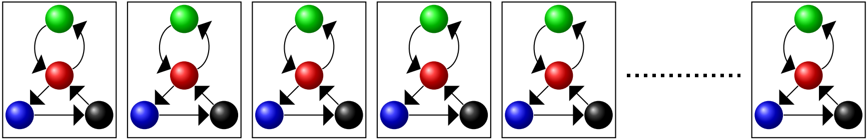
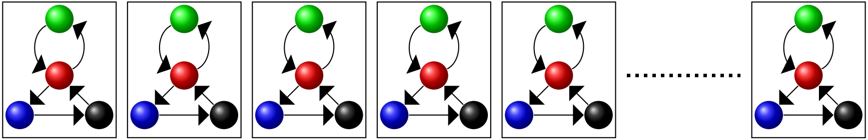
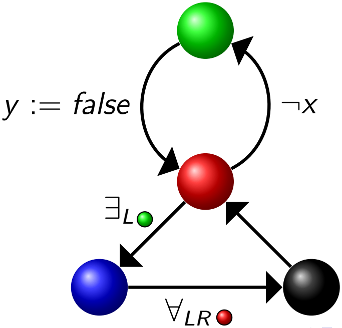
Linearly Ordered Parameterized Systems (of finite-state processes)

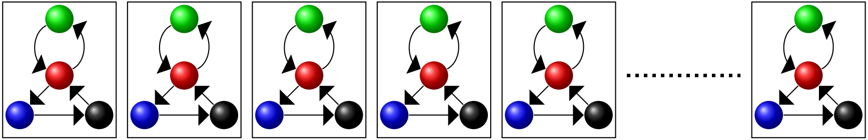


Linearly Ordered Parameterized Systems (of finite-state processes)





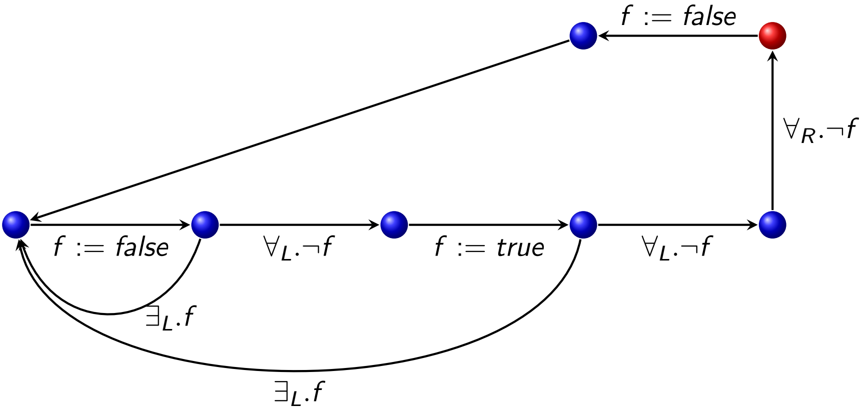
Linearly Ordered Parameterized Systems (of finite-state processes)

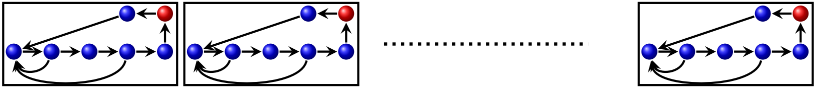


Remarks

- Infinite-state system:
  - unbounded number of processes.
  - Parameterized Verification: verify correctness regardless of the number of processes.
- Problem undecidable in general.
  - Challenge: find abstractions which work often.

Parameterized Burns' Mutual Exclusion Protocol





Burns Algorithm

Instance

Q: $q_1, \ldots, q_7$
X: $f \in \mathcal{B}$
T:

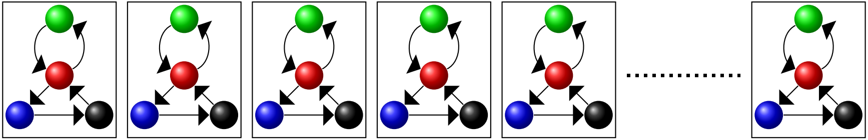
$$t_1 : \left[\begin{array}{c} q_1 \\ \mathbf{tt} \to f = \mathbf{ff} \\ q_2 \end{array}\right] \quad t_2 : \left[\begin{array}{c} q_2 \\ \exists_L f \to \{\} \\ q_1 \end{array}\right] \quad t_3 : \left[\begin{array}{c} q_2 \\ \forall_L \neg f \to \{\} \\ q_3 \end{array}\right]$$

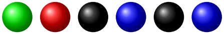
$$t_4 : \left[\begin{array}{c} q_3 \\ \mathbf{tt} \to f = \mathbf{tt} \\ q_4 \end{array}\right] \quad t_5 : \left[\begin{array}{c} q_4 \\ \exists_L f \to \{\} \\ q_1 \end{array}\right] \quad t_6 : \left[\begin{array}{c} q_4 \\ \forall_L \neg f \to \{\} \\ q_5 \end{array}\right]$$

$$t_7 : \left[\begin{array}{c} q_5 \\ \forall_R \neg f \to \{\} \\ q_6 \end{array}\right] \quad t_8 : \left[\begin{array}{c} q_6 \\ \mathbf{tt} \to f = \mathbf{ff} \\ q_7 \end{array}\right] \quad t_9 : \left[\begin{array}{c} q_7 \\ \mathbf{tt} \to \{\} \\ q_1 \end{array}\right]$$

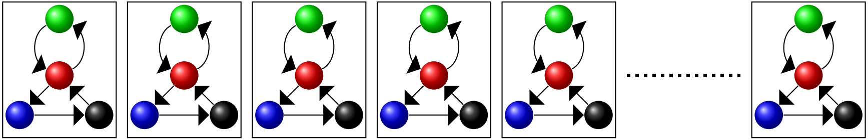
Initial Process State u_{init}: $q_1, f \mapsto \mathbf{ff}$
Final Constraints Φ_F: $q_6 q_6$

