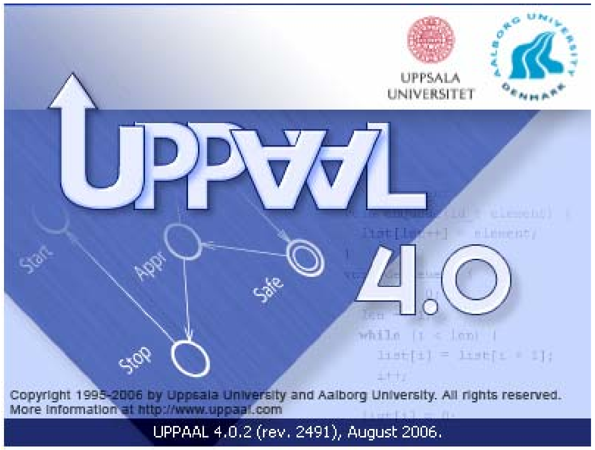


Verification & Modelling

Options & Patterns

Kim G Larsen

AALBORG UNIVERSITY DENMARK

BRICS
Basic Research in Computer Science

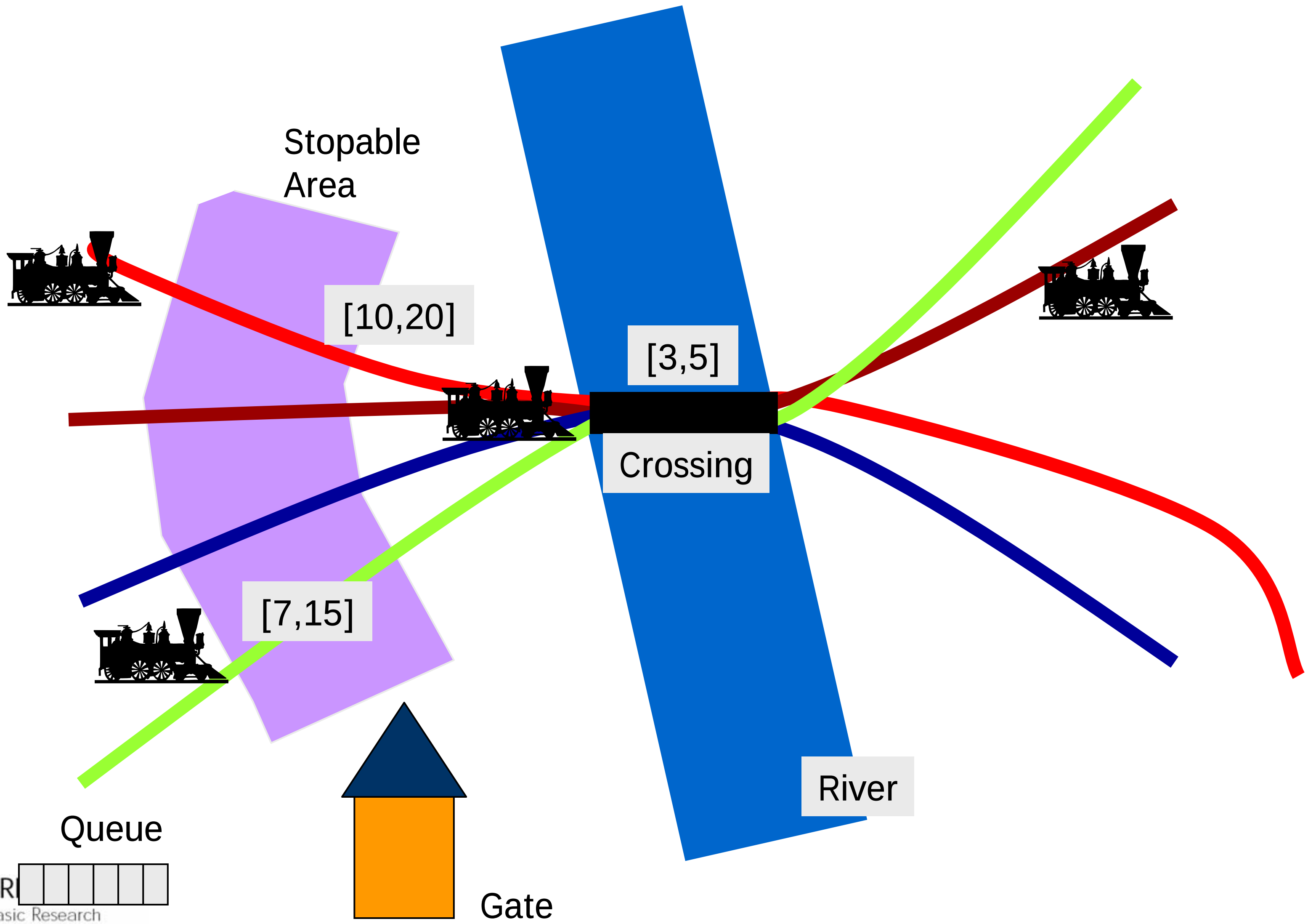
CISS
CENTER FOR INDLEJREDE SOFTWARE SYSTEMER

Outline

- UPPAAL
  - Modelling Formalism
  - Specification Formalism
- UPPAAL Verification Engine
- Verification Options & Modelling Patterns
- Real-Time Planning & Scheduling

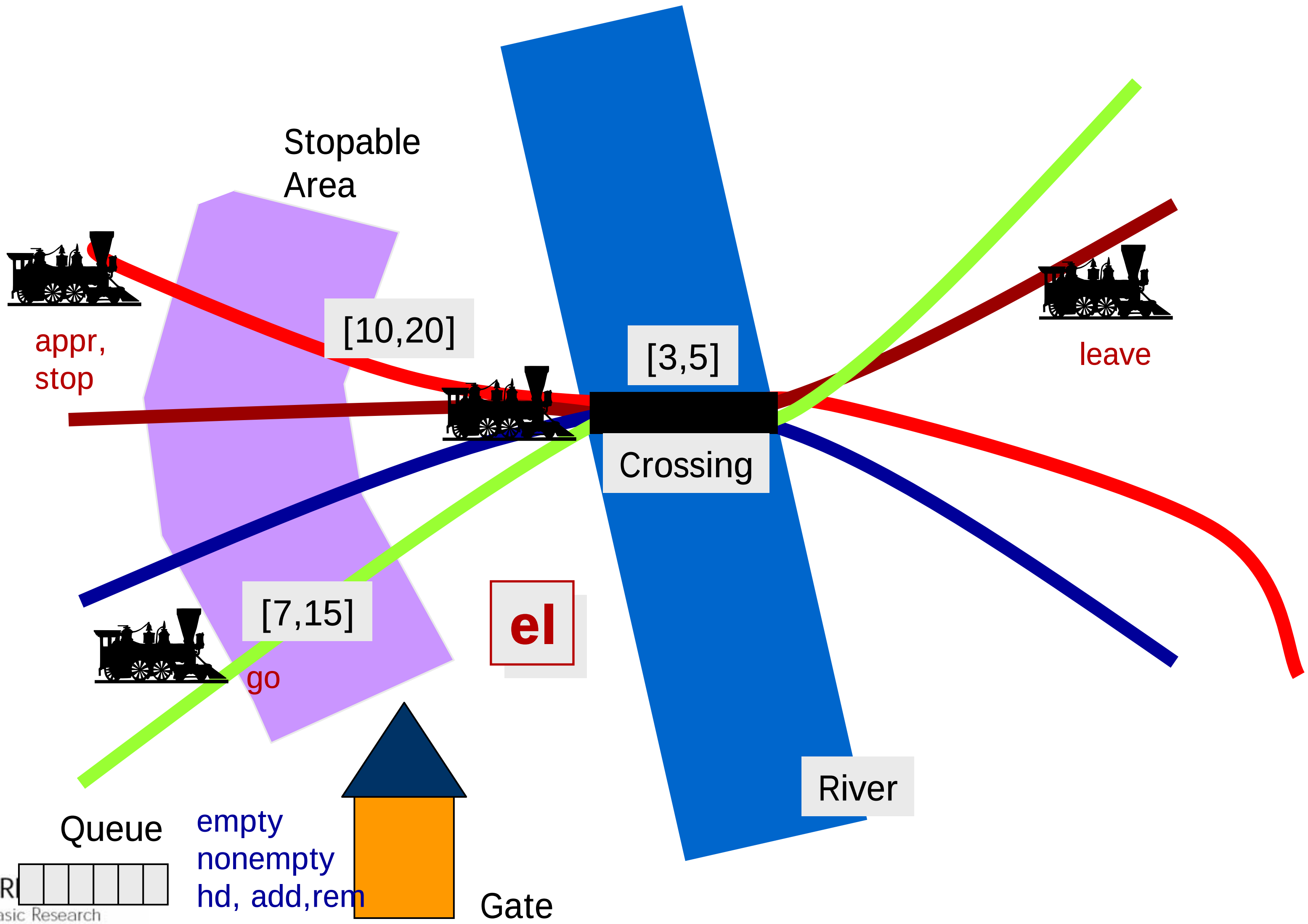
Train Crossing

Train Crossing



Train Crossing

Communication via channels and shared variable.



Specification Language

Logical Specifications

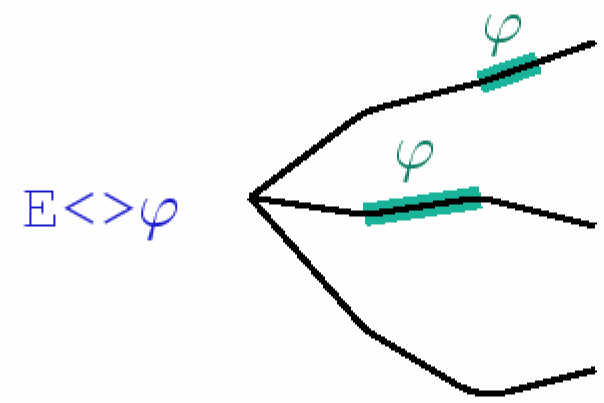
- Validation Properties
  - Possibly: E<> *P*
- Safety Properties
  - Invariant: A[] *P*
  - Pos. Inv.: E[] *P*
- Liveness Properties
  - Eventually: A<> *P*
  - Leadsto: $P \rightarrow Q$
- Bounded Liveness
  - Leads to within: $P \rightarrow_{\leq t} Q$

The expressions *P* and *Q* must be type safe, side effect free, and evaluate to a boolean.

Only references to integer variables, constants, clocks, and locations are allowed (and arrays of these).

Logical Specifications

- Validation Properties
  - Possibly: E<> *P*



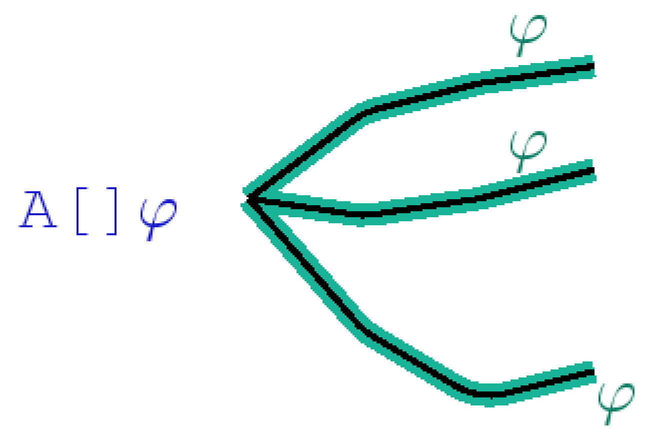
- Safety Properties
  - Invariant: A[] *P*
  - Pos. Inv.: E[] *P*

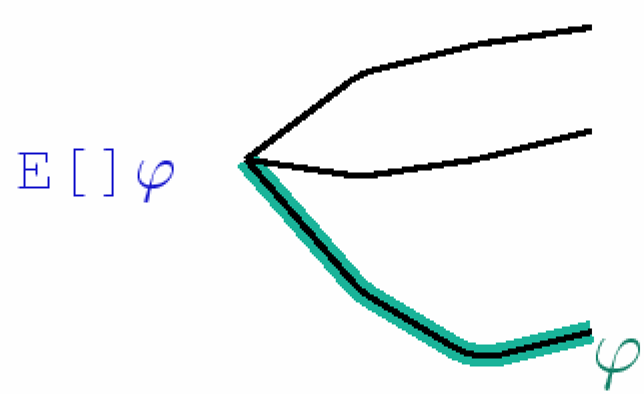
- Liveness Properties
  - Eventually: A<> *P*
  - Leadsto: P → *Q*

- Bounded Liveness
  - Leads to within: $P \rightarrow_{\leq t} Q$

Logical Specifications

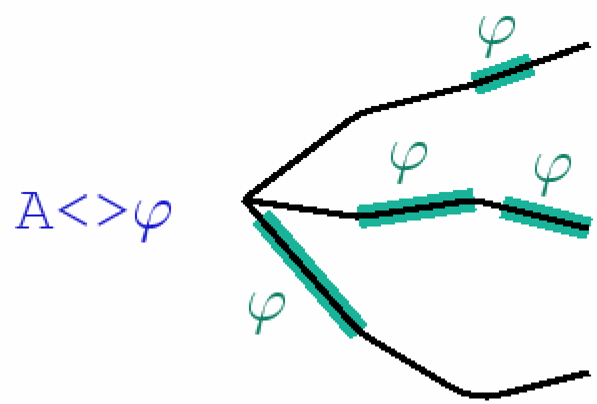
- Validation Properties
  - Possibly: E<> P
- Safety Properties
  - Invariant: A[] P
  - Pos. Inv.: E[] P
- Liveness Properties
  - Eventually: A<> P
  - Leadsto: $P \rightarrow Q$
- Bounded Liveness
  - Leads to within: $P \rightarrow_{\leq t} Q$

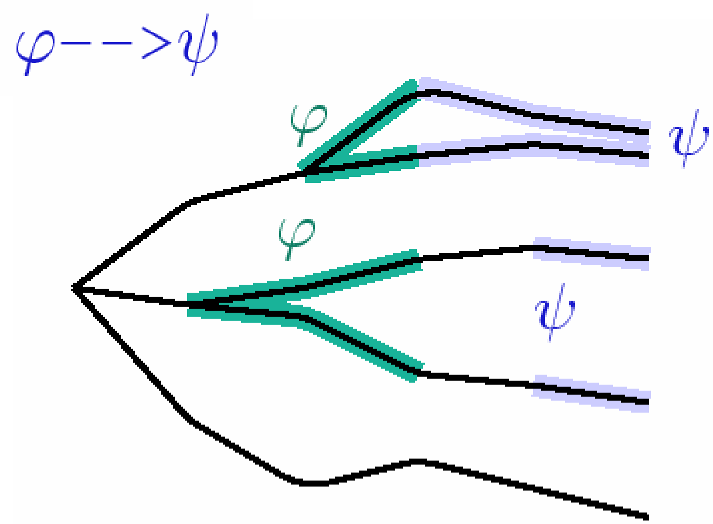




Logical Specifications

- Validation Properties
  - Possibly: E<> *P*
- Safety Properties
  - Invariant: A[] *P*
  - Pos. Inv.: E[] *P*
- Liveness Properties
  - Eventually: A<> *P*
  - Leadsto: P → *Q*
- Bounded Liveness
  - Leads to within: *P* $\rightarrow_{\leq t}$ *Q*

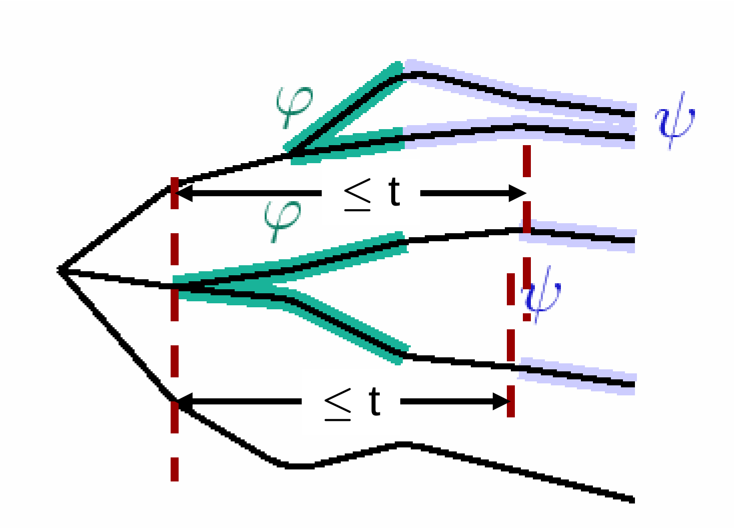




BRICS
Basic Research
in Computer Science

Logical Specifications

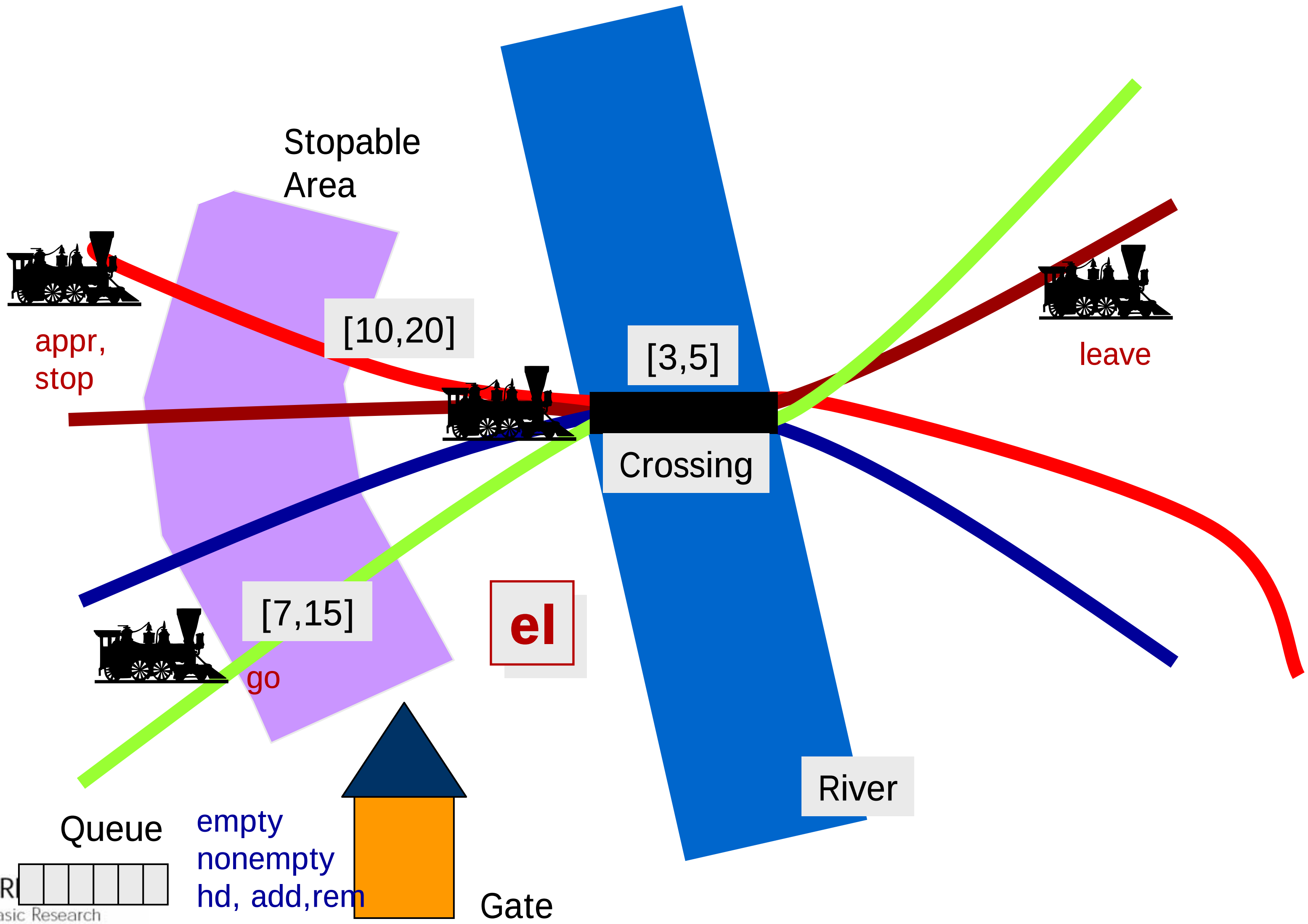
- Validation Properties
  - Possibly: E<> *P*
- Safety Properties
  - Invariant: A[] *P*
  - Pos. Inv.: E[] *P*
- Liveness Properties
  - Eventually: A<> *P*
  - Leadsto: P → *Q*
- Bounded Liveness
  - Leads to within: $P \rightarrow_{\leq t} Q$



BRICS Basic Research in Computer Science

Train Crossing

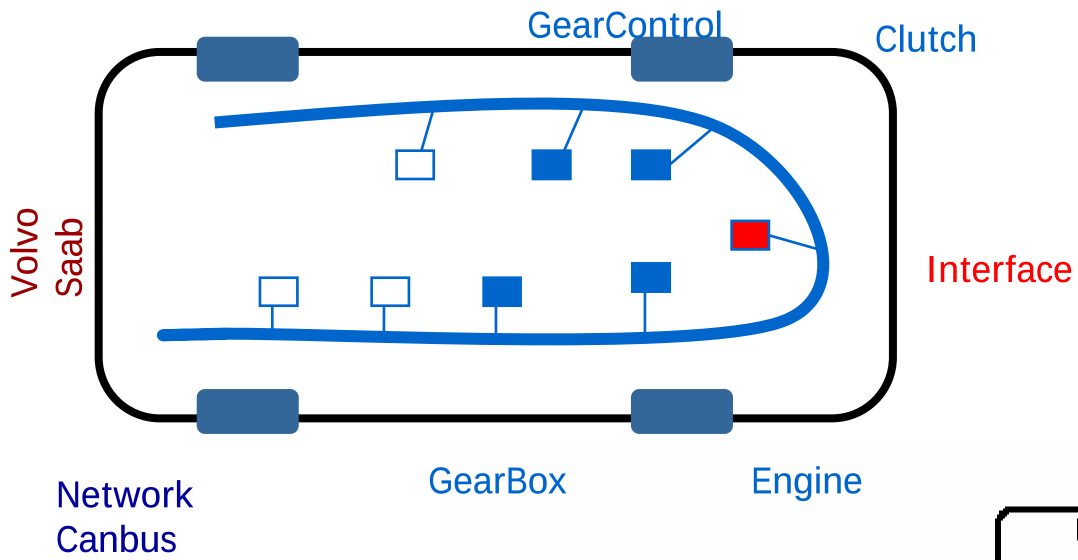
Communication via channels and shared variable.



