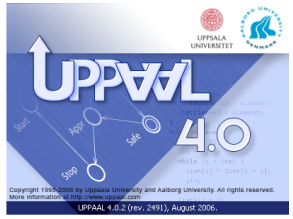
Synthesis of Safe and Optimal Strategies for Stochastic Hybrid Games

Kim G Larsen

Aalborg University, Denmark



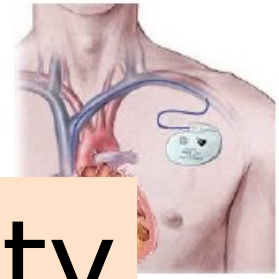
AALBORG UNIVERSITET

Cyber Physical Syst

OPTIMAL:
finding a **controller** for a given system such that a certain optimality criterion is achieved.

Discrete

Real Time



Resources

Stochasticity

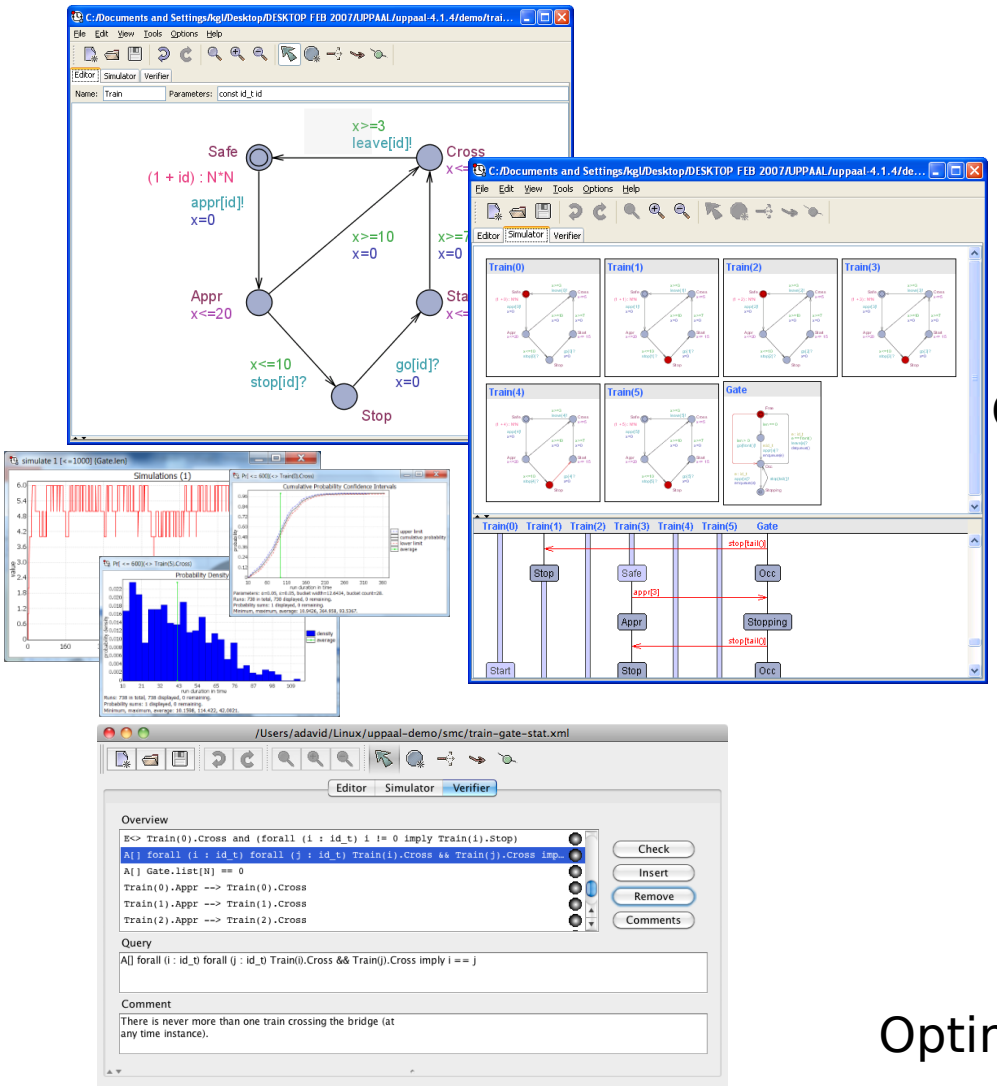
DEPENDABLE:
the ability of a **controller** to function (correct) under stated conditions for a specified period of time.

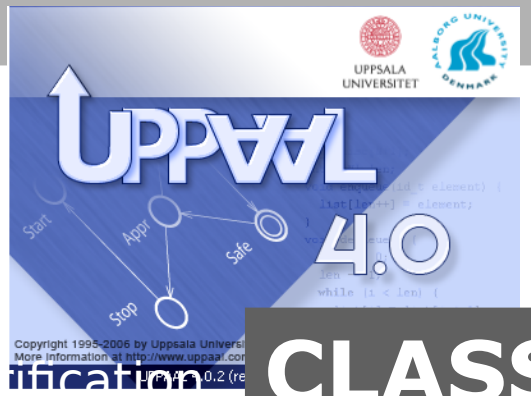
IoT

Cyber-Physical Systems

Hybrid

UPPAAL Tool Suite





Verification	CLASSIC	1995
Optimization	CORA	2001
Testing	TRON	2004
Synthesis	TIGA	2005
Component	ECDAR	2010
Performance Analysis	SMC	2011
Optimal Synthesis	STRATEGO	2014

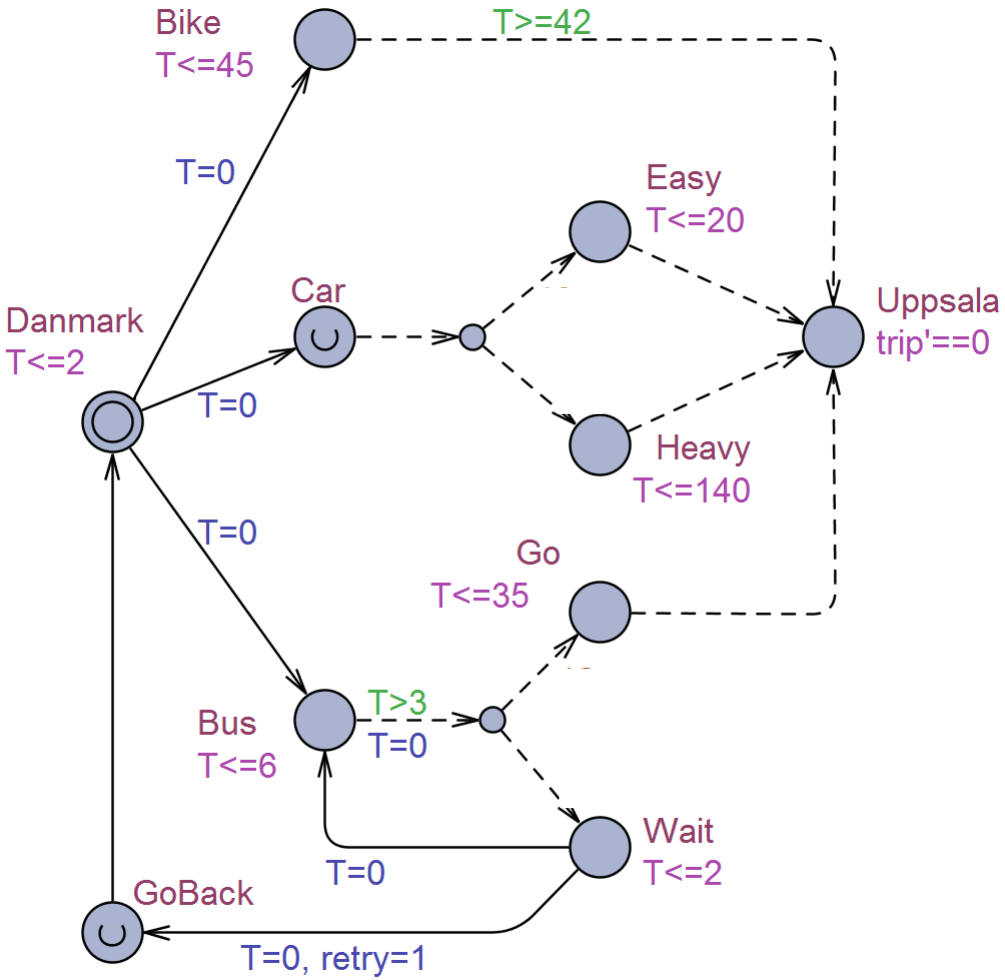
Outline

1. Stochastic (Priced) Timed Games
   - Going from Denmark to Uppsala
   - Intelligent Traffic Control

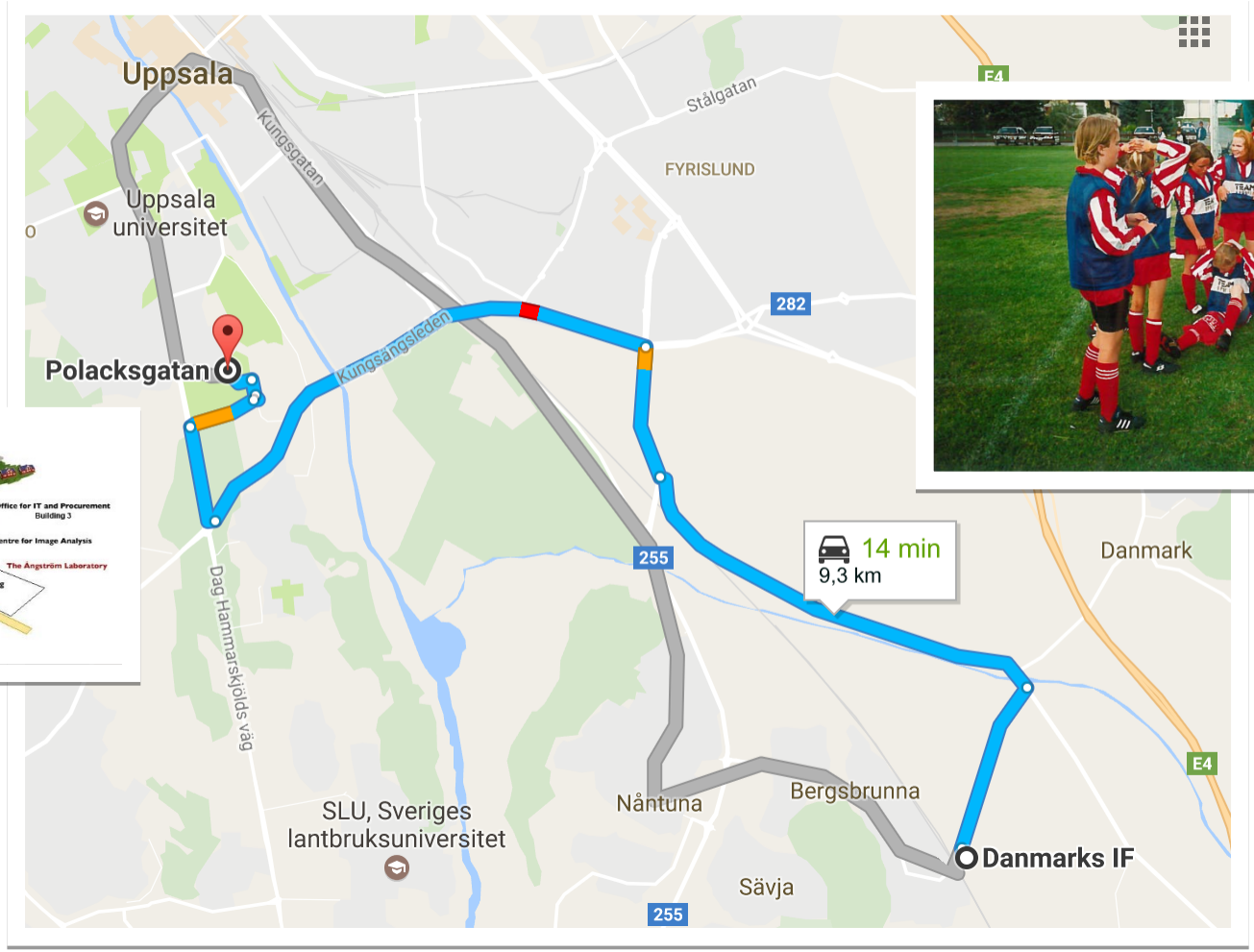
2. Stochastic Hybrid Games
   - Bouncing and Hitting Balls
   - Floor Heating

3. Safety for hybrid games using Euler.
   - Safe Adaptive Cruice Control

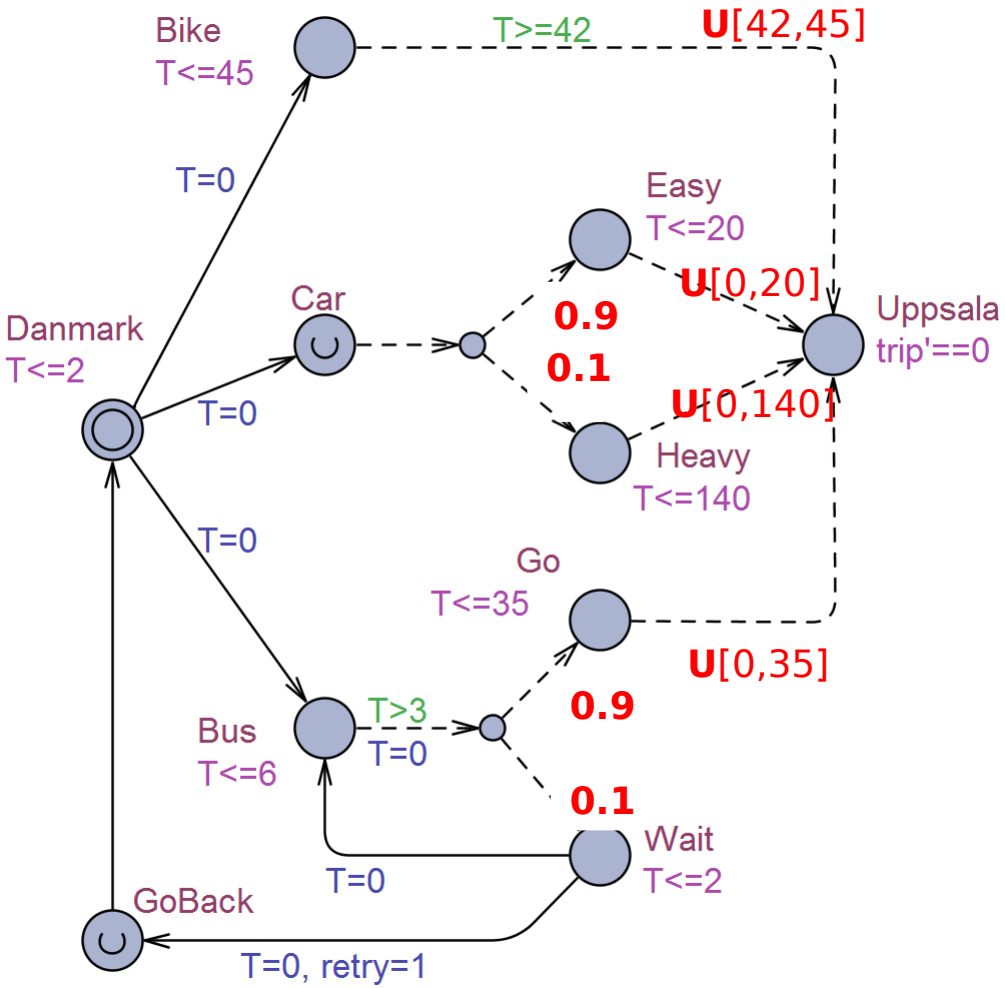
Going to Uppsala – in 1 hour



Going to Uppsala – in 1 hour



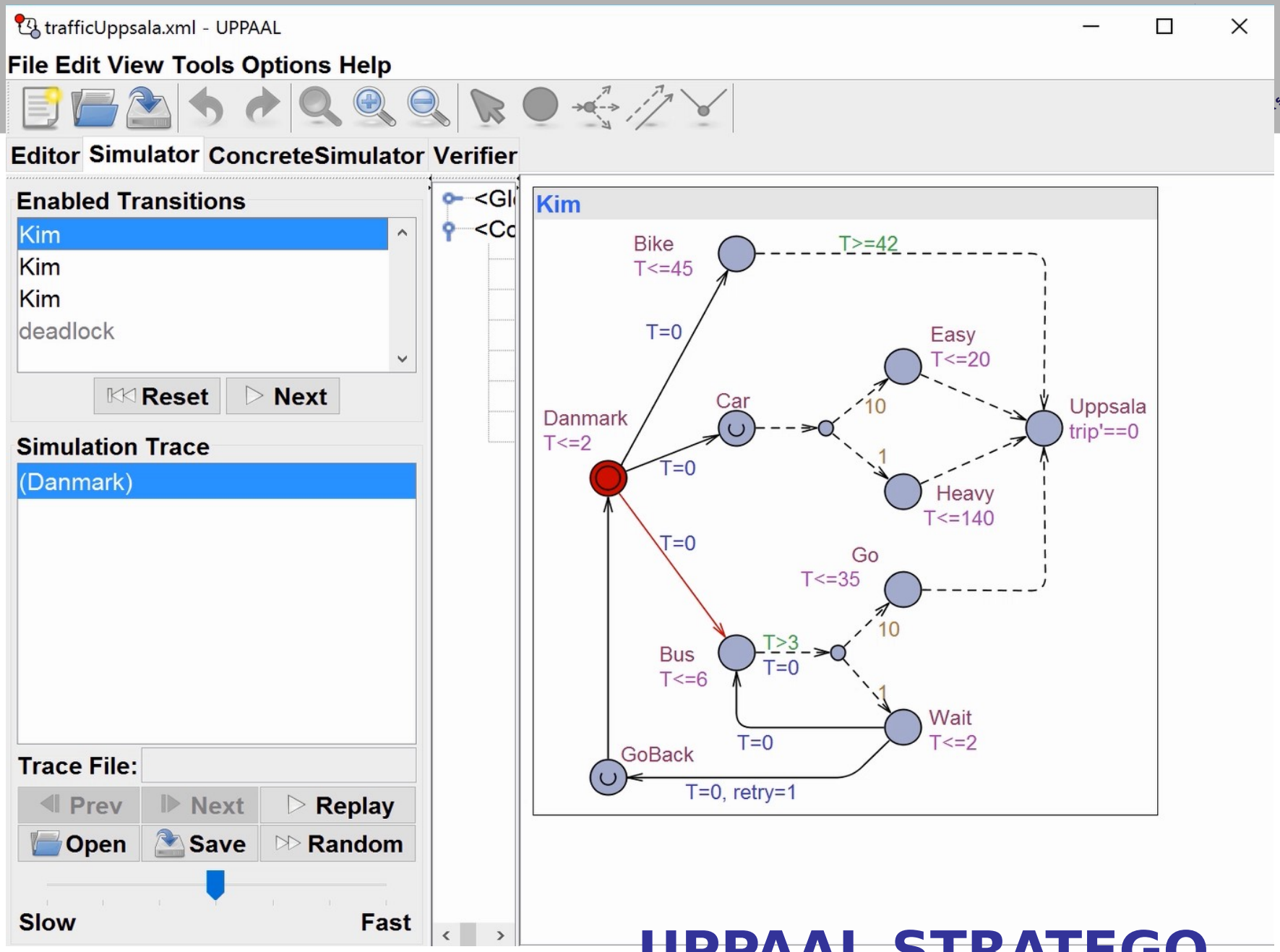
Going to Uppsala – in 1 hour



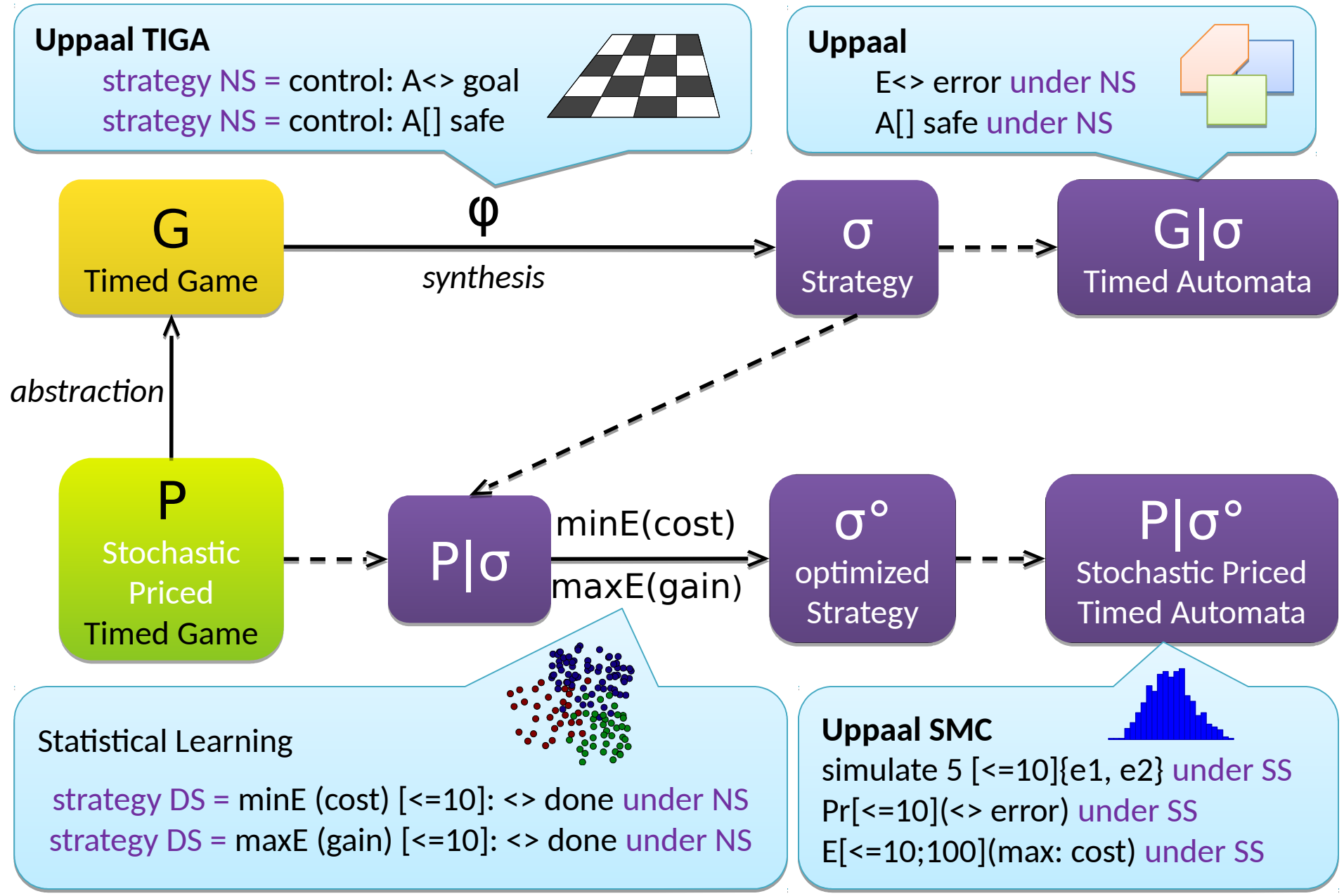
Optimal WC Strategy
(2-player)
Take bike
WC=45

Optimal Expected Strategy
(1½ player)
Take car
E = 16
WC = 140

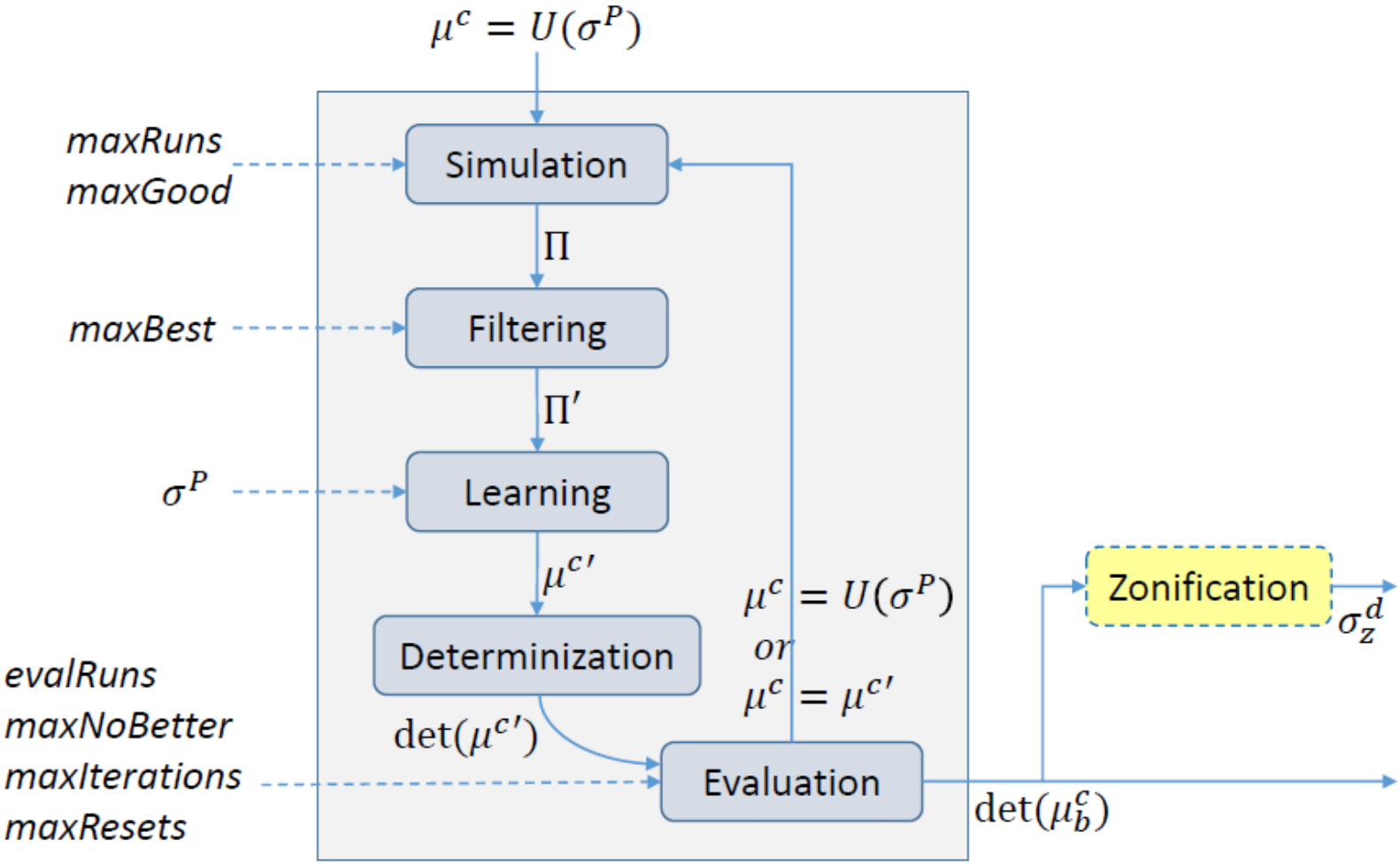
Optimal Expected Strategy guaranteeing WC<=60
?????



UPPAAL STRATEGO

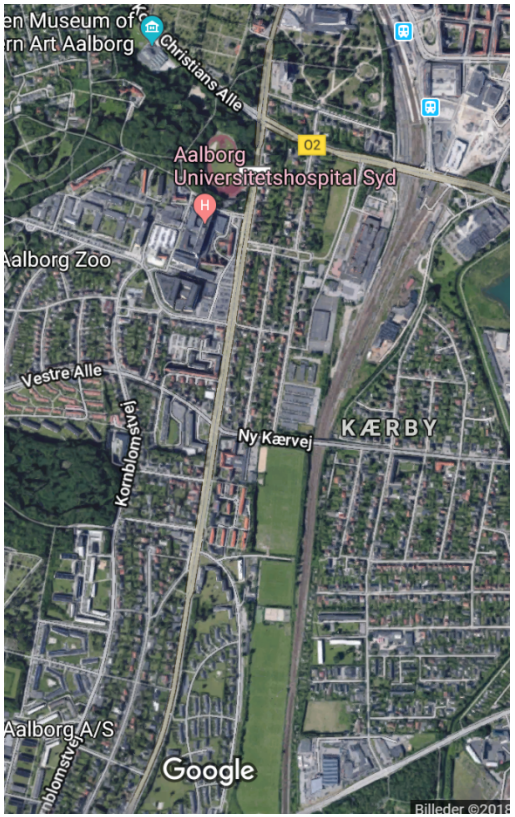

Uppaal TIGA
strategy NS = control: A<> goal
strategy NS = control: A[] safe
Uppaal
E<> error under NS
A[] safe under NS
G
Timed Game
φ
synthesis
σ
Strategy
G|σ
Timed Automata
abstraction
P
Stochastic
Priced
Timed Game
P|σ
minE(cost)
maxE(gain)
σ°
optimized
Strategy
P|σ°
Stochastic Priced
Timed Automata
Statistical Learning
strategy DS = minE (cost) [<=10]: <> done under NS
strategy DS = maxE (gain) [<=10]: <> done under NS
Uppaal SMC
simulate 5 [<=10]{e1, e2} under SS
Pr[<=10](<> error) under SS
E[<=10;100](max: cost) under SS

Learning Method



Intelligent
Traffic Control

Hobrovej, Aalborg
2 km stretch
6 signalized intersections
ÅDT 20.000-30.000
VISSEM (7.00-9.00)



AALBORG UNIVERSITET

DCyPS

w/ Marco Muniz
Jakob Taankvist
Harry Lahrman
Andeas Eriksen

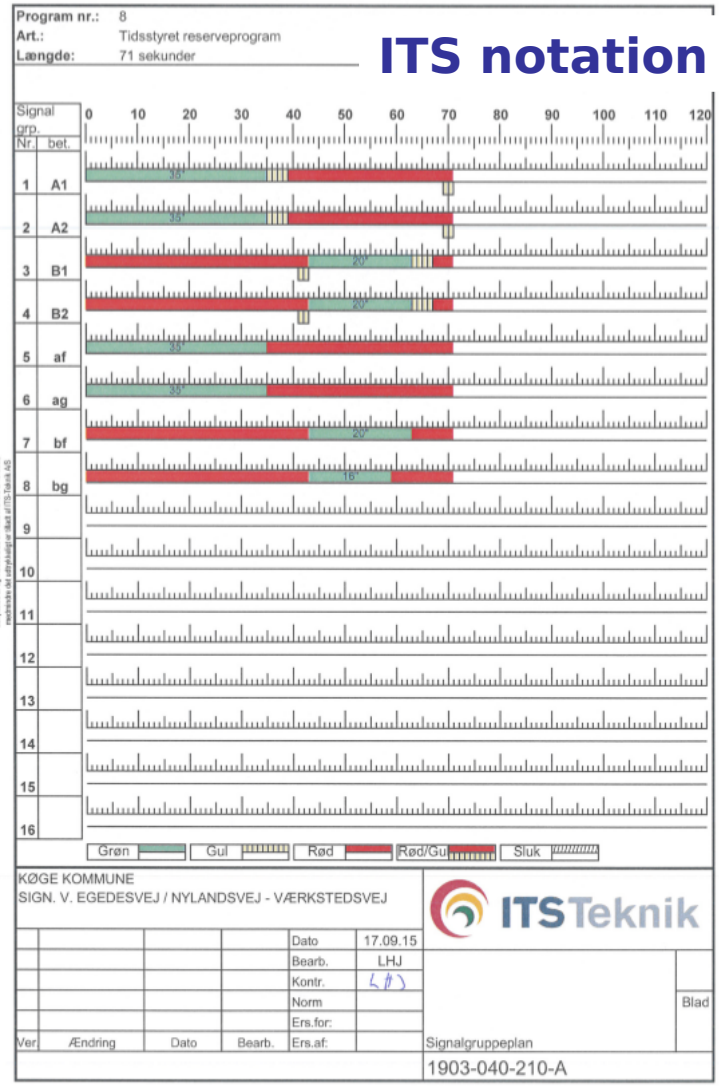
Eriksen, Huang, Kildebogaard, Lahrmann, Larsen, Muniz, Taankvist:
Uppaal Stratego for Intelligent Traffic Lights, ITS Europe 2017

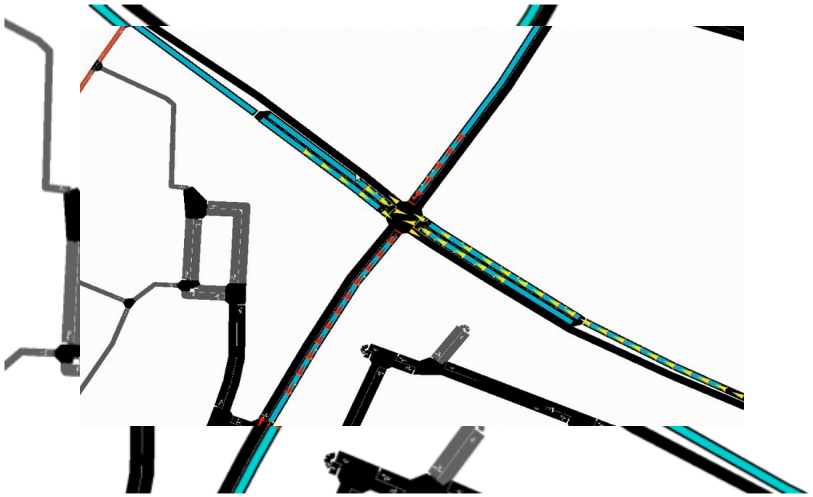
Objective

- Observed a lot of **unnecessary** waiting time for red light in signalized intersections

- Aim:
  Design an intelligent controller that can realize a near optimal signal control using UPPAAL Stratego

Time Triggered Control (Static)

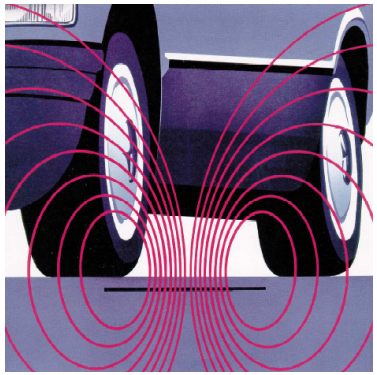




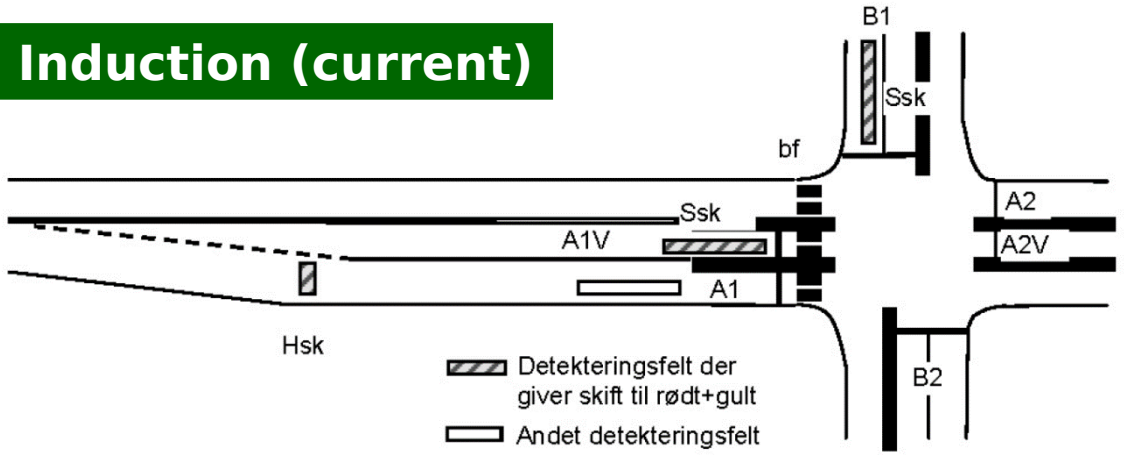
```
<tlLogic id="1693132977" type="static" programID="0">
<phase duration="35" state="rrGGgrrGGg"/>
<phase duration="4" state="rryyyrryyy"/>
<phase duration="4" state="rrrrrrrrrr"/>
<phase duration="20" state="GGrrrGGrrr"/>
<phase duration="4" state="yyrrryyrrr"/>
<phase duration="4" state="rrrrrrrrrr"/>
</tlLogic>
```

SUMO encoding

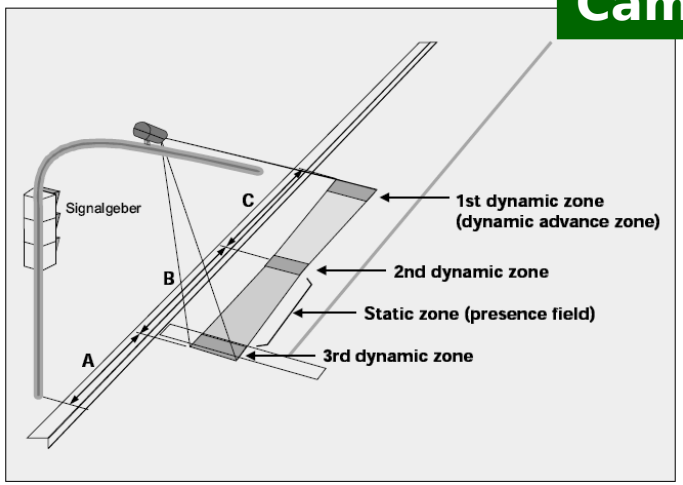
Detection



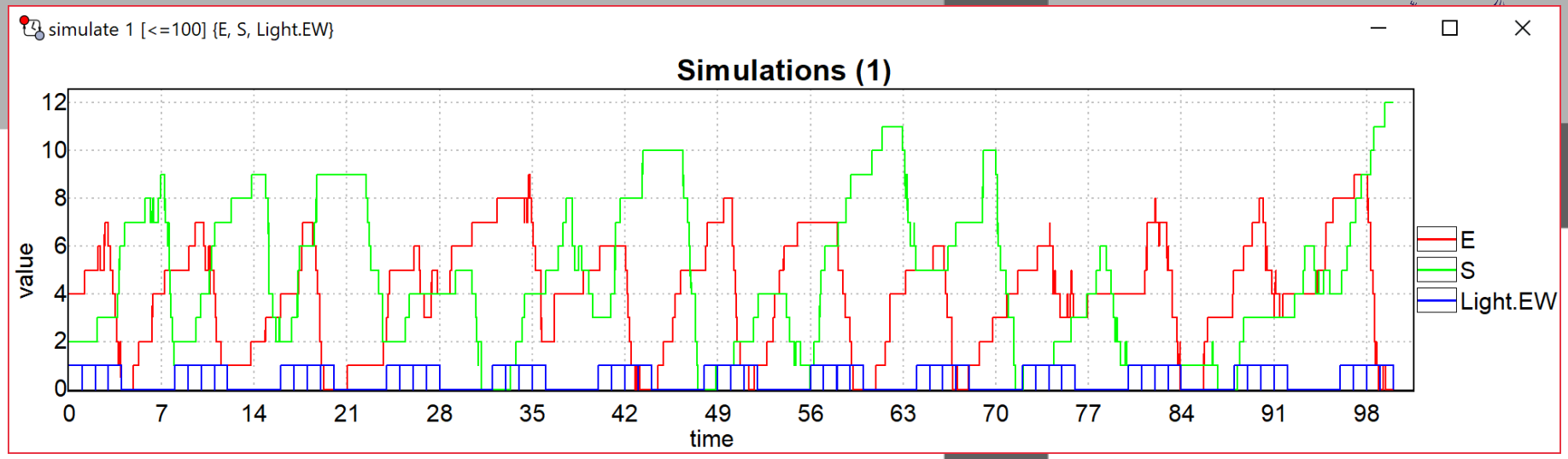
Loop Induction (current)

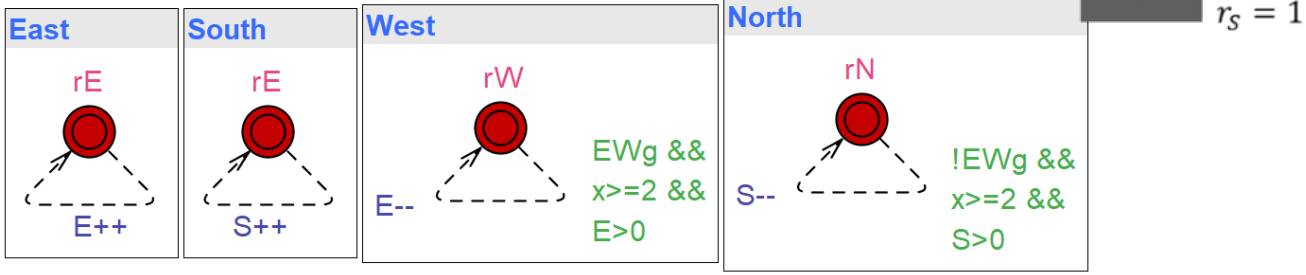


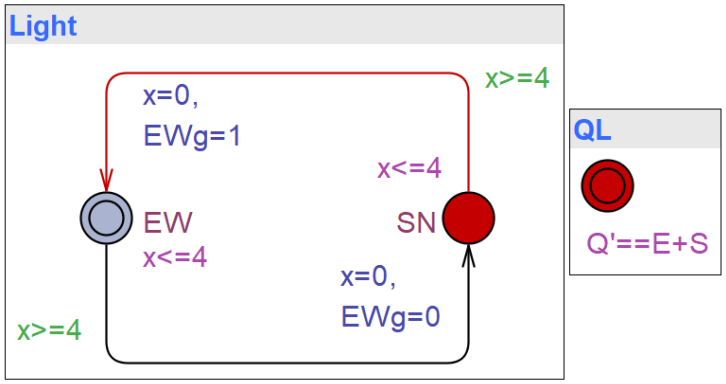
Camera (future)



Detection fields

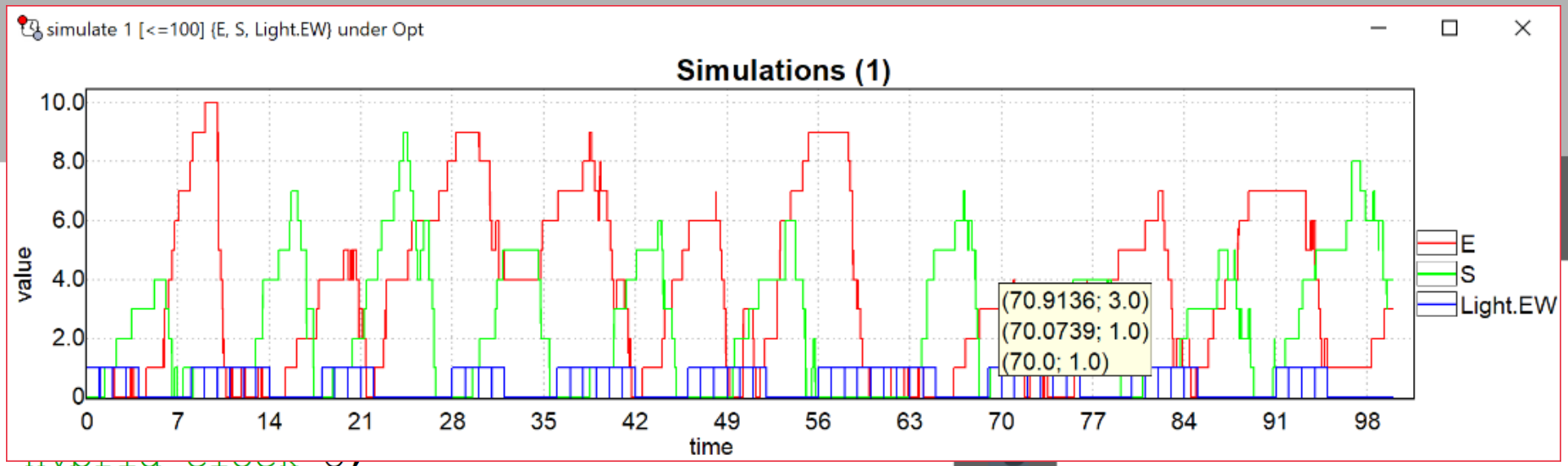


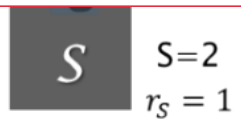




E[<=100;1000] (max:Q): 1296.29 +- 30.577

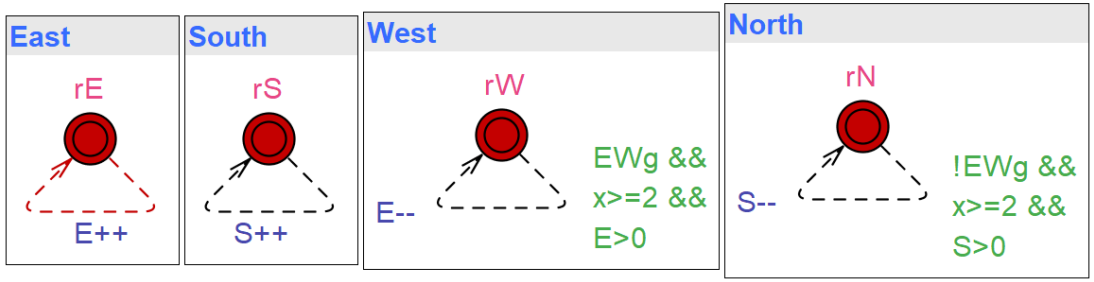
Time Driven Controller



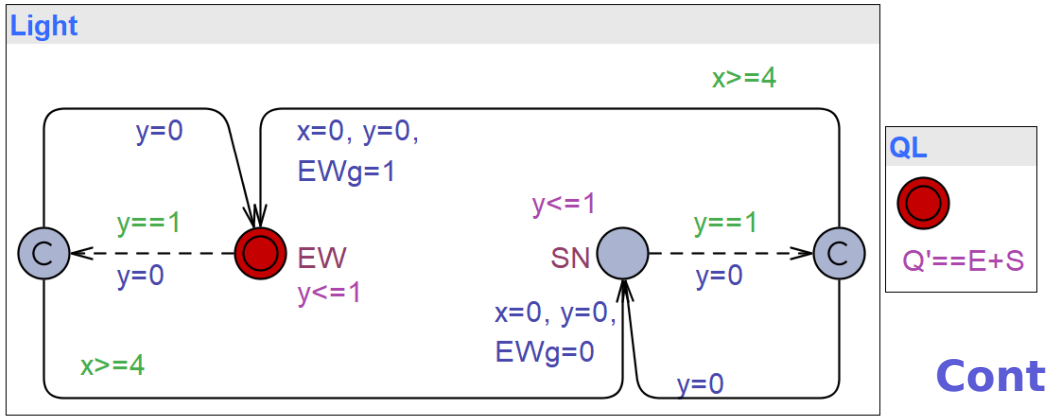


$S=2$

$r_s = 1$



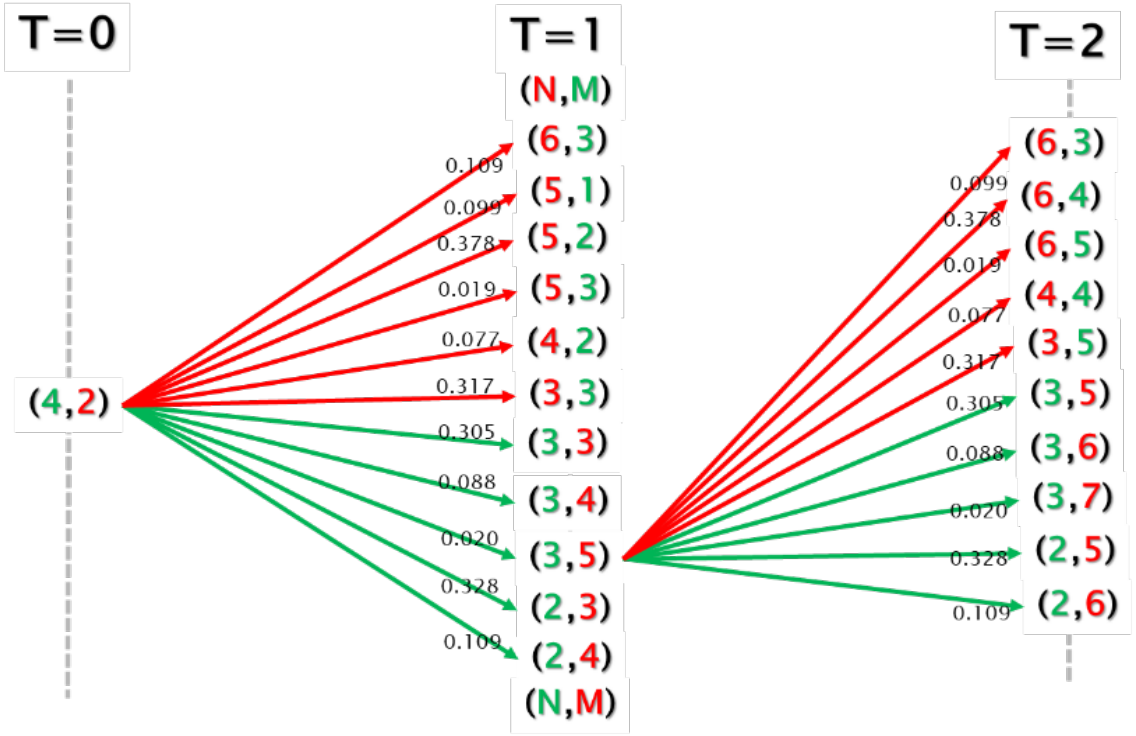
E[<=100;1000] (max:Q)
under Opt
819.36 +- 11.4



Controller Synthesis

strategy Opt = minE (Q) [<=100] {Light.location} -> {E, S, x} : <> t>=99

MDP and Strategies



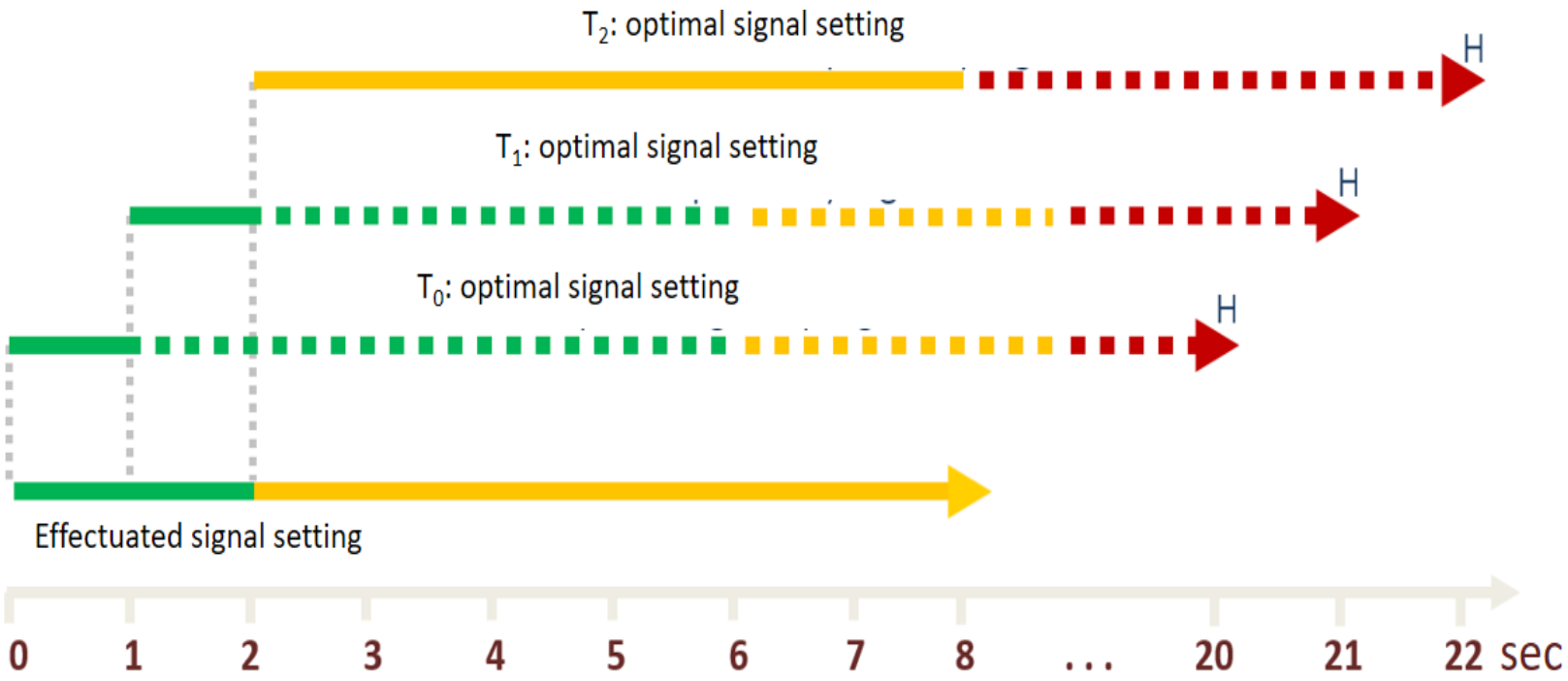
Stochastic Strategy:

$$\mu^c : \mathbb{N} \times \mathbb{N} \times \mathbb{R} \to (\{r, g\} \to [0, 1])$$

Deterministic Strategy

$$\mu_d : \mathbb{N} \times \mathbb{N} \times \mathbb{R} \to \{r, g\}$$

Online Learning (MPC)



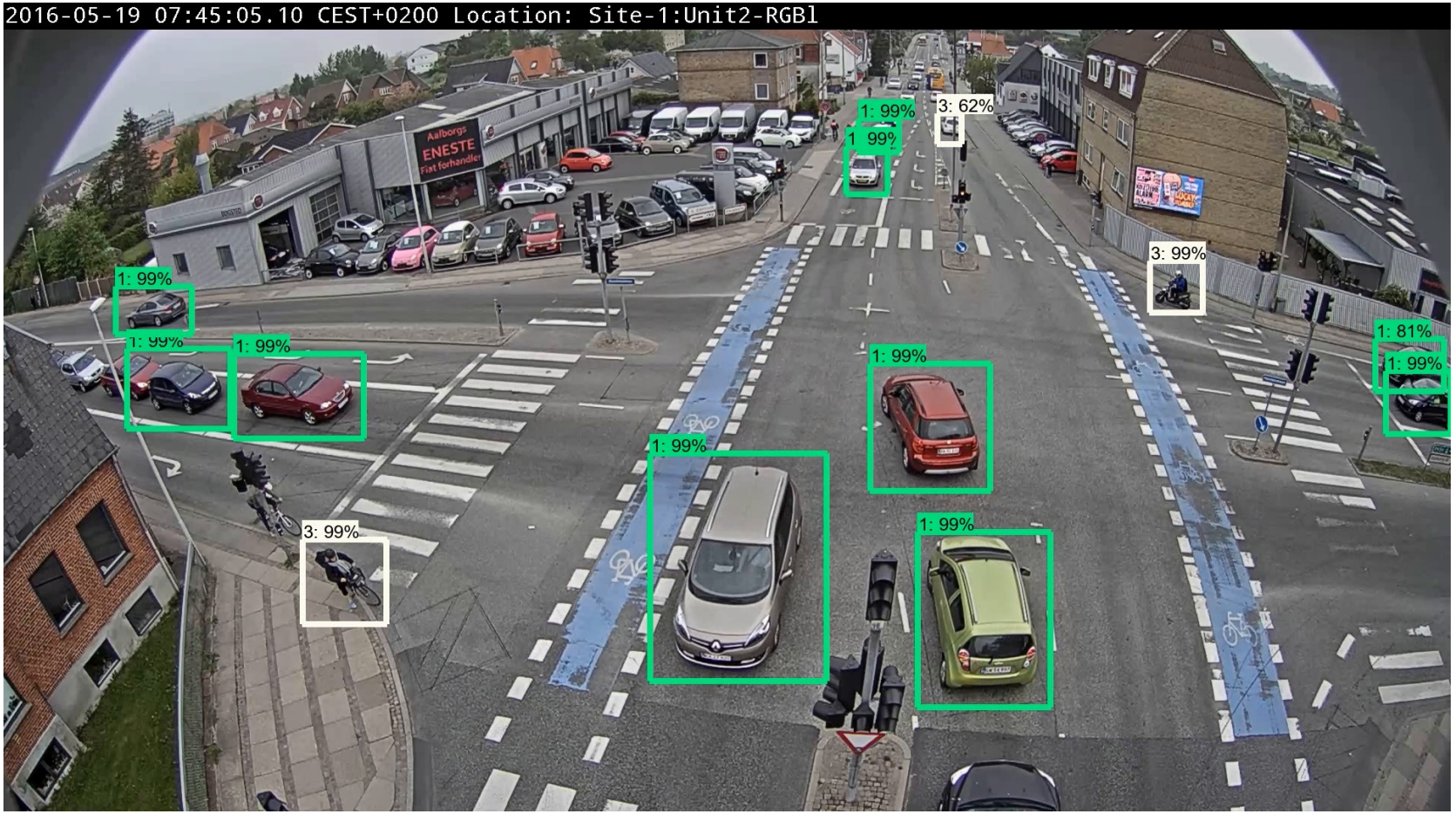
We learn a strategy up to a horizon, we then after a second learn a new strategy using the updated information from the radar.