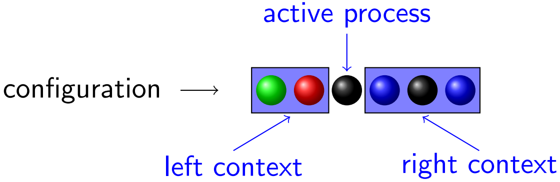
Configurations



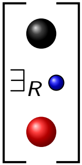
configuration ⟶ 

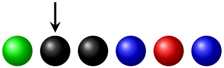
Configurations



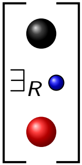


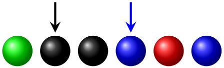
Existential Global Transitions



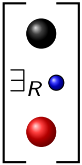


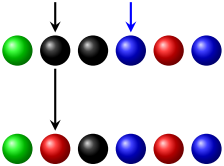
Existential Global Transitions



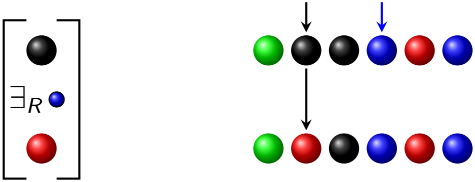


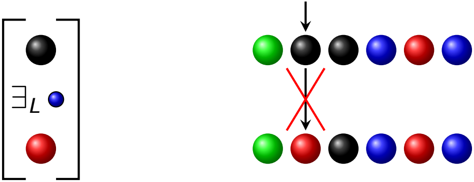
Existential Global Transitions



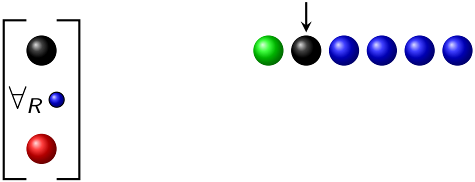


Existential Global Transitions

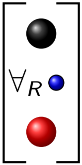


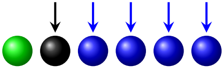


Universal Global Transitions

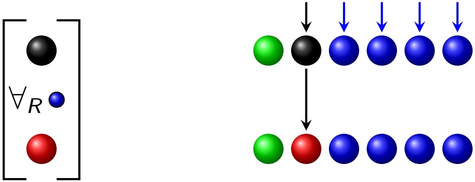


Universal Global Transitions

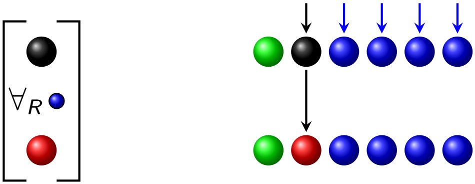


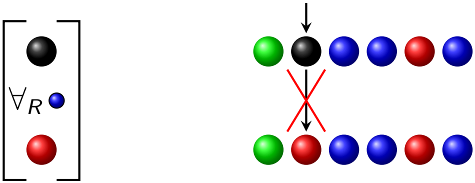


Universal Global Transitions

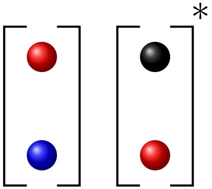


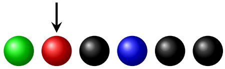
Universal Global Transitions



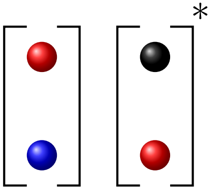


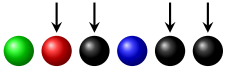
Broadcast Transitions



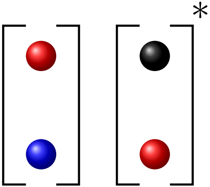


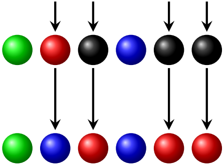
Broadcast Transitions





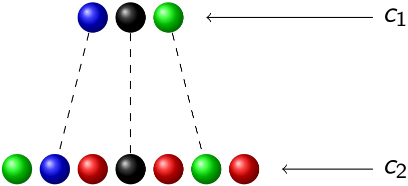
Broadcast Transitions



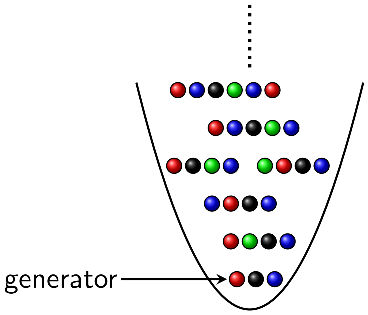


Ordering on Configurations

c_1
$\sqsubseteq$
c_2

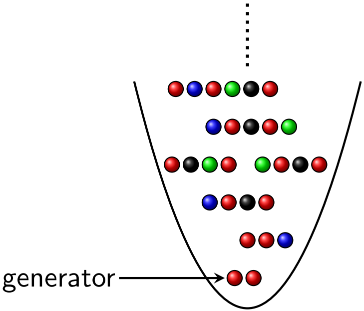


Upward-Closed Sets (UC)

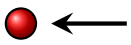


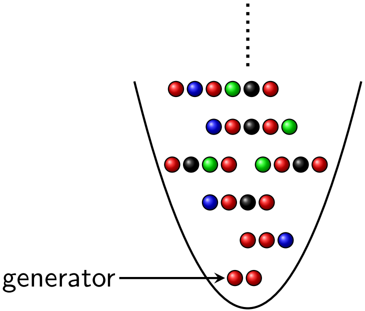
Upward-Closed Sets (UC)

← critical section



Upward-Closed Sets (UC)

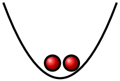
 critical section



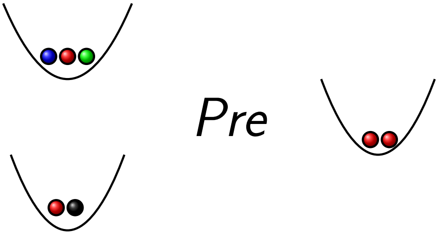
Why UC?

- Bad sets of states are UC
  - safety properties = reachability of UC
- Uniquely characterized by generator
  - simple representation = finite word

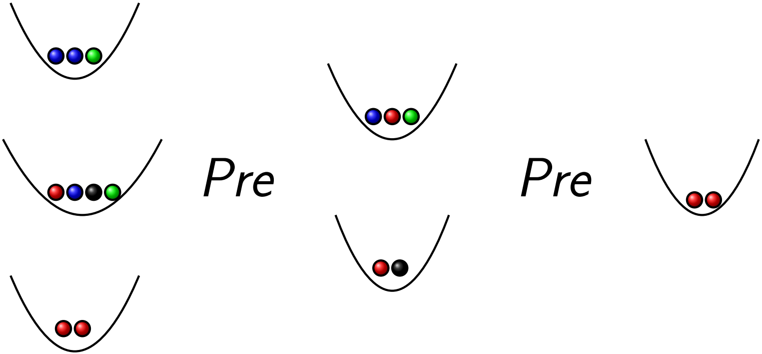
Backward Reachability Analysis (on UC)



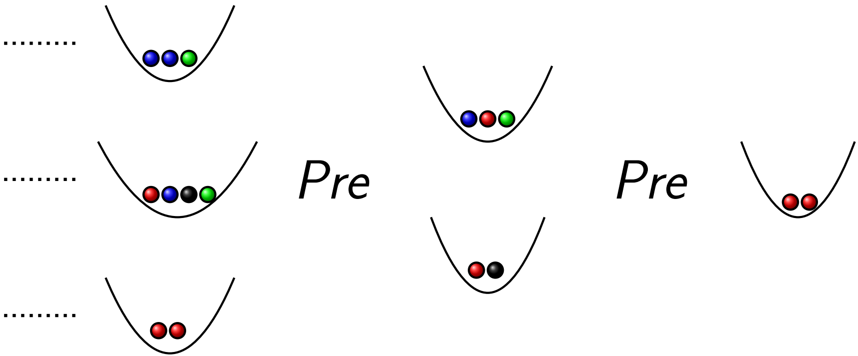
Backward Reachability Analysis (on UC)



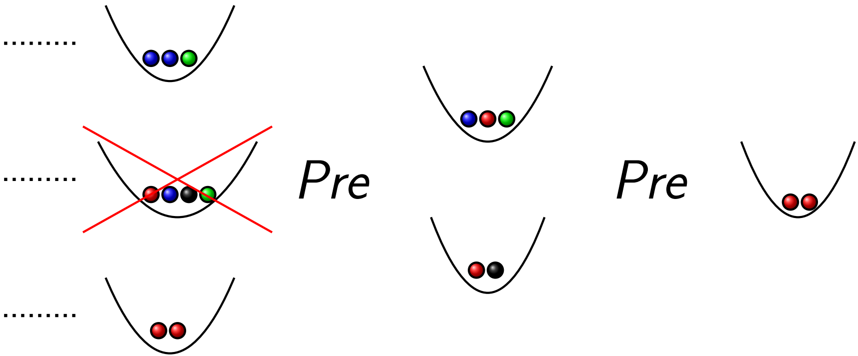
Backward Reachability Analysis (on UC)



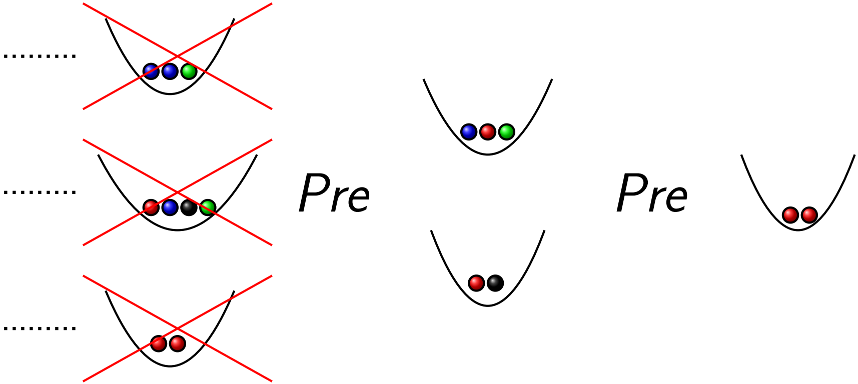
Backward Reachability Analysis (on UC)



Backward Reachability Analysis (on UC)

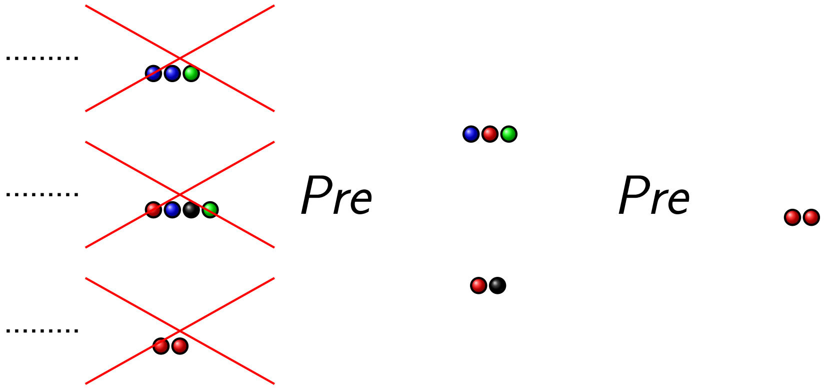


Backward Reachability Analysis (on UC)



Backward Reachability Analysis (on UC)

symbolic representation = finite words.



Required:

UC closed under Pre !!

