UPPAAL - Present and Future

Gerd Behrmann[1], Alexandre David[2], Kim G. Larsen[1], M. Oliver Möller[3], Paul Pettersson[2], Wang Yi[2]

[1] Aalborg University, [2] Uppsala University, [3] BRICS Århus

Outline:

1. Model-checking Timed Automata
2. Internal Optimizations
3. Applications: Protocols & Controllers
4. Extensions of the Modeling Language

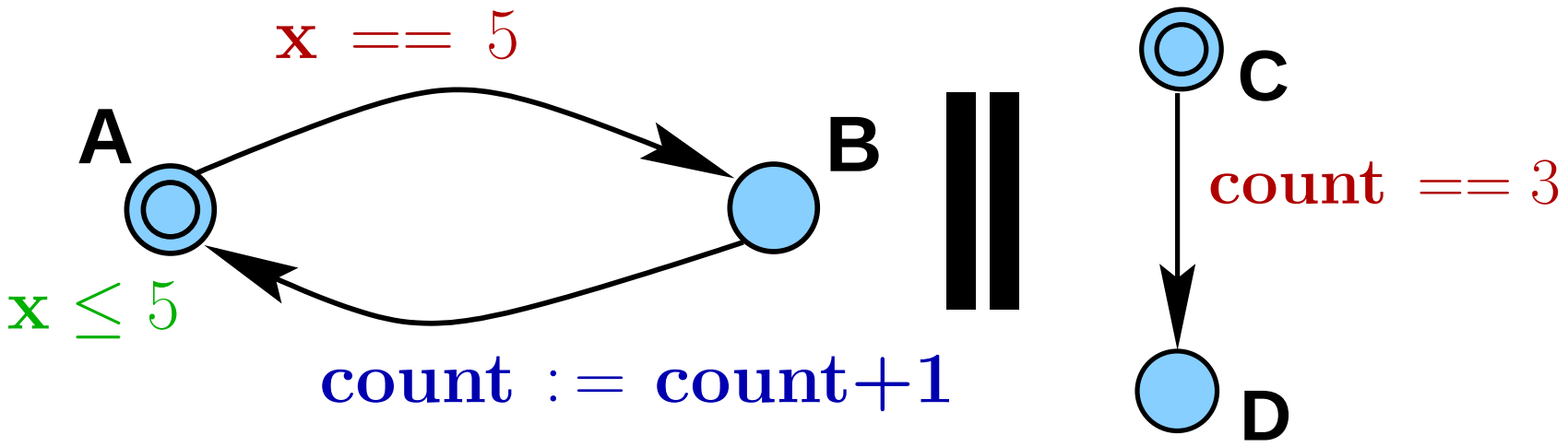
Collaborators

@ UPPsala	@ AALborg
Wang Yi	Kim G. Larsen
Johan Bengtsson	Arne Skou
Paul Pettersson	Carsten Weise
Fredrik Larsson	Kåre J. Kristoffersen
Alexandre David	Gerd Behrmann
Tobias Amnell	Thomas Hune
Elena Fersmann	M. Oliver Möller

@ many other places

David Griffioen, Ansgar Fehnker, Frits Vandraager, Klaus Havelund, Theo Ruys, Pedro DArgenio, J-P Katoen, J. Tretmans, Judi Romijn, Ed Brinksma, Franck Cassez, Magnus Lindahl, Francois Laroussinie, Patricia Bouyer, Augusto Burgueno, H. Bowmann, D. Latella, M. Massink, G. Faconti, Kristina Lundqvist, Lars Asplund, Justin Pearson, ...

UPPAAL: Model checking Timed Automata



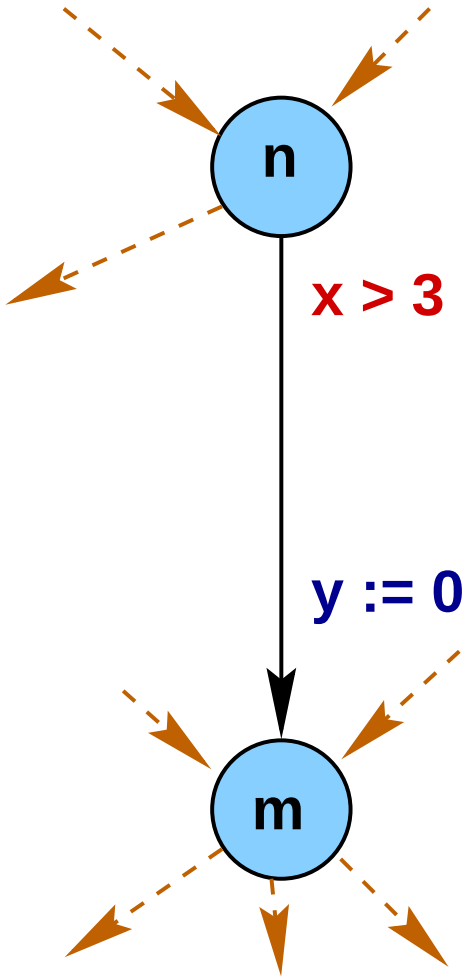

```
clock x; int count
```

- network of timed automata
- discrete data types
- arrays
- hand-shake synchronization
- urgency

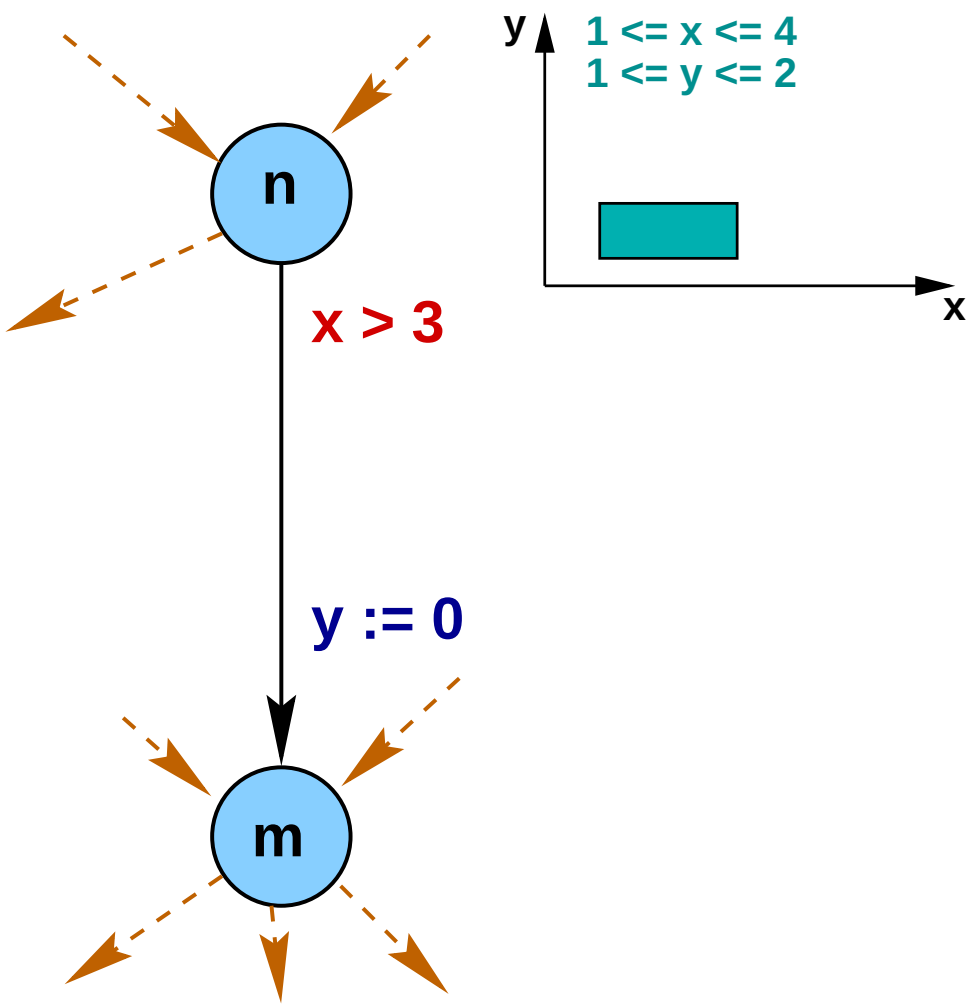
- template mechanism
- committed locations

