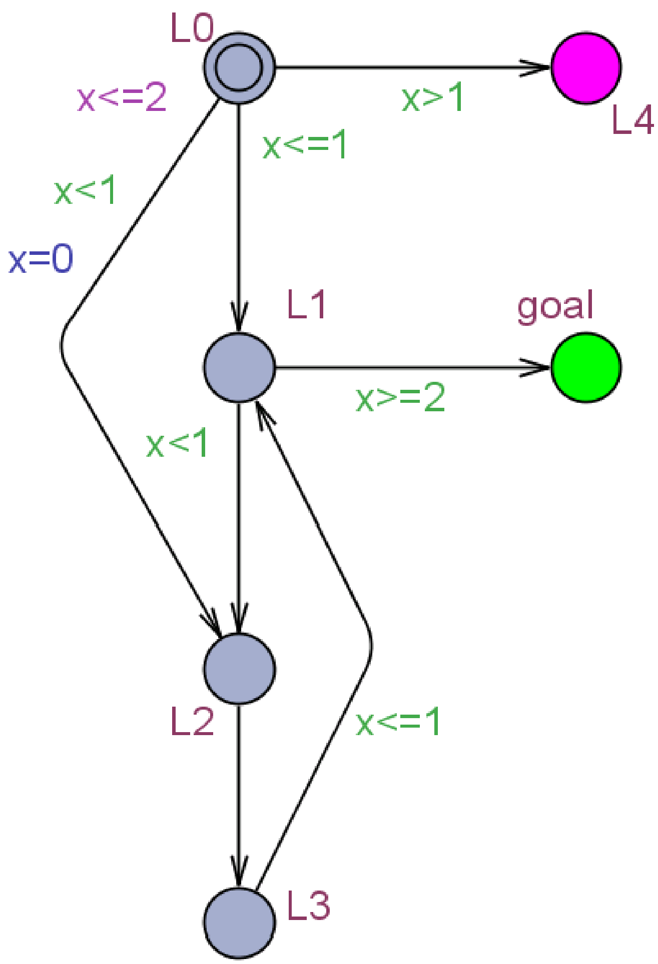
Timed Games & Timed Interfaces

TIGA

ECDAR

Kim G. Larsen
CISS – Aalborg University
DENMARK

Timed Automata & Model Checking



State (L1, x=0.81)

Transitions

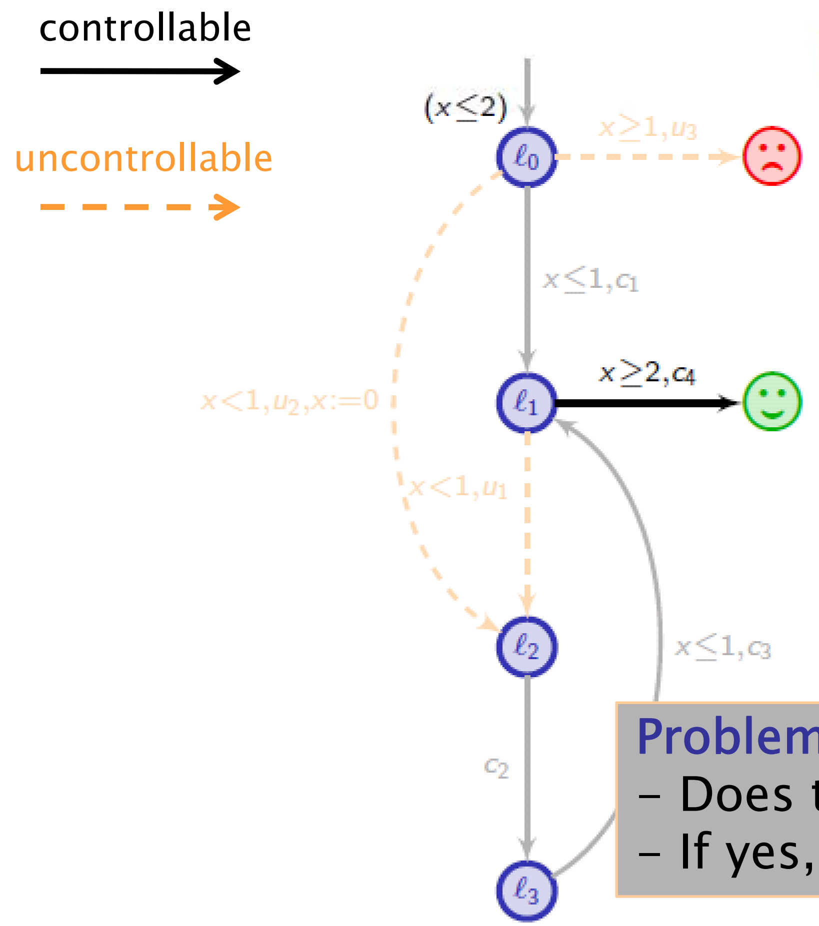
(L1 , x=0.81)
– 2.1 ->
(L1 , x=2.91)
->
(goal , x=2.91)

E⟨⟩ goal ?

A⟨⟩ goal ?

A[ ] ¬ L4 ?

Timed Game Automata & Synthesis



A (memoryless) winning strategy

- from $(\ell_0, 0)$, play $(0.5, c_1)$
  $\rightsquigarrow$ can be preempted by u_2

Problems to be considered:
- Does there exist a winning strategy?
- If yes, compute one (as simple as possible)

Decidability of Timed Games

Theorem [AMPS98,HK99]

Reachability and safety timed games are decidable and EXPTIME-complete. Furthermore memoryless and "region-based" strategies are sufficient.

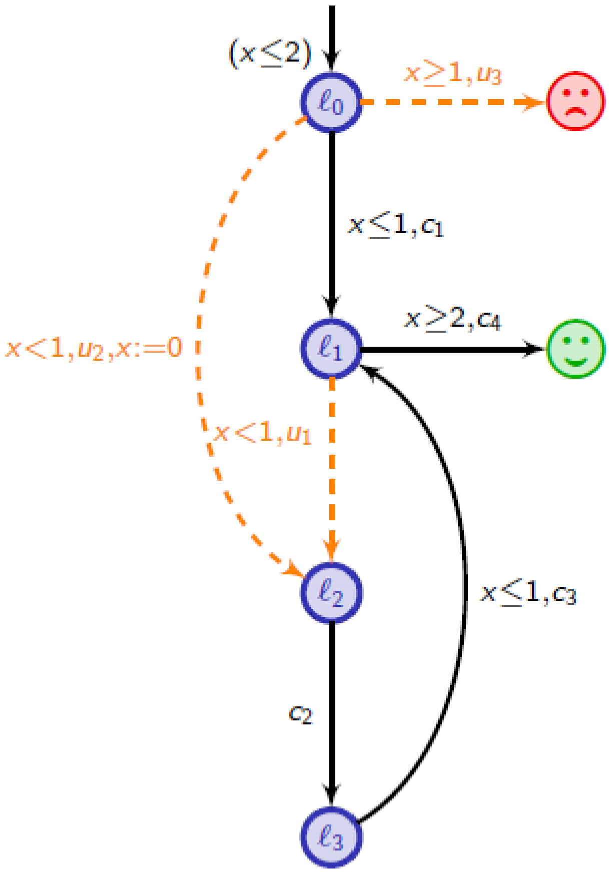
$\rightsquigarrow$ classical regions are sufficient for solving such problems

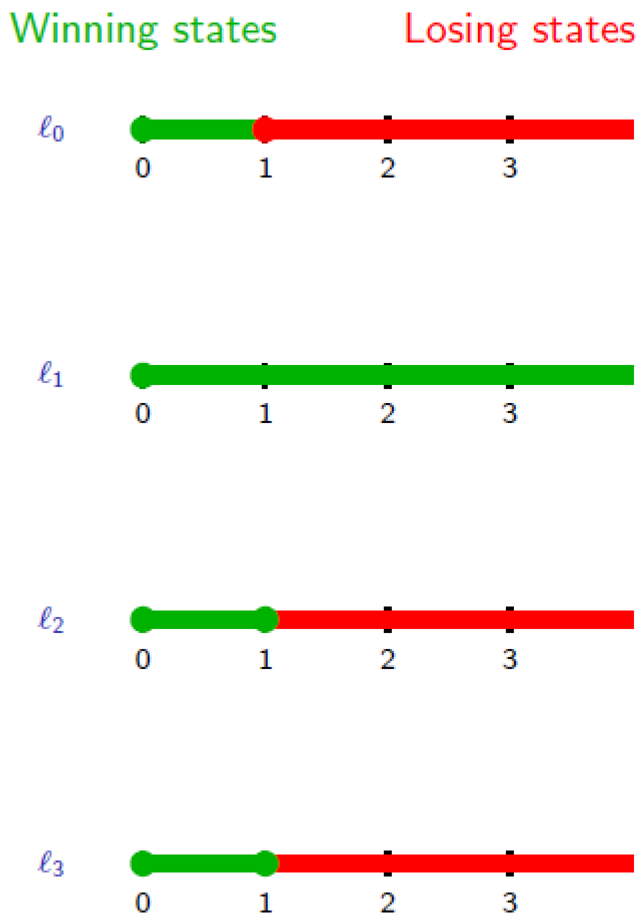
Theorem [AM99,BHPR07,JT07]

Optimal-time reachability timed games are decidable and EXPTIME-complete.

[AM99] Asarin, Maler. As soon as possible: time optimal control for timed automata *(HSCC'99)*.
[BHPR07] Brihaye, Henzinger, Prabhu, Raskin. Minimum-time reachability in timed games *(ICALP'07)*.
[JT07] Jurdziński, Trivedi. Reachability-time games on timed automata *(ICALP'07)*.

Computing Winning States





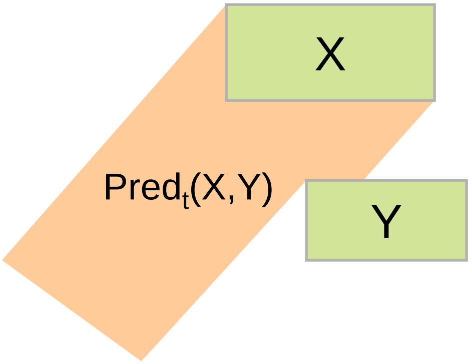
Reachability Games

Backwards Fixed-Point Computation

Definitions

$$\text{cPred}(X) = \{ q \in Q \mid \exists\, q' \in X.\ q \rightarrow_c q' \}$$
$$\text{uPred}(X) = \{ q \in Q \mid \exists\, q' \in X.\ q \rightarrow_u q' \}$$
$$\text{Pred}_t(X,Y) = \{ q \in Q \mid \exists\, t.\ q^t \in X \ \text{and}\ \forall s \leq t.\ q^s \in Y^C \}$$

$$\pi(X) = \text{Pred}_t[\, X \cup \text{cPred}(X)\ ,\ \text{uPred}(X^C)\,]$$



Theorem:
The set of winning states is obtained as the least fixpoint of the function: $X \mapsto \pi(X) \cup \text{Goal}$

Symbolic On-the-fly Algorithms for Timed Games [CDF+05, BCD+07]

- $S, S', \ldots$
  are *symbolic* states, i.e. sets of concrete states;
- G
  is the set of (concrete) goal states;
- $E = \{S \xrightarrow{c} S', S \xrightarrow{u} S'\}$
  the (finite) set of symbolic transitions (controlla
- $Waiting \subseteq E$
  is the list of symbolic transitions waiting to be p
- $Passed$
  is the list of the passed symbolic states;
- $Win[S] \subseteq S$
  is the subset of S currently known to be winnin
- $Depend[S] \subseteq E$
  indicates the edges (predecessors) of S which mu
  information about S is obtained.

Initialization:

$Passed \leftarrow \{S_0\}$ **where** $S_0 = \{(\ell_0, \vec{0})\}^{\nearrow}$;
$Waiting \leftarrow \{(S_0, \alpha, S') \mid S' = \mathsf{Post}_\alpha(S_0)^{\nearrow}\}$;
$Win[S_0] \leftarrow S_0 \cap (\{\mathsf{Goal}\} \times \mathbb{R}^X_{\geq 0})$;
$Depend[S_0] \leftarrow \emptyset$;

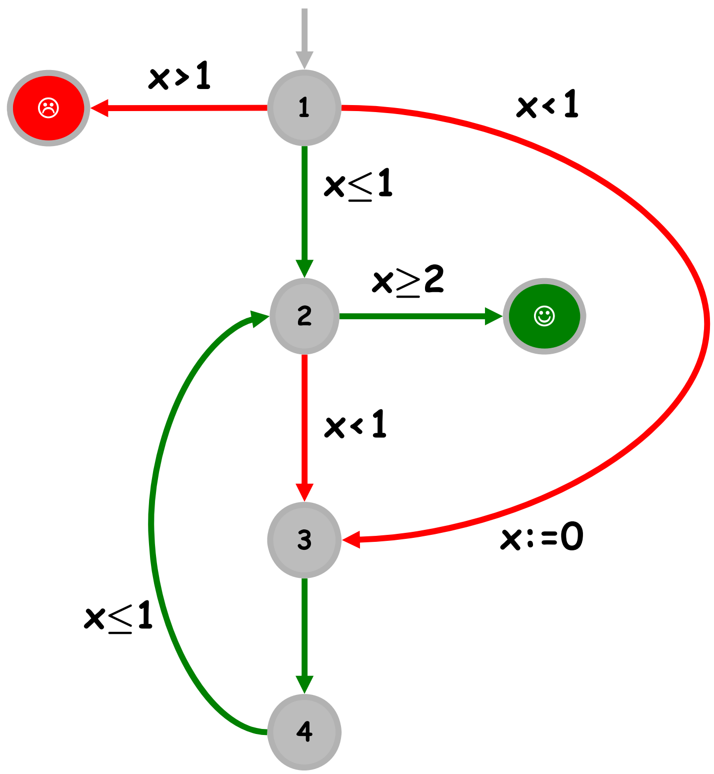
Main:

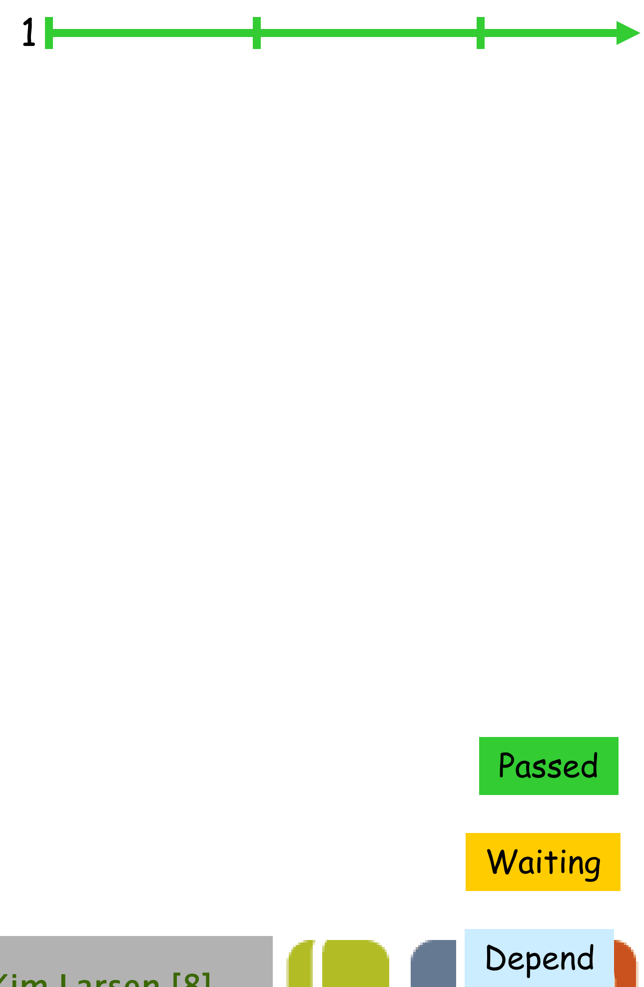
while $((Waiting \neq \emptyset) \wedge (s_0 \notin Win[S_0]))$ **do**
  $e = (S, \alpha, S') \leftarrow pop(Waiting)$;
  if $S' \notin Passed$ **then**
    $Passed \leftarrow Passed \cup \{S'\}$;
    $Depend[S'] \leftarrow \{(S, \alpha, S')\}$;
    $Win[S'] \leftarrow S' \cap (\{\mathsf{Goal}\} \times \mathbb{R}^X_{\geq 0})$;
    $Waiting \leftarrow Waiting \cup \{(S', \alpha, S'') \mid S'' = \mathsf{Post}_\alpha(S')^{\nearrow}\}$;
    if $Win[S'] \neq \emptyset$ **then** $Waiting \leftarrow Waiting \cup \{e\}$;
  e (* reevaluate *)[a]
    $Win^* \leftarrow \mathsf{Pred}_t(Win[S] \cup \bigcup_{S \xrightarrow{c} T} \mathsf{Pred}_c(Win[T]),$
      $\bigcup_{S \xrightarrow{u} T} \mathsf{Pred}_u(T \setminus Win[T])) \cap S$;
    if $(Win[S] \subsetneq Win^*)$ **then**
      $Waiting \leftarrow Waiting \cup Depend[S]$; $Win[S] \leftarrow Win^*$;
    $Depend[S'] \leftarrow Depend[S'] \cup \{e\}$;
  endif
endwhile

symbolic version of on-the-fly MC algorithm
for modal mu-calculus
Liu & Smolka 98

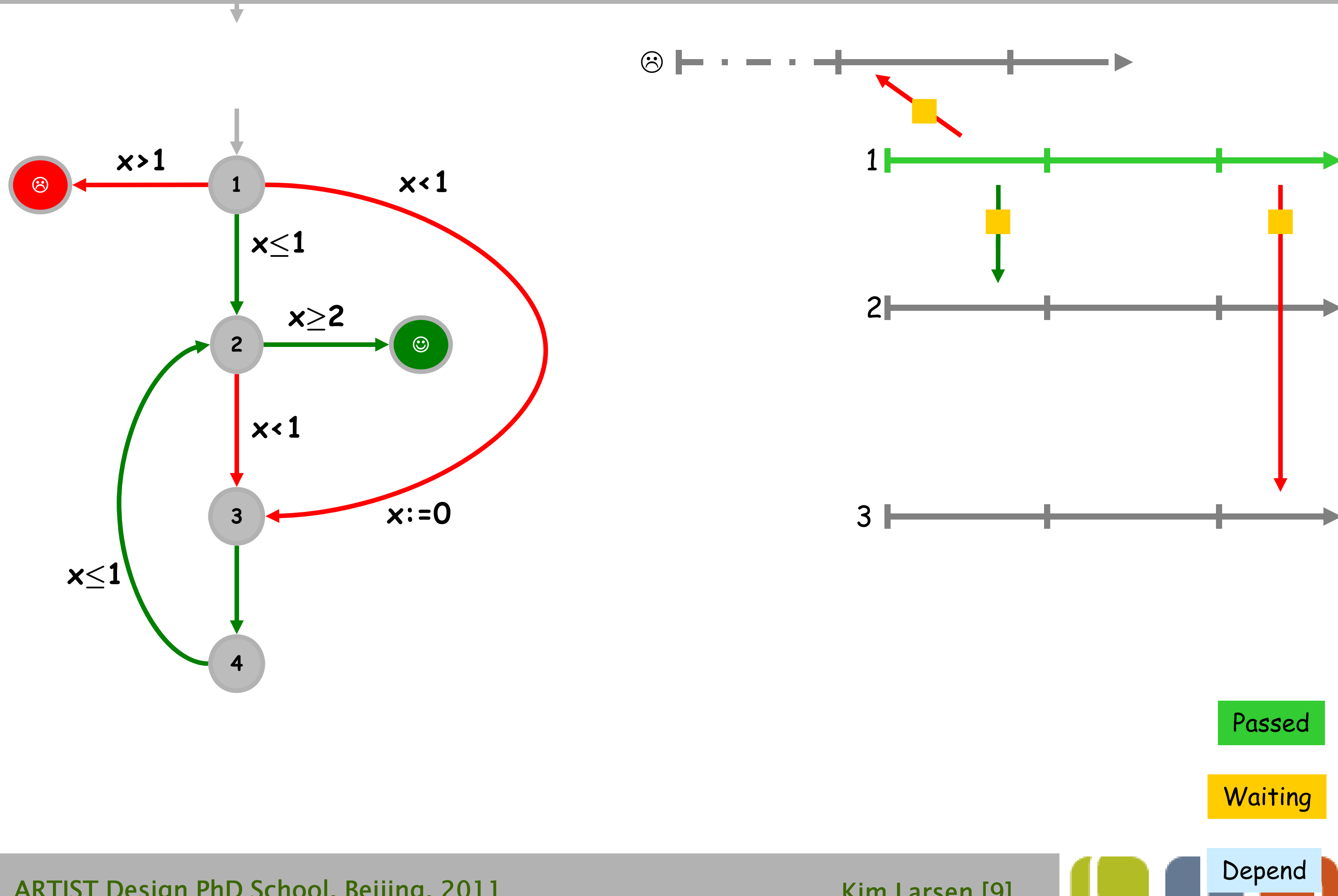
[CDF+05] Cassez, David, Fleury, Larsen, Lime. Efficient on-the-fly algorithms for the analysis of timed games *(CONCUR'05)*.
[BCD+07] Berhmann, Cougnard, David, Fleury, Larsen, Lime. Uppaal-Tiga: Time for playing games! *(CAV'07)*.

Symbolic On-the-fly Algorithms for Timed Games

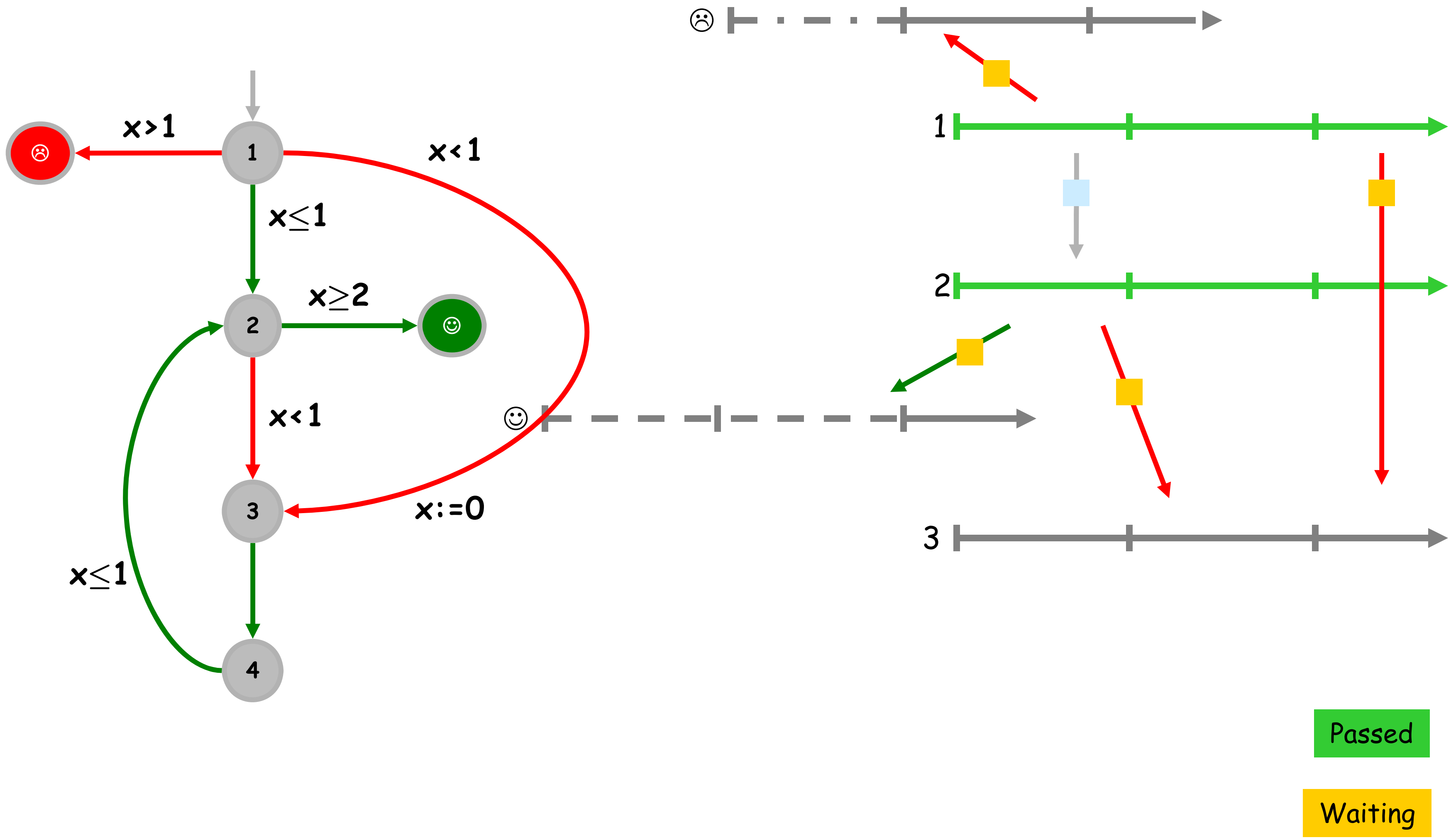




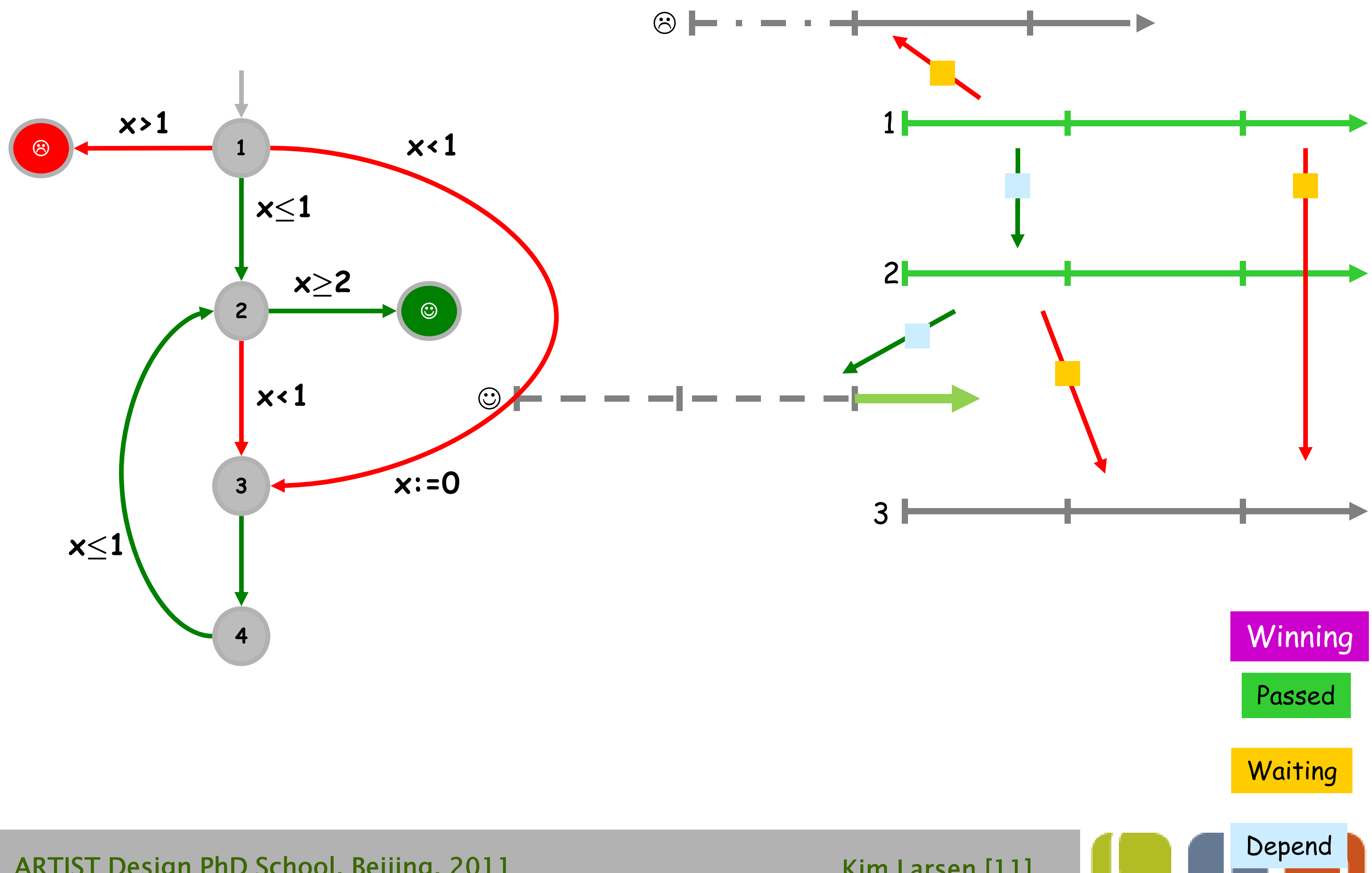
Symbolic On-the-fly Algorithms for Timed Games



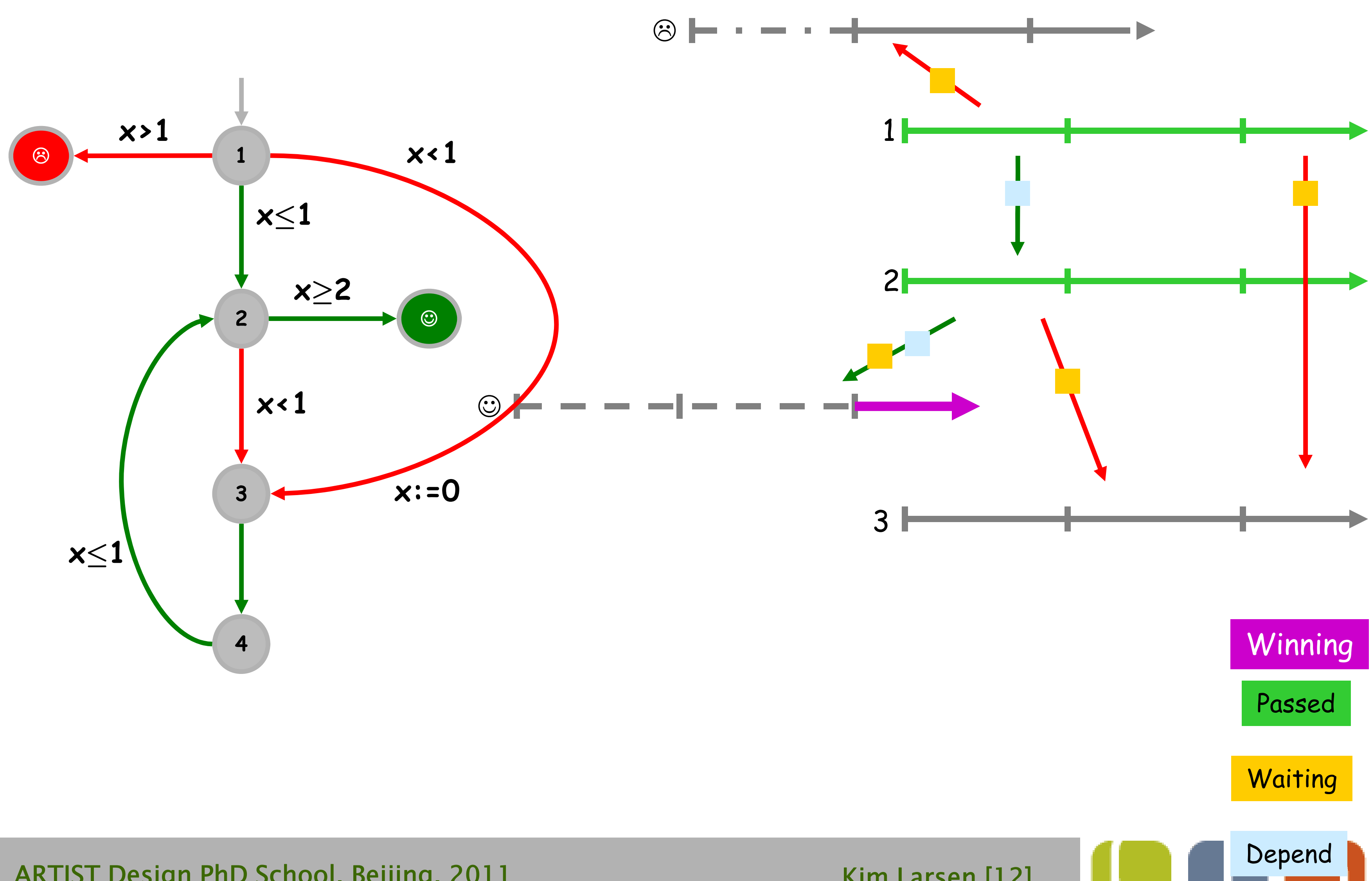
Symbolic On-the-fly Algorithms for Timed Games



