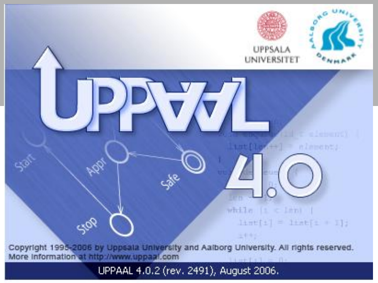
Symbolic and Statistical Model Checking in UPPAAL



Alexandre David
Kim G. Larsen

Marius Mikucionis, Peter Bulychev, Axel Legay, Dehui Du, Guangyuan Li, Danny B. Poulsen, Amélie Stainer, Zheng Wang

CAV11, FORMATS11, PDMC11, QAPL12, LPAR12, NFM12, iWIGP12, RV12, FORMATS12, HBS12, ISOLA12, SCIENCE China

Overview

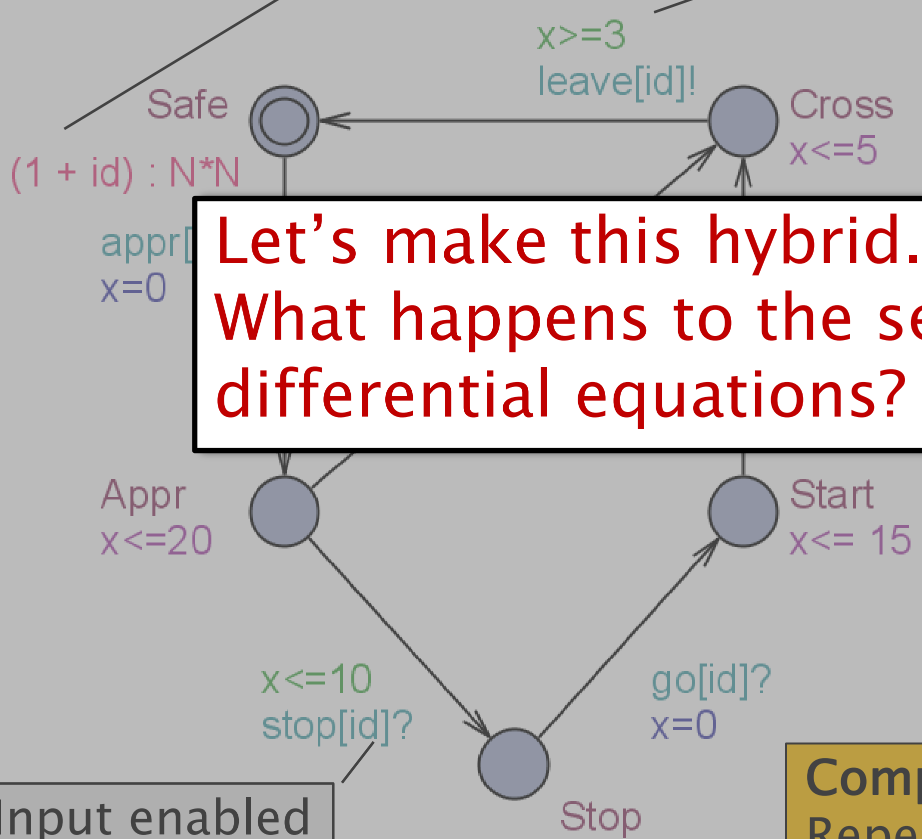
- Stochastic Hybrid Automata
- Biological Oscillator
  - Continuous vs. Stochastic Models
- Parameter Optimization – ANOVA
  - Energy Aware Building
- Controller Synthesis for Hybrid Systems

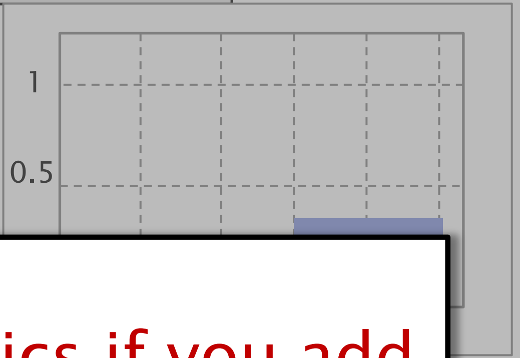
Stochastic Hybrid Automata

Stochastic Semantics of TA

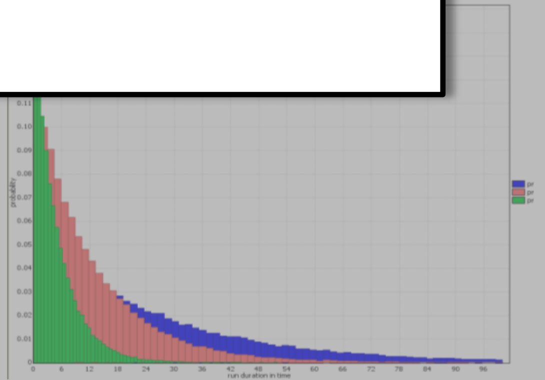
Exponential Distribution

Uniform Distribution





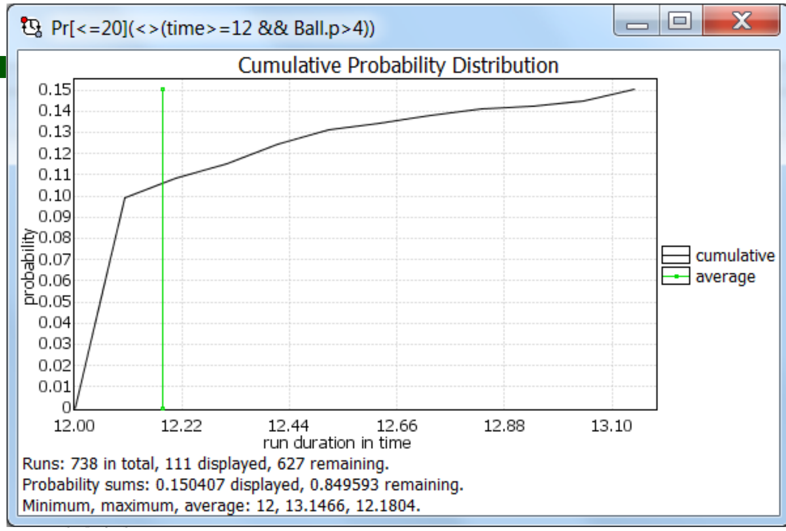
Let’s make this hybrid.
What happens to the semantics if you add differential equations?

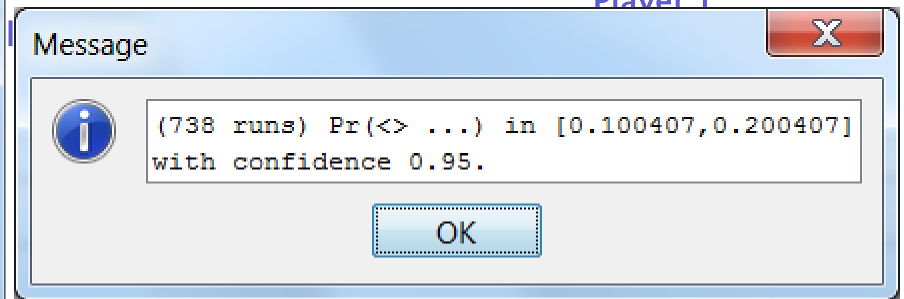


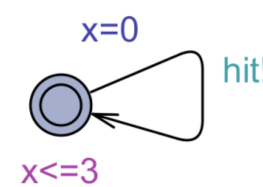
Input enabled

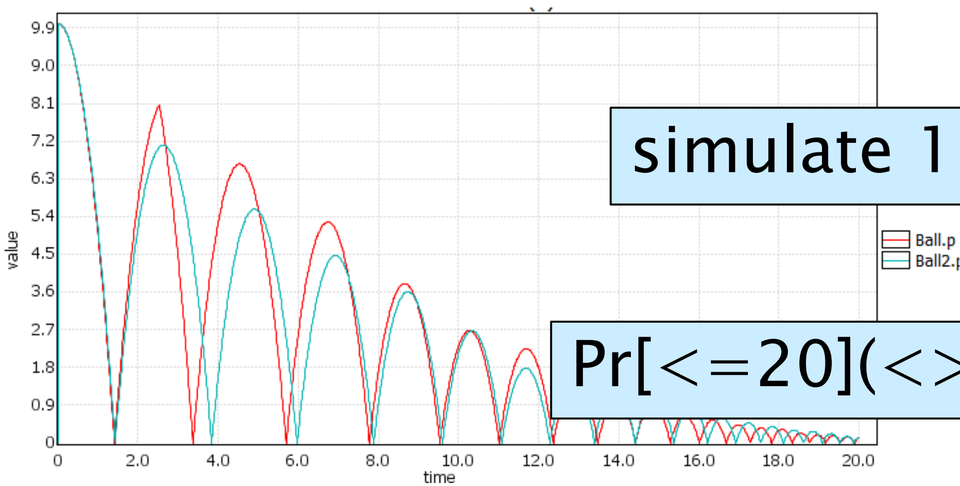
Composition =
Repeated races between components for **outputting**

Stochastic Hybrid Systems







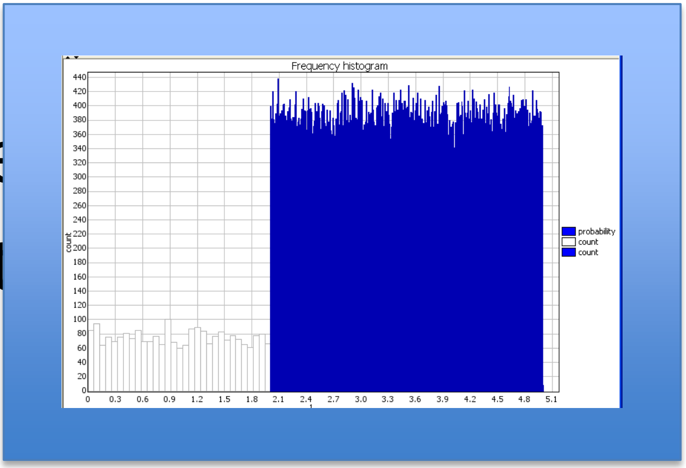


simulate 1 [<=20]{Ball1.p, Ball2.p}

Pr[<=20](<>(time>=12 && Ball.p>4))

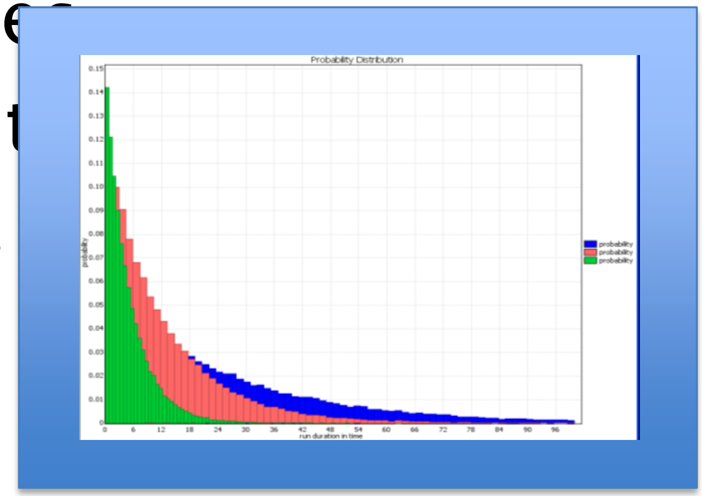
UPPAAL SMC

- **Uniform distributions (bounded delay)**
- Exponential distributions (u
- Discrete probabilistic choice
- Distribution on successor st
- Hybrid flow by use of ODEs
- + usual UPPAAL features
- Logic: MITL support.



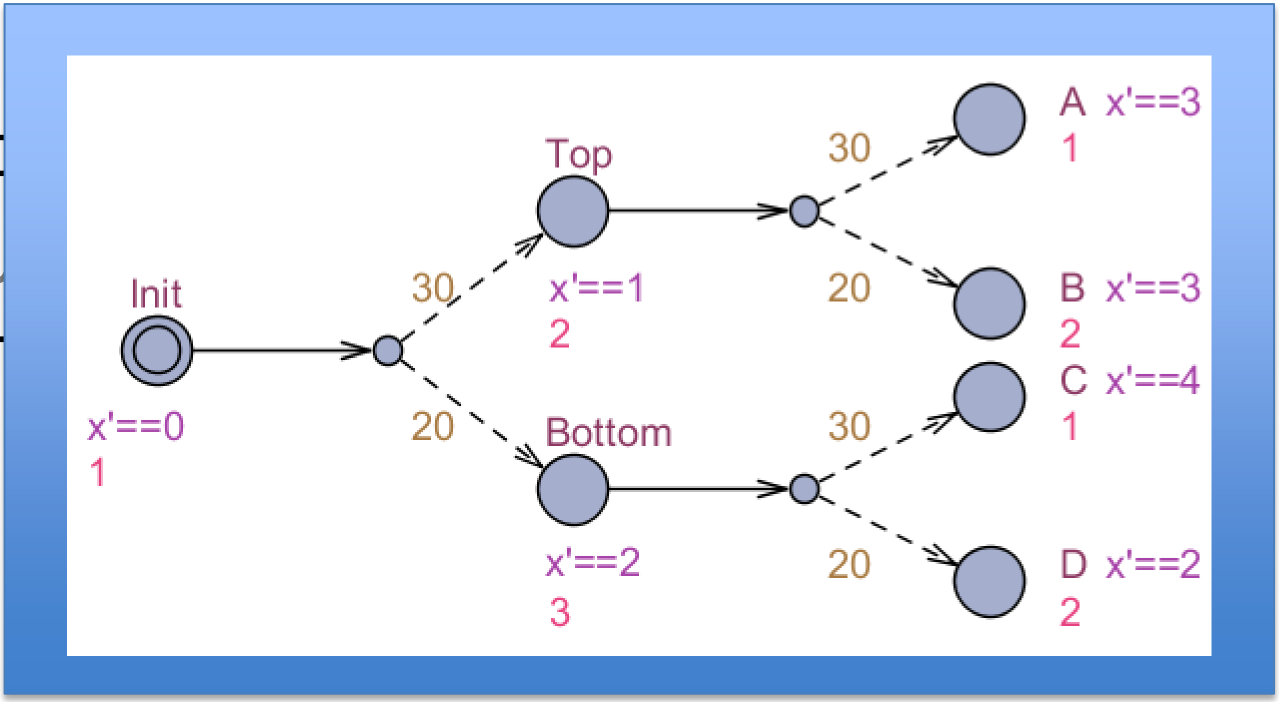
UPPAAL SMC

- Uniform distributions (bounded delay)
- **Exponential distributions (unbounded delay)**
- Discrete probabilistic choice
- Distribution on successor st
- Hybrid flow by use of ODEs
- + usual UPPAAL features
- Logic: MITL support.



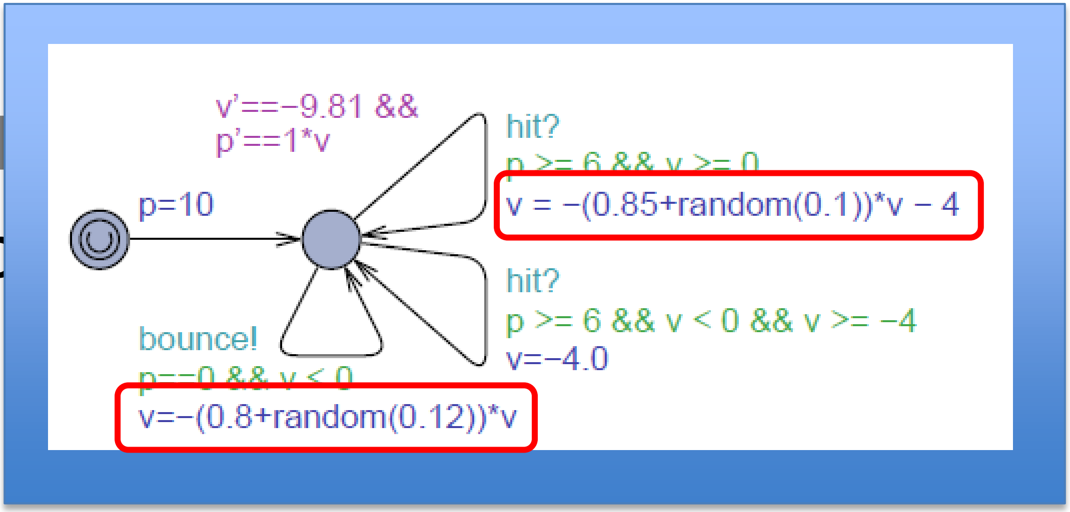
UPPAAL SMC

- Uniform distributions (bounded delay)
- Exponential distributions (unbounded delay)
- **Discrete probabilistic choices**
- Distributi
- Hybrid flo
- + usual U
- Logic: MI



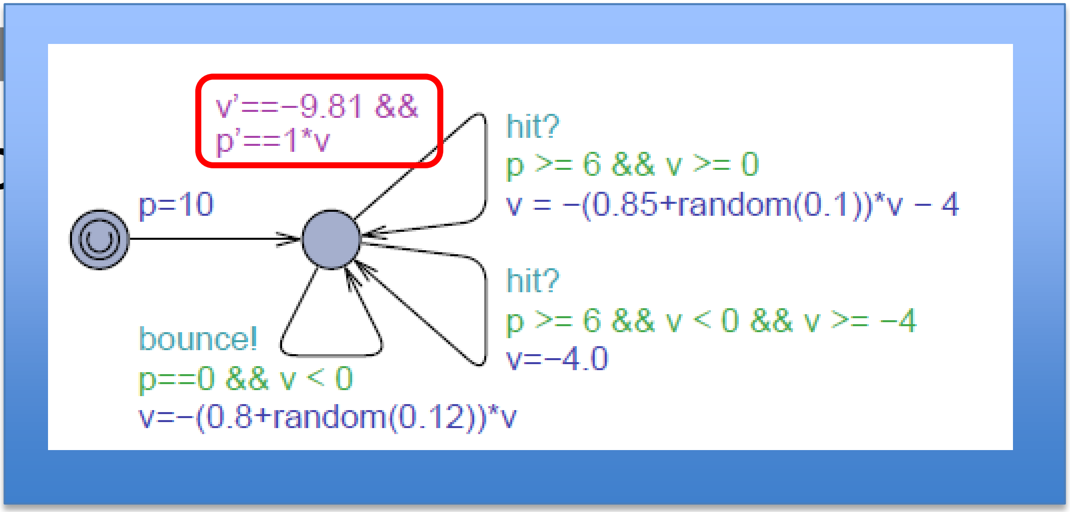
UPPAAL SMC

- Uniform distributions (bounded delay)
- Exponential distributions (unbounded delay)
- Discrete probabilistic choices
- **Distribution on successor state – random**
- Hybrid flow by
- + usual UPPAAL
- Logic: MITL sup