Monotonicity

$$\begin{array}{ccc} c_1 & \longrightarrow & c_2 \\ \rotatebox{90}{$\sqsubseteq$} & & \\ c_3 & & \end{array}$$

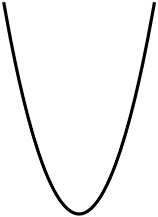
Monotonicity

$$\begin{array}{ccc} c_1 & \longrightarrow & c_2 \\ \rotatebox{-90}{$\sqsubseteq$} & & \rotatebox{-90}{$\sqsubseteq$} \\ c_3 & \longrightarrow & c_4 \end{array}$$

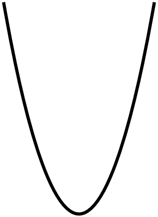
Monotonicity

$$\begin{array}{ccc} c_1 & \longrightarrow & c_2 \\ \sqsubseteq & & \sqsubseteq \\ c_3 & \longrightarrow & c_4 \end{array}$$

Monotonicity implies UC is closed under *Pre*



$Pre(U)$:Upward Closed?

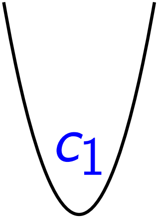


U:Upward Closed

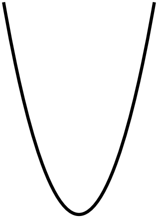
Monotonicity

$$\begin{array}{ccc} c_1 & \longrightarrow & c_2 \\ \sqsubseteq & & \sqsubseteq \\ c_3 & \longrightarrow & c_4 \end{array}$$

Monotonicity implies UC is closed under *Pre*



$Pre(U)$:Upward Closed?

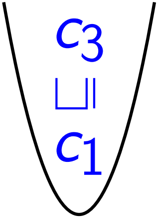


U:Upward Closed

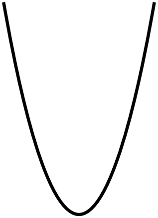
Monotonicity

$$\begin{array}{ccc} c_1 & \longrightarrow & c_2 \\ \sqcap\!\mid & & \sqcap\!\mid \\ c_3 & \longrightarrow & c_4 \end{array}$$

Monotonicity implies UC is closed under *Pre*

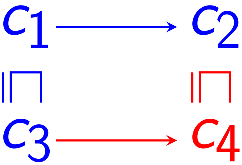


$Pre(U)$:Upward Closed?

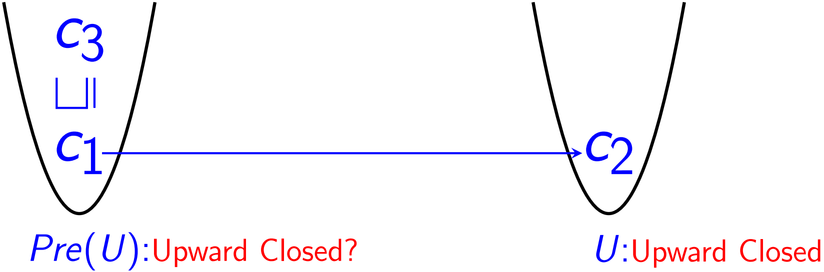


U:Upward Closed

Monotonicity



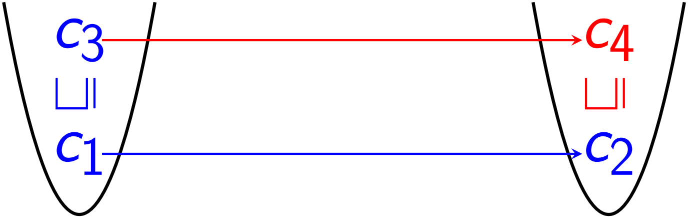
Monotonicity implies UC is closed under *Pre*



Monotonicity

$$
\begin{array}{ccc}
c_1 & \longrightarrow & c_2 \\
\sqsubseteq & & \sqsubseteq \\
c_3 & \longrightarrow & c_4
\end{array}
$$

Monotonicity implies UC is closed under *Pre*

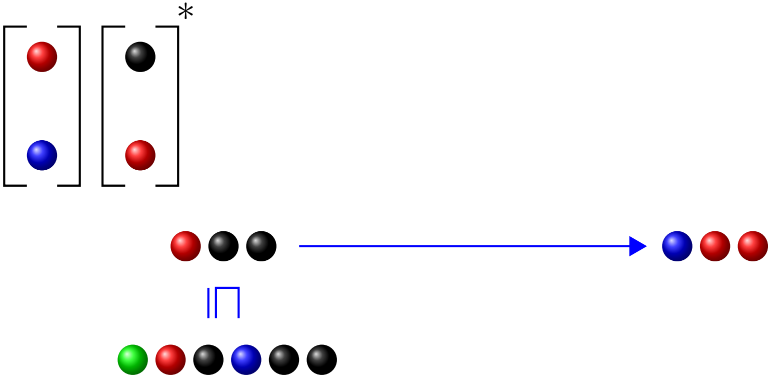


$Pre(U)$:Upward Closed? Yes

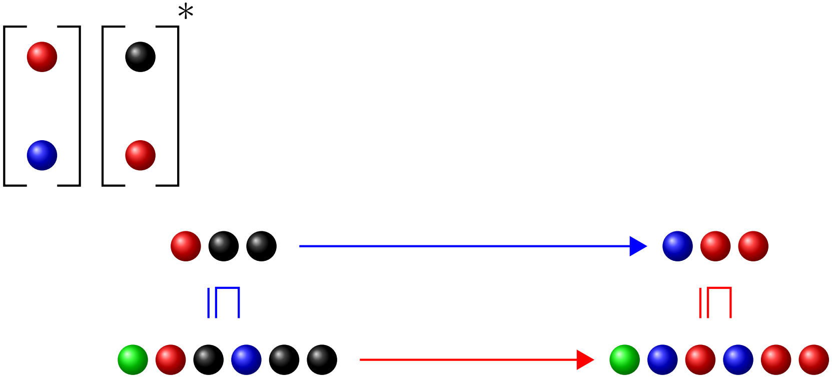
U:Upward Closed

Which transitions are monotonic?

Broadcast Transitions

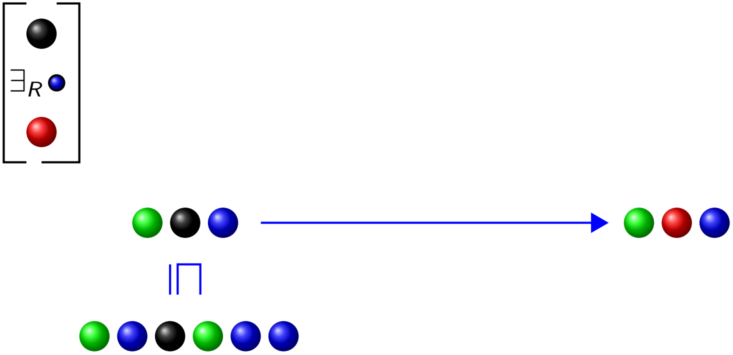


Yes



Which transitions are monotonic?

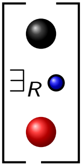
Existential Global Transitions

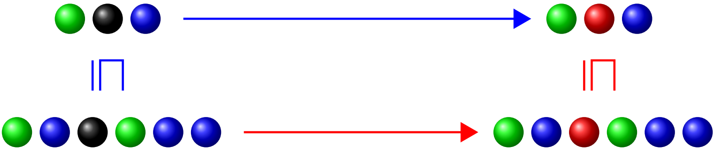


Which transitions are monotonic?

Existential Global Transitions

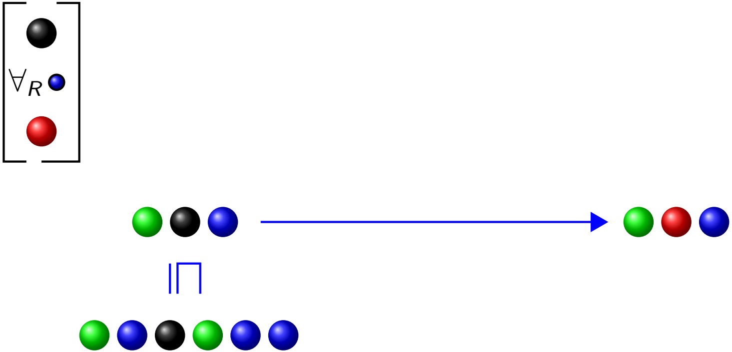
Yes





Which transitions are monotonic?

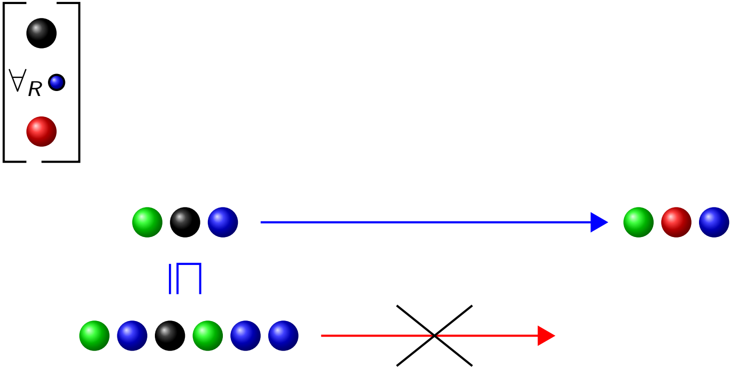
Universal Global Transitions



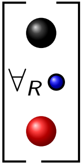
Which transitions are monotonic?