VISSIM Simulations

Traditional Signal Control

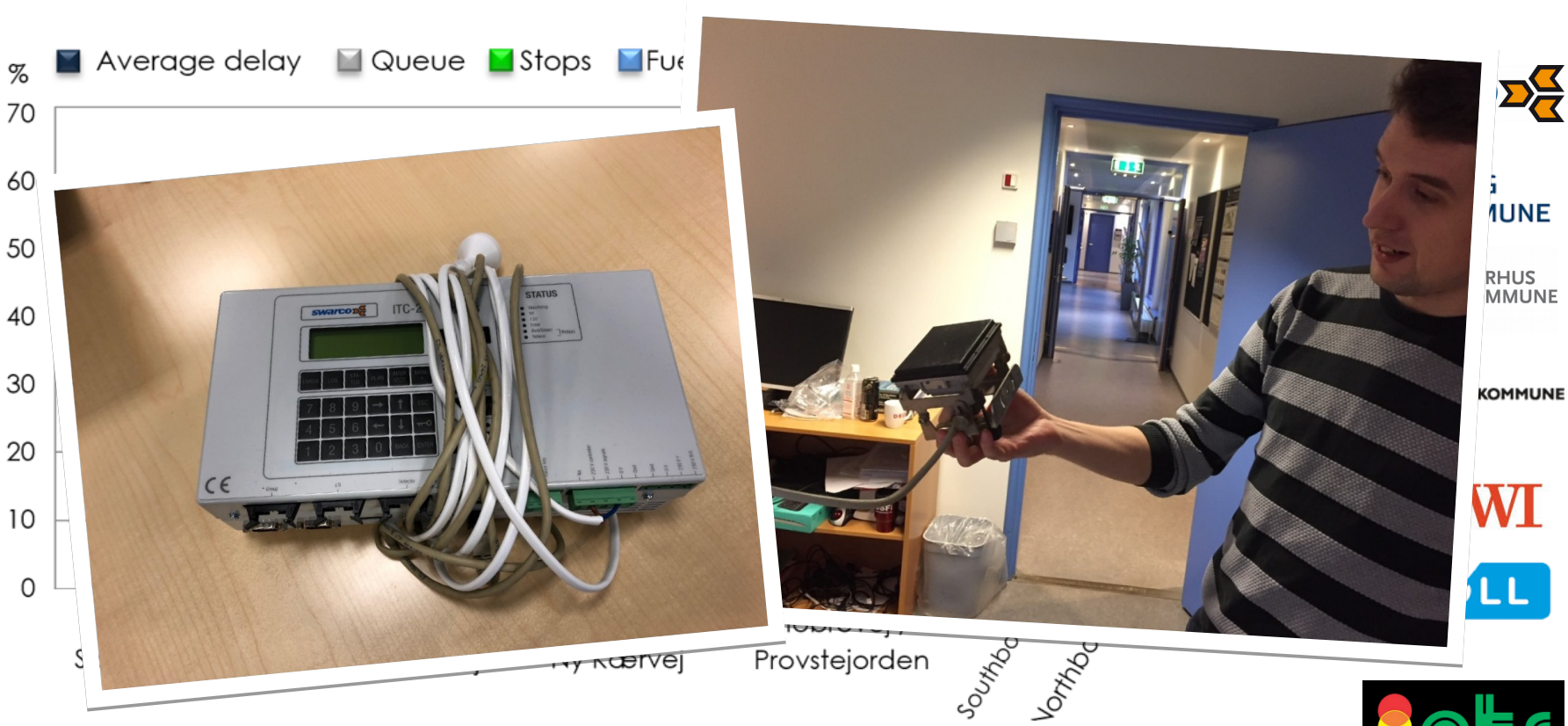
Intelligent Signal Control

Object Identification & Tracking



ATS: Googles tensorflow / convolutional neural networks

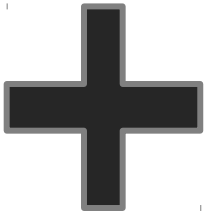
Results



Current status

Aarhus Kommune, Aalborg Kommune, Pune,
+Bus (request from municipality)
Publishing at Traffic Research Outlets (ITS).
UPPAAL Stratego on Microcontroller communicating with Radar

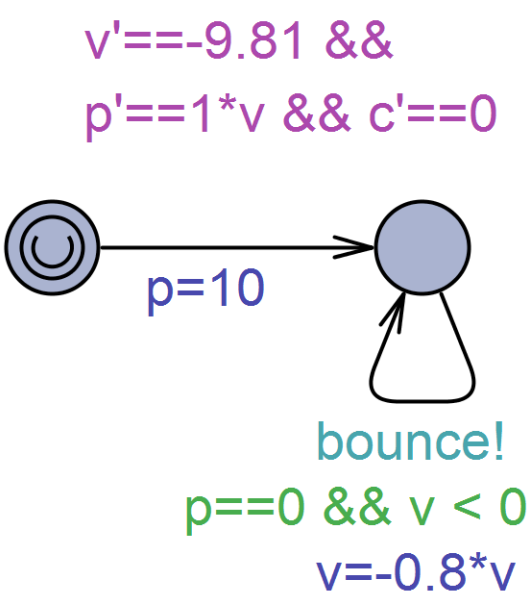
Stochastic Hybrid Games

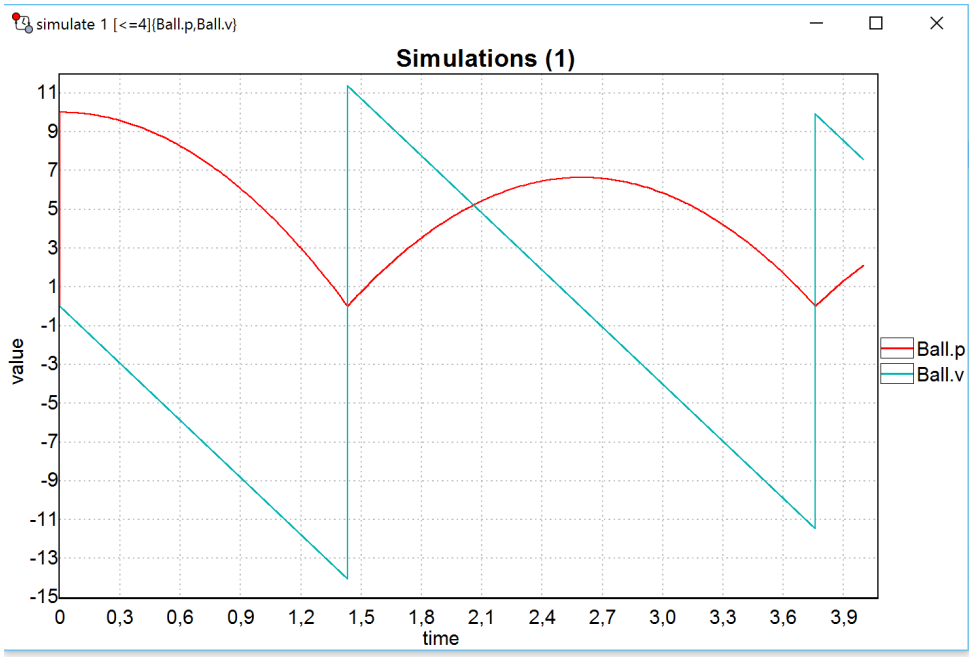


AALBORG UNIVERSITET

Hybrid Automata

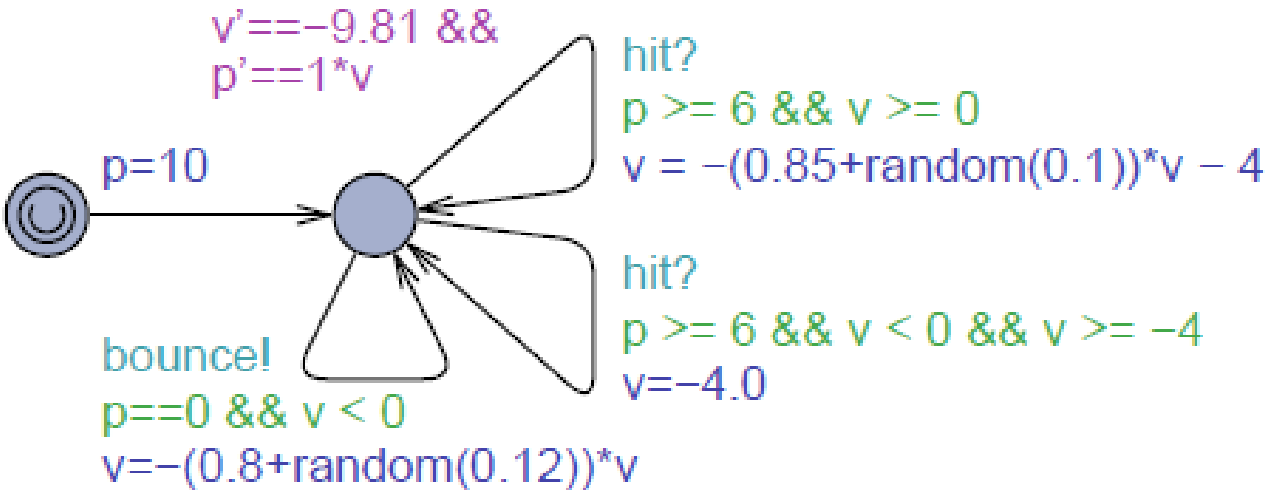
Bouncing Ball





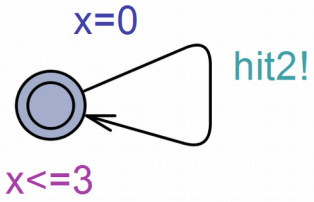
Stochastic Hybrid Systems

Bouncing Ball

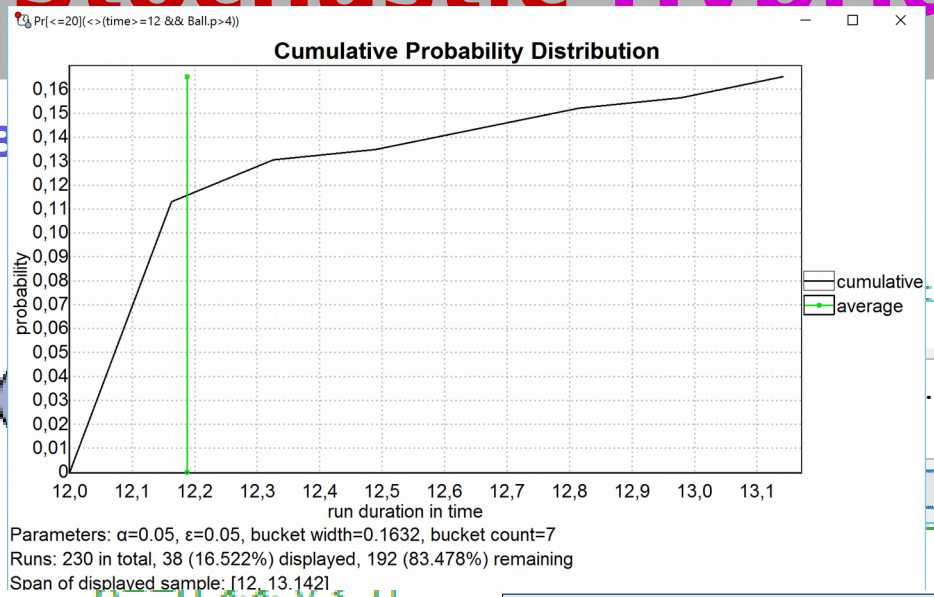


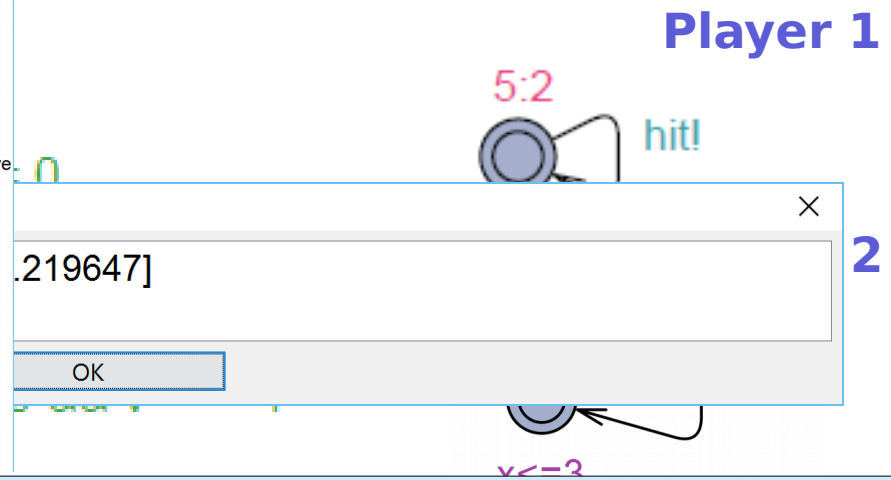
Player 1

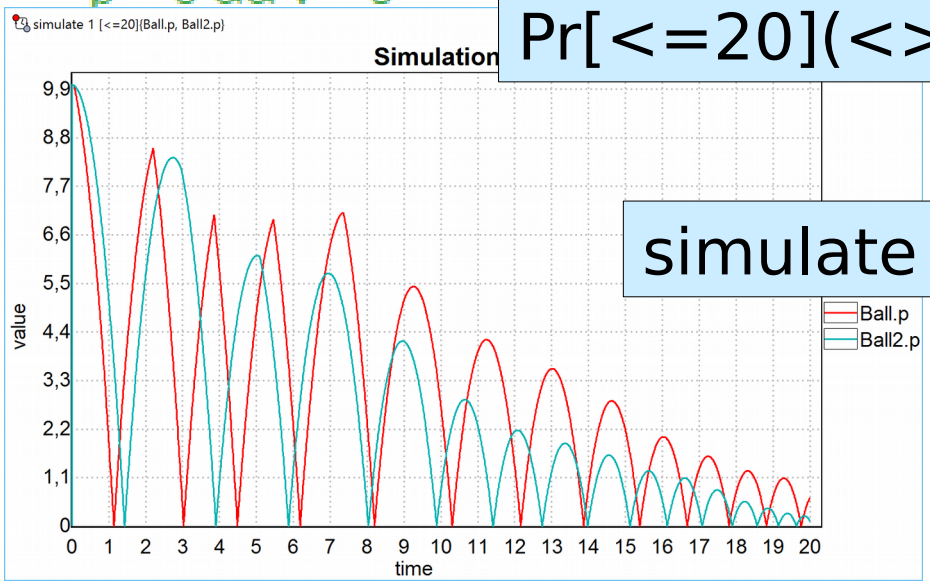
Player 2



Stochastic Hybrid Systems

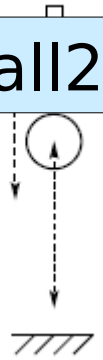




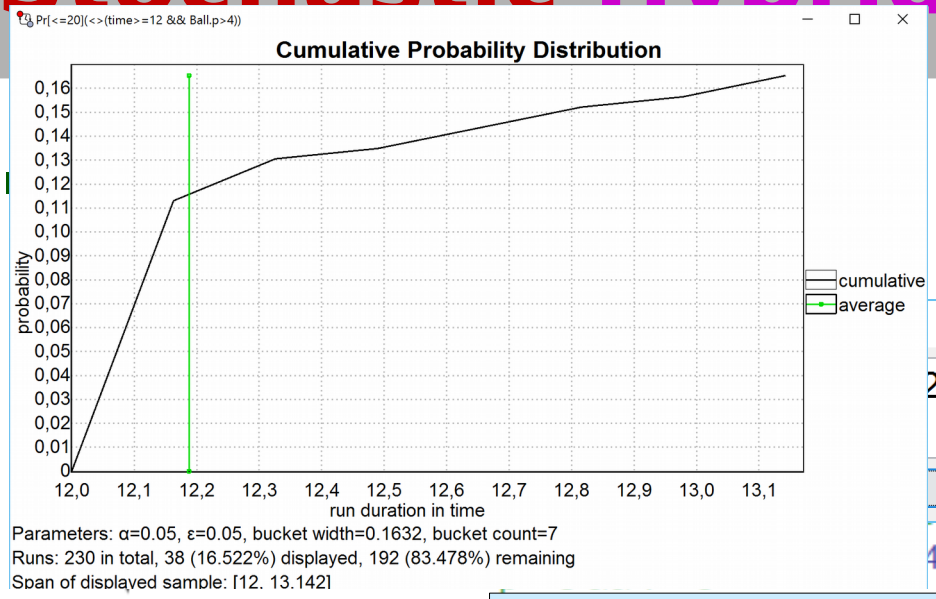


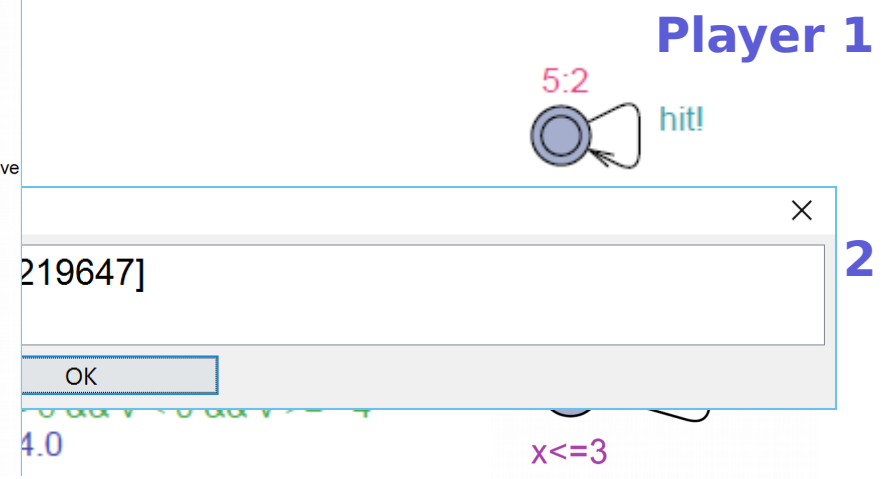
Pr[<=20](<>(time>=12 && Ball1.p>4))

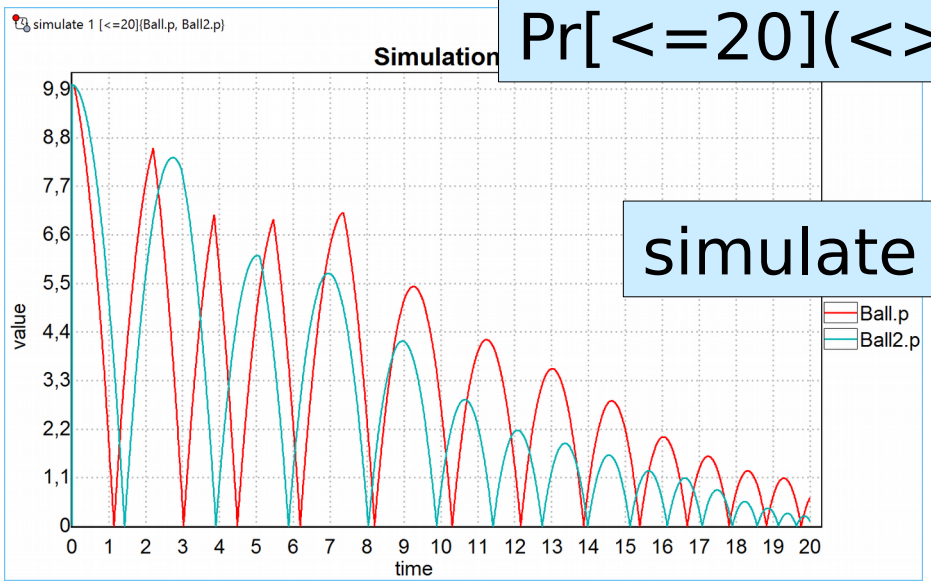
simulate 1 [<=20]{Ball1.p, Ball2.p}



Stochastic Hybrid Systems





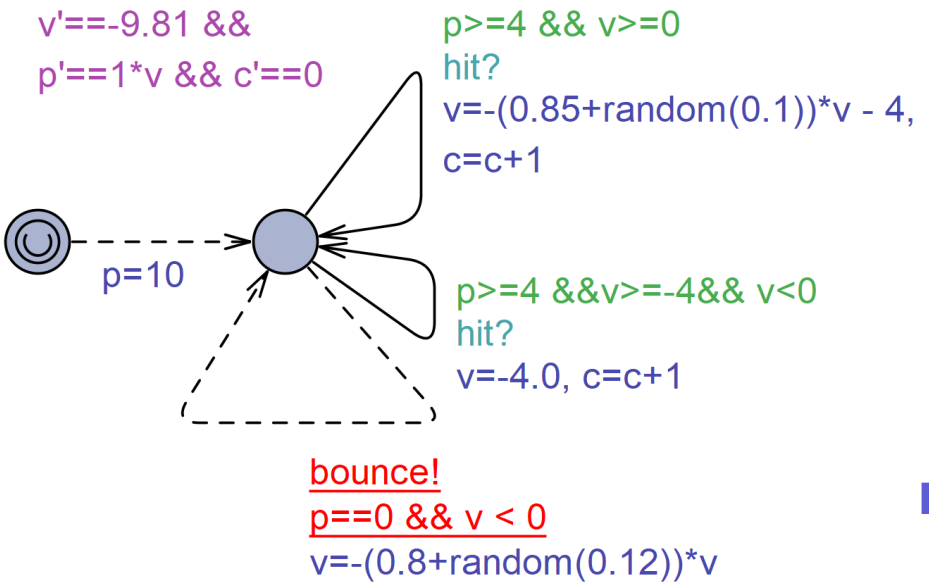


Pr[<=20](<>(time>=12 && Ball1.p>4))

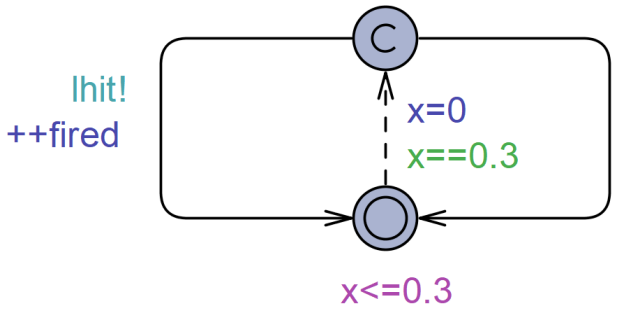
simulate 1 [<=20]{Ball1.p, Ball2.p}

Optimal Player

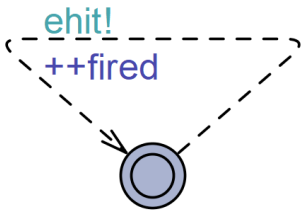
Ball



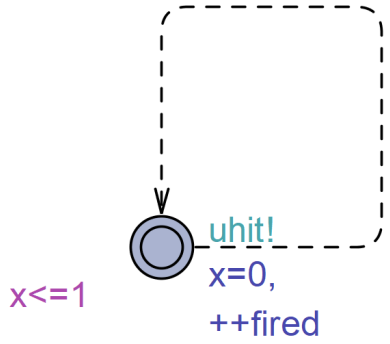
Learner Player

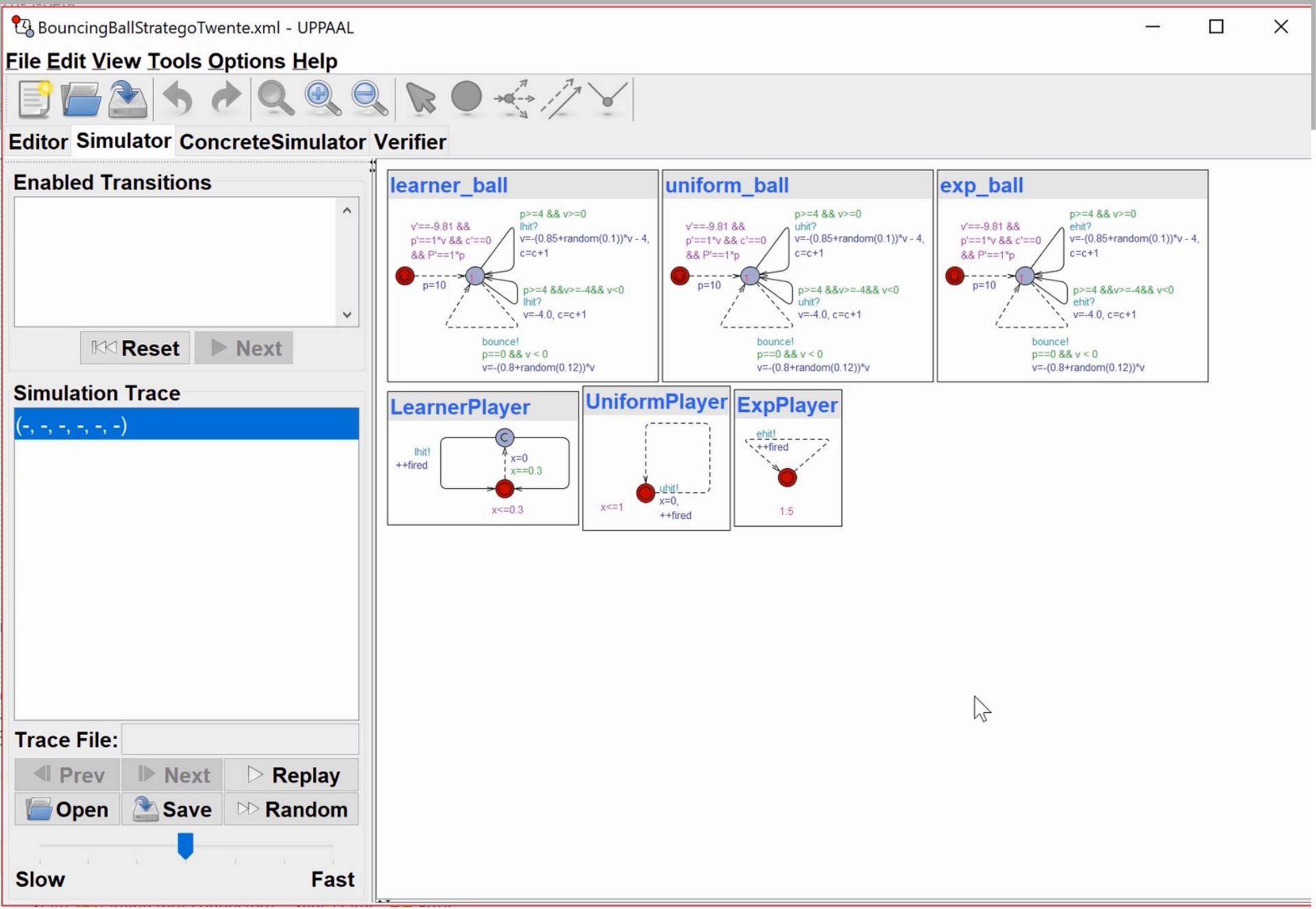


Expo Player

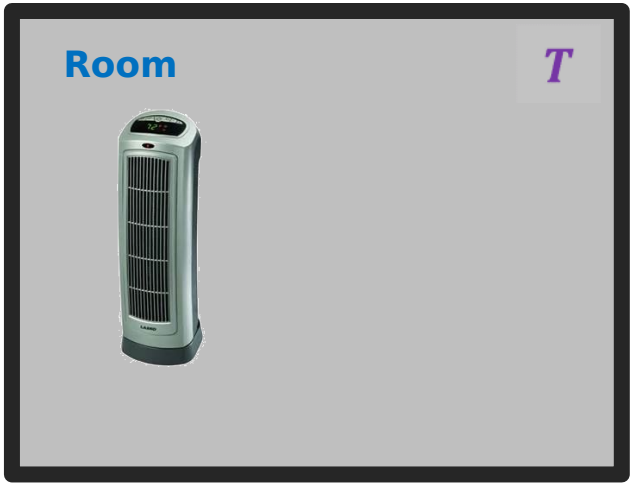


Uni Player

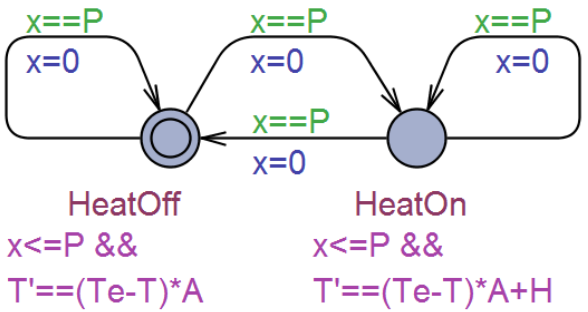



BouncingBallStrategoTwente.xml - UPPAAL
File Edit View Tools Options Help
Editor Simulator ConcreteSimulator Verifier
Enabled Transitions
Reset
Next
Simulation Trace
(-, -, -, -, -, -)
Trace File:
Prev
Next
Replay
Open
Save
Random
Slow
Fast
learner_ball
v'==-9.81 &&
p'==1*v && c'==0
&& P'==1*p
p>=4 && v>=0
lhit?
v=-(0.85+random(0.1))*v - 4,
c=c+1
p=10
p>=4 &&v>=-4&& v<0
lhit?
v=-4.0, c=c+1
bounce!
p==0 && v < 0
v=-(0.8+random(0.12))*v
uniform_ball
v'==-9.81 &&
p'==1*v && c'==0
&& P'==1*p
p>=4 && v>=0
uhit?
v=-(0.85+random(0.1))*v - 4,
c=c+1
p=10
p>=4 &&v>=-4&& v<0
uhit?
v=-4.0, c=c+1
bounce!
p==0 && v < 0
v=-(0.8+random(0.12))*v
exp_ball
v'==-9.81 &&
p'==1*v && c'==0
&& P'==1*p
p>=4 && v>=0
ehit?
v=-(0.85+random(0.1))*v - 4,
c=c+1
p=10
p>=4 &&v>=-4&& v<0
ehit?
v=-4.0, c=c+1
bounce!
p==0 && v < 0
v=-(0.8+random(0.12))*v
LearnerPlayer
lhit!
++fired
x=0
x==0.3
x<=0.3
UniformPlayer
x<=1
uhit!
x=0,
++fired
ExpPlayer
ehit!
++fired
1:5

1-Room / 1-Window Game

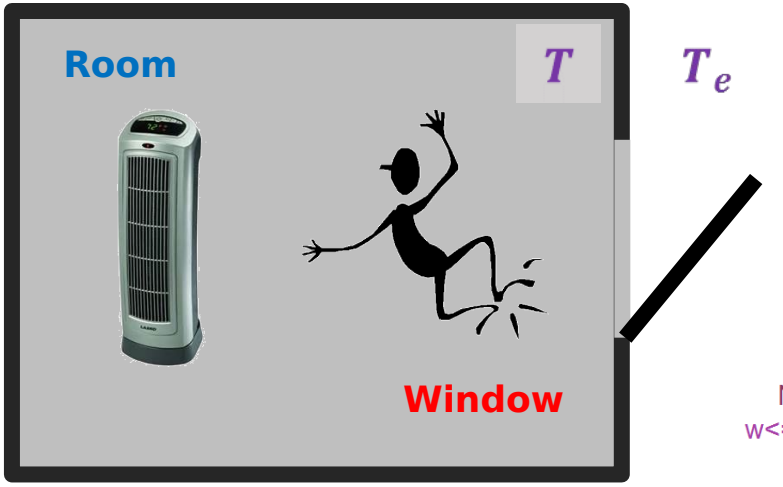


T_e




```
const double Tg = 21.0; // room temp. goal
const double Te = 15.0; // environment temp.
const double H = 0.04;  // power of heater
const double Aclosed = 0.002; // heat loss when window closed
const double Aopen = 0.004;   // heat loss when window open
const int P = 15; // heater switching period
const int h = 60; // 1 hour = 60 time units
```

1-Room / 1-Window Game

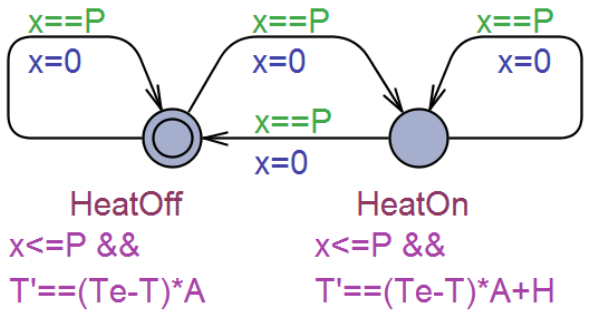


T_e

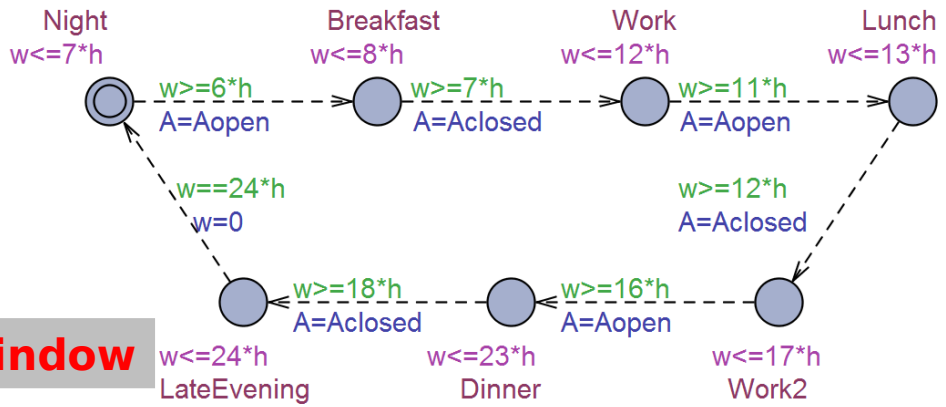
Find **strategy** that minimizes expected **discomfort**:

$$D(H) = \int_{t=0}^{t=H} \left(T(t) - T_g(t)\right)^2 dt$$

Room

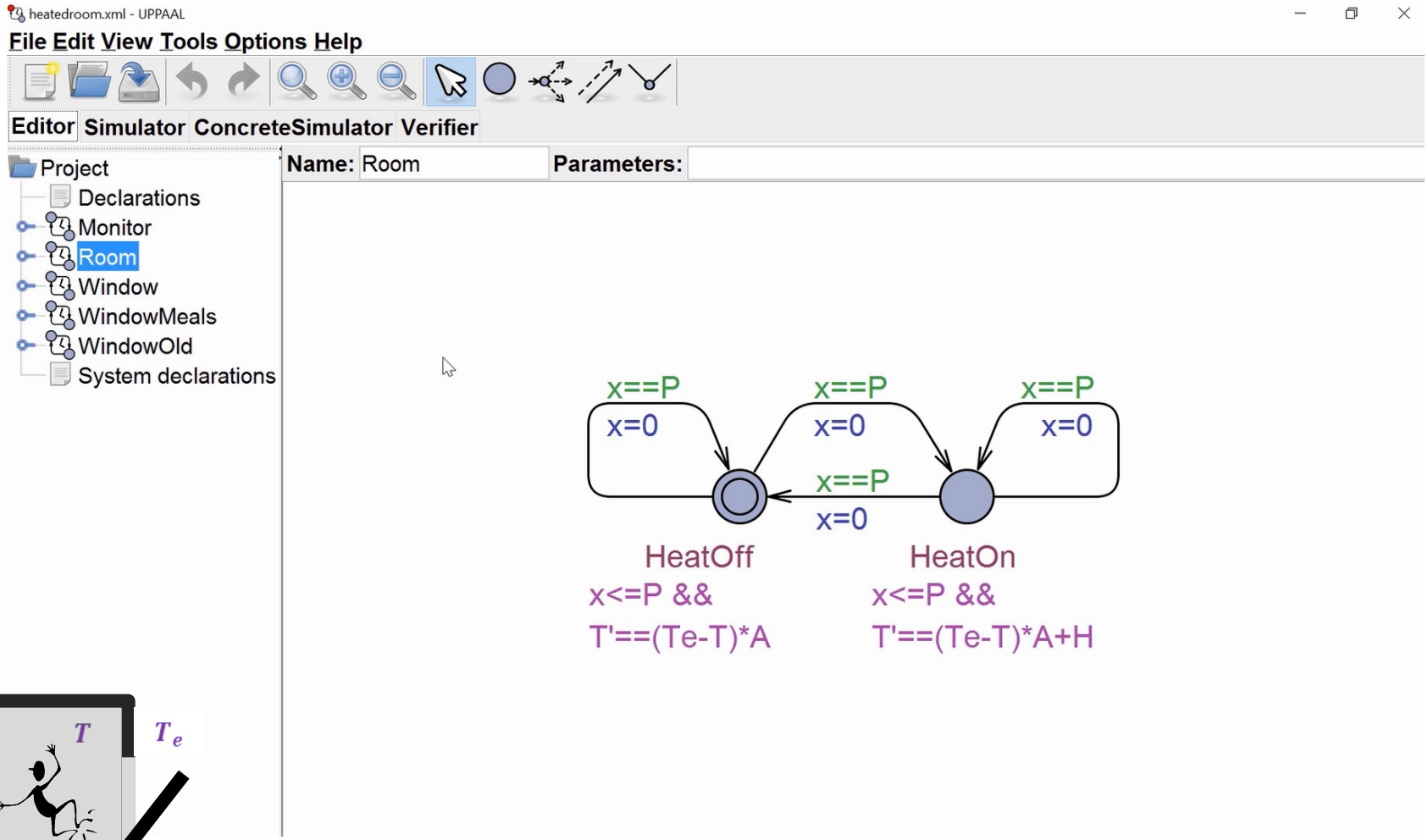


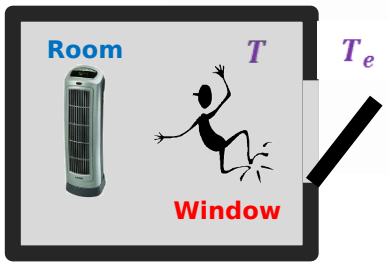
Window



Uniform Dist

UPPAAL Synthesis – Demo



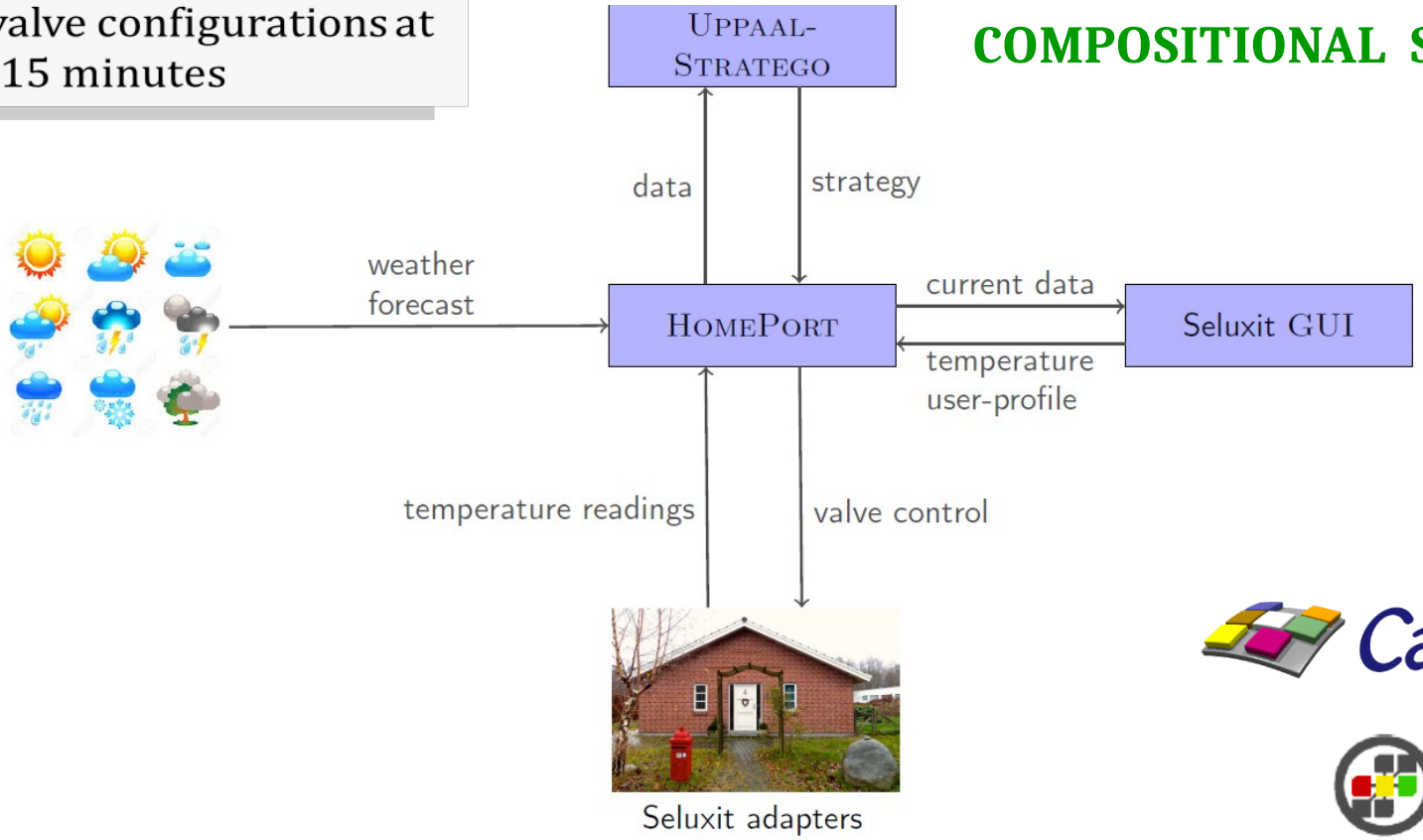


Full Floor Heating Case

CHALLENGE
2^{11} valve configurations at each 15 minutes

ON-LINE SYNTHESIS

COMPOSITIONAL SYNTHESIS



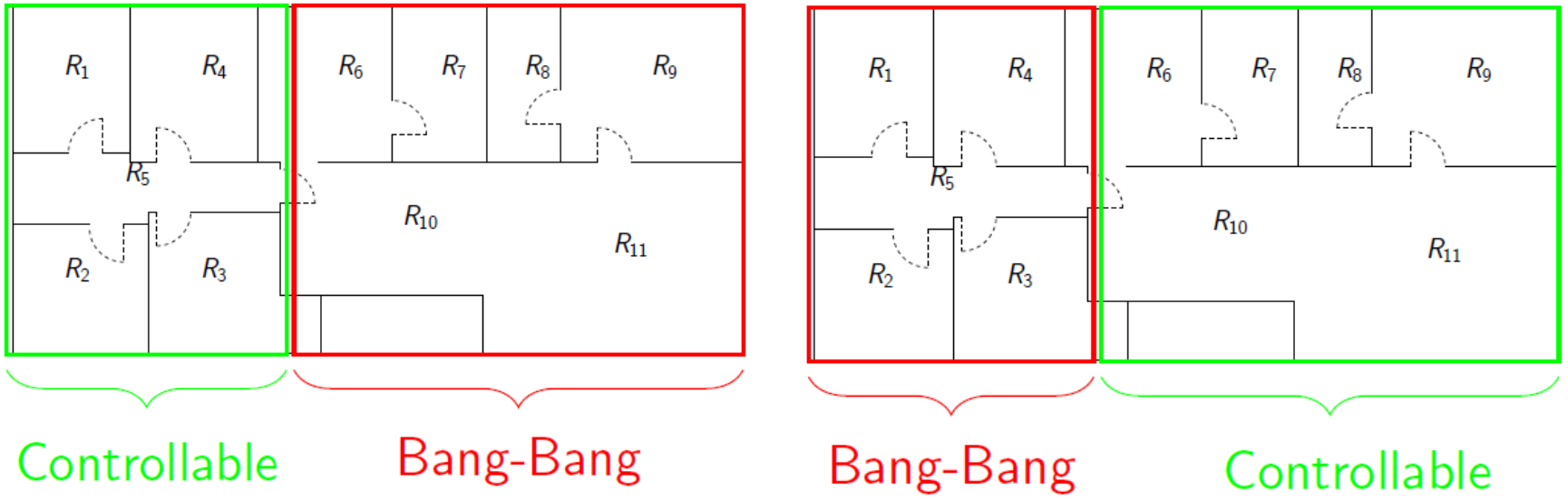
Cassting

seluxit

Larsen, Mikucionis, Muñiz, Srba, Taankvist, TACAS 16

Compositional Synthesis

- Split the valves into controllable and fixed (controlled via Bang-Bang)
- Synthesize a strategy for controllable valves
- Swap the controllable and fixed valves and synthesise another strategy
- Merge strategies.



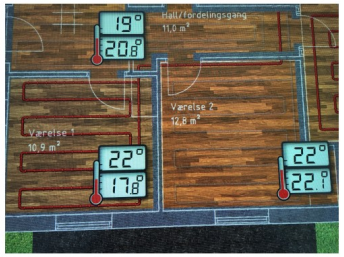
$(2^{5h} + 2^{6h})$ instead of 2^{11h} decision choices (in our case $h = 3$)

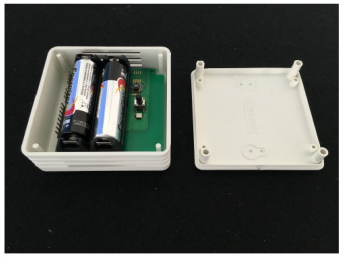
Full Floor Heating Case

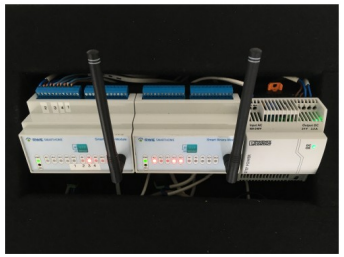
3 day scenario

Weather	Distance			Energy		
	Bang-Bang	Stratego	imp.	Bang-Bang	Stratego	imp.
Aalborg	14583	8342	**43%**	14180	12626	**10%**
Anadyr	2385515	1483272	**37%**	23040	22475	**2%**
Ankara	17985	10464	**41%**	17468	15684	**10%**
Minneapolis	22052	12175	**44%**	18165	15882	**12%**
Murmansk	399421	187941	**52%**	22355	21011	**6%**

Weather	Distance			Energy		
	Bang-Bang	Stratego	imp.	Bang-Bang	Stratego	imp.
Aalborg	14583	8552	**41%**	14180	12590	**11%**
Anadyr	2385515	1503448	**36%**	23040	22371	**2%**
Ankara	17985	10511	**41%**	17468	15697	**10%**
Minneapolis	22052	12725	**42%**	18165	15837	**12%**
Murmansk	399421	191441	**52%**	22355	20923	**6%**



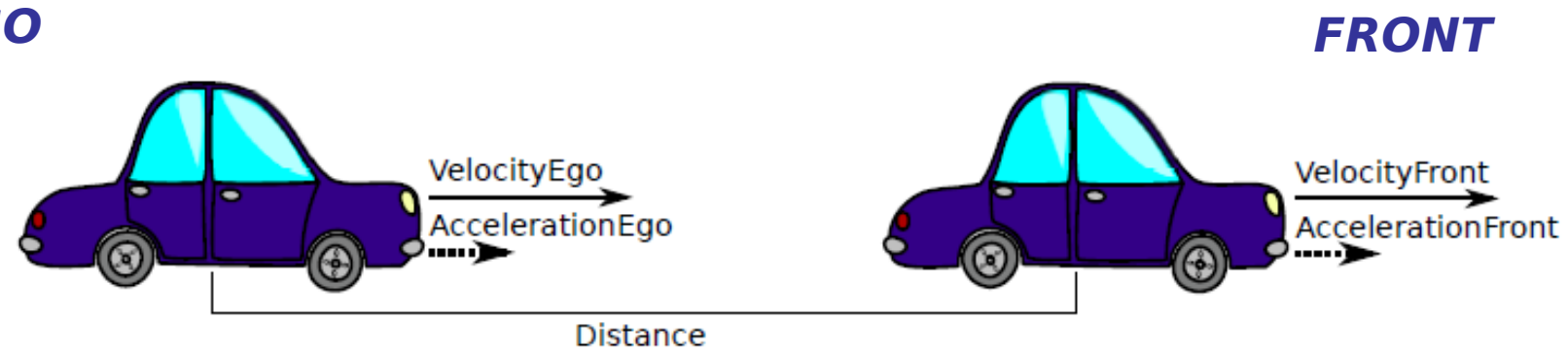




Evaluation of under modified parameters (0-20%)

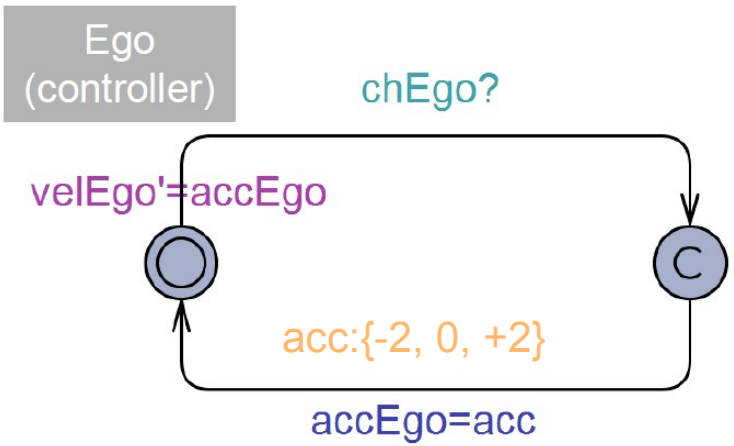
Safe & Adaptive Cruice Control

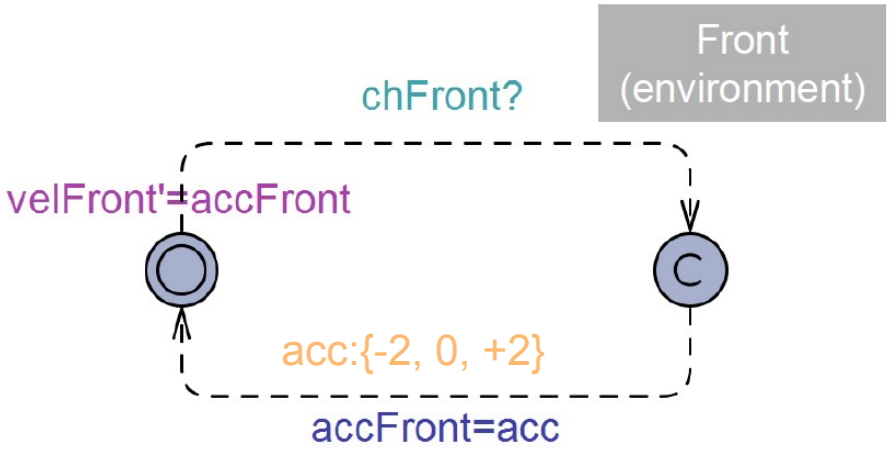
Controller Synthesis for Switched Hybrid Games

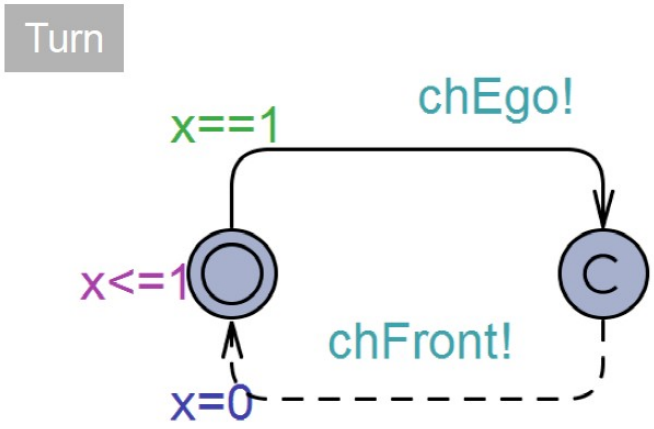


AALBORG UNIVERSITET

Two Player Game (simplified)







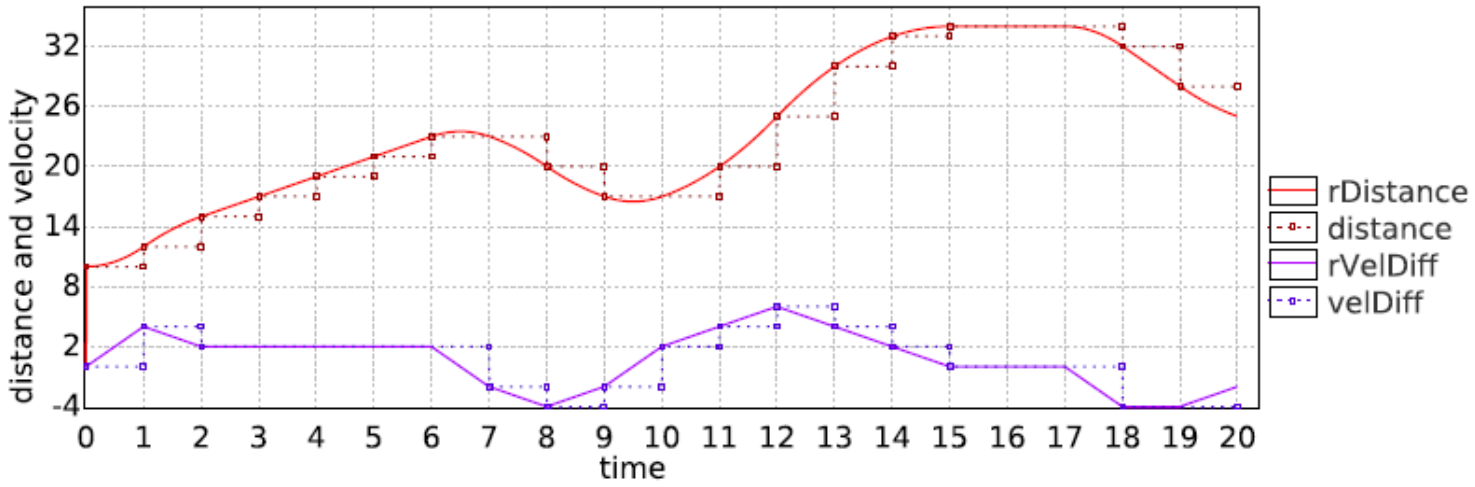
distance' == (velEgo-velFront) &&
D' == distance

Q: find strategy for Ego

Discretization

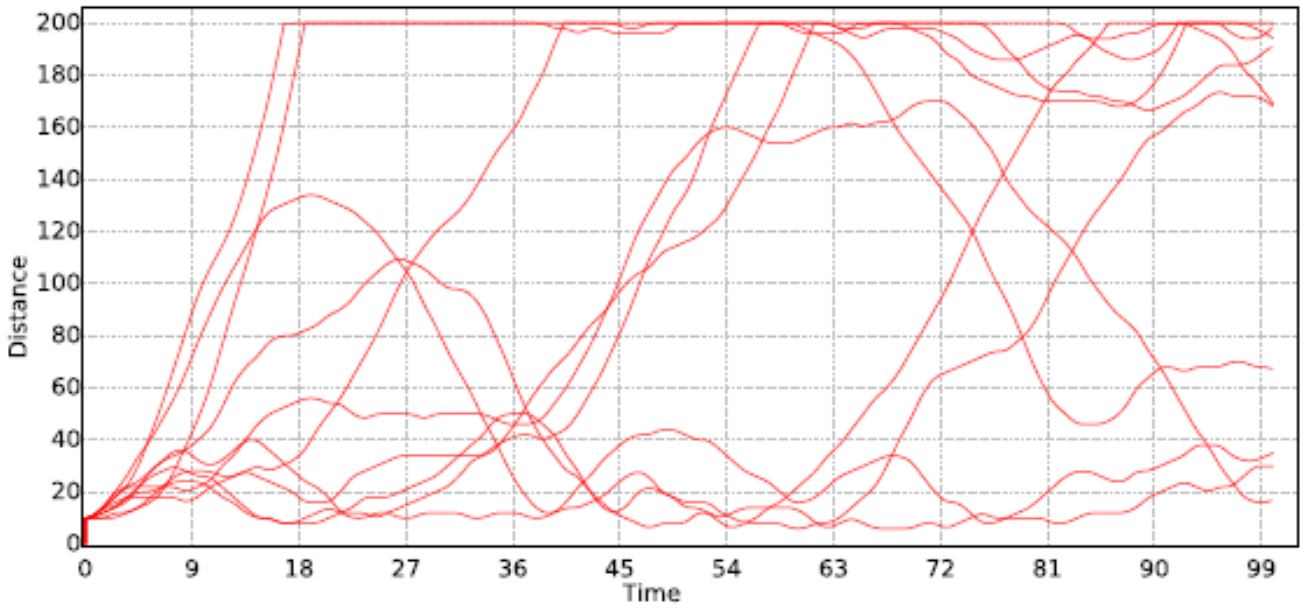
Discrete

```
void updateDiscrete(){
    int oldVel, newVel;
    oldVel = velocityFront - velocityEgo;
    velocityEgo = velocityEgo + accelerationEgo;
    velocityFront = velocityFront + accelerationFront;
    newVel = velocityFront - velocityEgo;
    if (distance > maxSensorDistance) {
        distance = maxSensorDistance + 1;
```



Safety Strategy

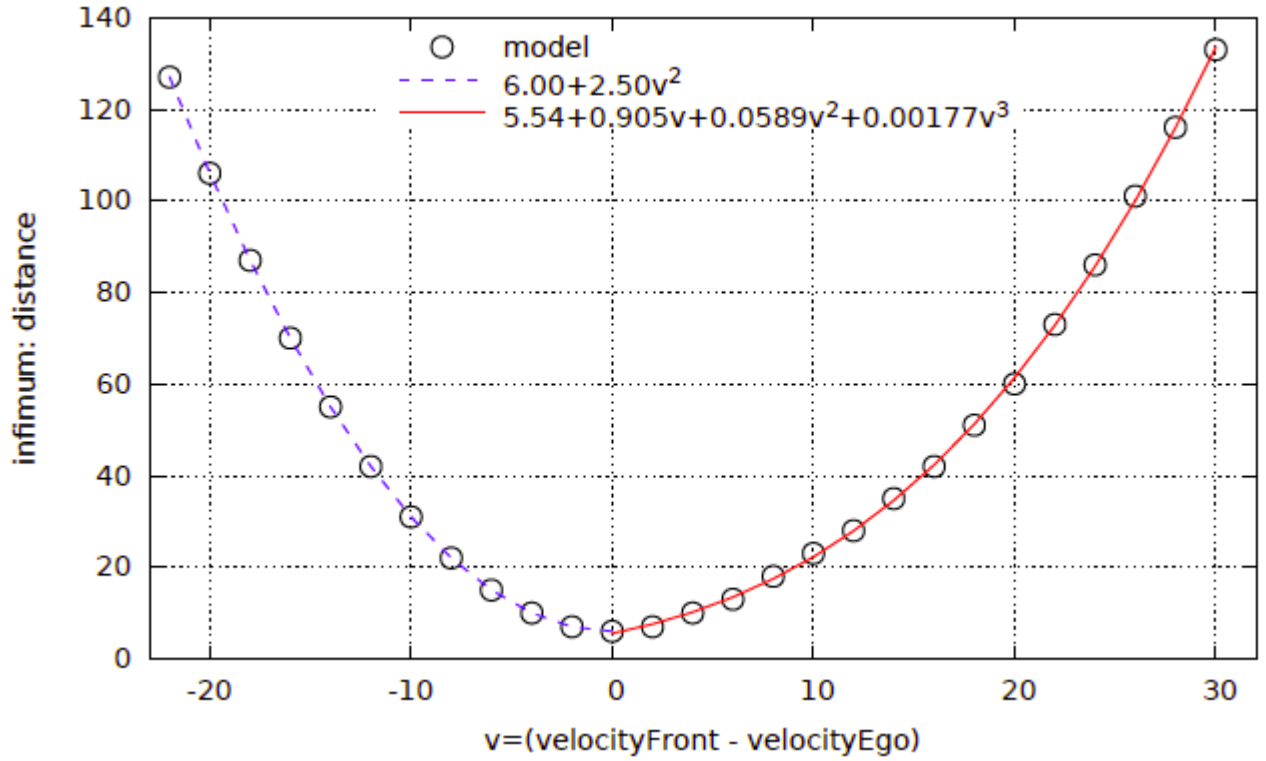
```
strategy safe = control: A[] distance > 5
```



Safety Strategy

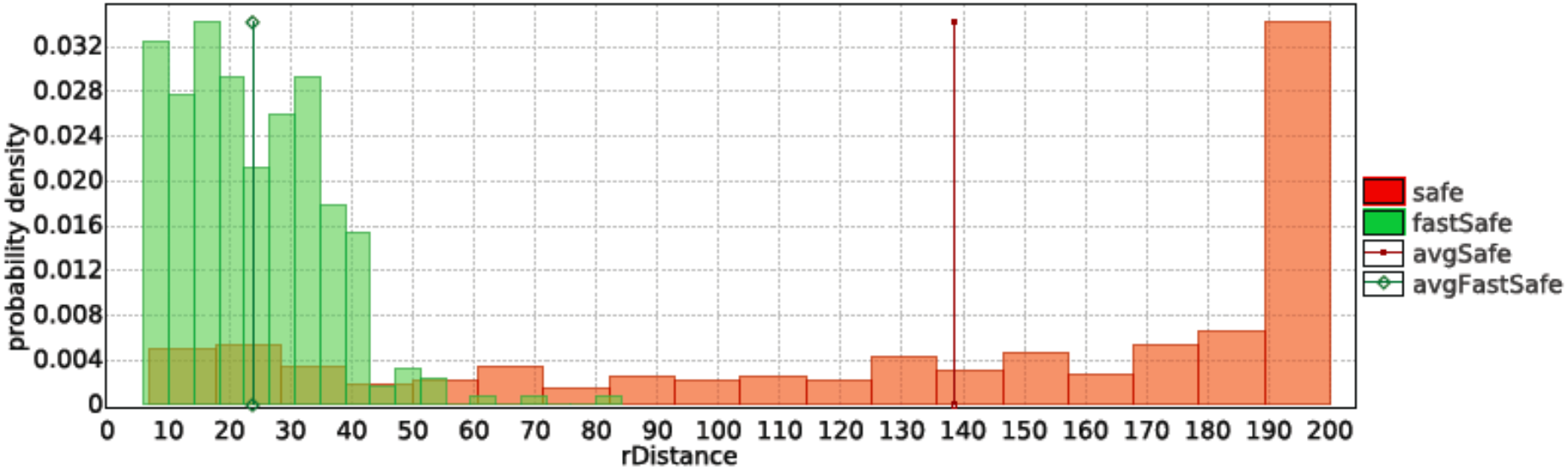
```
strategy safe = control: A[] distance > 5
```

```
inf{velosityFront-velosityEgo==v}: distance under safe
```

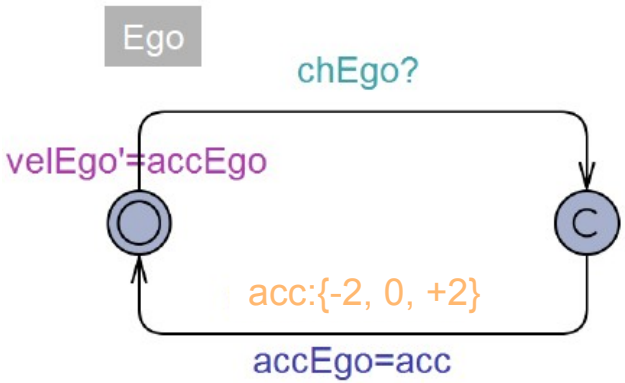


Optimal and Safe Strategy

```
strategy safeFast = minE (D) [<=100]: <> time >= 100 under safe
```



Safety Strategy (Code)



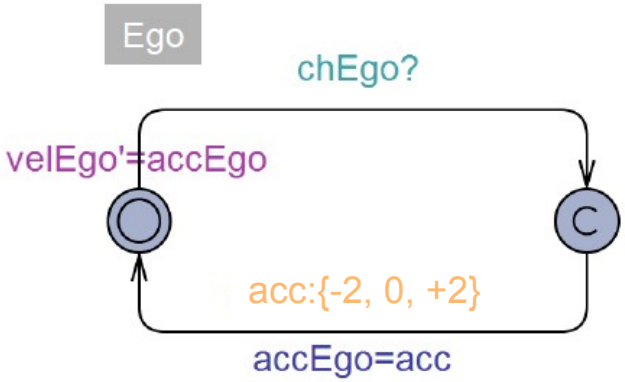
adaptiveCruiseControl - Notepad

File Edit Format View Help

```
State: ( Ego.Negative_acc Front.No_acceleration System.Wait Monitor._id12 ) #action=0
distance=47 velocityEgo=6 accelerationEgo=-2 velocityFront=12 accelerationFront=0
While you are in        (waitTimer<=1), wait.

State: ( Ego.No_acc Front.Positive_acc System.Wait Monitor._id12 ) #action=0 distance=83
velocityEgo=13 accelerationEgo=0 velocityFront=14 accelerationFront=2
While you are in        true, wait.

State: ( Ego.Choose Front.Choose System.Done Monitor._id12 ) #action=0
distance=181 velocityEgo=0 accelerationEgo=0 velocityFront=0 accelerationFront=0
When you are in         take transition Ego.Choose->Ego.Neg, tau, accelerationEgo := 0
}
When you are in         take transition Ego.Choose->Ego.Po   { velocityEgo <
maxVelocity             accelerationEgo := 2 }
When you are in         take transition Ego.Choose->Ego.Neg  { velocityEgo >
minVelocity             accelerationEgo := -2 }

State: ( Ego Front.Choose System.Done Monitor._id12 ) #action=0 distance=199
velocityEgo=7 velocityEgo=-2 velocityFront=15 accelerationFront=0
While you are in        true, wait.

State: ( Ego.Negative_acc Front.No_acceleration System.Wait Monitor._id12 ) #action=0
distance=49 velocityEgo=6 velocityFront=14 accelerationFront=2
While you are in        true, wait.

State: ( Ego.Positive_acc Front.Choose System.Done Monitor._id12 ) #action=0 distance=88
velocityEgo=0 accelerationEgo=2 velocityFront=11 accelerationFront=0
While you are in        true, wait.

State: ( Ego.Positive_acc Front.Choose System.Done Monitor._id12 ) #action=0 distance=174
velocityEgo=18 accelerationEgo=2 velocityFront=17 accelerationFront=2
While you are in        true, wait.

State: ( Ego.No_acc Front.Negative_acc System.Done Monitor._id12 ) #action=0 distance=147
```

4Mb

Safety Strategy (Code)



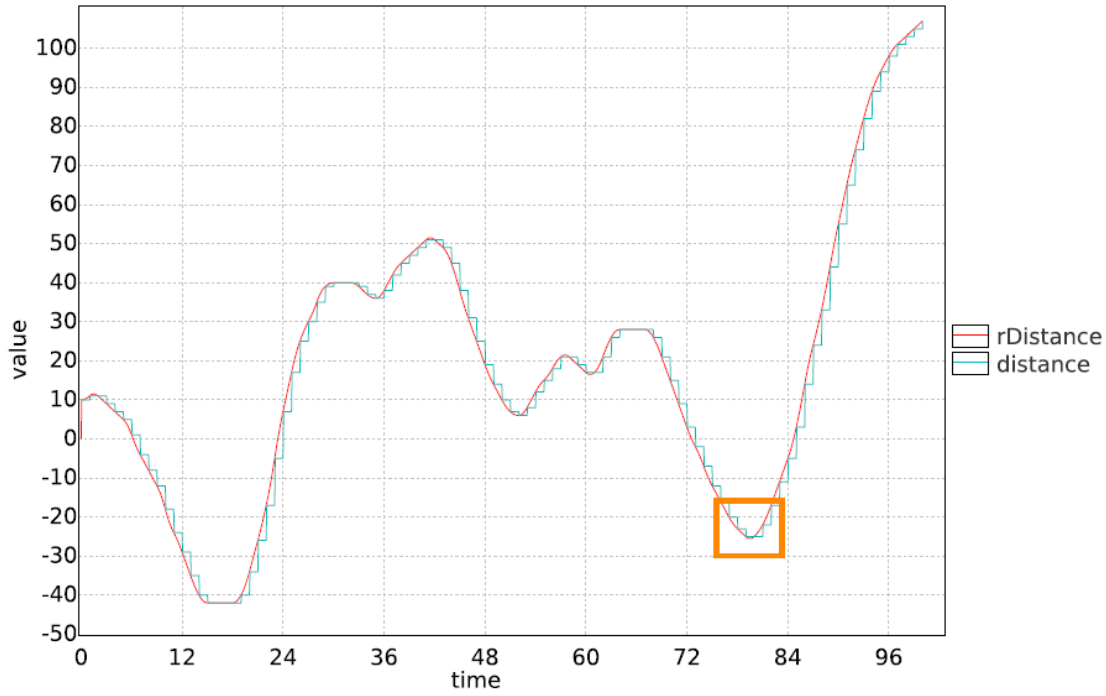

```
Ego.Choose <= 0: 3 (1481817.0)
Ego.Choose > 0
|  velocityEgo <= -10: 0 (39705.0/18380.0)
|  velocityEgo > -10
|  |  distance <= 200
|  |  |  velocityEgo <= 18
|  |  |  |  velocityEgo <= 12
|  |  |  |  |  distance <= 184
|  |  |  |  |  |  velocityEgo <= 0: 2 (331464.0/208577.0)
|  |  |  |  |  |  velocityEgo > 0
|  |  |  |  |  |  |  distance <= 122
|  |  |  |  |  |  |  |  distance <= 102: 2 (132918.0/80433.0)
|  |  |  |  |  |  |  |  distance > 102
|  |  |  |  |  |  |  |  |  velocityEgo <= 2
|  |  |  |  |  |  |  |  |  |  velocityFront <= 12: 1 (6255.0/4155.0)
|  |  |  |  |  |  |  |  |  |  velocityFront > 12
|  |  |  |  |  |  |  |  |  |  |  velocityFront <= 13: 2 (162.0)
|  |  |  |  |  |  |  |  |  |  |  velocityFront > 13
|  |  |  |  |  |  |  |  |  |  |  |  distance <= 110: 2 (870.0/363.0)
|  |  |  |  |  |  |  |  |  |  |  |  distance > 110
|  |  |  |  |  |  |  |  |  |  |  |  |  velocityFront <= 15
|  |  |  |  |  |  |  |  |  |  |  |  |  |  velocityFront <= 14: 1 (207.0/99.0)
|  |  |  |  |  |  |  |  |  |  |  |  |  |  velocityFront > 14: 2 (63.0)
|  |  |  |  |  |  |  |  |  |  |  |  |  velocityFront > 15
|  |  |  |  |  |  |  |  |  |  |  |  |  |  distance <= 116
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  velocityFront <= 17: 2 (126.0/54.0)
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  velocityFront > 17: 1 (129.0/39.0)
|  |  |  |  |  |  |  |  |  |  |  |  |  |  distance > 116: 1 (108.0)
|  |  |  |  |  |  |  |  |  |  velocityEgo > 2
|  |  |  |  |  |  |  |  |  |  velocityEgo <= 4: 1 (7689.0/4929.0)
```

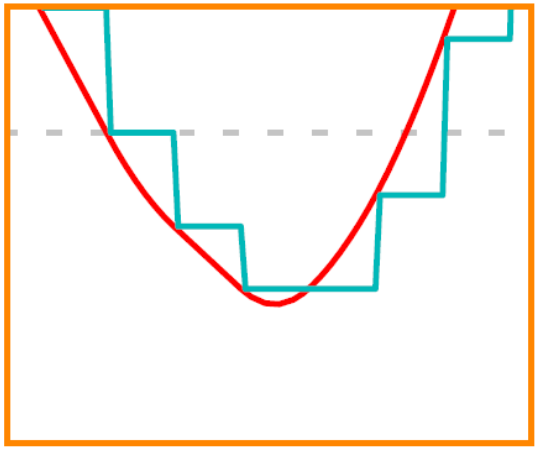
Learning Algorithm for Decision Tre

→65 line

(Jan Kretisnsky, Pranav Ashkot, TUM

Safe?
Problem of Discretization





Euler Approximate Solutions

Constants

$$L_j = \max_{x,y \in S,\ x \neq y} \frac{\|f_j(y) - f_j(x)\|}{\|y - x\|}$$

j'th modes

$$\dot{x} = f_j(x)$$

$$C_j = \max_{x \in S} L_j \|f_j(x)\|$$

$$\lambda_j = \max_{x,y \in T,\ x \neq y} \frac{\langle f_j(y) - f_j(x), y - x \rangle}{\|y - x\|^2}$$

Provable Upper Bound Error

- *if* $\lambda_j < 0$:

$$\delta_j(\rho, t) = \left(\rho^2 e^{\lambda_j t} + \frac{C_j^2}{\lambda_j^2} \left(t^2 + \frac{2t}{\lambda_j} + \frac{2}{\lambda_j^2} \left(1 - e^{\lambda_j t}\right) \right) \right)^{\frac{1}{2}}$$

- *if* $\lambda_j = 0$:

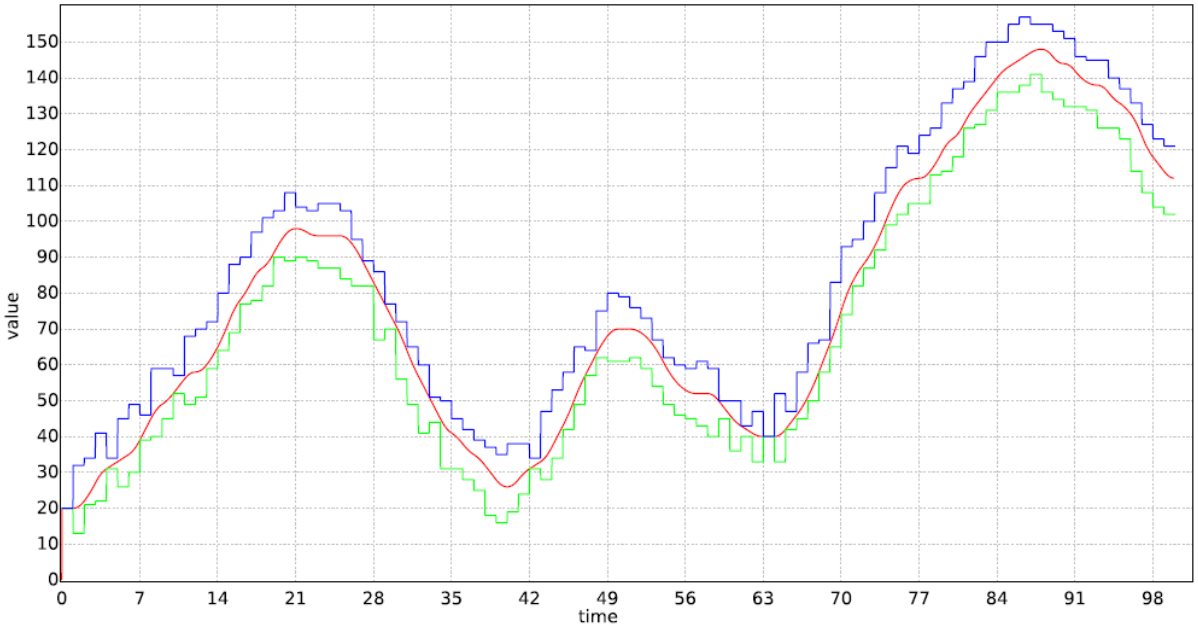
$$\delta_j(\rho, t) = \left(\rho^2 e^t + C_j^2 (-t^2 - 2t + 2(e^t - 1)) \right)^{\frac{1}{2}}$$

- *if* $\lambda_j > 0$:

$$\delta_j(\rho, t) = \left(\rho^2 e^{3\lambda_j t} + \frac{C_j^2}{3\lambda_j^2} \left(-t^2 - \frac{2t}{3\lambda_j} + \frac{2}{9\lambda_j^2} \left(e^{3\lambda_j t} - 1\right) \right) \right)^{\frac{1}{2}}$$

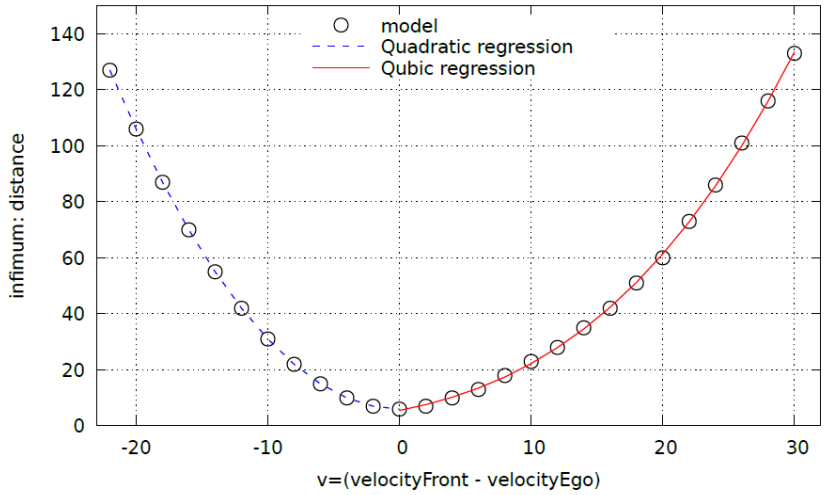
Le Coent, De Vuyst, Chamoin, Fribourg. Control synthesis of nonlinear sampled switched systems using Euler's method. 3rd International Workshop on Symbolic and Numerical Methods for Reachability Analysis, 2017,

Results

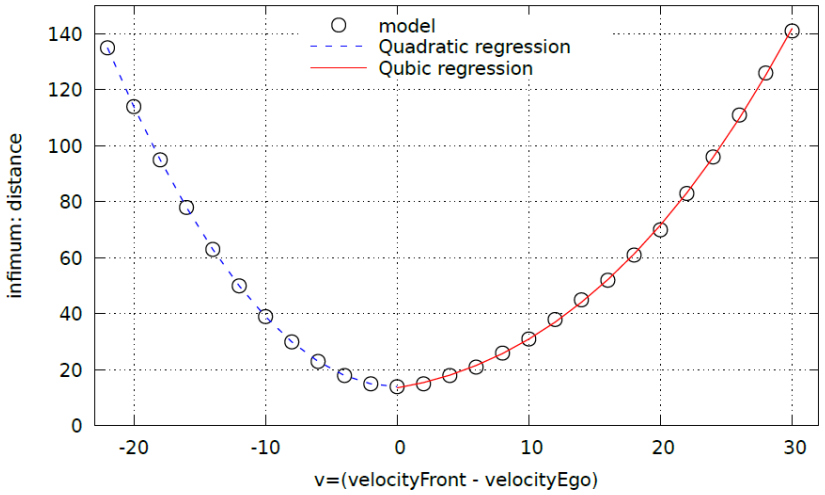


Guaranteed Bounds Using Euler

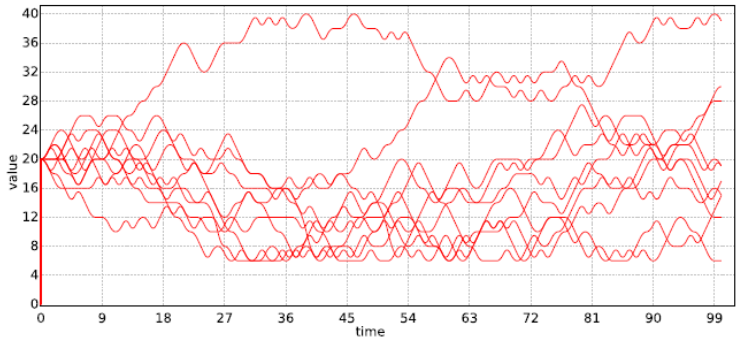
Results



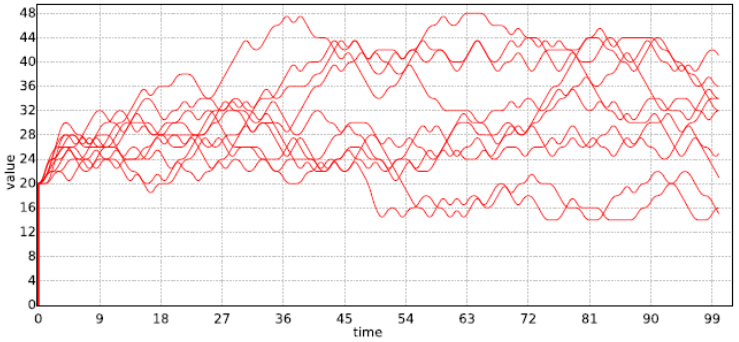
(a) Strategy computed in [12]. Figure from [12].