UPPAAL SMC

- Uniform distributions (bounded delay)
- Exponential distributions (unbounded delay)
- Discrete probabilistic choices
- Distribution on successor state – **random**
- **Hybrid flow by use of ODEs**
- + usual UPPAAL
- Logic: MITL sup



UPPAAL SMC

- Uniform distributions (bounded delay)
- Exponential distributions (unbounded delay)
- Discrete probabilistic choices
- Distribution on successor state – **random**
- Hybrid flow by use of ODEs
- \+ usual UPPAAL features
- **Logic: MITL support.**

$$\phi = \phi \vee \phi \mid \phi \wedge \phi \mid \neg\phi \mid \top \mid \bot \mid \phi \mathrm{U}_{[a;b]}\phi \mid \phi \mathrm{R}_{[a;b]}\phi \mid \mathrm{X}\phi \mid \alpha$$

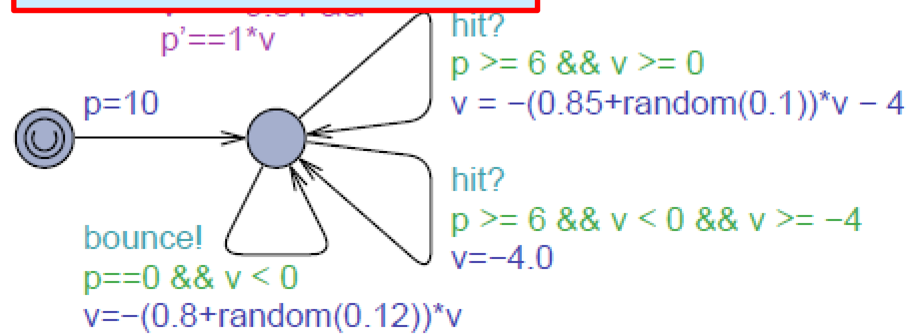
$$\Box_{\leq 1000}(\phi_{peakN} \implies \Diamond_{\leq p}\phi_{peakN})$$

Hybrid Automata

$H=(L, l_0, \Sigma, X, E, F, Inv)$
where

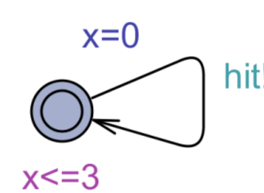
- L set of locations
- l_0 initial location
- $\Sigma = \Sigma_i \cup \Sigma_o$ set of actions
- X set of continuous variables
  valuation $\nu: X \to \mathbf{R}$ $(= R^X)$
- E set of edges (l,g,a,ϕ,l') with $g \subseteq R^X$ and $\phi \subseteq R^X \times R^X$ and $a \in \Sigma$
- For each l a delay function $F(l): R_{>0} \times R^X \to R^X$
- For each l an invariant $Inv(l) \subseteq R^X$

I/O – broadcast sync
$\Rightarrow$ input-enabled



Player 1

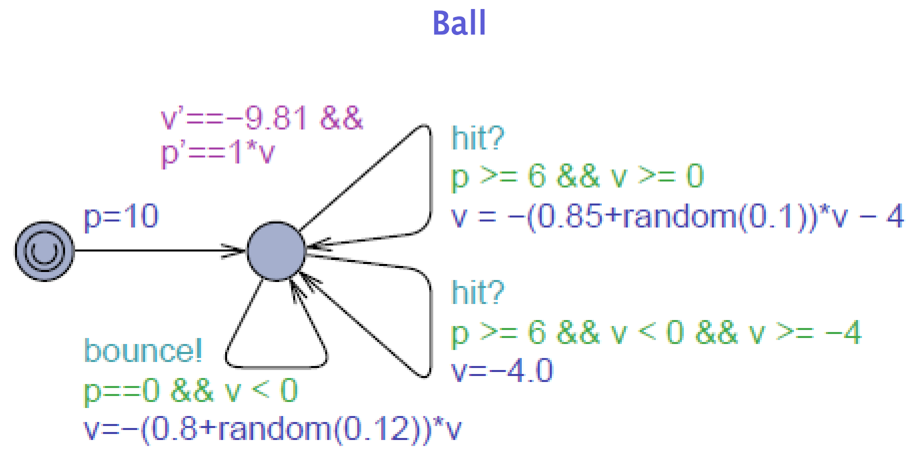
Player 2

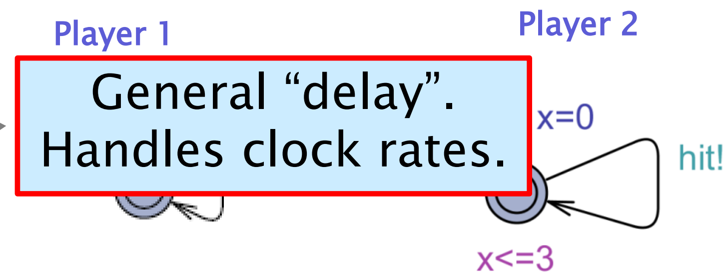


Hybrid Automata

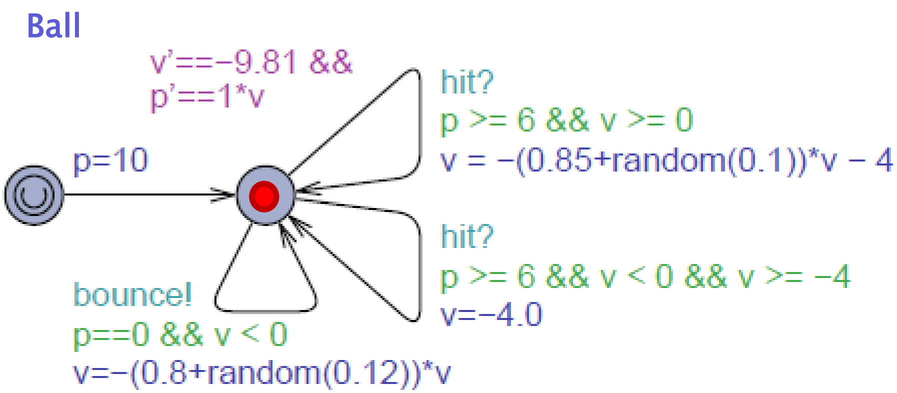
$H=(L, l_0, \Sigma, X, E, F, Inv)$
where

- L set of locations
- l_0 initial location
- $\Sigma = \Sigma_i \cup \Sigma_o$ set of actions
- X set of continuous variables
  valuation $\nu: X \to \mathbf{R}$ $(=R^X)$
- E set of edges (l,g,a,ϕ,l') with $g \subseteq R^X$ and $\phi \subseteq R^X \times R^X$ and $a \in \Sigma$
- For each l a delay function
  $F(l): R_{>0} \times R^X \to R^X$
- For each l an invariant
  $Inv(l) \subseteq R^X$





Hybrid Automata



$$
\begin{array}{ll}
(p=10, v=0) \xrightarrow{d} (p = 10 - 9.81/2d^2, v = -9.81d) & \\
\xrightarrow{\text{bounce!}} & (p=0, v=14.02 \cdot 0.83) \text{ at } d = 1.43 \\
\xrightarrow{d} & (p=6.92, v=0) \text{ at } d = 1.18 \\
\xrightarrow{d} & (p=0, v=11.51) \text{ at } d = 1.18 \\
\xrightarrow{\text{bounce!}} & \ldots
\end{array}
$$

Semantics

- **States**

  (l,ν) where $\nu \in R^X$

- **Transitions**

  $(l,\nu) \rightarrow_d (l,\nu')$ where

  $\nu' = F(l)(d,\nu)$

  provided $\nu' \in Inv(l)$

  $(l,\nu) \rightarrow_a (l',\nu')$ if

  there exists $(l,g,a,\phi,l') \in E$

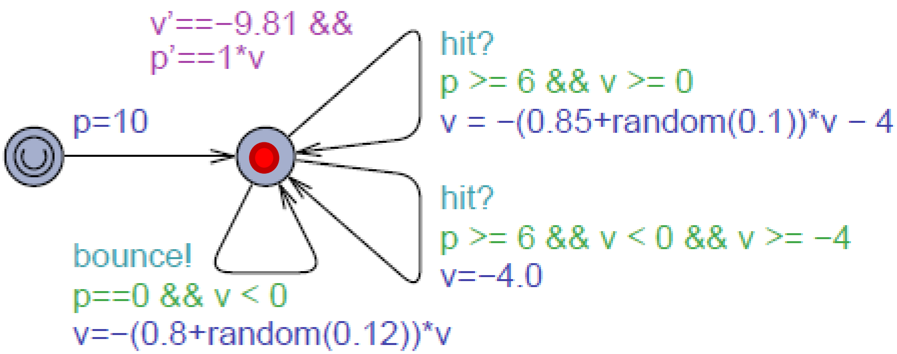
  with $\nu \in g$ and

  $(\nu,\nu') \in \phi$ and

  $\nu' \in Inv(l')$

Stochastic Hybrid Automata

Ball

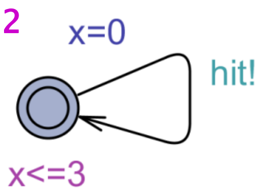


$$(p = 10, v = 0) \xrightarrow{d} (p = 10 - 9.81/2d^2, v = -9.81d)$$

$$\xrightarrow{\text{bounce!}} (p = 0, v = 14.02 \cdot 0.83) \text{ at } d = 1.43$$

Player 1

Player 2



$$Pr_1[hit!\, bounce!] = \int_{t=0}^{t=1.43} 2.5\, e^{-2.5t}\, dt$$

$$= [-e^{-2.5t}]_0^{1.43} = 0.97$$

$$Pr_2[hit!\, bounce!] = \int_{t=0}^{t=1.43} {}^1\!/_3\, dt$$

$$= [{}^1\!/_3\ t\,]_0^{1.43} = 0.48$$

Stochastic Semantics

For each state $s=(l,\nu)$

Delay density function*

μ_s: $R_{>0} \rightarrow R$

Output Probability Function

γ_s: $\Sigma_o \rightarrow [0,1]$

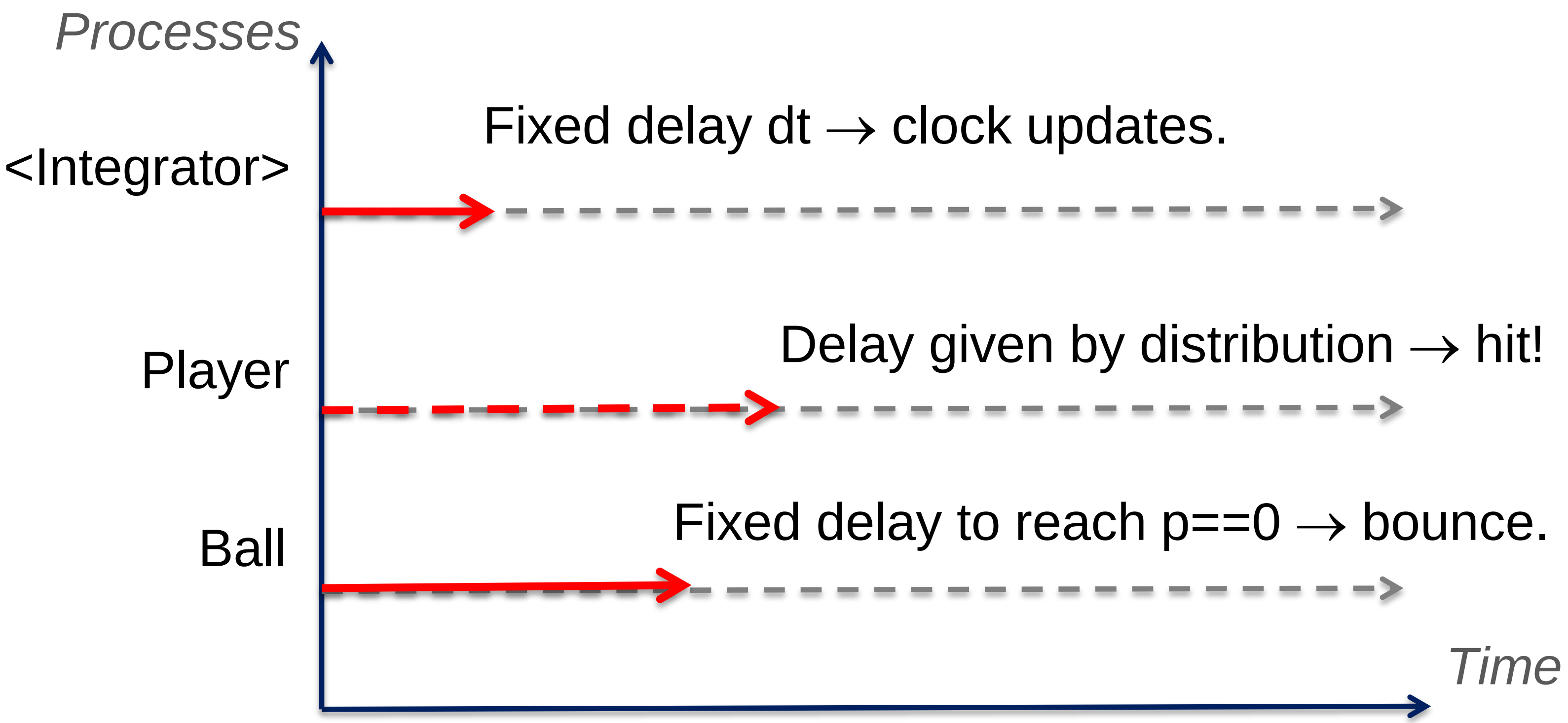
Next-state density function*

$\eta_{a\ s}$: $St \rightarrow R$

where $a \in \Sigma$.

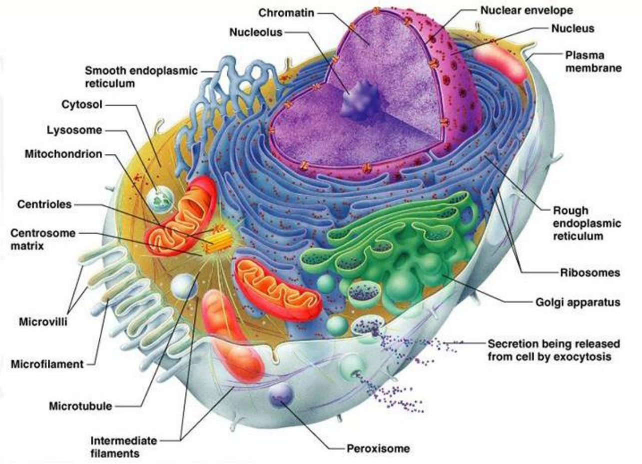
* Dirac's delta functions for deterministic delays / next state

Solving ODEs/Stochastic Semantics



Race between processes.
Choice of dt and clock updates can be changed (solver).

Biological Oscillator



A Biological Oscillator

- Circadian oscillator.
  N. Barkai and S. Leibler. Biological rhythms: Circadian clocks limited by noise. Nature, 403:267–268, 2000
- Two ways to model:
  1. Stochastic model that follow the reactions.
  2. Continuous model solving the ODEs.
- Analysis:
  - Evaluate time between peaks.
  - The continuous model is the limit behavior of the stochastic model.
  - Use frequency analysis for comparison.

Stochastic Model

$$\mathrm{A} + \mathrm{D}_A \xrightarrow{\gamma_A} \mathrm{D}'_A \qquad \mathrm{M}_A \xrightarrow{\beta_A} \mathrm{M}_A + \mathrm{A}$$

$$\mathrm{D}'_A \xrightarrow{\theta_A} \mathrm{D}_A + \mathrm{A} \qquad \mathrm{M}_R \xrightarrow{\beta_R} \mathrm{M}_R + \mathrm{R}$$

$$\mathrm{A} + \mathrm{D}_R \xrightarrow{\alpha_R} \mathrm{D}'_R \qquad \mathrm{A} + \mathrm{R} \xrightarrow{\gamma_C} \mathrm{C}$$

$$\mathrm{D}'_R \xrightarrow{\theta_R} \mathrm{D}_R + \mathrm{A} \qquad \mathrm{C} \xrightarrow{\delta_A} \mathrm{R}$$

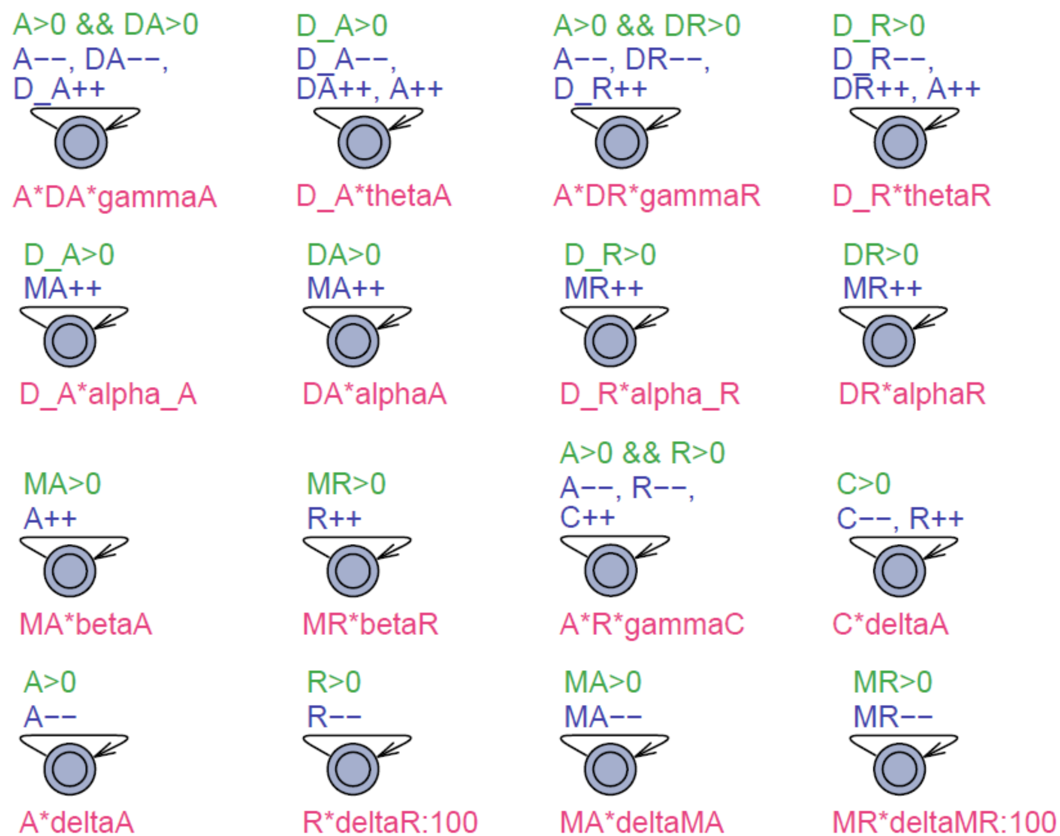
$$\mathrm{D}'_A \xrightarrow{\alpha'_A} \mathrm{M}_A + \mathrm{D}'_A \qquad \mathrm{A} \xrightarrow{\delta_A} \emptyset$$

$$\mathrm{D}_A \xrightarrow{\alpha_A} \mathrm{M}_A + \mathrm{D}_A \qquad \mathrm{R} \xrightarrow{\delta_R} \emptyset$$

$$\mathrm{D}'_R \xrightarrow{\alpha'_R} \mathrm{M}_R + \mathrm{D}'_R \qquad \mathrm{M}_A \xrightarrow{\delta_{M_A}} \emptyset$$

$$\mathrm{D}_R \xrightarrow{\alpha_R} \mathrm{M}_R + \mathrm{D}_R \qquad \mathrm{M}_R \xrightarrow{\delta_{M_R}} \emptyset$$

(a) Reactions.

