Playing with probabilities in selective broadcast networks

Cassting 2015

Paulin Fournier

Université Rennes1

joint work with Nathalie Bertrand and Arnaud Sangnier

Motivation

- Distributed algorithms (*mutual exclusion, leader election, ...*)
- Telecommunication protocols (*routing, ...*)
- Algorithms for *ad-hoc* networks
- Model for biological systems

Motivation

- Distributed algorithms (*mutual exclusion, leader election, ...*)
- Telecommunication protocols (*routing, ...*)
- Algorithms for *ad-hoc* networks
- Model for biological systems

All participants have the same behavior

They form a **crowd** [Esparza, STACS'14]

Crowd networks

- Every process follows a same given protocol
- Processes can communicate, by either
  - Message passing
  - Shared variables
  - *Rendez-vous* communications
  - Broadcast communications
  - **Multi-diffusion (selective broadcasts)**

Crowd networks

- Every process follows a same given protocol
- Processes can communicate, by either
  - Message passing
  - Shared variables
  - *Rendez-vous* communications
  - Broadcast communications
  - **Multi-diffusion (selective broadcasts)**

Parameterized verification of crowd networks

Does the network conform to a given specification independently of the number of participants?

In this talk

Decidability and complexity
of parameterized reachability problems
in probabilistic networks

Features:

- **Probabilistic protocols**
- **Multi-diffusion communications**
- **Simple reachability questions**

In this talk

**Decidability and complexity
of parameterized reachability problems
in probabilistic networks**

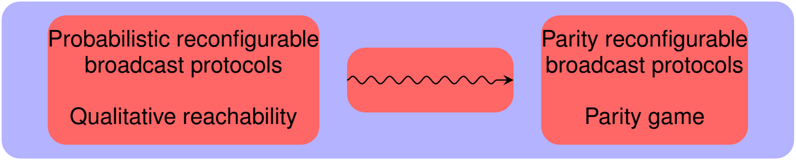
Features:

- **Probabilistic protocols**
- **Multi-diffusion communications**
- **Simple reachability questions**

**Challenge:
parameterized system + non-determinism + probabilities**

Outline

Parameterized verification of probabilistic reconfigurable networks



Section 1 Section 3 Section 2

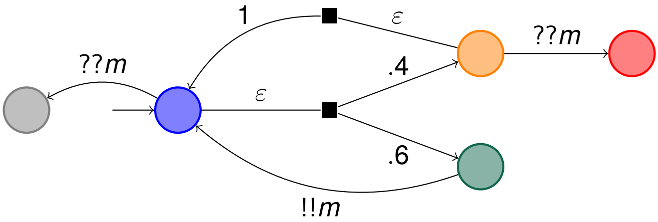
Outline

1. Probabilistic reconfigurable broadcast networks

2. Parity reconfigurable broadcast networks

3. Solving probabilistic networks via parity networks

A model for probabilistic protocols



Probabilistic protocol

Finite state system whose transitions are labelled with:

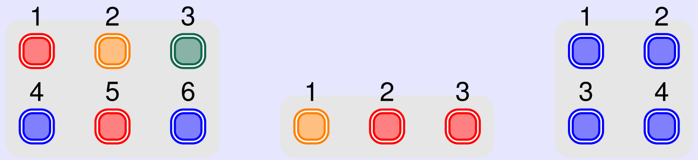
1. probabilistic internal actions - ε
2. broadcast of messages - $!!m$
3. reception of messages - $??m$

for m in a finite alphabet Σ.

A probabilistic protocol defines a probabilistic network

Configurations

Configurations: vectors of arbitrary size



- **Initial configurations**: **all** nodes are in the initial state

Remarks:

- Size of configurations is not bounded

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

Probabilistic Networks: semantics

Markov decision process over set of configurations.

- C: (infinite) set of configurations
- $\Rightarrow$: $C \times C \cup C \times Dist(C)$: Transition relation
- C_0: (infinite) set of initial configurations

The number of nodes does not change along an execution

Probabilistic Networks: semantics

Markov decision process over set of configurations.