(b) Safe and guaranteed strategy.



(a) Optimised strategy computed in [12]. Figure from [12].



(b) Optimised safe and guaranteed strategy.

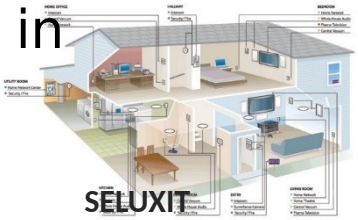
Updated UPPAAL Model

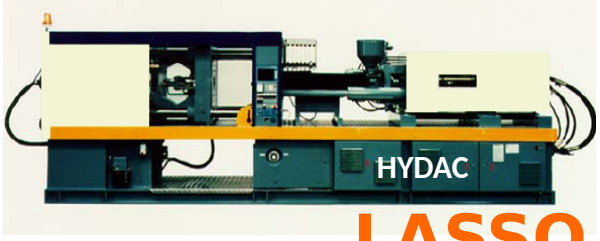
```
void updateDiscrete(){

    velocityEgo_gua_evol[0] = velocityEgo_gua
    velocityEgo_gua_evol[1] = velocityEgo_gua
    velocityFront_gua_evol[0] = velocityFront
    ;
    velocityFront_gua_evol[1] = velocityFro
    ;

    if (distance_gua_evol[0] > maxSensorDis
        distance_gua_evol[0] = maxSensorDis
        distance_gua_evol[1] = maxSensorDis
    } else {
        eulerDiscrete();
        distance_gua_evol[0] = distance_gua
    [0]-accelerationFront) - (velocityEgo_gu
    accelerationFront - accelerationEgo)/2;
        distance_gua_evol[1] = distance_gua
    [0]-accelerationFront) - (velocityEgo_gu
    accelerationFront - accelerationEgo)/2;

        if (distance_gua_evol[1] > maxSenso
            distance_gua_evol[1] = maxSenso
        }
    }
}
```

```
void eulerDiscrete(){
    //double velEgo, velFront,
    double dist, velEgo, velFront, delta;
    double memdist_min, memdist_max, memVF_min, memVF_max, memVE_min,
    memVE_max;

    int i;

    dist = (distance_gua_evol[0]+distance_gua_evol[1])/2;
    velFront = (velocityFront_gua_evol[0]+velocityFront_gua_evol[1])/2;
    velEgo = (velocityEgo_gua_evol[0]+velocityEgo_gua_evol[1])/2;

    delta = sqrt((distance_gua_evol[1]-distance_gua_evol[0])*(
    distance_gua_evol[1]-distance_gua_evol[0])/4 + (velocityFront_gua_evol
    [1]-velocityFront_gua_evol[0])*(velocityFront_gua_evol[1]-
    velocityFront_gua_evol[0])/4 + (velocityEgo_gua_evol[1]-
    velocityEgo_gua_evol[0])*(velocityEgo_gua_evol[1]-velocityEgo_gua_evol
    [0])/4);
    memdist_min = dist - delta;
    memdist_max = dist + delta;
    memVF_min = velFront - delta;
    memVF_max = velFront + delta;
    memVE_min = velEgo - delta;
    memVE_max = velEgo + delta;

    for (i=0;i<=euler_sub_step;i++){
        dist = dist + tau*(velFront - velEgo);
        velEgo = velEgo + tau*accelerationEgo;
        velFront = velFront + tau*accelerationFront;
        delta = delta_mode(delta,find_mode(accelerationFront,accelerationEgo)
    );
        memdist_min = mini(memdist_min,dist-delta);
        memdist_max = maxi(memdist_max,dist+delta);
        memVF_min = mini(memVF_min,velFront-delta);
        memVF_max = maxi(memVF_max,velFront+delta);
        memVE_min = mini(memVE_min,velEgo-delta);
        memVE_max = maxi(memVE_max,velEgo+delta);

    }

    distance_gua_evol[0] = floor(dist-delta);
    distance_gua_evol[1] = ceil(dist+delta);
    velocityFront_gua_evol[0] = floor(velFront-delta);
    velocityFront_gua_evol[1] = ceil(velFront+delta);
    velocityEgo_gua_evol[0] = floor(velEgo-delta);
    velocityEgo_gua_evol[1] = ceil(velEgo+delta);

    distance_gua[0] = floor(memdist_min);
    distance_gua[1] = ceil(memdist_max);
    velocityFront_gua[0] = floor(memVF_min);
    velocityFront_gua[1] = ceil(memVF_max);
    velocityEgo_gua[0] = floor(memVE_min);
    velocityEgo_gua[1] = ceil(memVE_max);

}
```

More Practical Synthesis …

- Zone-based climate control in pig-stables
- Profit-optimal, minimal-wear and energy-aware schedules for satelittes
- Personalized light control in home automation
- Energy- and comfort-optimal floor heating
- Safe and energy optimal control of hydralic pumps
- Safe and optimal car maneuvers (cruice control)



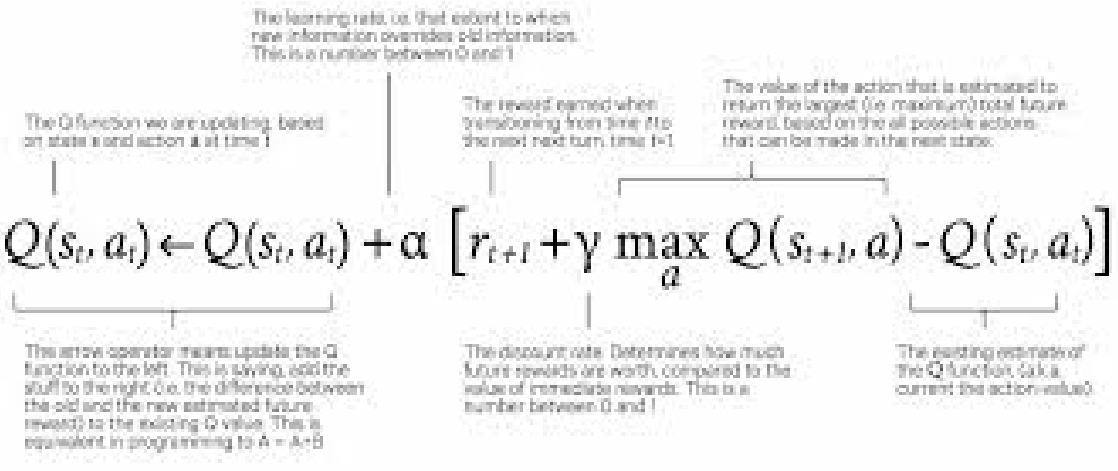


LASSO

Learning, Analysis, SynthesiS and Optimization of Cyber-Physical Systems

Thanks for your attention!

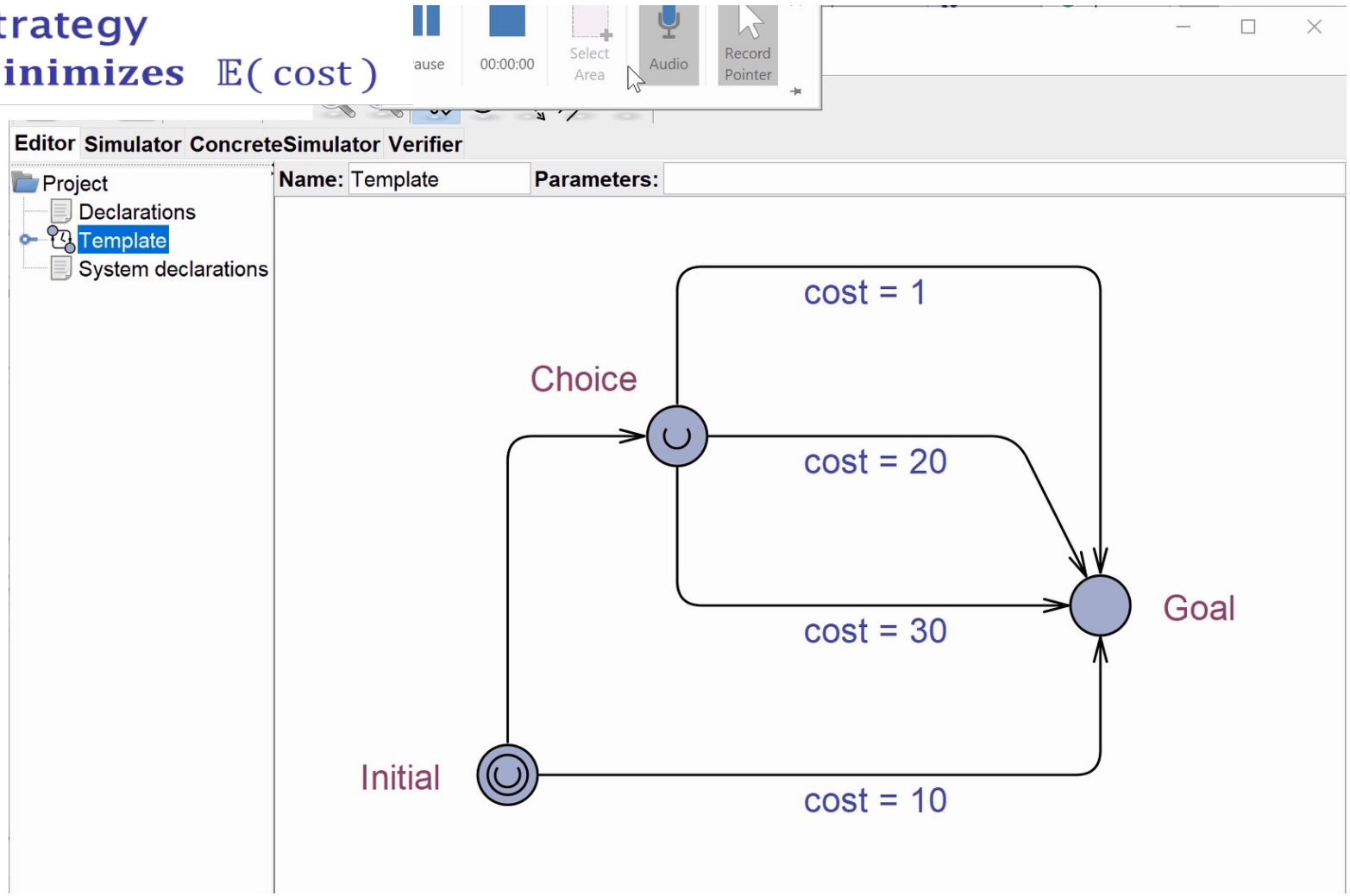
New Learning Algorithm



AALBORG UNIVERSITET

Bad Example

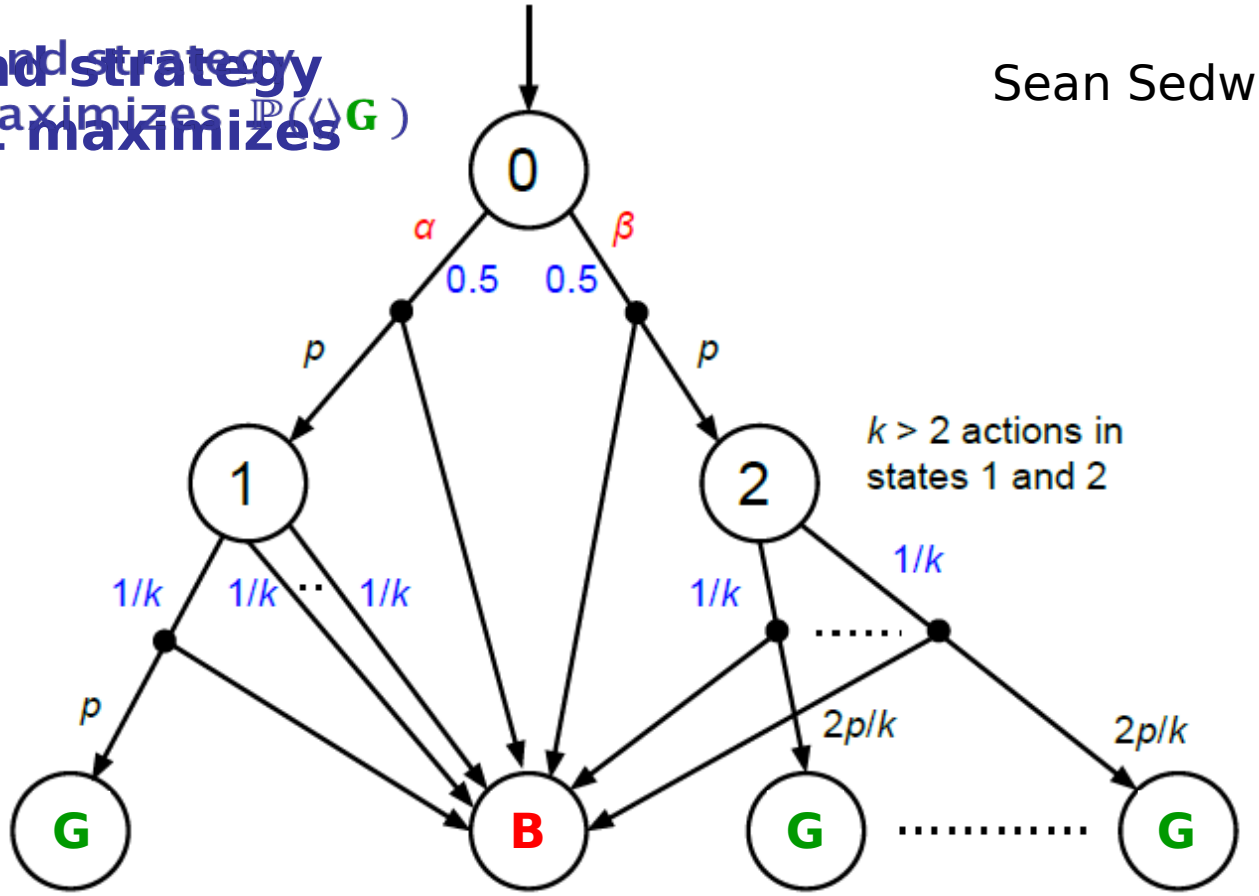
Find strategy
that minimizes $\mathbb{E}(\text{cost})$



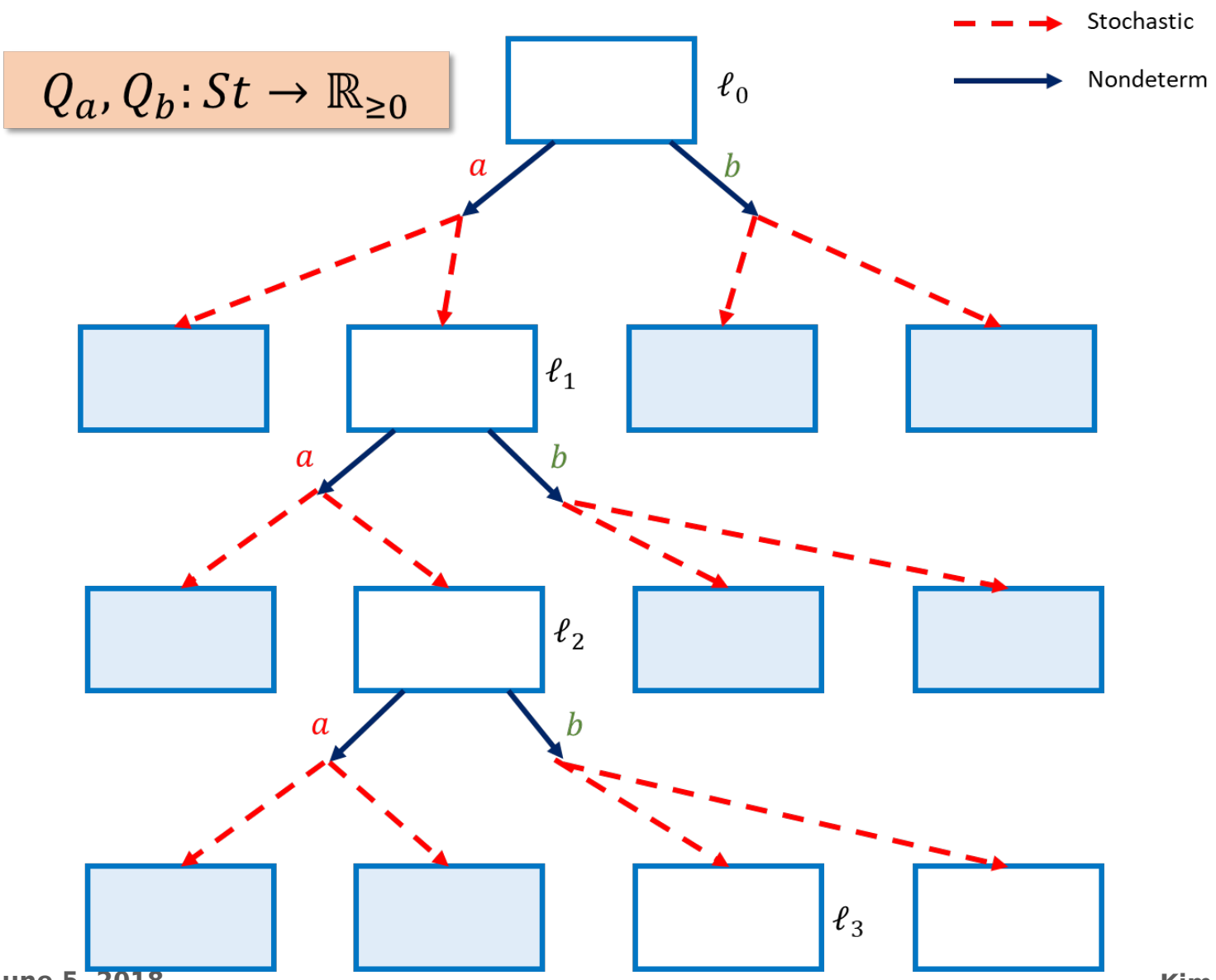
Bad Example

Find strategy that maximizes $\mathbb{P}(\Diamond \mathbf{G})$

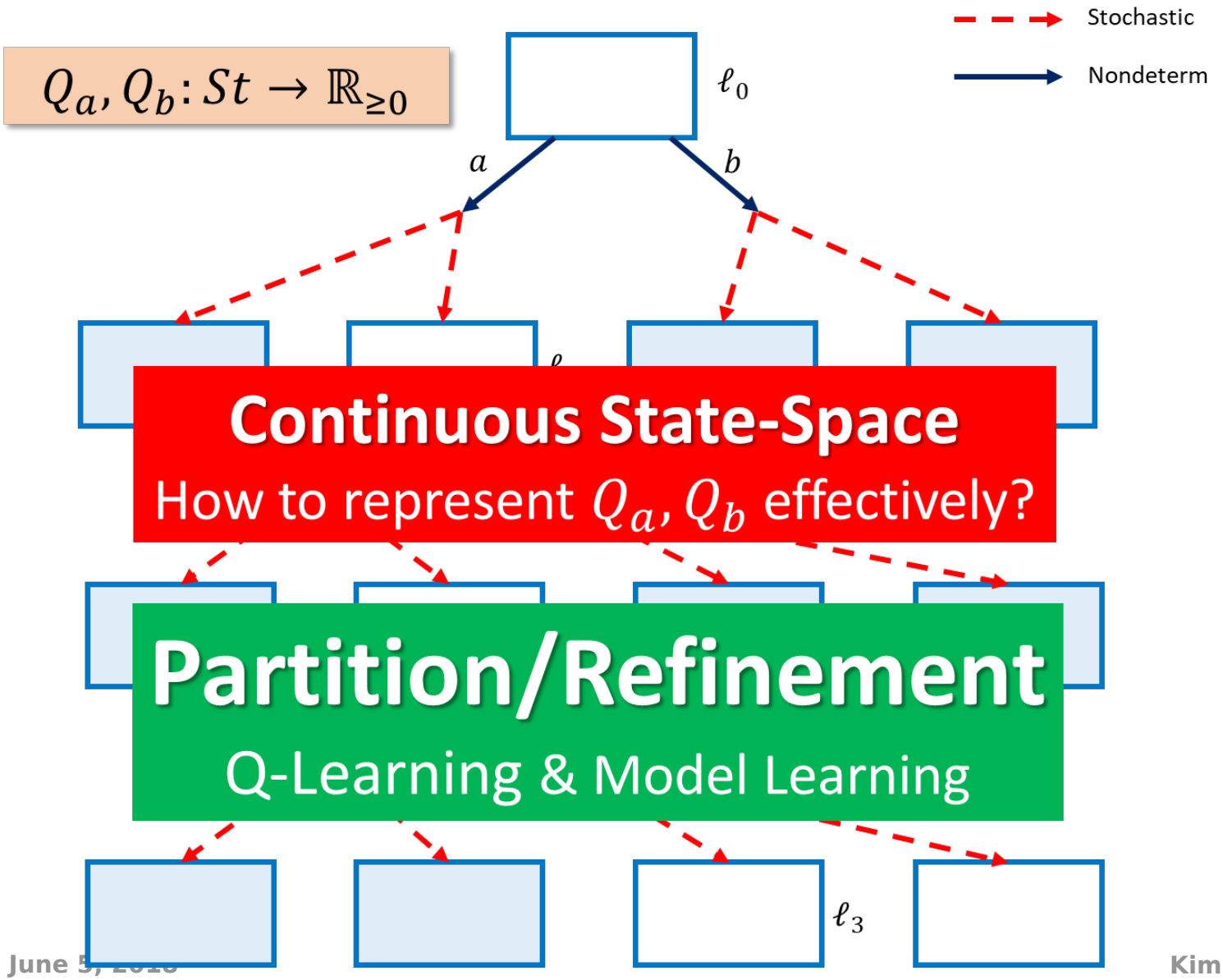
Sean Sedwards



Q- & M-Learning


$Q_a, Q_b : St \to \mathbb{R}_{\geq 0}$
Stochastic
Nondeterm
ℓ_0
ℓ_1
ℓ_2
ℓ_3
a
b
a
b
a
b

Q- & M-Learning


Stochastic
Nondeterm
$Q_a, Q_b : St \to \mathbb{R}_{\geq 0}$
ℓ_0
a
b
ℓ
Continuous State-Space
How to represent Q_a, Q_b effectively?
Partition/Refinement
Q-Learning & Model Learning
ℓ_3

Q- & M-Learning

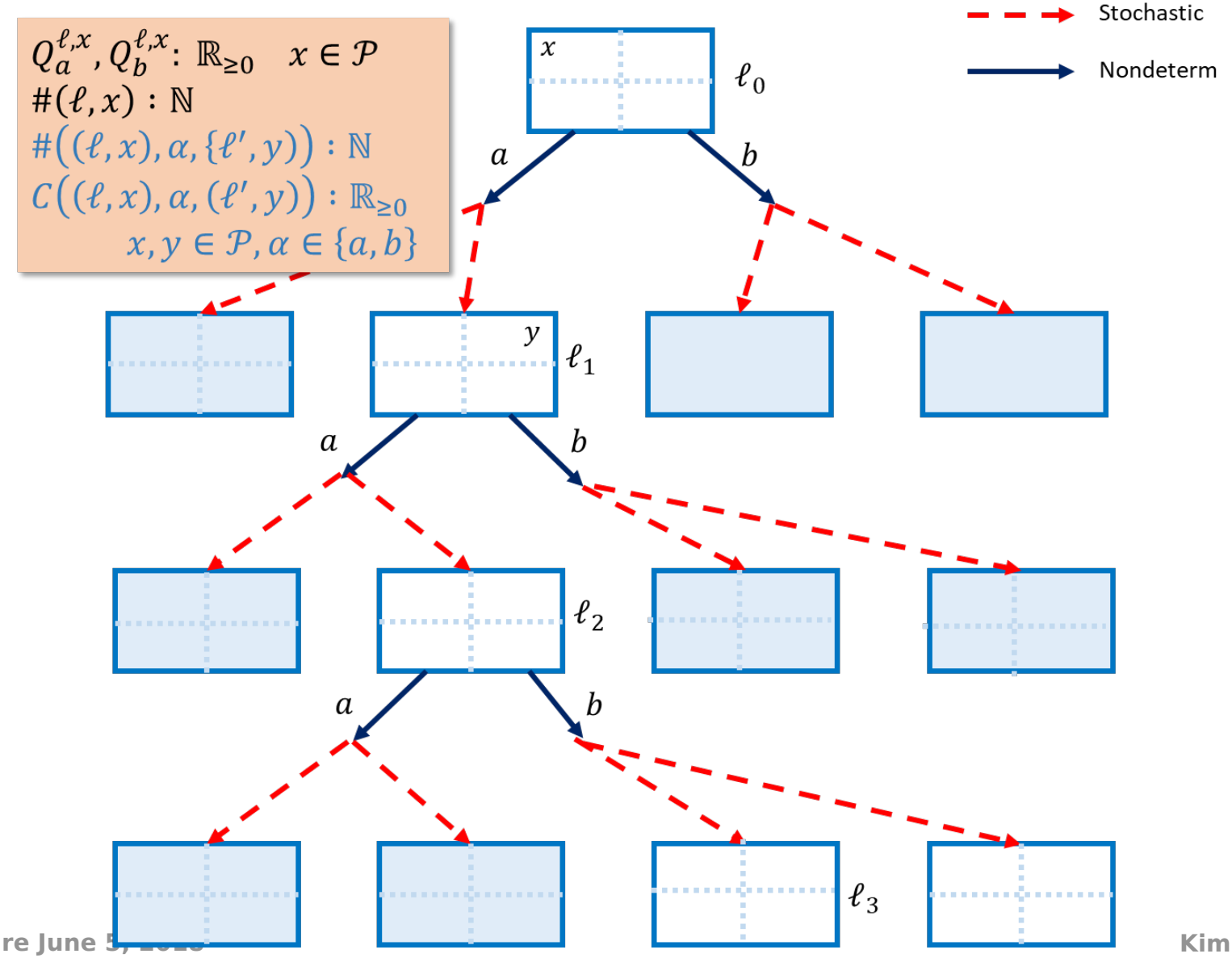
$Q_a^{\ell,x}, Q_b^{\ell,x}: \mathbb{R}_{\geq 0} \quad x \in \mathcal{P}$

$\#(\ell, x) : \mathbb{N}$

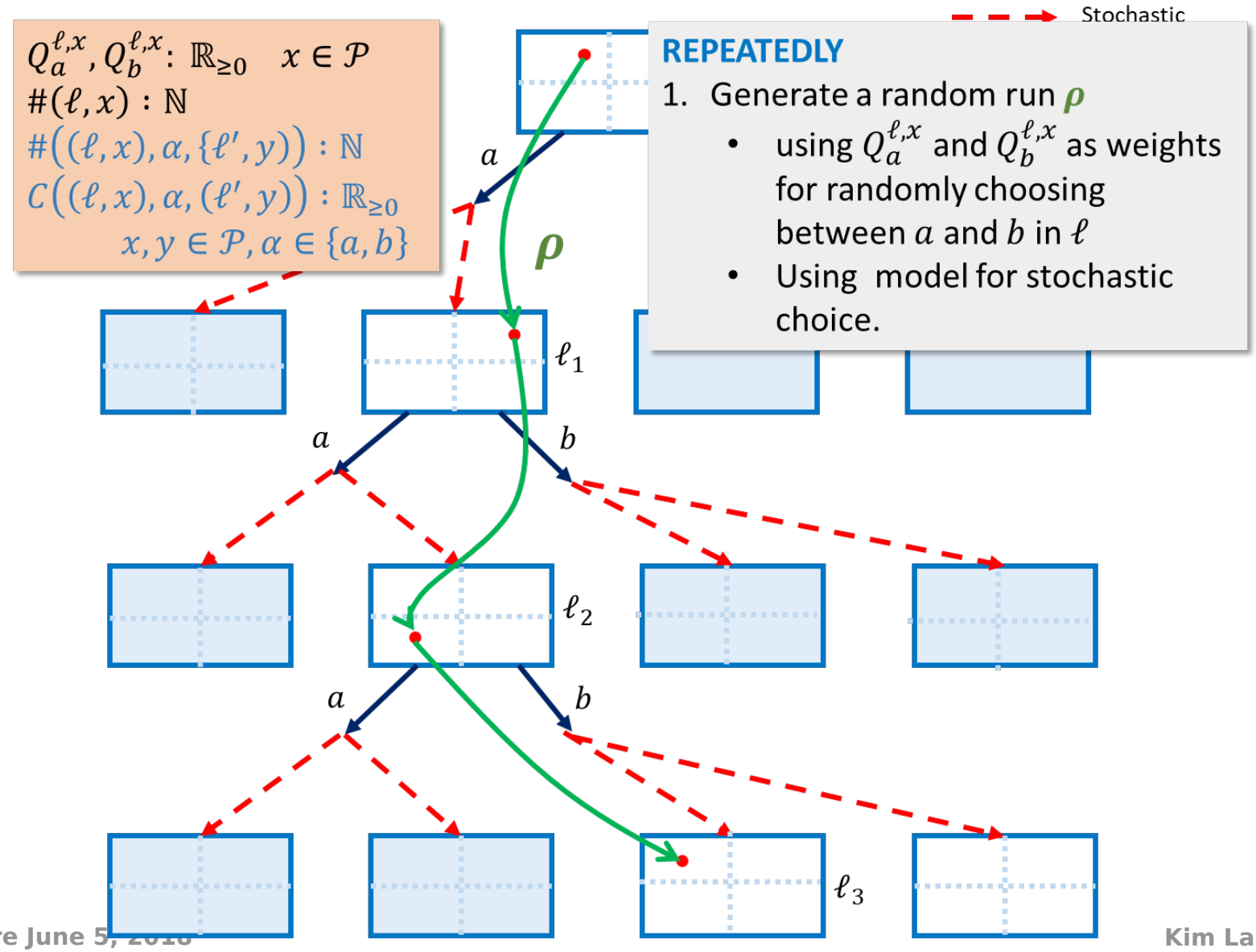
$\#\big((\ell, x), \alpha, \{\ell', y)\big) : \mathbb{N}$

$C\big((\ell, x), \alpha, (\ell', y)\big) : \mathbb{R}_{\geq 0}$

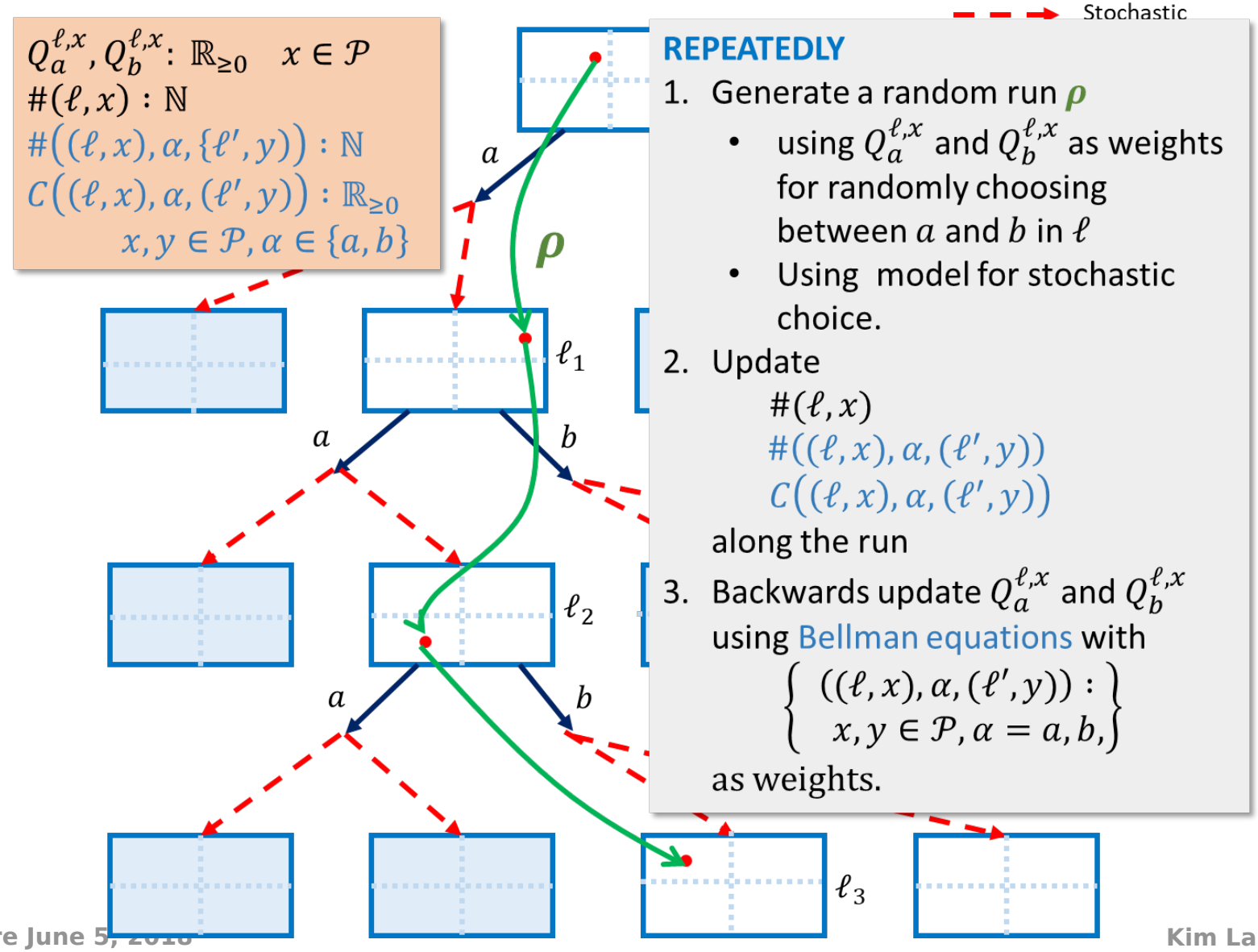
$x, y \in \mathcal{P}, \alpha \in \{a, b\}$



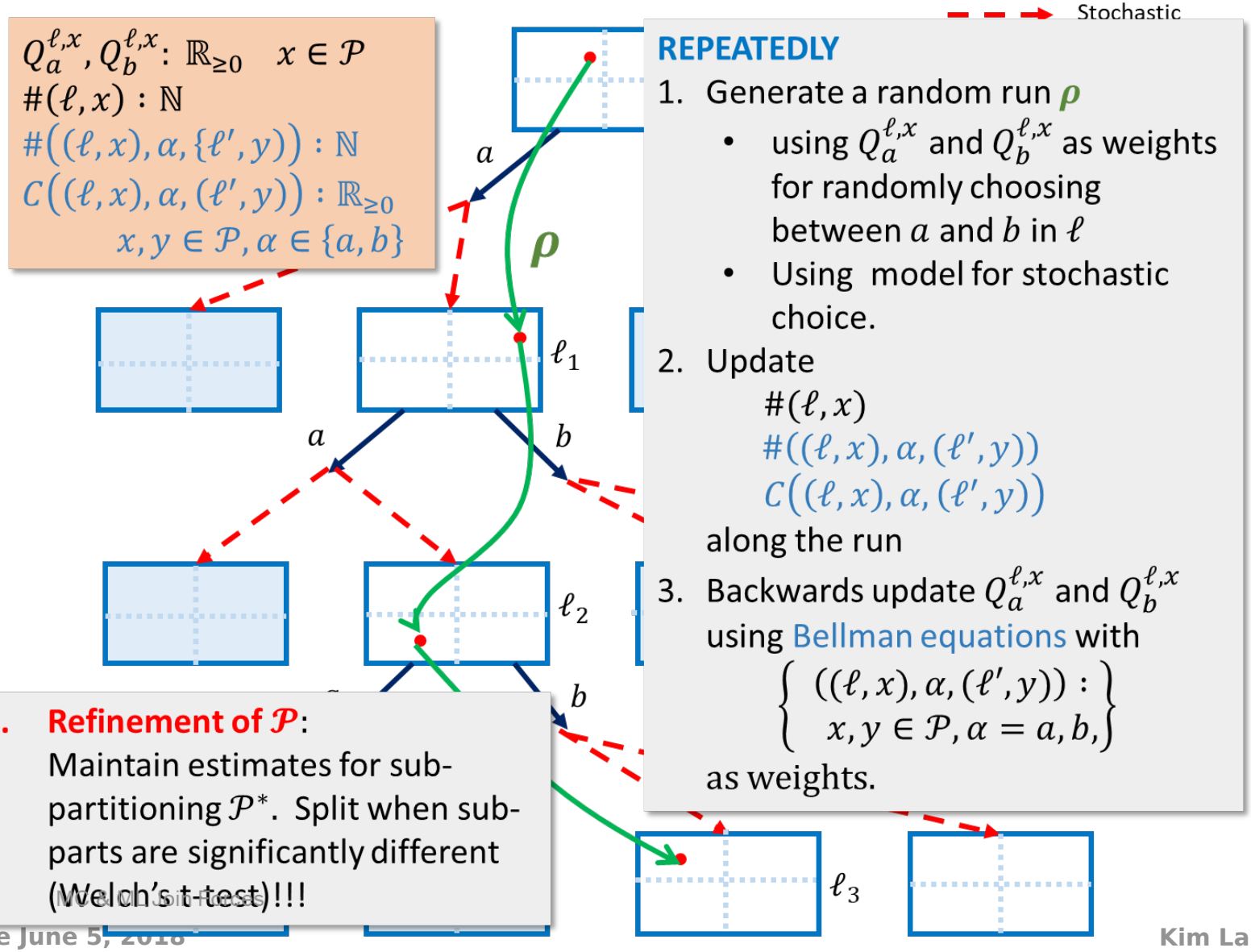
Q- & M-Learning


$Q_a^{\ell,x}, Q_b^{\ell,x}: \mathbb{R}_{\geq 0} \quad x \in \mathcal{P}$
$\#(\ell, x) : \mathbb{N}$
$\#\big((\ell, x), \alpha, \{\ell', y)\big) : \mathbb{N}$
$C\big((\ell, x), \alpha, (\ell', y)\big) : \mathbb{R}_{\geq 0}$
$x, y \in \mathcal{P}, \alpha \in \{a, b\}$
Stochastic
REPEATEDLY
1. Generate a random run ρ
• using $Q_a^{\ell,x}$ and $Q_b^{\ell,x}$ as weights for randomly choosing between a and b in ℓ
• Using model for stochastic choice.
ρ
a
b
ℓ_1
ℓ_2
ℓ_3

Q- & M-Learning


$Q_a^{\ell,x}, Q_b^{\ell,x}: \mathbb{R}_{\geq 0} \quad x \in \mathcal{P}$
$\#(\ell, x) : \mathbb{N}$
$\#\big((\ell, x), \alpha, \{\ell', y)\big) : \mathbb{N}$
$C\big((\ell, x), \alpha, (\ell', y)\big) : \mathbb{R}_{\geq 0}$
$x, y \in \mathcal{P}, \alpha \in \{a, b\}$
Stochastic
REPEATEDLY
1. Generate a random run ρ
   - using $Q_a^{\ell,x}$ and $Q_b^{\ell,x}$ as weights for randomly choosing between a and b in ℓ
   - Using model for stochastic choice.
2. Update
   $\#(\ell, x)$
   $\#((\ell, x), \alpha, (\ell', y))$
   $C\big((\ell, x), \alpha, (\ell', y)\big)$
   along the run
3. Backwards update $Q_a^{\ell,x}$ and $Q_b^{\ell,x}$ using Bellman equations with
   $\{ ((\ell, x), \alpha, (\ell', y)) : x, y \in \mathcal{P}, \alpha = a, b, \}$
   as weights.
a
b
ρ
ℓ_1
ℓ_2
ℓ_3

Q- & M-Learning


Stochastic
$Q_a^{\ell,x}, Q_b^{\ell,x}$: $\mathbb{R}_{\geq 0}$ $x \in \mathcal{P}$
$\#(\ell, x) : \mathbb{N}$
$\#\big((\ell, x), \alpha, \{\ell', y)\big) : \mathbb{N}$
$C\big((\ell, x), \alpha, (\ell', y)\big) : \mathbb{R}_{\geq 0}$
$x, y \in \mathcal{P}, \alpha \in \{a, b\}$
REPEATEDLY
1. Generate a random run ρ
   - using $Q_a^{\ell,x}$ and $Q_b^{\ell,x}$ as weights for randomly choosing between a and b in ℓ
   - Using model for stochastic choice.
2. Update
   $\#(\ell, x)$
   $\#((\ell, x), \alpha, (\ell', y))$
   $C\big((\ell, x), \alpha, (\ell', y)\big)$
   along the run
3. Backwards update $Q_a^{\ell,x}$ and $Q_b^{\ell,x}$ using Bellman equations with
   $\left\{ \begin{array}{l} ((\ell, x), \alpha, (\ell', y)) : \\ x, y \in \mathcal{P}, \alpha = a, b, \end{array} \right\}$
   as weights.
a
ρ
ℓ_1
a
b
ℓ_2
b
ℓ_3
5. **Refinement of $\mathcal{P}$**:
Maintain estimates for sub-partitioning $\mathcal{P}^*$. Split when sub-parts are significantly different (Welch's t-test)!!!

Intelligent Traffic Control

- Current time-triggered method
- Small example
  - Performance Evaluation
- SMC in UPPAAL

- Controller Synthesis for small example
- Compare performance

- Current status
  - Aarhus Kommune, Aalborg Kommune, Pune,
  - +Bus (request from municipality)
  - Publishing at Traffic venues.
  - UPPAAL Stratego on Microcontroller communicating with Radar

Stochastic Hybrid Automata

- The ball and the stochastic players example.
- Want to synthesize
- Show problem.
- Indicate the new method
- Do synthesis (here we show the various features e.g. partial observability)

Safety for Hybrid Games

- Recall the Adaptive Cruise Control
- Problem with discretization
- Euler (show what it gives)

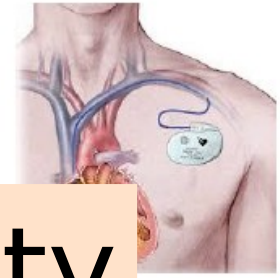
Cyber Physical Syste

OPTIMAL:
finding a **controller** for a given system such that a certain optimality criterion is achieved.

Discrete

Real Time

Resources



DEPENDABLE:
the ability of a **controller** to function (correct) under stated conditions for a specified period of time.

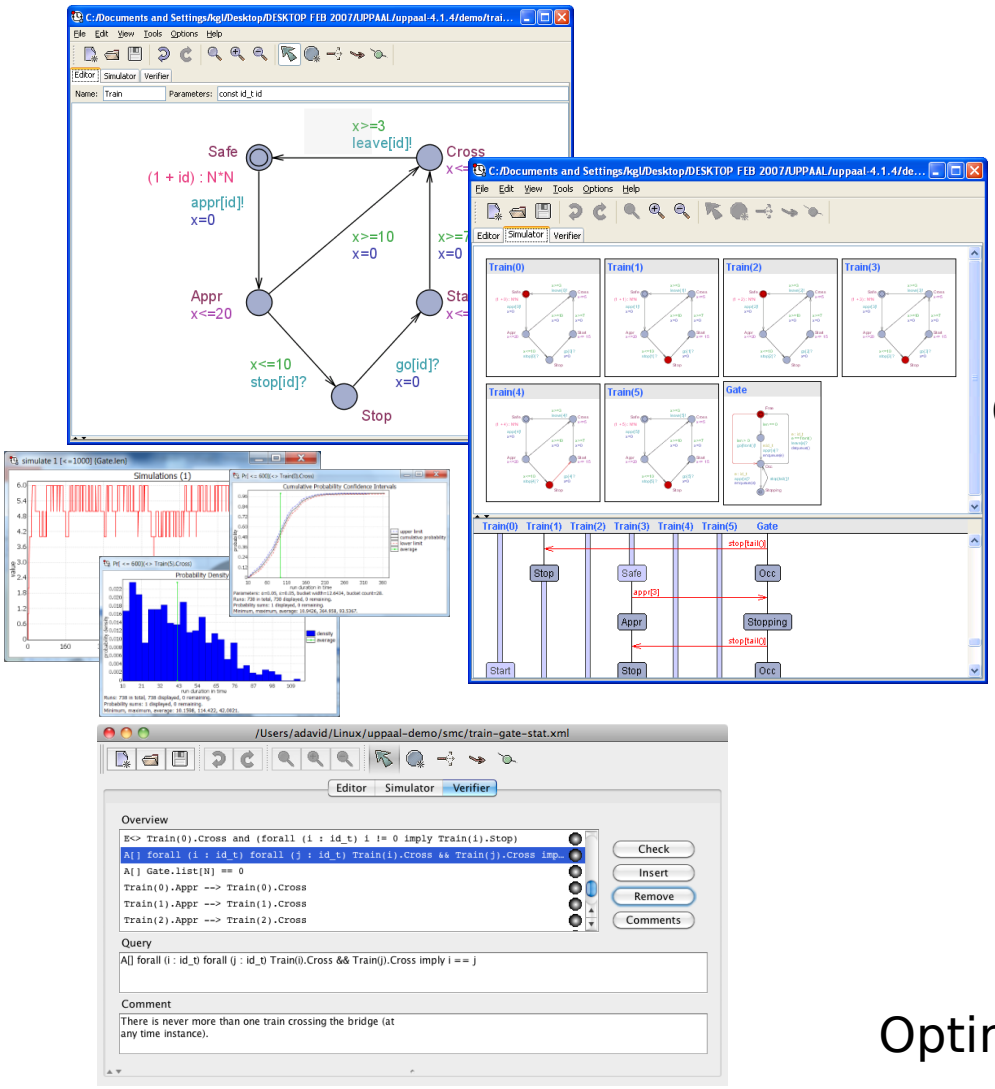
Stochasticity

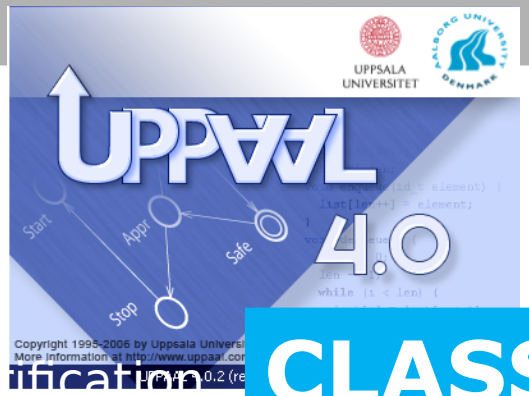
IoT

Cyber-Physical Systems

Hybrid

UPPAAL Tool Suite





1995

Verification — CLASSIC

Optimization — CORA — 2001

Testing — TRON — 2004

Synthesis — TIGA — 2005

Component — ECDAR — 2010

Performance Analysis — SMC — 2011, 2014

Optimal Synthesis — STRATEGO — 2018

Overview

- Timed Automata
  - Model Checking
- Stochastic Timed/Hybrid Automata
  - Statistical Model Checking
- Stochastic Timed/Hybrid Games
  - Synthesis
  - Reinforcement Learning
- Cases
  - Floor Heating
  - Adaptive Cruise Control
  - Intelligent Traffic Control
- Better Schedules
  - -Q- & M-Learning

CLASSIC

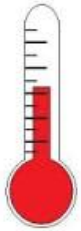
SMC

STRATEGO

STRATEGO 2.0

Verification & Performance Analysis

Stochastic Timed & Hybrid Automata

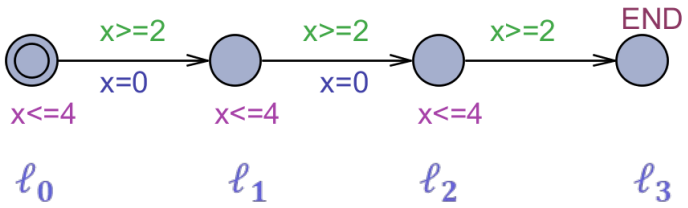


FORMATS11+12, CAV11,
RV12, HSB12, QAPL12,
NaSA12+13,SCIENCE CH13, STTT15

SMC

AALBORG UNIVERSITET

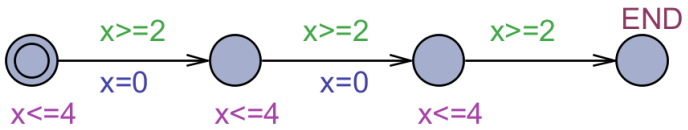
Timed Automata



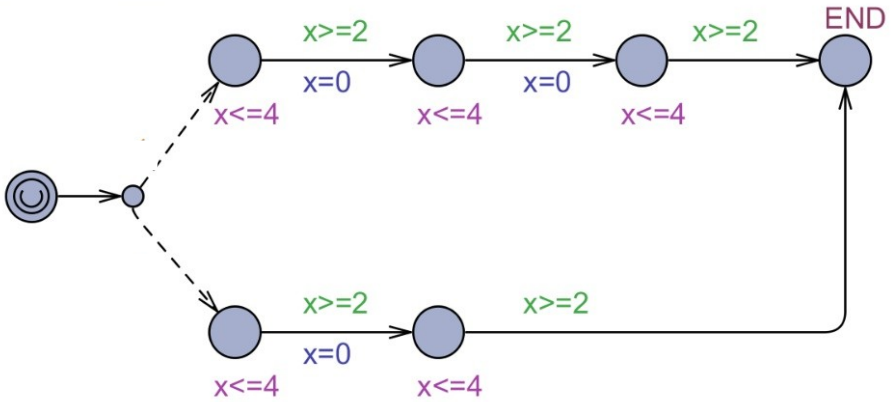
$[6,12]$

$$(\ell_0, x = 0) \xrightarrow{2.3} (\ell_0, x = 2.3) \rightarrow$$
$$(\ell_1, x = 0) \xrightarrow{3.4} (\ell_1, x = 3.4) \rightarrow$$
$$(\ell_2, x = 0) \xrightarrow{4.0} (\ell_2, x = 4.0) \rightarrow$$
$$(\ell_3, x = 0)$$

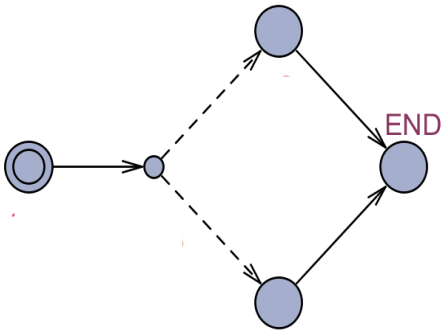
Timed Automata



[6,12]

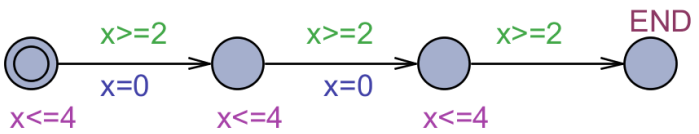


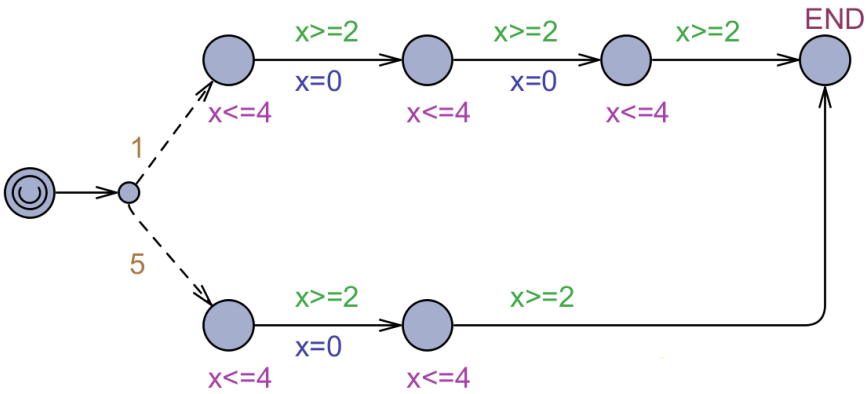
[4,12]

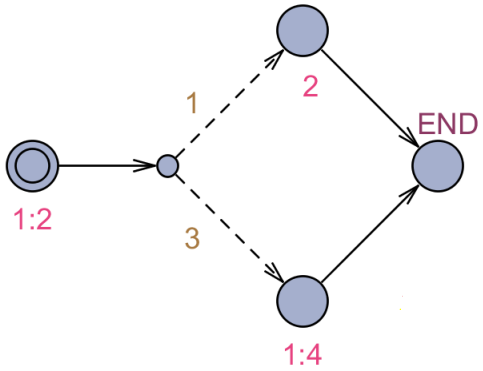


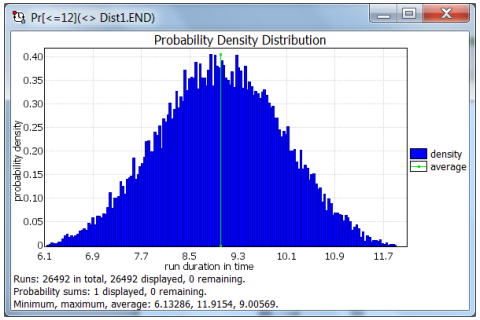
[0,∞]

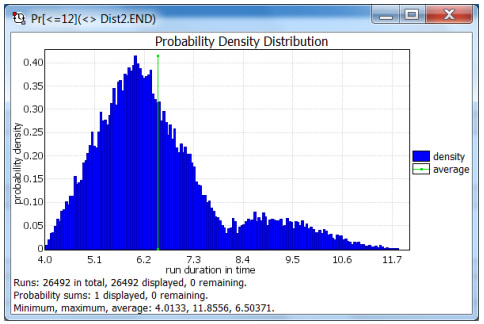
Stochastic Timed Automata

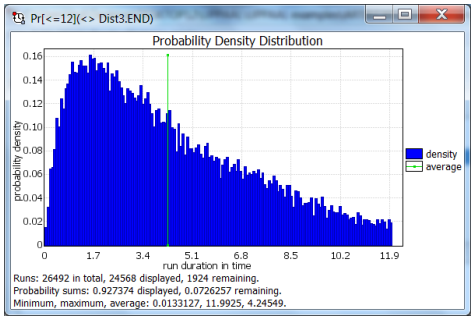




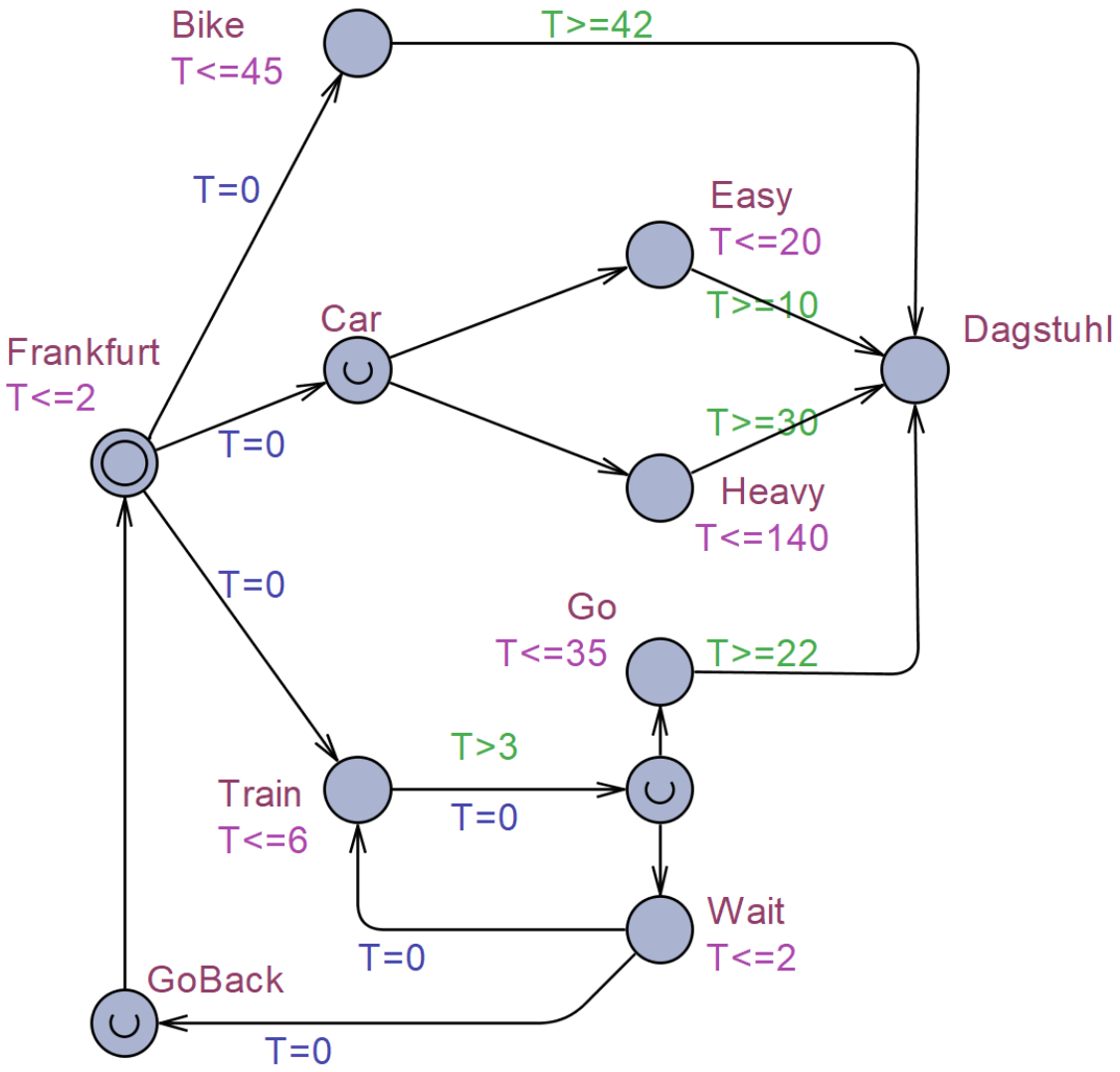




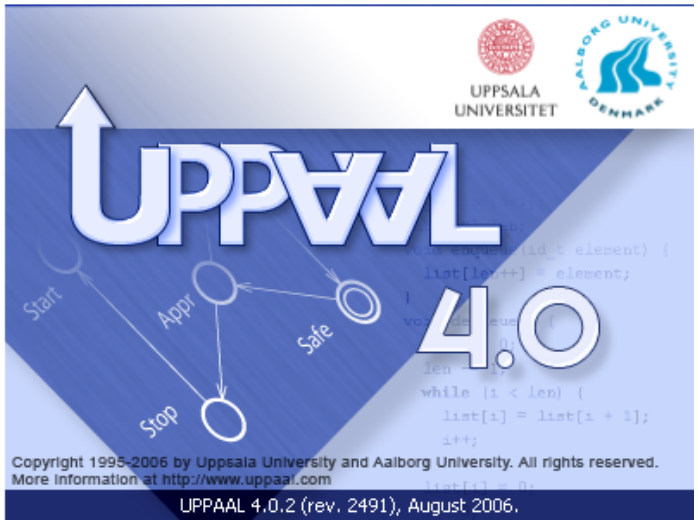




Going to Dagstuhl



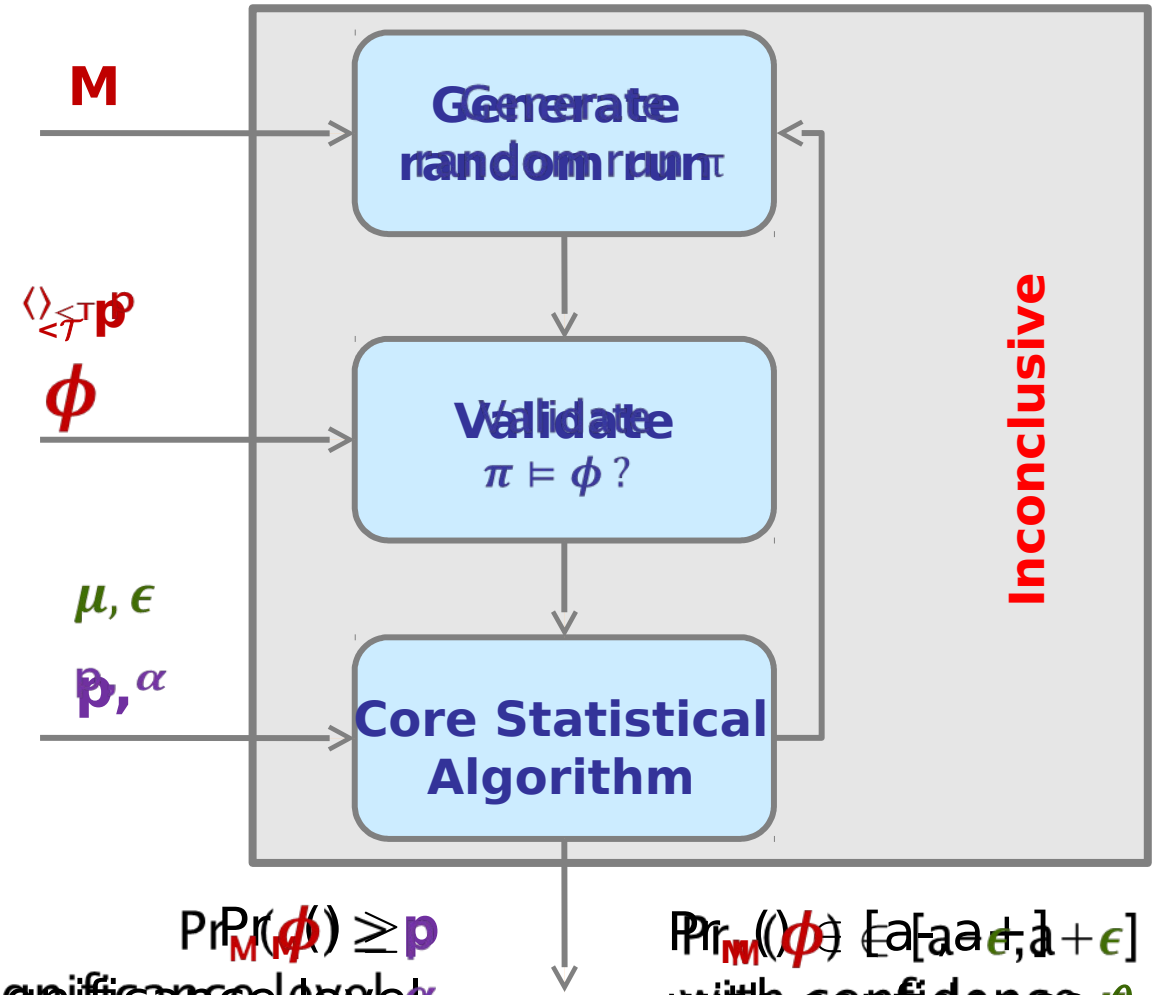
DEMO



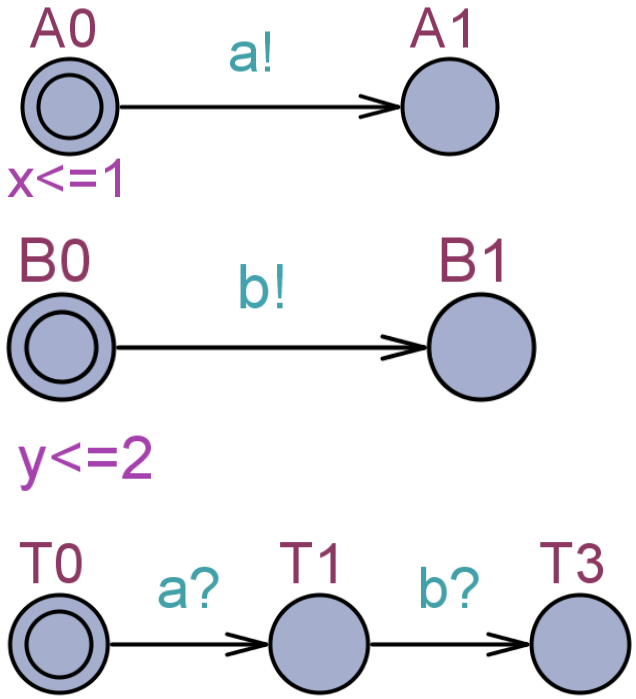
AALBORG UNIVERSITET

Statistical Model Checking

[FORMATS11, LPAR12, RV12]



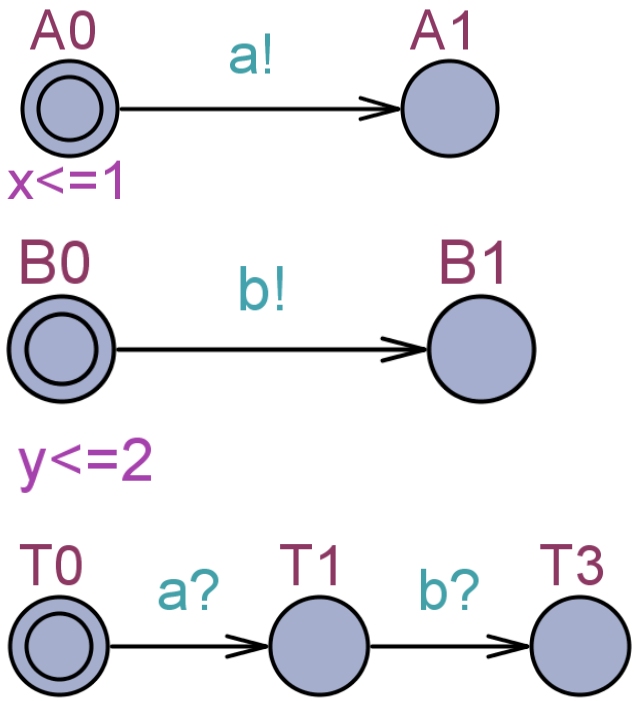
Composition of STA



E<> T.T3 & time <= 0.3

A<> T.T3 & time <= 2

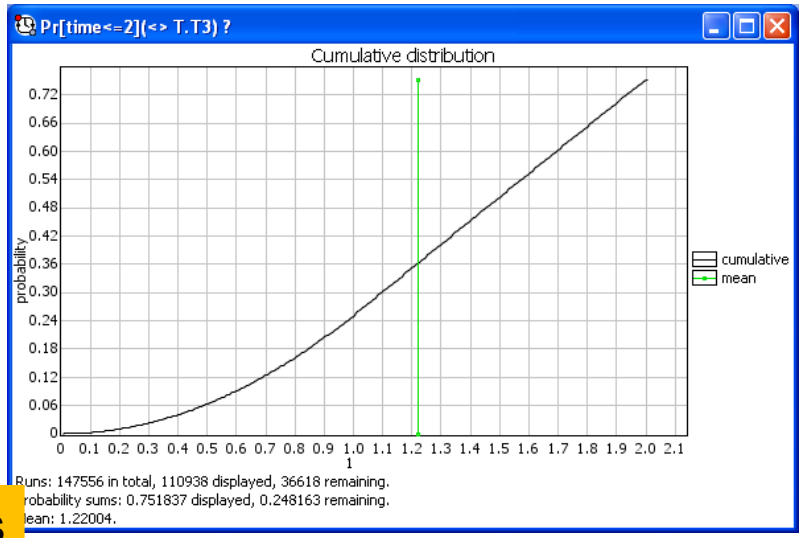
Composition of STA



Pr[time<=2](<> T.T3) ?

$$= \int_{t_a=0}^{1} 1 \cdot \int_{t_b=t_a}^{2} ½ \; dt_b \;\; dt_a = 3/4$$

Pr[time<=T](<> T.T3) ?

