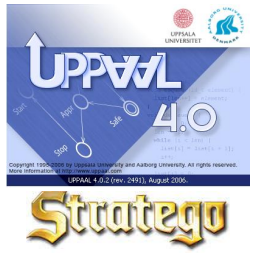
Synthesis of Safe, Small and Optimal Strategies for Cyber-Physical Systems

Kim G Larsen

Aalborg University, DENMARK



Collaborators

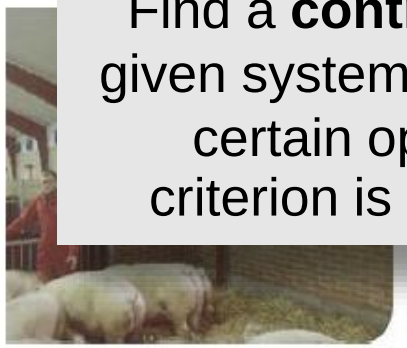
- Alexandre David
- Marius Mikucionis
- Peter Jensen
- Jakob Taankvist
- Manfred Jaeger
- Axel Legay
- Sean Sedwards
- Pranav Ashok
- Jan Kretinsky
- Adrien Le Coënt
- Maximilian Weininger

[ATVA19] Teaching Stratego to play Ball [QEST19] SOS: Synthesis for Hybrid MDP
[ISOLA20] Approximating Euclidain by Imprecise MDPs
[ADHS21] Learning Safe and Optimal Control m Strategies for Storm Water Detention Ponds

Cyber Physical Systems
Games Everywhere

SAFE:
Find **controller** that function (correct) under stated conditions for a specified period of time.

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



Discrete

Real Time

Resources

Stochasticity

Hybrid

COMPACT:
Find a representation of the **controller** that may fit a given embedded platform

Networked ES

Cloud Computing Big Data

Cyber-Physical Systems

Overview

- UPPAAL STRATEGO
  - Dagstuhl 19432
    Smart Farming Challenge

- Better Learning
- Approximation Methods
- Compact Strategies

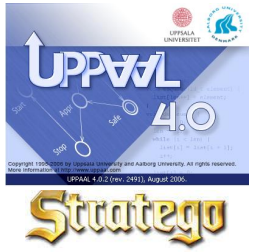
- Applications
- Future Challenges

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

SMART FARMING
&
UPPAAL STRATEGO

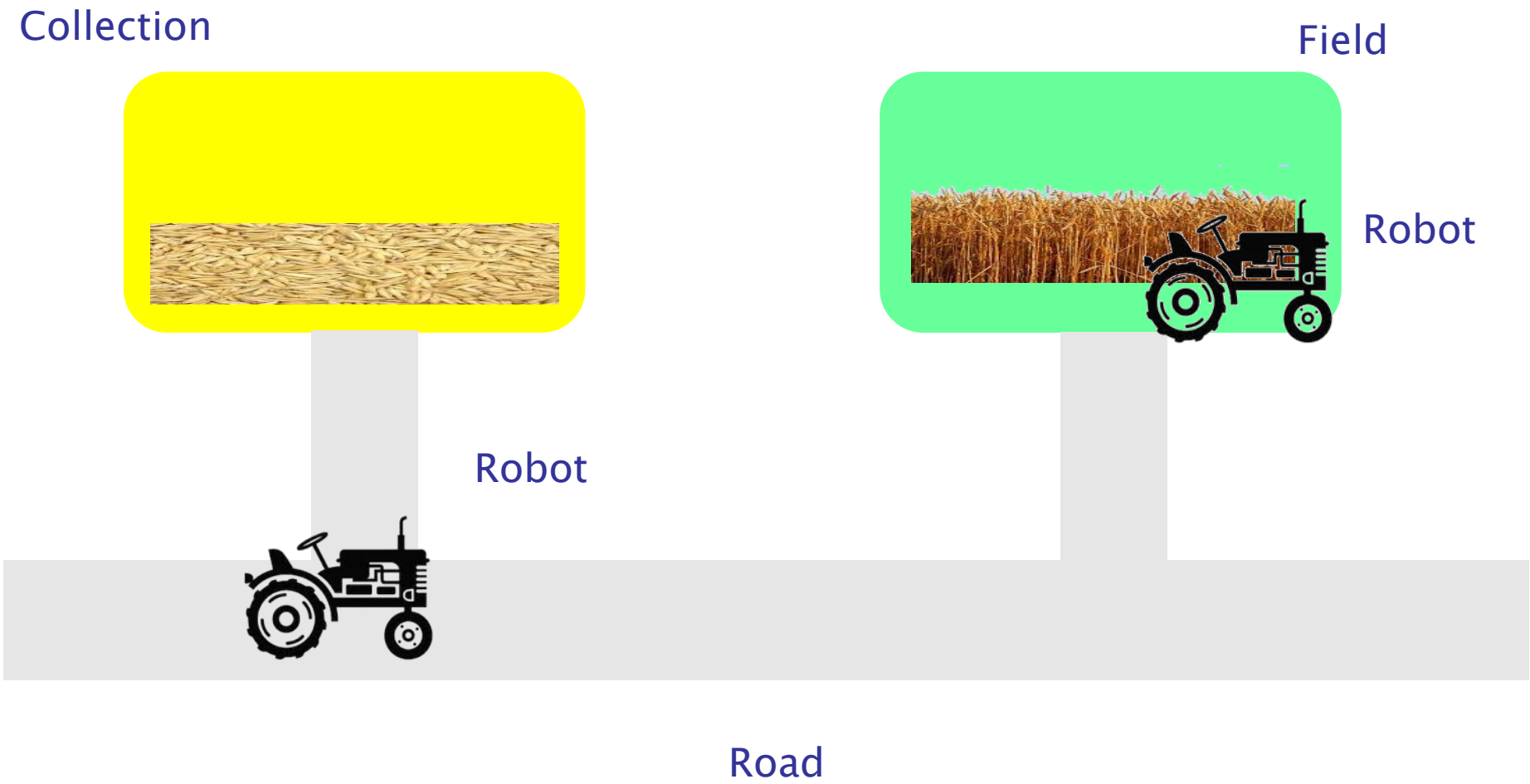
AALBORG UNIVERSITET

VILLUM FONDEN



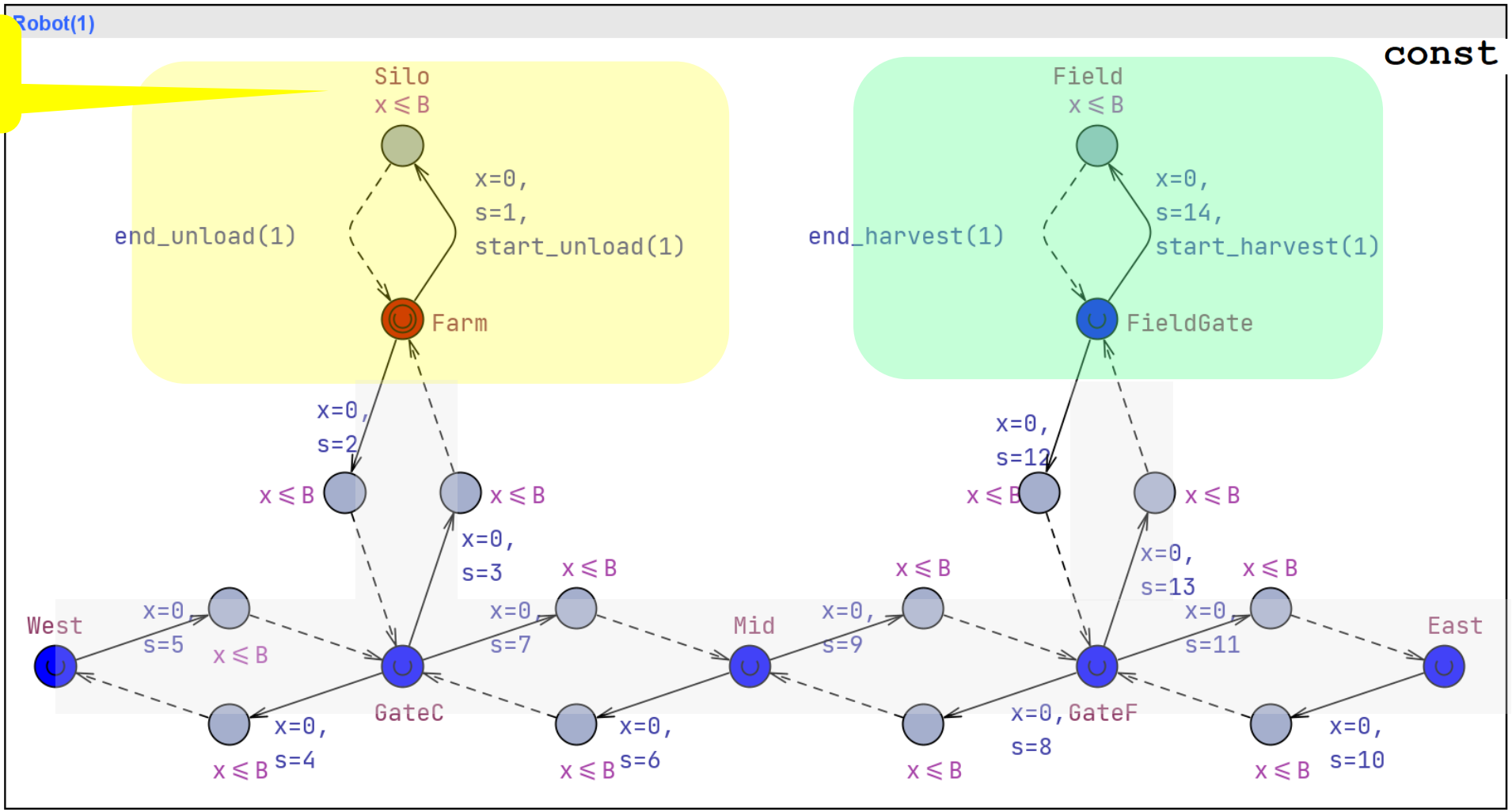
TACAS15: David, Jensen, L, Mikucionis, Taankvist: Uppaal Stratego.

Smart Farming / Dagstuhl 19432 Oct19



Smart Farming – Timed Automata

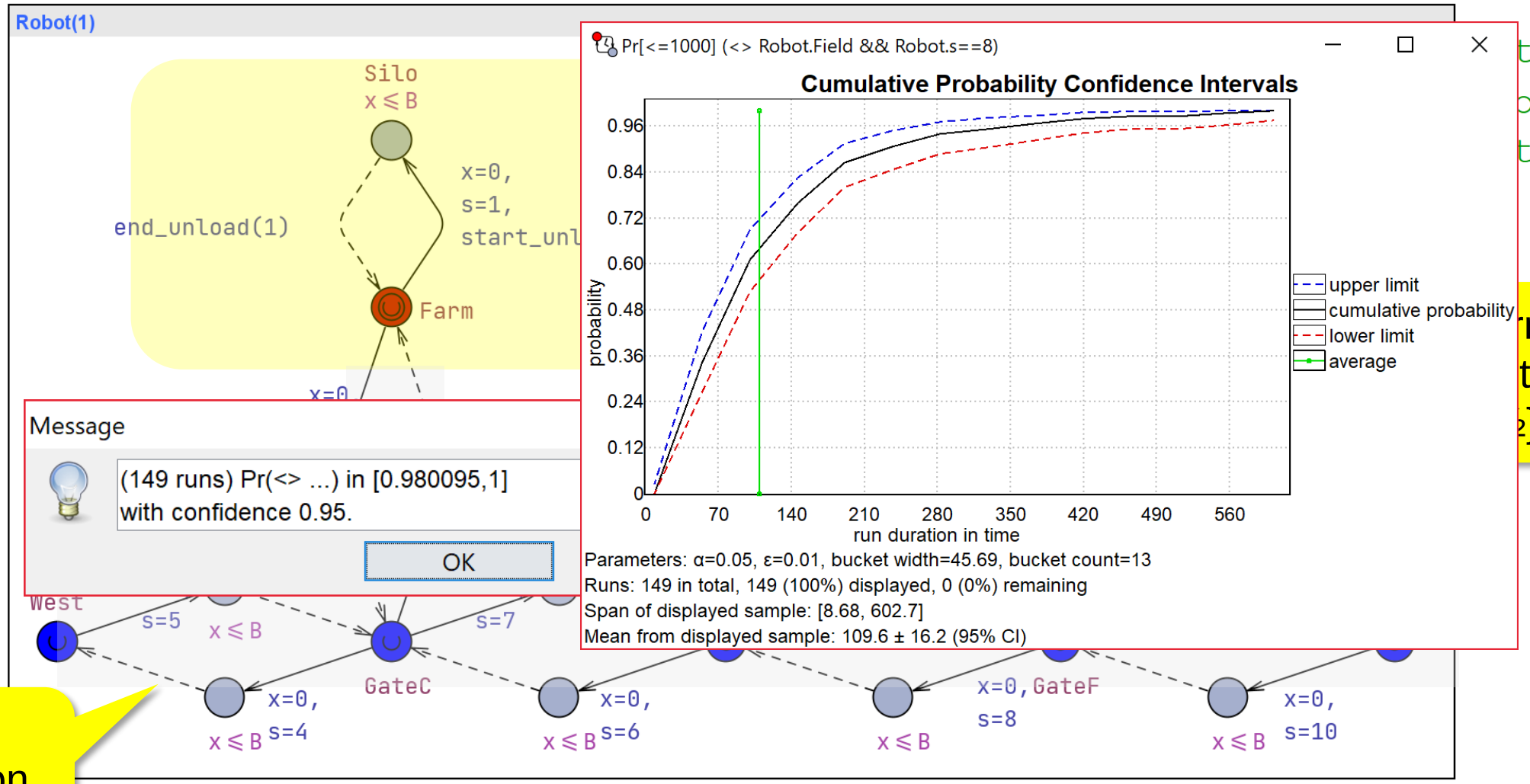
Invariant




```
const int B=5;
clock x;
int s=1;
```