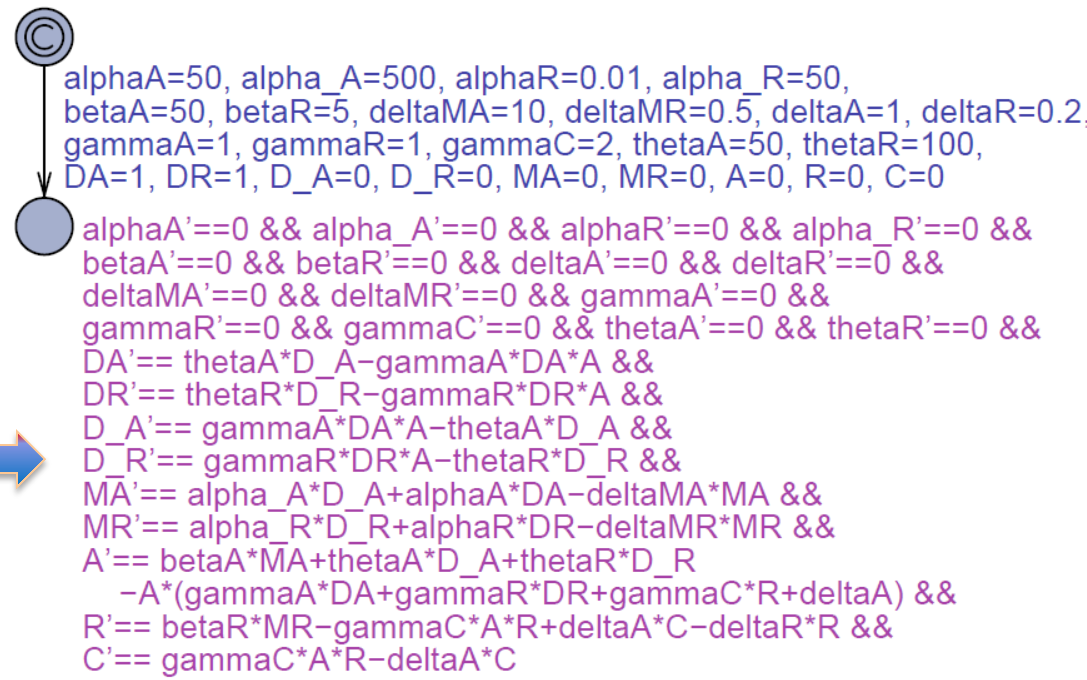


(b) UPPAAL model representation.

Continuous Model

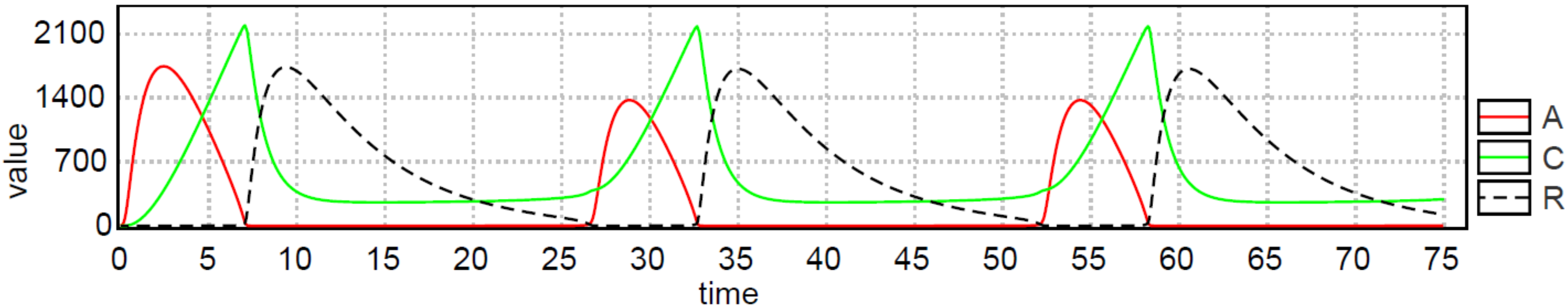
$$
\begin{aligned}
dD_A/dt &= \theta_A D'_A - \gamma_A D_A A \\
dD_R/dt &= \theta_R D'_R - \gamma_R D_R A \\
dD'_A/dt &= \gamma_A D_A A - \theta_A D'_A \\
dD'_R/dt &= \gamma_R D_R A - \theta_R D'_R \\
dM_A/dt &= \alpha'_A D'_A + \alpha_A D_A - \delta_{M_A} M_A \\
dM_R/dt &= \alpha'_R D'_R + \alpha_R D_R - \delta_{M_R} M_R \\
dA/dt &= \beta_A M_A + \theta_A D'_A + \theta_R D'_R \\
&\quad - A(\gamma_A D_A + \gamma_R D_R + \gamma_C R + \delta_A) \\
dR/dt &= \beta_R M_R - \gamma_C A R + \delta_A C - \delta_R R \\
dC/dt &= \gamma_C A R - \delta_A C
\end{aligned}
$$

(a) Ordinary differential equations.

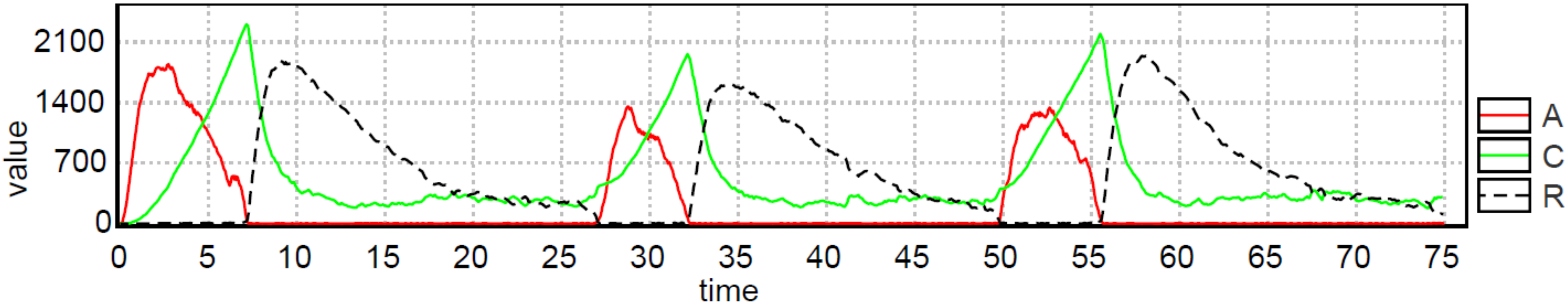


(b) UPPAAL automaton representation.

Results of Simulations



(a) ODE model simulation plot.



(b) Stochastic model simulation plot.

Frequency Domain Analysis *(Fourrier Transform)*

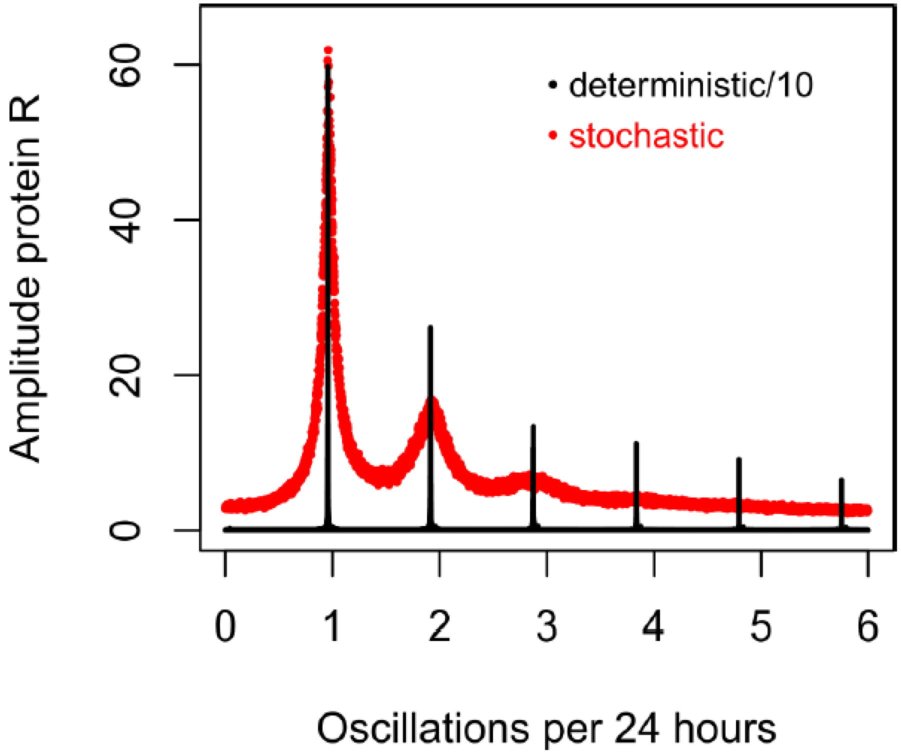


Fig. 11: Average frequency spectra of protein R. $\delta t = 2h, K = 12500$. $N = 100$ ($N = 1$) for the stochastic (deterministic) model.

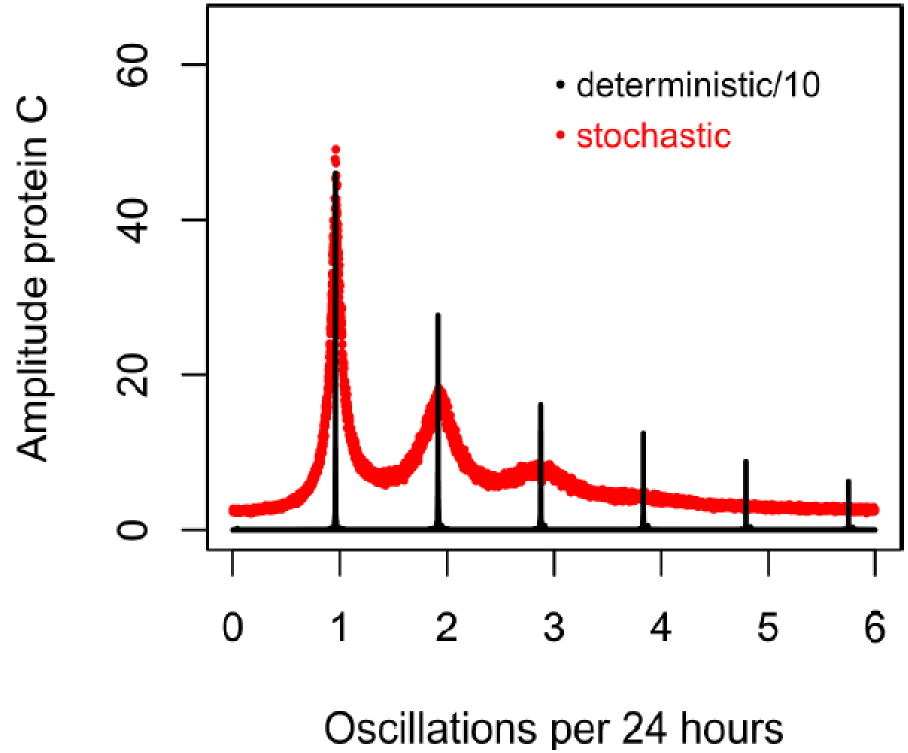
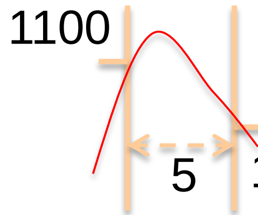
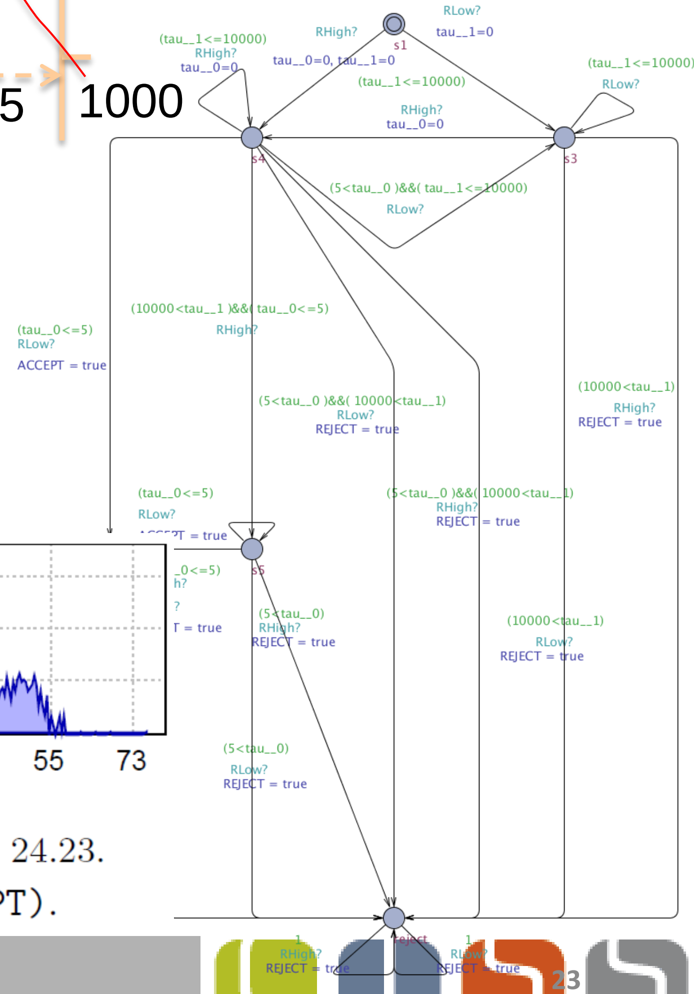


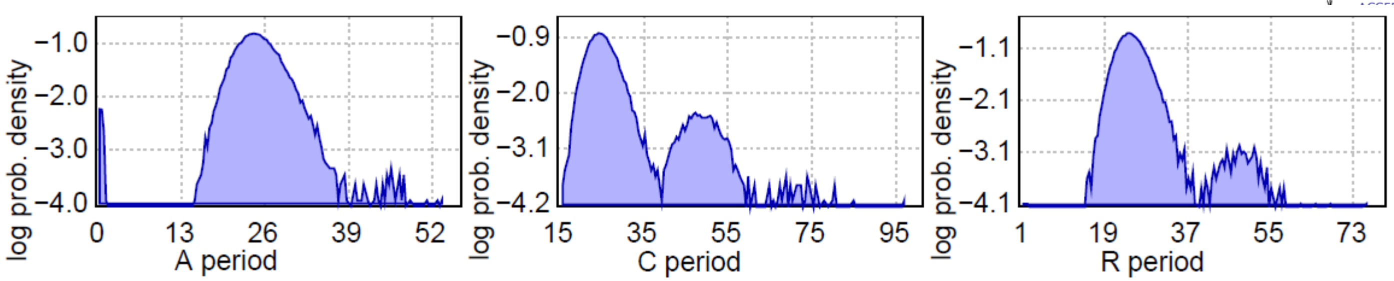
Fig. 12: Average frequency spectra of protein C. $\delta t = 2h, K = 12500$. $N = 100$ ($N = 1$) for the stochastic (deterministic) model.

Time Between Peaks

- Use the MITL formula
  `true U[<=1000] (A>1100 & true U[<=5] A<=1000).`
- Generate monitors (one shown).
- Run SMC.







(a) A, peak at 24.22. (b) C, peak at 24.21. (c) R, peak at 24.23.

Fig. 9: Estimated periods: `Pr[x<=100](<> secondPeak.ACCEPT)`.

Energy Aware Buildings

What This Work is About

- Find optimal parameters for, e.g., a controller.
  - Applied to stochastic hybrid systems.
  - Suitable for different domains: biology, avionics…

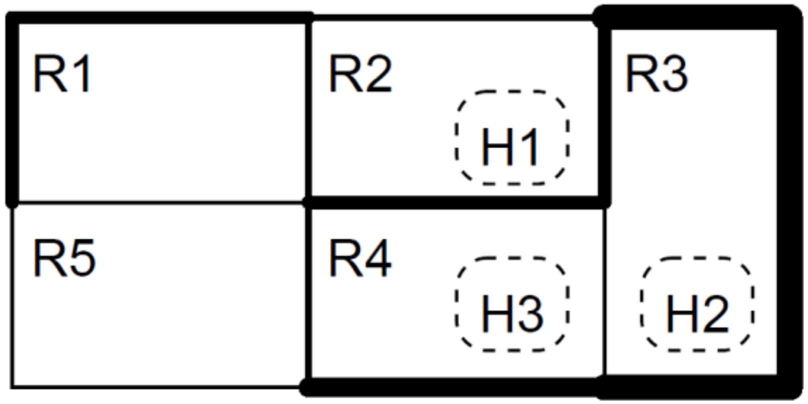
- Technique: statistical model-checking.
  - This work: Apply ANOVA to reduce the number of needed simulations.

Overview

- Energy aware buildings
  - The case-study in a nutshell
- Choosing the parameters
  - Naïve approach
- Efficiently choosing the (best) parameters
  - ANOVA

Energy Aware Buildings

- The case:
  - Building with rooms separated by doors or walls.
  - Contact with the environment by windows or walls.
  - Few transportable heat sources between the rooms.
  - Objective: maintain the temperature within range.



(a) Rooms R_i with heaters H_k.

