Model Checking of Infinite-State Systems

Parosh Aziz Abdulla

Bengt Jonsson

Uppsala University

Outline

Part 1

- Introduction
- Motivation
- Symbolic model checking and Well Quasi-Orderings
- Models: Petri nets, lossy channel systems, timed Petri nets, broadcast protocols, etc.
- Better Quasi-Orderings

Part 2

- Systems using word-shaped data structures: queues, stacks, integers, clocks
- Parameterized networks with linear or ring-shaped topology
- Regular sets as symbolic representation

Program Verification

- **Instance**
  - P: **Program**
  - ϕ: **Specification**
- **Question**
  - $P \models \phi$?

Model Checking

- **Automatic Verification**

Finite-State Systems

$$P \models \phi$$

- P: finite state space

  ϕ: CTL, PTL, etc.

- Applications
  - hardware and protocol verification

- Limitations
  - state explosion
  - infinite-state systems

Sources of Infiniteness

1. Unbounded Data Structures
   - Timed Automata
   - Push-down Automata
   - Communicating Finite-State Machines
   - Counter Machines

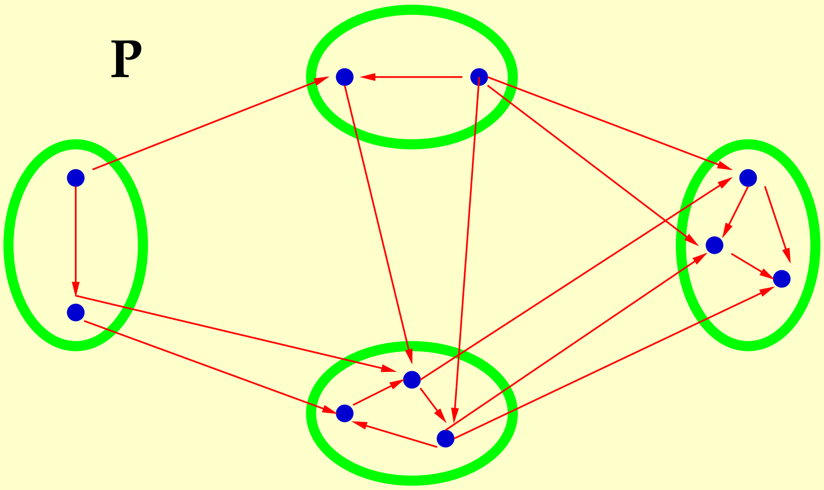
2. Unbounded Control Structures
   - Petri Nets
   - Parametrized Systems
     - e.g. mutual exclusion protocols, cache coherence protocols, broadcast protocols

Finite Partitioning

equivalence $\equiv$ on state space

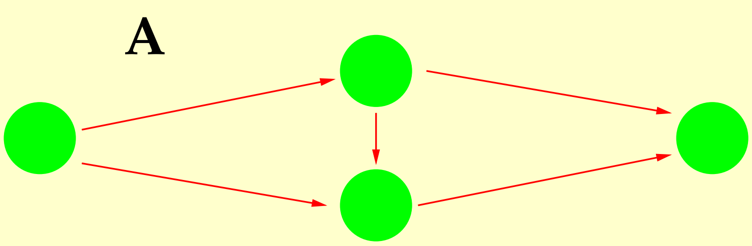
- $\equiv$ finite
- $\equiv$ bisimulation

$$\begin{tikzcd} s1 \arrow[r] \arrow[d, phantom, "|||"] & t1 \arrow[d, phantom, "|||"] \\ s2 \arrow[r] & t2 \end{tikzcd}$$



$\models \; \phi$

iff

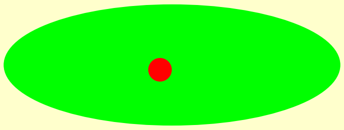
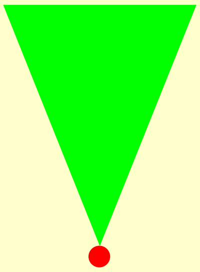


$\models \; \phi$

Finite Partitioning

- possible
  - timed automata
  - data-independent systems
- not possible
  - Petri nets
  - lossy channel systems
  - parametrized systems
  - relational automata
  - timed Petri nets

⋮

Equivalence $\equiv$	Preorder $\preceq$
Equivalence Class 	**Ideal** 
Bisimulation $\begin{array}{ccc} s_1 & \longrightarrow & t_1 \\ \rotatebox{90}{$\equiv$} & & \rotatebox{90}{$\equiv$} \\ s_2 & \longrightarrow & t_2 \end{array}$	**Simulation** $\begin{array}{ccc} s_1 & \longrightarrow & t_1 \\ \rotatebox{90}{$\preceq$} & & \rotatebox{90}{$\preceq$} \\ s_2 & \longrightarrow & t_2 \end{array}$
Finite number of equivalence classes	$\preceq$ **is WQO** $s_0, s_1, s_2, \ldots$ $\exists i < j.\ s_i \preceq s_j$

Transition Systems $(S, \longrightarrow)$

- S: (infinite) set of states
- $\longrightarrow \subseteq S \times S$: transition relation

---

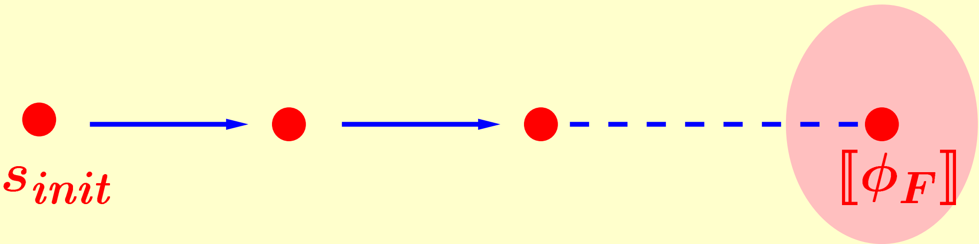
Constraints

- ϕ: constraint
- $[\![\phi]\!] \;\subseteq\; S$

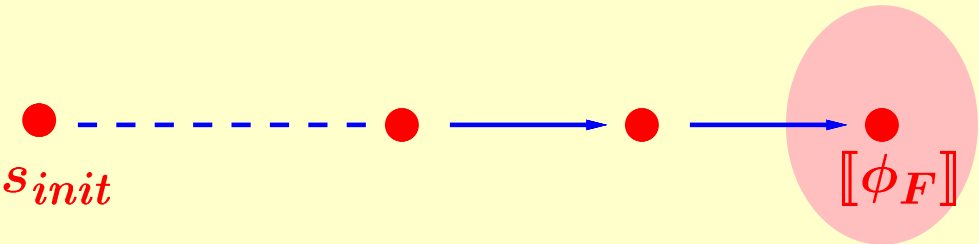
Reachability

- **Instance**
  - s_{init}: initial state
  - ϕ_F: final states
- **Question**
  - ϕ_F reachable from s_{init}?