E<> Robot(1).Field && Robot(1).s==14

Smart Farming – Stochastic Timed Automata



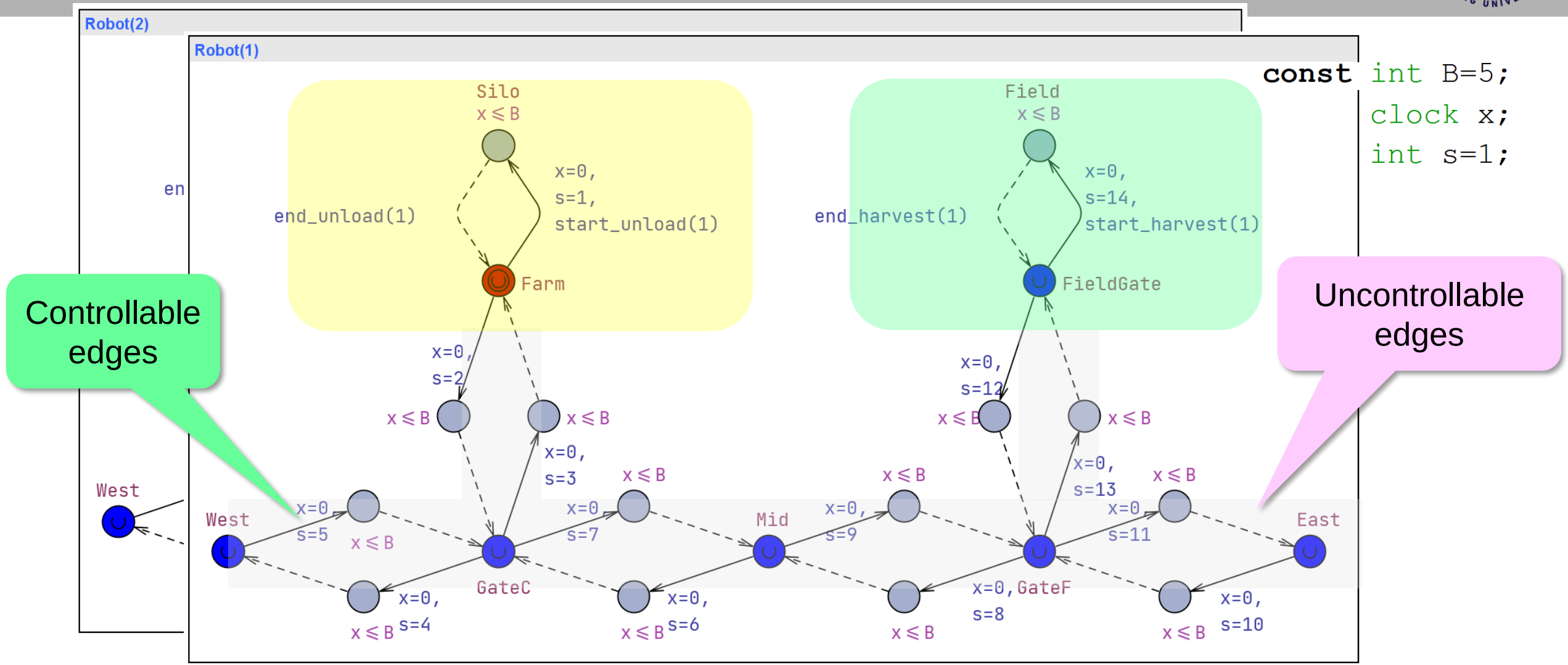

```
t B=5;
ock x;
t s=1;
```

rm
tion
2}

Uniform distribution [0,B]

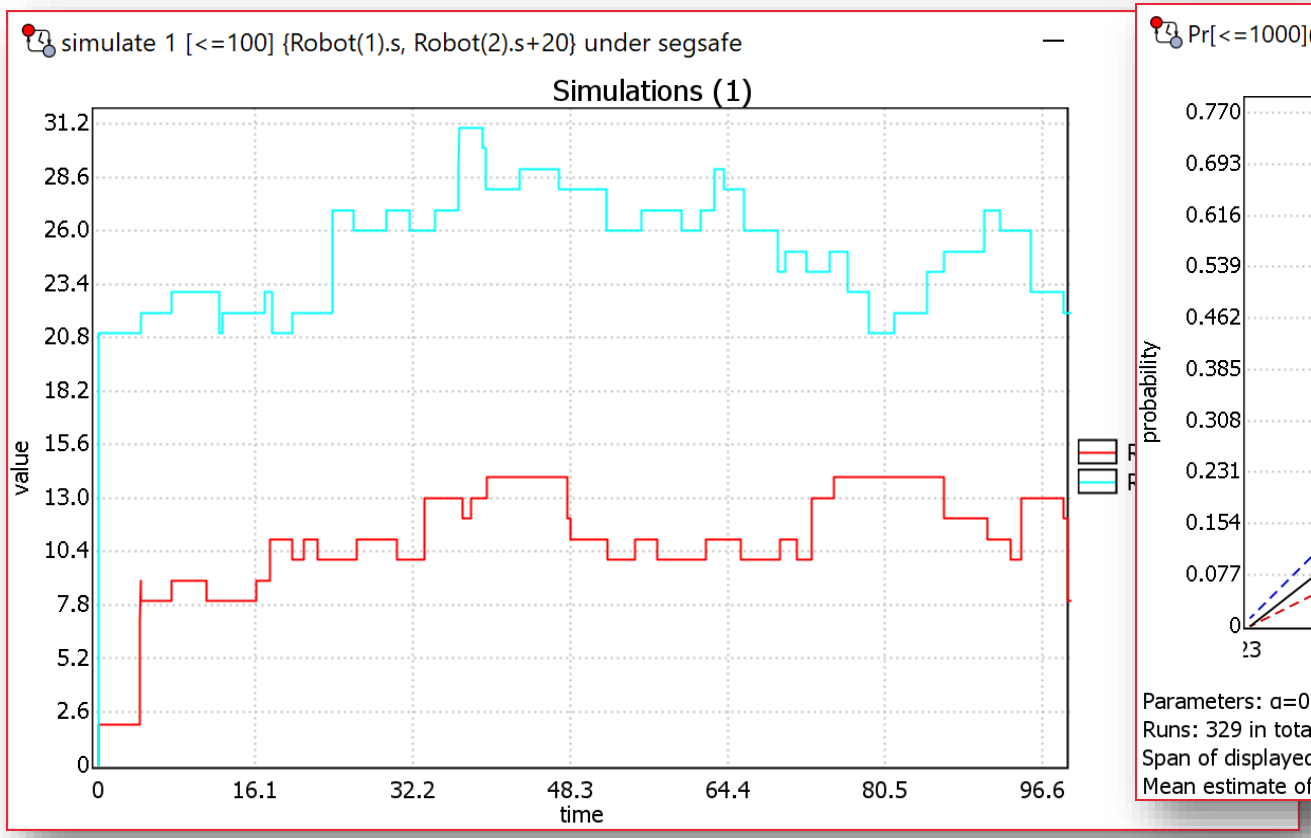
Pr[<=1000] (<> Robot(1).Field && Robot(1).s==14)

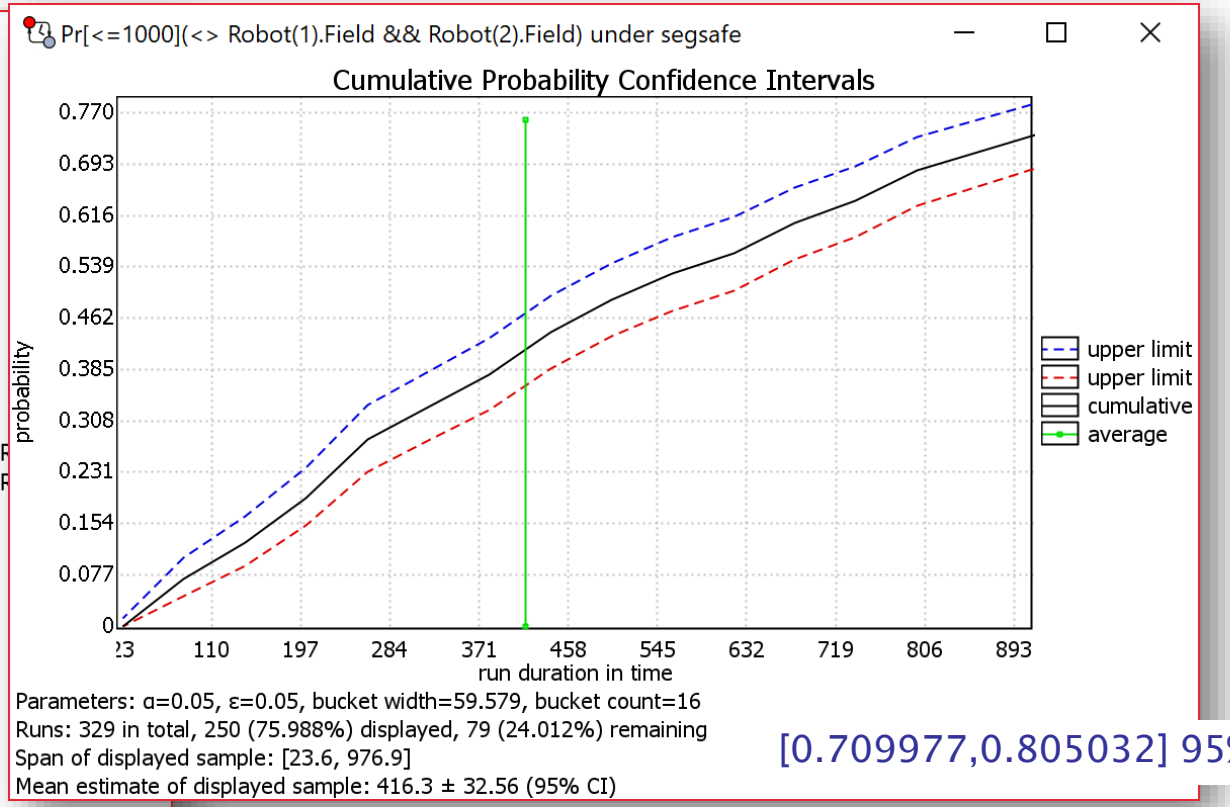
Smart Farming –Timed Game



strategy segsafe = control: A[] ! (Robot(1).s>1 && Robot(1).s<14 && Robot(1).s==Robot(2).s)

Smart Farming – Timed Games





[0.709977,0.805032] 95%

simulate 1 [<=100] {Robot(1).s, Robot(2).s+20} under segsafe

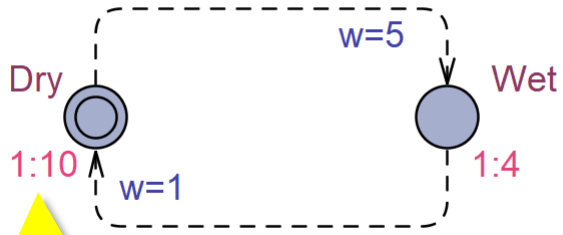
E<> Robot(1).Field && Robot(2).Field under segsafe

E<> Robot(1).s>1 && Robot(1).s<14 && Robot(1).s==Robot(2).s under segsafe

Pr[<=1000] (<> Robot(1).Field && Robot(2).Field==14) under segsafe

Smart Farming – Stochastic & Hybrid Stuff

Rain



Rates of exponential distributions

Field

```
f'==(((F-f)/(0.5*F))*w)*scale-
(harvesting[1]*f*(
 harvesting[2]*f*(
```

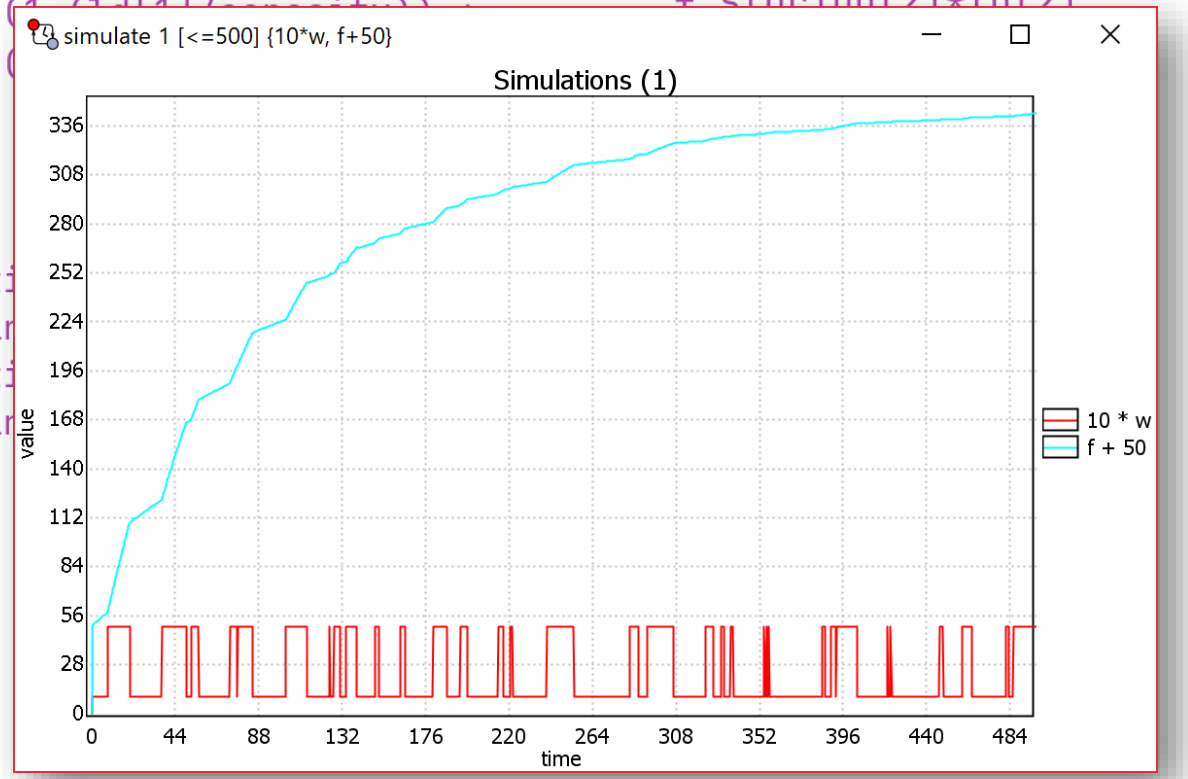
Load

```
ld[1]'==harvesti
        - storin
ld[2]'==harvesti
        - storin
```

Store

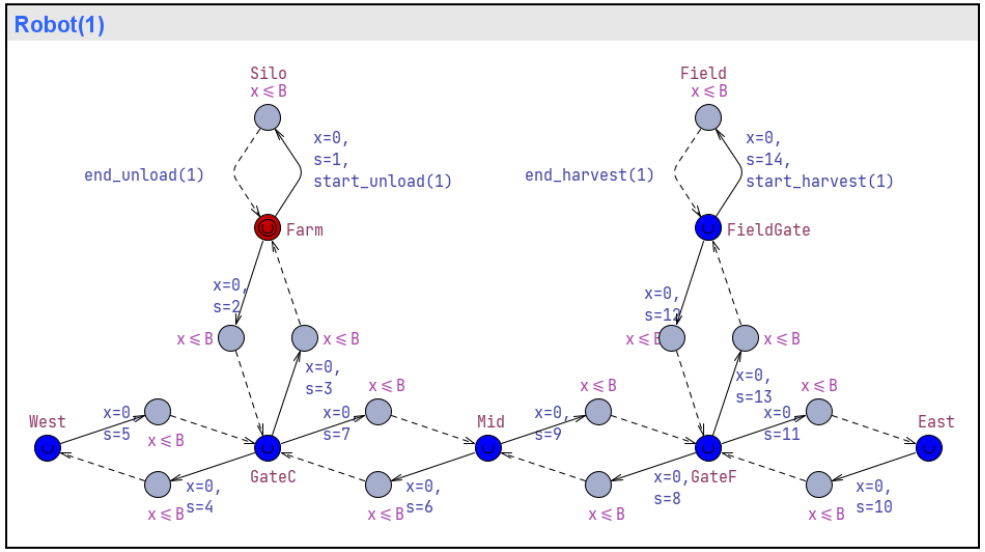
ODEs

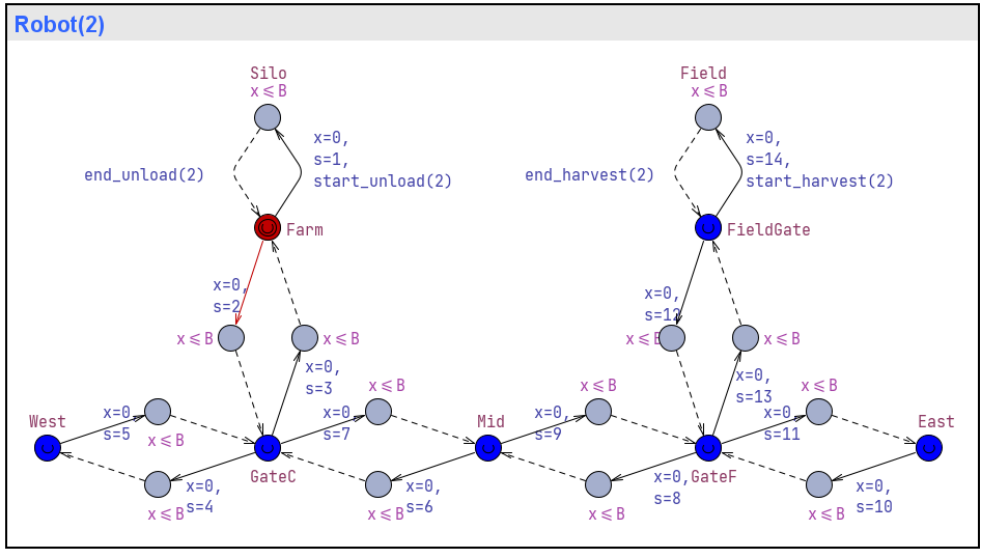
```
c'==storing[1]*ld[1]
```

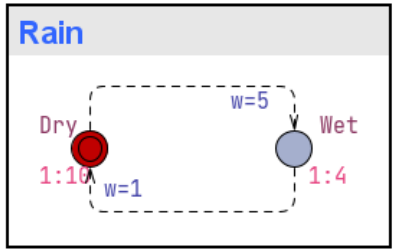


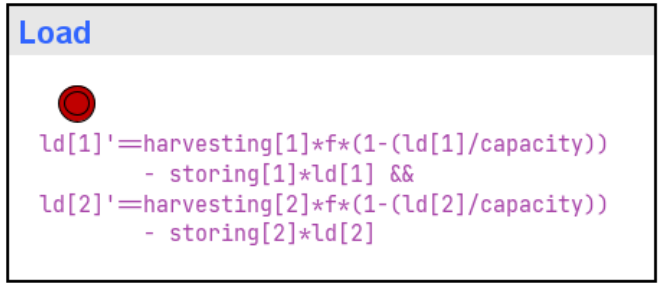

```
const double capacity = 40.0;
hybrid clock ld[id_t];
hybrid clock f, c;
```