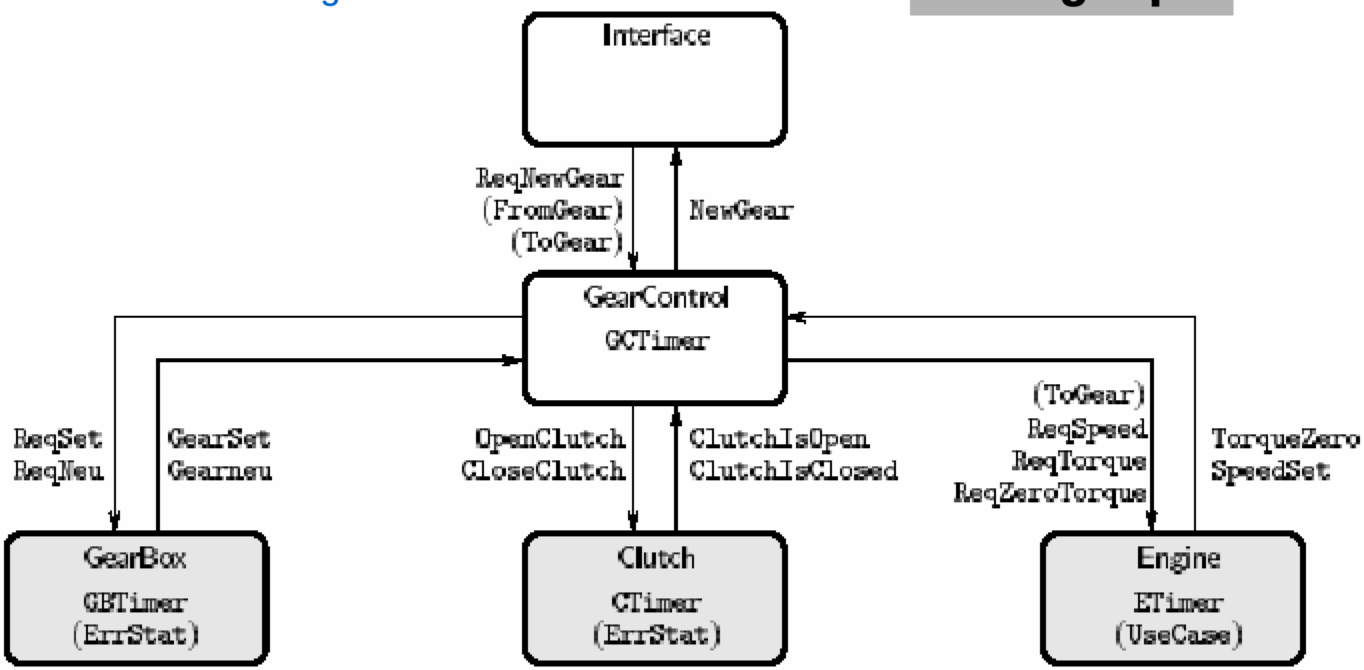
Gear Controller

with MECEL AB

Lindahl, Pettersson, Yi 1998



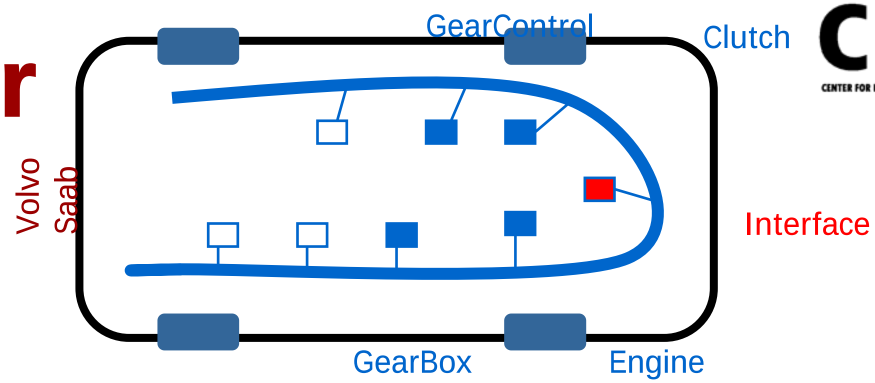
Flowgraph



BRICS
Basic Research
in Computer Science

Gear Controller

with MECEL AB



Requirements

$$\text{GearControl@Initiate} \leadsto_{\leq 1500} (\ (\ \mathtt{ErrStat} = 0\ )\ \Rightarrow\ \text{GearControl@GearChanged}\ ) \tag{1}$$

$$\text{GearControl@Initiate} \leadsto_{\leq 1000} (\ (\ \mathtt{ErrStat} = 0\ \wedge\ \mathtt{UseCase} = 0\ )\ \Rightarrow\ \text{GearControl@GearChanged}\ ) \tag{2}$$

$$\text{Clutch@ErrorClose} \leadsto_{\leq 200} \text{GearControl@CCloseError} \tag{3}$$

$$\text{Clutch@ErrorOpen} \leadsto_{\leq 200} \text{GearControl@COpenError} \tag{4}$$

$$\text{GearBox@ErrorIdle} \leadsto_{\leq 350} \text{GearControl@GSetError} \tag{5}$$

$$\text{GearBox@ErrorNeu} \leadsto_{\leq 200} \text{GearControl@GNeuError} \tag{6}$$

$$Inv\ (\ \text{GearControl@CCloseError}\ \Rightarrow\ \text{Clutch@ErrorClose}\ ) \tag{7}$$

$$Inv\ (\ \text{GearControl@COpenError}\ \Rightarrow\ \text{Clutch@ErrorOpen}\ ) \tag{8}$$

$$Inv\ (\ \text{GearControl@GSetError}\ \Rightarrow\ \text{GearBox@ErrorIdle}\ ) \tag{9}$$

$$Inv\ (\ \text{GearControl@GNeuError}\ \Rightarrow\ \text{GearBox@ErrorNeu}\ ) \tag{10}$$

$$Inv\ (\ \text{Engine@ErrorSpeed}\ \Rightarrow\ \mathtt{ErrStat} \neq 0\ ) \tag{11}$$

$$Inv\ (\ \text{Engine@Torque}\ \Rightarrow\ \text{Clutch@Closed}\ ) \tag{12}$$

$$\bigwedge_{i \in \{R,N,1,\ldots,5\}} Poss\ (\ \text{Gear@Gear}_i\ ) \tag{13}$$

$$\bigwedge_{i \in \{R,1,\ldots,5\}} Inv\ (\ (\ \text{GearControl@Gear}\ \wedge\ \text{Gear@Gear}_i\ )\ \Rightarrow\ \text{Engine@Torque}\ ) \tag{14}$$

Case-Studies: Controllers

- Gearbox Controller [TACAS'98]
- Bang & Olufsen Power Controller [RTPS'99,FTRTFT'2k]
- SIDMAR Steel Production Plant [RTCSA'99, DSVV'2k]
- Real-Time RCX Control-Programs [ECRTS'2k]
- Experimental Batch Plant (2000)
- RCX Production Cell (2000)
- Terma, Verification of Memory Management for Radar (2001)
- Scheduling Lacquer Production (2005)
- Memory Arbiter Synthesis and Verification for a Radar Memory Interface Card [NJC'05]

Case Studies: Protocols

- Philips Audio Protocol [HS'95, CAV'95, RTSS'95, CAV'96]
- Collision-Avoidance Protocol [SPIN'95]
- Bounded Retransmission Protocol [TACAS'97]
- Bang & Olufsen Audio/Video Protocol [RTSS'97]
- TDMA Protocol [PRFTS'97]
- Lip-Synchronization Protocol [FMICS'97]
- Multimedia Streams [DSVIS'98]
- ATM ABR Protocol [CAV'99]
- ABB Fieldbus Protocol [ECRTS'2k]
- IEEE 1394 Firewire Root Contention (2000)
- Distributed Agreement Protocol [Formats05]
- Leader Election for Mobile Ad Hoc Networks [Charme05]

BRICS Basic Research in Computer Science

The UPPAAL Verification Engine

Fm

AALBORG UNIVERSITY DENMARK

BRICS
Basic Research in Computer Science

CISS
CENTER FOR INDLEJREDE SOFTWARE SYSTEMER

Overview

- Zones and DBMs
- Minimal Constraint Form
- Clock Difference Diagrams

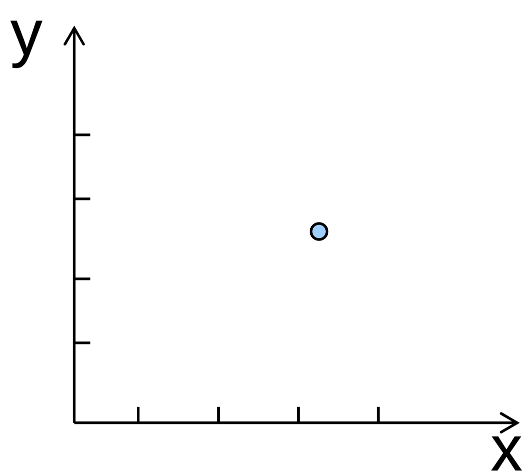
- Distributed UPPAAL [CAV2000, STTT2004]
- Unification & Sharing [FTRTFT2002, SPIN2003]
- Acceleration [FORMATS2002]
- Static Guard Analysis [TACAS2003,TACAS2004]
- Storage-Strategies [CAV2003]

Zones

From infinite to finite

State

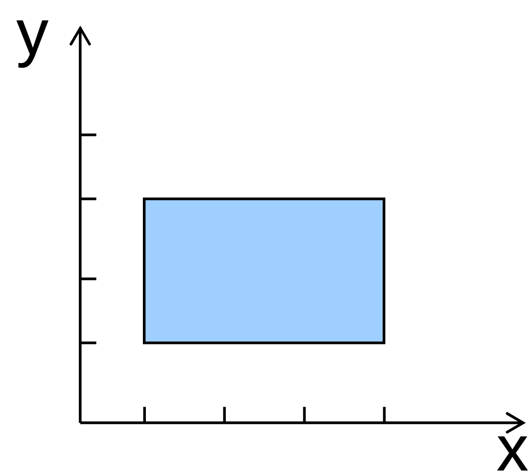
(n, x=3.2, y=2.5)



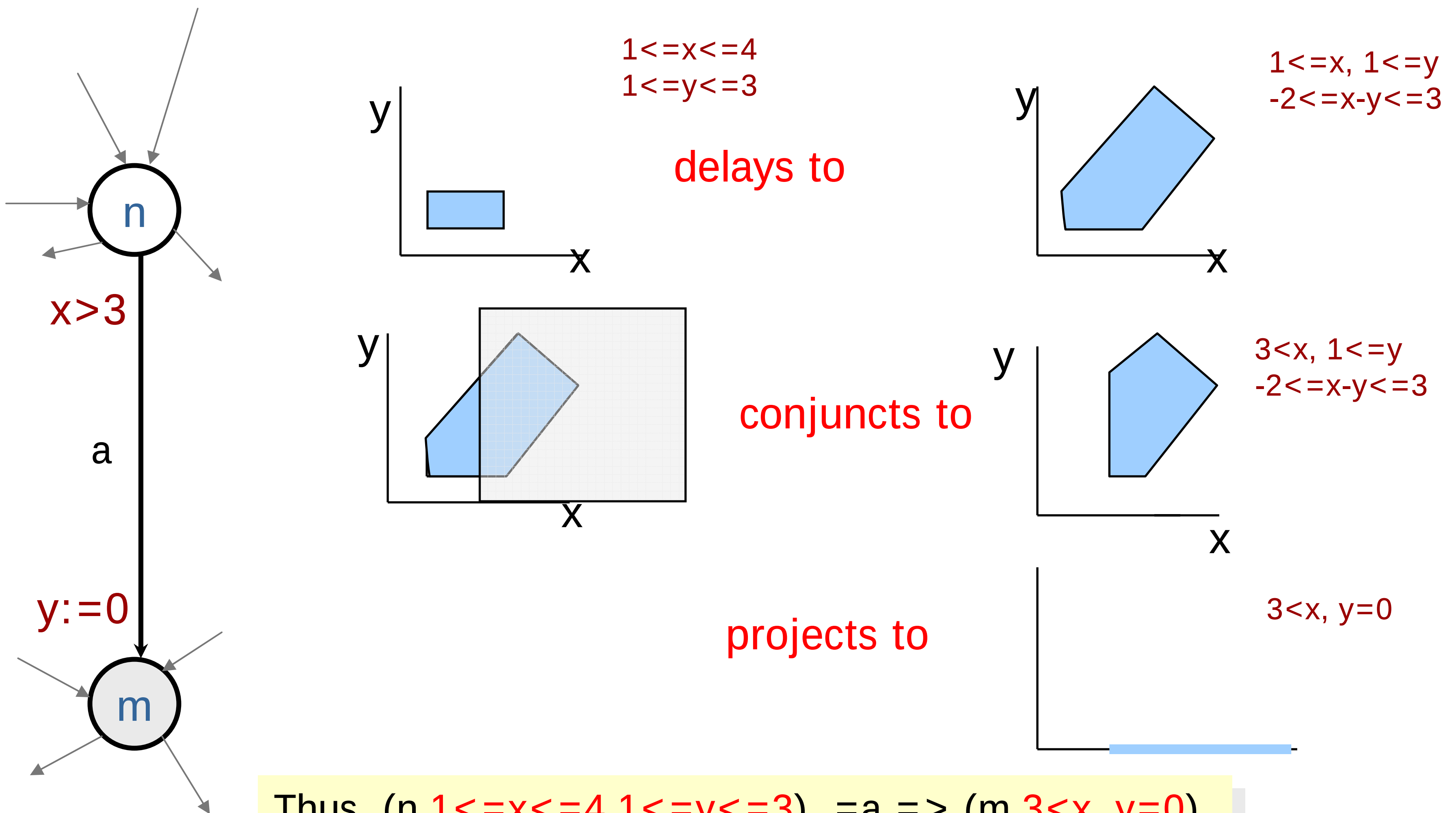
Symbolic state (set)

(n, $1 \le x \le 4$, $1 \le y \le 3$)

Zone:
conjunction of
x-y<=n, x<=>n



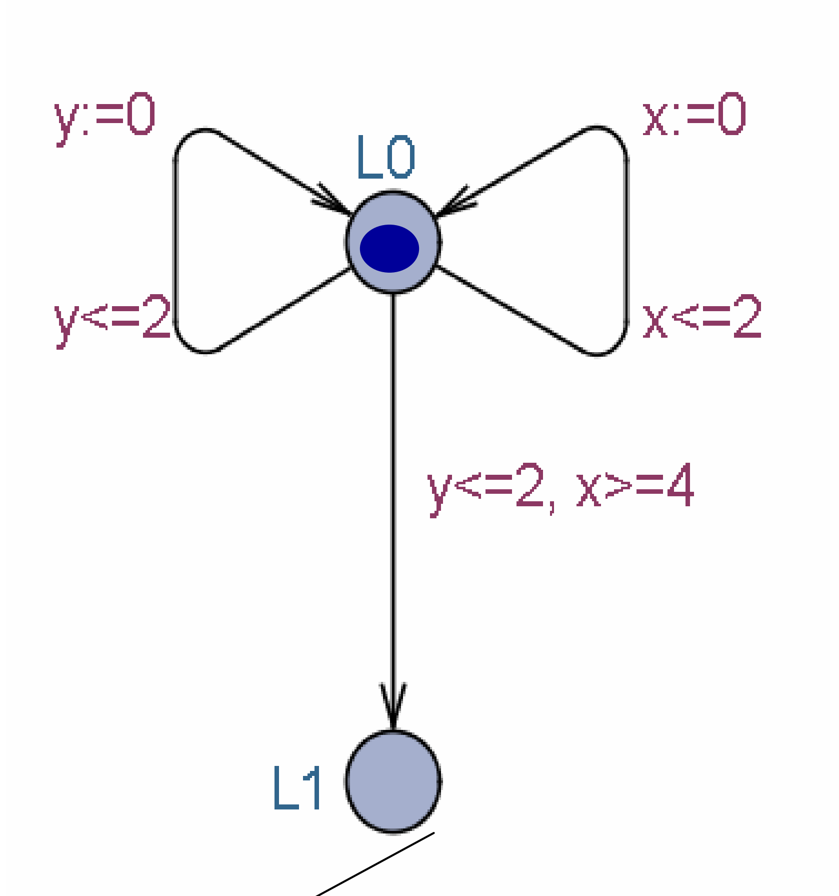
Symbolic Transitions



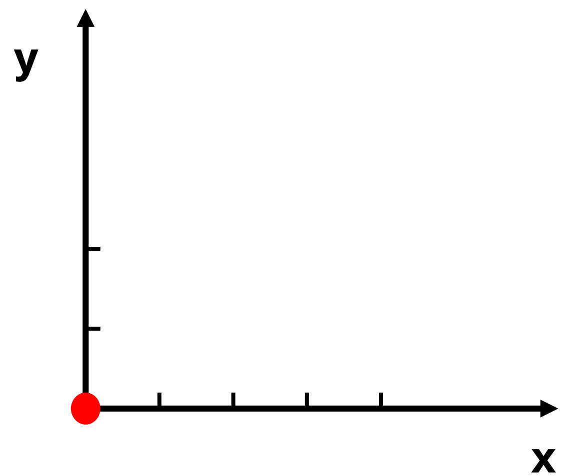
Thus (n,1<=x<=4,1<=y<=3) =a => (m,3<x, y=0)

BRICS Basic Research in Computer Science

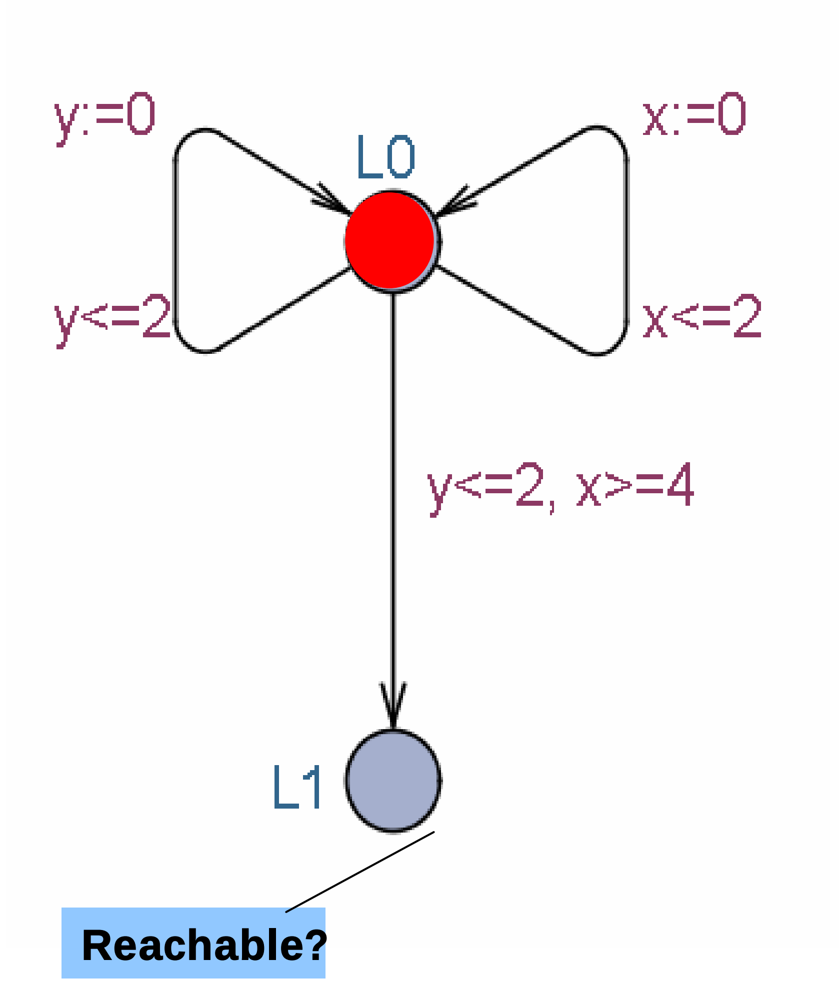
Symbolic Exploration

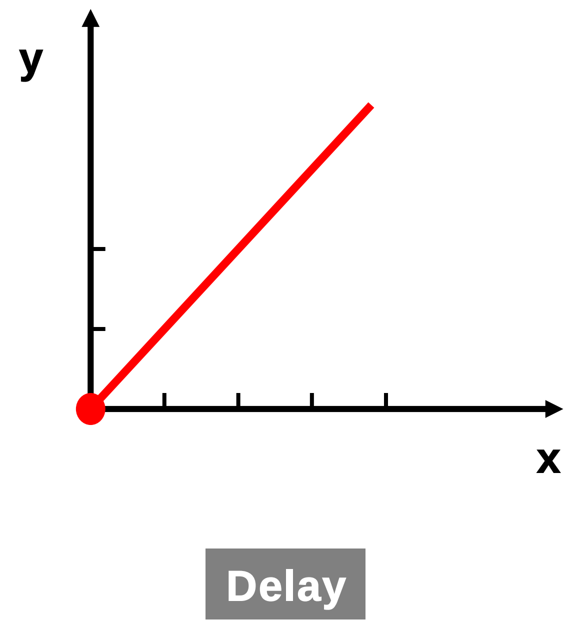


Reachable?