Forward Reachability Analysis



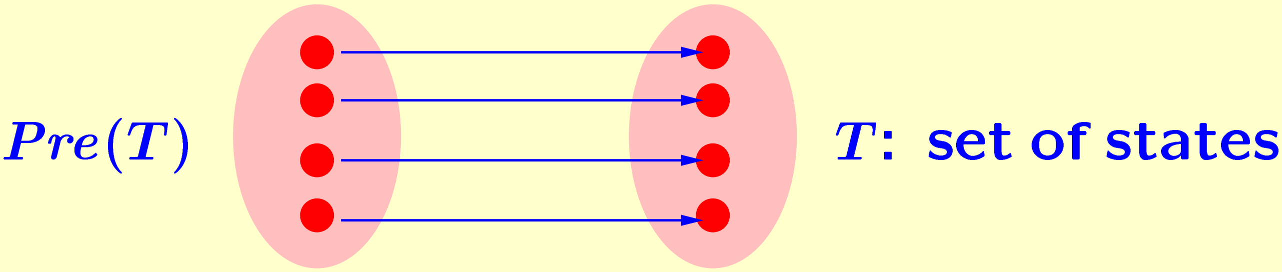
Backward Reachability Analysis



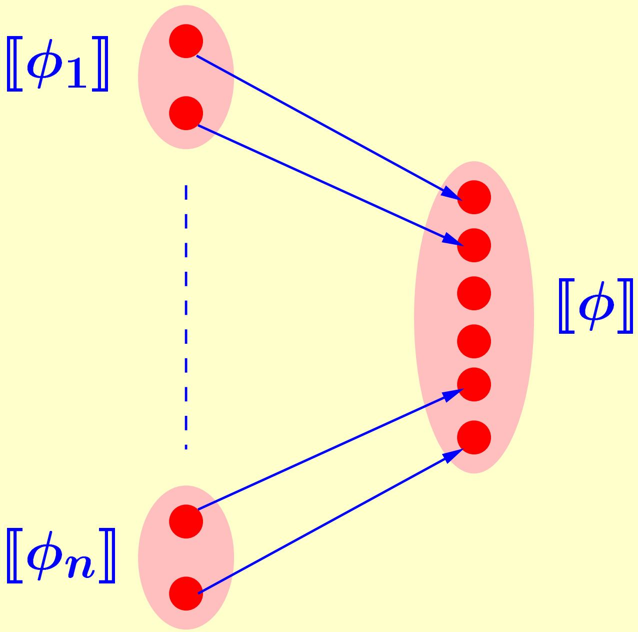
Forward vs. Backward

	finite-state	∞-state
Symmetry	Yes	No
Termination	Guaranteed	Not Guar.

$\boldsymbol{Pre}$



$\boldsymbol{Pre}$ $\quad$ ϕ: **constraint**



$$\boldsymbol{Pre}(\phi) = \{\phi_1, \ldots, \phi_n\}$$

$$\boldsymbol{Pre}([\![\phi]\!]) = [\![\phi_1]\!] \cup \cdots \cup [\![\phi_n]\!]$$

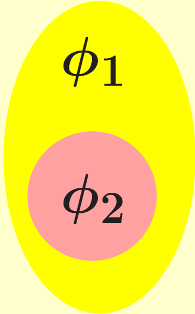
Finite sets of Constraints

- $\psi = \{\phi_1, \ldots, \phi_n\}$

- $[\![\psi]\!] = [\![\phi_1]\!] \cup \cdots \cup [\![\phi_n]\!]$

- $Pre(\psi) = Pre(\phi_1) \cup \cdots \cup Pre(\phi_n)$

Entailment $\sqsubseteq$



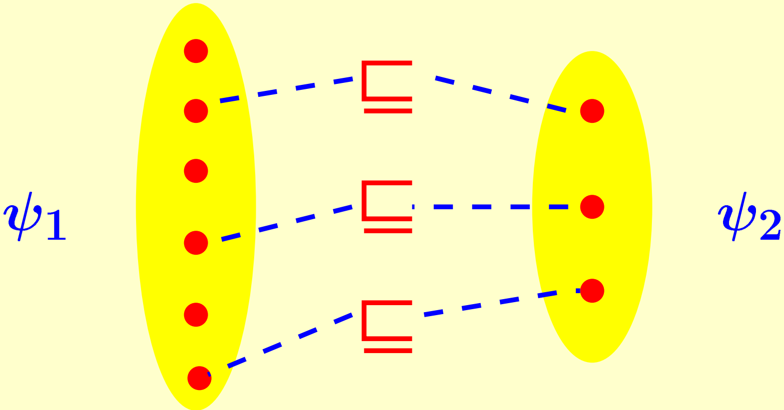
- ϕ_1, ϕ_2 : constraints

  $\phi_1 \sqsubseteq \phi_2$ **iff** $[\![\phi_2]\!] \subseteq [\![\phi_1]\!]$

- ψ_1, ψ_2: finite sets of constraints

  $\psi_1 \sqsubseteq \psi_2$ **iff**

  $\forall \phi_2 \in \psi_2. \quad \exists \phi_1 \in \psi_1. \quad \phi_1 \sqsubseteq \phi_2$



- $\psi_1 \sqsubseteq \psi_2$ **implies** $[\![\psi_2]\!] \subseteq [\![\psi_1]\!]$

Backward Reachability Algorithm

ϕ_F: final states $\quad s_{init}$:initial state

ψ_0, ψ_1, ψ_2, ψ_3, $\ldots$

$\psi_0 = \{\phi_F\} \qquad \psi_{i+1} = \psi_i \cup Pre(\psi_i)$

ψ_i: ϕ_F reachable within i steps

$\psi_0 \sqsupseteq \psi_1 \sqsupseteq \psi_2 \sqsupseteq \quad \cdots \quad \sqsupseteq \psi_j \sqsupseteq \psi_{j+1}$

Termination: $\psi_j \sqsubseteq \psi_{j+1}$

Reachability: $s_{init} \in [\![\psi_j]\!]$

Needed properties

1. Computability of Membership

2. Computability of Entailment

3. Computability of Pre

4. Termination: $\sqsubseteq$ is WQO

Well Quasi-Ordering (WQO)

$(A, \sqsubseteq)$ is WQO if

for each: $a_0,\ a_1,\ a_2,\ \ldots\ \in A$:
$\exists\ i,\ j.\ \ (i < j)$ and $(a_i \sqsubseteq a_j)$

WQO: Simple Example

$(Nat, \leq)$ is WQO:

$x_0,\ x_1,\ x_2,\ \ldots$: natural numbers
$\exists\ i,\ j.\ \ (i < j)$ and $(x_i \leq x_j)$

Properties OF WQO

Finite Sets

$(A, =)$ **WQO, if** A **finite**

$a_1\ a_2\ a_3\ b\ a_4\ a_5\ a_6\ b\ a_7\ \cdots$

Strings

if $(A, \sqsubseteq)$ **WQO then** $(A^*, \sqsubseteq^*)$ **WQO**

$$\begin{array}{cccccccccc}
w_1 & : & & a_1 & & a_2 & & a_3 & \\
\sqsupseteq^* & & & \sqsupseteq & & \sqsupseteq & & \sqsupseteq & \\
w_2 & : & b_1 & b_2 & b_3 & b_4 & b_5 & b_6 & b_7
\end{array}$$

$$w_1 = a_1 a_2 \cdots a_m \sqsubseteq^* b_1 b_2 \cdots b_n = w_2$$

iff

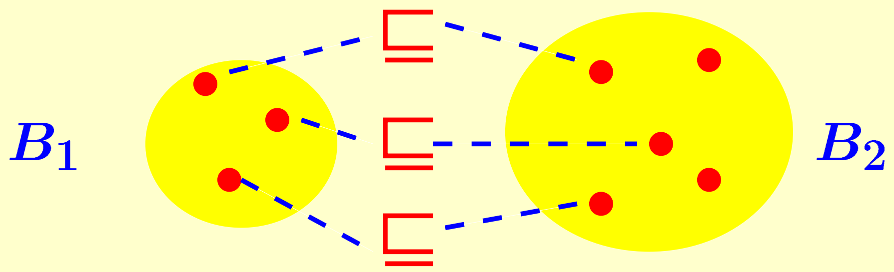
$\exists h : \{1, \ldots, m\} \mapsto \{1, \ldots, n\}$:

- h **injection**
- $i < j \Longrightarrow h(i) < h(j)$
- $a_i \sqsubseteq b_{h(i)}$ **for all** $i : 1 \leq i \leq m$

Bags (Clusters)

if $(A, \sqsubseteq)$ **WQO** **then** $(Bags(A), \sqsubseteq^B)$ **WQO**

$B_1 \sqsubseteq^B B_2$:



Similar results for finite sets, vectors, lists, etc.

Methodology

1. start from simple constraints: finite domains

2. build more complicated data structures: strings, bags, sets, lists, vectors, etc.

