Parameterized Reachability in Networks with Many Identical Processes

Arnaud Sangnier

LIAFA - Université Paris Diderot-Paris 7

Workshop Displexity
Cortina D'Ampezzo - 16th December 2014

Motivation

Verify network of processes of unbounded size

Why to consider such networks?

- Classical distributed algorithms (*mutual exclusion, leader election,...*)
- Telecommunication protocols (*routing,...*)
- Algorithms for ad-hoc networks
- Model for biological systems
- and many more applications ...

Hypothesis

All the processes have the same behavior

In [Esparza, STACS'14], such networks are called **crowd**

More precisely:

- Each process will follow the same protocol
- Process can communicate
- Communication way:
  - Message passing
  - Shared variable
  - **Rendez-vous communication**
  - **Broadcast communication**
  - **Multi-diffusion (selective broadcast)**

Question:
Is there a network with N processes which allows to reach a goal ?

Verification of parameterized systems

Different kind of parameters

Parametric data manipulated by the system

- Parametric timed automata [Alur et al., STOC'93]
- Counter machines with parameterized constants [Ibarra et al., TCS'02]
- Parametric one-counter automata [Haase et al., CONCUR'09]

Parametric number of processes in the system

- Seminal paper about counting abstraction [German & Sistla, ACM'92]
  - **Main idea:** Forget about the identities of each process
- We call such systems **parameterized networks**

In this talk

Main issues in verification of parameterized systems

- Infinite state systems
- Decidability and complexity issues
- Representation of possible infinite sets of configurations

Today:

Decidability and complexity of reachability problems on parameterized networks

Features

- Simple protocols with different means of communication
- Simple reachability questions

Outline

Outline

Parameterized Networks with rendez-vous

[German & Sistla, ACM'92]

Main characteristics

- No creation/deletion of processes
- Each process executes the same finite state protocol
- Synchronization through rendez-vous
- Each process can synchronize with any other process

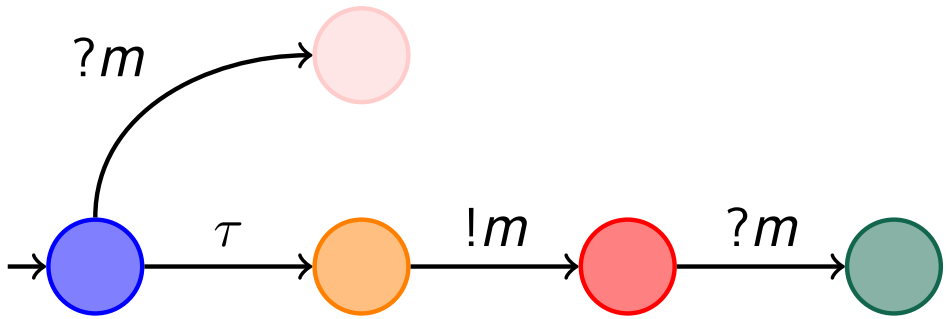
Rendez-vous Networks: syntax

A protocol $P = \langle Q, \Sigma, R, Q_0 \rangle$

Finite state system whose transitions are labeled with:

1. Request for rendez-vous with m - $!m$
2. Acknowledged rendez-vous with - $?m$
3. Internal actions - τ

where m belongs to the finite alphabet Σ

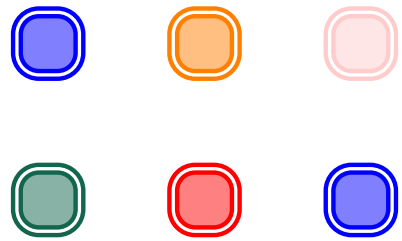


A protocol defines a Rendez-vous network (RN)

Rendez-vous Networks: configurations

A configuration is a multiset $\gamma : Q \mapsto \mathbb{N}$

- $\gamma(q)$ tells how many processes are in state q



- **Initial configurations**: $\gamma(q) > 0$ iff $q \in Q_0$

Remarks:

- The number of processes in a configuration is not bounded
- Infinite number of configurations

$\Rightarrow$ **RN are infinite state systems**

Rendez-voust Networks: semantics

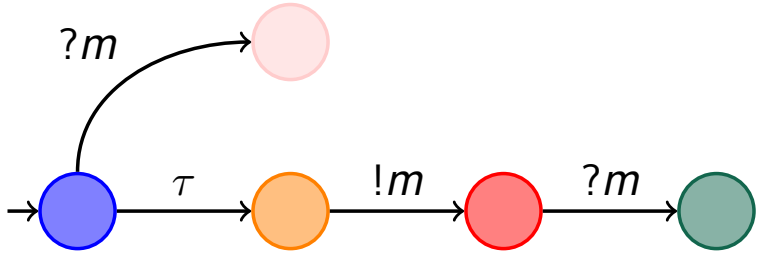
Transition system $RN(P) = \langle \mathcal{C}, \rightarrow, \mathcal{C}_0 \rangle$ associated to P

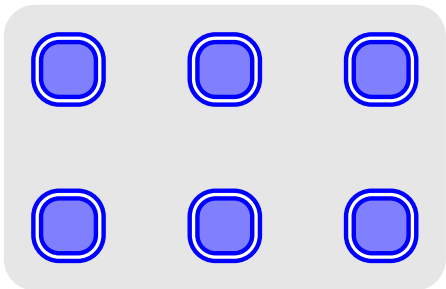
- $\mathcal{C}$: set of configurations
- $\rightarrow$: $\mathcal{C} \times \mathcal{C}$: transition relation
- $\mathcal{C}_0$: initial configurations

The relation $\rightarrow$ respects the following rules during an execution:

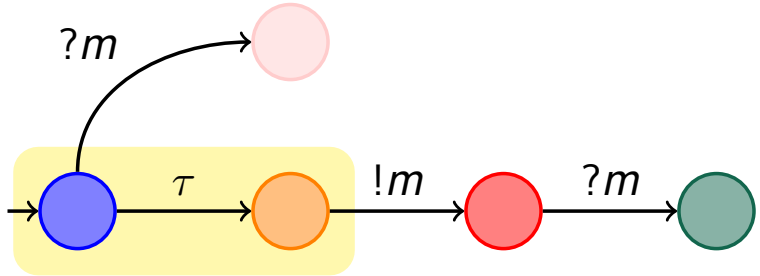
- The number of processes in an execution remain unchanged
- Processes can only change their state
- Two kind of transitions according to the given process
  1. **local actions** - one process performs an internal action τ
  2. **rendez-vous** - one process performs $!m$ and another process performs $?m$

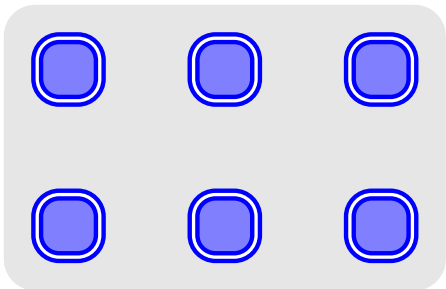
Rendez-vous Networks: an example



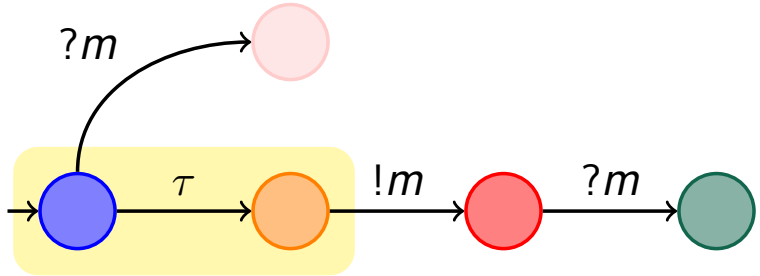


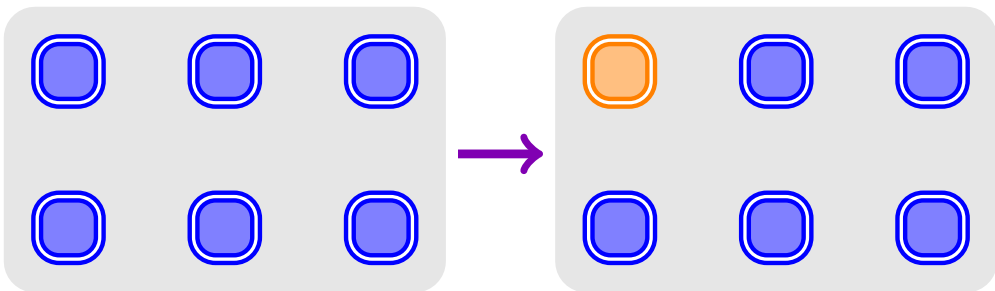
Rendez-vous Networks: an example



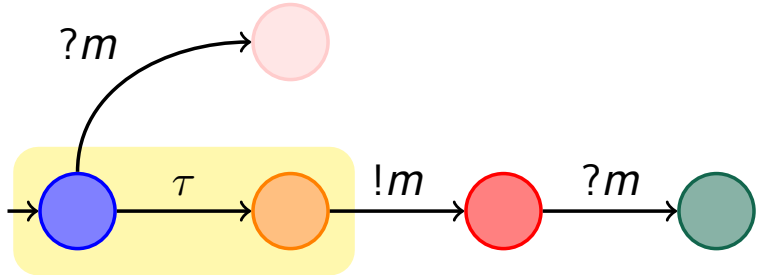


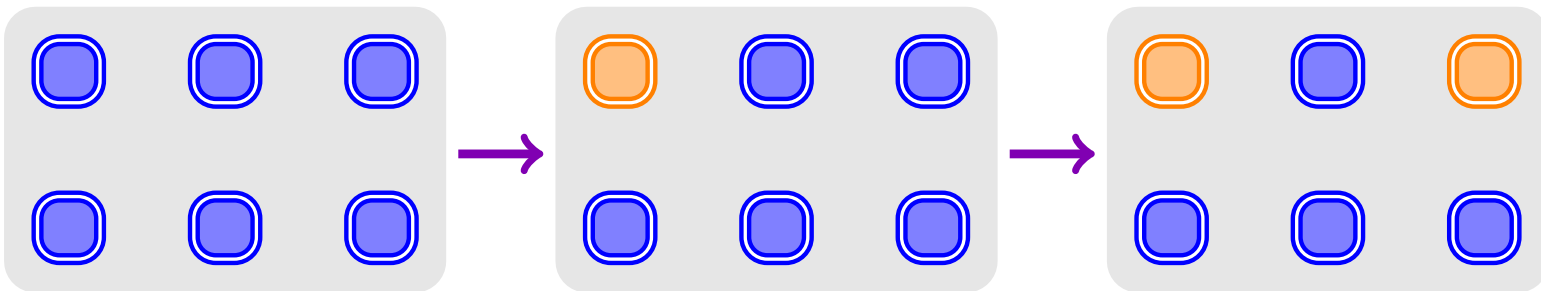
Rendez-vous Networks: an example



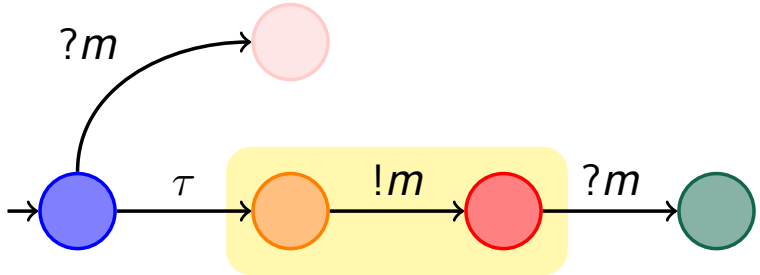


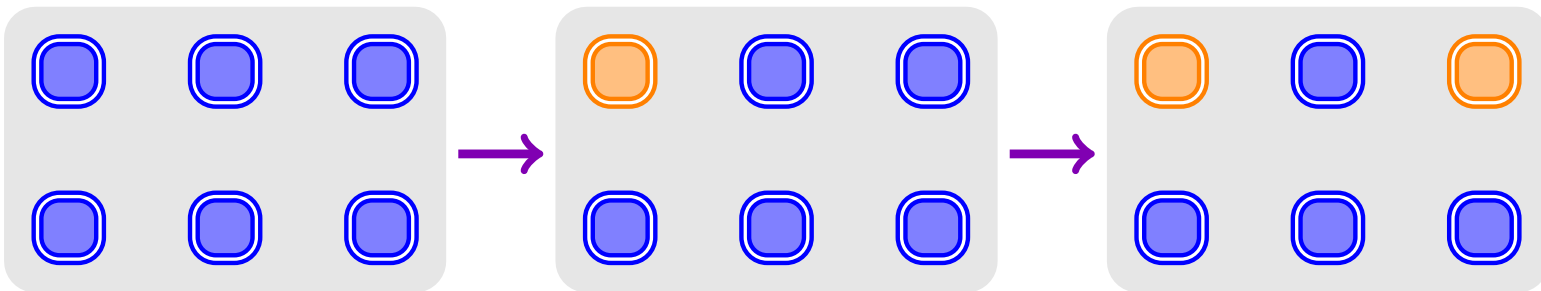
Rendez-vous Networks: an example



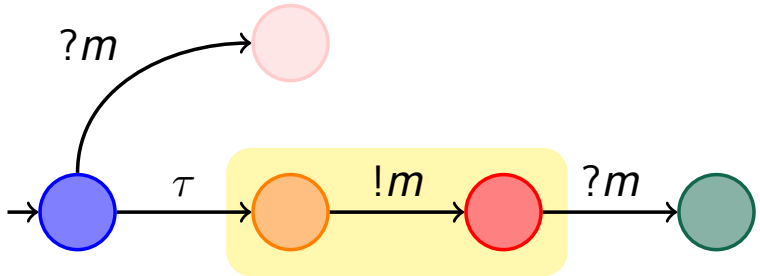


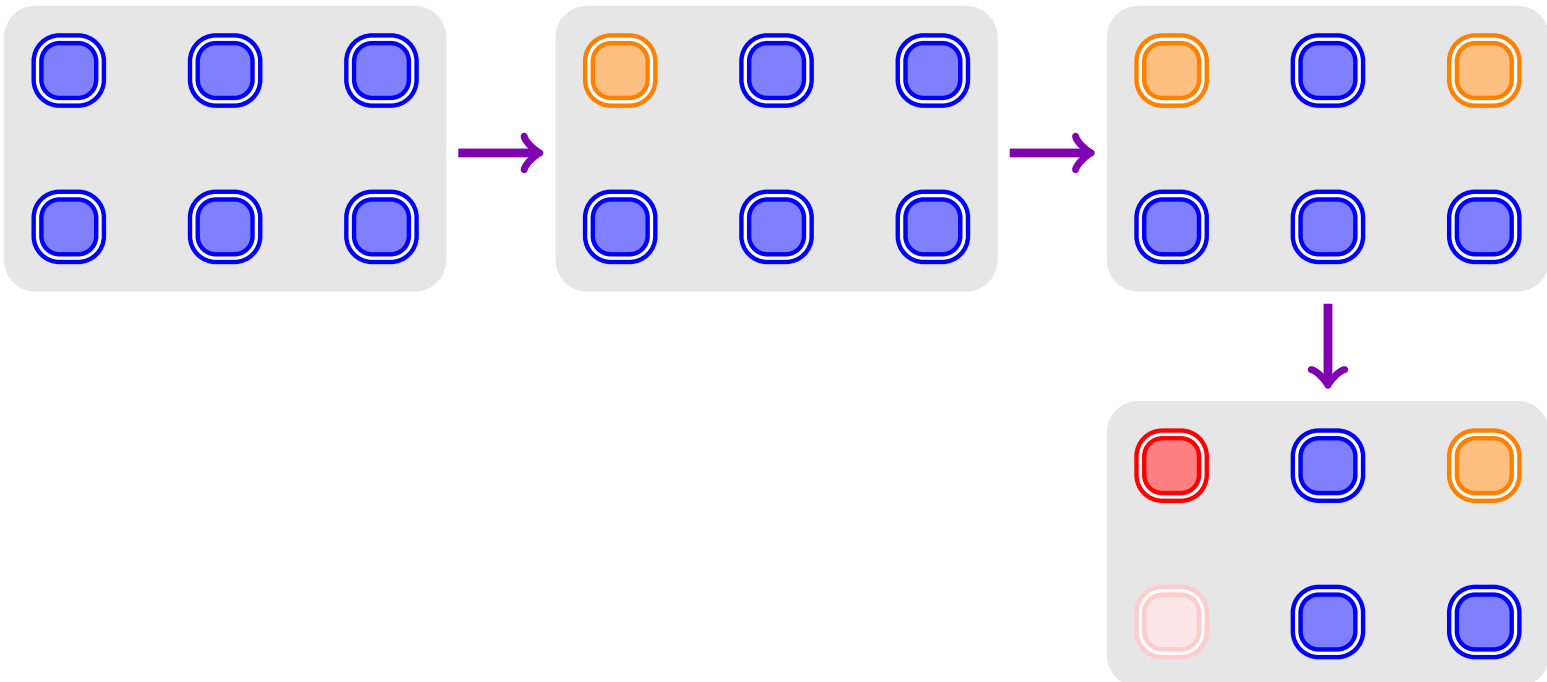
Rendez-vous Networks: an example



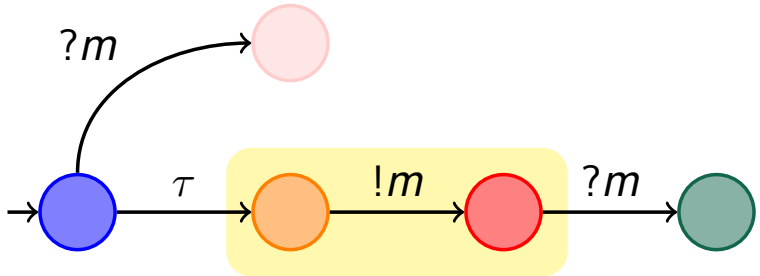


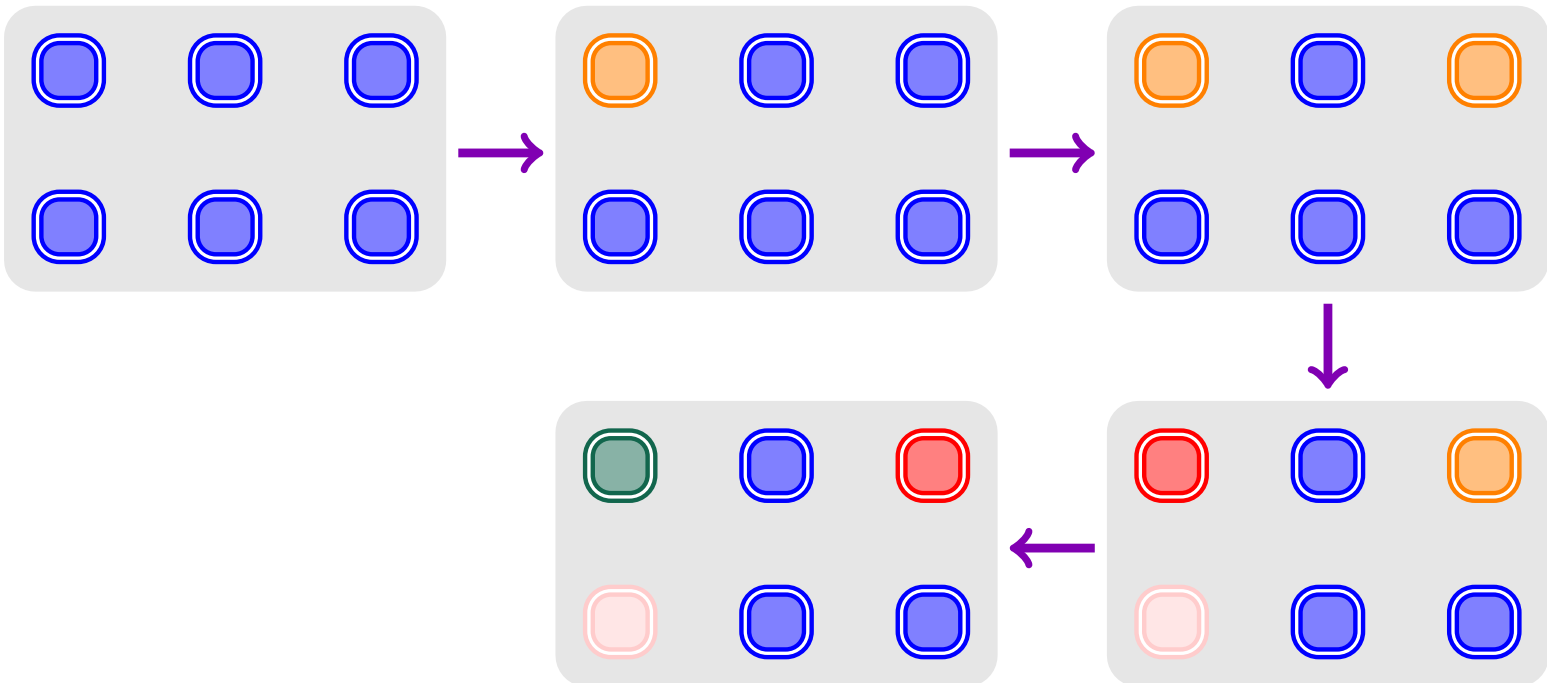
Rendez-vous Networks: an example





Rendez-vous Networks: an example





Reachability question

Parameters: Number of processes

Control State Reachability (REACH)

Input: A protocol and a control state $q \in Q$;

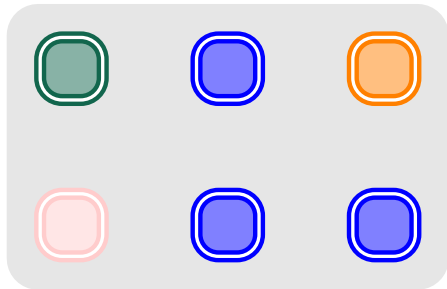
Output: Does there exist $\gamma \in \mathcal{C}_0$ and $\gamma' \in \mathcal{C}$ s.t. $\gamma \to^* \gamma'$ and $\gamma(q) > 0$?