Energy Aware Buildings

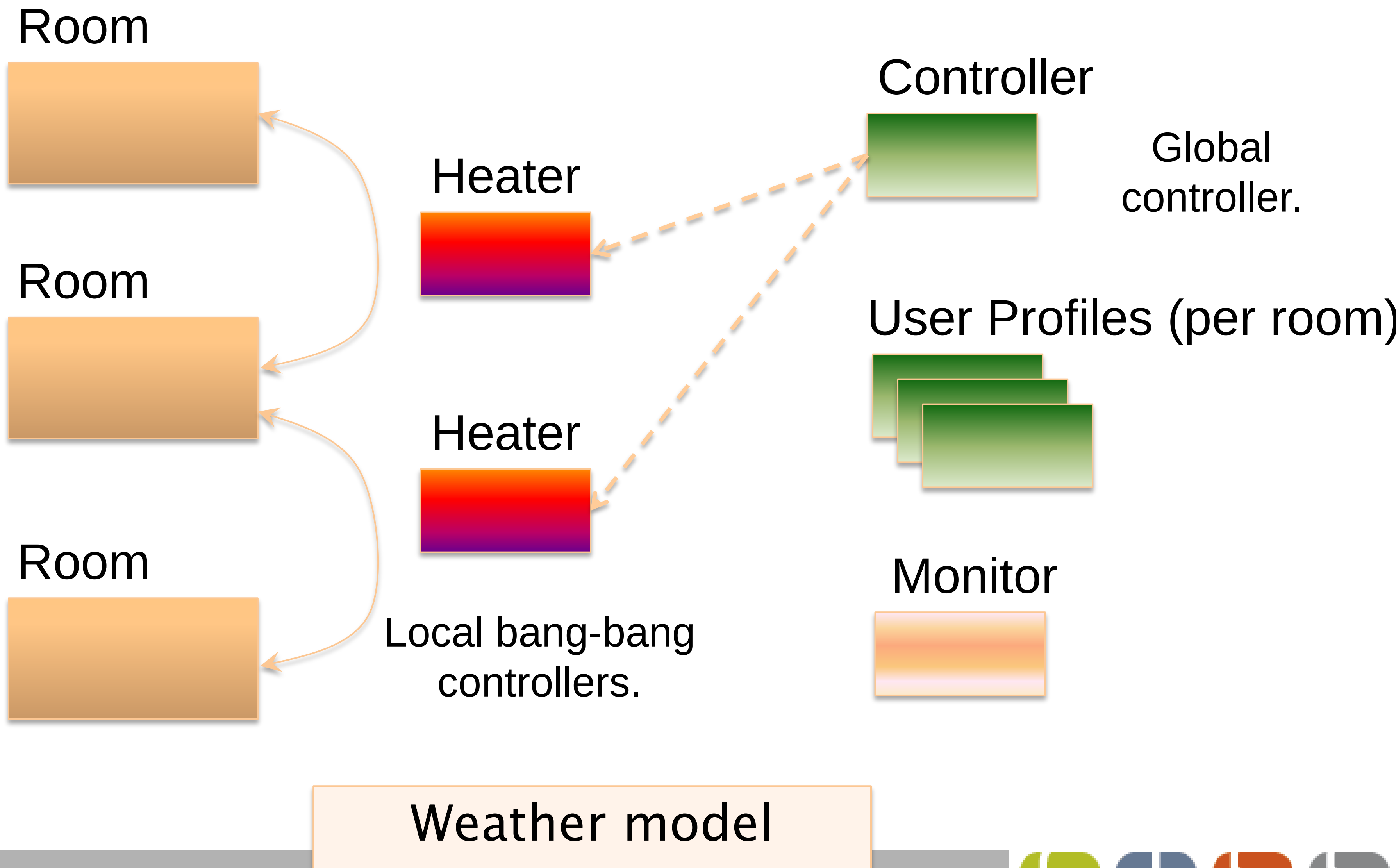
- Model:
  - Matrix of coefficients for heat transfer between *rooms*.

$$T_i' = \sum_{j \neq i} a_{i,j}(T_j - T_i) + b_i(u - T_i) + c_i h_i$$

  - Environment temperature $\rightarrow$ *weather* model.
  - Different controllers $\rightarrow$ *user* profiles.
- Goal:
  - *Optimize the controller.*

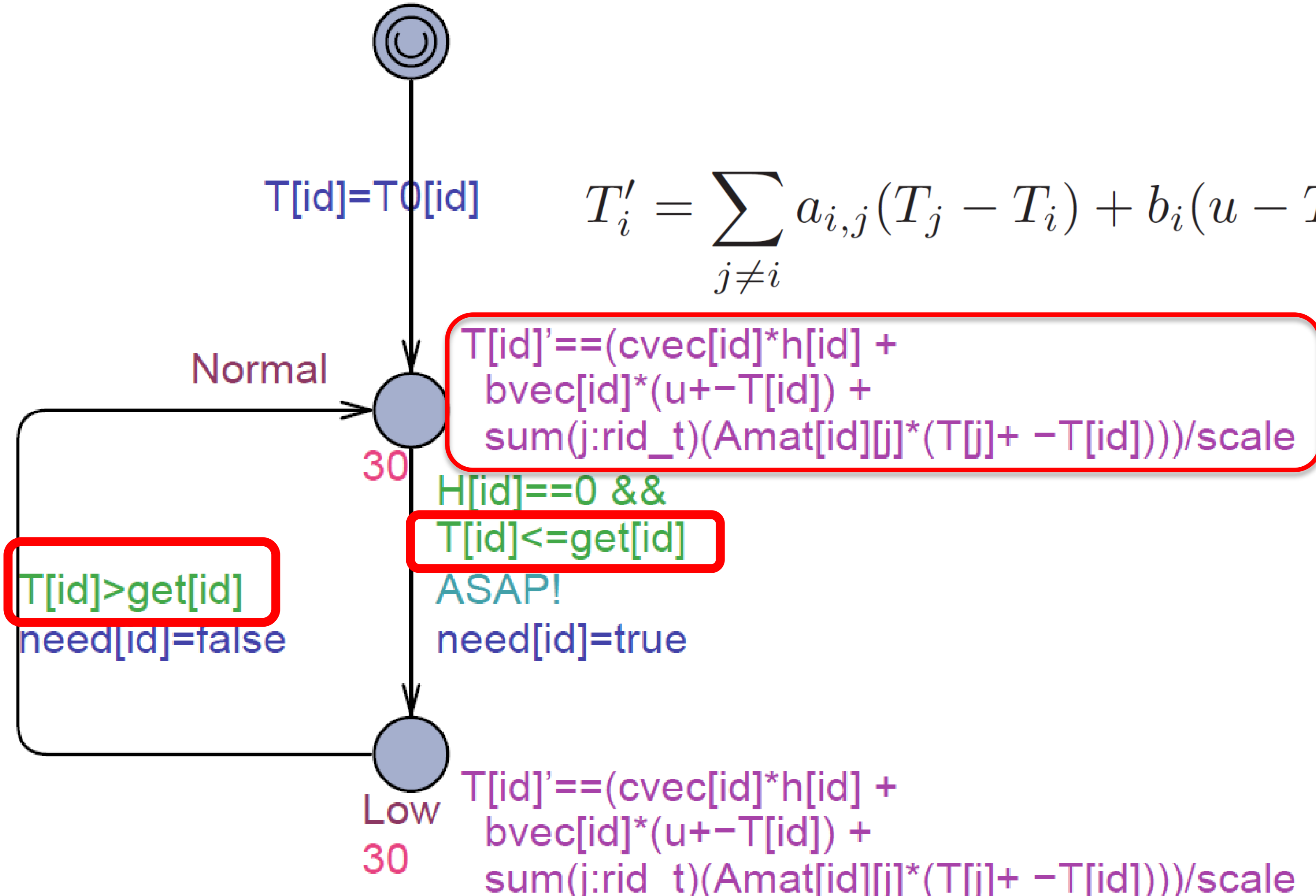
Science China 2012

Model Overview



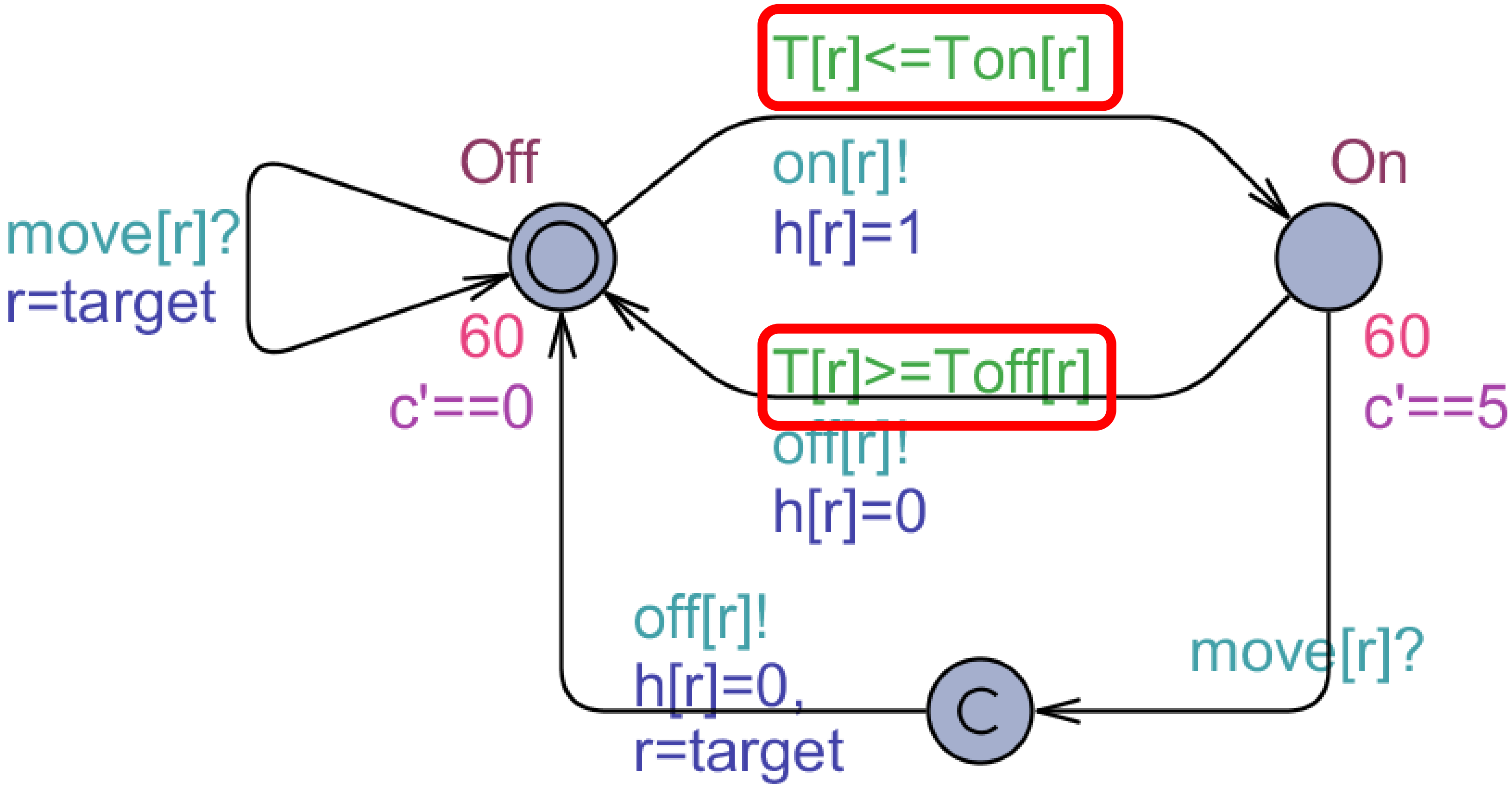
Stochastic Hybrid Model of the Room

$$T_i' = \sum_{j \neq i} a_{i,j}(T_j - T_i) + b_i(u - T_i) + c_i h_i$$



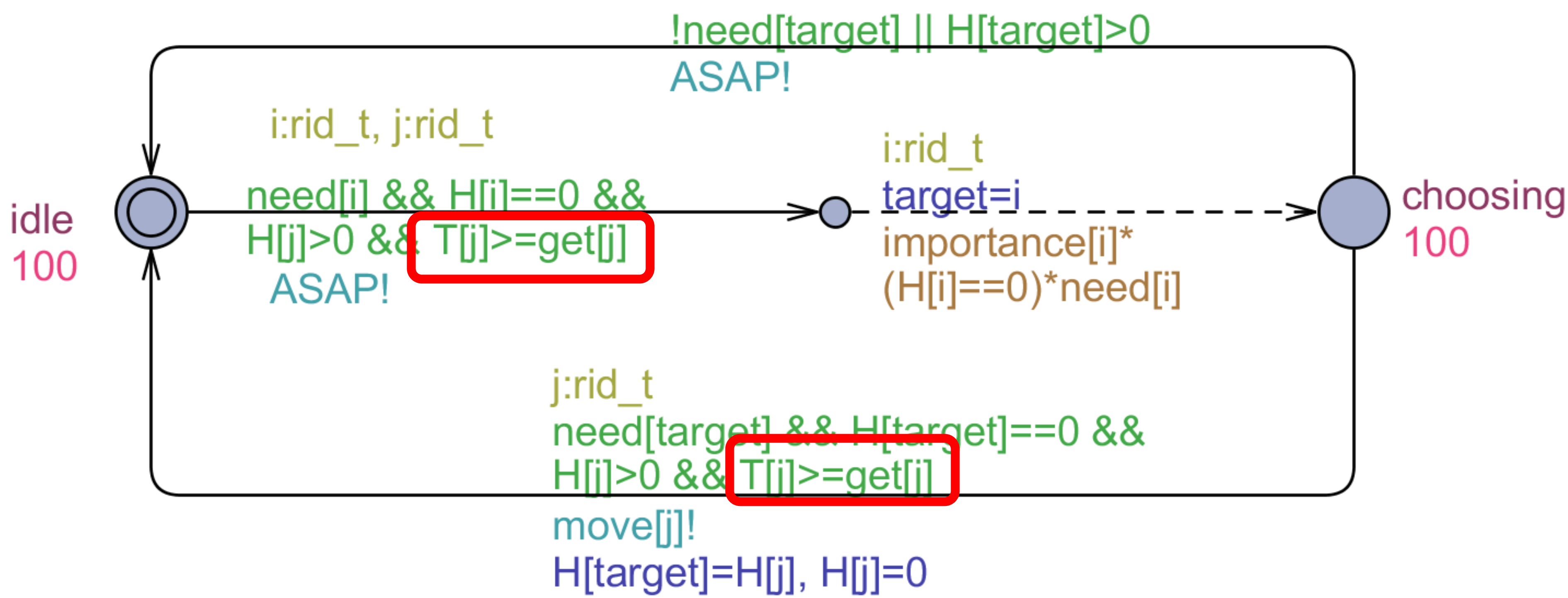
(a) Template for Room temperature.

Model of the Heater

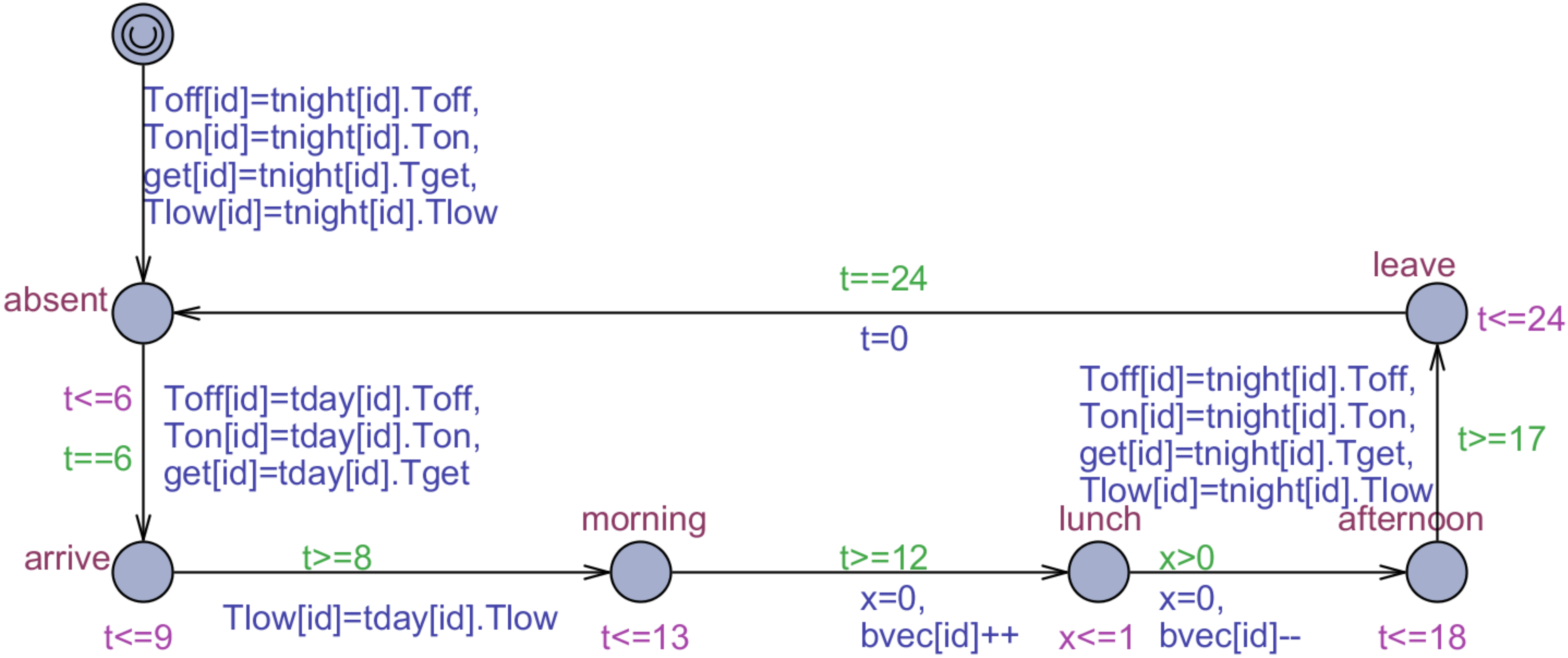


Local “bang-bang” controller.