- forward state-space exploration

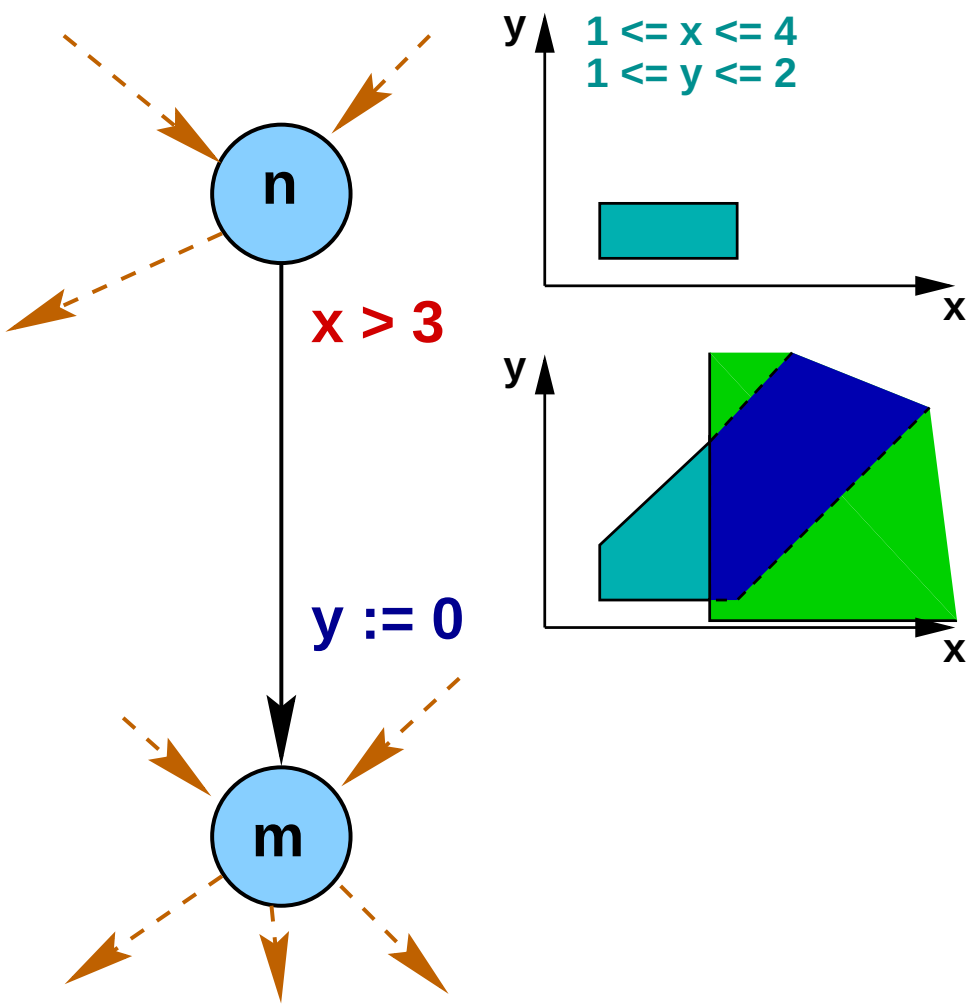
Symbolic Transitions



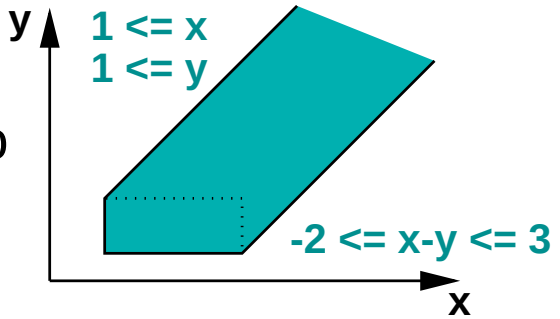
Symbolic Transitions



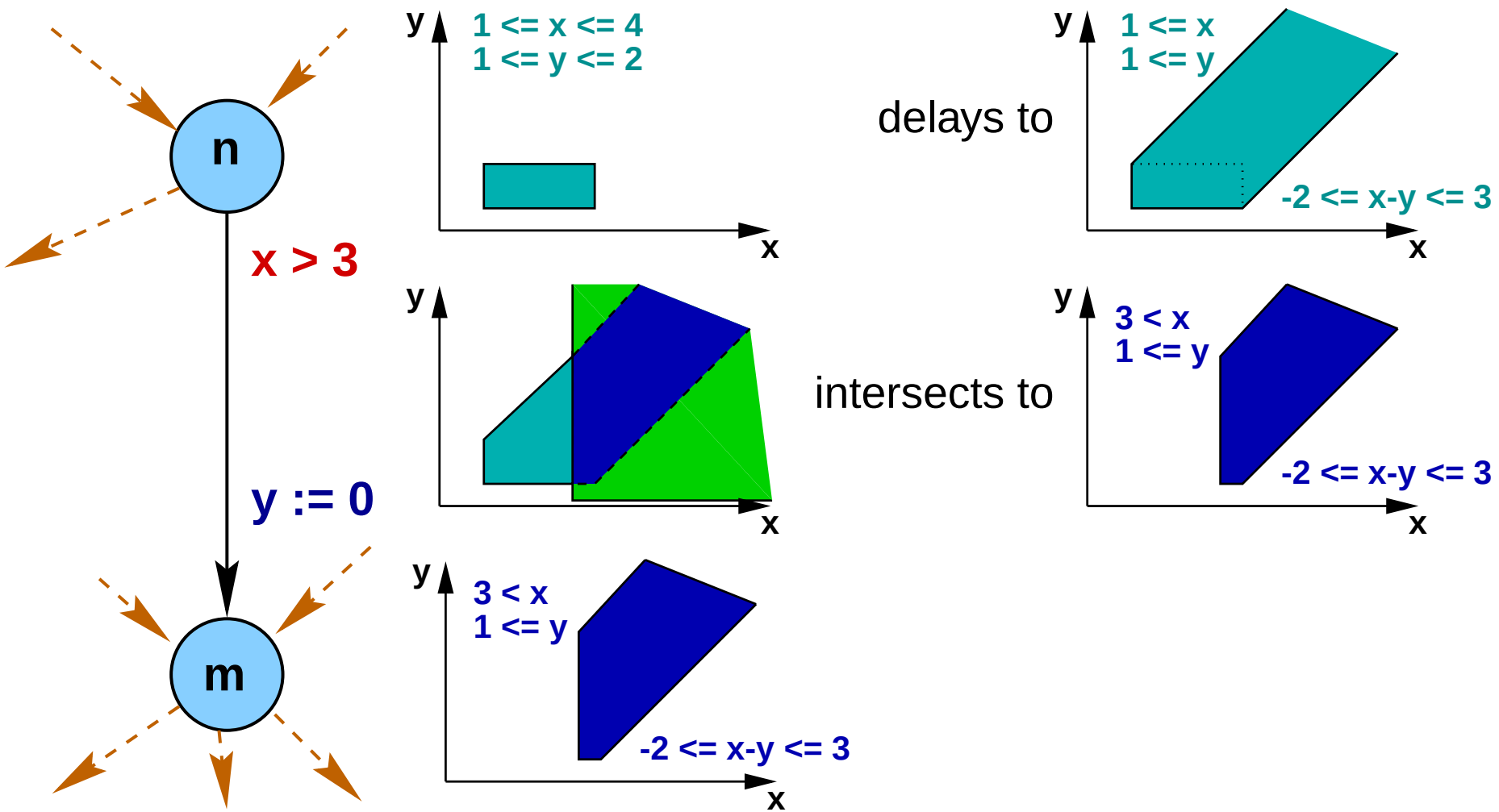
Symbolic Transitions



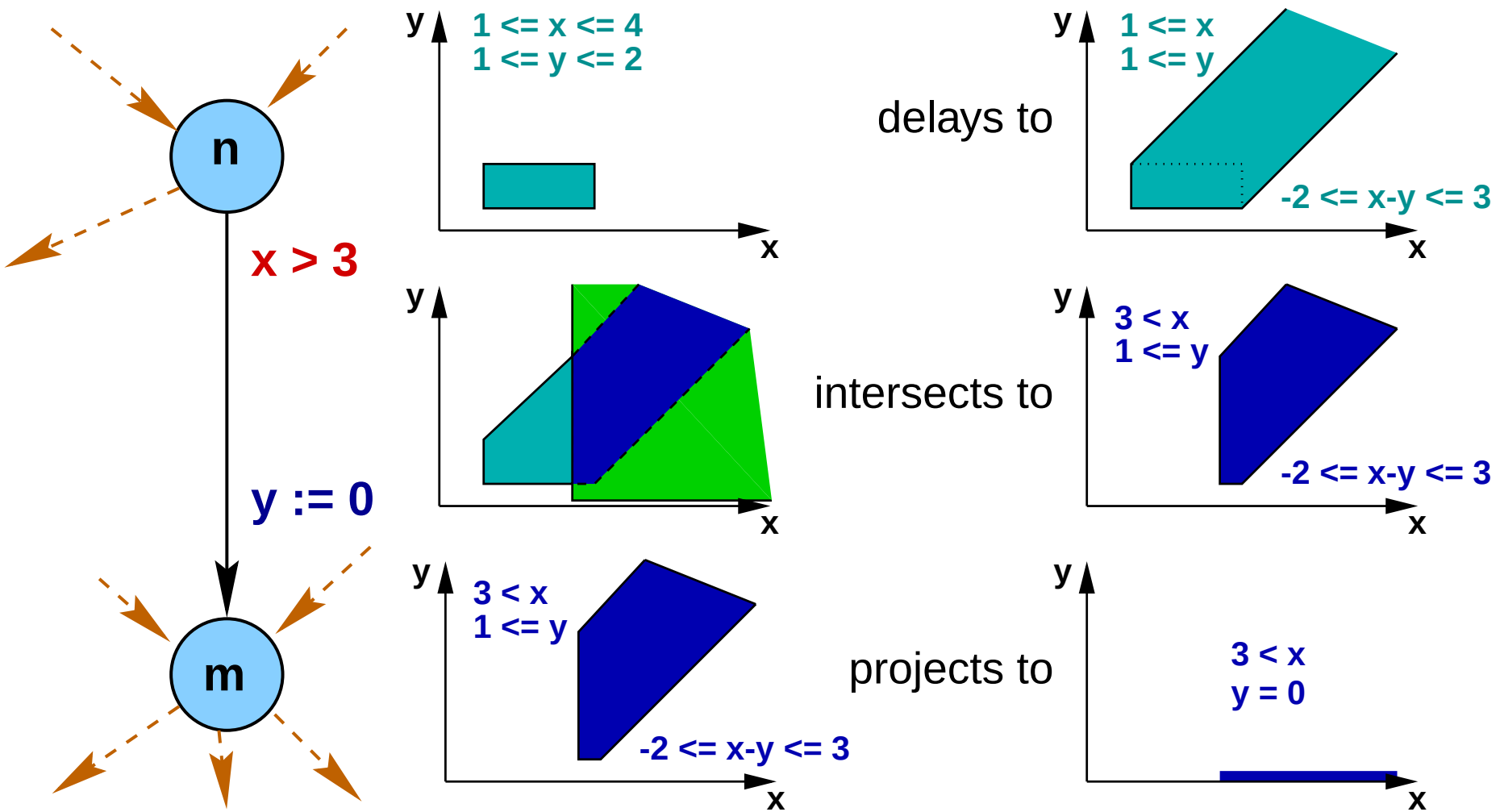
delays to



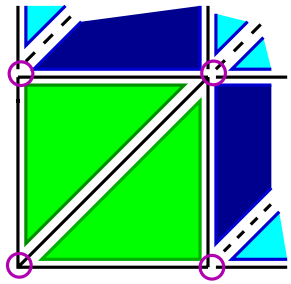
Symbolic Transitions

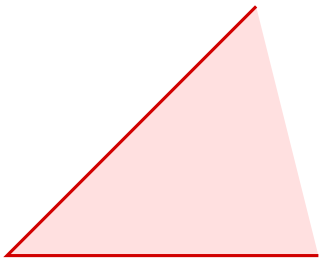


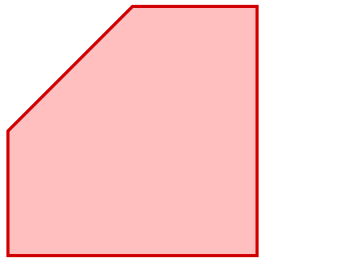
Symbolic Transitions



Sets of Clock-Evaluations

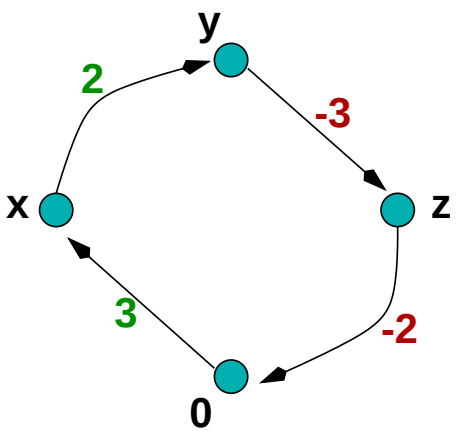






regions: smallest distinguishable sets **zones:** convex unions of regions

representing (unions of) zones: DBMs, CDDs, DDDs, ...

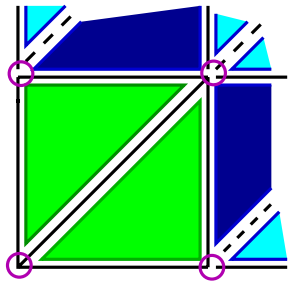


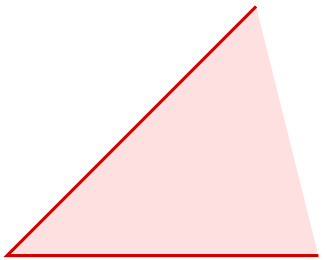
difference-bounded matrices

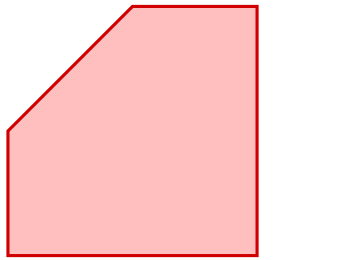
canonical

static

Sets of Clock-Evaluations

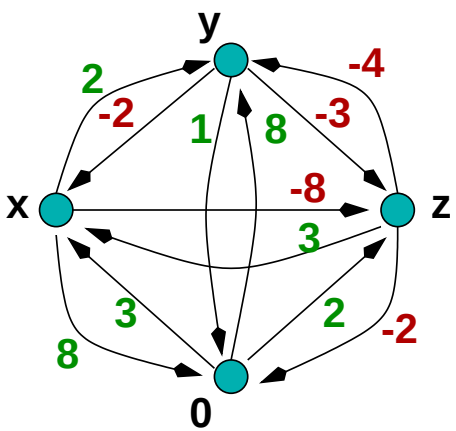






regions: smallest distinguishable sets **zones:** convex unions of regions

representing (unions of) zones: DBMs, CDDs, DDDs, ...