Examples – $(A, \sqsubseteq)$

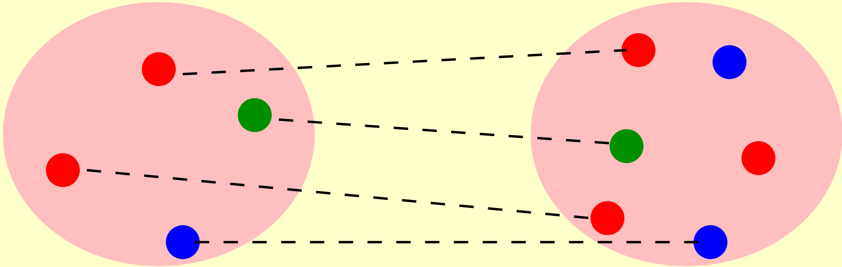
Strings over a Finite Set – $(A^*, \preceq)$

- A: finite alphabet $\sqsubseteq$: equality
- $w_1 \preceq w_2$: w_1 a (not necessarily contiguous) substring of w_2
- e.g. $ab \preceq xaybz$

Lists of Natural Numbers

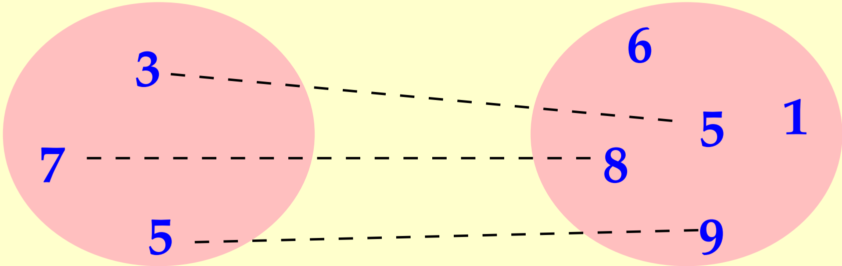
$$
\begin{array}{cccccccl}
 & 5 & & 2 & & 7 & & \leftarrow \ell_1 \\
 & \text{|}\!\wedge & & \text{|}\!\wedge & & \text{|}\!\wedge & & \\
3 & 7 & 1 & 4 & 2 & 8 & & \leftarrow \ell_2
\end{array}
\qquad \ell_1 \sqsubseteq^* \ell_2
$$

Bags over a Finite Alphabet



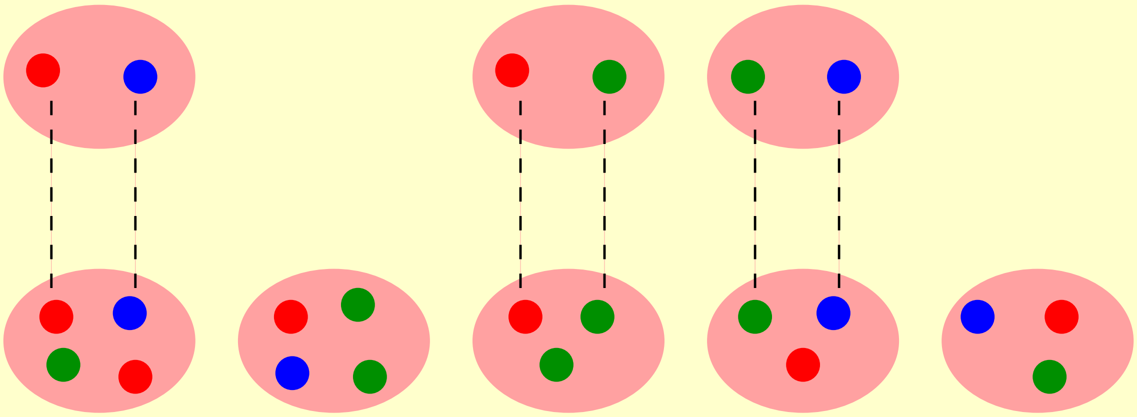
$B_1 \quad \sqsubseteq^* \quad B_2$

Sets of Natural Numbers

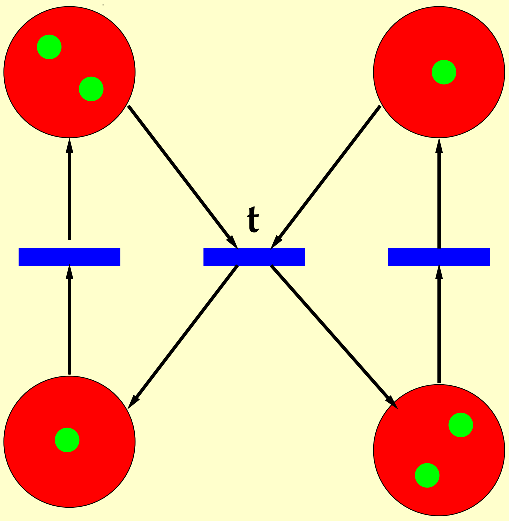


$S_1 \quad \sqsubseteq^* \quad S_2$

Lists of Bags



Petri Nets



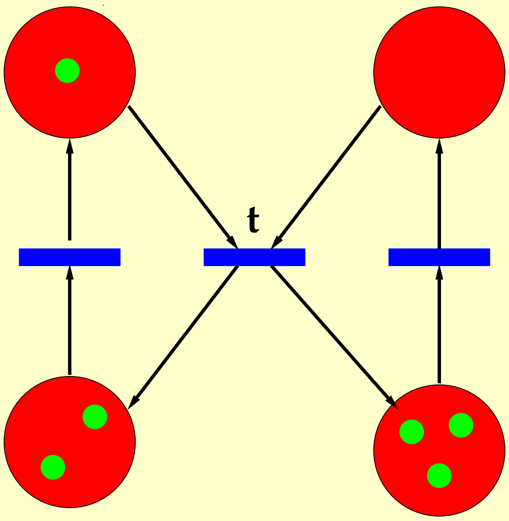
P : set of places

T : set of transitions

state=marking

$M(p)$: no. of tokens in p

Firing t:



Constraints

$\phi(p)$: minimum no. of tokens in p

Membership

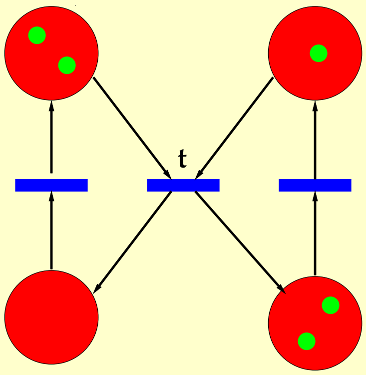
$M \in [\![\phi]\!]$ **iff** $\phi(p) \leq M(p)$, for all $p \in P$

if $M' \geq M$ **then** $M' \in [\![\phi]\!]$

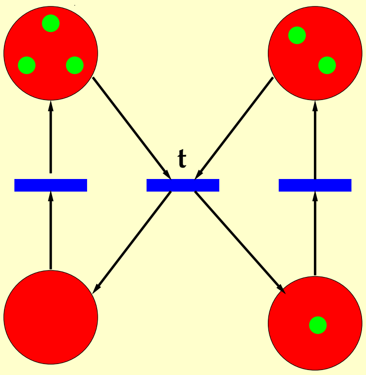
Entailment

$\phi_1 \sqsubseteq \phi_2$ **iff** $\phi_1(p) \leq \phi_2(p)$, for all $p \in P$

Pre



firing
t
backwards



WQO

Constraint:

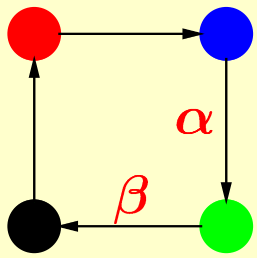
$\phi = (\phi(p_1), \ldots, \phi(p_n))$

vector of natural numbers (WQO)