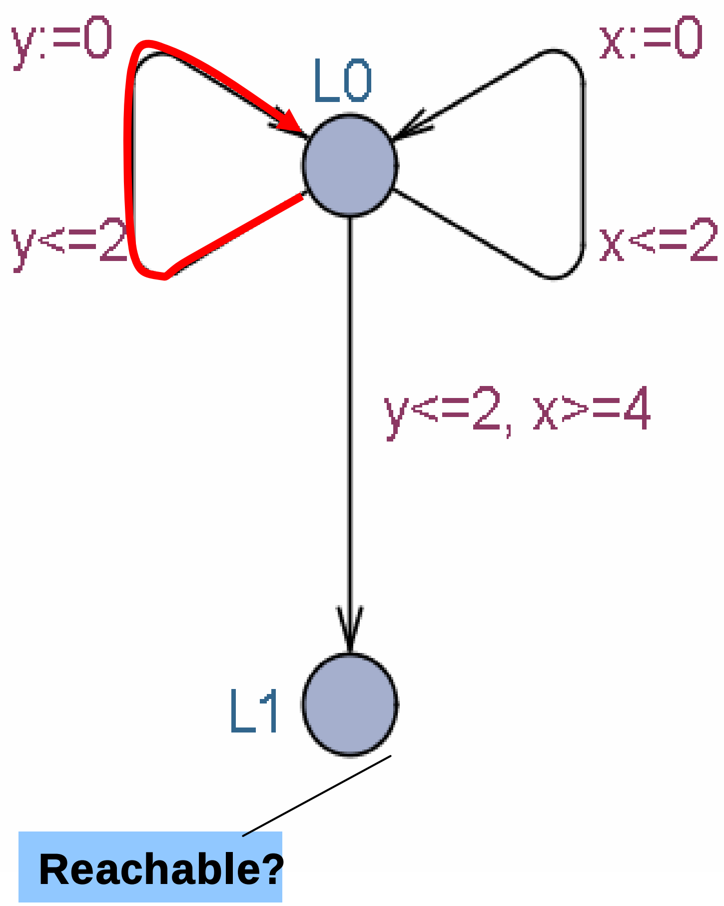


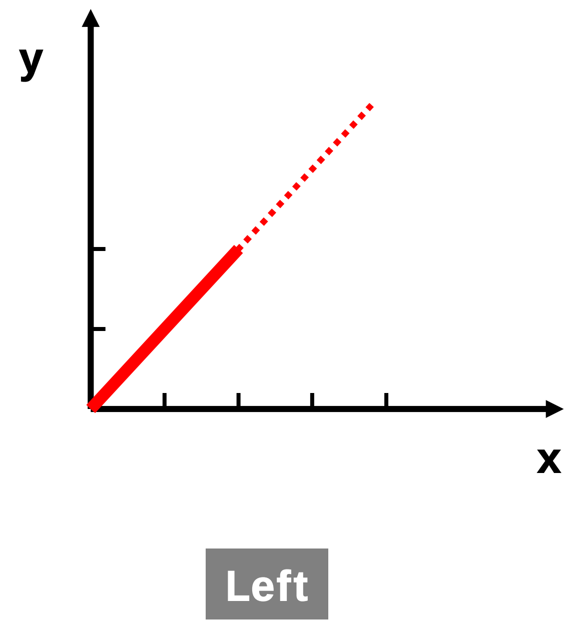
Symbolic Exploration



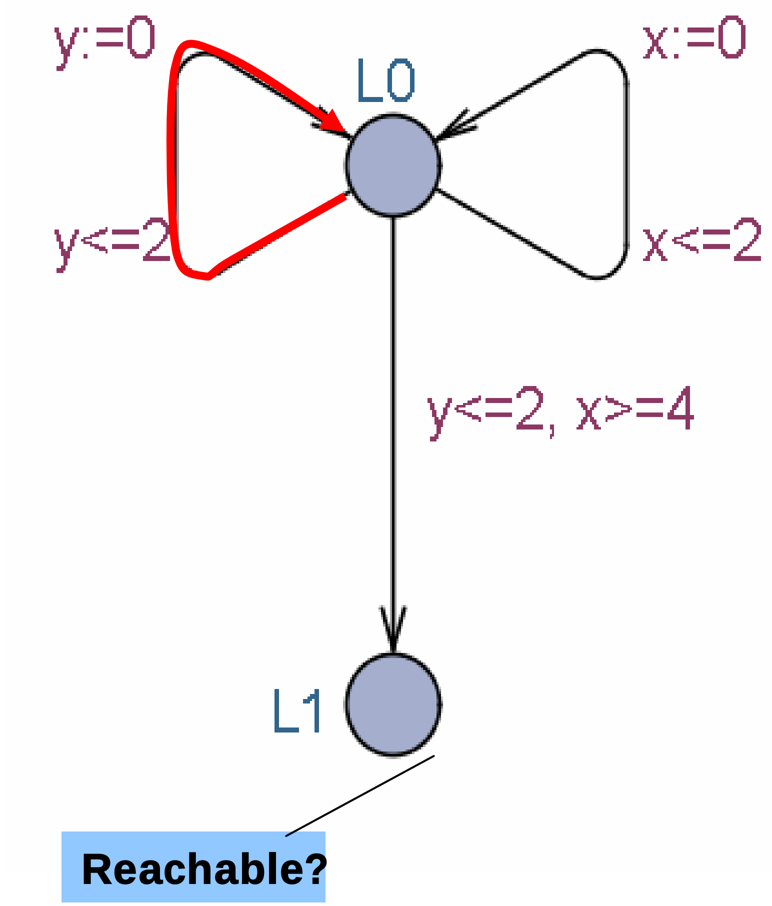


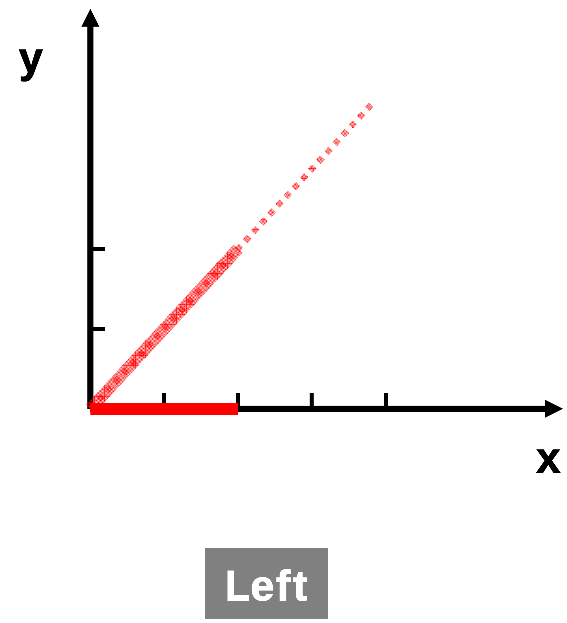
Symbolic Exploration





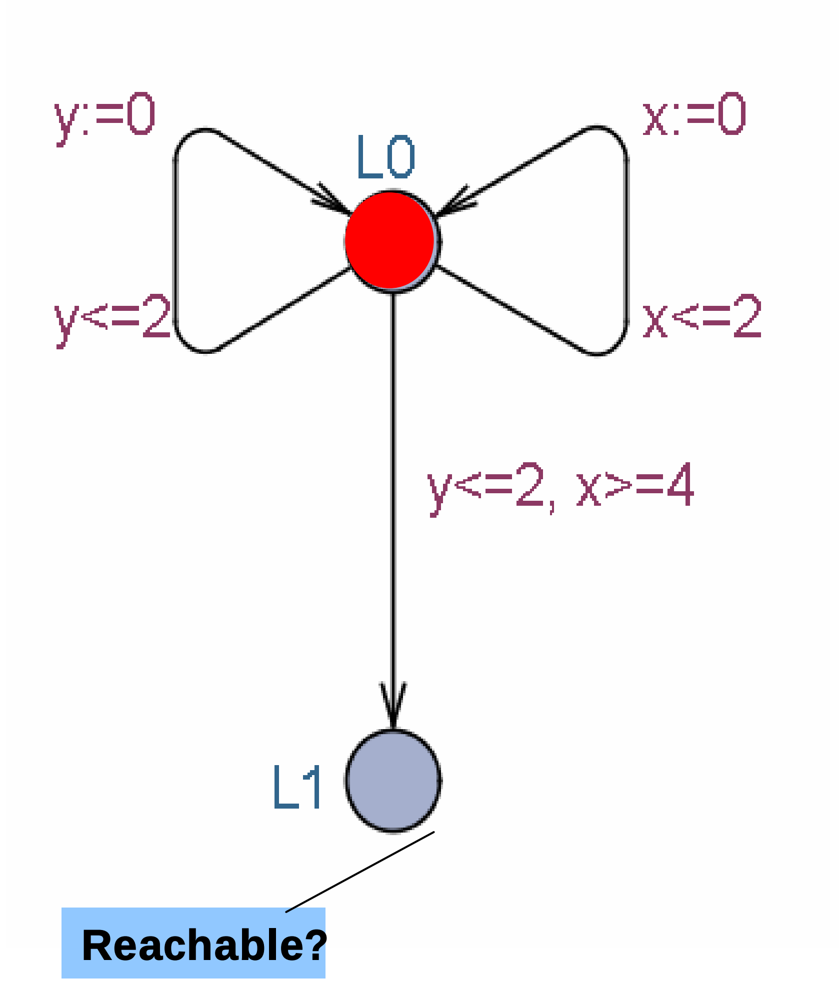
Symbolic Exploration

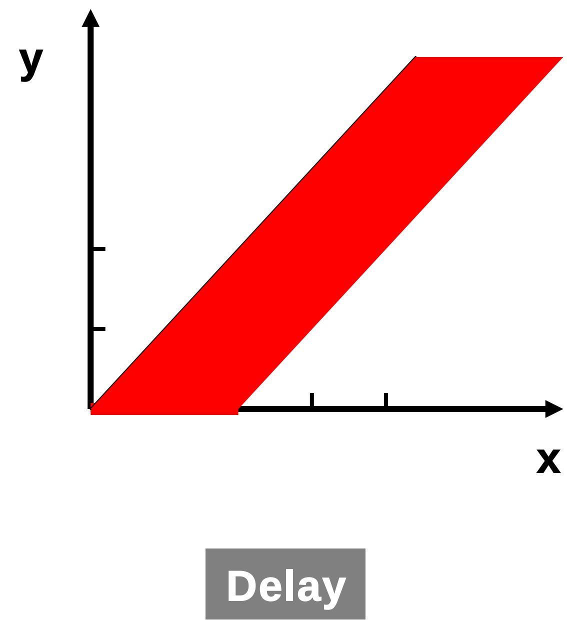




BRICS
Basic Research
in Computer Science

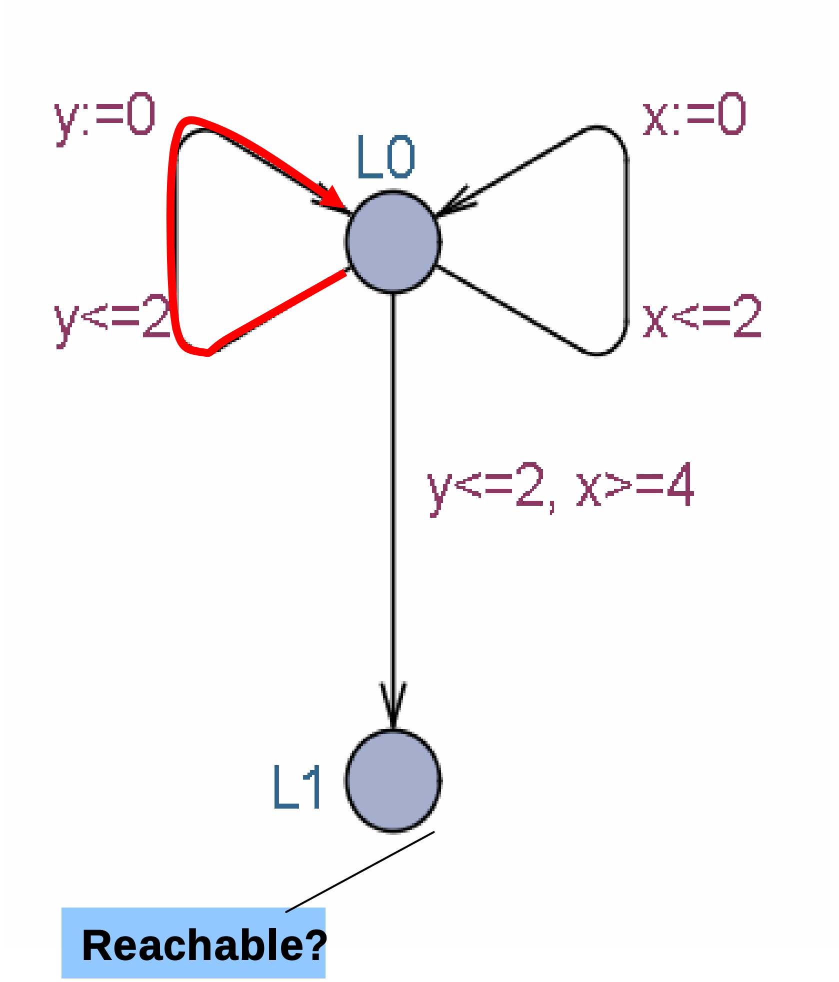
Symbolic Exploration

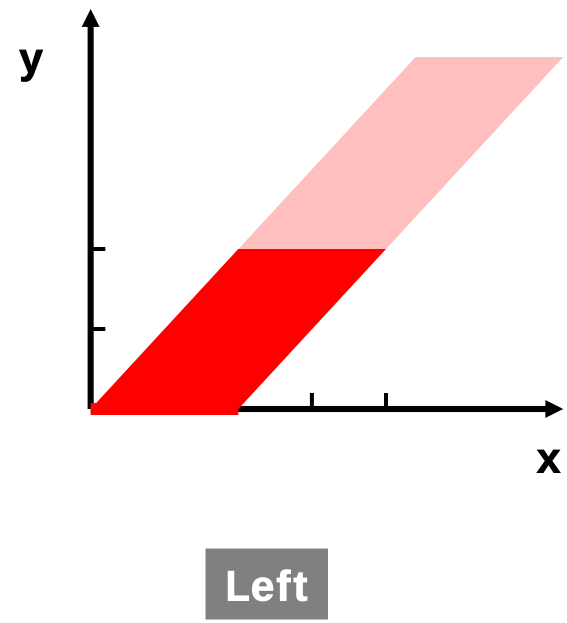




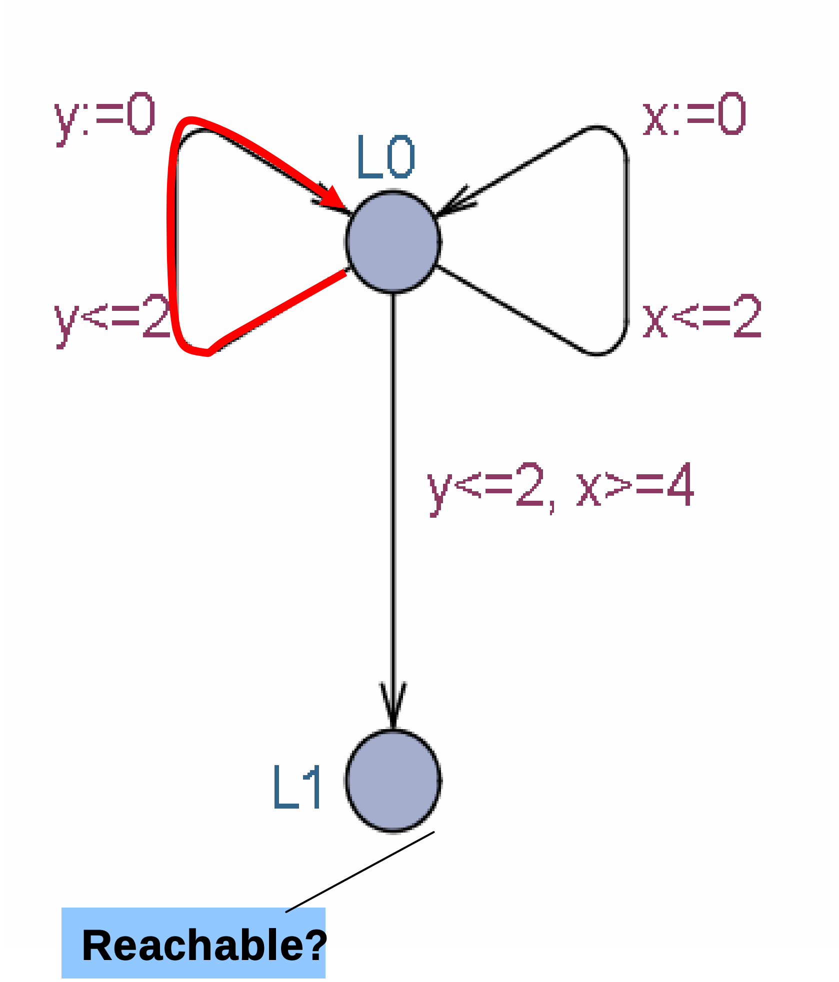
BRICS
Basic Research
in Computer Science