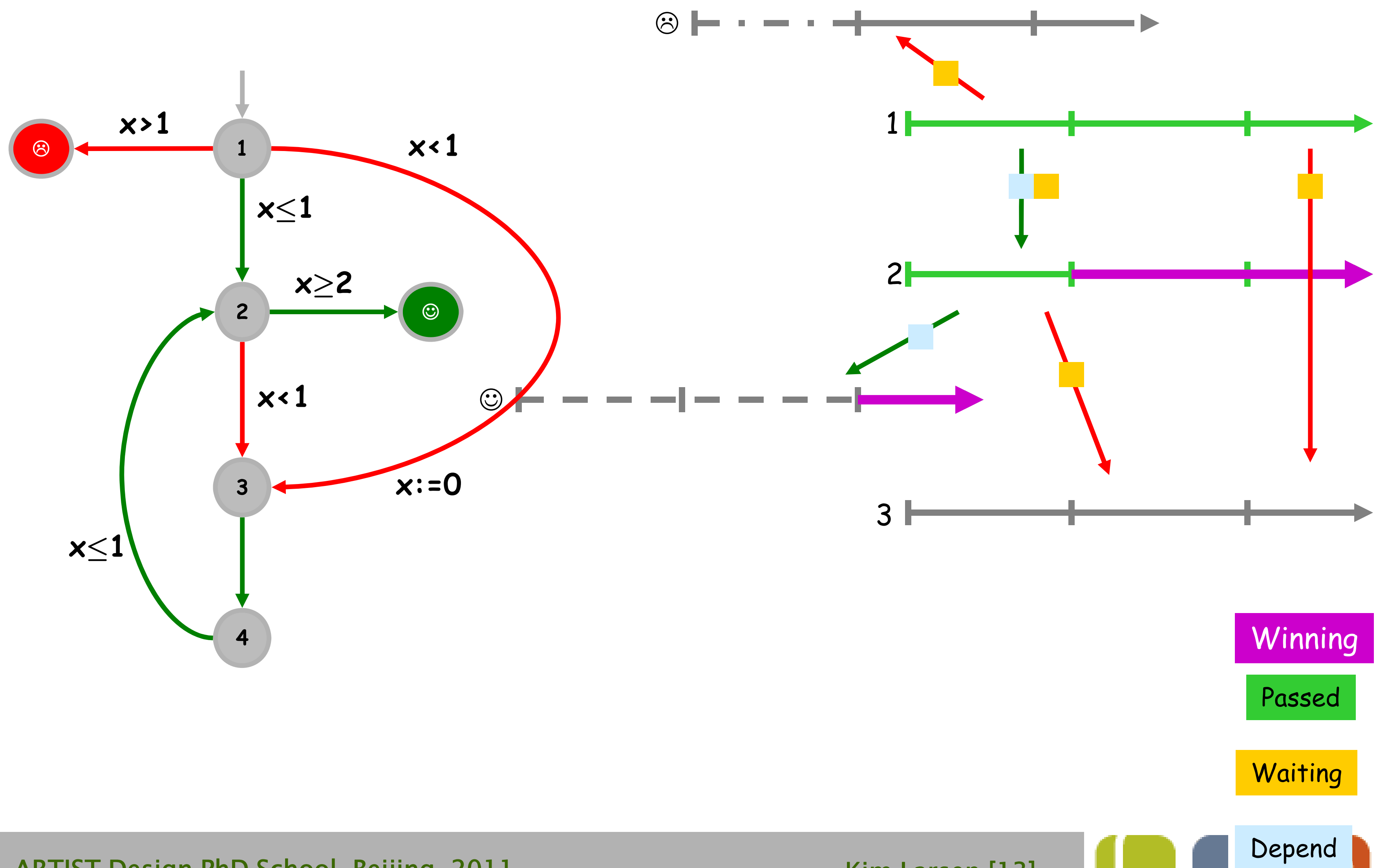
Symbolic On-the-fly Algorithms for Timed Games



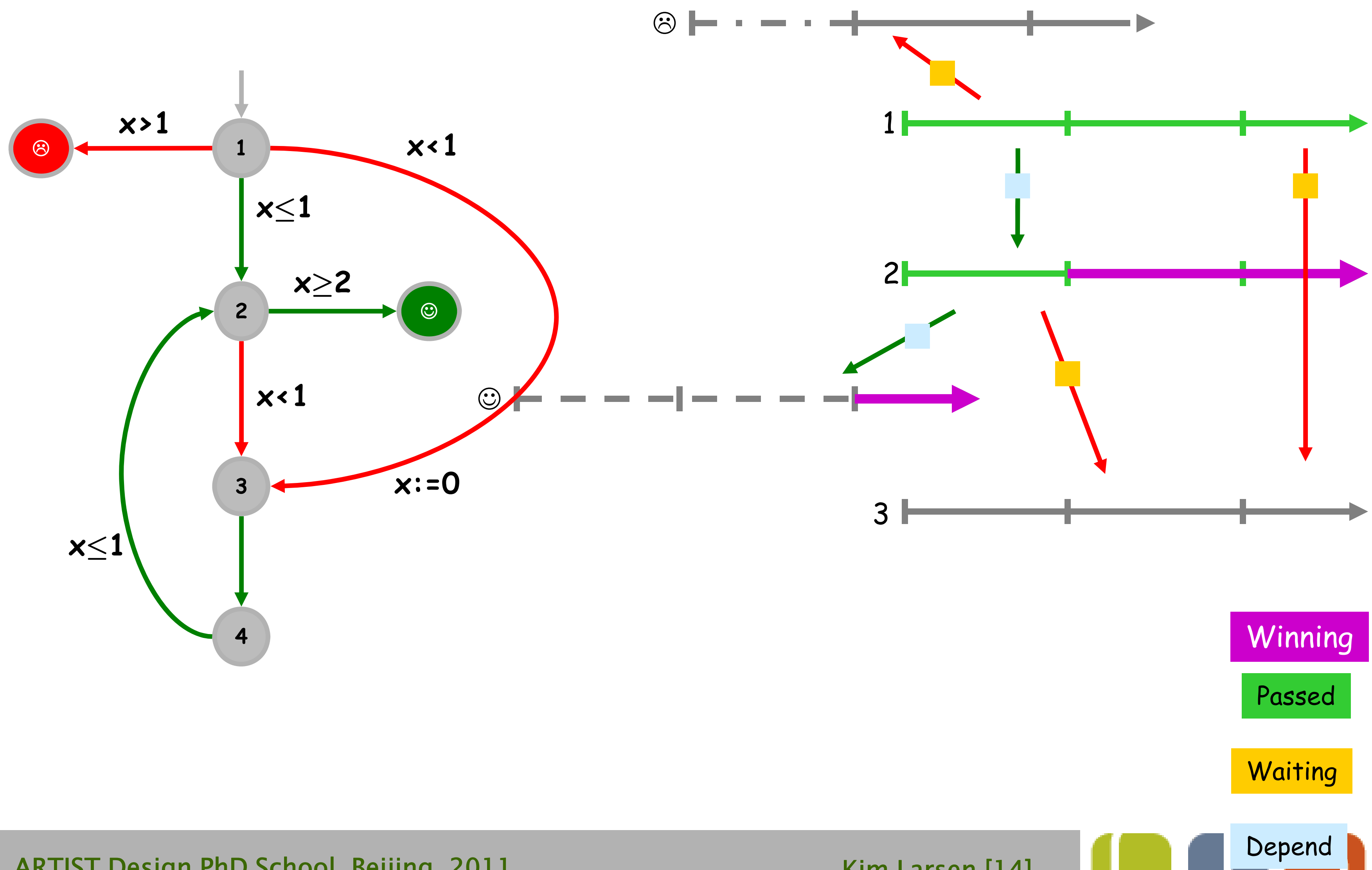
Symbolic On-the-fly Algorithms for Timed Games



Symbolic On-the-fly Algorithms for Timed Games



Symbolic On-the-fly Algorithms for Timed Games

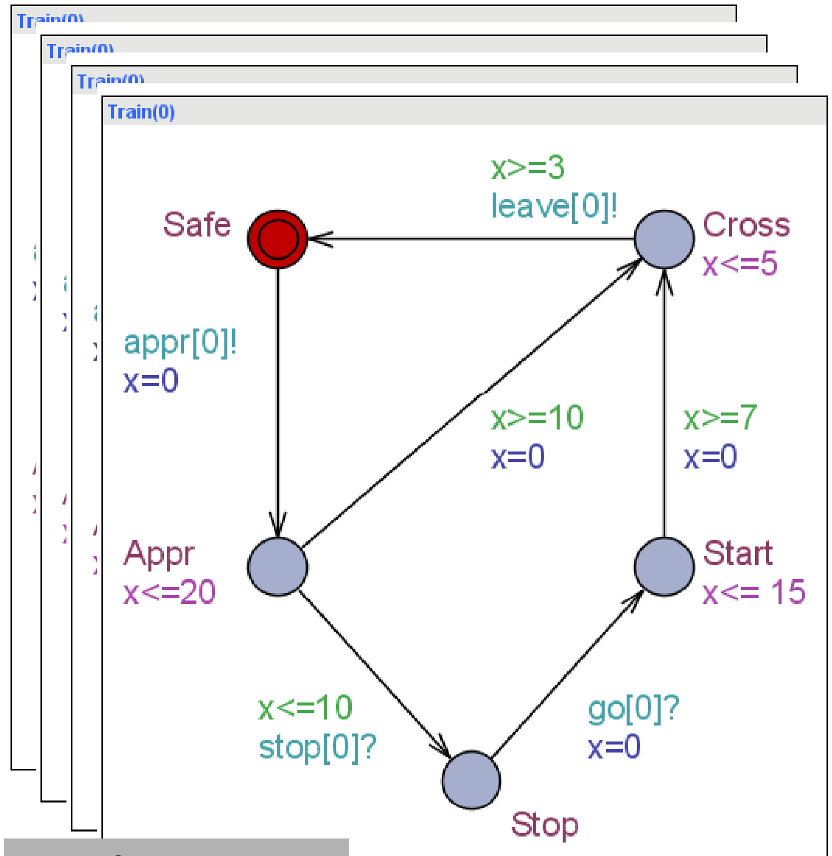


UPPAAL Tiga [CDF+05, BCD+07]

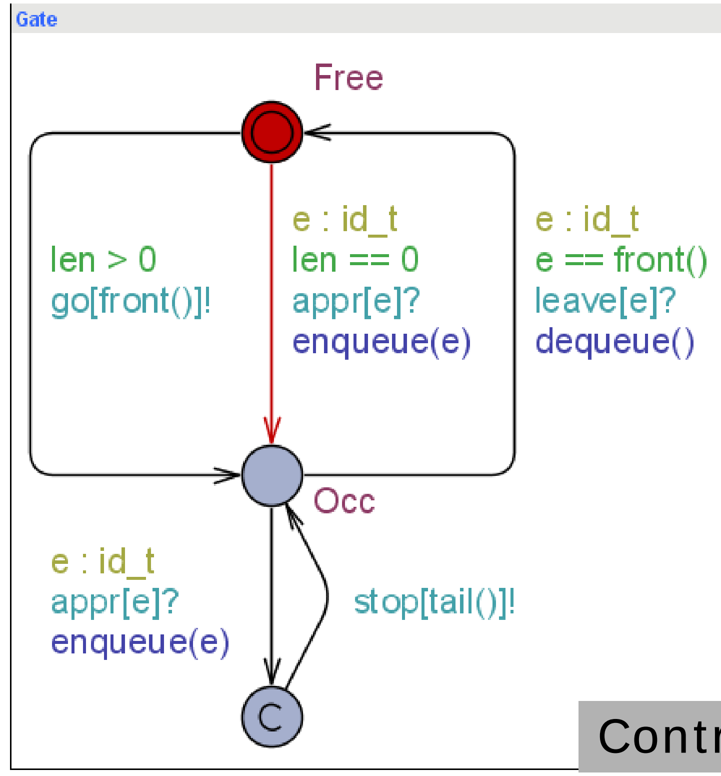
- Reachability properties:
  - control: A[p U q] *until*
  - control: A⟨⟩ q ⇔ control: A[true U q]
- Safety properties:
  - control: A[p W q] *weak until*
  - control: A[] p ⇔ control: A[p W false]
- Time-optimality :
  - control_t*(u,g): A[p U q]
    - u is an upper-bound to prune the search
    - g is the time to the goal from the current state

[CDF+05] Cassez, David, Fleury, Larsen, Lime. Efficient on-the-fly algorithms for the analysis of timed games *(CONCUR'05)*.
[BCD+07] Berhmann, Cougnard, David, Fleury, Larsen, Lime. Uppaal-Tiga: Time for playing games! *(CAV'07)*.

Model Checking (ex Train Gate)



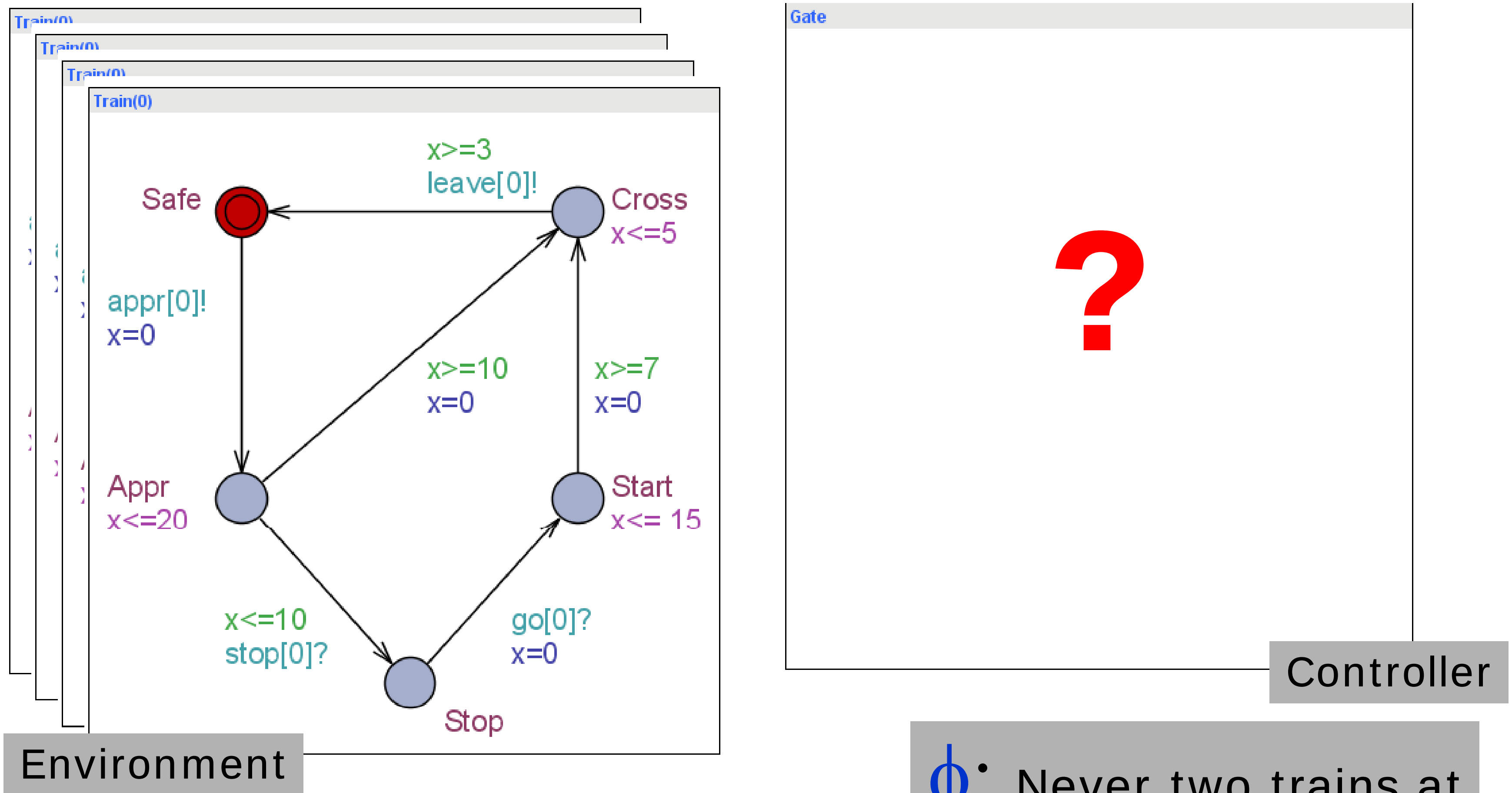
Environment



Controller

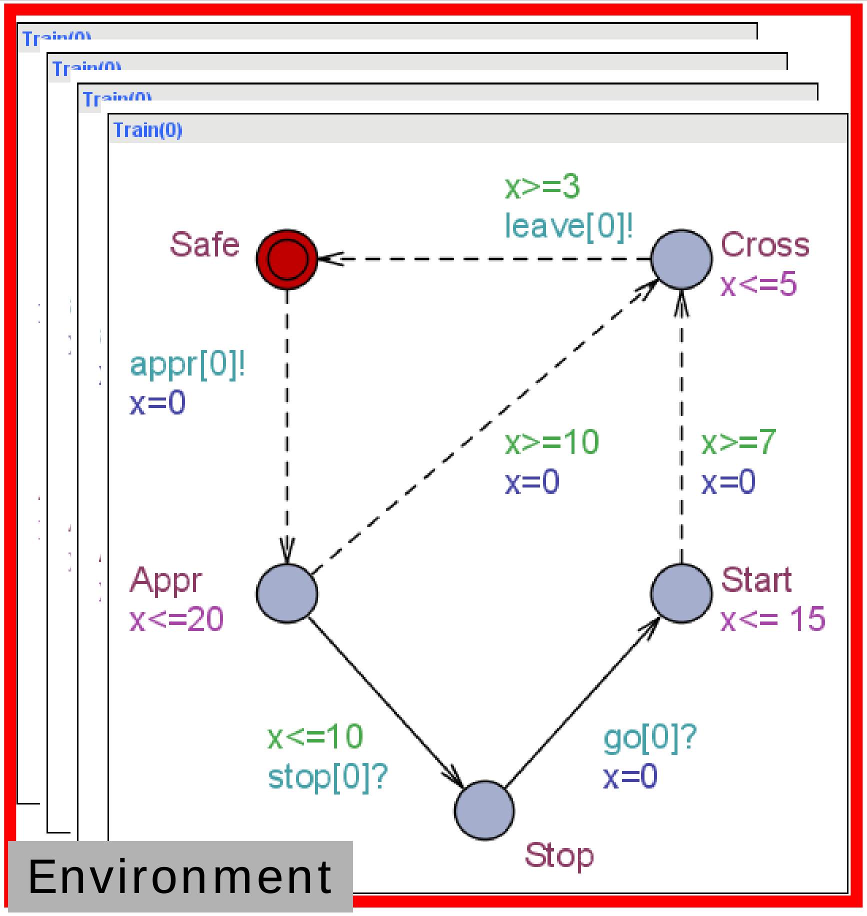
ϕ: Never two trains at the crossing at the same time

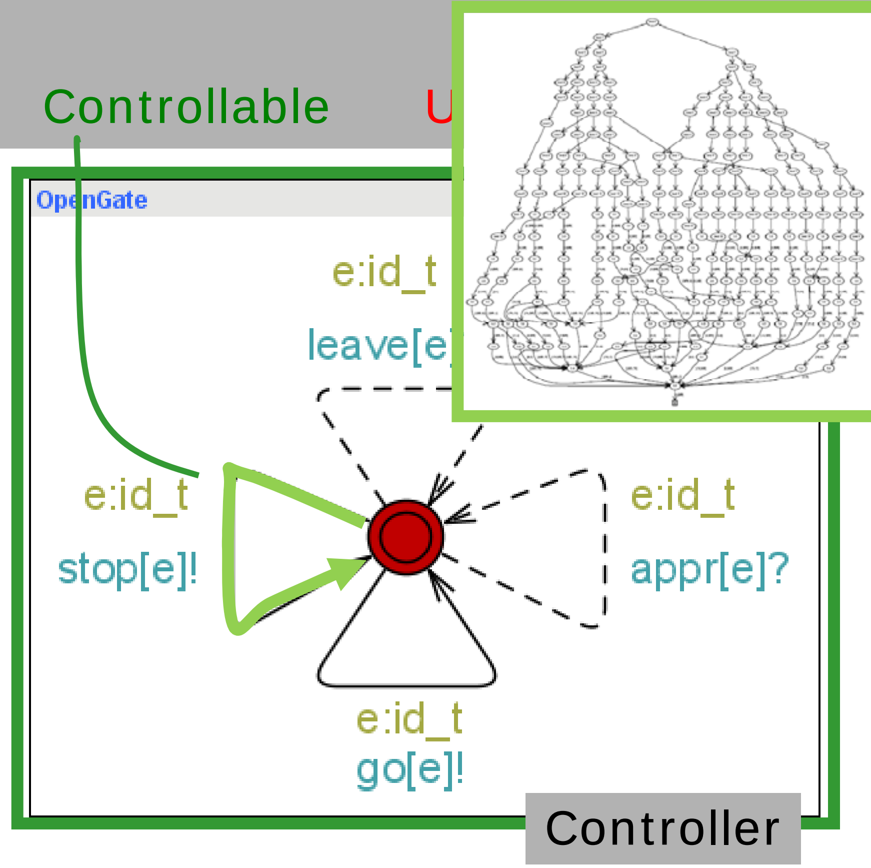
Synthesis (ex Train Gate)



ϕ: Never two trains at the crossing at the same time

Timed Games



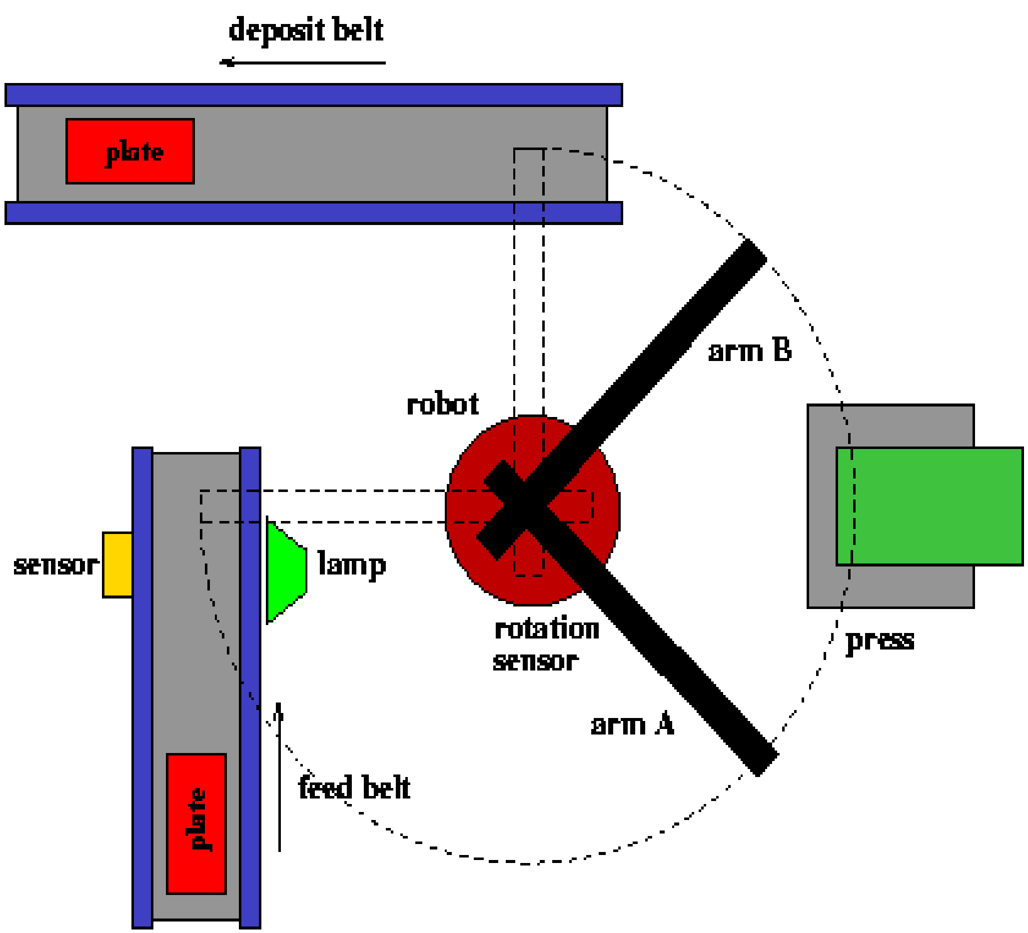


Find strategy for controllable actions st behaviour satisfies ϕ

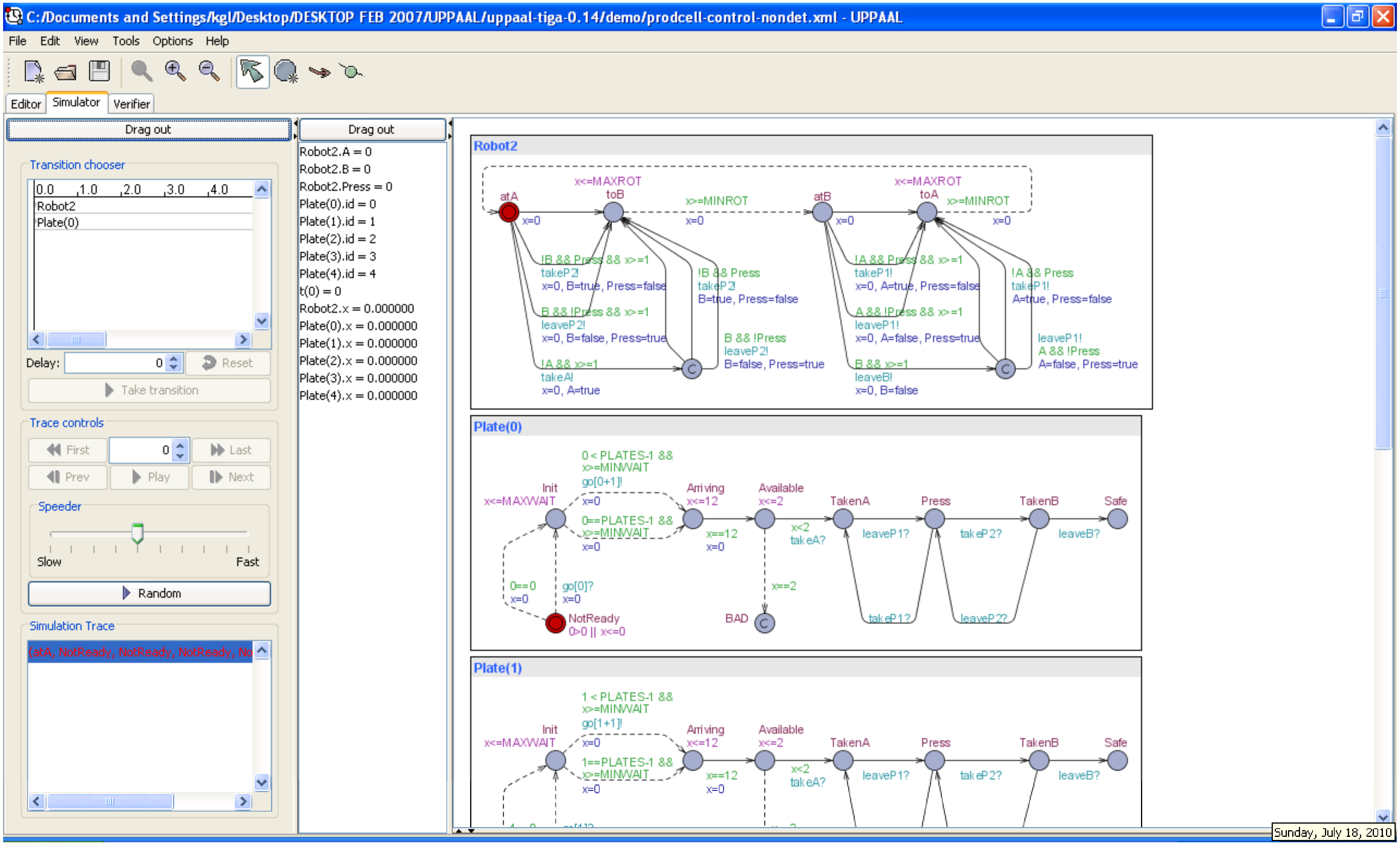
ϕ: Never two trains at the crossing at the same time

Production Cell Overview

- Realistic case-study described in several formalisms (1994 and later).
- **Objective**: stamp metal plates in press.
- feed belt, two-armed robot, press, and deposit belt.