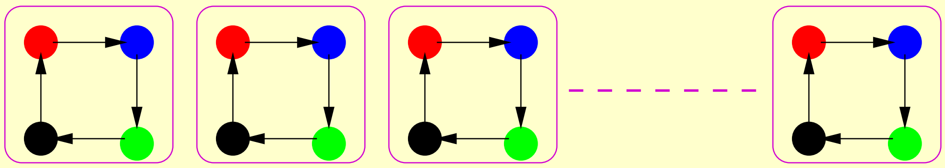
Parametrized System of Finite-State Processes (German and Sistla)

Finite-state process $\rightarrow$

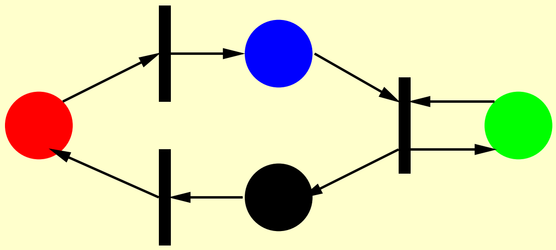
synchronize: α, β



Parametrized system



Petri net representation $\rightarrow$

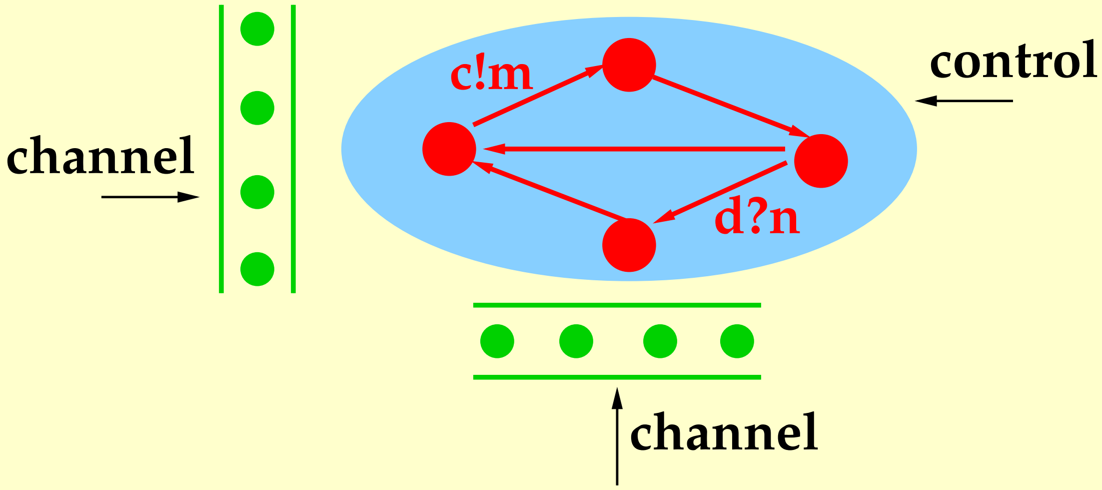


Applications

Broadcast Protocols Esparza, Finkel, Mayr

Cache Coherence Protocols Delzanno

Lossy Channel Systems



Model

- Control Part: finite LTS
- channels: FIFO, unbounded, lossy

Facts

- infinite state space
- perfect channels=Turing machine

Motivation

- communication protocols

Assumption: only one channel

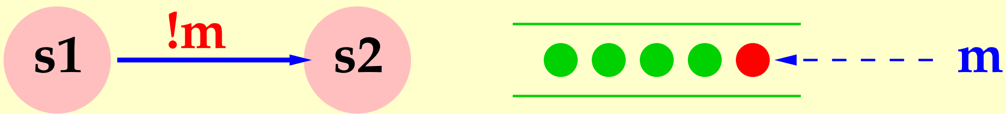
State $\quad M$: channel alphabet

$$(s \quad , \quad w)$$
$$\uparrow \qquad \uparrow$$

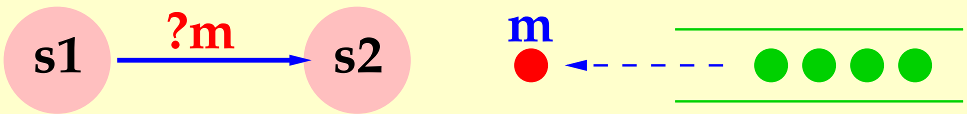
control state $\qquad$ channel state, $w \in M^*$

Transitions

- Send:

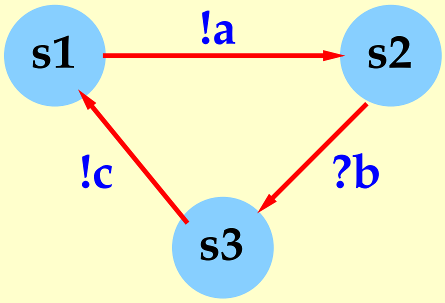


- Receive:



- messages may be lost

Example



$$(s_1, cb) \longrightarrow (s_2, ba) \longrightarrow (s_3, a) \longrightarrow (s_1, ac)$$

Constraints

- $\underset{\substack{\uparrow \\ \text{constraint}}}{\phi} = (\underset{\substack{\uparrow \\ \text{control} \\ \text{state}}}{s} , \underset{\substack{\uparrow \\ \text{channel} \\ \text{state}}}{w})$

- $(s', w') \in [\![\phi]\!]$ iff

  $s' = s$ and $w \preceq w'$

- **Example:** $\phi = (s_1, ab)$

  $(s_1, xaybz) \in [\![\phi]\!]$ $(s_2, xaybz) \notin [\![\phi]\!]$
  $(s_1, xay) \notin [\![\phi]\!]$

Entailment

$(s_1, w_1) = \phi_1 \sqsubseteq \phi_2 = (s_2, w_2)$ iff

$s_1 = s_2$ and $w_1 \preceq w_2$

$Pre(\phi)$ $\qquad\qquad \phi = (s, w)$

- **if** $s' \xrightarrow{!m} s$ **and** $w = w' \bullet m$

  then $(s', w') \in Pre(\phi)$

- **if** $s' \xrightarrow{!m} s$ **and** $last(w) \neq m$

  then $(s', w) \in Pre(\phi)$

- **if** $s' \xrightarrow{?m} s$

  then $(s', m \bullet w) \in Pre(\phi)$

Example $\qquad\qquad \phi = (s_2, ab)$

$s1 \xrightarrow{!b} s2$ - - - - - - (s_1, a)

$s1 \xrightarrow{!d} s2$ - - - - - - (s_1, ab)

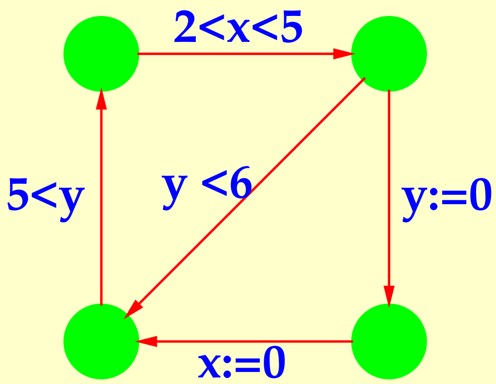
$s1 \xrightarrow{?d} s2$ - - - - - - (s_1, dab)

WQO of constraint system

constraint: (s, w)

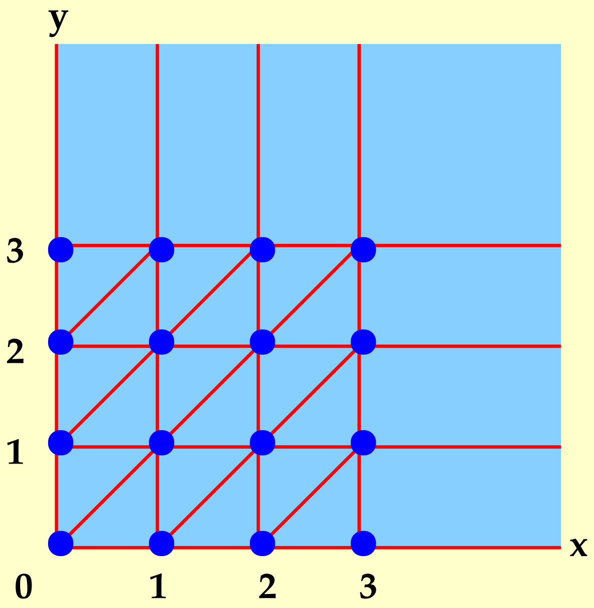
- control states WQO
  - equality on finite alphabet
- channel states WQO
  - substring relation $\preceq$
- so, constraints WQO
  - vectors of WQOs