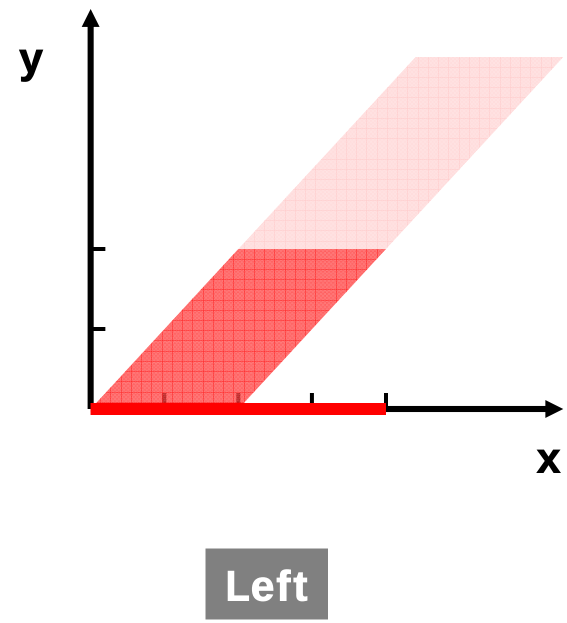
Symbolic Exploration



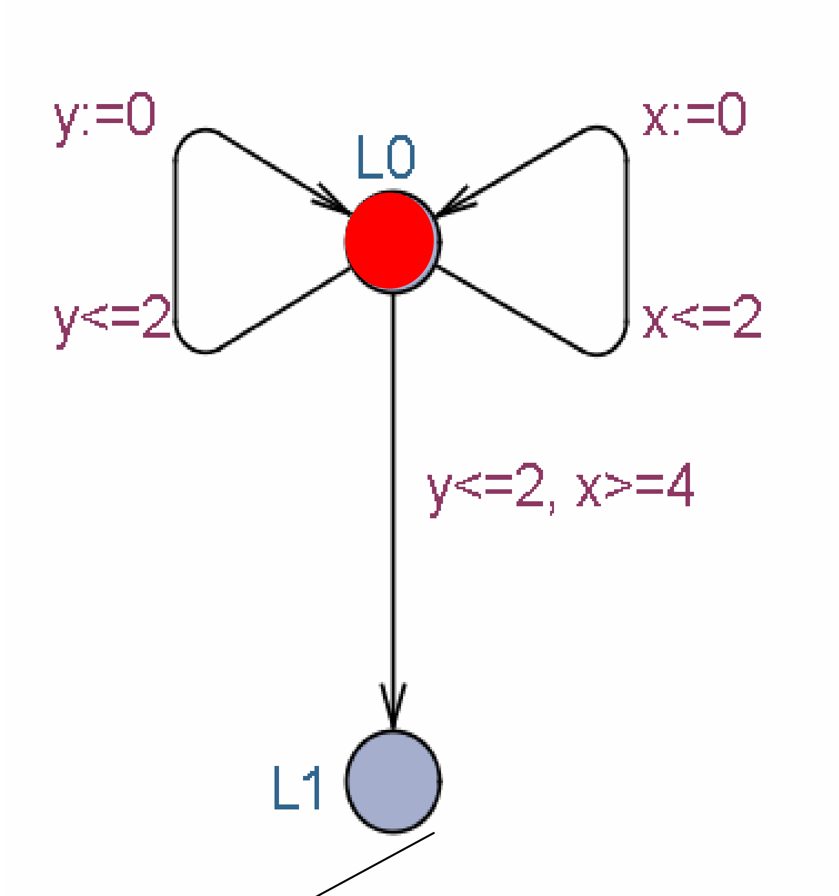


Symbolic Exploration

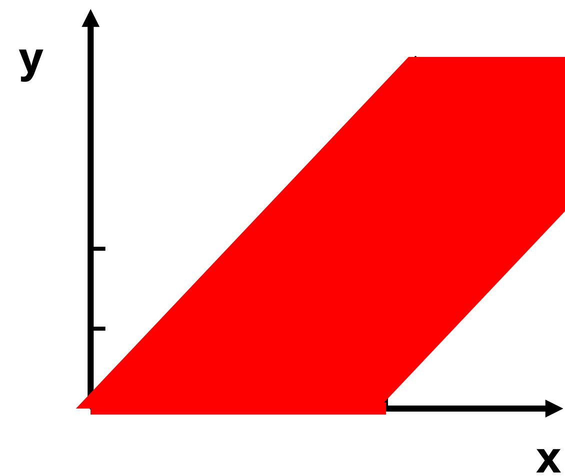




Symbolic Exploration

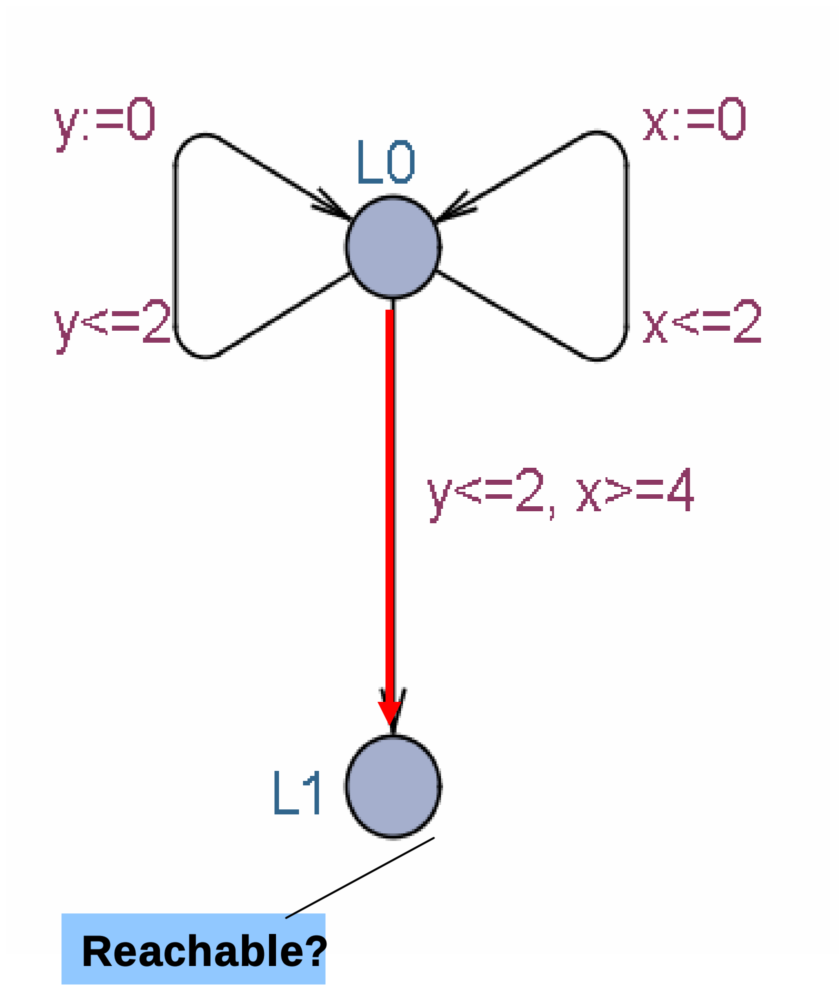


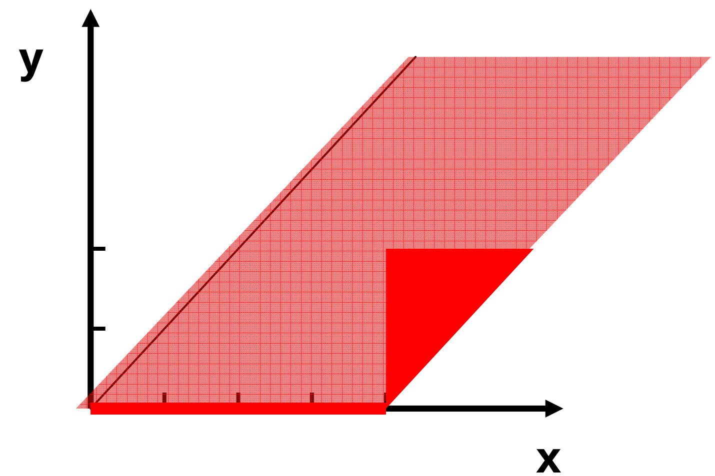
Reachable?



Delay

Symbolic Exploration

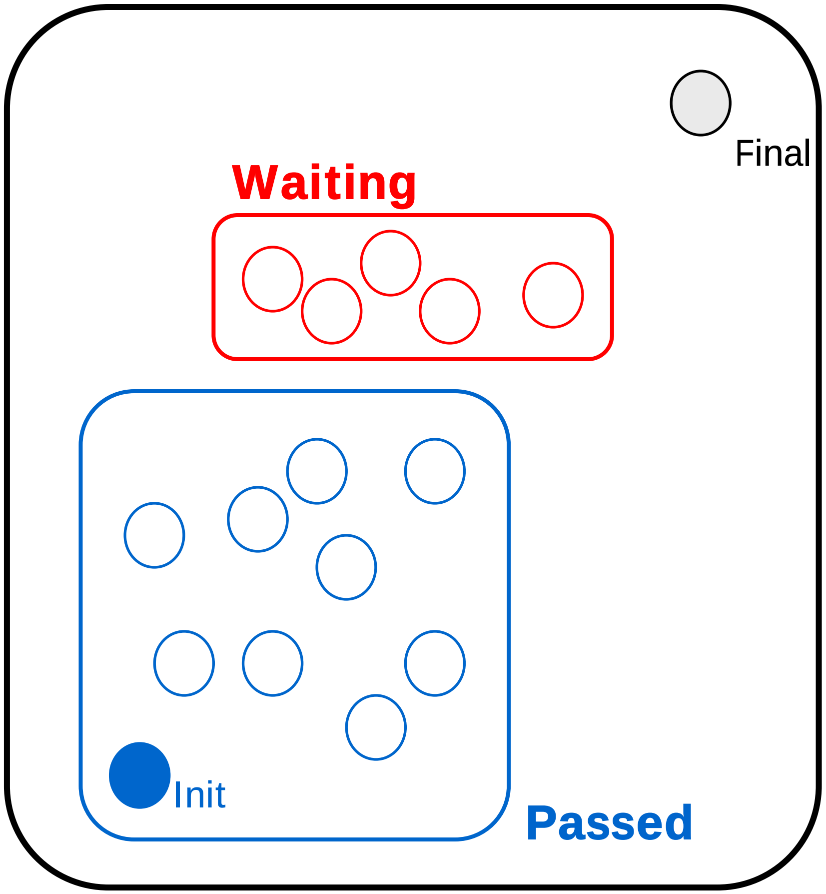




Down

Forward Rechability

Init -> Final ?



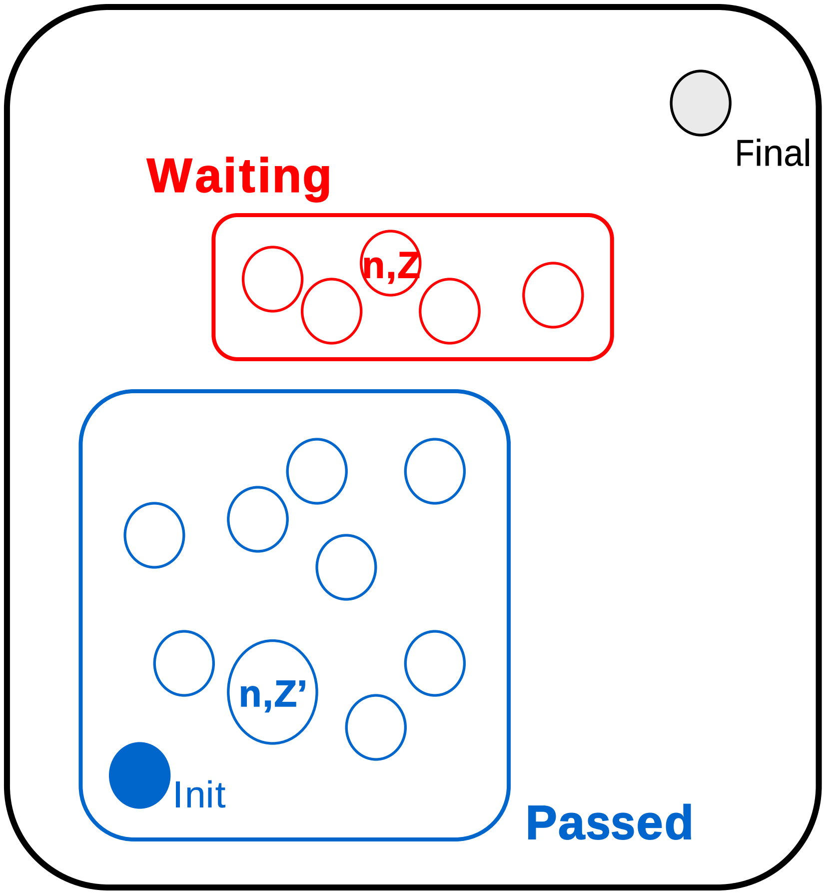
INITIAL **Passed** := Ø;
Waiting := {(n0,Z0)}

REPEAT

UNTIL **Waiting** = Ø
or
Final is in **Waiting**

Forward Rechability

Init -> Final ?



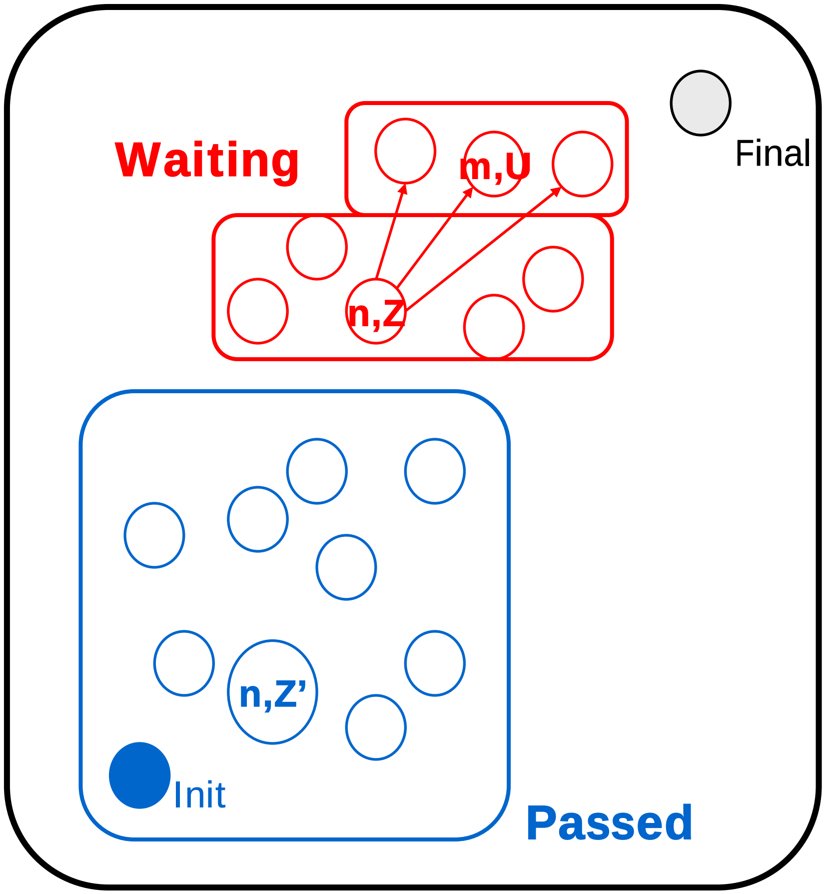
INITIAL **Passed** := Ø;
Waiting := {(n0,Z0)}

REPEAT
- pick (n,Z) in **Waiting**
- **if** for some Z' ⊇ Z (n,Z') in **Passed** **then STOP**

UNTIL **Waiting** = Ø
or
Final is in **Waiting**

Forward Rechability

Init -> Final ?



INITIAL **Passed** := Ø;
Waiting := {(n0,Z0)}

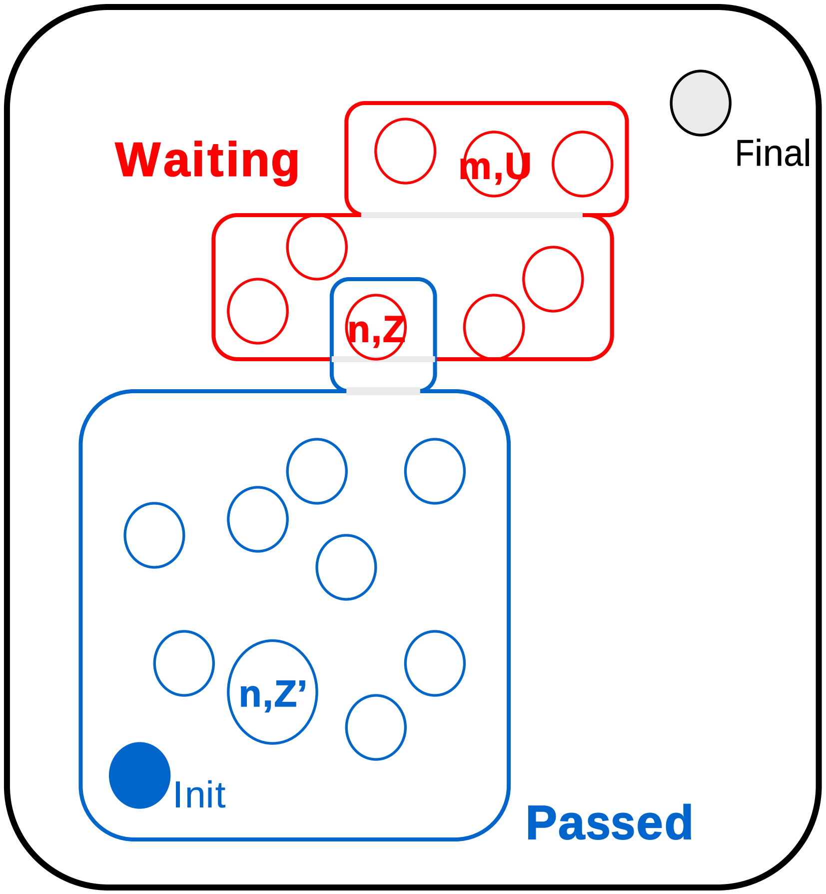
REPEAT

- pick (n,Z) in **Waiting**
- **if** for some Z' ⊇ Z (n,Z') in **Passed** **then STOP**
- **else** /explore/ add { (m,U) : (n,Z) => (m,U) } to **Waiting**;

UNTIL **Waiting** = Ø
or
Final is in **Waiting**

Forward Rechability

Init -> Final ?



INITIAL **Passed** := Ø;
Waiting := {(n0,Z0)}

REPEAT
- pick (n,Z) in **Waiting**
- **if** for some Z' ⊇ Z (n,Z') in **Passed** **then** **STOP**
- **else** /explore/ add { (m,U) : (n,Z) => (m,U) } to **Waiting**; Add (n,Z) to **Passed**

UNTIL **Waiting** = Ø
or
Final is in **Waiting**

Zones

Difference Bounded Matrices

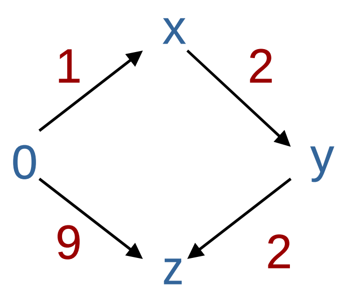
Bellman 1958, Dill 1989

Inclusion

D1

x<=1
y-x<=2
z-y<=2
z<=9

Graph

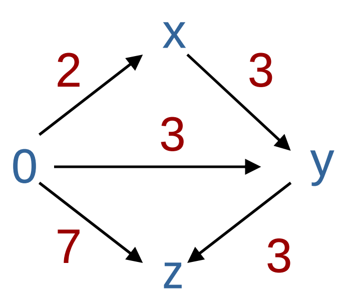


? ⊆ **?**

D2

x<=2
y-x<=3
y<=3
z-y<=3
z<=7

Graph



BRICS Basic Research in Computer Science

Zones

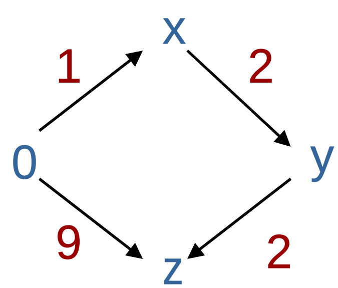
Difference Bounded Matrices

Inclusion

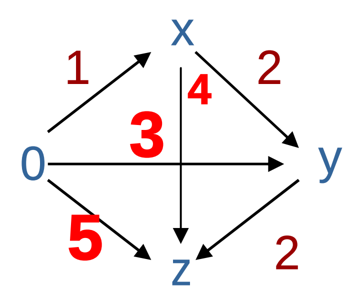
D1

x<=1
y-x<=2
z-y<=2
z<=9

Graph



Shortest Path Closure

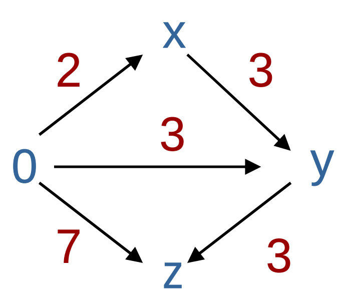


? $\subseteq$?