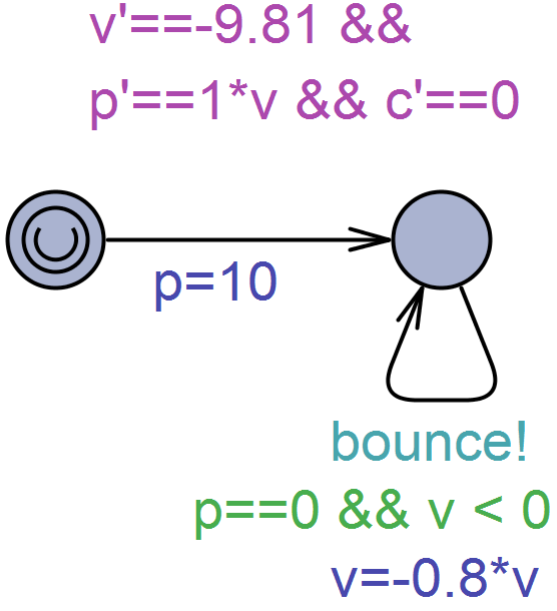


Composition = Race between components for outputting

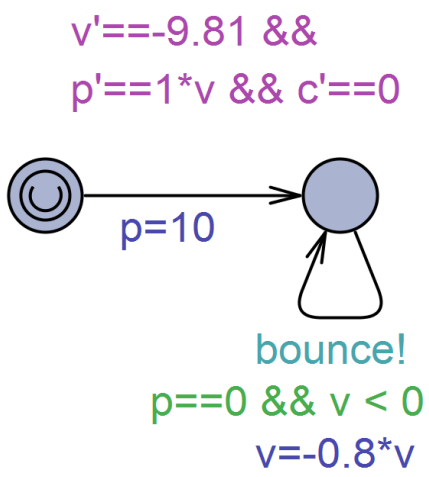
Hybrid Automata

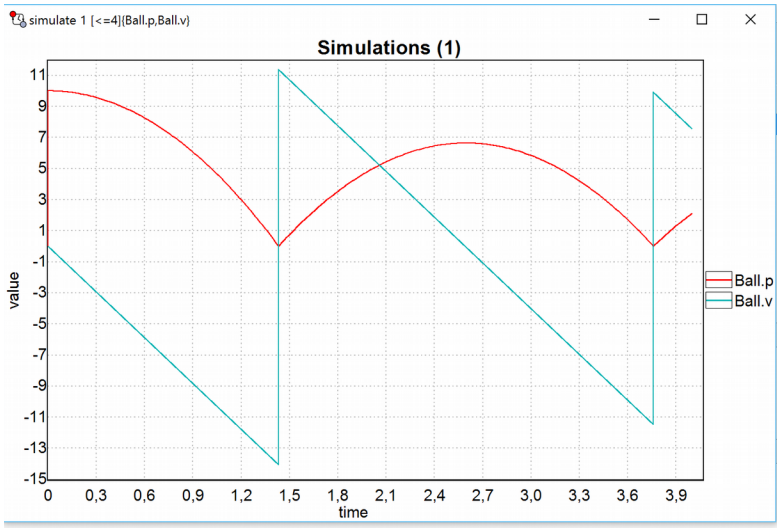
$H=(L, l_0, \Sigma, X, E, F, Inv)$
where

- L set of locations
- l_0 initial location
- $\Sigma=\Sigma_i \cup \Sigma_o$ set of actions
- X set of continuous variables
  valuation $\nu: X \rightarrow \mathbf{R}$ $(=R^X)$
- 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
  $F(l): R_{>0} \times R^X \rightarrow R^X$
- For each l an invariant
  $Inv(l) \subseteq R^X$



Hybrid Automata





Semantics

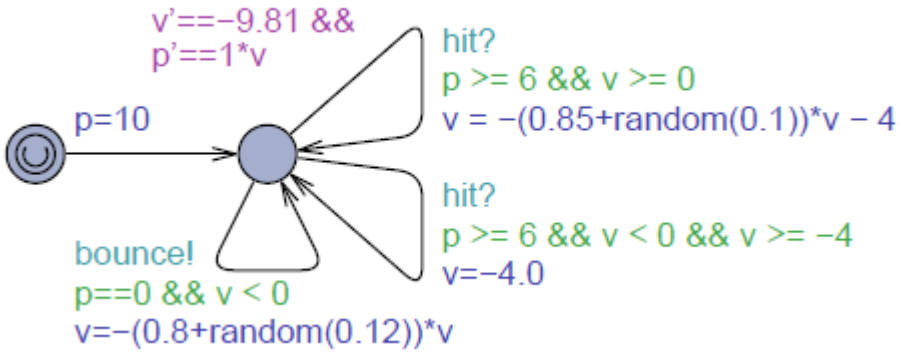
- States
  (l,ν) where $\nu \in R^X$
- Transitions
  $(l,\nu) \rightarrow_d (l,\nu')$ where
  $\nu' = F(l)(d)(\nu)$
  provided $\nu' \in Inv(l)$

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

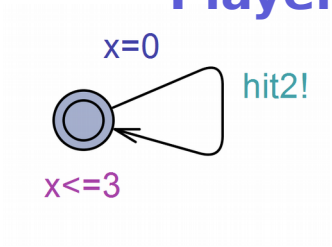
Networks of ..

Ball



Player 1

Player 2



Stochastic Semantics

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

Delay density function*

$$\mu_s : R_{>0} \rightarrow R$$

Output Probability Function

$$\gamma_s : \Sigma_o \rightarrow [0,1]$$

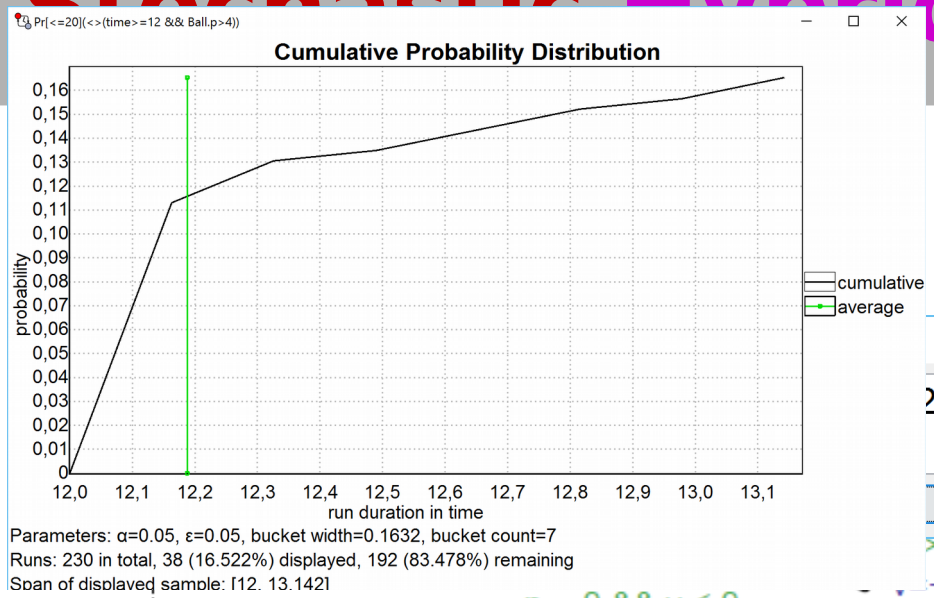
Next-state density function*

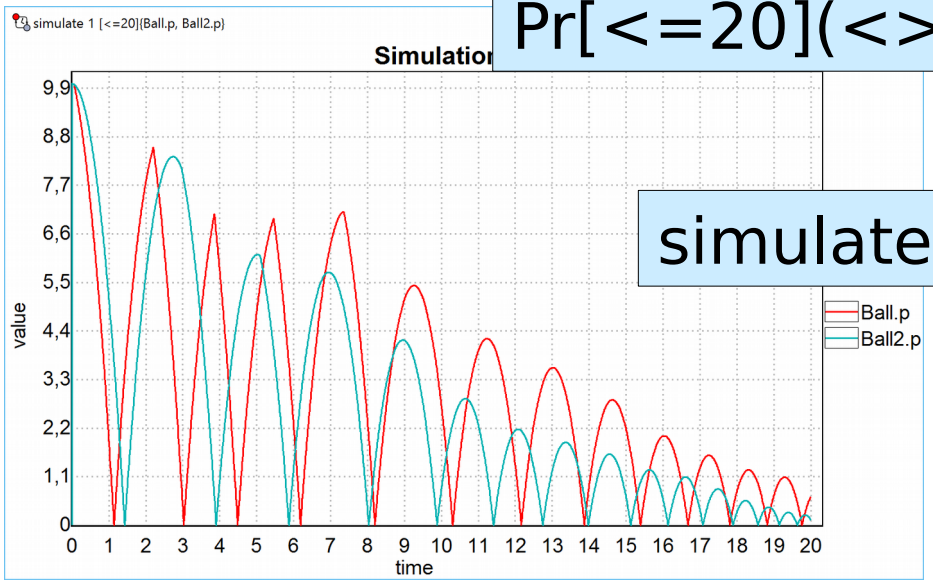
$$\eta_{a\ s} : St \rightarrow R$$

where $a \in \Sigma$.

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

Stochastic Hybrid Systems

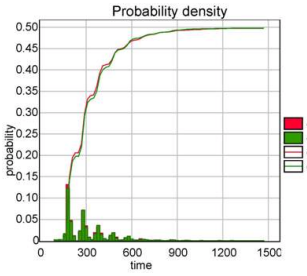




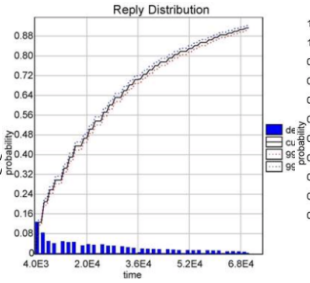
Pr[<=20](<>(time>=12 && Ball1.p>4))

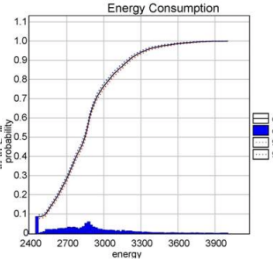
simulate 1 [<=20]{Ball1.p, Ball2.p}

Applications

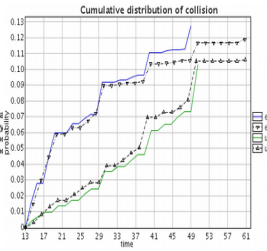


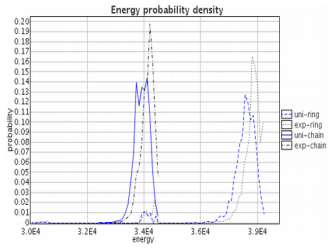
FIREWIRE



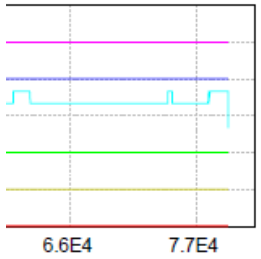


BLUETOOTH

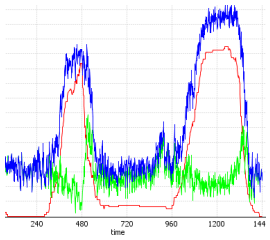




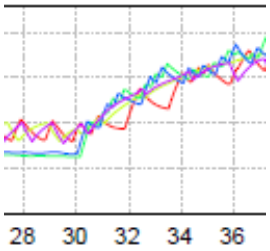
10 node LMAC



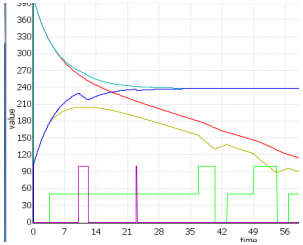
Schedulability Analysis for Mix Cr Sys



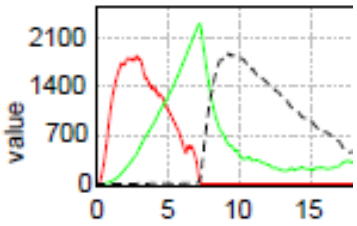
Smart Grid Demand / Response



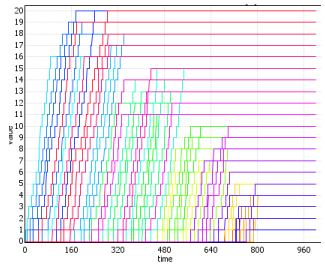
Energy Aware Buildings



Battery Scheduling



Genetic Oscilator (HBS)



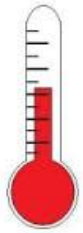
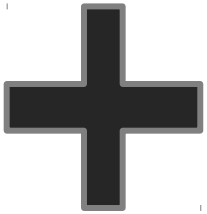
Passenger Seating in Aircraft

More

- Statistical MC of $\mathrm{WMLT}_{\leq}$
- Statistical MC of Liveness Pr
- Distributed UPPAAL
- Dynamic instantiati
  - Useful for modeling c
  - Quantified extension
- Support of FMI/FMU
  - Co-simulation with MA
- Support of ANOVA
  - Minimization of numbe
- Rare Event Methods (Im sampling)

Synthesis & Optimization

Stochastic Timed & Hybrid GAMES

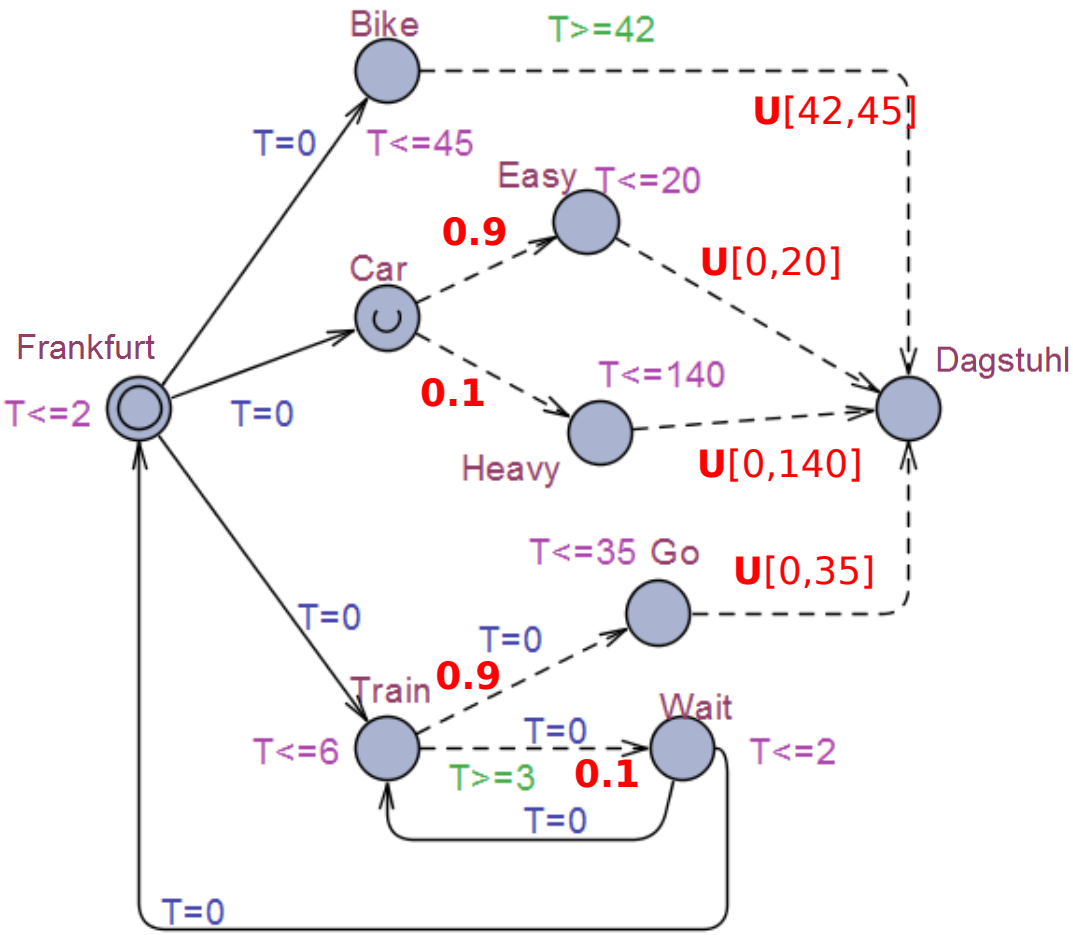

ATVA'14, TACAS'15, CSD'15, FORMATS'15,
ISOLA'16, SETTA'16, TACAS'16,
SOFSEM'17, TACAS'17, SPIN'17, CIAA'17

STRATEGO

AALBORG UNIVERSITY DENMARK

LUSS

Going to Dagstuhl – in 1 hour



Optimal WC Strategy
- (2-player)
- Take bike
- WC=45

Optimal Expected Strategy
- (1½ player)
- Take car
- E = 16
- WC = 140

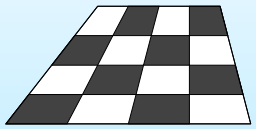
Optimal Expected Strategy guaranteeing WC<=60
- ?????

DEMO

Uppaal TIGA

strategy NS = control: A<> goal

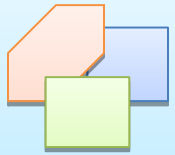
strategy NS = control: A[] safe



Uppaal

E<> error under NS

A[] safe under NS



G
Timed Game

φ

synthesis

σ
Strategy

G|σ
Timed Automata

abstraction

P
Stochastic Priced Timed Game

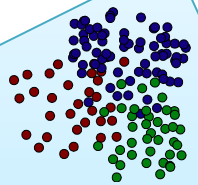
P|σ

minE(cost)

maxE(gain)

σ°
optimized Strategy

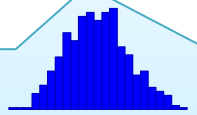
P|σ°
Stochastic Priced Timed Automata



Statistical Learning

strategy DS = minE (cost) [<=10]: <> done under NS

strategy DS = maxE (gain) [<=10]: <> done under NS



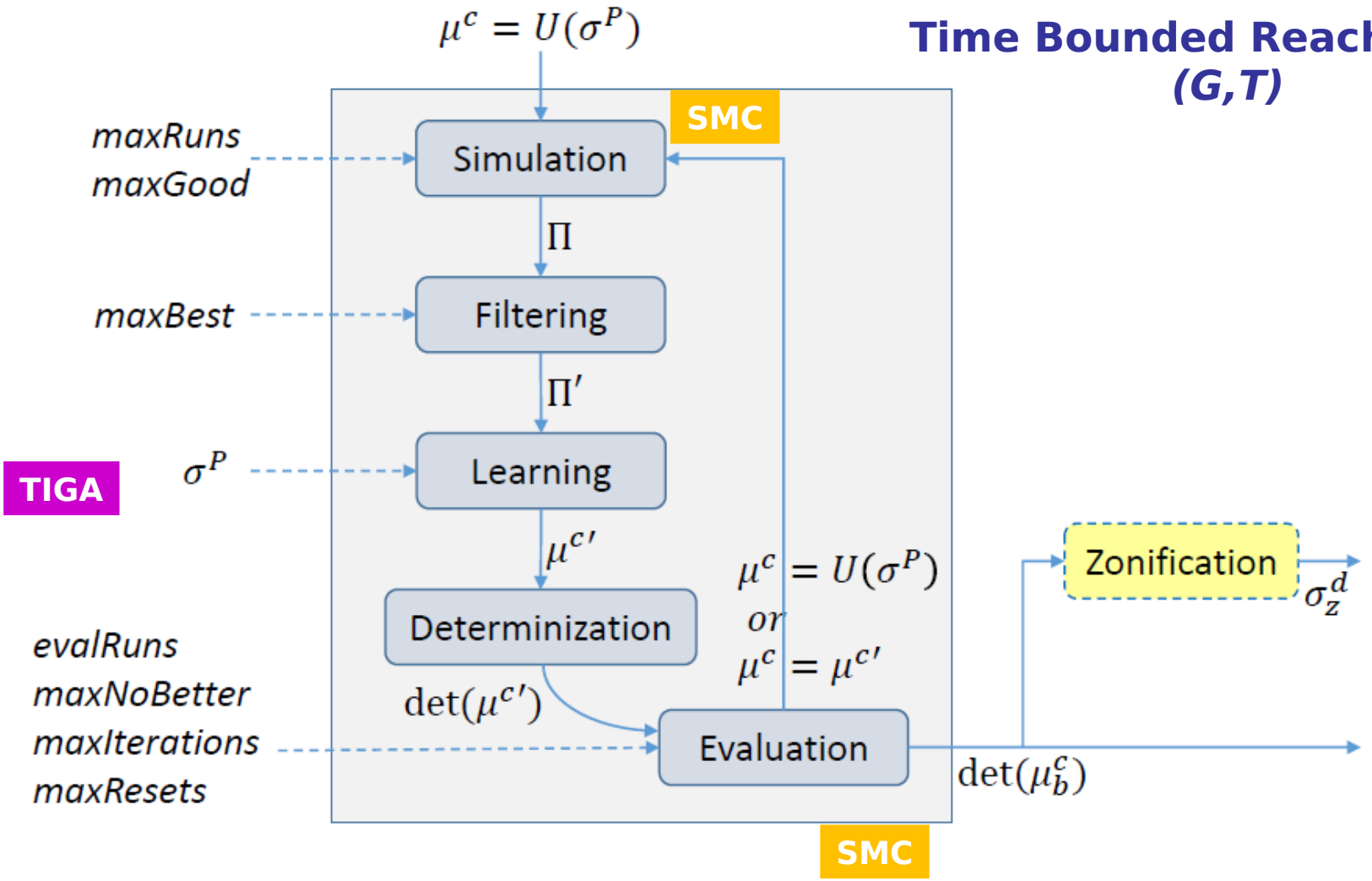
Uppaal SMC

simulate 5 [<=10]{e1, e2} under SS

Pr[<=10](<> error) under SS

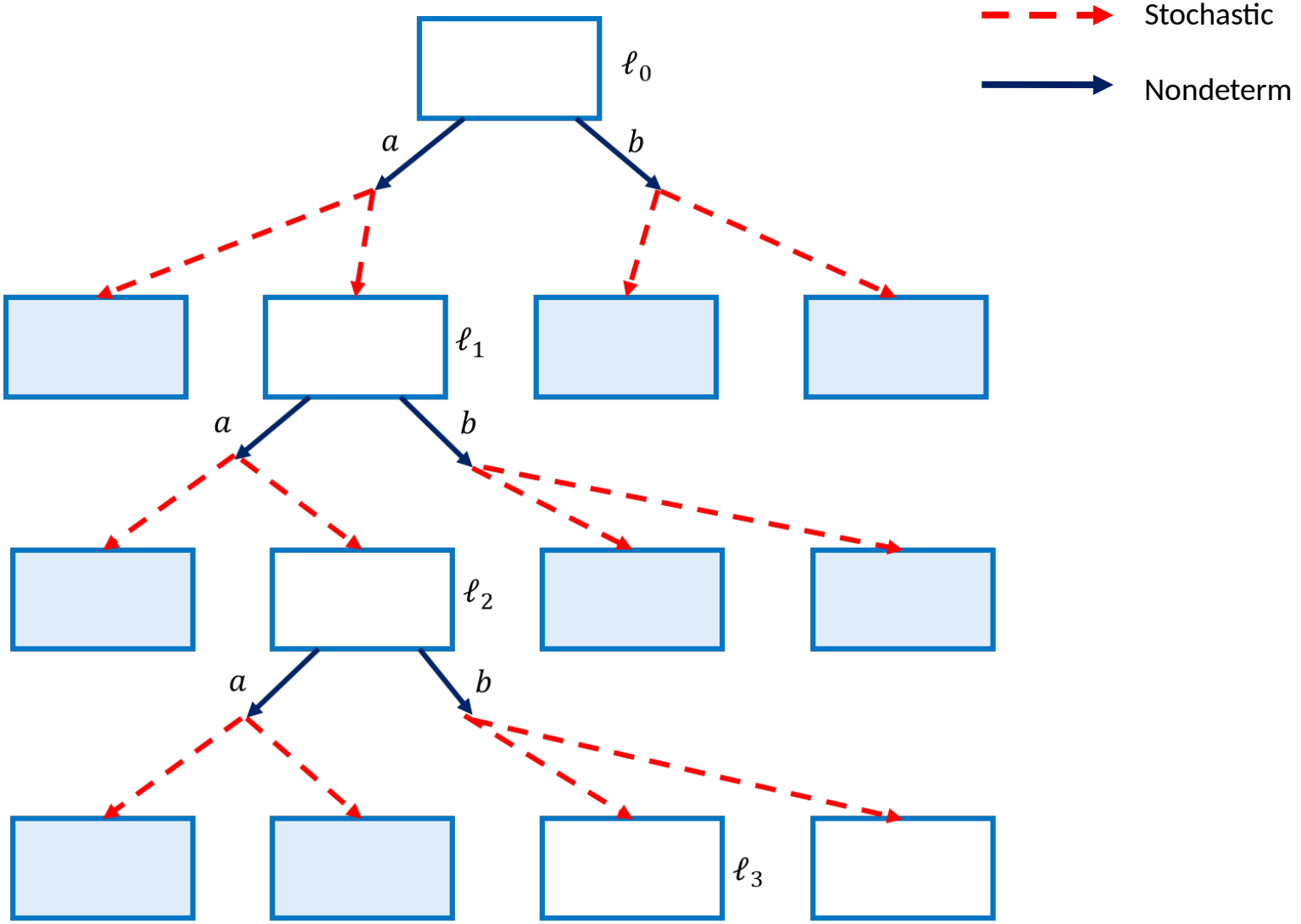
E[<=10;100](max: cost) under SS

Reinforcement Learning

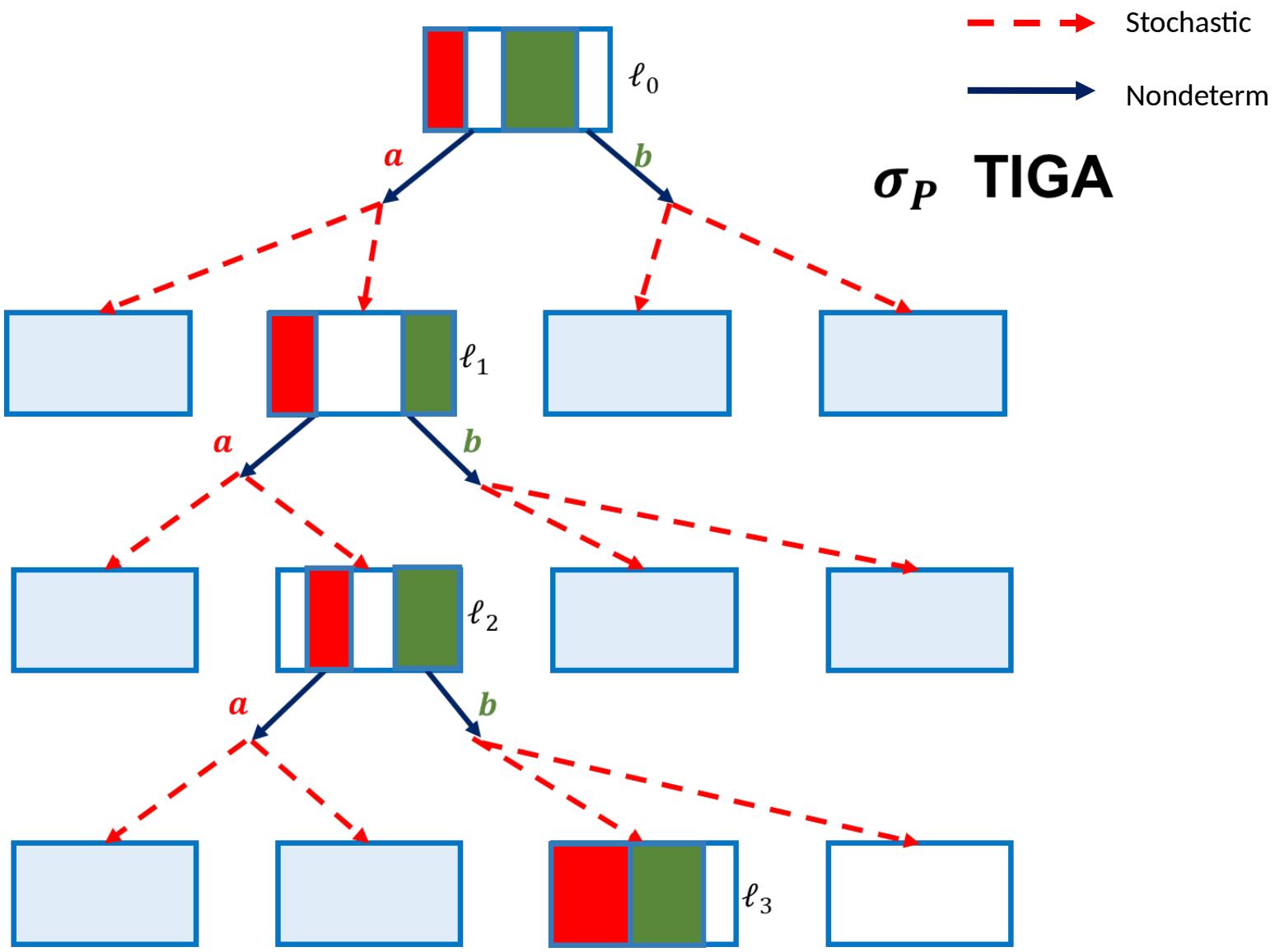


David, Jensen, Larsen, Legay, Lime, Sørensen, Taankvist:On Time with Minimal Expected Cost! ATVA 2014
Henriques, Martins, Zuliani, Platzer, Clarke: Statistical Model Checking for Markov Decision Processes. QEST 2012

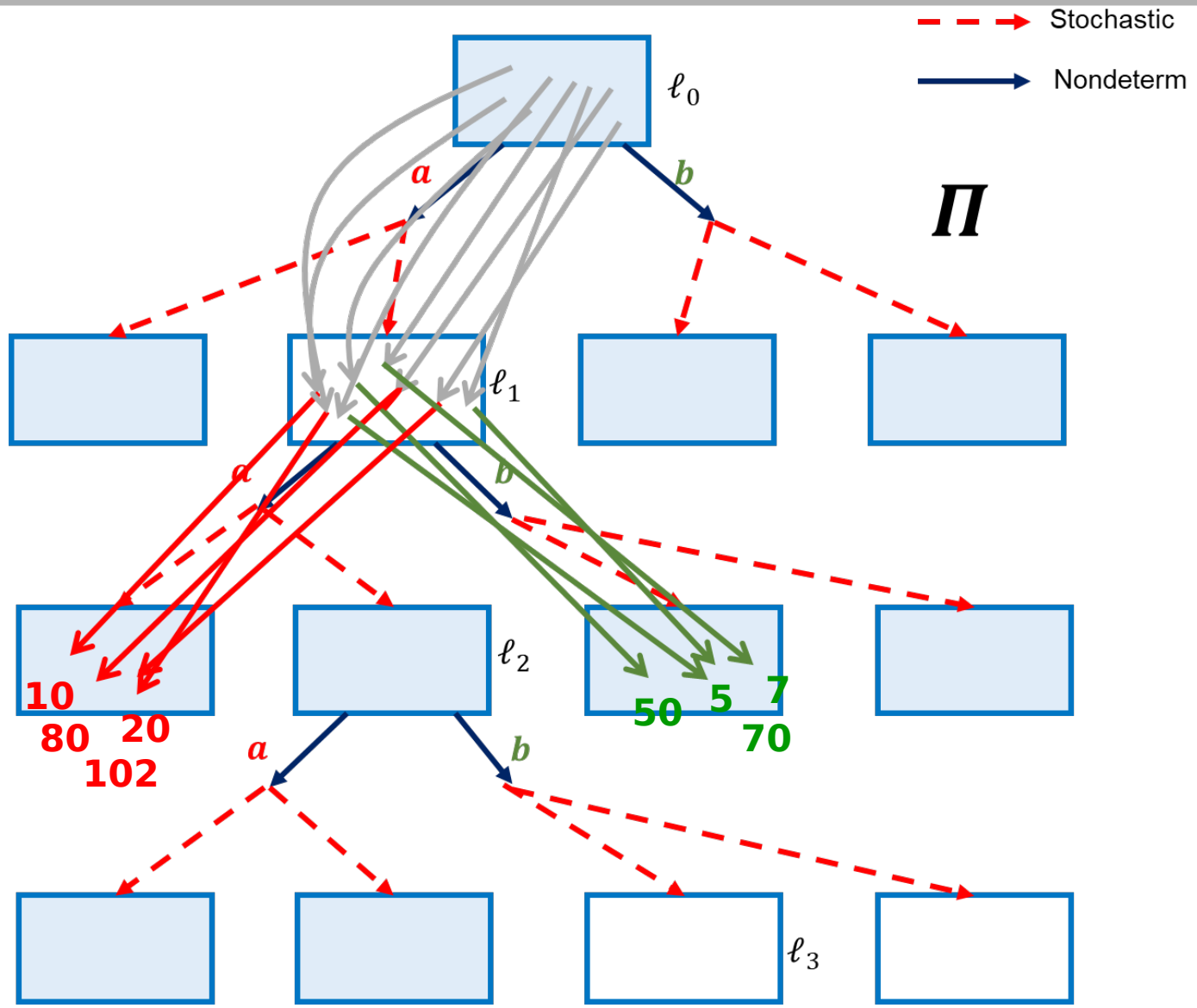
Reinforcement Learning



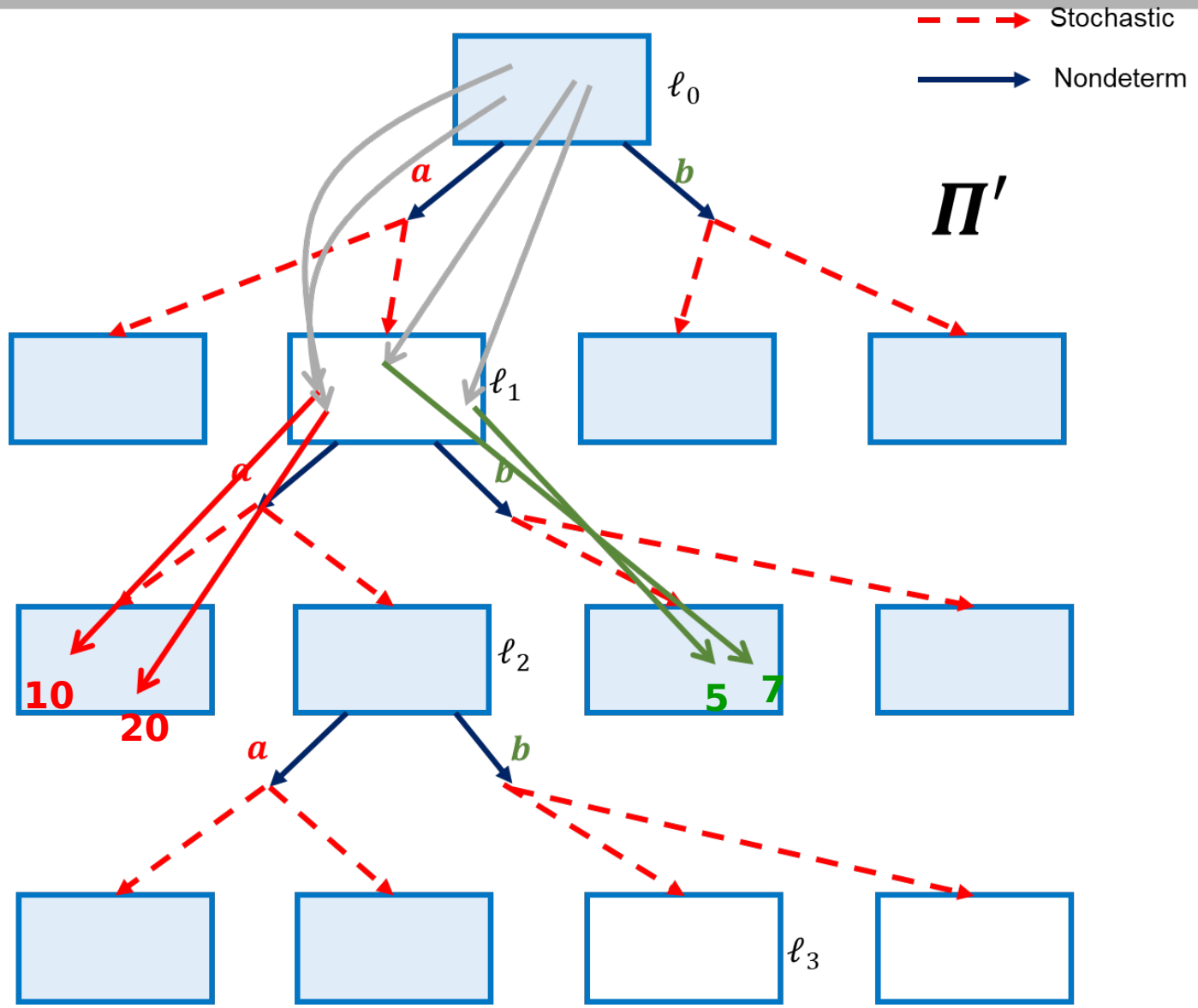
Most Permissive Safety



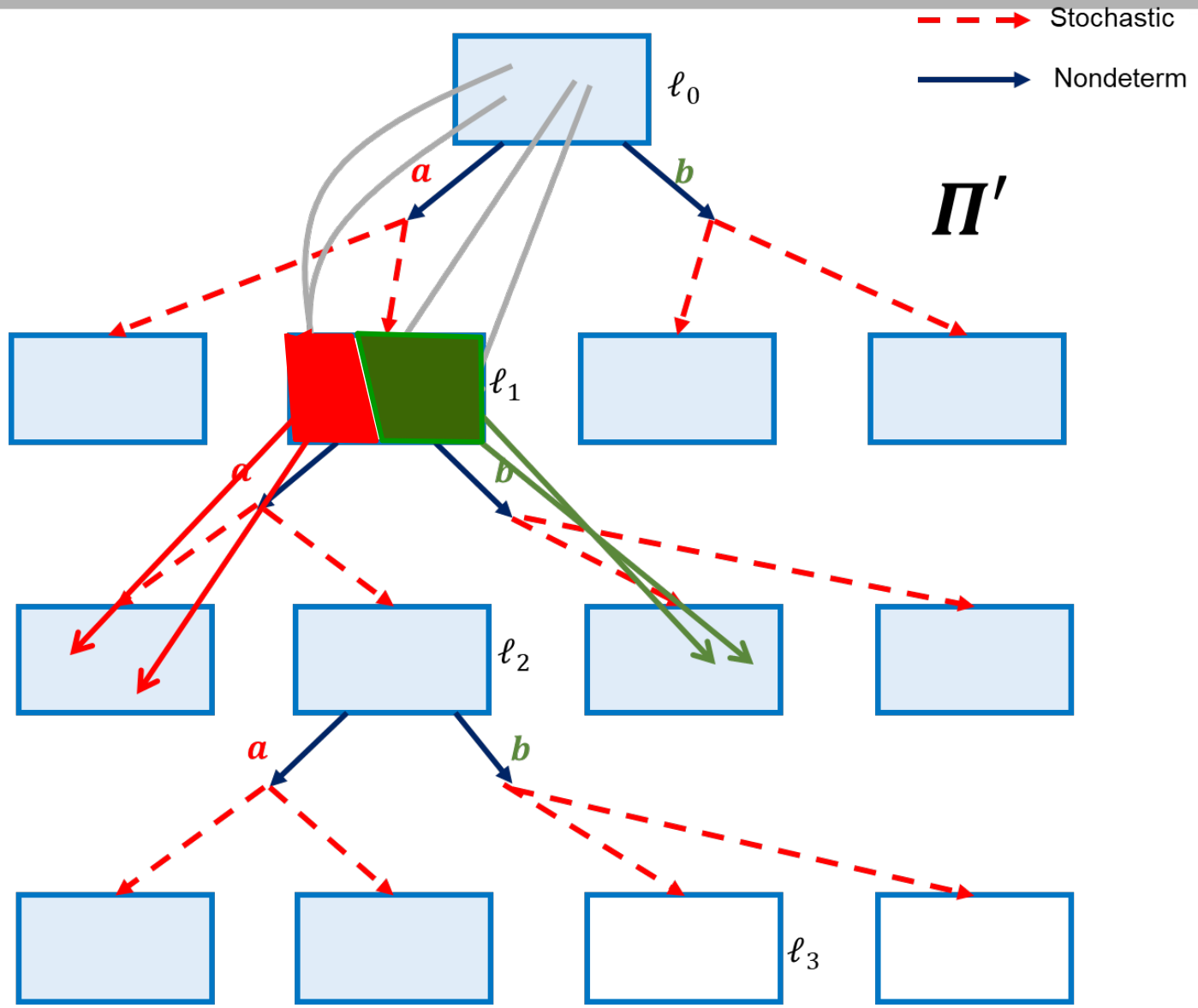
Random Runs


Stochastic
Nondeterm
ℓ_0
ℓ_1
ℓ_2
ℓ_3
a
b
Π
10
80
20
102
50
5
7
70

Best Random Runs



Strategy

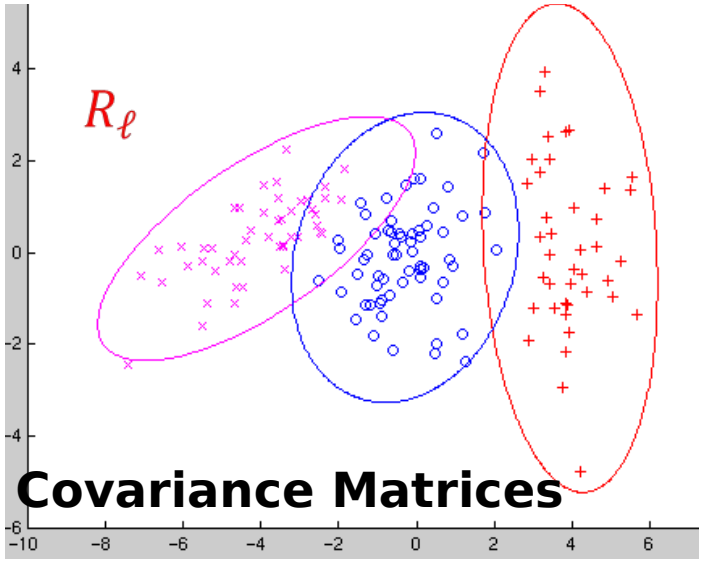


Strategies – Representation

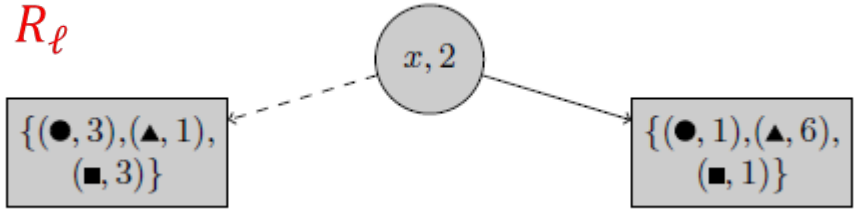
Nondeterministic Strategies $\sigma^{n}_{(\ell,\nu)} \subseteq (\Sigma_c \cup \{\lambda\})$

$$R_\ell = \{(Z_1, a_1), \ldots, (Z_k, a_k)\}, \text{ where } a_i \in \Sigma_c \cup \{\lambda\}.$$

Stochastic Strategies $\mu^{s}{}_{(\ell,\nu)} : (\Sigma_c \cup \{\lambda\}) \to [0,1]$



Covariance Matrices



Splitting

Logistic Regression

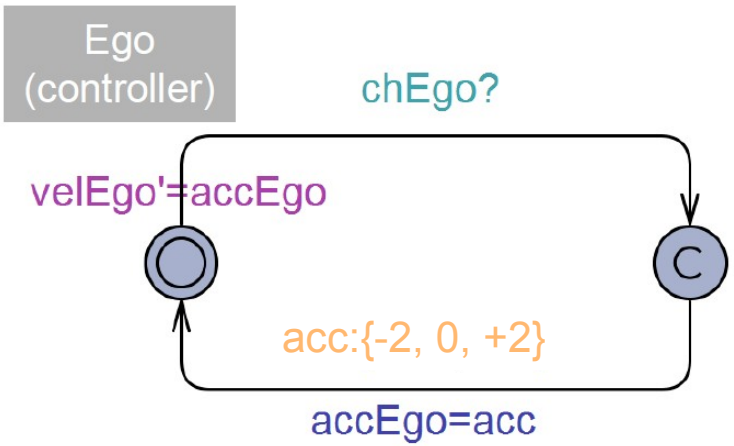
Application
Safe & Adaptive Cruise Control

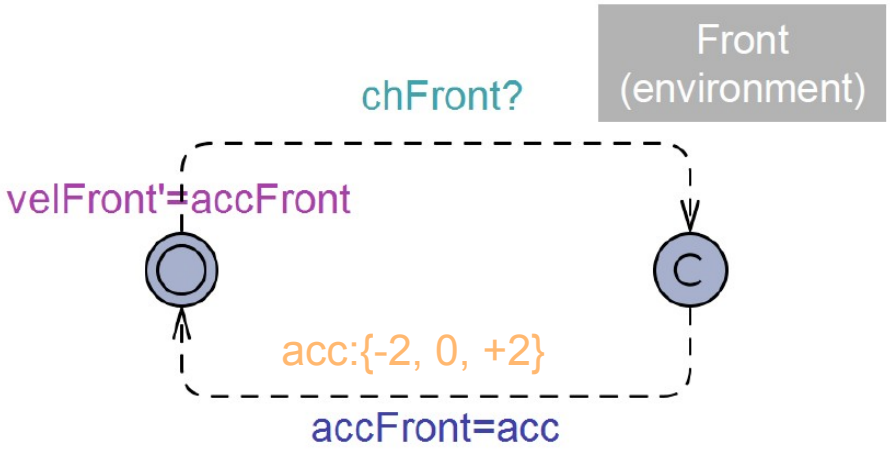
Q1: Find a safety **strategy** for ***Ego*** such no crash will ever occur no matter what ***Front*** is doing.

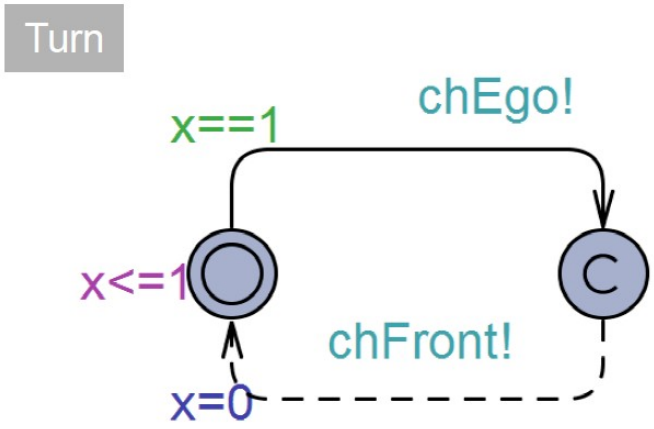
Q2: Find the **optimal sub-strategy** that will allow *Ego* to go as far as possible (without overtaking).

Larsen, Mikucionis, Taankvist: Safe and Optimal Adaptive Cruise Control. Correct System Design 2015

Two Player Game (simplified)







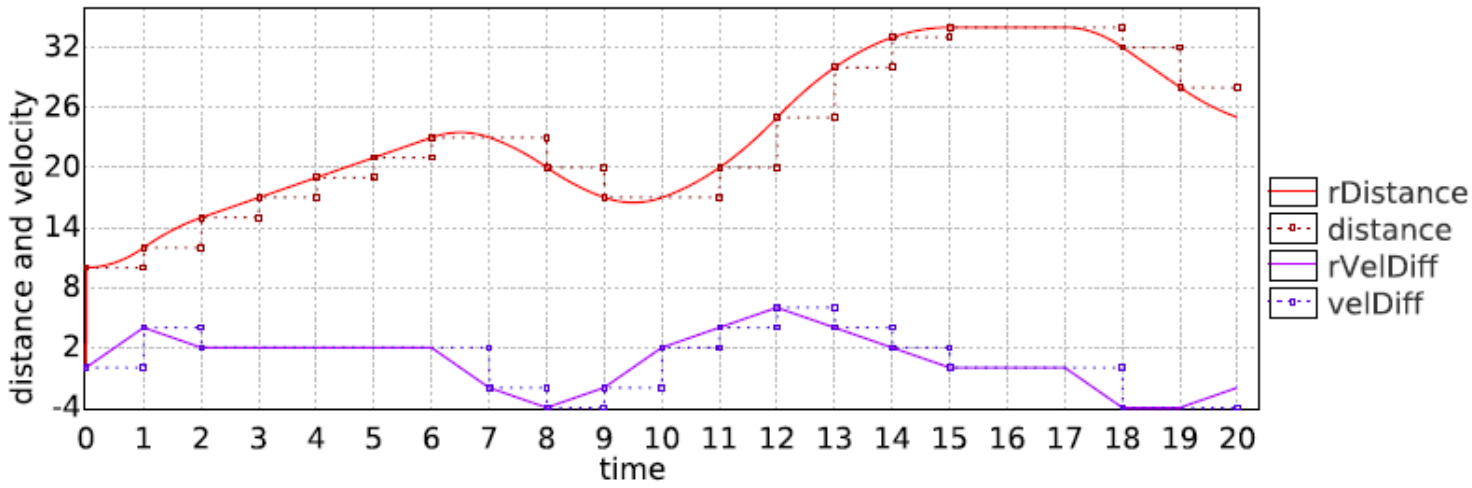
distance' == (velEgo-velFront) &&
D' == distance

Q: find strategy for Ego

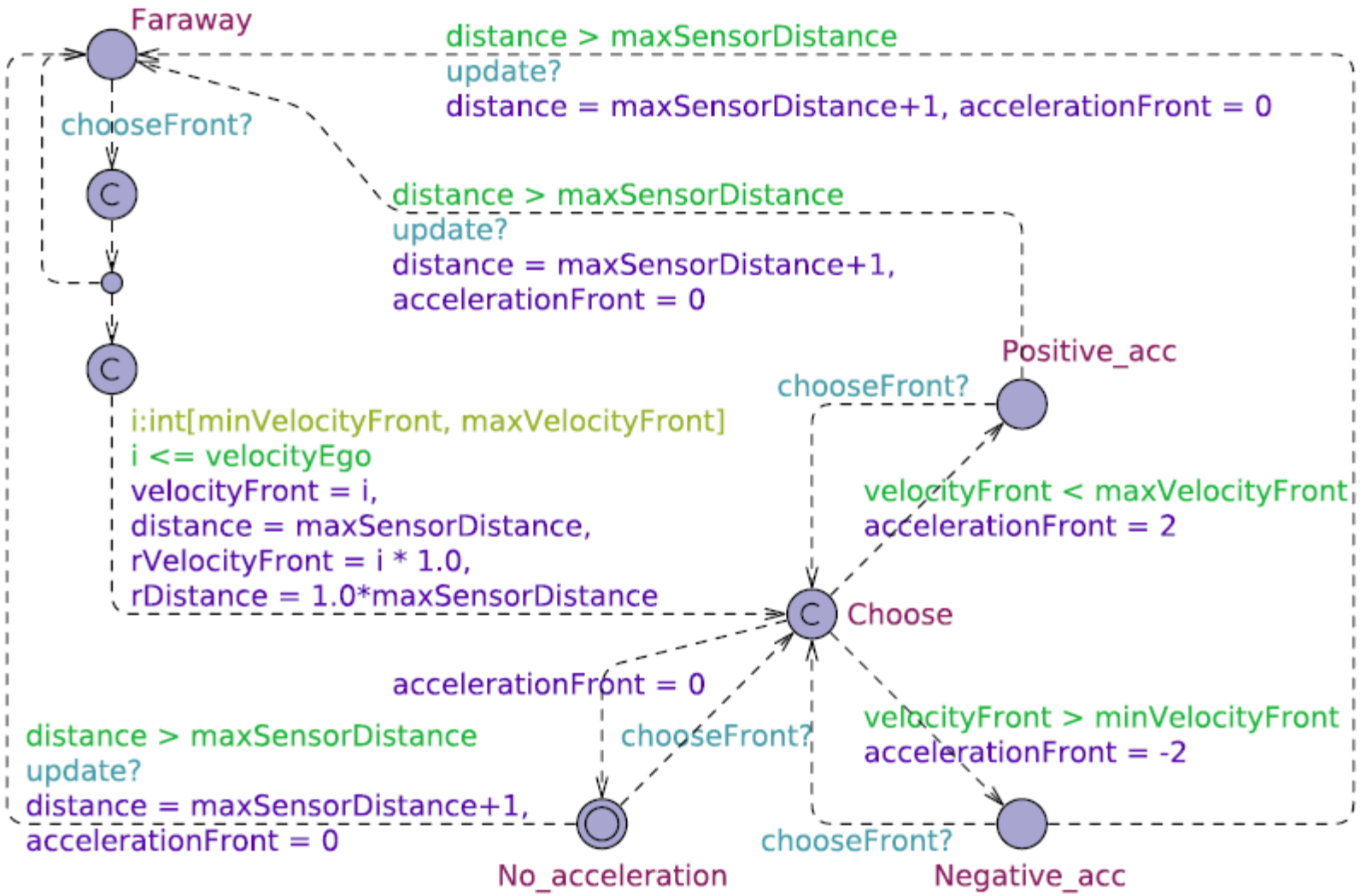
Discretization

Discrete

```
void updateDiscrete(){
    int oldVel, newVel;
    oldVel = velocityFront - velocityEgo;
    velocityEgo = velocityEgo + accelerationEgo;
    velocityFront = velocityFront + accelerationFront;
    newVel = velocityFront - velocityEgo;
    if (distance > maxSensorDistance) {
        distance = maxSensorDistance + 1;
```



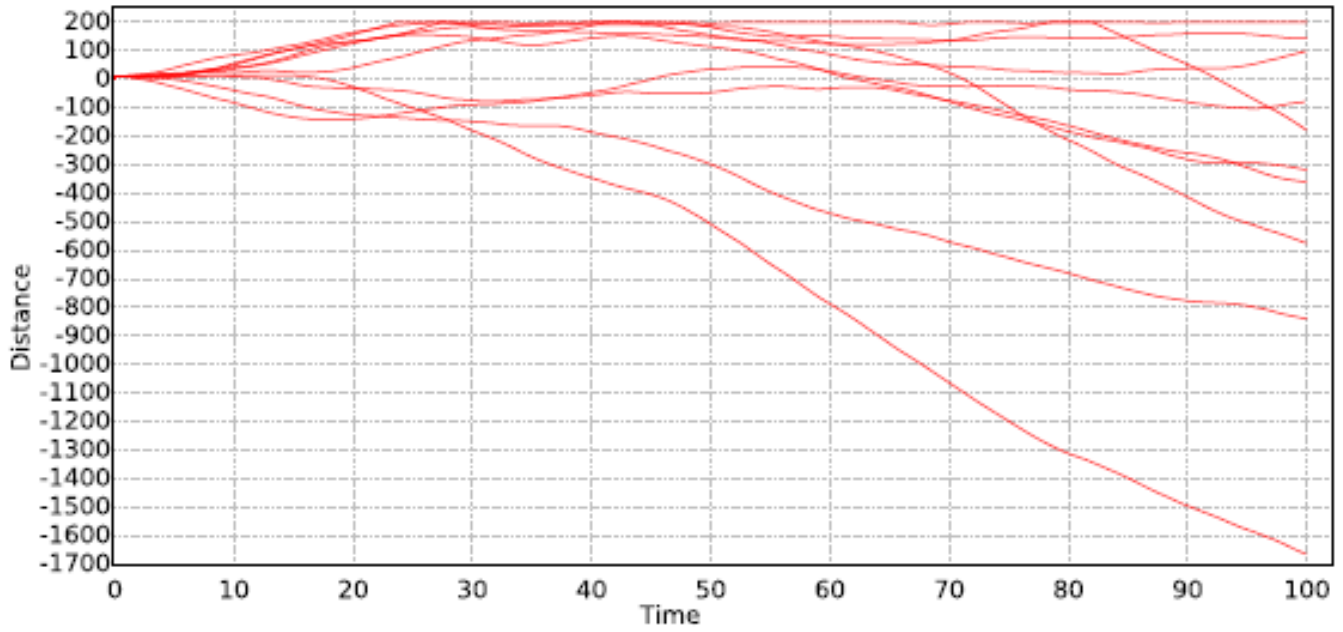
Front (complete)



No Strategy

```
Pr[<=100] (<> distance <= 5)
```

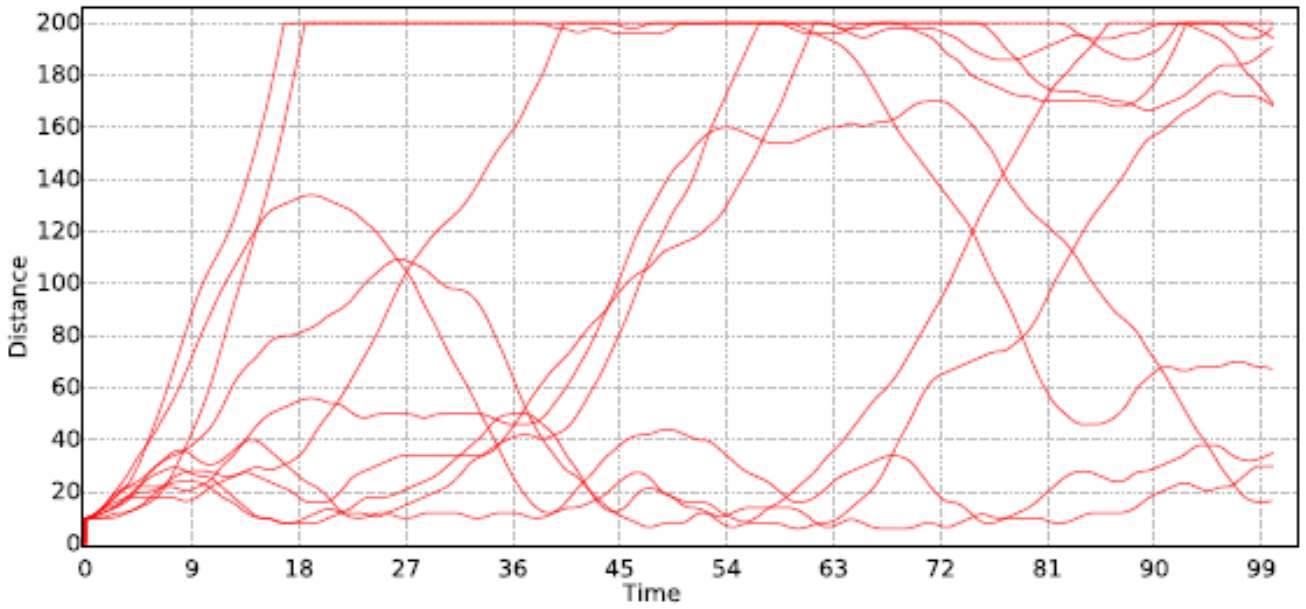
```
A[] distance > 5
```



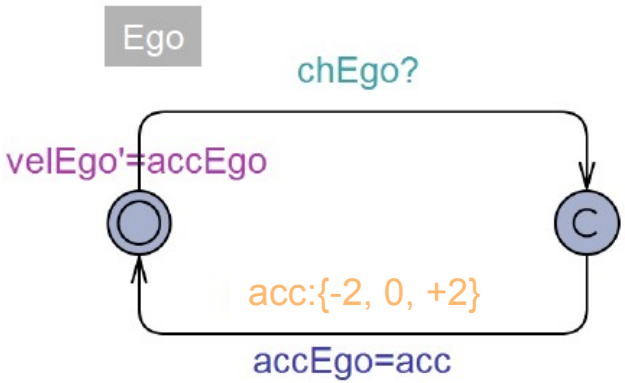
Safety Strategy

```
strategy safe = control: A[] distance > 5
```

```
A[] distance > 5 under safe
```