Timed Automata



- finite-state automaton
- finite set of real-valued clocks
- state: (q, V)
  - q: control state
  - V: clock state = clock values
- $cmax$: maximum occurring natural number

Constraint System - Regions

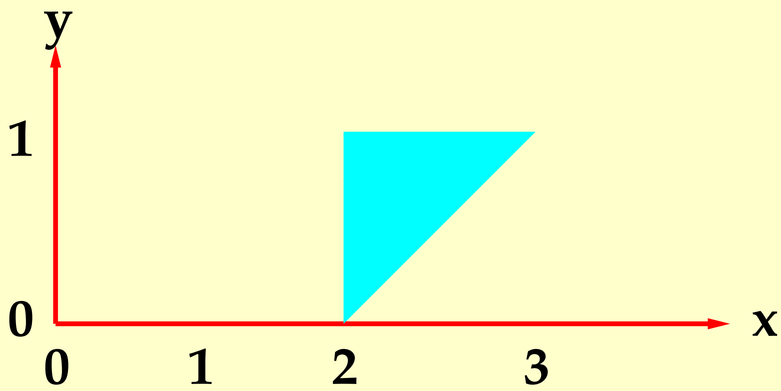


two clocks: x, y

$cmax = 3$

- Equivalence relation on clock values
- clock values equivalent if
  - same integral parts
  - same ordering of fractional parts
  - clock values beyond $cmax$ equivalent
- finite number of regions