Remarks:

- This problem considers an infinite number of possible initial configurations
- Reachability of a configuration γ' is easier, **the number of processes is in fact fixed**

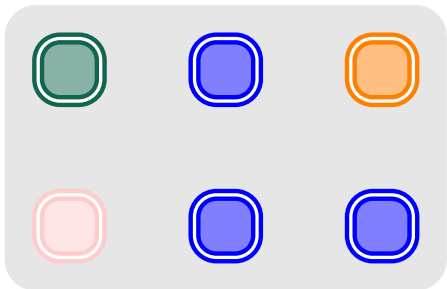
A crucial characteristic

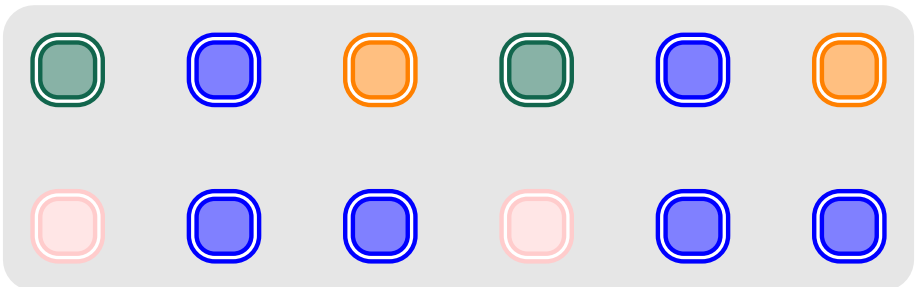
What you can do with N processes



A crucial characteristic

What you can do with N processes
you can do it similarly with 2N processes





Solving REACH with Rendez-vous is easy

Theorem [German & Sistla, ACM'92]

For Rendez-vous Networks, `Reach` is in PTIME

Idea of the proof:

- Algorithm which builds the set of reachable states
- Each time, do all the possible transactions in the protocol
- Saturation obtained after a polynomial number of steps

Algorithm to solve REACH with Rendez-vous

PTIME algorithm to compute the set of reachable states

Input : $P = \langle Q, \Sigma, R, Q_0 \rangle$ a protocol
Output : $S \subseteq Q$ the set of reachable control states in $RN(P)$

1: $S := Q_0$
2: $oldS := \emptyset$
3: **while** $S \neq oldS$ **do**
4: $\quad oldS := S$
5: $\quad$ **for all** $\langle q_1, !m, q_2 \rangle \in R$ such that $q_1 \in oldS$ **do**
6: $\quad\quad S := S \cup \{q_2\} \cup \{q' \in Q \mid \langle q, ?m, q' \rangle \in R \wedge q \in oldS\}$
7: $\quad$ **end for**
8: **end while**

Controlled Networks with rendez-vous

[German & Sistla, ACM'92]

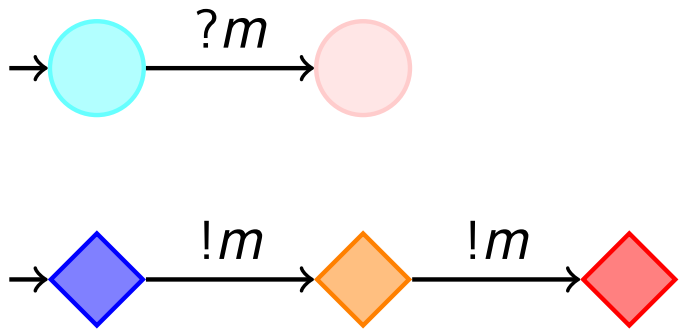
Main characteristics

- No creation/deletion of processes
- Each process executes the same finite state protocol
- **There is also a single controller in the system**
- Synchronization through rendez-vous
- Each process can synchronize with **the controller**

Controlled Rendez-vous Networks: syntax

A protocol $P = \langle Q, Q_c\Sigma, R, Q_0, q_{c,0} \rangle$

- $Q \cap Q_c = \emptyset$
- Edges do not go from Q to Q_c or from Q_c to Q
- Only the Controller do $!m$
- The Controller does not do $?m$

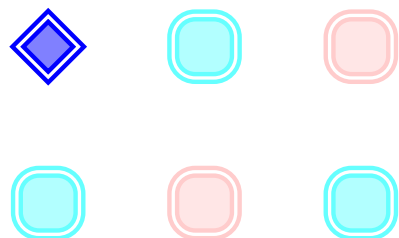


A protocol defines a Controlled Rendez-vous network (CRN)

Controlled Rendez-vous Networks: configurations

A configuration is a pair (q_c, γ)

- q_c is a current state of the controller
- $\gamma : Q \mapsto \mathbb{N}$
- $\gamma(q)$ tells how many processes are in state q



- **Initial configurations**: $(q_{c,0}, \gamma(q))$ s.t. $\gamma(q) > 0$ iff $q \in Q_0$

Remarks:

- Infinite number of configurations
- **You cannot get rid of the controller**

$\Rightarrow$ **CRN are infinite state systems**

Controlled Rendez-voust Networks: semantics

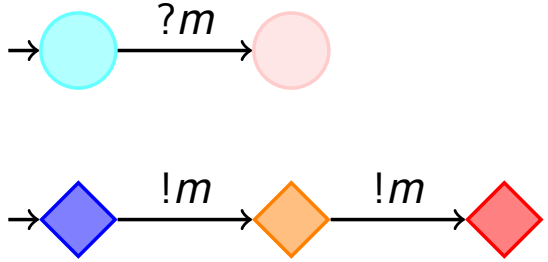
Transition system $CRN(P) = \langle \mathcal{C}, \rightarrow, \mathcal{C}_0 \rangle$ associated to P

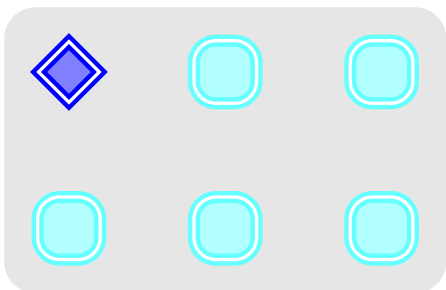
- $\mathcal{C}$: set of configurations
- $\rightarrow$: $\mathcal{C} \times \mathcal{C}$: transition relation
- $\mathcal{C}_0$: initial configurations

The relation $\rightarrow$ respects the following rules during an execution:

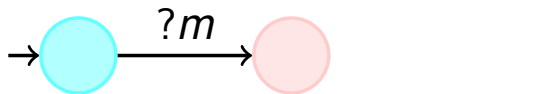
- The number of processes in an execution remain unchanged
- Processes can only change their state
- Two kind of transitions according to the given process
  1. **local actions** - one process or controller performs an internal action τ
  2. **rendez-vous** - the controller performs $!m$ and a process performs $?m$

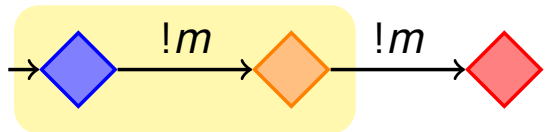
Controlled Rendez-vous Networks: an example

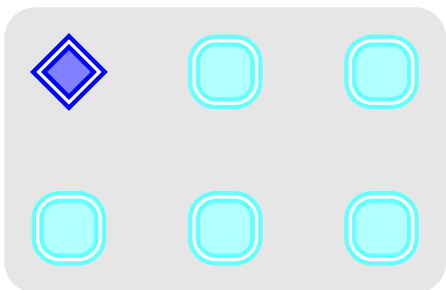




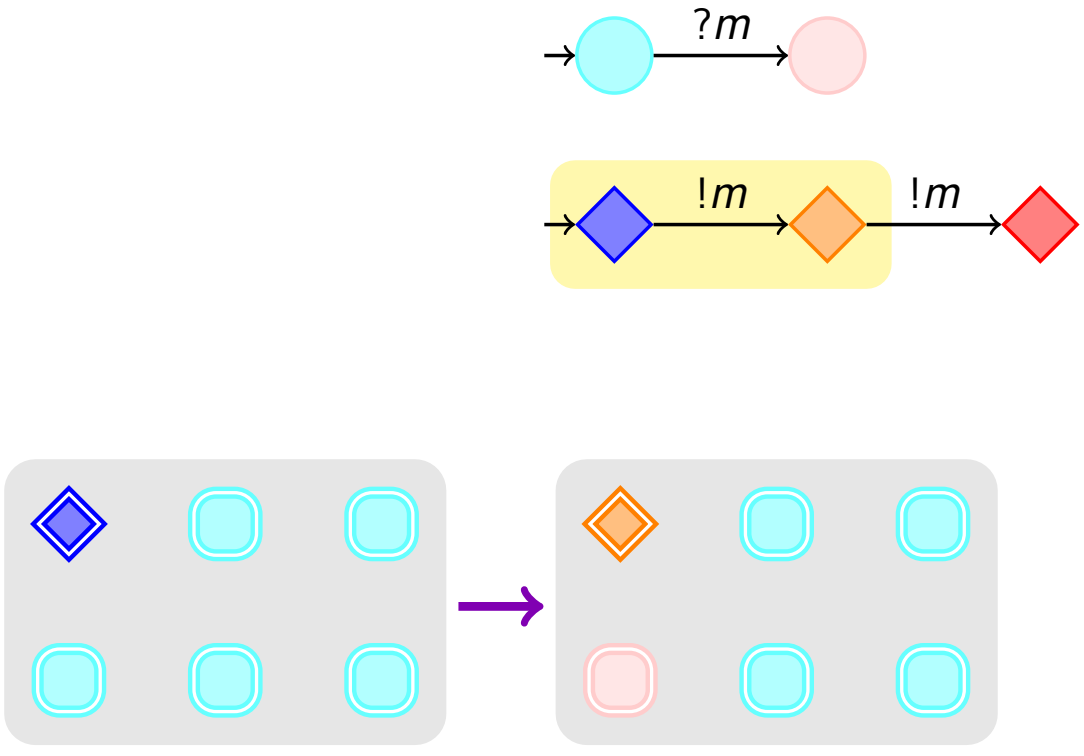
Controlled Rendez-vous Networks: an example



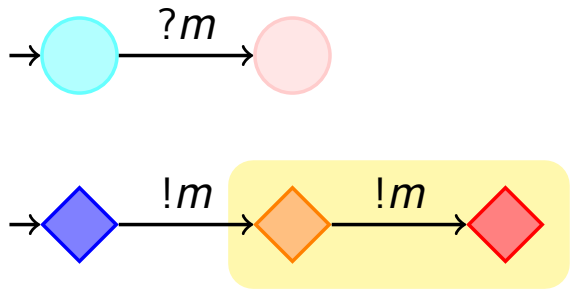


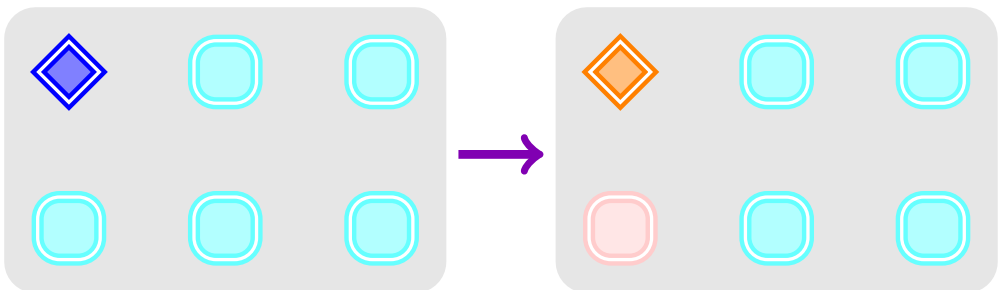


Controlled Rendez-vous Networks: an example

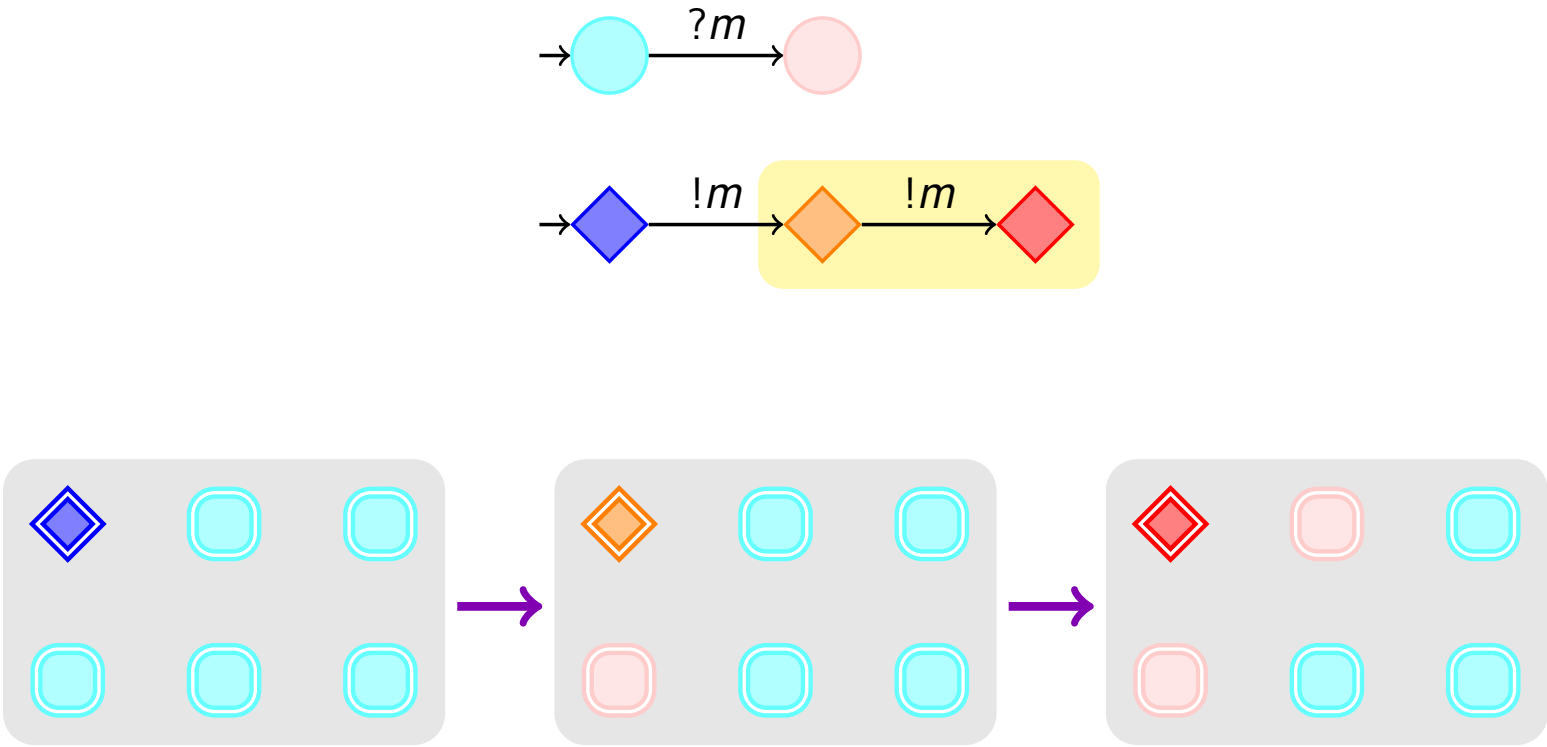


Controlled Rendez-vous Networks: an example

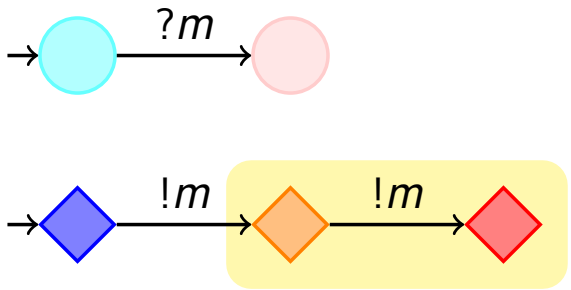


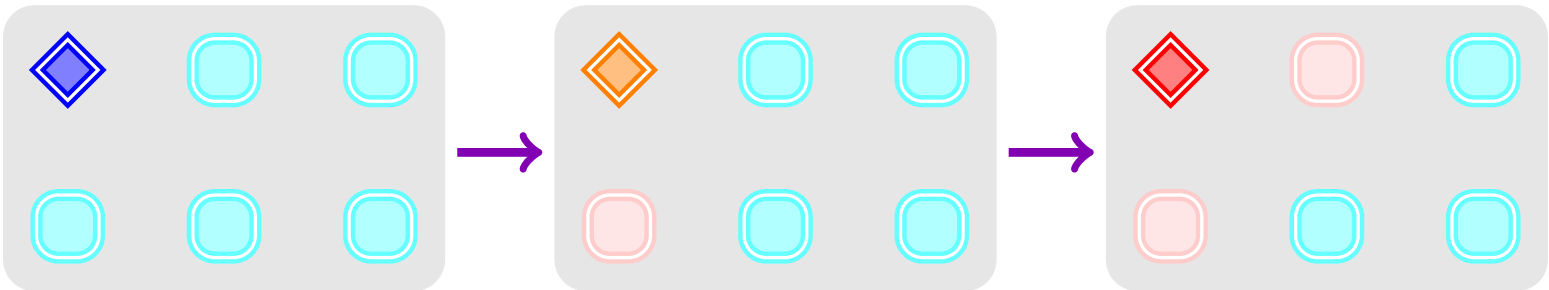


Controlled Rendez-vous Networks: an example



Controlled Rendez-vous Networks: an example





We cannot have as many processes as we want in pink!

Reachability question

Parameters: Number of processes

Control State Reachability (REACH)

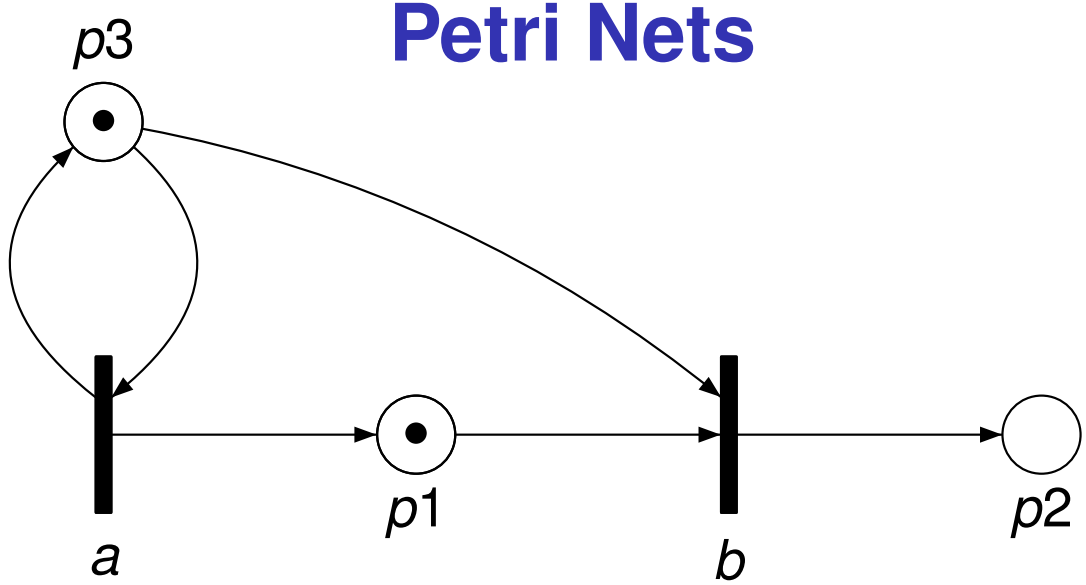
Input: A protocol and a control state $q_c \in Q_c$;

Output: Does there exist $(q_{c,0}, \gamma) \in \mathcal{C}_0$ and $(q_c, \gamma') \in \mathcal{C}$ s.t. $(q_{c,0}.\gamma) \rightarrow^* (q_c, \gamma')$?

Remarks:

- This problem considers an infinite number of possible initial configurations
- Reachability of a configuration (q_c, γ') is easier, **the number of processes is in fact fixed**

Petri Nets

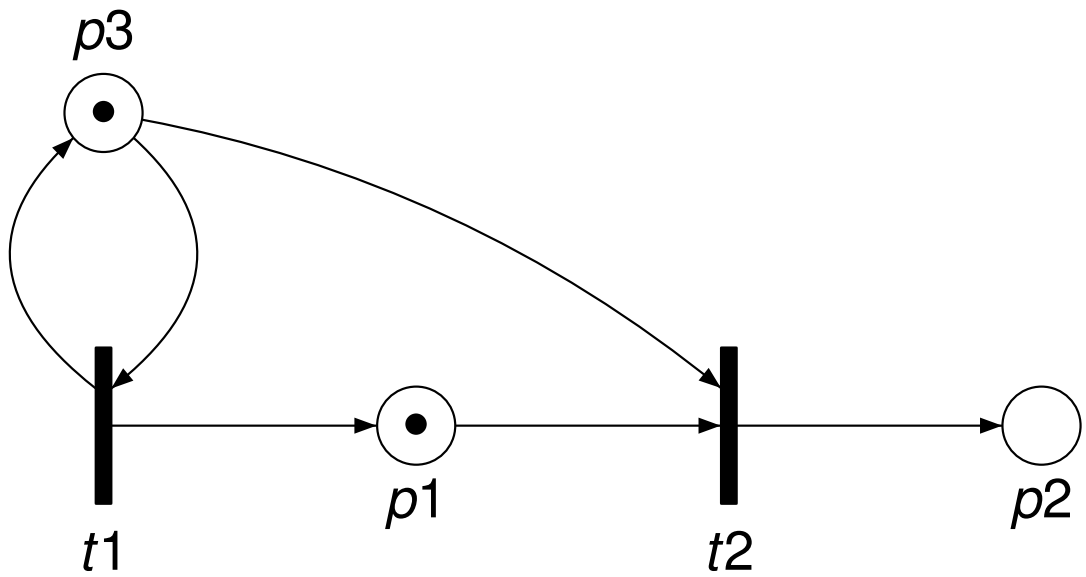


A Petri net $\mathcal{N} = (P, T, {}^\bullet(), ()^\bullet, M_0)$

- P: finite set of places
- T: finite set of transitions
- ${}^\bullet() : T \mapsto \mathbb{N}^P$: backward incidence mapping
- $()^\bullet : T \mapsto \mathbb{N}^P$: forward incidence mapping
- $M_0 \in \mathbb{N}^P$: initial marking

- **Reach**$(\mathcal{N}) = \{M \in \mathbb{N}^P \mid M_0 \rightarrow^* M\}$

Semantics of Petri nets

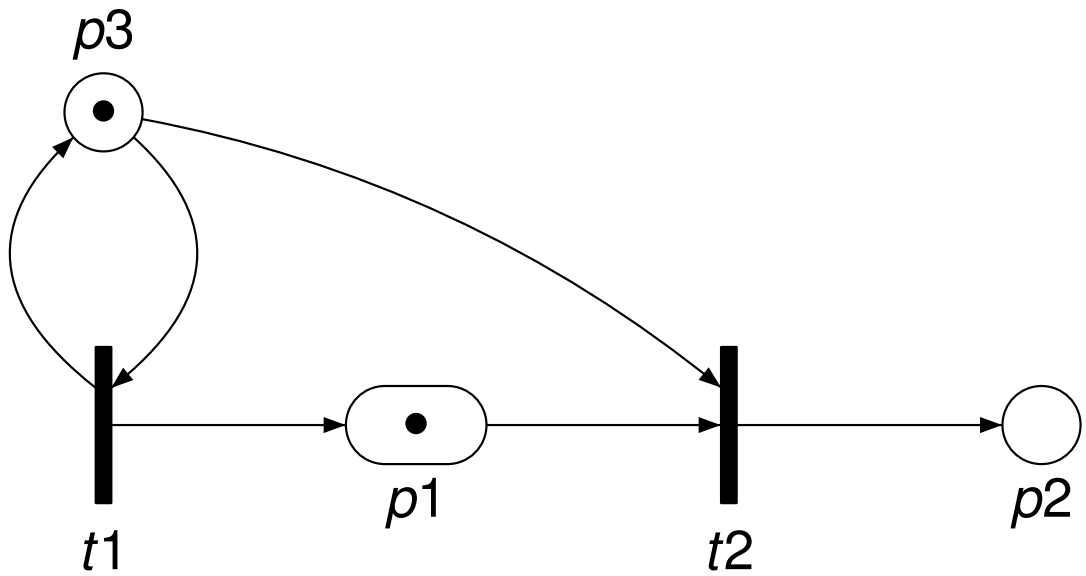


Transition system $[\![\mathcal{N}]\!] = (\mathbb{N}^P, M_0, \rightarrow)$

For $M, M' \in \mathbb{N}^P$, $M \rightarrow M'$ iff $\exists t \in T$ s.t.: $M \geq {}^\bullet t \ \wedge \ M' = M - {}^\bullet t + t^\bullet$