Safety Strategy (Code)



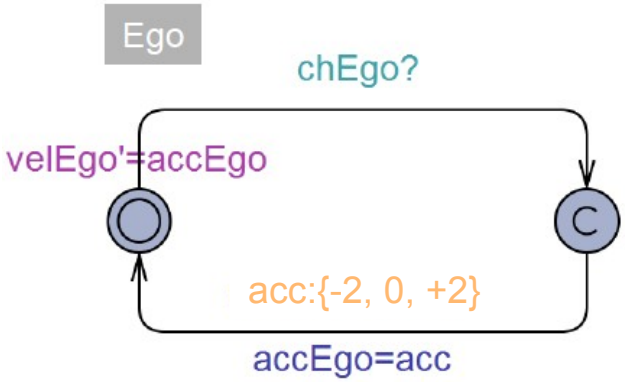
adaptiveCruiseControl - Notepad

File Edit Format View Help

```
State: ( Ego.Negative_acc Front.No_acceleration System.Wait Monitor._id12 ) #action=0
distance=47 velocityEgo=6 accelerationEgo=-2 velocityFront=12 accelerationFront=0
While you are in        (waitTimer<=1), wait.

State: ( Ego.No_acc Front.Positive_acc System.Wait Monitor._id12 ) #action=0 distance=83
velocityEgo=13 accelerationEgo=0 velocityFront=14 accelerationFront=2
While you are in        true, wait.

State: ( Ego.Choose Front.Choose System.Done Monitor._id12 ) #action=0
distance=181 velocityEgo=0 accelerationEgo=0 velocityFront=15 accelerationFront=0
When you are in         take transition Ego.Choose->Ego.Neg, tau, accelerationEgo := 0
}
When you are in         take transition Ego.Choose->Ego.Pos { velocityEgo <
maxVelocity             accelerationEgo := 2 }
When you are in         take transition Ego.Choose->Ego.Neg { velocityEgo >
minVelocity             accelerationEgo := -2 }

State: ( Ego.Positive_acc Front.Choose System.Done Monitor._id12 ) #action=0 distance=199
velocityEgo=7 accelerationEgo=-2 velocityFront=15 accelerationFront=0
While you are in        true, wait.

State: ( Ego.Negative_acc Front.Choose System.Done Monitor._id12 ) #action=0
distance=49 velocityEgo=0 accelerationEgo=-2 velocityFront=14 accelerationFront=2
While you are in        true, wait.

State: ( Ego.Positive_acc Front.Choose System.Done Monitor._id12 ) #action=0 distance=88
velocityEgo=0 accelerationEgo=2 velocityFront=11 accelerationFront=0
While you are in        true, wait.

State: ( Ego.Positive_acc Front.Choose System.Done Monitor._id12 ) #action=0 distance=174
velocityEgo=18 accelerationEgo=2 velocityFront=17 accelerationFront=2
While you are in        true, wait.

State: ( Ego.No_acc Front.Negative_acc System.Done Monitor._id12 ) #action=0 distance=147
```

4Mb

Safety Strategy (Code)




```
Ego.Choose <= 0: 3 (1481817.0)
Ego.Choose > 0
|  velocityEgo <= -10: 0 (39705.0/18380.0)
|  velocityEgo > -10
|  |  distance <= 200
|  |  |  velocityEgo <= 18
|  |  |  |  velocityEgo <= 12
|  |  |  |  |  distance <= 184
|  |  |  |  |  |  velocityEgo <= 0: 2 (331464.0/208577.0)
|  |  |  |  |  |  velocityEgo > 0
|  |  |  |  |  |  |  distance <= 122
|  |  |  |  |  |  |  |  distance <= 102: 2 (132918.0/80433.0)
|  |  |  |  |  |  |  |  distance > 102
|  |  |  |  |  |  |  |  |  velocityEgo <= 2
|  |  |  |  |  |  |  |  |  |  velocityFront <= 12: 1 (6255.0/4155.0)
|  |  |  |  |  |  |  |  |  |  velocityFront > 12
|  |  |  |  |  |  |  |  |  |  |  velocityFront <= 13: 2 (162.0)
|  |  |  |  |  |  |  |  |  |  |  velocityFront > 13
|  |  |  |  |  |  |  |  |  |  |  |  distance <= 110: 2 (870.0/363.0)
|  |  |  |  |  |  |  |  |  |  |  |  distance > 110
|  |  |  |  |  |  |  |  |  |  |  |  |  velocityFront <= 15
|  |  |  |  |  |  |  |  |  |  |  |  |  |  velocityFront <= 14: 1 (207.0/99.0)
|  |  |  |  |  |  |  |  |  |  |  |  |  |  velocityFront > 14: 2 (63.0)
|  |  |  |  |  |  |  |  |  |  |  |  |  velocityFront > 15
|  |  |  |  |  |  |  |  |  |  |  |  |  |  distance <= 116
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  velocityFront <= 17: 2 (126.0/54.0)
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  velocityFront > 17: 1 (129.0/39.0)
|  |  |  |  |  |  |  |  |  |  |  |  |  |  distance > 116: 1 (108.0)
|  |  |  |  |  |  |  |  |  |  velocityEgo > 2
|  |  |  |  |  |  |  |  |  |  velocityEgo <= 4: 1 (7689.0/4929.0)
```

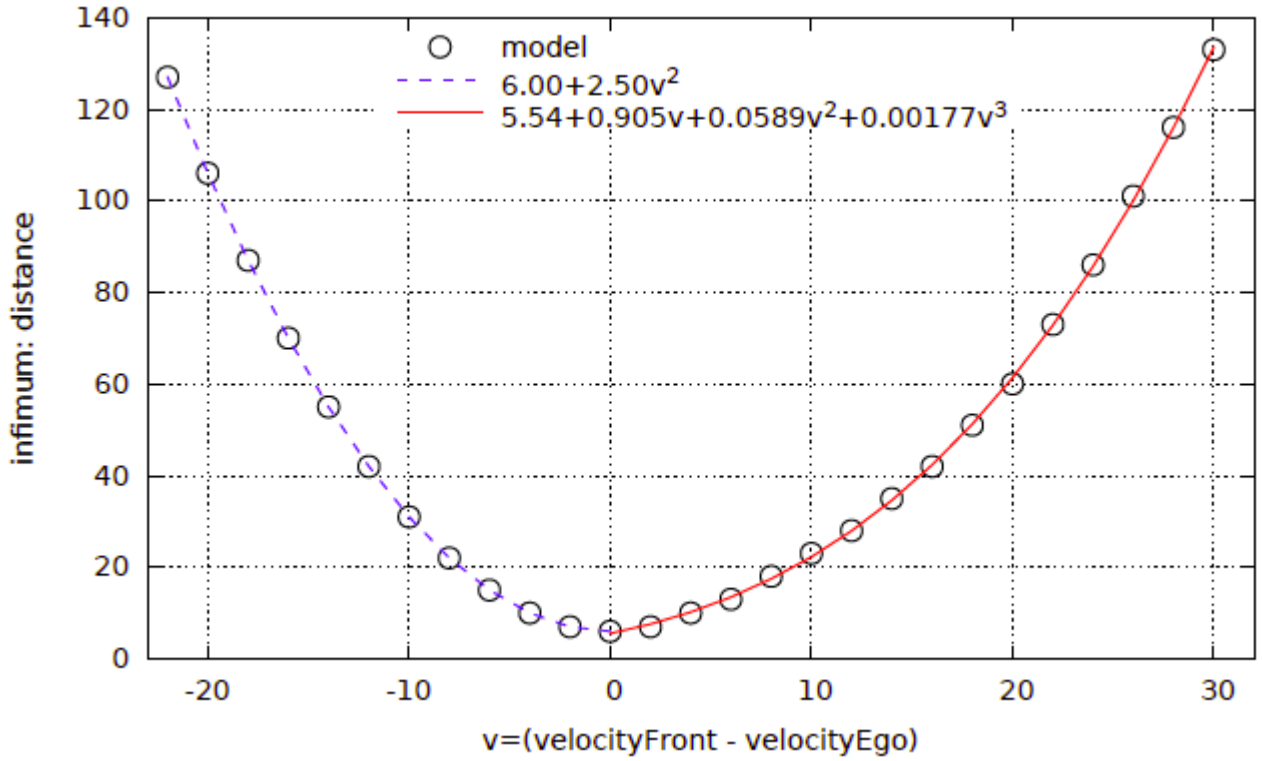
Learning Algorithm for Decision Tre

→65 line

(Jan Kretisnsky, Pranav Ashkot, TUM

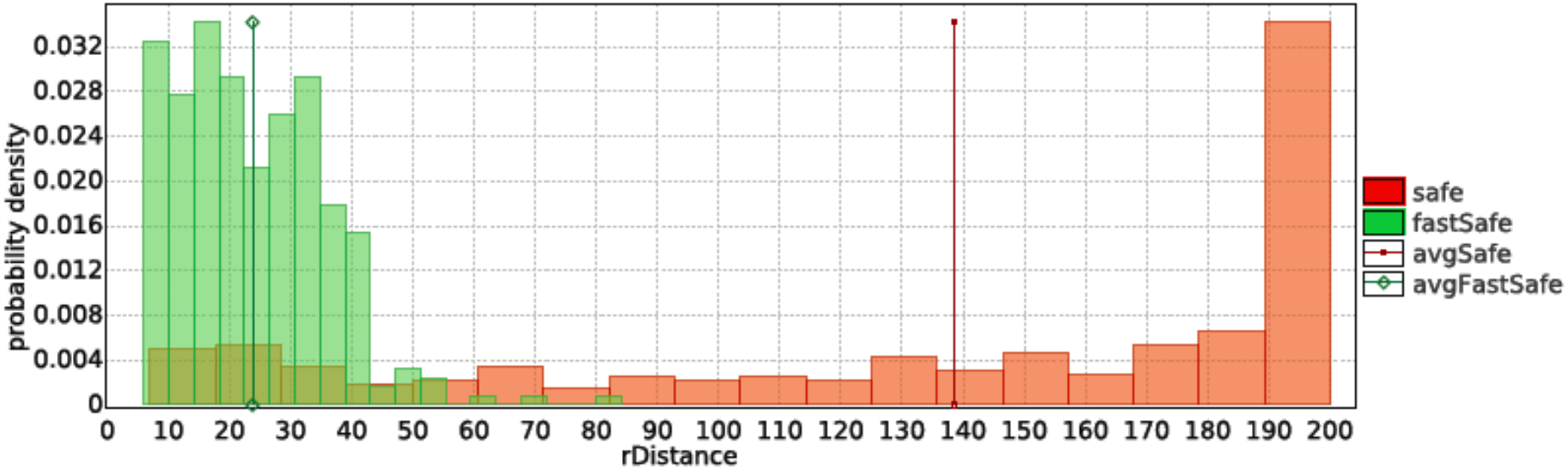
Safety Strategy

```
inf{velosityFront-velosityEgo==v}: distance under safe
```

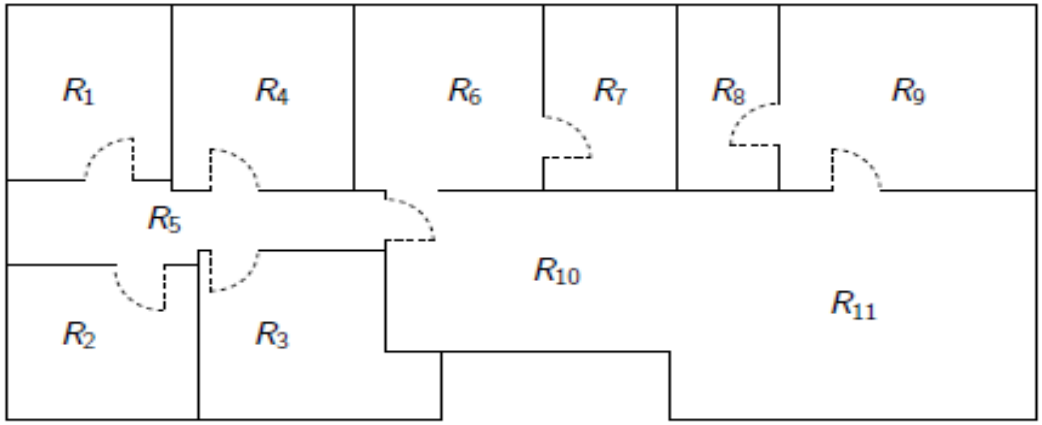


Optimal and Safe Strategy

```
strategy safeFast = minE (D) [<=100]: <> time >= 100 under safe
```



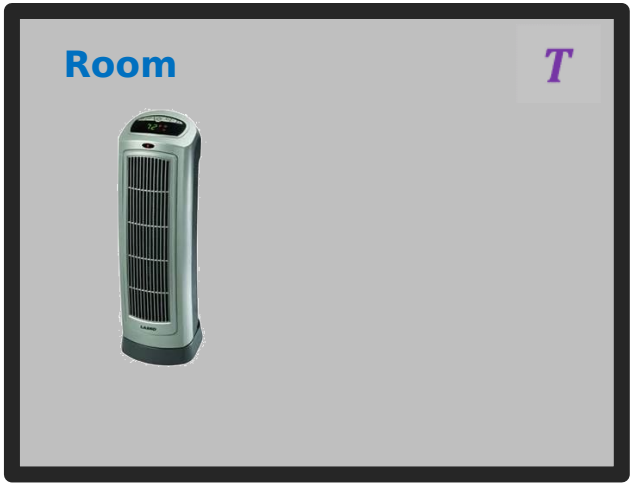
Application
Floor Heating



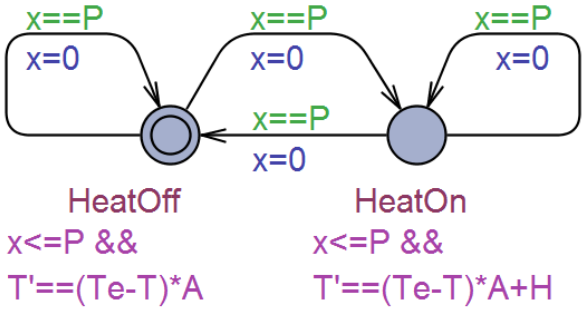
AALBORG UNIVERSITET

1-Room / 1-Window Game

Room

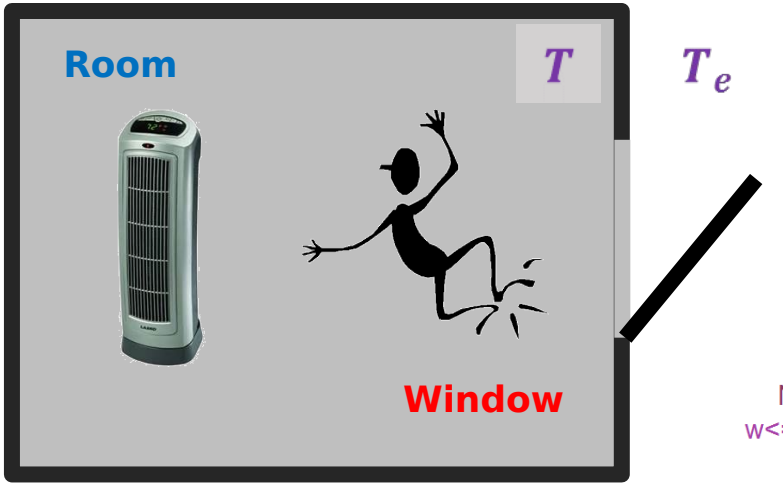


T_e




```
const double Tg = 21.0; // room temp. goal
const double Te = 15.0; // environment temp.
const double H = 0.04;  // power of heater
const double Aclosed = 0.002; // heat loss when window closed
const double Aopen = 0.004;   // heat loss when window open
const int P = 15; // heater switching period
const int h = 60; // 1 hour = 60 time units
```

1-Room / 1-Window Game

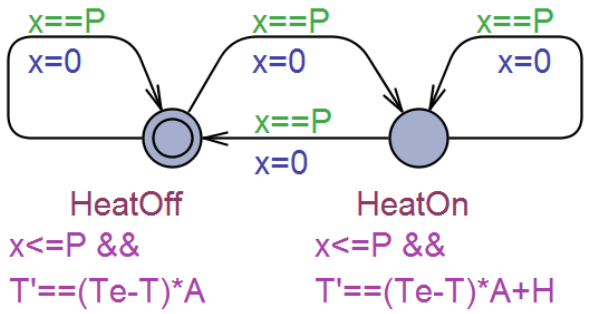


T_e

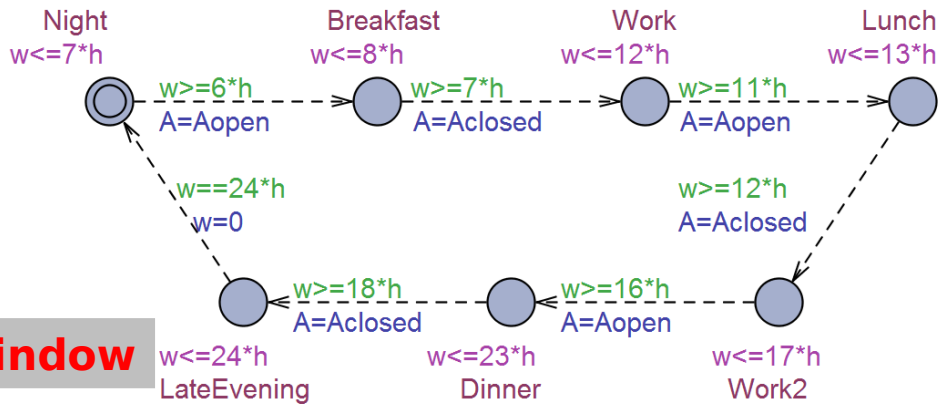
Find **strategy** that minimizes expected **discomfort**:

$$D(H) = \int_{t=0}^{t=H} \left(T(t) - T_g(t)\right)^2 dt$$

Room

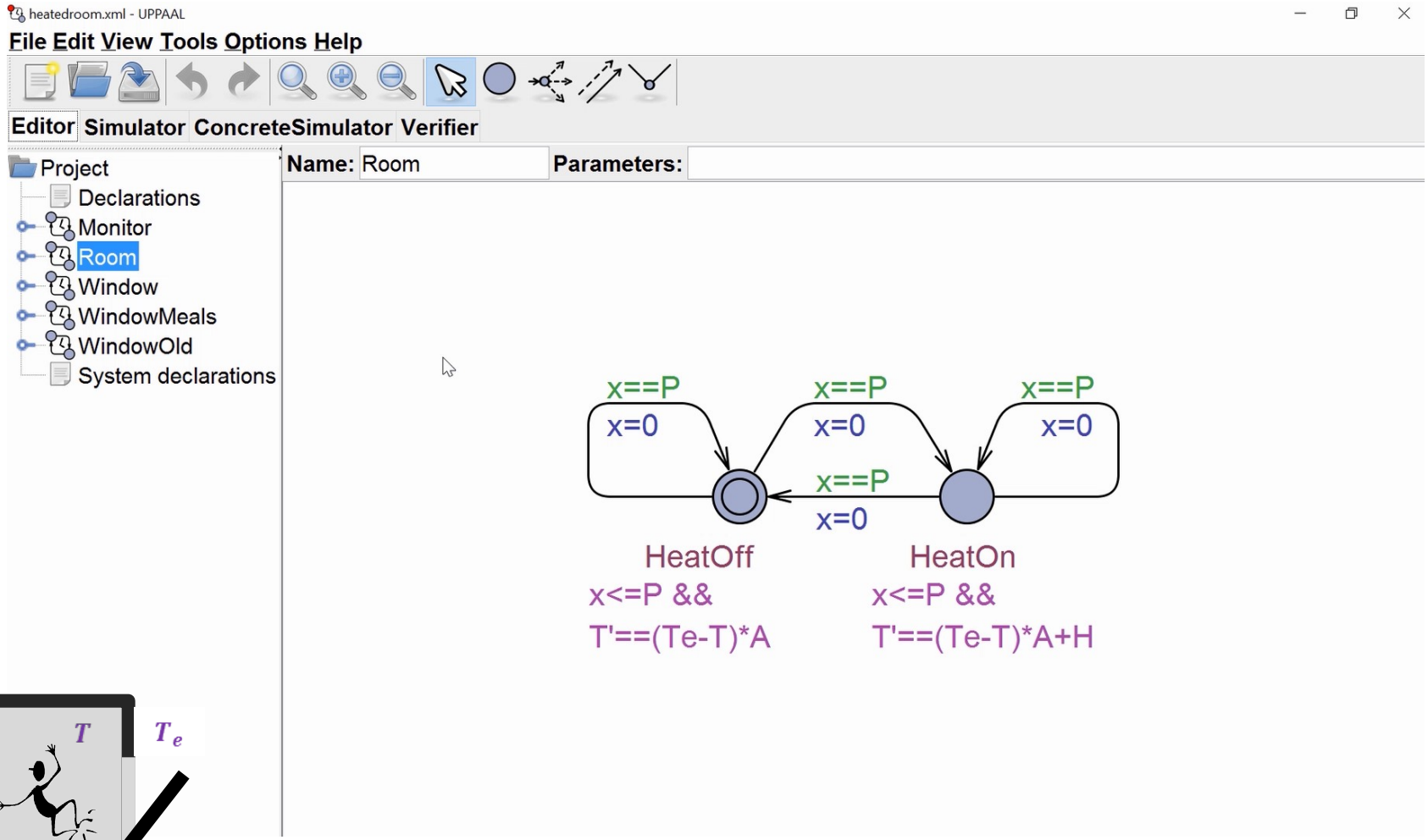


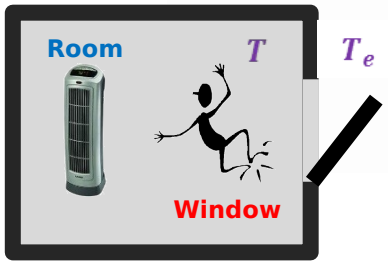
Window



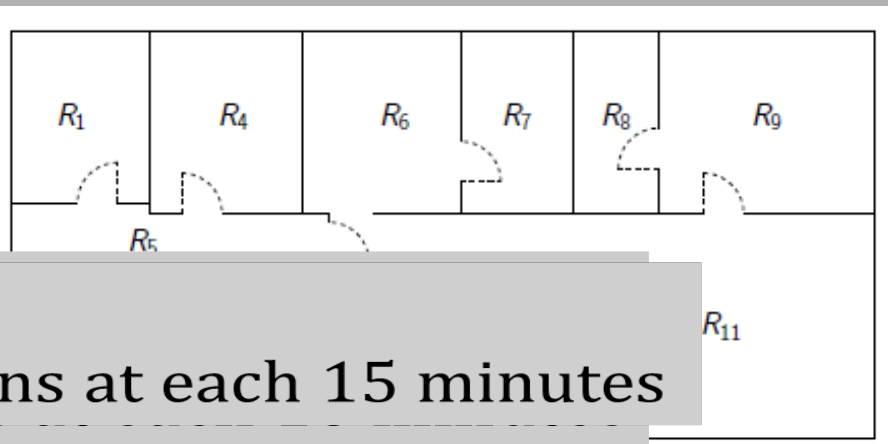
Uniform Dist

UPPAAL Synthesis – Demo





Floorheating



CHALLENGE

2^{11} valve configurations at each 15 minutes

The evolution of the room temperatures $\mathcal{F}_{v,d}$ are the solutions to the following differential equations:

$$\frac{d}{dt} T_i(t) = \text{ON-LINE SYNTHESIS} \; H^v_{j,i} \cdot v_j \; dt$$

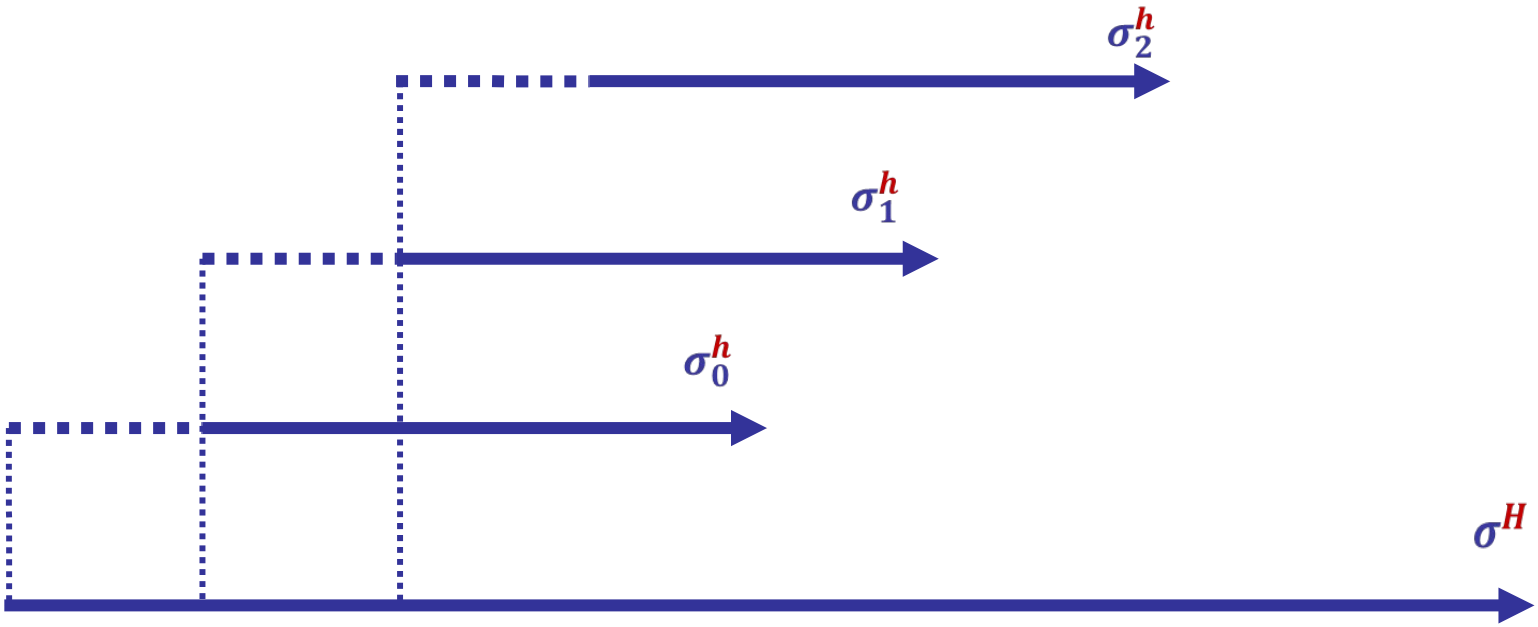
Where:

- A^d represents fferent rooms given the door confi
- B represents t nvironment and each room,
- H^v represents the heat exchange coefficients among each pipe and the rooms it heats given the valve configuration v,

COMPOSITIONAL SYNTHESIS

On-Line Synthesis

Instead of learning a strategy upto ***H*** repeated learn strategy for next 3 periods (***h***=45 minutes) ≈ model predictive control

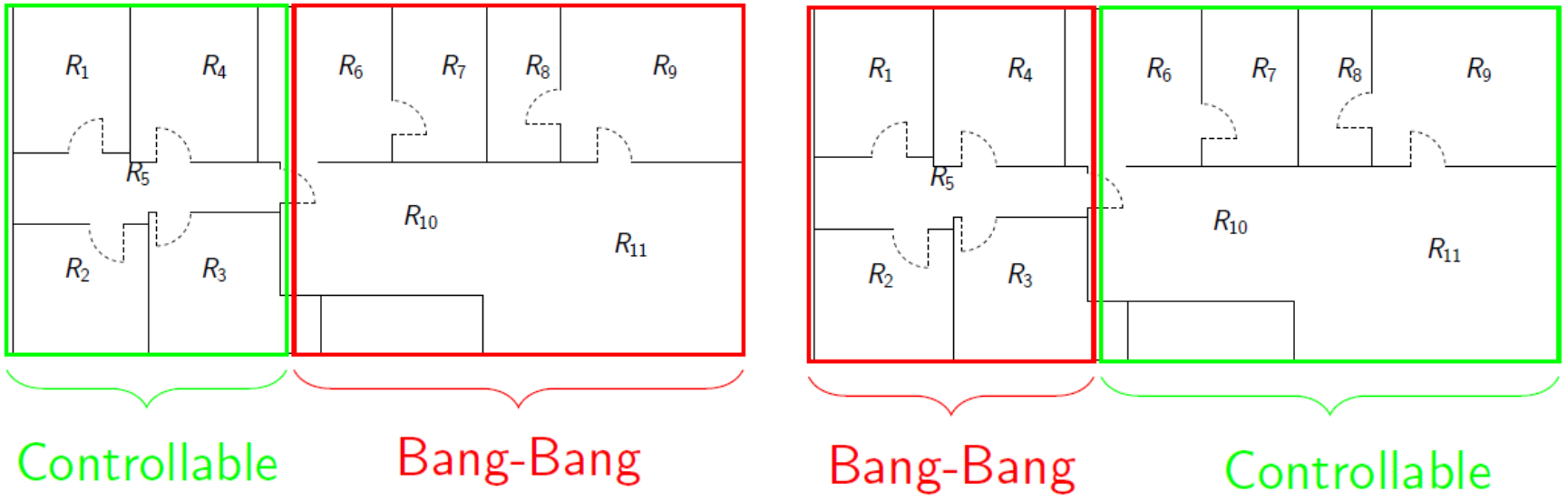


Still there are $2^{3 \cdot 11}$ valve combinations to consider.

Might not scale to 11 rooms?

Compositional Synthesis

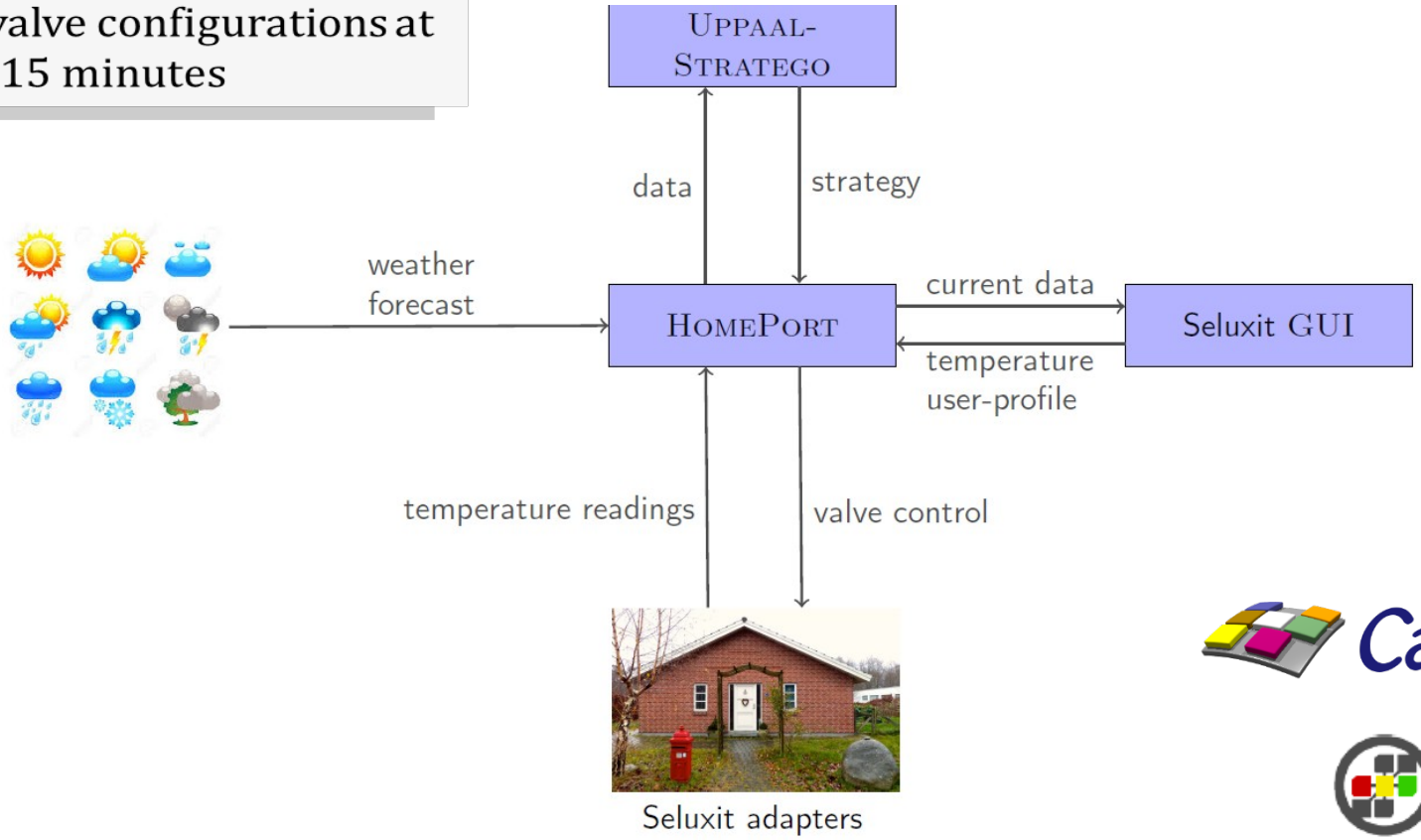
- Split the valves into controllable and fixed (controlled via Bang-Bang)
- Synthesize a strategy for controllable valves
- Swap the controllable and fixed valves and synthesise another strategy
- Merge strategies.



$(2^{5h} + 2^{6h})$ instead of 2^{11h} decision choices (in our case $h = 3$)

Full Floor Heating Case

CHALLENGE
2^{11} valve configurations at each 15 minutes



Cassting

seluxit

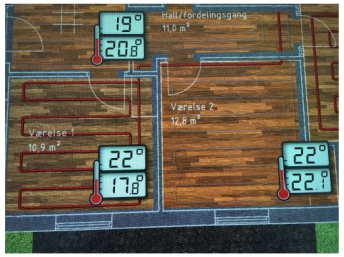
Larsen, Mikucionis, Muñiz, Srba, Taankvist, TACAS 16

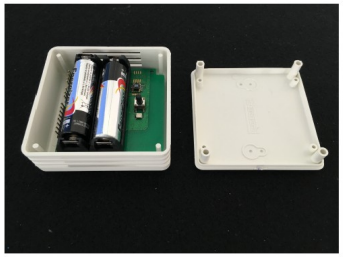
Full Floor Heating Case

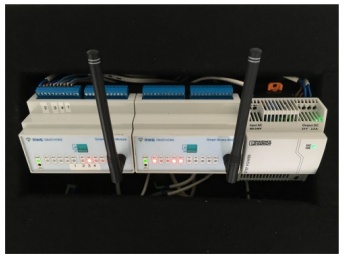
3 day scenario

Weather	Distance			Energy		
	Bang-Bang	Stratego	imp.	Bang-Bang	Stratego	imp.
Aalborg	14583	8342	**43%**	14180	12626	**10%**
Anadyr	2385515	1483272	**37%**	23040	22475	**2%**
Ankara	17985	10464	**41%**	17468	15684	**10%**
Minneapolis	22052	12175	**44%**	18165	15882	**12%**
Murmansk	399421	187941	**52%**	22355	21011	**6%**

Weather	Distance			Energy		
	Bang-Bang	Stratego	imp.	Bang-Bang	Stratego	imp.
Aalborg	14583	8552	**41%**	14180	12590	**11%**
Anadyr	2385515	1503448	**36%**	23040	22371	**2%**
Ankara	17985	10511	**41%**	17468	15697	**10%**
Minneapolis	22052	12725	**42%**	18165	15837	**12%**
Murmansk	399421	191441	**52%**	22355	20923	**6%**





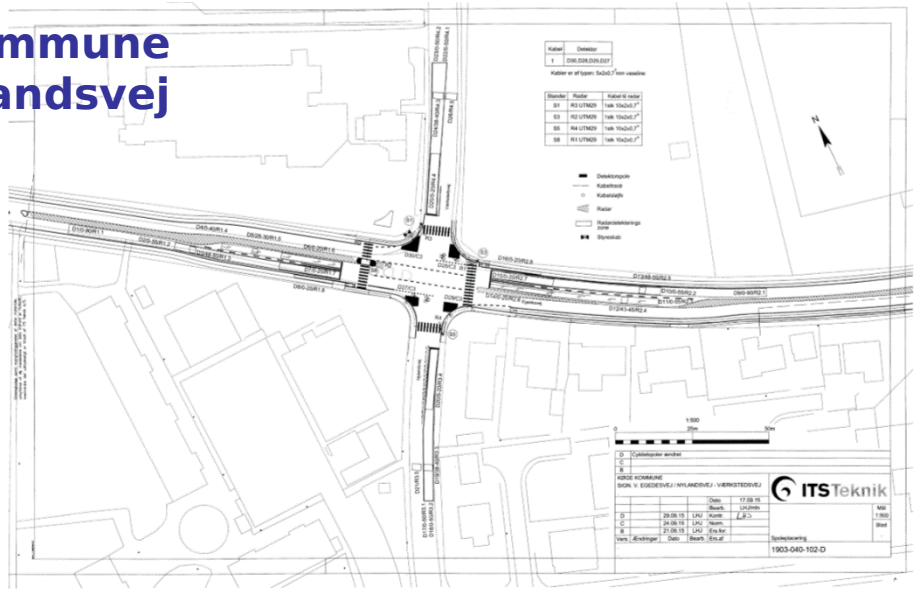


Evaluation of under modified parameters (0-20%)

A short movie of demonstrator at
www.casting-project.eu

Application
Traffic Control

Køge Kommune
Egedesvej/Nylandsvej



DiCyPS

w/ Marco Muniz
Jakob Taankvist
Harry Lahrman
Andeas Eriksen

AALBORG UNIVERSITET

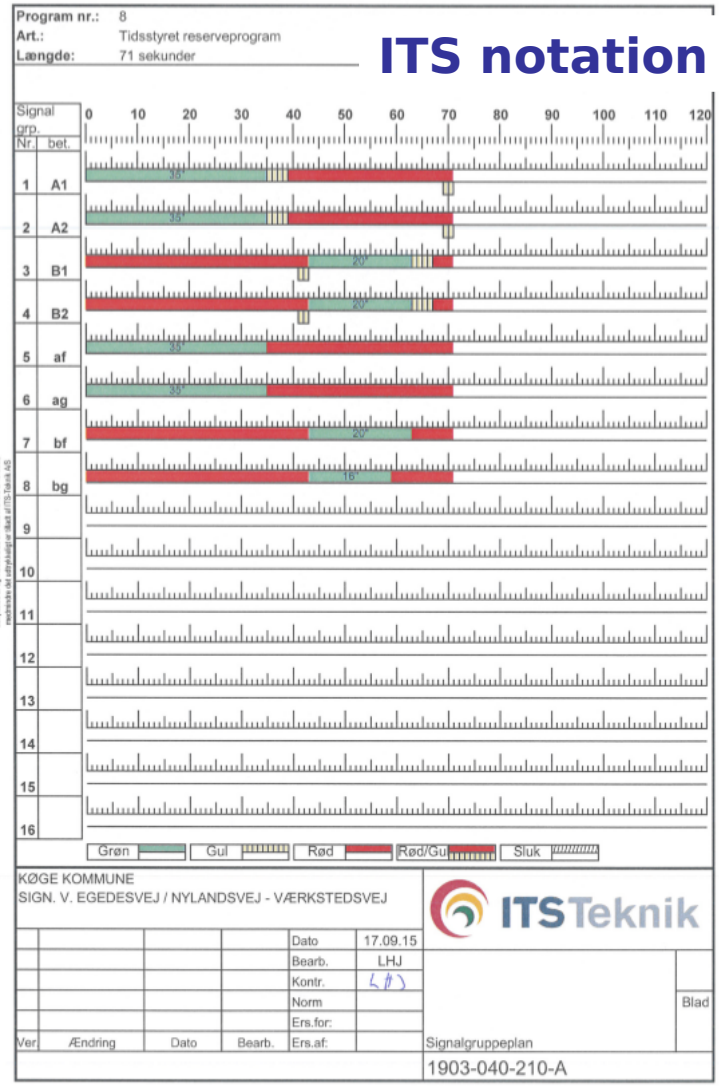
Eriksen, Huang, Kildebogaard, Lahrmann, Larsen, Muniz, Taankvist:
Uppaal Stratego for Intelligent Traffic Lights, ITS Europe 2017

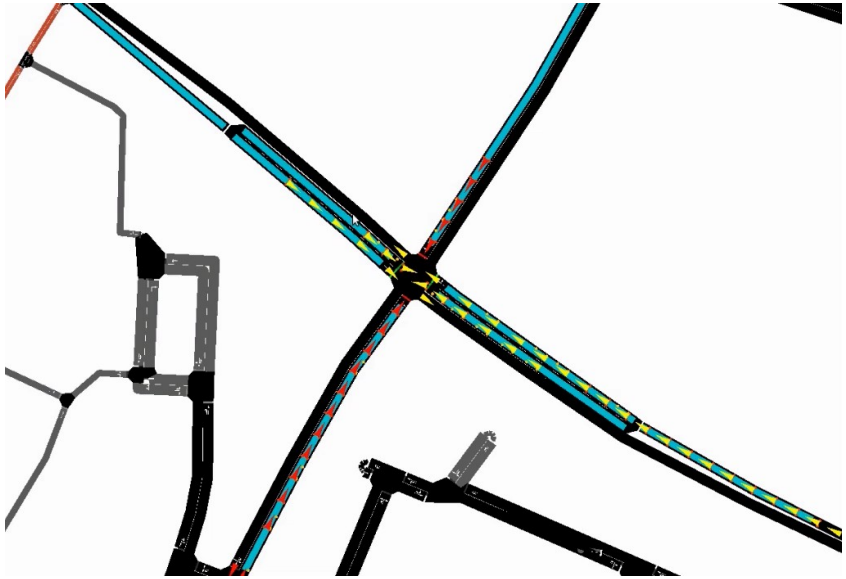
Objective

- Observed a lot of **unnecessary** waiting time for red light in signalized intersections

- Aim:
  Design an intelligent controller that can realize a near optimal signal control using UPPAAL Stratego

Time Triggered Control (Static)

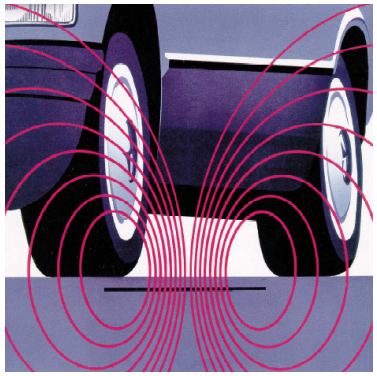




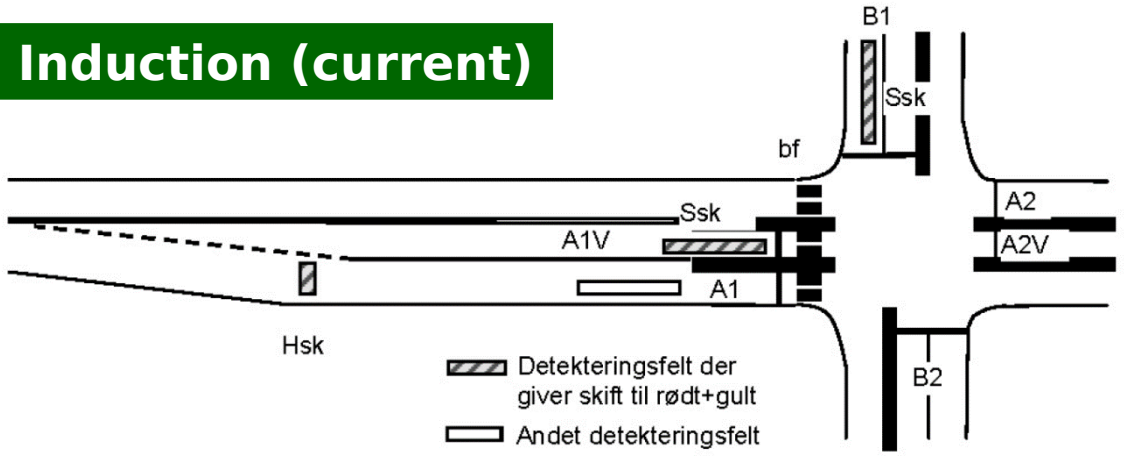
```
<tlLogic id="1693132977" type="static" programID="0">
<phase duration="35" state="rrGGgrrGGg"/>
<phase duration="4" state="rryyyrryyy"/>
<phase duration="4" state="rrrrrrrrrr"/>
<phase duration="20" state="GGrrrGGrrr"/>
<phase duration="4" state="yyrrryyrrr"/>
<phase duration="4" state="rrrrrrrrrr"/>
</tlLogic>
```

SUMO encoding

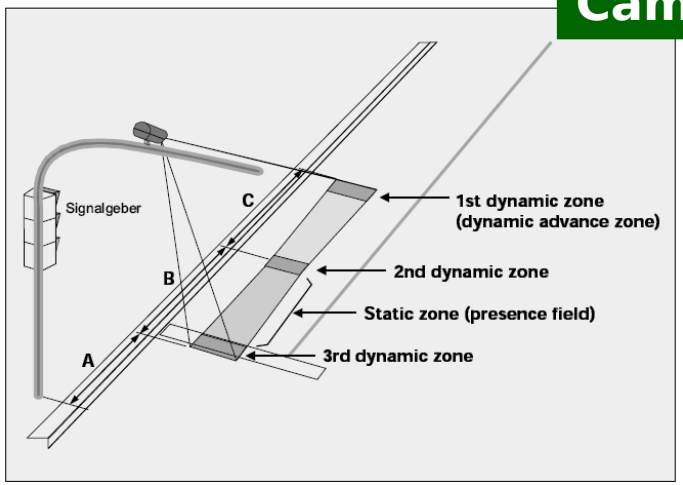
Detection



Loop Induction (current)



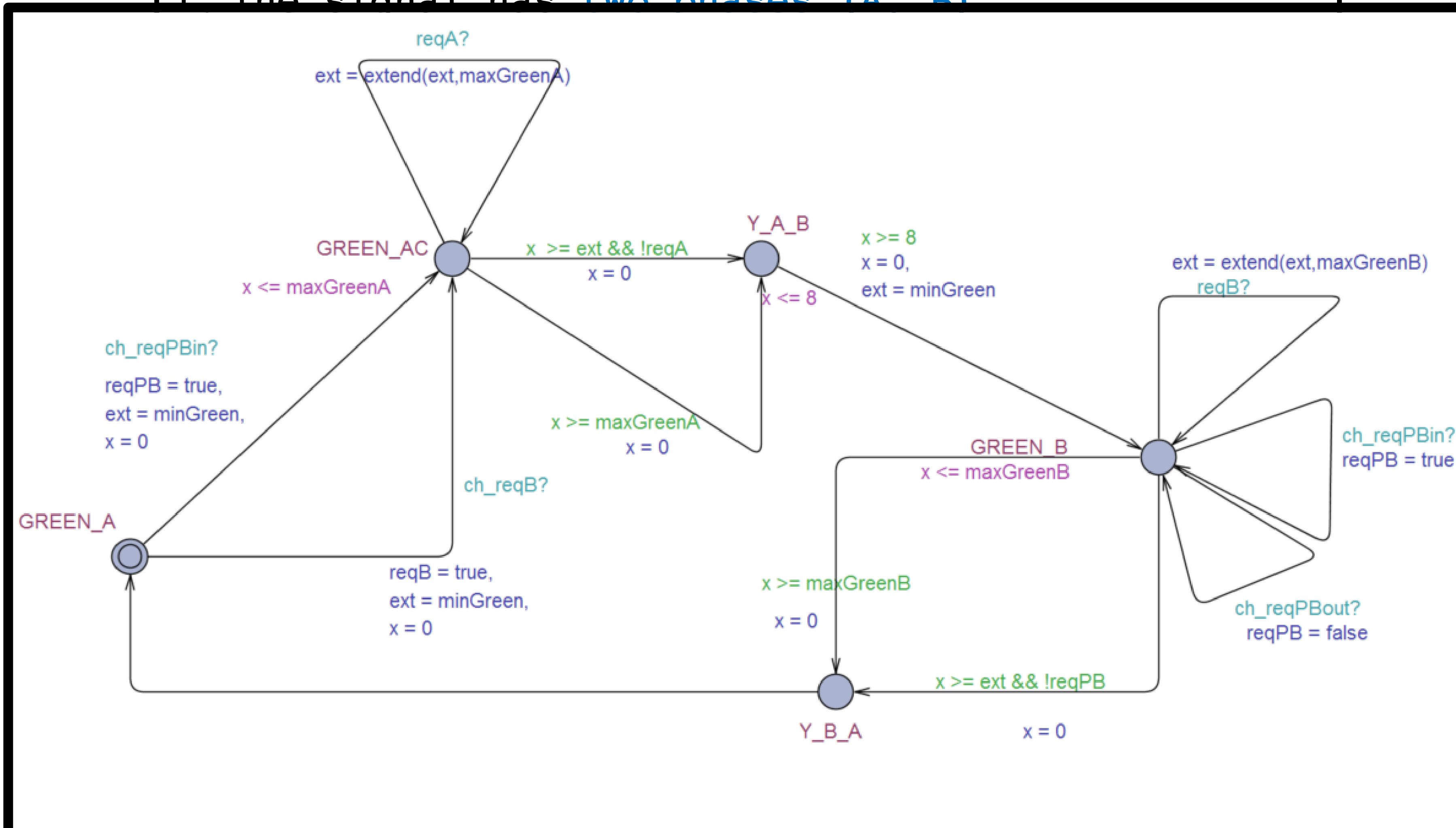
Camera (future)



Detection fields

Traffic Control (Induction Loop)

1. The signal has two phases (A, B)



until a max green time of 60 second is reached.

Learn Lib

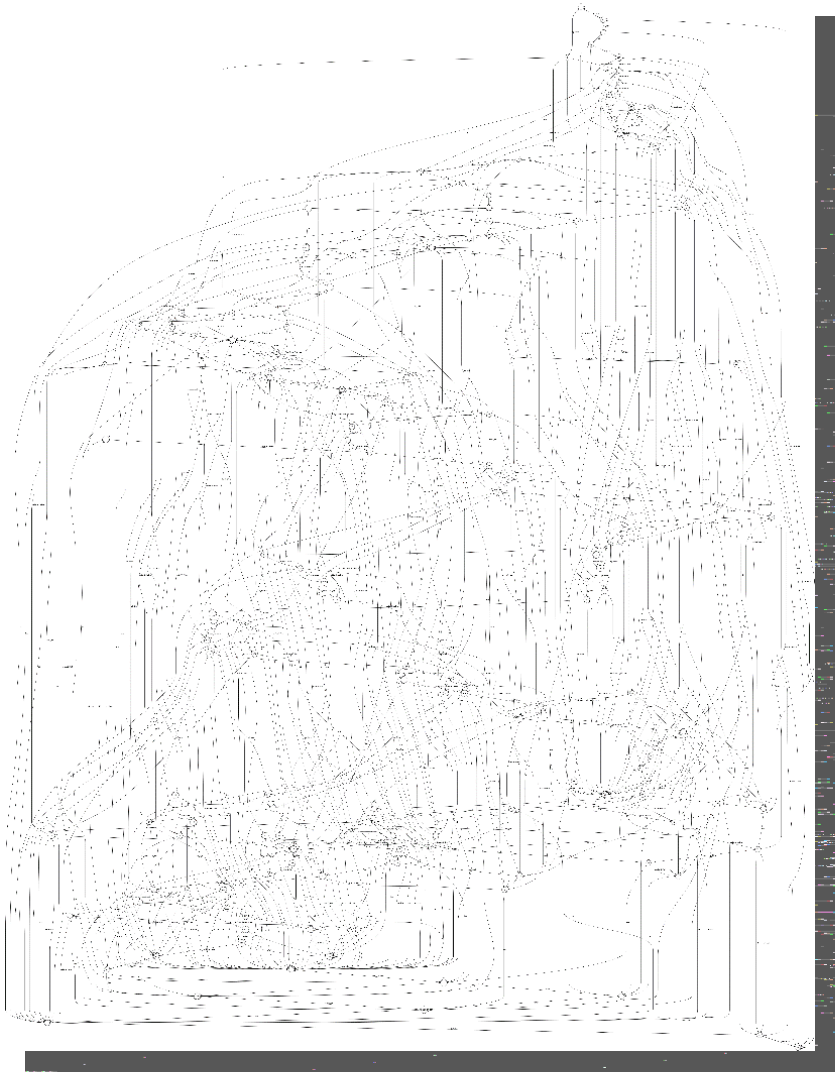
Angluin L* Algorithm

Complexity Simulation:

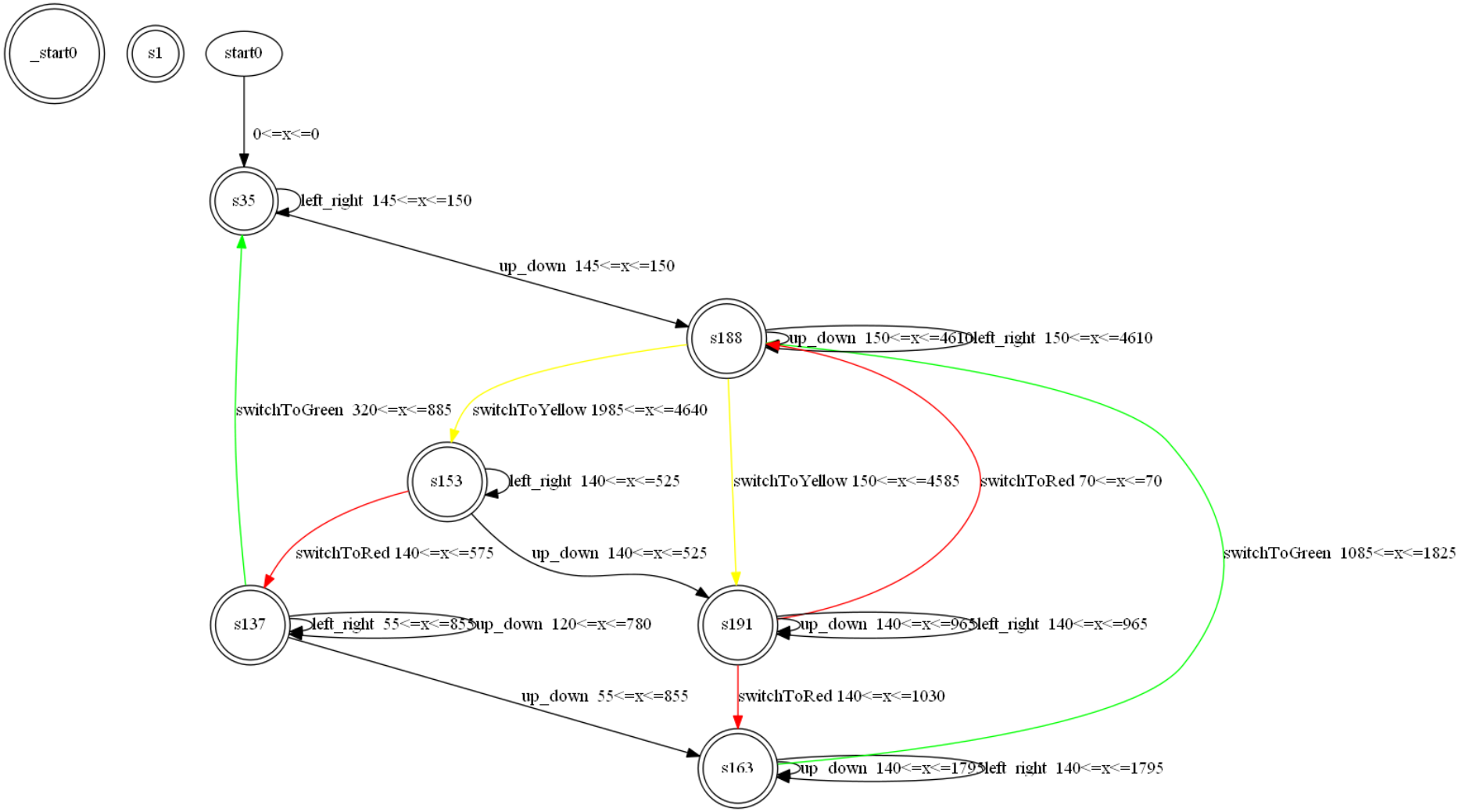
Reach depth 15 in 2 days
Over 100 000 simulations
~1 second per simulation

Complexity Real Execution:

Only estimations yet
Still over 100 000 runs needed
~2-5 minutes per run
(without preparation)
About a year (347 days)
for 5 minute runs



Reduced


_start0
s1
start0
0<=x<=0
s35
left_right 145<=x<=150
up_down 145<=x<=150
s188
up_down 150<=x<=4610
left_right 150<=x<=4610
switchToGreen 320<=x<=885
switchToYellow 1985<=x<=4640
s153
left_right 140<=x<=525
switchToYellow 150<=x<=4585
switchToRed 70<=x<=70
switchToRed 140<=x<=575
up_down 140<=x<=525
switchToGreen 1085<=x<=1825
s137
left_right 55<=x<=855
up_down 120<=x<=780
s191
up_down 140<=x<=965
left_right 140<=x<=965
up_down 55<=x<=855
switchToRed 140<=x<=1030
s163
up_down 140<=x<=1795
left_right 140<=x<=1795

Stratego Controller (Cameras)

Number of cars waiting in each lane *(full information)*

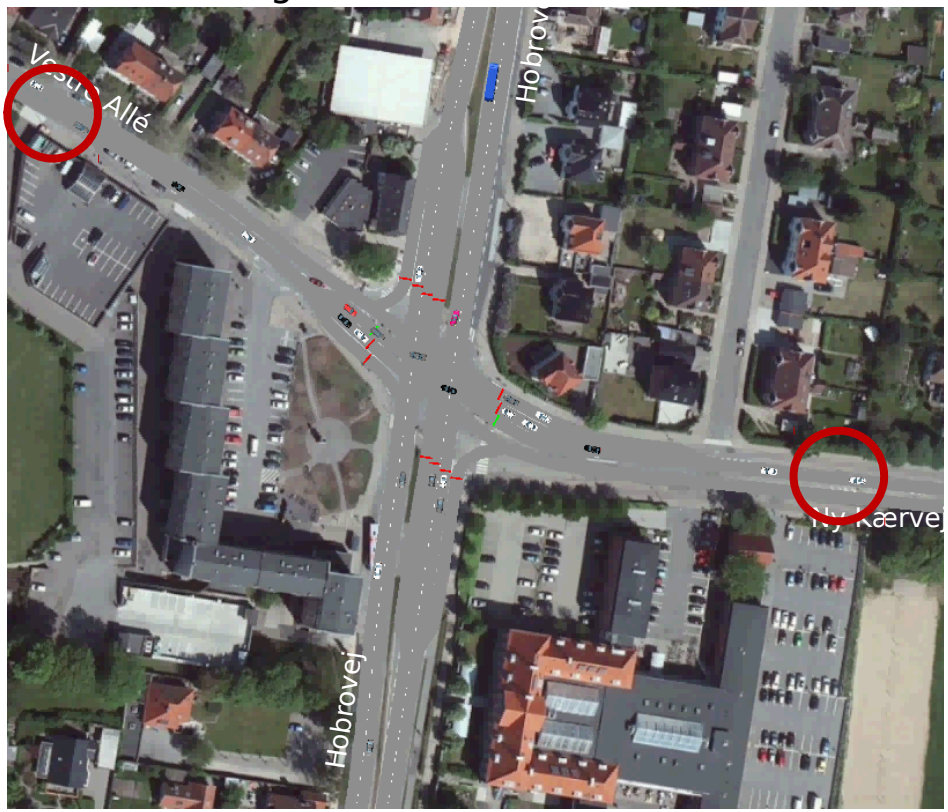
1: Every 5 to 8 sec read sensor data
2: **if** Traffic Light in yellow phase **then**
3: Run UPPAAL STRATEGO – decide next green phase
4: **else if** Traffic Light in green phase **then**
5: Run UPPAAL STRATEGO – extend green phase or go to yellow
6: **end if**

ONLINE Synthesis

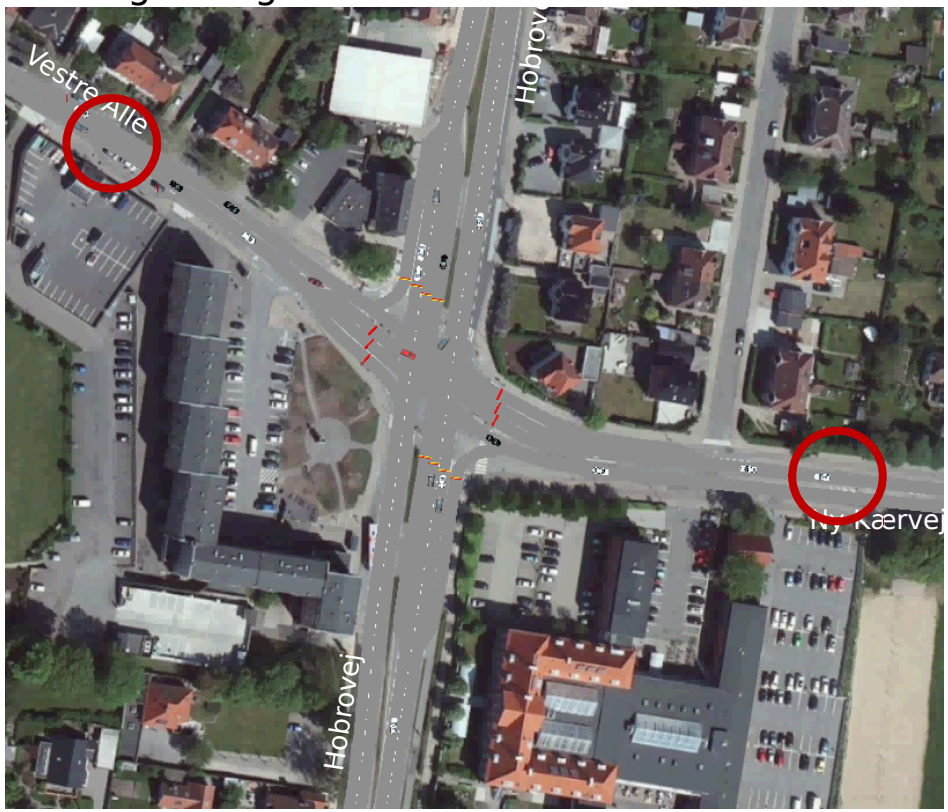
- Identify optimal strategy up to horizon H=90sec.
  - Strategy changes phase (at least 5 sec).
    - Modelling of stochastic arrival of cars in different directions (from 60-850 cars/hour)
- Minimize waiting time or jam (# of waiting >2sec)

