Parameterized Verification Distributed Broadcast Protocols

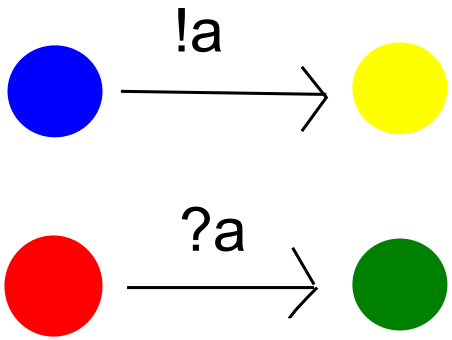
Giorgio Delzanno
Joint work with A. Sangnier, G. Zavattaro, R. Traverso, P. Abdulla, O. Rezine

DIBRIS, Università di Genova, Italy

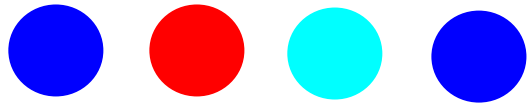
PV, Madrid, September 2015

Background

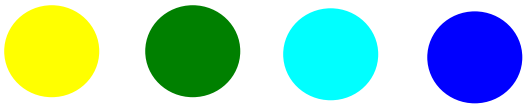
Petri Nets: Rendez-vous Synchronization



Rendez-vous

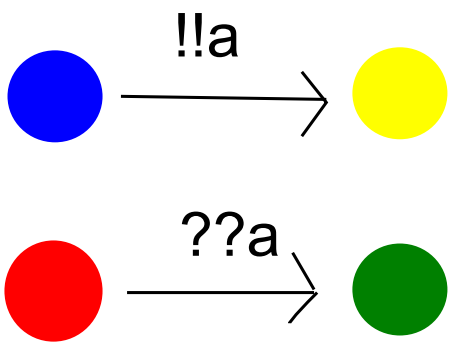


current marking

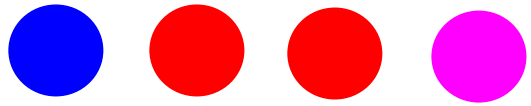


new marking

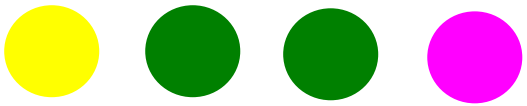
Broadcast Protocols: Transfer Arcs



Broadcast



current marking

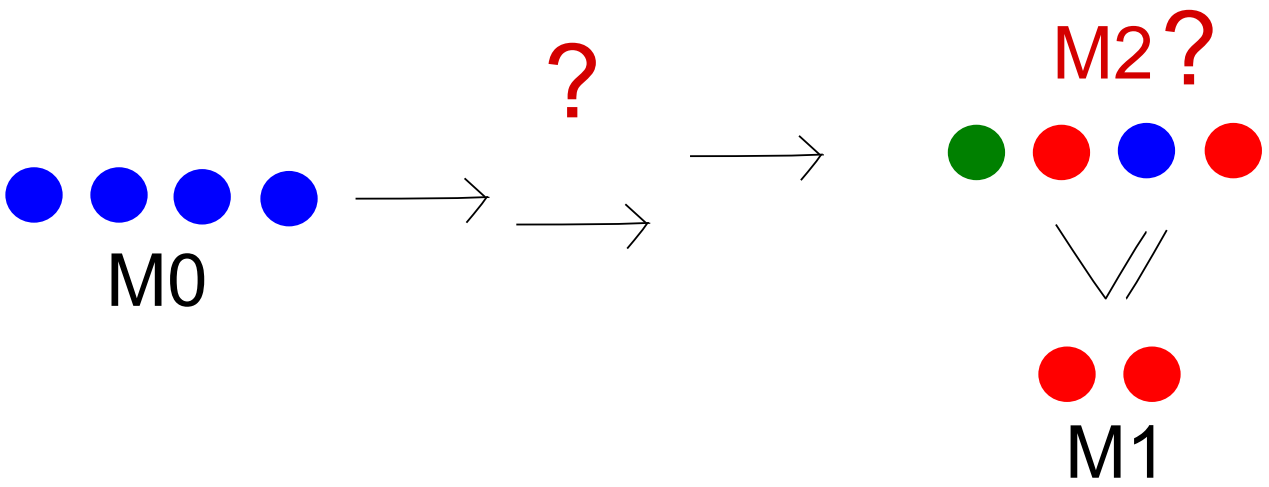


new marking

Coverability

Let $M_1 \leq M_2$ if, for any color c, the number of c-tokens in M_1 is less than that in M_2

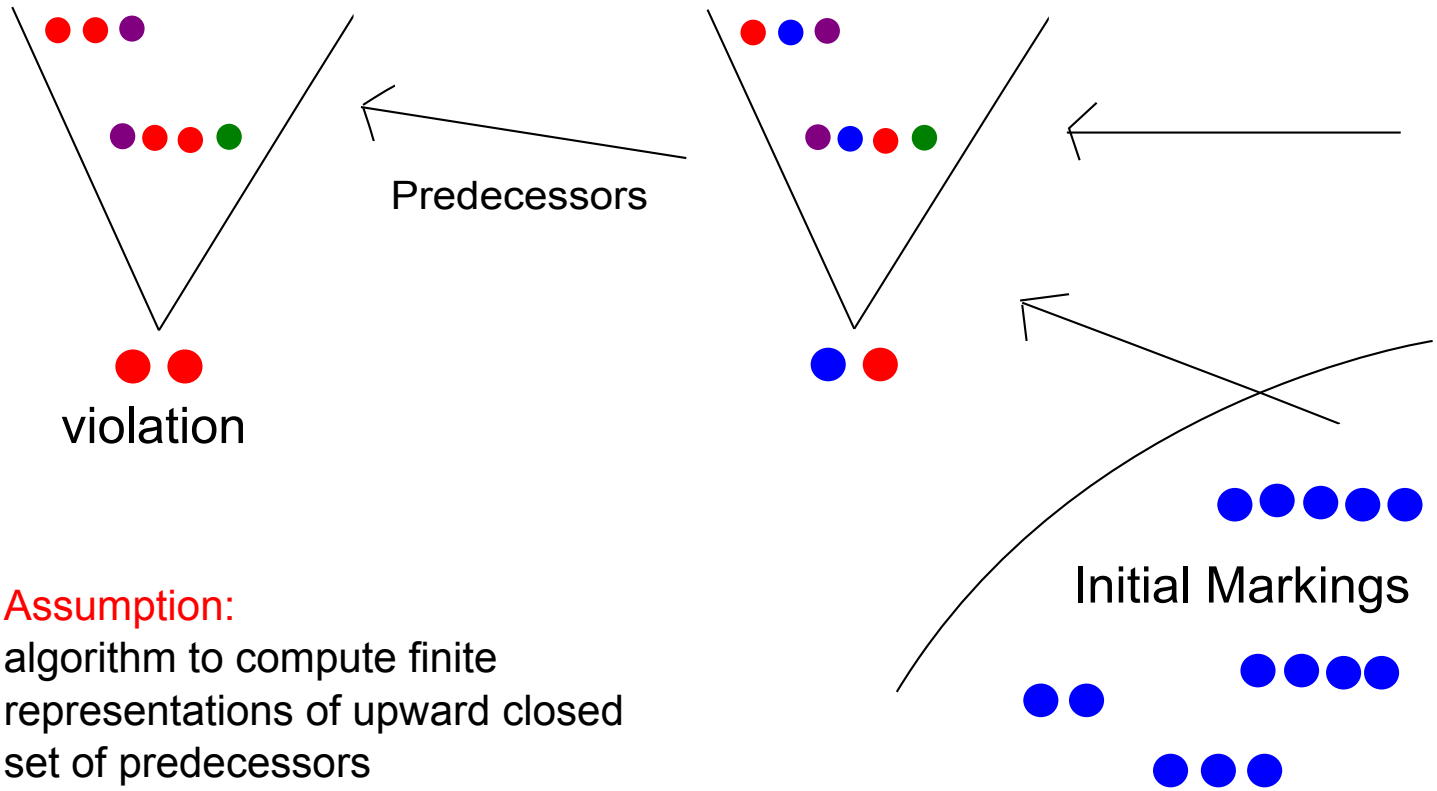
- Input: Markings M_0 and M_1
- Problem: Is there M_2 s.t. $M_1 \leq M_2$ and there is a computation from M_0 to M_2?



Known Properties

- Reachability and Coverability are decidable for Petri Nets
  - Karp-Miller construction solves Coverability
- Coverability is decidable for Broadcast Protocols
  - Backward Reachability using upward closed sets of markings is guaranteed to terminate
  - Complexity is non-primitive recursive

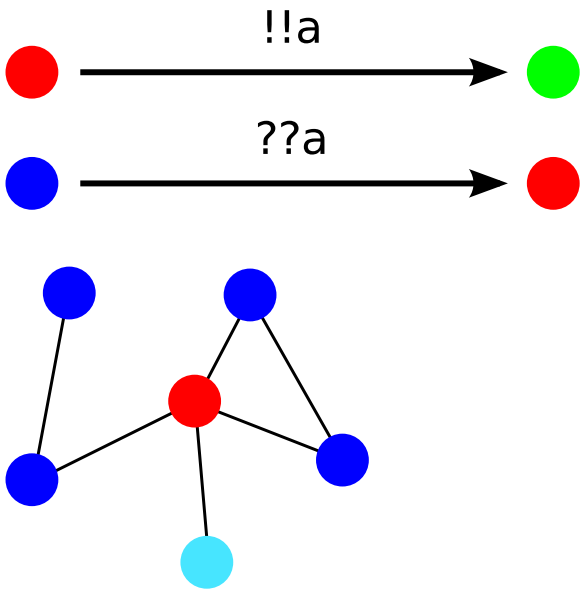
Coverability vs Parameterized Verification



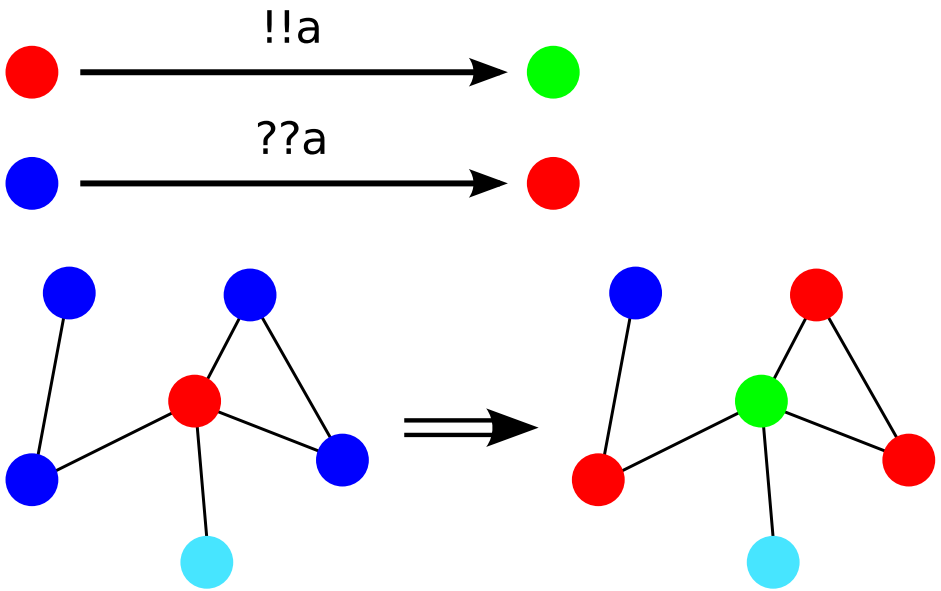
Assumption:
algorithm to compute finite representations of upward closed set of predecessors

What if nodes are distributed on a graph?