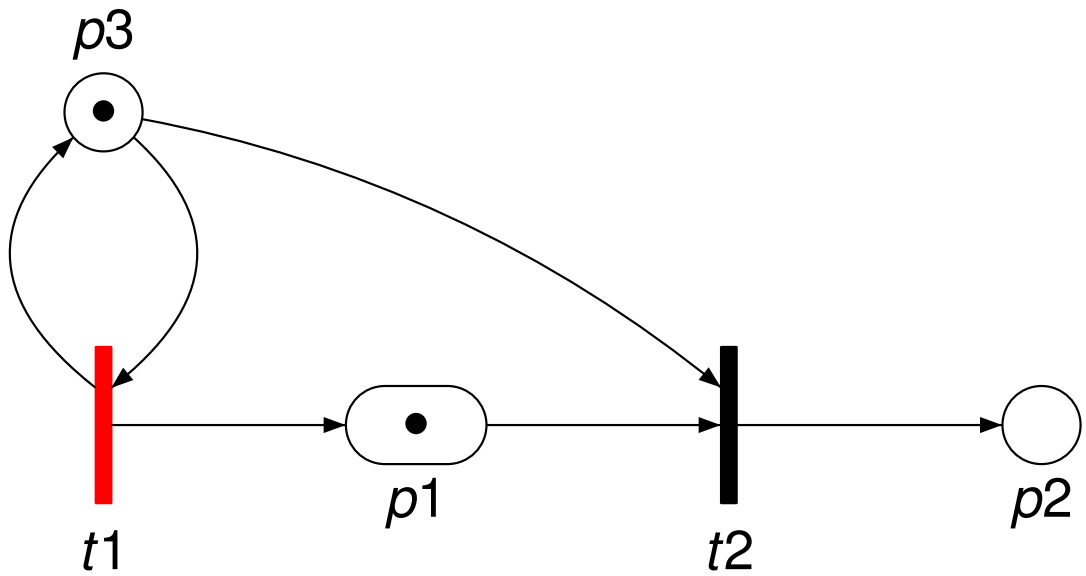
- **Reach**$(\mathcal{N}) = \{M \in \mathbb{N}^P \mid M_0 \rightarrow^* M\}$

Execution in Petri nets



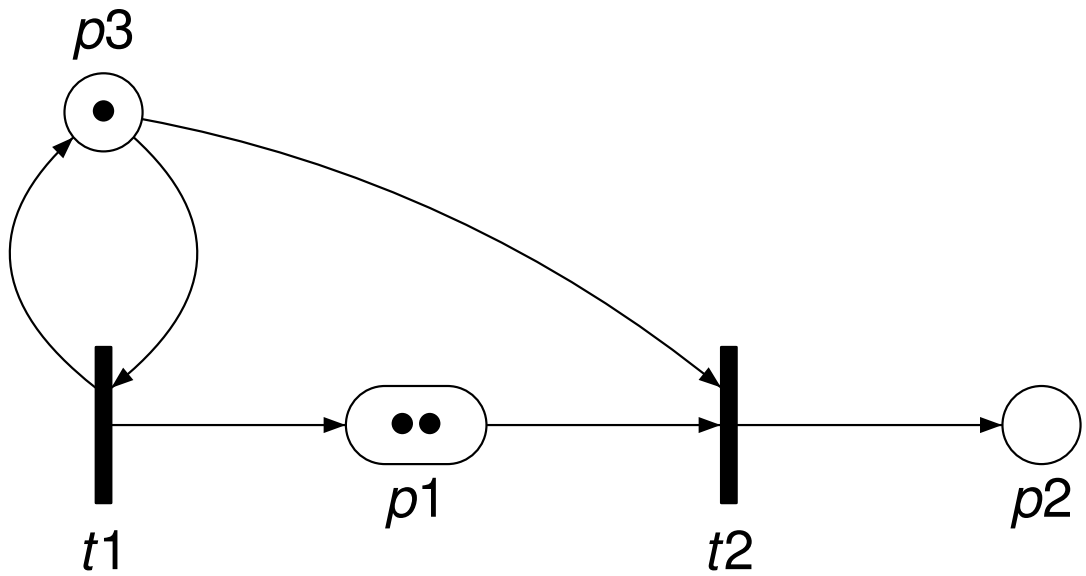
$$\begin{pmatrix} 1 \\ 0 \\ 1 \end{pmatrix}$$

Execution in Petri nets



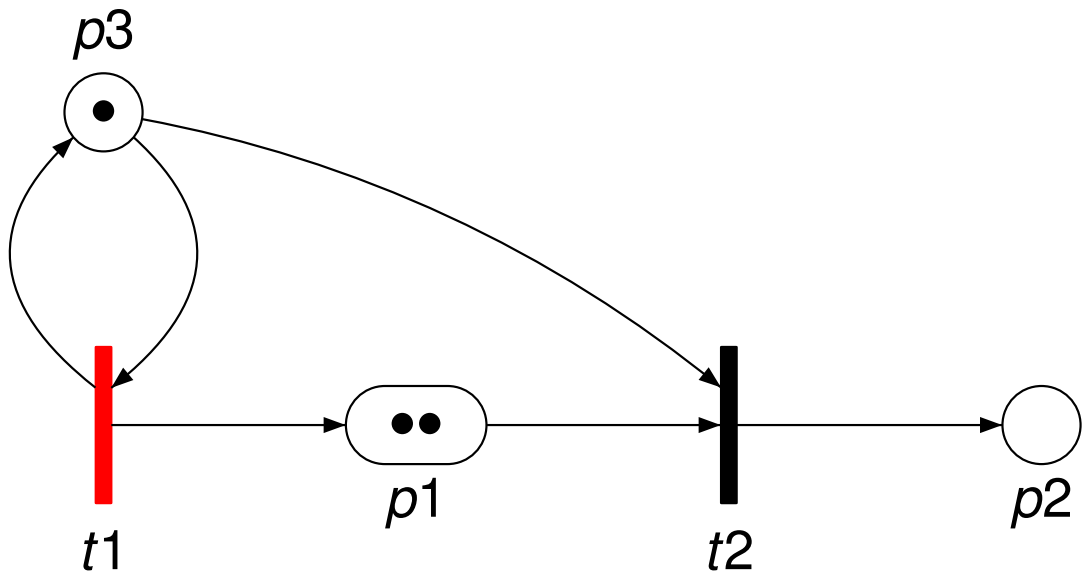
$$\begin{pmatrix} 1 \\ 0 \\ 1 \end{pmatrix} \rightarrow$$

Execution in Petri nets



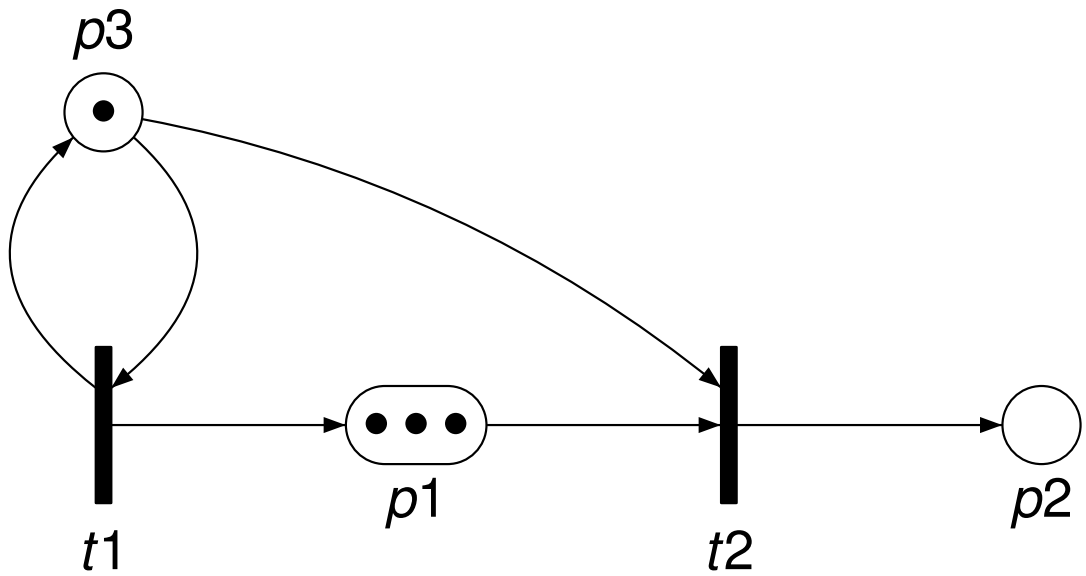
$$\begin{pmatrix} 1 \\ 0 \\ 1 \end{pmatrix} \rightarrow \begin{pmatrix} 2 \\ 0 \\ 1 \end{pmatrix}$$

Execution in Petri nets



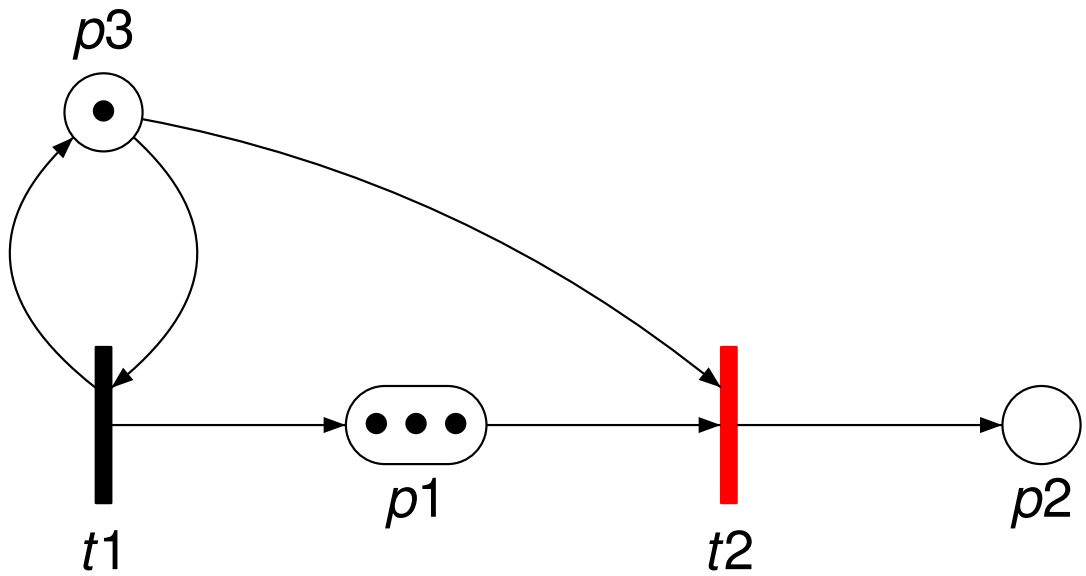
$$\begin{pmatrix} 1 \\ 0 \\ 1 \end{pmatrix} \rightarrow \begin{pmatrix} 2 \\ 0 \\ 1 \end{pmatrix} \rightarrow$$

Execution in Petri nets



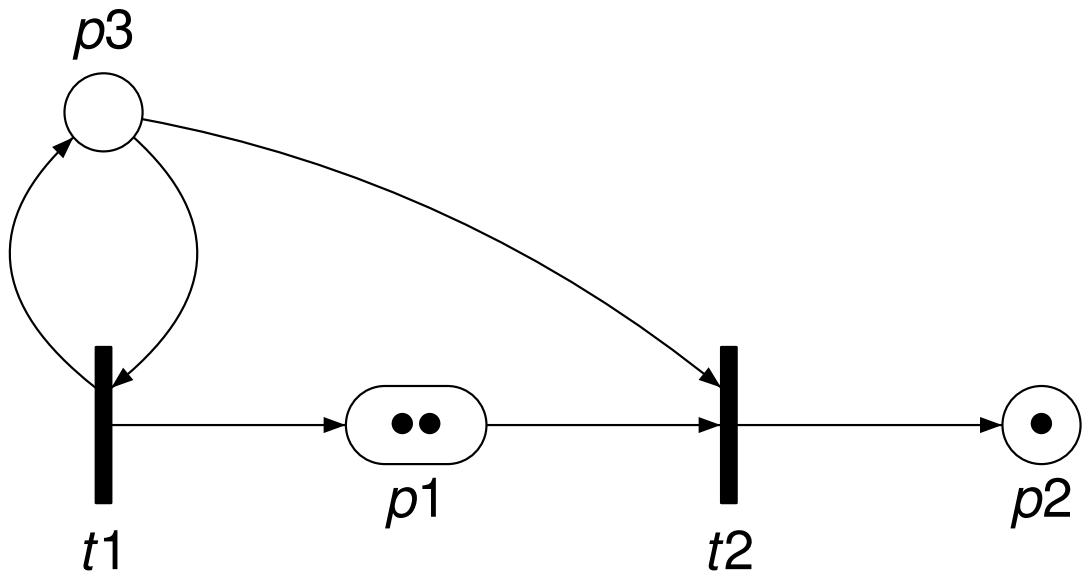
$$\begin{pmatrix} 1 \\ 0 \\ 1 \end{pmatrix} \rightarrow \begin{pmatrix} 2 \\ 0 \\ 1 \end{pmatrix} \rightarrow \begin{pmatrix} 3 \\ 0 \\ 1 \end{pmatrix}$$

Execution in Petri nets



$$\begin{pmatrix} 1 \\ 0 \\ 1 \end{pmatrix} \rightarrow \begin{pmatrix} 2 \\ 0 \\ 1 \end{pmatrix} \rightarrow \begin{pmatrix} 3 \\ 0 \\ 1 \end{pmatrix} \rightarrow$$

Execution in Petri nets



$$\begin{pmatrix} 1 \\ 0 \\ 1 \end{pmatrix} \rightarrow \begin{pmatrix} 2 \\ 0 \\ 1 \end{pmatrix} \rightarrow \begin{pmatrix} 3 \\ 0 \\ 1 \end{pmatrix} \rightarrow \begin{pmatrix} 2 \\ 1 \\ 0 \end{pmatrix}$$

Coverability problem

Coverability [Karp & Miller, J. Comput. Syst. Sci'69]

Input: A Petri net $\mathcal{N}$ and a marking M;

Output: Does there exist $M' \in \mathbf{Reach}(\mathcal{N})$ s.t. $M' \geq M$?

Theorem [Lipton, Tech. Rep.'76; Rackoff TCS'78]

The coverability problem is EXPSPACE-complete.

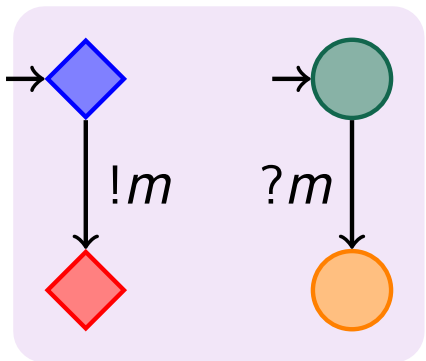
Thanks to this result:

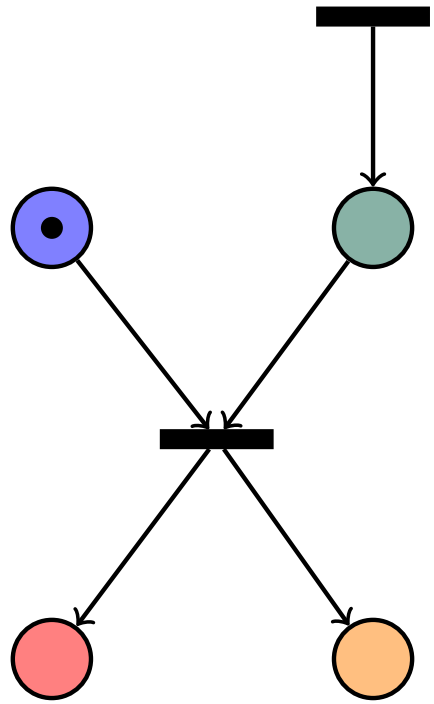
Theorem [German & Sistla, ACM'92]

REACH for Controlled Rendez-vous Networks is EXPSPACE-complete.

Simultating a CRN with a Petri net

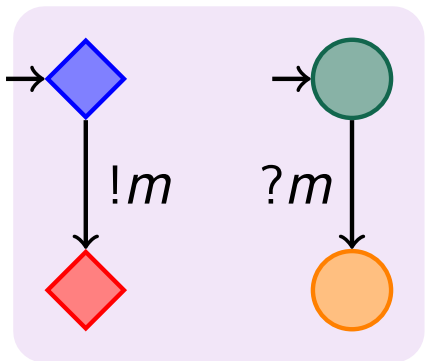
Counting abstraction

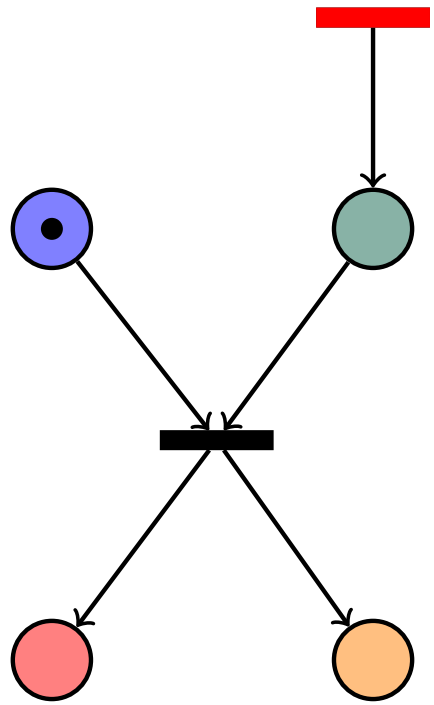




Simultating a CRN with a Petri net

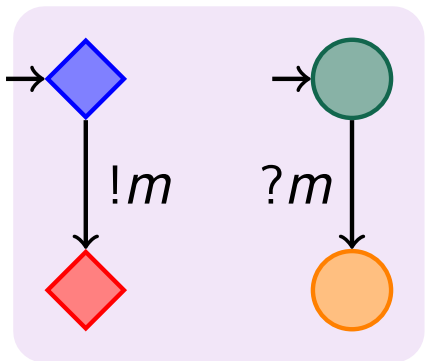
Counting abstraction

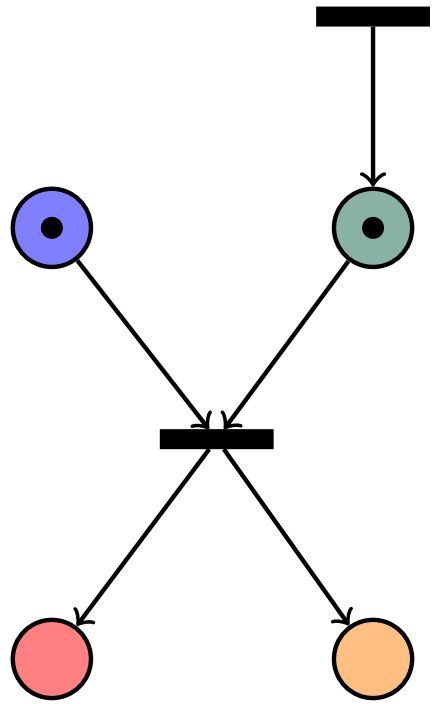




Simultating a CRN with a Petri net

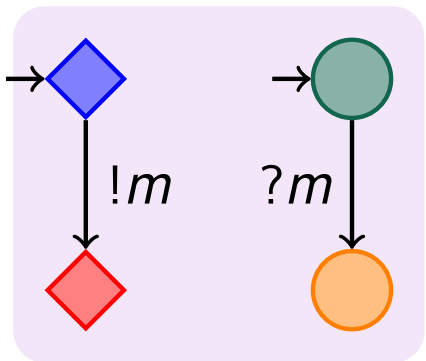
Counting abstraction

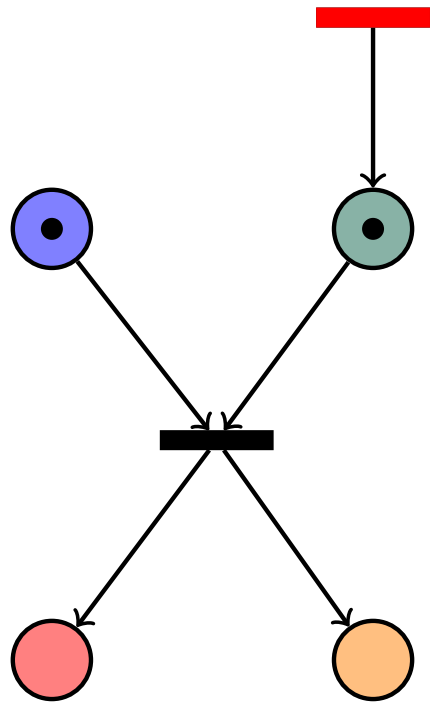




Simultating a CRN with a Petri net

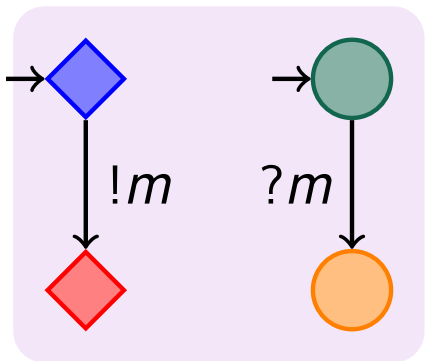
Counting abstraction

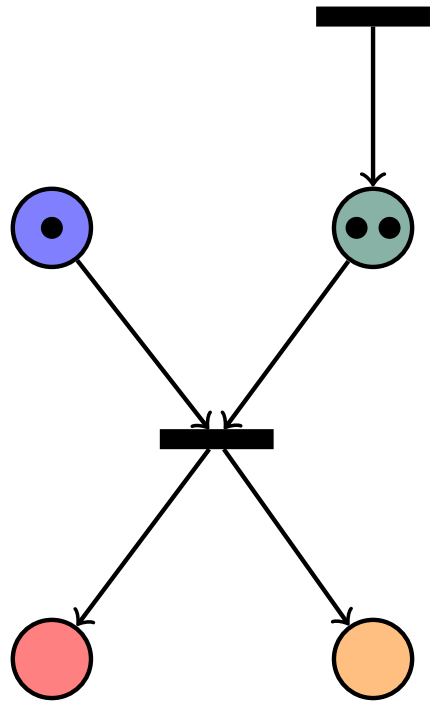




Simultating a CRN with a Petri net

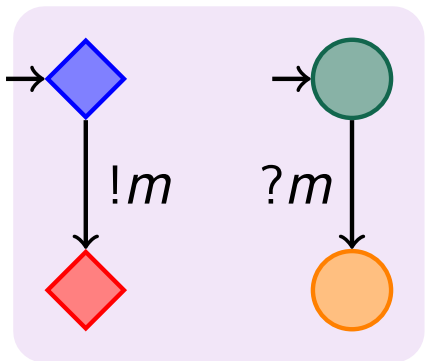
Counting abstraction

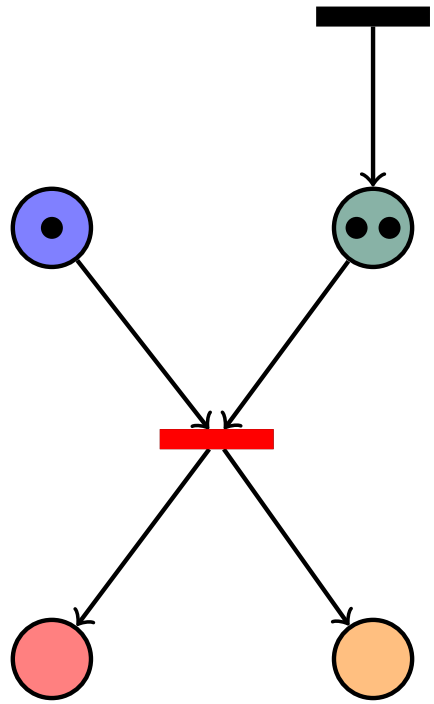




Simultating a CRN with a Petri net

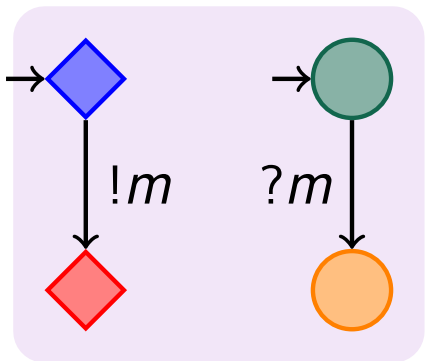
Counting abstraction

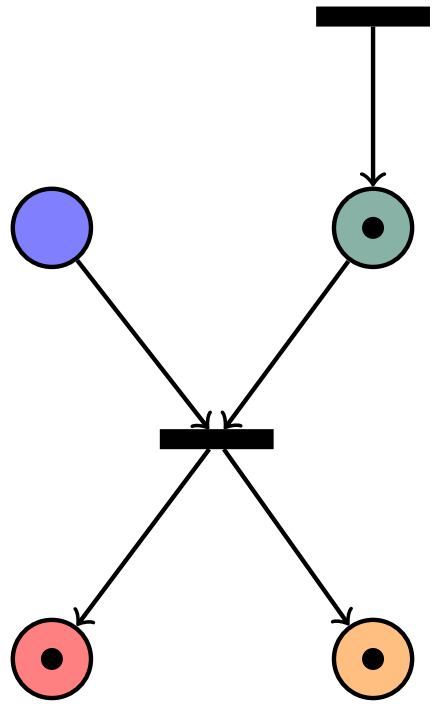




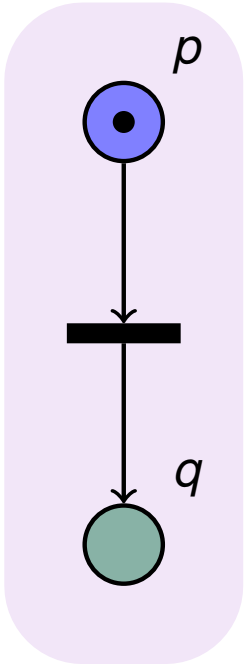
Simultating a CRN with a Petri net

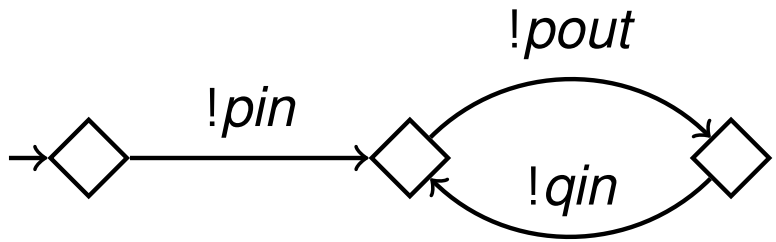
Counting abstraction

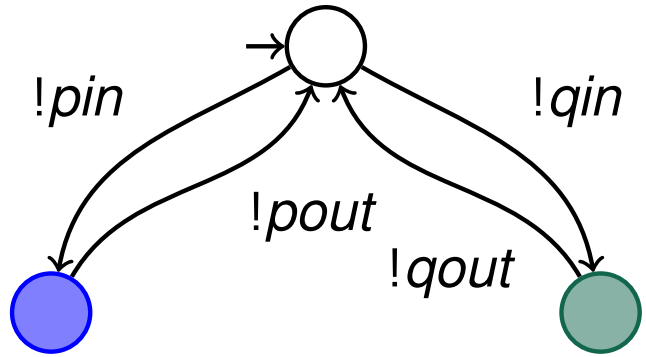




Simulating a Petri net with a CRN







Number of processes : maximum number of tokens needed for coverability

Outline

Parameterized Networks with Broadcast

[Esparza et al., LICS'99]