- C: (infinite) set of configurations
- $\Rightarrow$: $C \times C \cup C \times Dist(C)$: Transition relation
- C_0: (infinite) set of initial configurations

The number of nodes does not change along an execution

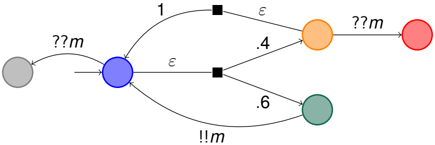
Transition relation

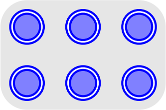
Decomposed in three steps

1. Choice of a process
2. Choice of a reception set (= set of neighbours)
3. Execution of an action
   - **local action** - the process performs an internal action ε
   - **communication** - the process sends a message ($!!m$), and its neighbours receive it ($??m$) if they can

Schedulers to resolve non-determinism

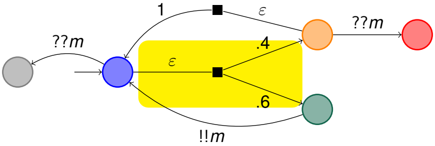
Scheduler π resolves the non-determinism by choosing the active process, its action and its neighbours

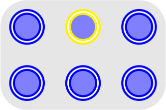




Schedulers to resolve non-determinism

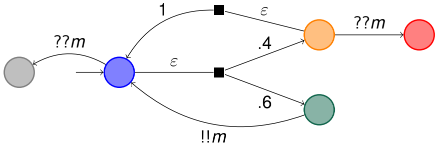
Scheduler π resolves the non-determinism
by choosing the active process, its action and its neighbours

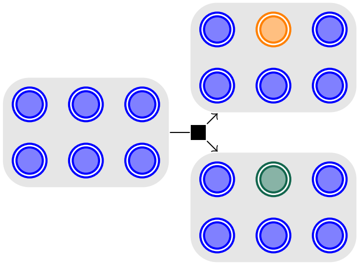




Schedulers to resolve non-determinism

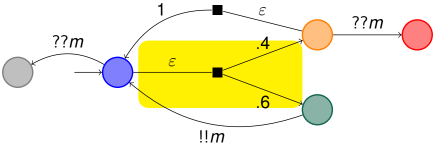
Scheduler π resolves the non-determinism
by choosing the active process, its action and its neighbours

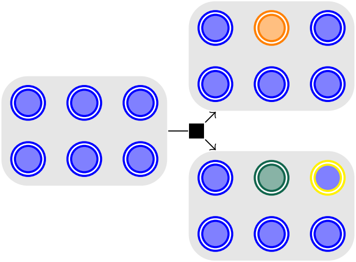




Schedulers to resolve non-determinism

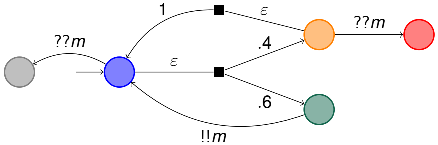
Scheduler π resolves the non-determinism
by choosing the active process, its action and its neighbours

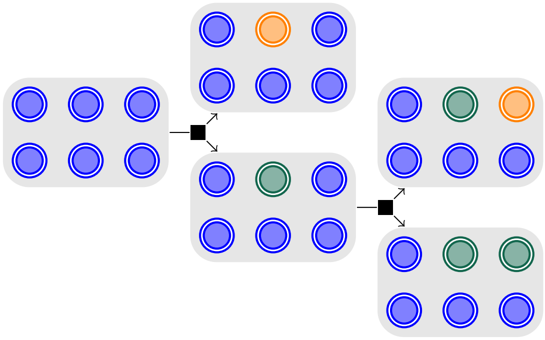




Schedulers to resolve non-determinism

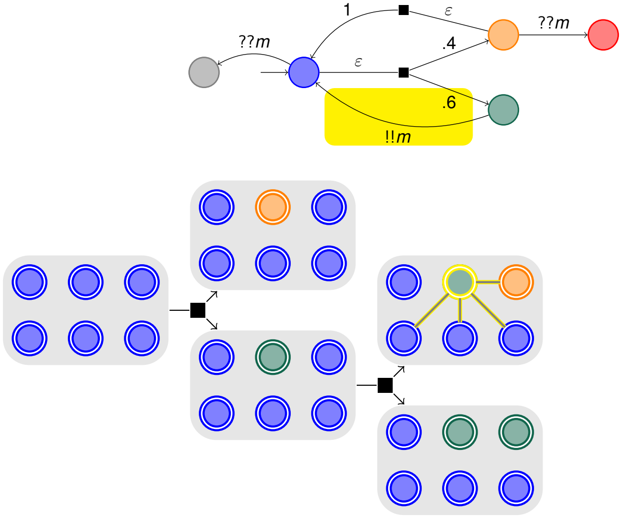
Scheduler π resolves the non-determinism
by choosing the active process, its action and its neighbours