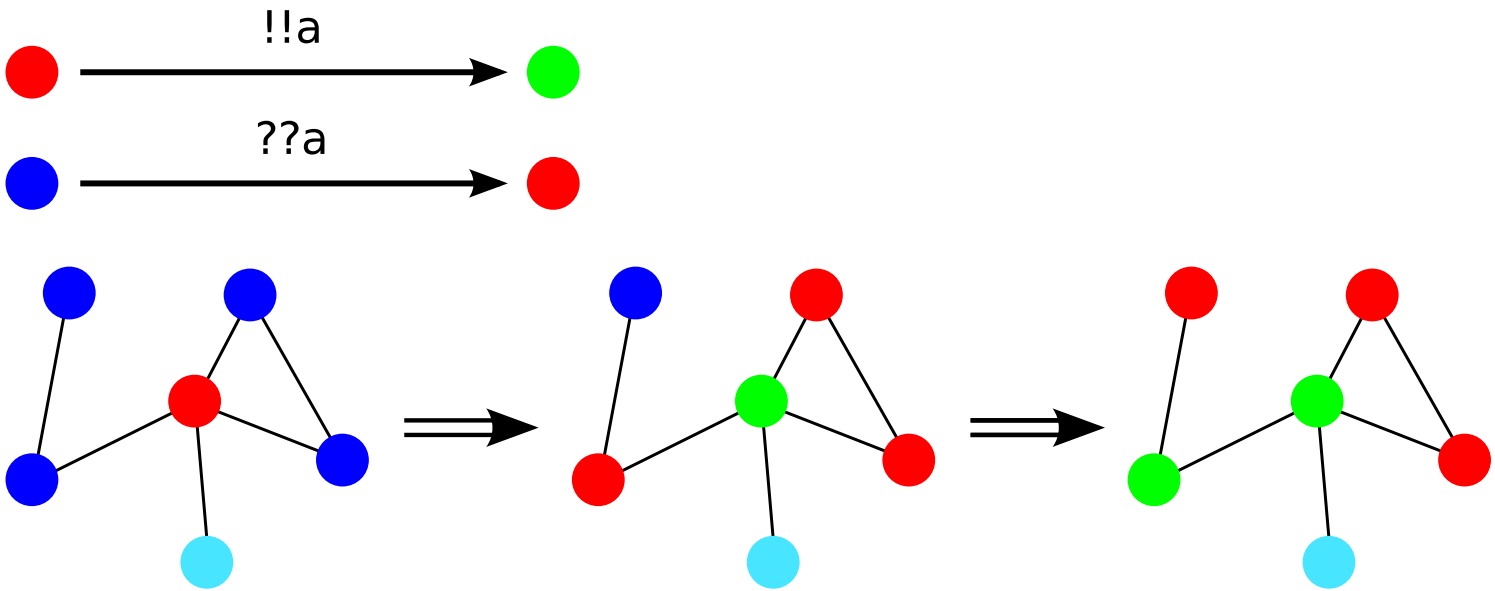
Broadcast Communication on a Graph



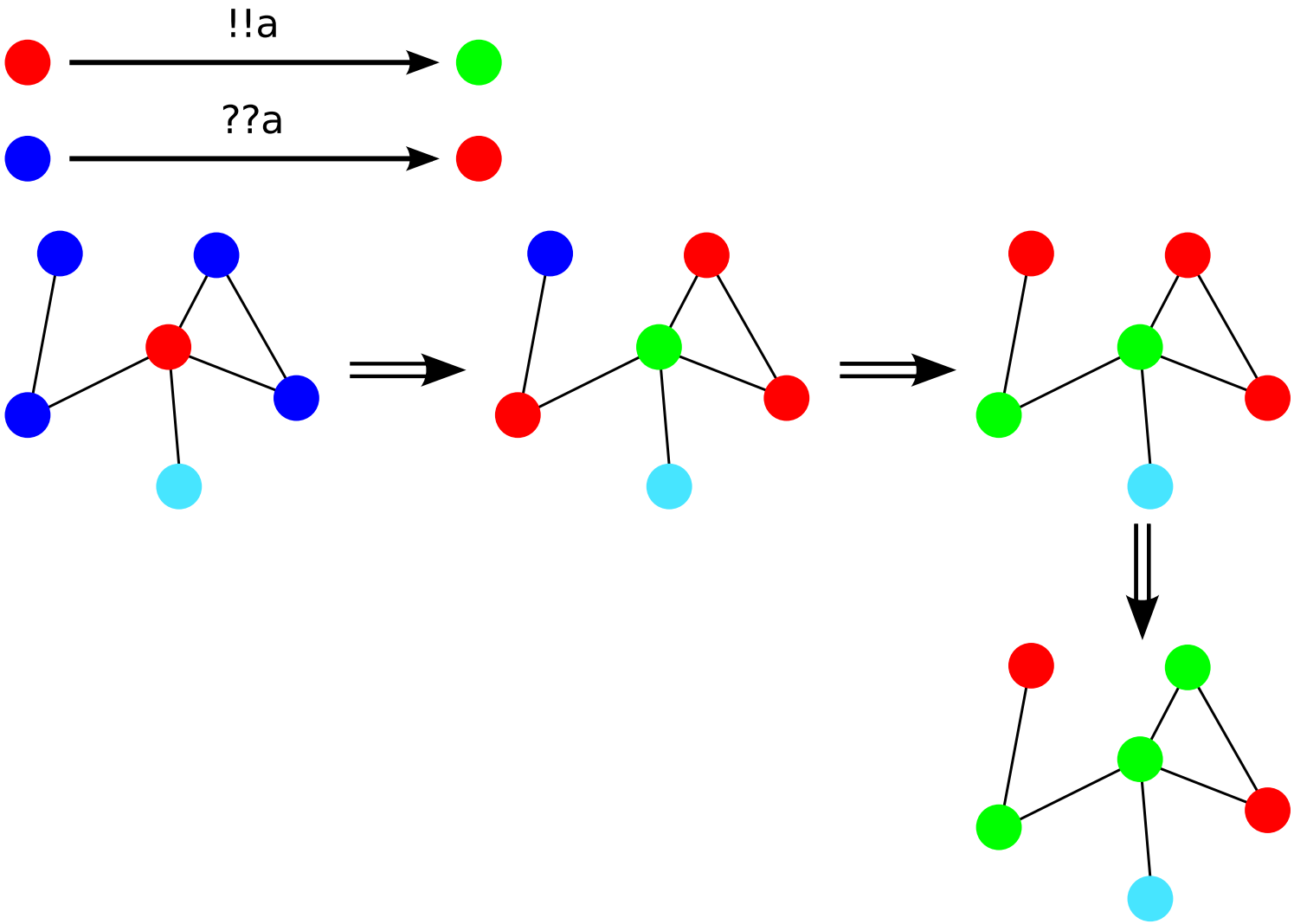
Broadcast Communication on a Graph



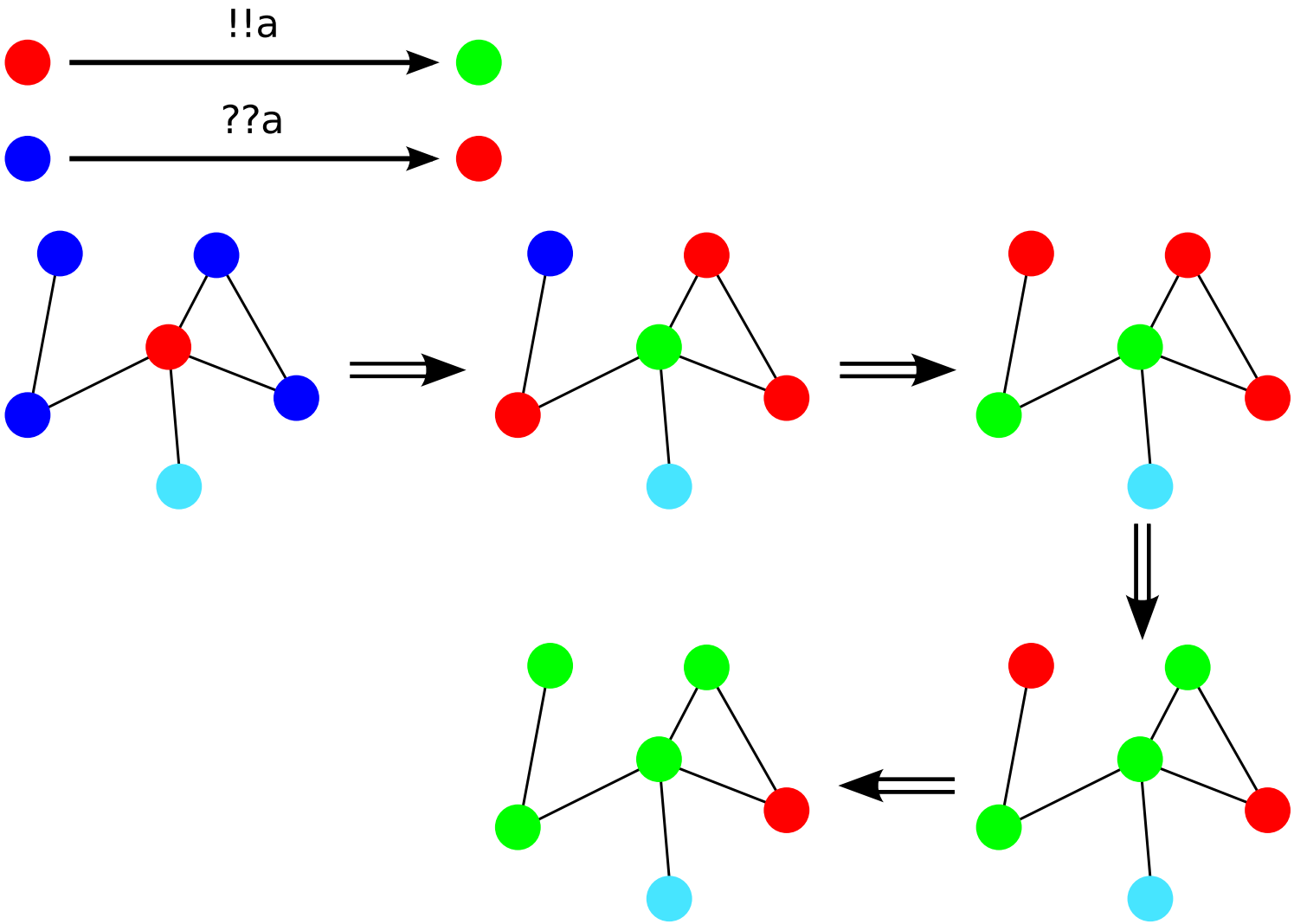
Broadcast Communication on a Graph



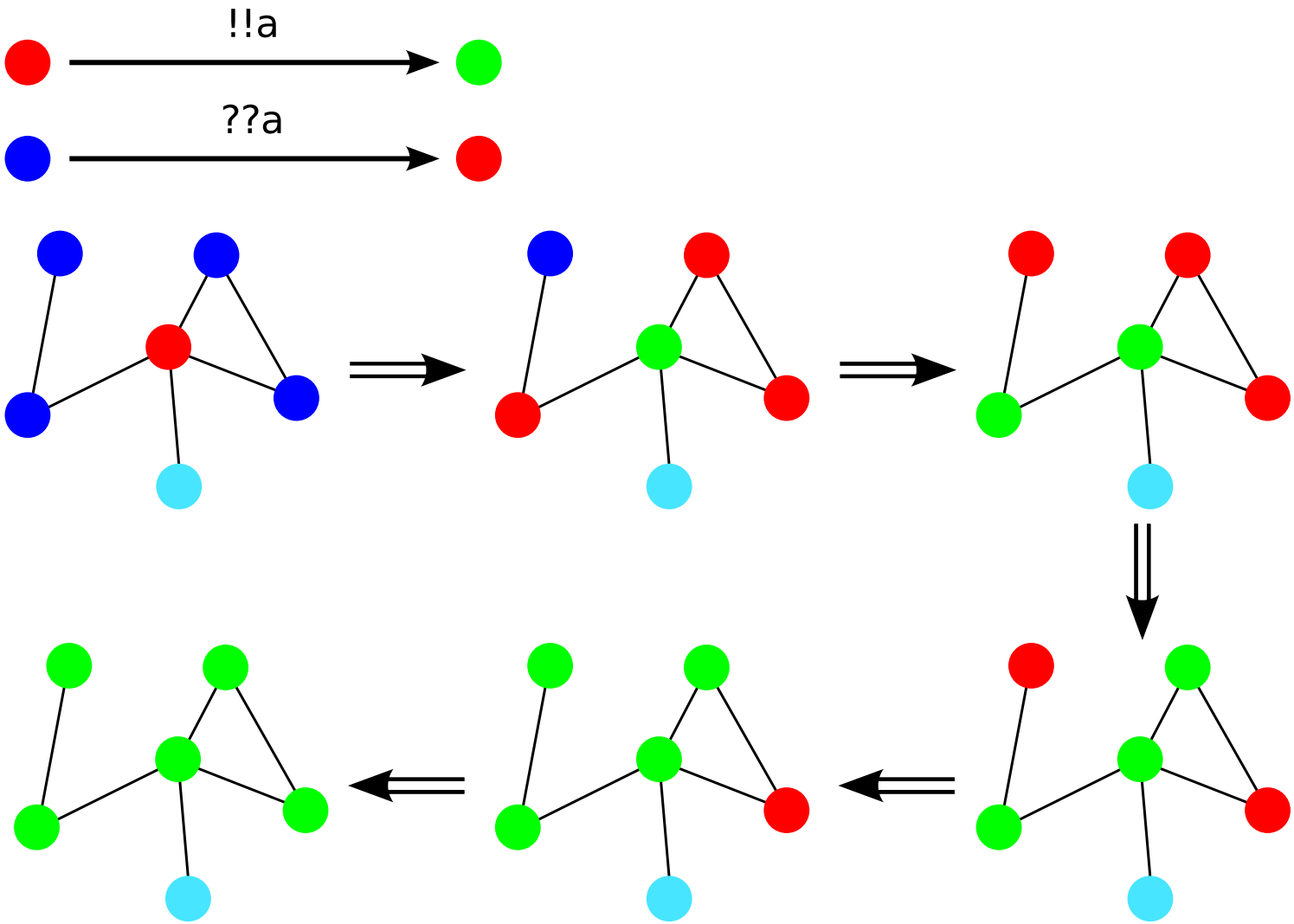
Broadcast Communication on a Graph



Broadcast Communication on a Graph



Broadcast Communication on a Graph



Basic Model for Ad Hoc Networks (AHN)

- An Ad Hoc Network (AHN)[1] is a network of communicating automata distributed over an undirected graph.
- Each node runs a predefined common protocol.
- Synchronous selective broadcast messages.

[1] Delzanno, Sangnier, Zavattaro: Parameterized verification of ad hoc networks. CONCUR'10.
Delzanno, Sangnier, Zavattaro: On the Power of Cliques in the Parameterized Verification of Ad Hoc Networks. FOSSACS'11.

Generalization of Coverability (COVER)

For:

- a (possibly infinite) class of graphs $\mathcal{G}$ (initial network configurations)

Given:

- an automaton/protocol $\mathcal{P}$
- a specification φ of target states (e.g. is there a red node?)

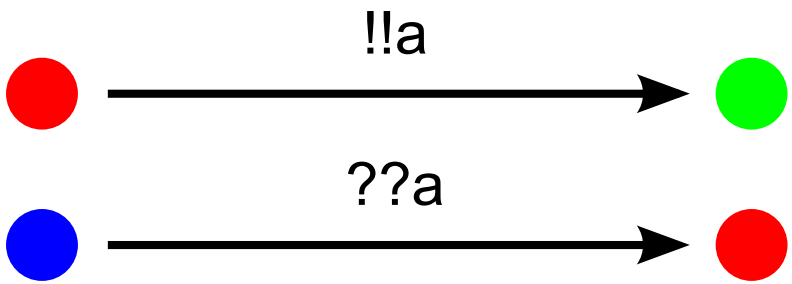
Output:

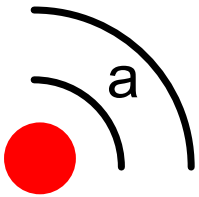
- True iff there is an initial configuration in $\mathcal{G}$ s.t., by executing $\mathcal{P}$ on each node, the network may reach a configuration satisfying φ

Expressive Power?

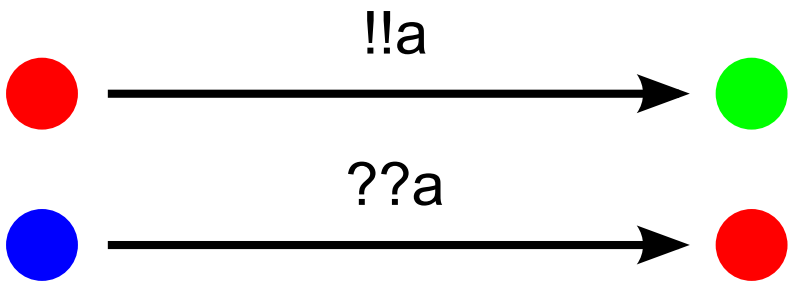
- Fixed initial configuration:
  - Finite-state systems that are computationally expensive to verify with existing model checkers like Spin, SMV, Uppaal, Groove.
- Unknown initial configuration, specification: existence of a "red" node
  - Verification becomes undecidable: discovery protocols can be used to build unbounded data structures, e.g., a list of arbitrary length.