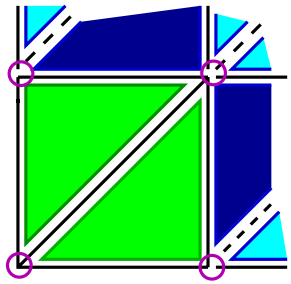


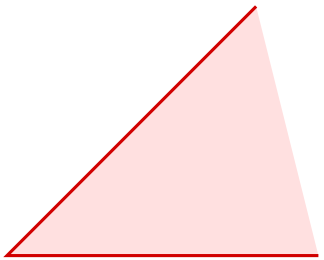
difference-bounded matrices

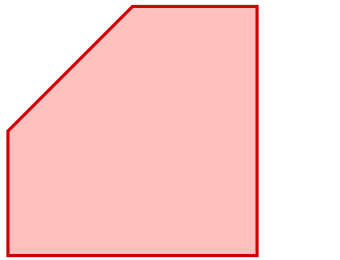
canonical

static

Sets of Clock-Evaluations

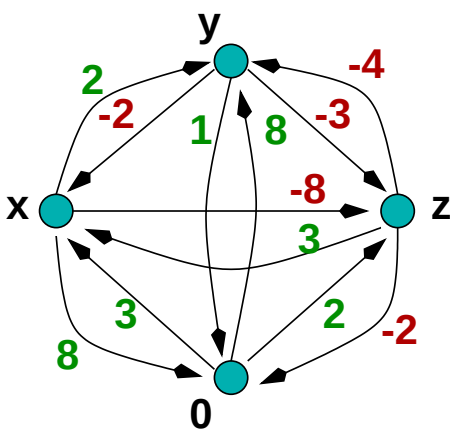


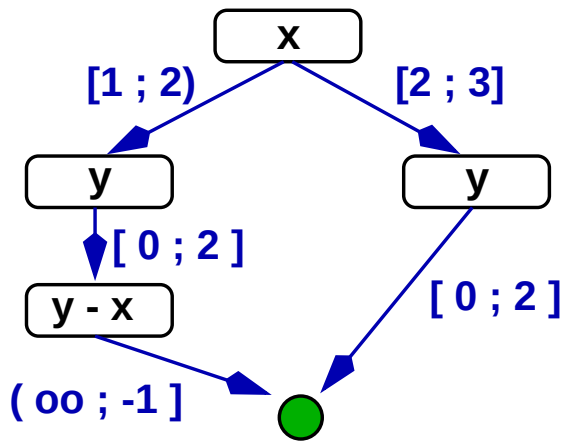




regions: smallest distinguishable sets **zones:** convex unions of regions

representing (unions of) zones: DBMs, CDDs, DDDs, ...



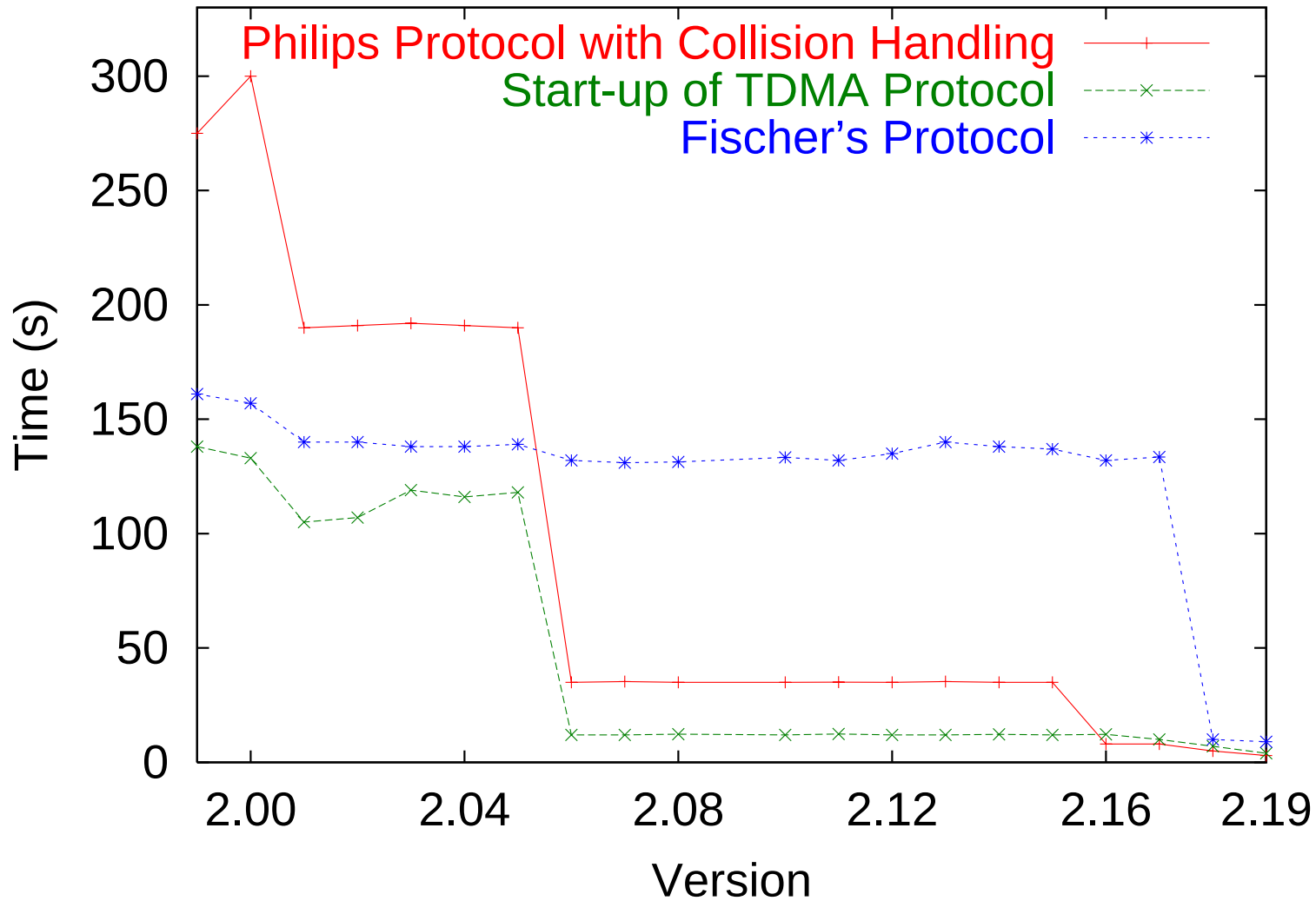


difference-bounded matrices **clock difference diagrams**

canonical *non-canonical*

static *flexible*

Engineering Improvements Dec ’96 - Sept ’98

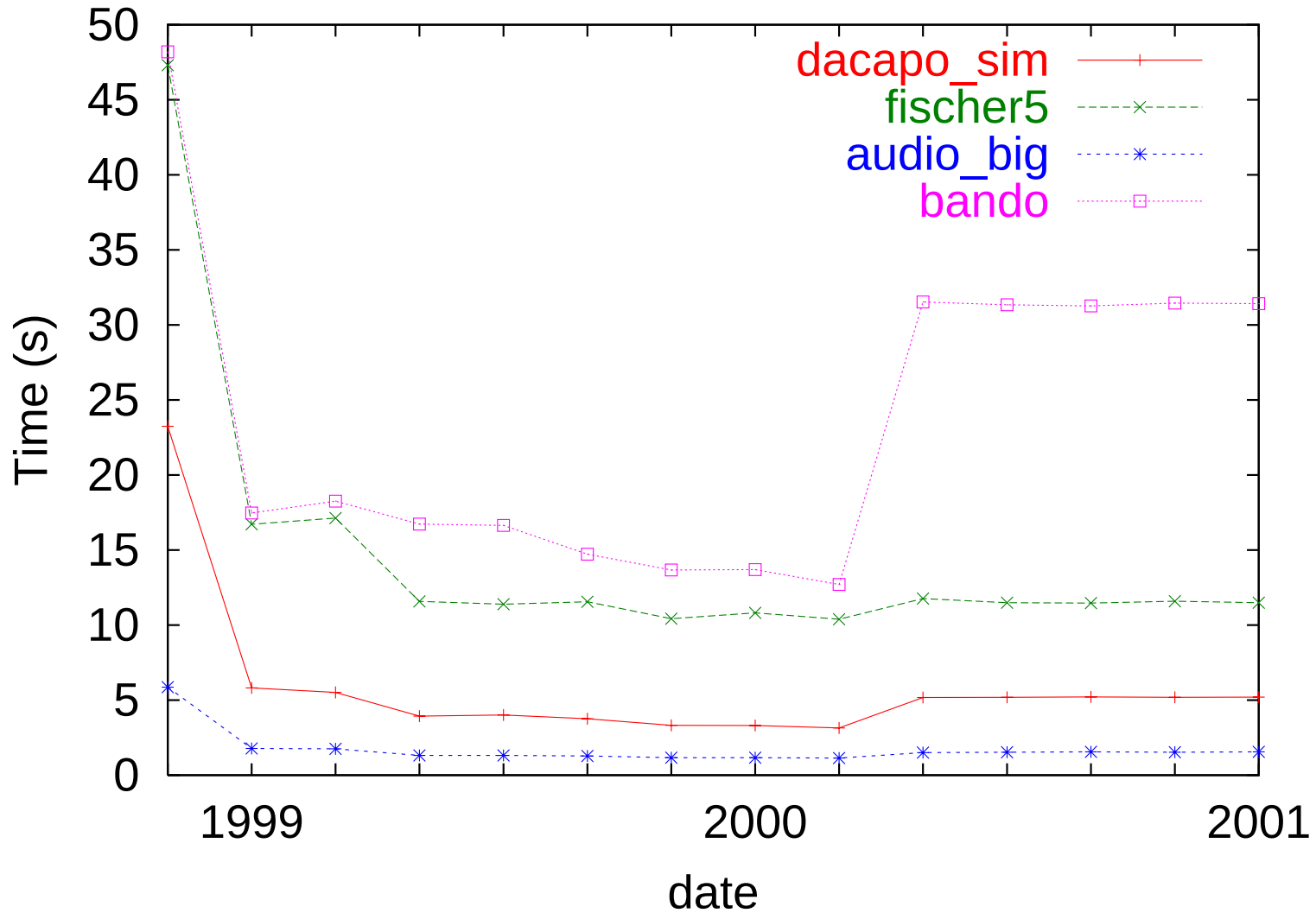


Internal Optimizations

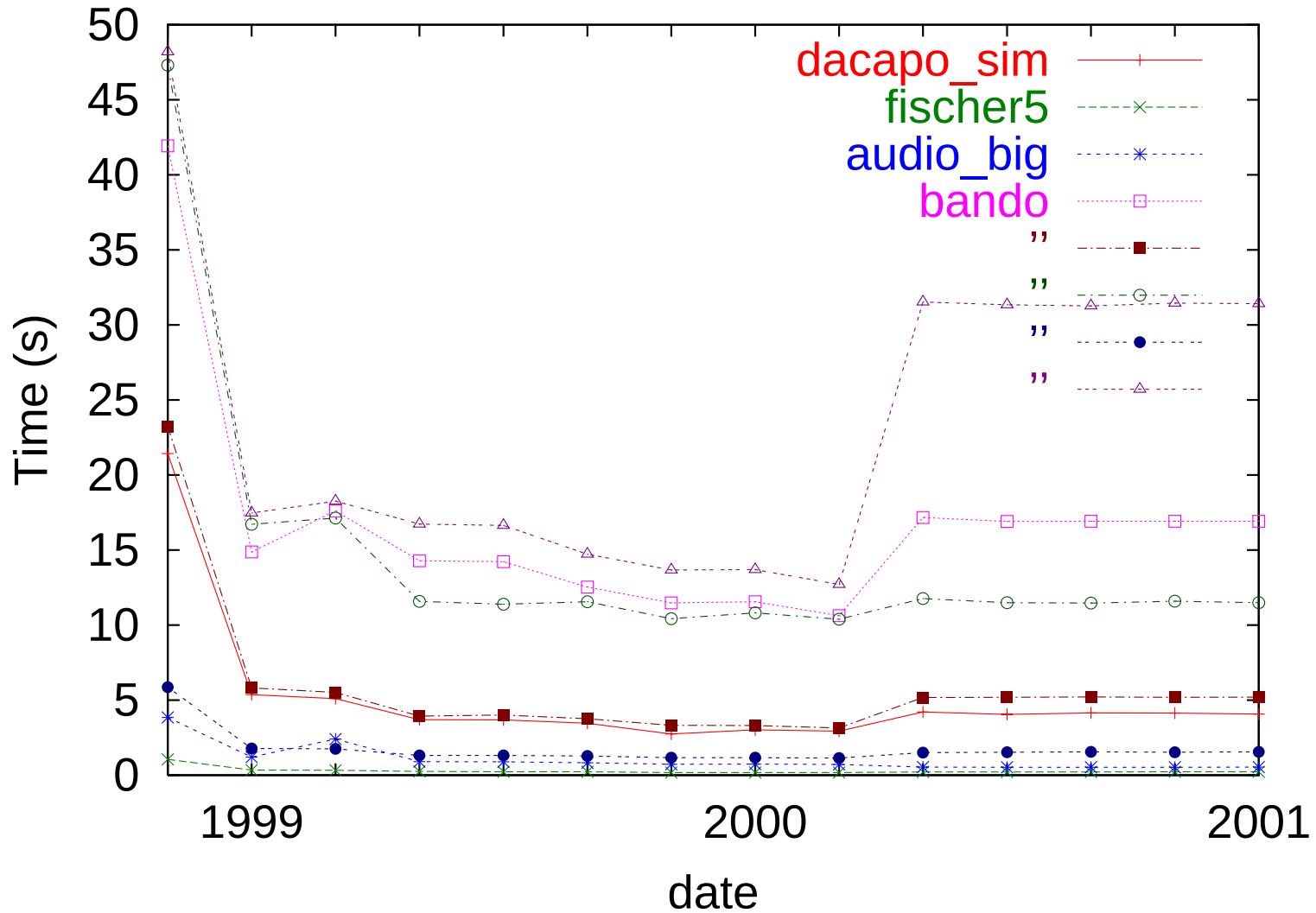
- × committed locations (to reduce interleavings)
- × active clock reduction
- × variation of search order
- × local reduction (compact DBM representation)
- × global reduction (remove covered states from *Passed*)
- ≈ convex hull over-approximation [safe]
- ≈ bit-state hashing [sound]