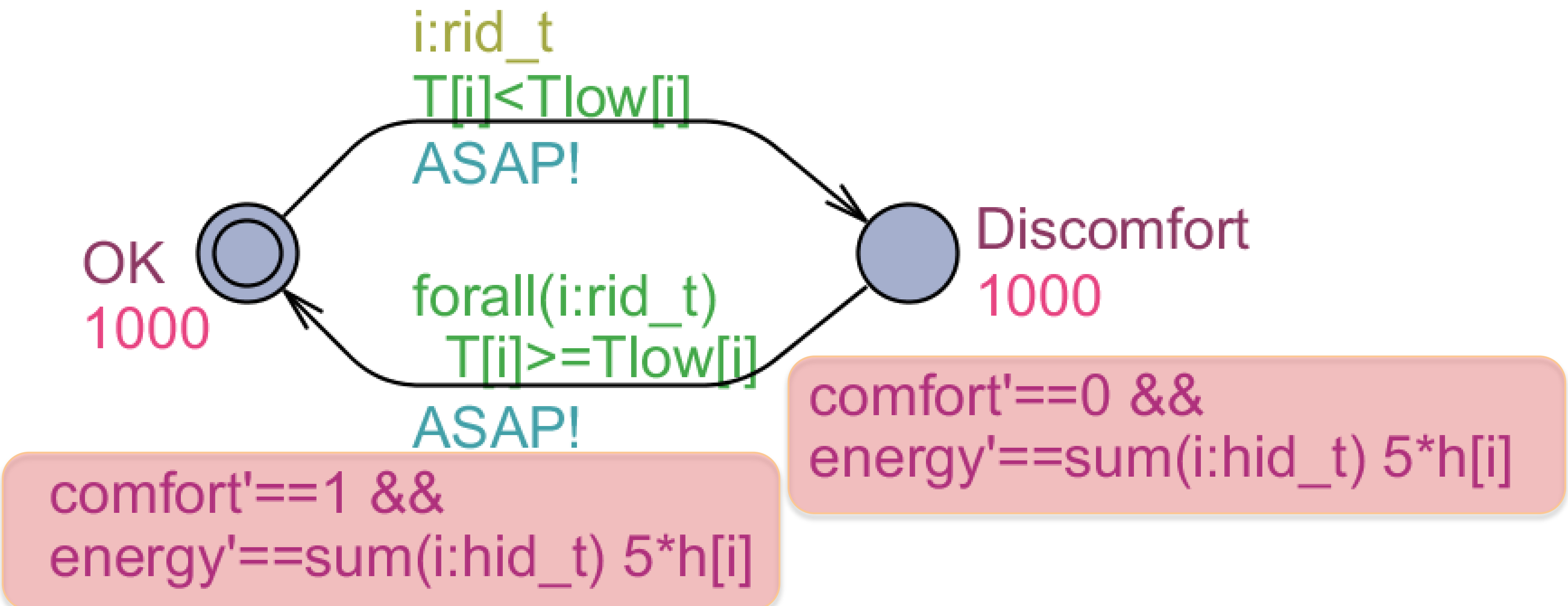
Main Controller



Dynamic User Profile



Global Monitoring



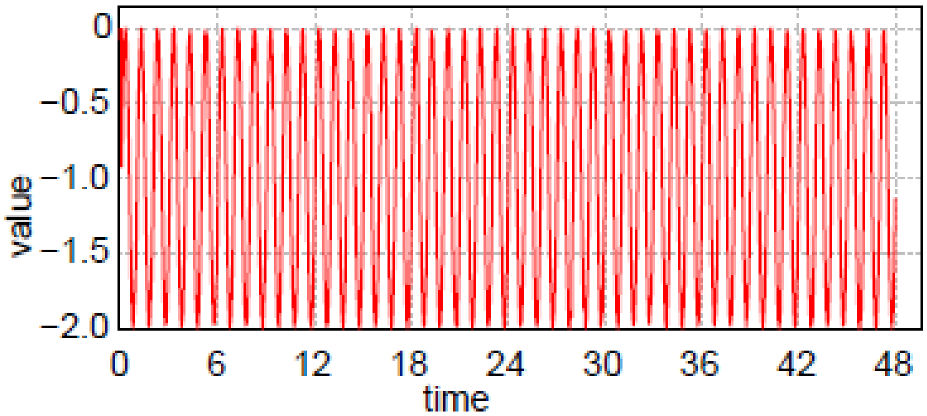
\+ Maximize comfort.

\- Minimize energy.

? Play with Ton and Tget.

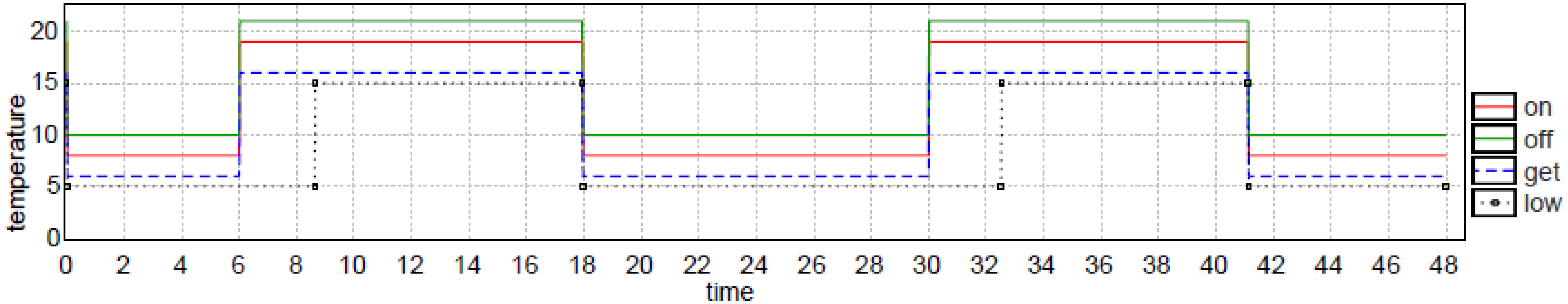
(Possible with Toff but not here).

Simulations

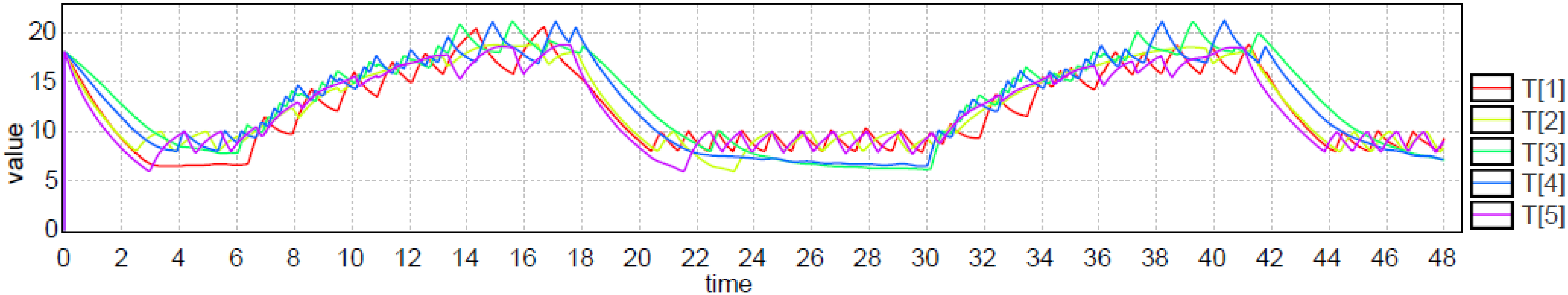


Weather Model

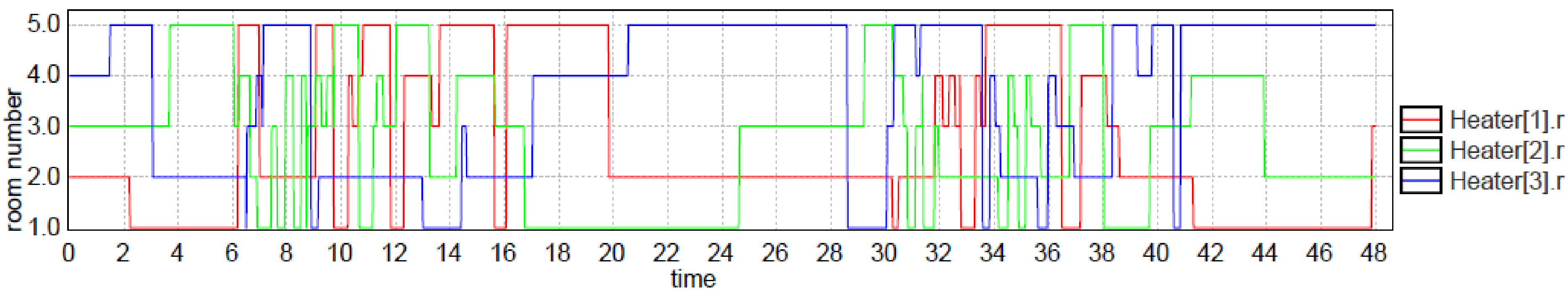
User Profile



Simulations



simulate 1 [<=2*day]{ T[1], T[2], T[3], T[4], T[5] }



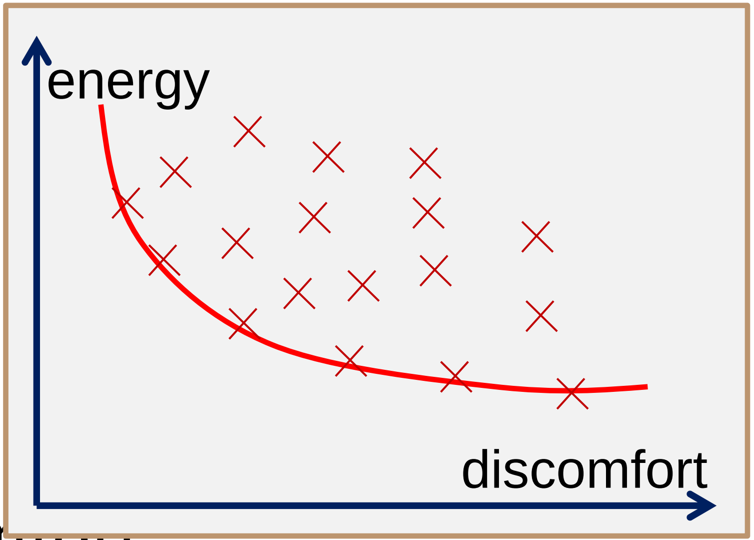
simulate 1 [<=2*day]{ Heater(1).r,Heater(2).r,Heater(3).r }

How to Pick the Parameter Values?

- $\mathrm{T_{on}}, \mathrm{T_{get}} \in [16{,}22] \rightarrow 49\ \textit{discrete choices}$.
  More if considering other parameters.
- Stochastic simulations.
  - Weather not deterministic.
  - User not deterministic (present, absent…)
- How to decide that one combination is better?
  - Probabilistic comparisons?
    49*48 comparisons * number of runs.
  - To optimize what? Discomfort or energy?

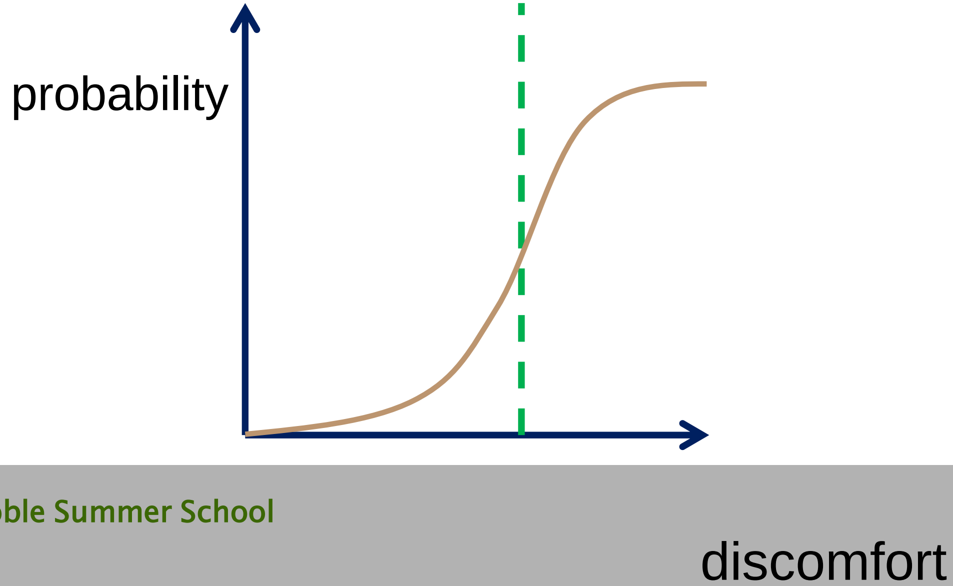
How to Pick the Parameter Values?

- Remark:
  - Stochastic hybrid system $\Rightarrow$ SMC
- Idea:
  - Generate runs.
  - Plot the result energy/comfort.
  - Pick the Pareto frontier of the means.
- How many runs do you need?
  - What's the significance of the results?

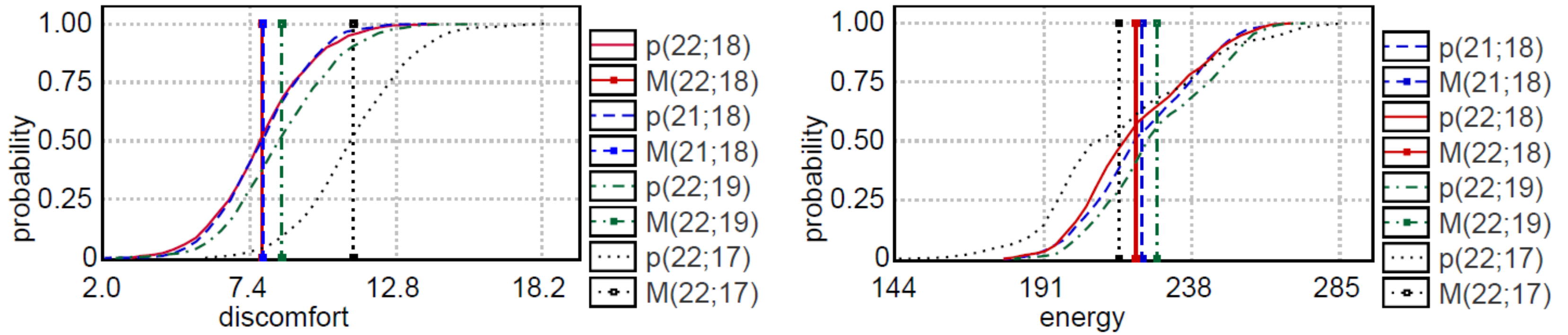


“Naïve” Solution

- Estimate the probabilities
  Pr[discomfort<=100](<> time >= 2*day)
  Pr[energy<=1000](<> time >= 2*day)
- From the obtained distributions (confidence known), compute the means.
- Pick the Pareto frontier of the means.

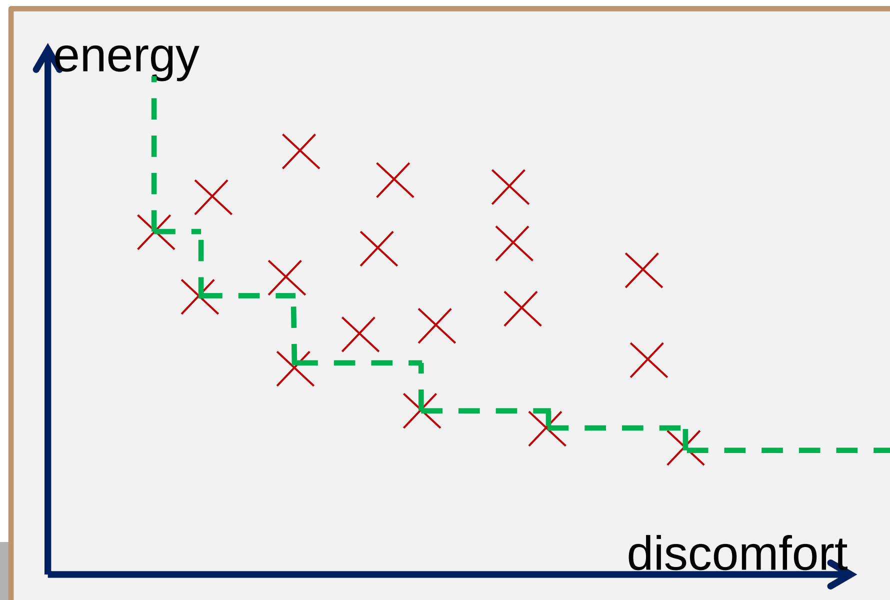


“Naïve” Approach



(a) Over discomfort. (b) Over energy.

For each (Ton,Tget)



ANOVA Method

- Compare several distributions.
  - Evaluate influence of each factor on the outcome.
- Generalization of Student's t-test.
  - Compare 2 distributions using the mean of their difference.
  - If confidence interval does not include zero, distributions are significantly different.
  - Cheaper than evaluating 2 means + on-the-fly possible.

ANOVA Method

- 2-factor factorial experiment design
  - Ton, Tget are our 2 factors.
  - Each combination gives a distribution to compare.
  - Measure cost outcome (discomfort or energy).
- ANOVA estimates a linear model and computes the influence of each factor.
  - The measure of the influence is the F-statistic.
  - This is translated into P-value, the factor significance.
  - Assume balanced experiments.

ANOVA Method

Fewer runs, more efficient than before.

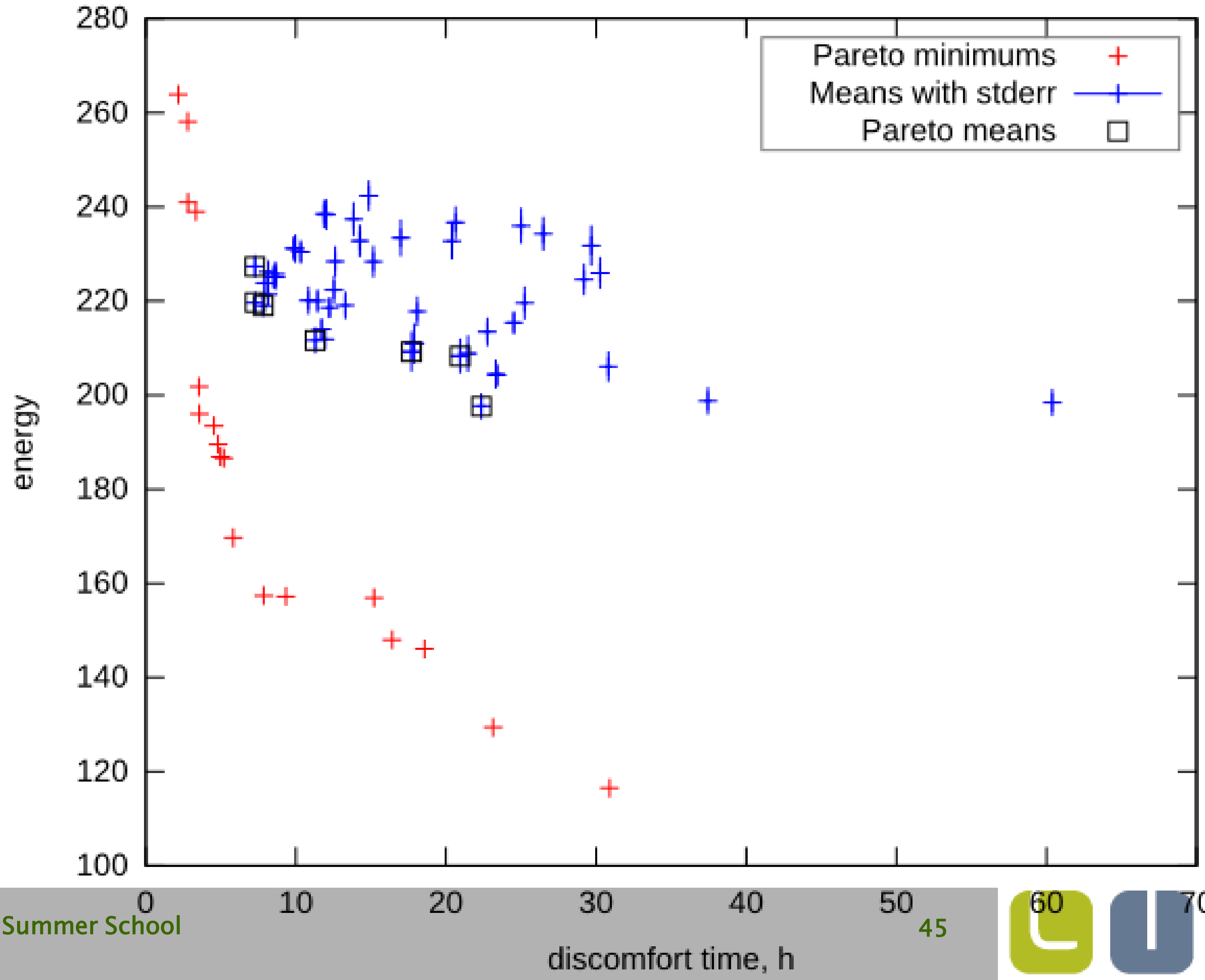
- Generate balanced measurements for each configuration to compare.
- Apply ANOVA on the data (used the tool R).
- If the factors are not significant, goto 1.
- Reuse the data and compute the confidence intervals of the means for each comparison.
- Compute the Pareto frontier.