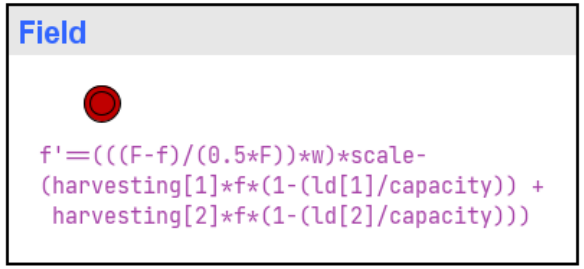
Smart Farming – Complete Model

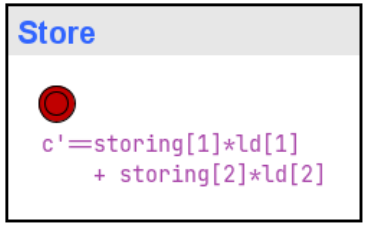








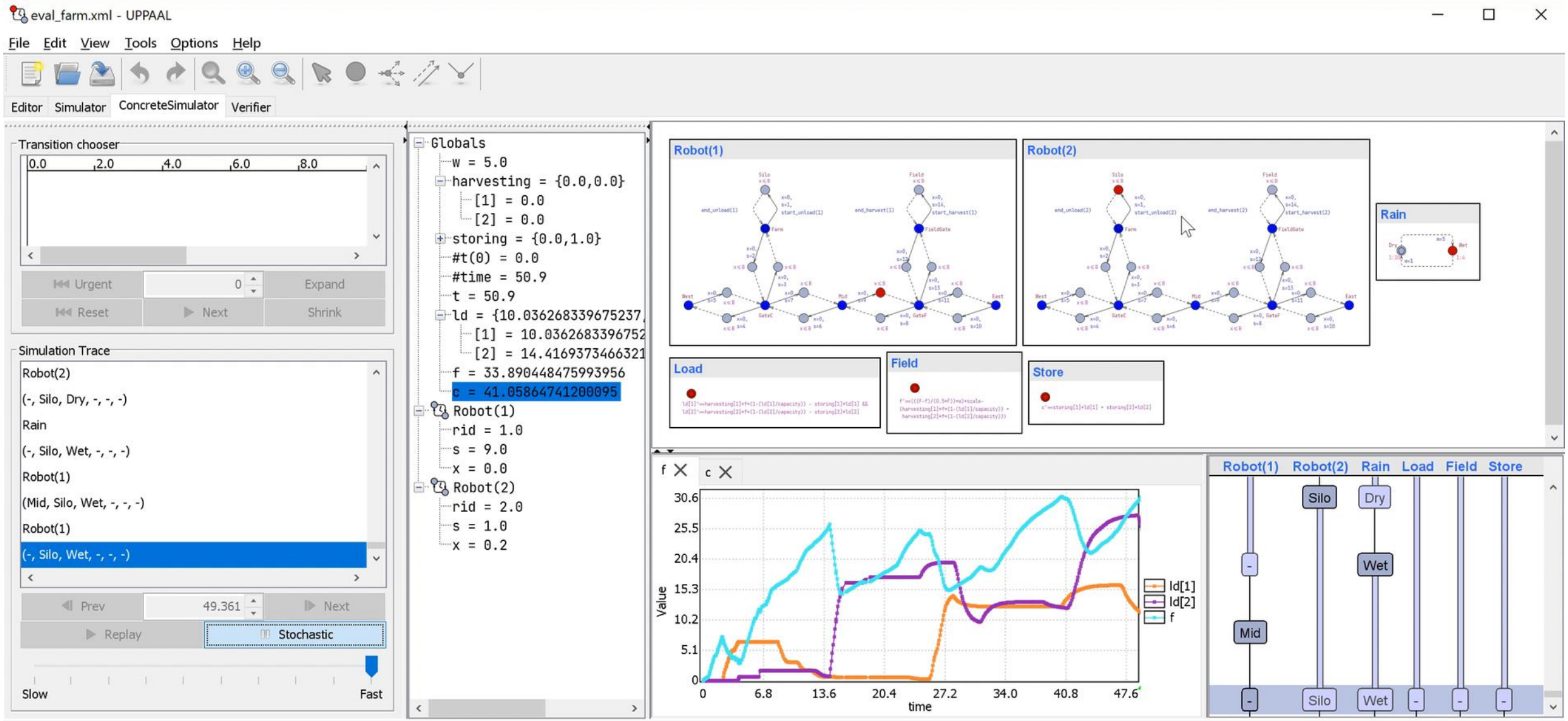




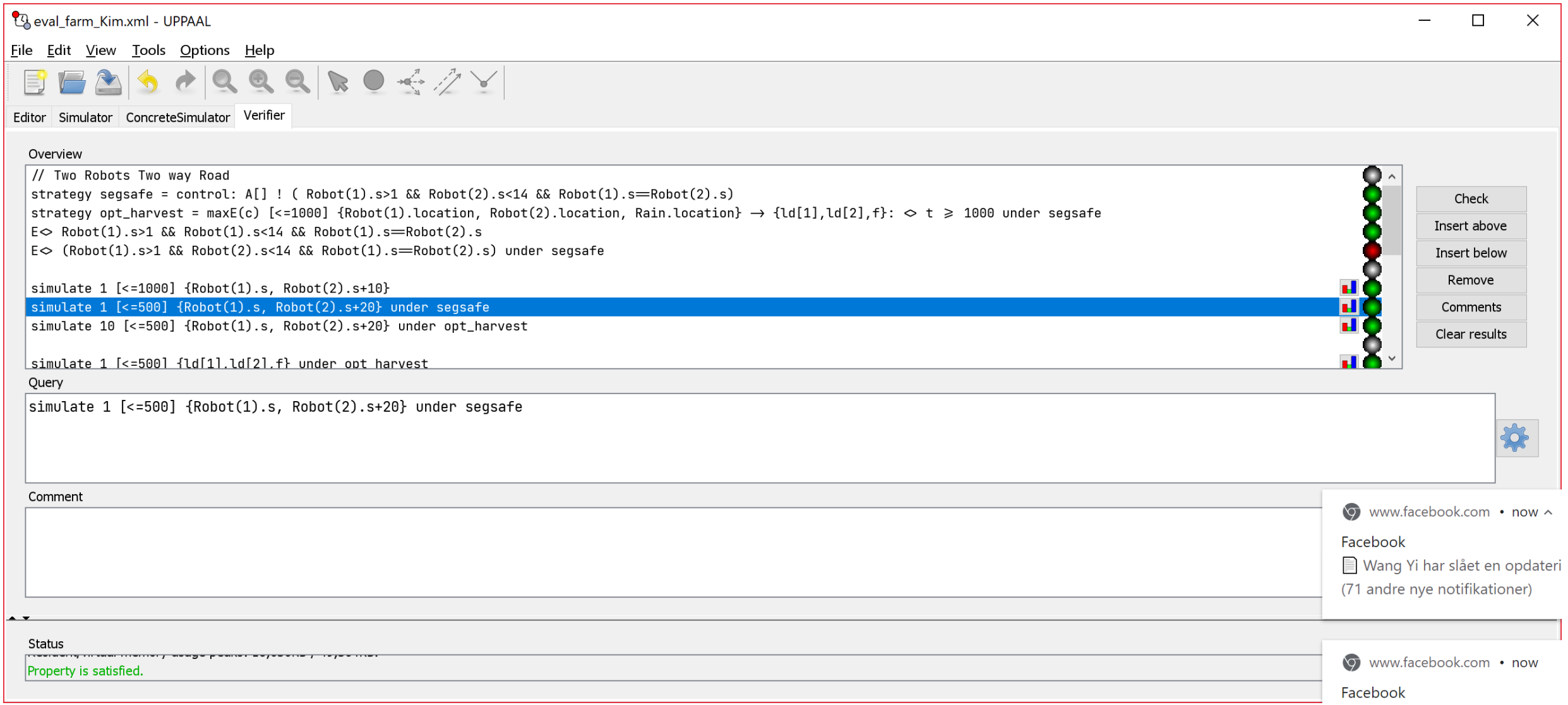

```
typedef int[1,2] id_t;
const int B=5;
int w=1;
clock t;


const double capacity = 40.0;
hybrid clock ld[id_t];
hybrid clock f, c;
```

```
bool harvesting[id_t] = {false, false};
bool storing[id_t] = {false,false};

bool capacity_check(id_t rid) {
    if(ld[rid] ⩾ capacity)
        return 0;
    else
        return 1;
}

void start_unload(id_t rid) {
    storing[rid] = true;
}
```

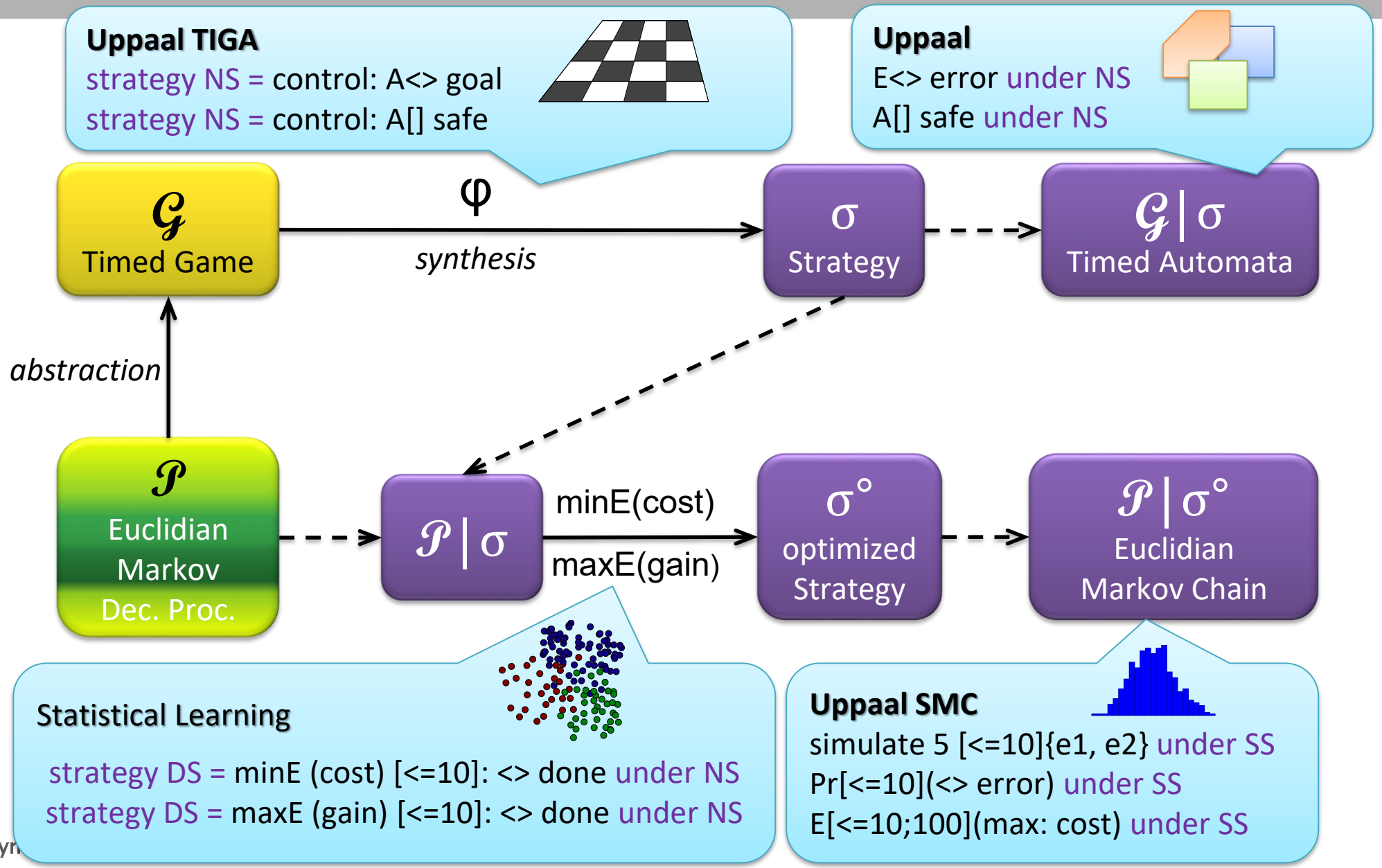
Farming Benchmark – in Stratego



Farming Benchmark – in Stratego



Workflow under UPPAAL Stratego



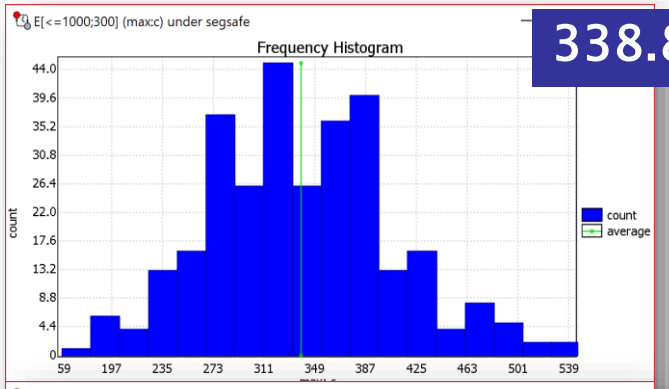
Smart Farming

Q1: strategy segsafe = control: A[] ! (Robot(1).s>1 && Robot(1).s<14 && Robot(1).s==Robot(2).s)