Production Cell in UPPAAL Tiga



Experimental Results

[CDF+05]

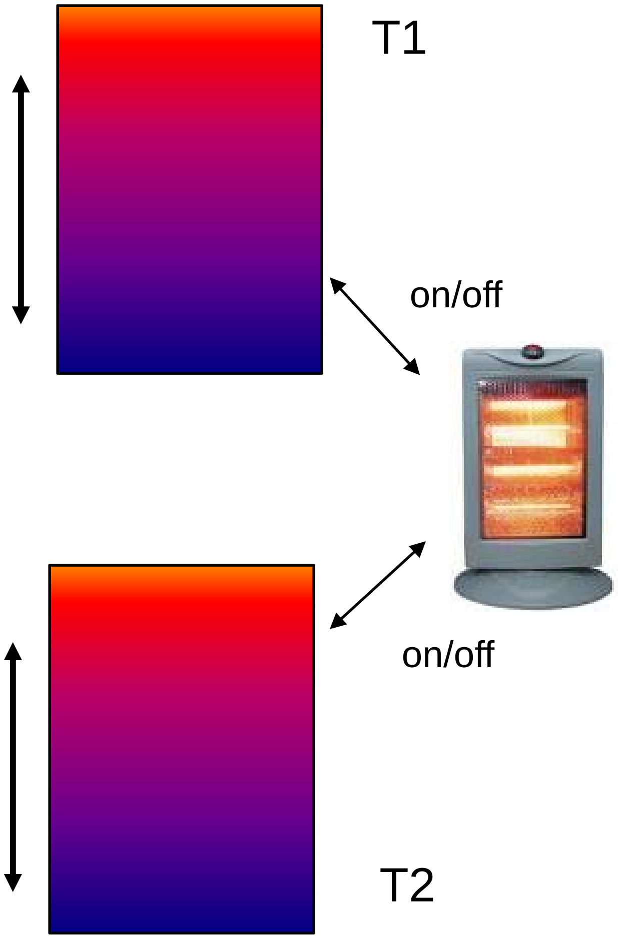
Plates		Basic		Basic +inc		Basic +inc +pruning		Basic+lose +inc +pruning		Basic+lose +inc +topt	
		time	mem	time	mem	time	mem	time	mem	time	mem
2	win	0.0s	1M	0.0s	1M	0.0s	1M	0.0s	1M	0.04s	1M
	lose	0.0s	1M	0.0s	1M	0.0s	1M	0.0s	1M	n/a	n/a
3	win	0.5s	19M	0.0s	1M	0.0s	1M	0.1s	1M	0.27s	4M
	lose	1.1s	45M	0.1s	1M	0.0s	1M	0.2s	3M	n/a	n/a
4	win	33.9s	1395M	0.2s	8M	0.1s	6M	0.4s	5M	1.88s	13M
	lose	-	-	0.5s	11M	0.4s	10M	0.9s	9M	n/a	n/a
5	win	-	-	3.0s	31M	1.5s	22M	2.0s	16M	13.35s	59M
	lose	-	-	11.1s	61M	5.9s	46M	7.0s	41M	n/a	n/a
6	win	-	-	89.1s	179M	38.9s	121M	12.0s	63M	220.3s	369M
	lose	-	-	699s	480M	317s	346M	135.1s	273M	n/a	n/a
7	win	-	-	3256s	1183M	1181s	786M	124s	319M	6188s	2457M
	lose	-	-	-	-	16791s	2981M	4075s	2090M	n/a	n/a

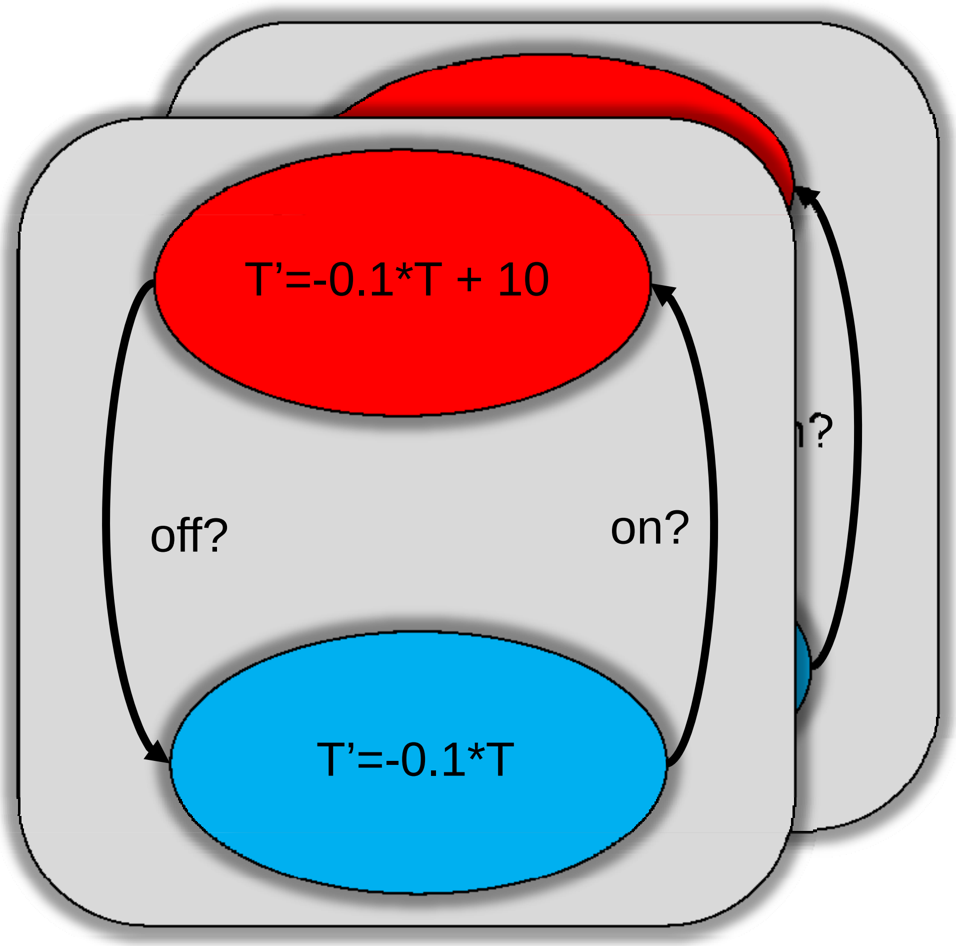
Model	c3		c6		c12		u3		u6		u12	
Old	0.1s	1M	12s	63M	-	-	0.2s	3M	235s	273M	-	-
New	0.05s	3.5M	0.05s	3.5M	0.14s	55M	0.02s	3.5M	0.04s	3.5M	0.12s	55M

[BCD+07]

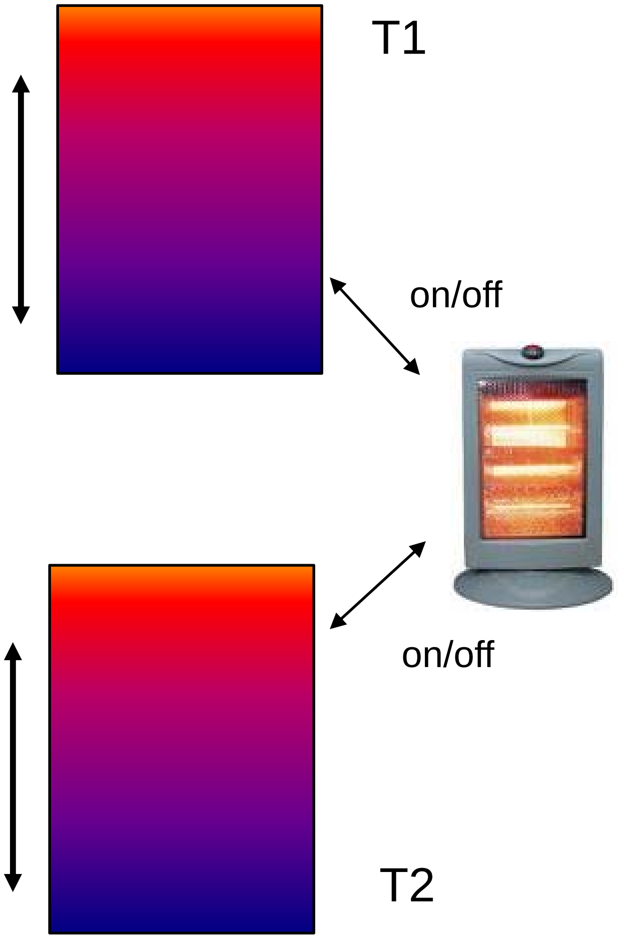
[CDF+05] Cassez, David, Fleury, Larsen, Lime. Efficient on-the-fly algorithms for the analysis of timed games *(CONCUR'05)*.
[BCD+07] Berhmann, Cougnard, David, Fleury, Larsen, Lime. Uppaal-Tiga: Time for playing games! *(CAV'07)*.

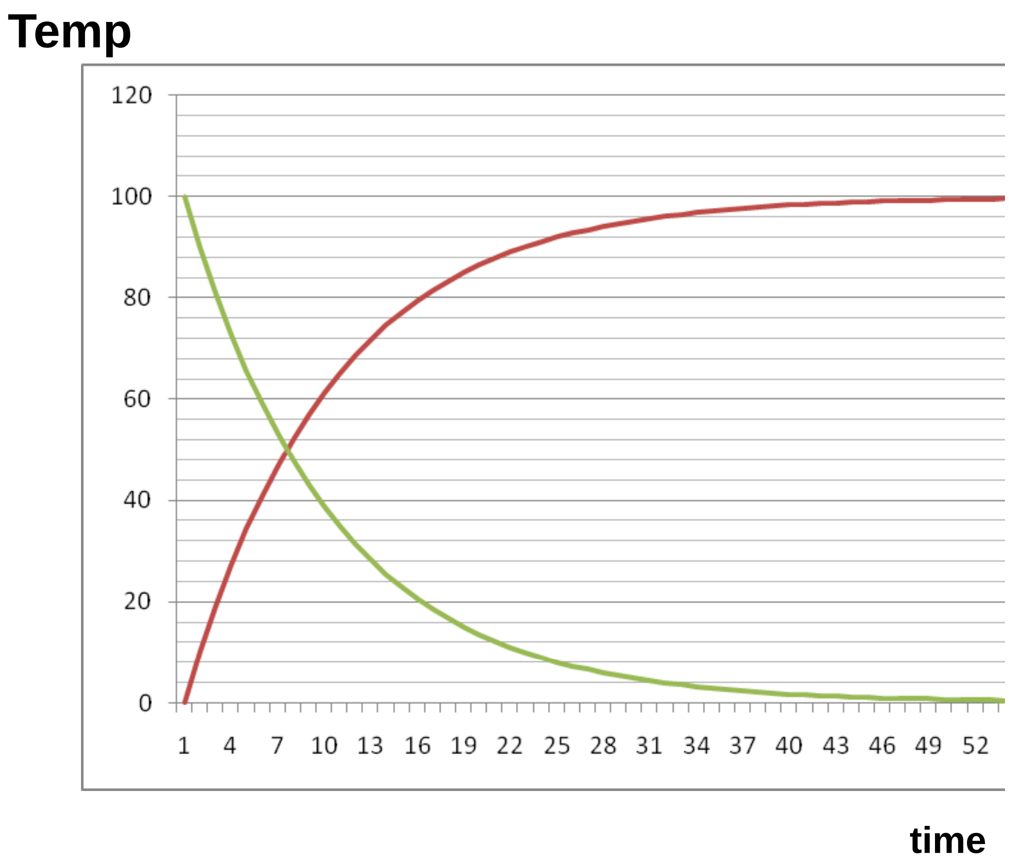
Two Tank Example



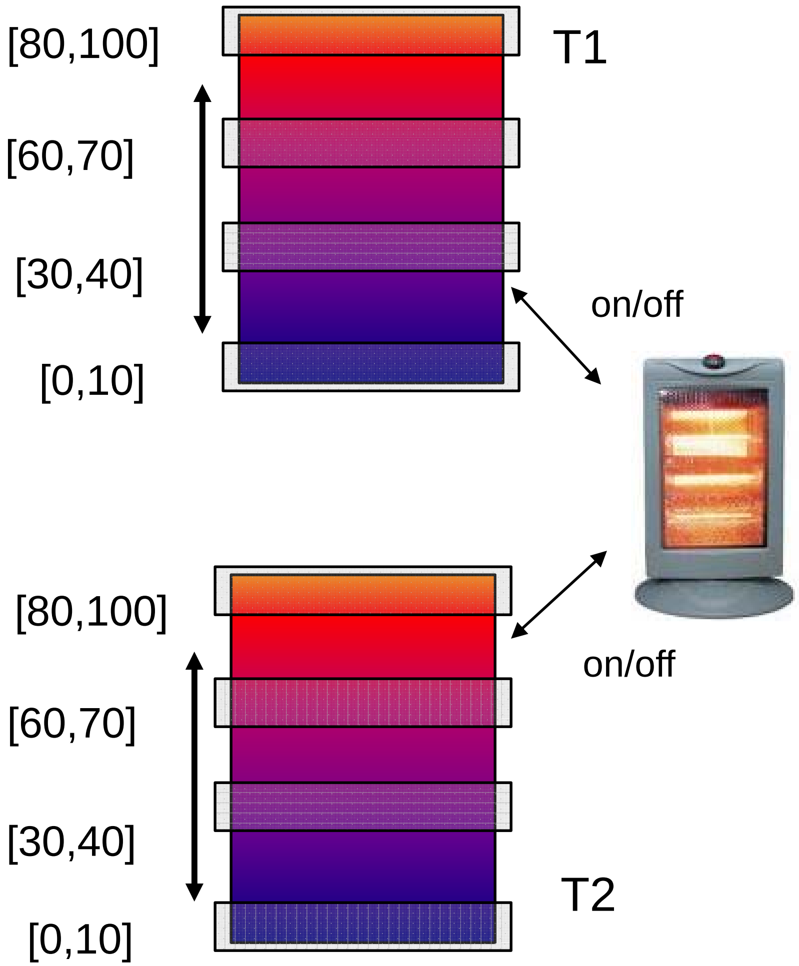


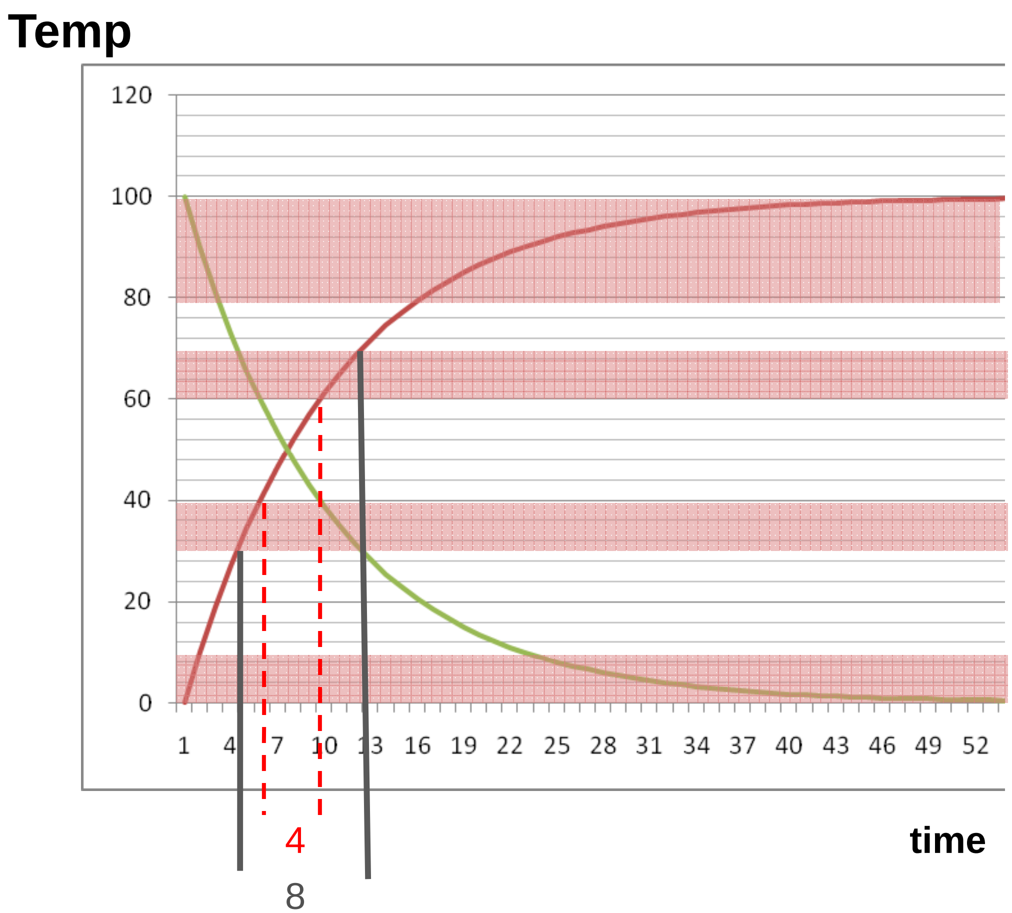
Two Tank Example



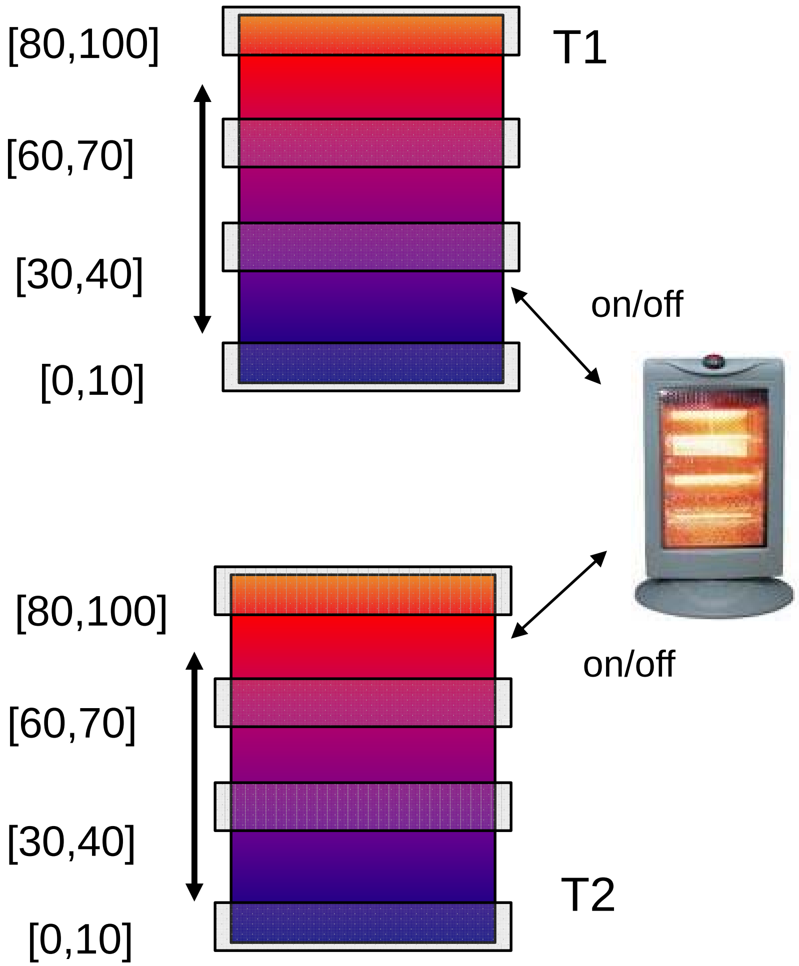


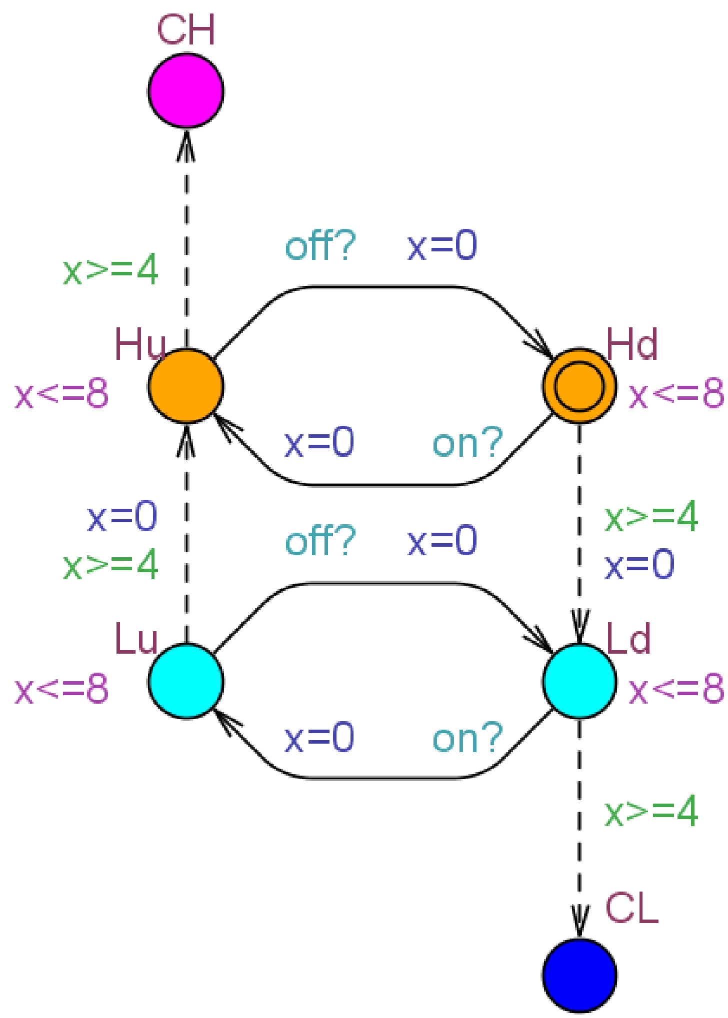
Two Tank Example





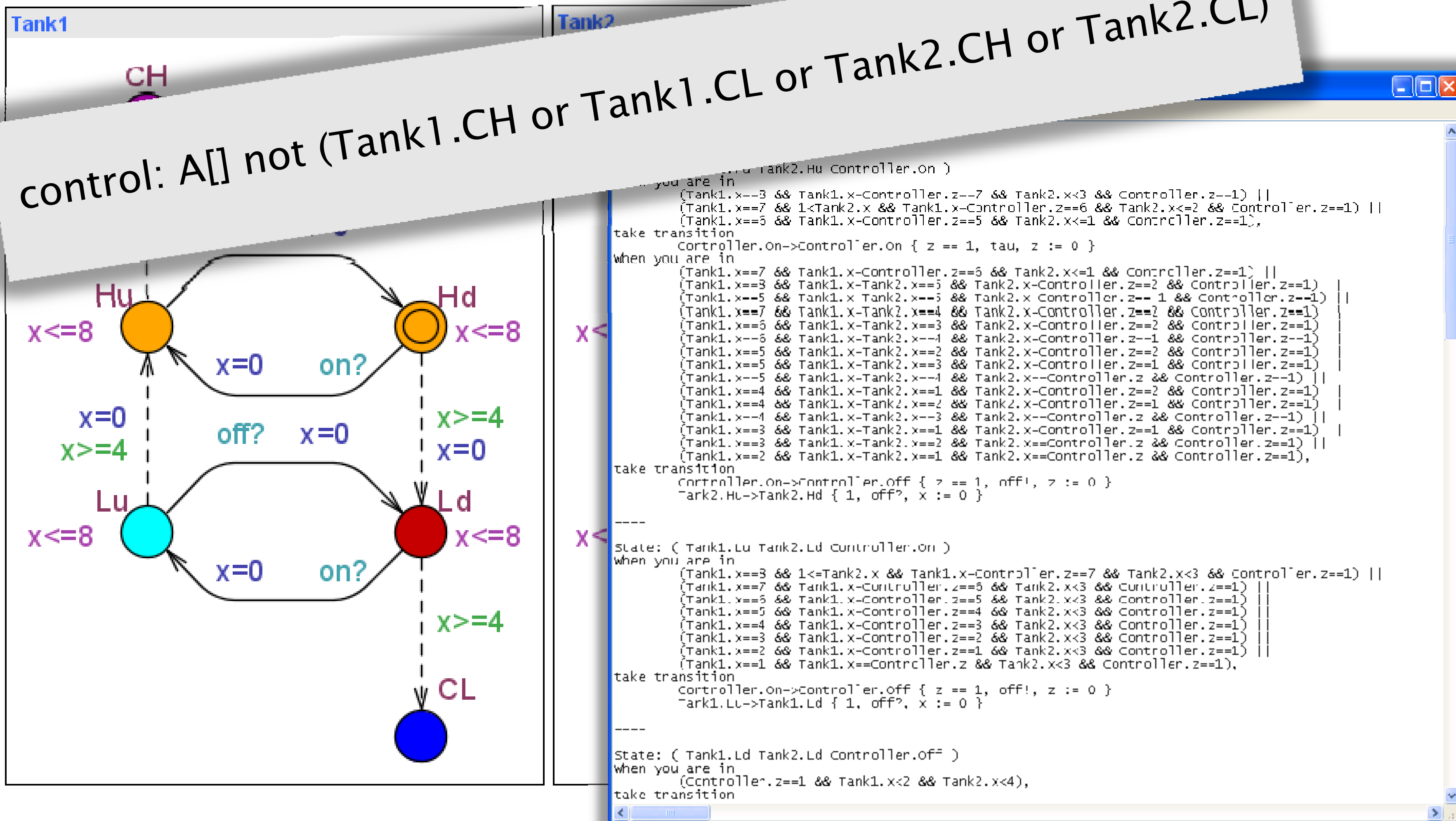
Two Tank Example





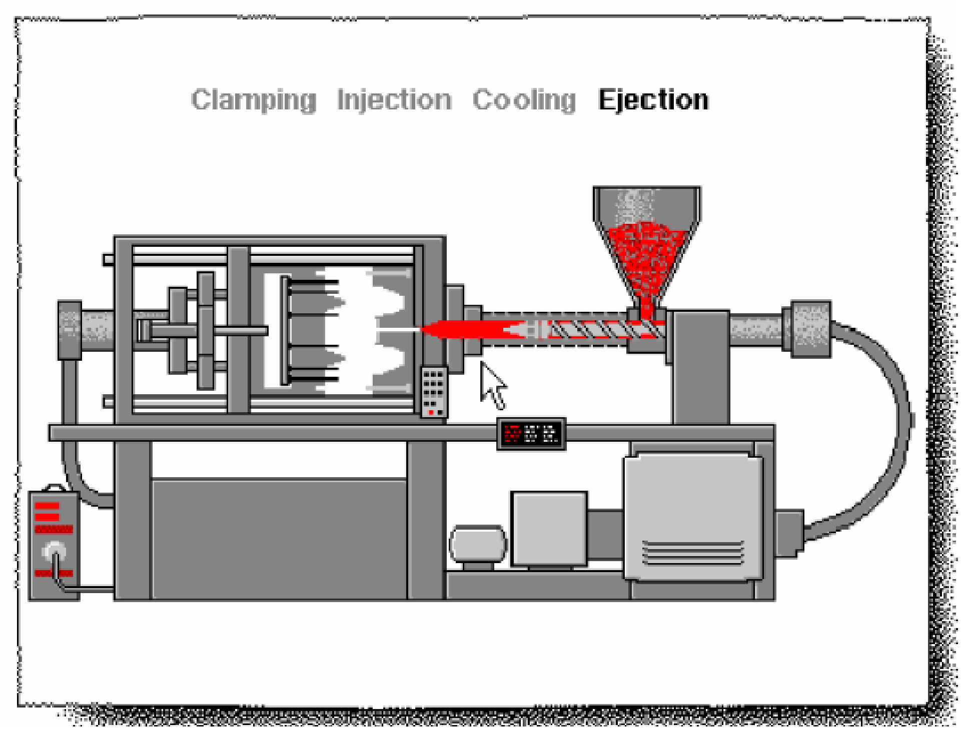
Two Tank Example

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



Plastic Injection Molding Machine

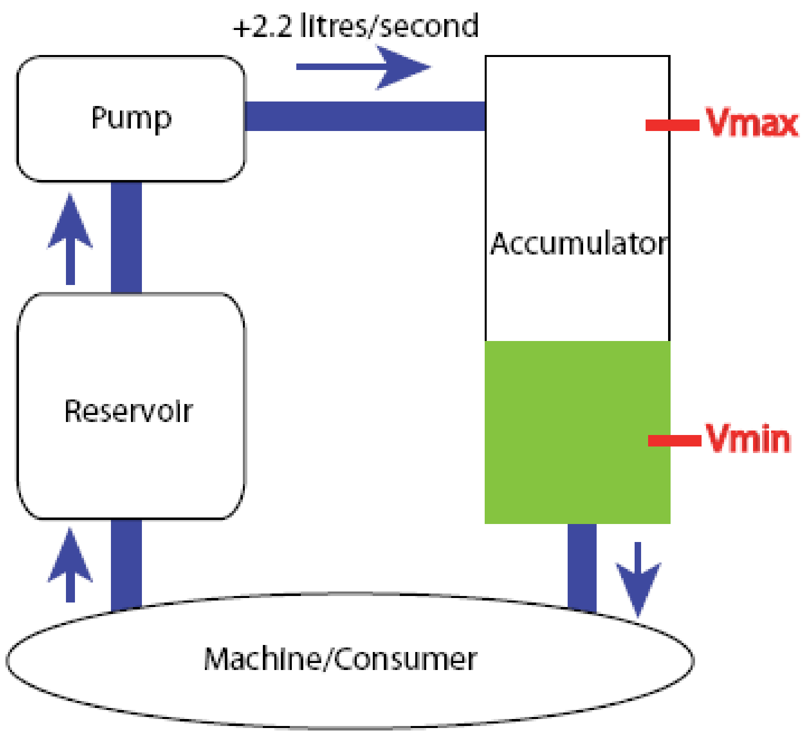
[CJL+09]



- Robust and optimal control
- Tool Chain
  - Synthesis: UPPAAL TIGA
  - Verification: PHAVer
  - Performance: SIMULINK
- 40% improvement of existing solutions..

[CJL+09] Cassez, Jessen, Larsen, Raskin, Reynier.Automatic Synthesis of Robust and Optimal Controllers – An Industrial Case Study *(HSCC'09)*.

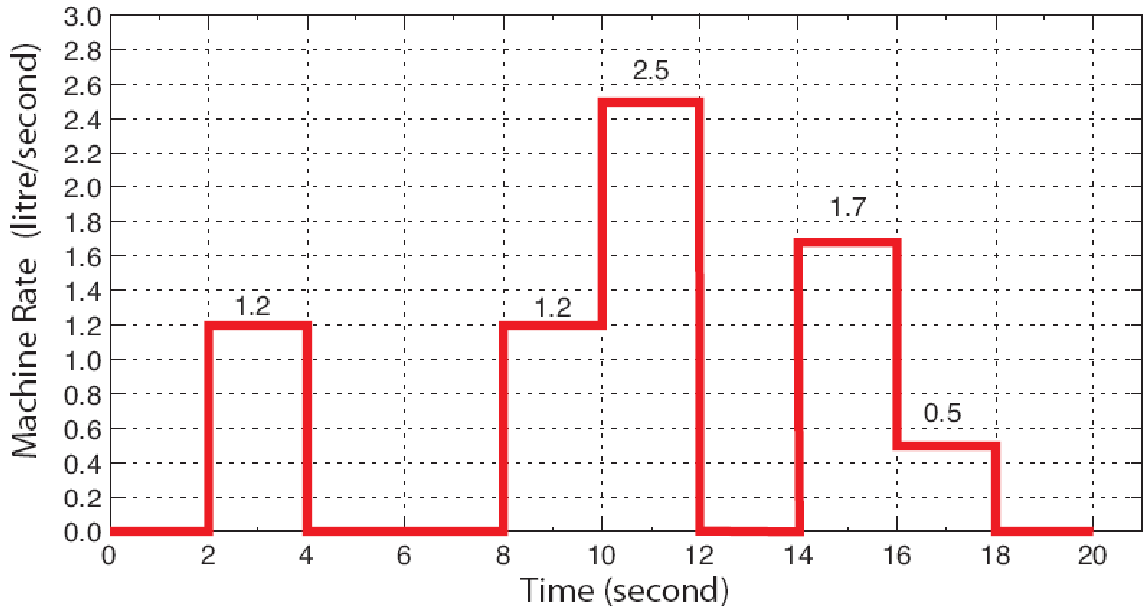
Oil Pump Control Problem



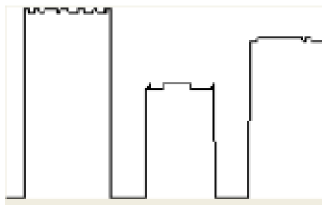
- R1: stay within safe interval [4.9,25.1]
- R2: minimize average/overall oil volume

$$\int_{t=0}^{t=T} v(t)dt \;/\; T$$

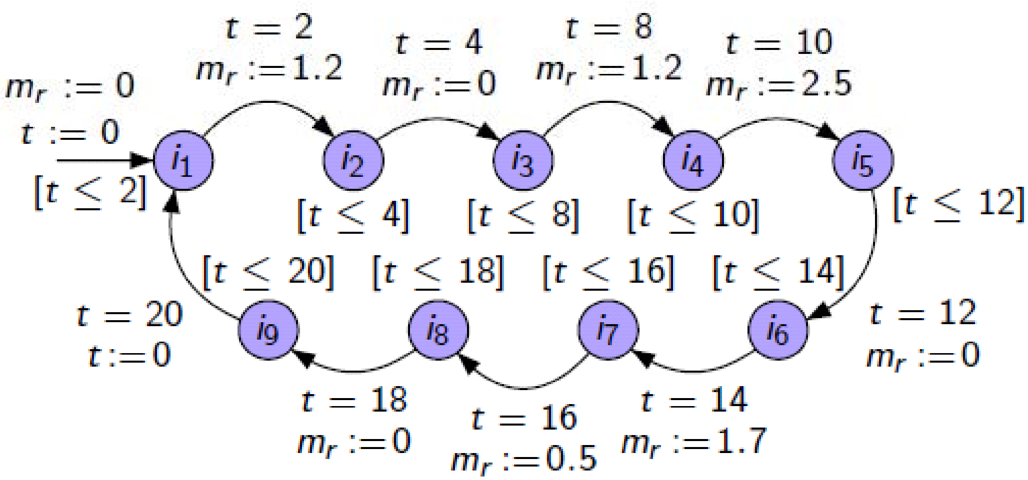
The Machine (consumption)



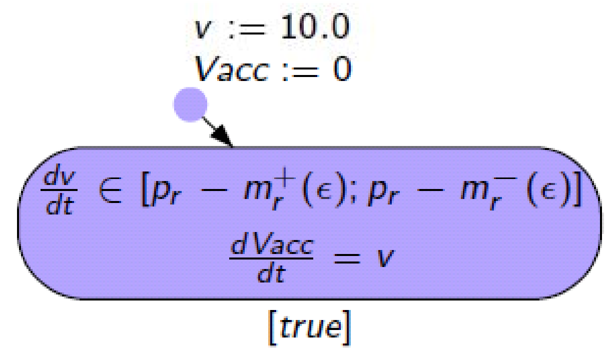
- Infinite cyclic demand to be satisfied by our control strategy.
- P: latency 2 s between state change of pump
- F: noise 0.1 l/s



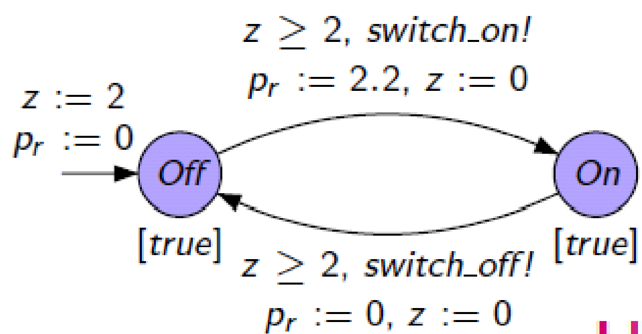
Hybrid Game Model



(a) The Machine



(b) The Accumulator



Undecidable problem!