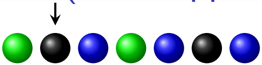
Universal Global Transitions

No

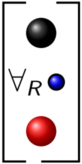


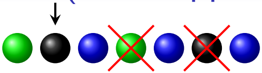
Monotonic Abstraction (Over-Approximation)



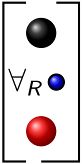


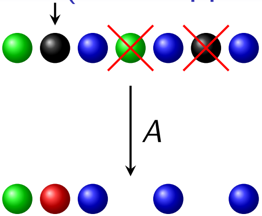
Monotonic Abstraction (Over-Approximation)



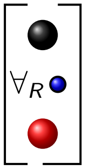


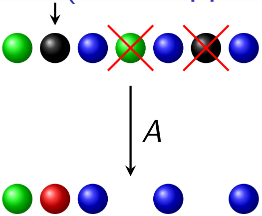
Monotonic Abstraction (Over-Approximation)



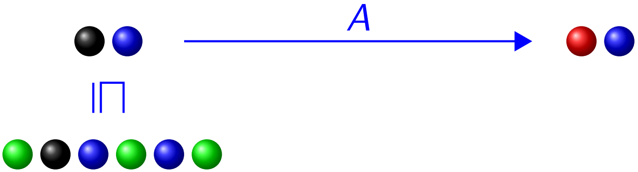


Monotonic Abstraction (Over-Approximation)

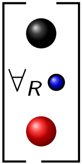


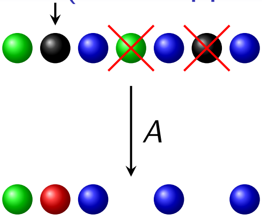


Monotonic?

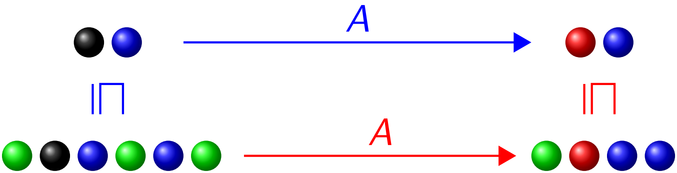


Monotonic Abstraction (Over-Approximation)

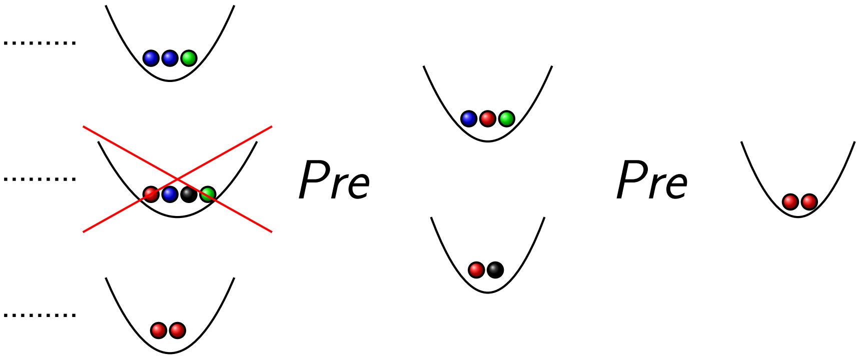




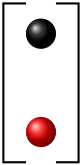
Monotonic? Yes

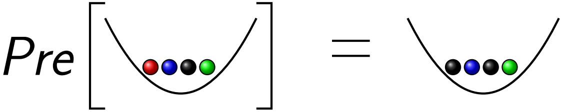


Backward Reachability Analysis on Abstract System

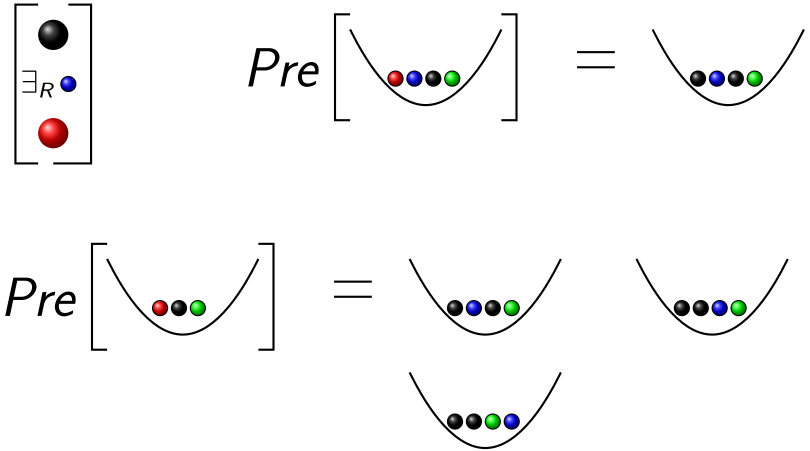


Pre - Local Transitions

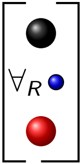


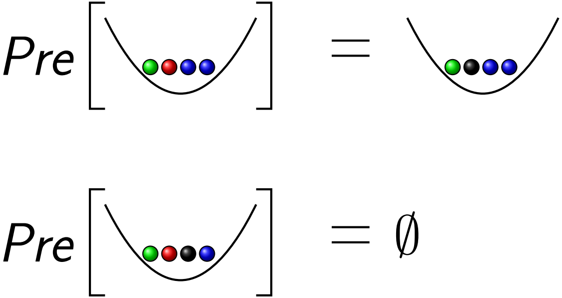


Pre - Existential Global Transitions

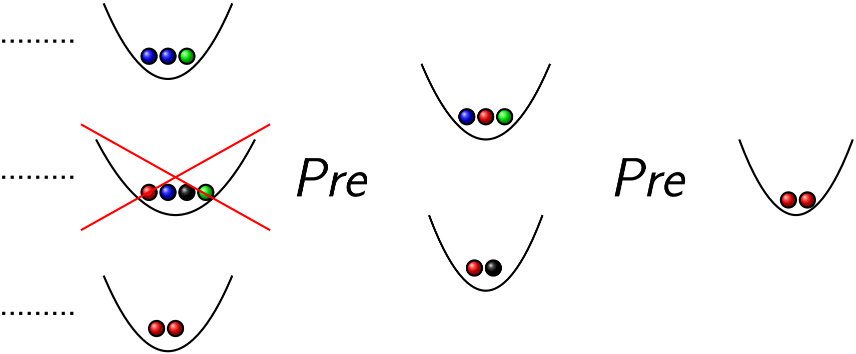


Pre - Universal Global Transitions



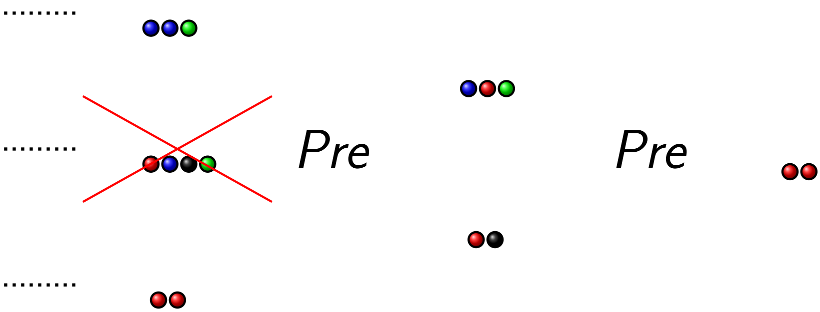


Backward Reachability Analysis on Abstract System



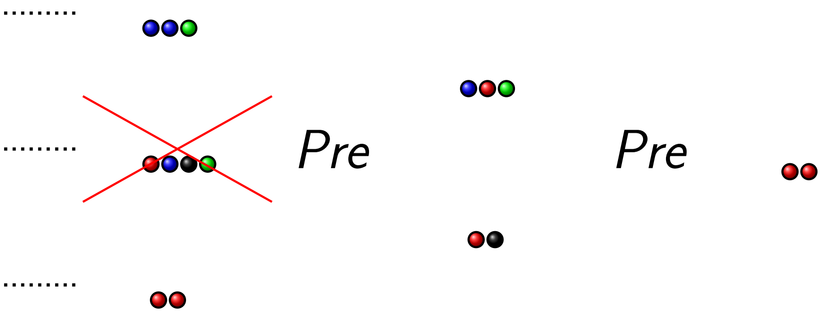
Backward Reachability Analysis on Abstract System

symbolic representation $=$ finite words



Backward Reachability Analysis on Abstract System

symbolic representation = finite words



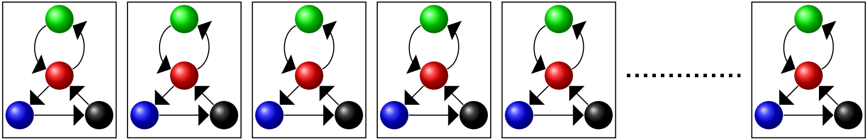
Termination

- Subword relation is a well quasi-ordering.
- Reachability algorithm guaranteed to terminate.