VISSEM / STRATEGO co-simulation

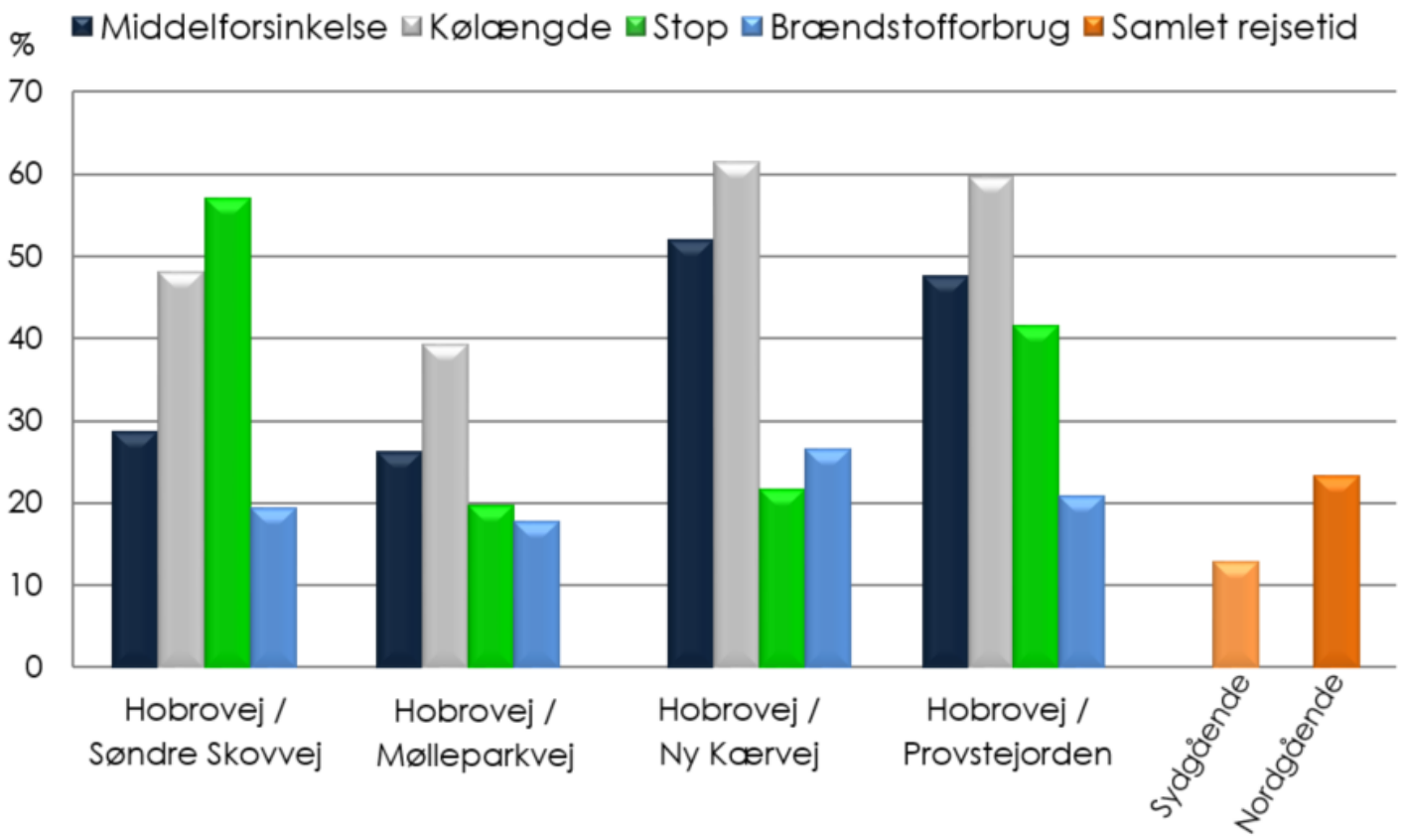
Traditional signal control



Intelligent signal control



Results


%
Middelforsinkelse
Kølængde
Stop
Brændstofforbrug
Samlet rejsetid
70
60
50
40
30
20
10
0
Hobrovej / Søndre Skovvej
Hobrovej / Mølleparkvej
Hobrovej / Ny Kærvej
Hobrovej / Provstejorden
Sydgående
Nordgående

Better Schedules

UPPAAL Stratego 2.0

Andreas B Eriksen, Manfred Jaeger, Peter Gjøl Jensen, Kim G Larsen, Axel Legay, Sean Sedwards,

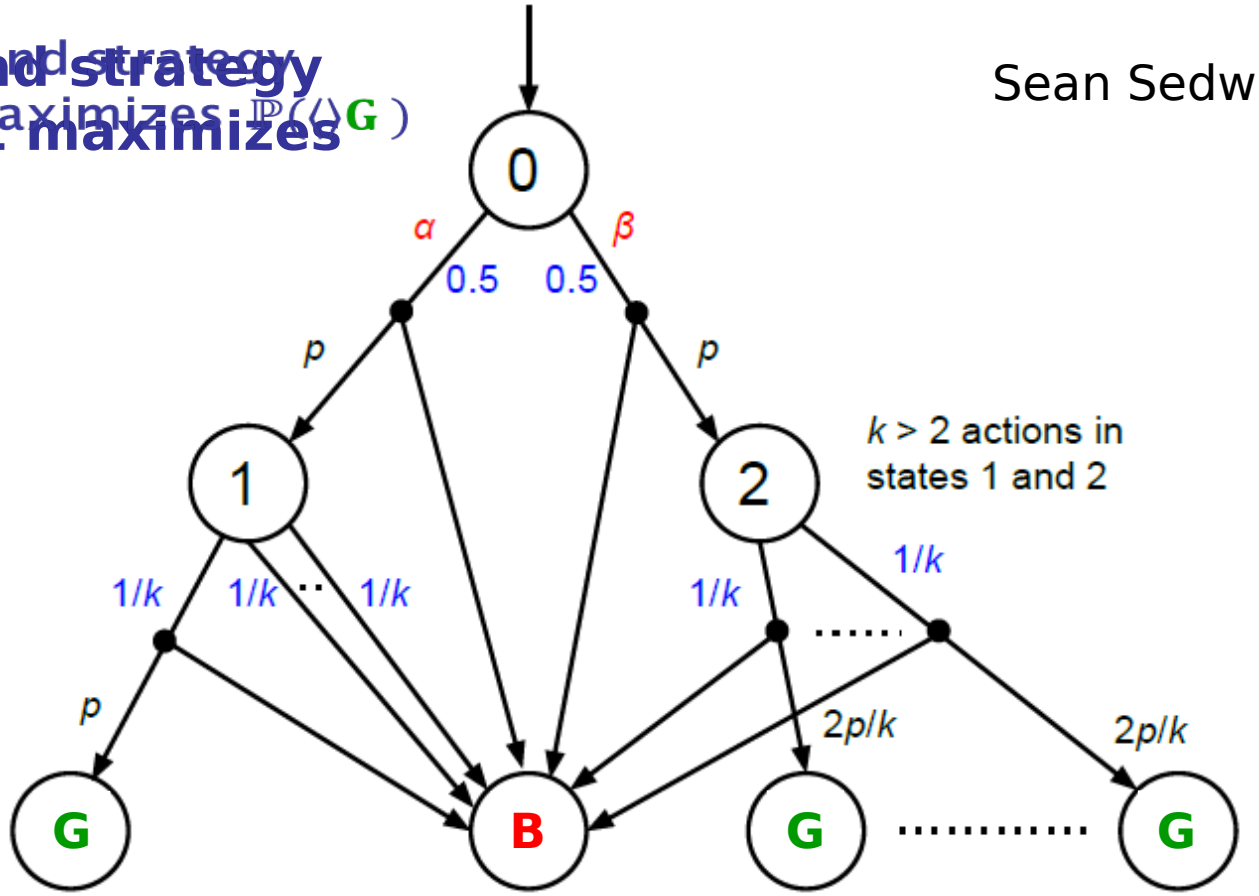
Jakob H Taankvist

AALBORG UNIVERSITET

Bad Example

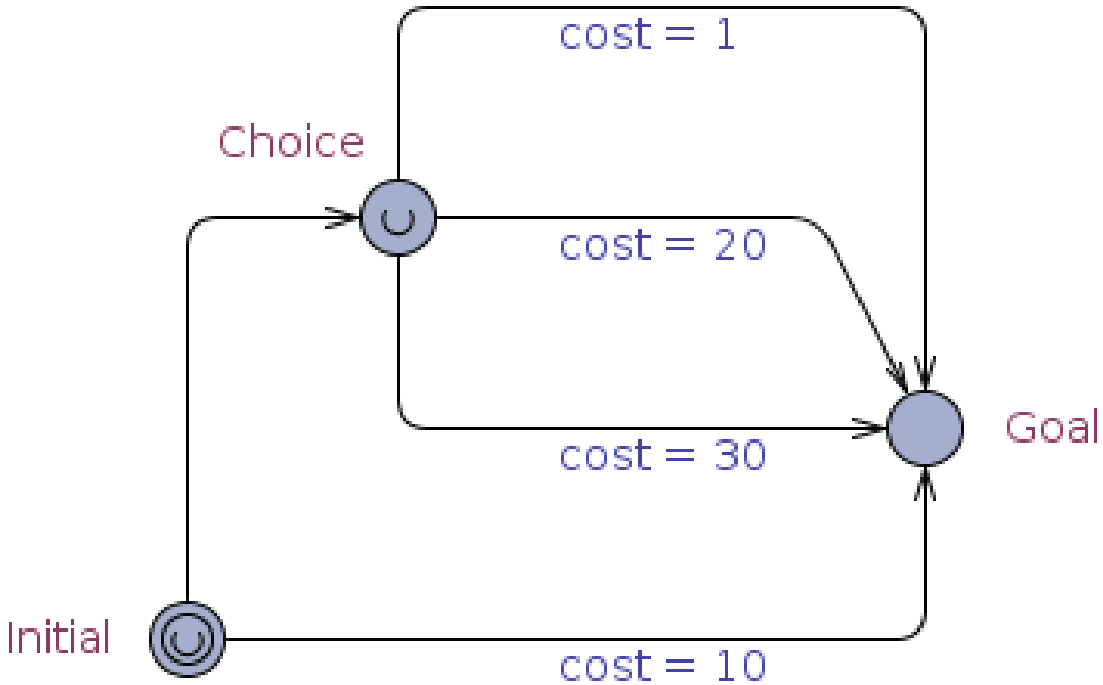
Find strategy that maximizes $\mathbb{P}(\Diamond \mathbf{G})$

Sean Sedwards



Bad Example

Find strategy
that minimizes $\mathbb{E}(\text{cost})$

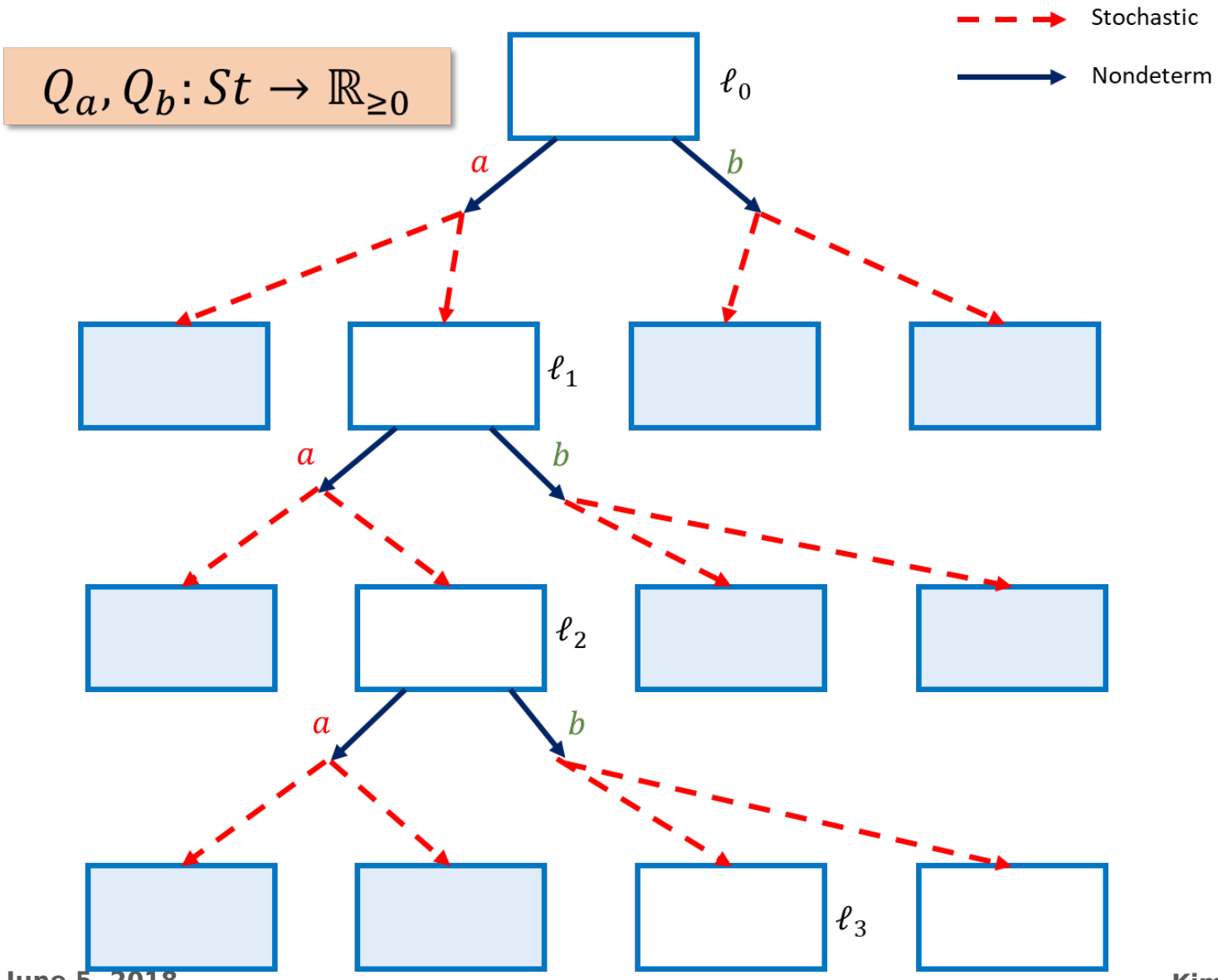


Peter Gjøl

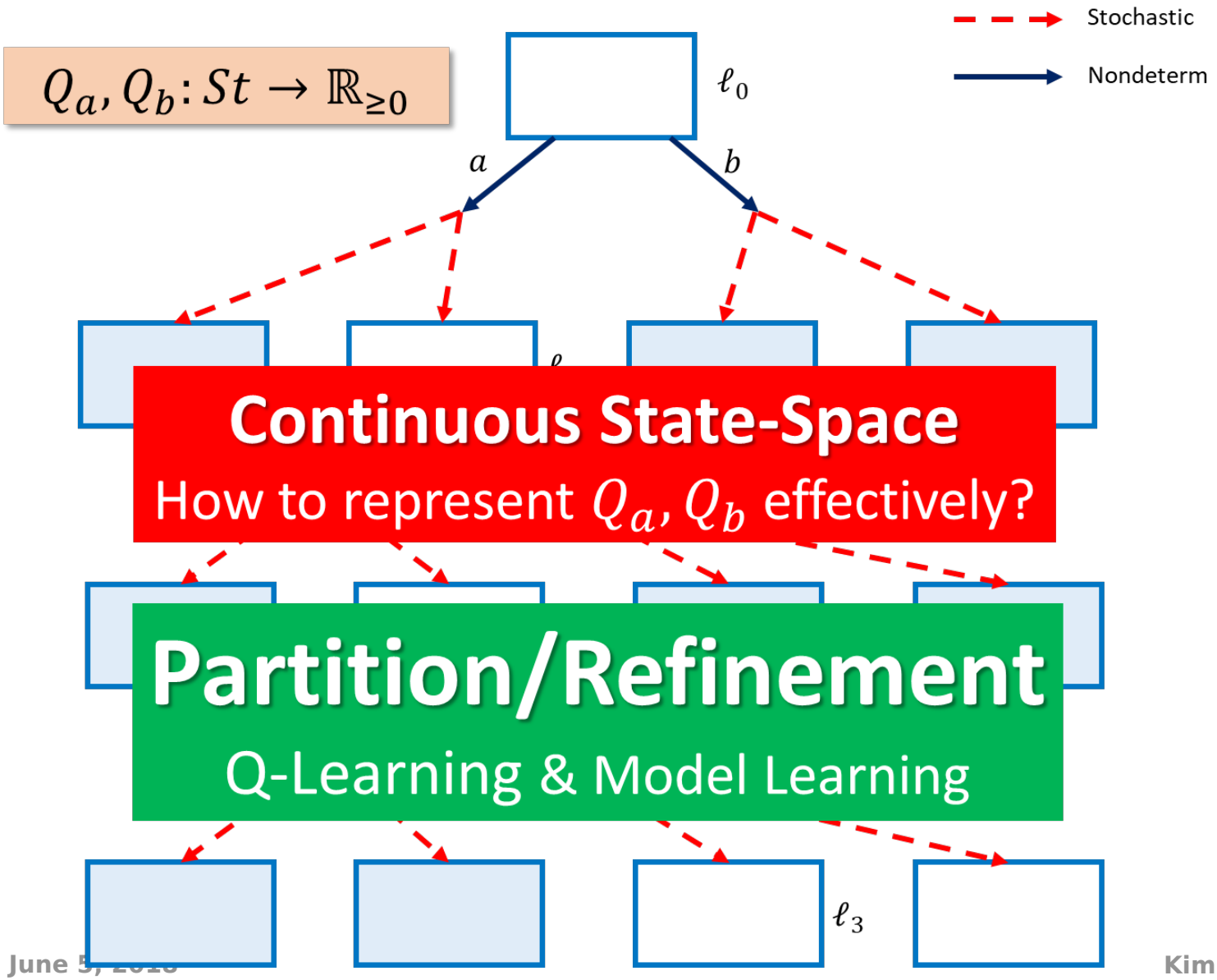
DEMO

AALBORG UNIVERSITET

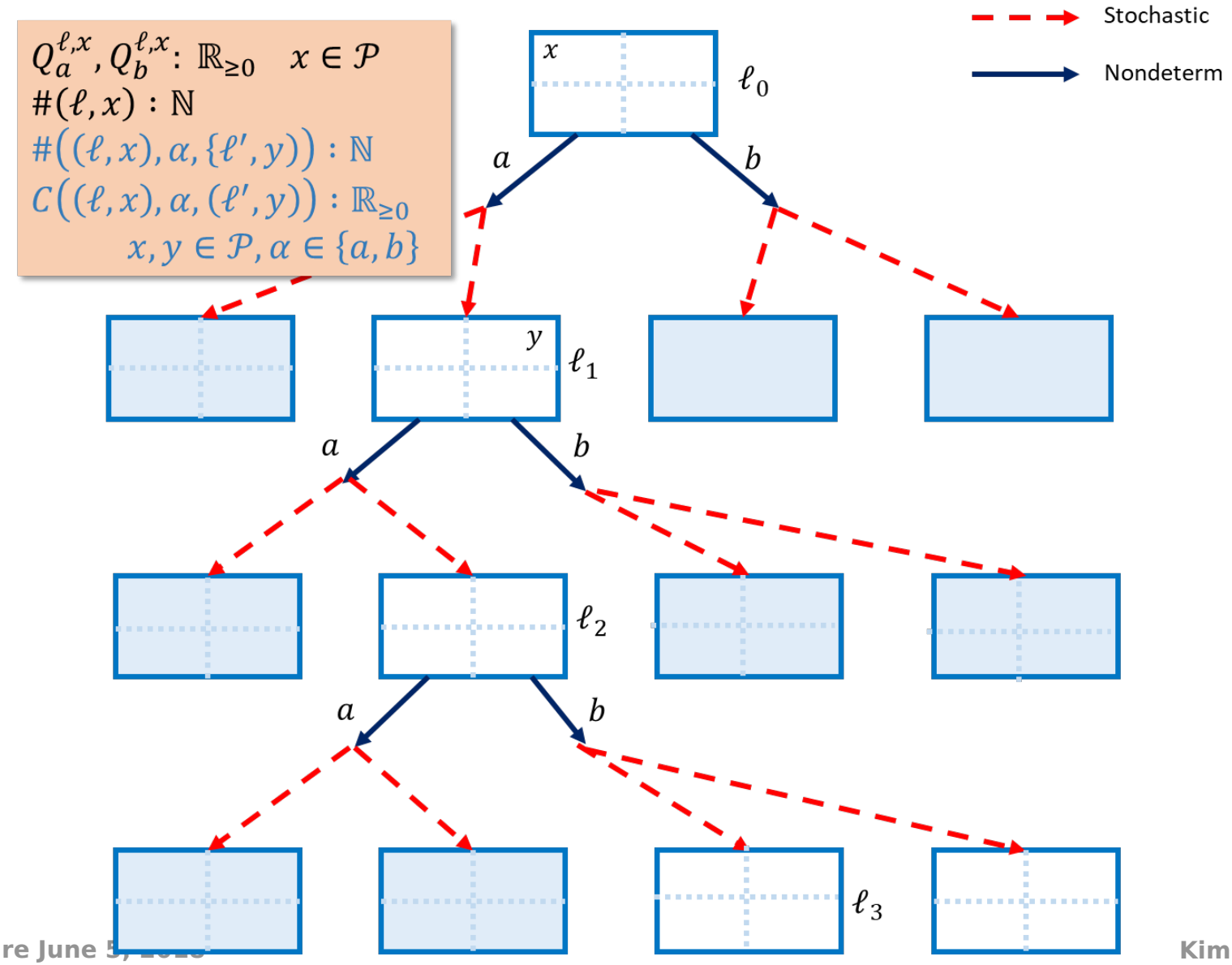
Q- & M-Learning


$Q_a, Q_b : St \rightarrow \mathbb{R}_{\geq 0}$
Stochastic
Nondeterm
ℓ_0
ℓ_1
ℓ_2
ℓ_3
a
b
a
b
a
b

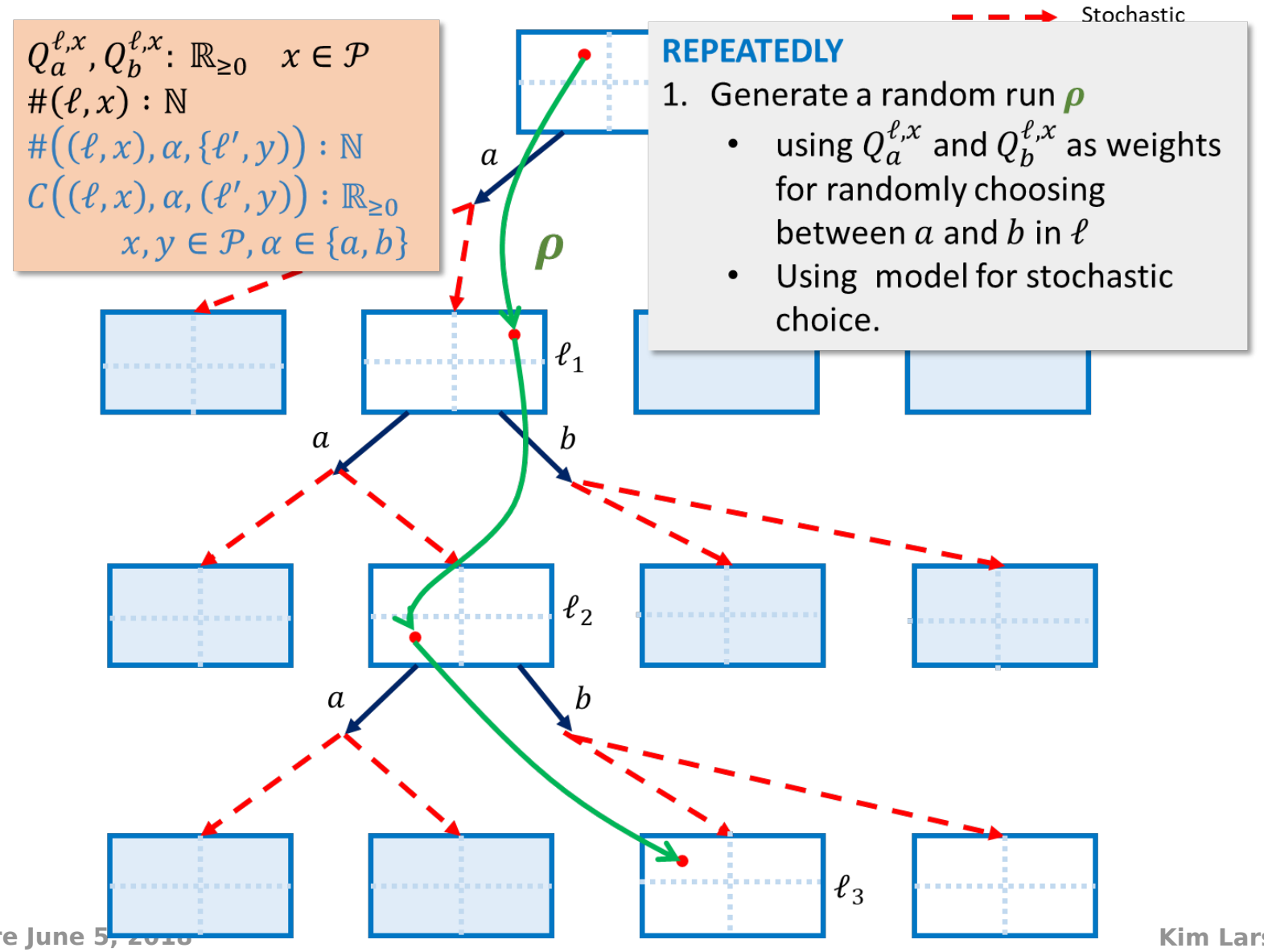
Q- & M-Learning


Stochastic
Nondeterm
$Q_a, Q_b : St \to \mathbb{R}_{\geq 0}$
ℓ_0
a
b
ℓ_3
Continuous State-Space
How to represent Q_a, Q_b effectively?
Partition/Refinement
Q-Learning & Model Learning

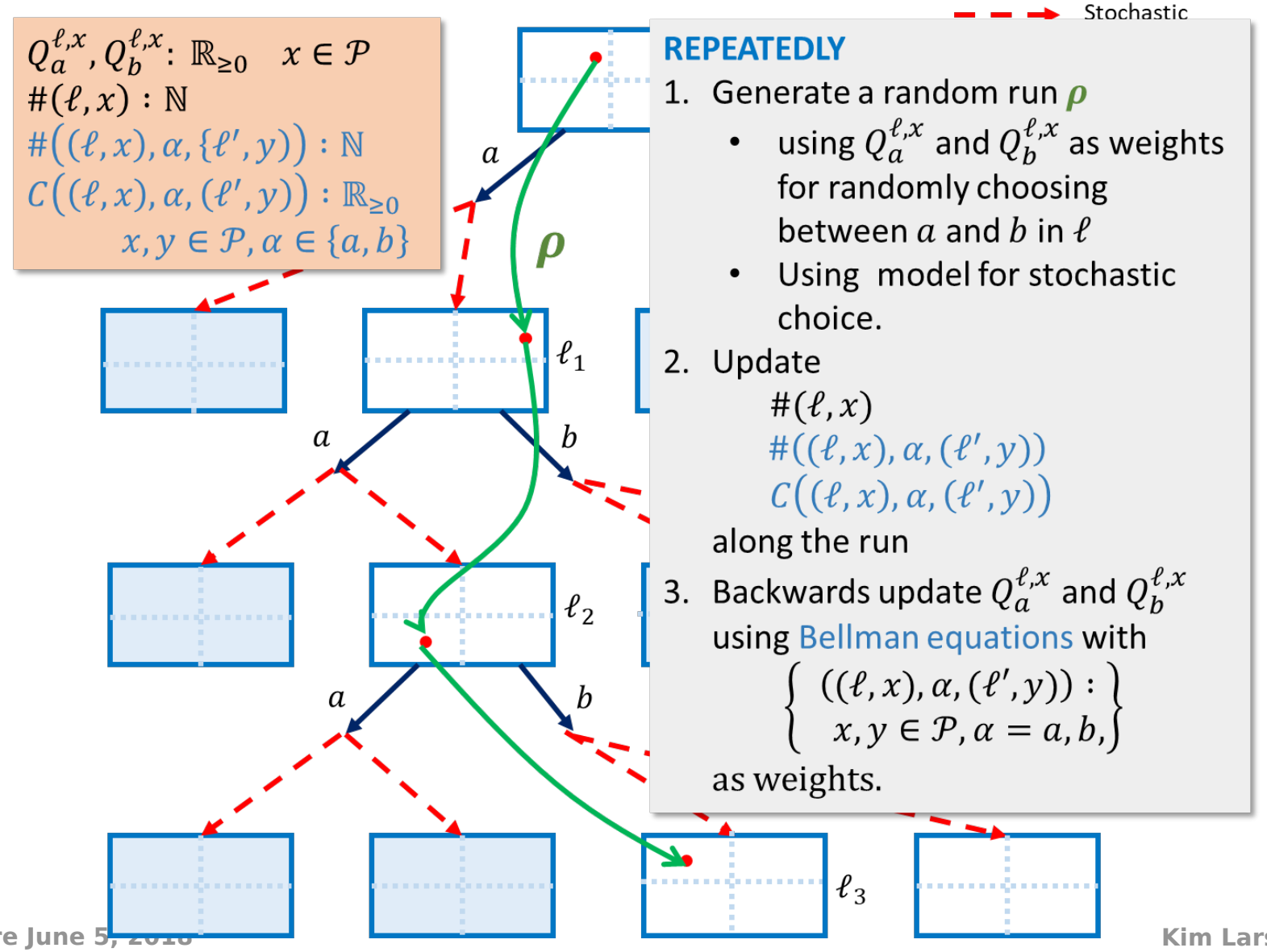
Q- & M-Learning


$Q_a^{\ell,x}, Q_b^{\ell,x}: \mathbb{R}_{\geq 0} \quad x \in \mathcal{P}$
$\#(\ell, x) : \mathbb{N}$
$\#\big((\ell, x), \alpha, \{\ell', y)\big) : \mathbb{N}$
$C\big((\ell, x), \alpha, (\ell', y)\big) : \mathbb{R}_{\geq 0}$
$x, y \in \mathcal{P}, \alpha \in \{a, b\}$
Stochastic
Nondeterm
x
ℓ_0
a
b
y
ℓ_1
a
b
ℓ_2
a
b
ℓ_3

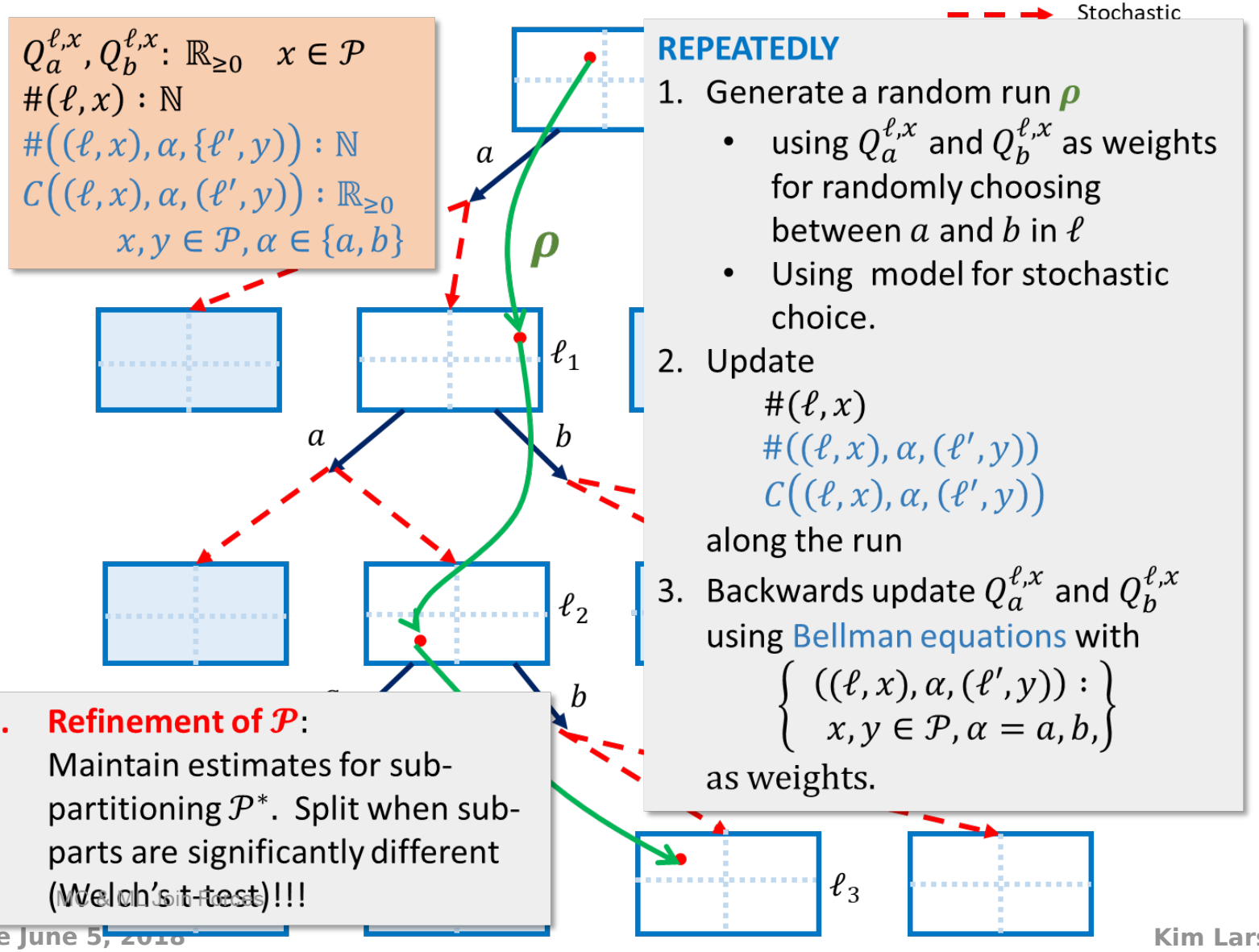
Q- & M-Learning


$Q_a^{\ell,x}, Q_b^{\ell,x}: \mathbb{R}_{\geq 0} \quad x \in \mathcal{P}$
$\#(\ell, x) : \mathbb{N}$
$\#\big((\ell, x), \alpha, \{\ell', y)\big) : \mathbb{N}$
$C\big((\ell, x), \alpha, (\ell', y)\big) : \mathbb{R}_{\geq 0}$
$x, y \in \mathcal{P}, \alpha \in \{a, b\}$
Stochastic
REPEATEDLY
1. Generate a random run ρ
• using $Q_a^{\ell,x}$ and $Q_b^{\ell,x}$ as weights for randomly choosing between a and b in ℓ
• Using model for stochastic choice.
ρ
a
b
ℓ_1
ℓ_2
ℓ_3

Q- & M-Learning


$Q_a^{\ell,x}, Q_b^{\ell,x}: \mathbb{R}_{\geq 0} \quad x \in \mathcal{P}$
$\#(\ell, x) : \mathbb{N}$
$\#\big((\ell, x), \alpha, \{\ell', y)\big) : \mathbb{N}$
$C\big((\ell, x), \alpha, (\ell', y)\big) : \mathbb{R}_{\geq 0}$
$x, y \in \mathcal{P}, \alpha \in \{a, b\}$
Stochastic
REPEATEDLY
1. Generate a random run ρ
   - using $Q_a^{\ell,x}$ and $Q_b^{\ell,x}$ as weights for randomly choosing between a and b in ℓ
   - Using model for stochastic choice.
2. Update
   $\#(\ell, x)$
   $\#((\ell, x), \alpha, (\ell', y))$
   $C\big((\ell, x), \alpha, (\ell', y)\big)$
   along the run
3. Backwards update $Q_a^{\ell,x}$ and $Q_b^{\ell,x}$ using Bellman equations with
   $\left\{ \begin{array}{l} ((\ell, x), \alpha, (\ell', y)) : \\ x, y \in \mathcal{P}, \alpha = a, b, \end{array} \right\}$
   as weights.
a
b
ρ
ℓ_1
ℓ_2
ℓ_3

Q- & M-Learning


Stochastic
$Q_a^{\ell,x}, Q_b^{\ell,x}$: $\mathbb{R}_{\geq 0}$ $x \in \mathcal{P}$
$\#(\ell, x) : \mathbb{N}$
$\#\big((\ell, x), \alpha, \{\ell', y)\big) : \mathbb{N}$
$C\big((\ell, x), \alpha, (\ell', y)\big) : \mathbb{R}_{\geq 0}$
$x, y \in \mathcal{P}, \alpha \in \{a, b\}$
REPEATEDLY
1. Generate a random run ρ
   - using $Q_a^{\ell,x}$ and $Q_b^{\ell,x}$ as weights for randomly choosing between a and b in ℓ
   - Using model for stochastic choice.
2. Update
   $\#(\ell, x)$
   $\#((\ell, x), \alpha, (\ell', y))$
   $C\big((\ell, x), \alpha, (\ell', y)\big)$
   along the run
3. Backwards update $Q_a^{\ell,x}$ and $Q_b^{\ell,x}$ using Bellman equations with
   $\left\{ \begin{array}{l} ((\ell, x), \alpha, (\ell', y)) : \\ x, y \in \mathcal{P}, \alpha = a, b, \end{array} \right\}$
   as weights.
a
ρ
ℓ_1
a
b
ℓ_2
b
ℓ_3
5. **Refinement of $\mathcal{P}$**:
Maintain estimates for sub-partitioning $\mathcal{P}^*$. Split when sub-parts are significantly different (Welch's t-test)!!!

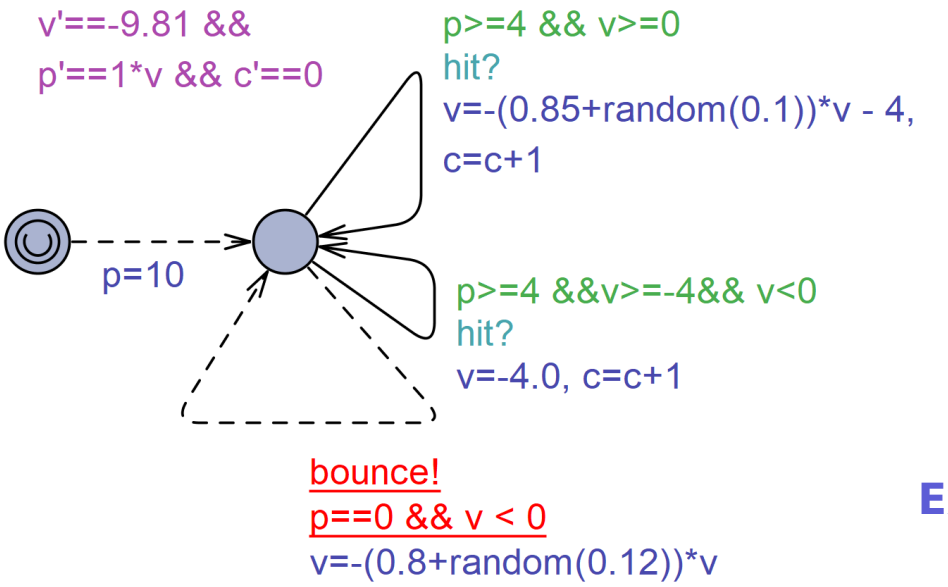
DEMO

2.0

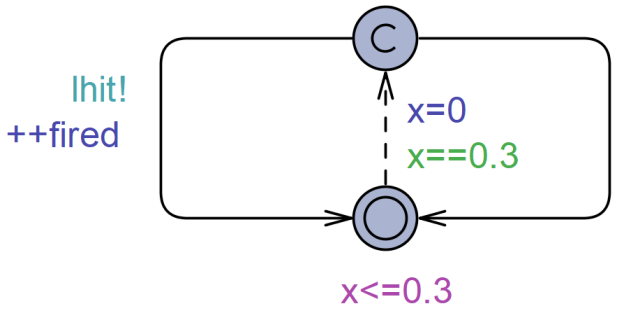
AALBORG UNIVERSITET

Optimal Player

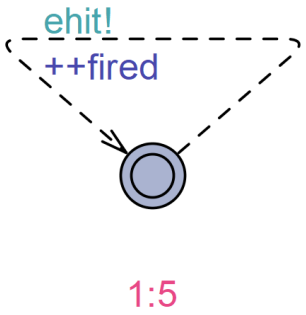
Ball



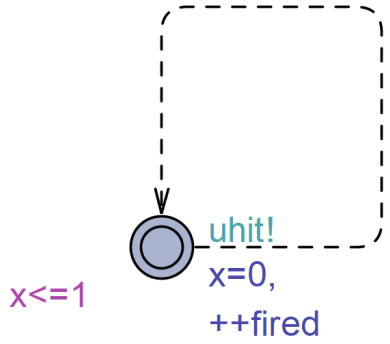
Learner Player



Expo Player

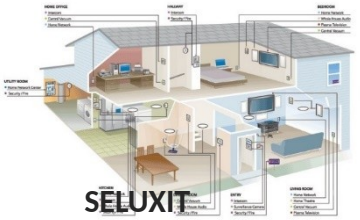


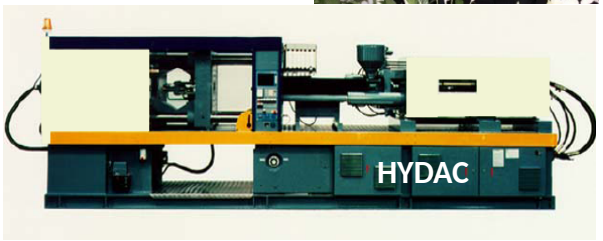
Uni Player



More Synthesis …

- Zone-based climate control pig-stables
- Profit-optimal, energy-aware schedules for satelittes
- Personalized light control in home automation
- Energy- and comfort-optimal floor heating
- **Safe** and energy **optimal** control of hydralic pumps
- **Safe** and **optimal** adaptive cruise control
- Intelligent Traffic Control





LASSO
Learning, Analysis, SynthesiS and Optimization of Cyber-Physical Systems

Conclusion

- Verification of learned strategy (zonification)
- Full correctness of hybrid control obtained from discretization.
  - Verification using SpaceEX;
- From optimal strategy of bounded horizon to optimal infinite strategy.
- Strategies: complexity (space and time)
- On-line synthesis may be too slow for some application
  - Satellites > Floor-heating > Traffic > Power Electronics
  - Learn Neural Network representation of optimal strategy

Stochastic Hybrid Games

- $\mathcal{G} = (C, U, X, \mathcal{F}, \delta, P)$

- Controllable modes $C = \{c_1, \ldots, c_m\}$
- Environment modes $U = \{u_1, \ldots, u_k\}$
- Continuous real-valued variables $X = \{x_1, \ldots, x_n\}$
- Flow functions

$$\mathcal{F}_{c,u} : \mathbb{R}_{\geq 0} \times \mathbb{R}^X \to \mathbb{R}^X$$

  for $c \in C, u \in U$.

- Density functions for environment

$$\delta_\gamma : \mathbb{R}_{\geq 0} \times U \to \mathbb{R}_{\geq 0}$$

  where $\gamma = (c, u, v) \in C \times U \times \mathbb{R}^X$ (configuration)

- Period $P \in \mathbb{R}_{\geq 0}$

Strategy $\sigma : C \times U \times \mathbb{R}^X \to C$

- Given configuration then is the controllable mode to use in the next period
- Run under σ:

$$\pi = \gamma_1 :: \mathbf{0} :: \gamma_2 :: \tau_2 :: \gamma_3 :: \tau_3 :: \gamma_4 :: \tau_4 :: \gamma_5 :: \mathbf{0} :: \gamma_6 :: \cdots$$

u u

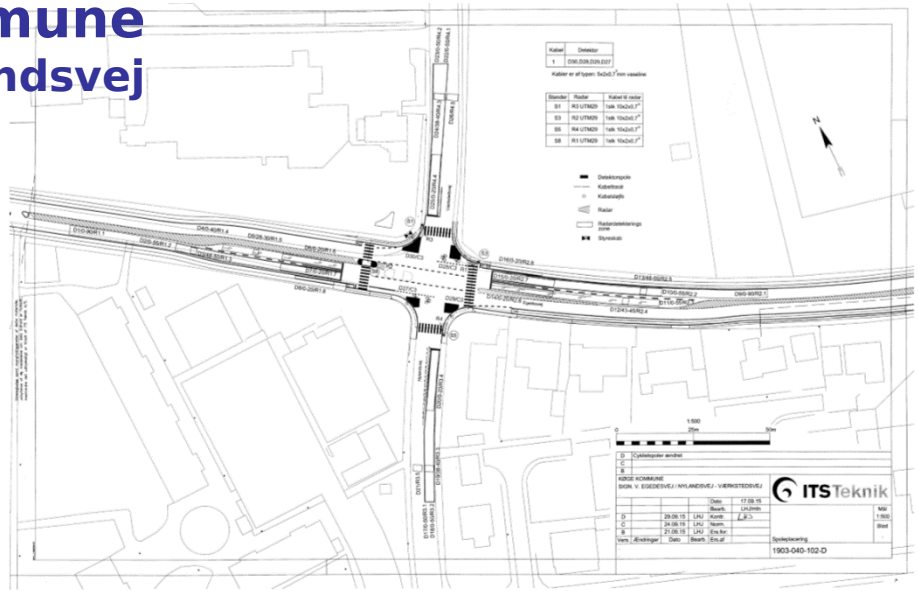
$c = \sigma(\gamma_1)$ $c = \sigma(\gamma_5)$

nP $(n+1)P$

Optimal synthesis problem:
Let D be random variable on runs and H horizon.
Find strategy σ such that $\mathbb{E}_\sigma(D)$ is minimal

Application
Traffic Control

Køge Kommune
Egedesvej/Nylandsvej



AALBORG UNIVERSITET

AALBORG KOMMUNE

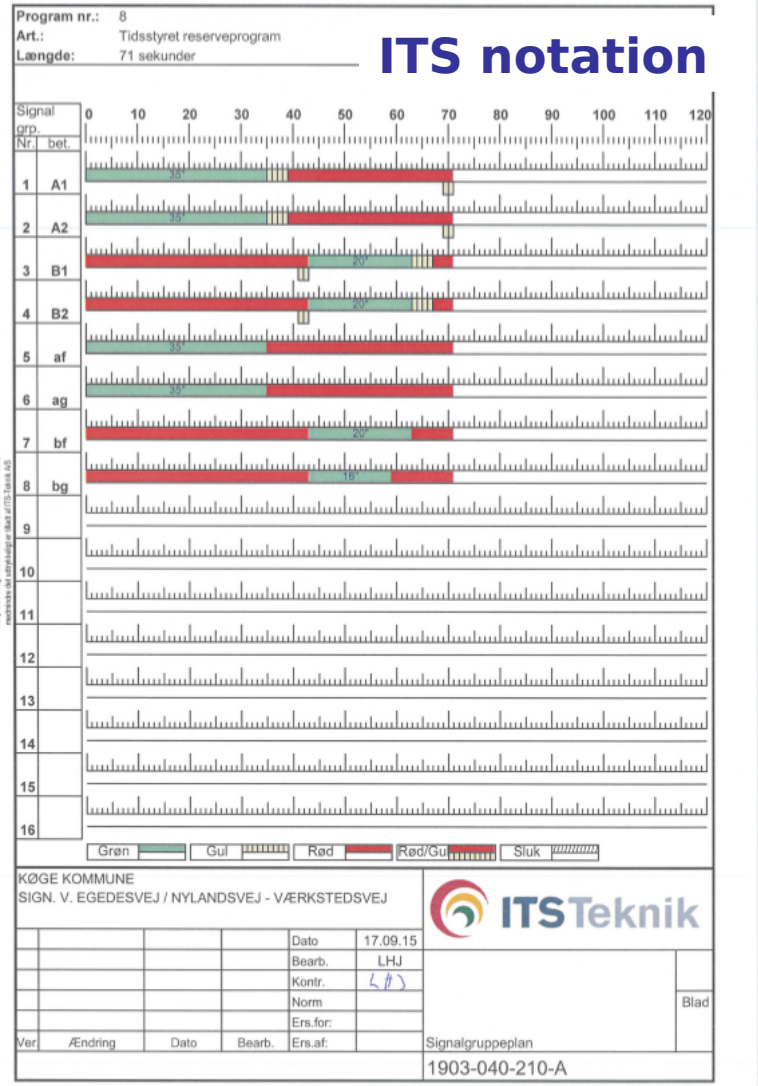
Eriksen, Huang, Kildebogaard, Lahrmann, Larsen, Muniz, Taankvist
ITS Europe 2017 & LiVE@ETAPS17

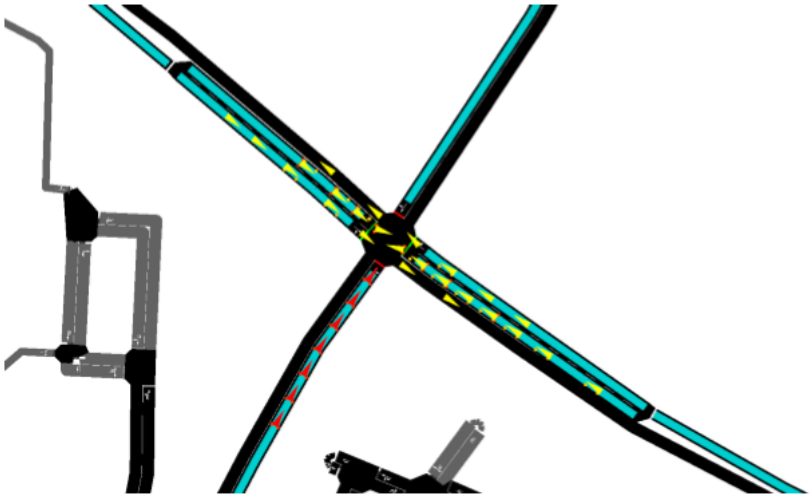
Objective

- Observed a lot of **unnecessary** waiting time for red light in signalized intersections

- Aim:
  Design an intelligent controller that can realize a near optimal signal control using UPPAAL Stratego

Time Triggered Control (Static)

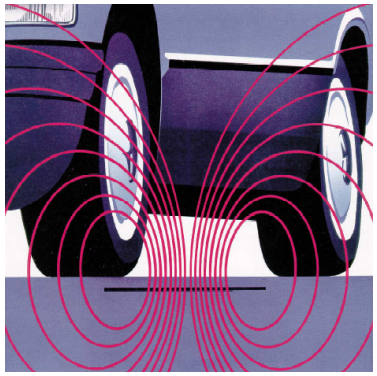




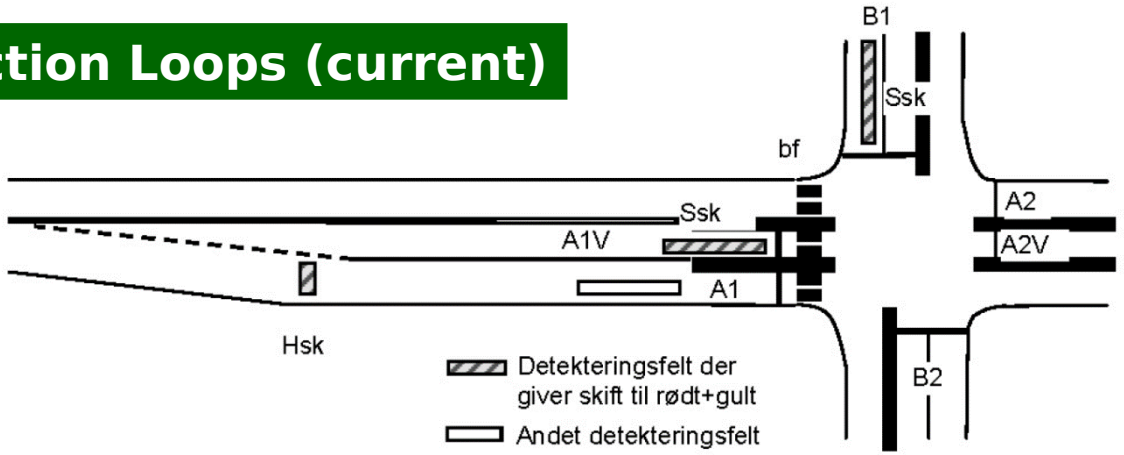
```
<tlLogic id="1693132977" type="static" programID="0">
<phase duration="35" state="rrGGgrrGGg"/>
<phase duration="4" state="rryyyrryyy"/>
<phase duration="4" state="rrrrrrrrrr"/>
<phase duration="20" state="GGrrrGGrrr"/>
<phase duration="4" state="yyrrryyrrr"/>
<phase duration="4" state="rrrrrrrrrr"/>
</tlLogic>
```

SUMO encoding

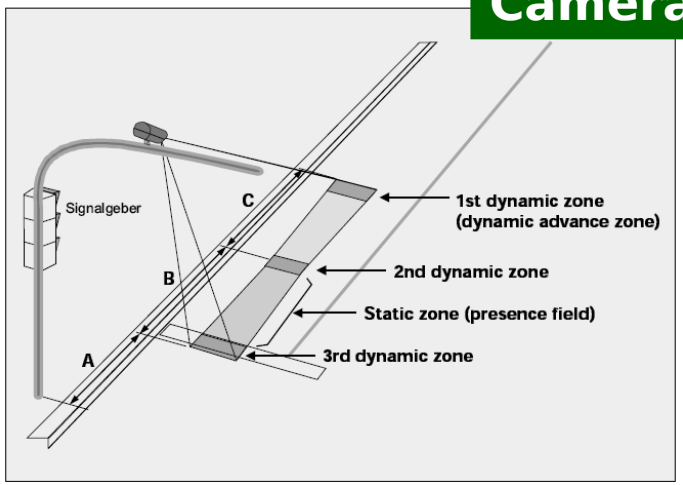
Induction Loops



Induction Loops (current)

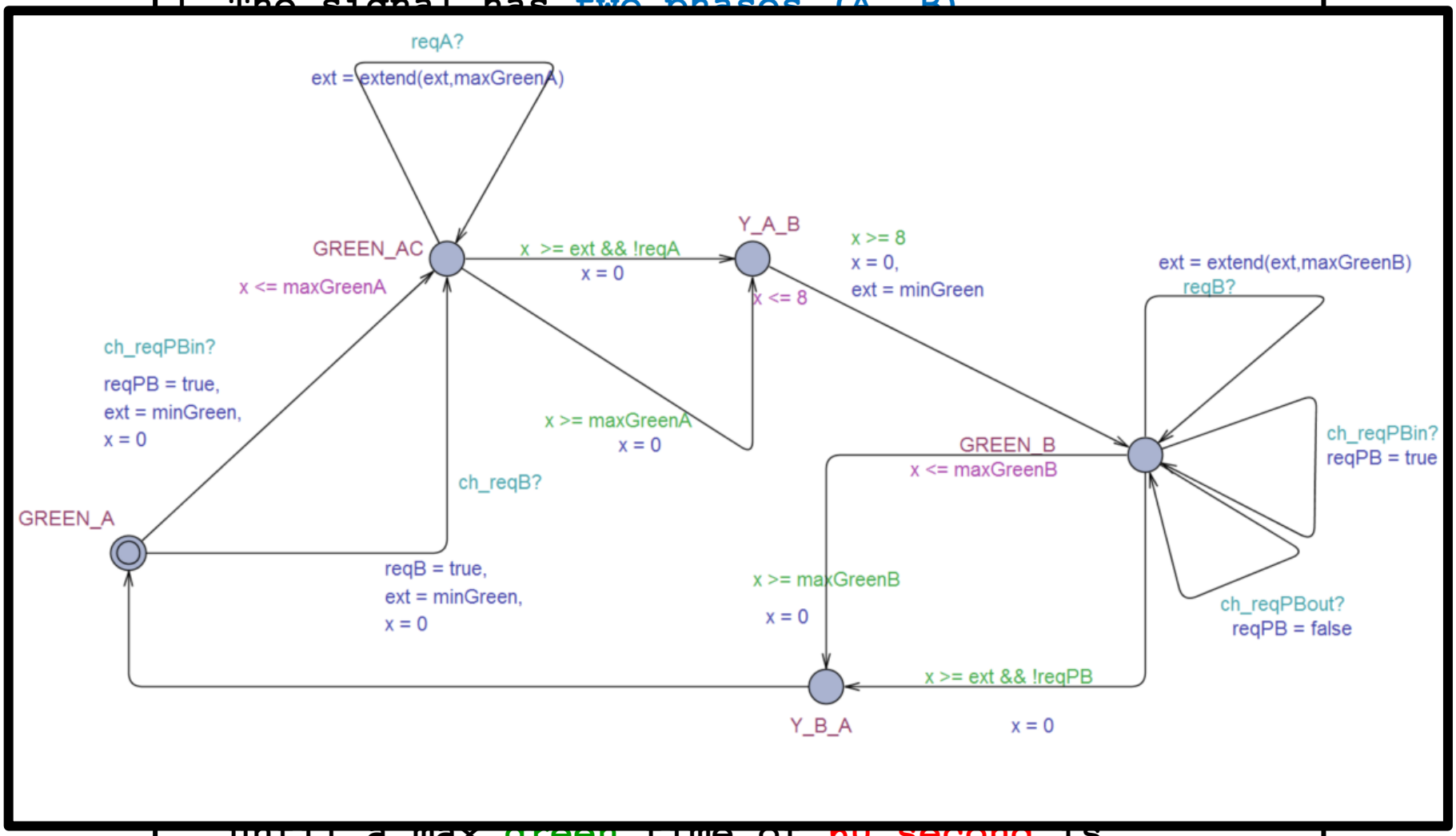


Camera (coming soon)



Detection fields

Traffic Control (Induction Loop)



until a max green time of 60 second is reached.