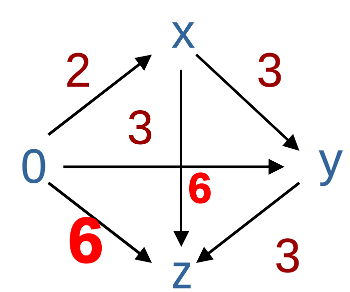
D2

x<=2
y-x<=3
y<=3
z-y<=3
z<=7

Graph



Shortest Path Closure



Zones

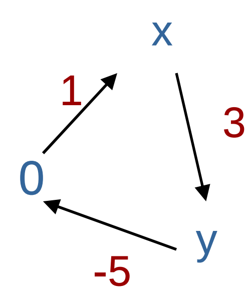
Difference Bounded Matrices

Emptiness

D

x<=1
y>=5
y-x<=3

Graph



Negative Cycle
iff
empty solution set

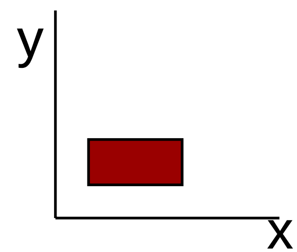
Compact

Zones

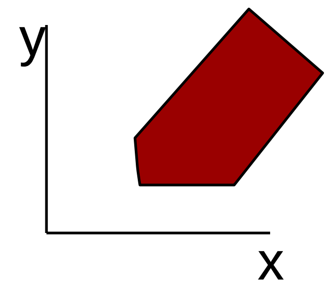
Difference Bounded Matrices

Future

D

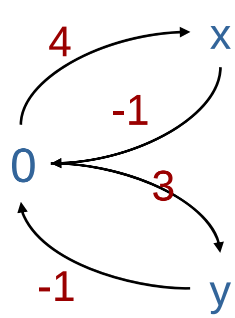


1<= x <=4
1<= y <=3

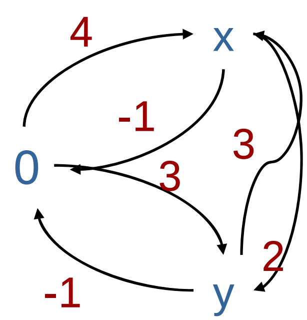


Future D

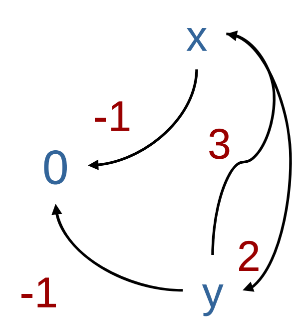
1<=x, 1<=y
-2<=x-y<=3



Shortest Path Closure



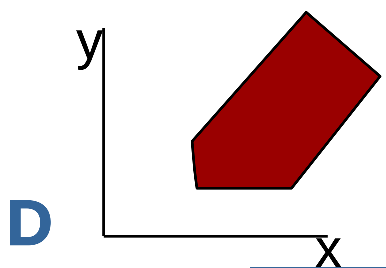
Remove upper bounds on clocks



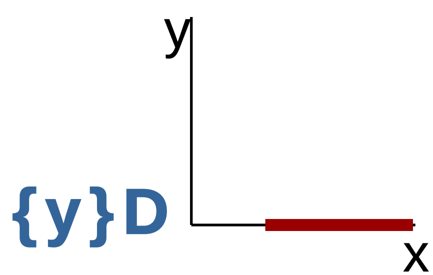
Zones

Difference Bounded Matrices

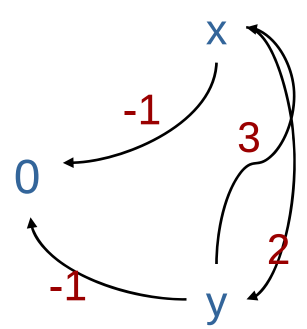
Reset



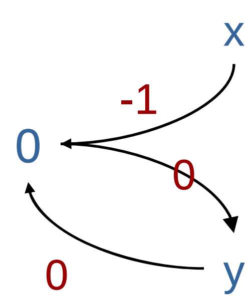
1<=x, 1<=y
-2<=x-y<=3



y=0, 1<=x



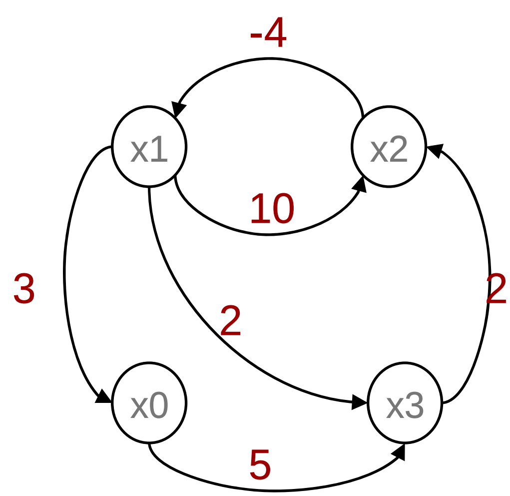
Remove all bounds involving y and set y to 0



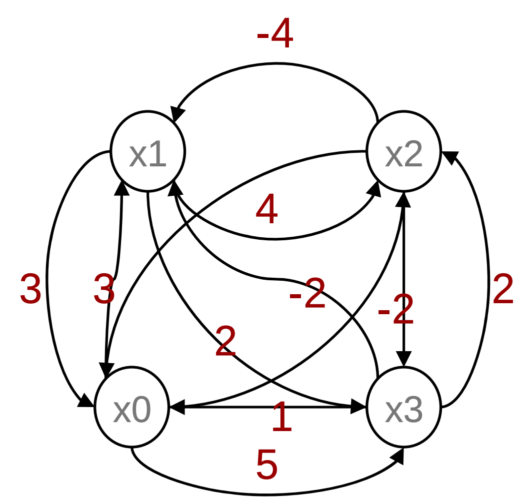
Zones

Difference Bounded Matrices

x1-x2<=4
x2-x1<=10
x3-x1<=2
x2-x3<=2
x0-x1<=3
x3-x0<=5



Shortest Path Closure O(n^3)

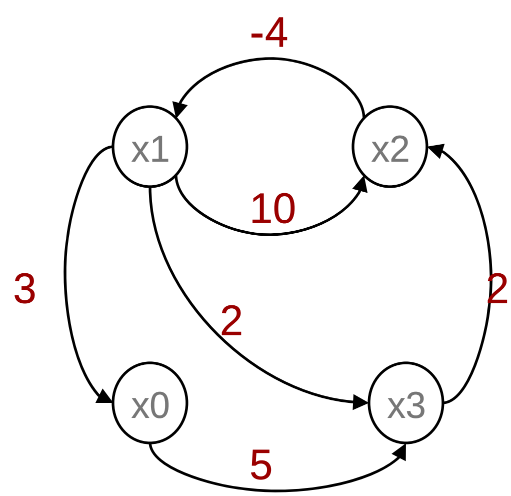


Zones

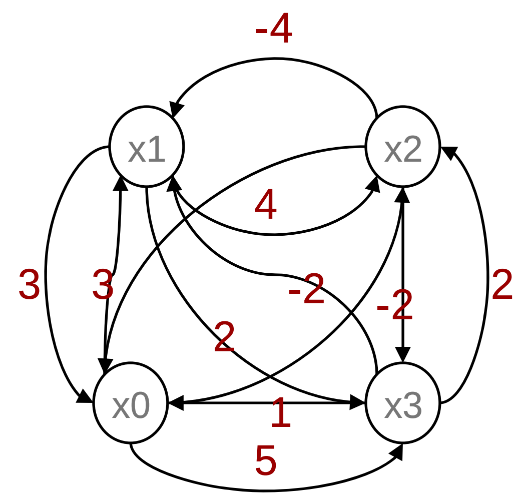
Minimal Constraint Form

RTSS 1997

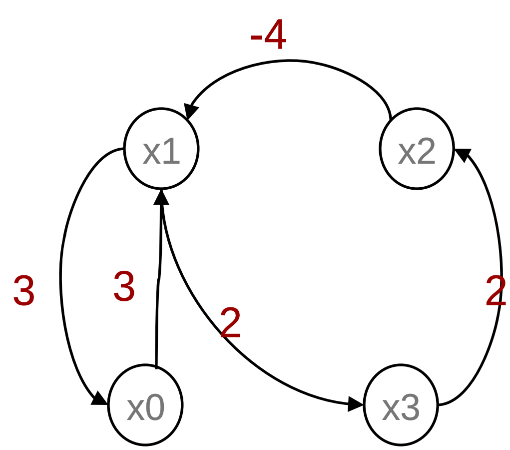
x1-x2<=4
x2-x1<=10
x3-x1<=2
x2-x3<=2
x0-x1<=3
x3-x0<=5



Shortest Path Closure O(n^3)

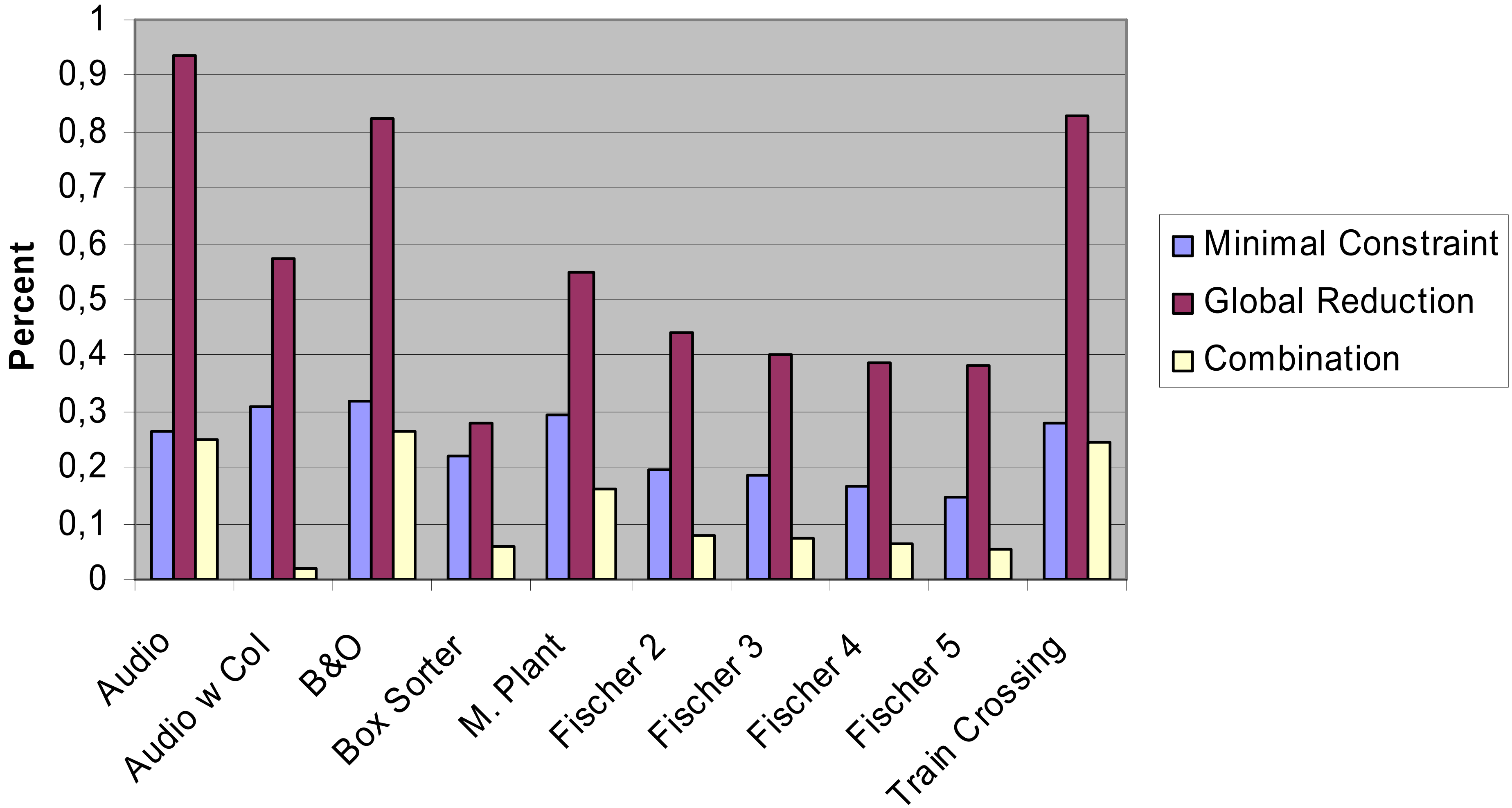


Shortest Path Reduction O(n^3)

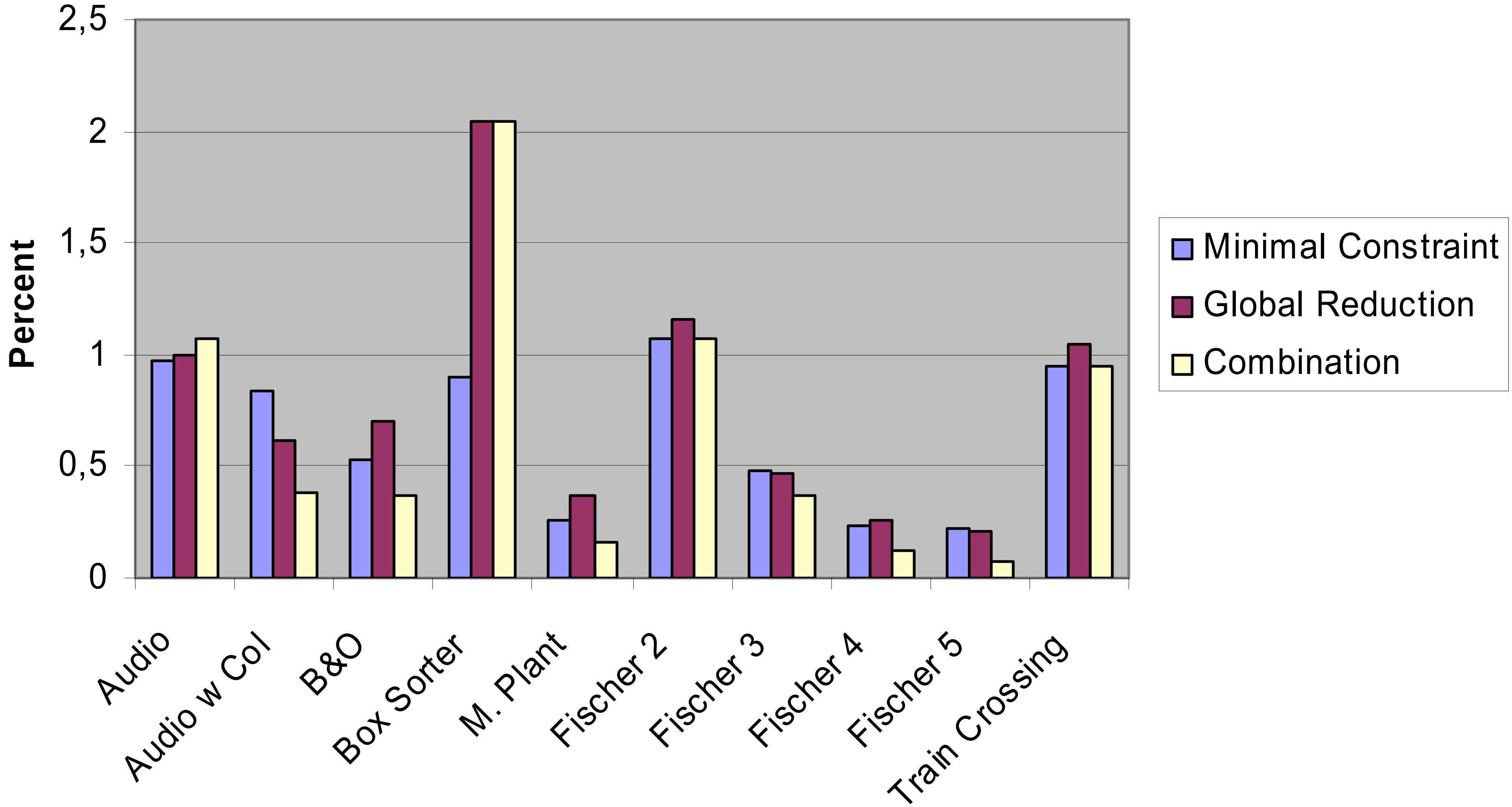


Space worst O(n^2)
practice O(n)

SPACE PERFORMANCE

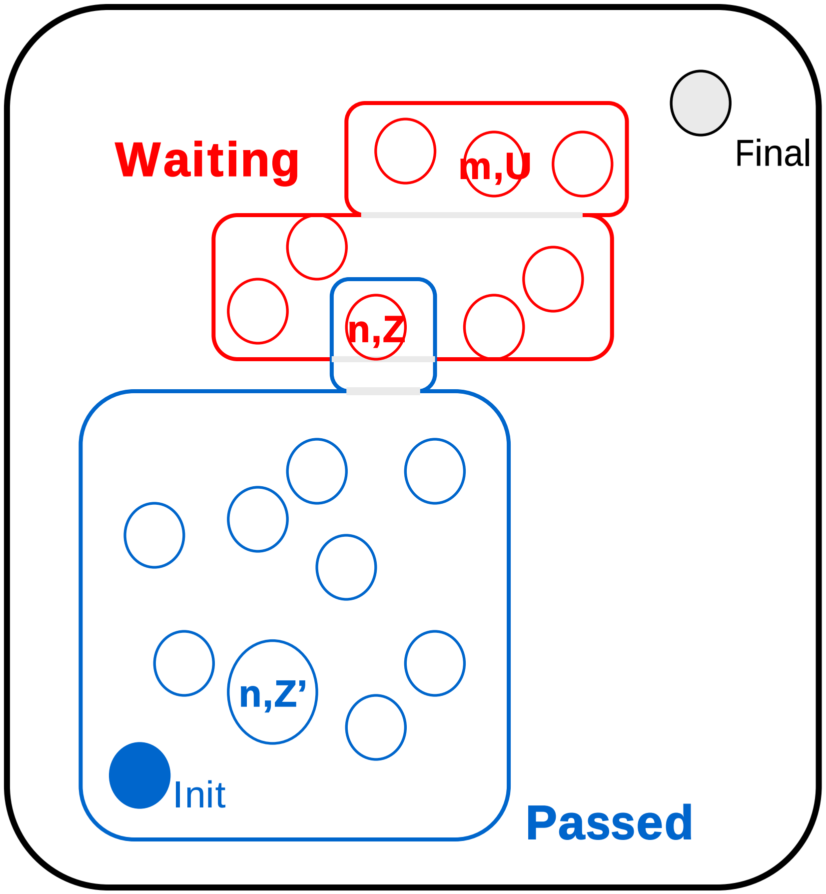


TIME PERFORMANCE



Earlier Termination

Init -> Final ?