Example

- $R:$ 

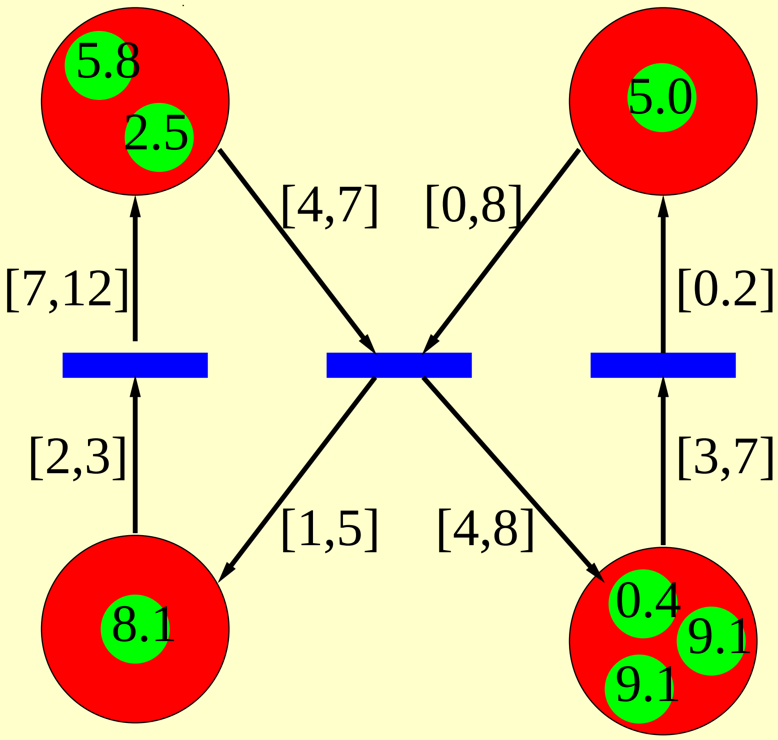
- $V(x) = 2.4,\ V(y) = 0.8$

- $V \in [\![R]\!]$

WQO

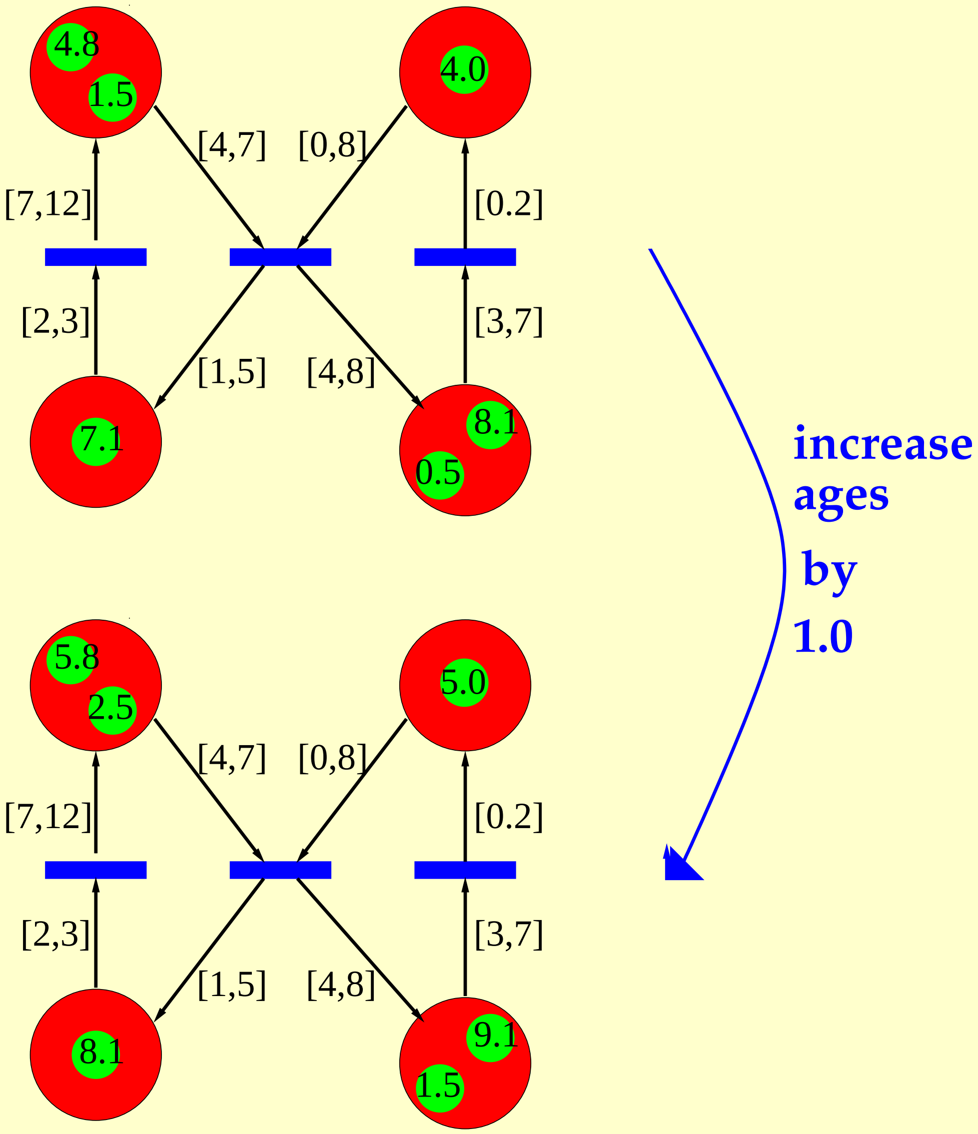
finite number of regions

Timed Petri Nets

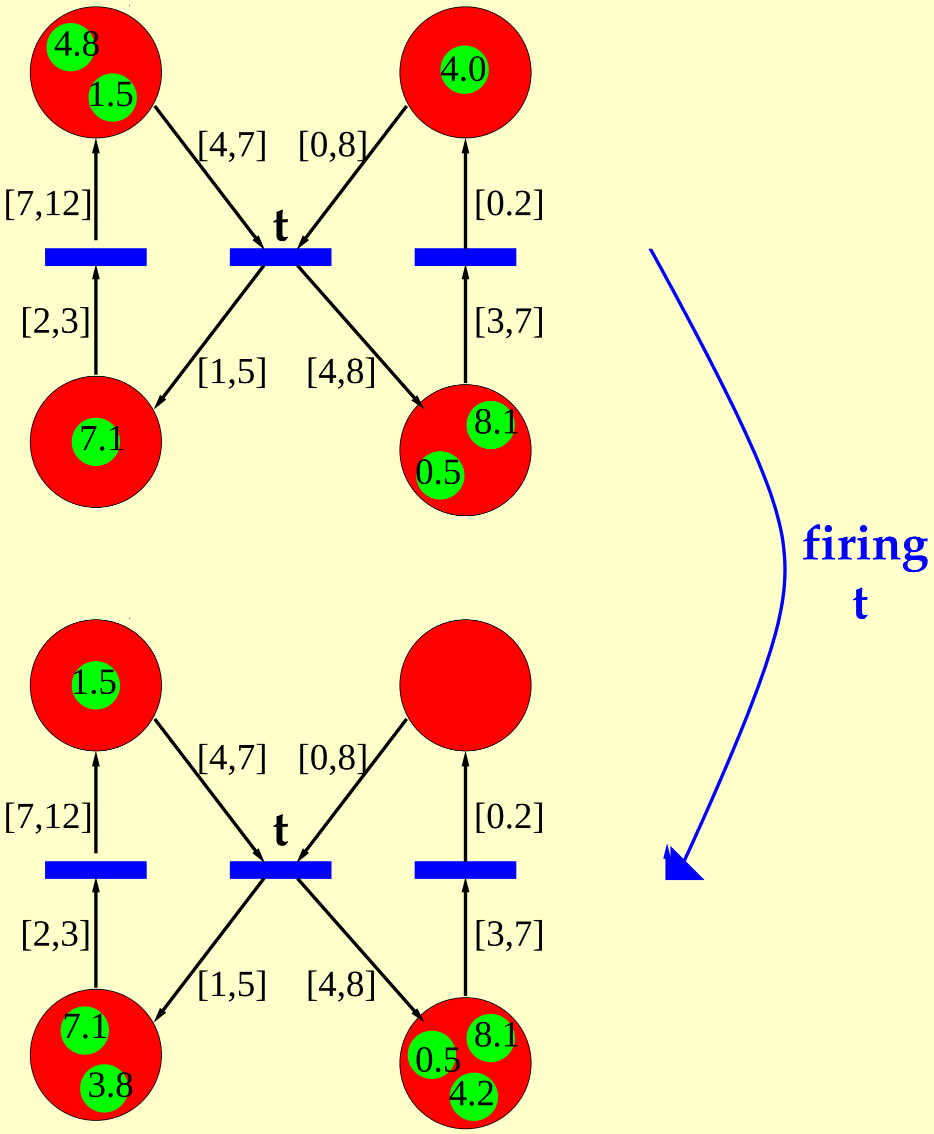


- Tokens have “ages”: Real numbers
- Conditions on ages: Intervals
- cannot be modelled as timed automata
- Motivation:
  Parametrized timed processes

Timed Transitions



Discrete Transitions



Constraints – Existential Regions

$$ER:$$
$$\exists x_1, \ldots, x_n.\ R(x_1, \ldots, x_n)\ \wedge\ P(x_1, \ldots, x_n)$$

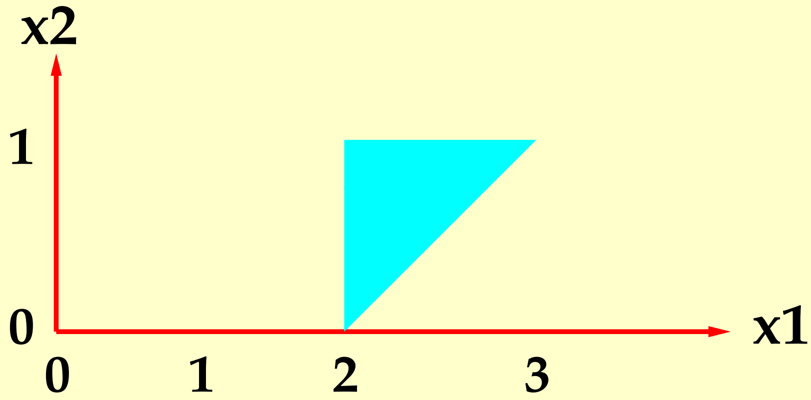
- $x_1, \ldots, x_n$: tokens
- $R(x_1, \ldots, x_n)$: region
- $P(x_1, \ldots, x_n)$: assign tokens to places

- ER: minimal set of conditions

Existential Regions – Example

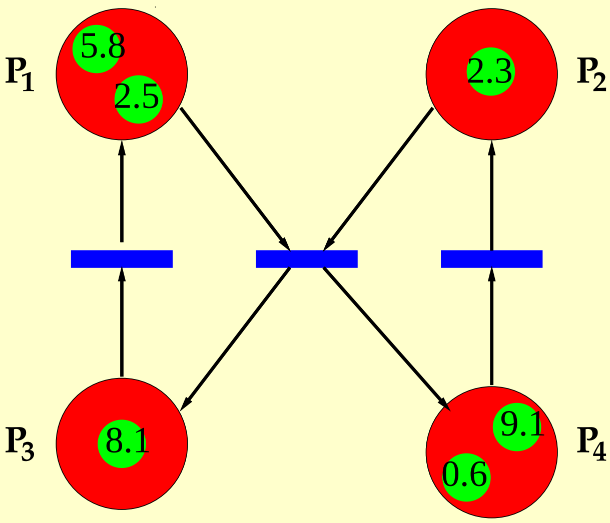
- $ER: \quad \exists x_1, x_2.\ R(x_1, x_2) \ \wedge \ P(x_1, x_2)$

- $R(x_1, x_2)$



- $P(x_1, x_2)$: x_1 in p_2, x_2 in p_4

- $M:$



$\in \ [\![ER]\!]$

- if $M \leq M'$ then $M' \in [\![ER]\!]$

Regions vs. Existential Regions

- Regions $\longrightarrow$ Equivalence Classes

  Existential Regions $\longrightarrow$ Ideals

- Regions $\longrightarrow$ no. of constraints bounded

  Existential Regions $\longrightarrow$ no. of constraints not bounded: WQO?