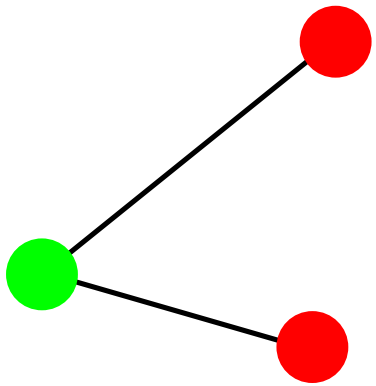
Discovery



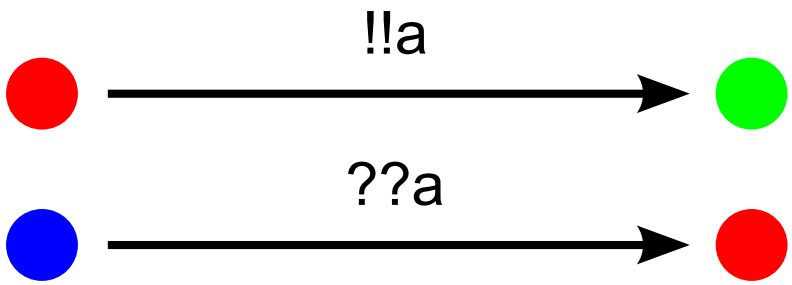


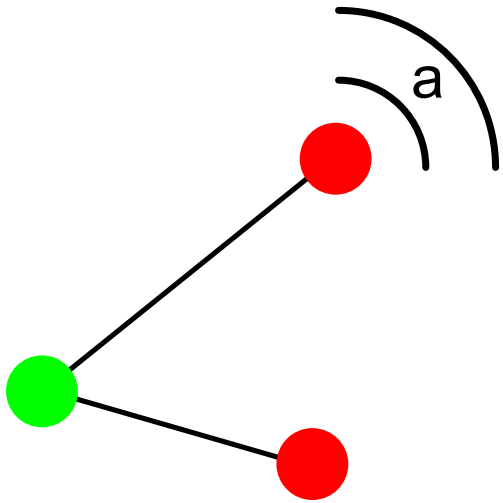
Discovery



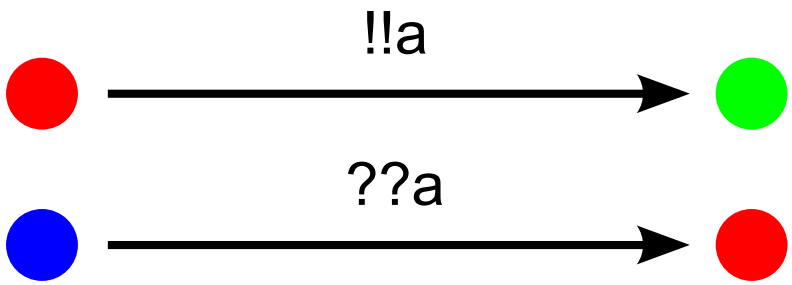


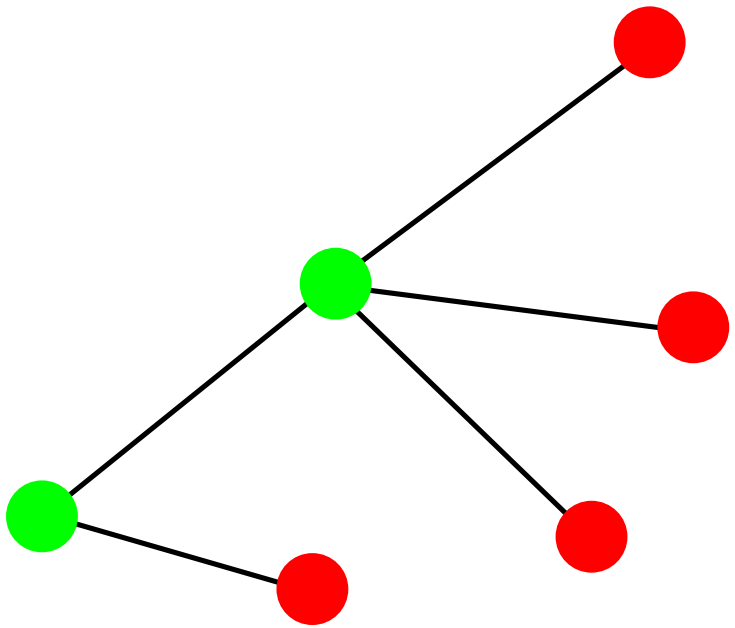
Discovery



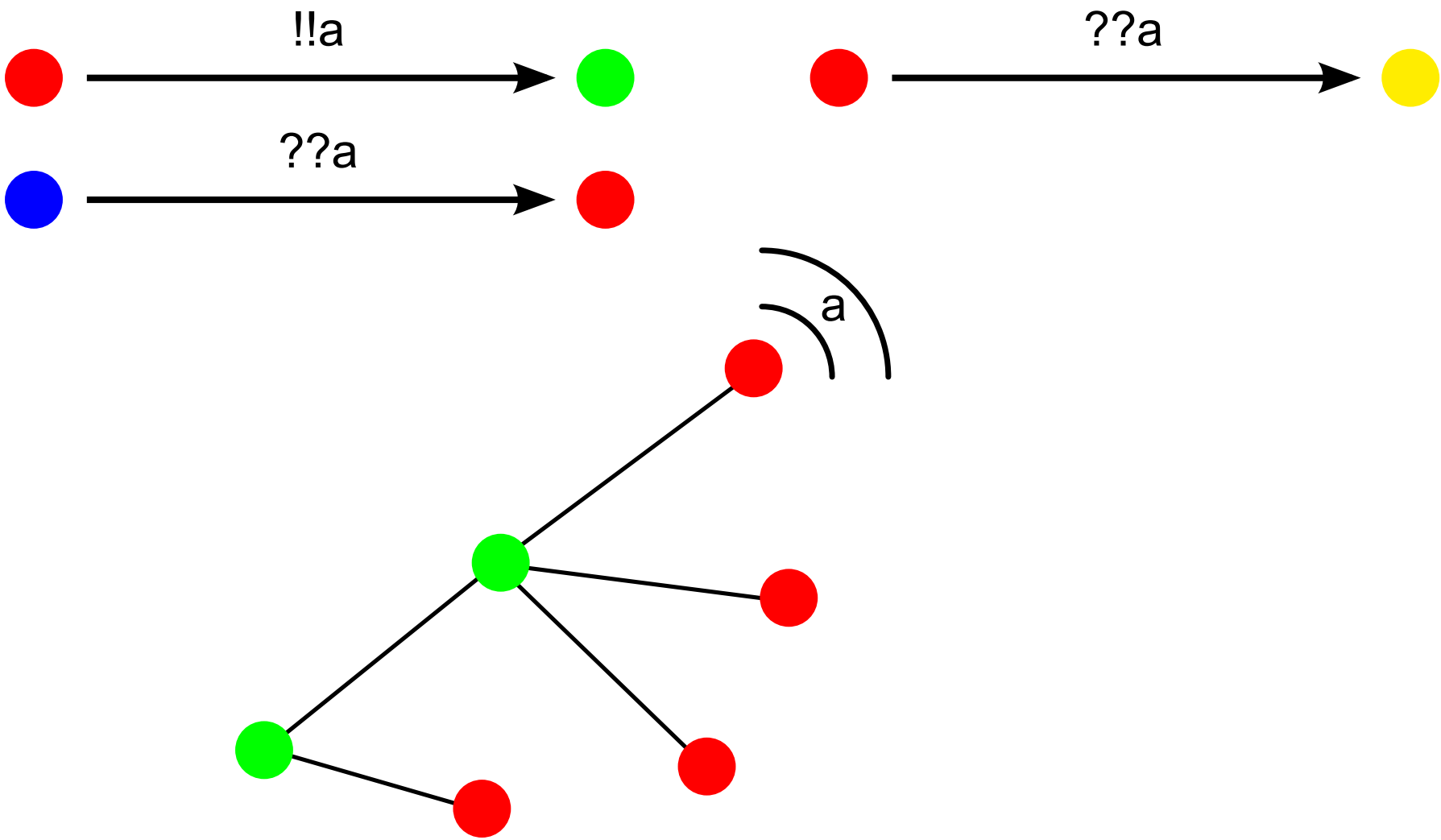


Discovery

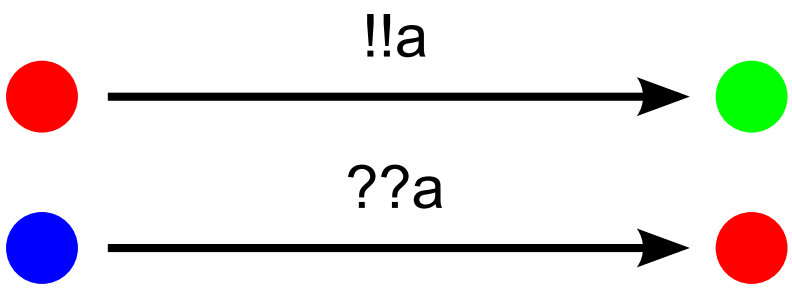


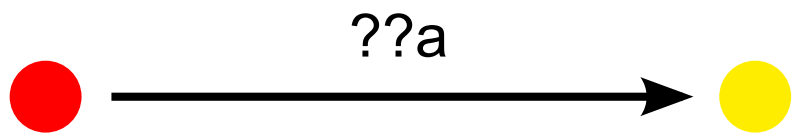


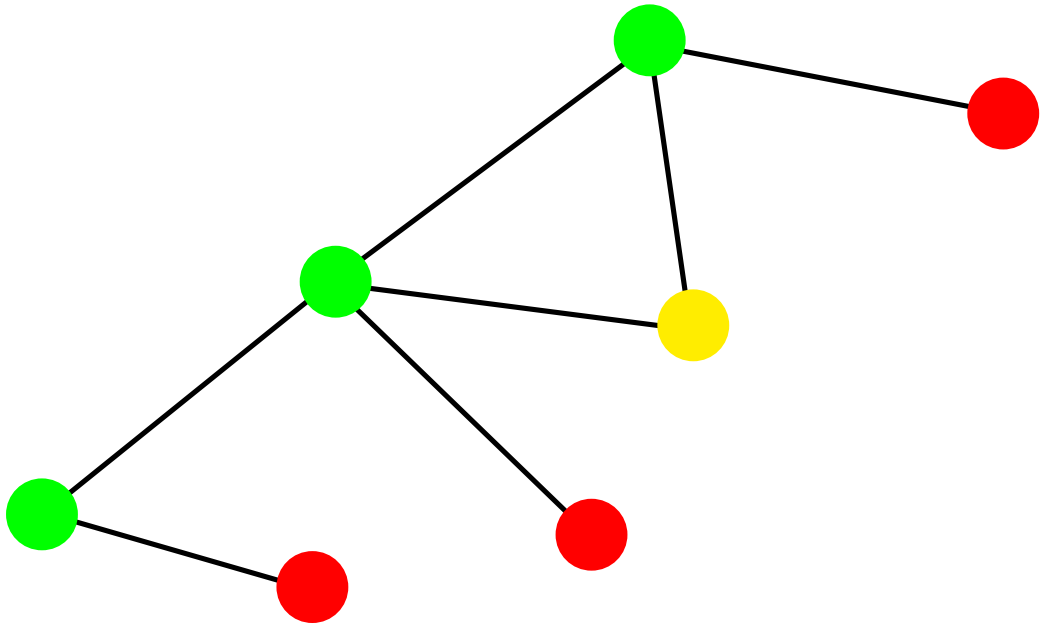
Discovery



Discovery