Main characteristics

- No creation/deletion of processes
- Each process executes the same finite state protocol
- Synchronization through broadcast of a message
- **All the processes** receive the message

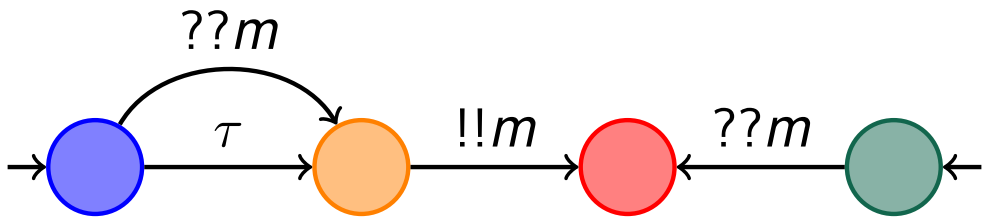
Broadcast Networks: syntax

A protocol $P = \langle Q, \Sigma, R, Q_0 \rangle$

Finite state system whose transitions are labeled with:

1. broadcast of messages - $!!m$
2. reception of messages - $??m$
3. internal actions - τ

where m belongs to the finite alphabet Σ

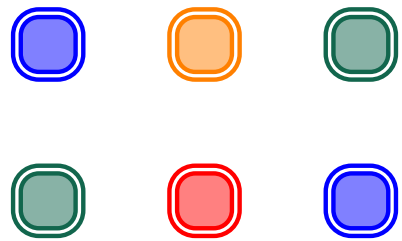


A protocol defines a Broadcast Network (BN)

Broadcast Networks: configurations

A configuration is a multiset $\gamma : Q \mapsto \mathbb{N}$

- Same as for Rendez-vous Networks



- **Initial configurations**: $\gamma(q) > 0$ iff $q \in Q_0$

Remarks:

- The size of configurations is not bounded
- Infinite number of configurations

$\Rightarrow$ **BN are infinite state systems**

Broadcast Networks: semantics

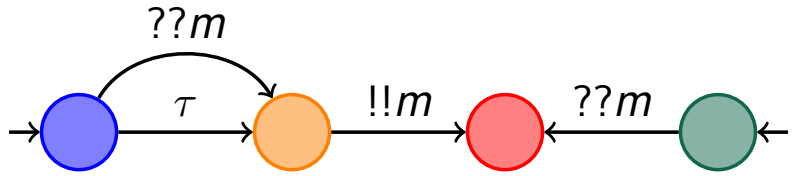
Transition system $BN(P) = \langle \mathcal{C}, \rightarrow, \mathcal{C}_0 \rangle$ associated to P

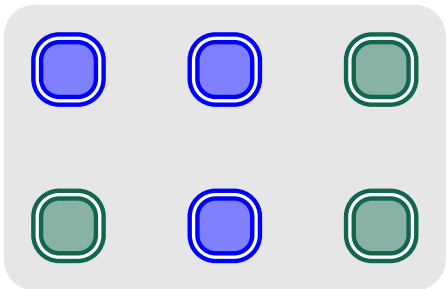
- $\mathcal{C}$: set of configurations
- $\rightarrow$: $\mathcal{C} \times \mathcal{C}$: transition relation
- $\mathcal{C}_0$: initial configurations

The relation $\rightarrow$ respects the following rules during an execution:

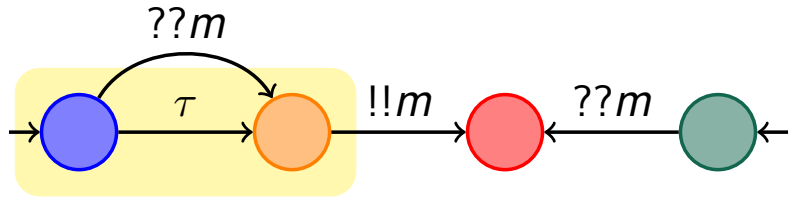
- The number of processes in an execution does not change
- Processes can only change their state
- Two kind of transitions according to the given process
  1. **local actions** - one process performs an internal action τ
  2. **broadcast** - one process emits a message with $!!m$, **all the processes** that can receive it with $??m$ have to receive it

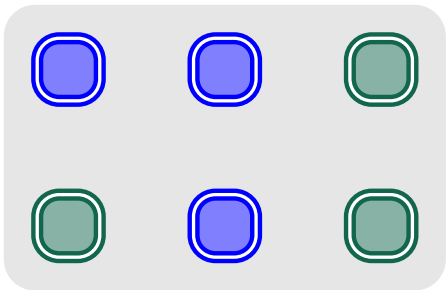
Broadcast Networks: an example



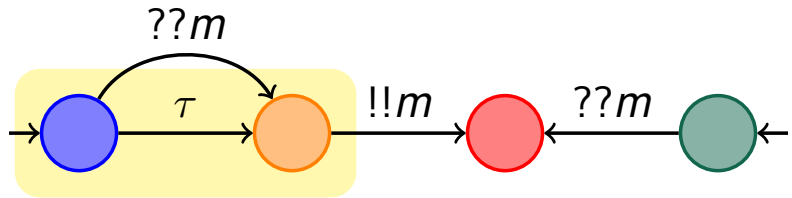


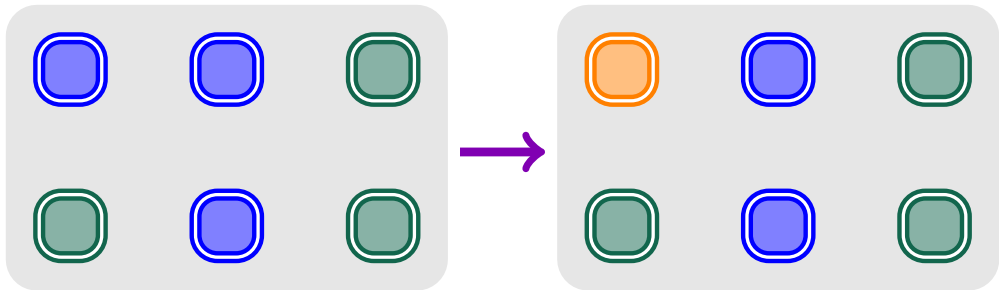
Broadcast Networks: an example



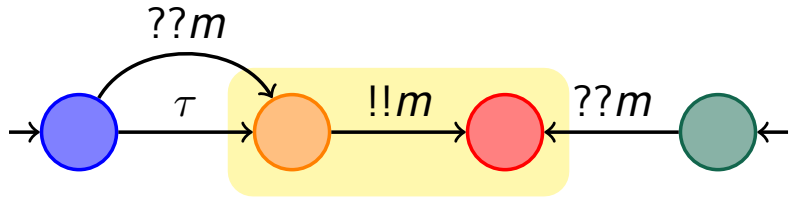


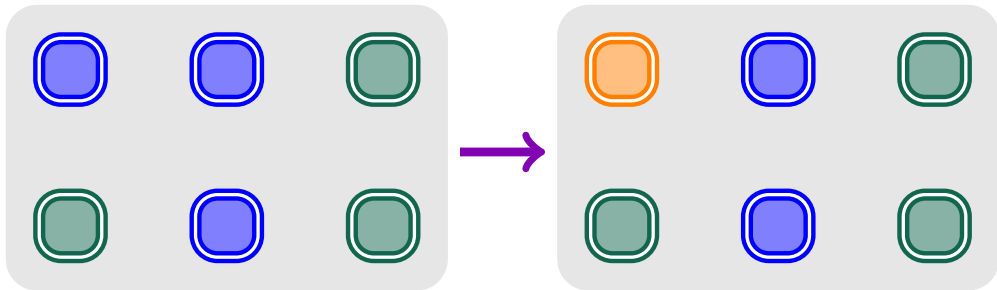
Broadcast Networks: an example



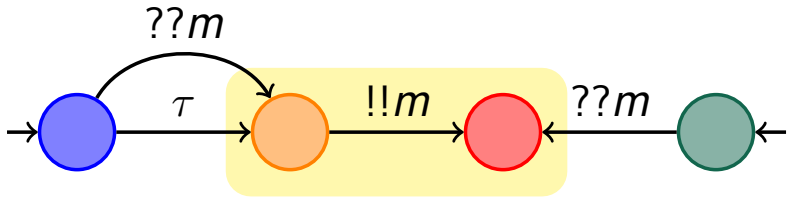


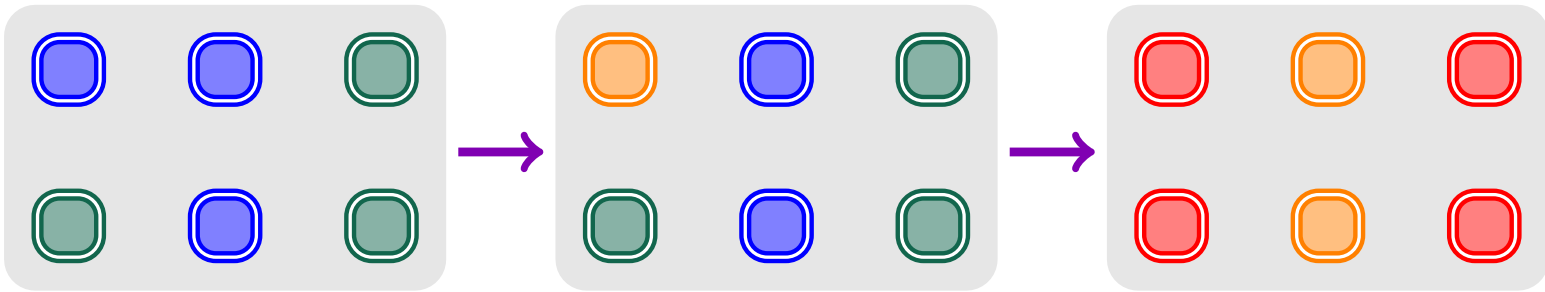
Broadcast Networks: an example



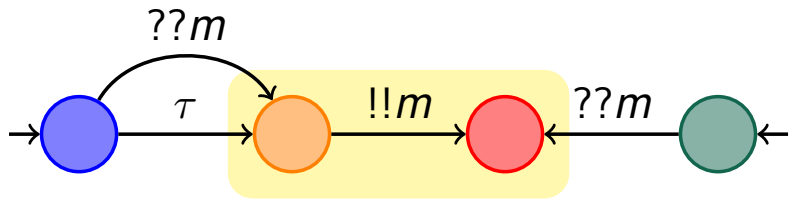


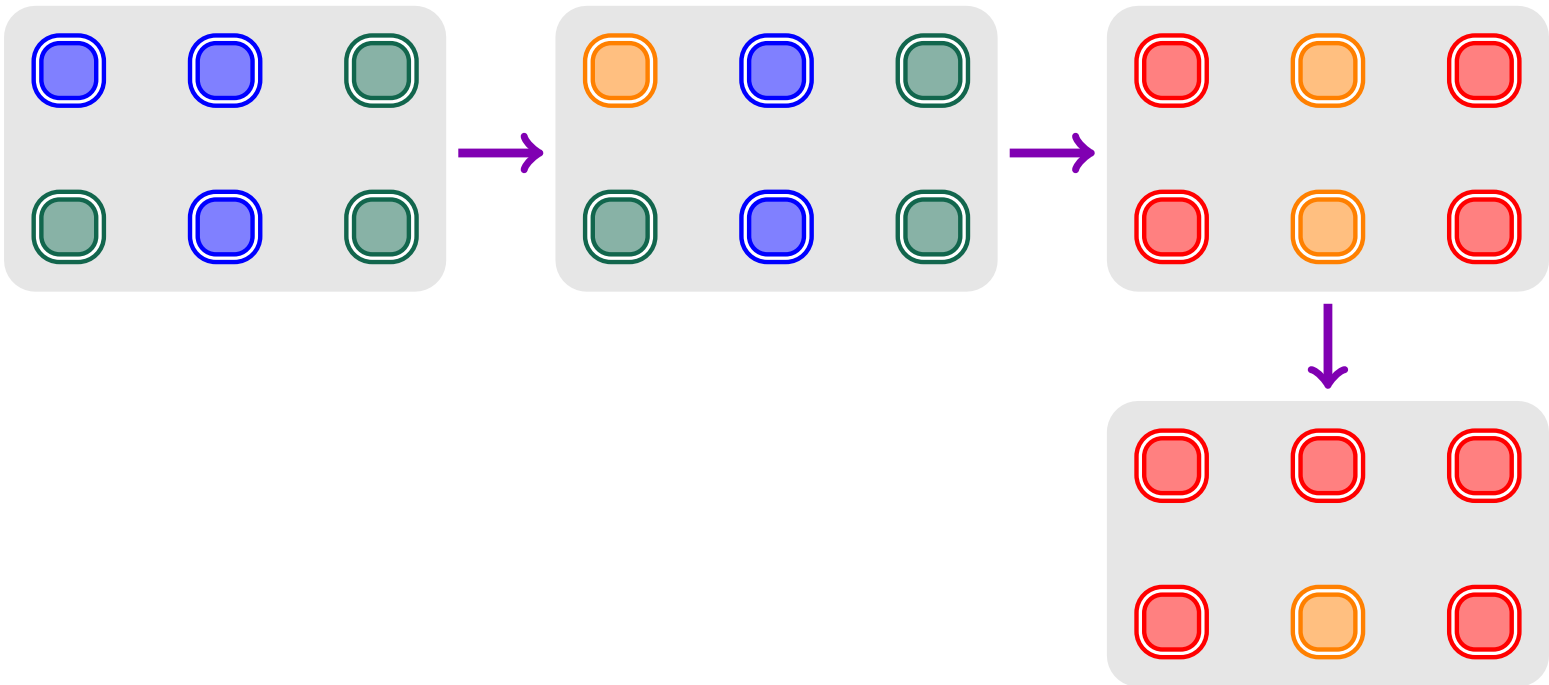
Broadcast Networks: an example



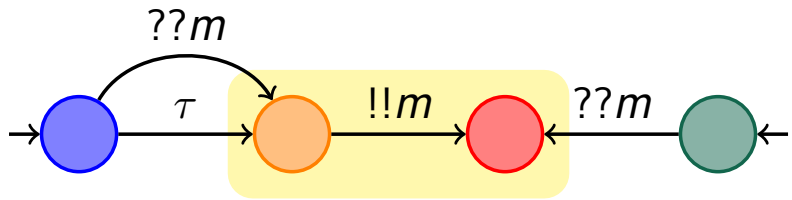


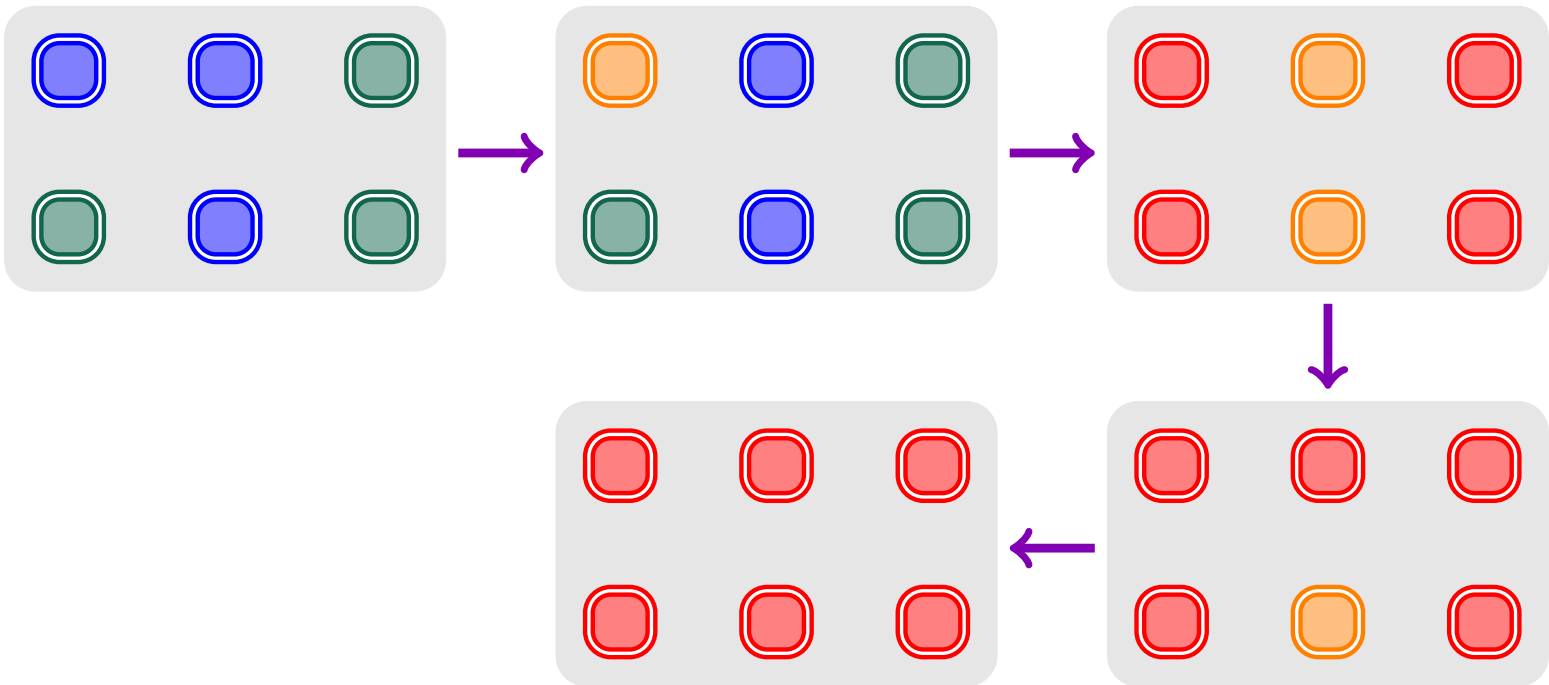
Broadcast Networks: an example



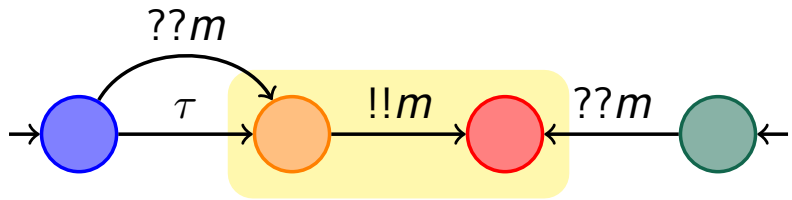


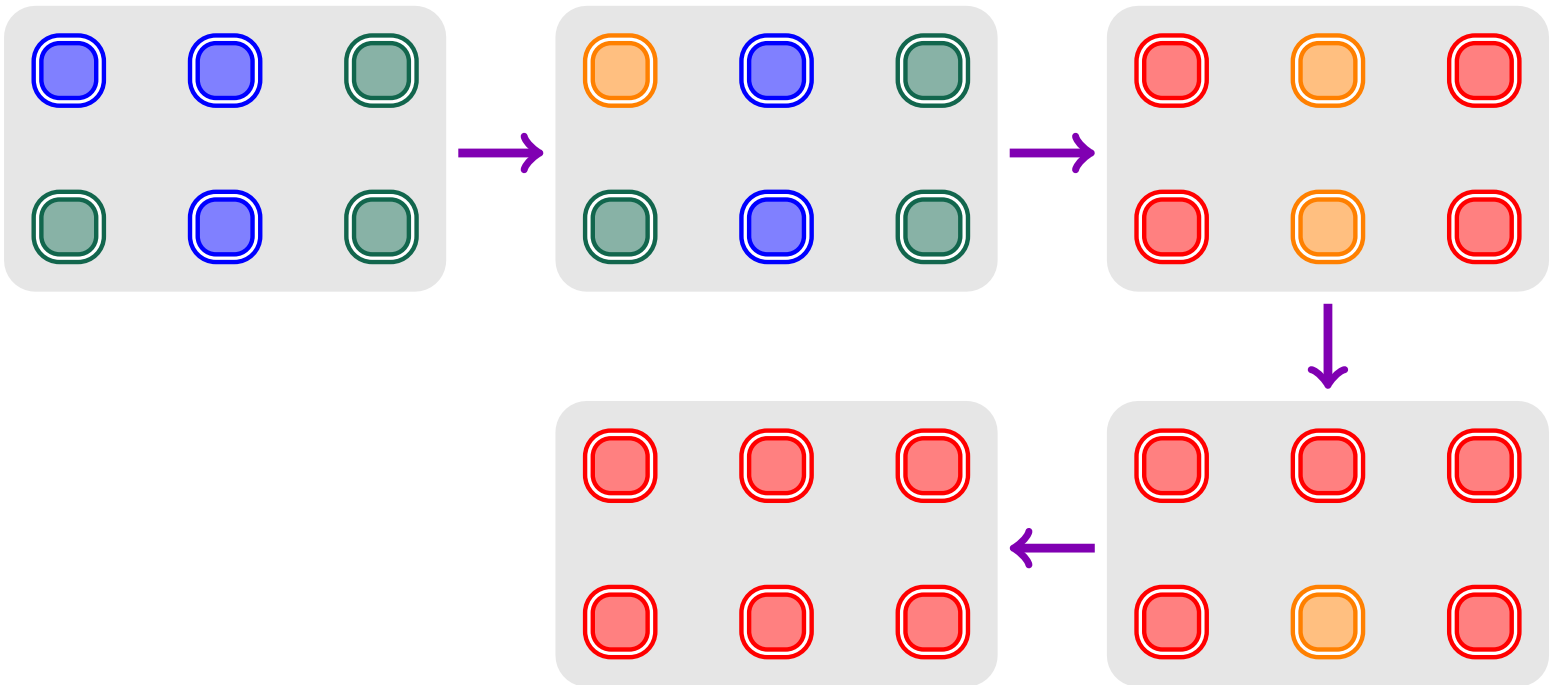
Broadcast Networks: an example





Broadcast Networks: an example





One cannot simulate broadcast with rendez-vous

WQO and upward closed sets

Well Quasi Ordering (wqo)

$(X, \leq)$ is a well-quasi ordering if for all infinite sequences $s_1, s_2, \ldots$, there exists $i < j$ such that $s_i \leq s_j$.