(c) The Pump

Abstract Game Model

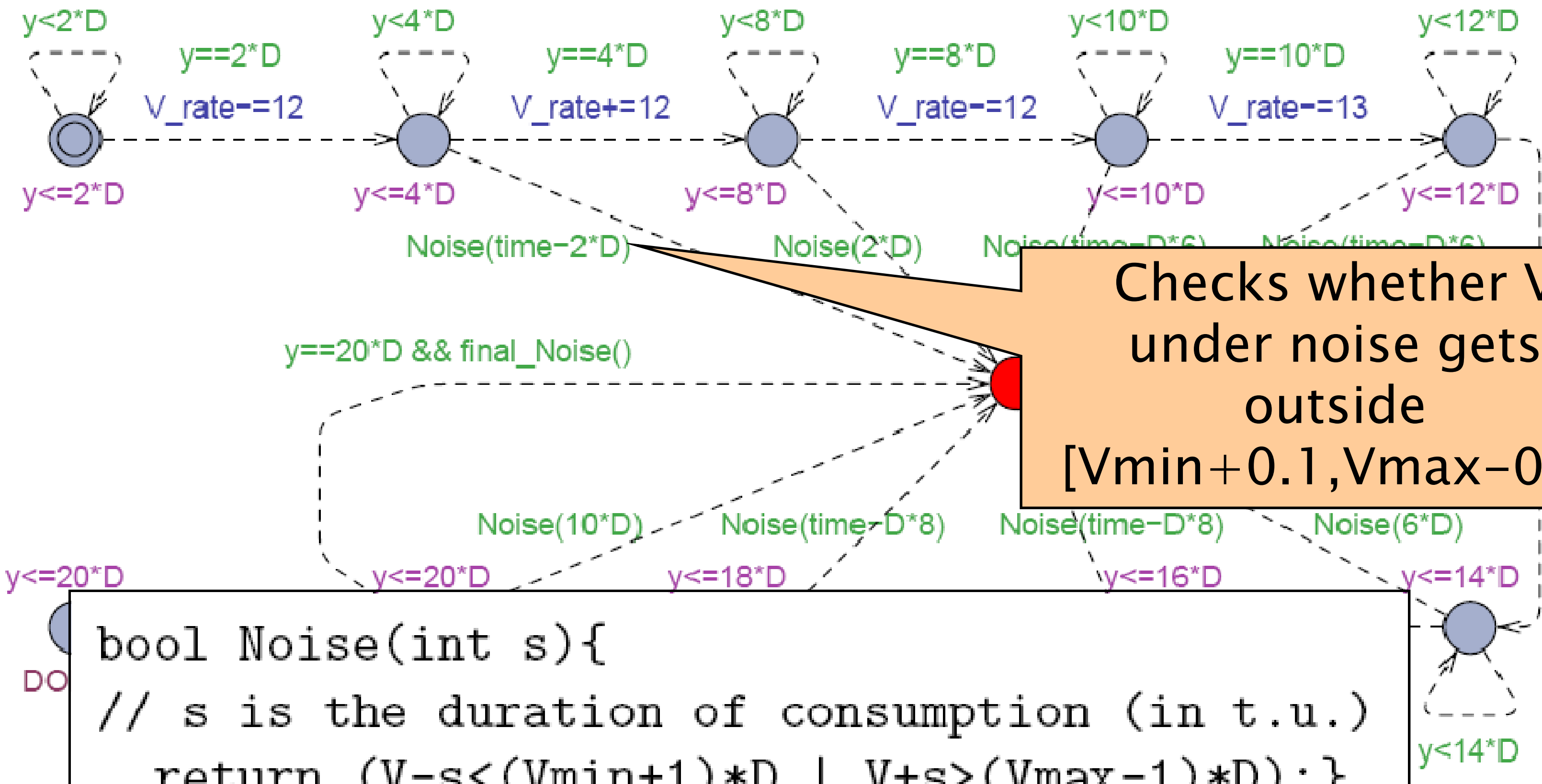
Quasiomodo

- UPPAAL Tiga
  offers games of perfect information

- Abstract game model such that states only contain information about:
  - Volume of oil at the beginning of cycle
  - The ideal volume as predicted by the consumption cycle
  - Current time within the cycle
  - State of the Pump (on/off)
  - Discrete model

D
V, V_rate
V_acc
time

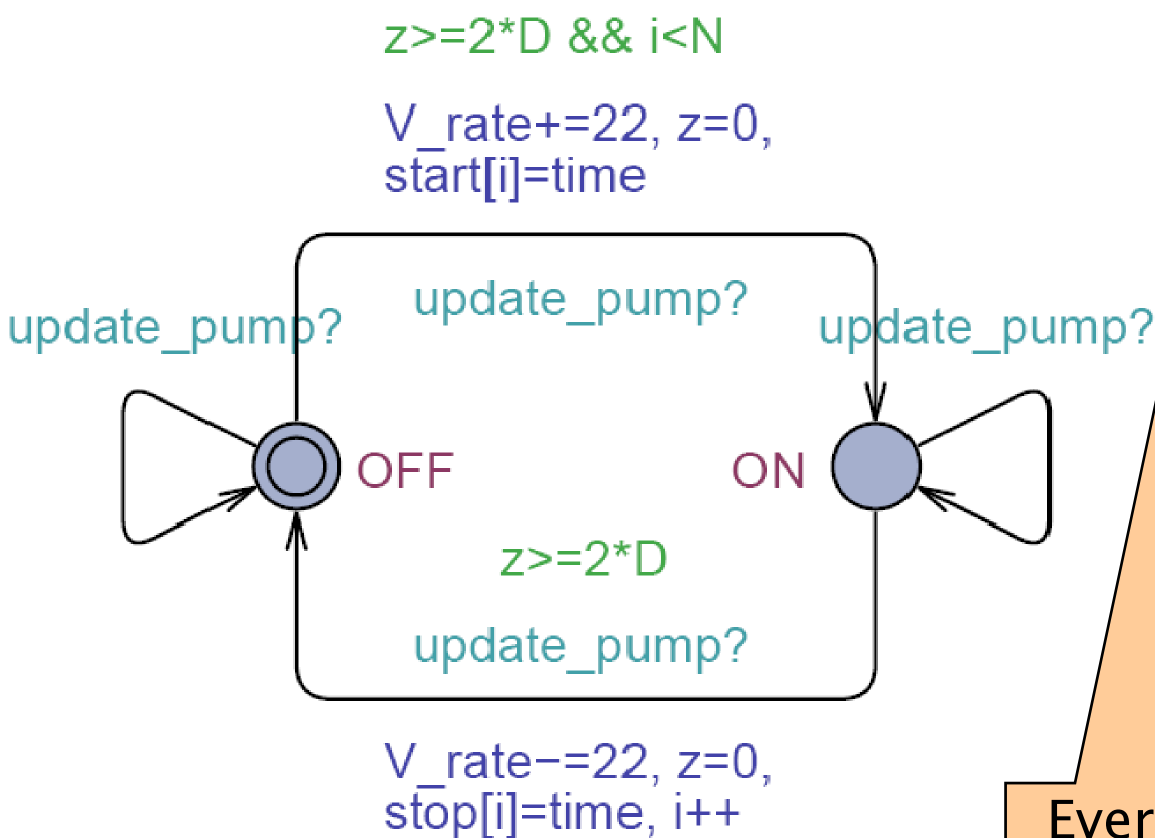
Machine (uncontrollable)



Checks whether V under noise gets outside [Vmin+0.1,Vmax-0.1]

```
bool Noise(int s){
// s is the duration of consumption (in t.u.)
  return (V-s<(Vmin+1)*D | V+s>(Vmax-1)*D);}
```

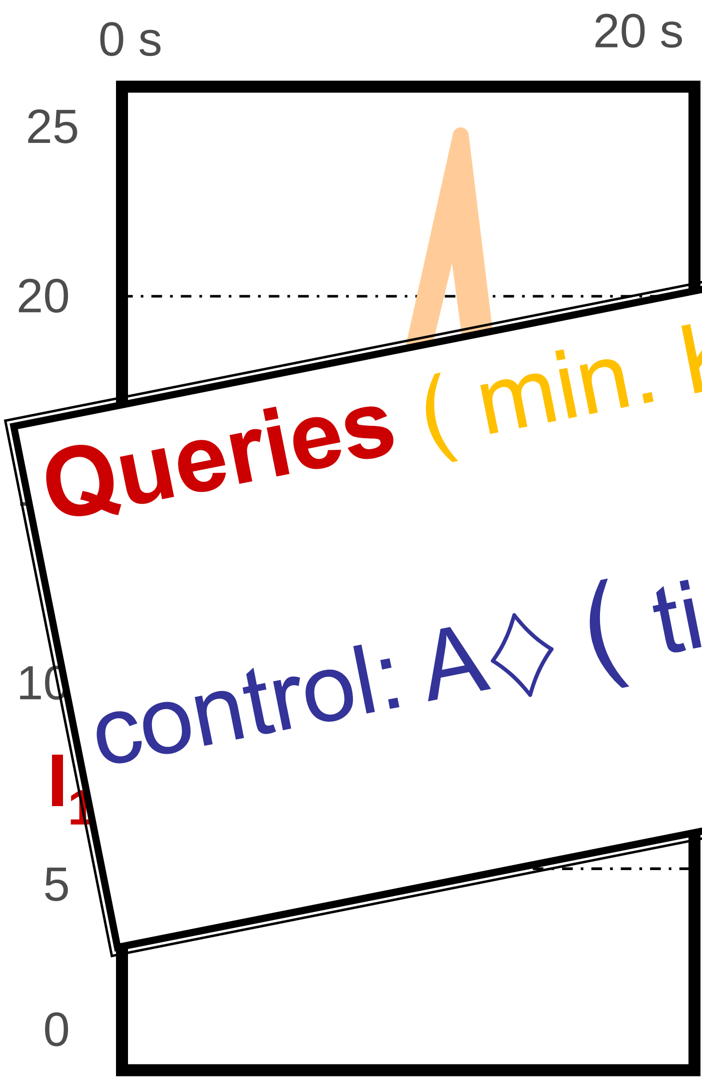
Pump (controllable)




```
void update_val(){
   int V_pred = V;
   time++;
   V+=V_rate;
   V_acc+=V+V_pred;
}
```

Every 1 (one) seconds

Global Approach



- Find some interval $I_1=[V_1,V_2] \subseteq [4.9,25.1]$ s.t.

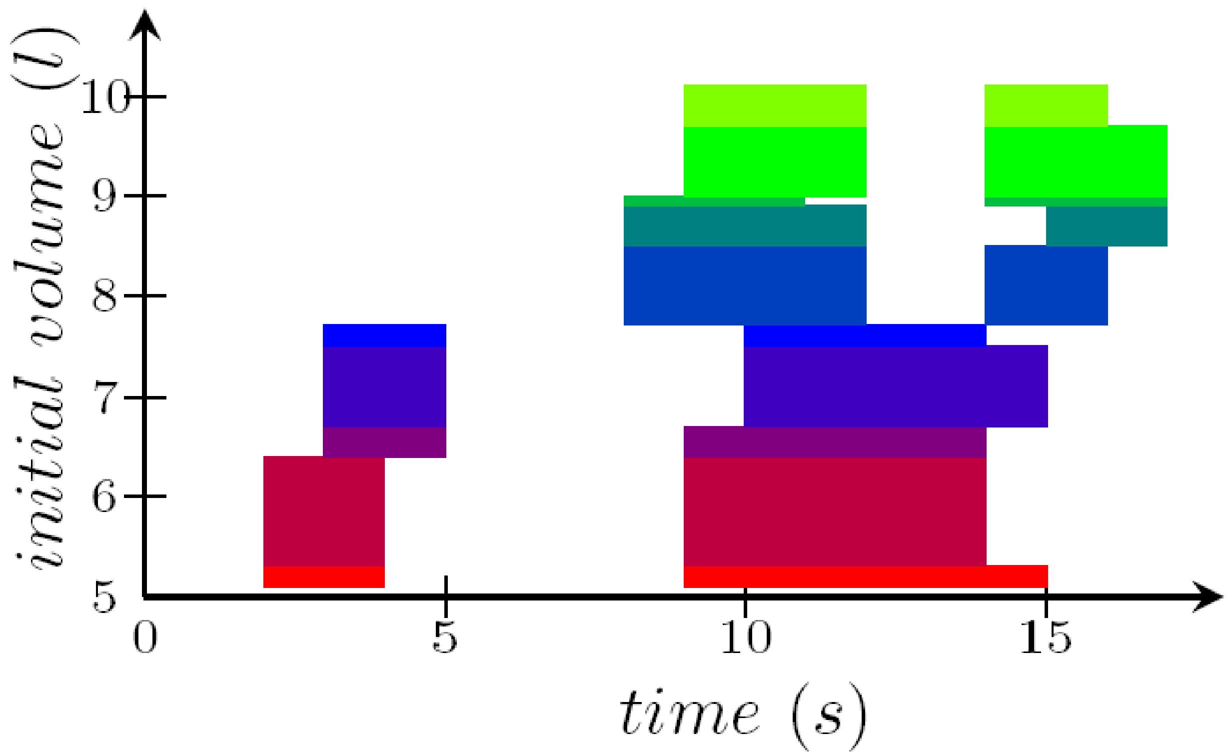
  …m, V_1−m]

- I_1 is optimal among all m−stable intervals.

Queries (min. K)

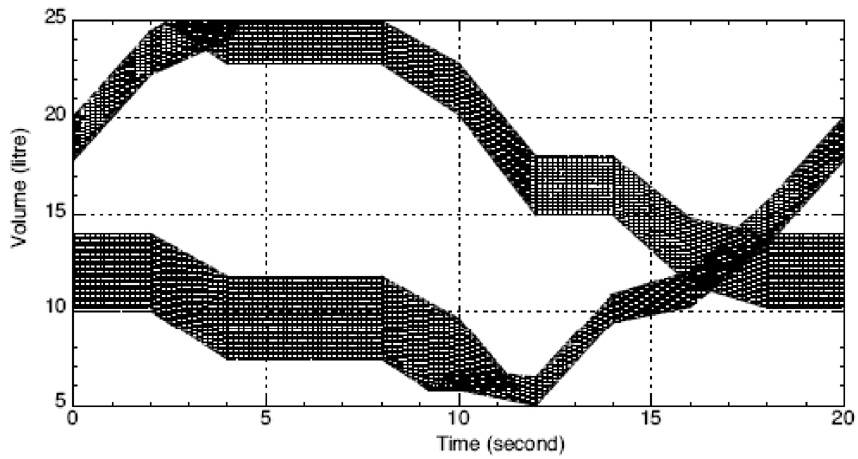
control: $A\Diamond$ (time=20 $\wedge$ not BAD $\wedge$ V $\in I_2$ $\wedge$ V_acc $\leq$ K)

Synthesized Strategy



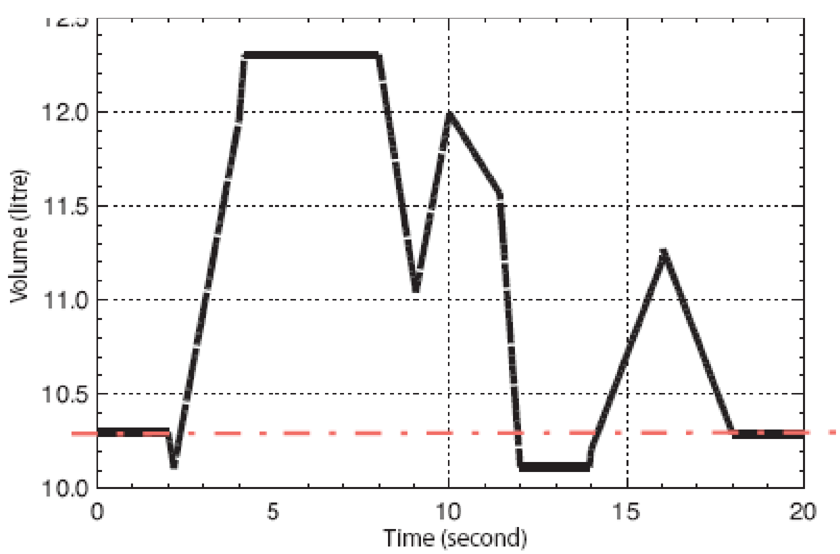
D=1, m=0.4: Optimal stable interval I_1=[5.1,10]

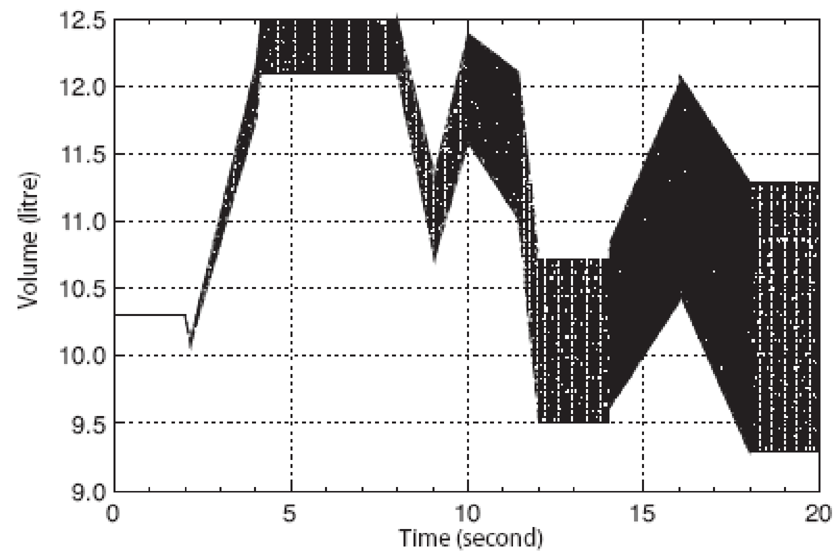
Verification Using PHAVER



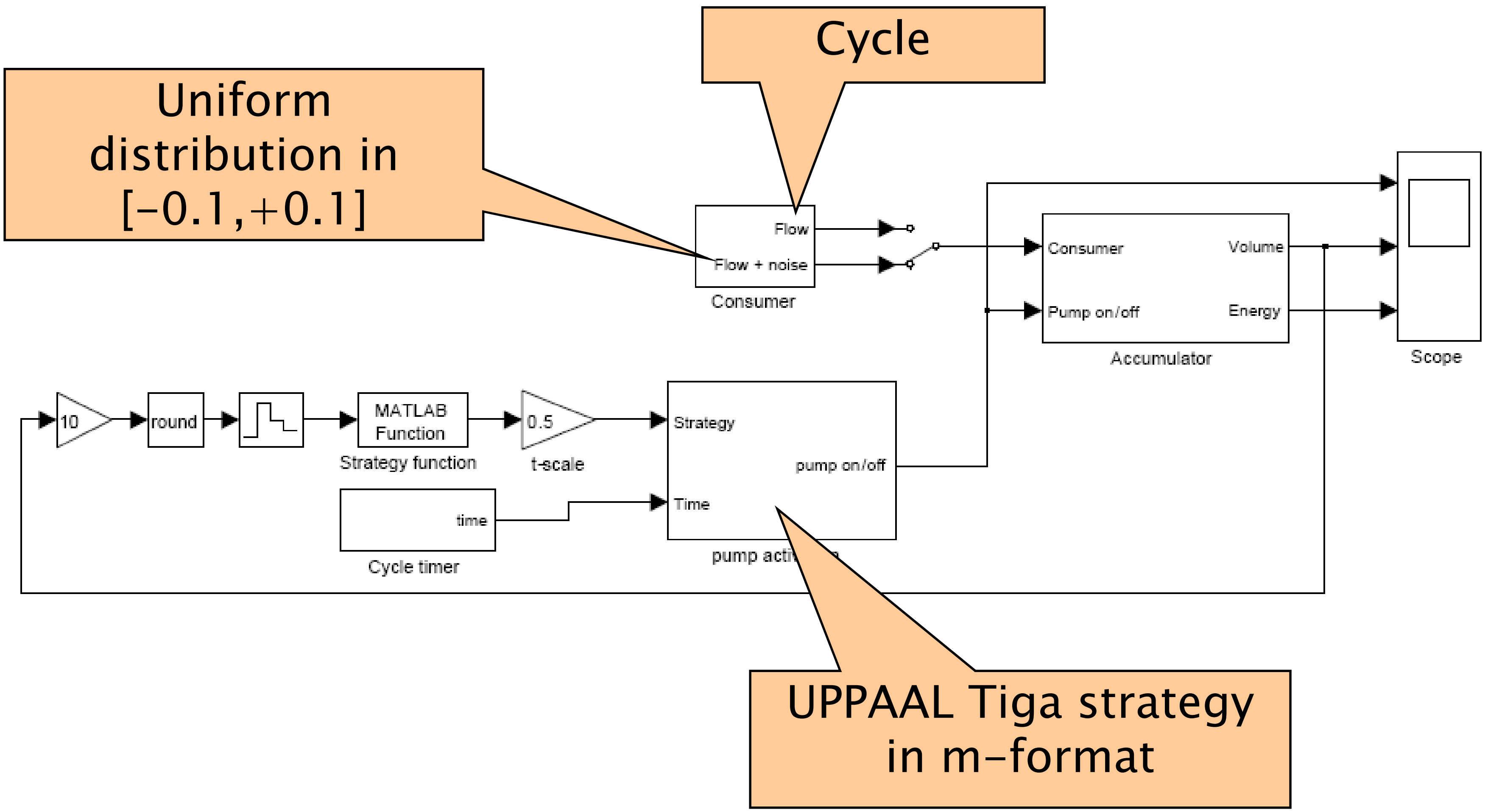
Bang-Bang safe and robust

HyDAC optimized
possibly unsafe under fluctuation

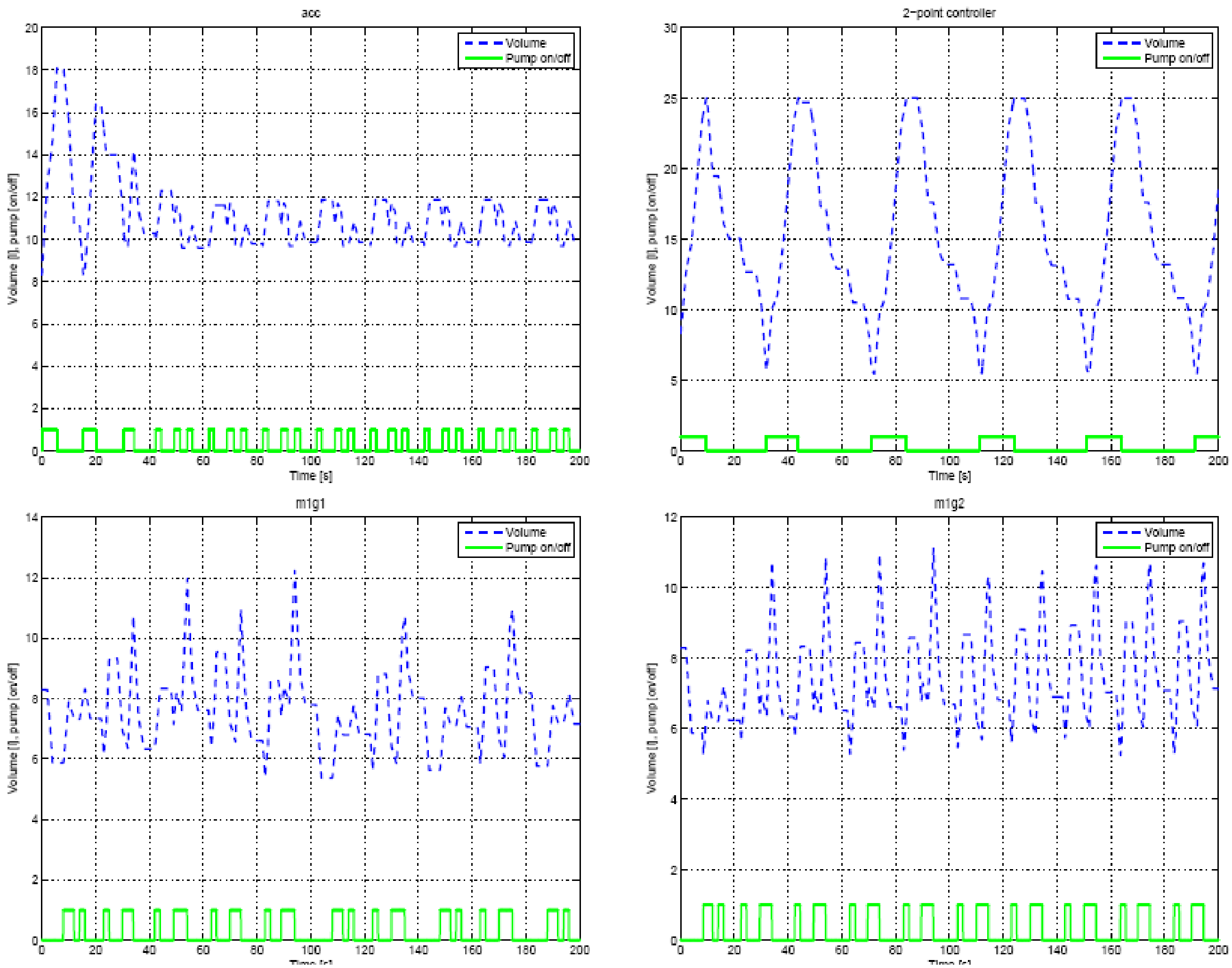




Performance SIMULINK



Results


acc
2−point controller
m1g1
m1g2
Volume
Pump on/off
Volume [l], pump [on/off]
Time [s]

Results

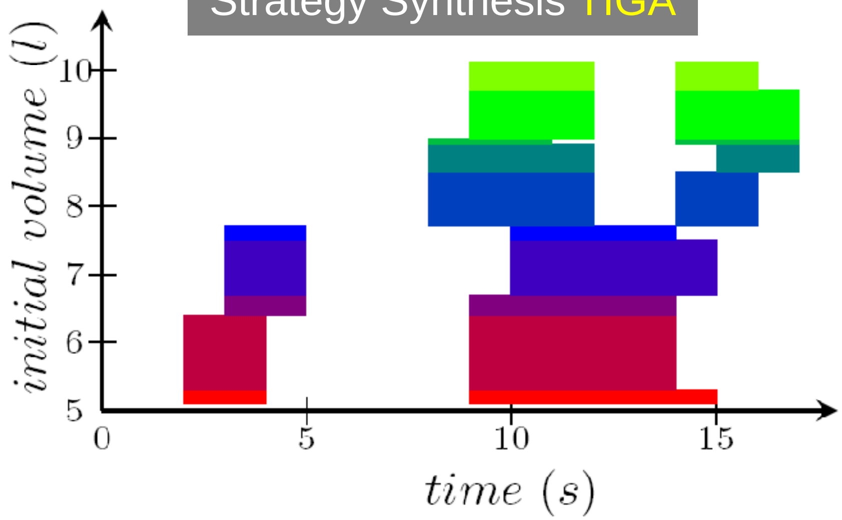
Controller	Acc. volume	Mean volume	Mean volume (TIGA)
Bang-Bang	2689	13.45	[illegible]
Hy…	[illegible]	[illegible]	[illegible]
G1…	[illegible]	[illegible]	[illegible]
G1…	[illegible]	[illegible]	[illegible]
G1…	[illegible]	[illegible]	[illegible]
G1…	[illegible]	[illegible]	[illegible]
G2…	[illegible]	[illegible]	[illegible]
G2M…	[illegible]	[illegible]	[illegible]
G2M…	[illegible]	7.5	7.95
G2M1	1489	7.44	7.95

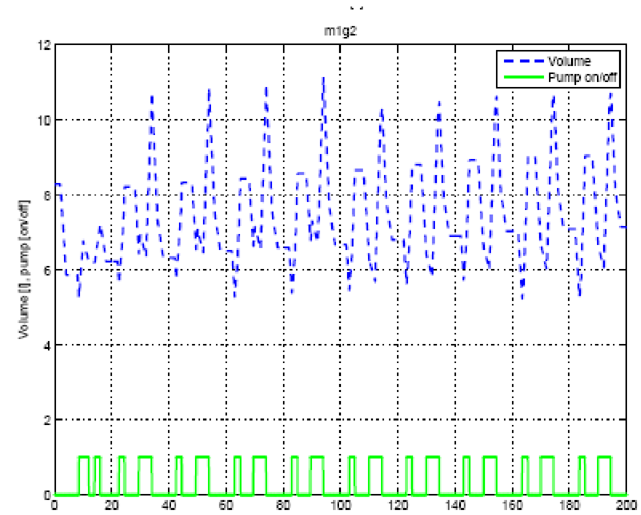
Guaranteed
Correctness
Robustness

with
40% improvement in performance

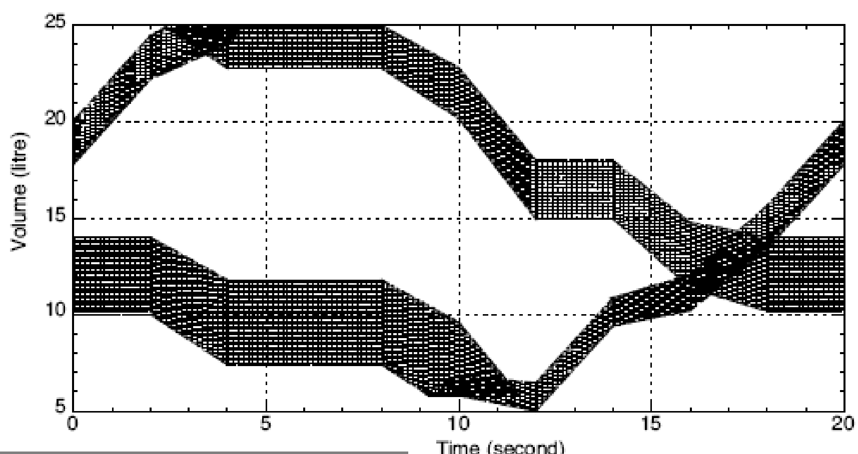
Tool Chain

Strategy Synthesis TIGA





Performance Evaluation
SIMULINK



Verification PHAVER

Guaranteed

Correctness

Robustness

with

40% Improvement

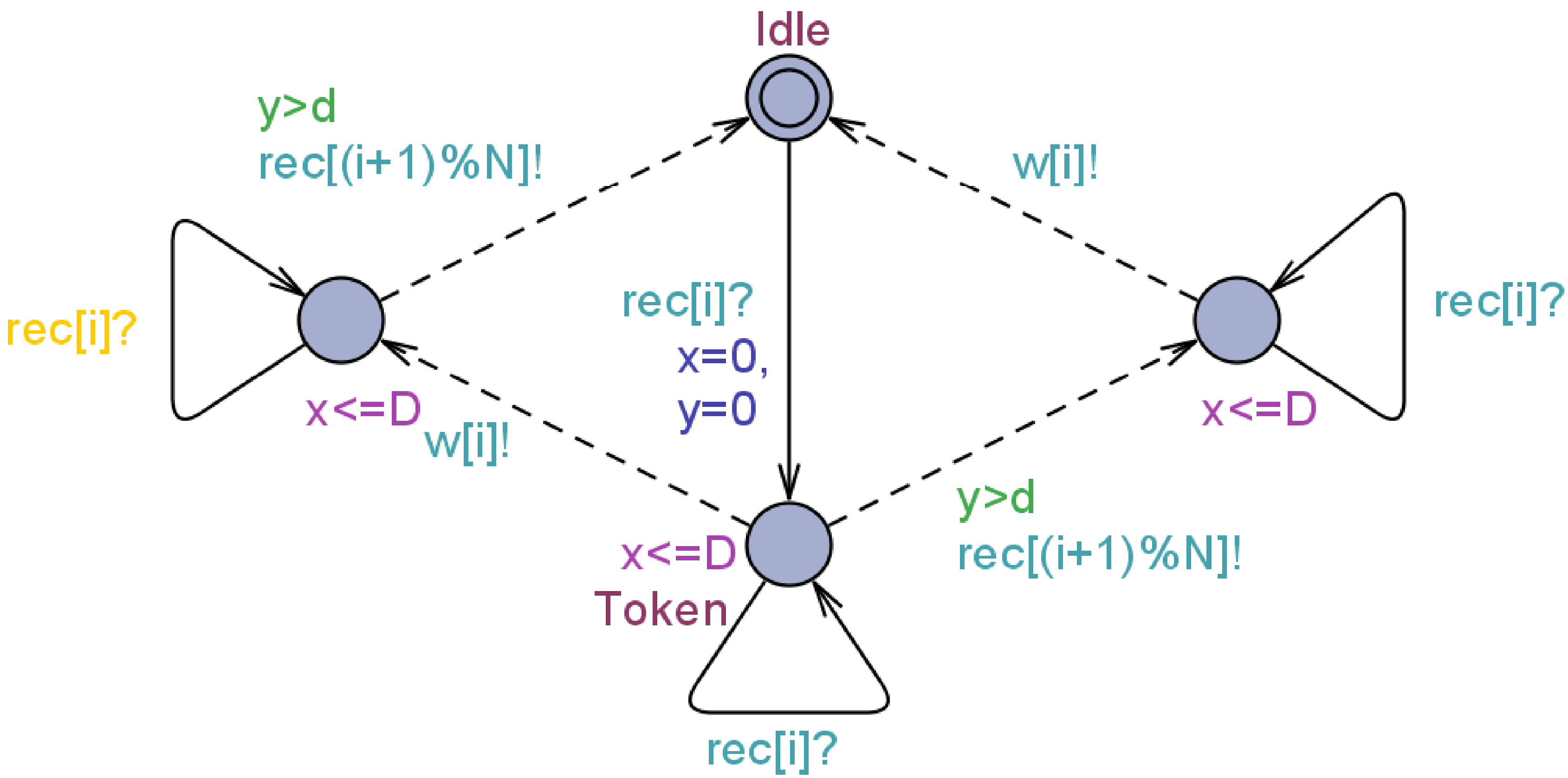
What else ? What next ?