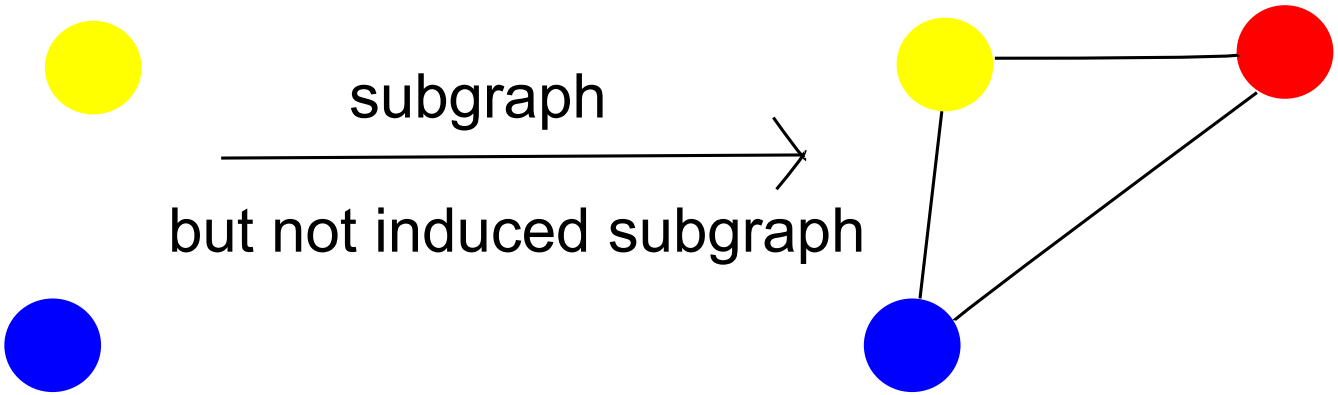
Decidability

- COVER becomes decidable by restricting configurations in specific classes of graphs.
  - Fully connected topologies
  - K-bounded path graphs, i.e., graphs in which all simple paths have length $\leq K$ (fully connect graphs $\neq$ not bounded path graphs)
  - Graphs in which when collapsing maximal cliques we still have K-bounded paths only

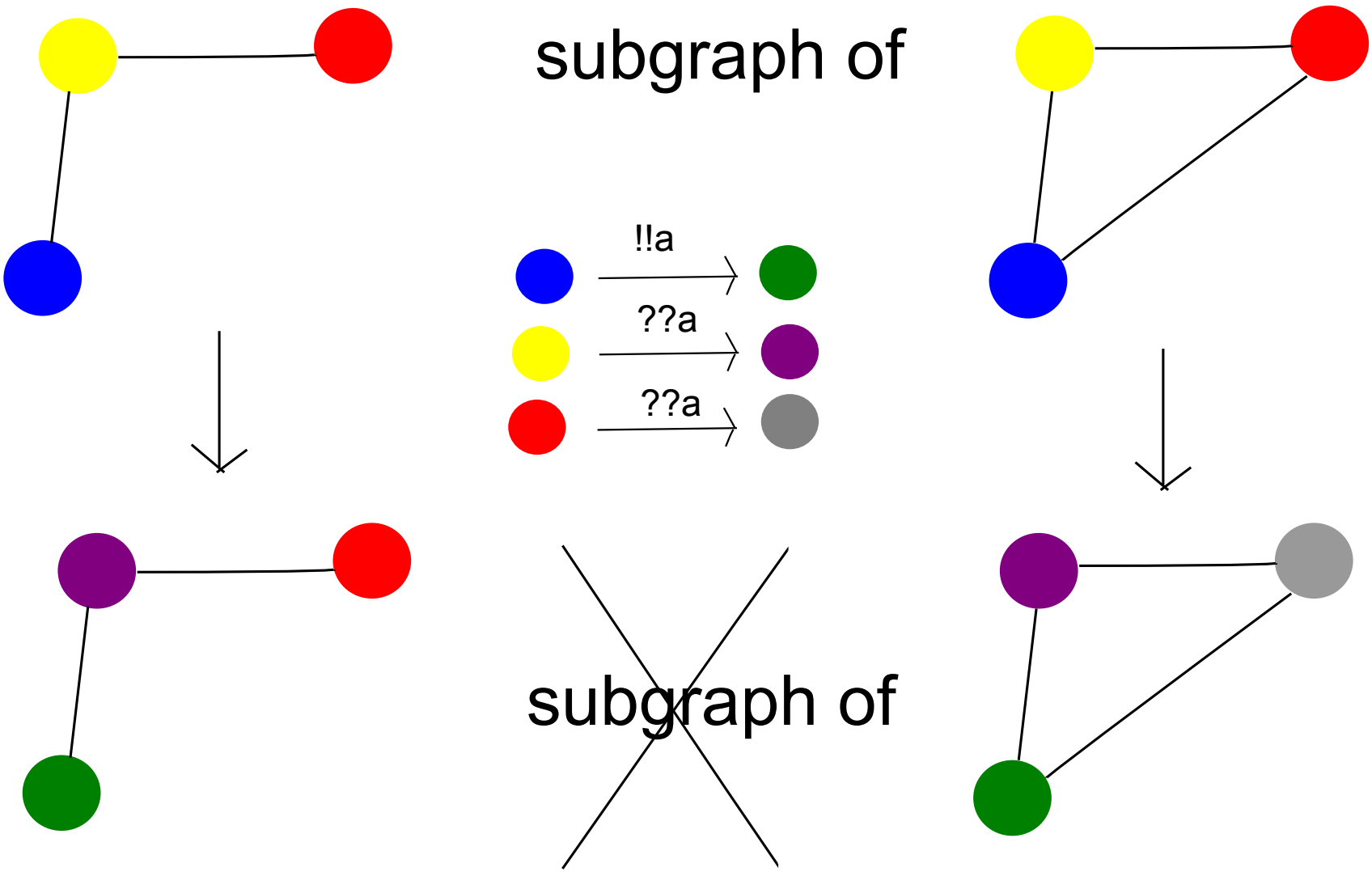
Bounded Path Graphs: WSTS as proof technique