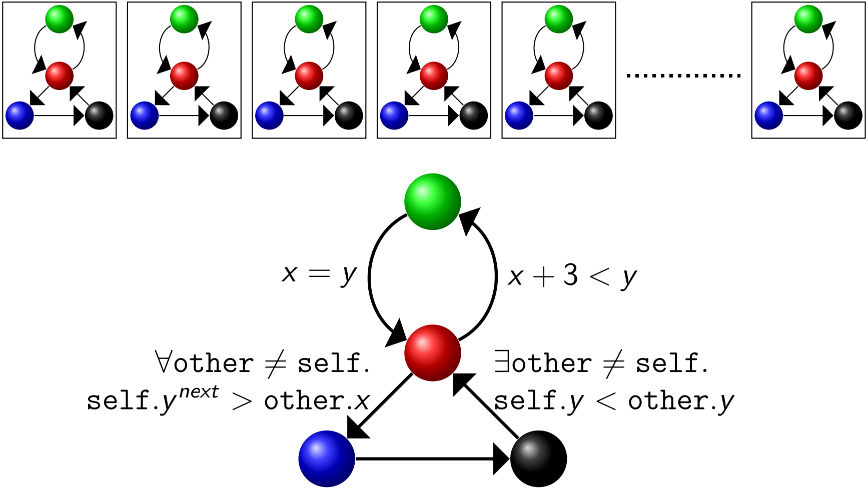
Summary

- Monotonicity allows working with upward closed sets
- Symbolic representation = words:
  - More powerful than finite-state abstraction
  - More powerful than counter abstraction
  - Less heavy than general regular expressions (transducer-based methods, e.g., regular model checking)
- Simple abstraction gives monotonicity
- Works on difficult examples !!

Parameterized Systems with variables



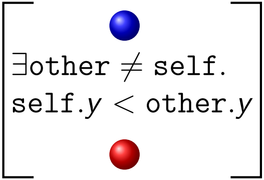
Parameterized Systems with variables

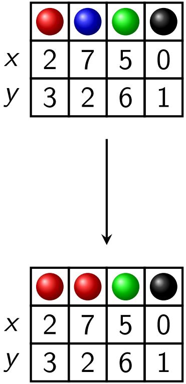


Configurations

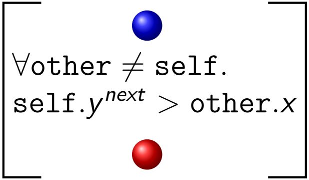
				
x	2	7	5	0
y	3	2	6	1

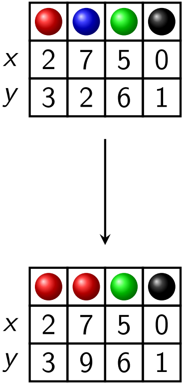
Transitions





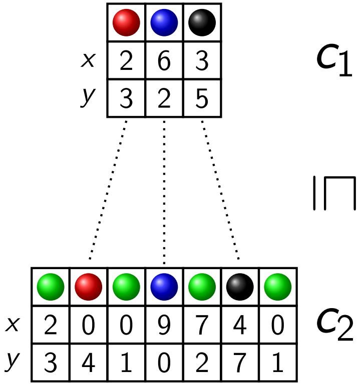
Transitions





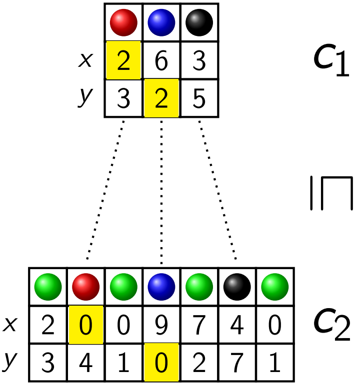
Ordering on Configurations (gap-order)

- Identical control states
- Preserves equality
- Gaps in c_1 $\leq$ Gaps in c_2



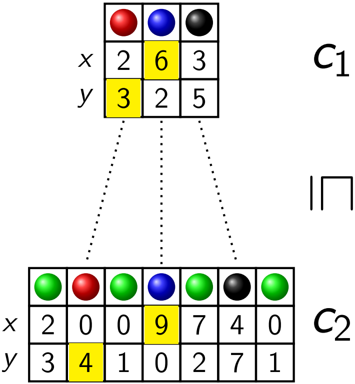
Ordering on Configurations (gap-order)

- Identical control states.
- Preserves equality
- Gaps in c_1 $\leq$ Gaps in c_2

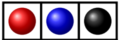


Ordering on Configurations (gap-order)

- Identical control states.
- Preserves equality
- Gaps in c_1 $\leq$ Gaps in c_2



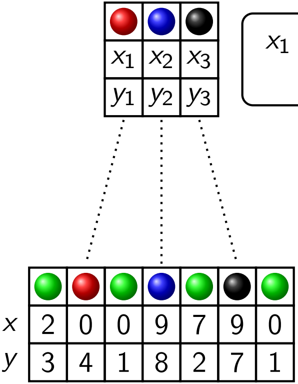
Gap-Order Constraints



x_1	x_2	x_3
y_1	y_2	y_3

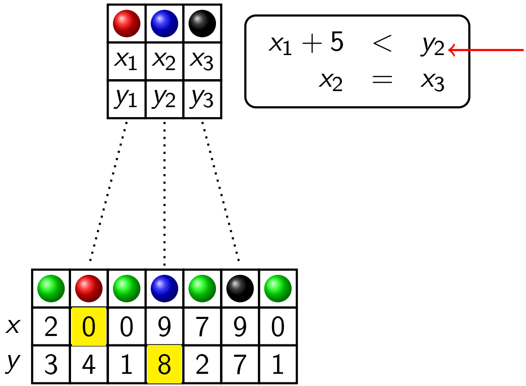
$$\begin{aligned} x_1 + 5 &< y_2 \\ x_2 &= x_3 \end{aligned}$$