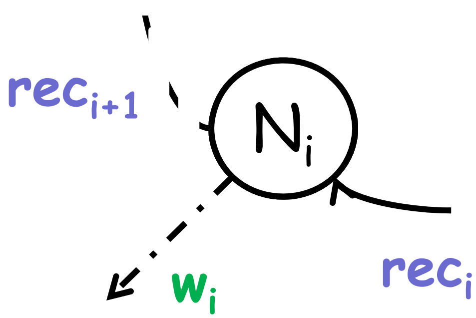
- Timed Games w Partial Observability
  - Action-based Observation: undecidable [BDMP03]
  - Finite-observation of states: decidable [CDL+07]
- Priced Timed Games:
  - Acyclic, cost non-zeno: decidable [LTMM02] [BCFL04]
  - 1 clock: decidable [BLMR06]
  - >2 clocks: undecidable [BBR05, BBM06]
  - 2 clocks: open
- Energy Games:
  - Several Open Problems
  - Exponential Observers
- Climate Controller in Pig Stables [JRLD07]
- CHESS Way [Quasimodo@ESWEEK]

[BDMP03] Bouyer, D'Souza, Madhusudan, Petit. Timed control with partial observability *(CAV'03)*.
[CDL+07] Cassez, David, Larsen, Lime, Raskin. Timed control with observation based and stuttering invariant strategies *(ATVA'07)*.
[JRLD07] Jessen, Rasmussen, Larsen, David. Guided Controller Synthesis for Climate Controller Using Uppaal Tiga *(FORMATS'07)*.

Timed Interfaces & Compositionality

Real-Time version of Milner's Scheduler

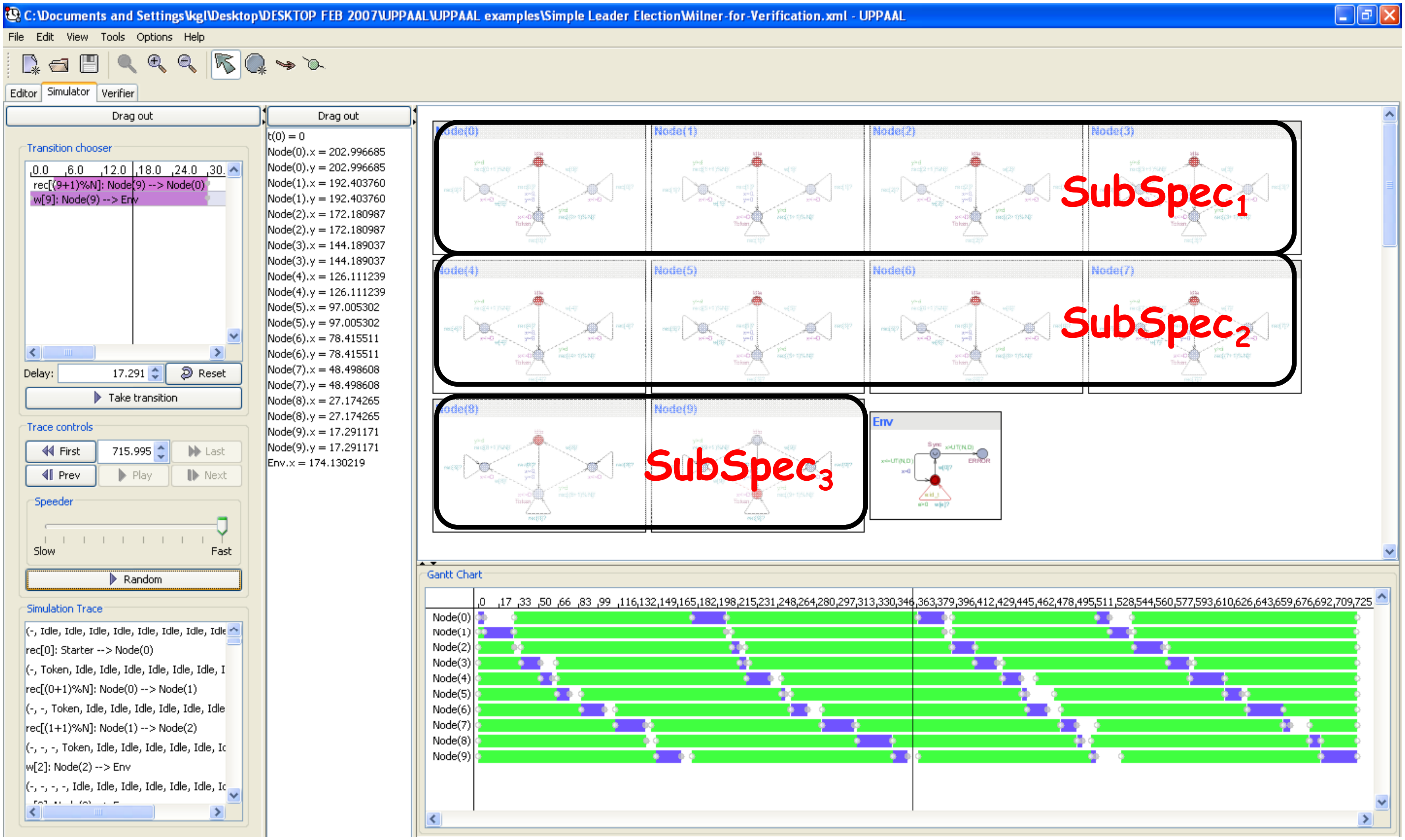




Demo

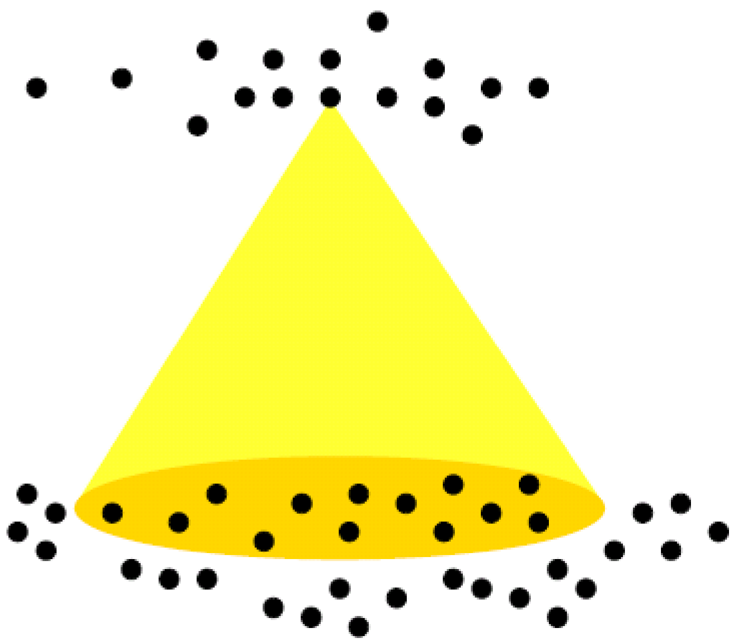
C:\Documents and Settings\kgl\Desktop\DESKTOP FEB 2007\UPPAAL\UPPAAL examples\Simple Leader Election\Milner-for-Verification.xml - UPPAAL
File Edit View Tools Options Help
Editor Simulator Verifier
Drag out
Transition chooser
rec[(9+1)%N]: Node(9) --> Node(0)
w[9]: Node(9) --> Env
Delay: 17.291
Reset
Take transition
Trace controls
First 715.995 Last
Prev Play Next
Speeder
Slow Fast
Random
Simulation Trace
(-, Idle, Idle, Idle, Idle, Idle, Idle, Idle, Idle
rec[0]: Starter --> Node(0)
(-, Token, Idle, Idle, Idle, Idle, Idle, Idle, I
rec[(0+1)%N]: Node(0) --> Node(1)
(-, -, Token, Idle, Idle, Idle, Idle, Idle, Idle
rec[(1+1)%N]: Node(1) --> Node(2)
(-, -, -, Token, Idle, Idle, Idle, Idle, Ic
w[2]: Node(2) --> Env
(-, -, -, -, Idle, Idle, Idle, Idle, Idle, Ic
Drag out
t(0) = 0
Node(0).x = 202.996685
Node(0).y = 202.996685
Node(1).x = 192.403760
Node(1).y = 192.403760
Node(2).x = 172.180987
Node(2).y = 172.180987
Node(3).x = 144.189037
Node(3).y = 144.189037
Node(4).x = 126.111239
Node(4).y = 126.111239
Node(5).x = 97.005302
Node(5).y = 97.005302
Node(6).x = 78.415511
Node(6).y = 78.415511
Node(7).x = 48.498608
Node(7).y = 48.498608
Node(8).x = 27.174265
Node(8).y = 27.174265
Node(9).x = 17.291171
Node(9).y = 17.291171
Env.x = 174.130219
Node(0) Node(1) Node(2) Node(3)
Node(4) Node(5) Node(6) Node(7)
Node(8) Node(9) Env
Gantt Chart
0 17 33 50 66 83 99 116 132 149 165 182 198 215 231 248 264 280 297 313 330 346 363 379 396 412 429 445 462 478 495 511 528 544 560 577 593 610 626 643 659 676 692 709 725
Node(0)
Node(1)
Node(2)
Node(3)
Node(4)
Node(5)
Node(6)
Node(7)
Node(8)
Node(9)

Compositional Verification



Specification Theory

Spec: set of specifications



Imp: set of implementations

Specification Formalism

$$SPF = (Imp, Spec, sat)$$

where

$$sat \subseteq Imp \times Spec$$

$$|S| = \{ I : I \text{ sat } S \}$$

Refinement:

$$S \leq T \text{ iff } |S| \subseteq |T|$$

Consistency:

$$|S| \neq \emptyset$$

$$|S| \cap |T| \neq \emptyset$$

Operations on Specifications

- **Logical Conjunction:**
  - Given S_1 and S_2 construct $S_1 \wedge S_2$ such that
    $$|S_1 \wedge S_2| = |S_1| \cap |S_2|$$

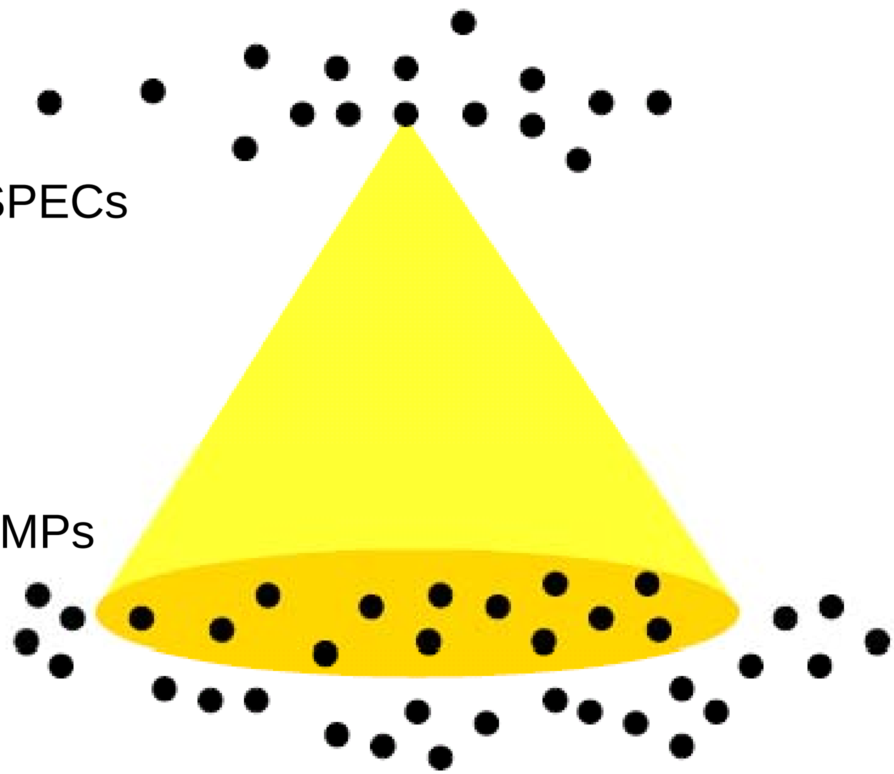
- **Structural Composition:**
  - Given S_1 and S_2 construct S_1 **par** S_2 such that
    $$| S_1 \textbf{ par } S_2 | = |S_1| \textbf{ par } |S_2|$$
  - $\leq$ should be precongruence wrt **par** to allow for compositional analysis !

- **Quotienting:**
  - Given overall specification T and component specification S construct the quotient specification T\S such that
    $$S \textbf{ par } X \leq T \quad \text{iff} \quad X \leq T\backslash S$$

Specifications and Implementations

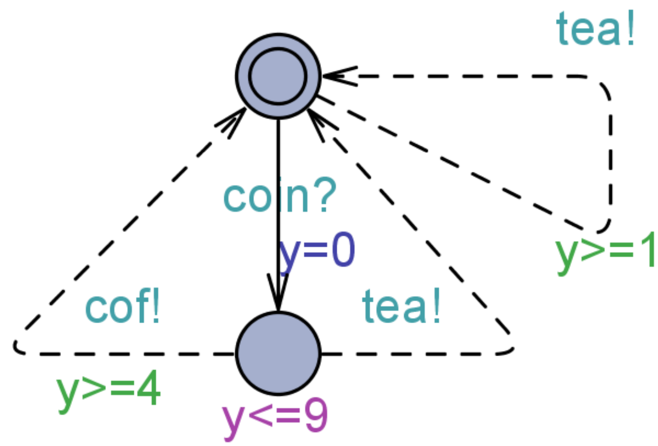
Timed I/O Transition Systems

SPECs

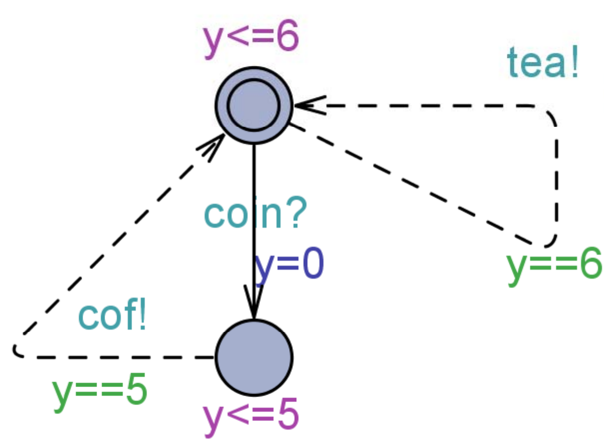


IMPs

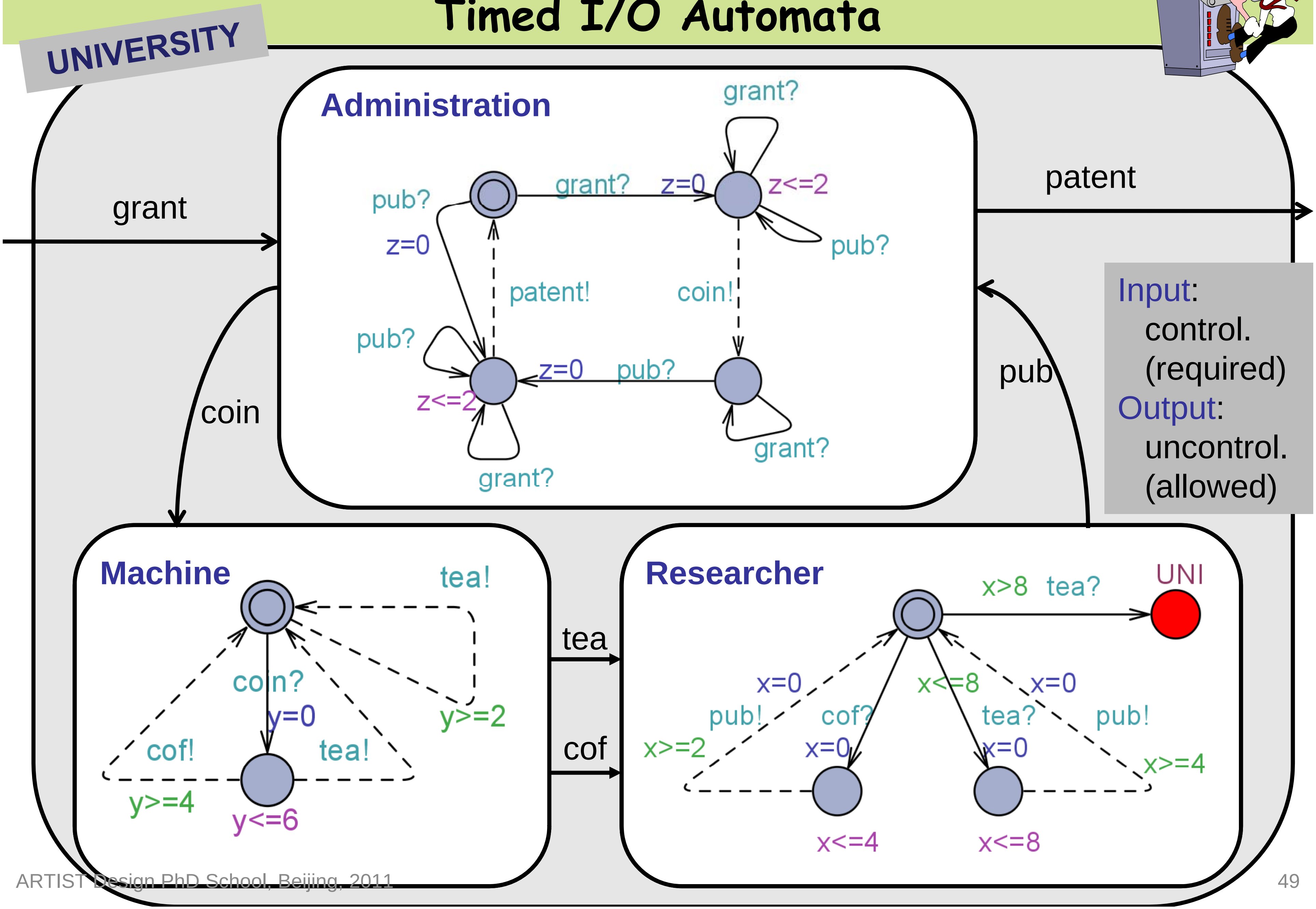
Timed I/O Transition Systems



Timed I/O Automata



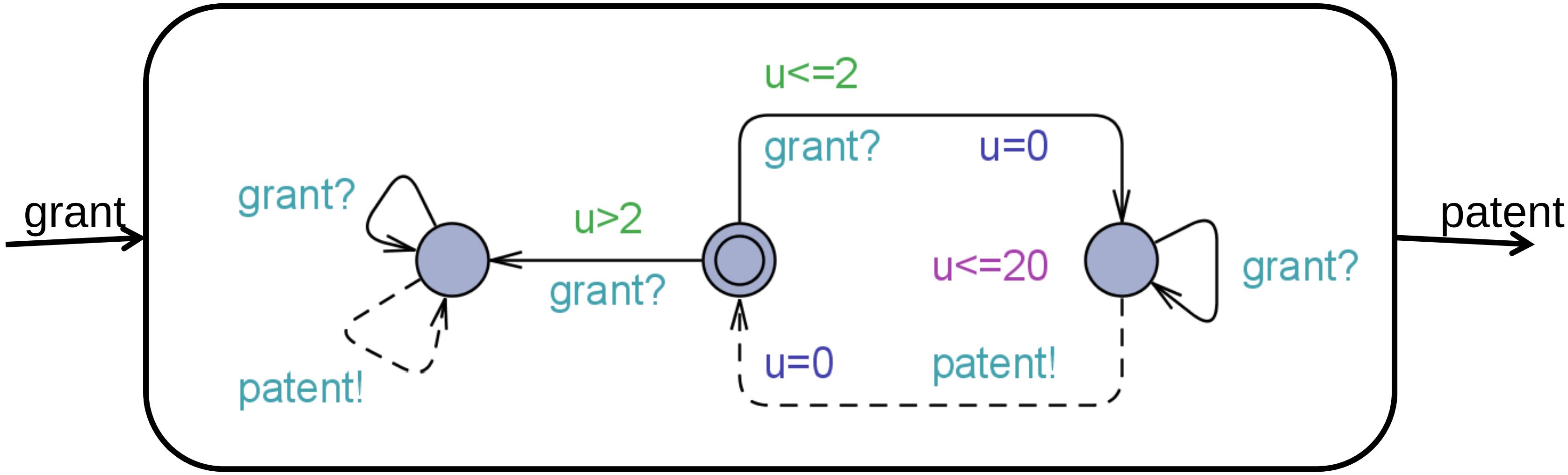
Timed Systems Specifications = Timed I/O Automata

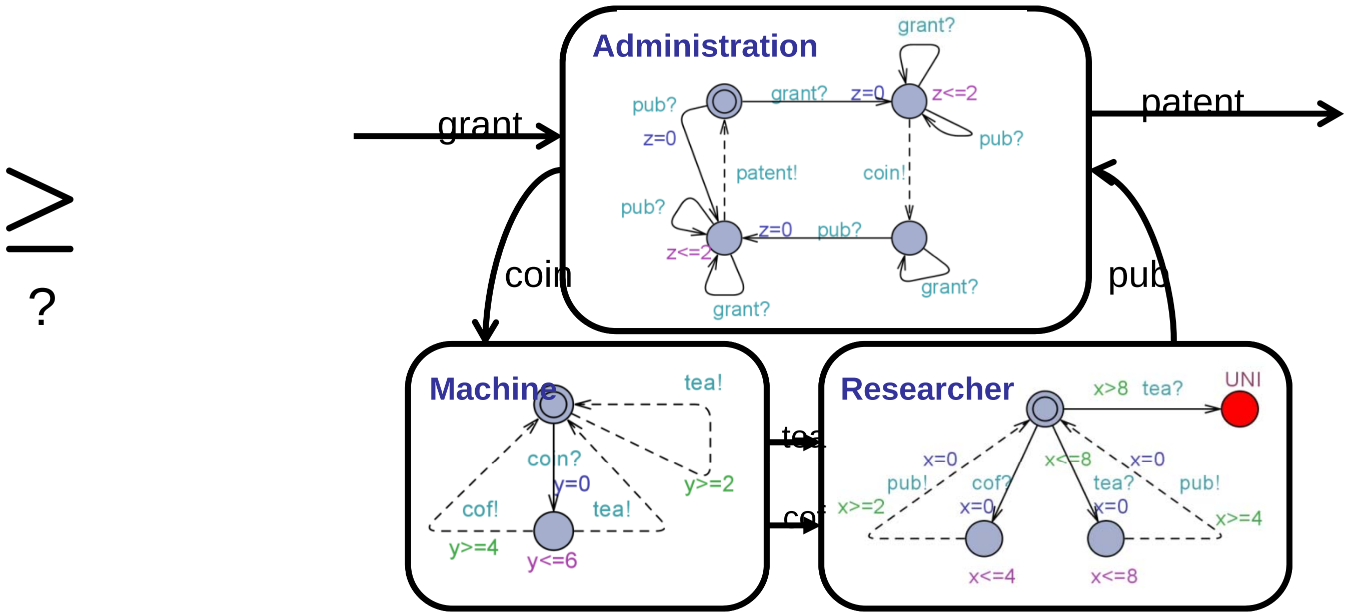


Input: control. (required)

Output: uncontrol. (allowed)

Overall Specification





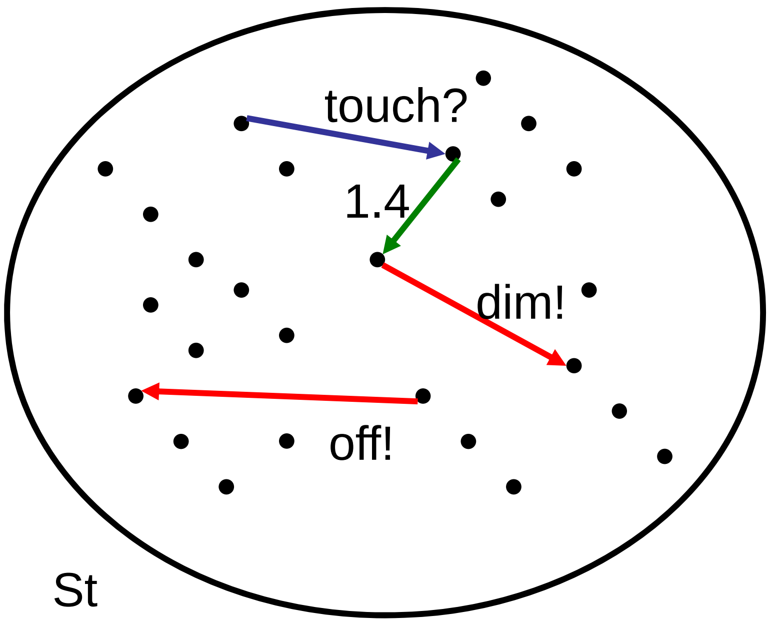
Timed I/O Transition Systems

TIOTS:

$(St, Act, \rightarrow)$

where $\rightarrow : St \times (Act \cup \Re) \times St$

and $Act = \Sigma_i \cup \Sigma_o$



Time determinism $(d \in \Re)$

$\quad$ *if* $s \xrightarrow{d} s'$ *and* $s \xrightarrow{d} s''$ *then* $s' = s''$

Input enabledness

$\quad$ *for all* s *and* $i \in \Sigma_i.\ \ s \xrightarrow{i}$

Determinism $(a \in Act \cup \Re)$

$\quad$ *if* $s \xrightarrow{a} s'$ *and* $s \xrightarrow{a} s''$ *then* $s' = s''$

Output Urgency

$\quad$ *whenever* $s \xrightarrow{o}$

$\quad\quad$ *then* $s \xrightarrow{d}$ *implies* $d = 0$

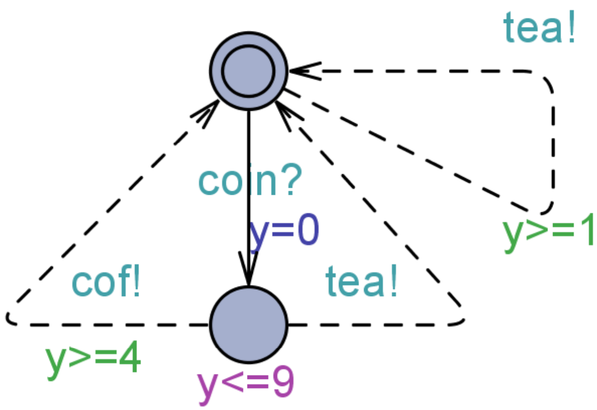
Independent Progress

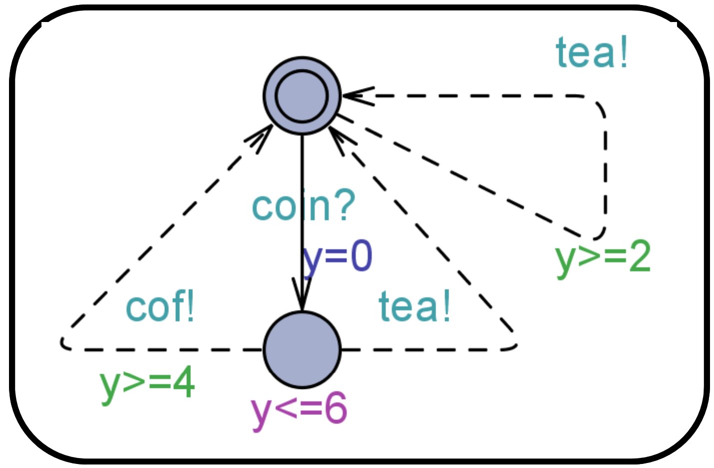
$\quad$ *Either* $\forall d \geq 0.\ \ s \xrightarrow{d}$

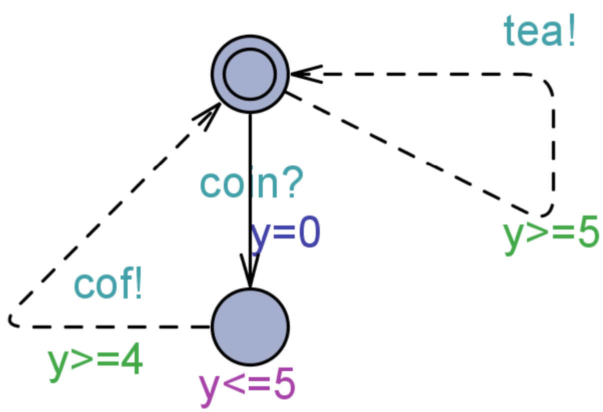
$\quad$ *or* $\quad \exists d, o.\ s \xrightarrow{d} \xrightarrow{o}$

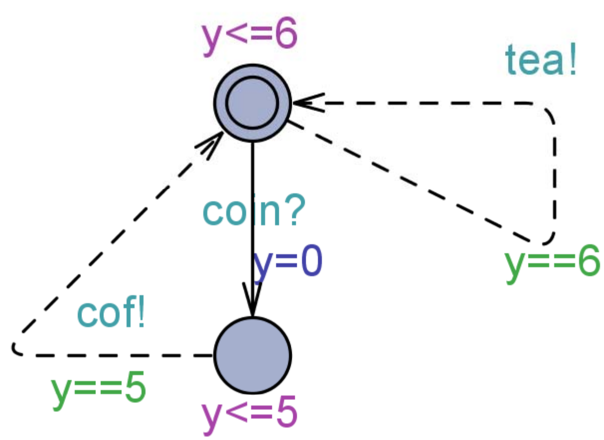
Implementations

Refinements, Implementations, Consistency

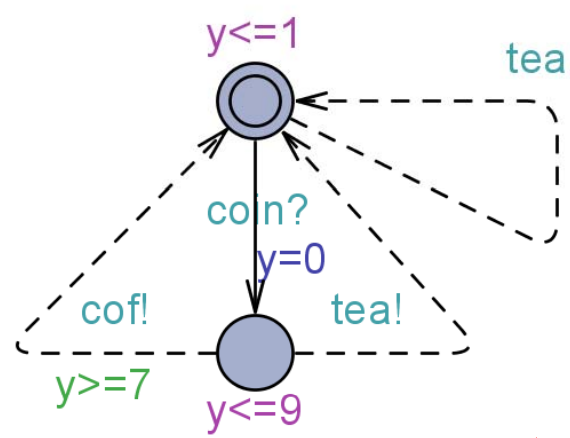








An Implementation



Inconsistent

Refinement = Timed Alternating Simulation

Let S and T be TIOGA.