ANOVA Results

P<0.05⇒significant

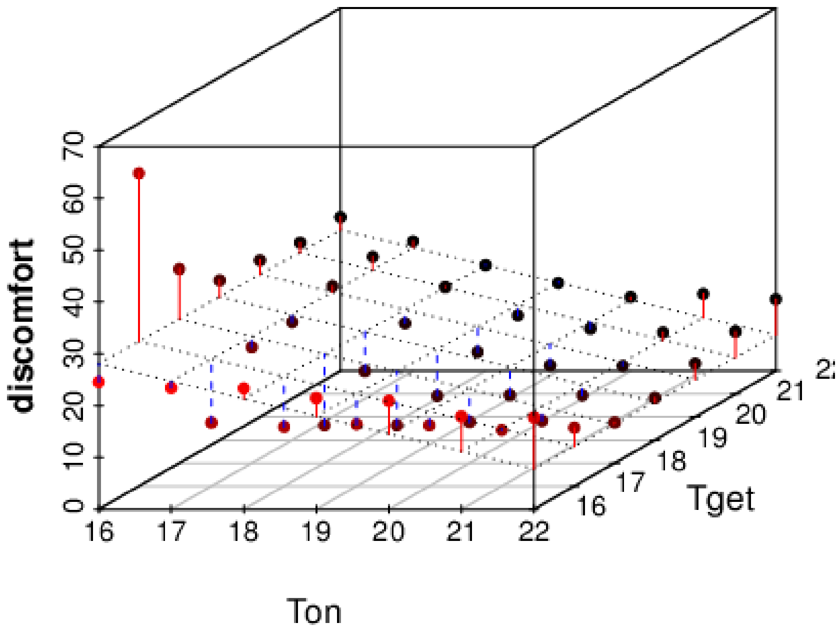
Number of runs	Factor	Discomfort time F value	Discomfort time P-value	Energy consumption F value	Energy consumption P-value
2 · 49	T_{on}	63.8874	**3.30e-12**	0.7147	0.4000
	T_{get}	0.0063	0.9369	17.5777	**6.24e-05**
	$T_{on} : T_{get}$	0.0629	0.8026	0.7181	0.3989
4 · 49	T_{on}	136.1676	**<2e-16**	1.1647	0.2818
	T_{get}	0.1537	0.6955	17.9283	**3.55e-05**
	$T_{on} : T_{get}$	0.0003	0.9869	0.0582	0.8096
8 · 49	T_{on}	315.7978	**<2e-16**	2.4425	0.1189
	T_{get}	0.1202	0.7290	35.8938	**4.76e-09**
	$T_{on} : T_{get}$	0.0096	0.9218	0.8253	0.3642
16 · 49	T_{on}	629.1384	**<2e-16**	6.5909	0.01044
	T_{get}	0.5895	0.4429	90.9612	**<2e-16**
	$T_{on} : T_{get}$	0.2852	0.5935	5.3053	0.02152
32 · 49	T_{on}	1263.5390	**<2e-16**	27.9527	**1.42e-07**
	T_{get}	1.0840	0.2980	172.3296	**<2.2e-16**
	$T_{on} : T_{get}$	0.5401	0.4625	3.2632	0.07104
64 · 49	T_{on}	2575.3208	**<2e-16**	65.6245	**7.74e-16**
		4.6682	0.0308	405.4892	**<2.2e-16**
	$T_{on} : T_{get}$	0.5949	0.4406	0.1926	0.6608

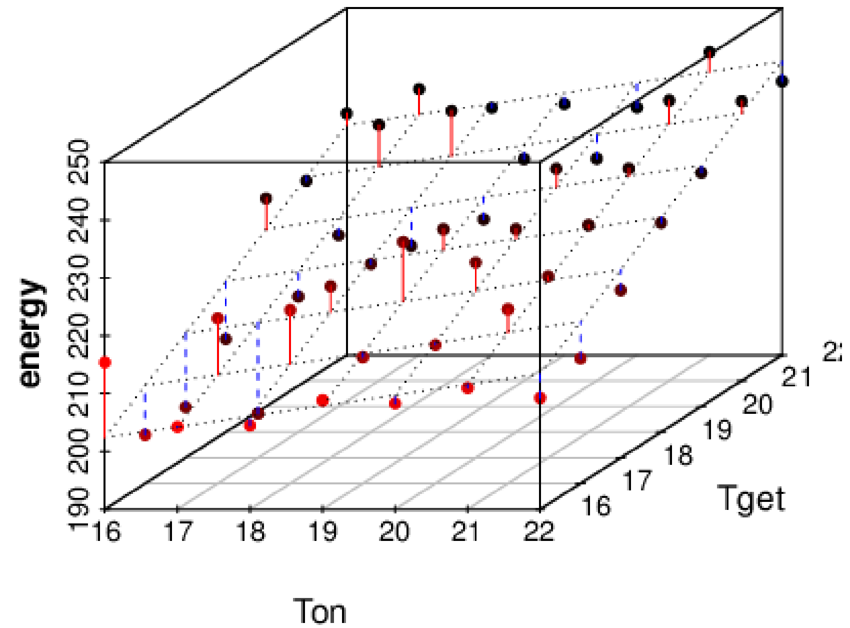
Results



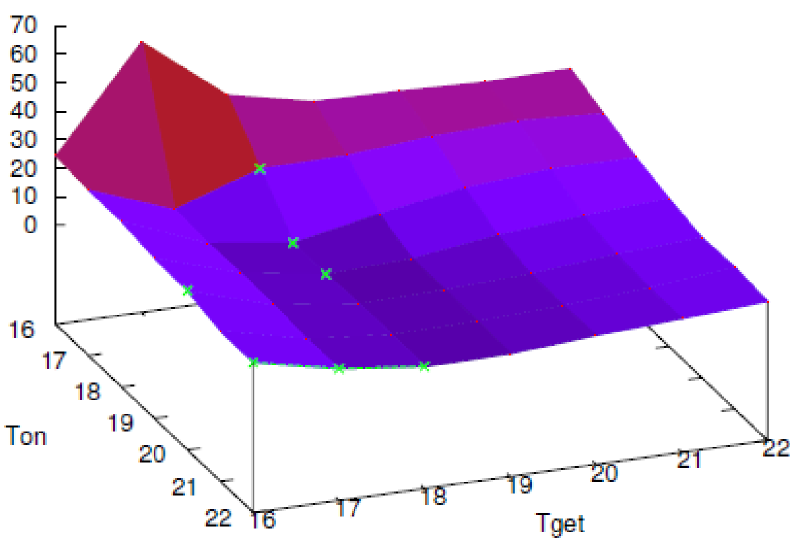
Pareto minimums
Means with stderr
Pareto means
energy
discomfort time, h
280
260
240
220
200
180
160
140
120
100
0
10
20
30
40
50
60
70

Visualization of the Cost Model

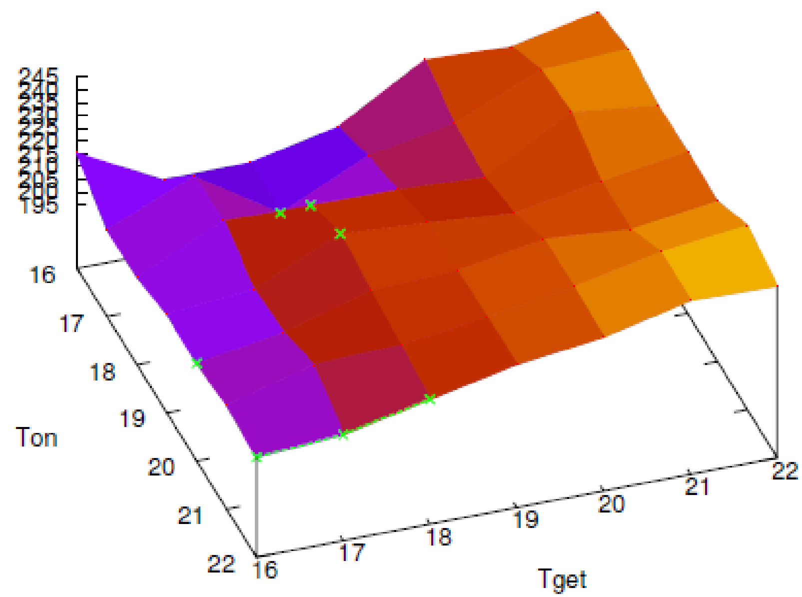




Results



(a) Discomfort.



(b) Energy.

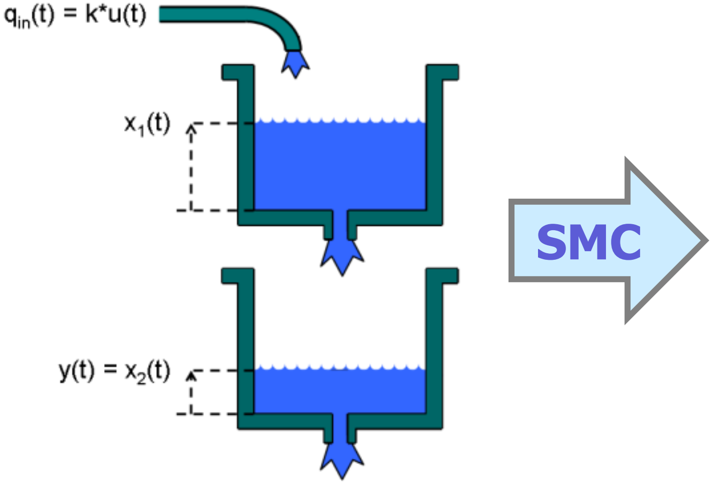
Comparison

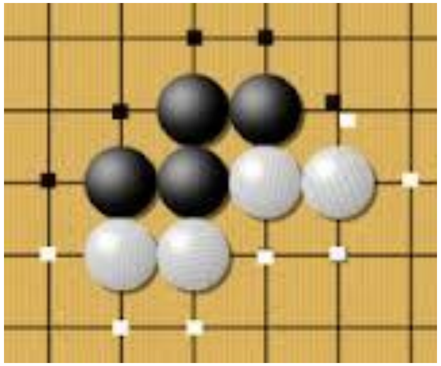
- Naïve approach:
  738 runs per evaluation per cost
  *2 (energy & discomfort) *49
  (configurations).
  $\Rightarrow$ 1h 5min
- ANOVA:
  3136 runs $\Rightarrow$ 6min 6s.
- Core i7 2600

Discussion

- Analysis of variance used sequentially to decide when there is enough data to distinguish the effect of 2 factors.
  - Efficient use of SMC.
- What if the factor has no influence?
  - Need an alternative test.
- Possible to distribute.
- Future work: Integrate ANOVA in UPPAAL

Hybrid Controller Synthesis





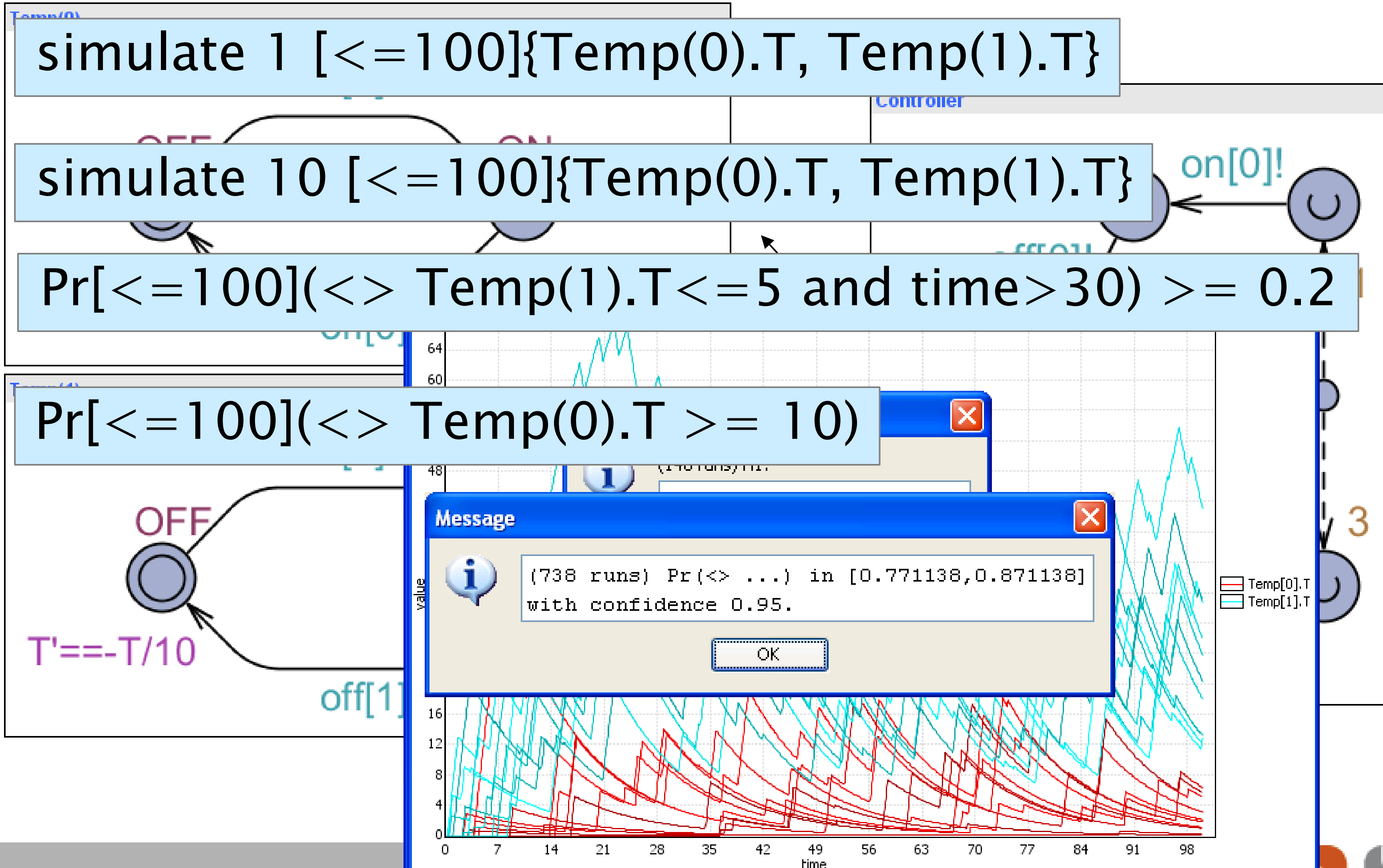
Stochastic Hybrid Systems

simulate 1 [<=100]{Temp(0).T, Temp(1).T}

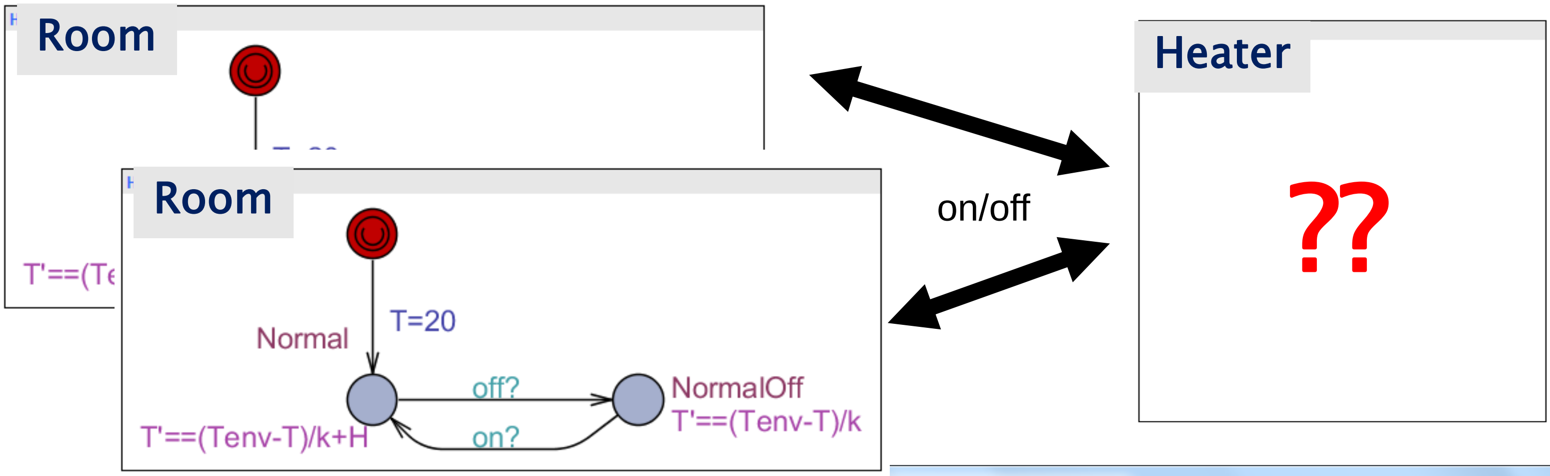
simulate 10 [<=100]{Temp(0).T, Temp(1).T}