... and of course: *a lot of software engineering!*

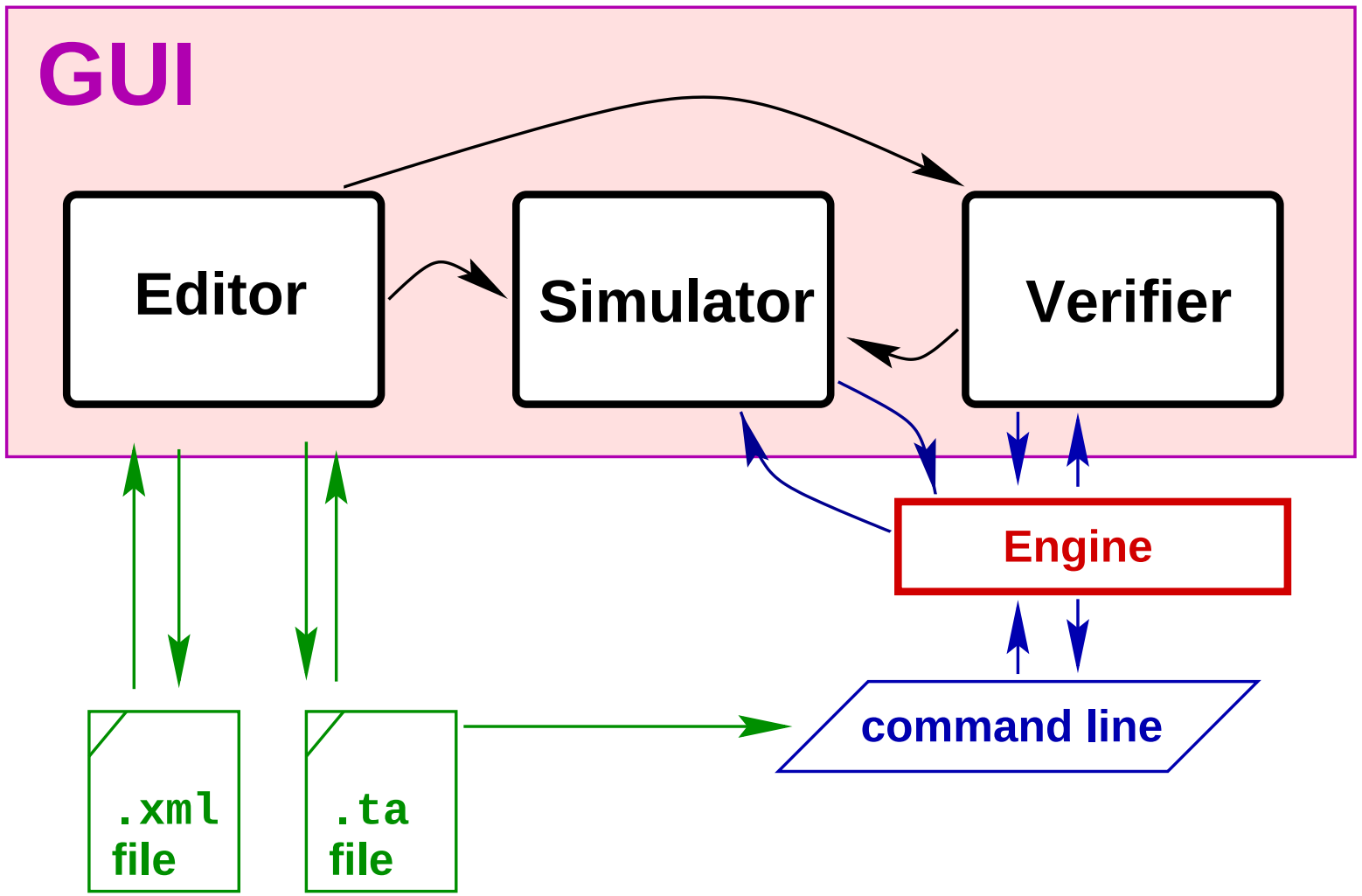
Benchmarks (without optimizations)



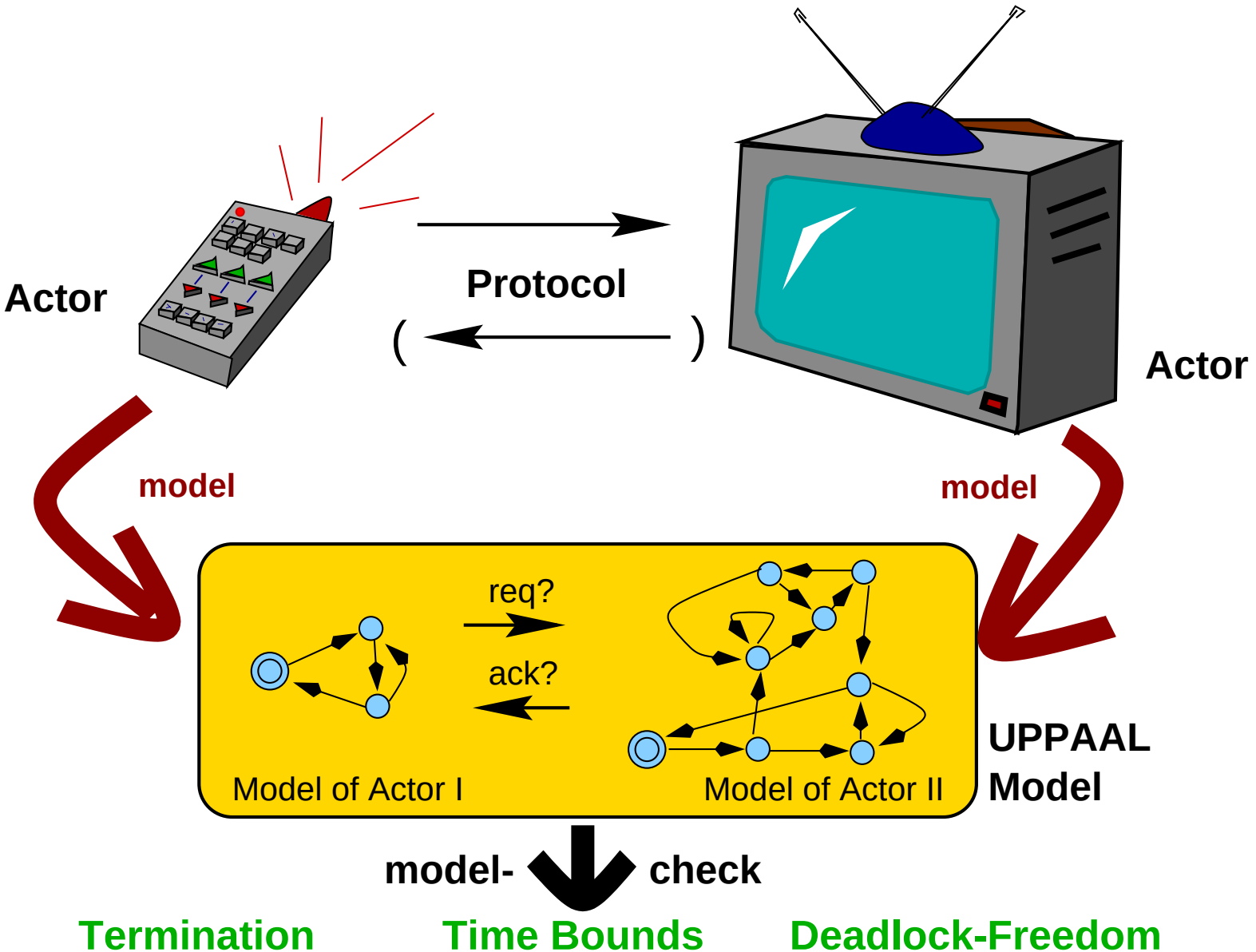
Benchmarks (with optimizations)



Architecture of UPPAAL



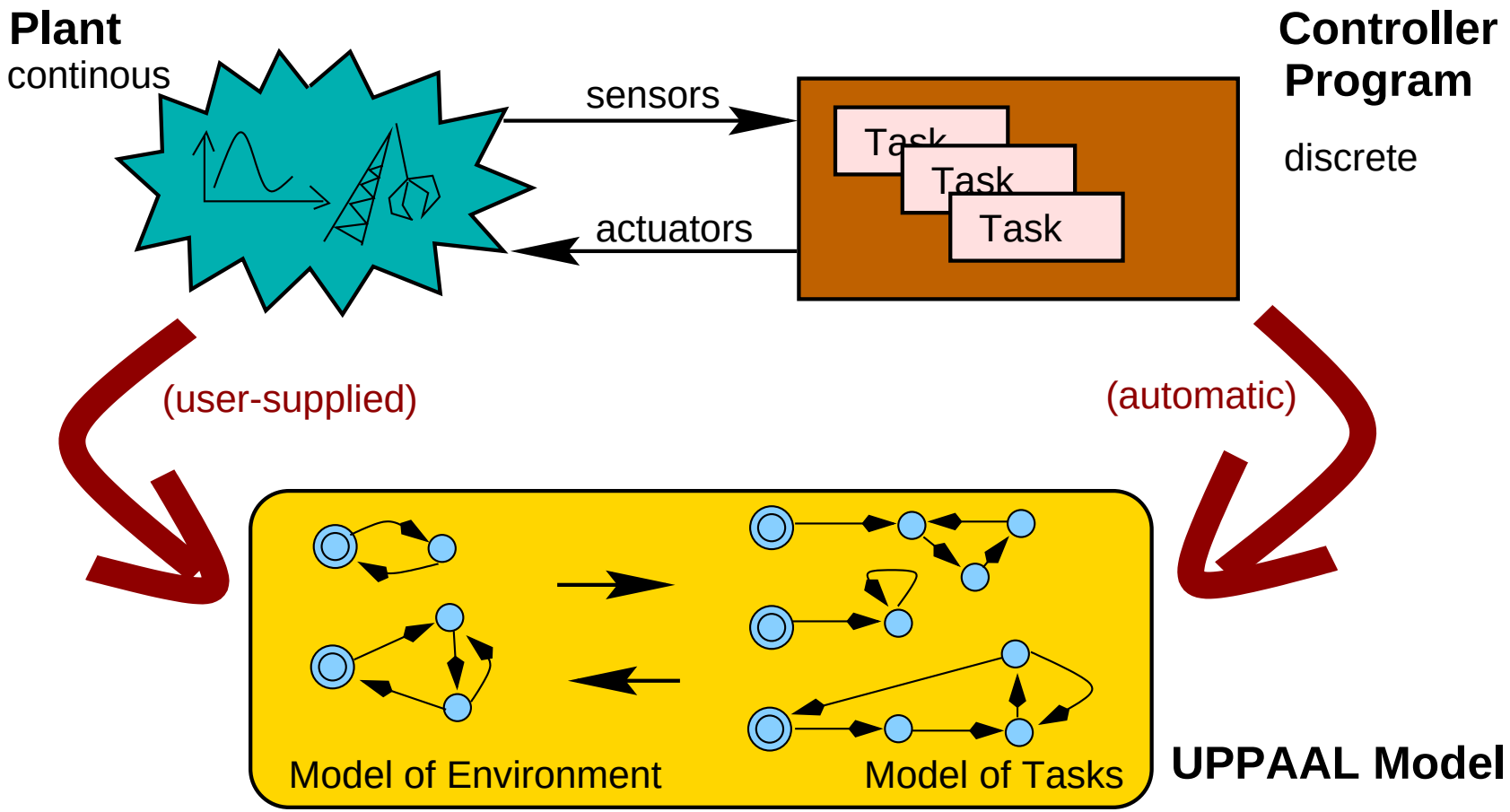
Communication Protocols



Case Studies: Protocols

- Philips Audio Protocol [HS95, CAV95, RTSS95, CAV96]
- Collision-Avoidance Protocol [SPIN95]
- Bounded Retransmission Protocol [TACAS97]
- Bang & Olufsen Audio/Video Protocol [RTSS97]
- TDMA Protocol [PRFTS97]
- Lip-Synchronization Protocol [FMICS97]
- Multimedia Streams [DSVIS98]
- ATM ABR Protocol [CAV99]
- ABB Fieldbus Protocol [ECRTS2k]
- IEEE 1394 Firewire Root Contention [STTT'01]