$S \prec T$ iff

i. $T \xrightarrow{i?} T'$ then $S \xrightarrow{i?} S'$ with $S' \leq T'$

ii. $S \xrightarrow{o!} S'$ then $T \xrightarrow{o!} T'$ with $S' \leq T'$

iii. $S \xrightarrow{d} S'$ then $T \xrightarrow{d} T'$ with $S' \leq T'$

Intuition:
S leaves less choices than T
for an implementation.

Definition:

$I \text{ sat } S \overset{\Delta}{\Leftrightarrow} I \leq S$

Theorem

Whenever $S \leq T$ then $|S| \subseteq |T|$

Theorem:

Whenever $\emptyset \neq |S| \subseteq |T|$ then $S \leq T$

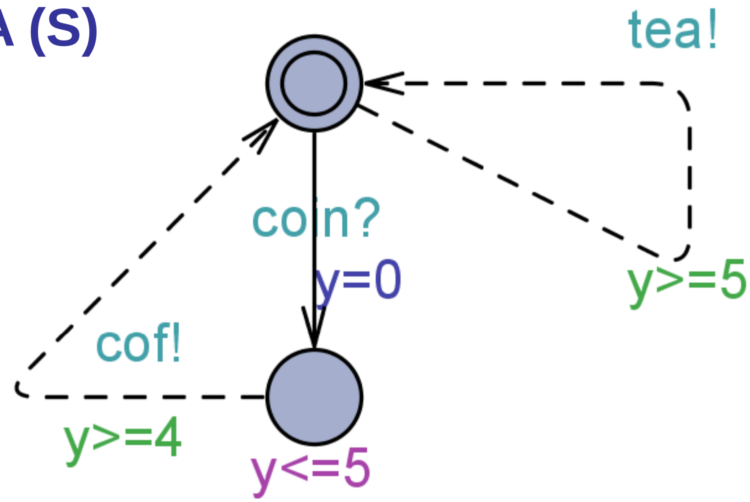
Theorem

$S \leq T \Rightarrow$

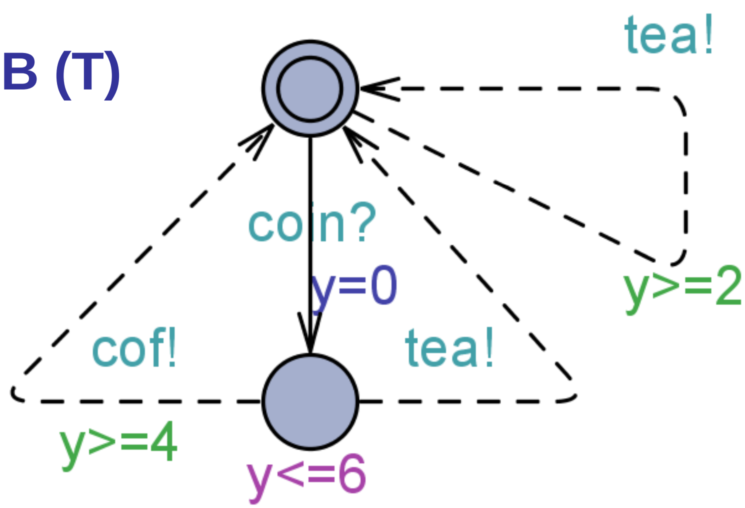
$(\forall \Phi \in ATCTL.\ T \text{ cntr } \Phi \Rightarrow S \text{ cntr } \Phi)$

Refinement (example)

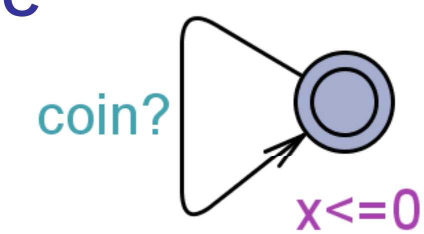
A (S)



B (T)



INC



Let S and T be TIOGA.

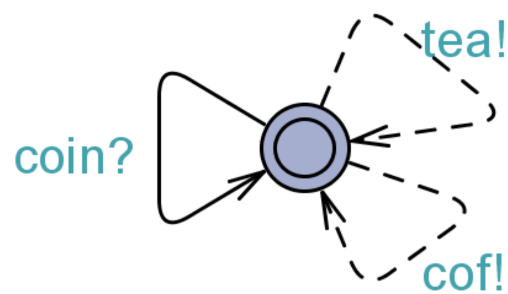
$S \prec T$ iff

i. $T \xrightarrow{i?} T'$ then $S \xrightarrow{i?} S'$ with $S' \leq T'$

ii. $S \xrightarrow{o!} S'$ then $T \xrightarrow{o!} T'$ with $S' \leq T'$

iii. $S \xrightarrow{d} S'$ then $T \xrightarrow{d} T'$ with $S' \leq T'$

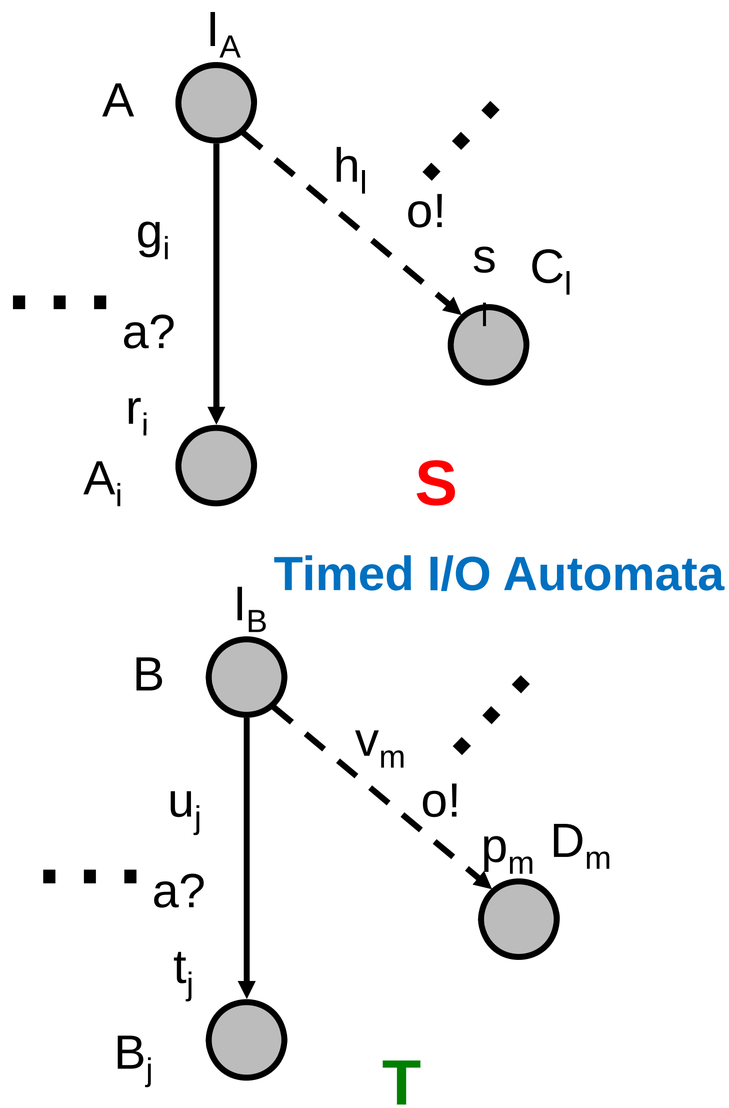
UNI

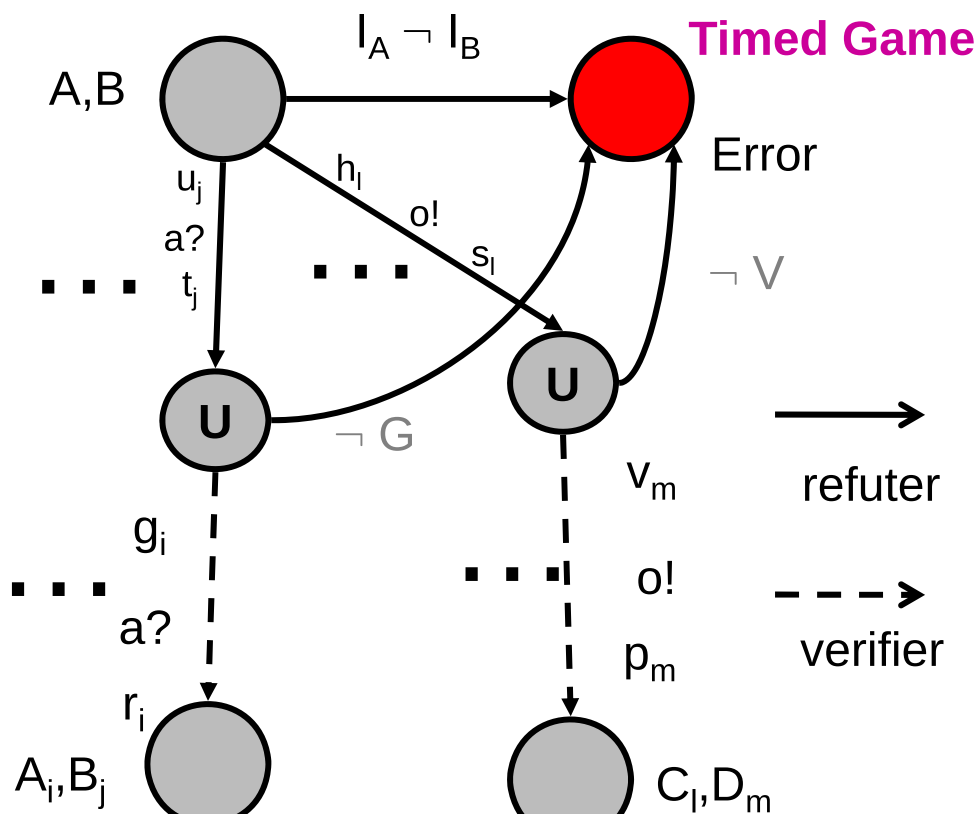


Refinement as a Game

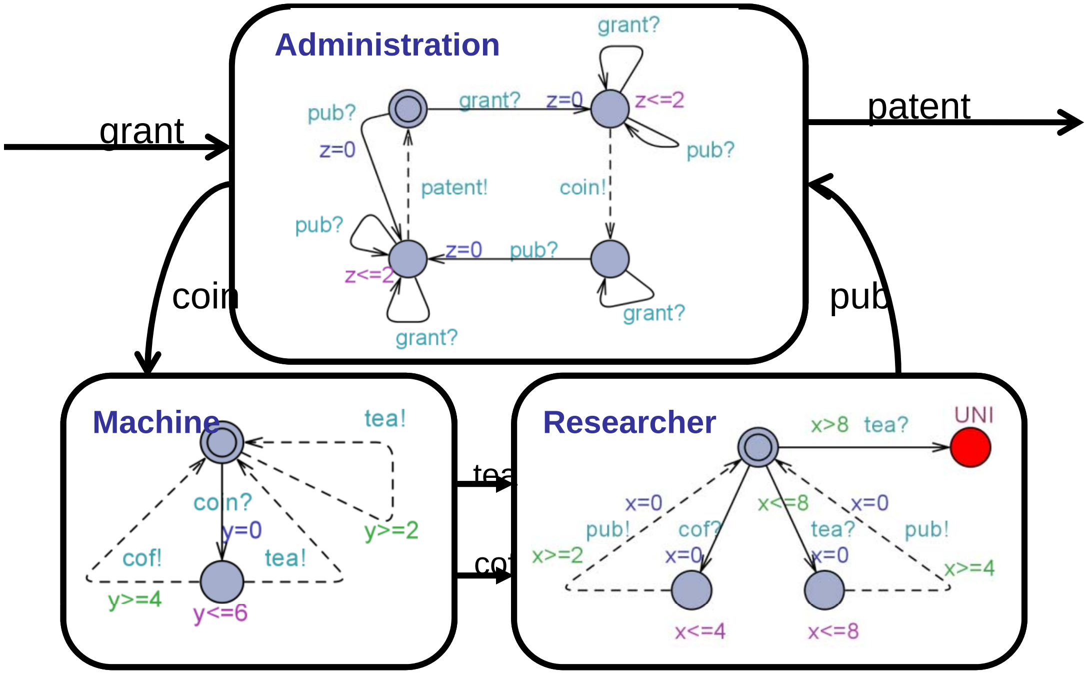
FORMATS09
Optimized Refinement Algorithm

not **A** $\leq$ **B**
iff
AxB **sat** control: A<> Error

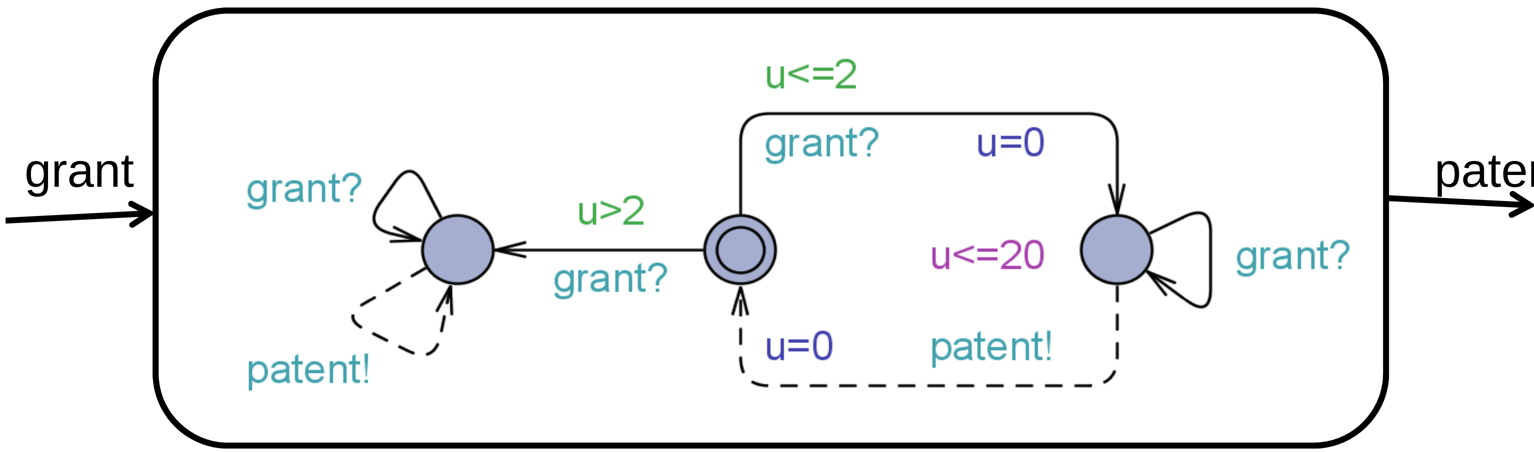




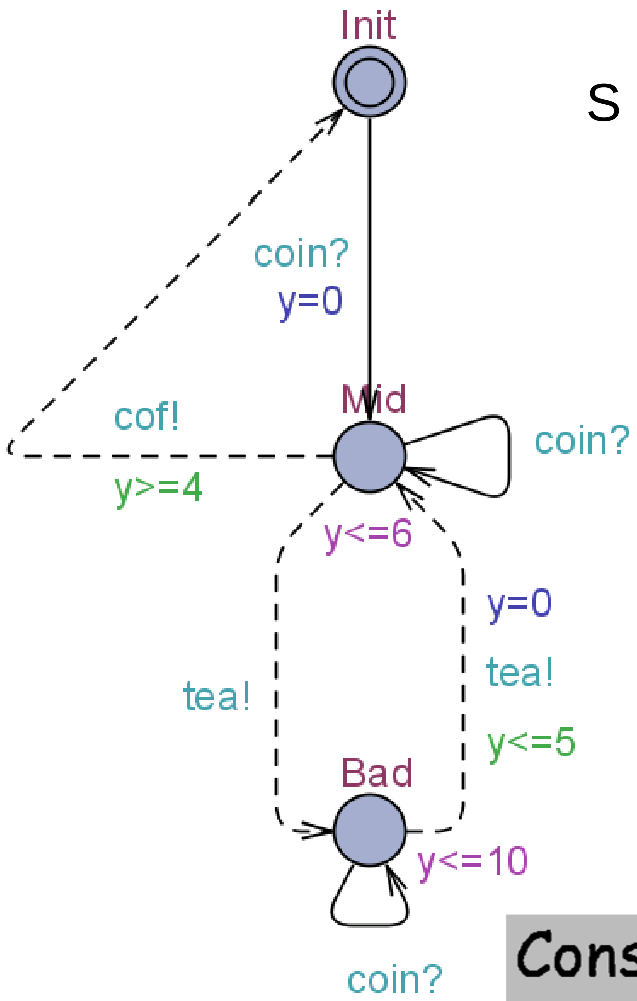
Refinement in ECDAR



$\leq$?????

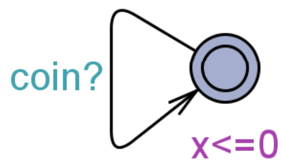


Consistency

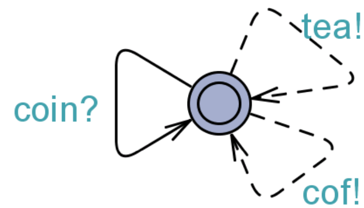


S

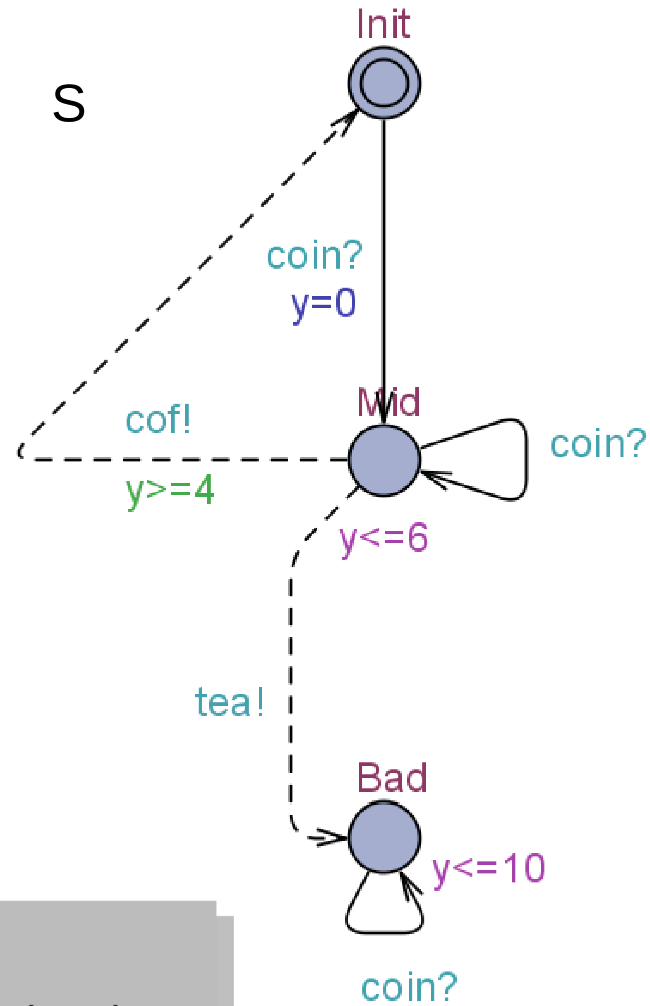
S



S



S

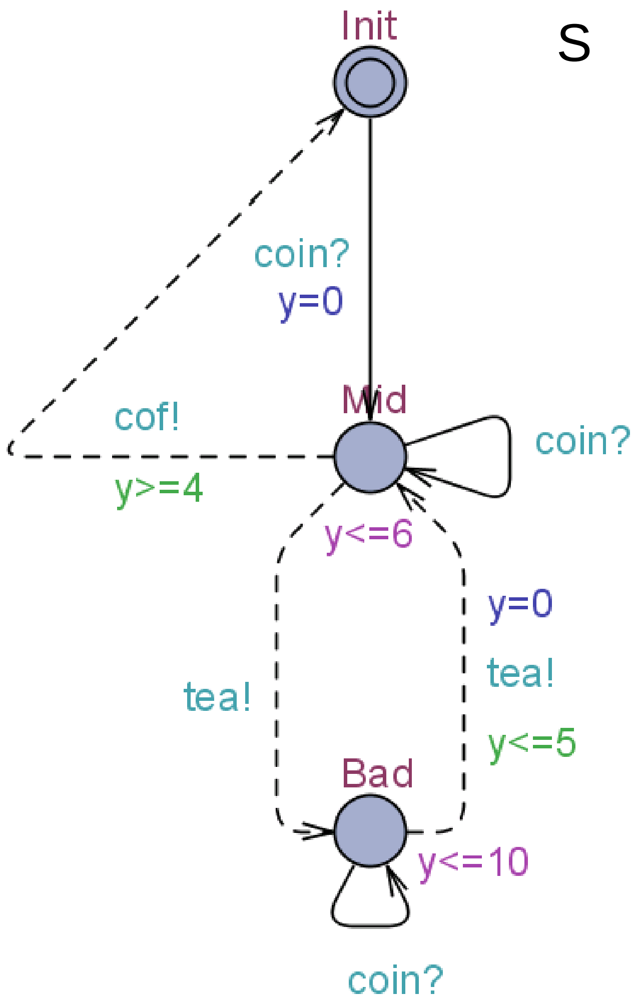


Consistency:

Does there exist I such that

$I \leq S$?

Consistency



Definitions

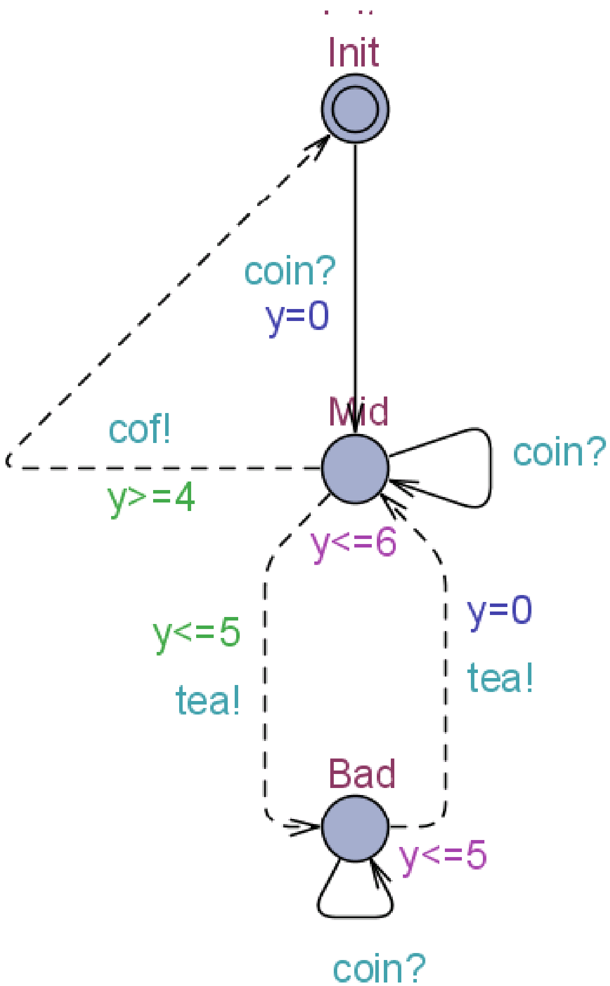
Err =

$$\{\, s \mid \forall d > 0. \neg s \xrightarrow{d} \wedge \forall o. \neg s \xrightarrow{o} \}$$

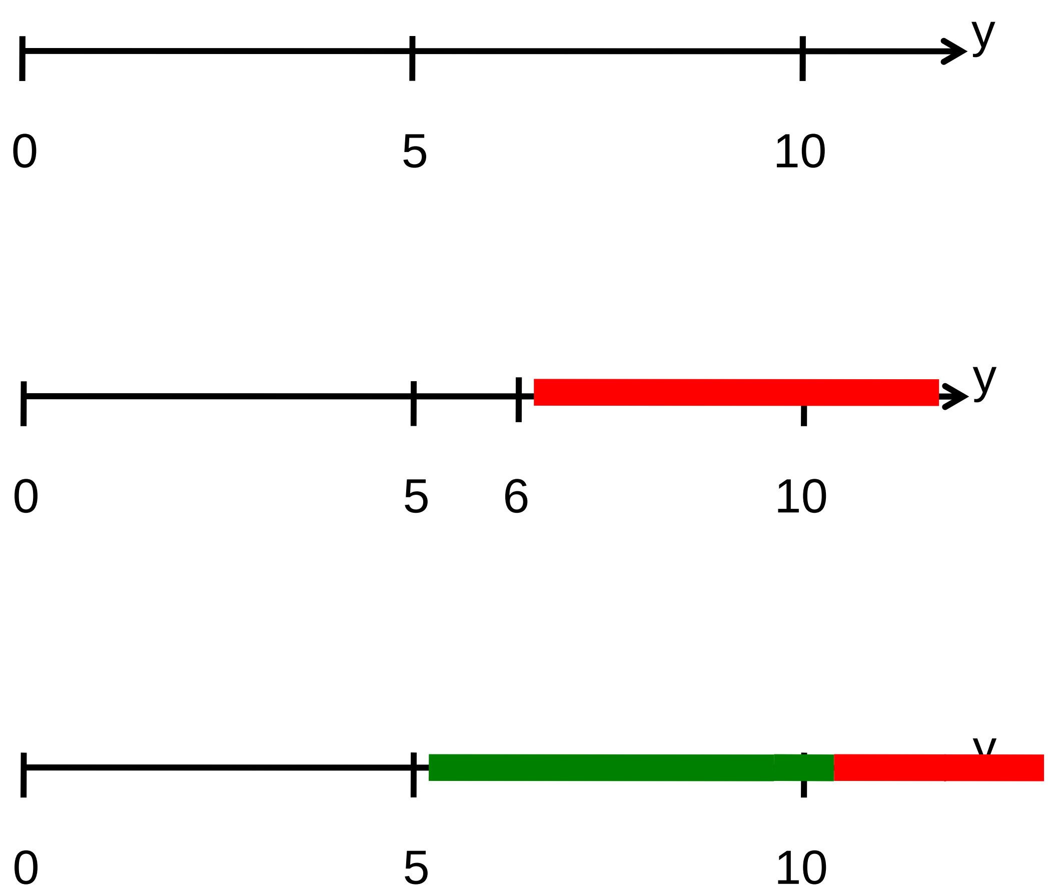
$\pi(X) =$
  Err $\cup$
  Pred_t[X $\cup$ iPred(X) , oPred(X^C)]

Theorem
A specificiation (state) s is
 inconsistent
iff
 $s \in \mu X.\ \pi(X)$

Consistency



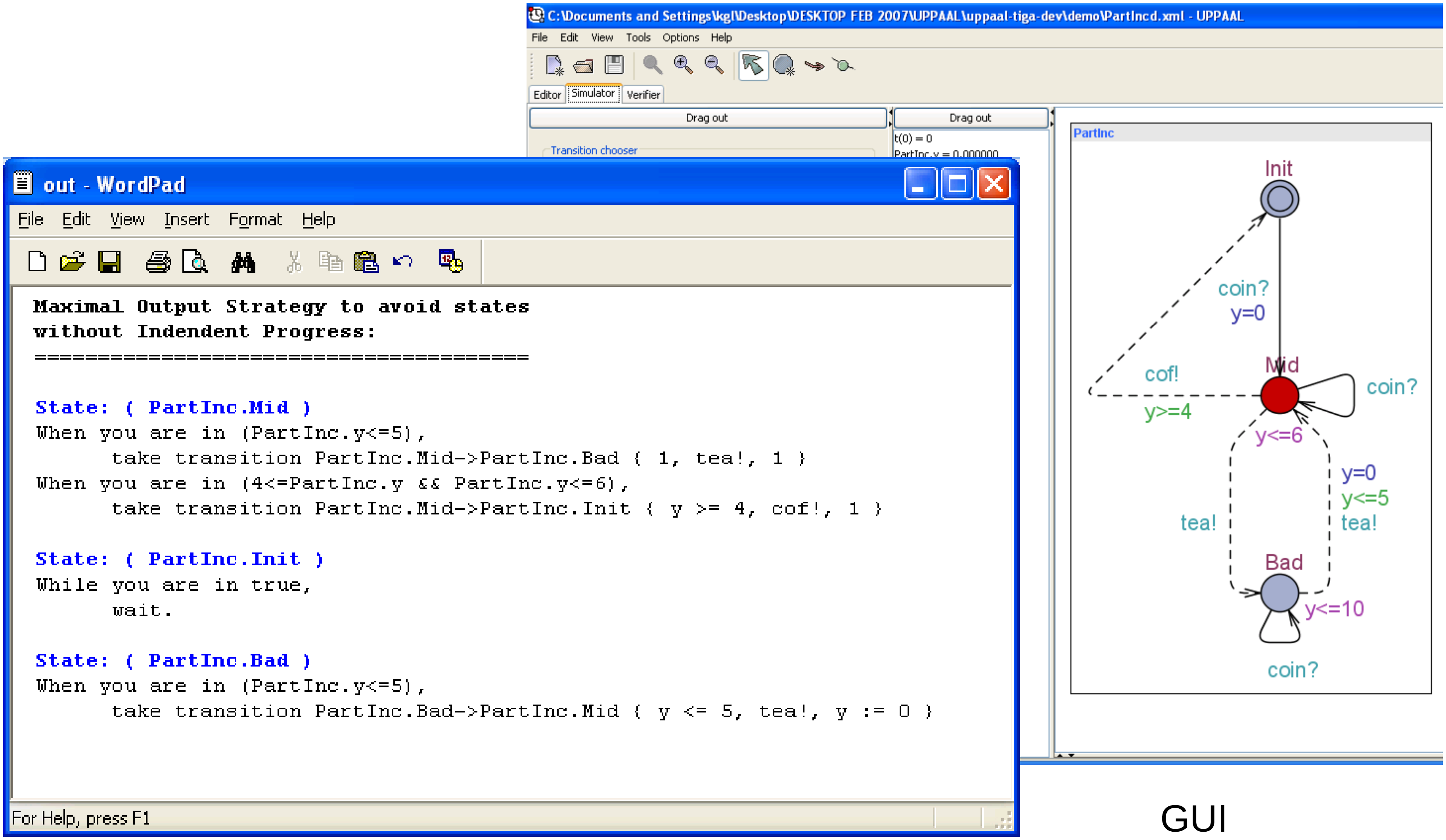
Pruned Version



$\pi(X) = Err \cup Pred_t[\ X \cup iPred(X)\ ,\ oPred(X^C)\ ]$

$Err = \{\ s \mid \forall d > 0. \neg s \xrightarrow{d} \ \wedge \forall o. \neg s \xrightarrow{o} \}$