Pr[<=100](<> Temp(1).T<=5 and time>30) >= 0.2

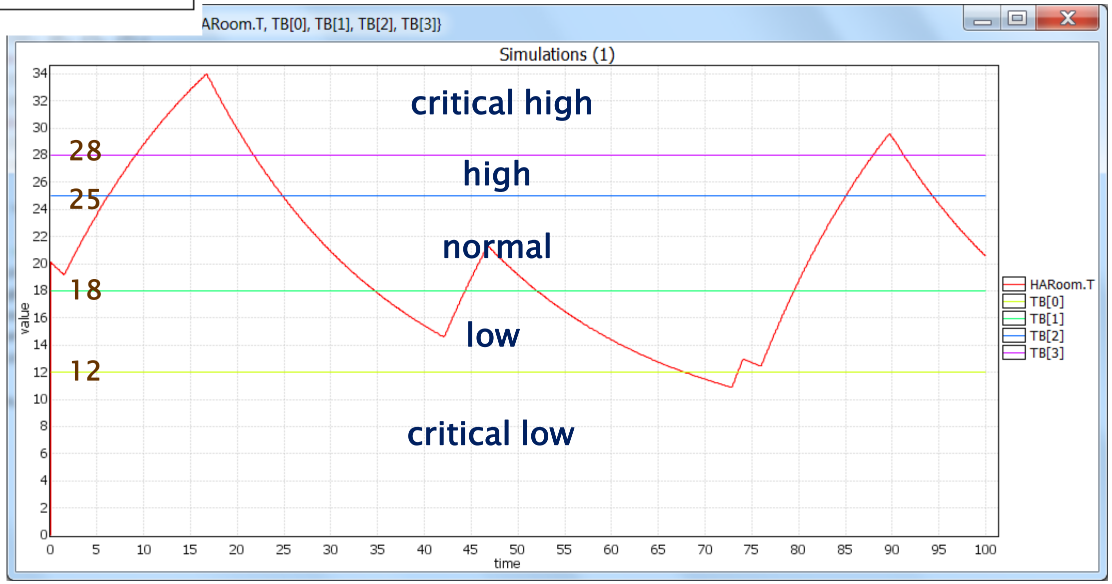
Pr[<=100](<> Temp(0).T >= 10)



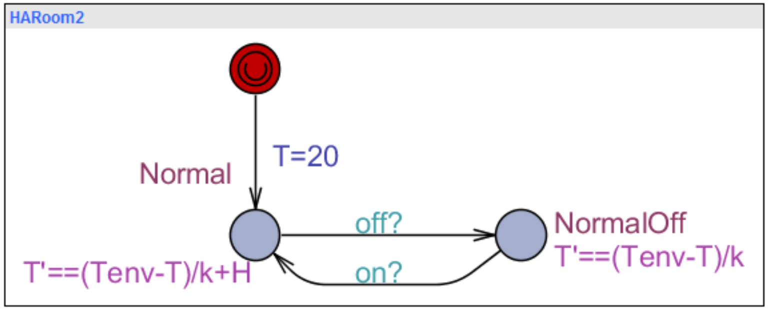
Controller Synthesis



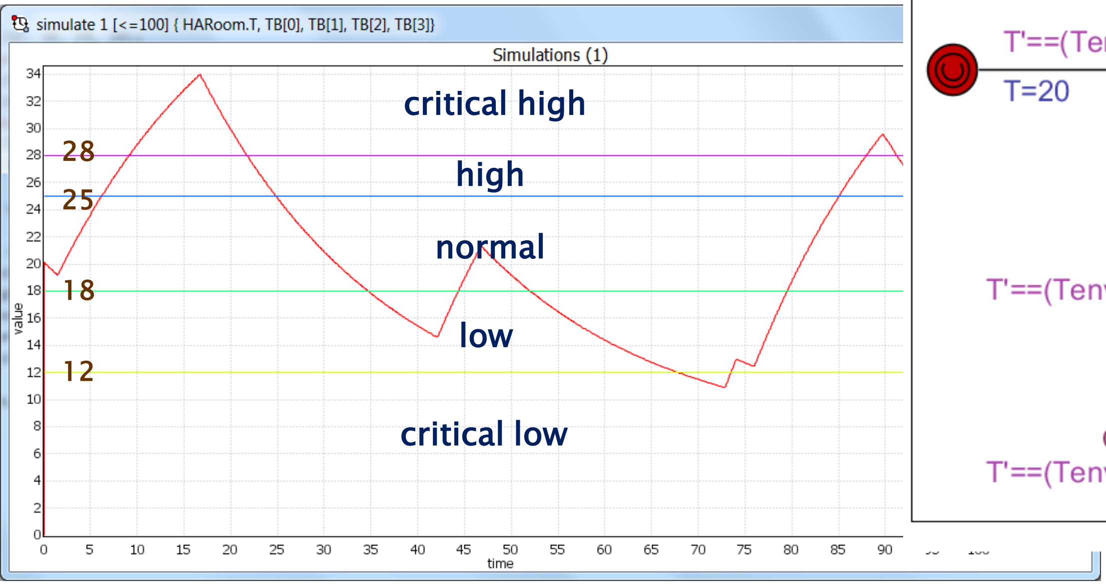

```
const int Tenv=7;
const int k=2;
const int H=20;
const int TB[4]=
        {12, 18, 25, 28};
```



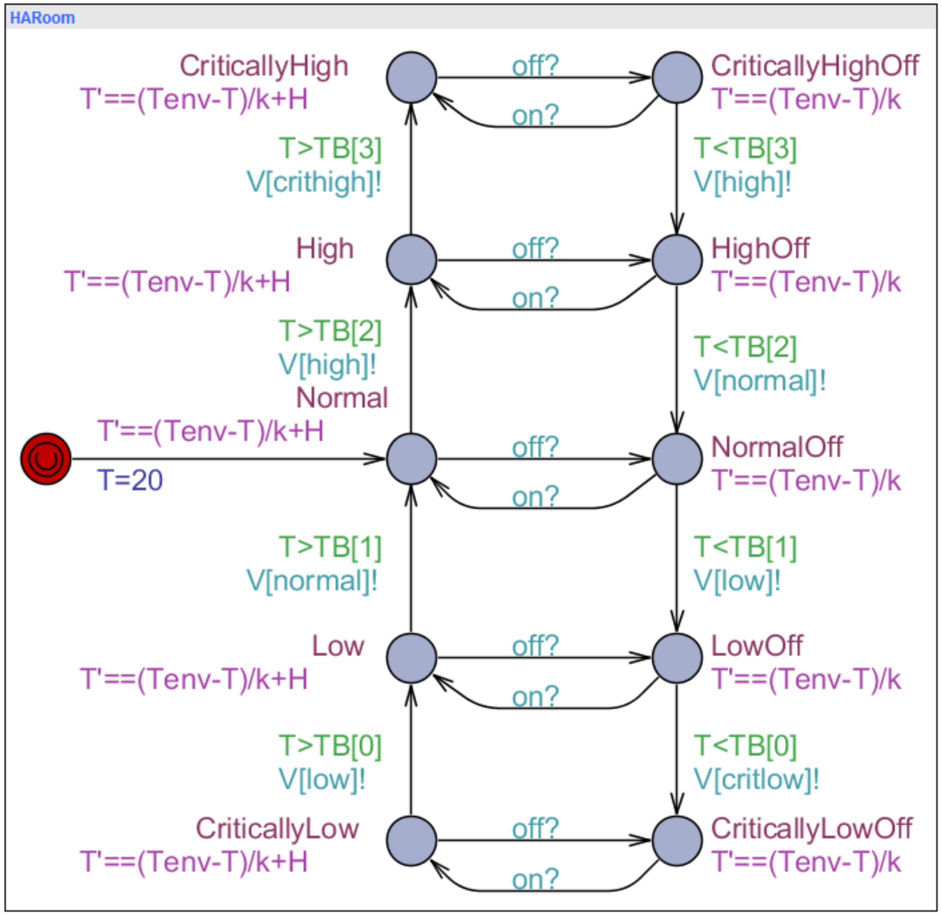
Unfolding



HARoom2
Normal
T=20
off?
on?
NormalOff
T'==(Tenv-T)/k+H
T'==(Tenv-T)/k

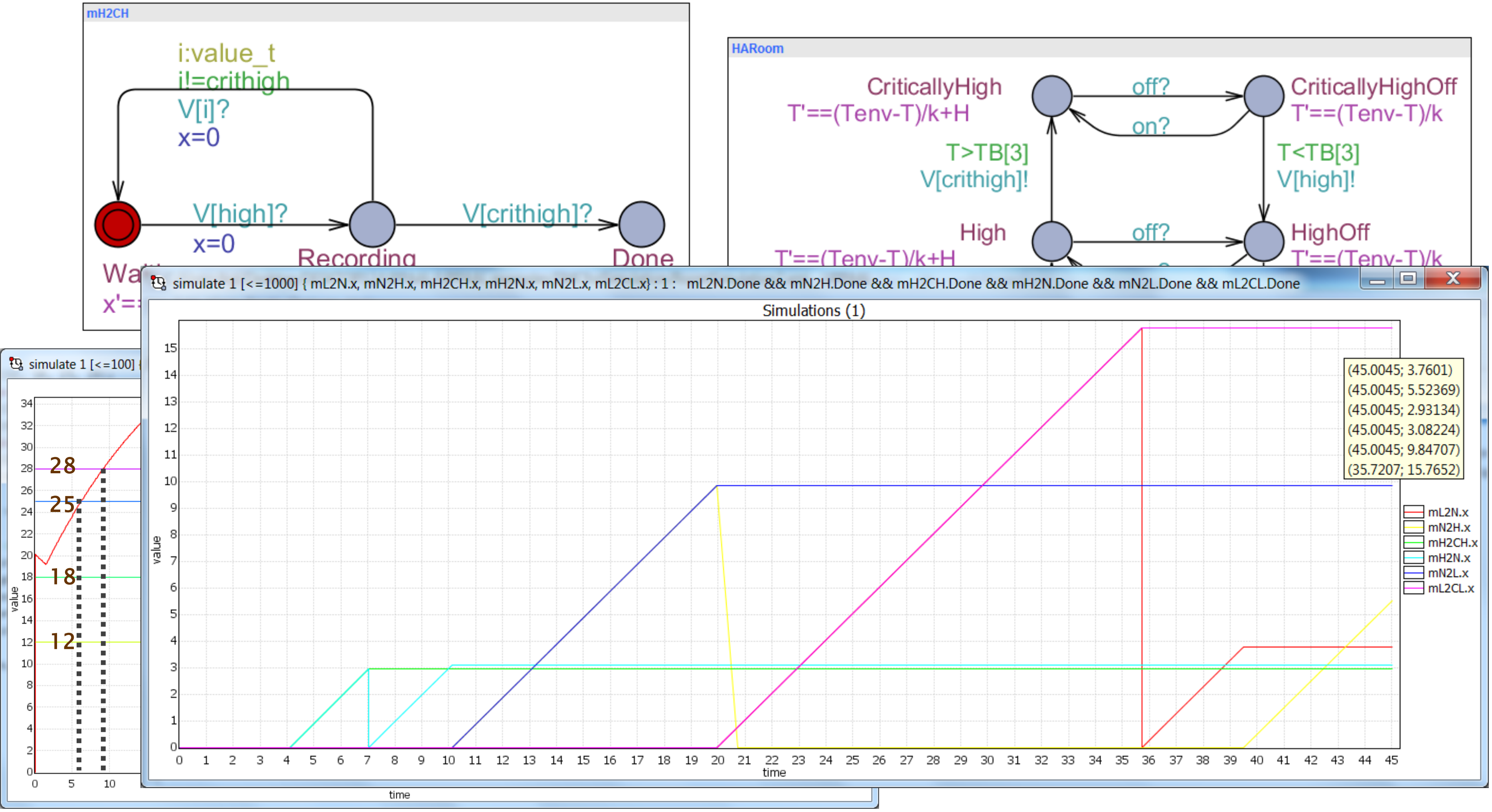


simulate 1 [<=100] { HARoom.T, TB[0], TB[1], TB[2], TB[3]}
Simulations (1)
critical high
high
normal
low
critical low
28
25
18
12
value
time



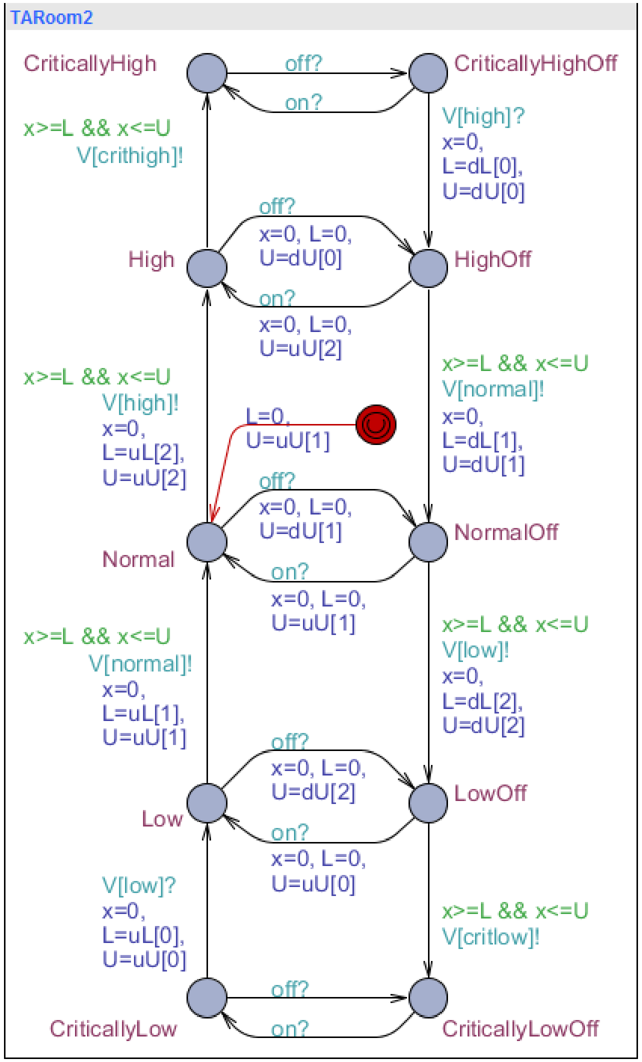
HARoom
CriticallyHigh
T'==(Tenv-T)/k+H
off?
on?
CriticallyHighOff
T'==(Tenv-T)/k
T>TB[3]
V[crithigh]!
T<TB[3]
V[high]!
High
T'==(Tenv-T)/k+H
HighOff
T'==(Tenv-T)/k
T>TB[2]
V[high]!
T<TB[2]
V[normal]!
Normal
T'==(Tenv-T)/k+H
T=20
NormalOff
T'==(Tenv-T)/k
T>TB[1]
V[normal]!
T<TB[1]
V[low]!
Low
T'==(Tenv-T)/k+H
LowOff
T'==(Tenv-T)/k
T>TB[0]
V[low]!
T<TB[0]
V[critlow]!
CriticallyLow
T'==(Tenv-T)/k+H
CriticallyLowOff
T'==(Tenv-T)/k

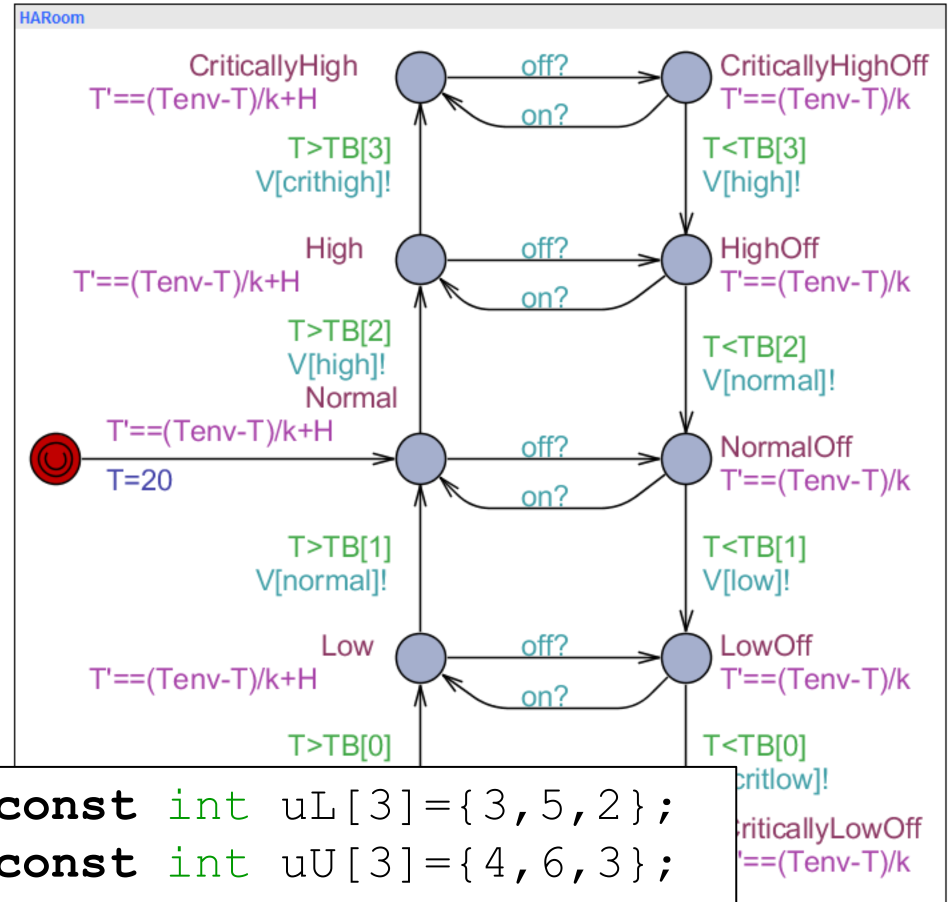
Timing



mH2CH
i:value_t
i!=crithigh
V[i]?
x=0
V[high]?
x=0
V[crithigh]?
Recording
Done
HARoom
CriticallyHigh
T'==(Tenv-T)/k+H
off?
on?
CriticallyHighOff
T'==(Tenv-T)/k
T>TB[3]
V[crithigh]!
T<TB[3]
V[high]!
High
HighOff
simulate 1 [<=1000] { mL2N.x, mN2H.x, mH2CH.x, mH2N.x, mN2L.x, mL2CL.x} : 1 : mL2N.Done && mN2H.Done && mH2CH.Done && mH2N.Done && mN2L.Done && mL2CL.Done
Simulations (1)
(45.0045; 3.7601)
(45.0045; 5.52369)
(45.0045; 2.93134)
(45.0045; 3.08224)
(45.0045; 9.84707)
(35.7207; 15.7652)
mL2N.x
mN2H.x
mH2CH.x
mH2N.x
mN2L.x
mL2CL.x
value
time
simulate 1 [<=100]
28
25
18
12