Gap-Order Constraints

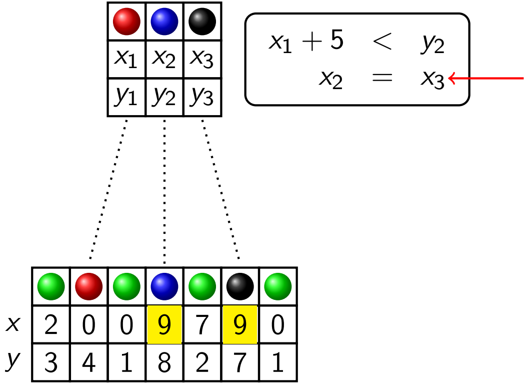


$$\begin{aligned} x_1 + 5 &< y_2 \\ x_2 &= x_3 \end{aligned}$$

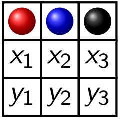
Gap-Order Constraints



Gap-Order Constraints



Gap-Order Constraints

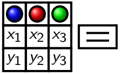


$$\begin{aligned} x_1 + 5 &< y_2 \\ x_2 &= x_3 \end{aligned}$$

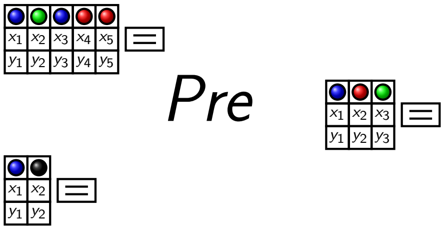
↑

upward closed set of configurations

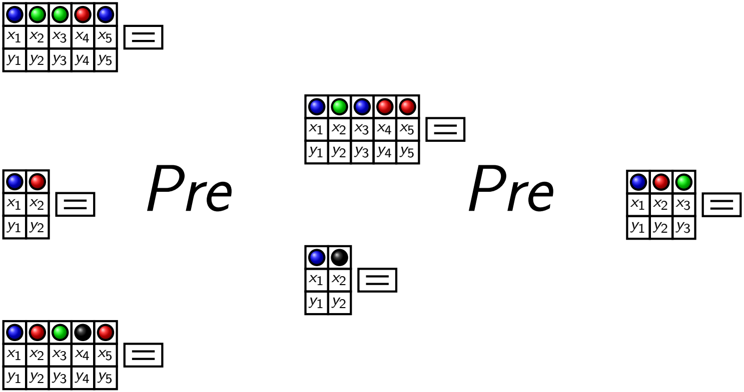
Backward Reachability Analysis



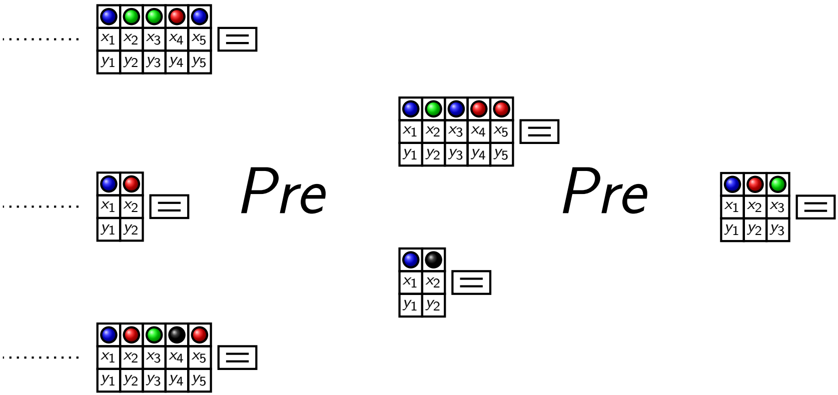
Backward Reachability Analysis



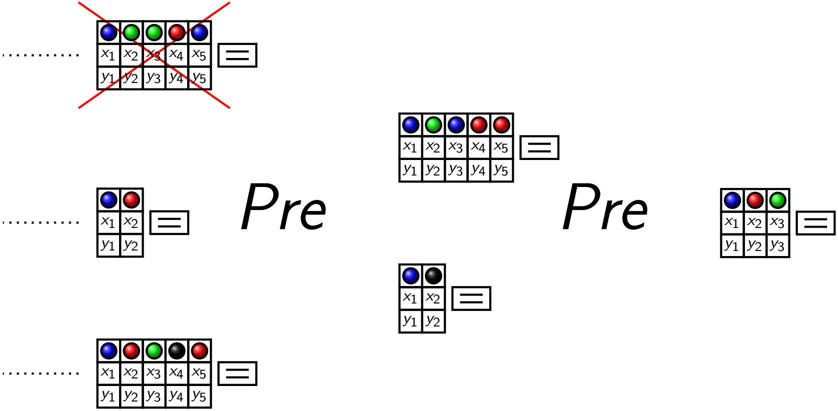
Backward Reachability Analysis



Backward Reachability Analysis

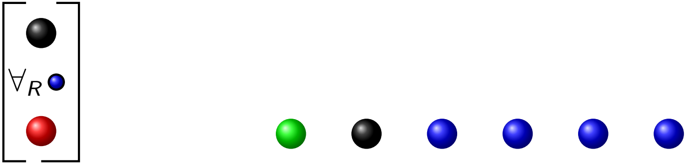


Backward Reachability Analysis



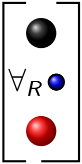
Non-Atomic Global Conditions

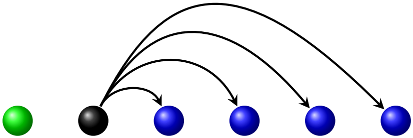
- Replace global condition with protocol:
  - Send request
  - Acks sent successively
  - Perform transition when all acks received



Non-Atomic Global Conditions

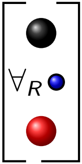
- Replace global condition with protocol:
  - Send request
  - Acks sent successively
  - Perform transition when all acks received

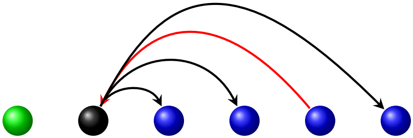




Non-Atomic Global Conditions

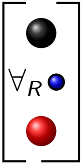
- Replace global condition with protocol:
  - Send request
  - Acks sent successively
  - Perform transition when all acks received

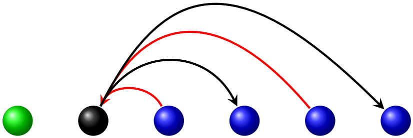




Non-Atomic Global Conditions

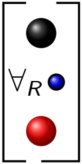
- Replace global condition with protocol:
  - Send request
  - Acks sent successively
  - Perform transition when all acks received

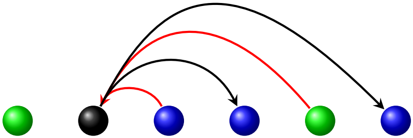




Non-Atomic Global Conditions

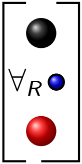
- Replace global condition with protocol:
  - Send request
  - Acks sent successively
  - Perform transition when all acks received

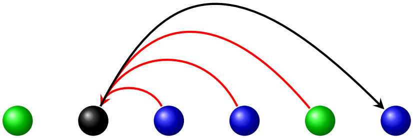




Non-Atomic Global Conditions

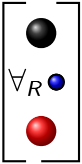
- Replace global condition with protocol:
  - Send request
  - Acks sent successively
  - Perform transition when all acks received

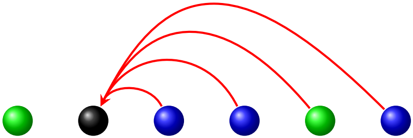




Non-Atomic Global Conditions

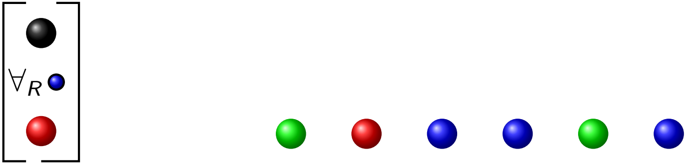
- Replace global condition with protocol:
    - Send request
    - Acks sent successively
    - Perform transition when all acks received





Non-Atomic Global Conditions

- Replace global condition with protocol:
  - Send request
  - Acks sent successively
  - Perform transition when all acks received



Lamport's Distributed Mut-Ex

Q_P : q_{idle}, q_{wait}, q_{use}
Q_C : q_{empty}, q_{req_1}, q_{ack_1}, q_{ok_1}, q_{req_2}, q_{ack_2}, q_{ok_2}
X_P : $\{id, num, aux \in \mathcal{N}\}$
X_C : $\{s\_id, r\_id, v \in \mathcal{N}\}$

Part I: Distributed Computation of Number

$$t_1 : \left[q_{idle} \rightarrow q_{choose} \;\triangleright\; \left(\begin{array}{l} aux' = num \wedge \\ \quad \forall\, \texttt{other} \neq \texttt{self} \cdot \\ \quad \left(\begin{array}{l} \texttt{other} \cdot state = empty \;\wedge\; \texttt{other} \cdot s\_id = \texttt{self} \cdot id \\ \supset \\ \texttt{other} \cdot state' = req_1 \;\wedge\; \texttt{other} \cdot v' = \texttt{self} \cdot num \end{array} \right) \end{array} \right) \right]$$

$$t_2 : \left[q_{choose} \rightarrow q_{choose} \;\triangleright\; \left(\begin{array}{l} \exists\, \texttt{other} \neq \texttt{self} \cdot \\ \left(\begin{array}{l} \texttt{other} \cdot state = ack_1 \;\wedge\; \texttt{other} \cdot s\_id = \texttt{self} \cdot id \\ \quad \wedge\; \texttt{other} \cdot v > \texttt{self} \cdot aux \\ \supset \\ \texttt{other} \cdot state' = ok_1 \;\wedge\; \texttt{self} \cdot aux' = \texttt{other} \cdot v \end{array} \right) \end{array} \right) \right]$$

$$t_3 : \left[q_{choose} \rightarrow q_{choose} \;\triangleright\; \left(\begin{array}{l} \exists\, \texttt{other} \neq \texttt{self} \cdot \\ \left(\begin{array}{l} \texttt{other} \cdot state = ack_1 \;\wedge\; \texttt{other} \cdot s\_id = \texttt{self} \cdot id \\ \quad \wedge\; \texttt{other} \cdot v \leq \texttt{self} \cdot aux \\ \supset \\ \texttt{other} \cdot state' = ok_1 \end{array} \right) \end{array} \right) \right]$$

$$t_4 : \left[q_{choose} \rightarrow q_{wait} \;\triangleright\; \left(\begin{array}{l} num' > aux \\ \wedge \\ \quad \forall\, \texttt{other} \neq \texttt{self} \cdot \\ \quad \texttt{other} \cdot s\_id = \texttt{self} \cdot id \;\supset\; \texttt{other} \cdot state = ok_1 \end{array} \right) \right]$$