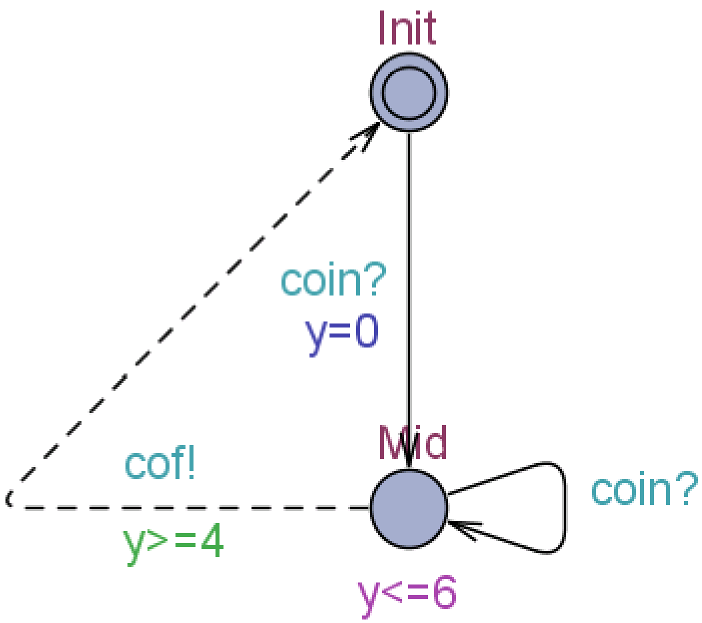
Consistency in UPPAAL Tiga+

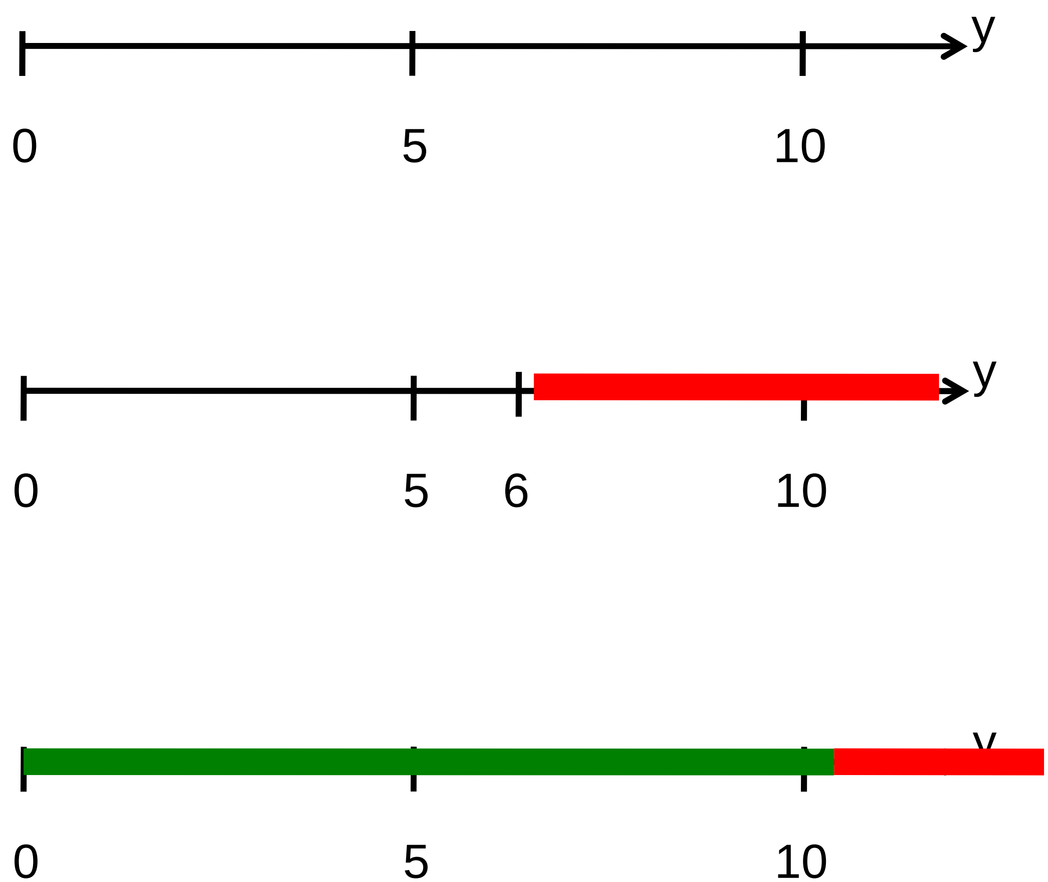


GUI

Command-Line

Consistency



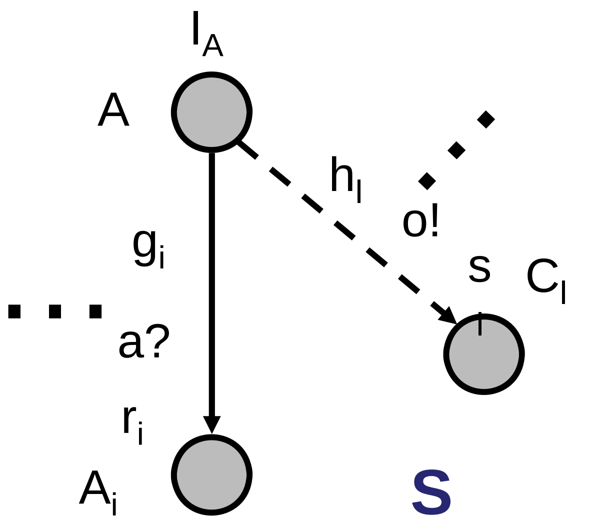


Pruned Version

$$\pi(X) = Err \cup Pred_t[\, X \cup iPred(X)\,,\, oPred(X^C)\,]$$

$$Err = \{\, s \mid \forall d > 0. \neg s \xrightarrow{d} \wedge \forall o. \neg s \xrightarrow{o} \}$$

Conjunction, S∧T

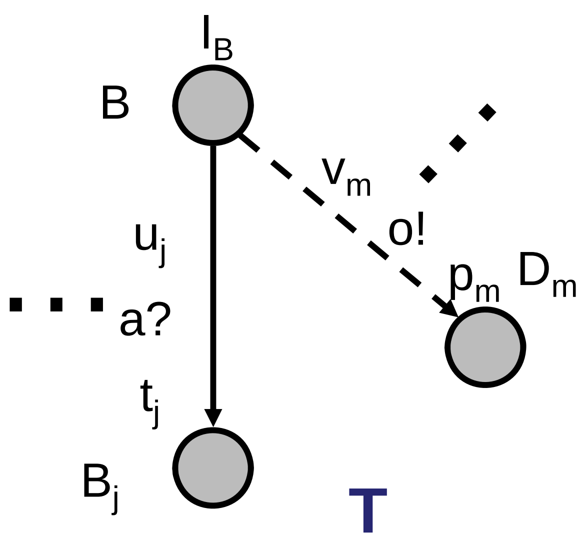


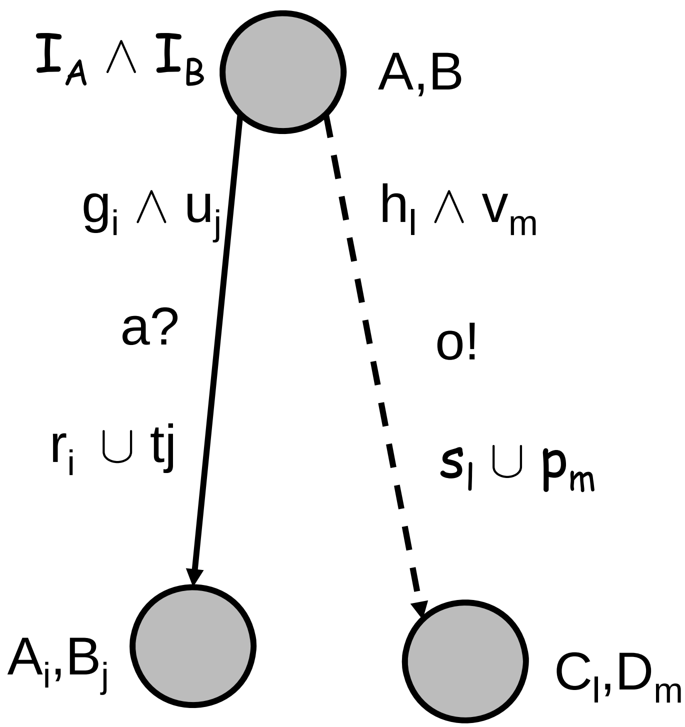
Theorem

$S \wedge T \leq S$

$S \wedge T \leq T$

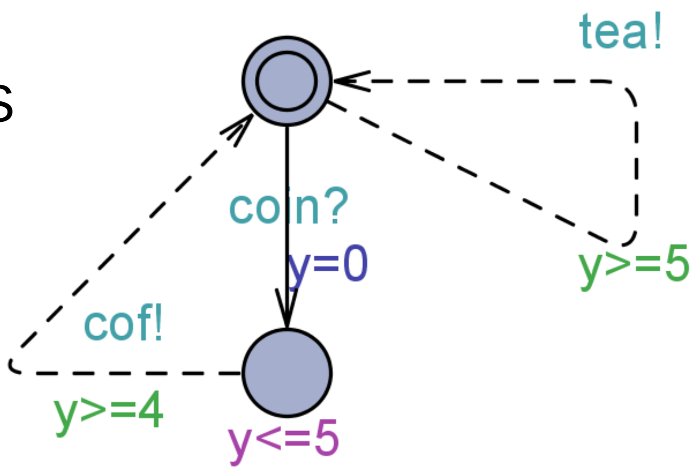
$(U \leq S)$ and $(U \leq T) \Rightarrow U \leq (S \wedge T)$



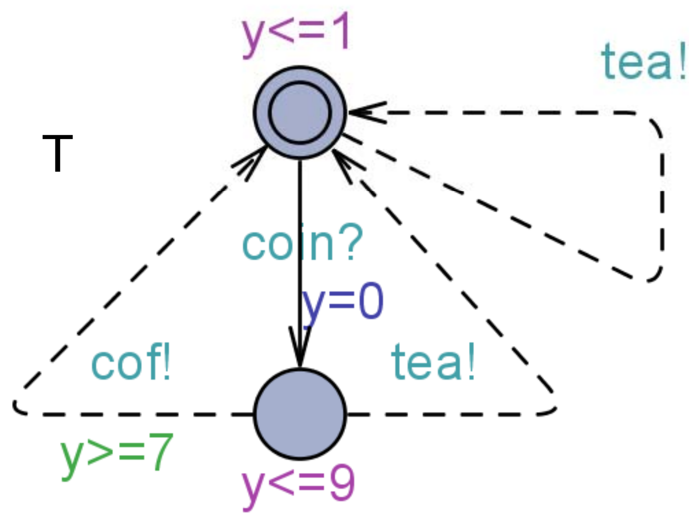


Conjunction, Ex.

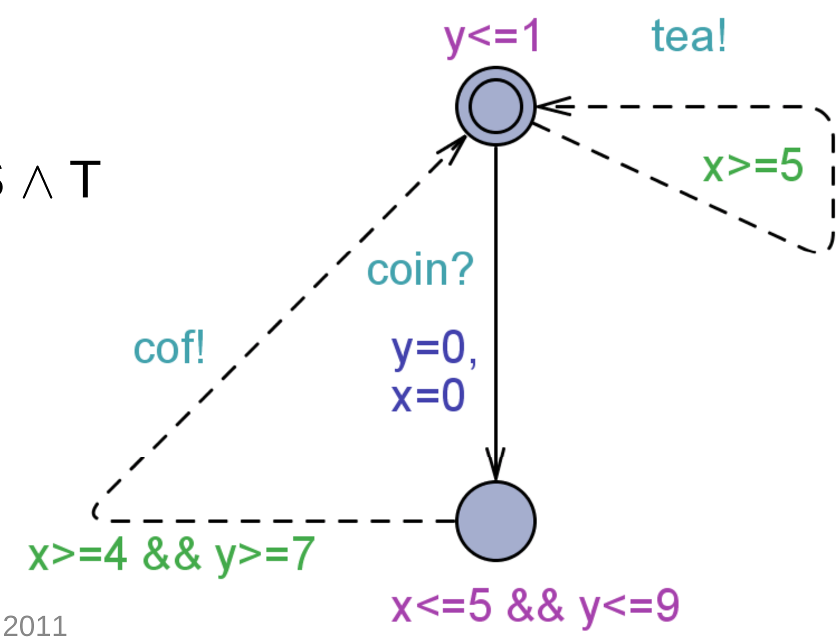
S



T



S ∧ T



Clearly Inconsistent !

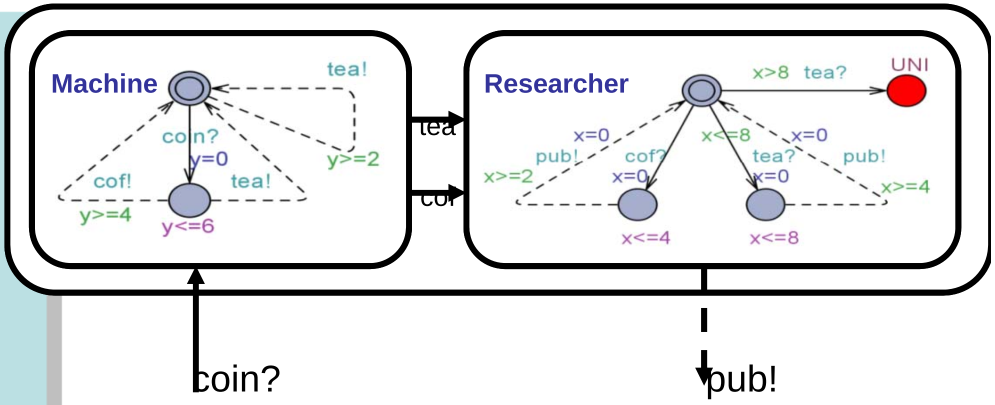
Composition, S|T

$$\frac{s_1 \xrightarrow{i?} s_1'}{s_1 | s_2 \xrightarrow{i?} s_1' | s_2}\; i \in \Sigma_i^1 - \Sigma_0^2$$

$$\frac{s_1 \xrightarrow{o!} s_1'}{s_1 | s_2 \xrightarrow{o!} s_1' | s_2}\; o \in \Sigma_o^1 - \Sigma_i^2$$

$$\frac{s_1 \xrightarrow{a!} s_1' \quad s_2 \xrightarrow{a?} s_2'}{s_1 | s_2' \xrightarrow{a!} s_1' | s_2'}\; a \in \Sigma_o^1 \cap \Sigma_i^2$$

$$\frac{s_1 \xrightarrow{d} s_1' \quad s_2 \xrightarrow{d} s_2'}{s_1 | s_2' \xrightarrow{d} s_1' | s_2'}\; d \in \Re$$

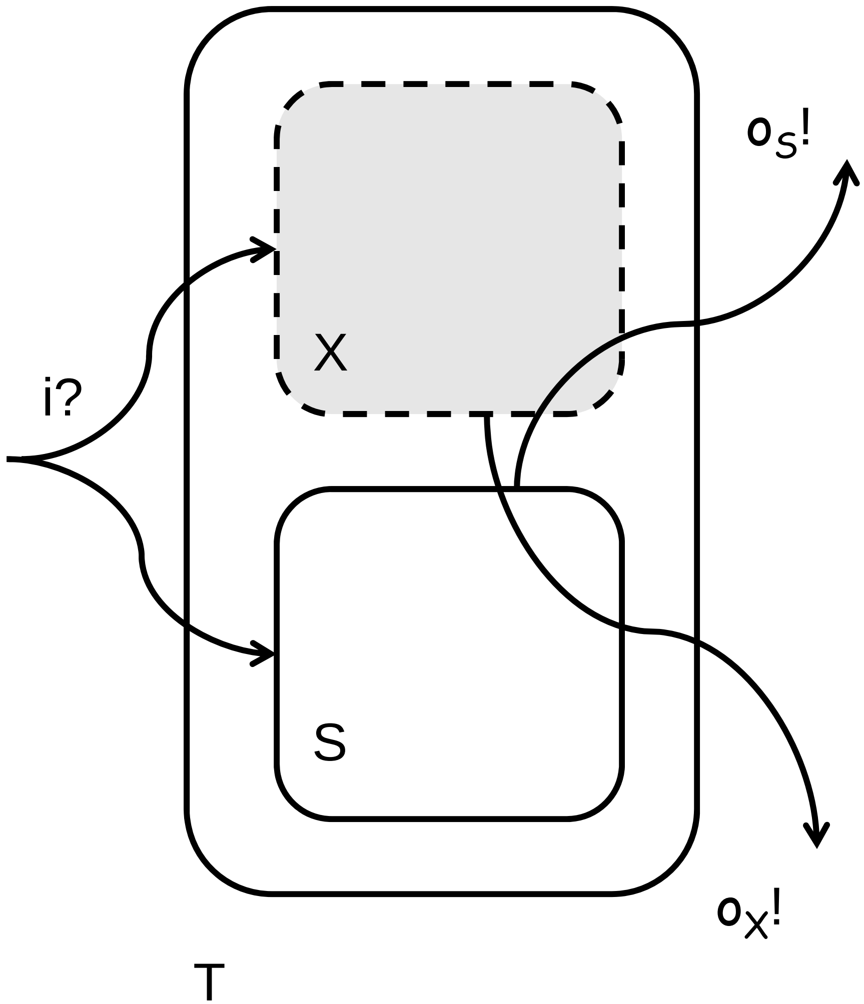


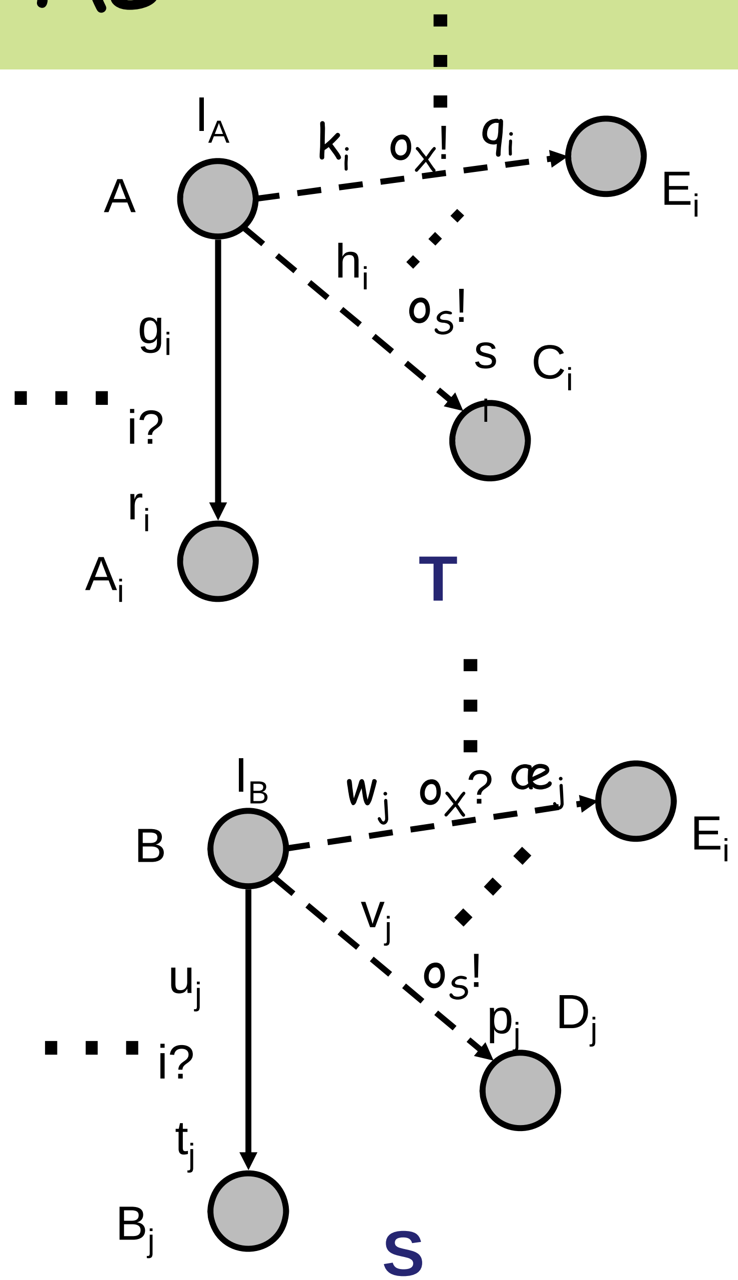
Theorem

If $A_1 \leq B_1$ and
$A_2 \leq B_2$
then
$A_1|A_2 \leq B_1|B_2$

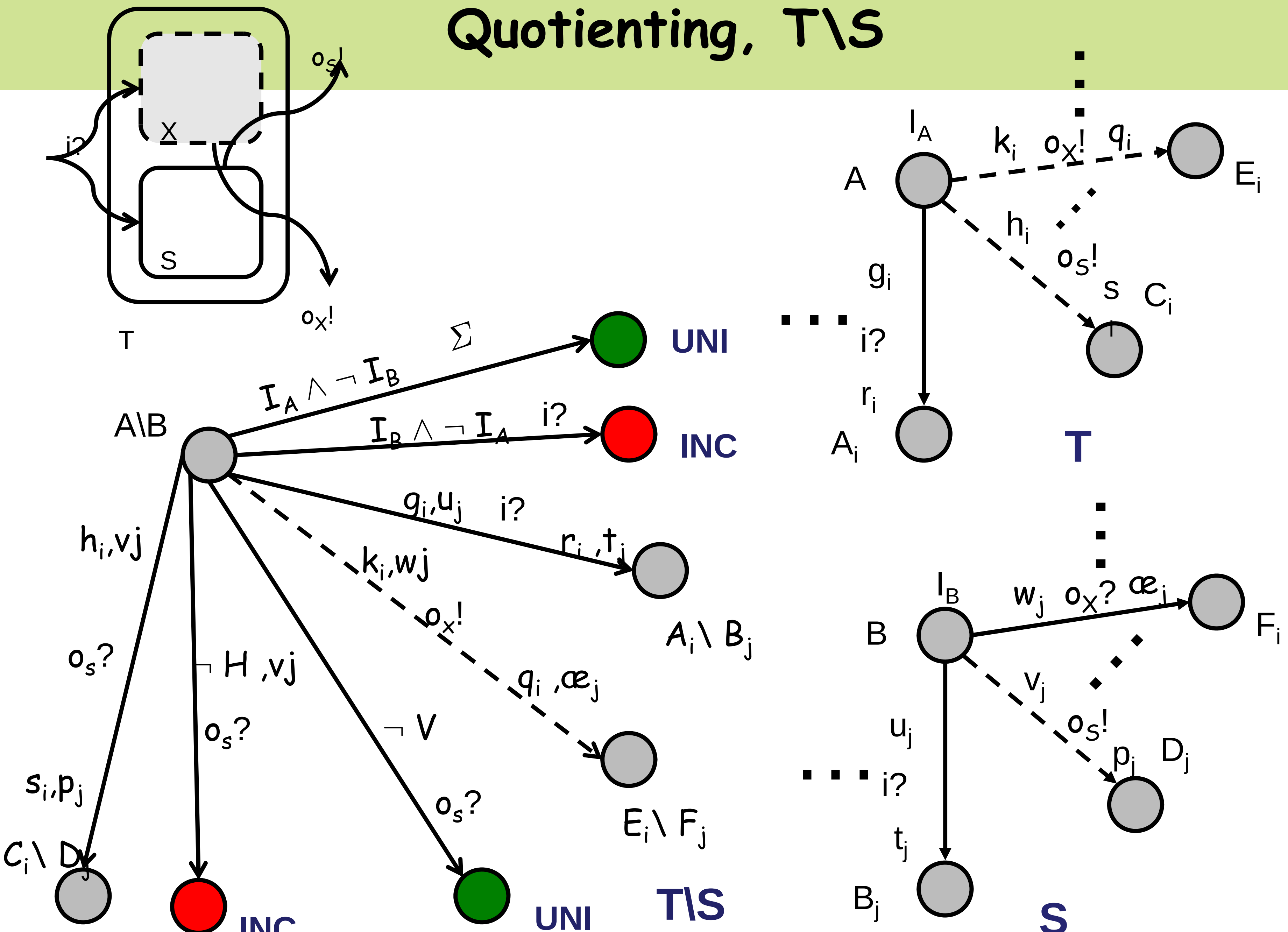
Classical rules for Composition of I/O transition Systems

Quotienting, T\S

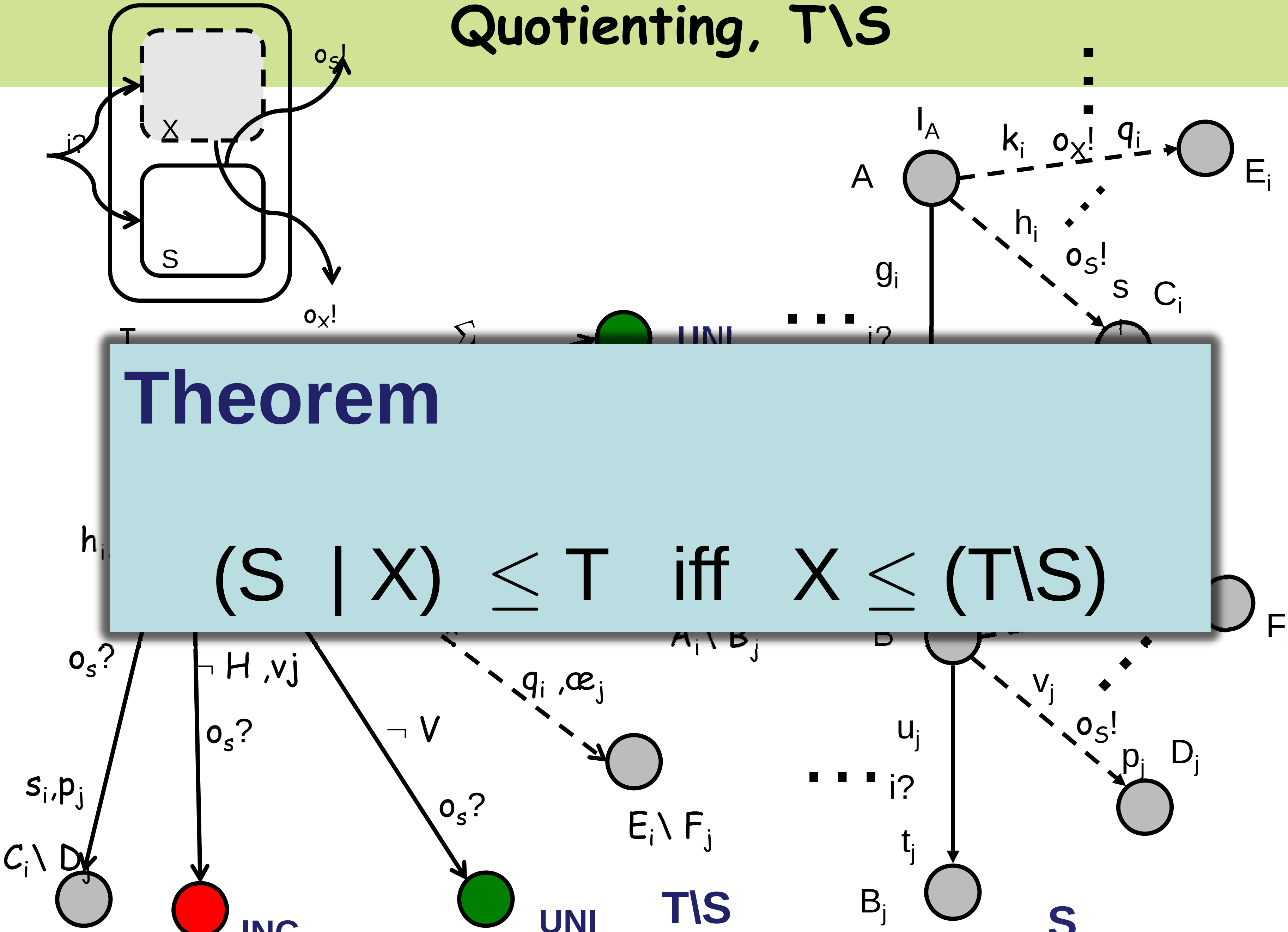




Quotienting, T\S



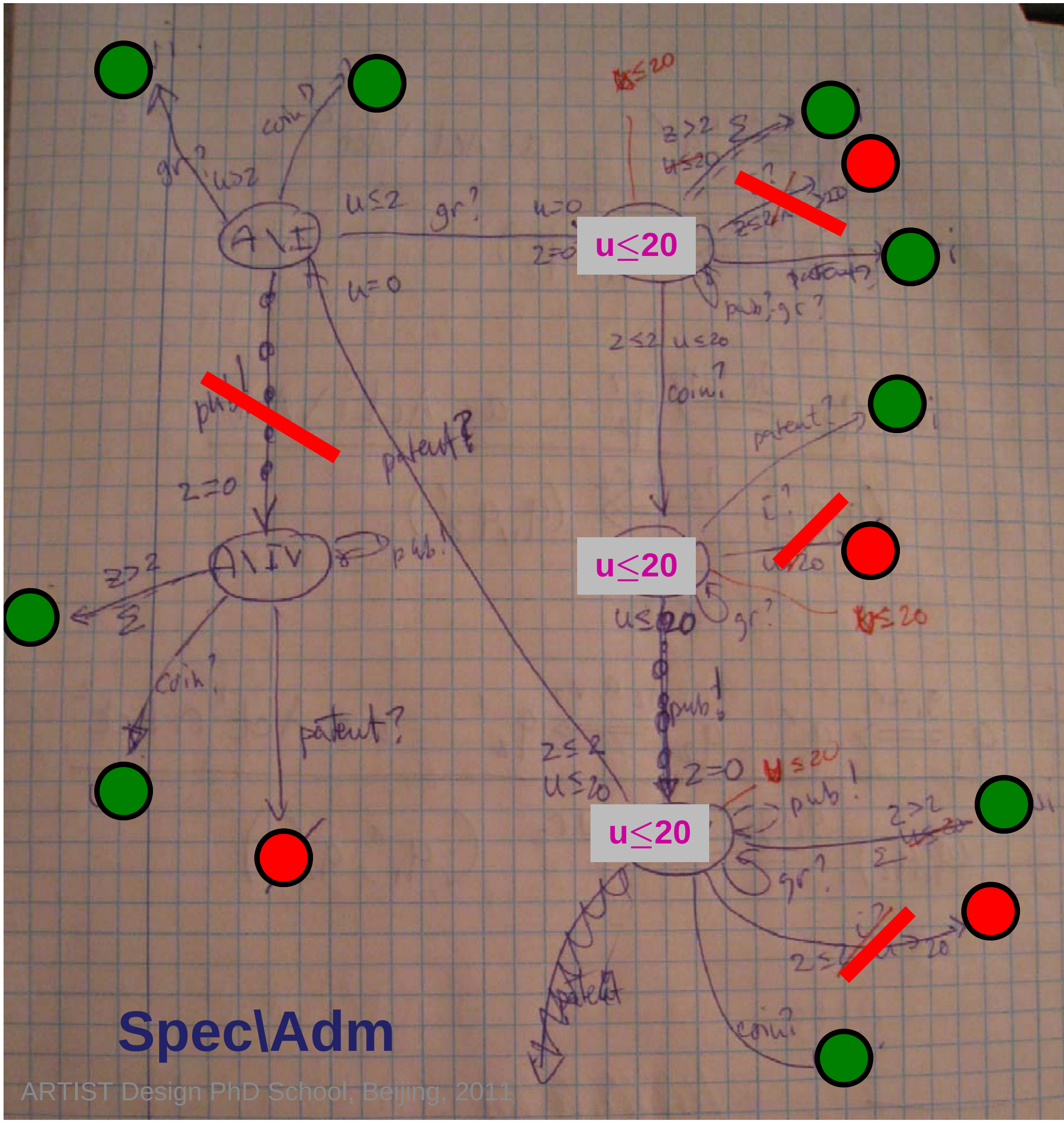
Quotienting, T\S

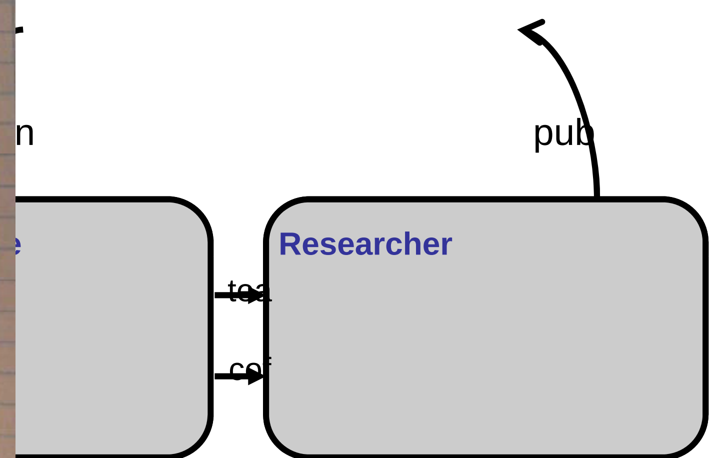


Theorem

$$(S \mid X) \leq T \quad \text{iff} \quad X \leq (T\backslash S)$$

Quotienting, "Application"