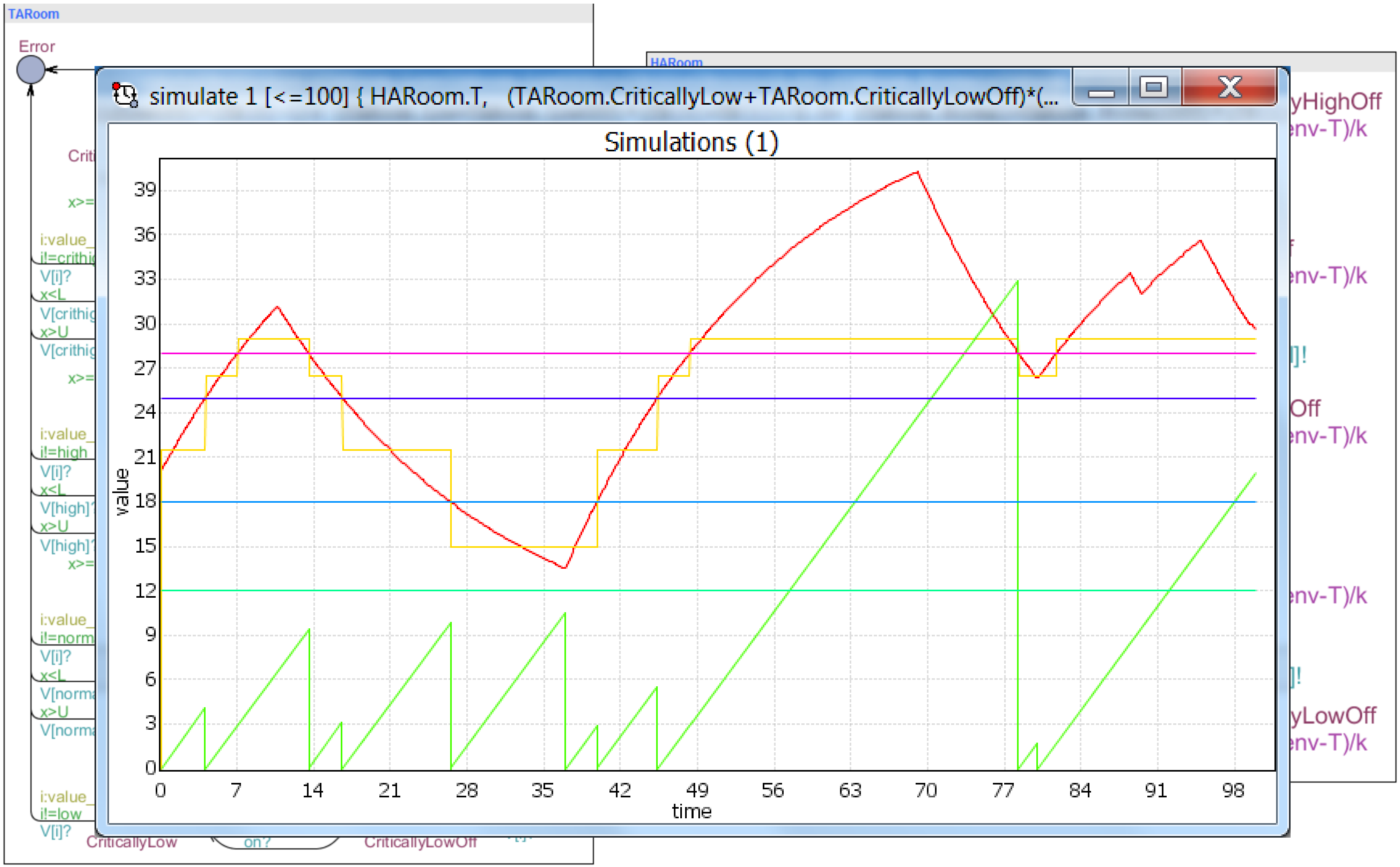
TA Abstraction



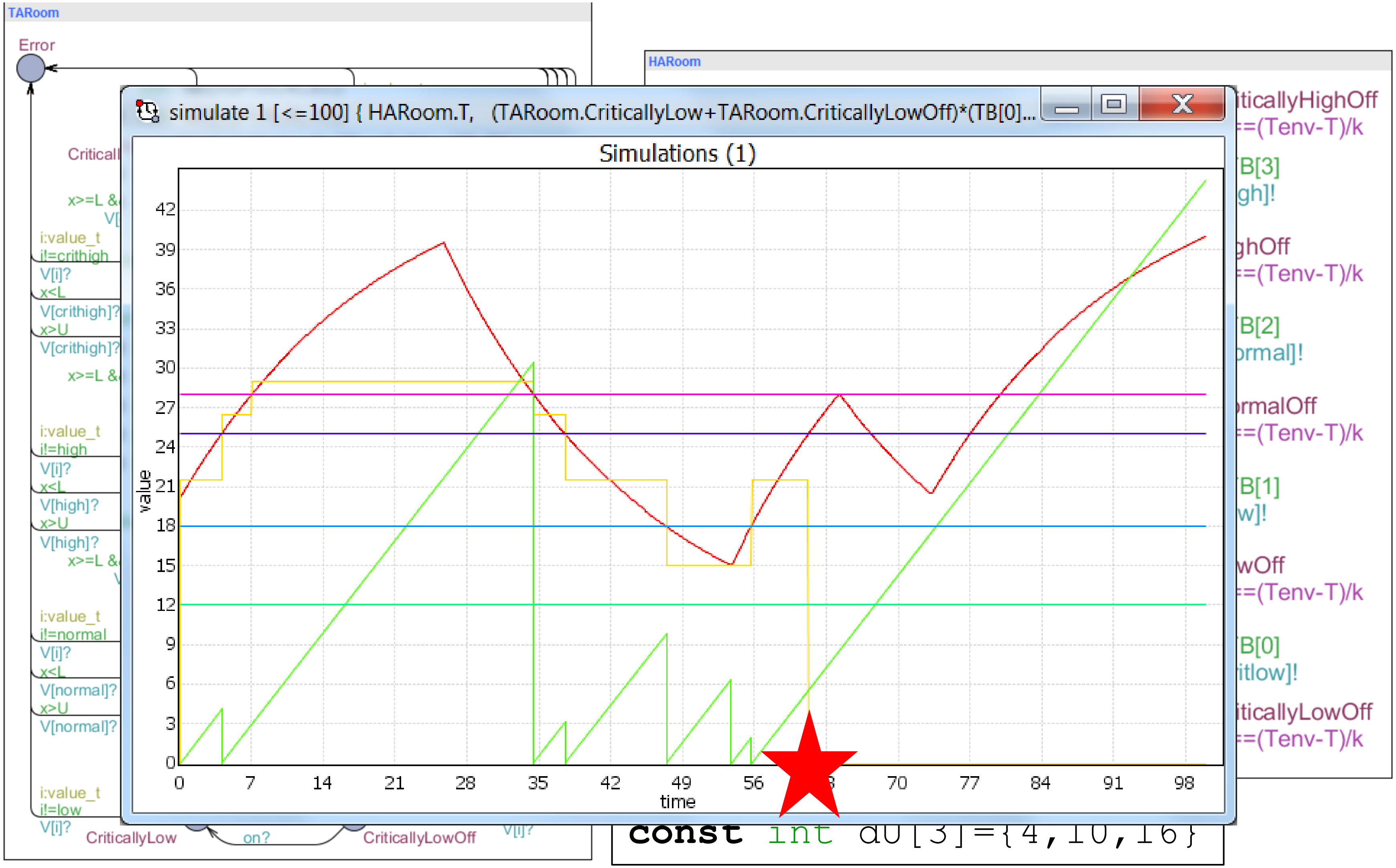



```
const int uL[3]={3,5,2};
const int uU[3]={4,6,3};
const int dL[3]={3,9,15};
const int dU[3]={4,10,16}
```

Validation by co-Simulation

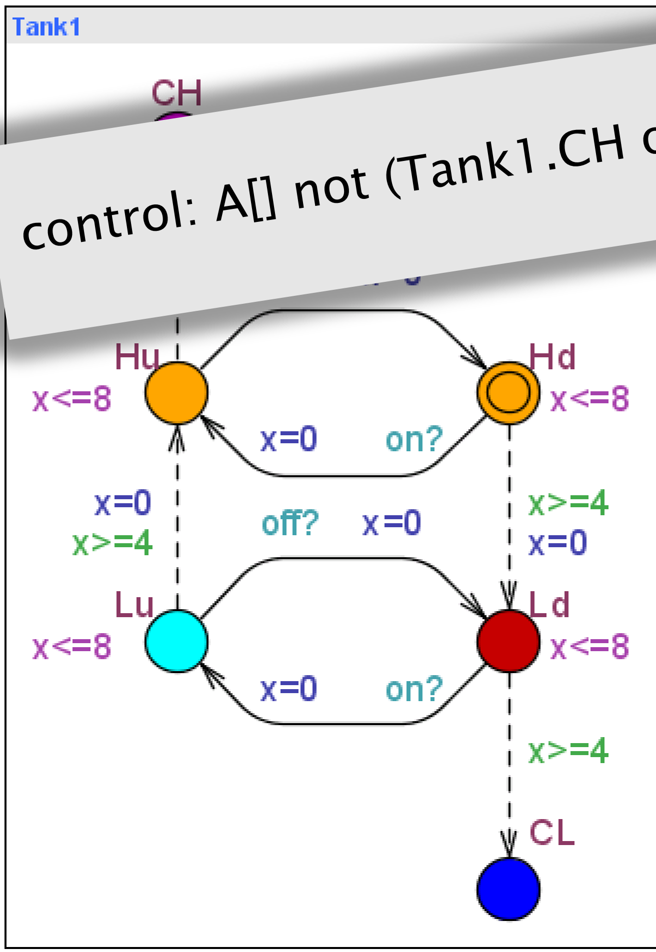


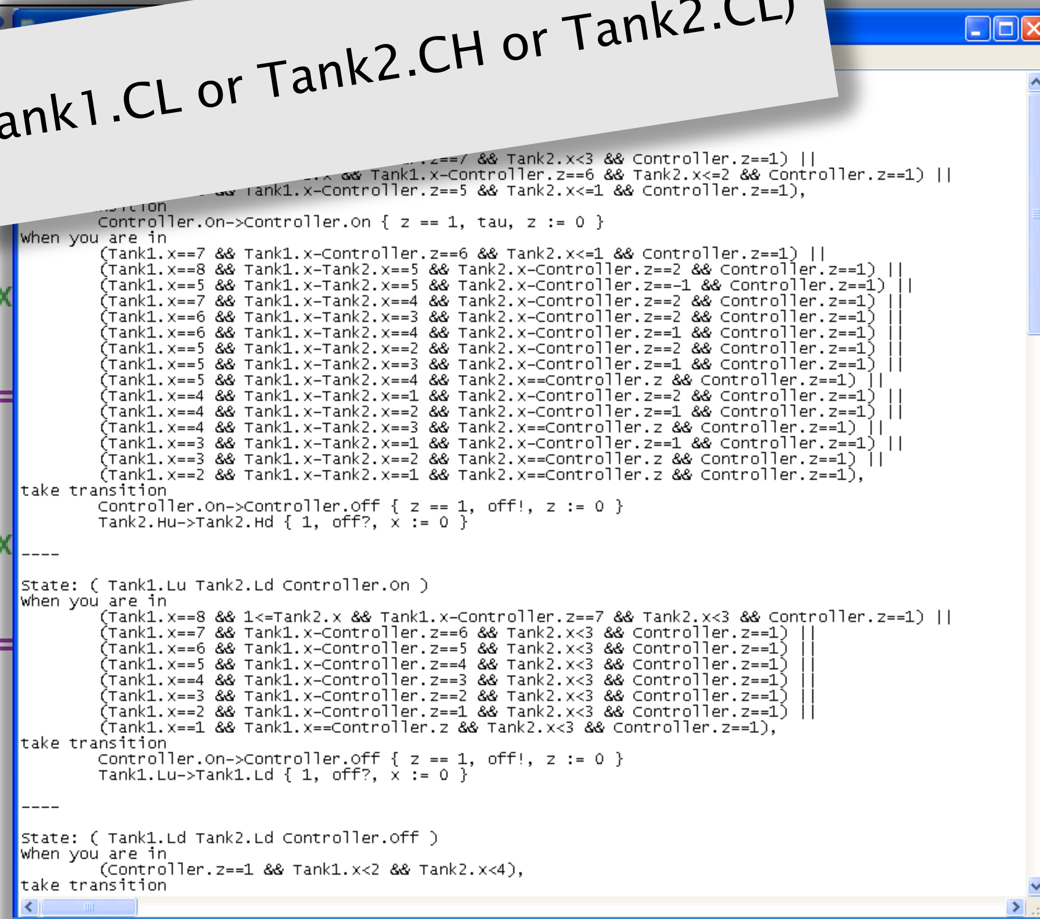
Validation by co–Simulation



Synthesis using TIGA

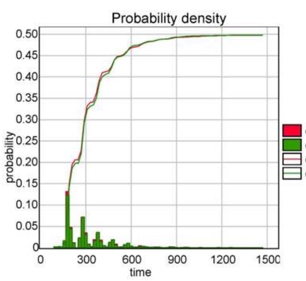
control: A[] not (Tank1.CH or Tank1.CL or Tank2.CH or Tank2.CL)



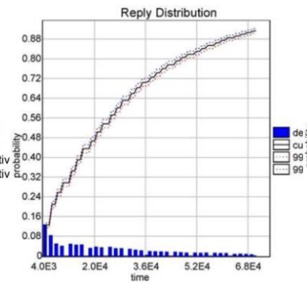


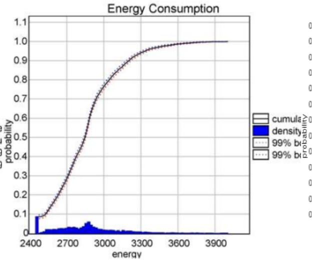
```
        Controller.On->Controller.On { z == 1, tau, z := 0 }
when you are in
        (Tank1.x==7 && Tank1.x-Controller.z==6 && Tank2.x<=1 && Controller.z==1) ||
        (Tank1.x==8 && Tank1.x-Tank2.x==5 && Tank2.x-Controller.z==2 && Controller.z==1) ||
        (Tank1.x==5 && Tank1.x-Tank2.x==5 && Tank2.x-Controller.z==-1 && Controller.z==1) ||
        (Tank1.x==7 && Tank1.x-Tank2.x==4 && Tank2.x-Controller.z==2 && Controller.z==1) ||
        (Tank1.x==6 && Tank1.x-Tank2.x==3 && Tank2.x-Controller.z==2 && Controller.z==1) ||
        (Tank1.x==6 && Tank1.x-Tank2.x==4 && Tank2.x-Controller.z==1 && Controller.z==1) ||
        (Tank1.x==5 && Tank1.x-Tank2.x==2 && Tank2.x-Controller.z==2 && Controller.z==1) ||
        (Tank1.x==5 && Tank1.x-Tank2.x==3 && Tank2.x-Controller.z==1 && Controller.z==1) ||
        (Tank1.x==5 && Tank1.x-Tank2.x==4 && Tank2.x==Controller.z && Controller.z==1) ||
        (Tank1.x==4 && Tank1.x-Tank2.x==1 && Tank2.x-Controller.z==2 && Controller.z==1) ||
        (Tank1.x==4 && Tank1.x-Tank2.x==2 && Tank2.x-Controller.z==1 && Controller.z==1) ||
        (Tank1.x==4 && Tank1.x-Tank2.x==3 && Tank2.x==Controller.z && Controller.z==1) ||
        (Tank1.x==3 && Tank1.x-Tank2.x==1 && Tank2.x-Controller.z==1 && Controller.z==1) ||
        (Tank1.x==3 && Tank1.x-Tank2.x==2 && Tank2.x==Controller.z && Controller.z==1) ||
        (Tank1.x==2 && Tank1.x-Tank2.x==1 && Tank2.x==Controller.z && Controller.z==1),
take transition
        Controller.On->Controller.Off { z == 1, off!, z := 0 }
        Tank2.Hu->Tank2.Hd { 1, off?, x := 0 }

----

State: ( Tank1.Lu Tank2.Ld Controller.On )
when you are in
        (Tank1.x==8 && 1<=Tank2.x && Tank1.x-Controller.z==7 && Tank2.x<3 && Controller.z==1) ||
        (Tank1.x==7 && Tank1.x-Controller.z==6 && Tank2.x<3 && Controller.z==1) ||
        (Tank1.x==6 && Tank1.x-Controller.z==5 && Tank2.x<3 && Controller.z==1) ||
        (Tank1.x==5 && Tank1.x-Controller.z==4 && Tank2.x<3 && Controller.z==1) ||
        (Tank1.x==4 && Tank1.x-Controller.z==3 && Tank2.x<3 && Controller.z==1) ||
        (Tank1.x==3 && Tank1.x-Controller.z==2 && Tank2.x<3 && Controller.z==1) ||
        (Tank1.x==2 && Tank1.x-Controller.z==1 && Tank2.x<3 && Controller.z==1) ||
        (Tank1.x==1 && Tank1.x==Controller.z && Tank2.x<3 && Controller.z==1),
take transition
        Controller.On->Controller.Off { z == 1, off!, z := 0 }
        Tank1.Lu->Tank1.Ld { 1, off?, x := 0 }

----

State: ( Tank1.Ld Tank2.Ld Controller.Off )
when you are in
        (Controller.z==1 && Tank1.x<2 && Tank2.x<4),
take transition
```

Other Case Studies

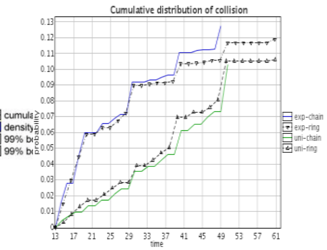


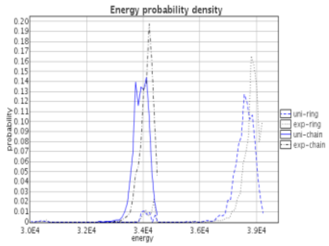
FIREWIRE



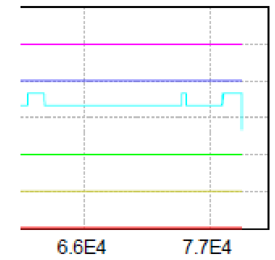


BLUETOOTH

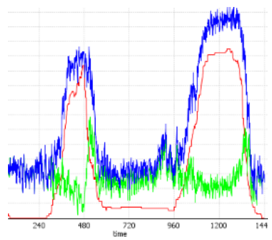




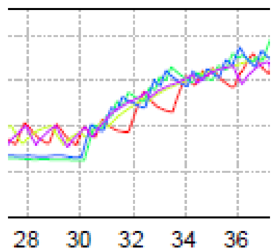
10 node LMAC



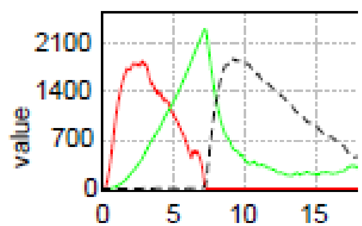
Schedulability Analysis for Mix Cr Sys



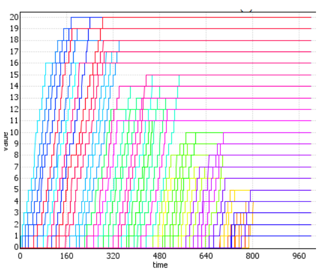
Smart Grid Demand / Response



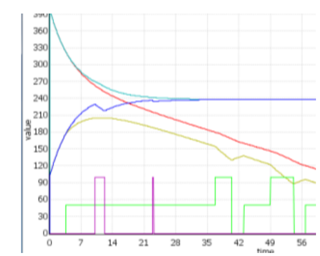
Energy Aware Buildings



Genetic Oscilator (HBS)



Passenger Seating in Aircraft



Battery Scheduling