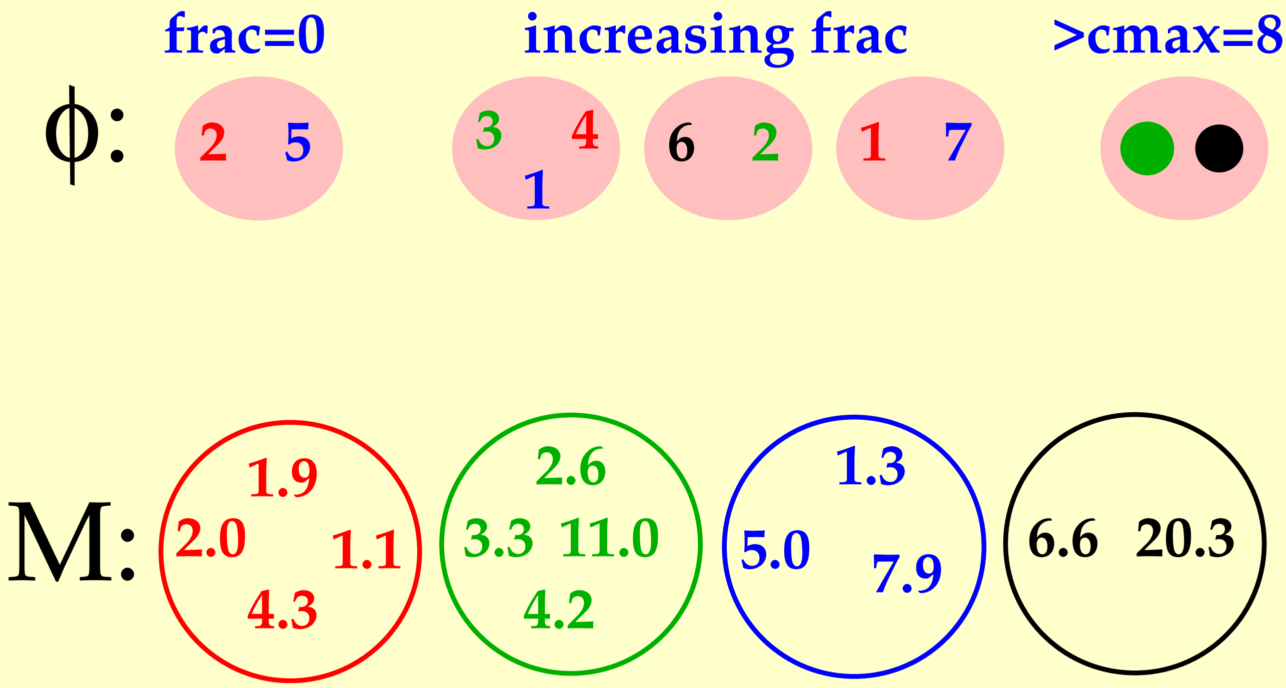
WQO

Existential Region = list of bags over a finite alphabet

- finite alphabet = integral parts of clock values + places of tokens

- ordering of list elements = ordering of fractional parts

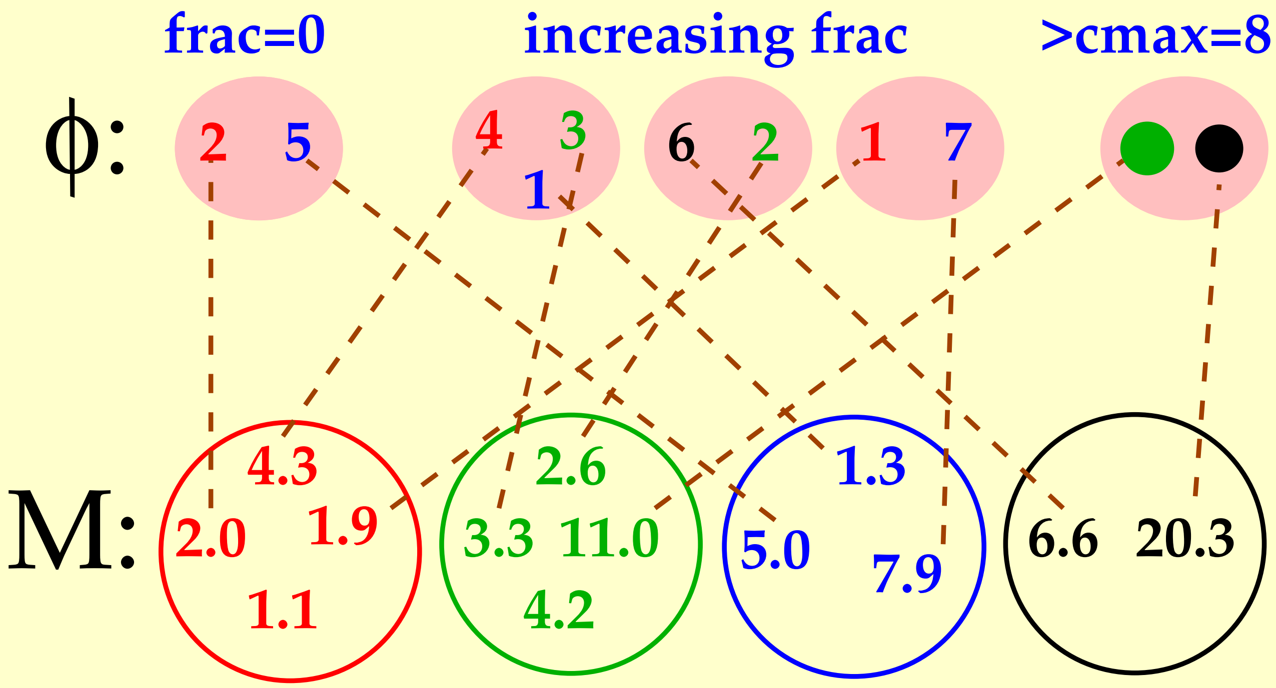
Characterization of Existential Regions



WQO

lists of bags over a finite alphabet !

Characterization of Existential Regions



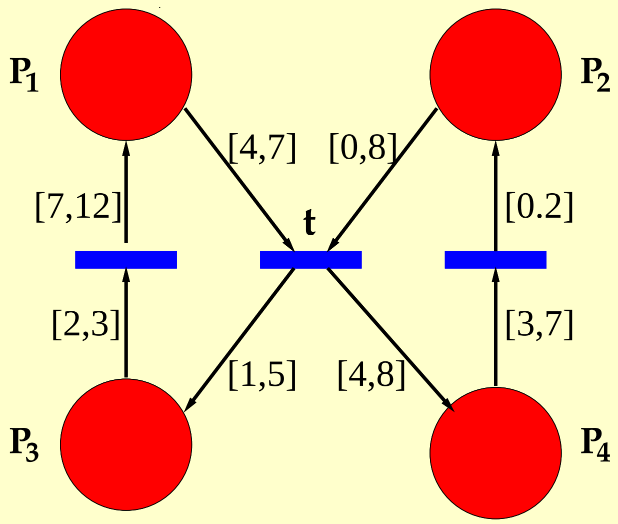
WQO

lists of bags over a finite alphabet !

Constraint Explosion Problem $Pre(\phi)$ large

$ER:$

$\exists x. \left(\begin{array}{c} 3 \text{ --- } 4 \\ x \text{ in } P_3 \end{array} \right)$

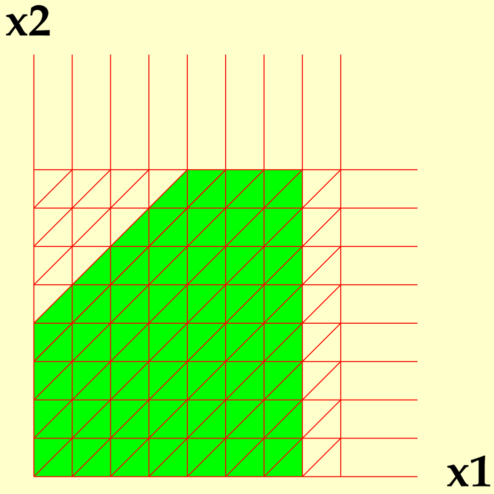


$Pre(ER, t):$

$\exists\ x_1\ x_2\ .$

x_1 in P_1

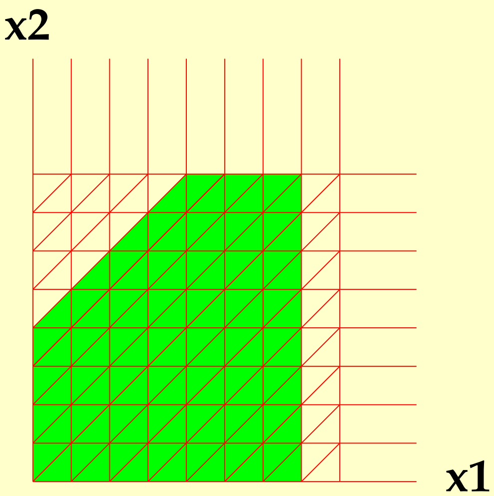
x_2 in P_2



130 existential regions !!

Solution: Use disjunctions of constraints

Zones



Z :

$$0 \leq x_1 \leq 8$$
$$\wedge$$
$$0 \leq x_2 \leq 8$$
$$\wedge$$
$$x_2 - x_1 \leq 4$$

Existential Zones

$$\exists x_1 \cdots x_n .\ P(x_1, \ldots, x_n) \wedge Z(x_1, \ldots, x_n)$$

Existential Zone $\equiv$
Disjunction of Existential Regions

Problem

- WQOs not closed under Disjunction
- Existential Zones WQO?