Q2: strategy opt_harvest =
maxE(c) [<=1000] {Robot(1).location, Robot(2).location, Rain.location} -> {ld1,ld2,t}:
<> t >= 1000 under segsafe

Q3: E[<=1000;300] (max:c) under segsafe

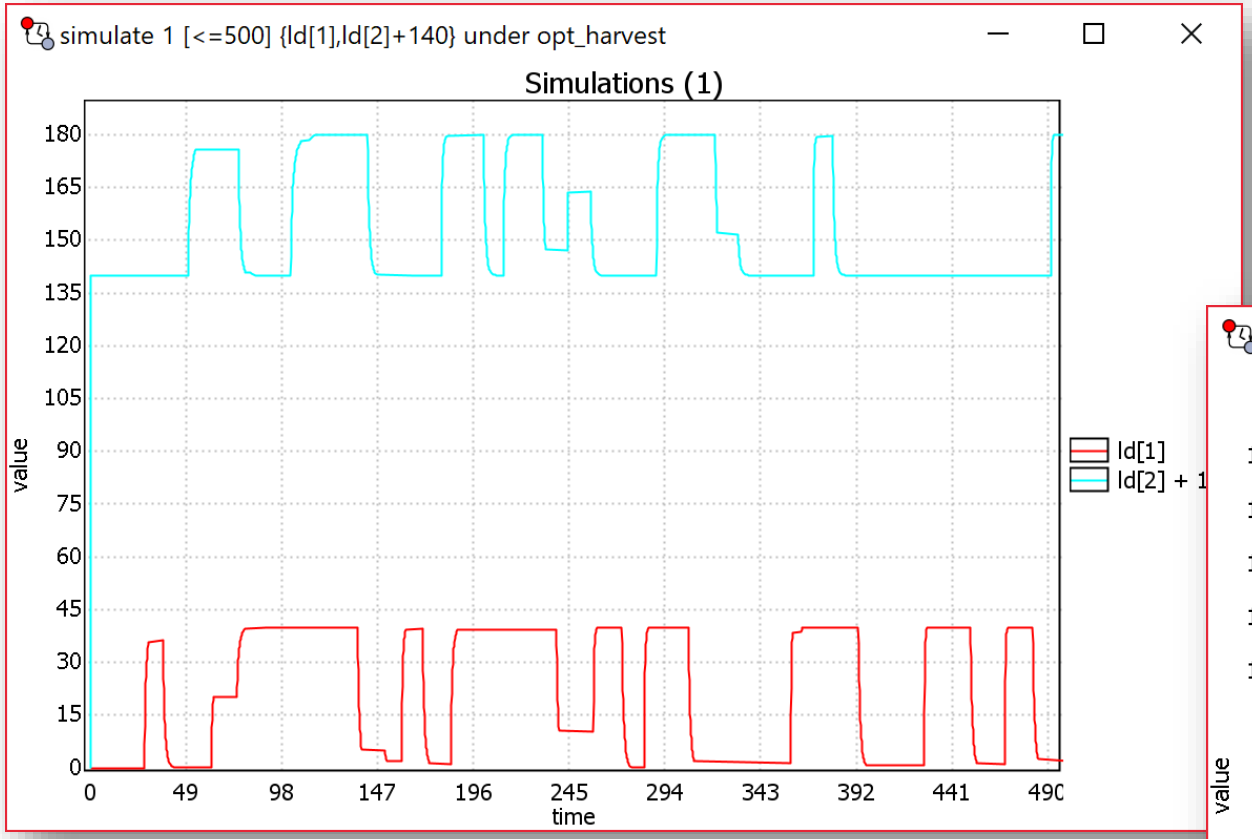
Q4: E[<=1000;300] (max:c) under opt_harvest

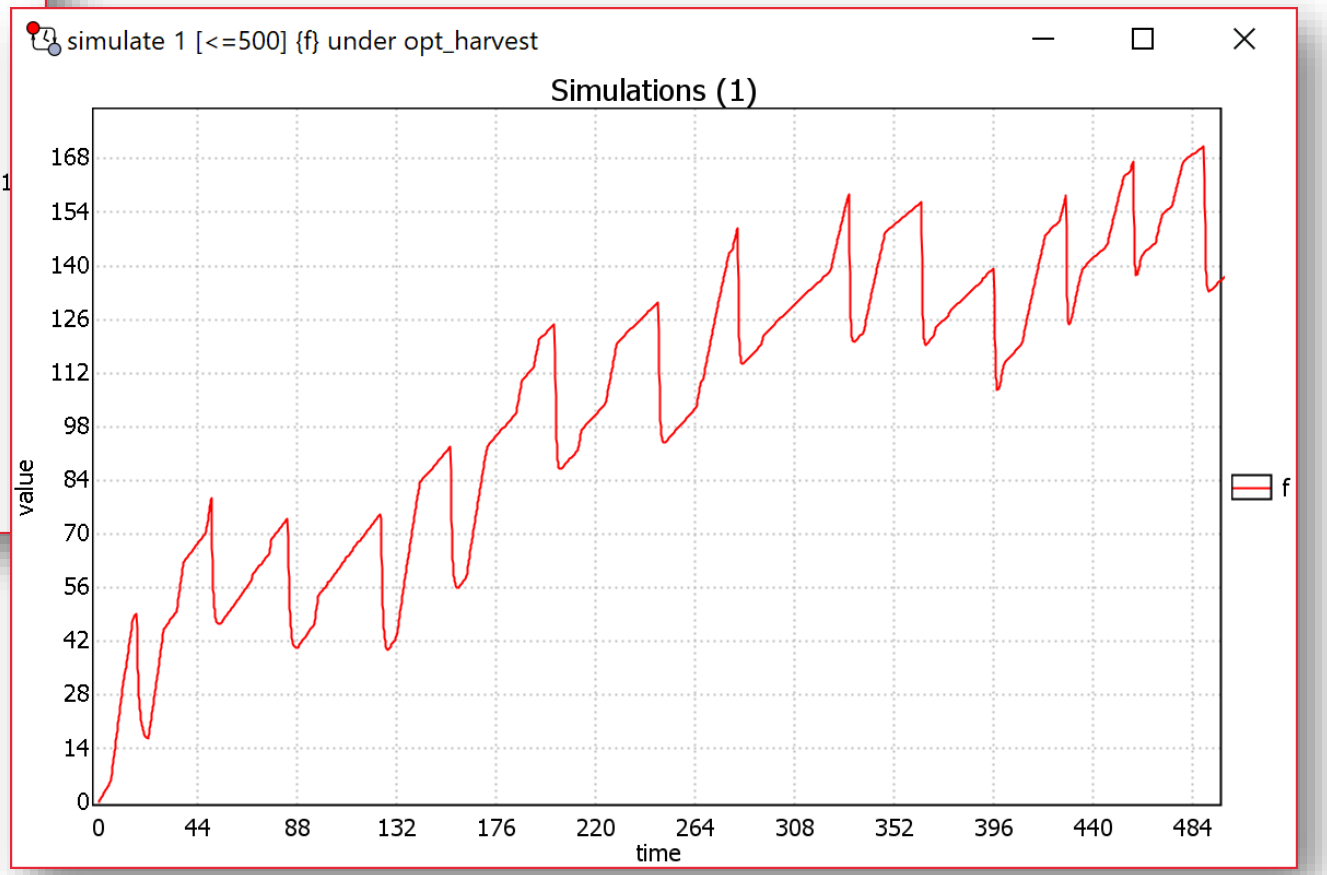


338.881 +/- 7.80819

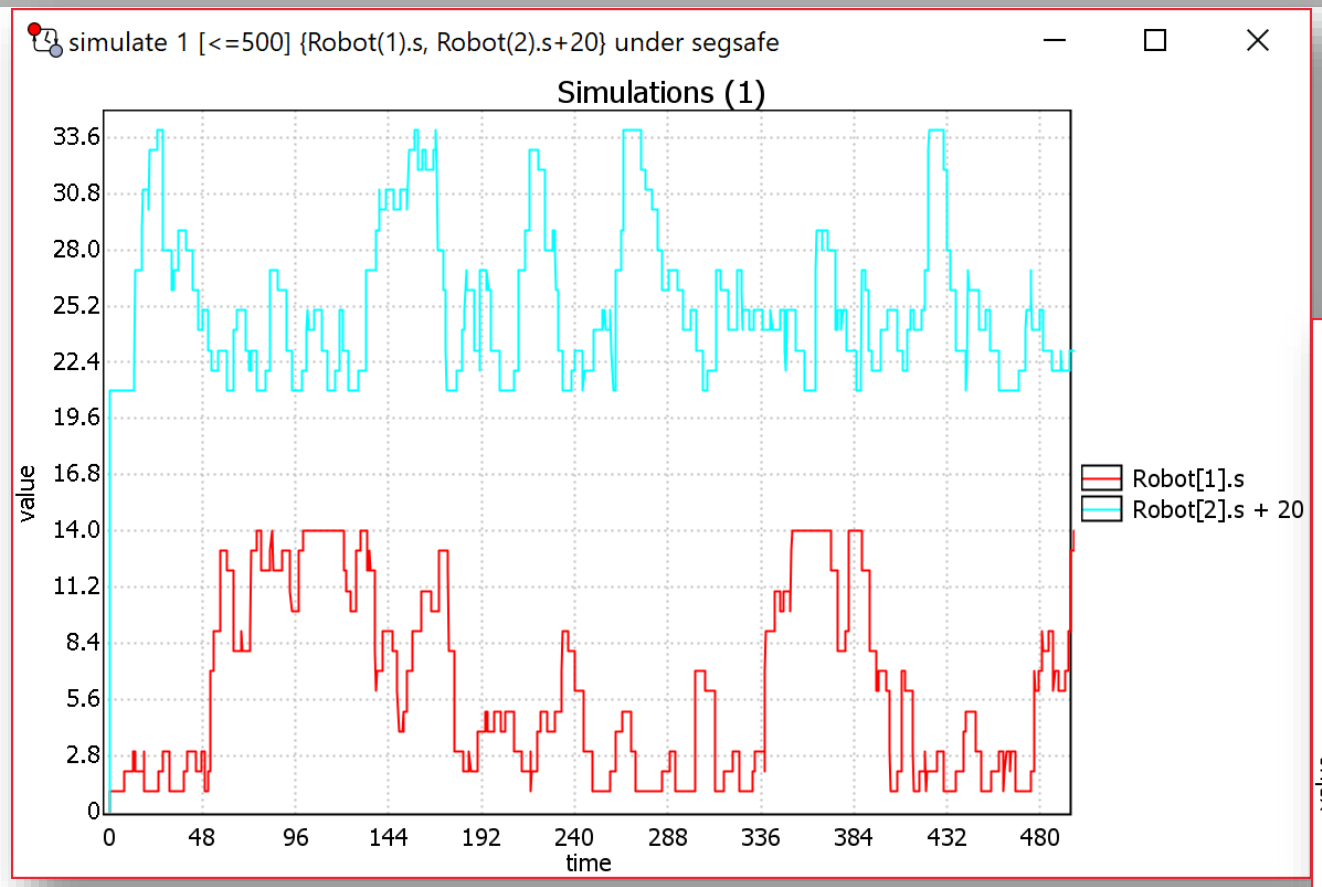
969.112 +/- 13.8988

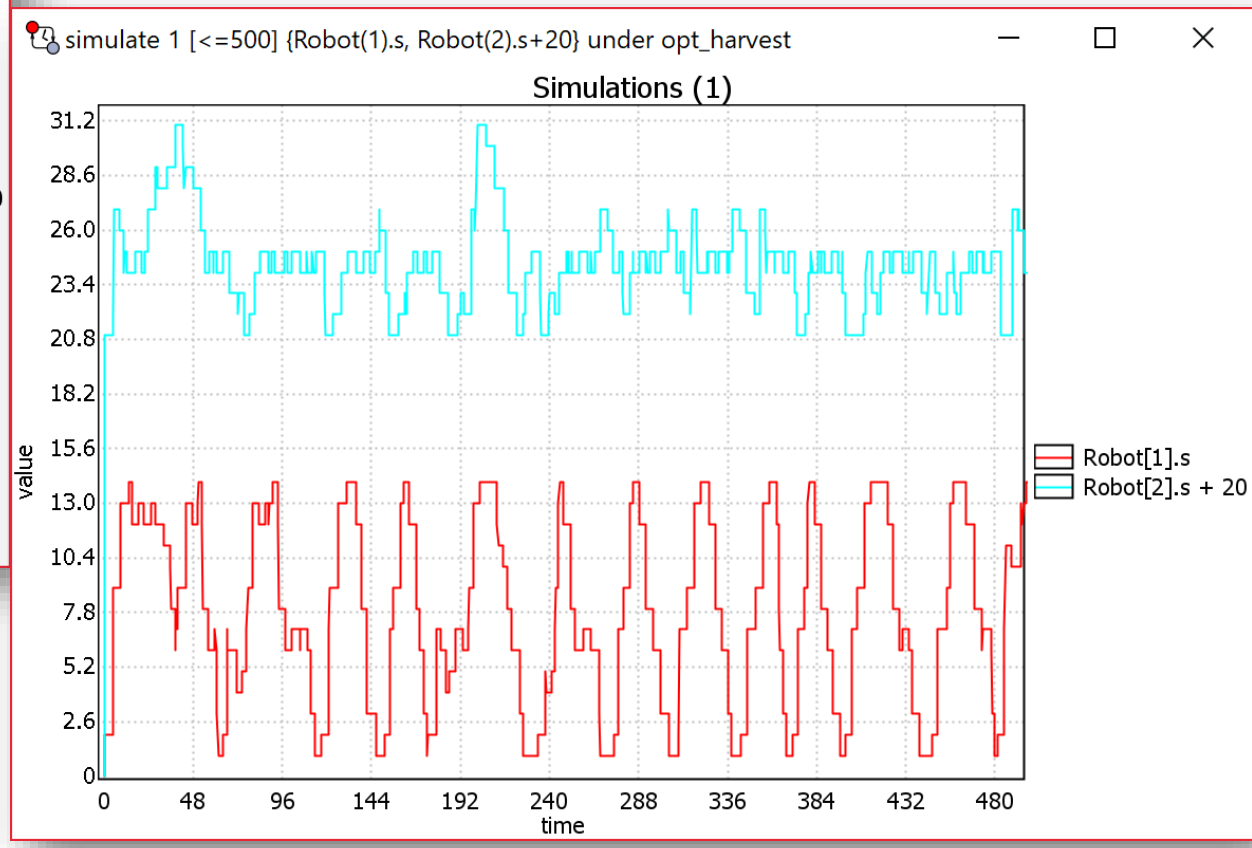
ld[1], ld[2] – f


simulate 1 [<=500] {ld[1],ld[2]+140} under opt_harvest
Simulations (1)
value
time
ld[1]
ld[2] + 1