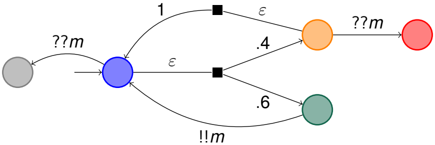
Schedulers to resolve non-determinism

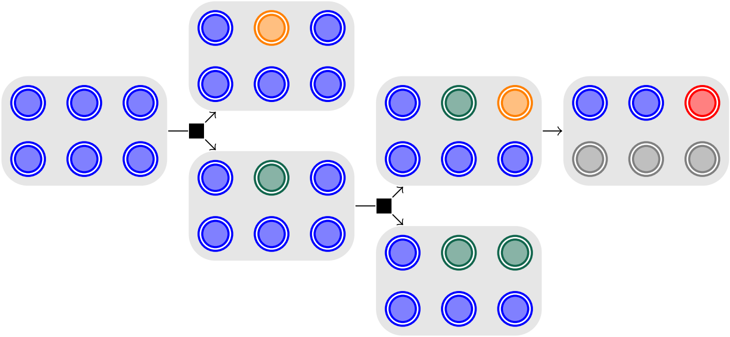
Scheduler π resolves the non-determinism
by choosing the active process, its action and its neighbours



Schedulers to resolve non-determinism

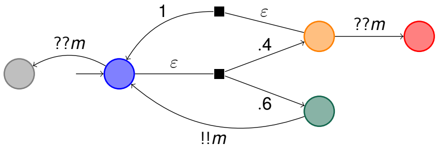
Scheduler π resolves the non-determinism
by choosing the active process, its action and its neighbours

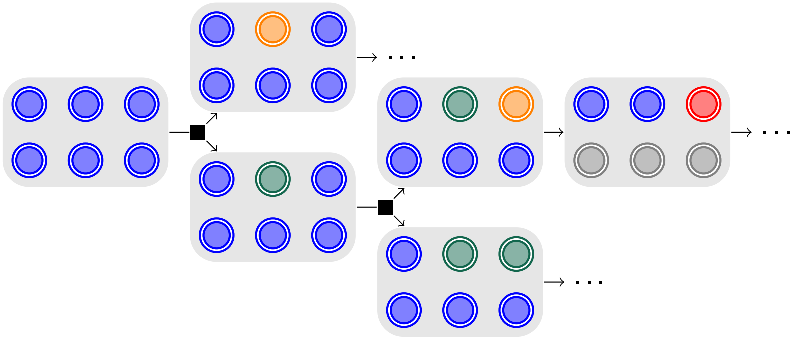




Schedulers to resolve non-determinism

Scheduler π resolves the non-determinism
by choosing the active process, its action and its neighbours





Parameterized reachability problems

scheduler π on N nodes induce a finite Markov chain of measure $\mathbb{P}_\pi^N$

Parameterized reachability problems

scheduler π on N nodes induce a finite Markov chain of measure $\mathbb{P}_\pi^N$

- Is an error state almost surely reachable, under some scheduler, and for some number of nodes?

$$\exists N,\ \exists \pi,\ \mathbb{P}_\pi^N(\Diamond q_{\mathtt{err}}) = 1$$

- Is an error state avoidable almost surely, under all adversarial schedulers, and for any number of nodes?

$$\forall N,\ \forall \pi,\ \mathbb{P}_\pi^N(\Diamond q_{\mathtt{err}}) = 0$$

Parameterized reachability problems

scheduler π on N nodes induce a finite Markov chain of measure $\mathbb{P}_\pi^N$

- Is an error state almost surely reachable, under some scheduler, and for some number of nodes? $\textbf{Reach}_{\text{max}}^{=1}$

$$\exists N,\ \exists \pi,\ \mathbb{P}_\pi^N(\Diamond q_{\texttt{err}}) = 1$$

- Is an error state avoidable almost surely, under all adversarial schedulers, and for any number of nodes? $\neg\textbf{Reach}_{\text{max}}^{>0}$

$$\forall N,\ \forall \pi,\ \mathbb{P}_\pi^N(\Diamond q_{\texttt{err}}) = 0$$

$\text{Reach}_{opt}^{\sim b}$ $\quad opt \in \{\min, \max\}, \sim \in \{>, <, \leq, \geq, =\}, b \in \{0, 1\}$

Input: A process and a control state $q_F \in Q$;

Output: Does there exists N such that $\underset{\pi}{opt}\left\{\mathbb{P}_\pi^N(\Diamond q_F)\right\} \sim b$?

Monotocity property and consequences

Monotonicity

With more nodes in the network, the maximum reachability probability can only increase.

Idea: ignore additional nodes

As a consequence, *e.g.*

$$\exists N,\ \exists \pi,\ \mathbb{P}^N_\pi(\Diamond q_F) = 0 \Longleftrightarrow \exists \pi,\ \mathbb{P}^1_\pi(\Diamond q_F) = 0$$

$\textsc{Reach}^{=0}_{\max}$ is decidable in $\textsc{PTime}$ by considering a single node.