Upward closed set

A set $Y \subseteq X$ is upward closed w.r.t $(X, \leq)$ if $y \in Y$ and $y \leq y'$ implies $y' \in Y$.

- Upward closure of $Y \subseteq X$: $Y\uparrow = \{x \in X \mid \exists y \in Y \ \wedge \ y \leq x\}$

Lemma

If $(X, \leq)$ is a wqo and if $Y \subseteq X$ is upward closed w.r.t. $(X, \leq)$, then there exists a finite set $B \subseteq X$ s.t. $Y = B\uparrow$.

Well structured transition systems everywhere

$$\gamma \preceq \gamma' \text{ iff } \forall q \in Q, \text{ we have } \gamma(q) \leq \gamma'(q)$$

Theorem

$(\mathcal{C}, \preceq)$ is a well-quasi-ordering.

Well structured transition systems everywhere

$\gamma \preceq \gamma'$ iff $\forall q \in Q$, we have $\gamma(q) \leq \gamma'(q)$

Theorem

$(\mathcal{C}, \preceq)$ is a well-quasi-ordering.

Monotonicity lemma

For $\gamma_1, \gamma_1', \gamma_2 \in \mathcal{C}$, if

- $\gamma_1 \Rightarrow \gamma_1'$ and $\gamma_1 \preceq \gamma_2$

then there exists $\gamma_2' \in \mathcal{C}$ s.t.

- $\gamma_2 \Rightarrow \gamma_2'$ and $\gamma_1' \preceq \gamma_2'$

- BN are **Well Structured Transition Systems**
  [Abdulla et al., LICS'96; Finkel & Schnoebelen, TCS'01]

Deciding REACH in Broadcast Networks

Theorem [Esperza et al., LICS'99]

REACH is decidable for Broadcast Networks

Idea of the proof

- For $S \subseteq \mathcal{C}$, $pre(S) = \{\gamma \in \mathcal{C} \mid \gamma \Rightarrow \gamma' \ \wedge \ \gamma' \in S\}$
- if S is upward-closed, then $pre(S)$ is upward closed
- let $\Gamma : \mathcal{C} \mapsto \mathcal{C}$ s.t. $\Gamma(S) = S \cup pre(S)$
- For S upward-closed, there exists $i \in \mathbb{N}$ s.t. $\Gamma^{i+1}(S) = \Gamma^{i}(S)$ and given a finite basis B of S, one can compute a finite basis B' of $\Gamma^{i}(S)$
- Take for S the configuration γ such that $\gamma(q) = 1$ and $\gamma(q') = 0$ for all $q' \neq q$

Deciding REACH in Broadcast Networks

Theorem [Esperza et al., LICS'99]

REACH is decidable for Broadcast Networks

Idea of the proof

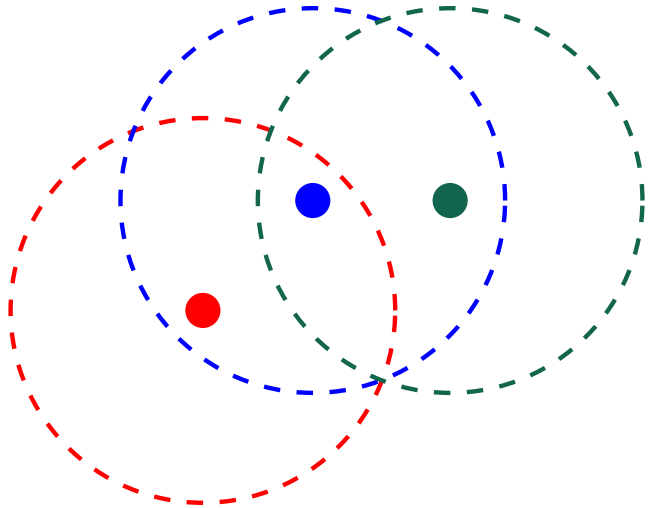
- For $S \subseteq \mathcal{C}$, $pre(S) = \{\gamma \in \mathcal{C} \mid \gamma \Rightarrow \gamma' \ \wedge \ \gamma' \in S\}$
- if S is upward-closed, then $pre(S)$ is upward closed
- let $\Gamma : \mathcal{C} \mapsto \mathcal{C}$ s.t. $\Gamma(S) = S \cup pre(S)$
- For S upward-closed, there exists $i \in \mathbb{N}$ s.t. $\Gamma^{i+1}(S) = \Gamma^i(S)$ and given a finite basis B of S, one can compute a finite basis B' of $\Gamma^i(S)$
- Take for S the configuration γ such that $\gamma(q) = 1$ and $\gamma(q') = 0$ for all $q' \neq q$

Theorem [Schmitz & Schnoebelen, CONCUR'13]

REACH for Broadcast Networks is Ackermann-complete.

Outline

Ad Hoc Networks



Main characteristics of Ad Hoc Networks

- Nodes can be mobile
- Topology is not known a priori
- Messages are broadcasted to the neighbours
- Problems linked to communication (collision, loss of messages, etc.)

Defining a model for Ad Hoc Networks

Main characteristics [Delzanno et al., CONCUR'10]

- No creation/deletion of nodes
- Each node executes the same finite state process
- Model based on the ω-calculus
- Broadcast of the messages to the neighbors
- Static topology represented by a connectivity graph

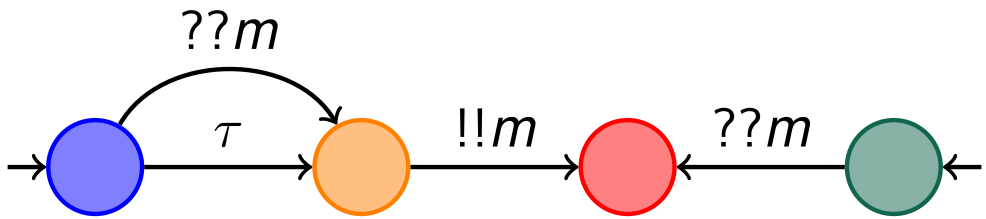
Ad Hoc Networks: syntax

A protocol $P = \langle Q, \Sigma, R, Q_0 \rangle$

Finite state system whose transitions are labeled with:

1. broadcast of messages - $!!m$
2. reception of messages - $??m$
3. internal actions - τ

where m belongs to the finite alphabet Σ

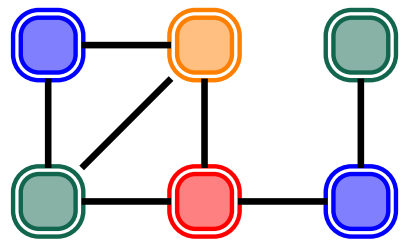


A protocol defines an Ad Hoc Network (AHN)

Ad Hoc Networks: configurations

A configuration is a graph $\gamma = \langle V, E, L \rangle$

- V : finite set of vertices
- $E : V \times V$: finite set of edges
- $L : V \to Q$: labeling function



- **Initial configurations**: **all** vertices are labeled with initial states
- *Notation : $L(\gamma)$ all the labels present in γ*

Remarks:

- The size of the considered graphs is not bounded
- Infinite number of configurations

$\Rightarrow$ **BN are infinite state systems**

Ad Hoc Networks: semantics

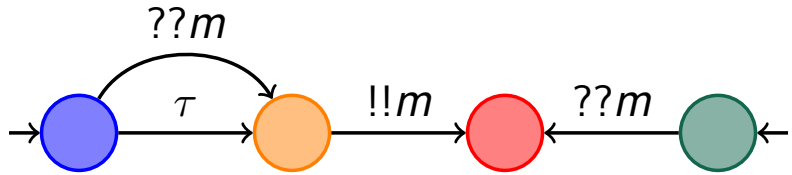
Transition system $BN(P) = \langle \mathcal{C}, \rightarrow, \mathcal{C}_0 \rangle$ associated to P

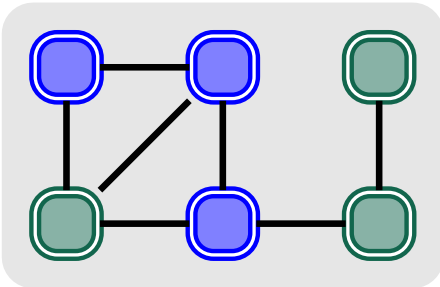
- $\mathcal{C}$: set of configurations
- $\rightarrow$: $\mathcal{C} \times \mathcal{C}$: transition relation
- $\mathcal{C}_0$: initial configurations

The relation $\rightarrow$ respects the following rules during an execution:

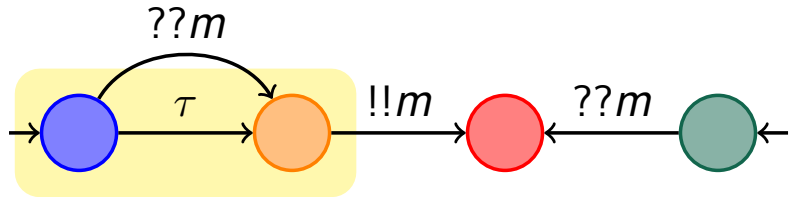
- The topology remains **static**
  - The number of vertices does not change
  - The edges do not change
  - Only the labels of the vertices can evolve
- Two kind of transitions according to the given protocol
  1. **local actions** - one process performs an internal action τ
  2. **broadcast** - one process emits a message with $!!m$, all its neighbors that can receive it with $??m$ have to receive it