Lamport's Distributed Mut-Ex, Part II: Reply

$$t_5 : \left[q_s \rightarrow q_s \;\triangleright\; \left(\begin{array}{l} \exists\ \texttt{other} \neq \texttt{self} \cdot \\ \quad \texttt{other} \cdot \mathit{state} = \mathit{req}_1 \;\wedge\; \texttt{other} \cdot \mathit{r\_id} = \texttt{self} \cdot \mathit{id} \\ \quad \supset \\ \quad \texttt{other} \cdot \mathit{state}' = \mathit{ack}_1 \;\wedge\; \texttt{other} \cdot v' = \texttt{self} \cdot \mathit{num} \end{array} \right) \right]$$

for any $s \in \{\mathit{idle}, \mathit{choose}, \mathit{wait}, \mathit{use}\}$

$$t_6 : \left[q_s \rightarrow q_s \;\triangleright\; \left(\begin{array}{l} \exists\ \texttt{other} \neq \texttt{self} \cdot \\ \quad \texttt{other} \cdot \mathit{state} = \mathit{req}_2 \;\wedge\; \texttt{other} \cdot \mathit{r\_id} = \texttt{self} \cdot \mathit{id} \\ \quad \supset \\ \quad \texttt{other} \cdot \mathit{state}' = \mathit{ack}_2 \;\wedge\; \texttt{other} \cdot v' = \texttt{self} \cdot \mathit{num} \end{array} \right) \right]$$

for any $s \in \{\mathit{idle}, \mathit{wait}, \mathit{use}\}$

Lamport's Distributed Mut-Ex, Part III: Entry and Exit

$$t_7 : \left[\; q_{wait} \rightarrow q_{wait} \;\; \triangleright \;\; \left(\begin{array}{l} \forall\ \texttt{other} \neq \texttt{self} \cdot \\ \quad \texttt{other} \cdot state = ok_1 \;\wedge\; \texttt{other} \cdot s\_id = \texttt{self} \cdot id \\ \quad \supset \\ \quad \texttt{other} \cdot state' = req_2 \end{array} \right) \right]$$

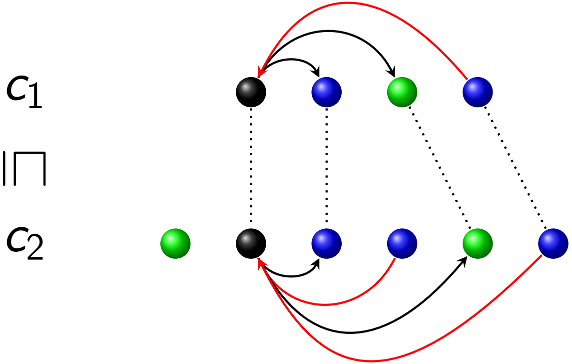
$$t_8 : \left[\; q_{wait} \rightarrow q_{wait} \;\; \triangleright \;\; \left(\begin{array}{l} \exists\ \texttt{other} \neq \texttt{self} \cdot \\ \quad \left(\begin{array}{l} \texttt{other} \cdot state = ack_2 \;\wedge \\ \texttt{other} \cdot s\_id = \texttt{self} \cdot id \;\wedge\; \texttt{other} \cdot v > 0 \wedge \\ \left(\begin{array}{l} \texttt{self} \cdot num > \texttt{other} \cdot v \;\vee \\ \left(\; \texttt{other} \cdot v = \texttt{self} \cdot num \wedge \; \texttt{self} \cdot id > r\_id \; \right) \end{array} \right) \end{array} \right) \\ \quad \supset \\ \quad \texttt{other} \cdot state' = req_2 \end{array} \right) \right]$$

$$t_9 : \left[\; q_{wait} \rightarrow q_{wait} \;\; \triangleright \;\; \left(\begin{array}{l} \exists\ \texttt{other} \neq \texttt{self} \cdot \\ \left(\begin{array}{l} \texttt{other} \cdot state = ack_2 \;\wedge \\ \texttt{other} \cdot s\_id = \texttt{self} \cdot id \;\wedge \\ \left(\begin{array}{l} \texttt{other} \cdot v = 0 \;\vee \\ \texttt{self} \cdot num < \texttt{other} \cdot v \;\vee \\ (\texttt{other} \cdot v = \texttt{self} \cdot num \;\wedge\; \texttt{self} \cdot id < r\_id) \end{array} \right) \end{array} \right) \\ \quad \supset \\ \texttt{other} \cdot state' = ok_2 \end{array} \right) \right]$$

$$t_{10} : \left[\; q_{wait} \rightarrow q_{use} \;\; \triangleright \;\; \left(\begin{array}{l} \forall\ \texttt{other} \neq \texttt{self} \cdot \\ \; s\_id = \texttt{self} \cdot id \;\; \supset \;\; \texttt{other} \cdot state = ok_2 \end{array} \right) \right]$$

$$t_{11} : \left[\; q_{use} \rightarrow q_{idle} \;\; \triangleright \;\; \left(\begin{array}{l} num' = 0 \\ \wedge \\ \quad \forall\ \texttt{other} \neq \texttt{self} \cdot \\ \quad\quad \texttt{other} \cdot state = ok_2 \;\wedge\; \texttt{other} \cdot s\_id = \texttt{self} \cdot id \\ \quad\quad \supset \\ \quad\quad \texttt{other} \cdot state' = empty \end{array} \right) \right]$$

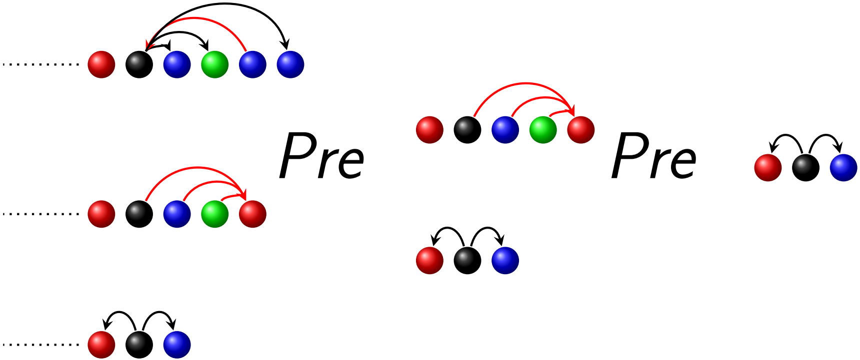
Ordering on Configurations



Approximation

- We apply monotonic abstraction when testing that all acknowledgments have been received (universal quantification)
- We delete all nodes and corresponding edges that have not acknowledged the request (i.e. they do not satisfy the condition we are checking)

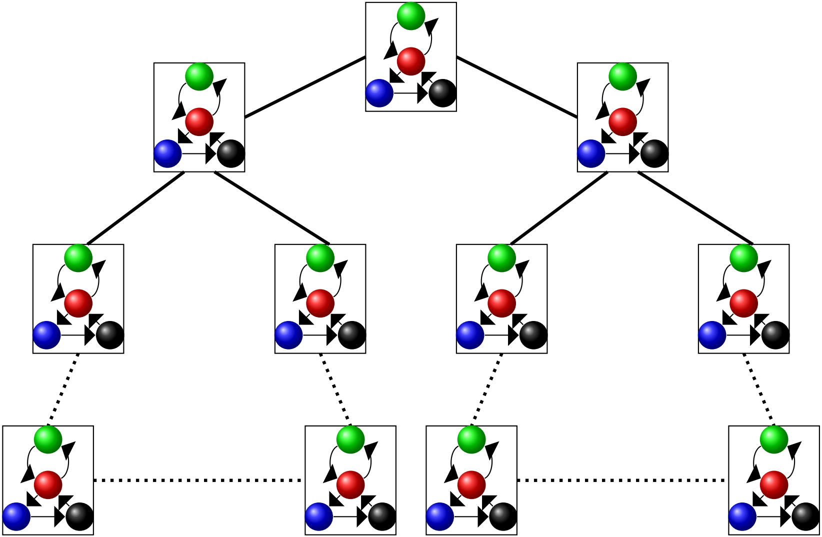
Predecessor Computation



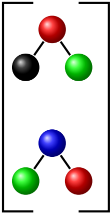
Termination

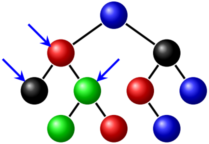
- Finite representation of upward closed sets of configurations (graphs)
- We use subgraph relation as entailment that is not a wqo for generic graphs
- Termination of the backward analysis is not guaranteed in general