INITIAL **Passed** := Ø;
Waiting := {(n0,Z0)}

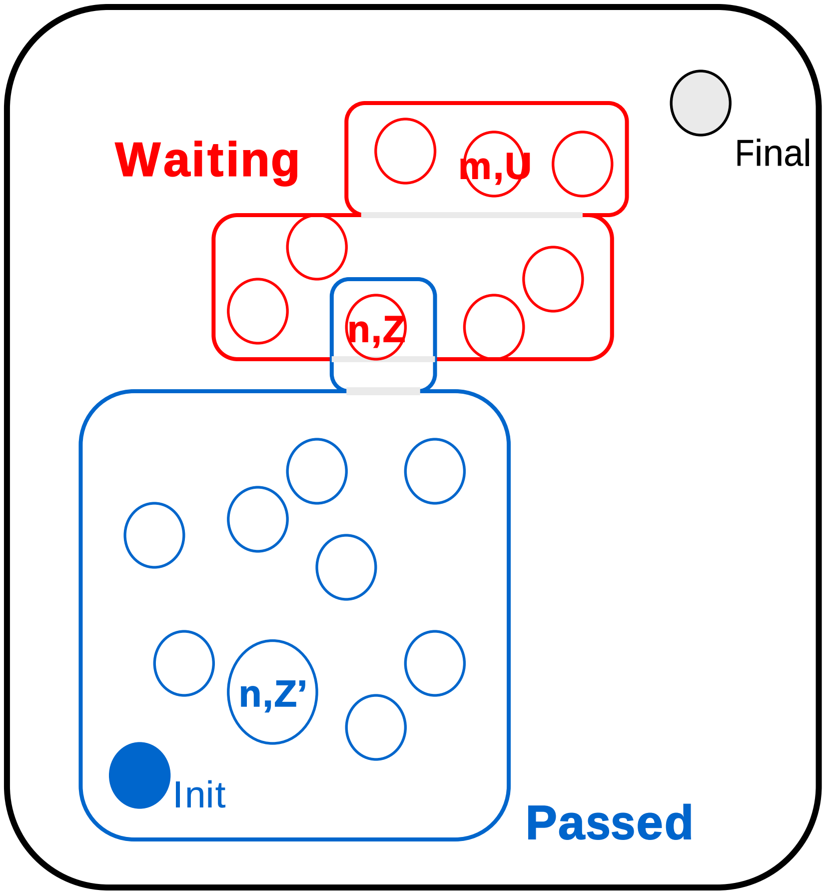
REPEAT
- pick (n,Z) in **Waiting**
- **if** for some Z' ⊇ Z (n,Z') in **Passed** **then STOP**
- **else** /explore/ add { (m,U) : (n,Z) => (m,U) } to **Waiting**;
  Add (n,Z) to **Passed**

UNTIL **Waiting** = Ø
or
Final is in **Waiting**

CISS
CENTER FOR INDLEJREDE SOFTWARE SYSTEMER

Earlier Termination

Init -> Final ?



INITIAL **Passed** := Ø;
Waiting := {(n0,Z0)}

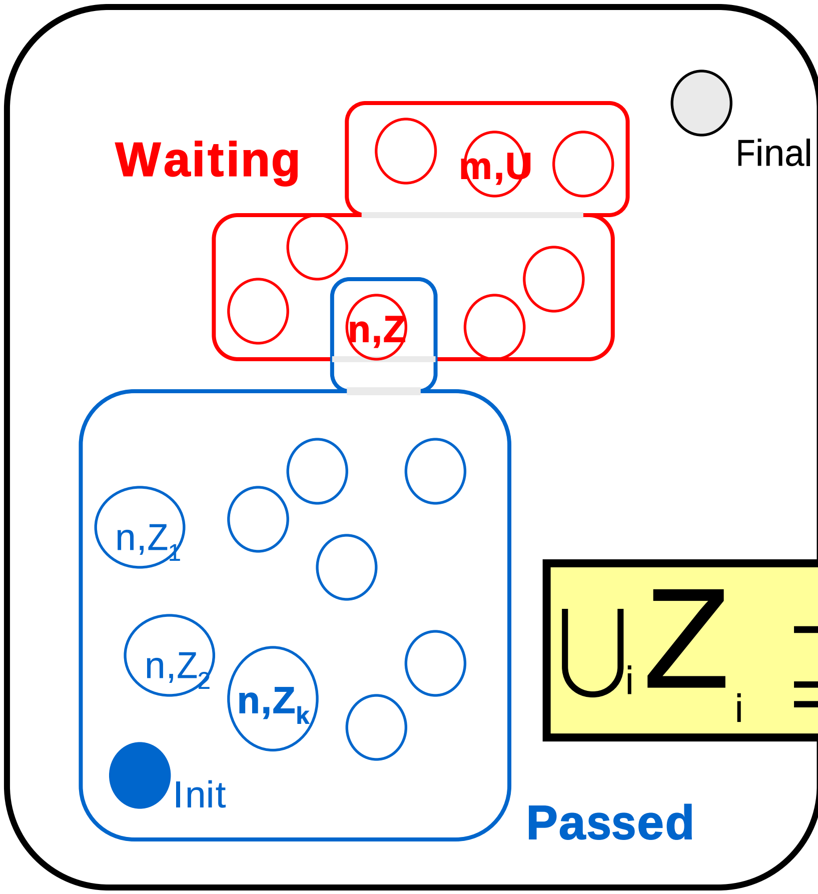
REPEAT

- pick (n,Z) in **Waiting**
- **if** for som $\boxed{Z' \supseteq Z}$ Z
  (n,Z') in **Passed** **then STOP**
- **else** /explore/ add
  { (m,U) : (n,Z) => (m,U) }
  to **Waiting**;
  Add (n,Z) to **Passed**

UNTIL **Waiting** = Ø
or
Final is in **Waiting**

Earlier Termination

Init -> Final ?



$$\bigcup_i Z_i \supseteq Z$$

INITIAL **Passed** := Ø;
Waiting := {(n0,Z0)}

REPEAT
- pick (n,Z) in **Waiting**
- **if** for som $Z' \supseteq Z$
  (n,Z') in **Passed** **then STOP**
- **el**... /explore/ add
  { (m,U) : (n,Z) => (m,U) }
  to **Waiting**;
  Add (n,Z) to **Passed**

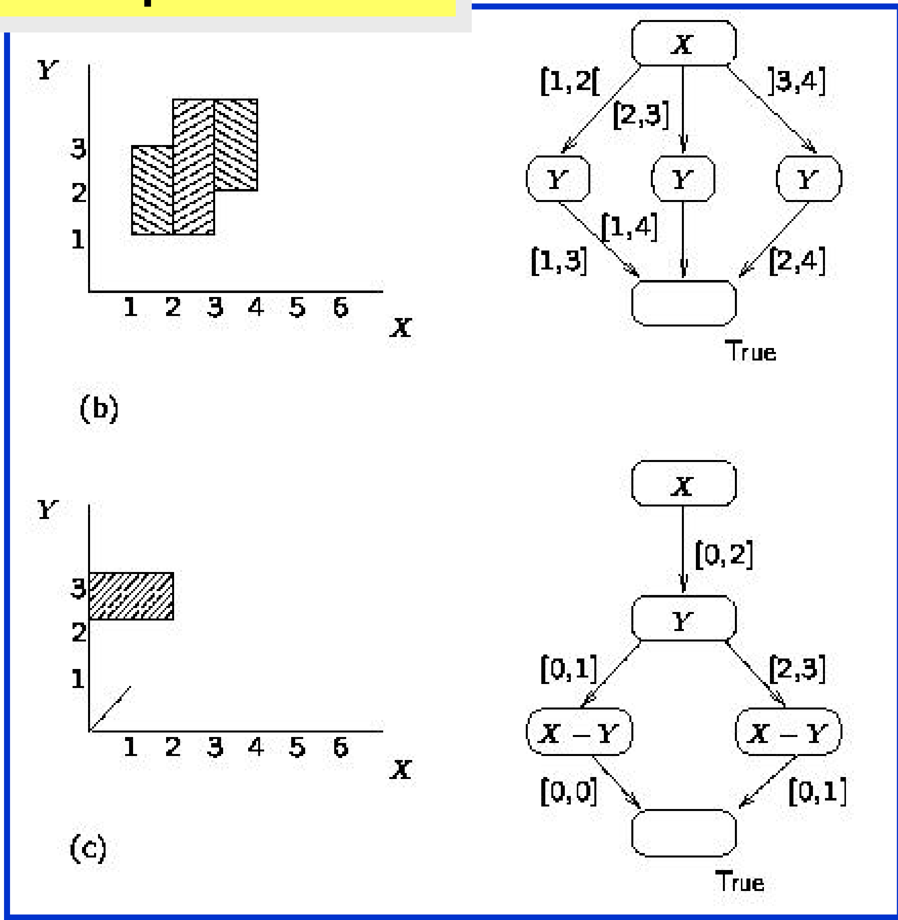
UNTIL **Waiting** = Ø
or
Final is in **Waiting**

BRICS
Basic Research
in Computer Science

Clock Difference Diagrams

= Binary Decision Diagrams + Difference Bounded Matrices **CAV99**

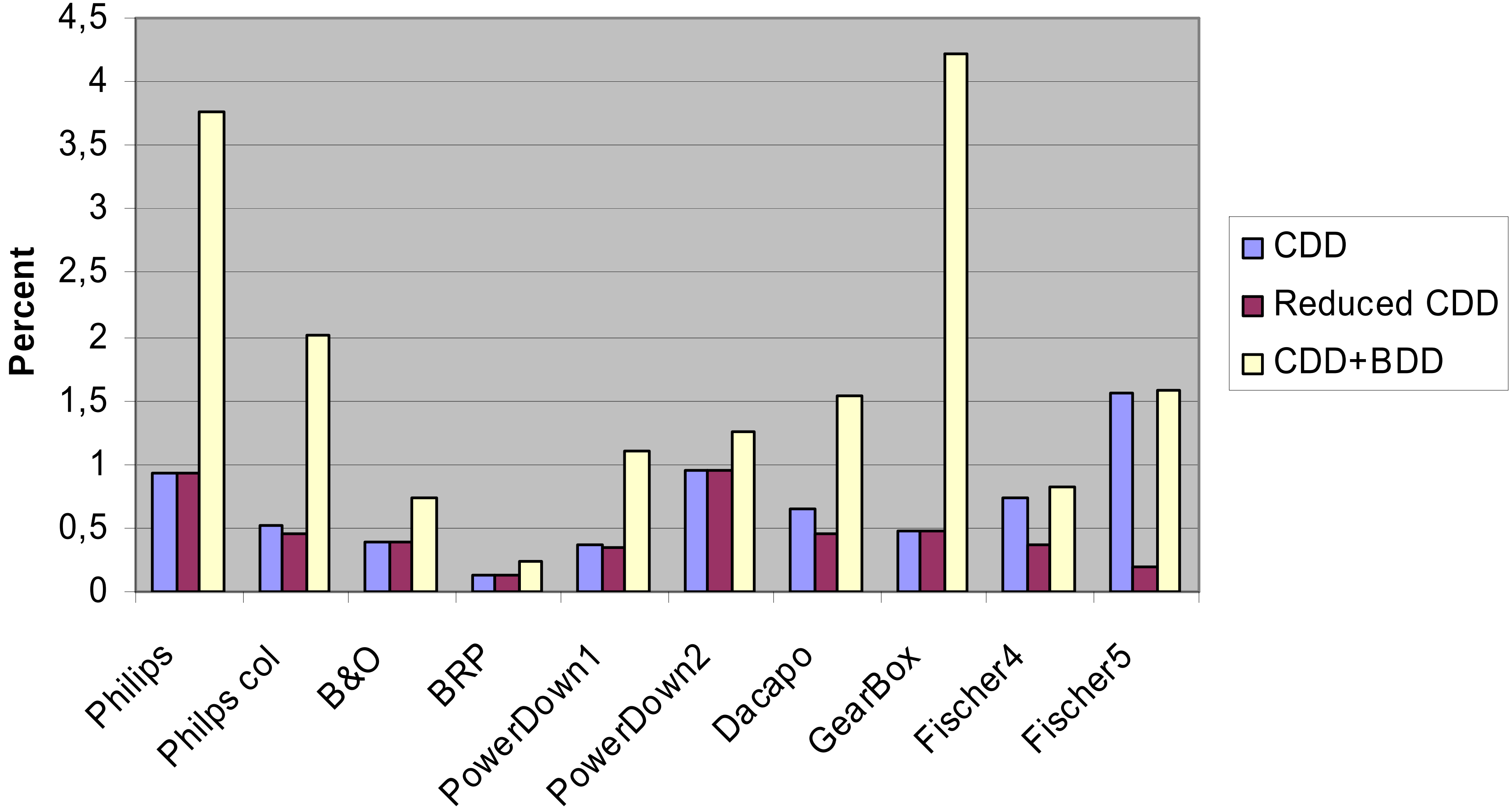
CDD-representations



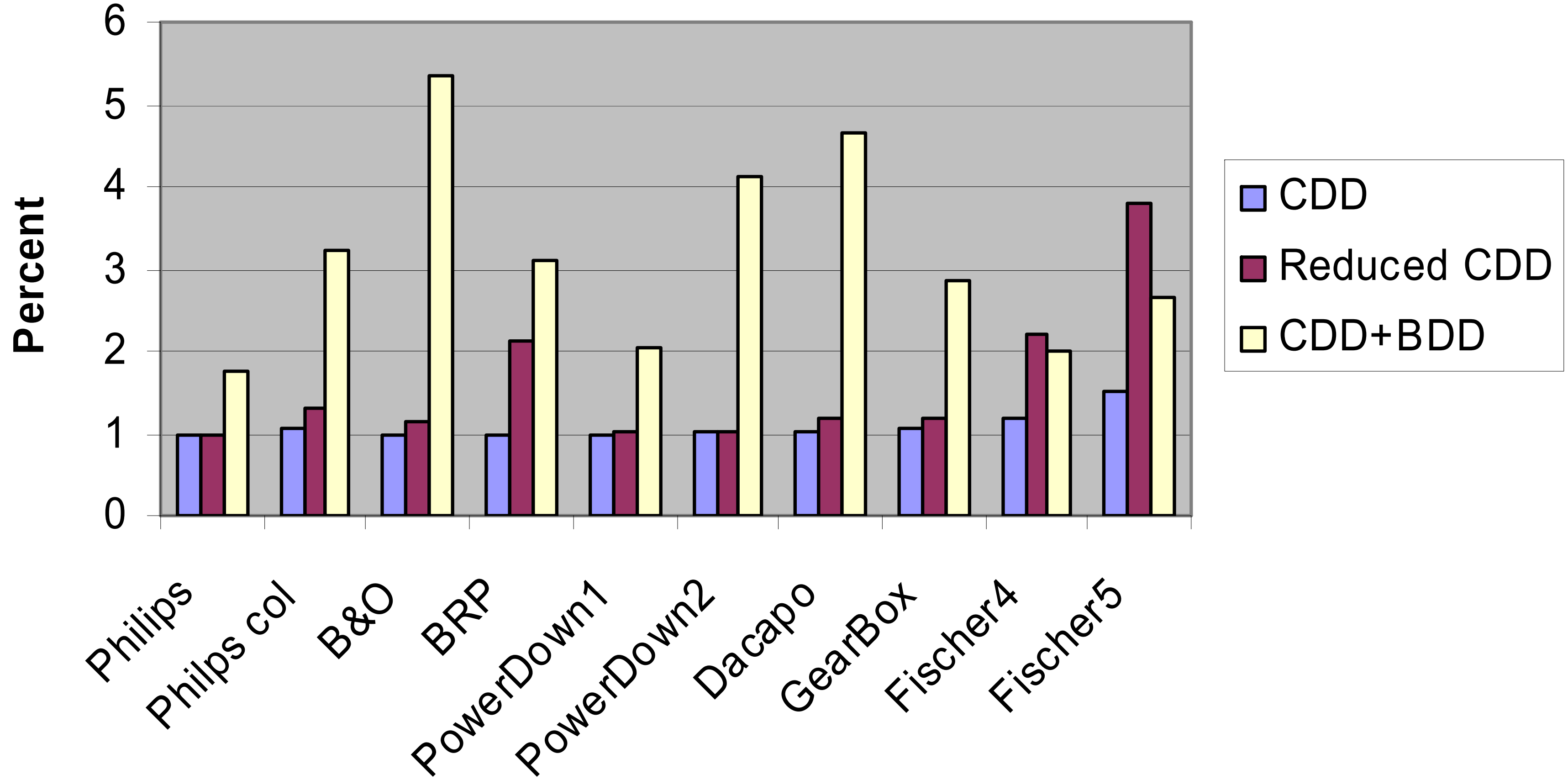
- Nodes labeled with differences
- Maximal sharing of substructures (also across different CDDs)
- Maximal intervals
- Linear-time algorithms for set-theoretic operations.

- NDD's Maler et. al
- DDD's Møller, Lichtenberg

SPACE PERFORMANCE



TIME PERFORMANCE



BRICS
Basic Research
in Computer Science

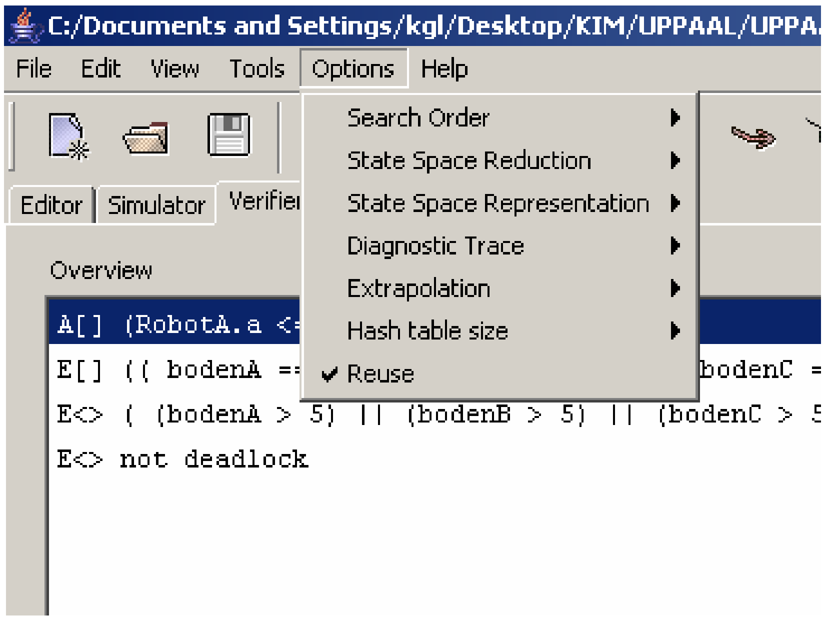
Verification Options

BRICS
Basic Research in Computer Science

CISS
CENTER FOR INDLEJREDE SOFTWARE SYSTEMER

AALBORG UNIVERSITY DENMARK

Verification Options



Search Order
- Depth First
- Breadth First

State Space Reduction
- None
- Conservative
- Aggressive

State Space Representation
- DBM
- Compact Form
- Under Approximation
- Over Approximation

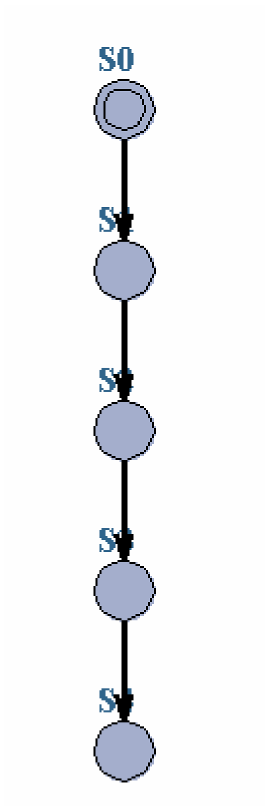
Diagnostic Trace
- Some
- Shortest
- Fastest

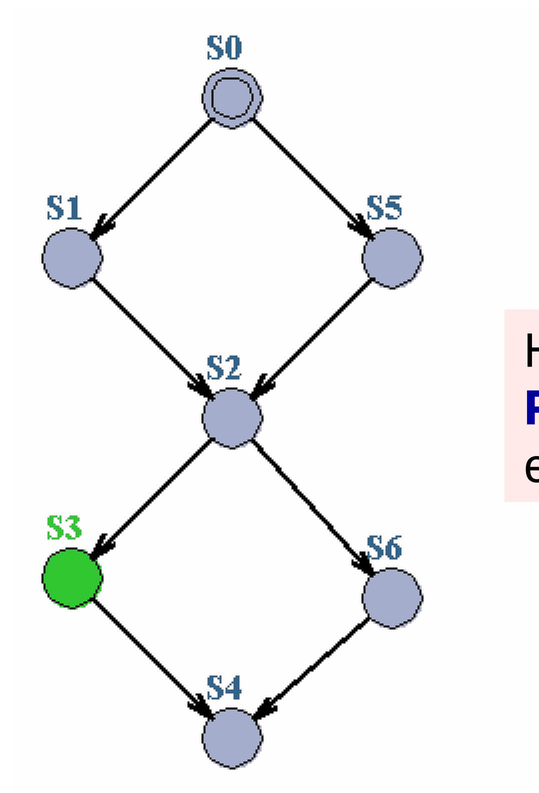
Extrapolation

Hash Table size

Reuse

State Space Reduction

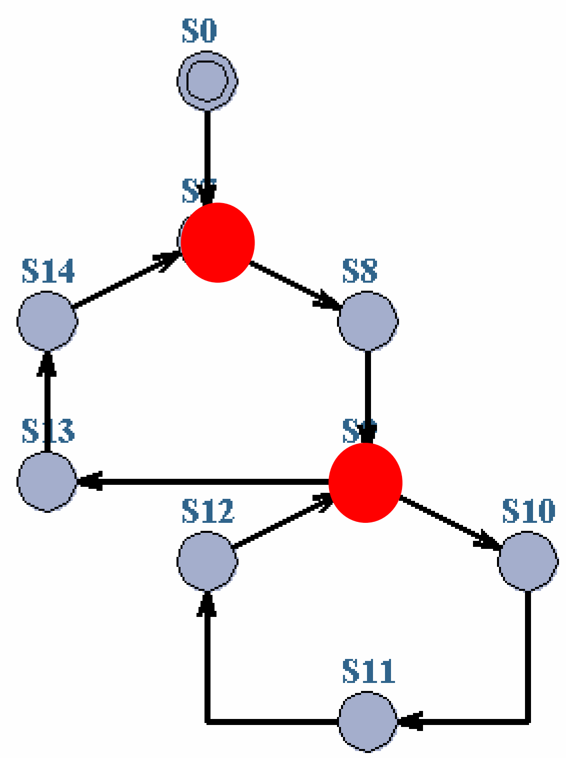




However, **Passed** list useful for efficiency

No Cycles: **Passed** list not needed for *termination*

State Space Reduction



Cycles:

Only symbolic states
involving loop-entry points
need to be saved on **Passed** list

To Store or Not To Store

Behrmann, Larsen, Pelanek 2003

117 states$_{total}$

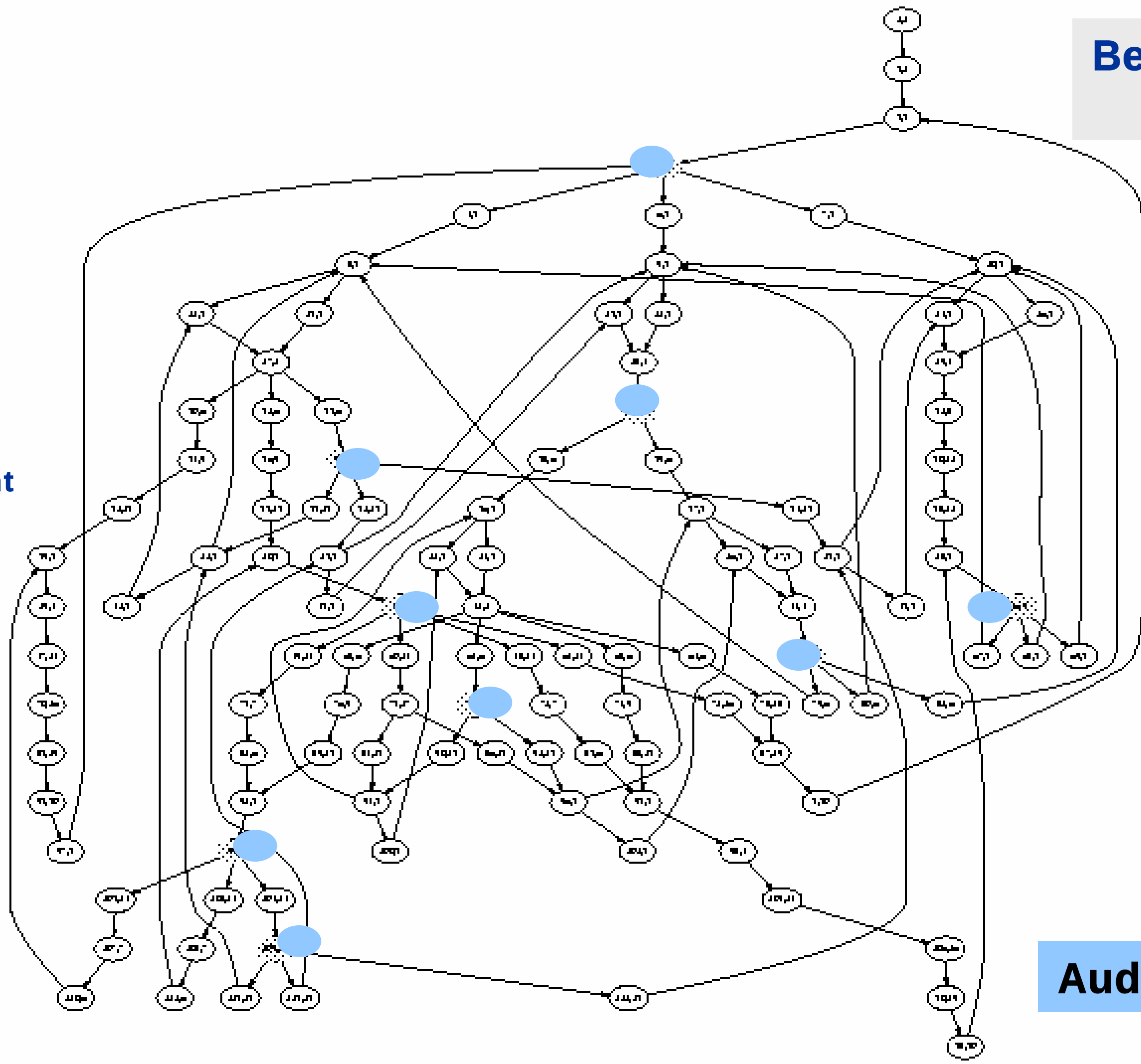
$\rightarrow$

81 states$_{entrypoint}$

$\rightarrow$

9 states

Time OH less than 10 %



Audio Protocol

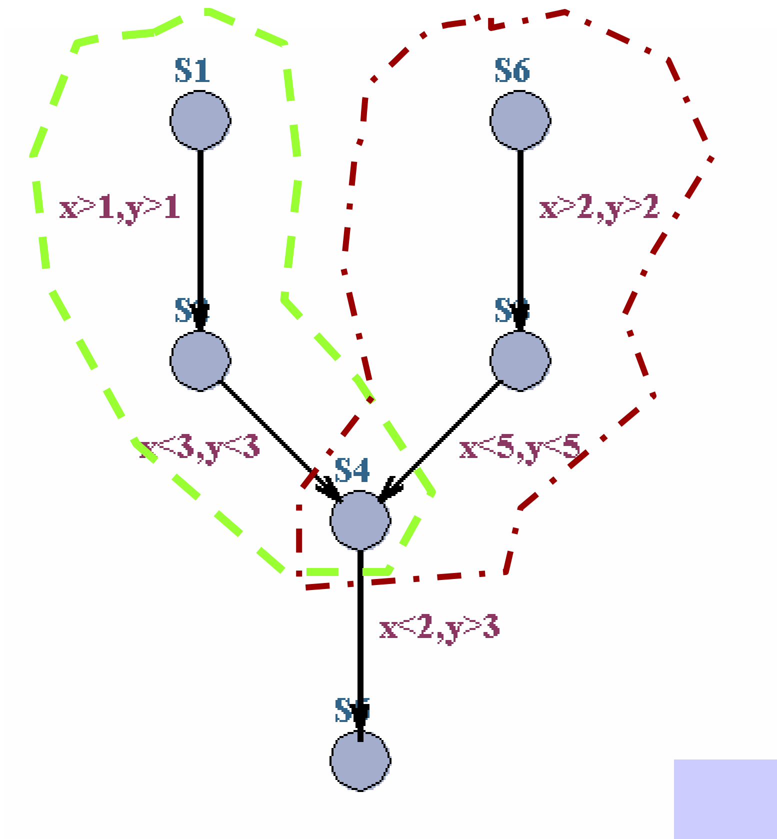
To Store or Not to Store

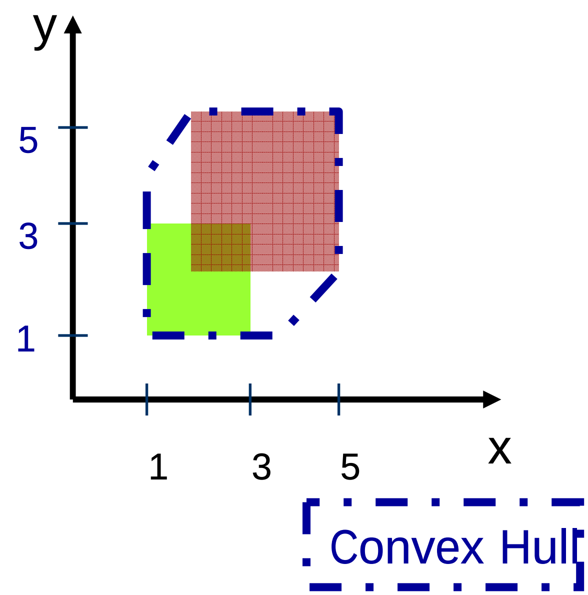
Behrmann, Larsen, Pelanek 2003

	entry points	covering set	successors	random $p = 0.1$	distance $k = 10$	combination $k = 3$
Fischer 3,077	27.1% 1.00	42.1% 1.66	47.9% 1.00	53.7% 4.51	67.6% 2.76	56.9% 6.57
BRP 6,060	70.5% 1.01	16.5% 1.20	19.8% 1.03	18.3% 1.78	15.8% 1.34	7.6% 1.68
Token Ring 15,103	33.0% 1.16	10.3% 1.46	20.7% 1.03	17.2% 1.63	17.5% 1.43	16.8% 7.40
Train-gate 16,666	71.1% 1.22	27.4% 1.55	24.2% 1.68	31.8% 2.90	24.2% 2.11	19.8% 5.08
Dacapo 30,502	29.4% 1.07	24.3% 1.08	24.9% 1.07	12.2% 1.21	12.7% 1.16	7.0% 1.26
CSMA 47,857	94.0% 1.06	75.9% 2.62	81.2% 1.40	105.9% 7.66	114.9% 2.83	120.3% 6.82
BOCDP 203,557	25.2% 1.00	22.5% 1.01	6.5% 1.08	10.2% 1.02	9.3% 1.01	4.5% 1.09
BOPDP 1,013,072	14.7% 2.40	13.2% 1.33	42.1% 1.02	15.2% 1.52	11% 1.14	4.3% 1.74
Buscoupler 3,595,108	53.2% 1.29	13.6% 2.48	40.5% 1.18	31.7% 3.17	24.6% 2.13	14.3% 8.73

Over-approximation

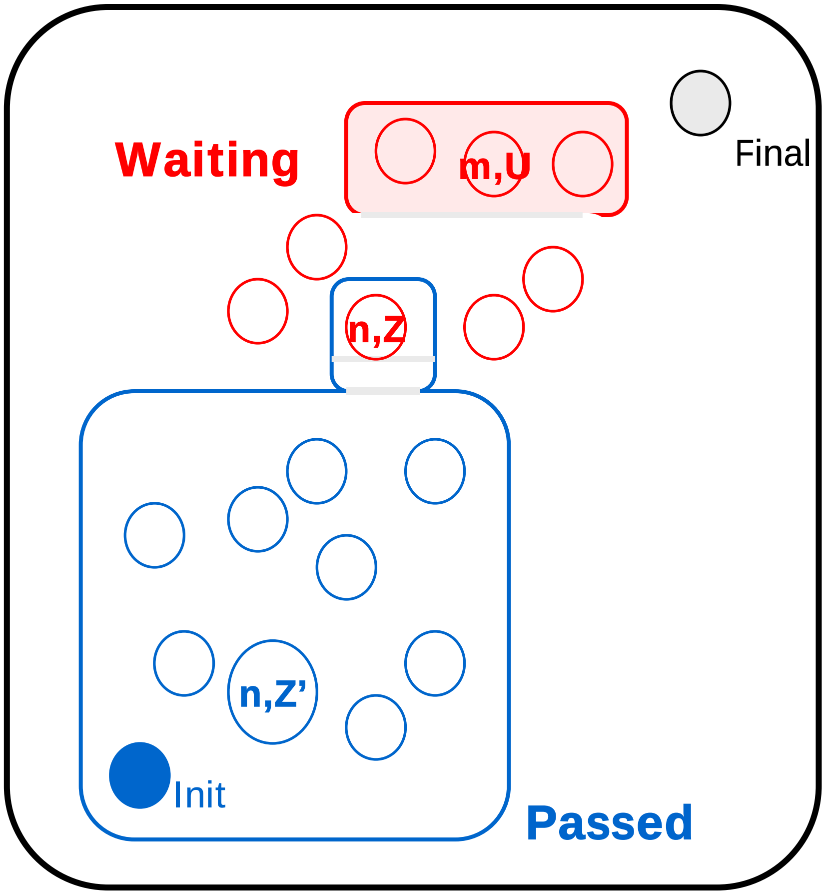
Convex Hull



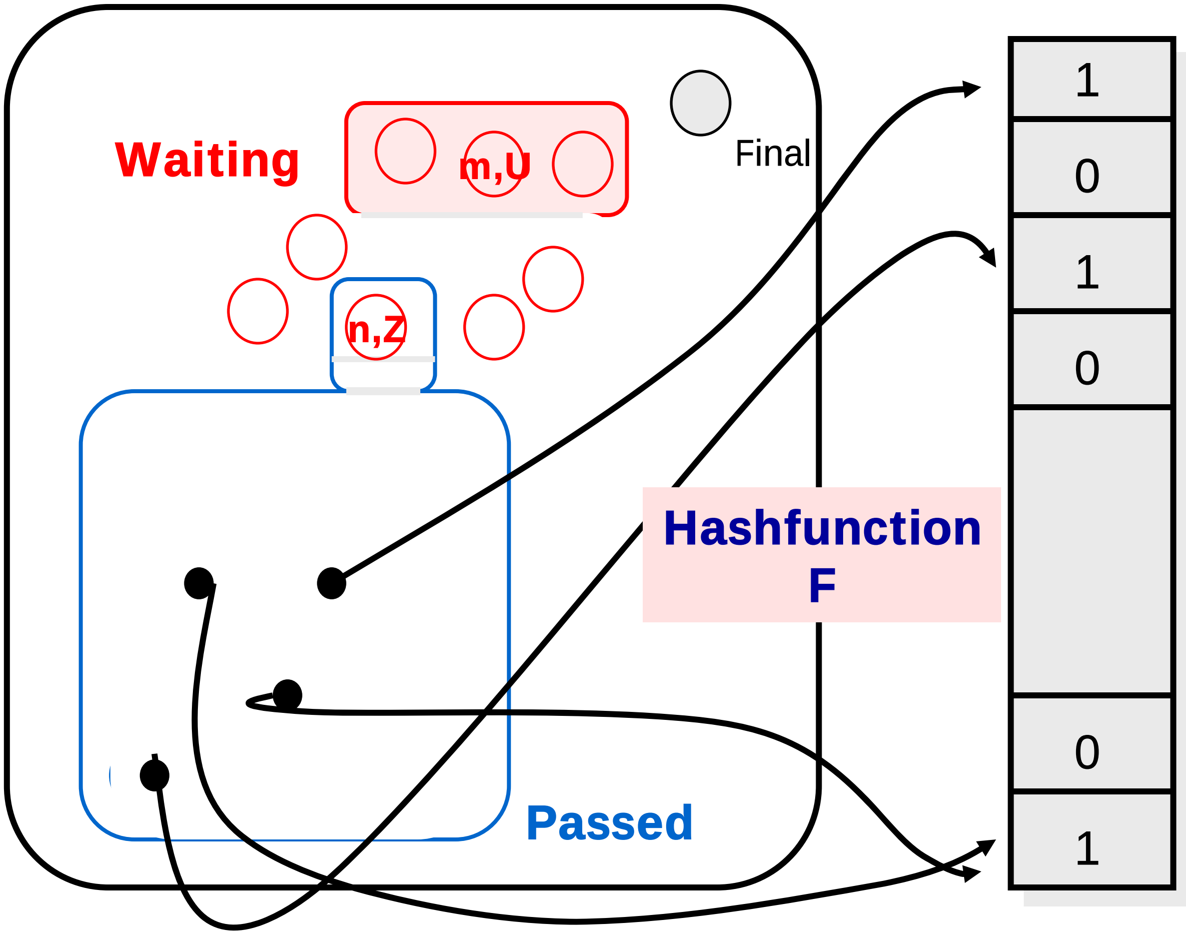


TACAS04: An **EXACT** method performing as well as Convex Hull has been developed based on abstractions taking max constants into account distinguishing between clocks, locations and $\leq$ & $\geq$

Under-approximation
Bitstate Hashing



Under-approximation
Bitstate Hashing



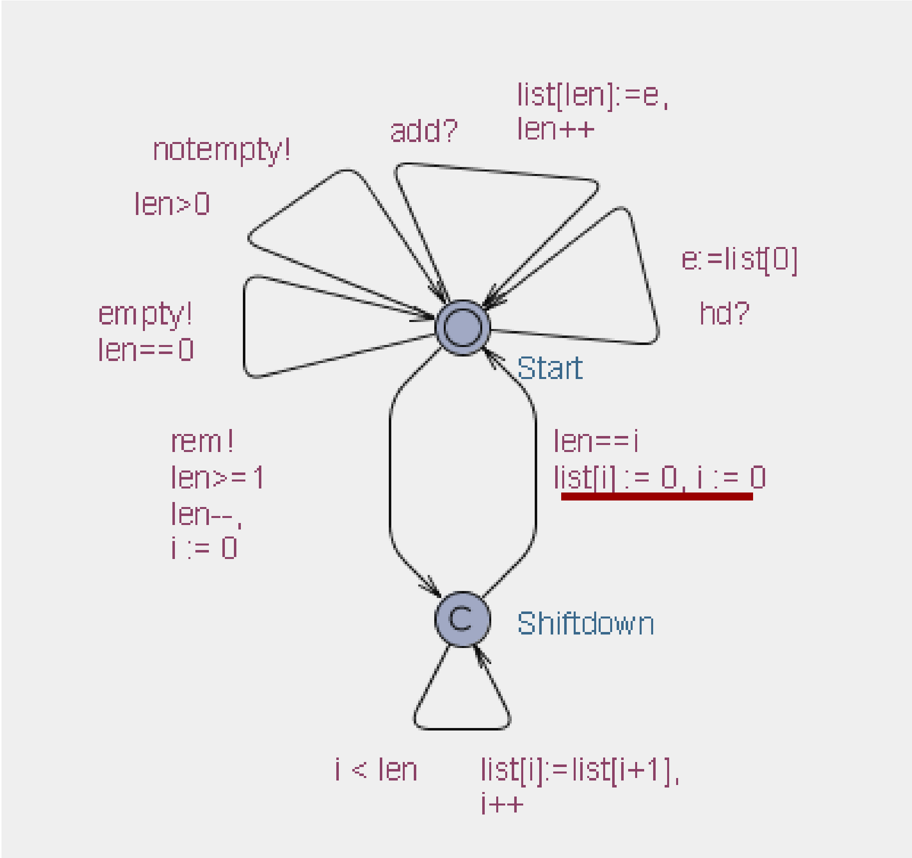
Passed= Bitarray

UPPAAL
8 Mbits

Modelling Patterns

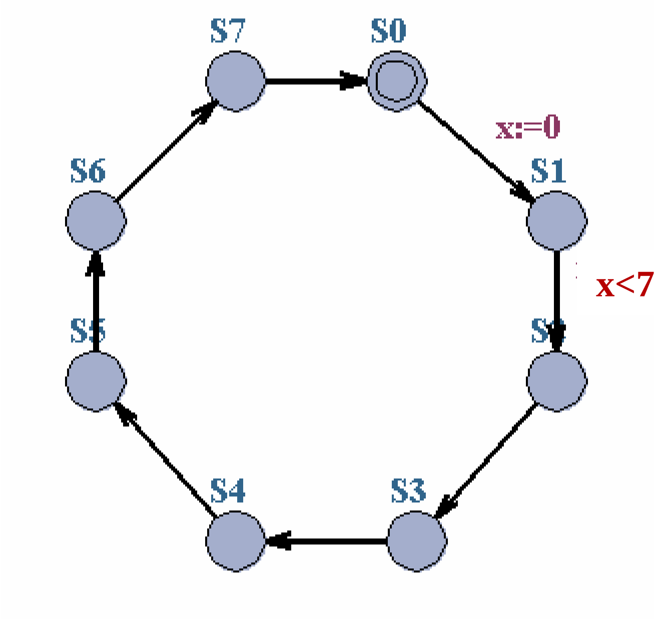
Variable Reduction

- Reduce size of state space by explicitely resetting variables when they are not used!
- Automatically performed for clock variables (active clock reduction)