Learn Lib

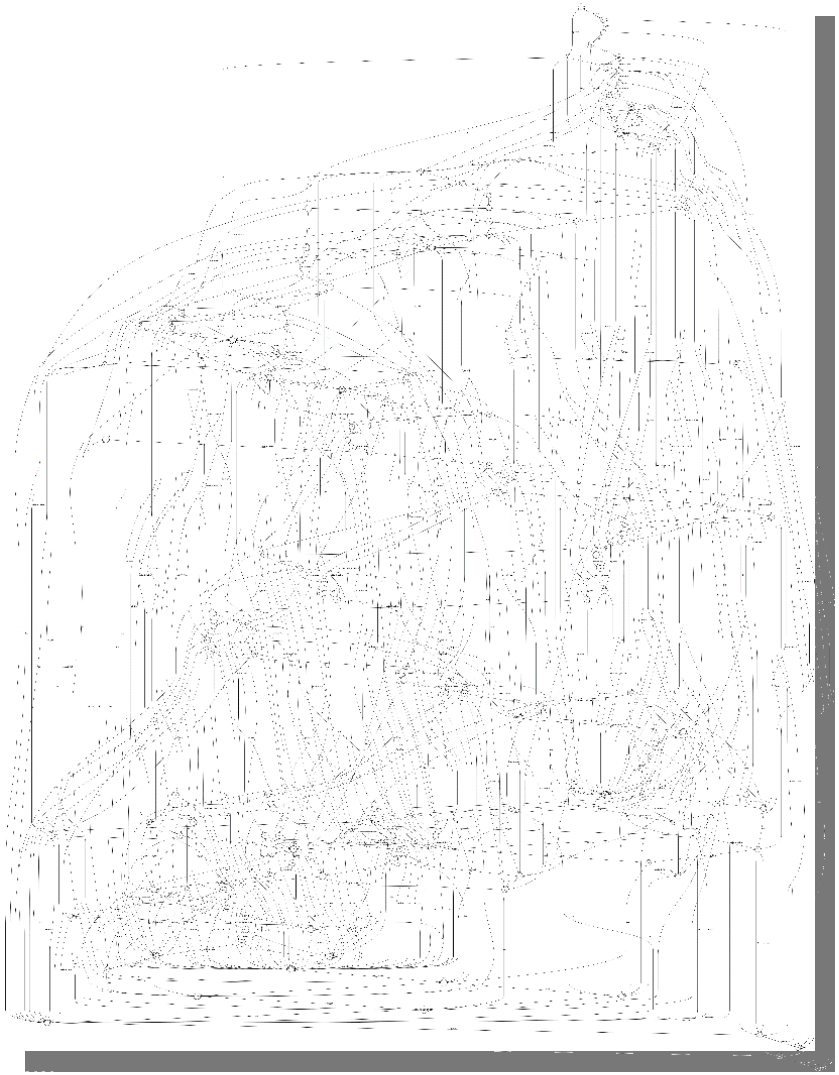
Angluin L* Algorithm

Complexity Simulation:

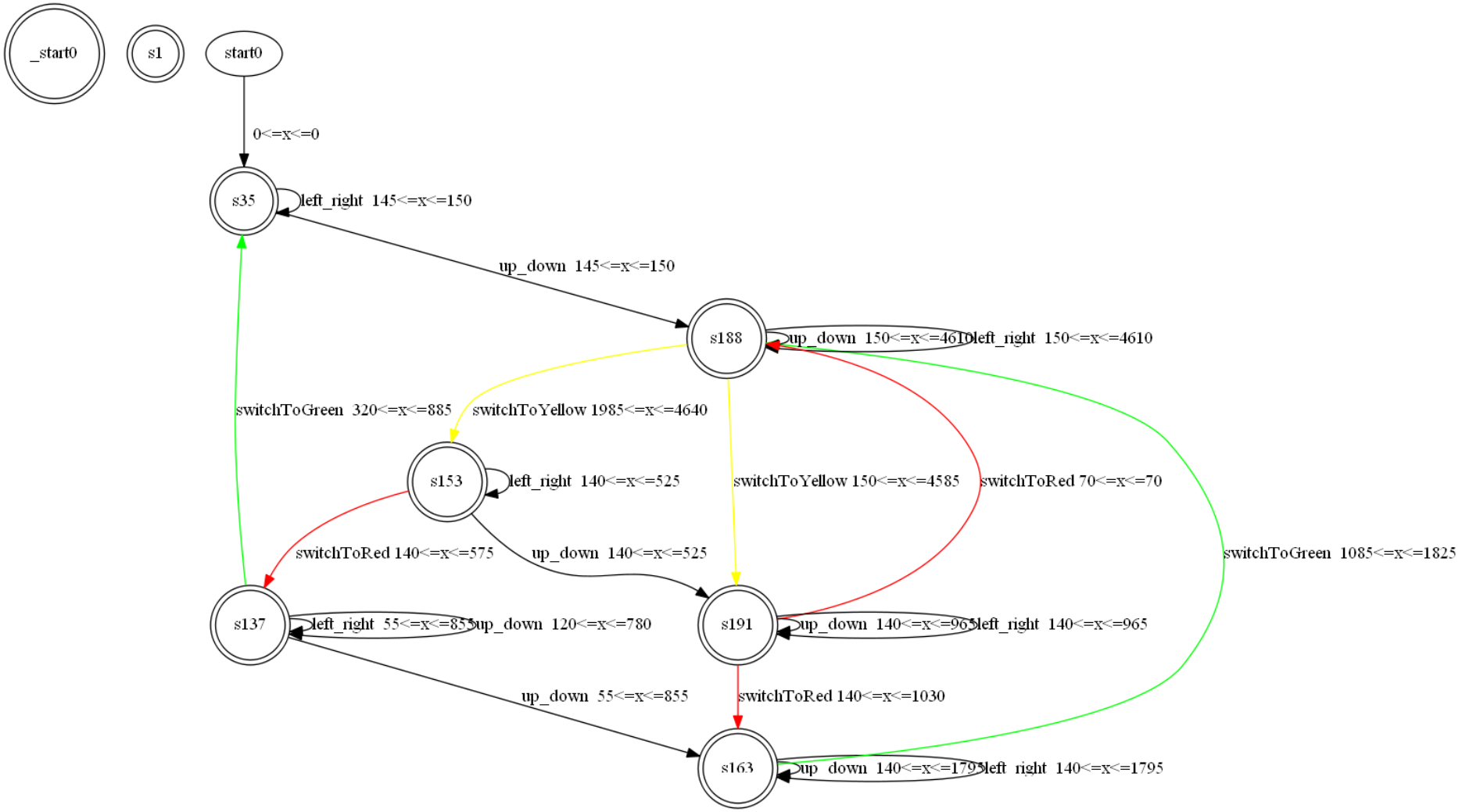
Reach depth 15 in 2 days
Over 100 000 simulations
~1 second per simulation

Complexity Real Execution:

Only estimations yet
Still over 100 000 runs needed
~2-5 minutes per run
(without preparation)
About a year (347 days)
for 5 minute runs



Reduced


_start0
s1
start0
0<=x<=0
s35
left_right 145<=x<=150
up_down 145<=x<=150
s188
up_down 150<=x<=4610
left_right 150<=x<=4610
switchToGreen 320<=x<=885
switchToYellow 1985<=x<=4640
s153
left_right 140<=x<=525
switchToYellow 150<=x<=4585
switchToRed 70<=x<=70
switchToRed 140<=x<=575
up_down 140<=x<=525
switchToGreen 1085<=x<=1825
s137
left_right 55<=x<=855
up_down 120<=x<=780
s191
up_down 140<=x<=965
left_right 140<=x<=965
up_down 55<=x<=855
switchToRed 140<=x<=1030
s163
up_down 140<=x<=1795
left_right 140<=x<=1795

Stratego Controller (Cameras)

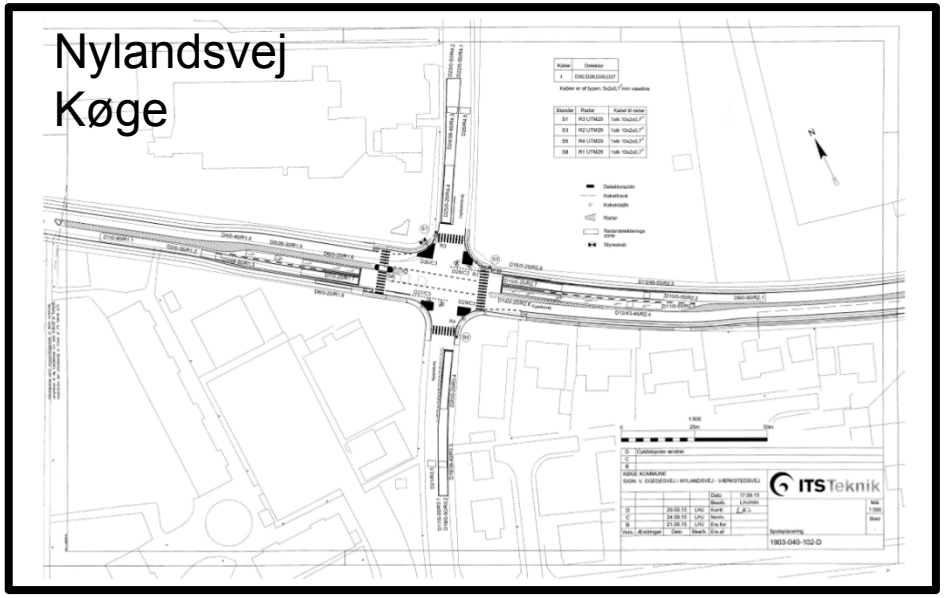
Number of cars waiting in each lane *(full information)*

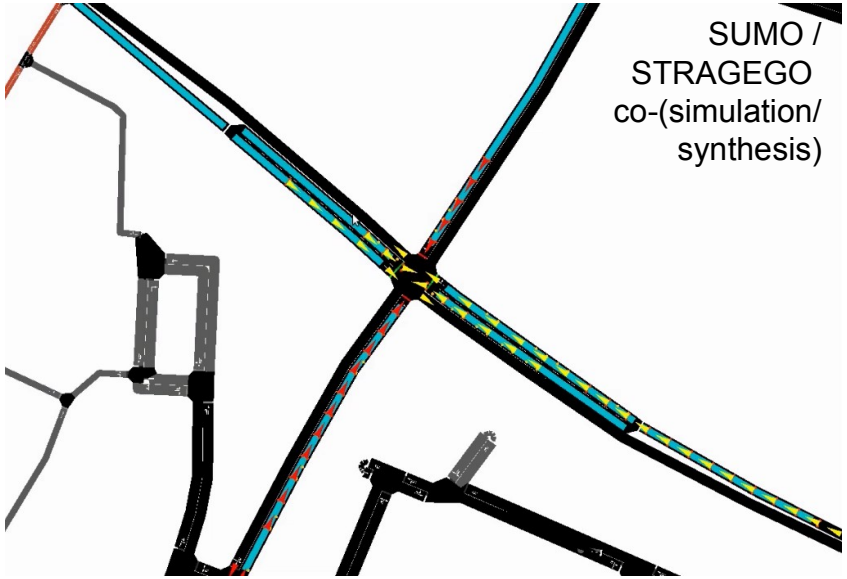
1: Every 5 to 8 sec read sensor data
2: **if** Traffic Light in yellow phase **then**
3: Run UPPAAL STRATEGO – decide next green phase
4: **else if** Traffic Light in green phase **then**
5: Run UPPAAL STRATEGO – extend green phase or go to yellow
6: **end if**

ONLINE Synthesis

- Identify optimal strategy up to horizon H=90sec.
  - Strategy changes phase (at least 5 sec).
    - Modelling of stochastic arrival of cars in different directions (from 60-850 cars/hour)
- Minimize waiting time or jam (# of waiting >2sec)

Preliminary Results



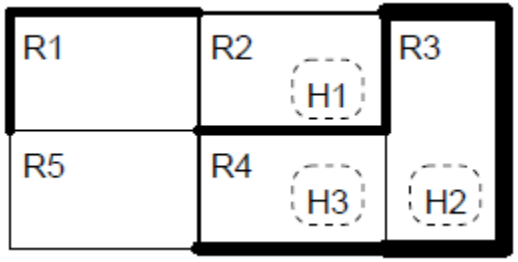


Scenario	Static		Loop Induction		Stratego		Imp W time over LI %
	Jam Km	W time s	Jam Km	W time s	Jam Km	W time s	
MAX	1451	191990	1185	157200	551	73001	53.5%
MID	456	60362	369	48936	331	43878	[illegible]
LOW	138	18425	139	18566	101	13451	27.5%

Scenario:
2 hours traffic

VISSEM / UPPAAL

Energy Aware Buildings

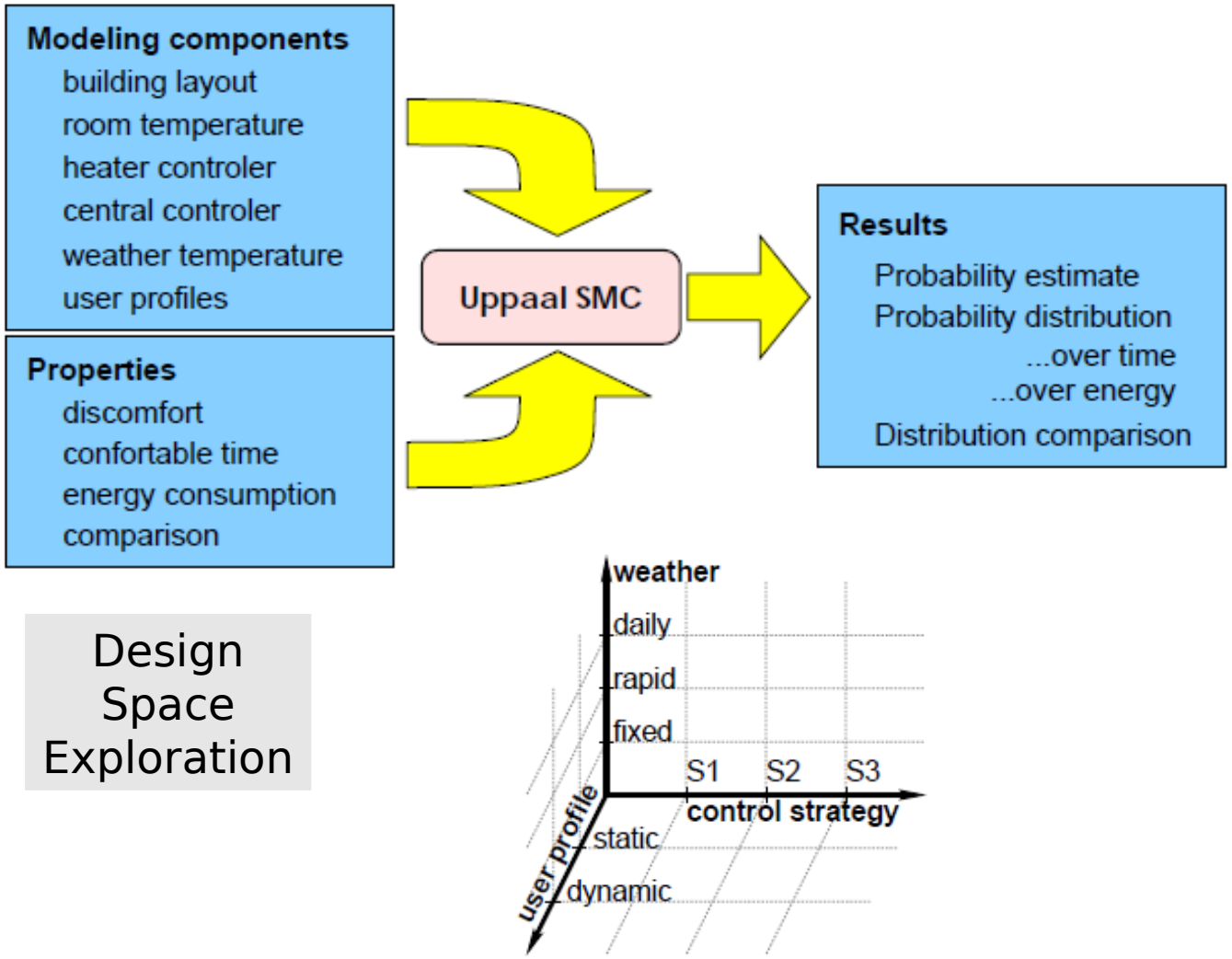


Fehnker, Ivancic.
Benchmarks for Hybrid Systems Verification.
HSCC04

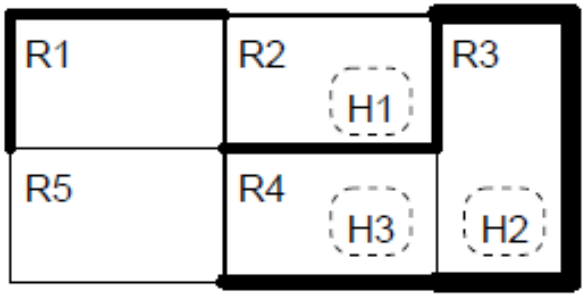
With Alexandre David,
Dehui Du
Marius Mikucionis
Arne Skou

AALBORG UNIVERSITET

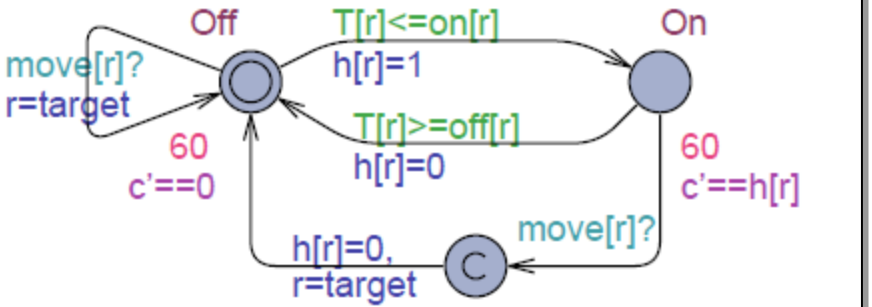
Framework


Modeling components
building layout
room temperature
heater controler
central controler
weather temperature
user profiles
Properties
discomfort
confortable time
energy consumption
comparison
Uppaal SMC
Results
Probability estimate
Probability distribution
...over time
...over energy
Distribution comparison
Design Space Exploration
weather
daily
rapid
fixed
S1
S2
S3
control strategy
user profile
static
dynamic

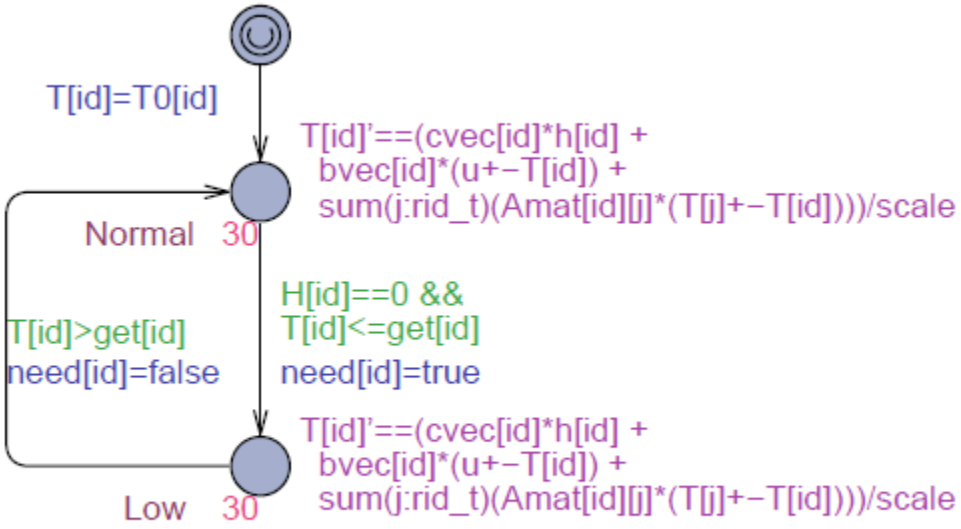
Rooms & Heaters – MODELS



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



(b) Template for `Heater` control.



(a) Template for `Room` temperature.

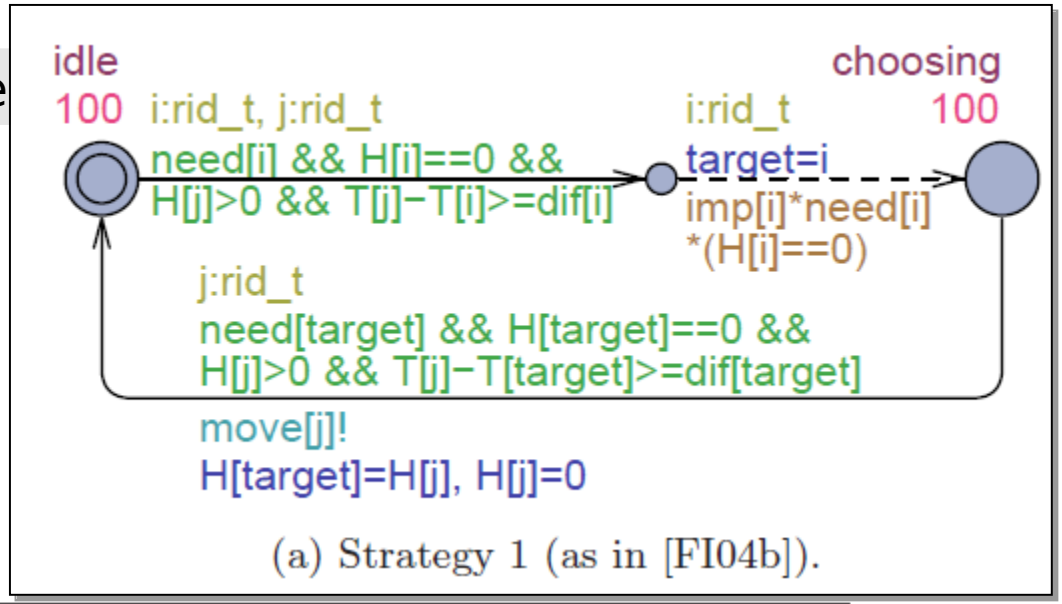
$$\begin{pmatrix} 10.0 & 7.0 & 10.0 & 11.0 & 9.0 \end{pmatrix}$$

(d) Heating vector c.

Control Strategies – MODELS

room	1	2	3	4	5
off	21	21	21	21	21
on	19	19	19	19	19
get	16	17	18	17	16
low	15	16	16	16	15
dif	1.0	1.0	1.0	1.0	1.0
imp	1	30	2	3	4
pow	5	5	5	5	5

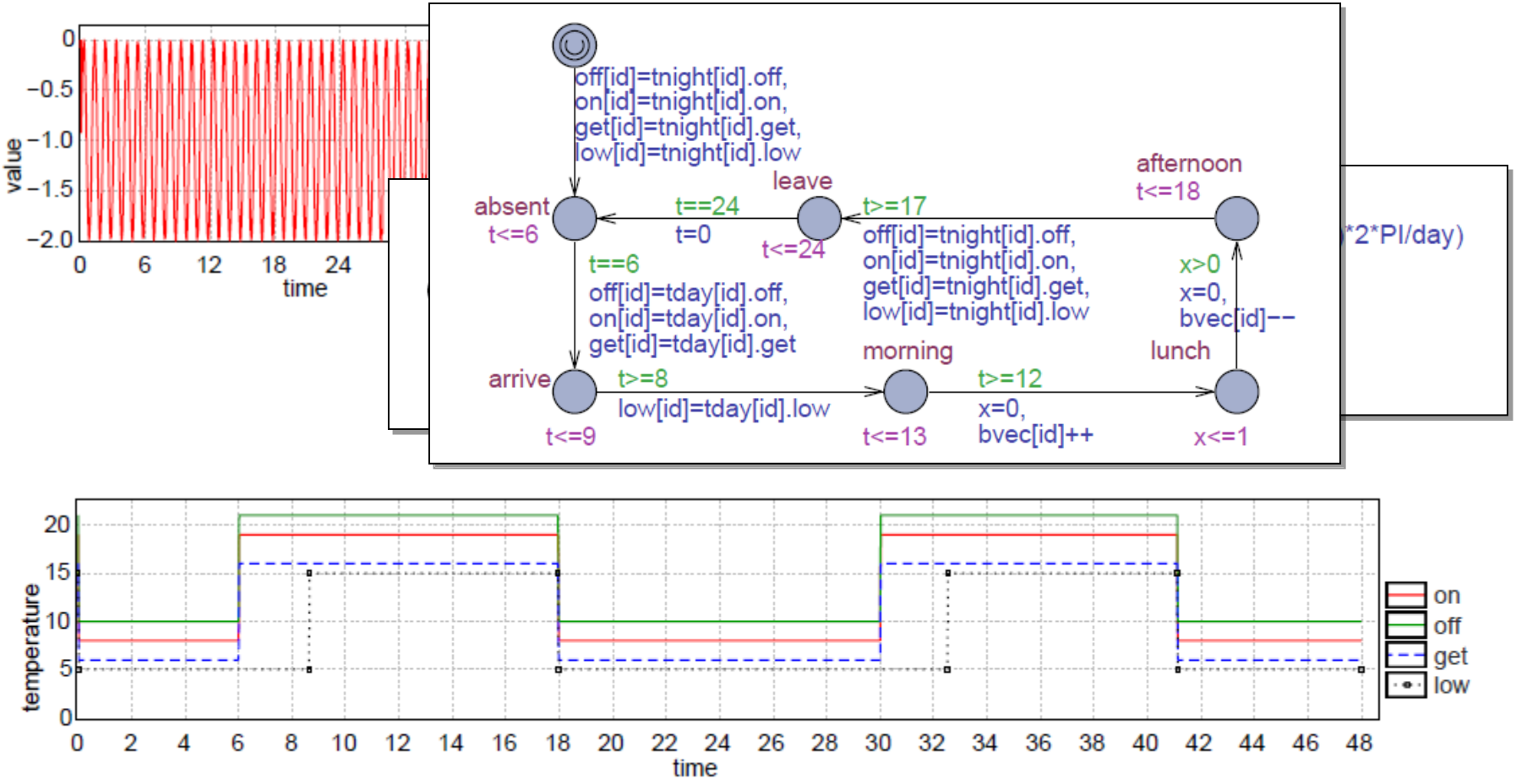
Te



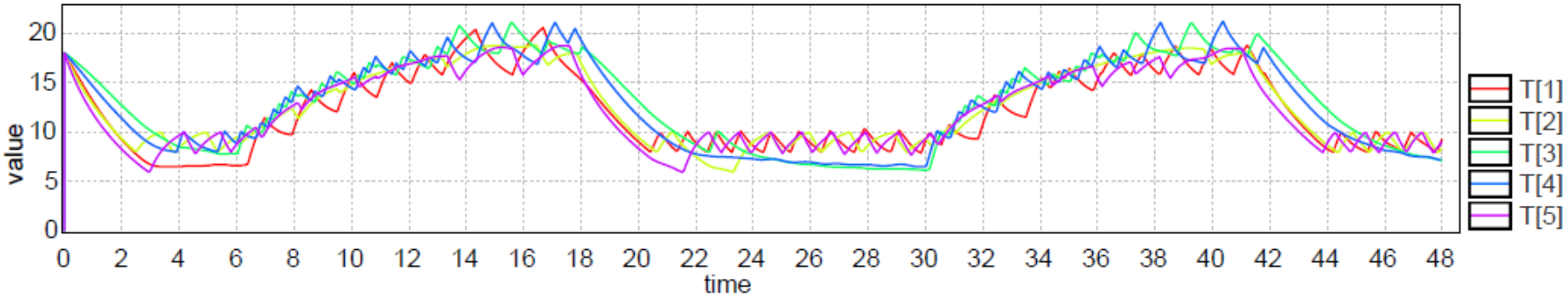
(a) Strategy 1 (as in [FI04b]).

Strategy 1	Strategy 2	Strategy 3
room i has no heater	room i has no heater	room i has no heater
room j has a heater	room j has a heater	room j has a heater
temperature $T_i \leqslant get_i$	temperature $T_i \leqslant get_i$	temperature $T_i \leqslant get_i$
difference $T_j - T_i \geqslant dif_i$	threshold $T_j \geqslant get_j$	threshold $T_j \geqslant on_j$

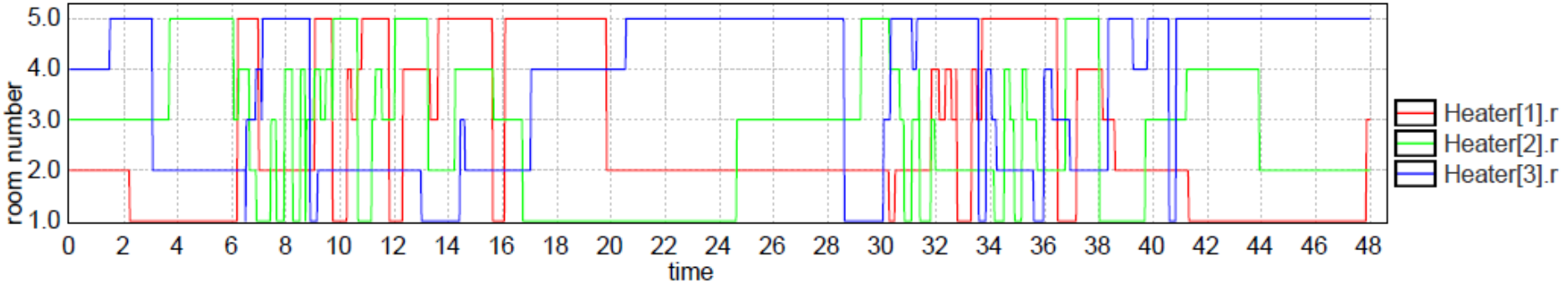
Weather & User Profile – MODELS



Results – 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 }

Results – Discomfort

Pr[<=2*day](<> time>0
&& Monitor.Discomfort)

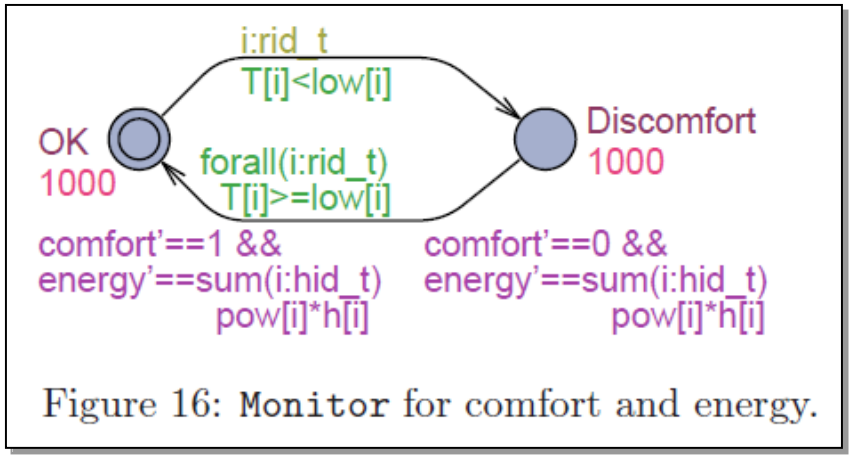
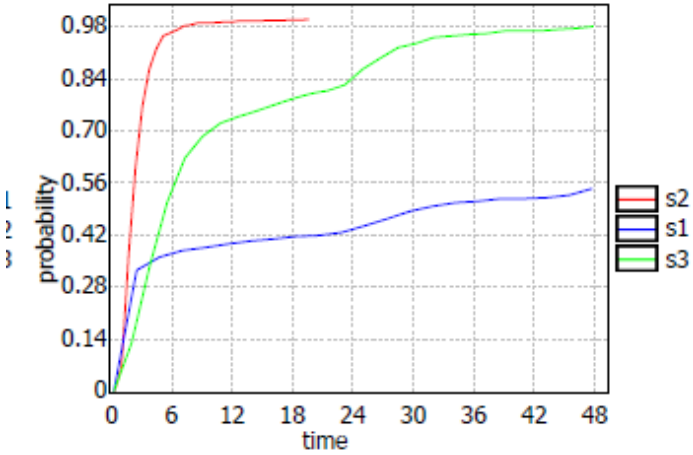
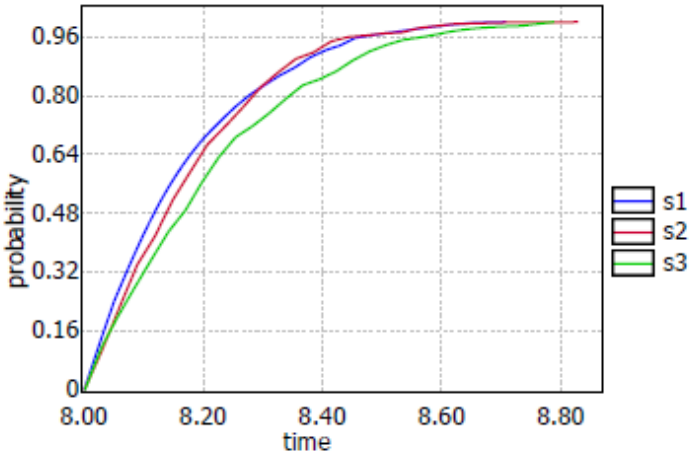


Figure 16: `Monitor` for comfort and energy.



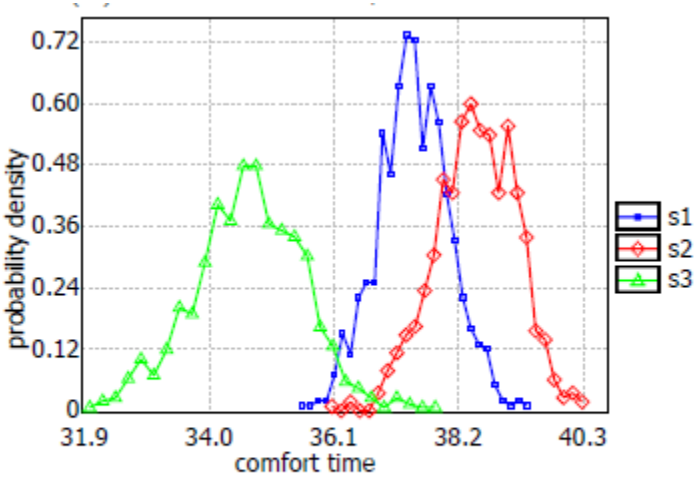
(c) Daily weather, Static user.



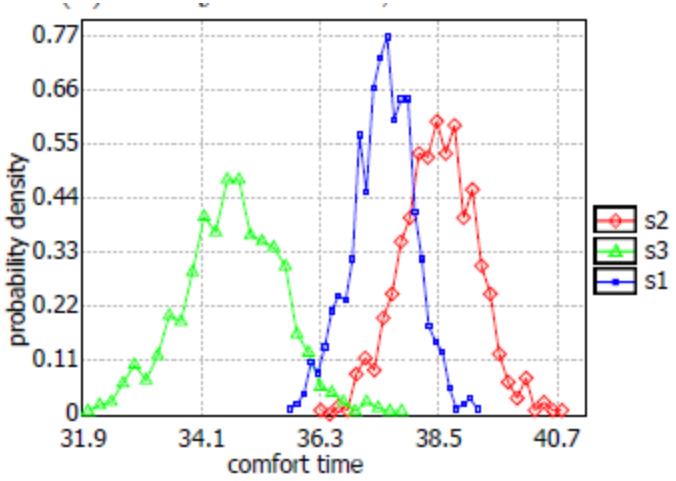
(f) Daily weather, Dynamic user.

Results – Comfort

Pr[comfort<=2*day] (<> time>=2*day)



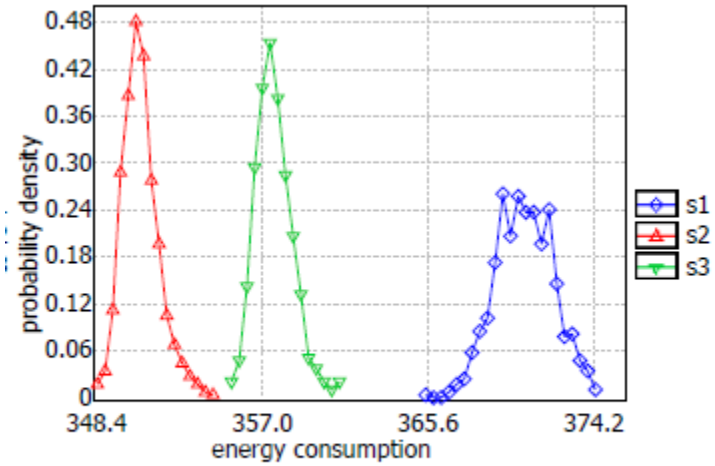
(d) Flat weather, Dynamic user.



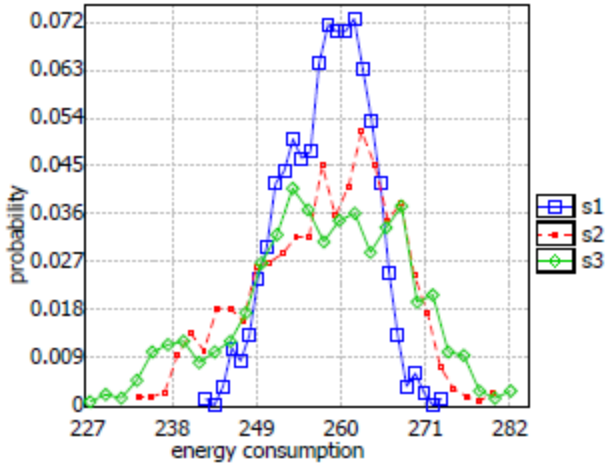
(f) Daily weather, Dynamic user.

Results – Energy

Pr[Monitor.energy<=1000000](<> time>=2*day)



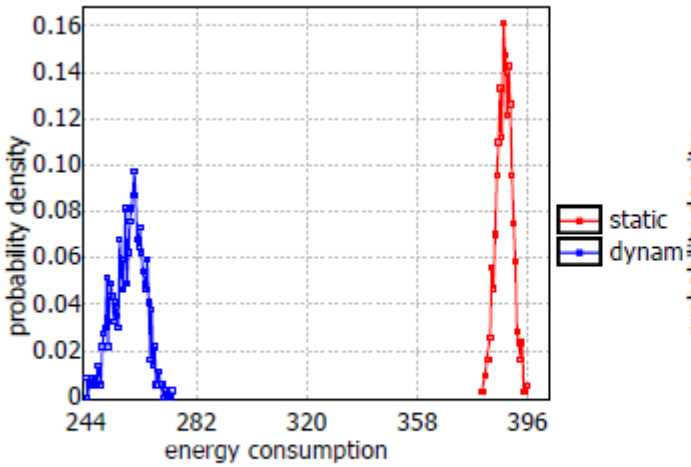
(c) Dail weather, Static user.



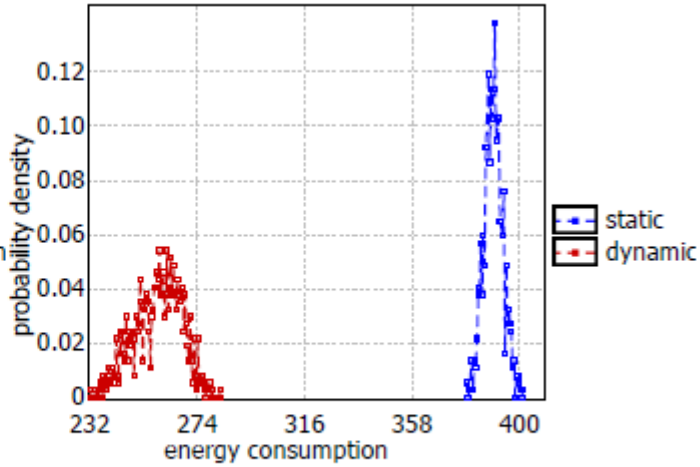
(f) Daily weather, Dynamic user.

Result – User Profile

Pr[Monitor.energy<=1000000](<> time>=2*day)

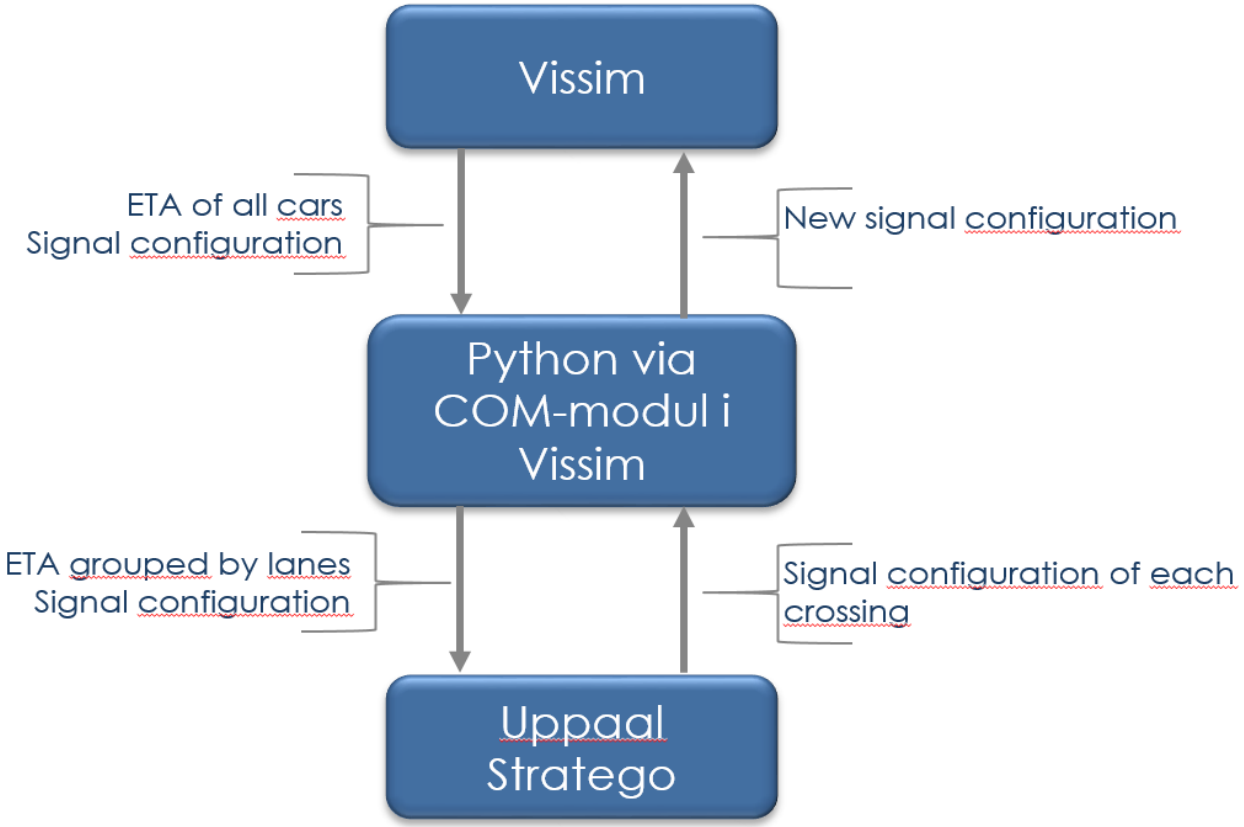


(a) Strategy 1.



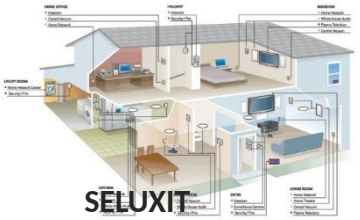
(b) Strategy 2.

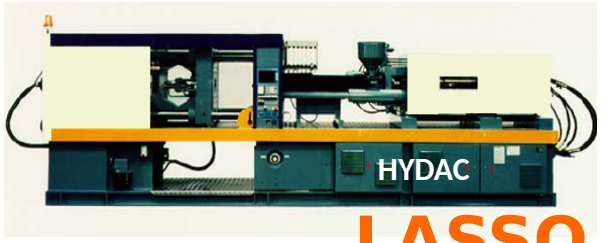
VISSEM / UPPAAL



More Practical Synthesis …

- Zone-based climate control pig-stables
- Profit-optimal, minimal-wear and energy-aware schedules for satelittes
- Personalized light control in home automation
- Energy- and comfort-optimal floor heating
- Safe and energy optimal control of hydralic pumps
- Safe and optimal car maneuvers





LASSO

Learning, Analysis, SynthesiS and Optimization of Cyber-Physical Systems

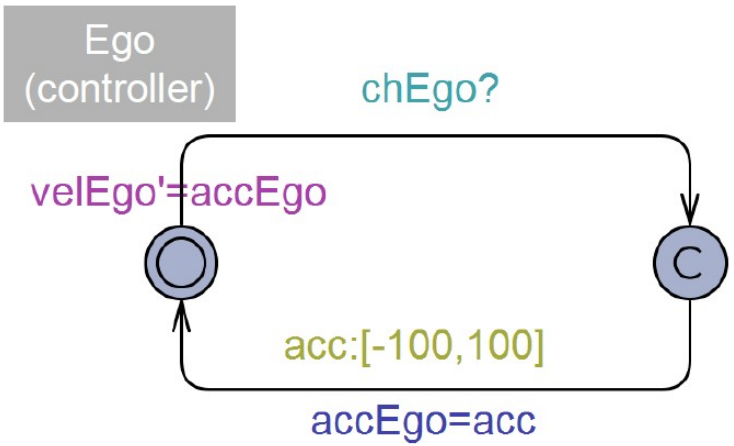
Application
Safe & Adaptive Cruise Control

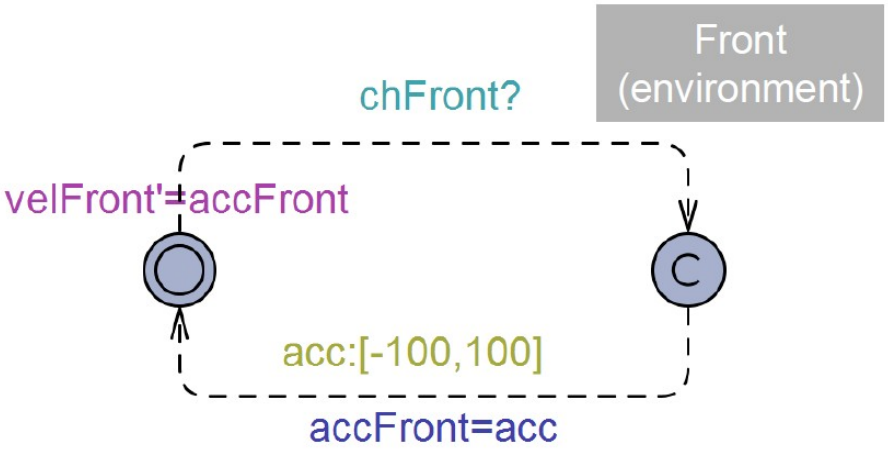
Q1: Find a safety **strategy** for ***Ego*** such no crash will ever occur no matter what ***Front*** is doing.

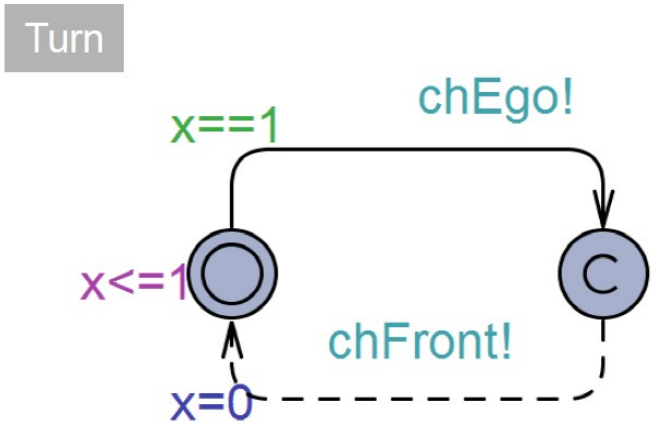
Q2: Find the **optimal sub-strategy** that will allow *Ego* to go as far as possible (without overtaking).

Larsen, Mikucionis, Taankvist: Safe and Optimal Adaptive Cruise Control. Correct System Design 2015

Two Player Game (simplified)







distance' == (velEgo-velFront) &&
D' == distance

Q: find strategy for Ego

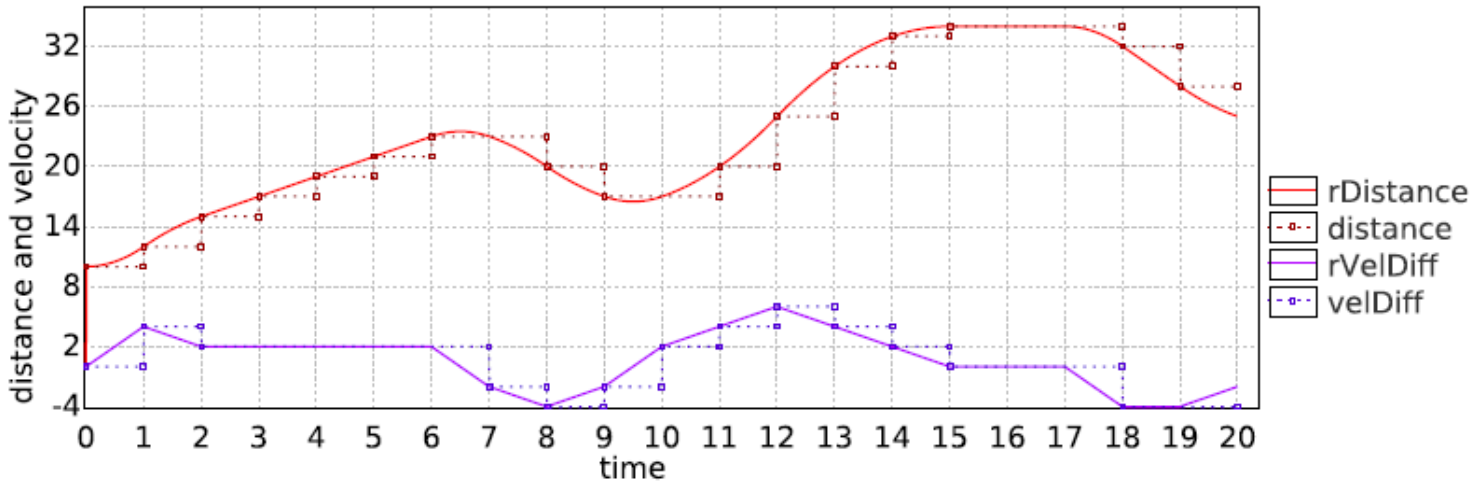
Discretization

Discrete

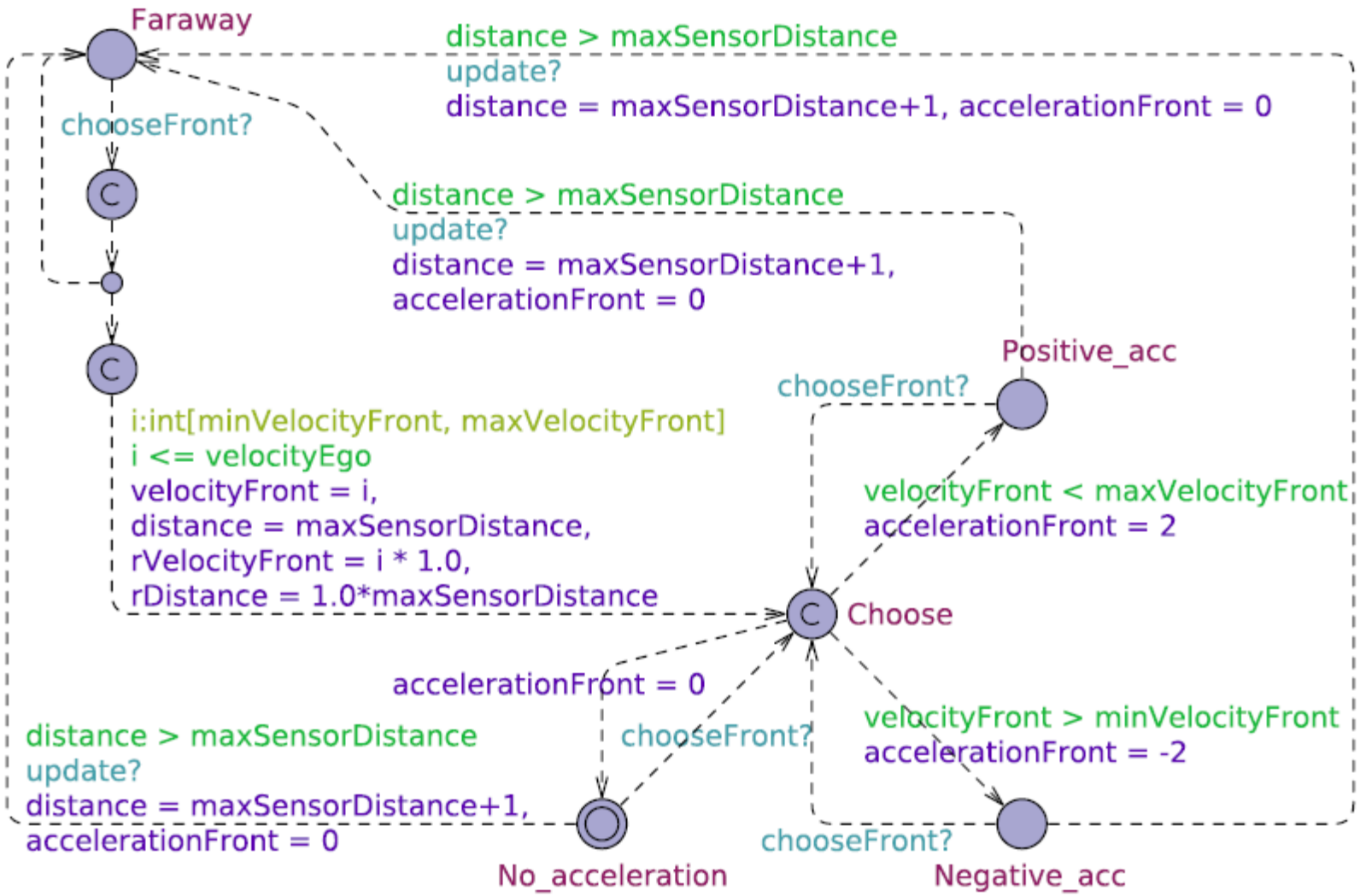
```
void updateDiscrete(){
    int oldVel, newVel;
    oldVel = velocityFront - velocityEgo;
    velocityEgo = velocityEgo + accelerationEgo;
    velocityFront = velocityFront + accelerationFront;
    newVel = velocityFront - velocityEgo;
    if (distance > maxSensorDistance) {
        distance
```

ie

```
double distanceRate(double velFront  double velEgo
```



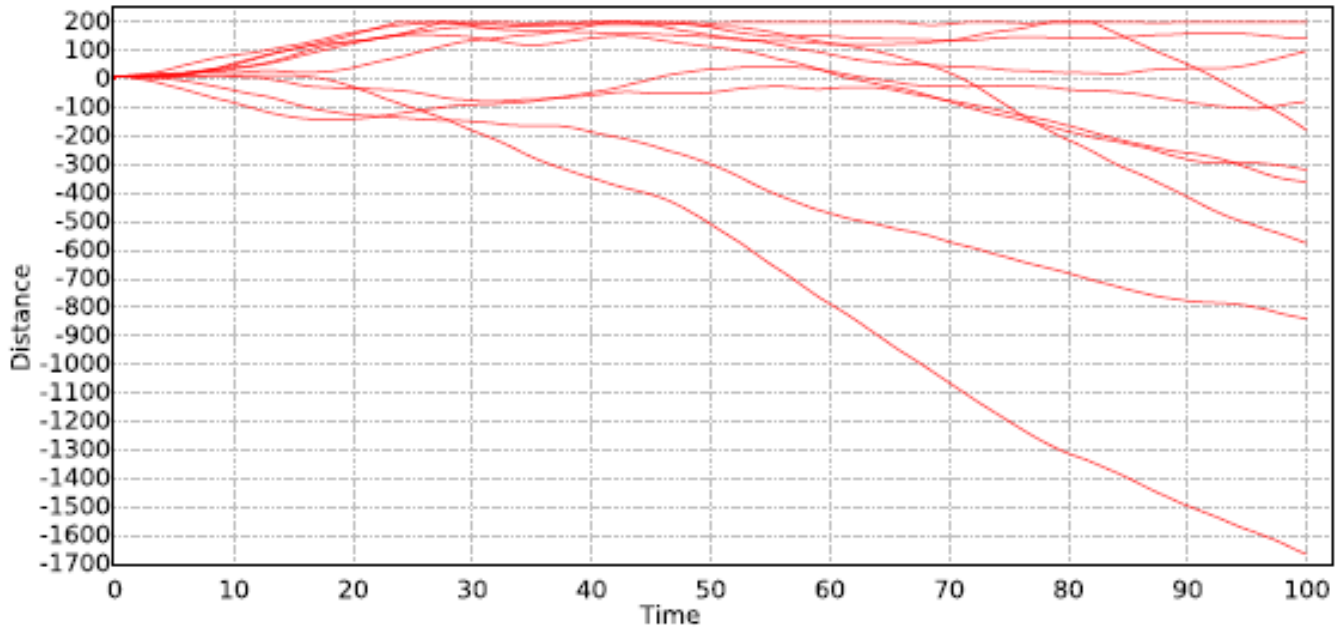
Front (complete)



No Strategy

```
Pr[<=100] (<> distance <= 5)
```

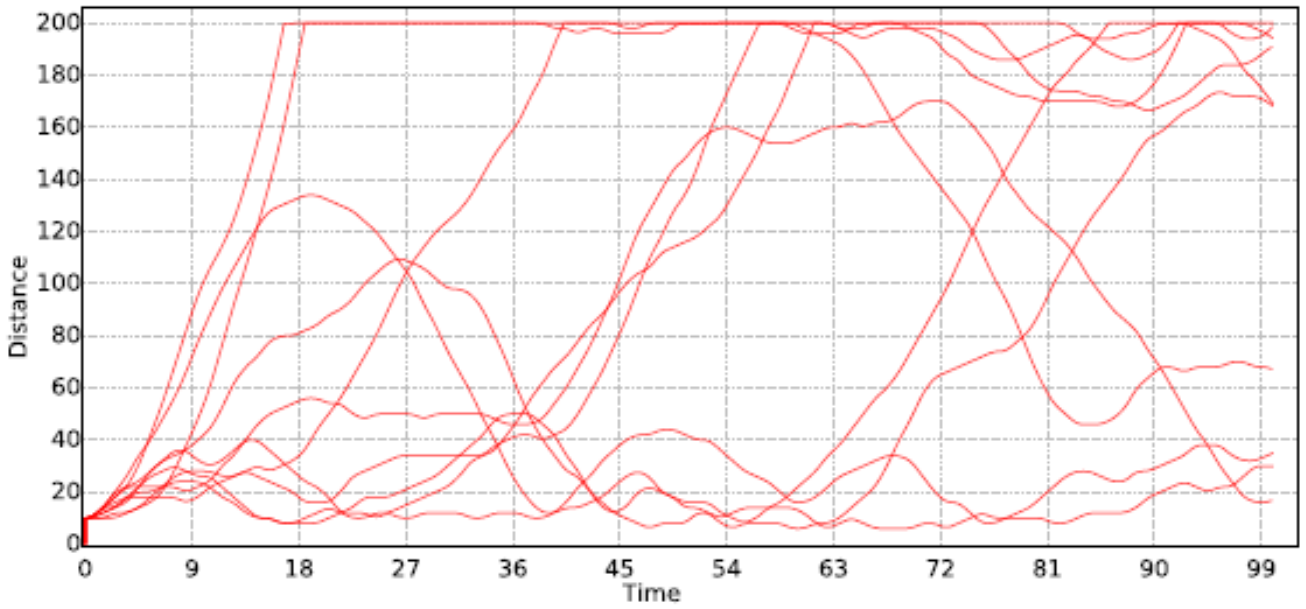
```
A[] distance > 5
```



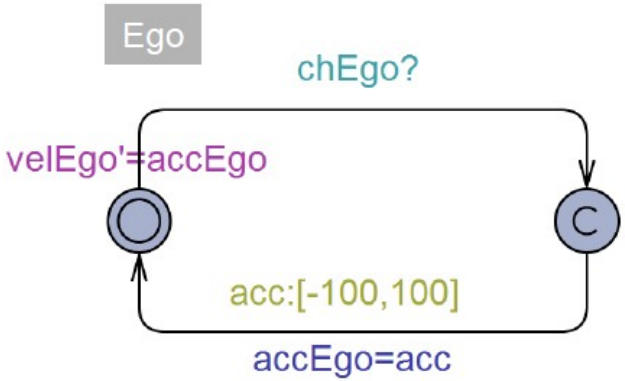
Safety Strategy

```
strategy safe = control: A[] distance > 5
```

```
A[] distance > 5 under safe
```



Safety Strategy (Code)



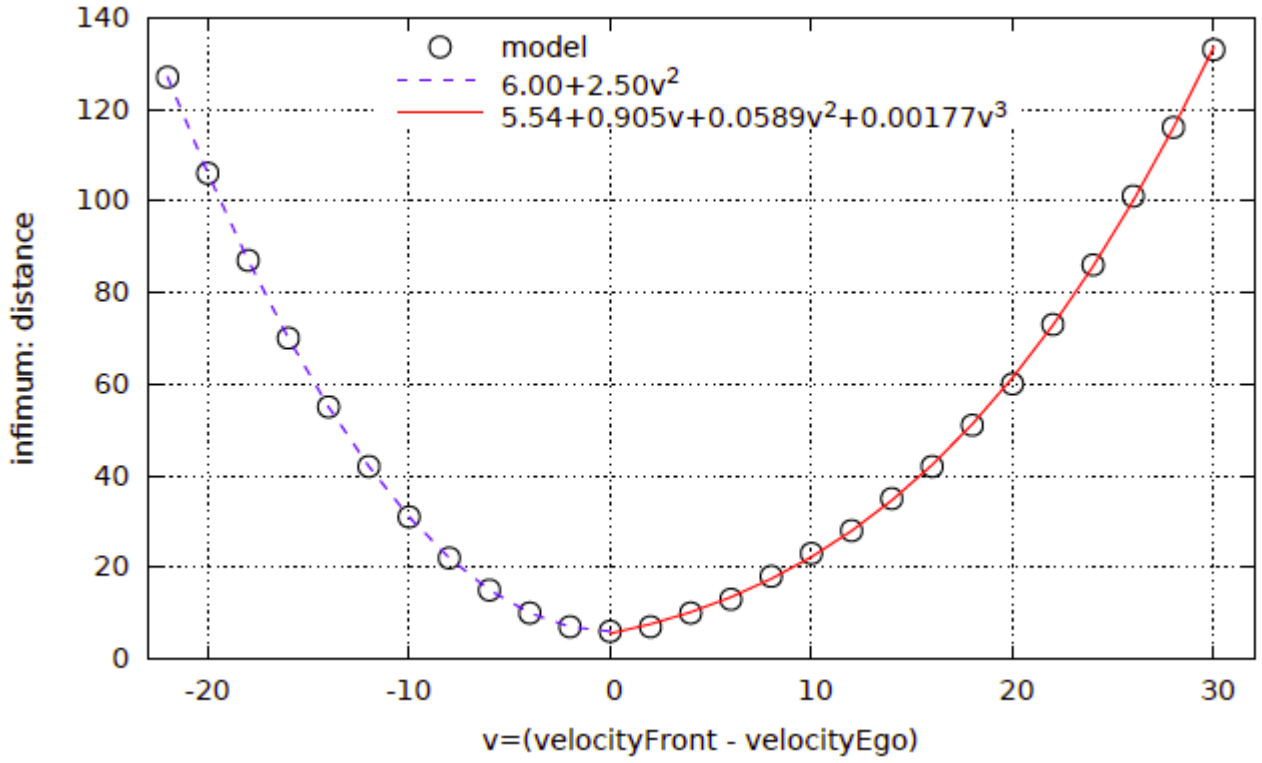
adaptiveCruiseControl - Notepad

File Edit Format View Help

```
State: ( Ego.Negative_acc Front.No_acceleration System.Wait Monitor._id12 ) #action=0
distance=47 velocityEgo=6 accelerationEgo=-2 velocityFront=12 accelerationFront=0
While you are in        (waitTimer<=1), wait.

State: ( Ego.No_acc Front.Positive_acc System.Wait Monitor._id12 ) #action=0 distance=83
velocityEgo=13 accelerationEgo=0 velocityFront=14 accelerationFront=2
While you are in        (waitTimer<=1), wait.

State: ( Ego.Choose Front.No_acceleration System.FrontNext Monitor._id12 ) #action=0
distance=181 velocityEgo=0 accelerationEgo=0 velocityFront=14 accelerationFront=0
When you are in true, take transition Ego.Choose->Ego.No_acc { 1, tau, accelerationEgo := 0
}
When you are in true, take transition Ego.Choose->Ego.Positive_acc { velocityEgo <
maxVelocityEgo, tau, accelerationEgo := 2 }
When you are in true, take transition Ego.Choose->Ego.Negative_acc { velocityEgo >
minVelocityEgo, tau, accelerationEgo := -2 }

State: ( Ego.Negative_acc Front.Choose System.Done Monitor._id12 ) #action=0 distance=199
velocityEgo=7 accelerationEgo=-2 velocityFront=15 accelerationFront=0
While you are in        true, wait.

State: ( Ego.Negative_acc Front.Positive_acc System.Done Monitor._id12 ) #action=0
distance=49 velocityEgo=4 accelerationEgo=-2 velocityFront=14 accelerationFront=2
While you are in        true, wait.

State: ( Ego.Positive_acc Front.Choose System.Done Monitor._id12 ) #action=0 distance=88
velocityEgo=0 accelerationEgo=2 velocityFront=11 accelerationFront=0
While you are in        true, wait.

State: ( Ego.Positive_acc Front.Choose System.Done Monitor._id12 ) #action=0 distance=174
velocityEgo=18 accelerationEgo=2 velocityFront=17 accelerationFront=2
While you are in        true, wait.

State: ( Ego.No_acc Front.Negative_acc System.Done Monitor._id12 ) #action=0 distance=147
```

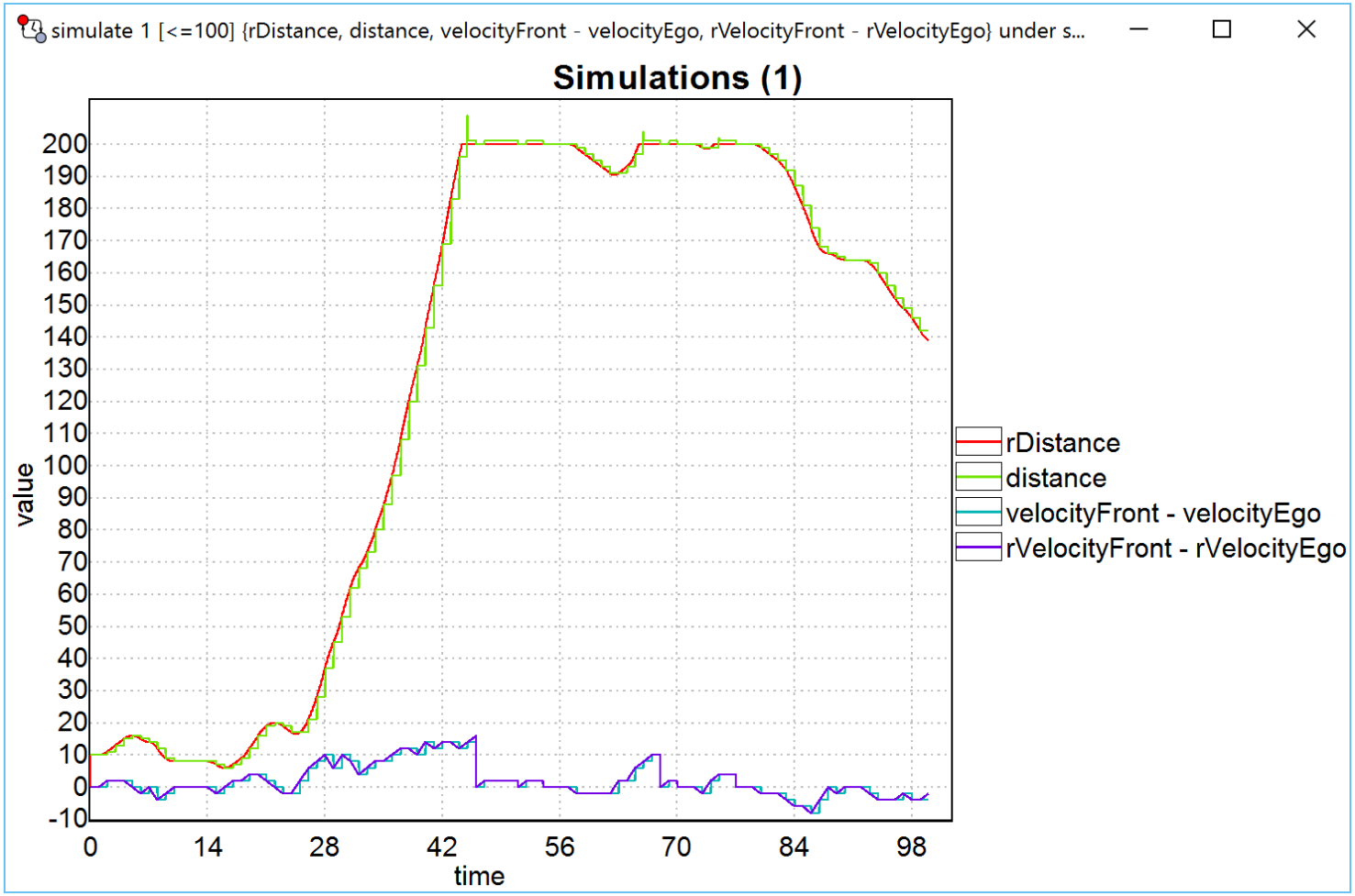
Safety Strategy

```
inf{velosityFront-velosityEgo==v}: distance under safe
```



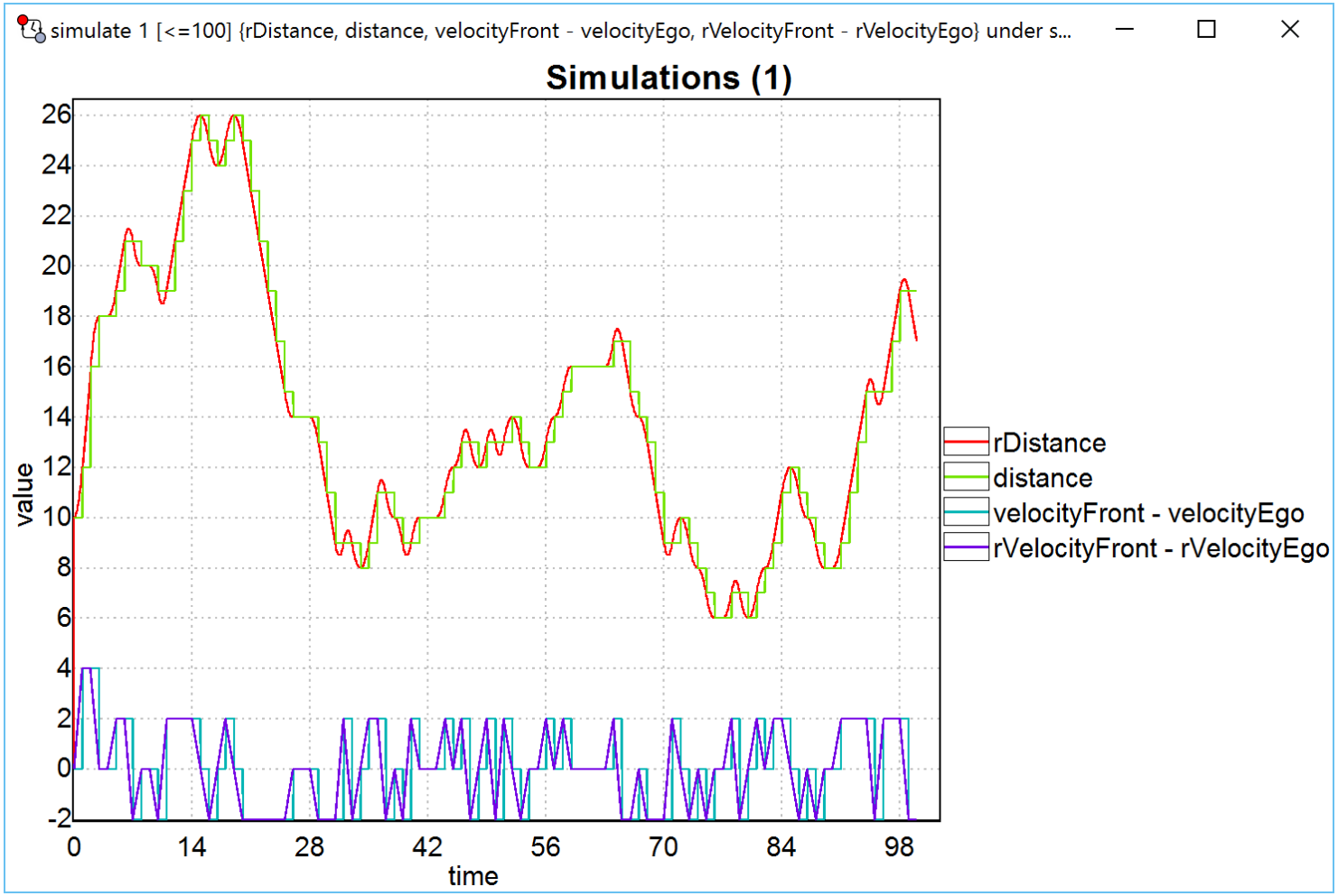
Safe and Optimal Strategy

```
strategy safeFast = minE (D) [<=100]: <> time >= 100 under safe
```



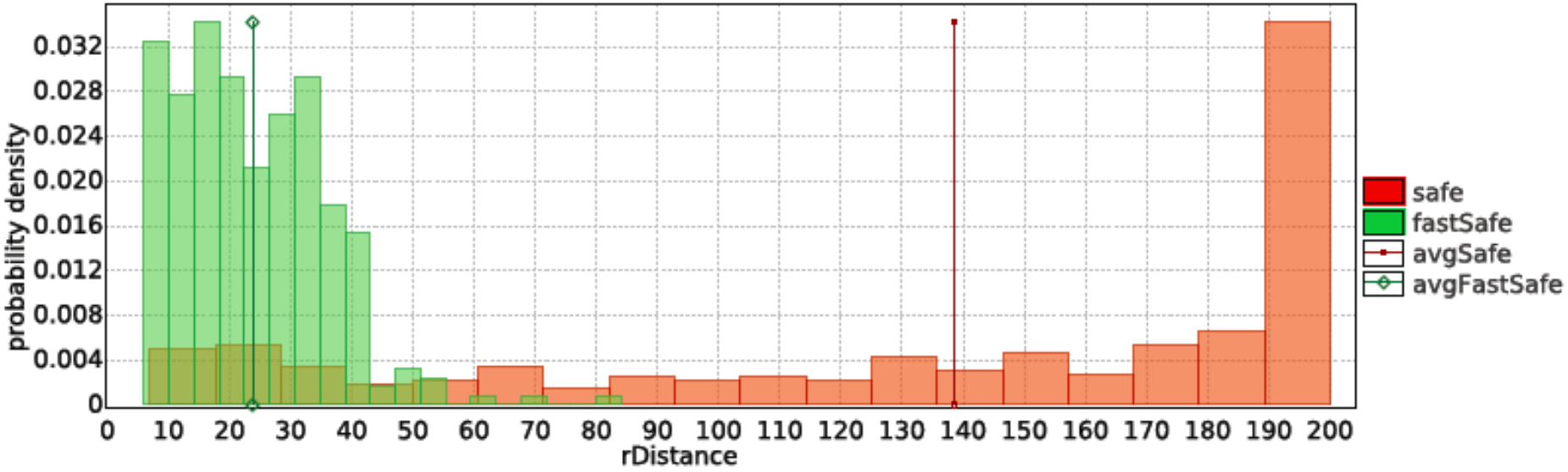
Safe and Optimal Strategy

```
strategy safeFast = minE (D) [<=100]: <> time >= 100 under safe
```



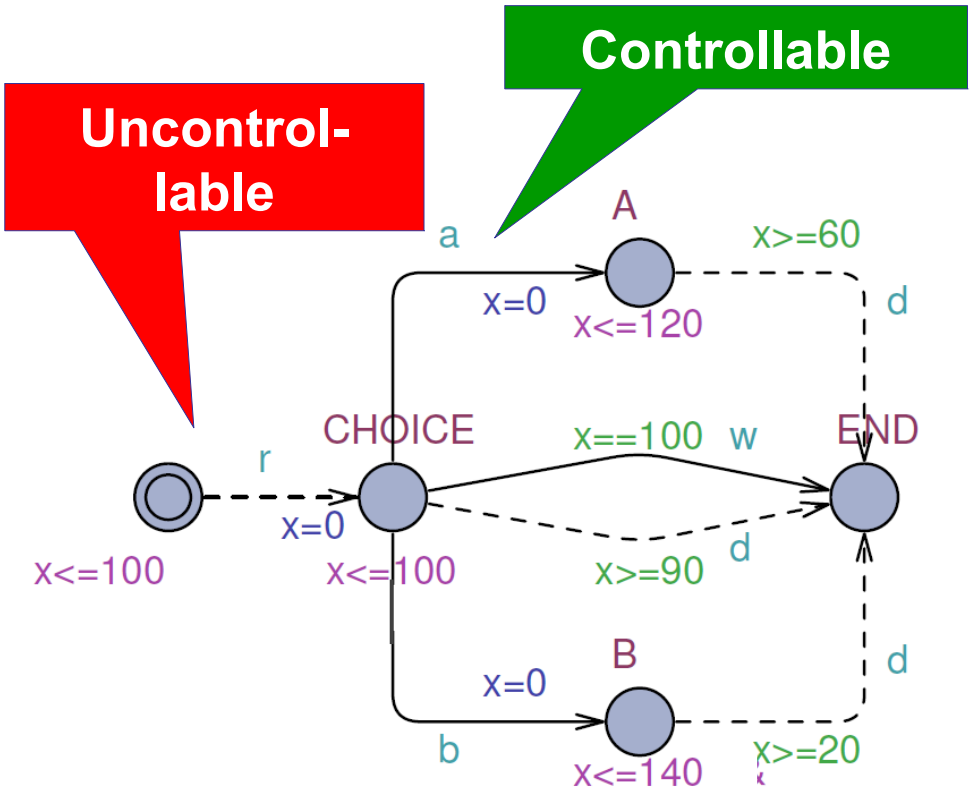
Optimal and Safe Strategy

```
strategy safeFast = minE (D) [<=100]: <> time >= 100 under safe
```



Timed Games

TIGA



- Strategy:

$$\sigma : Exec^{f}_{\mathcal{G}} \rightharpoonup \mathcal{P}\left(\Sigma_c \cup \{\lambda\}\right) \setminus \{\emptyset\}$$

- Memoryless, deterministic, most permissive.