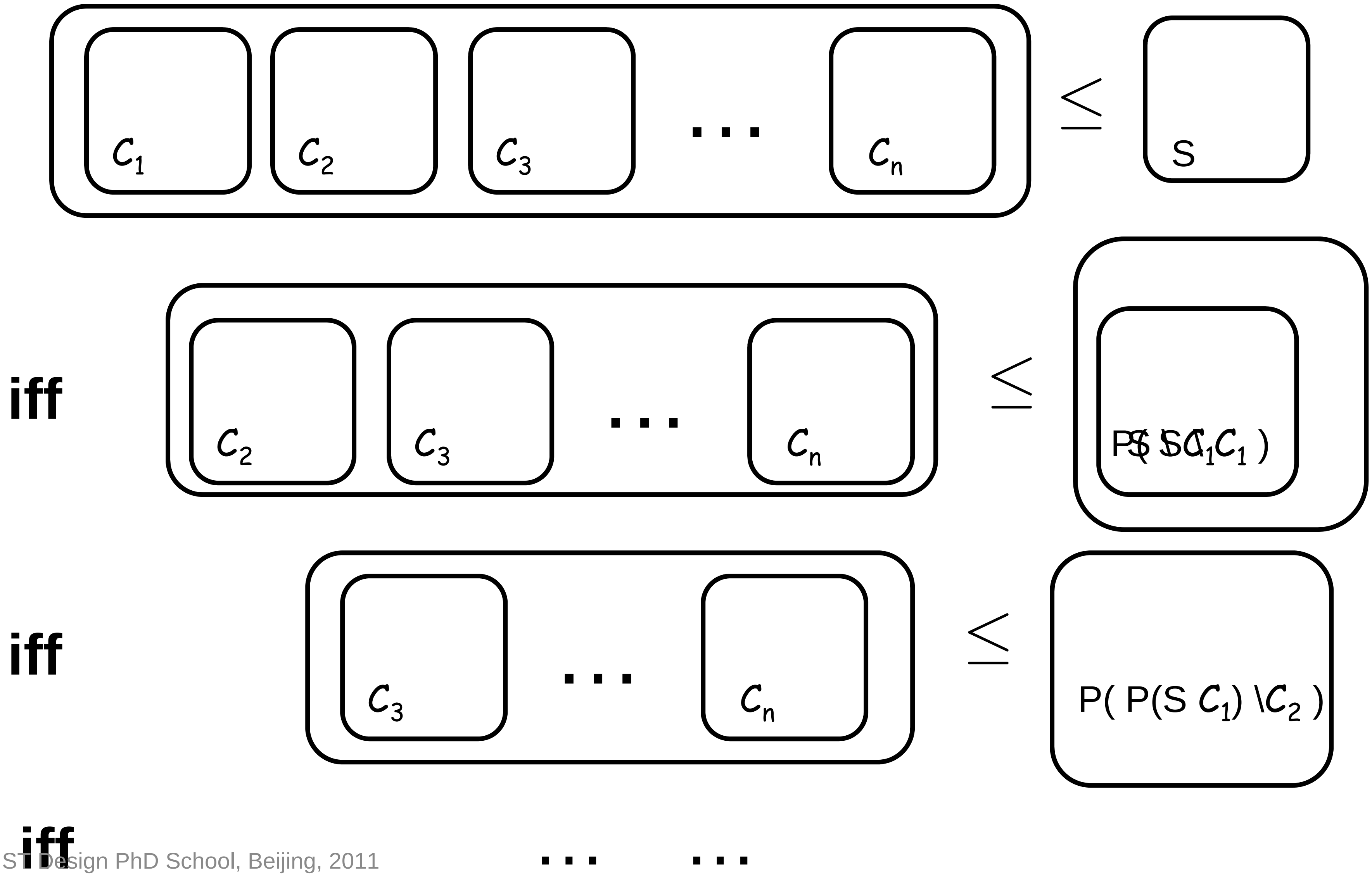




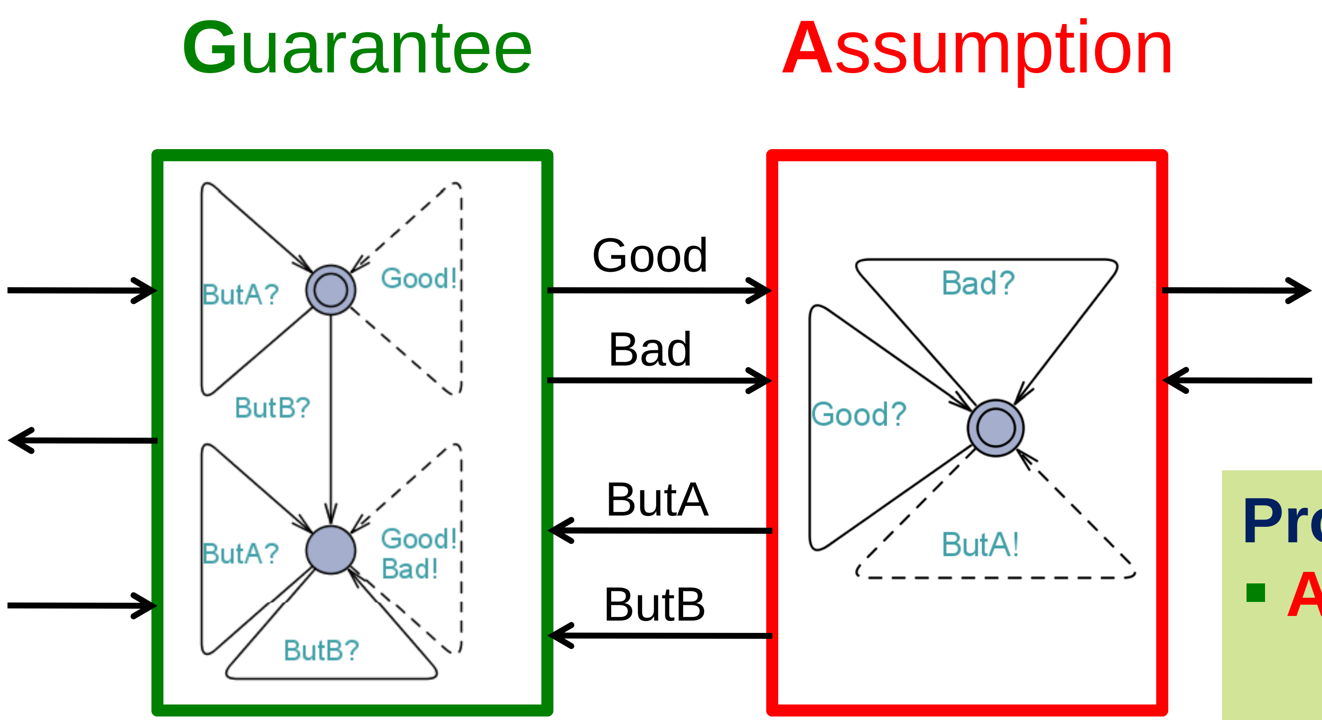
≤

Spec \ Adm

Compositional Refinement Checking



Assume-Guarantee

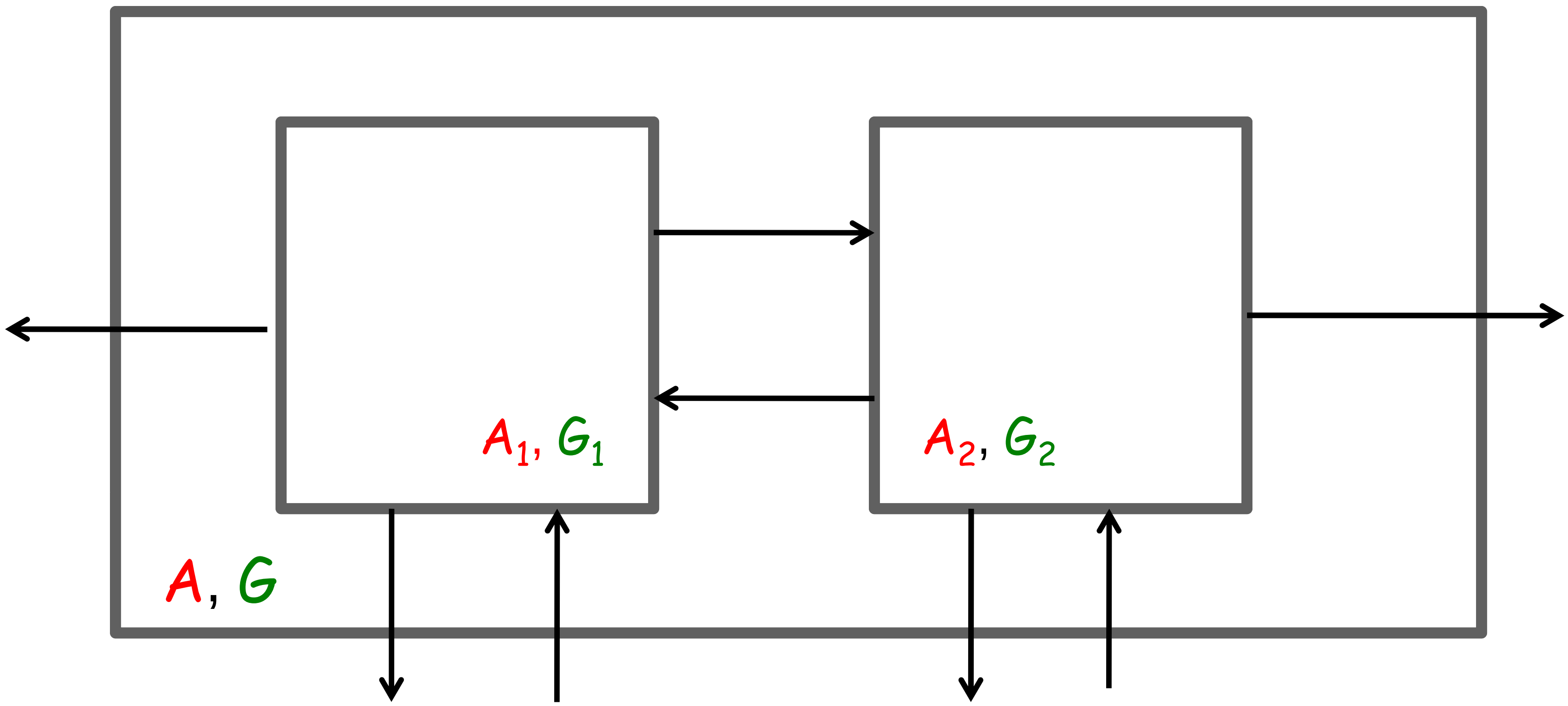


A>>G = (A | G) \ A

Properties

- **A>>G ≥ G**
- **A ≤ A' ⇒ A>>G ≥ A'>> G**
- **G ≤ G' ⇒ A>>G ≤ A>>G'**

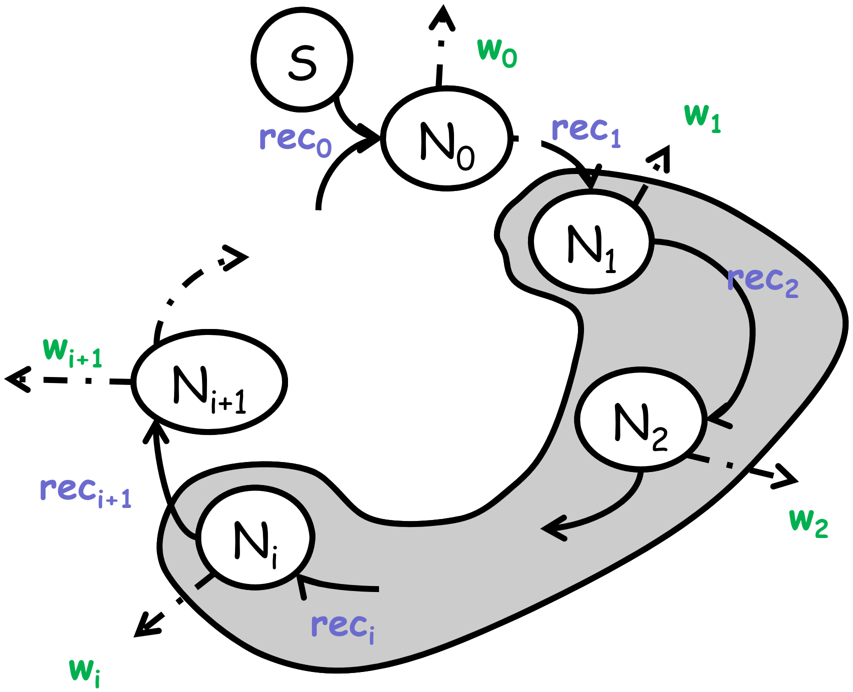
Assume-Guarantee Reasoning

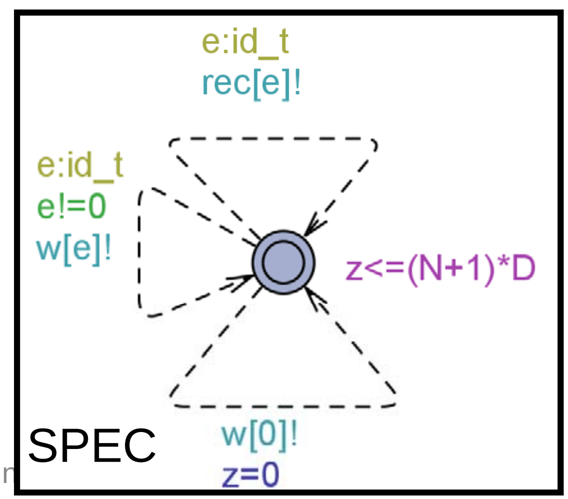


Proof Rule:

$A>>G \geq (A_1>>G_1 \mid A_2>>G_2)$

Milner's Scheduler Compositionaly





Find SS_i and verify:

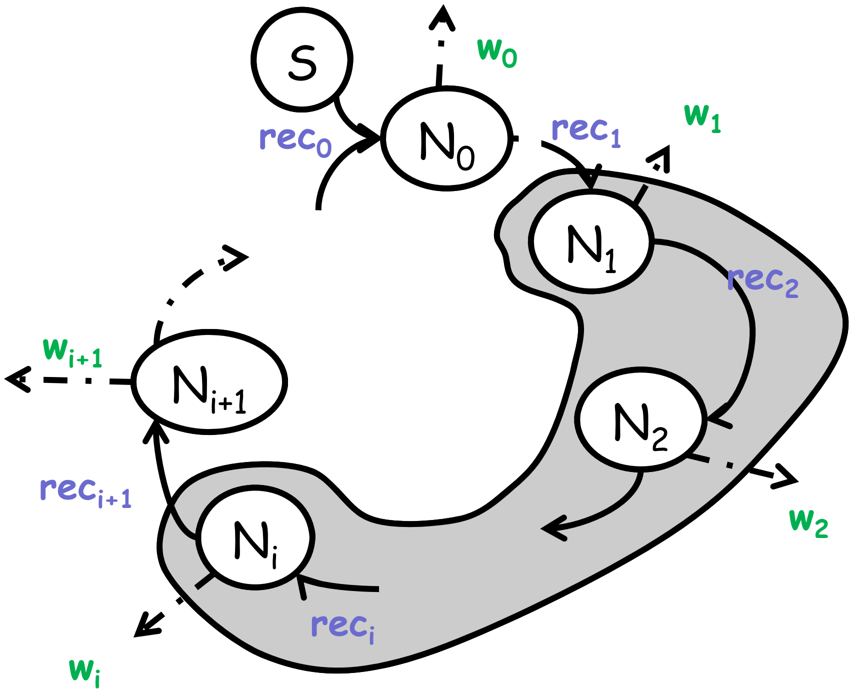
1. $N_1 \leq SS_1$
2. $SS_1 \mid N_2 \leq SS_2$
3. $SS_2 \mid N_3 \leq SS_3$

… …

n. $SS_{n-1} \mid N_n \leq SS_n$

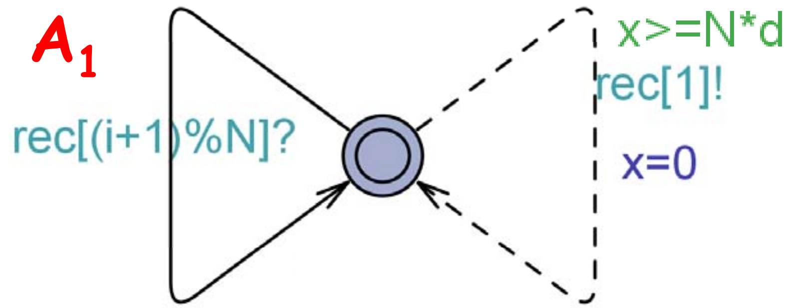
n+1. $SS_n \mid N_0 \leq$ SPEC

Milner's Scheduler Compositionaly

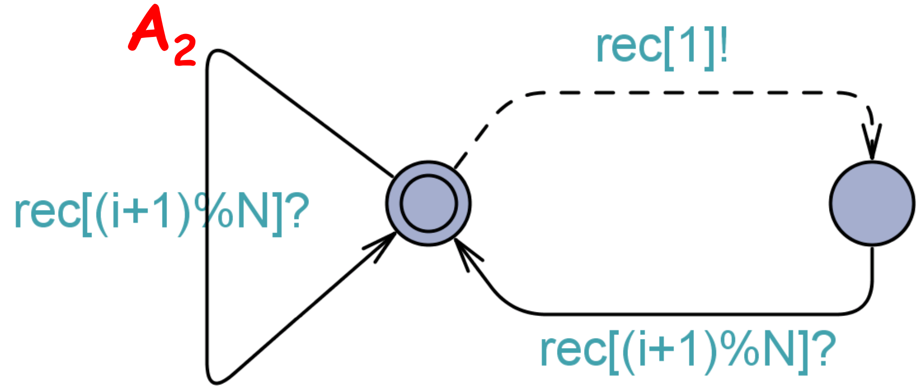


Find SS_i ......

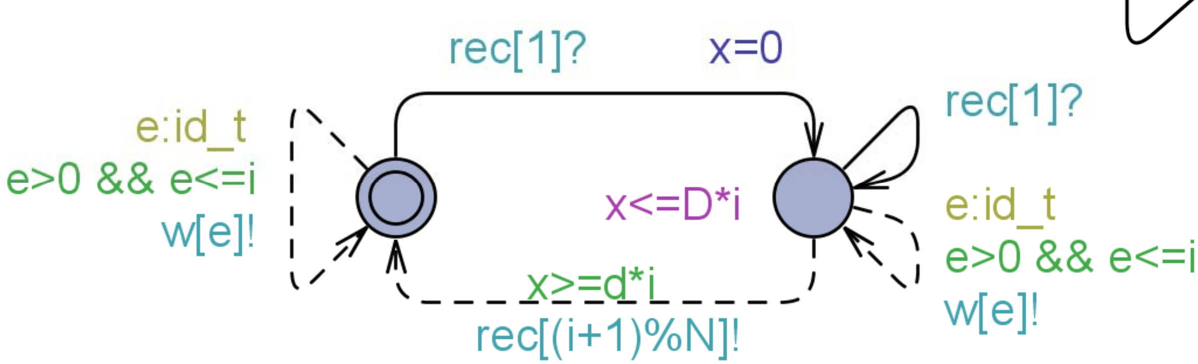
A_1



A_2

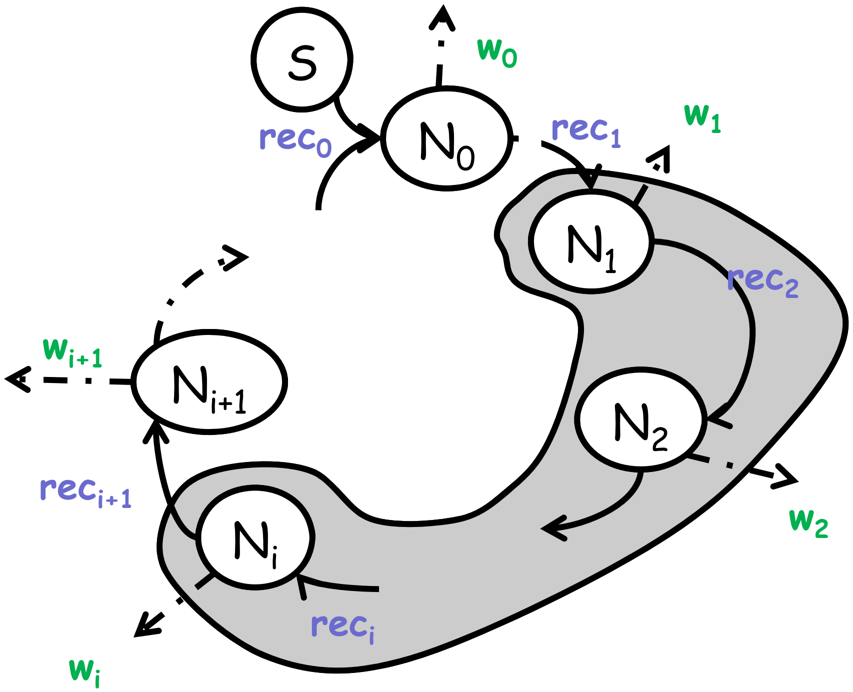


G

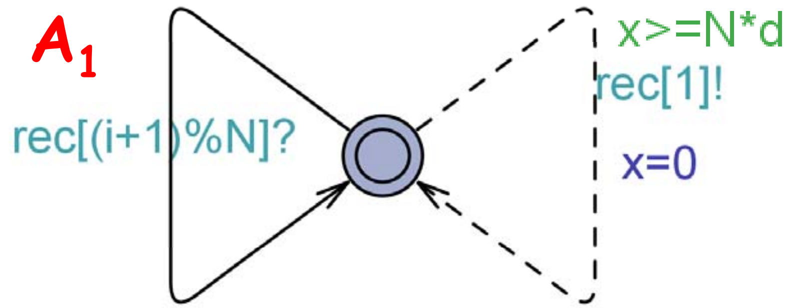


Milner's Scheduler Compositionaly

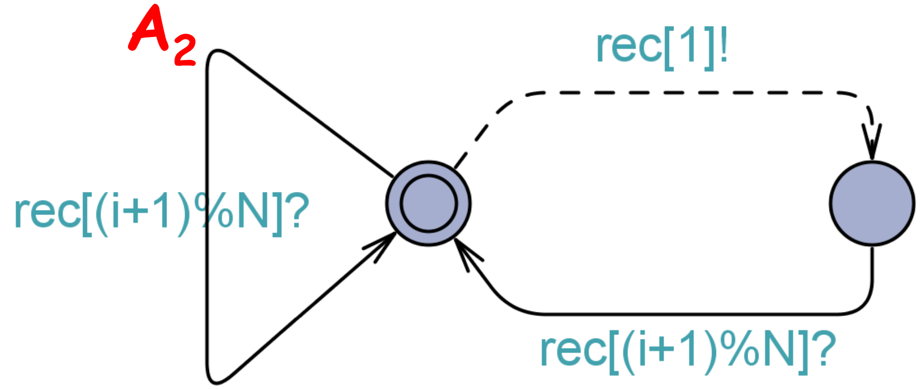
Take $SS_i = (A_1 \,\&\, A_2) \gg G$



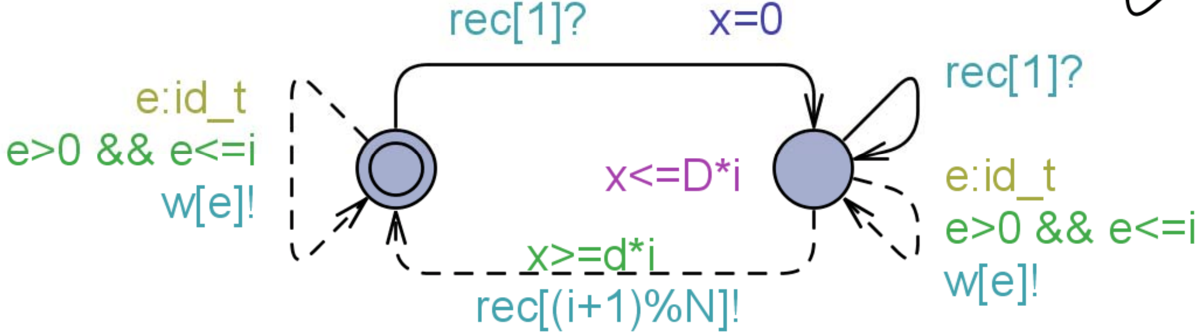
A_1



A_2

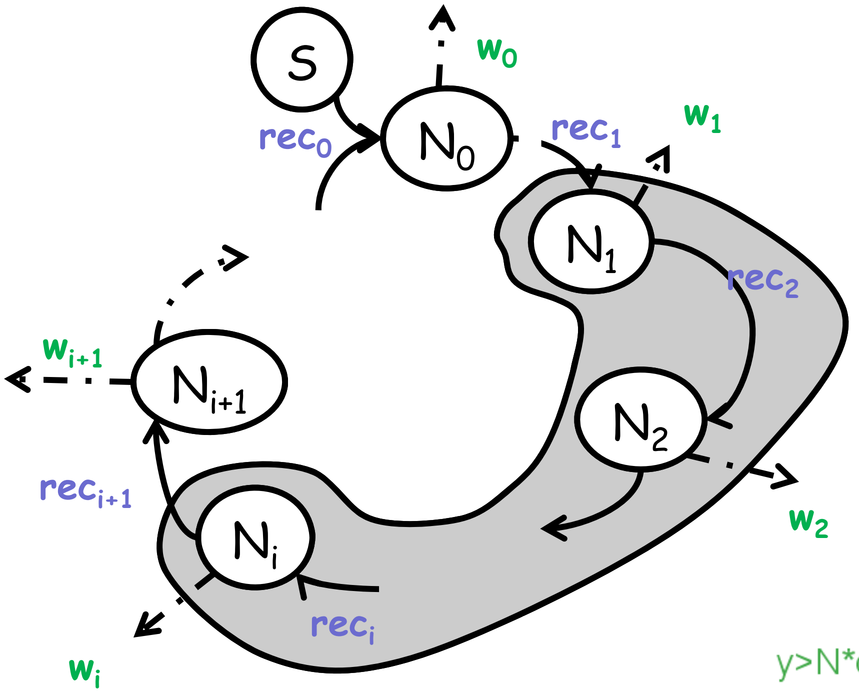


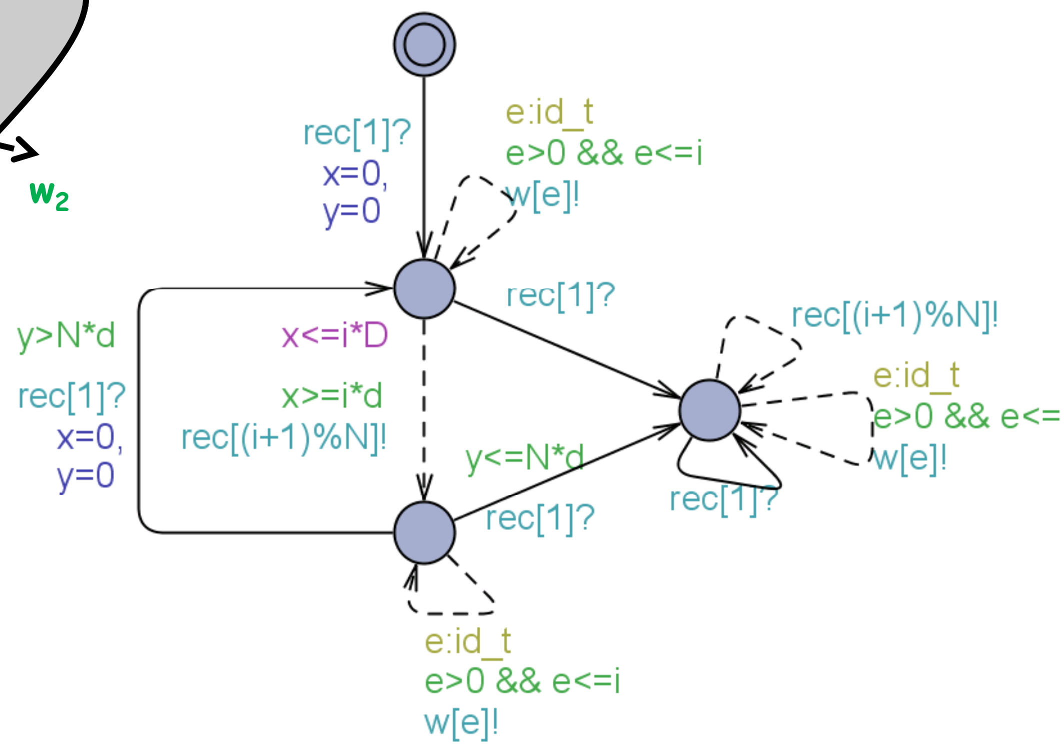
G



Milner's Scheduler Compositionaly

Take SS_i = (A_1 & A_2)>>G





Experiments

	$d = 29$	20	10	9	8	6	4
$n = 5$	0.080	0.097	0.191	*0.169*	*0.172*	*0.151*	*0.205*
monolithic	0.034	0.034	0.073	1.191	1.189	64.933	> 600
$n = 6$	0.102	0.133	0.231	*0.228*	*0.238*	*0.238*	*0.294*
monolithic	0.040	0.043	0.095	6.786	6.791	> 600	> 600
$n = 8$	0.225	0.349	0.516	**0.515**	*0.540*	*0.600*	*0.582*
monolithic	0.076	0.076	0.230	88.542	88.642	> 600	> 600
$n = 12$	0.830	1.414	1.802	**1.895**	**1.831**	*2.079*	*2.181*
monolithic	0.220	0.223	0.843	> 600	> 600	> 600	> 600
$n = 20$	4.990	9.739	12.377	**11.923**	**12.041**	**12.438**	*12.764*
monolithic	1.038	1.030	4.523	> 600	> 600	> 600	> 600
$n = 30$	22.053	45.709	55.728	**55.345**	**55.112**	**54.702**	*56.164*
monolithic	3.791	3.778	17.652	> 600	> 600	> 600	om

Conclusion & Future Work

- Complete specification theory based on Timed I/O Automata
- Analysis: refinement, consistency, compatibility
- Operations: conjunction, parallel composition, quotienting

- Implemented in the tool ECDAR using the engine of UPPAAL Tiga.

- Non-determinism ?
- Unobservable actions ?

Timed Games
W Partial Observability

- Applications :
  - Milners Scheduler
  - Leader Election Protocol
  - Fischers Protocol

- USE IT !!!

References

- HSCC'10
  Timed I/O Automata: A Complete Specification Theory for Real-time Systems
- FMCO'09
  Methodologies for Specification of Real-Time Systems Using Timed I/O Automata
- ATVA'10
  Ecdar: An Environment for Compositional Design and Analysis of Real Time Systems
- WADT'10
  An Interface Theory for Timed Systems

- www.cs.aau.dk/~adavid/ecdar
- www.uppaal.com