- Transitions are monotone w.r.t. the induced subgraph ordering
- Induced subgraph ordering over K-bounded path graph is a well quasi ordering [Ding '90]
- We can effectively compute predecessors of upward closed sets of graphs (for graphs of size n we need to consider at most graphs of size $n + 1$)

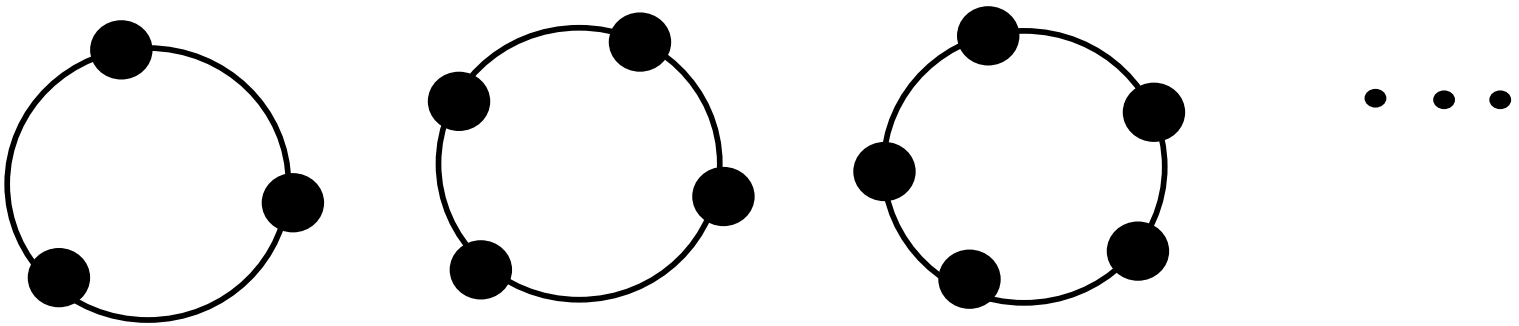
Remark 1: Induced vs Subgraph



Remark 2: Subgraphs is not OK



Remark 3: Induced is not a wqo for arbitrary graphs



infinite sequence of incomparable graphs

Possible Extensions

- . . . no fixed infrastructure, dynamically changing: reconfigurations
- . . . asynchronous communication
- . . . value passing
- . . . failures of nodes and/or communication channels
- . . . time constraints
- . . . permission and access control
- . . .

AHN + Reconfigurations

- Reconfiguration = the topology of the network can change non-deterministically during an execution (node mobility, link failures/intermittence) [2]

[2] Delzanno, Sangnier, Traverso, Zavattaro. On the Complexity of Parameterized Reachability in Reconfigurable Broadcast Networks. FSTTCS'12.

AHN + Reconf.: COVER

As specification language, we consider cardinality constraints (CC) on the number of processes in a given control state:

$$\varphi ::= a \leq \#q < b \mid \varphi \wedge \varphi \mid \varphi \vee \varphi \mid \neg\varphi$$

where $a \in \mathbb{N}$, q is a local control state, and $b \in (\mathbb{N} \setminus \{0\}) \cup \{+\infty\}$.

- COVER with no restrictions on the initial number of processes.

AHN + Reconf.: Results

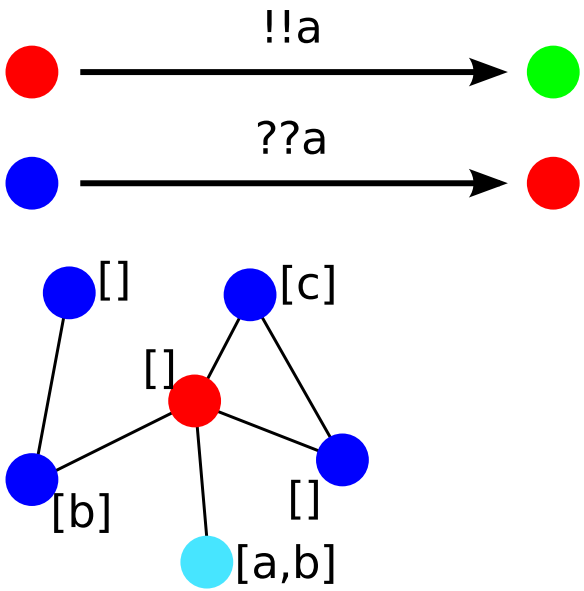
- COVER is PTIME-complete for CC without negation and with only $\#q \geq 1$ atoms
  - Keypoint: if two processes can make a transition, any number of processes can do the same transition
  - No need to count: Saturation procedures that collects the **set** of labels that can be produced by applying transitions
- For CC with $\#q \geq 1$ atoms and negation COVER is NP-complete.
- COVER is PSPACE-complete for unrestricted CC.

AHN + Asynchronous Communication

- A variant of AHNs with asynchronous broadcast communication.[3]
- Unread messages are kept in local mailboxes.
- We consider different disciplines for handling mailboxes:
  - **multisets**, to model the loss of the order of incoming messages;
  - **lossy FIFO queues**, to model the loss of messages;
  - **FIFO queues**, to model perfect communication.

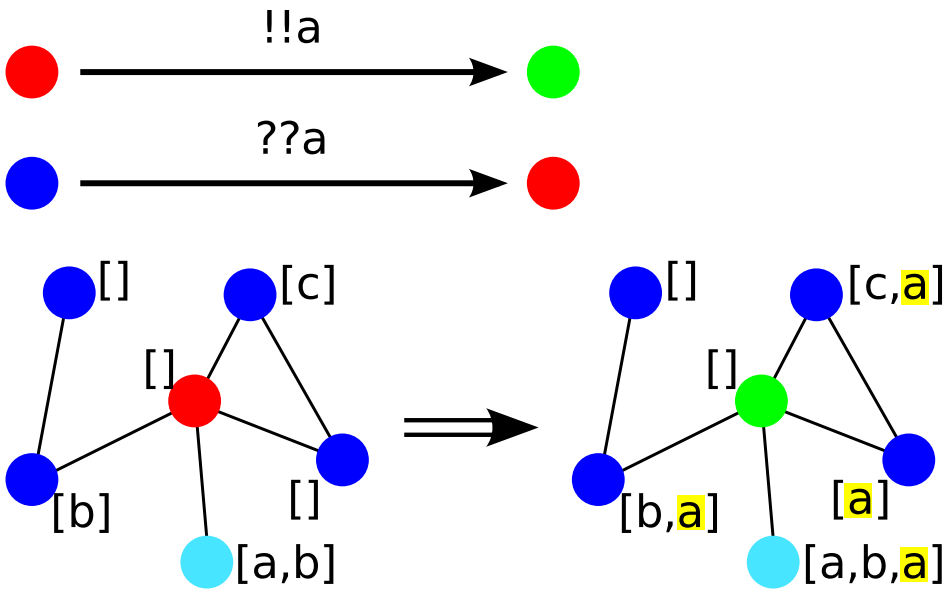
[3] Delzanno, Traverso: A Formal Model of Asynchronous Broadcast Communication. ICTCS'12.
Delzanno, Traverso: Decidability and Complexity Results for Verification of Asynchronous Broadcast Networks. LATA'13.

AHN + Asynch.: Example #1



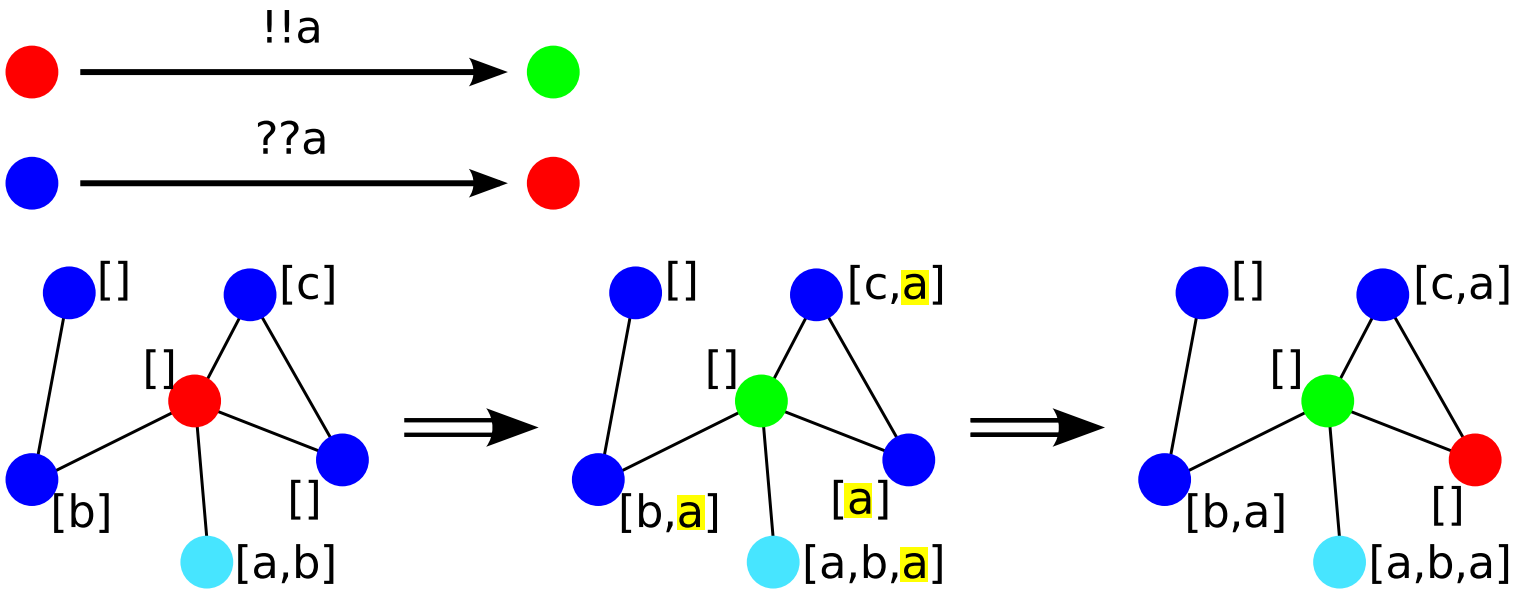
(with multisets as mailboxes)