Composing the Embedded System Model



Case Studies: Controllers

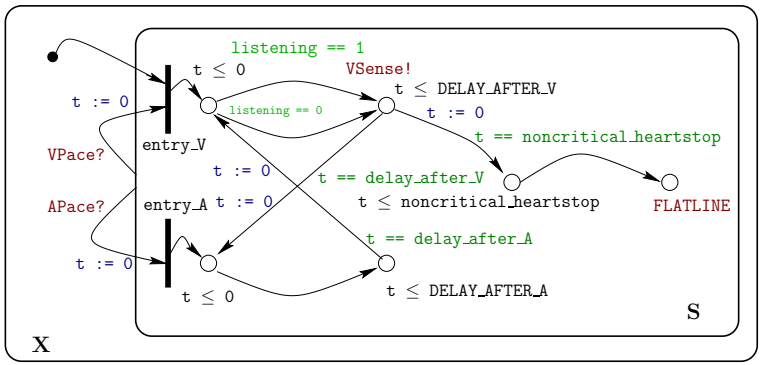
- Gearbox Controller [TACAS98]
- Bang & Olufsen Power Controller [RTPS99,FTRTFT2k]
- SIDMAR Steel Production Plant [RTCSA99, DSVV2k]
- Real-Time RCX Control-Programs [ECRTS2k]
- RCX Production Cell (2000)
- Experimental Batch Plant [ICDCS'01]
- Saab Car Locking System [RT-TOOLS'01]

Extensions of the Modeling Language

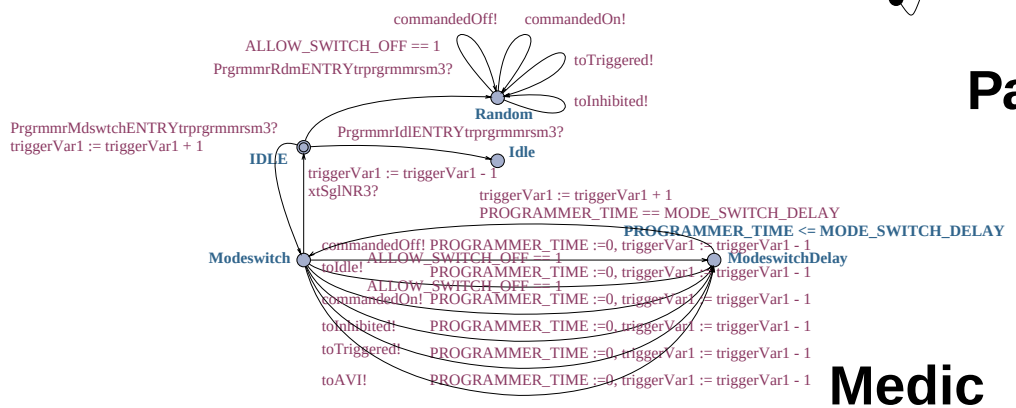
- **Stopwatch** extension
- **Probabilistic** timed automata
- **Hierarchical** timed automata
- **Parameters** on clock constraints
- **Cost-Optimal** timed automata
- **Executable** timed automata

Hierarchical UPPAAL

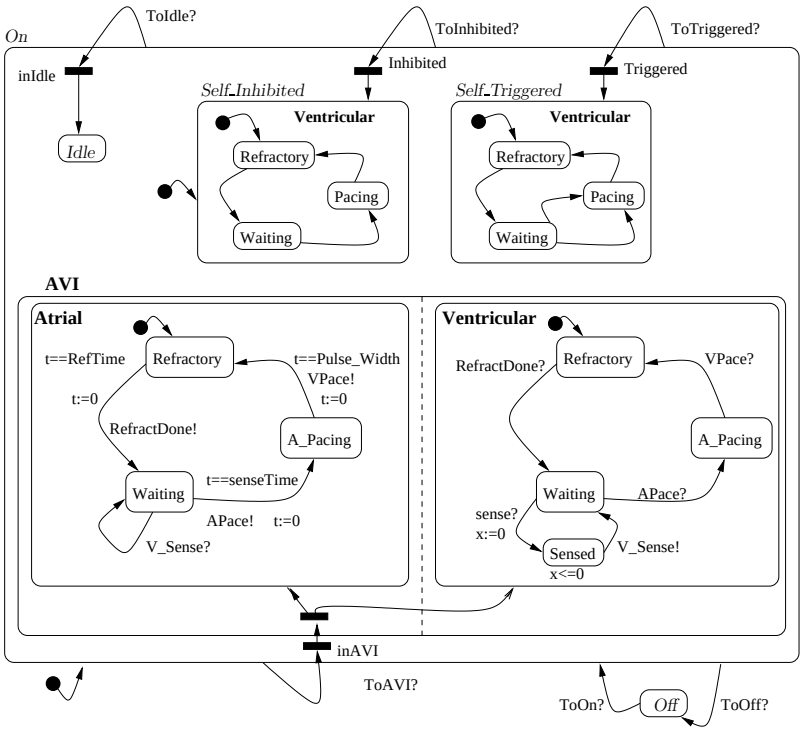
Use hierarchical timed automata:



Human Heart



Medic



Pacemaker

Flattened Version of the Pacemaker

HTA model	# XML tags	564	$\longrightarrow$	1191	UPPAAL model
	# proper control locations	35	$\longrightarrow$	45	

- SAFETY:
  `A[] ¬heart stops`
- LIVENESS:
  `A[] Vcontract => A<> Acontract`

Parameters:

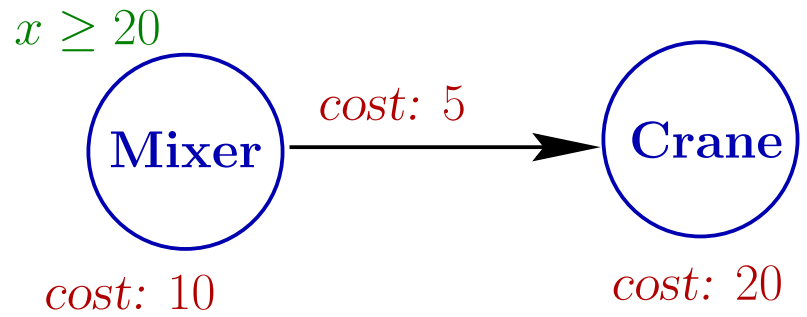
```
REFRACTORY_TIME  = 50
SENSE_TIMEOUT    = 15

DELAY_AFTER_V = 50
DELAY_AFTER_A =  5

MODE_SWITCH_DELAY  = 66
```

E.g. for `MODE_SWITCH_DELAY  = 65`, `A[] ¬heart stops` is violated

Cost-Optimality



Idea: Add *cost* to locations and actions

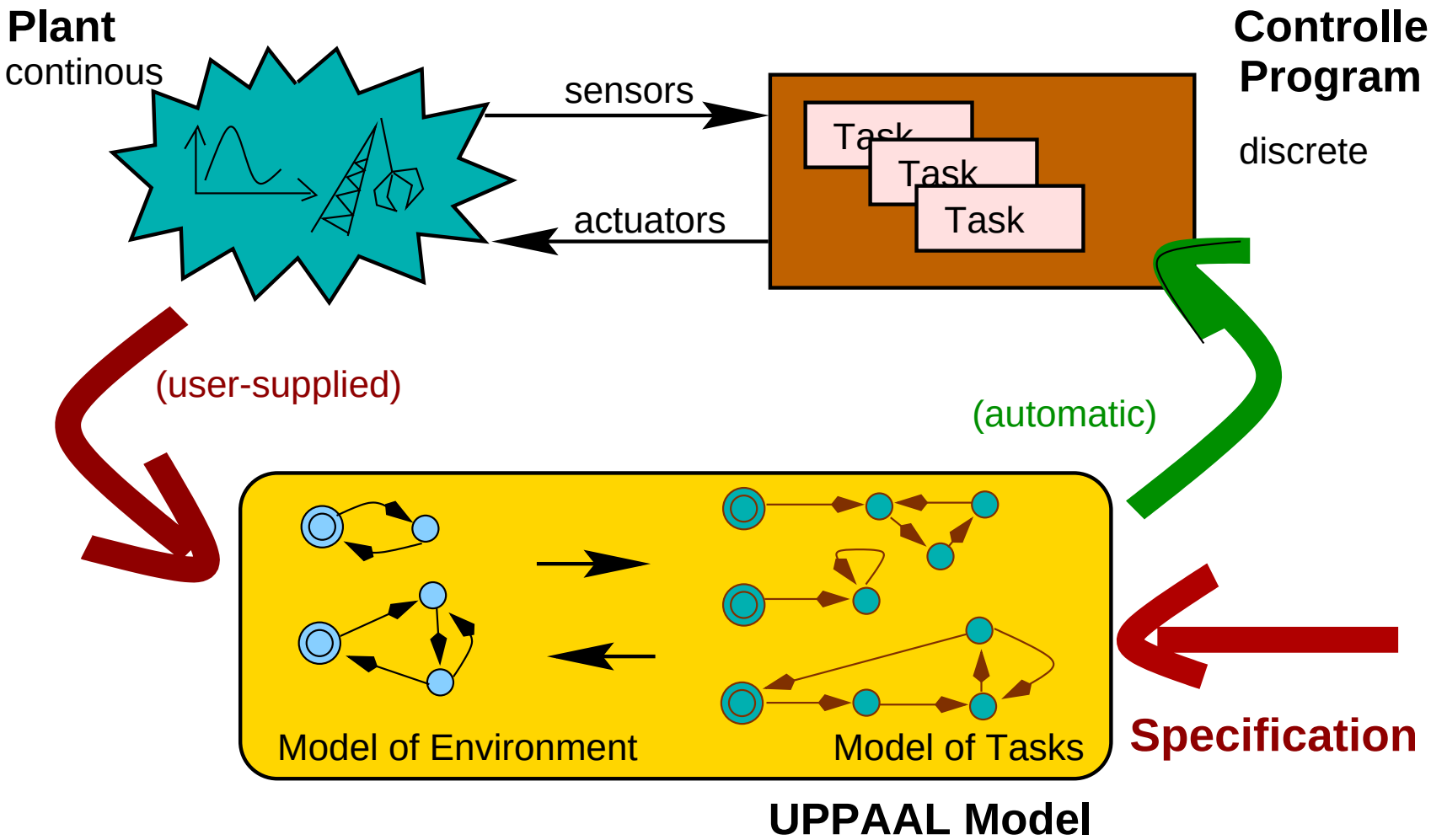
Starting Point: 'cost' not necessarily uniform

Approach: attach different (integer) prices to locations
treat algorithmically with *priced zones*

Applied: compute schedule for a steel batch plant in Gent
and a LEGO model of it [Feh99, HLP00]