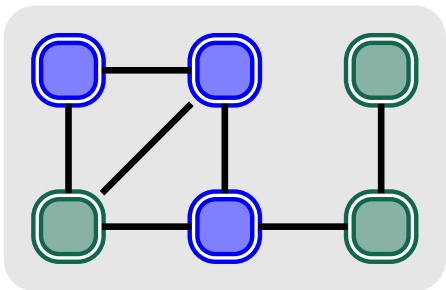
Ad Hoc Networks: an example



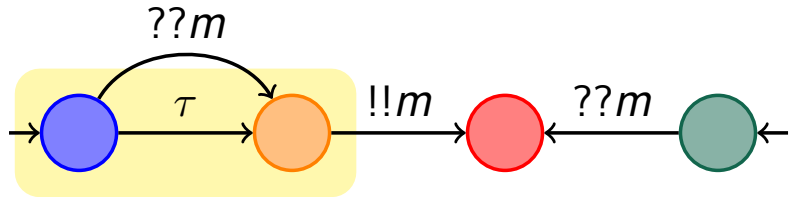


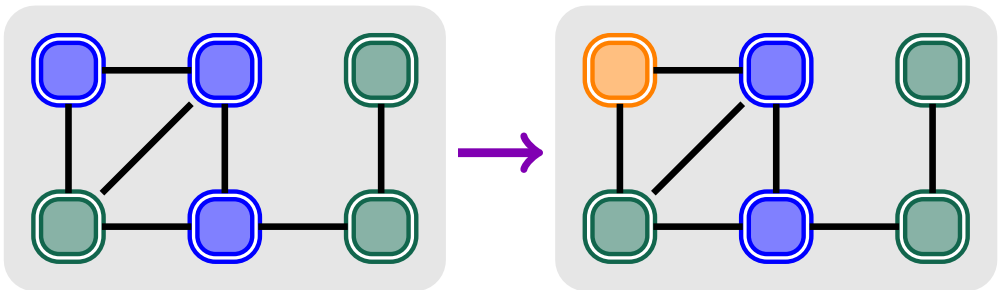
Ad Hoc Networks: an example



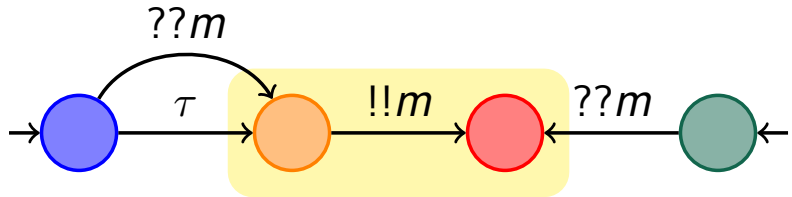


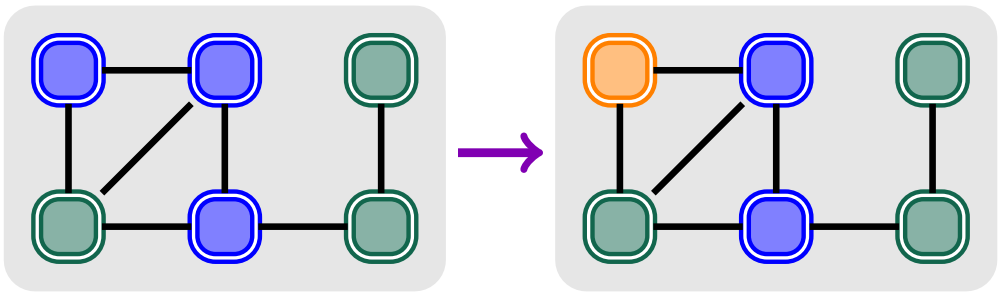
Ad Hoc Networks: an example



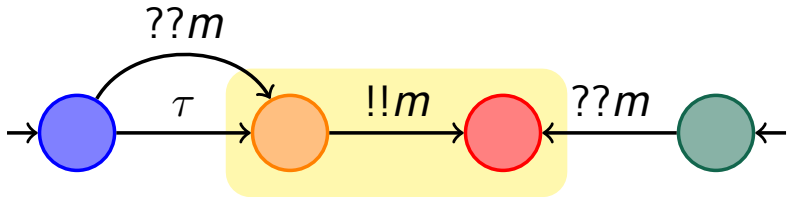


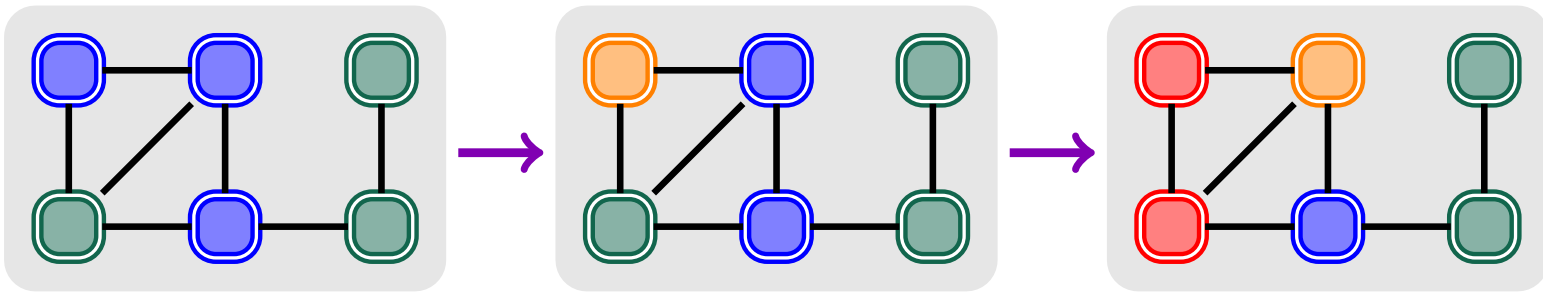
Ad Hoc Networks: an example



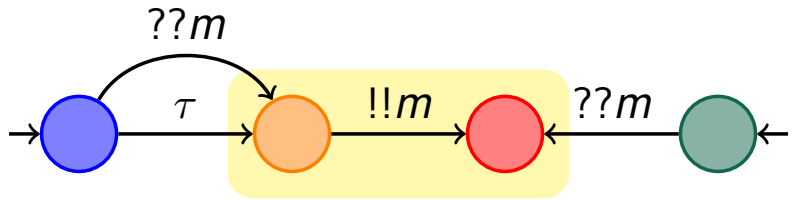


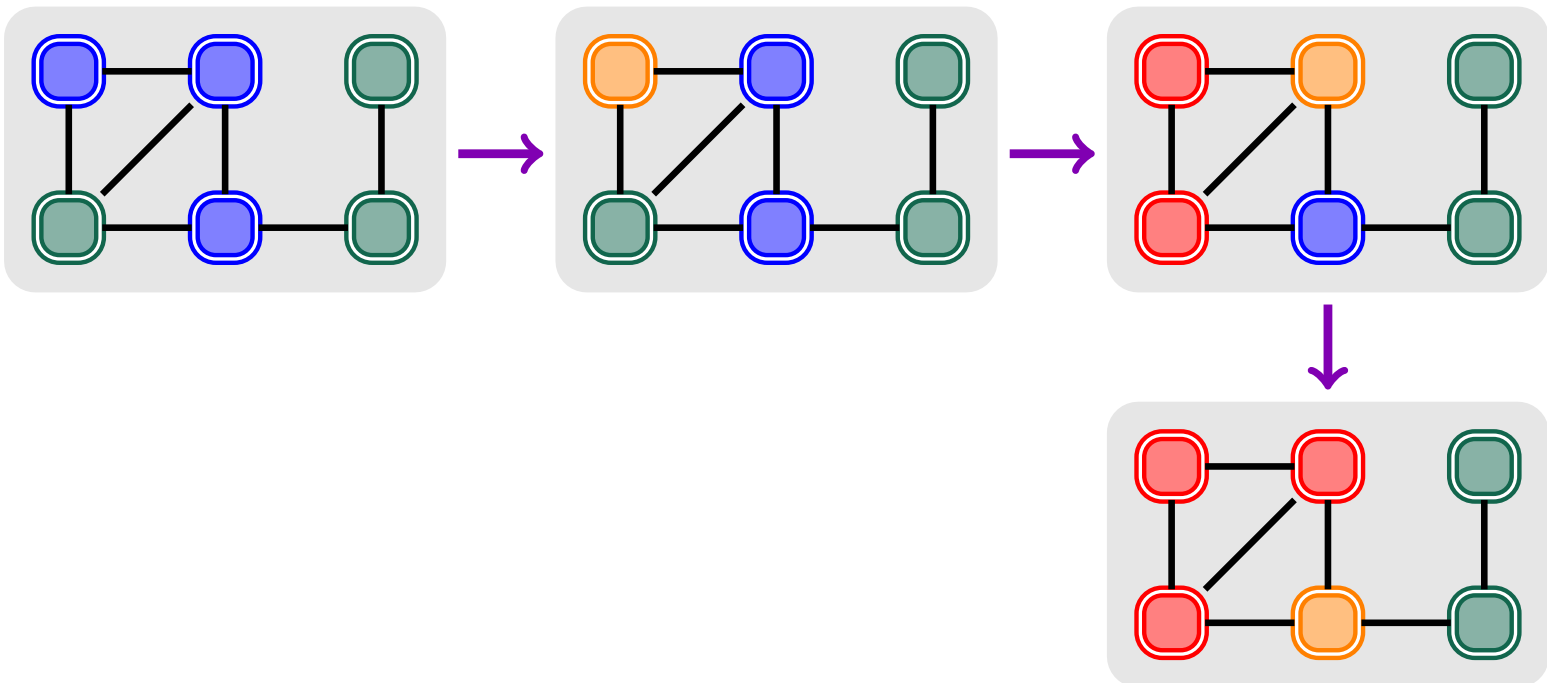
Ad Hoc Networks: an example



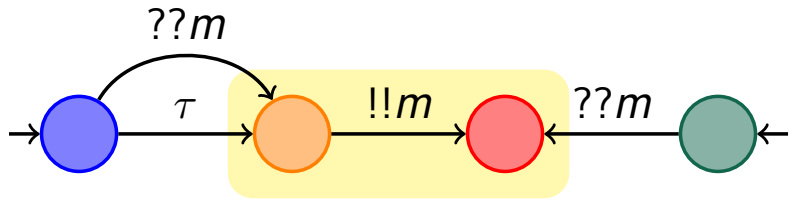


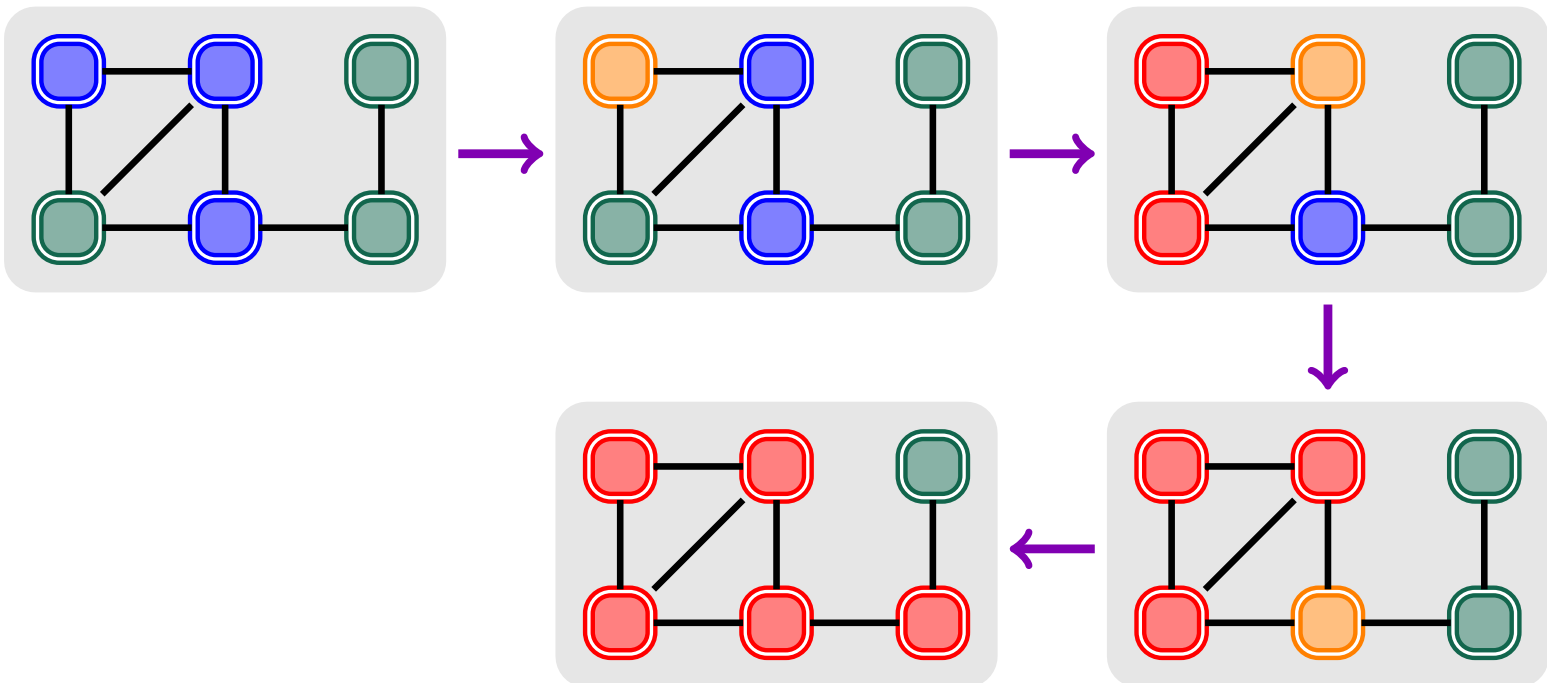
Ad Hoc Networks: an example





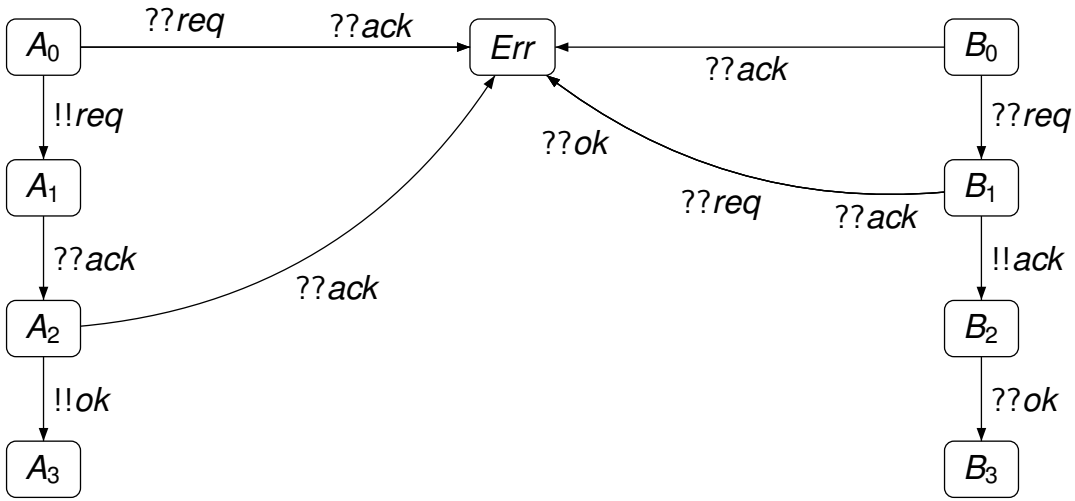
Ad Hoc Networks: an example





Ensuring the form of a topology

The Req/Ack/Ok-protocol

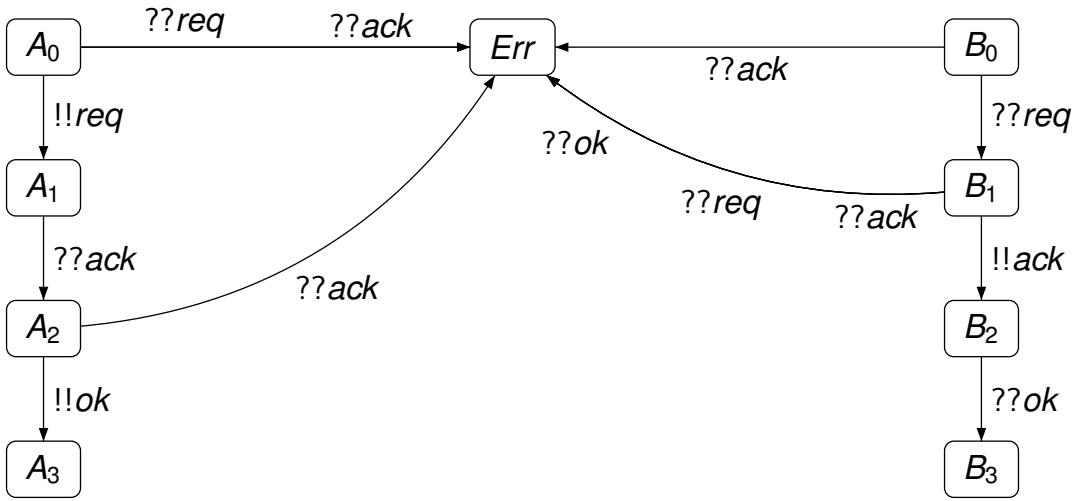


Properties

B_3

Ensuring the form of a topology

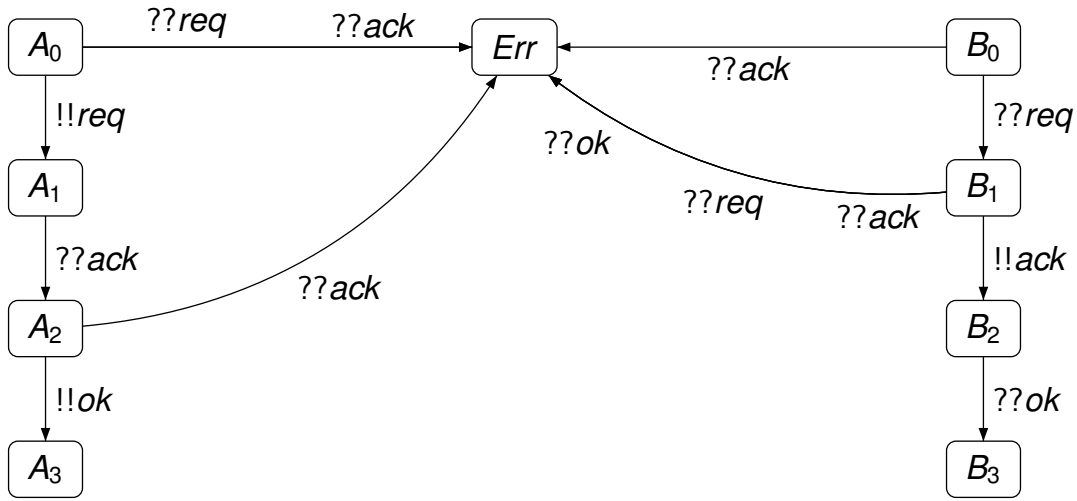
The Req/Ack/Ok-protocol



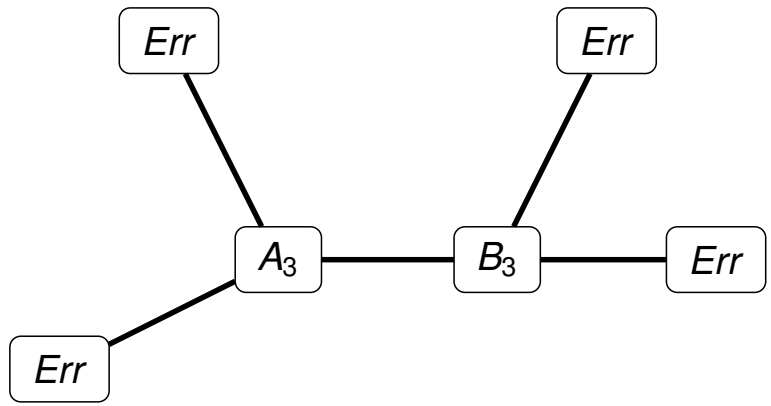
Properties

Ensuring the form of a topology

The Req/Ack/Ok-protocol

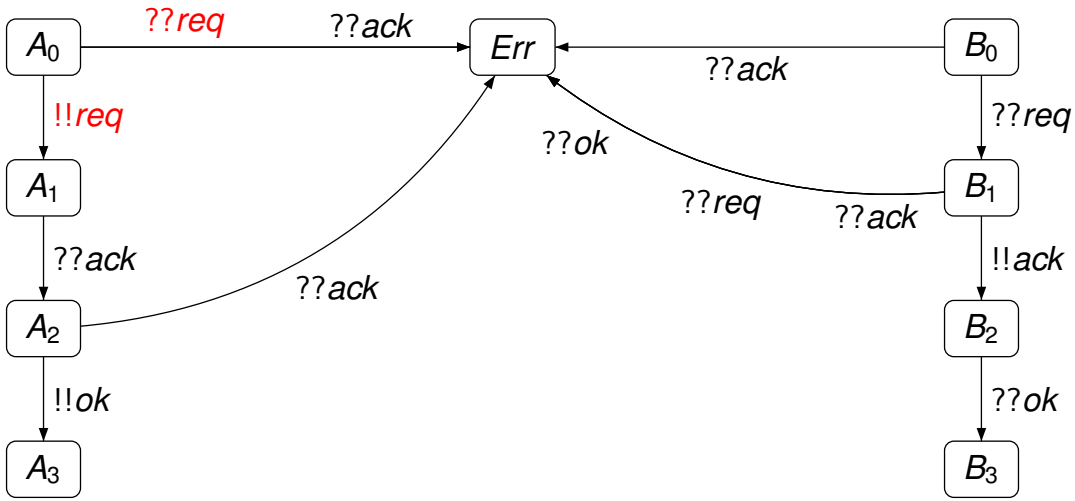


Properties

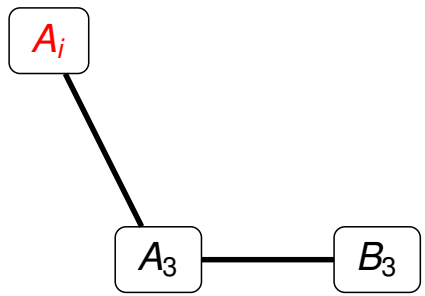


Ensuring the form of a topology

The Req/Ack/Ok-protocol

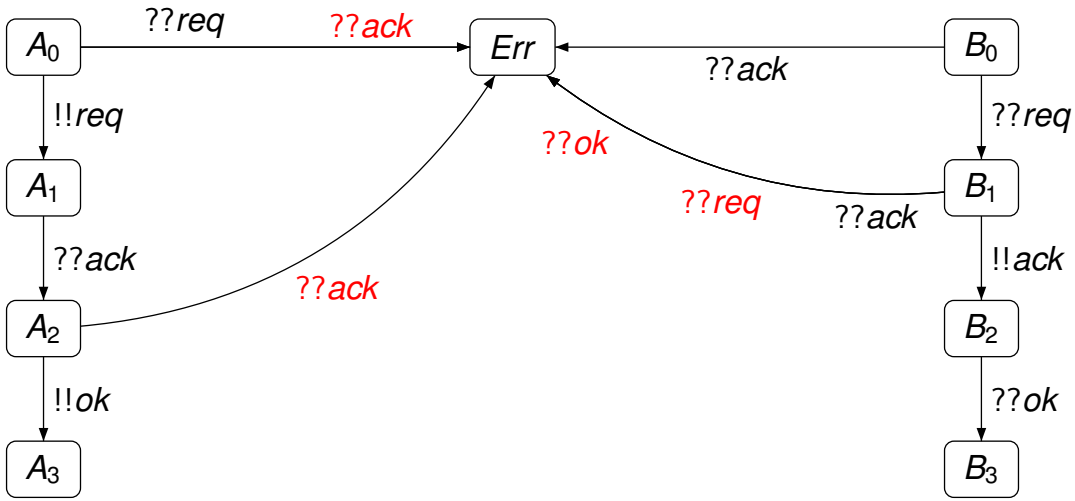


Properties

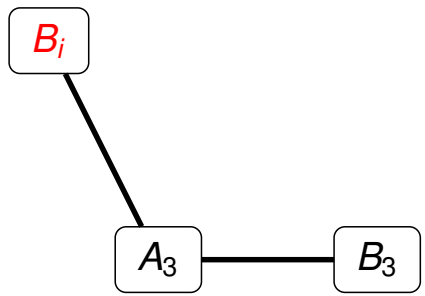


Ensuring the form of a topology

The Req/Ack/Ok-protocol

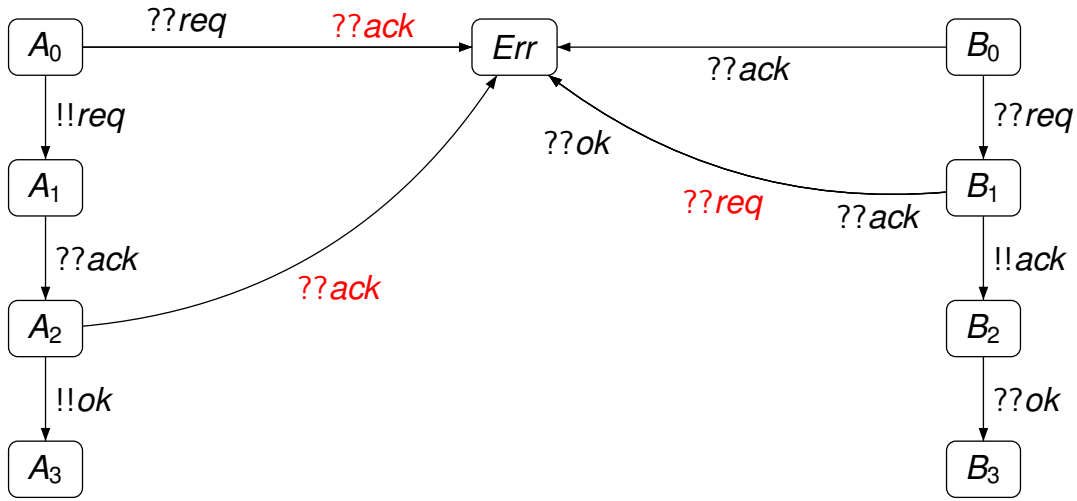


Properties

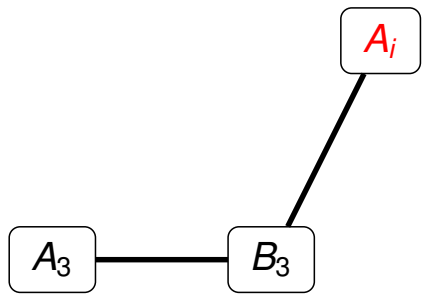


Ensuring the form of a topology

The Req/Ack/Ok-protocol

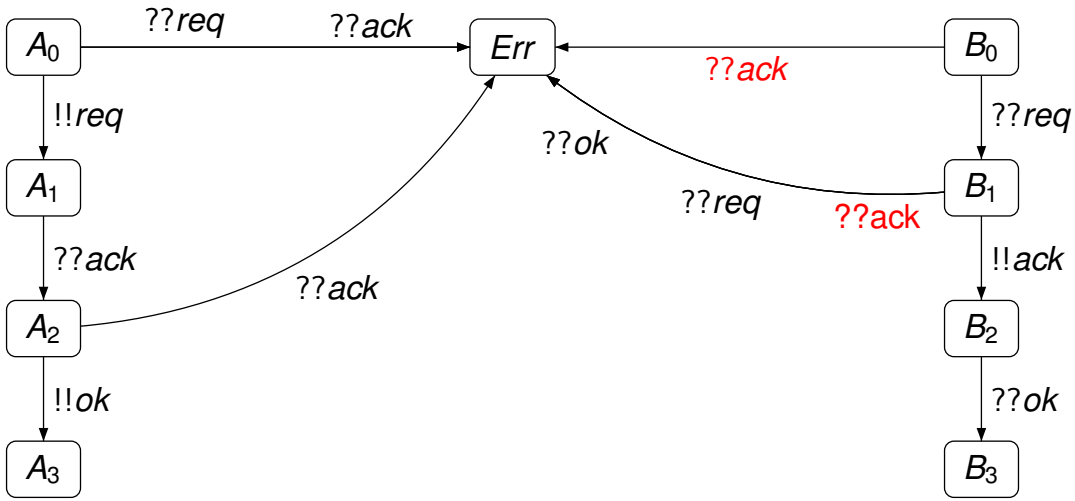


Properties

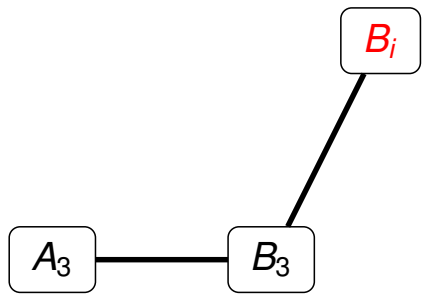


Ensuring the form of a topology

The Req/Ack/Ok-protocol

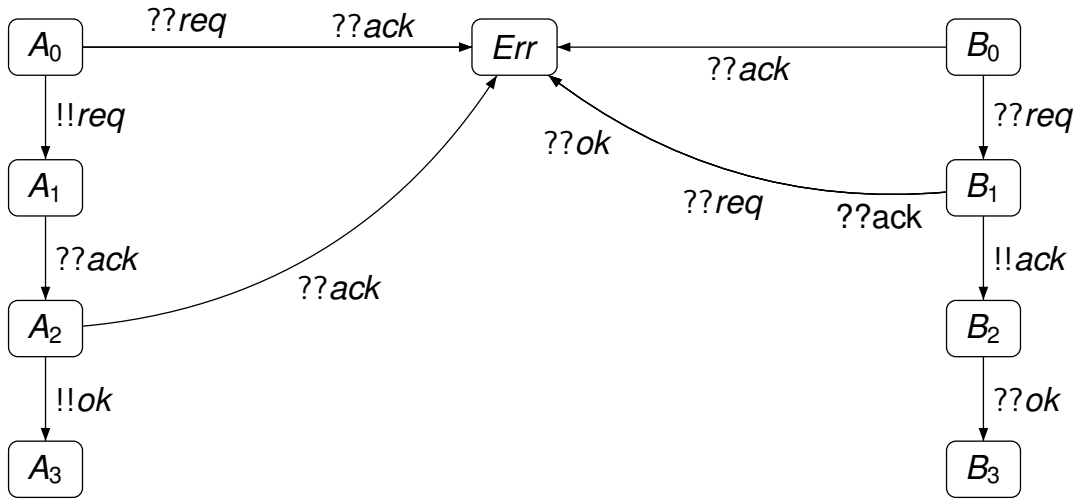


Properties

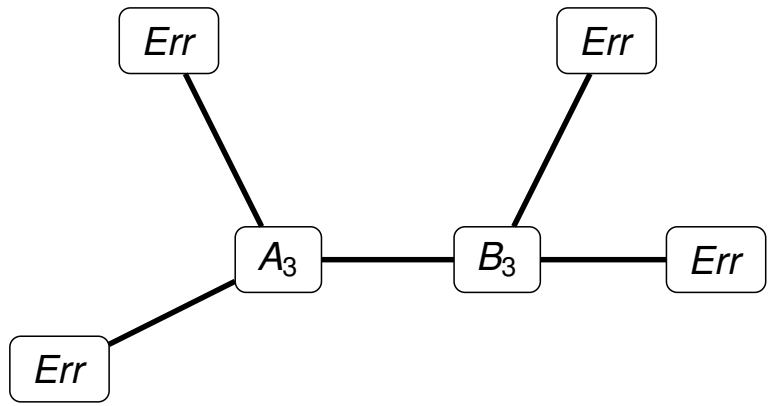


Ensuring the form of a topology

The Req/Ack/Ok-protocol



Properties



Encoding Minsky machine to prove undecidability

Minsky machine

- Manipulates two counters c_1 and c_2
- Finite set of labeled instructions of the form:
  1. $L : c_i := c_i + 1$; goto L'
  2. L : if $c_i = 0$ goto L' else $c_i := c_i - 1$; goto L''
- An initial label L_0
- A special label L_F with no output instruction

Halting problem: Is the label L_F eventually reached?

Theorem [Minsky, 67]

The halting problem for Minsky machines is undecidable.

Undecidability result

Theorem [Delzanno et al, CONCUR'10]

REACH for Ad Hoc Networks is undecidable.

Idea of the proof:

- Ensure that a topology is in a certain form
- Simulate the behavior of a Minsky machine

Undecidability result

Theorem [Delzanno et al, CONCUR'10]

REACH for Ad Hoc Networks is undecidable.

Idea of the proof:

- Ensure that a topology is in a certain form
- Simulate the behavior of a Minsky machine

One way to regain decidability:
restrict the considered graphs

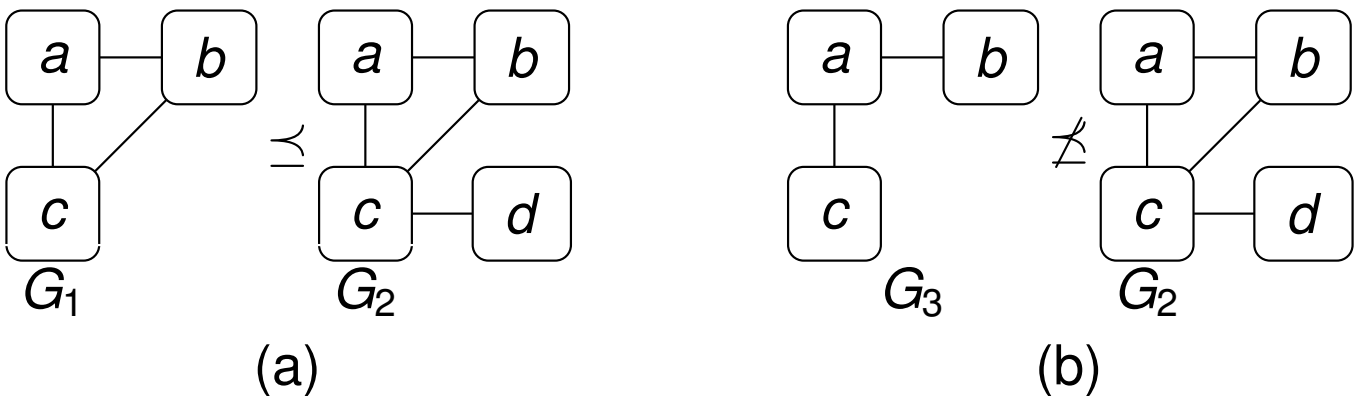
Considered order on graphs

- Given $\gamma \in \mathcal{C}$, $G(\gamma)$ is the associated graph

Induced subgraph relation

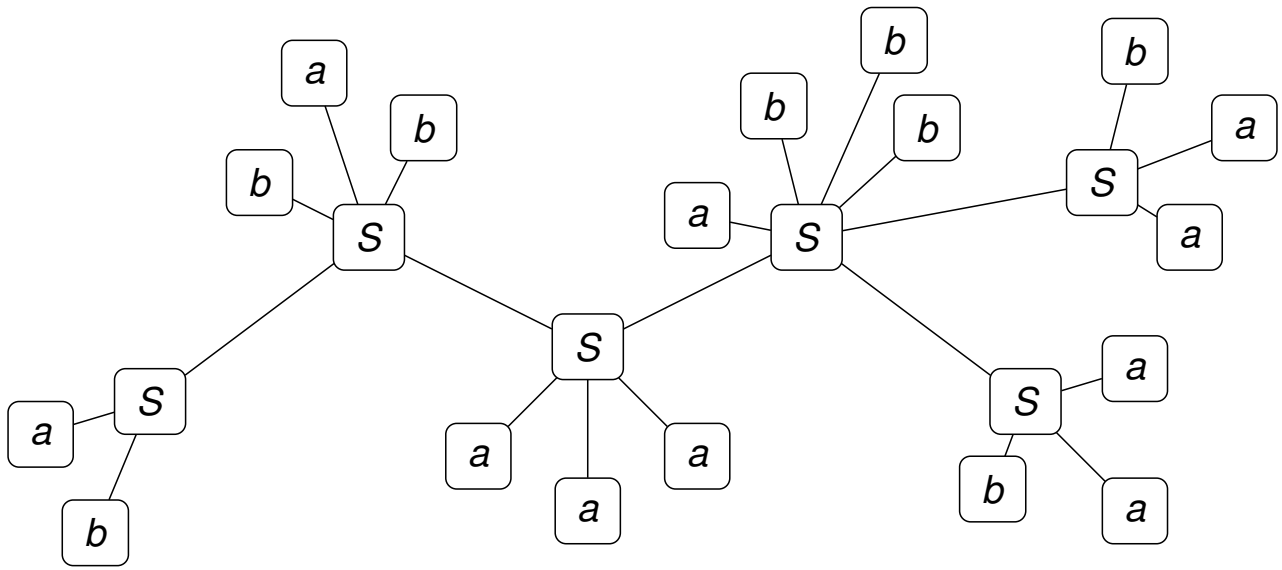
Given $\gamma_1, \gamma_2 \in \mathcal{C}$, $\gamma_1 \preceq \gamma_2$ if there exists a label preserving injection h from nodes of $G(\gamma_1)$ to nodes of $G(\gamma_2)$ s.t.:

- (n, n') is an edge in $G(\gamma_1)$ **if and only if** $(h(n), h(n')$ is an edge in $G(\gamma_2)$



Bounded path configurations

- $\mathcal{P}^K$: set of configurations $\gamma \in \mathcal{C}$ s.t. the length of the longest simple path in $G(\gamma)$ is smaller than K



Theorem [Ding, J. of Graph Theory'92]

For all $K \in \mathbb{N}$, $(\mathcal{P}^K, \preceq)$ is a well-quasi-ordering

Well structured transition systems everywhere

Monotonicity lemma

For $\gamma_1, \gamma_1', \gamma_2 \in \mathcal{P}^{\mathcal{K}}$, if

- $\gamma_1 \Rightarrow \gamma_1'$ and $\gamma_1 \preceq \gamma_2$

then there exists $\gamma_2' \in \mathcal{P}^{\mathcal{K}}$ s.t.