Run

$$\pi = (\mathrm{INIT}, x = 0) \xrightarrow{50.1\ \mathrm{r}} (\mathrm{CHOICE}, x = 0) \xrightarrow{2.4\ \mathrm{b}} (\mathrm{B}, x = 0) \xrightarrow{20.3\ \mathrm{d}} (\mathrm{END}, x = 20.3)$$

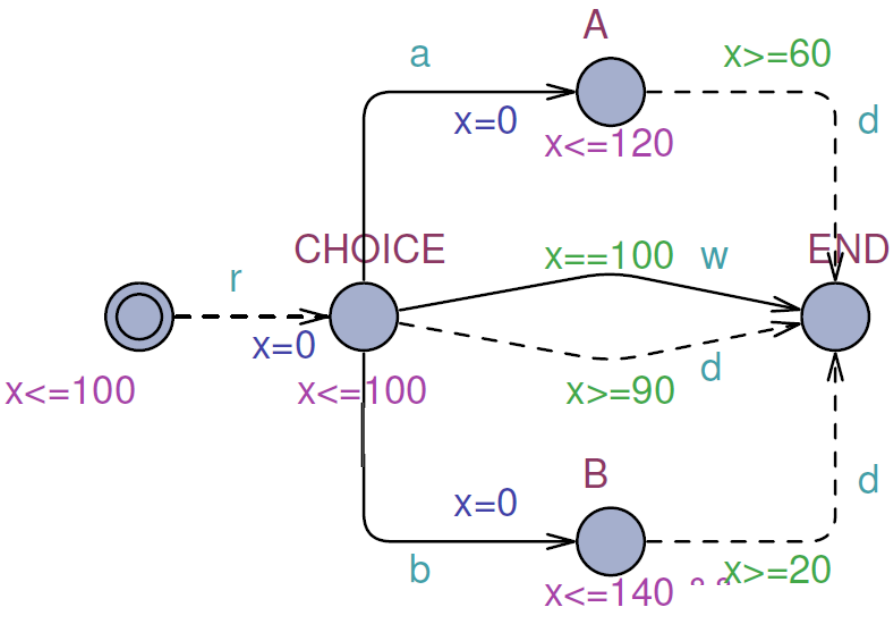
Run

$$\text{Total time} = 50.1 + 2.4 + 20.3 = 72.8$$

Asarin, Maler, Pnueli: Symbolic Controller Synthesis for Discrete and Timed Systems, HSCC94
Cassez, David, Fleury, Larsen, Lime: Efficient On-the-Fly Algorithms for the Analysis of Timed Games, CONCUR05.
Behrmann, Cougnard, David, Fleury, Larsen, Lime: UPPAAL-Tiga: Time for Playing Games! CAV05.

Timed Games –

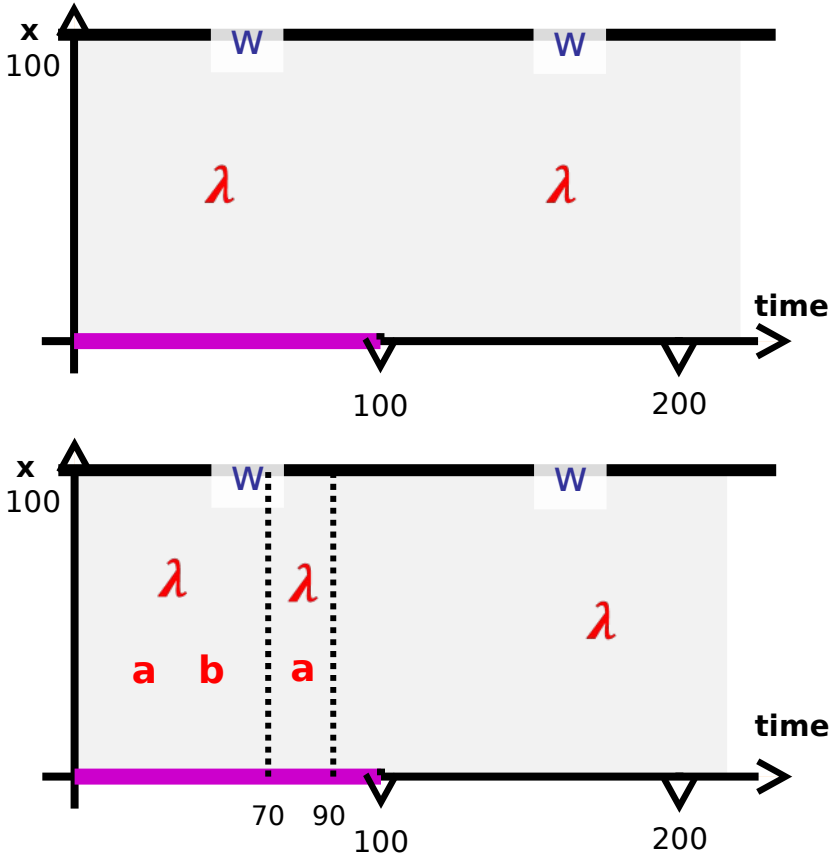
Time Bounded Reachability



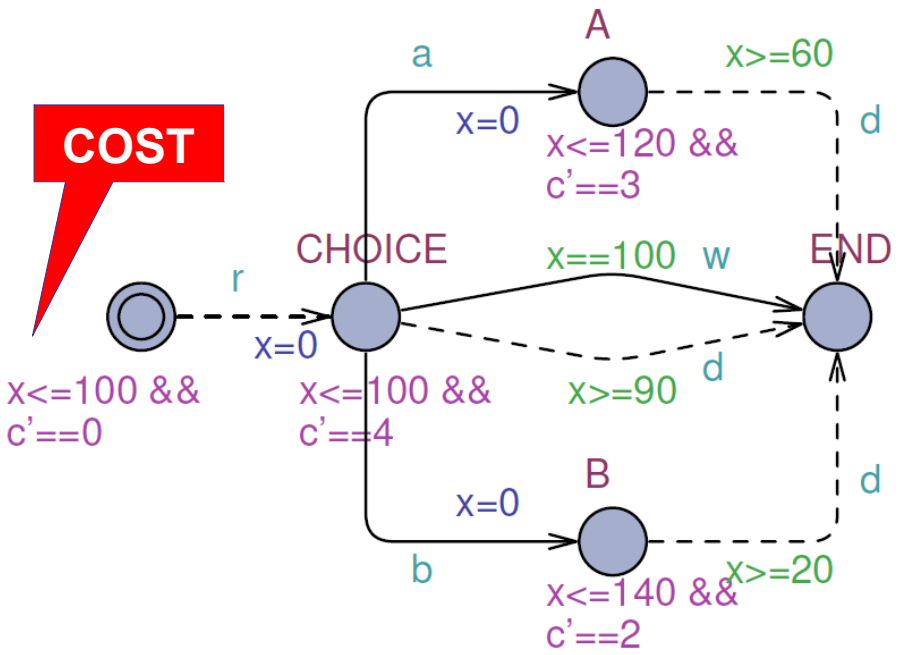
Most permissive, memoryless strategy

Objective: $A\langle\rangle(\text{END} \wedge \text{Time} \leq 210)$

Deterministic, memoryless strategy:



Priced Timed Games "CORA"



- Cost optimal strategy
  - take b immediately
    WC= 280

Priced

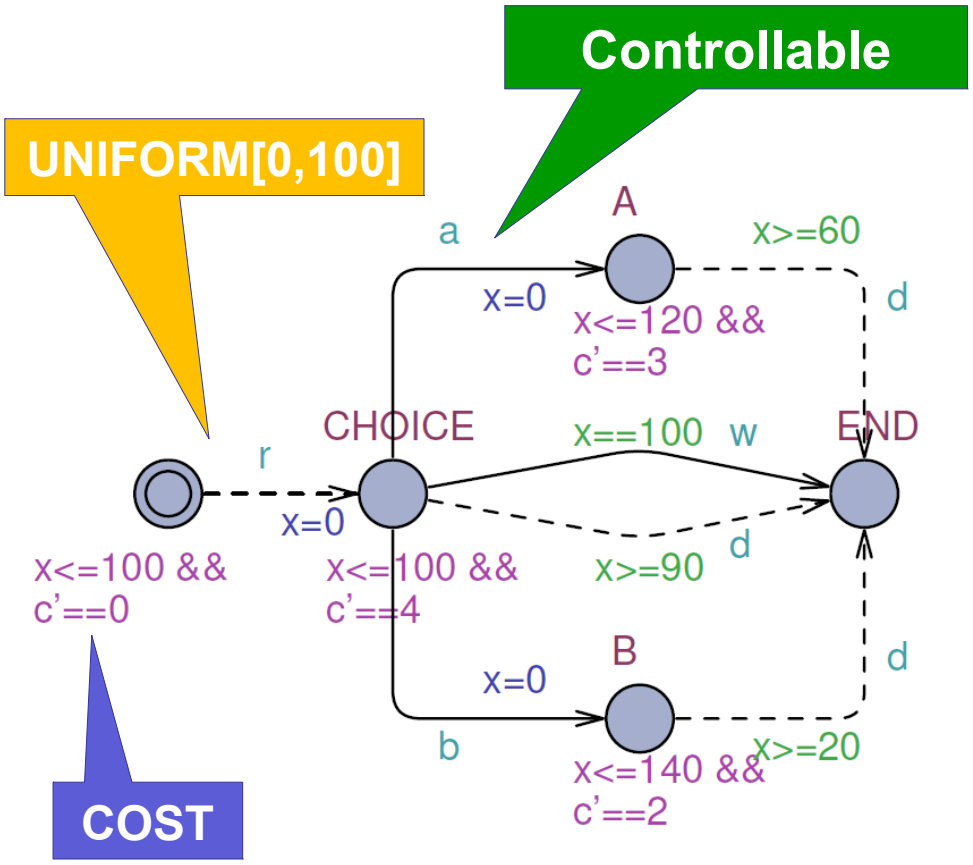
$$\pi = (\text{Init}, x = 0) \xrightarrow[\mathbf{0}]{50.1\ \ r} \rightarrow (\text{CHOICE}, x = 0) \xrightarrow[\mathbf{9.6}]{2.4\ \ b} \rightarrow (\text{B}, x = 0) \xrightarrow[\mathbf{40.6}]{20.3\ \ d} \rightarrow (\text{END}, x = 20.3)$$

Priced

$$\text{Total } cost = \mathbf{0} + \mathbf{9.6} + \mathbf{40.6} = \mathbf{50.5}$$

Bouyer, Cassez, Fleury, Kim Larsen:Optimal Strategies in Priced Timed Game Automata, FSTTCS04
Alur, Bernadsky, Madhusudan:Optimal Reachability for Weighted Timed Games. ICALP04
Bouyer, Brihaye, Markey: Improved undecidability results on weighted timed automata, IPL06
Bouyer, Larsen,, Rasmussen: Almost Optimal Strategies in One Clock Priced Timed Games. FSTTCS06
Hansen, Ibsen-Jensen, Miltersen: A Faster Algorithm for Solving One-Clock Priced Timed Games, CONCUR13

Stochastic Priced Timed Games

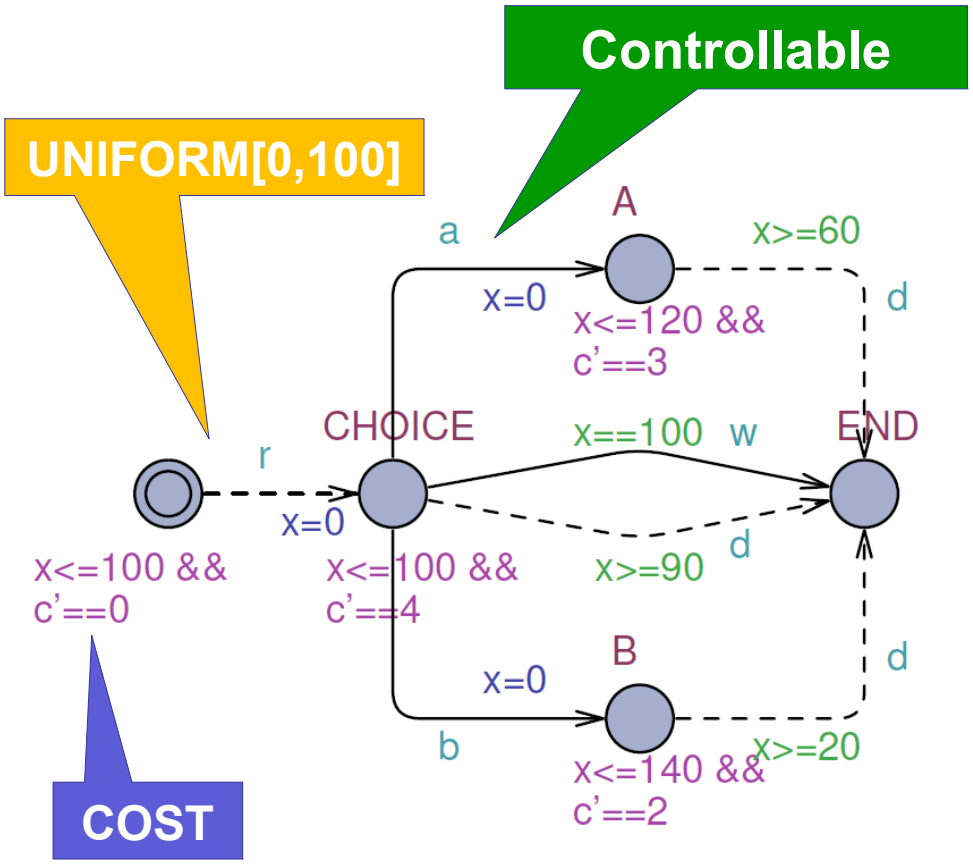


- Cost optimal strategy
  - take b immediately
    WC= 280

- Stochastic PTG
- Optimal **Expected** Cost str
  - take b immediately
    expectation = 160

Baier, Bertrand, Bouyer, Brihaye, Größer: Probabilistic and Topological Semantics for Timed Automata. FSTTCS07
David, Larsen, Legay, Mikucionis, Wang :Time for Statistical Model Checking of Real-Time Systems. CAV11
David, Jensen, Larsen, Legay, Lime, Sørensen, Taankvist: On Time with Minimal Expected Cost! ATVA14

Stochastic Priced Timed Games

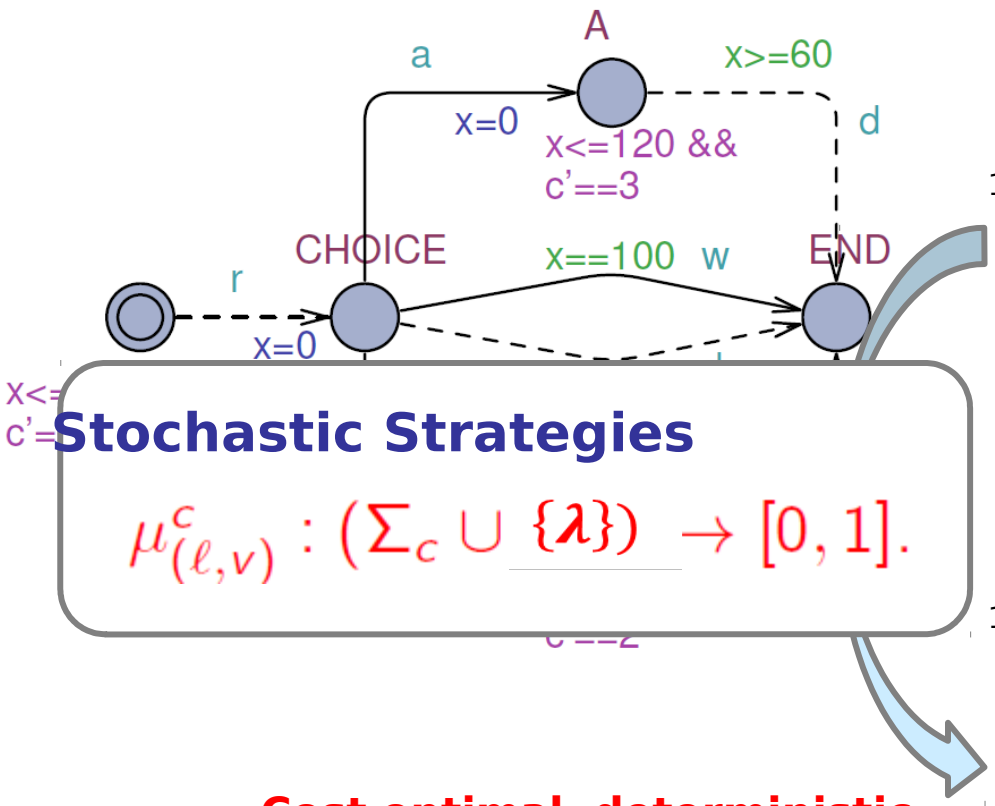


- Cost optimal strategy
  - take b immediately
    overall = 280
- Stochastic PTG
- Optimal **Expected** cost str
  - take b immediately
    expectation = 160
- Optimal Expected Cost while guaranteeing END is reached within time 210:

  Strat.: t>90 ▭ (100,w)
  t>70 ▭ (0,a)
  t<70 ▭ (0,b)
  = **204**

Baier, Bertrand, Bouyer, Brihaye, Größer: Probabilistic and Topological Semantics for Timed Automata. FSTTCS07
David, Larsen, Legay, Mikucionis, Wang :Time for Statistical Model Checking of Real-Time Systems. CAV11
David, Jensen, Larsen, Legay, Lime, Sørensen, Taankvist: On Time with Minimal Expected Cost! ATVA14.

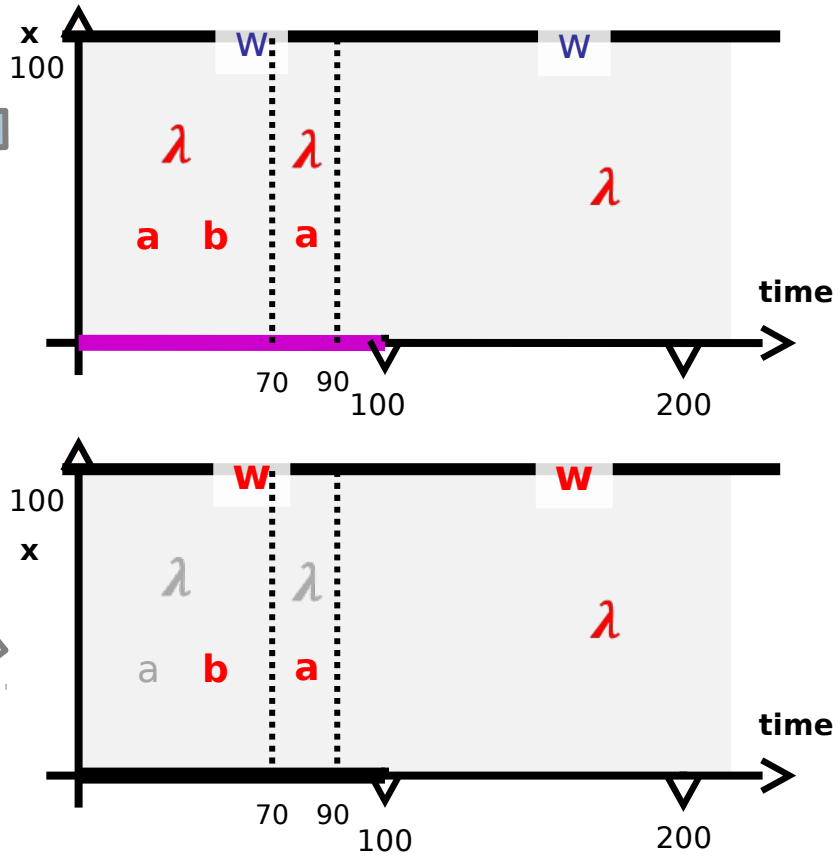
Stochastic Strategies for Learning!



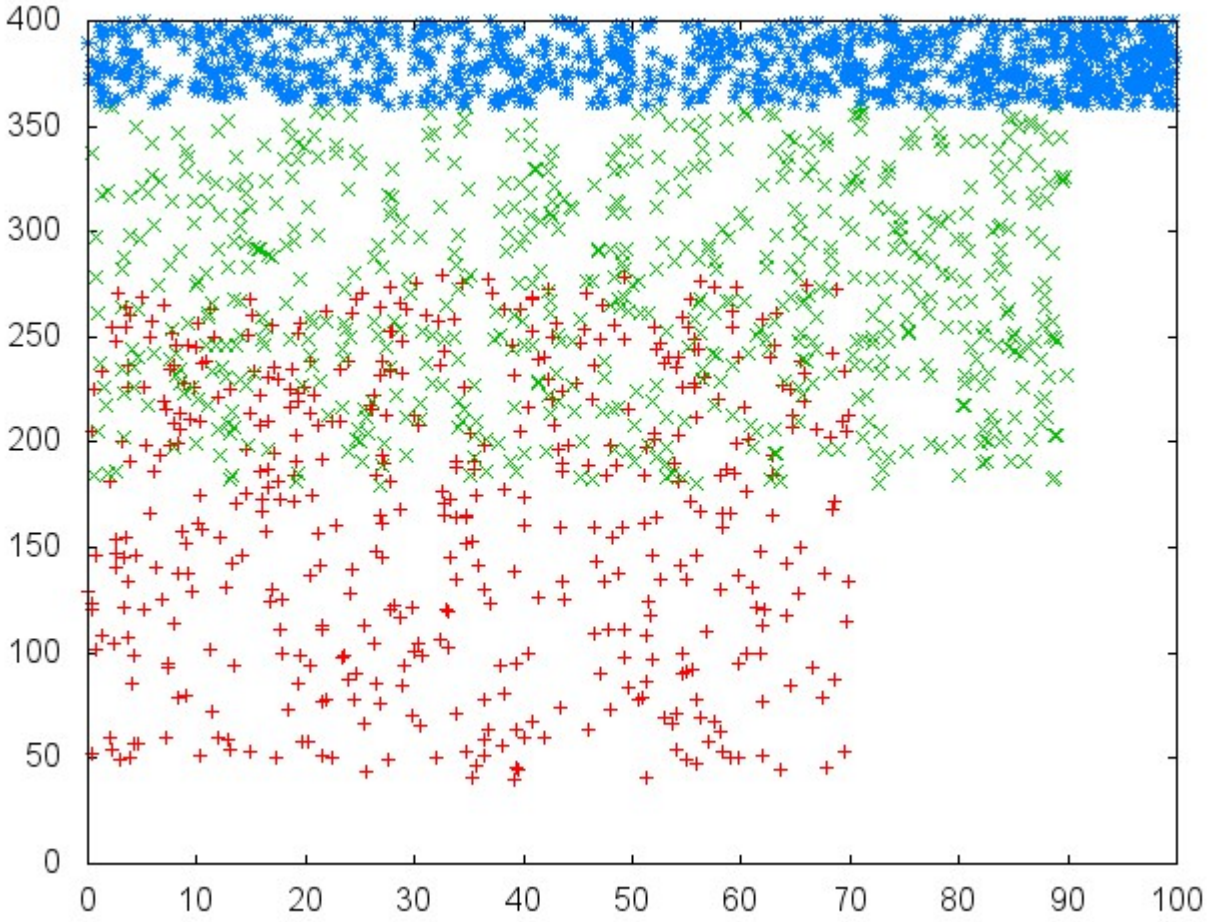
Cost optimal deterministic sub-strategy !

Objective: $A\langle\rangle(\text{ENDTime} \leq 210)$

Most permissive, memoryless strategy:



Learned Strategies Covariance Matrices

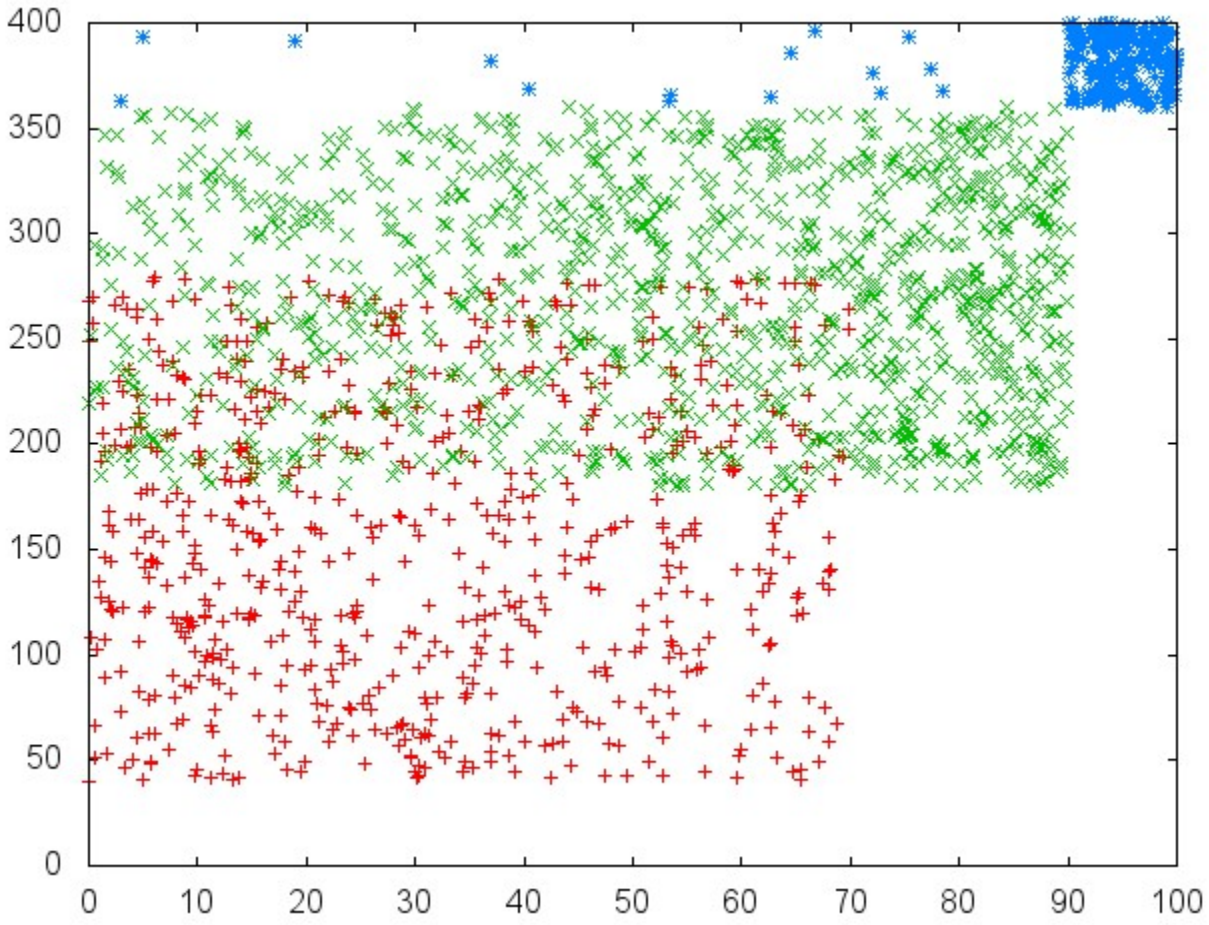


e. λ

a

b

Learned Strategies Covariance Matrices

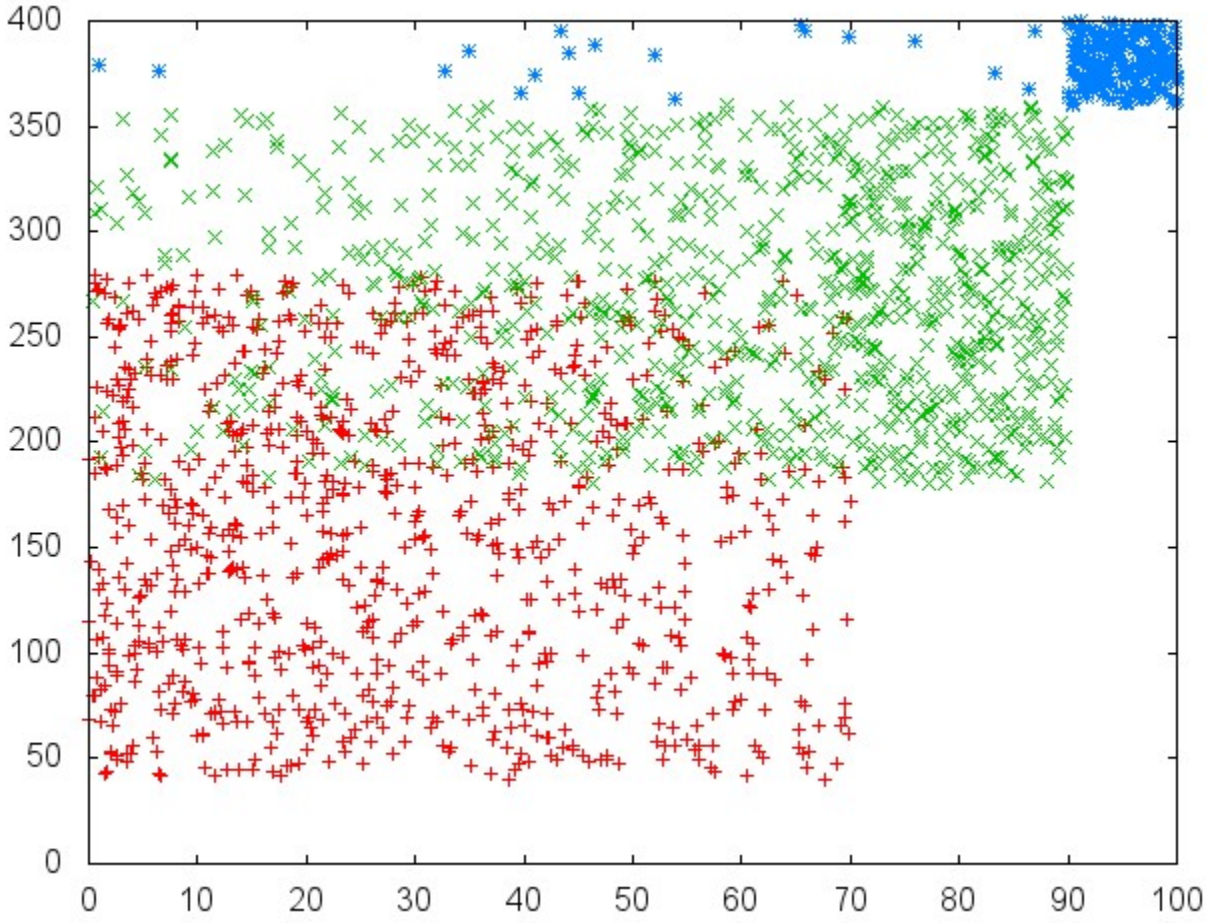


e. λ

a

b

Learned Strategies Covariance Matrices

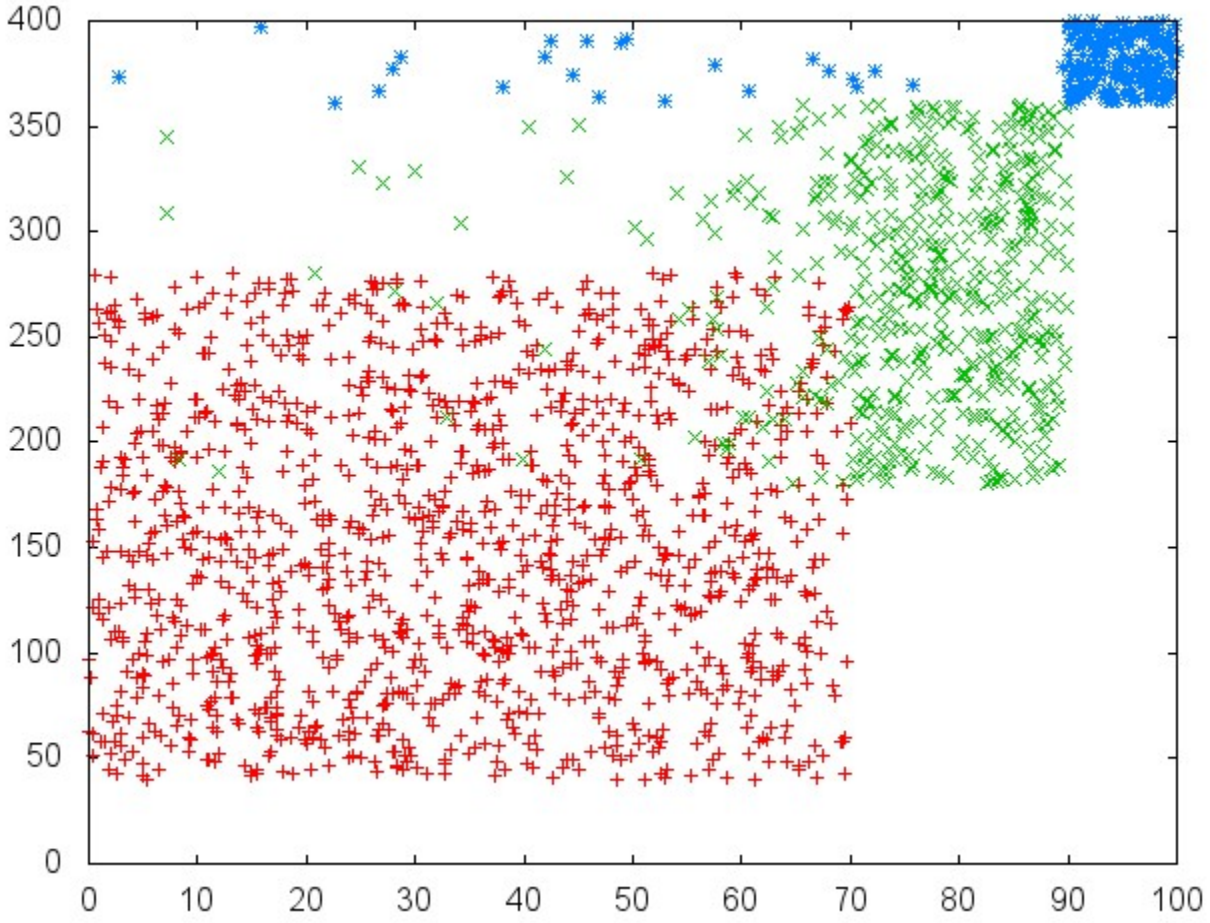


e. λ

a

b

Learned Strategies Covariance Matrices

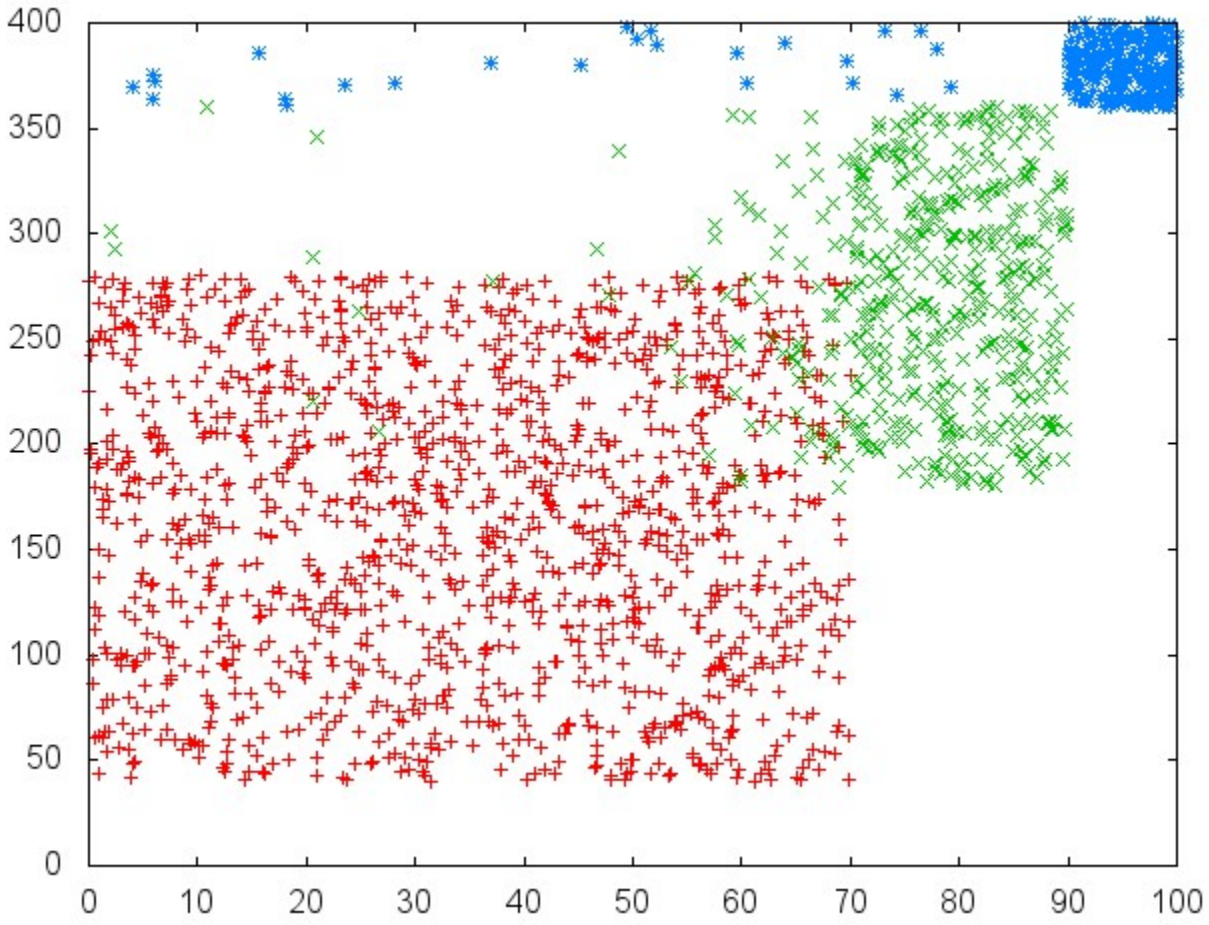


e. λ

a

b

Learned Strategies Covariance Matrices

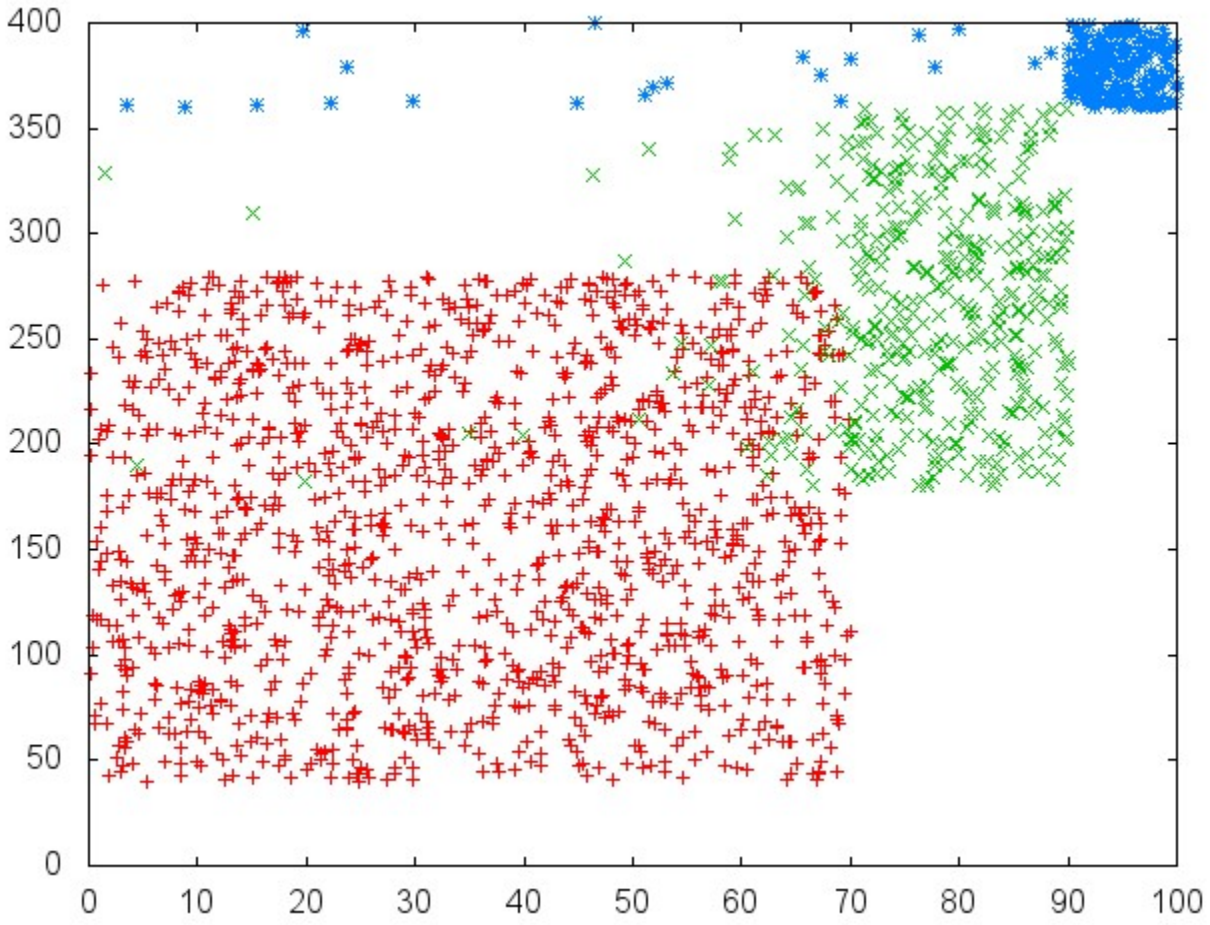


e. λ

a

b

Learned Strategies Covariance Matrices



e. λ

a

b

DEMO

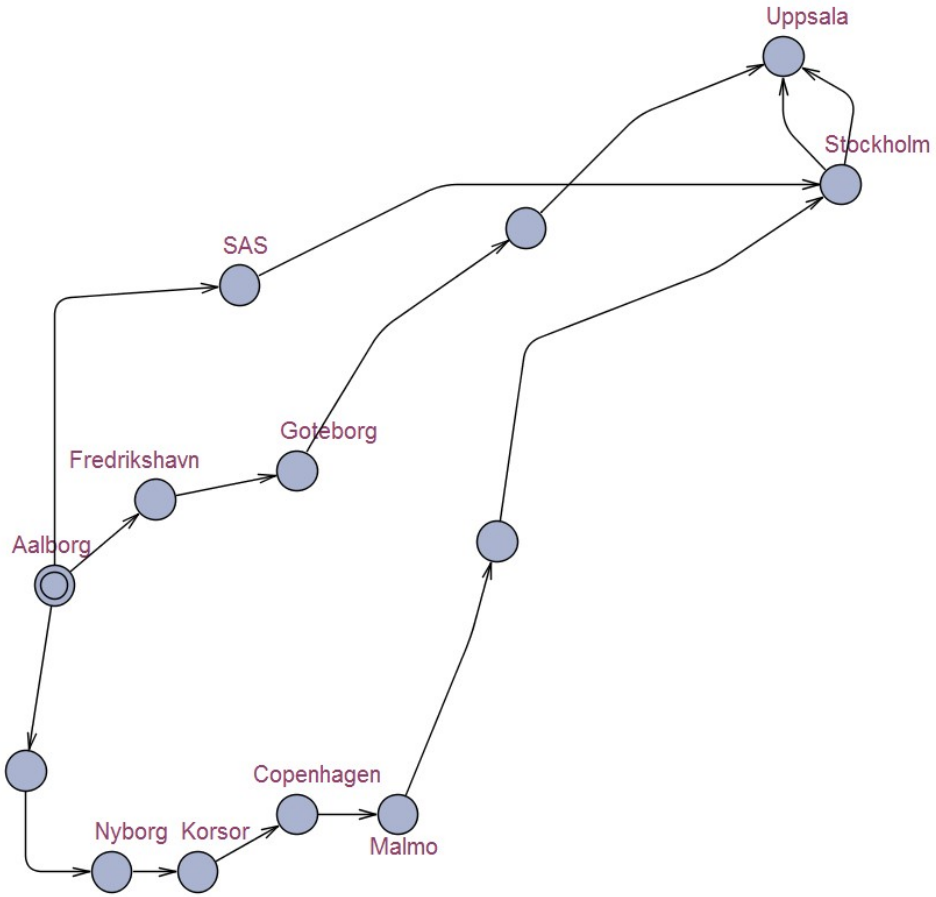
AALBORG UNIVERSITET


TEAM

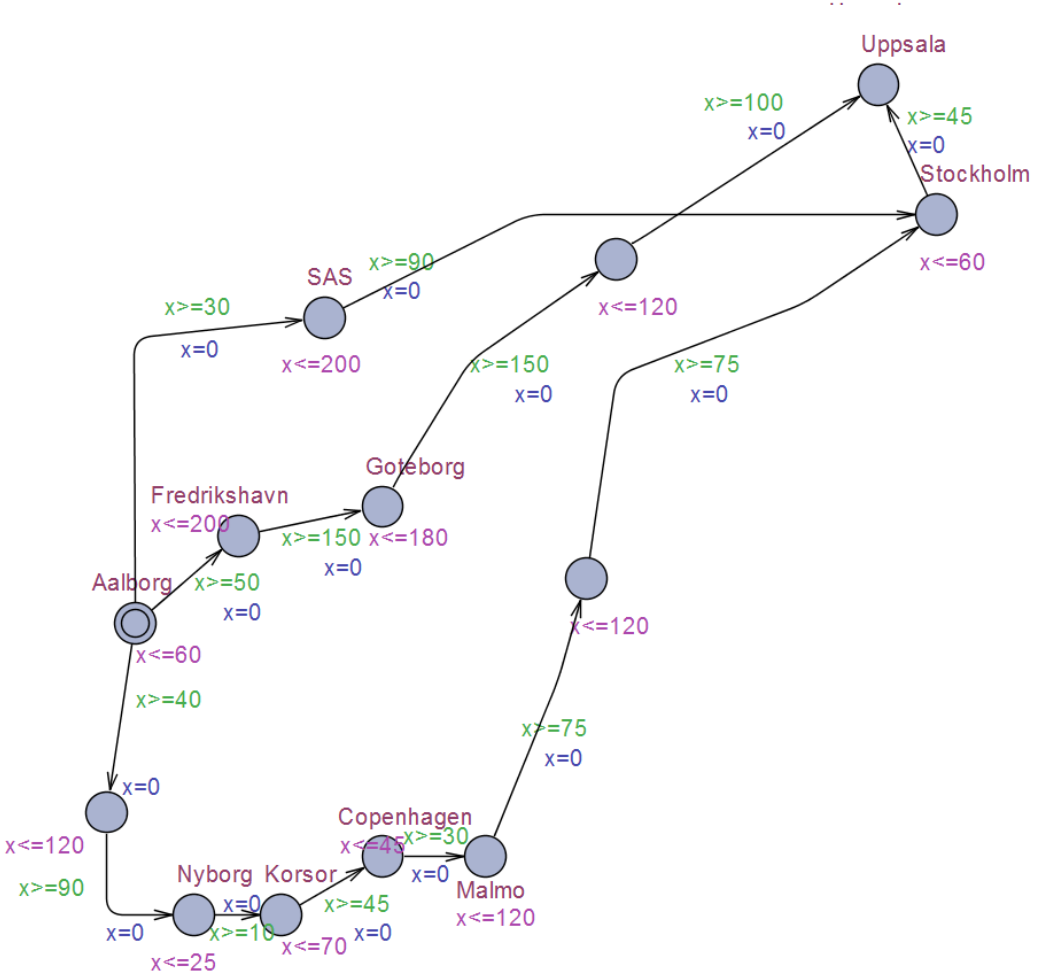

9 9 96

Planning the Travel

Planning the Travel



Planning the Travel



Before UPPAAL (<1995)

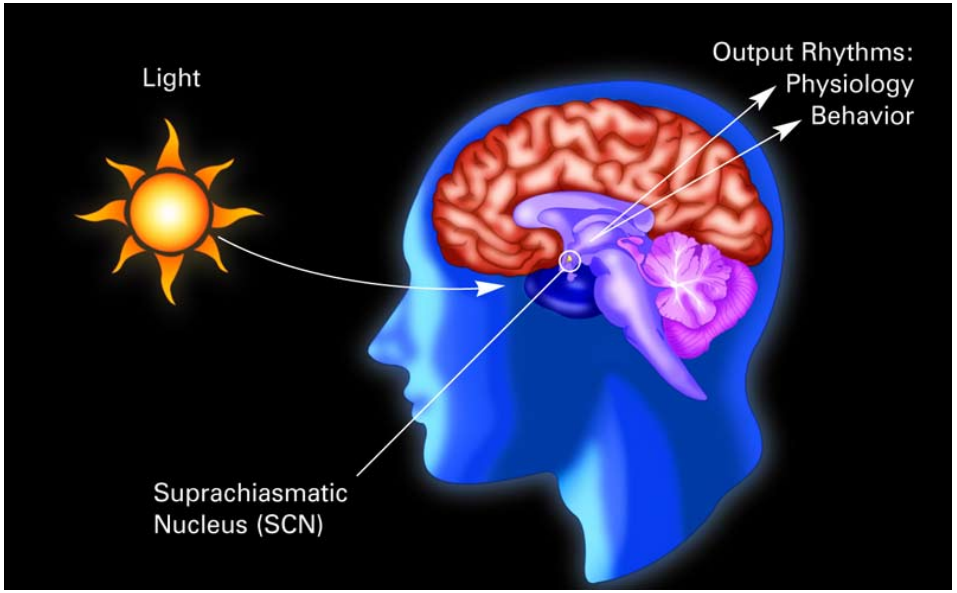
Genetic Oscillator

With
Alexandre David,, Axel Legay,
Marius Mikucionis,
Danny Bøgsted Poulsen, Sean Sedwards

AALBORG UNIVERSITET

Circadian Rhythm: circa-diem (~day)

- Genetically programmed chemical oscilation.
- Regulates sleep and activity cycles.
- Responds to light-dark periods
- Disruption causes sleeping disorders, jet-lag, depression, cancer.
- Understanding helps treating diabetes.



http://www.livescience.com/13123-circadian-rhythms-obesity-diabetes-nih.html

Continuous Time Markov Chain

simply count molecules

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

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

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

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

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

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

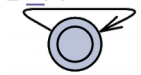
$$\mathrm{D}'_R \xrightarrow{\alpha'_R} \mathrm{M}_R + \mathrm{D}'_R \qquad \mathrm{M}_A \xrightarrow{\delta_{M_A}} \emptyset$$

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

A>0 && DA>0 **Require**: at least one of A and one of DA

A−−, DA−−, D_A++ **Consume**: one of A and one of DA

Produce: one of D_A

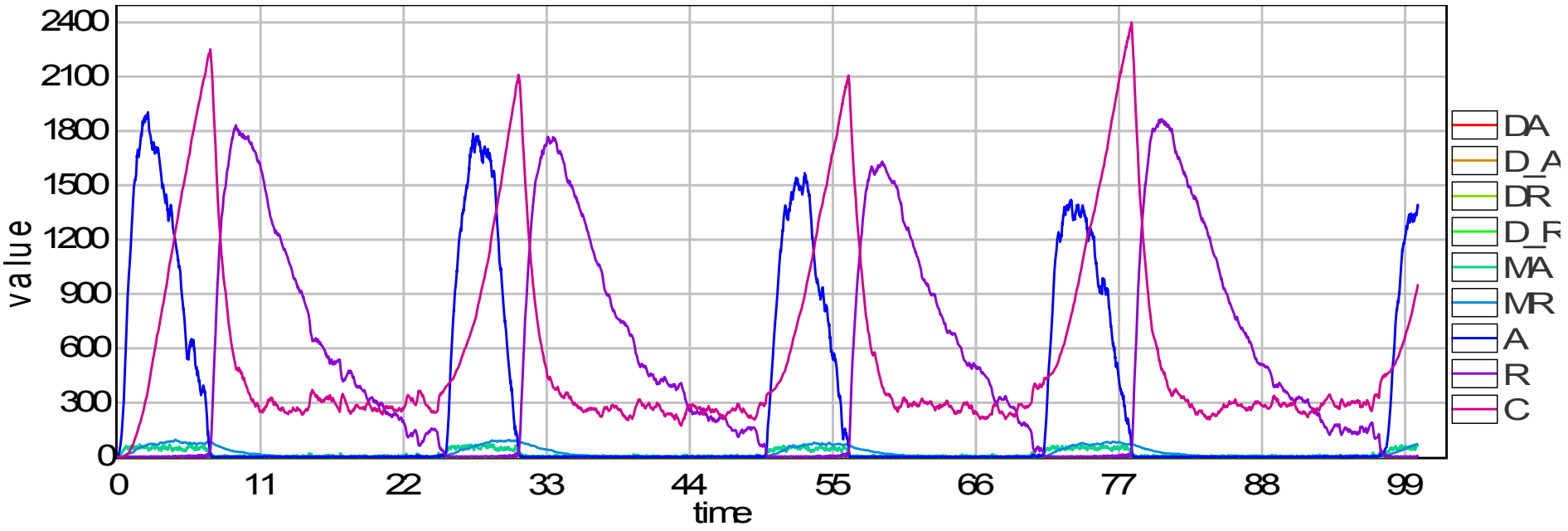


A*DA*gammaA **Rate**: proportional to gammaA and amounts of A and DA.

[Vilar, Kueh, Barkai, Leibler: Mechanisms of noise-resistance in genetic oscillators, PNAS, 2002]

UPPAAL SMC: quantities over time

```
simulate 1 [<=100]{ DA, D_A, DR, D_R, MA, MR, A, R, C}
```



Ordinary Differential Equations

$$\begin{array}{ll}
\mathrm{A} + \mathrm{D}_A \xrightarrow{\gamma_A} \mathrm{D}'_A & \mathrm{M}_A \xrightarrow{\beta_A} \mathrm{M}_A + \mathrm{A} \\
\mathrm{D}'_A \xrightarrow{\theta_A} \mathrm{D}_A + \mathrm{A} & \mathrm{M}_R \xrightarrow{\beta_R} \mathrm{M}_R + \mathrm{R} \\
\mathrm{A} + \mathrm{D}_R \xrightarrow{\alpha_R} \mathrm{D}'_R & \mathrm{A} + \mathrm{R} \xrightarrow{\gamma_C} \mathrm{C} \\
\mathrm{D}'_R \xrightarrow{\theta_R} \mathrm{D}_R + \mathrm{A} & \mathrm{C} \xrightarrow{\delta_A} \mathrm{R} \\
\mathrm{D}'_A \xrightarrow{\alpha'_A} \mathrm{M}_A + \mathrm{D}'_A & \mathrm{A} \xrightarrow{\delta_A} \emptyset \\
\mathrm{D}_A \xrightarrow{\alpha_A} \mathrm{M}_A + \mathrm{D}_A & \mathrm{R} \xrightarrow{\delta_R} \emptyset \\
\mathrm{D}'_R \xrightarrow{\alpha'_R} \mathrm{M}_R + \mathrm{D}'_R & \mathrm{M}_A \xrightarrow{\delta_{M_A}} \emptyset \\
\mathrm{D}_R \xrightarrow{\alpha_R} \mathrm{M}_R + \mathrm{D}_R & \mathrm{M}_R \xrightarrow{\delta_{M_R}} \emptyset
\end{array}$$

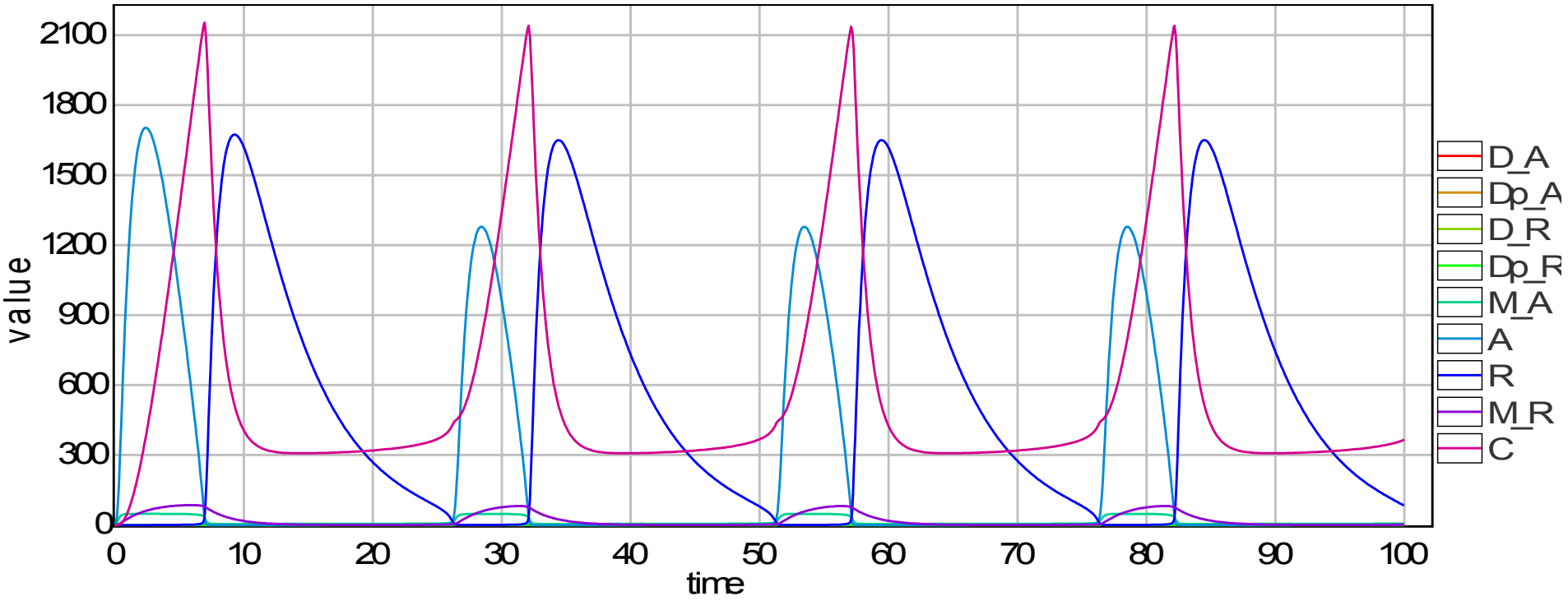
$$dD_A/dt = \theta_A D'_A - \gamma_A D_A A$$

```
D_A' = theta_A*Dp_A-gamma_A*D_A*A &&
D_R' = theta_R*Dp_R-gamma_R*D_R*A &&
Dp_A' = gamma_A*D_A*A-theta_A*Dp_A &&
Dp_R' = gamma_R*D_R*A-theta_R*Dp_R &&
M_A' = alphap_A*Dp_A+alpha_A*D_A-delta_MA*M_A &&
A' = beta_A*M_A+theta_A*D_A+theta_R*Dp_R
    -A*(gamma_A*D_A+gamma_R*D_R+gamma_C*R+delta_A) &&
M_R' = alphap_R*Dp_R+alpha_R*D_R-delta_MR*M_R &&
R' = beta_R*M_R-gamma_C*A*R+delta_A*C-delta_R*R &&
C' = gamma_C*A*R-delta_A*C
```

[Vilar, Kueh, Barkai, Leibler: Mechanisms of noise-resistance in genetic oscillators, PNAS, 2002]

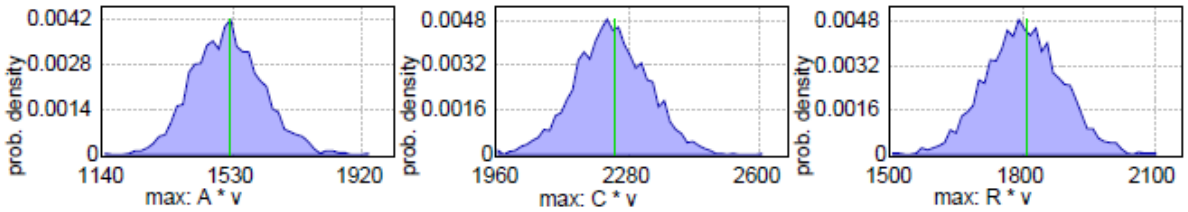
UPPAAL SMC: quantities over time

```
simulate 1 [<=100]{ DA, D_A, DR, D_R, MA, MR, A, R, C}
```



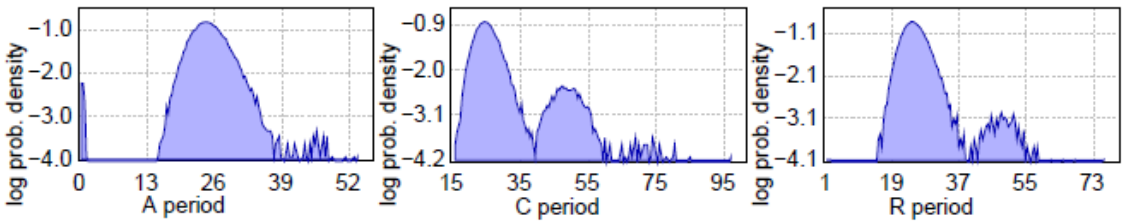
Amplitudes & Periods for CTMC

Amplitude `E[<=75; 2000](max: A ,C ,R )`



(a) A, mean 1522. (b) C, mean 2243. (c) R, mean 1807.

Periods: **PEAK** = `true U[<=1000] (A>1100 & true U[<=5] A<=1000)`
`Pr[x<=100](<> secondPeak.ACCEPT)`



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

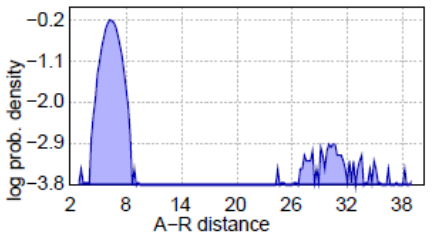


Fig. 10: Phase diff., peak at 6.21.