Fact: Cost-Optimal trace is computable

Controller Synthesis

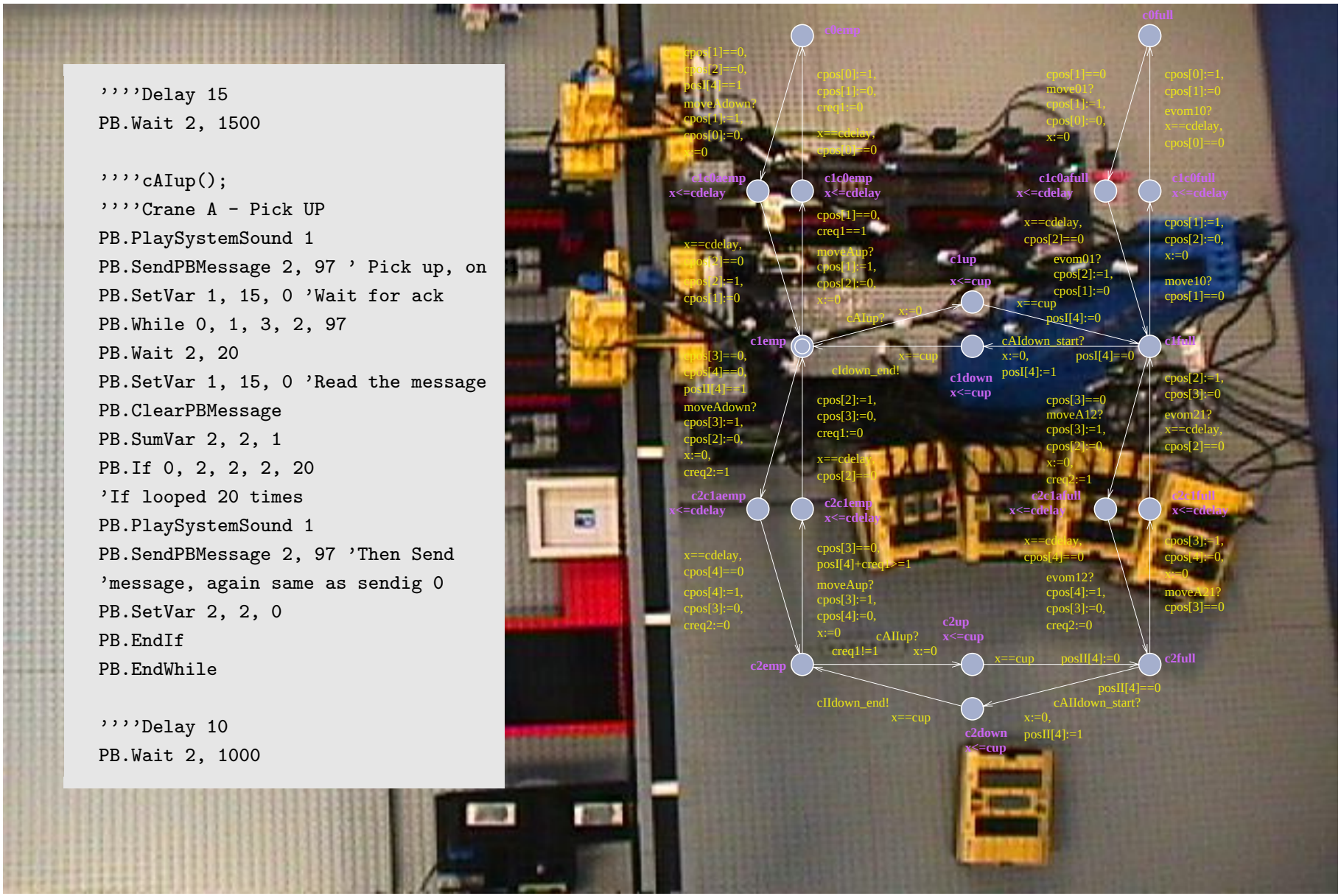


SIDMAR Steel Production Plant (LEGO Version)

```
''''Delay 15
PB.Wait 2, 1500

''''cAIup();
''''Crane A - Pick UP
PB.PlaySystemSound 1
PB.SendPBMessage 2, 97 ' Pick up, on
PB.SetVar 1, 15, 0 'Wait for ack
PB.While 0, 1, 3, 2, 97
PB.Wait 2, 20
PB.SetVar 1, 15, 0 'Read the message
PB.ClearPBMessage
PB.SumVar 2, 2, 1
PB.If 0, 2, 2, 2, 20
'If looped 20 times
PB.PlaySystemSound 1
PB.SendPBMessage 2, 97 'Then Send
'message, again same as sendig 0
PB.SetVar 2, 2, 0
PB.EndIf
PB.EndWhile

''''Delay 10
PB.Wait 2, 1000
```



Cost-Optimal Extension: Summary

- completely random schedules not analyzable
  → guides/optimality *restrict* behavior
- the LEGO model helped *debugging* the UPPAAL model

Compared to traditional (LP) methods:

- reasonably efficient
- more **flexible**
- *aircraft landing* case study:
  computed schedules either **better** or **substantially worse**

Completed Parts

- ✔ cost-optimal extension
- ✔ parametric extension
- ✔ stopwatch extension
- ✔ distributed UPPAAL

Work in Progress

- → probabilistic extension
- → hierarchical extension
- → executable UPPAAL

Work Planned

- ⋆ dynamic partitioning
- ⋆ hybrid animation

Go, Get It!

UPPAAL2k (3.2.1) available for

Linux, SunOS, and MS Windows

http://www.uppaal.com/

Since July 1999: $>$ 1˙000 downloads (from different users)

$>$ 60 countries

Open mailing list: http://groups.yahoo.com/group/uppaal

Bibliography

[AHV93] Rajeev Alur, Thomas A. Henzinger, and Moshe Y. Vardi. Parametric Real-time Reasoning. In *Proceedings of the Twenty-Fifth Annual ACM Symposium on the Theory of Computing*, pages 592–601, 1993.

[AJ01] Tobias Amnell and Pontus Jansson. In *Proc. of Workshop on Real-Time Tools*, August 2001.

[BSdRT01] GiosuéÁBandini, R. F. Lutje Spelberg, R. C. M. de Rooij, and W. J. Toetenel. Application of Parametric Model Checking - The Root Contention Protocol. In *Proc. of the 34th Annual Hawaii International Conference on System Sciences (HICSS-34)*, 2001.

[DY00] Alexandre David and Wang Yi. Modelling and analysis of a field bus protocol. In *proceedings of the 12th Euromicro Conference On ReaL-Time Systems*. IEEE Press, June 2000.

[Feh99] Ansgar Fehnker. Scheduling a steel plant with timed automata. In *Proceedings of the 6th International Conference on Real-Time Computing Systems and Applications (RTCSA99)*, pages 280–286. IEEE Computer Society, 1999.

[HLP00] Thomas Hune, Kim G. Larsen, and Paul Pettersson. Guided Synthesis of Control Programs Using UPPAAL. In Ten H. Lai, editor, *Proc. of the IEEE ICDCS International Workshop on Distributed Systems Verification and Validation*, pages E15–E22. IEEE Computer Society Press, April 2000.

[HSLL97] Klaus Havelund, Arne Skou, Kim G. Larsen, and Kristian Lund. Formal Modeling and Analysis of an Audio/Video Protocol: An Industrial Case Study Using UPPAAL. In *Proc. of the 18*th *IEEE Real-Time Systems Symposium*. IEEE Computer Society Press, December 1997.

[Hun99] Thomas Hune. Modelling a real-time language. In *Proc. 4*th *Workshop on Formal Methods for Industrial Critical Systems, FMICS*, 1999.

[Jen96] H.E. Jensen. Model checking probabilistic real time systems. In B. Bjerner, M. Larsson, and B. Nordström, editors, *Proceedings of the 7th Nordic Workshop on Programming Theory,* Göteborg Sweden, Report 86, pages 247–261. Chalmers University of Technolog, 1996.

[KGLP98] Wang Yi Kim G. Larsen, Carsten Weise and Justin Pearson. Clock difference diagrams. Technical Report 98/99, Department of Computer Systems, Uppsala University, P.O. Box 325, SE-751 05 Uppsala, Sweden., August 1998. Available as `http://www.docs.uu.se/docs/rtmv/papers/lwyp-sub98-1.ps.gz`.

[KLPW99] K. Kristoffersen, K. Larsen, P. Pettersson, and C. Weise. VHS Case Study 1 - Experimental Batch Plant using UPPAAL. BRICS, University of Aalborg, Denmark, May 1999.

[KNSS99] M. Kwiatkowska, G. Norman, R. Segala, and J. Sproston. Automatic veri£cation of real-time systems with probability distributions. In J.-P. Katoen, editor, *Proceedings of the 5th AMAST Workshop on Real-Time and Probabilistic System,* Bamberg, Germany, volume 1601 of *Lecture Notes in Computer Science*, pages 75–95. Springer-Verlag, 1999. An extended version will appear in Theoretical Computer

Science.

[LLPY97] Kim G. Larsen, Fredrik Larsson, Paul Pettersson, and Wang Yi. Efficient verification of real-time systems: Compact data structure and state-space reduction. In *proceedings of the 18th IEEE Real-Time Systems Symposium*. IEEE Press, December 1997.

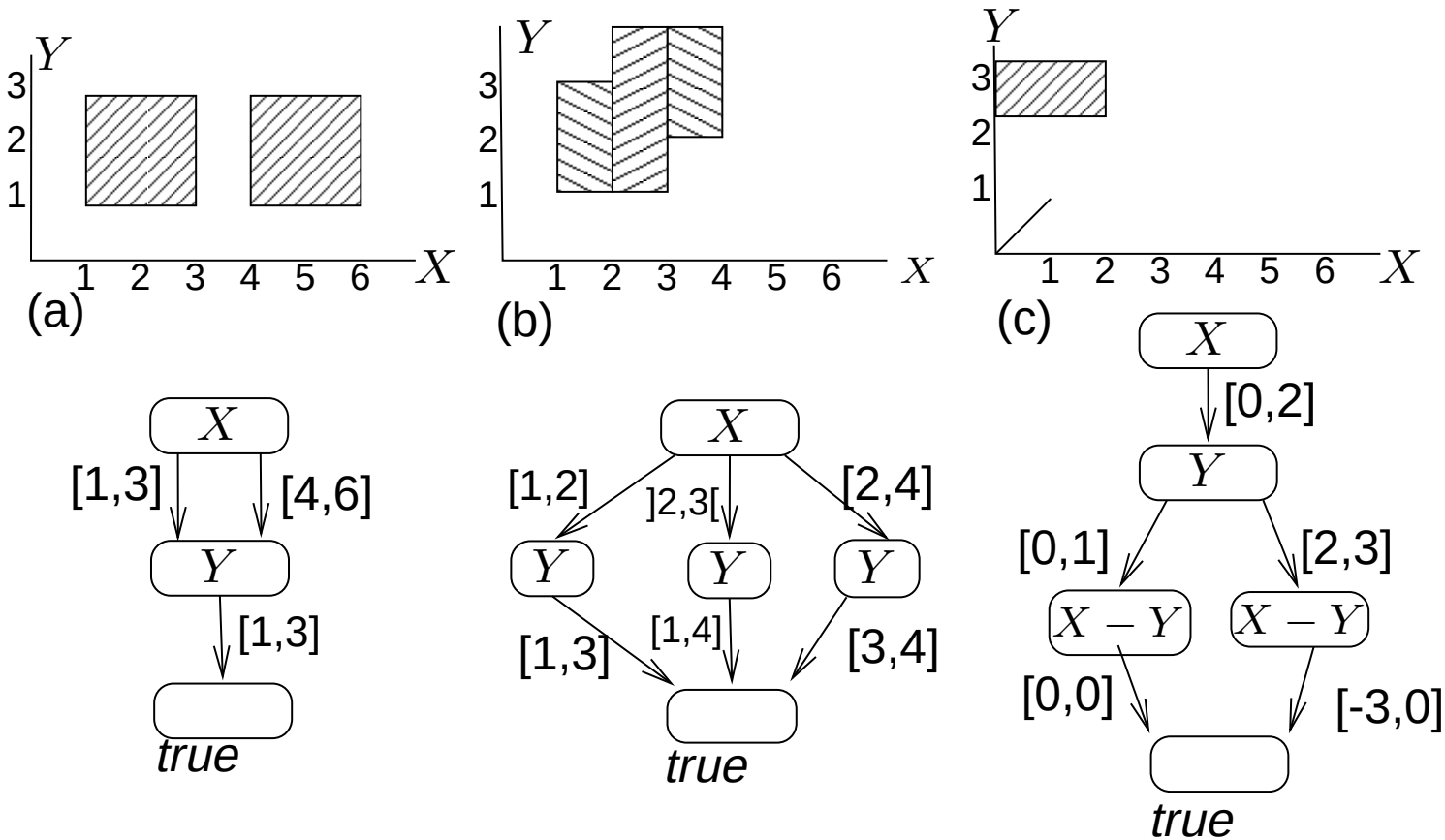
[SS01] D. Simons and M. Stoelinga. Mechanical verification of the ieee1394a root contention protocol using uppaal2k. 2001. To appear in International Journal on Software Tools for Technology Transfer.