Tree Topologies

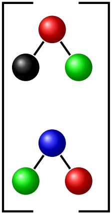


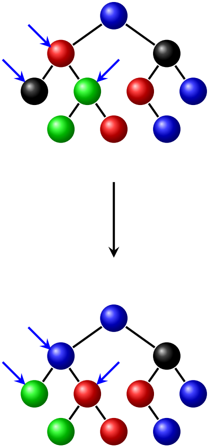
Transitions



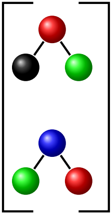


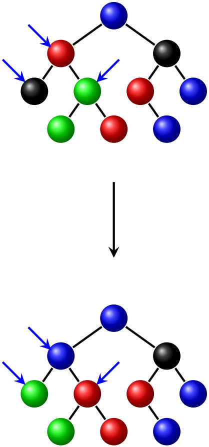
Transitions





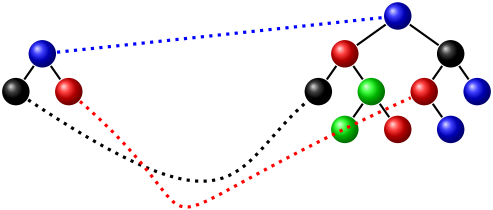
Transitions



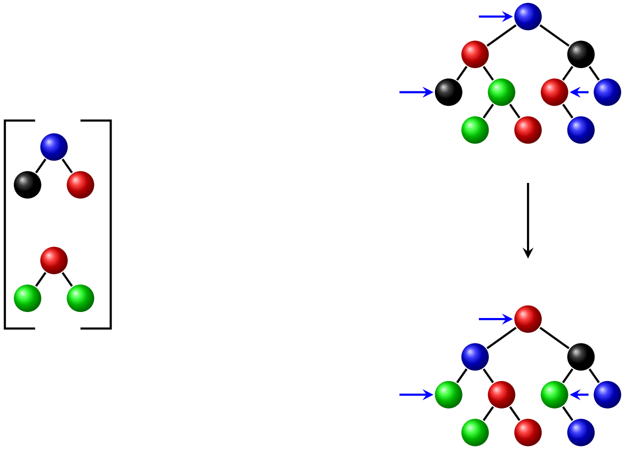


- Tree Arbiter Protocols
- Leader Election Protocols
- Distributed Protocols

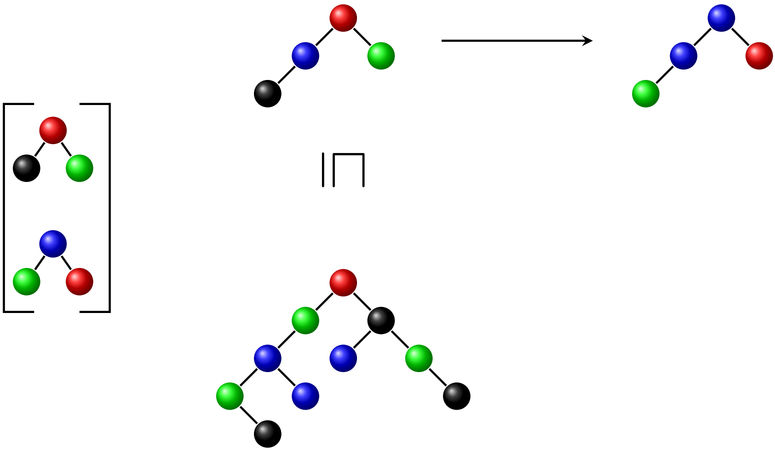
Ordering on Configurations



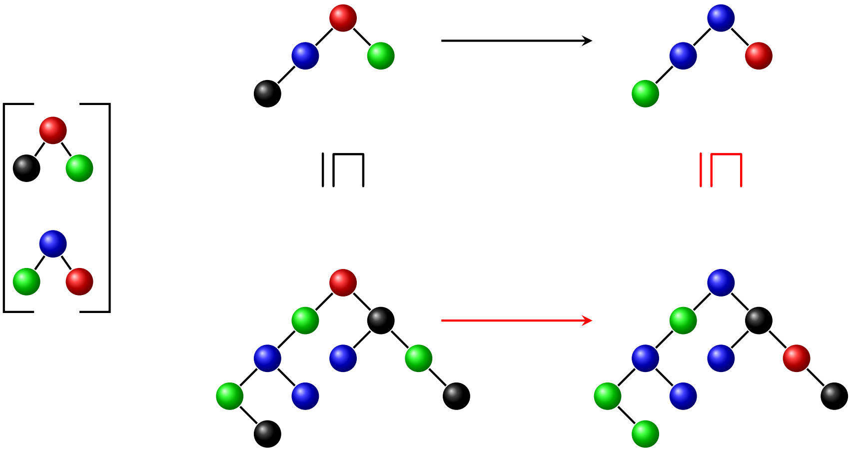
Monotonic Abstraction (Over-approximation)



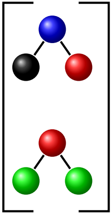
Monotonicity

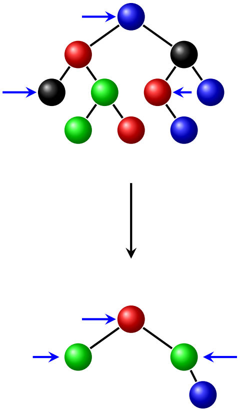


Monotonicity

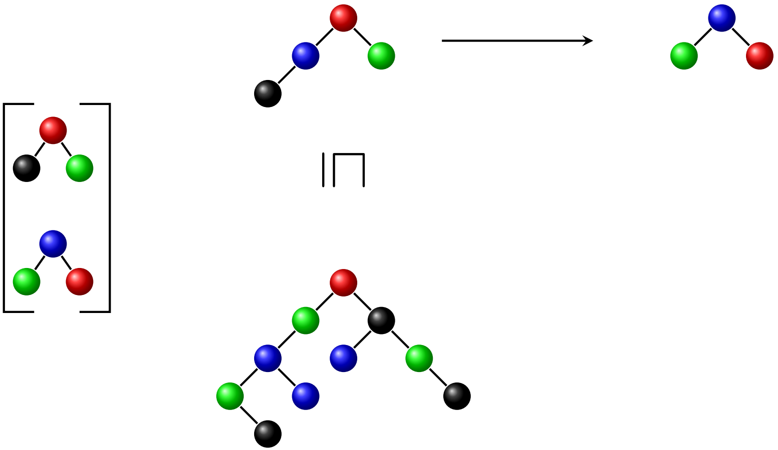


Monotonic Abstraction with Deletion

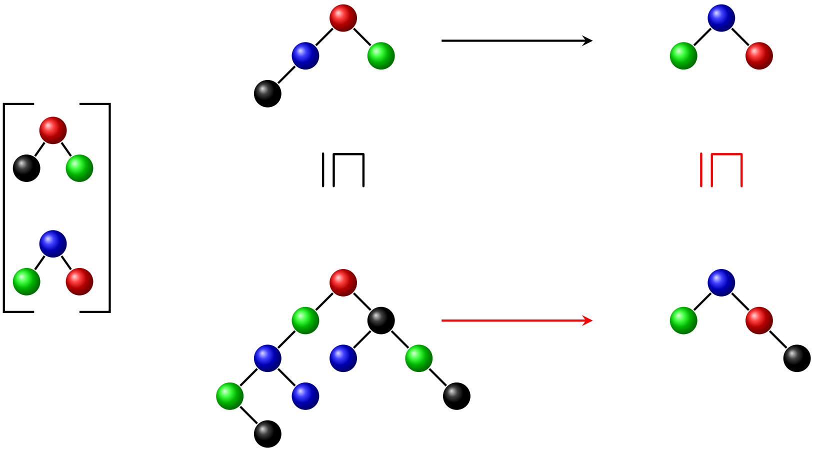




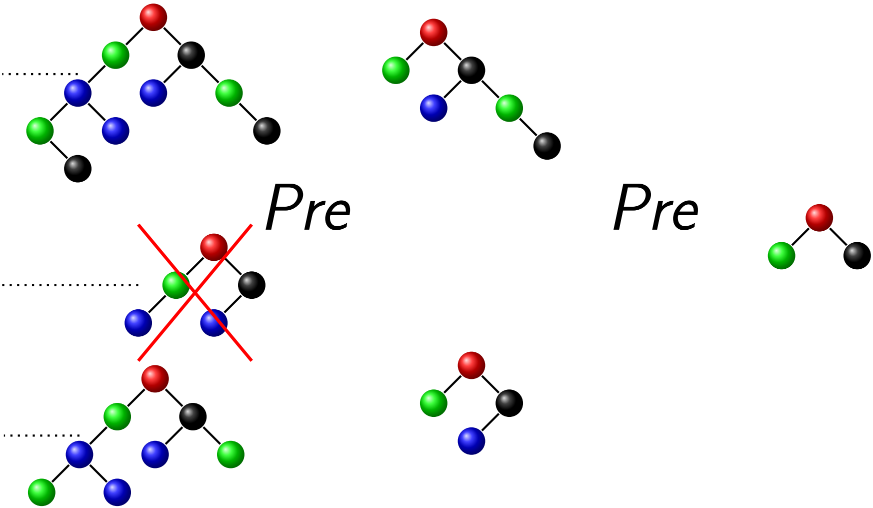
Monotonicity



Monotonicity



Backward Reachability on Trees



Termination

- Finite representation of upward closed sets of trees (with labels over a finite alphabet)
- Tree embedding as entailment: it is a wqo
- Termination of the backward analysis is guaranteed