BRICS
Basic Research
in Computer Science

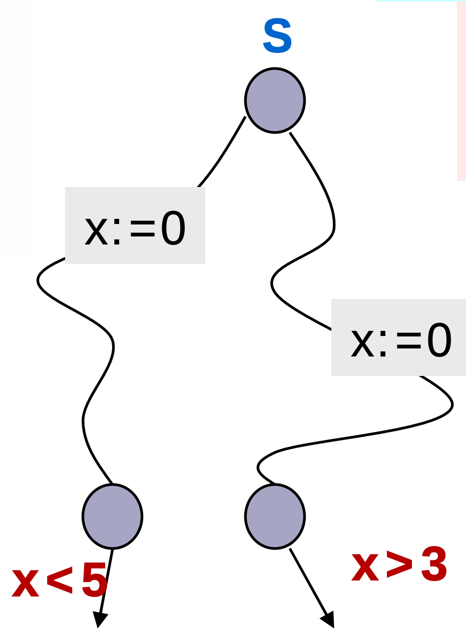
Variable Reduction



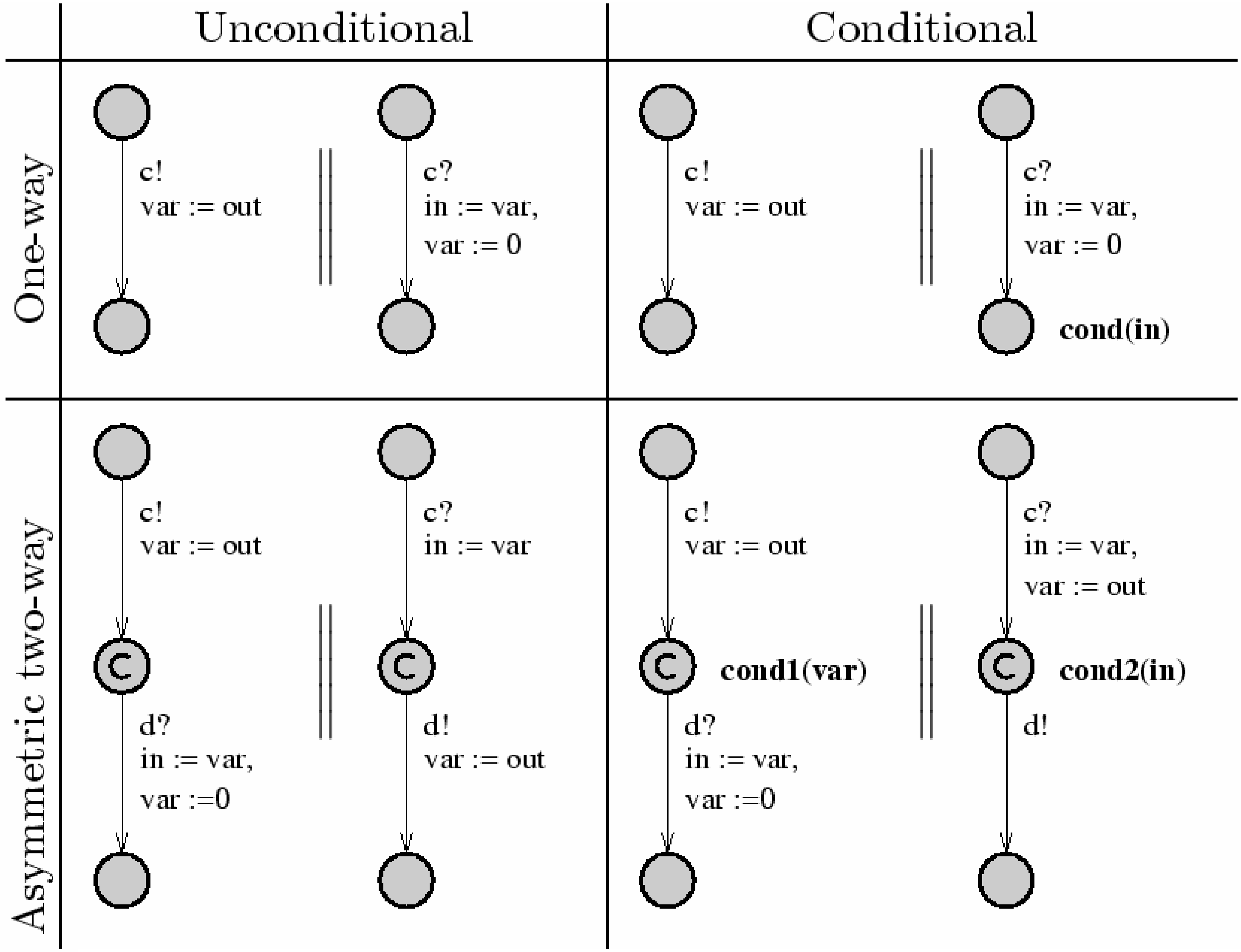
x is only ***active*** in location **S1**

Definition

x is ***inactive*** at **S** if on all path from **S**, x is always reset before being tested.

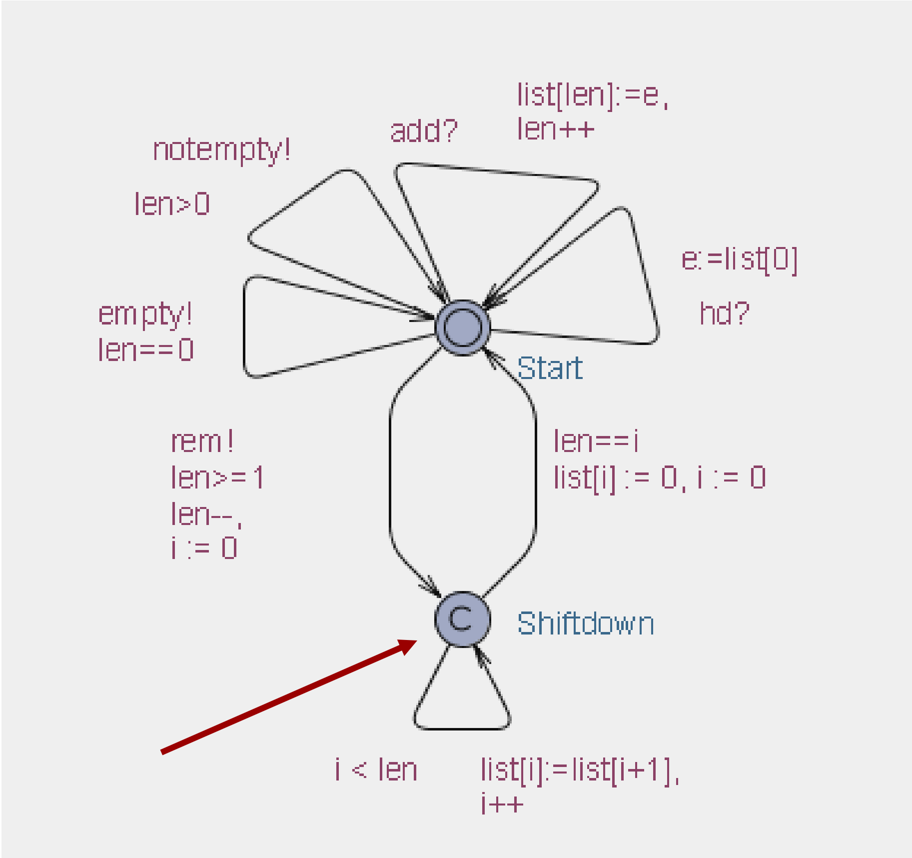


Synchronous Value Passing



Atomicity

- To allow encoding of control structure (for- or while-loops, conditionals, etc.) without erroneous interleaving
- To allow encoding of multicasting.
- Heavy use of committed locations.

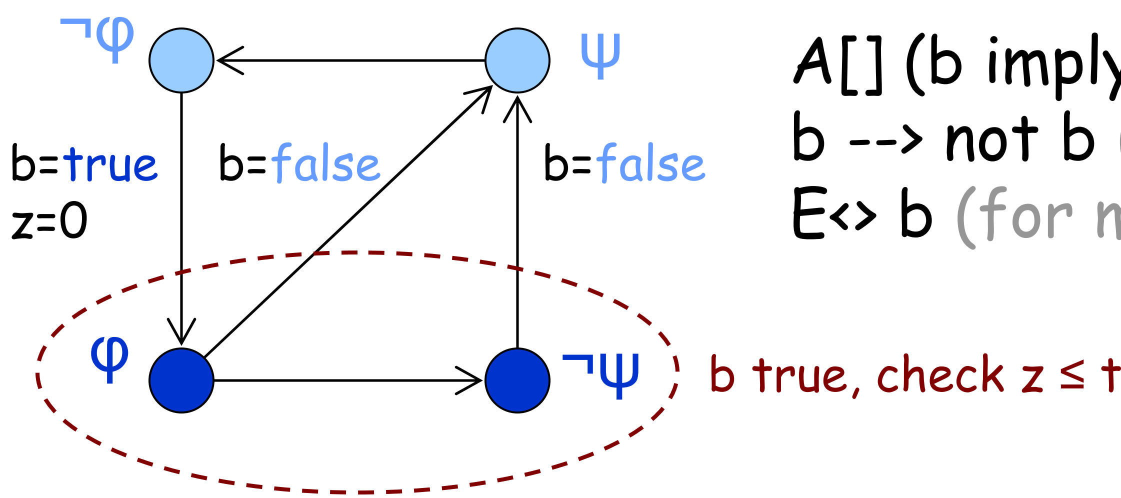


Bounded Liveness

- Intent: Check for properties that are guaranteed to hold eventually within some upper (time) bound.
  - Provide additional information (with a valid bound).
  - More efficient verification.
  - φ leadsto$_{\leq t}$ ψ reduced to $A\Box(b \Rightarrow z \leq t)$ with bool b set to *true* and clock z reset when φ starts to hold. When ψ starts to hold, set b to *false*.

Bounded Liveness

- The truth value of b indicates whether or not ψ should hold in the future.



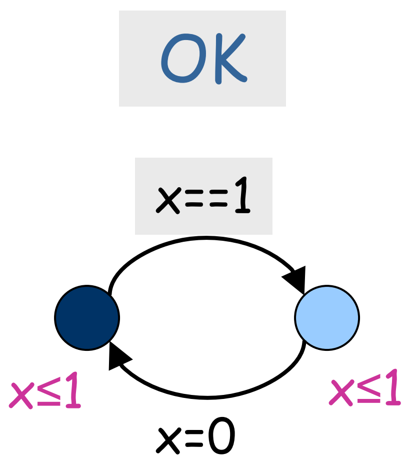
A[] (b imply z≤t)
b --> not b (for non zenoness)
E<> b (for meaningful check)

Zenoness

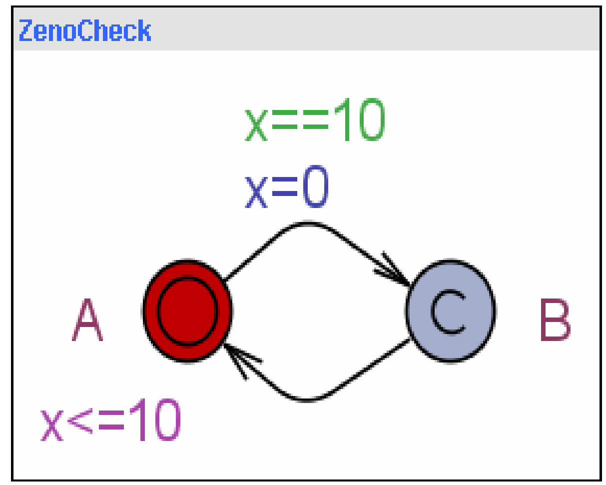
- Problem: UPPAAL does not check for zenoness directly.
  - A model has “zeno” behavior if it can take an infinite amount of actions in finite time.
  - That is usually not a desirable behavior in practice.
  - Zeno models may wrongly conclude that some properties hold though they logically should not.
  - Rarely taken into account.

- Solution: Add an observer automata and check for non-zenoness, i.e., that time will always pass.

Zenoness



Detect by
•adding the
observer:



Constant (10) can be anything (>0), but choose it well w.r.t. your model for efficiency.
Clocks ‘x’ are local.

•and check the property
`ZenoCheck.A --> ZenoCheck.B`

PRELIMINARY

more to come later
in lectures by
Jacob Illum Rasmussen

AMETIST

advanced methods for timed systems

Optimal Real Time Planning & Scheduling

with Gerd Behrmann, Ed Brinksma, Ansgar Fehnker,
Thomas Hune, Paul Pettersson, Judi Romijn,
Frits Vaandrager, Patricia Bouyer, Franck Cassez,
Emmanuel Fleury, Arne Skou, Jacob Rasmussen,
K. Subramani

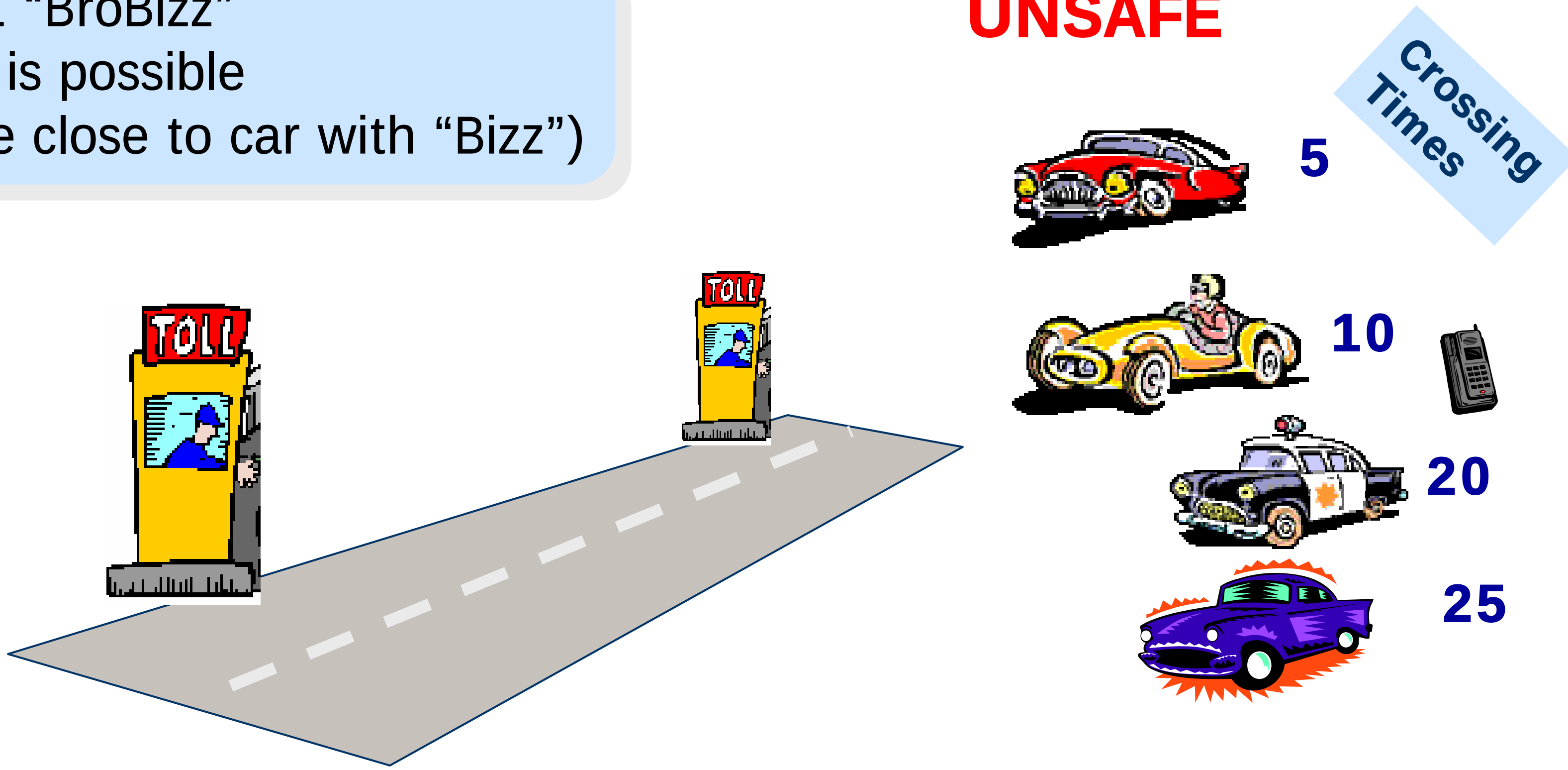
Aalborg University Denmark

BRICS
Basic Research in Computer Science

CISS
CENTER FOR INDLEJREDE SOFTWARE SYSTEMER

Real Time Scheduling

- Only 1 “BroBizz”
- Cheat is possible
  (drive close to car with “Bizz”)

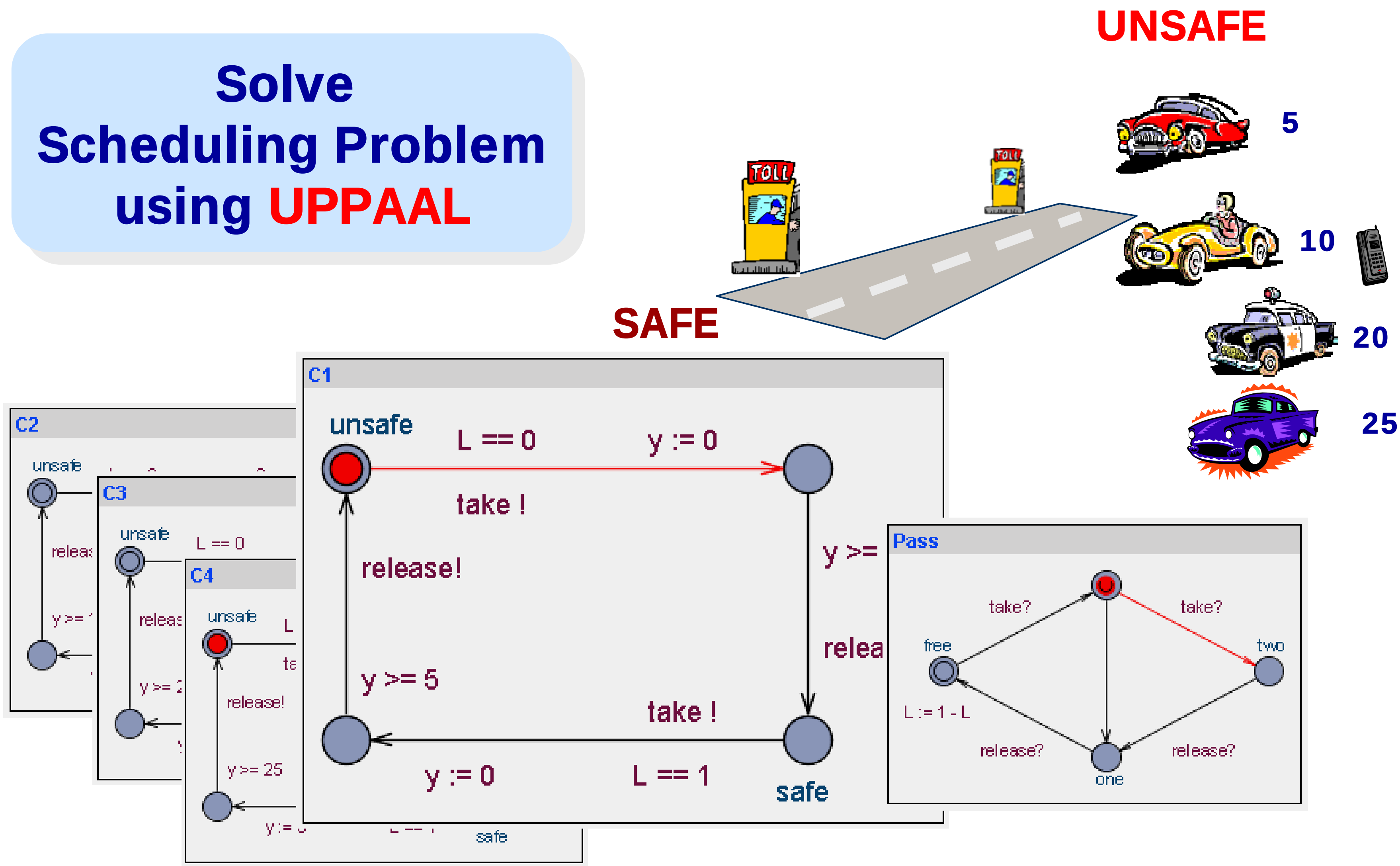


SAFE

CAN THEY MAKE IT TO SAFE WITHIN 70 MINUTES ???

Real Time Scheduling

Solve Scheduling Problem using UPPAAL



Rush Hour

OBJECTIVE: Get your CAR out

EEF Summerschool on Concurrency, Kapellerput

BRICS
Basic Research in Computer Science

END

☺☺☺