Clock Difference Diagrams (CDDs)

Data structure to express disjunction of zones

- → similar to BDDs
- → rooted, directed, acyclic graph
- → every node labeled x or $x - y$
- → every edge labeled with an interval
- → order of labels fixed
- → one terminal node: $\mathbf{true}$
- → missing edges lead to $\mathbf{false}$
- ⚠ *not canonical*

Clock Difference Diagrams (CDDs) (2)



Stopwatch UPPAAL

timed automaton + stopwatches = SWA

Fact: Any *timed language* accepted by a *linear hybrid automaton* can also be accepted by a *stopwatch automaton*

linear hybrid automaton —*translate*→ SWA

Problem: reachability analysis of SWA is undecidable

Observation: often it suffices to *over-approximate* reachability

Approach: run DBM-based SWA, with *approximative future*
(only differences of *two* stop-watches considered)

Notes: *way* to *translate* effects accuracy
more sophisticated translations could preserve *termination*

Probabilistic UPPAAL

Example Problem:

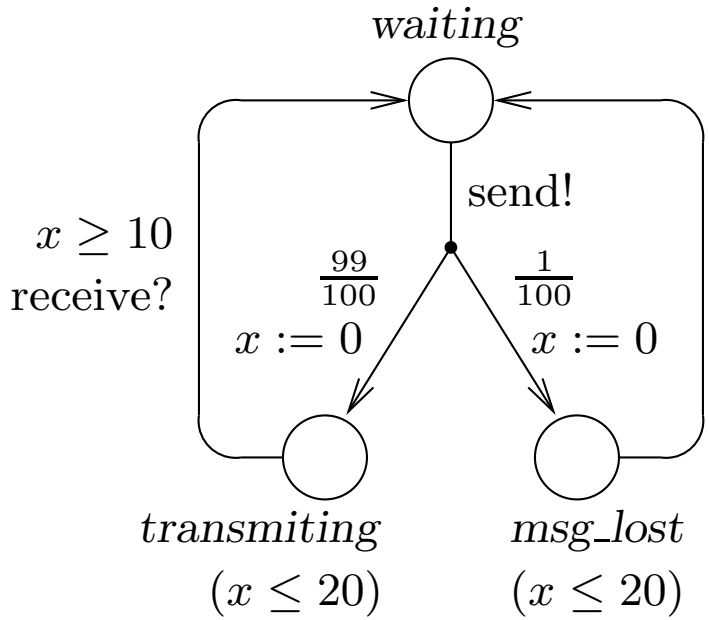
Lossy channel with known probabilites

Cannot prove:

in time X, message will arrive

But:

$P_{\geq 95\%}(\ \forall\Box_{\leq 1000}$ *received*)



existing Approaches: Jensen 96, Kwiatkowska et al. 99

Problem: based on region graph construction

new Approach: use minimization techniques to obtain stable *probabilistic zone graphs*

use matching data structure

Determining Parameters: *Parametric-Uppaal*

Parameters: in clock guards $x \bowtie p$, $x - y \bowtie p$

$\bowtie \in \{<, \leq, =, \geq, >\}$, p a linear expression

Fact: parameterized timed reachability undecidable for systems with $\geq$3 clocks [AHV93]

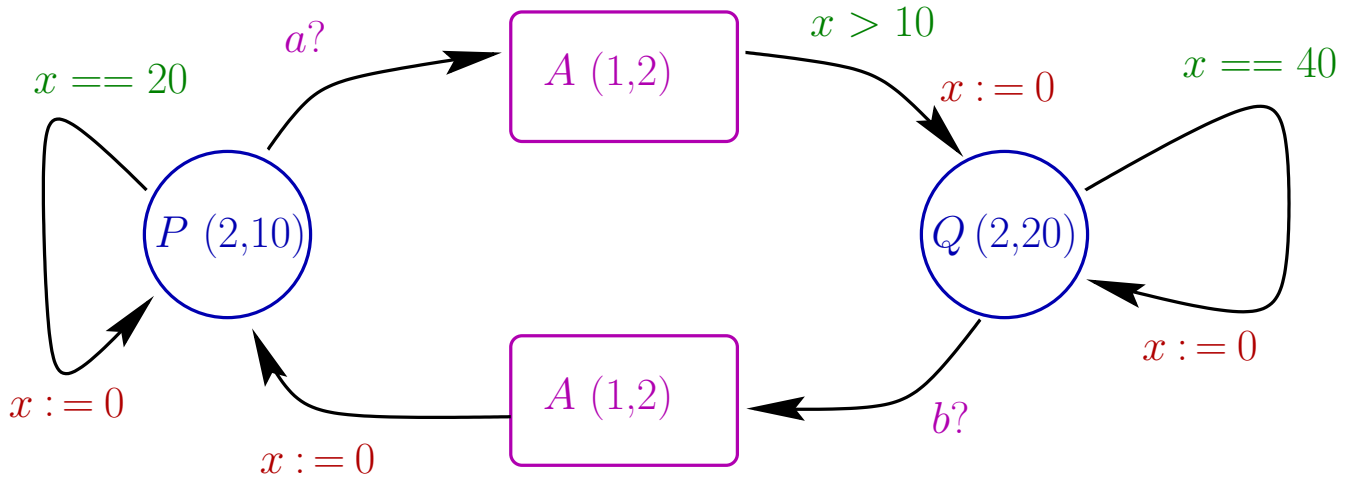
UPPAAL + LP solver (from PMC tool) = *semi*-algorithm

data-structure: parametric DBMs

modified algorithm: split, if the outcome of a comparison is dependent on parameter values

not guaranteed to terminate $\Rightarrow$ output partial solutions

Executable Timed Automata



Periodic Tasks P, Q Spontaneous Tasks A, B

Parameters: worst-case execution time, deadline

Delay transition	$\equiv$	execute task with earliest deadline
Action transition	$\equiv$	releases a new task
Automaton schedulable	$\Leftrightarrow$	every $a!, b!$-sequence schedulable

Fact: added *Preemption* is as expressive as TAs with stop watches

UPPAAL in the European WOODDES project

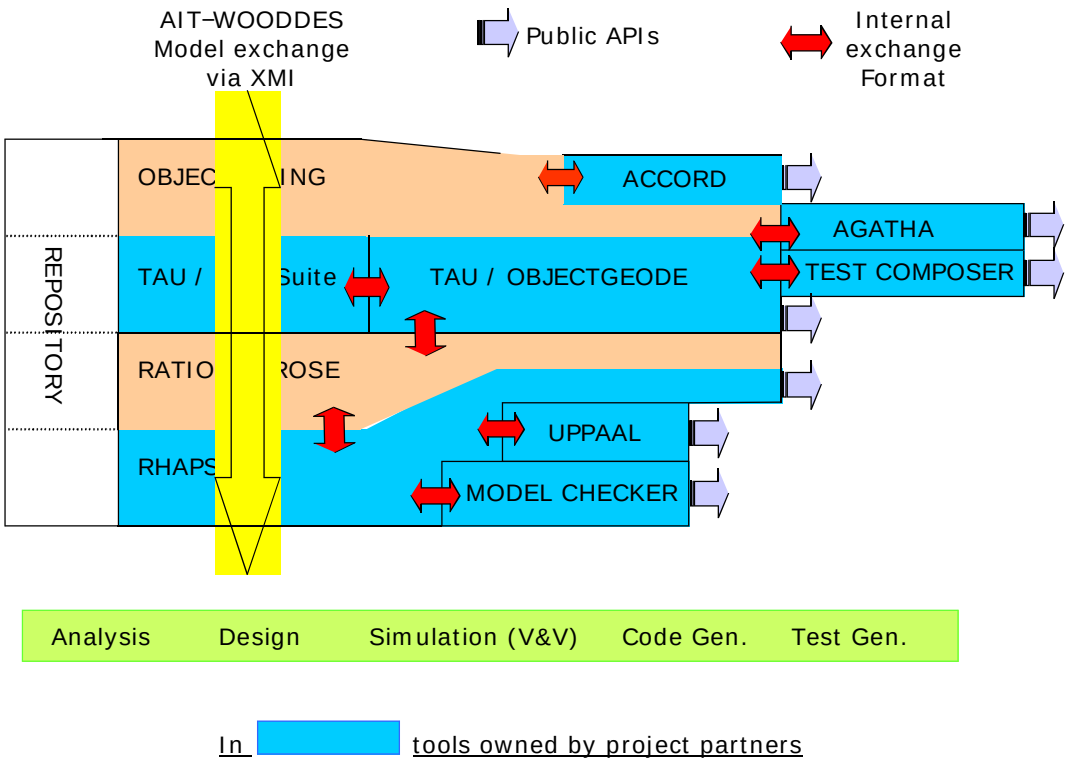
Workshop for Object-Oriented Design and Development of Embedded Systems

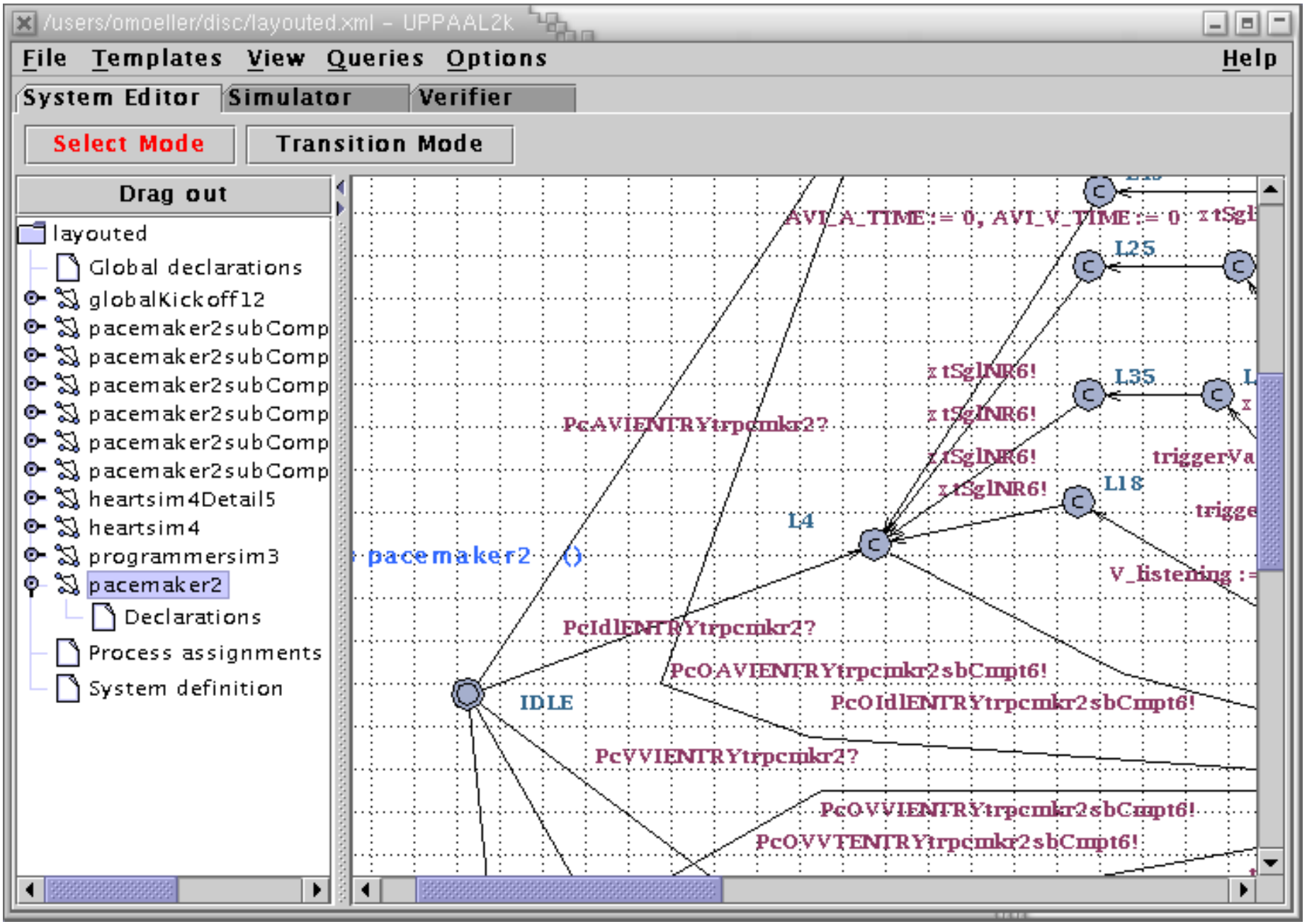
Partners:

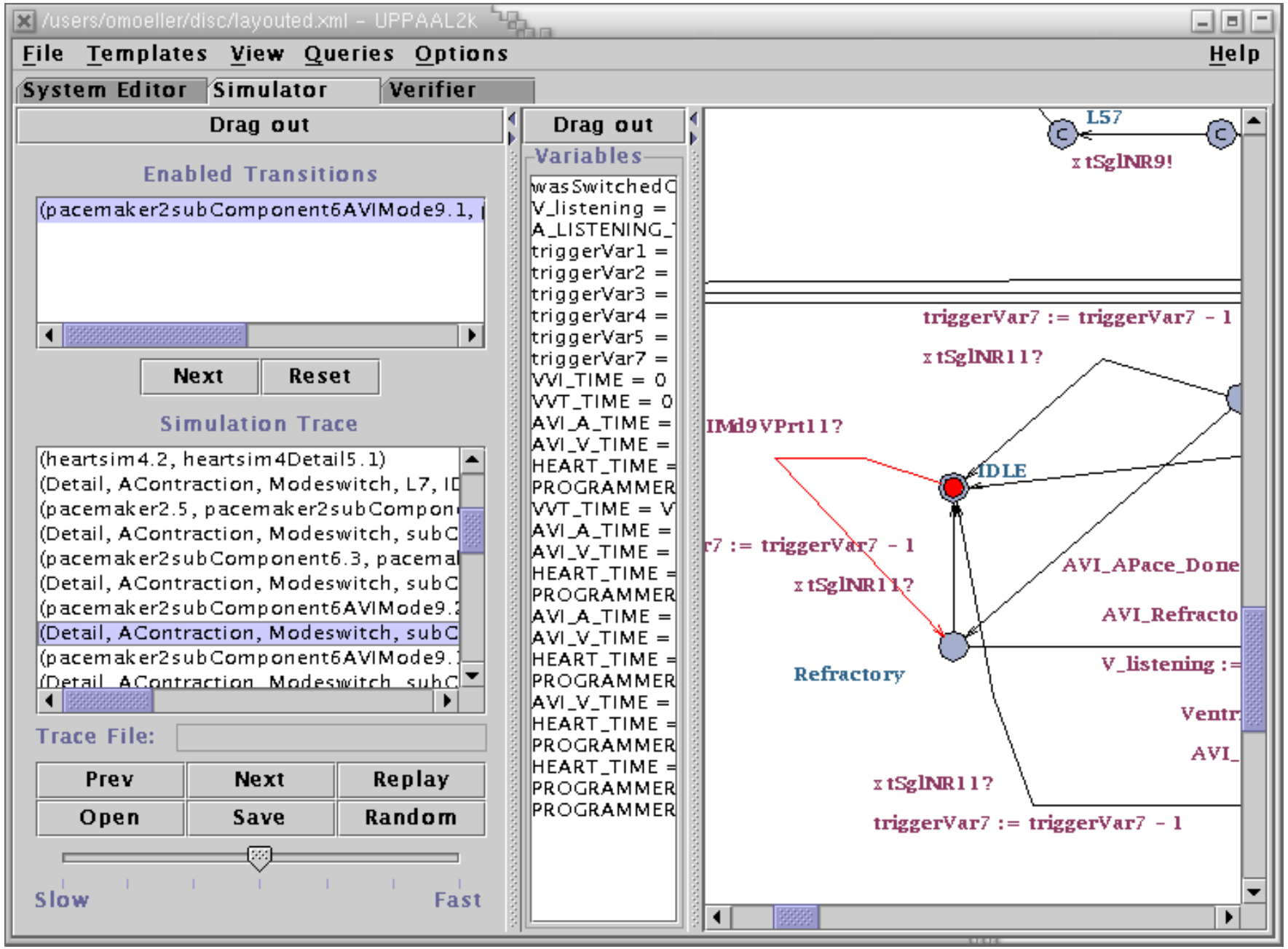
- PSA
- Mecel
- CEA
- SOFTEAM
- I-Logix
- Intracom
- Offis
- Uppsala
- Aalborg

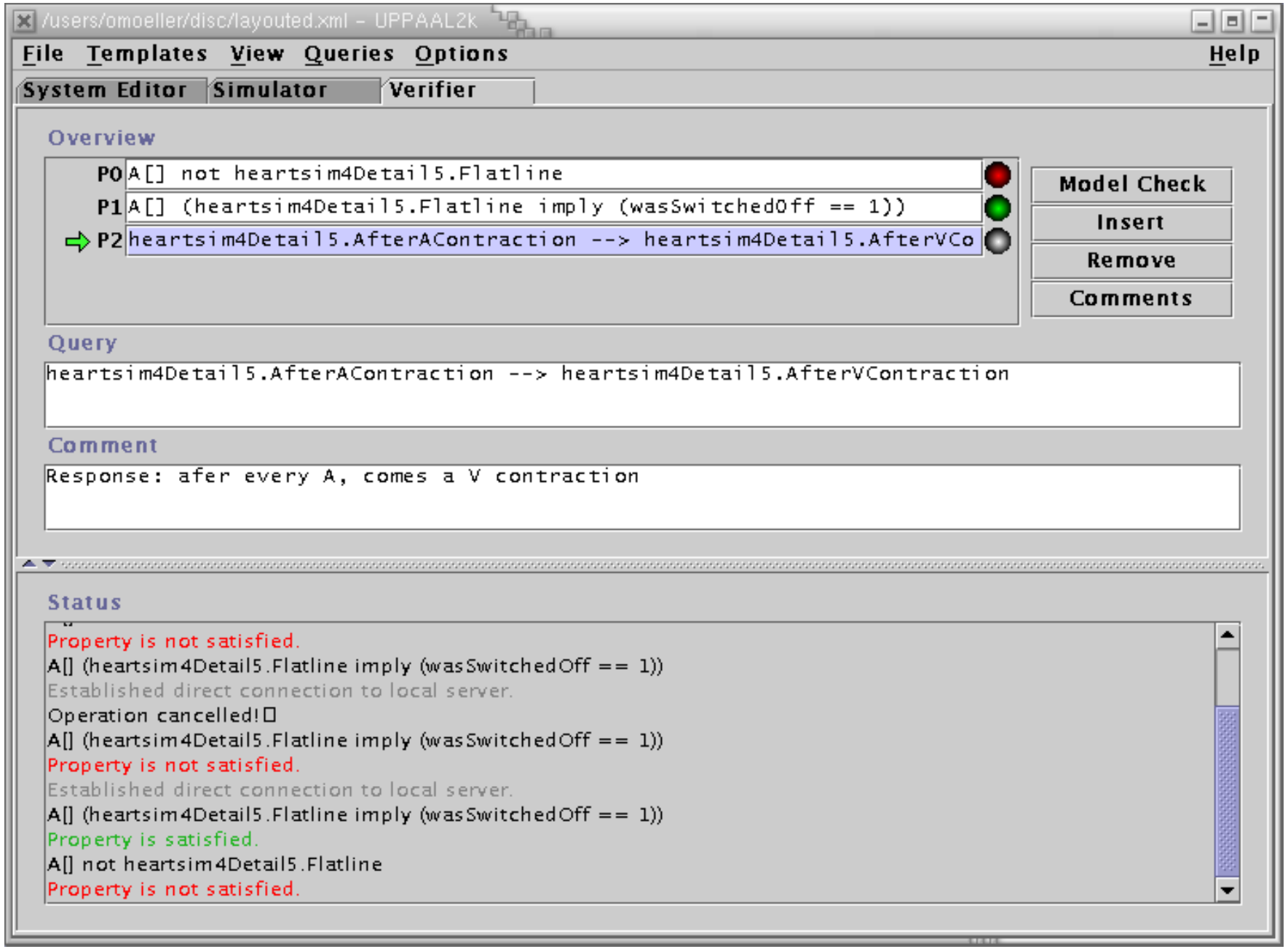
Objectives:

- UML Real-Time profile
- WOODDES methodology & tool platform




/users/omoeller/disc/layouted.xml – UPPAAL2k
File Templates View Queries Options Help
System Editor Simulator Verifier
Select Mode
Transition Mode
Drag out
layouted
Global declarations
globalKickoff12
pacemaker2subComp
pacemaker2subComp
pacemaker2subComp
pacemaker2subComp
pacemaker2subComp
pacemaker2subComp
heartsim4Detail5
heartsim4
programmersim3
pacemaker2
Declarations
Process assignments
System definition
AVI_A_TIME := 0, AVI_V_TIME := 0
L25
L35
L18
L4
IDLE
PcAVIENTRYtrpcmkr2?
PcIdlENTRYtrpcmkr2?
PcVVIENTRYtrpcmkr2?
PcOAVIENTRYtrpcmkr2sbCmpt6!
PcOIdlENTRYtrpcmkr2sbCmpt6!
PcOVVIENTRYtrpcmkr2sbCmpt6!
PcOVVTENTRYtrpcmkr2sbCmpt6!
V_listening :=


/users/omoeller/disc/layouted.xml – UPPAAL2k
File Templates View Queries Options Help
System Editor Simulator Verifier
Drag out
Enabled Transitions
(pacemaker2subComponent6AVIMode9.1,
Next Reset
Simulation Trace
(heartsim4.2, heartsim4Detail5.1)
(Detail, AContraction, Modeswitch, L7, ID
(pacemaker2.5, pacemaker2subCompon
(Detail, AContraction, Modeswitch, subC
(pacemaker2subComponent6.3, pacemal
(Detail, AContraction, Modeswitch, subC
(pacemaker2subComponent6AVIMode9.2
(Detail, AContraction, Modeswitch, subC
(pacemaker2subComponent6AVIMode9.1
Trace File:
Prev Next Replay
Open Save Random
Slow Fast
Drag out
Variables
wasSwitchedC
V_listening =
A_LISTENING_
triggerVar1 =
triggerVar2 =
triggerVar3 =
triggerVar4 =
triggerVar5 =
triggerVar7 =
VVI_TIME = 0
VVT_TIME = 0
AVI_A_TIME =
AVI_V_TIME =
HEART_TIME =
PROGRAMMER
VVT_TIME = V
AVI_A_TIME =
AVI_V_TIME =
HEART_TIME =
PROGRAMMER
AVI_A_TIME =
AVI_V_TIME =
HEART_TIME =
PROGRAMMER
AVI_V_TIME =
HEART_TIME =
PROGRAMMER
HEART_TIME =
PROGRAMMER
PROGRAMMER
L57
x tSglNR9!
triggerVar7 := triggerVar7 - 1
x tSglNR11?
IMd9VPrt11?
IDLE
r7 := triggerVar7 - 1
x tSglNR11?
AVI_APace_Done
AVI_Refracto
V_listening :=
Refractory
Ventr
AVI_
x tSglNR11?
triggerVar7 := triggerVar7 - 1


/users/omoeller/disc/layouted.xml - UPPAAL2k
File Templates View Queries Options Help
System Editor Simulator Verifier
Overview
P0 A[] not heartsim4Detail5.Flatline
P1 A[] (heartsim4Detail5.Flatline imply (wasSwitchedOff == 1))
P2 heartsim4Detail5.AfterAContraction --> heartsim4Detail5.AfterVCo
Model Check
Insert
Remove
Comments
Query
heartsim4Detail5.AfterAContraction --> heartsim4Detail5.AfterVContraction
Comment
Response: afer every A, comes a V contraction
Status
Property is not satisfied.
A[] (heartsim4Detail5.Flatline imply (wasSwitchedOff == 1))
Established direct connection to local server.
Operation cancelled!
A[] (heartsim4Detail5.Flatline imply (wasSwitchedOff == 1))
Property is not satisfied.
Established direct connection to local server.
A[] (heartsim4Detail5.Flatline imply (wasSwitchedOff == 1))
Property is satisfied.
A[] not heartsim4Detail5.Flatline
Property is not satisfied.