simulate 1 [<=500] {f} under opt_harvest
Simulations (1)
value
time
f

Robots Movement

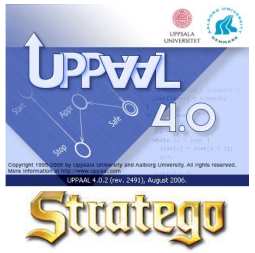




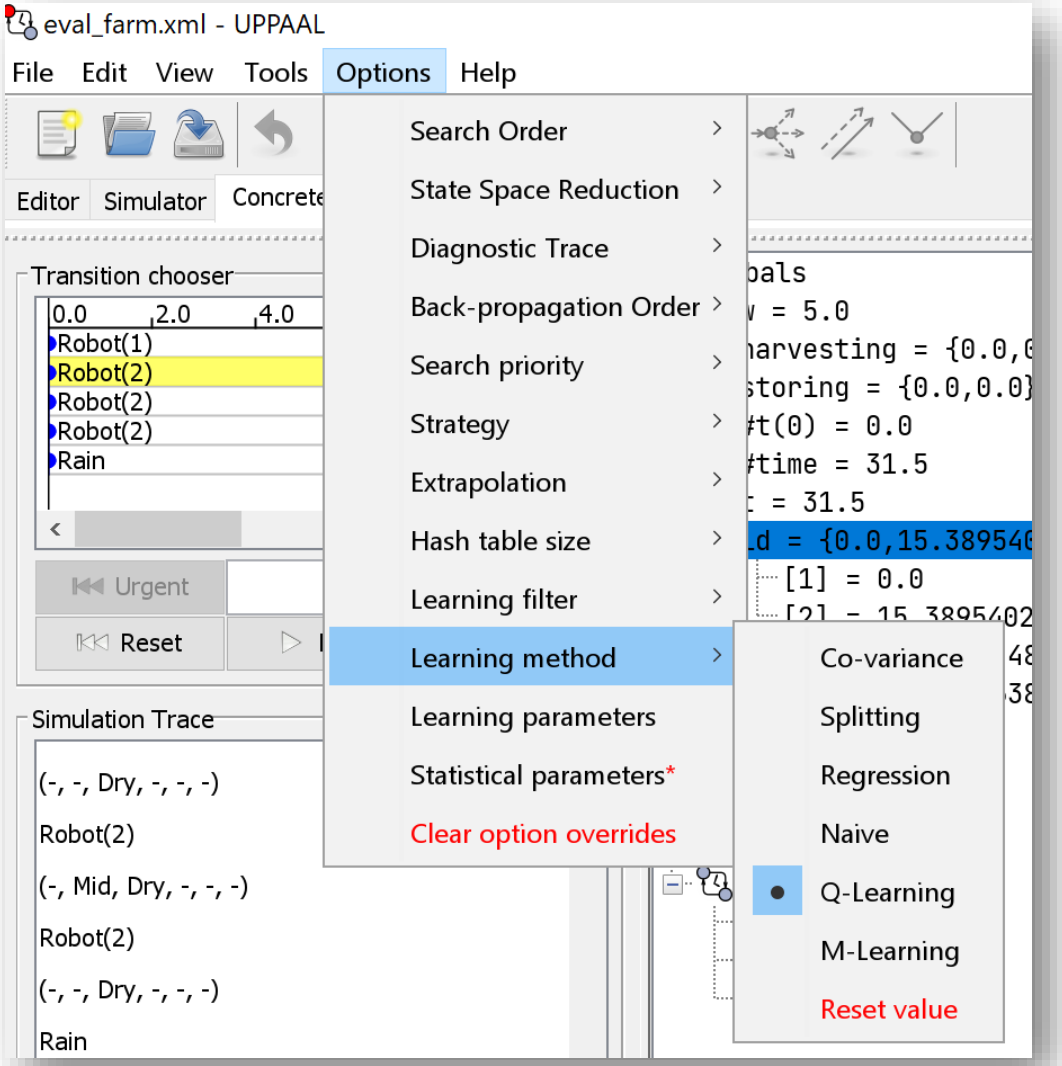
Better Learning

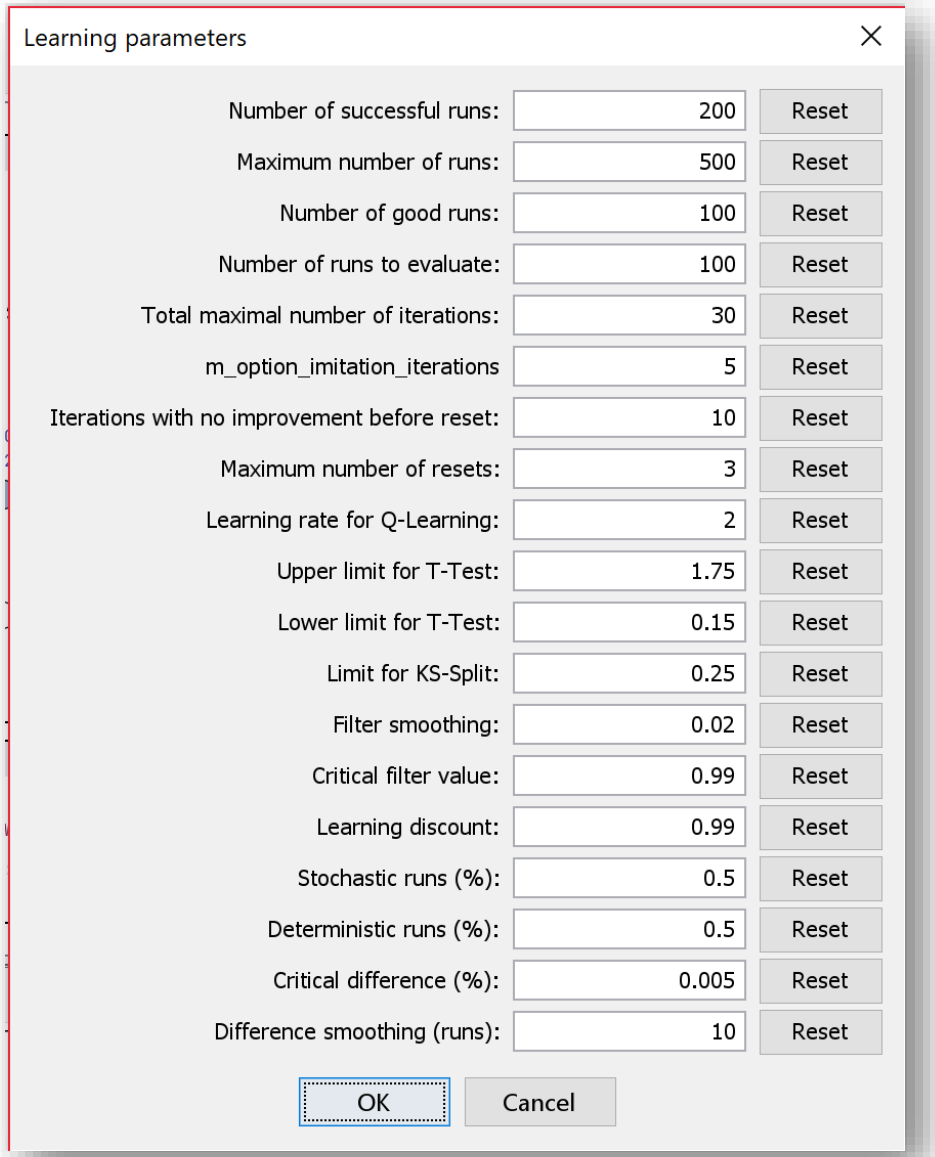
AALBORG UNIVERSITET

VILLUM FONDEN

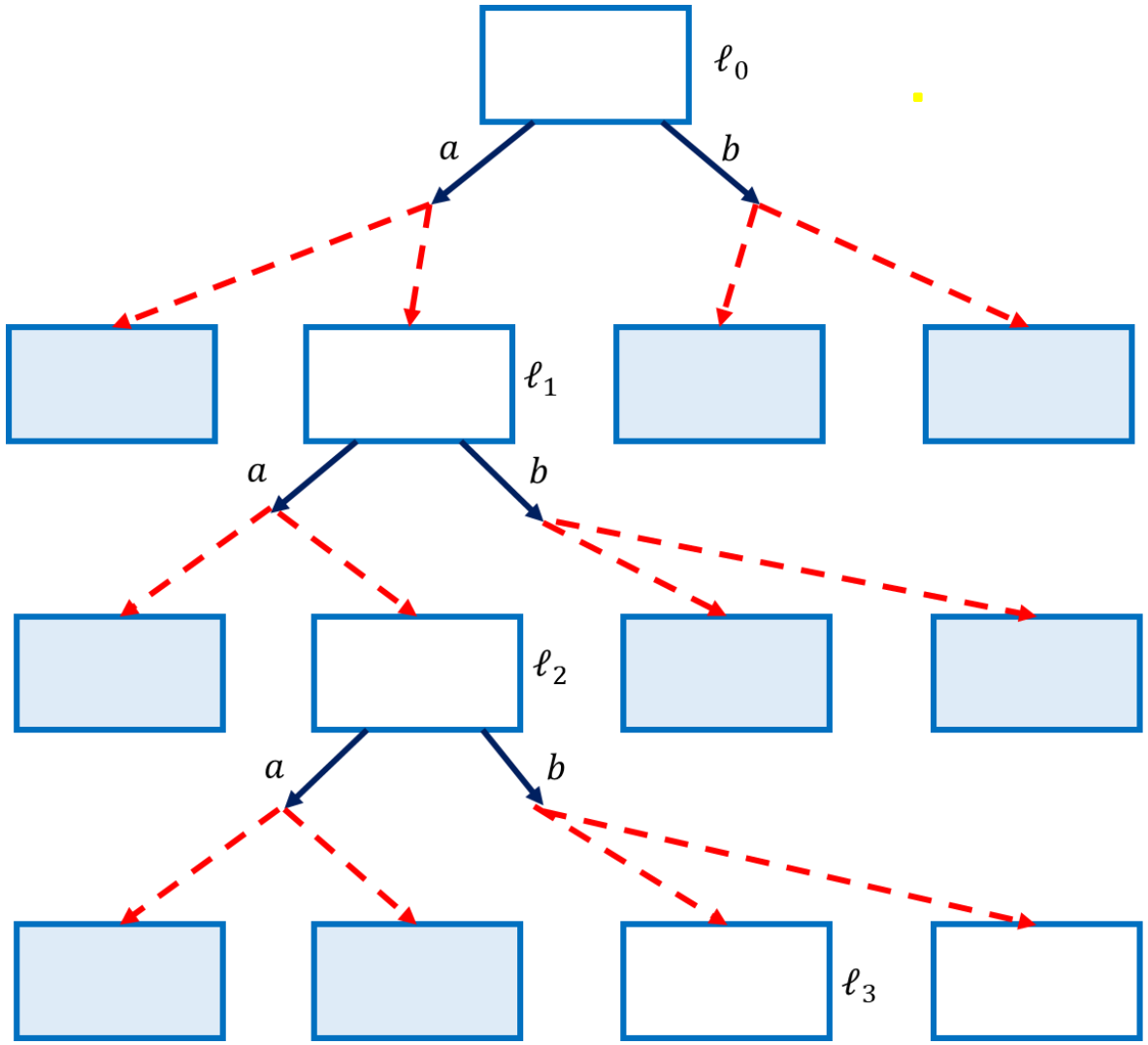


Learning Methods & Parameters





Euclidean MDP



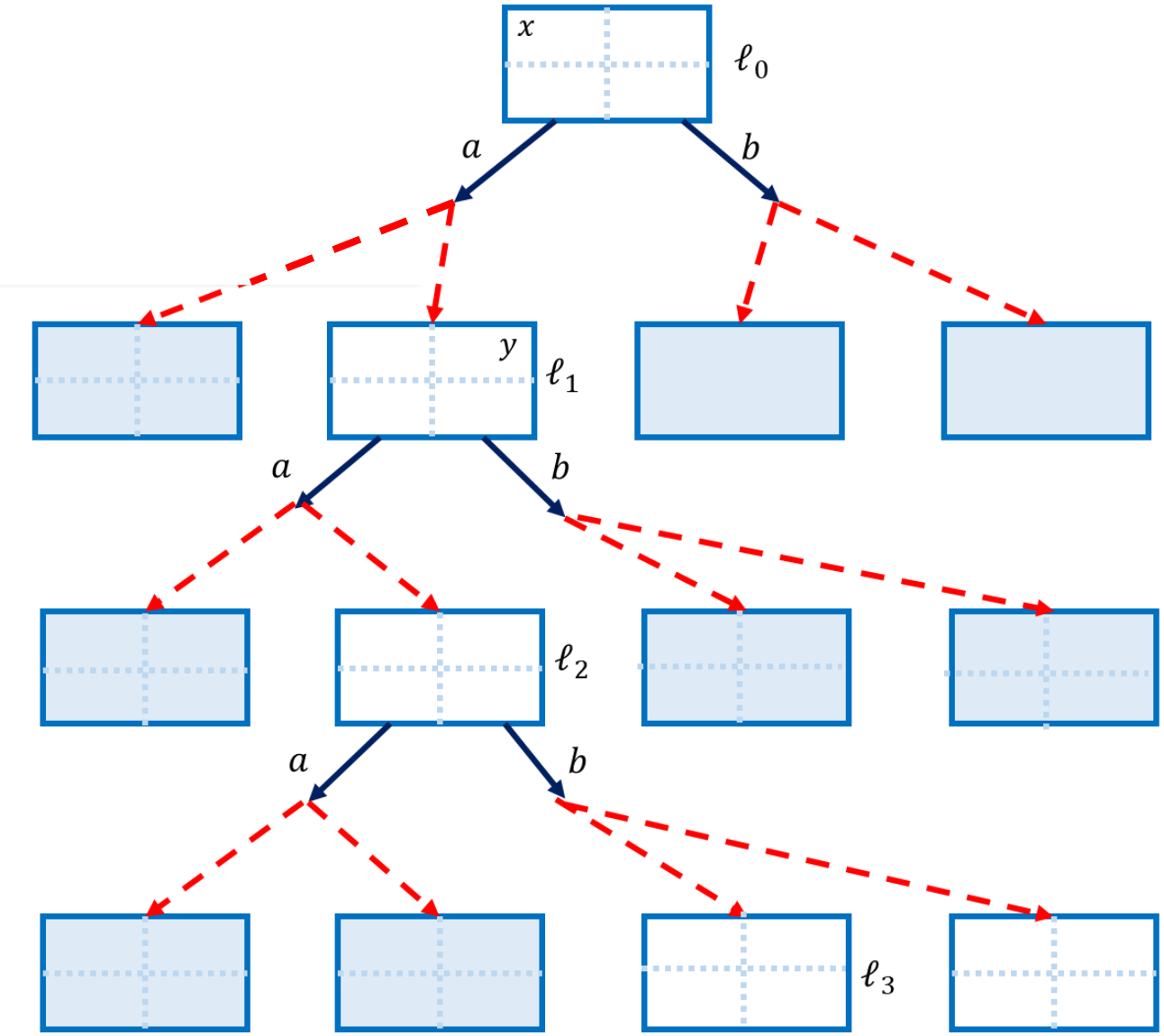
- States $S \subseteq \mathbb{R}^k$
- Act is a finite set of actions
- Initial state $s_0 \in S$
- Next state density function
  $T: S \times Act \to (S \to \mathbb{R}_{\geq 0})$
- Cost-function
  $C: S \times Act \times S \to \mathbb{R}$
- Goal states $G \subseteq S$

Strategy $\sigma: S \to (Act \to [0,1])$

$\mathbb{E}_\sigma(G, s)$:
Expected cost of reaching G under σ.