AHN + Asynch.: Example #1



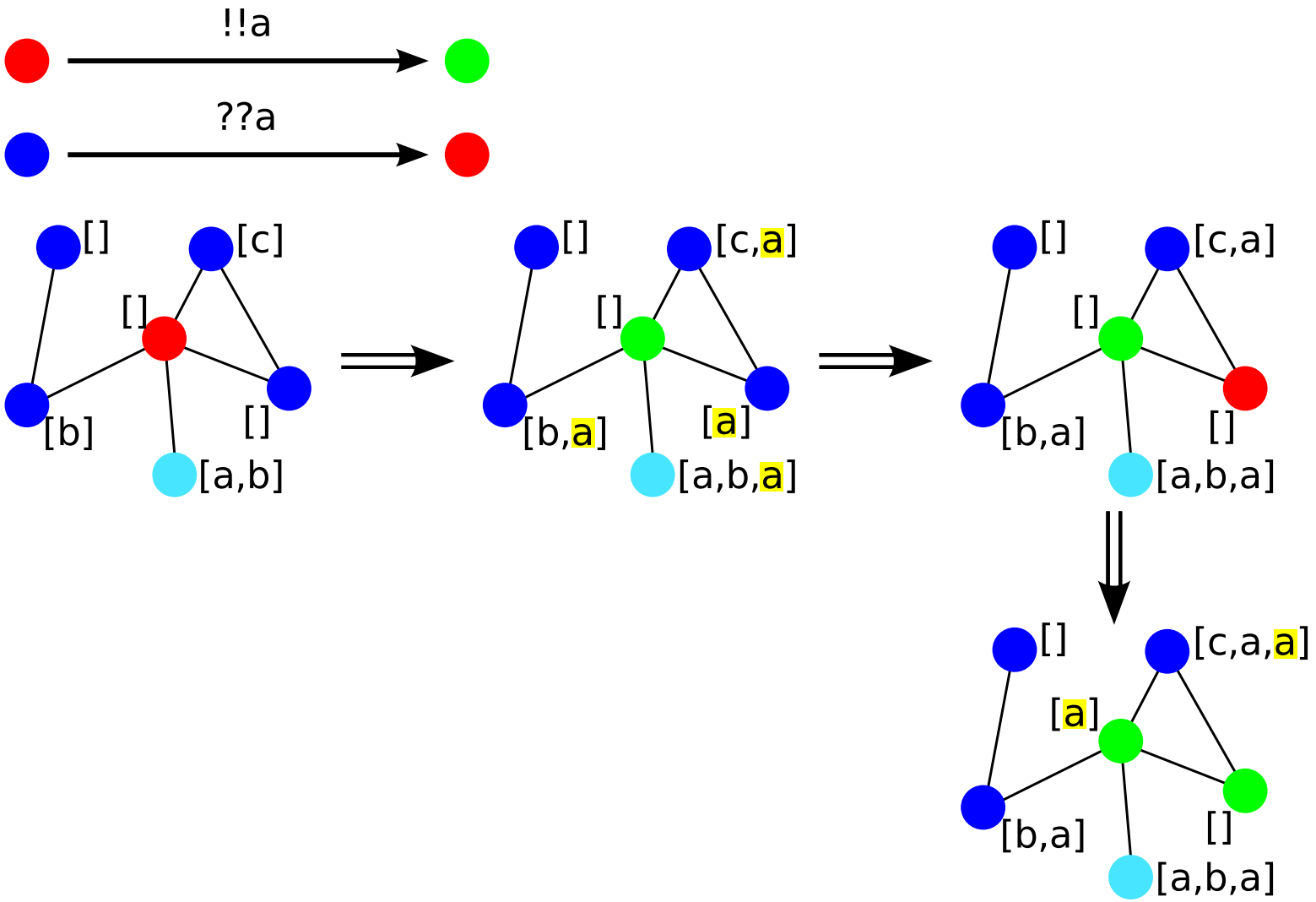
(with multisets as mailboxes)

AHN + Asynch.: Example #1



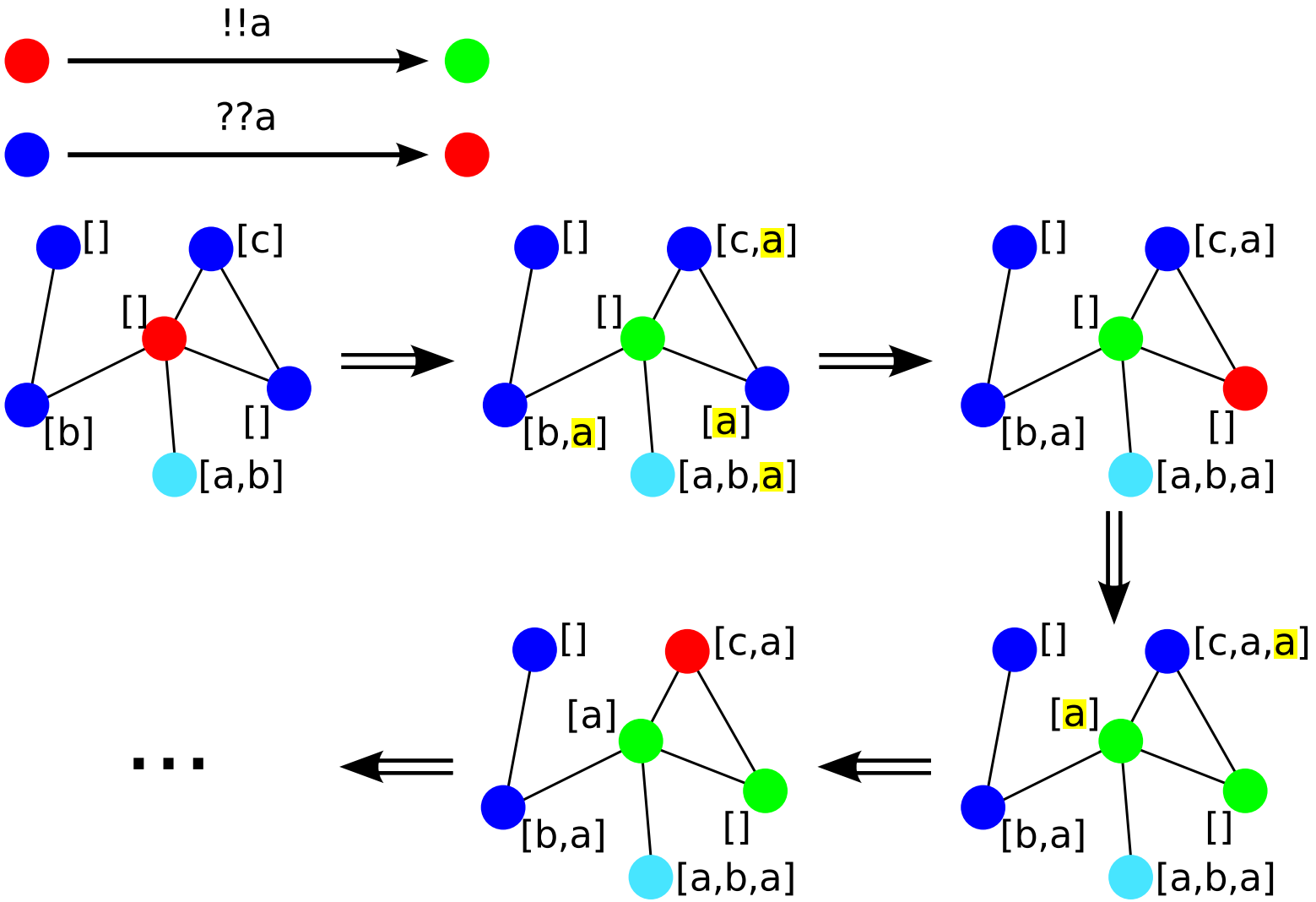
(with multisets as mailboxes)

AHN + Asynch.: Example #1



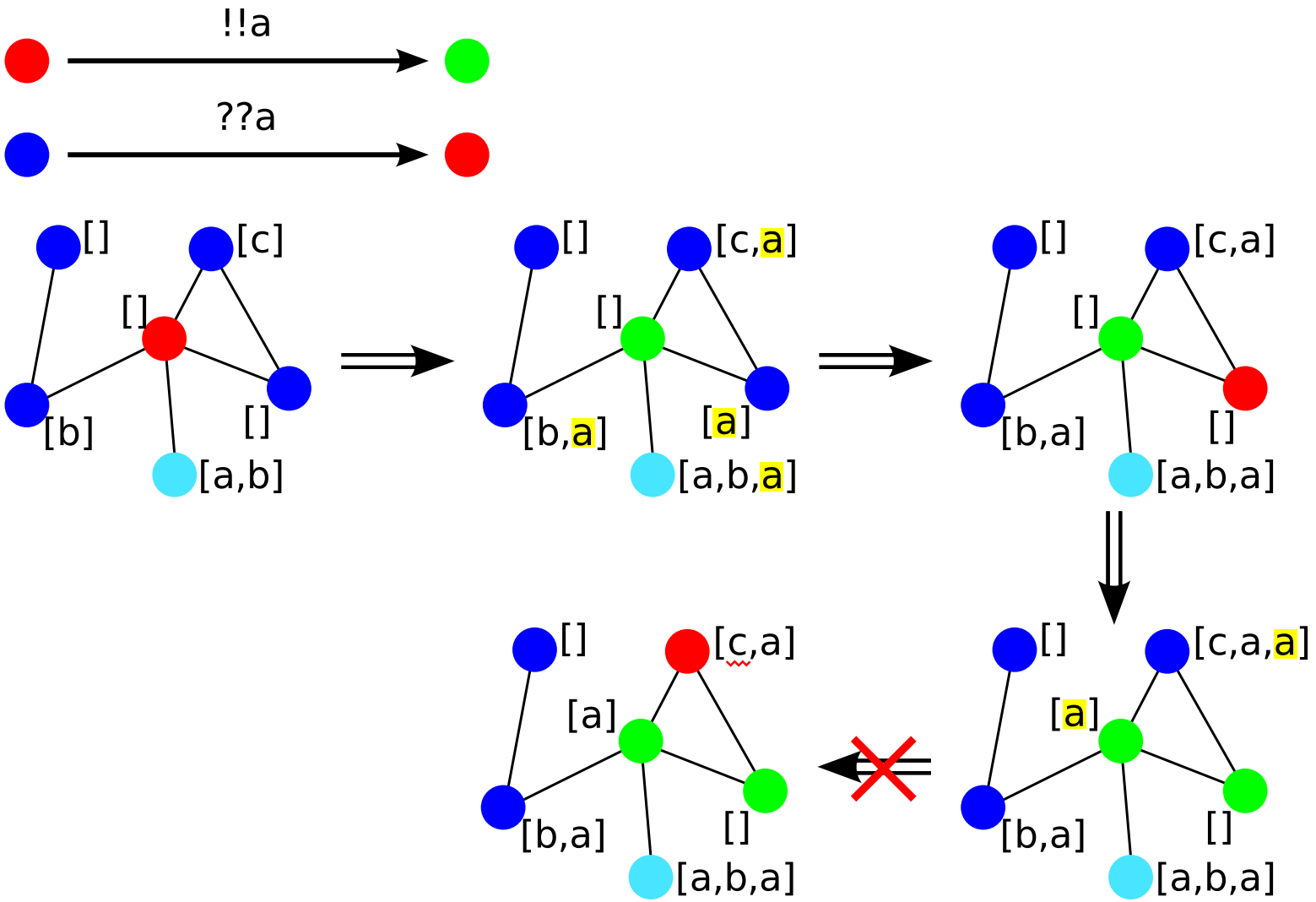
(with multisets as mailboxes)