- $\gamma_2 \Rightarrow \gamma_2'$ and $\gamma_1' \preceq \gamma_2'$

- AHN restricted to K-bounded path configurations are **Well Structured Transition Systems**

Remark:

- This is true with induced subgraph but not with subgraph (*Node c broadcast a message received by node a and b)*

Well structured transition systems everywhere

Monotonicity lemma

For $\gamma_1, \gamma_1', \gamma_2 \in \mathcal{P}^{\mathcal{K}}$, if

- $\gamma_1 \Rightarrow \gamma_1'$ and $\gamma_1 \preceq \gamma_2$

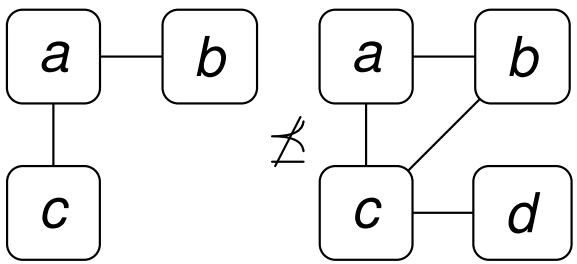
then there exists $\gamma_2' \in \mathcal{P}^{\mathcal{K}}$ s.t.

- $\gamma_2 \Rightarrow \gamma_2'$ and $\gamma_1' \preceq \gamma_2'$

- AHN restricted to K-bounded path configurations are **Well Structured Transition Systems**

Remark:

- This is true with induced subgraph but not with subgraph (*Node c broadcast a message received by node a and b)*



Decidability result

Theorem [Delzanno et al., CONCUR'10]

REACH is decidable for AHN restricted to K-bounded path configurations

Idea of the proof

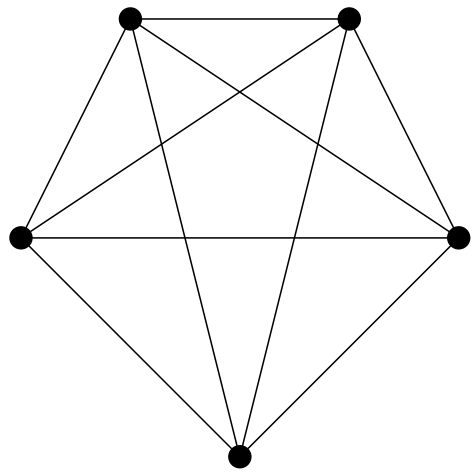
- For $S \subseteq \mathcal{P}^K$, $pre_K(S) = \{\gamma \in \mathcal{P}^K \mid \gamma \Rightarrow \gamma' \ \wedge \ \gamma' \in S\}$
- if S is upward-closed, then $pre_K(S)$ is upward closed
- let $\Gamma : \mathcal{P}^K \mapsto \mathcal{P}^K$ s.t. $\Gamma(S) = S \cup pre_K(S)$
- For S upward-closed, there exists $i \in \mathbb{N}$ s.t. $\Gamma^{i+1}(S) = \Gamma^i(S)$ and given a finite basis B of S, one can compute a finite basis B' of $\Gamma^i(S)$
- Take for S the graph with a single node labelled with q

The problem of cliques

Cliques in graph $\gamma = \langle V, E, L \rangle$

$V' \subseteq V$ such that for all $v, v' \in V'$, we have $(v, v') \in E$.

If $V = V'$, γ is *clique graph*.

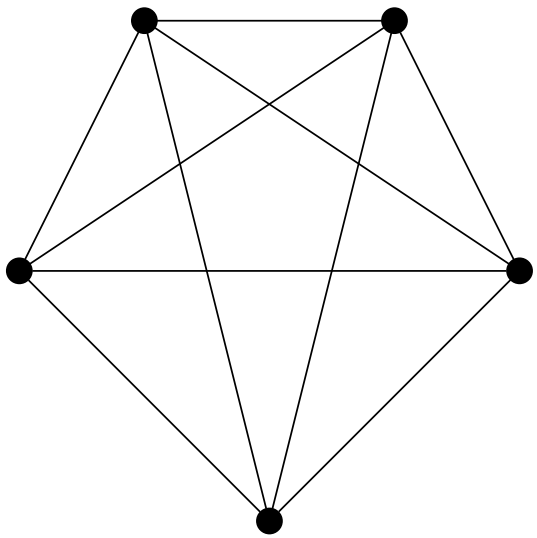


AHN restricted to clique graphs are Broadcast Networks

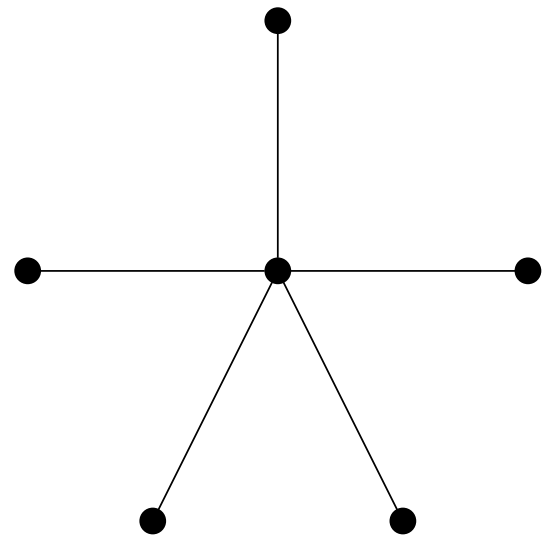
Bounded diameter graphs

Diameter of a graph

Length of the longest shortest path between any two vertices.



Diameter=1



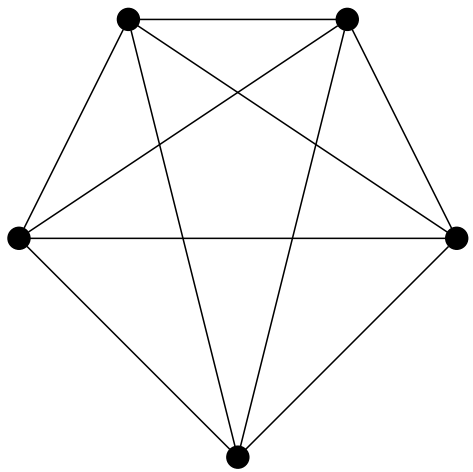
Diameter=2

$\mathcal{D}^K$: set of configurations whose diameter is smaller than K

Bounded path vs. bounded diameter

Proposition

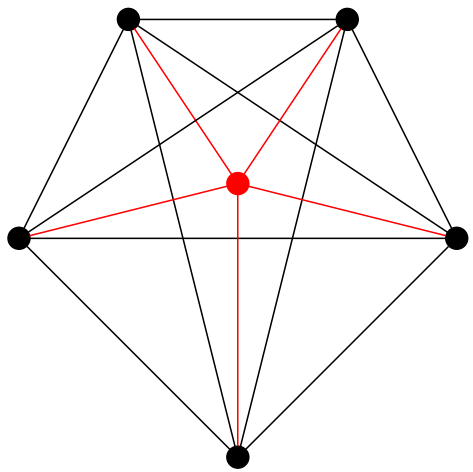
- $\mathcal{P}^K \subseteq \mathcal{D}^K$
- For every $K \in \mathbb{N}$, there exists a clique graph which does not belong to $\mathcal{P}^K$
- All clique graphs are in $\mathcal{D}^1$



Bounded path vs. bounded diameter

Proposition

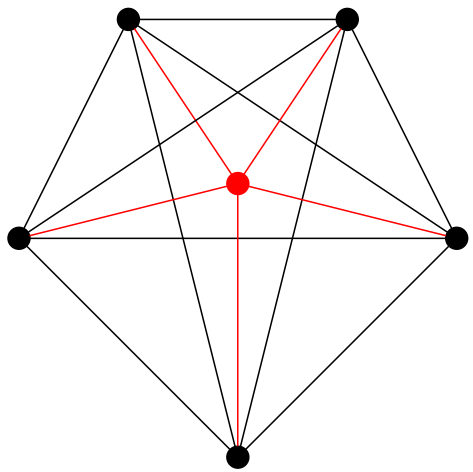
- $\mathcal{P}^K \subseteq \mathcal{D}^K$
- For every $K \in \mathbb{N}$, there exists a clique graph which does not belong to $\mathcal{P}^K$
- All clique graphs are in $\mathcal{D}^1$



Bounded path vs. bounded diameter

Proposition

- $\mathcal{P}^K \subseteq \mathcal{D}^K$
- For every $K \in \mathbb{N}$, there exists a clique graph which does not belong to $\mathcal{P}^K$
- All clique graphs are in $\mathcal{D}^1$



Theorem

For $K \geq 2$, REACH for AHN restricted to K-bounded diameter configurations is undecidable.

Restricting the class of bounded diameter graphs

Degree of a graph

The degree of a graph is the maximum number of outgoing edges for each node.

Theorem [Hoffman & Singleton, 60]

Given $K, D \in \mathbb{N}$, the number of graphs whose diameter is smaller than K and its degree is smaller that D is finite.

The number of nodes in such graphs is in fact smaller than the Moore bound:

$$M(K, D) = (K(K-1)^D - 2)/(K-2)$$

Theorem [Delzanno et al., FOSSACS'11]

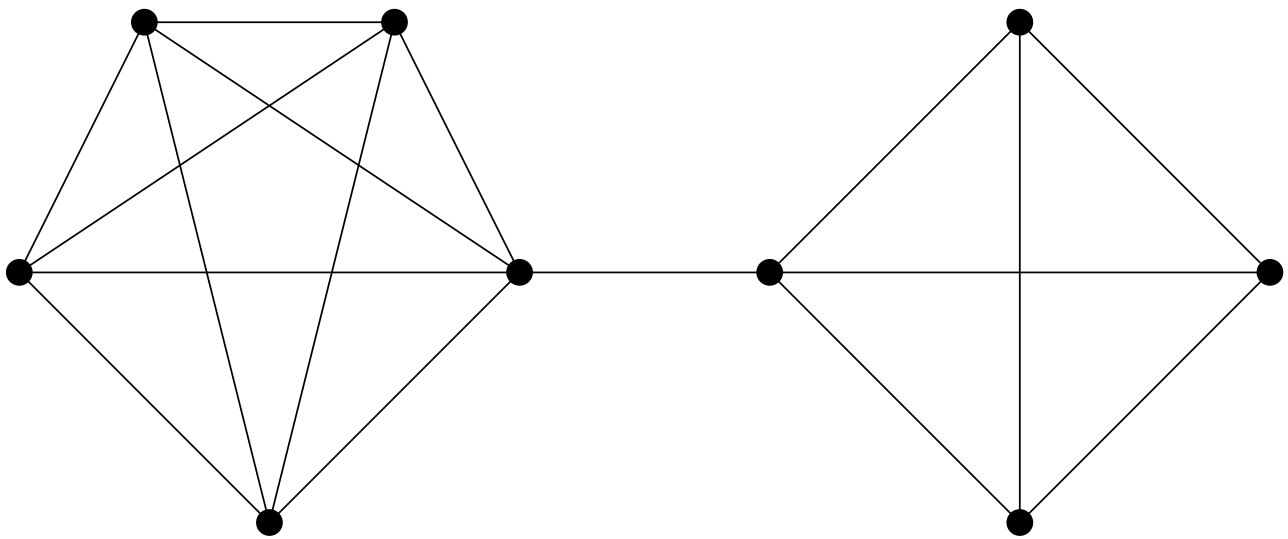
REACH in AHN restricted to configurations of bounded diameter and bounded degree is decidable.

How to capture the cliques ?

Bipartite Cliques graphs associated to $\gamma = \langle V, E, L \rangle$

$K_\gamma = \langle X, W, E', L' \rangle$ with:

- $X = V$
- $W \subseteq 2^V$ the set of maximal cliques of γ
- For $v \in V$, and $w \in W$, $(v, w) \in E'$ iff $v \in w$
- For all $v \in V$, $L'(v) = L(v)$

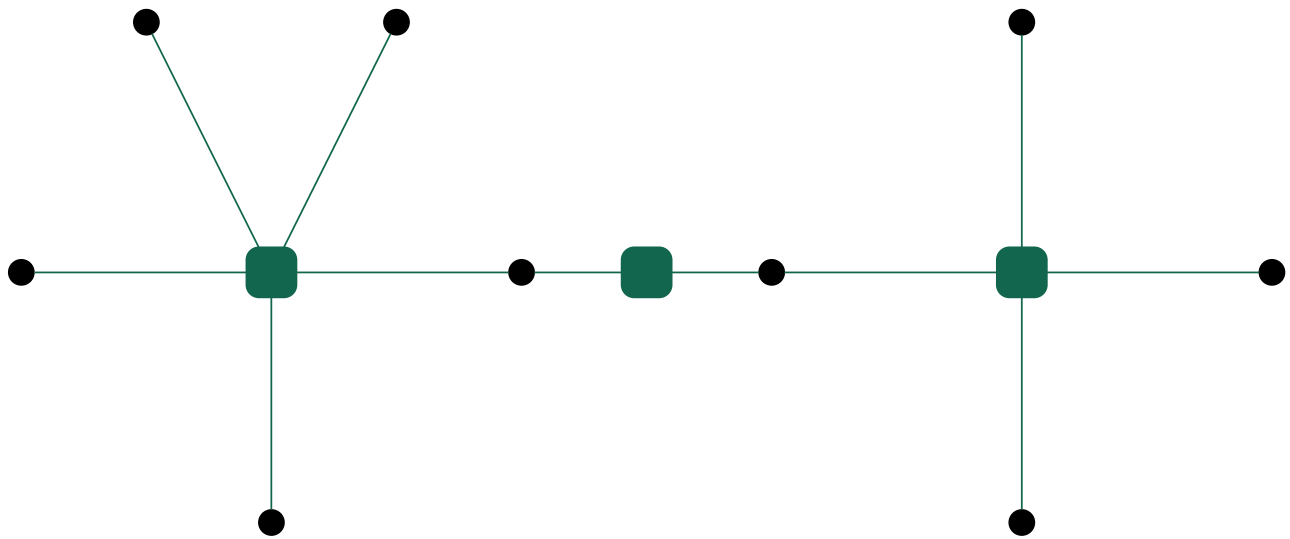


How to capture the cliques ?

Bipartite Cliques graphs associated to $\gamma = \langle V, E, L \rangle$

$K_\gamma = \langle X, W, E', L' \rangle$ with:

- $X = V$
- $W \subseteq 2^V$ the set of maximal cliques of γ
- For $v \in V$, and $w \in W$, $(v, w) \in E'$ iff $v \in w$
- For all $v \in V$, $L'(v) = L(v)$



Decidability for a bigger class of graphs

$\mathcal{PC}^K$

class of configurations γ such that K_γ is K-bounded path

Proposition

- All clique graphs are in $\mathcal{PC}^2$
- For all $K \in \mathbb{N}$, $\mathcal{P}^K \subseteq \mathcal{PC}^{2.K}$

Decidability for a bigger class of graphs

$\mathcal{PC}^K$

class of configurations γ such that K_γ is K-bounded path

Proposition

- All clique graphs are in $\mathcal{PC}^2$
- For all $K \in \mathbb{N}$, $\mathcal{P}^K \subseteq \mathcal{PC}^{2.K}$

Lemma [Delzanno et al., FOSSACS'11]

For all $K \in \mathbb{N}$, $(\mathcal{PC}^K, \preceq)$ is a well quasi ordering.

Decidability for a bigger class of graphs

$\mathcal{PC}^K$

class of configurations γ such that K_γ is K-bounded path

Proposition

- All clique graphs are in $\mathcal{PC}^2$
- For all $K \in \mathbb{N}$, $\mathcal{P}^K \subseteq \mathcal{PC}^{2.K}$

Lemma [Delzanno et al., FOSSACS'11]

For all $K \in \mathbb{N}$, $(\mathcal{PC}^K, \preceq)$ is a well quasi ordering.

Theorem [Delzanno et al., FOSSACS'11]

For all $K \in \mathbb{N}$, REACH for AHN restricted to configurations in $\mathcal{PC}^K$ is decidable and non primitive recursive.

Outline

Adding non deterministic reconfiguration

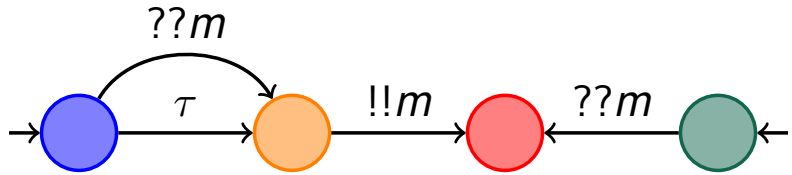
Transition system $RAN(P) = \langle \mathcal{C}, \rightarrow, \mathcal{C}_0 \rangle$ associated to P

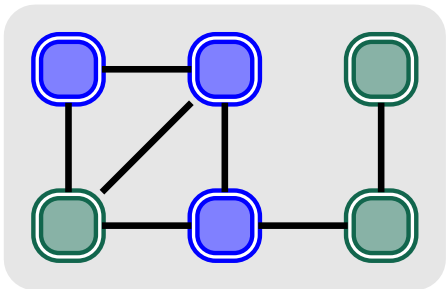
- $\mathcal{C}$: set of configurations
- $\Rightarrow$: $\mathcal{C} \times \mathcal{C}$: transition relation
- $\mathcal{C}_0$: initial configurations

The relation $\Rightarrow$ respects the following rules during an execution:

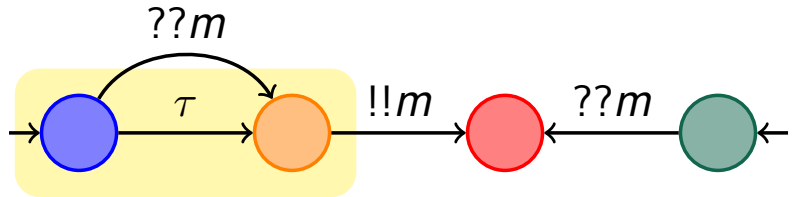
- The topology is **not static** anymore
  - The number of vertices does not change
  - The edges can change non deterministically
  - The labels of the vertices can evolve
- Three kind of transitions according to the given protocol
  1. **local actions**
  2. **broadcast**
  3. **reconfiguration** - the edges can change with no restriction

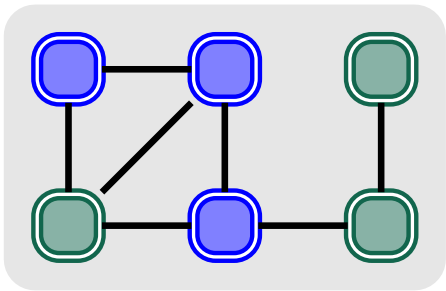
Reconfigurable Ad Hoc Networks: an example



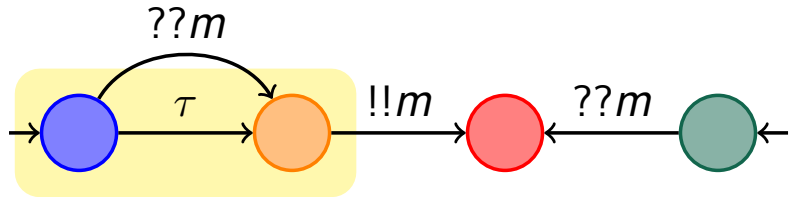


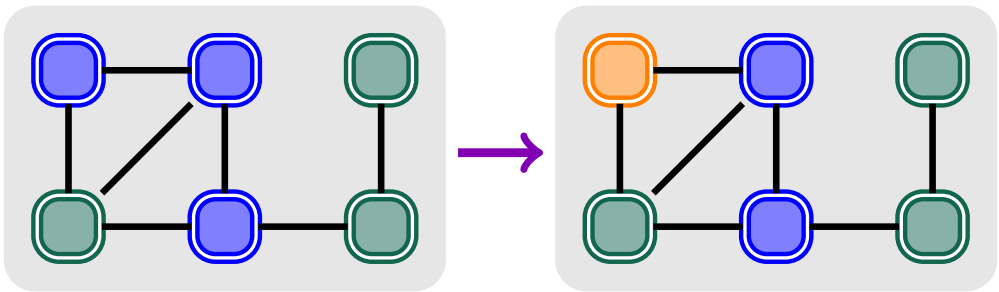
Reconfigurable Ad Hoc Networks: an example



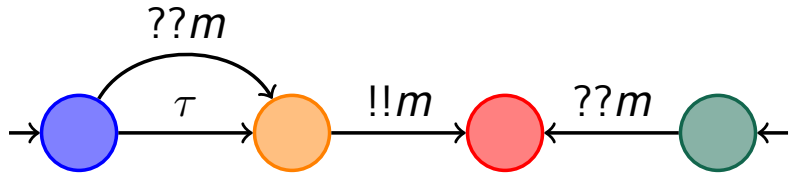


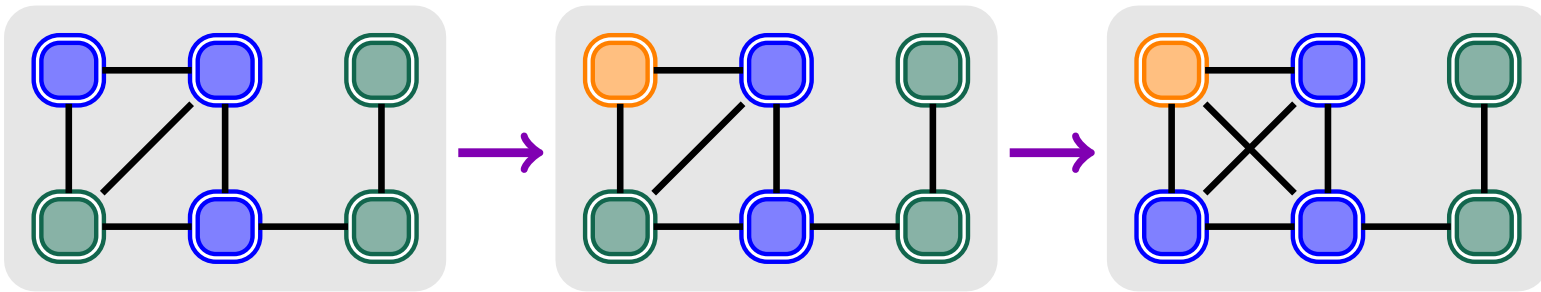
Reconfigurable Ad Hoc Networks: an example