Find σ^* st. $\mathbb{E}_{\sigma^*}(G, s) = \inf_\sigma \mathbb{E}_\sigma(G, s)$

Partitioning (IMDP)



- Finite partition $\boldsymbol{A}$ (respecting G)
- Next state transition-function
  $\boldsymbol{T_A: A \times Act \times A \rightarrow [0,1] \times [0,1]}$
- Cost-function
  $\boldsymbol{C_A: A \times Act \times A \rightarrow \mathbb{R} \times \mathbb{R}}$

$$\mathbb{E}_A^{min}(G,x) \qquad \mathbb{E}_A^{max}(G,x)$$

Theorem

$$\mathbb{E}_A^{min}(G,[s]) \leq \inf_\sigma \mathbb{E}_\sigma(G,s) \leq \mathbb{E}_A^{max}(G,[s])$$

Theorem
(Bounded horizon, continuous cost & transition)
Let $A_0 \sqsubseteq A_1 \sqsubseteq \cdots \sqsubseteq A_i \sqsubseteq \cdots$ a be refining sequence "of arbitrary precision". Then

$$\inf_{i\to\infty} \mathbb{E}_{A_i}^{min}(G,[s]_{A_i}) = \inf_\sigma \mathbb{E}_\sigma(G,s) = \inf_{i\to\infty} \mathbb{E}_{A_i}^{max}(G,[s]_{A_i})$$

Q- & M-Learning

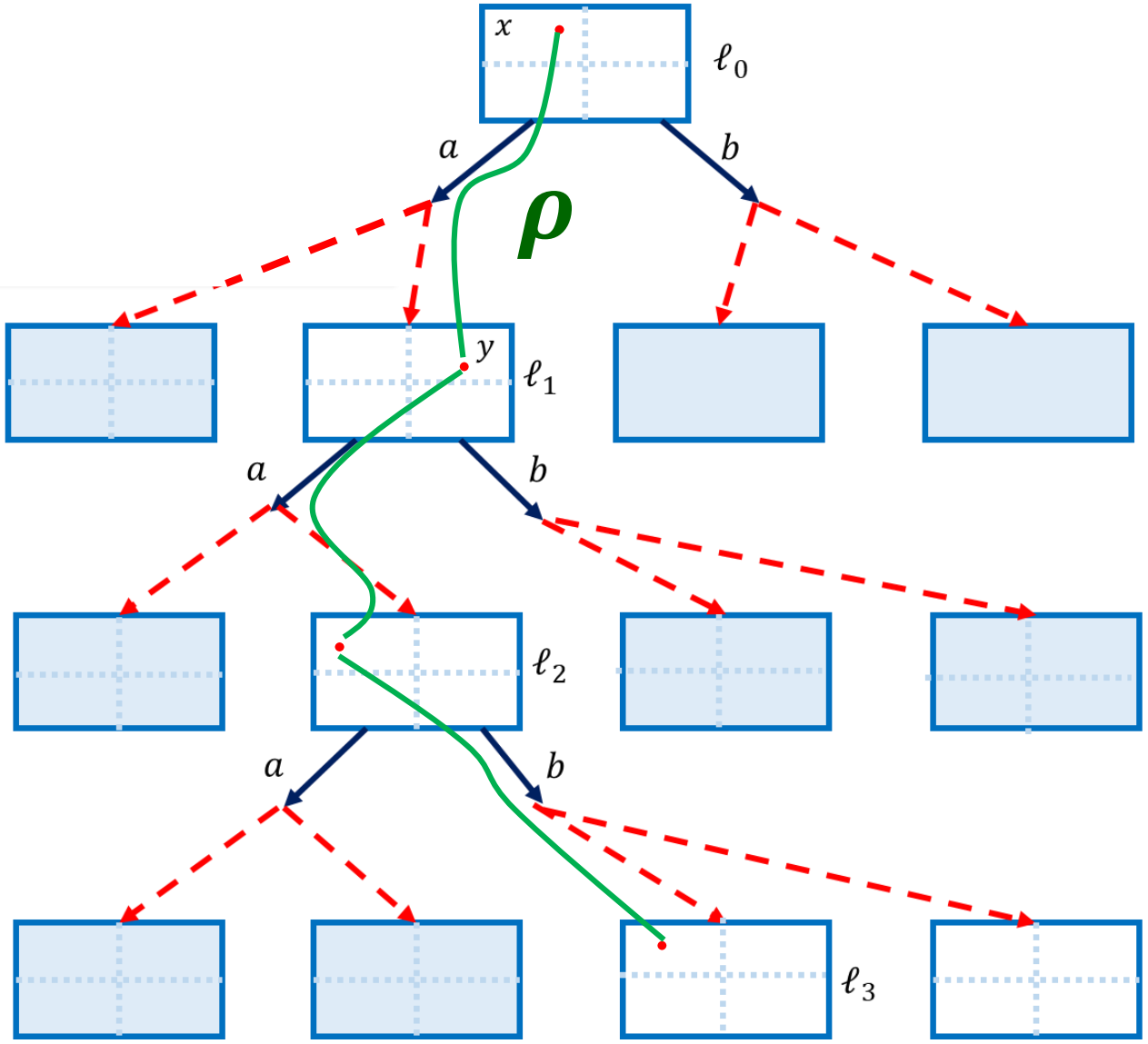
$Q_a^{\ell,x}, Q_b^{\ell,x} : \mathbb{R}_{\geq 0} \quad x \in \mathcal{P}$

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

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

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

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



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.

Q- and M-Learning

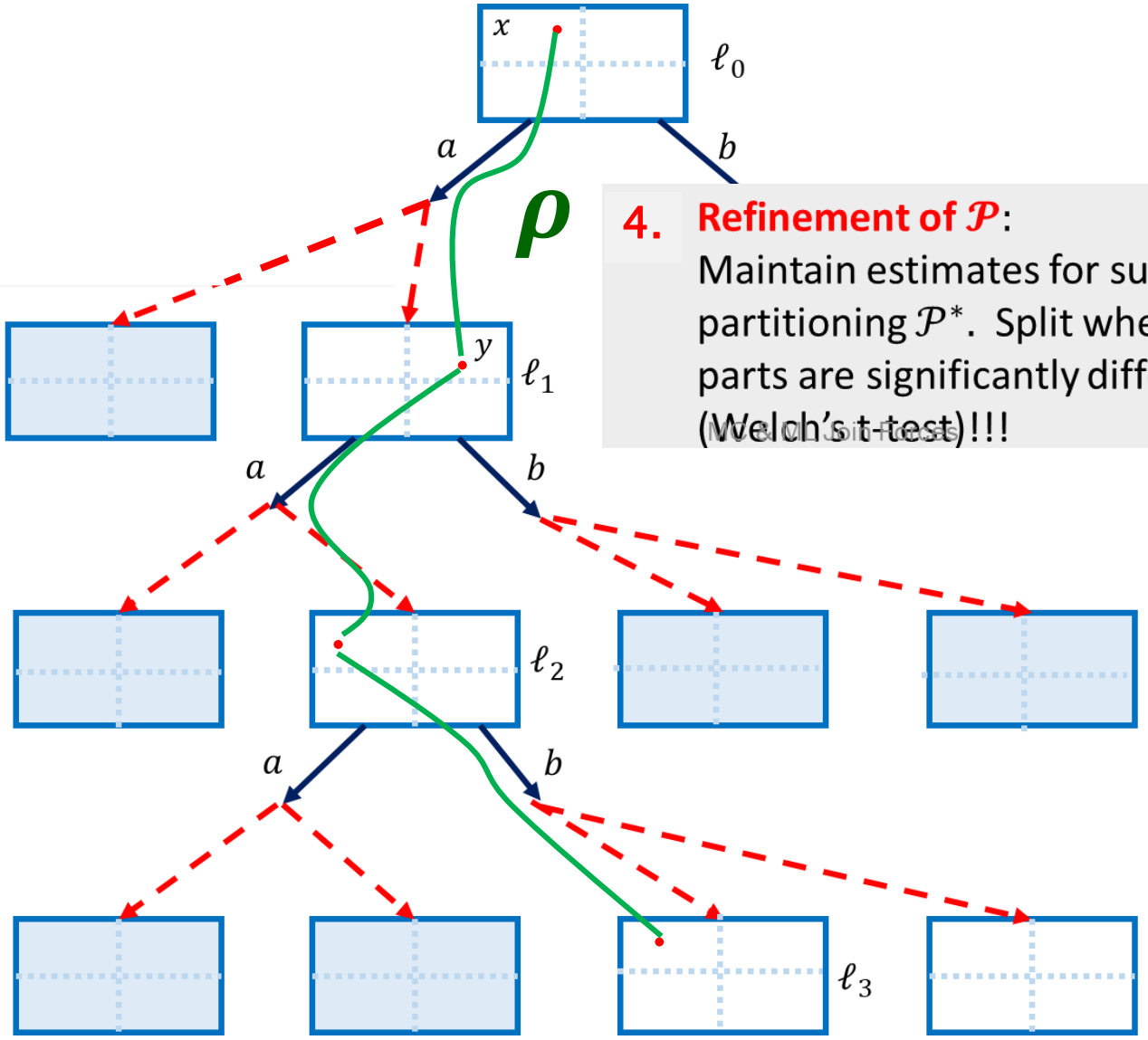
$Q_a^{\ell,x}, Q_b^{\ell,x} : \mathbb{R}_{\geq 0} \quad x \in \mathcal{P}$

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

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

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

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



4. **Refinement of $\mathcal{P}$:**
Maintain estimates for sub-partitioning $\mathcal{P}^*$. Split when sub-parts are significantly different (Welch's t-test)!!!

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 $Q_b^{\ell,x}$

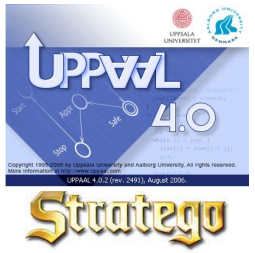
$$Q_a^{\ell_0,x} = \left(1 - \frac{1}{\#(\ell_0, x)}\right) \cdot Q_a^{\ell_0,x} + \frac{1}{\#(\ell_0, x)} \cdot \left(C(s_0, a, s_1) + \min_\alpha Q_\alpha^{\ell_1,y}\right)$$

Kim Larsen [24]