AHN + Asynch.: Example #1



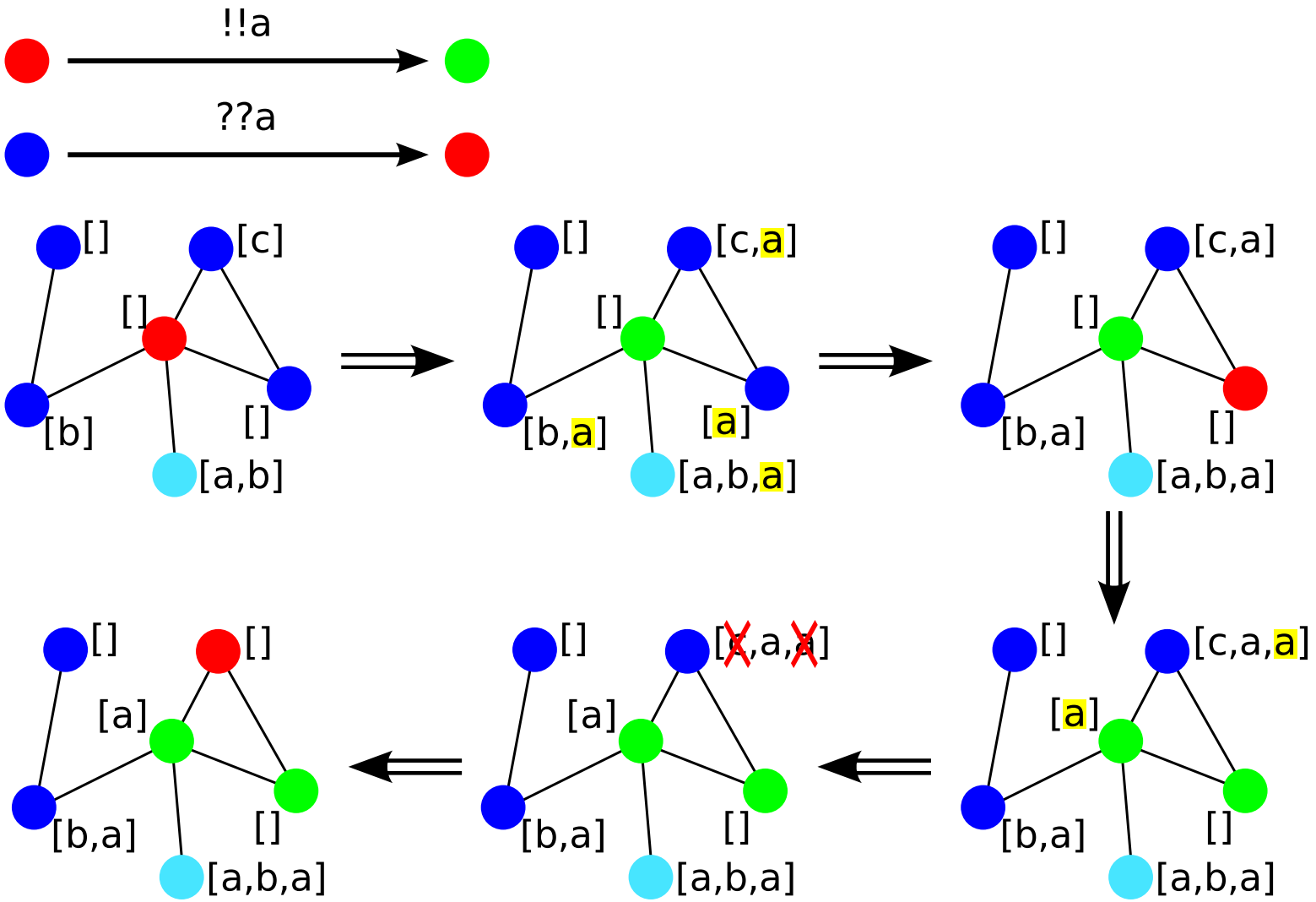
(with multisets as mailboxes)

AHN + Asynch.: Example #2



(with FIFO queues as mailboxes)

AHN + Asynch.: Example #3



(with lossy FIFO queues as mailboxes)

AHN + Asynch.: Goal

To decidability boundaries for COVER we must consider

- the **policy** to handle local mailboxes;
- the **shape** of connection graph;
- the capability to **recognize empty mailboxes** (ϵ-transitions).

AHN + Asynch.: Undecidability with Emptiness Test

COVER for multisets and emptiness test is undecidable

Reduction of halting problem for two-counter machines.
Proof idea: The emptiness test can be exploited both for zero-testing and interference detection.

AHN + Asynch.: Undecidability with Emptiness Test

Reduction of halting problem for two-counter machines in which the emptiness test can be exploited both for zero-testing and interference detection

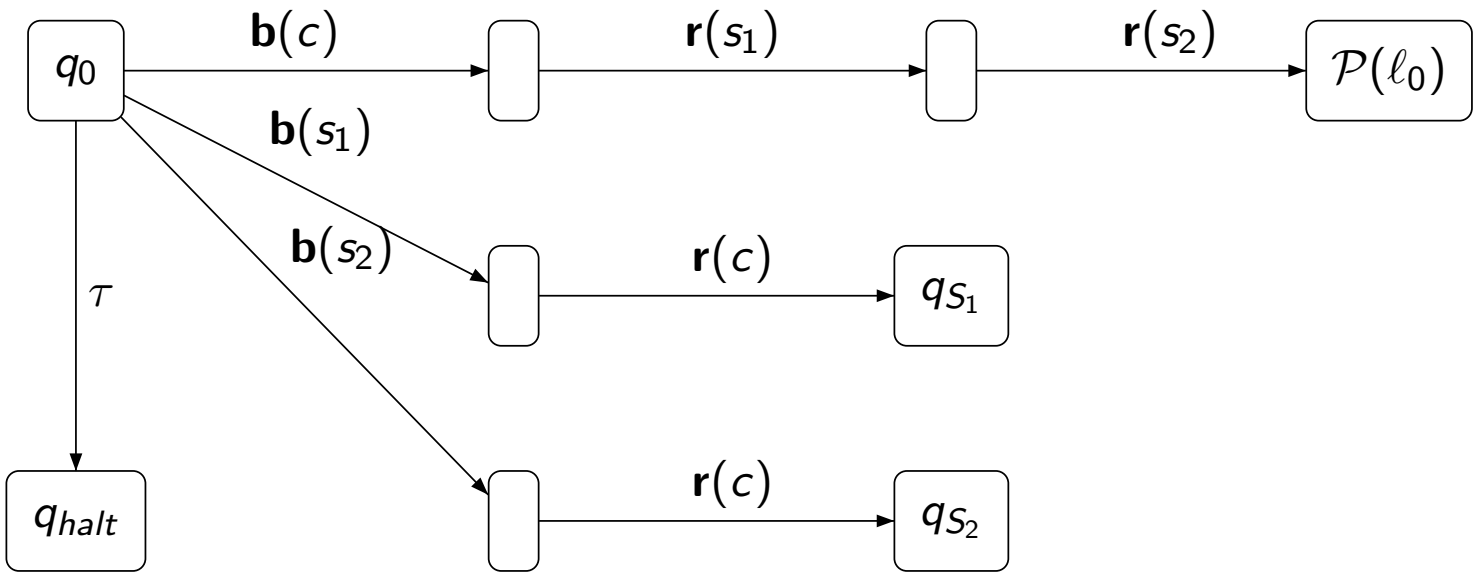
- The encoded protocol is split in two phases: **election** and **simulation**.
- During the election processes choose their role.
- The simulation requires a leader process directly connected to two slaves (one per counter).

AHN + Asynch.: Undecidability with Emptiness Test

Election

- Each node chooses a role and searches for appropriate neighbours accordingly.
- Election only tests for the presence of the required links between nodes with the various roles.
- Messages exchanged during the election can never be consumed afterwards.
- A successful election ends leaving empty mailboxes in the involved nodes.

AHN + Asynch.: Undecidability with Emptiness Test

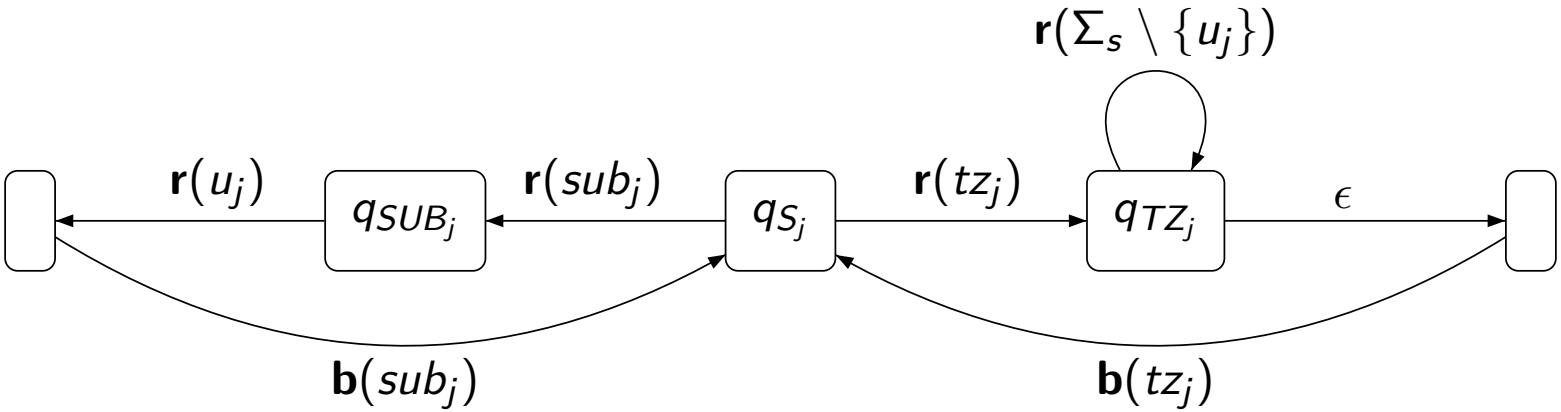


AHN + Asynch.: Undecidability with Emptiness Test

Simulation

- Counters are encoded (in unary) through messages in the mailboxes.
  - For increment it is sufficient to send a broadcast with a unit.
  - A decrement forces the removal of a unit from the mailbox of the slave.
  - Tests for zero are performed by exploiting tests for emptiness.

AHN + Asynch.: Undecidability with Emptiness Test



AHN + Asynch.: Undecidability with Emptiness Test

Warning

What if other neighbours wake up and start a simulation leading to interferences with the current one?

AHN + Asynch.: Undecidability with Emptiness Test

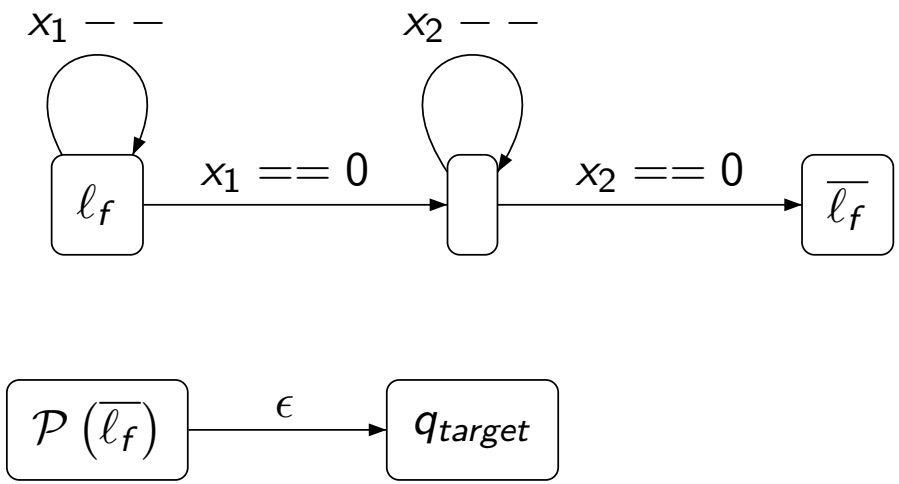
Warning

What if other neighbours wake up and start a simulation leading to interferences with the current one?

Finalization

- Reminder: we cannot consume messages from the election during the simulation.
- After reaching the target control state, we reset both counters to zero in order to try to empty all mailboxes.
- If all mailboxes are empty the simulation ends successfully, otherwise it blocks just before completing.

AHN + Asynch.: Undecidability with Emptiness Test



AHN + Asynch.: Results

	$COVER^{\mathcal{K}}(\mathbb{M})$		$COVER(\mathbb{M})$	
	without ϵ	with ϵ	without ϵ	with ϵ
LFIFO	PTIME	PTIME	PTIME	PTIME
Multiset	PTIME	undec.	PTIME	undec.
FIFO	undec.	undec.	undec.	undec.