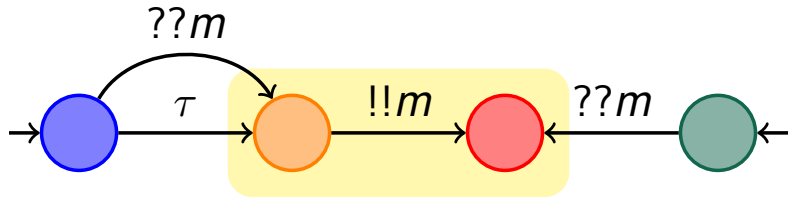


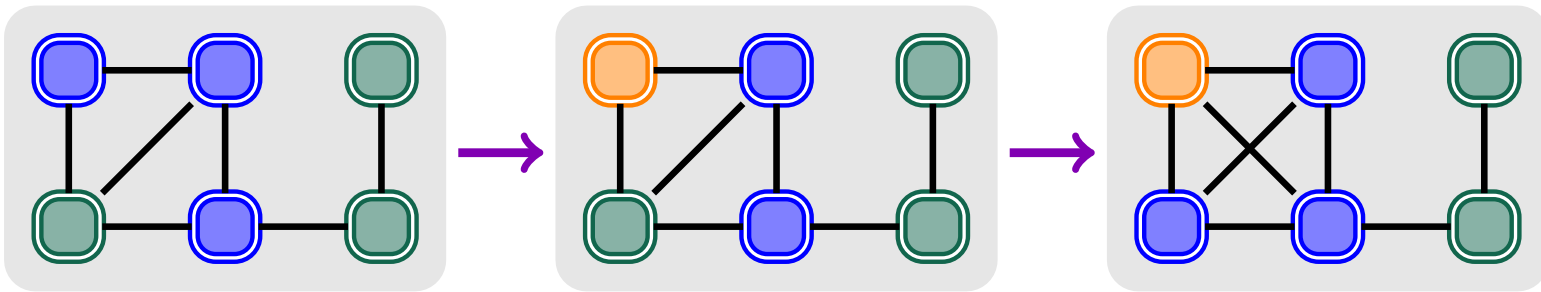
Reconfigurable Ad Hoc Networks: an example



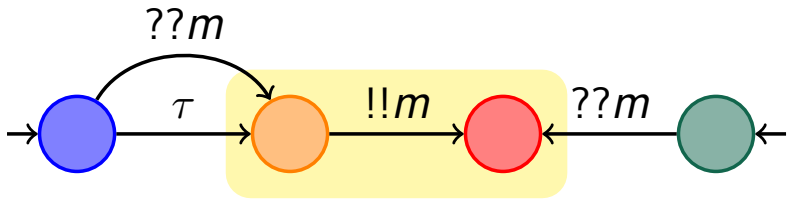


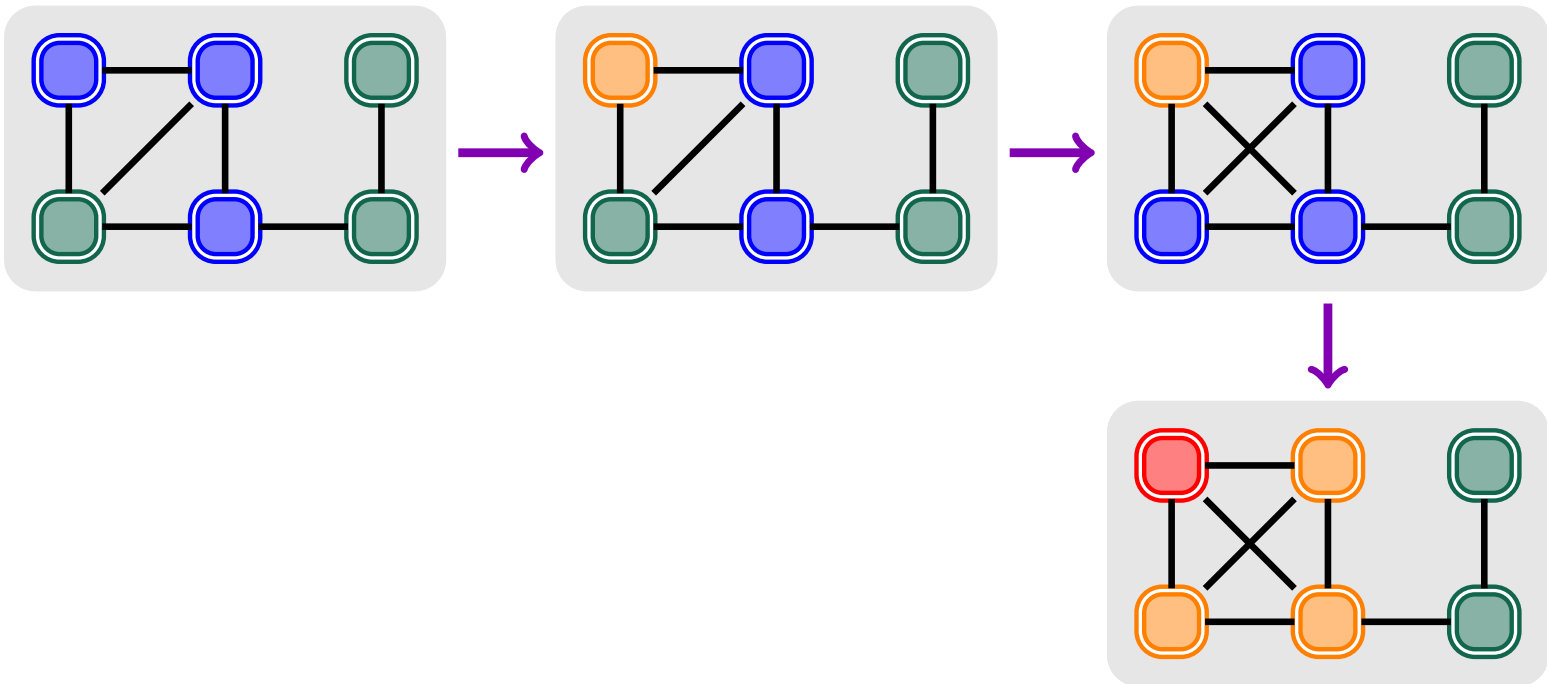
Reconfigurable Ad Hoc Networks: an example



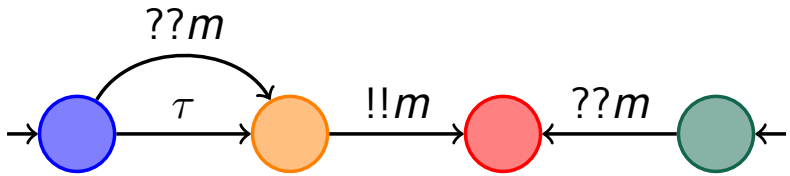


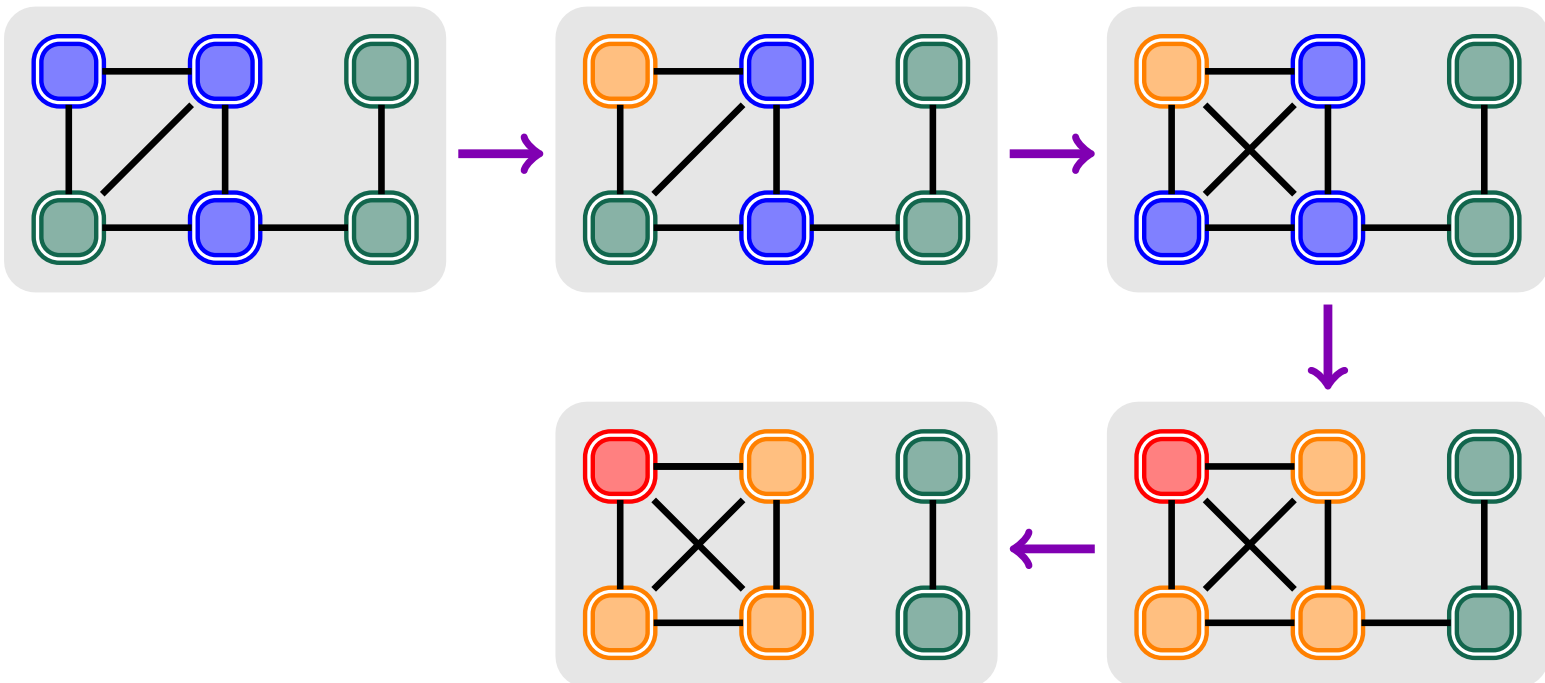
Reconfigurable Ad Hoc Networks: an example



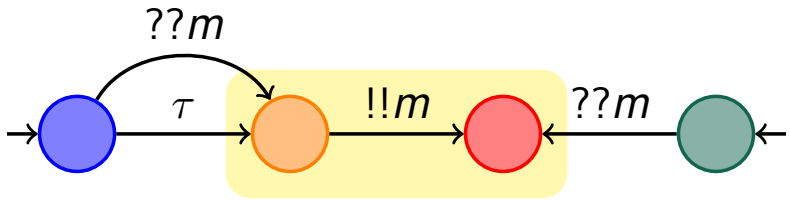


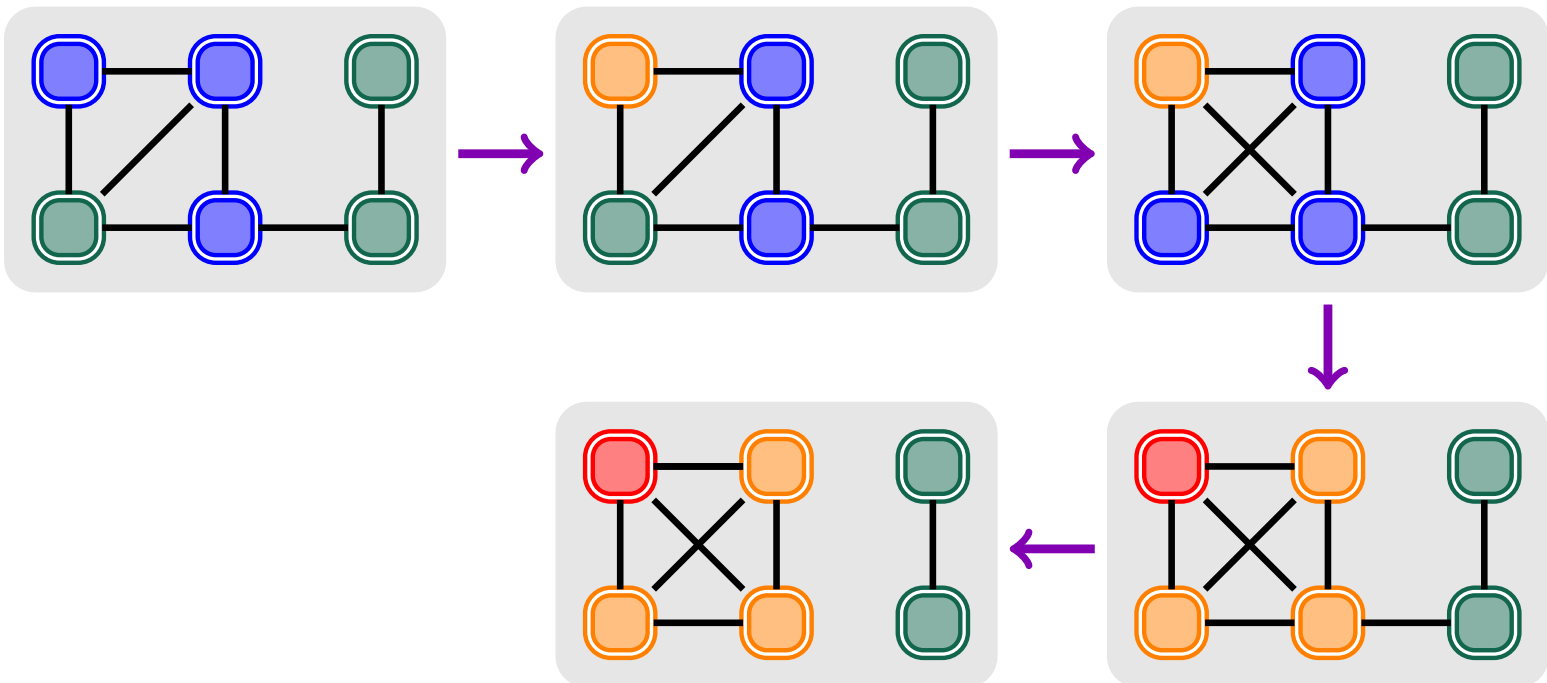
Reconfigurable Ad Hoc Networks: an example



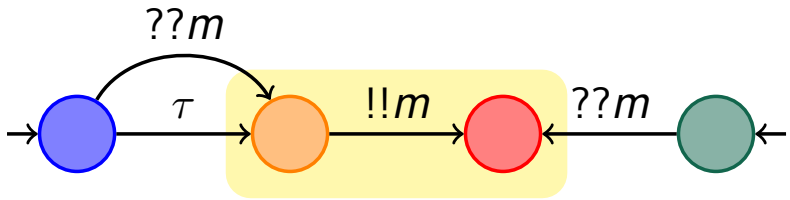


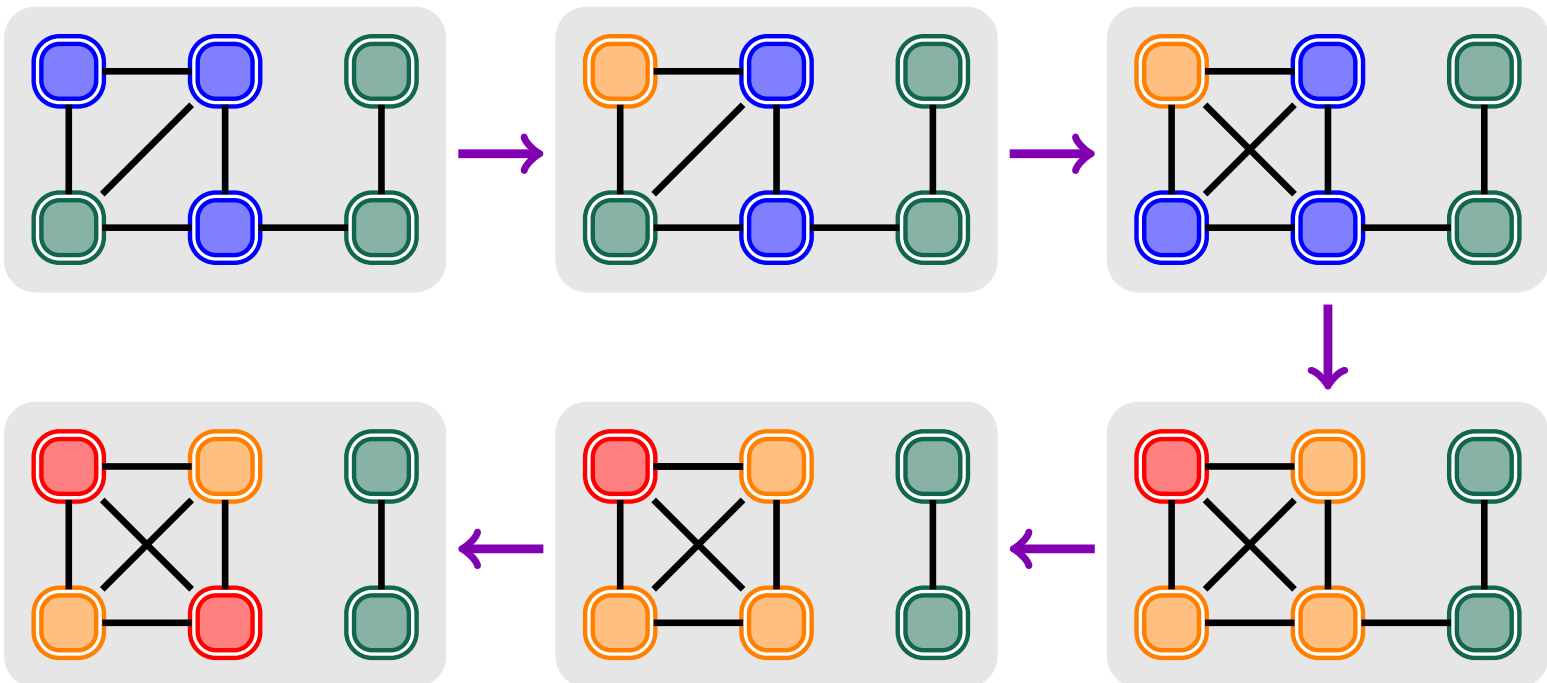
Reconfigurable Ad Hoc Networks: an example





Reconfigurable Ad Hoc Networks: an example





Cardinality constraints

Counting in Reconfigurable Ad Hoc Networks

Cardinality constraint over a protocol P

$$\varphi ::= b > \#q \geq a \mid \varphi \wedge \varphi \mid \varphi \vee \phi \mid \neg\varphi$$

with $a \in \mathbb{N}$, $b \in \mathbb{N} \setminus \{0\} \cup \{+\infty\}$ and $q \in Q$

Satisfaction relation $\models$ for cardinality constraints:

- For a configuration γ, $\gamma \models b > \#q \geq a$ iff the number of vertices labeled by q is greater than a and strictly smaller than b
- For boolean combinations, classical definition

Restrictions:

- CC[≥ 1] : negation is forbidden and atomic formuale are of the form $\#q \geq 1$
- CC[$\geq 1, = 0$] : as CC[≥ 1] adding atoms of the form $\#q = 0$

Remark: naturals are encoded in unary

Quantitative verification problem

Cardinaliy Reachability Problem (CRP)

Input: A protocol and a cardinality constraint φ;

Output: Does there exist $\gamma \in \mathcal{C}_0$ and $\gamma' \in \mathcal{C}$ s.t. $\gamma \rightarrow^* \gamma'$ and $\gamma' \models \varphi$?

- REACH can be encoded into CRP restricted to CC[$\geq$ 1]
- CRP restricted to CC[$\geq$ 1,=0] allows to test absence of label
- CRP is **decidable** using an encoding into Petri nets

What is the complexity of CRP?

CRP restricted to CC[≥ 1]

Theorem [Delzanno et al.,FSTTCS'12]

For RAN, CRP restricted to CC[≥ 1] is PTIME-complete

Idea of the proof:

- **Lower bound:** LOGSPACE reduction from the Circuit Value Problem
- **Upper bound:** algorithm which builds the set of reachable states

Corollary [Delzanno et al.,FSTTCS'12]

For RAN, Reach is PTIME-complete

Solving CRP restricted to CC[≥ 1]

PTIME algorithm to compute the set of reachable states

Input : $P = \langle Q, \Sigma, R, Q_0 \rangle$ a protocol
Output : $S \subseteq Q$ the set of reachable control states in $RAN(P)$

1: $S := Q_0$
2: $oldS := \emptyset$
3: **while** $S \neq oldS$ **do**
4: $\quad oldS := S$
5: $\quad$ **for all** $\langle q_1, !!a, q_2 \rangle \in R$ such that $q_1 \in oldS$ **do**
6: $\quad\quad S := S \cup \{q_2\} \cup \{q' \in Q \mid \langle q, ??a, q' \rangle \in R \wedge q \in oldS\}$
7: $\quad$ **end for**
8: **end while**

- Each time, do all the possible transactions in the network
- Terminates in at most $|P|$ iterations of the main loop

CRP restricted to CC[$\geq 1, = 0$]

Possibility to test for the absence of control states

Theorem [Delzanno et al.,FSTTCS'12]

For RAN, CRP restricted to CC[$\geq 1, = 0$] is NP-complete

Idea of the proof:

- **Lower bound:**
  - Reduction from 3SAT
  - No need for communication
  - For each variable v, two initial state v and $\neg v$
  - The cardinality constraint is built as follows:
    - The states v and $\neg v$ cannot both be present
    - In the 3SAT formula v is replaced by $\#v \geq 1$ and $\neg v$ by $\#(\neg v) \geq 1$
- **Upper bound:** algorithm adapted from the one for CC[≥ 1]

CRP with no restriction

Theorem [Delzanno et al.,FSTTCS'12]

For RAN, CRP is PSPACE-complete

Idea of the proof:

- **Lower bound:** reduction from marking reachability in 1-safe nets
- **Upper bound:** reachability in symbolic graph of exponential size

Outline

Conclusion

Complexity result for REACH in parameterized networks

Communication	Complexity
Rendez-vous	PTIME
Controlled Rendez-vous	EXPSPACE-complete
Broadcast	Ackermann-complete
Ad Hoc	Undecidable
Reconfigurable Ad Hoc	PTIME

- For Ad Hoc Networks, decidability can be regained by restricting the class of topologies
- For Reconfigurable Ad Hoc Networks, verifying absence of labels leads to PSPACE-complete algorithms

Last remarks

Many many papers on this subject

- See the survey [Esparza, STACS'14]
- Aminof et al. studied model-checking with branching time logic
- Esparza & Ganty studied communication through shared variables with no locking mechanism
- Bollig et al. studied expressivity of parameterized networks
- Bertrand et al. studied Broadcast Networks and Ad Hoc Networks with probability

And now ?

- How can this knowledge be used to verify real distributed algorithms ?
- Often you need identity (from an infinite alphabet)
- You might have message passing systems with queues
- Or parameterized shared memory (an array whose size depends on the number of processes)