Better Quasi-Ordering (BQO)

Properties of BQOs

- Each BQO is WQO
- $(A, =)$ is BQO if A finite
- BQO preserved by finite sets, lists, etc.
- BQO preserved by disjunction

Methodology

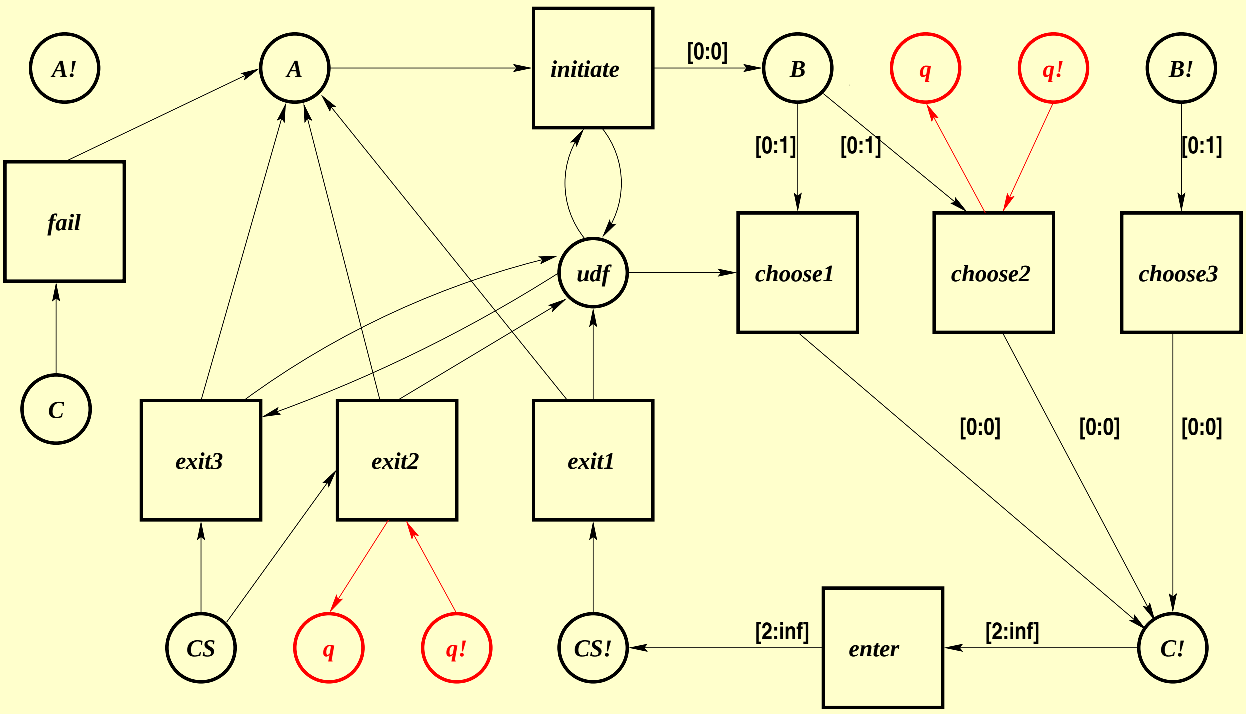
- Build Constraints as in WQOs
- Build new constraints using disjunction

BQO of Existential Zones

- Existential Regions are BQO:
  - Existential regions are lists of bags over a finite domain

- Existential Zone $\equiv$ disjunction of Existential Regions

Example: Fischer's Protocol



Broadcast and Cache Coherence Protocols

Delzanno, Esparza, Podelski

$\phi : x_1 + \cdots + x_n \geq \beta$

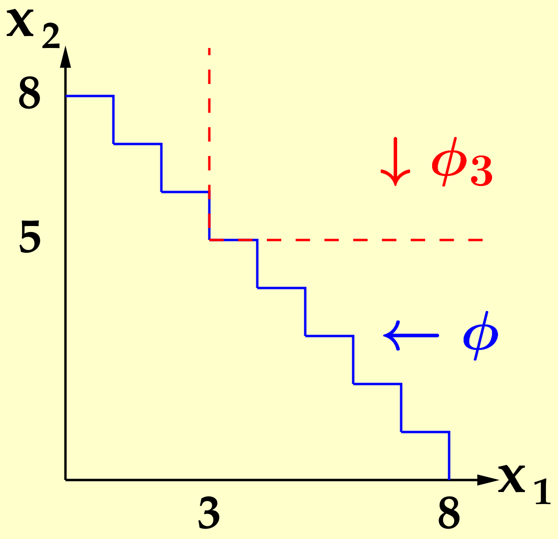
$\phi \equiv \phi_0 \vee \cdots \vee \phi_m$

$\phi_i : (x_1 \geq \alpha_1) \wedge \cdots \wedge (x_n \geq \alpha_n)$

$\phi : x_1 + x_2 \geq 8$

$\phi \equiv \phi_0 \vee \cdots \vee \phi_8$

$$\phi_i : \begin{pmatrix} x_1 \geq i \\ \wedge \\ x_2 \geq 8 - i \end{pmatrix}$$



Constraint System BQO
(and hence WQO)

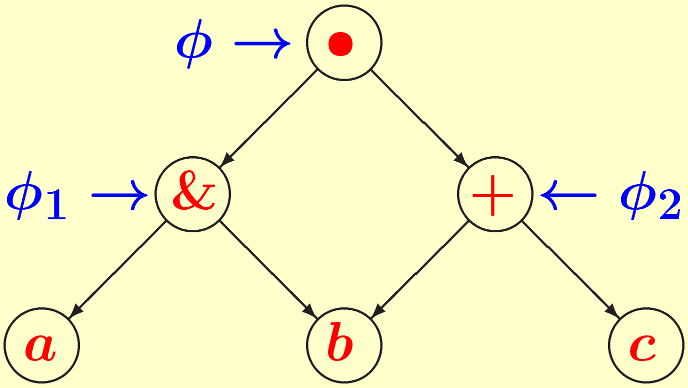
Lossy Channel Systems

$w \in [\![a]\!]$ **iff** $a \preceq w$

$w \in [\![\phi_1 \& \phi_2]\!]$ **iff** $w \in [\![\phi_1]\!]$ **and** $w \in [\![\phi_2]\!]$

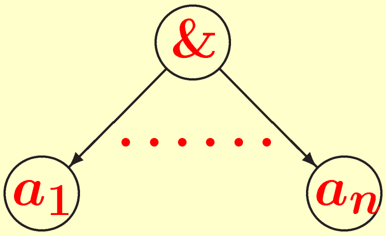
$w \in [\![\phi_1 + \phi_2]\!]$ **iff** $w \in [\![\phi_1]\!]$ **or** $w \in [\![\phi_2]\!]$

$w \in [\![\phi_1 \bullet \phi_2]\!]$ **iff** $w = w_1 \bullet w_2$ **and**
$w_1 \in [\![\phi_1]\!]$ **and** $w_2 \in [\![\phi_2]\!]$



$w \in [\![\phi]\!]$ **iff**

$abb \preceq w$ **or**
$abc \preceq w$ **or**
$bab \preceq w$ **or**
$bac \preceq w$



$w \in [\![\phi]\!]$ **iff**

$b_1 \cdots b_n \preceq w$, **where**
$b_1 \cdots b_n \in perm(a_1 \cdots a_n)$

Forward vs. Backward Reachability Analysis

(s, w): state of lossy channel system

$Pre^*(s, w)$, $Post^*(s, w)$: regular

$Pre^*(s, w)$ computable

$Post^*(s, w)$ not computable

Conclusions of Part 1

- Combining WQOs and symbolic model checking
- Uniform presentation of existing results: e.g. Petri nets, lossy channel systems, timed automata
- New model checking algorithms: e.g. timed Petri nets, broadcast protocols, cache coherence protocols
- efficient algorithms: BQOs