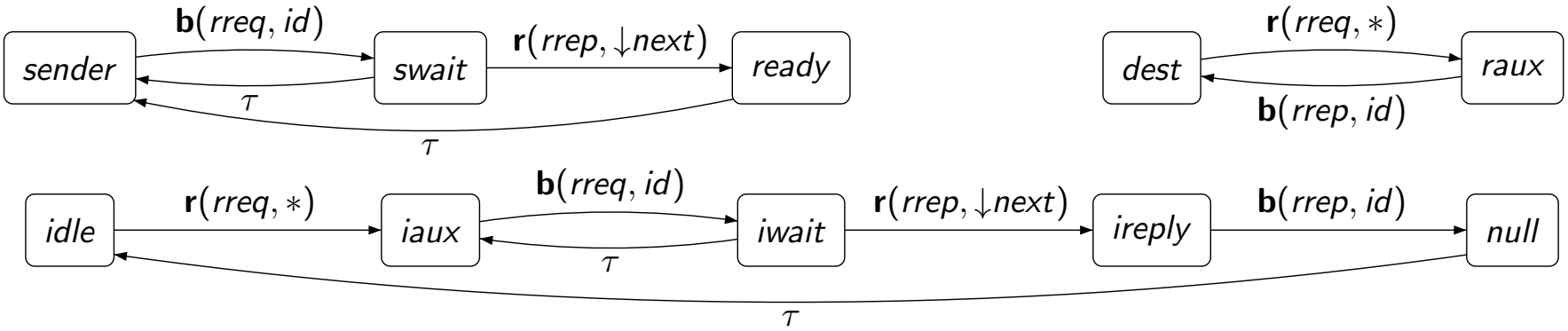
		Asynch. without ϵ / with ϵ		
	AHN	LFIFO	Multiset	FIFO
Fully connected graphs	✓	PTIME	PTIME/undec.	undec.
Arbitrary graphs	undec.	PTIME	PTIME/undec.	undec.

AHN + Data + Reconfigurations

- Broadcast Networks of Register Automata[4].
- Each process has local registers:
  - with values ranging over $\mathbb{N}$;
  - initialized with fresh values w.r.t. the whole network;
  - read-only ones act as process identifiers.
- Messages have some payload fields to exchange data.
- Upon reception of a message with a payload, processes may:
  - test (in)equality w.r.t. local registers;
  - store values in local registers;
  - ignore (a part of) the payload.
- Dynamic network reconfigurations (w.r.t. edges).

[4] Delzanno, Sangnier, Traverso. Parameterized Verification of Broadcast Networks of Register Automata. RP'13.

AHN + Data + Reconf.: Example



- *sender* nodes want to keep a route towards some *dest* node.
- Two registers: *id* and *next*.
- Initial states: *sender*, *idle*, *dest*.

AHN + Data + Reconf.: COVER

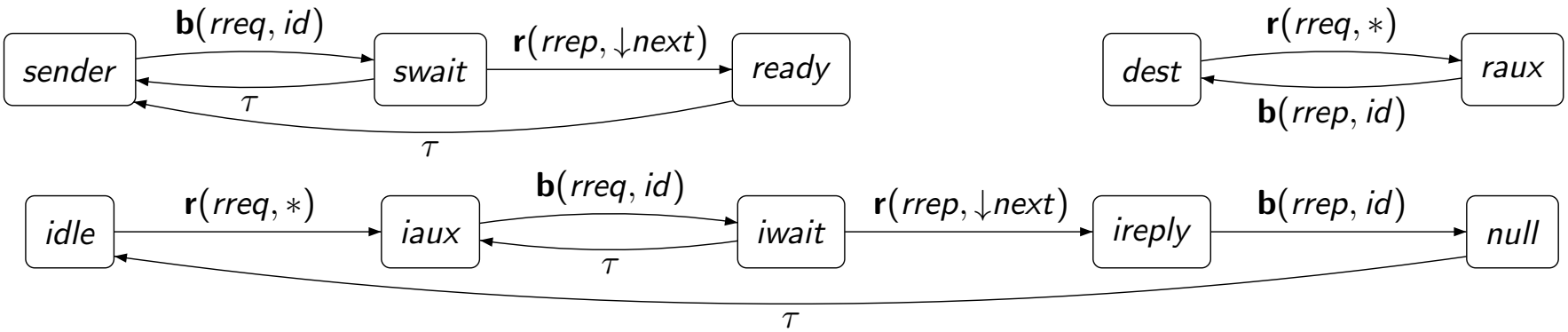
We consider queries expressing graph inclusion patterns:

$$\varphi ::= q(z) \mid M_i(z) = M_j(z') \mid M_i(z) \neq M_j(z') \mid \varphi \wedge \varphi$$

where z, z' are from a denumerable set of variables, q is a local control state, and $i, j \in \mathbb{N}$ are register indexes.

- No restrictions on the initial number of processes or network topology.

AHN + Data + Reconf.: Example Query



$$ready(z_1) \wedge ready(z_2) \wedge M_{id}(z_1) = M_{next}(z_2) \wedge M_{next}(z_1) = M_{id}(z_2)$$

AHN + Data + Reconf.: Goal

Explore decidability boundaries for COVER:

- restrictions on the **number of registers** in each node;
- restrictions on the **number of fields** in each message;
- **static topology** vs **dynamic reconfigurations**.

AHN + Data + Reconf.: Results

Without network reconfigurations (fully connected case):

- 1 RW registers, 1 payload field $\Longrightarrow$ non-elementary
- 1 RO + 1 RW registers, 1 payload field $\Longrightarrow$ undecidable

With reconfigurations:

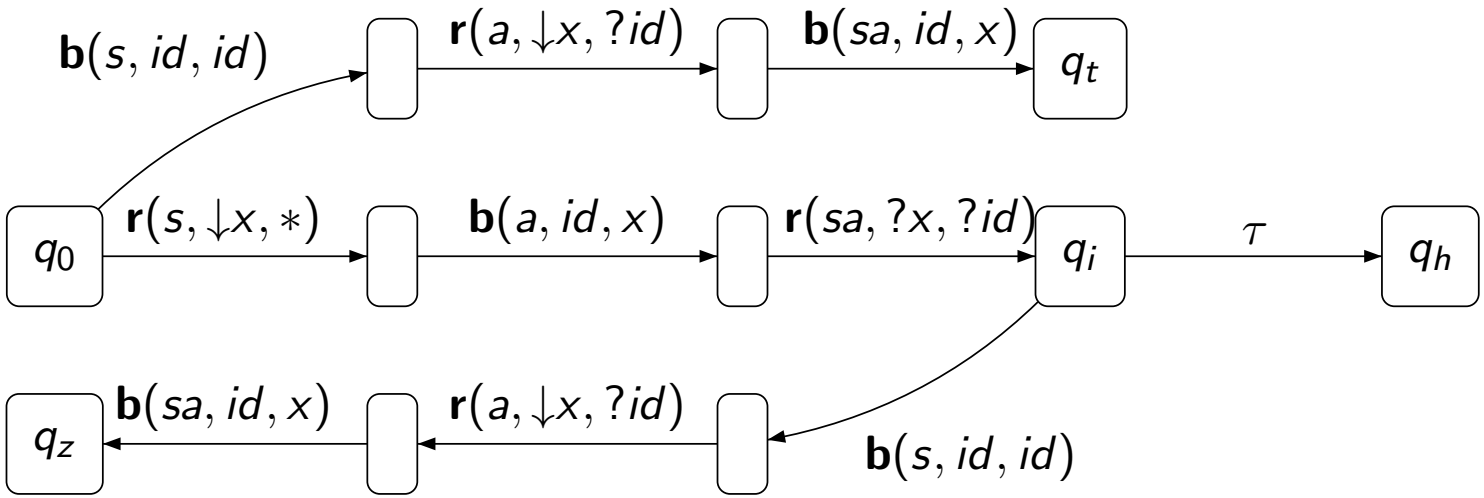
- $k \geq 0$ RW registers, 1 payload field $\Longrightarrow$ PSPACE-complete
- 1 RO + 1 RW registers, 2 payload fields $\Longrightarrow$ undecidable

AHN + Data + Reconf.: Undecidability

- For each restricted model for which it's possible, we provide a list-builder protocol.
- A list-builder uses registers as pointers to build linked lists of nodes.
- It is enough to define a list-builder to prove undecidability:
  - the protocol is extended in order to encode a 2-counter machine;
  - only the subset of features common to all models is used.

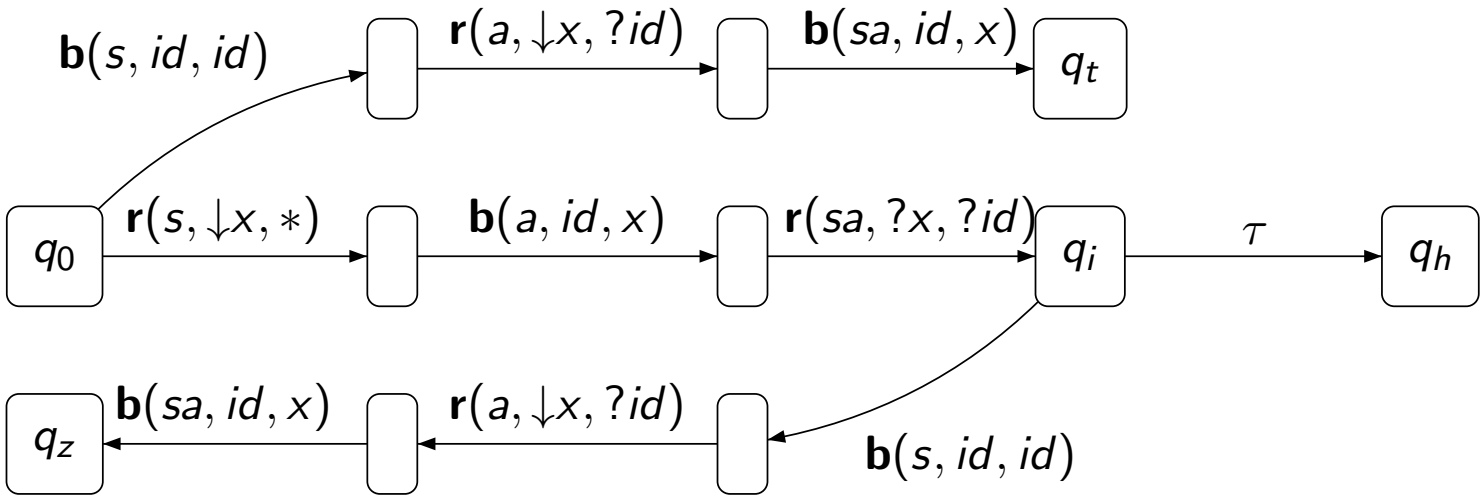
AHN + Data + Reconf.: List-Builder

network reconfigurations, 2 registers, 2 fields:



AHN + Data + Reconf.: List-Builder

network reconfigurations, 2 registers, 2 fields:



$\Longrightarrow$ undecidability

More Recent Work [TGC 2015, GandALF 2015]

Concurrent model for an abstraction of the life cycle of Android Components

- Activities are abstracted into finite state automata with local registers (to model data)
- Callbacks are defined via value passing messages
- Permissions are statically defined via a dependency graph between definitions

We study decidability boundaries for the following problems

- Violation of the permission model during component execution (run time errors)
- Detection of conflicts due to data exchanged by components with incompatible permissions

Decidability using History WSTS in which monadic predicates maintain footprints of data along a computation

Summarizing...

- ...we studied the impact on decidability and complexity for several basic features of ad hoc protocols, considering also the interplay between them:
  - synchronous communication,
  - asynchronous communication,
  - local clocks,
  - dynamic network reconfigurations,
  - data.
- ...decidability is hard to achieve in general, but not impossible:
  - search for good compromises between features of the model, approximations, and expressivity;
  - there is a relation between asynchronous communication and reconfigurations, but the latter is easier to handle.