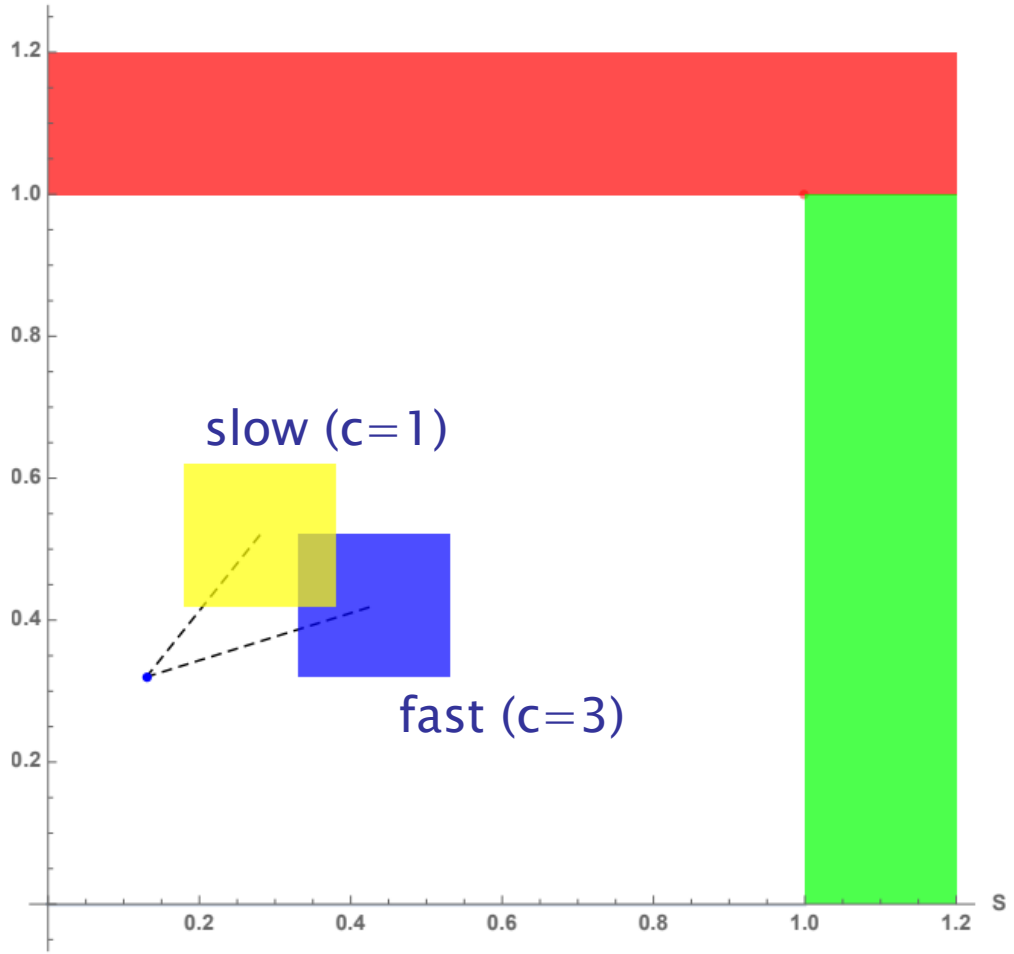
Approximations

AALBORG UNIVERSITET

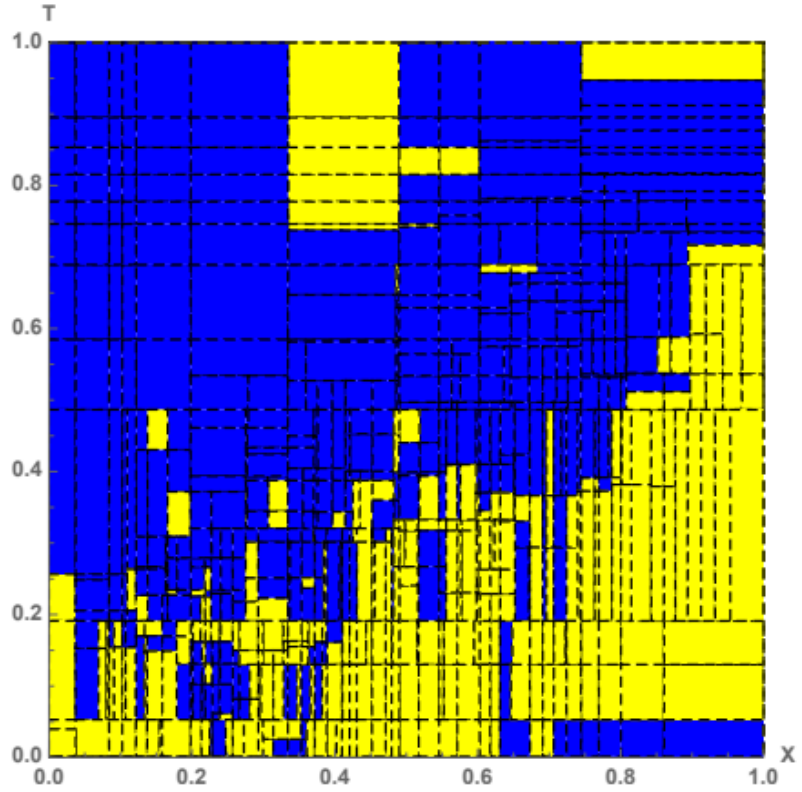
VILLUM FONDEN



Semi-Random Walk

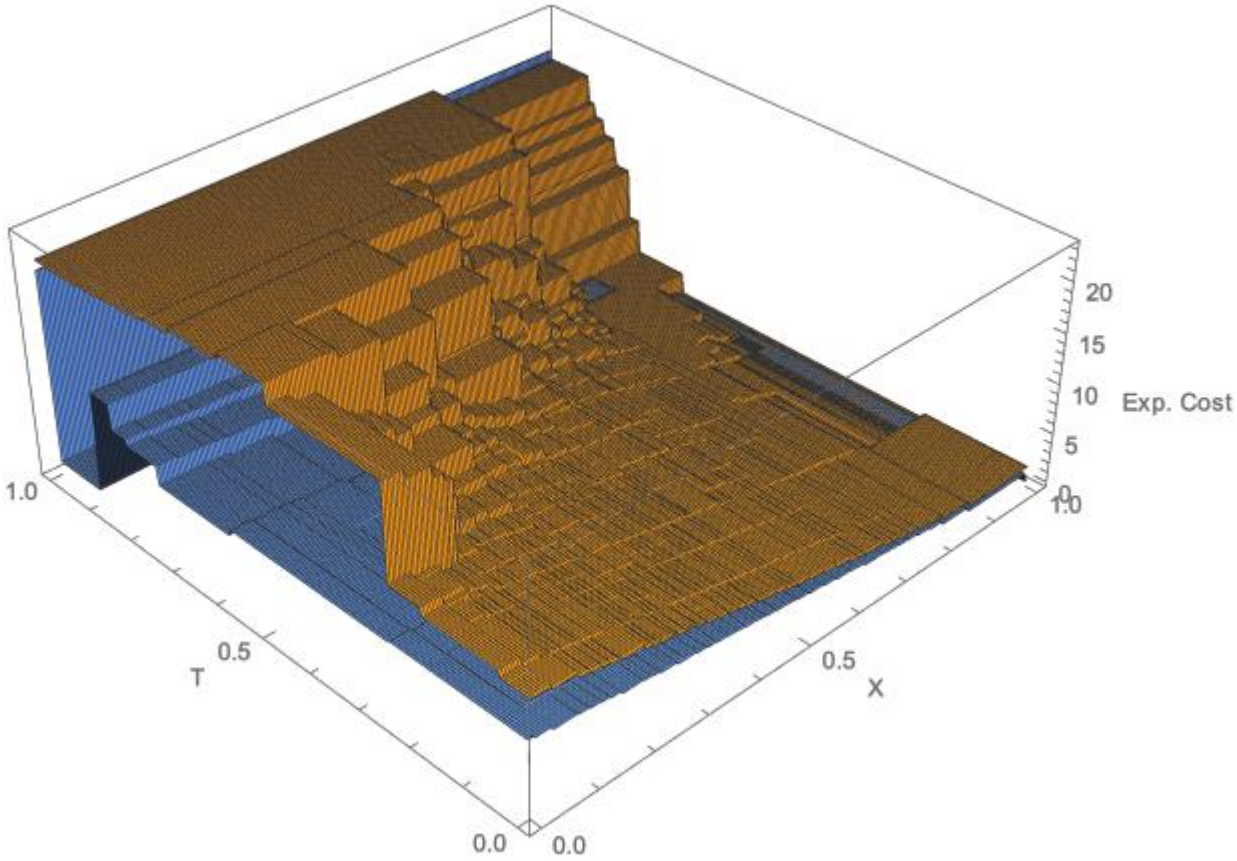


EMDP

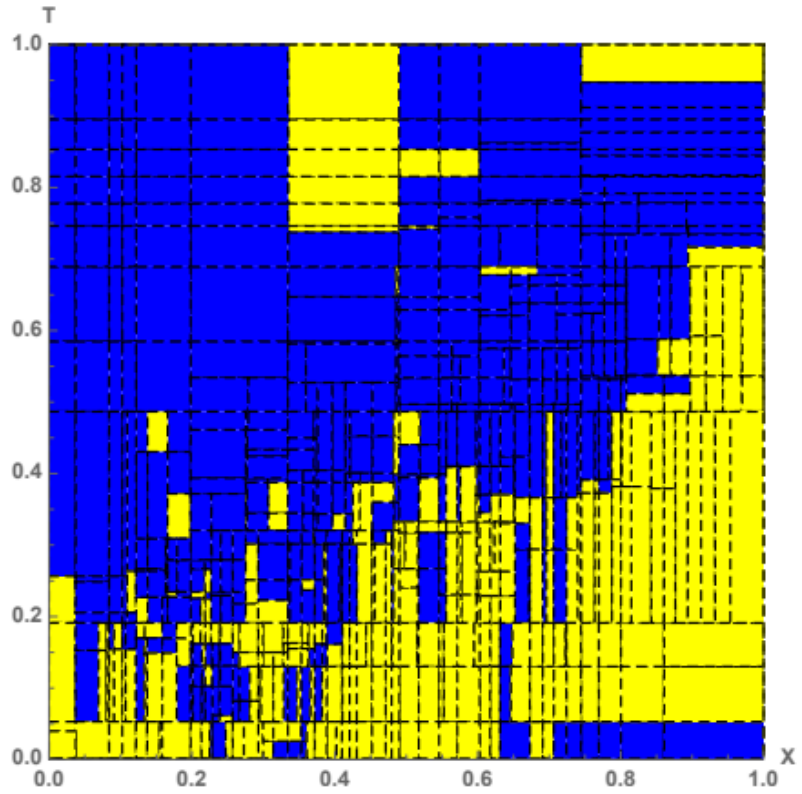


Strategy learned by STRATEGO

Semi-Random Walk

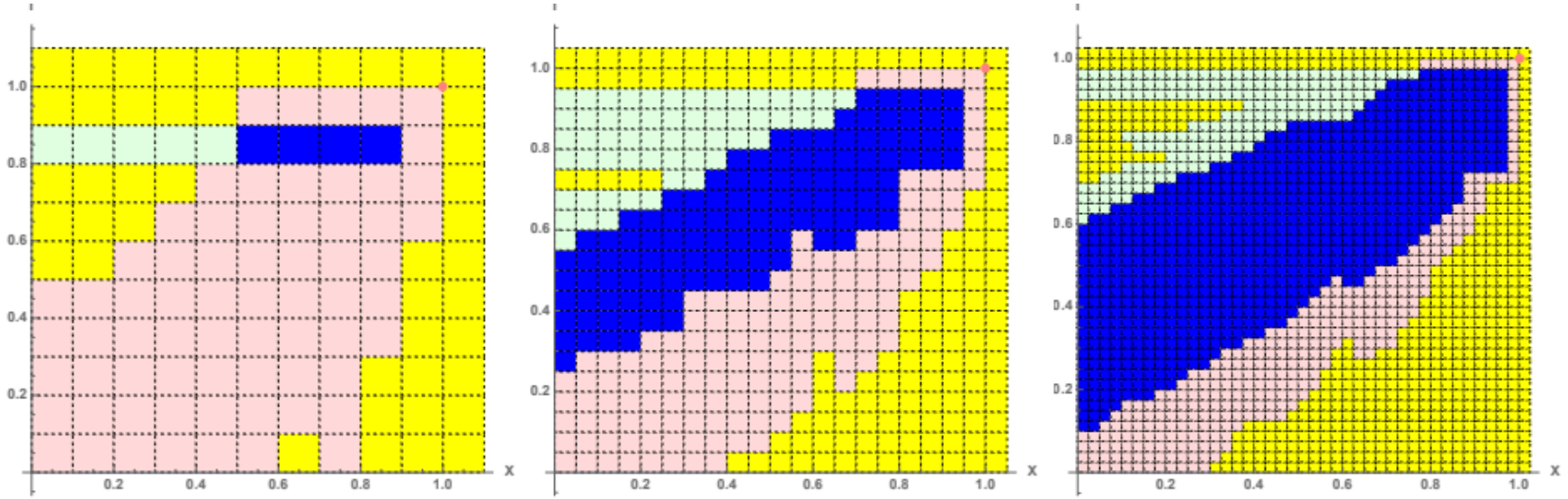


Lower/Upper Cost-bound



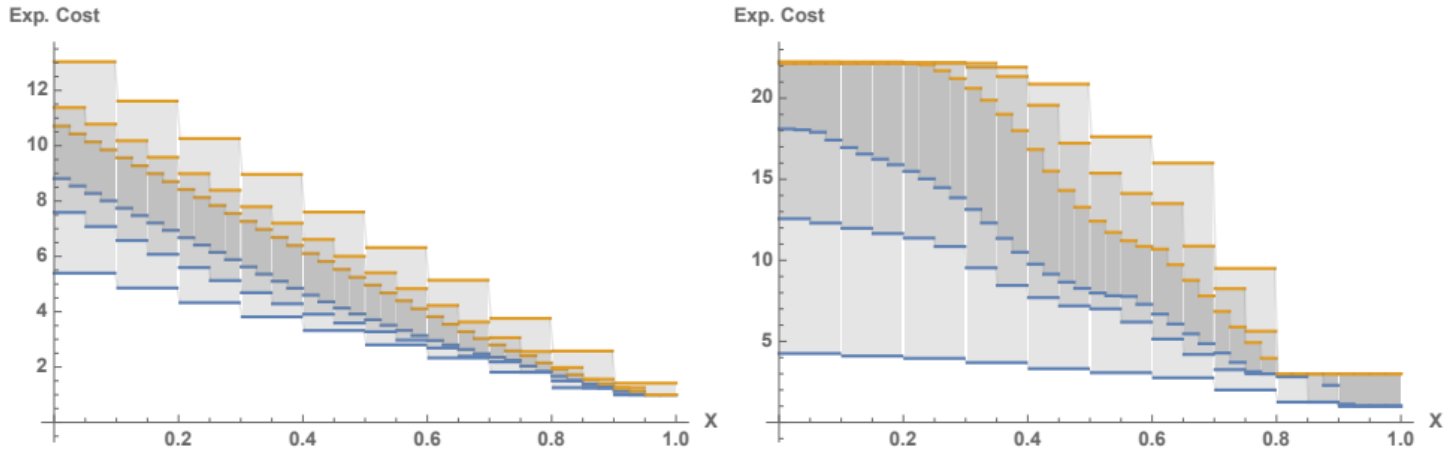
Strategy learned by STRATEGO

Strategies from Successive Partitioning

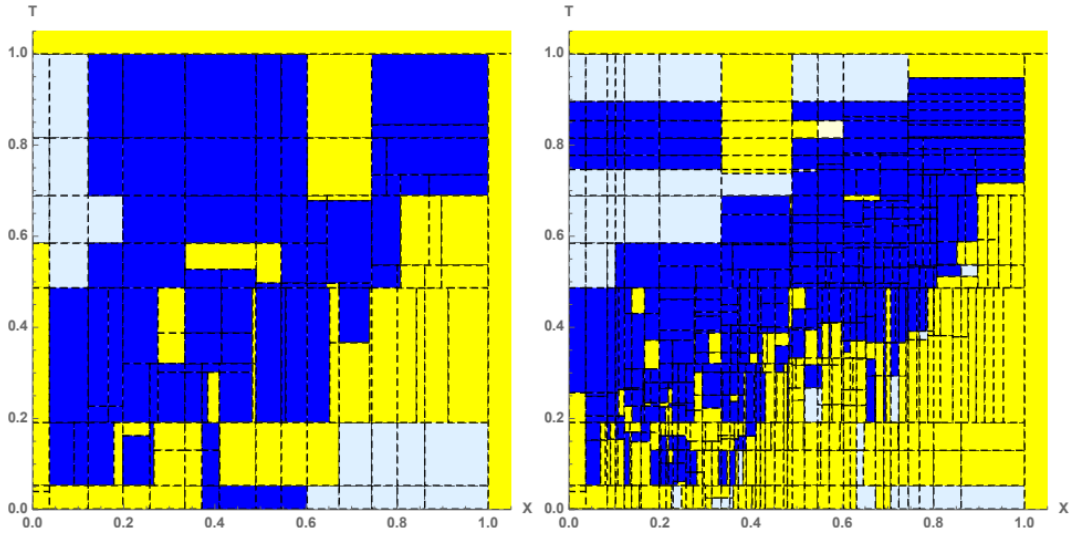


Obtained from lower and upper expected cost approximations

L/U expected costs
for t=0.0, t=0.7



Strategies from Learning



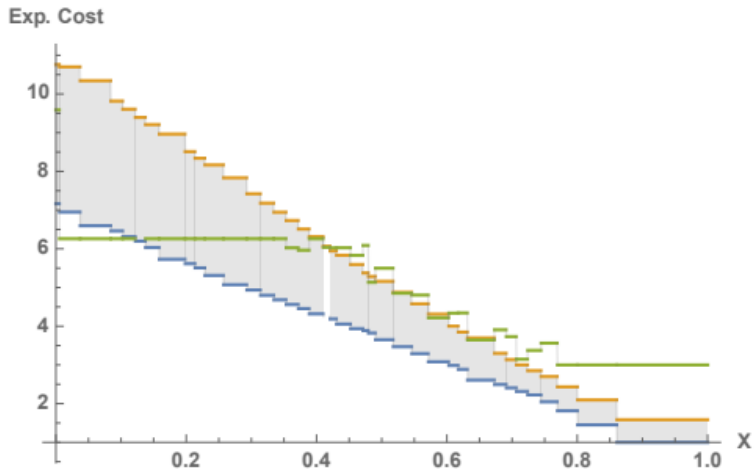
k=27

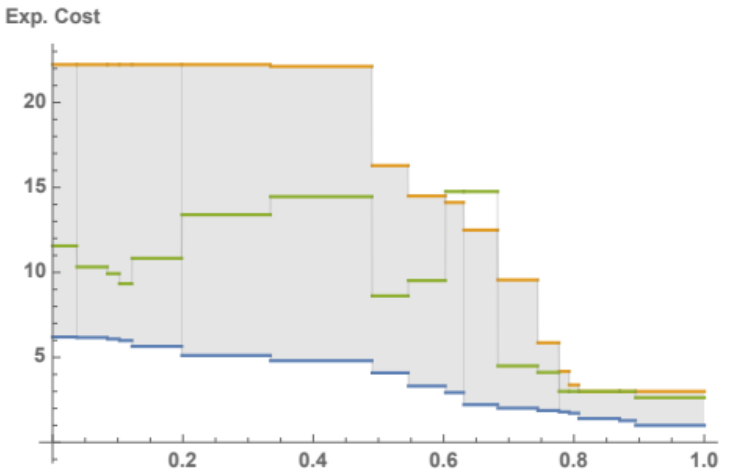
k=205

Expected cost functions for k=205
Along t=0.0, t=0.7.

Yellow: Upper expected cost
Blue: Lower expected cost
Green: Learned expected cost

CHALLENGE
Prove convergence of Q-learning for EMDPs

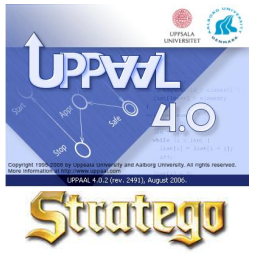




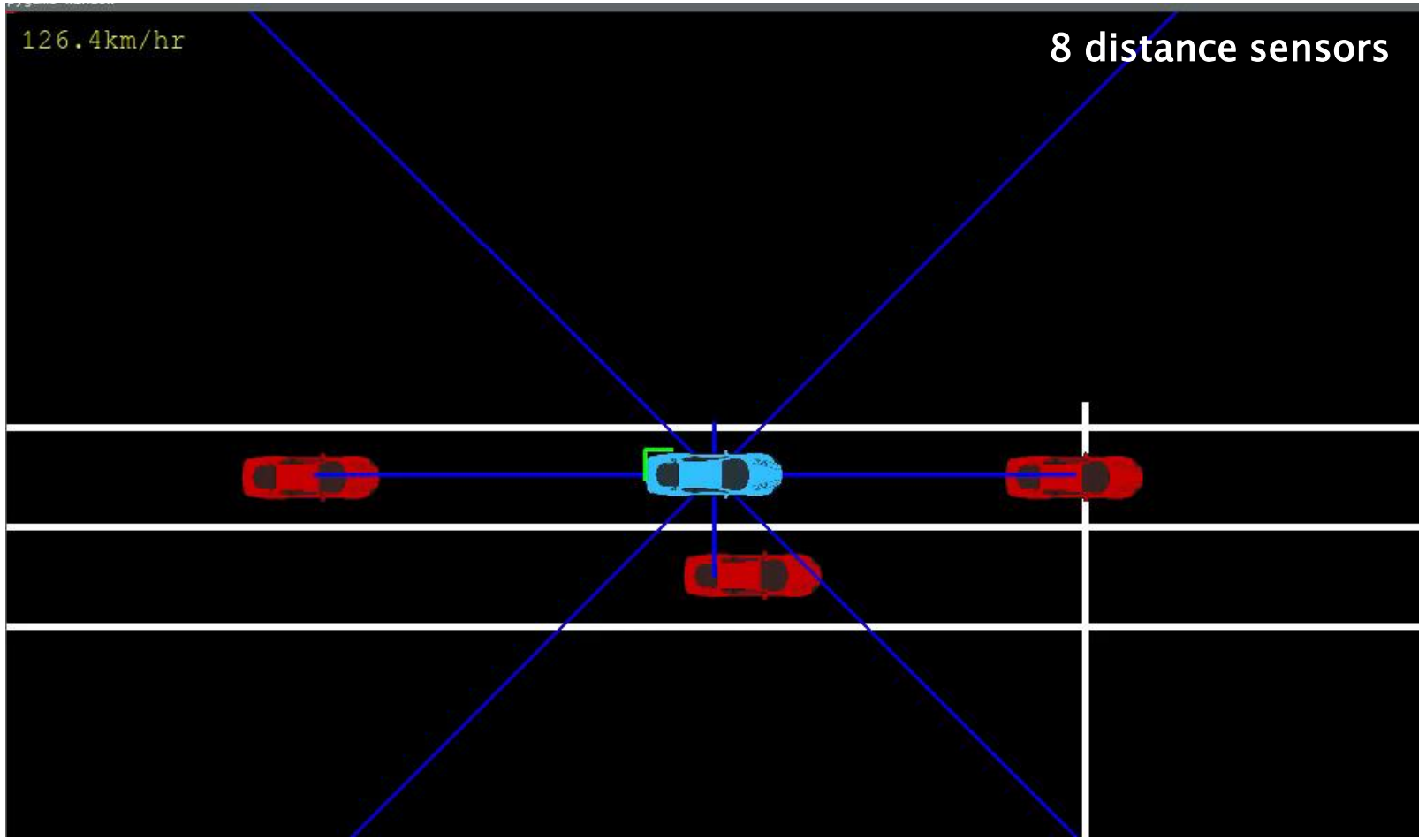
Compact Strategies

AALBORG UNIVERSITET

VILLUM FONDEN



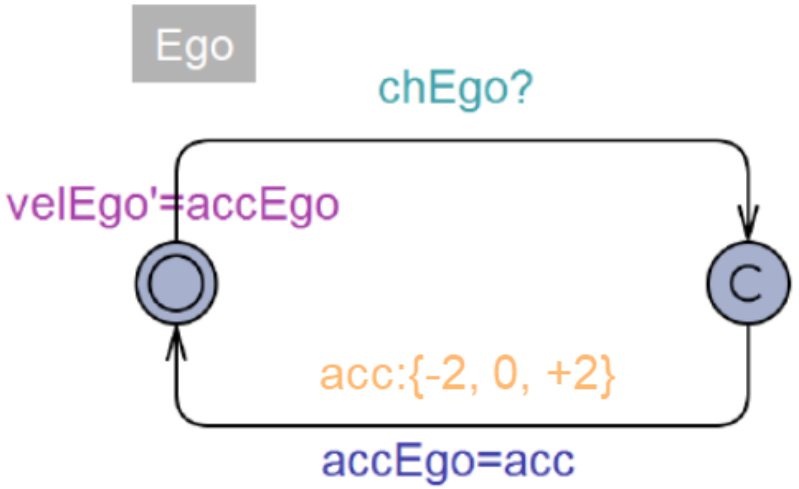
UPPAAL Adaptive Cruise Control



"Optimal" Strategy

Safe and "Optimal" Strategy

Strategy – Explicit



adaptiveCruiseControl - Notepad

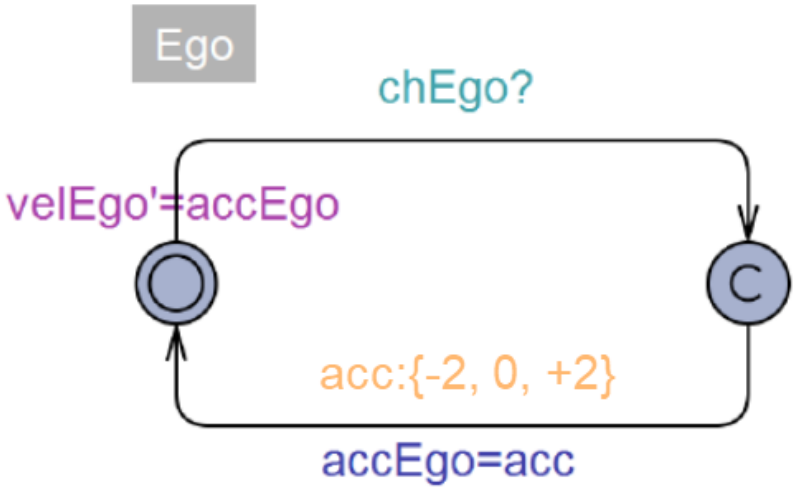
File Edit Format View Help

```
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
velocityEgo=13 accelerati                  Front=14 acceler
While you are in

State: ( Ego                                              Next
distance=1                 0 accelerationEgo
When you                   transition Ego.Choo            cc { 1, tau, accelerationEgo := 0
}
When yo                    tion Ego.Choose->E             ve_acc { velocityEgo <
maxVel                     := 2 }
When y                     go.Choose->Eg                  e_acc { velocityEgo >
minVel
State:                     _id12 ) #action=0 distance=199
velocity                   ionFront=0
While you                  true, wait.

State: ( Ego.                                  ne Monitor._id12 ) #action=0
distance=49 velo                               tyFront=14 accelerationFront=2
While you are in

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

6 mio configurations

Strategy – Decision Tree




```
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 (331464.0/20857
|  |  |  |  |  |  velocityEgo > 0
|  |  |  |  |  |  |  distance <= 122
|  |  |  |  |  |  |  |  distance <= 102: 2 (132918.0/8(
|  |  |  |  |  |  |  |  distance > 102
|  |  |  |  |  |  |  |  |  velocityEgo <= 2
|  |  |  |  |  |  |  |  |  |  velocityFront <= 12: 1 (625
|  |  |  |  |  |  |  |  |  |  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
```

Learning Algorithms for Decision Tree (ID3, D4.5, CART)

→ 65 lines

Jan Kretisnsky, Pranav Ashkot, TUM

[QEST19] SOS: Synthesis for Hybrid MDP

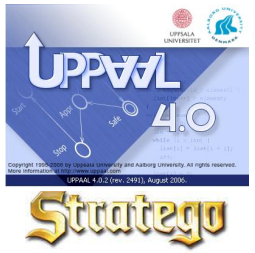
[TACAS21]
dtControl 2.0: Explainable Strategy Representation via Decision Tree Learning Steered by Experts.

Safe

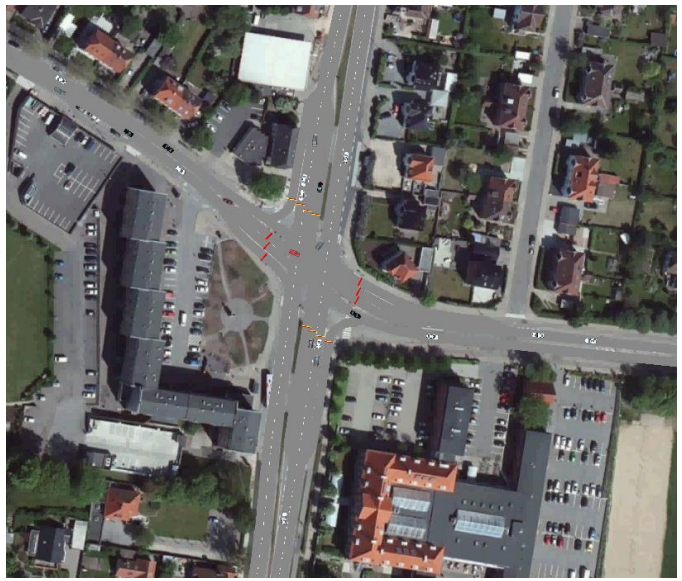
Applications

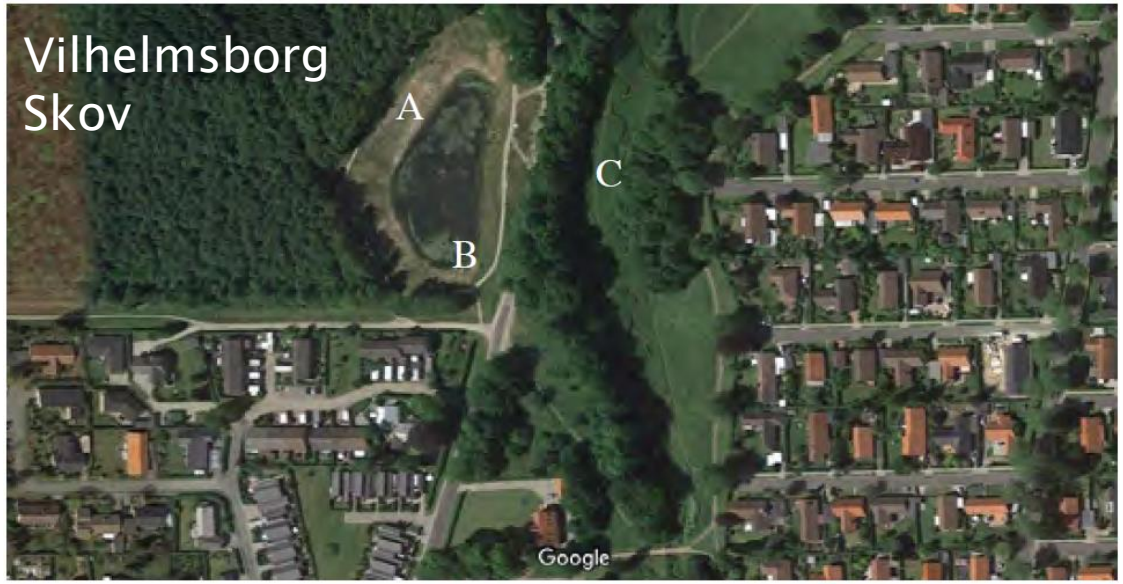
AALBORG UNIVERSITET

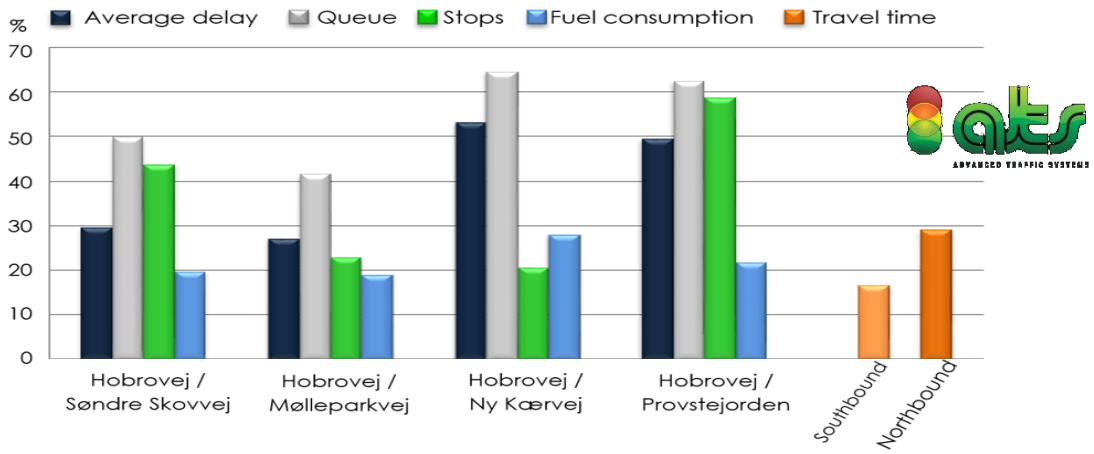
VILLUM FONDEN

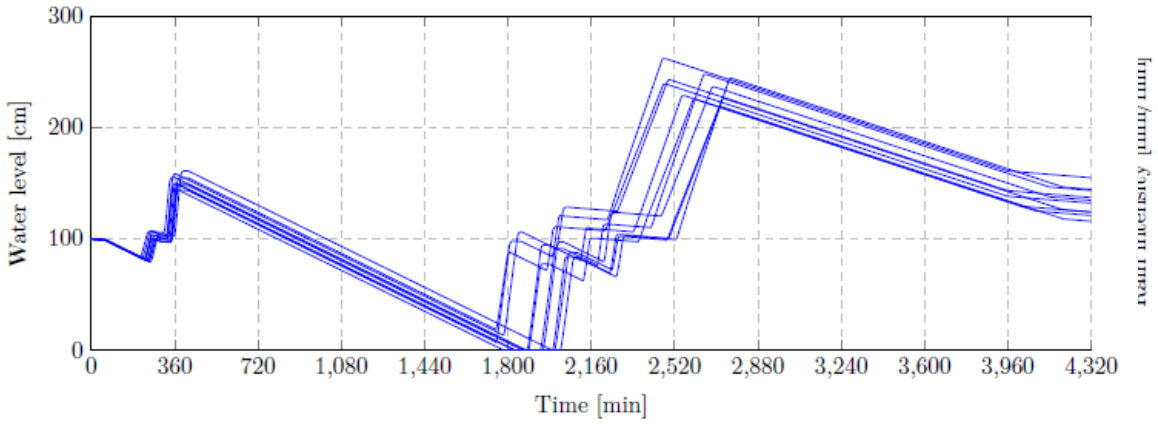


Intellingent Transport -- Smart Water



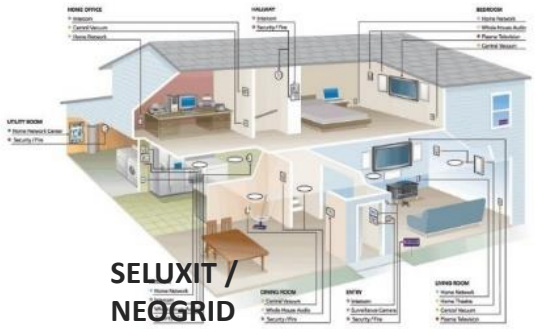






More Synthesis & Ongoing Research

- Convergence of Q-learning for IMDP.
- Partial Observability
- Learning strategy profiles for composite systems
- Online/Offline
- ..



VILLUM FONDEN