Solving $\text{REACH}_{\text{max}}^{>0}$

Does there exists a N and a scheduler π such that $\mathbb{P}_{\pi}^{N}(\Diamond q_F) > 0$?

- Equivalent to parameterized control state reachability
- **Decidable in PTIME** [Delzanno et al., FSTTCS'12]
- One can compute the set of reachable control states in PTIME

Solving $\textsc{Reach}^{>0}_{\max}$

Does there exists a N and a scheduler π such that $\mathbb{P}^N_\pi(\Diamond q_F) > 0$?

- Equivalent to parameterized control state reachability
- **Decidable in PTIME** [Delzanno et al., FSTTCS'12]
- One can compute the set of reachable control states in PTIME

Not as easy for $\textsc{Reach}^{=1}_{\max}$!

Finite vs infinite MDPs

- Qualitative reachability is solvable in PTIME for finite MDPs by simple graph algorithms.
- Qualitative reachability in infinite-state MDPs: restricted to particular classes with *ad hoc* algorithms
  - non-deterministic and Probabilistic Lossy Channel Systems [Baier et al. 2007]
  - recursive Markov Decision Processes [Etessami et al. 2015]

Finite vs infinite MDPs

- Qualitative reachability is solvable in PTIME for finite MDPs by simple graph algorithms.
- Qualitative reachability in infinite-state MDPs: restricted to particular classes with *ad hoc* algorithms
  - non-deterministic and Probabilistic Lossy Channel Systems [Baier et al. 2007]
  - recursive Markov Decision Processes [Etessami et al. 2015]
- Alternative technique in the finite case: transformation into μ-calculus formula or parity game. [Chatterjee et al. 2007]

Finite vs infinite MDPs

- Qualitative reachability is solvable in PTIME for finite MDPs by simple graph algorithms.
- Qualitative reachability in infinite-state MDPs: restricted to particular classes with *ad hoc* algorithms
  - non-deterministic and Probabilistic Lossy Channel Systems [Baier et al. 2007]
  - recursive Markov Decision Processes [Etessami et al. 2015]
- Alternative technique in the finite case: transformation into μ-calculus formula or parity game. [Chatterjee et al. 2007]

How to adapt this methodology to probabilistic networks?

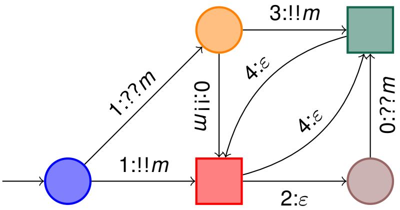
Main issues:

1. Transform MDP into equivalent parity game **at the protocol level**
2. Solve parity networks

Outline

1. Probabilistic reconfigurable broadcast networks
2. Parity reconfigurable broadcast networks
3. Solving probabilistic networks via parity networks

A model for parity protocol

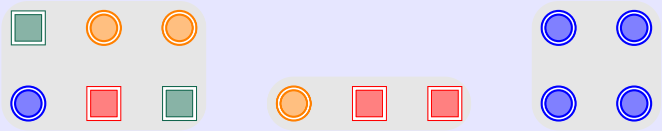


Parity protocol

-  states of Player 1
-  states of Player 2
- Transitions are labelled with:
  1. internal actions from Player 2's states – ε
  2. broadcast of messages – !!m
  3. reception of messages – ??m
  4. parities in $\mathbb{N}$

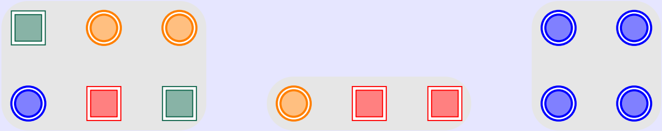
Semantics

Configurations: vectors of arbitrary size



Semantics

Configurations: vectors of arbitrary size



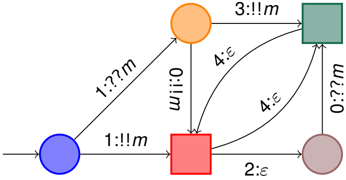
Roles are asymmetric

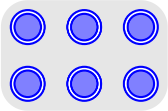
- Player 1 chooses the active process, and its neighbours
- If the active process is in a Player i's state, Player i chooses its action

Strategy profile (σ, τ) yields a play ρ

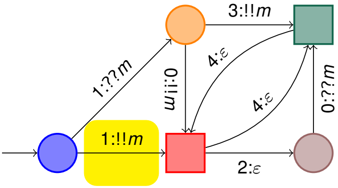
In communication transitions, the parity is the one of the corresponding broadcast

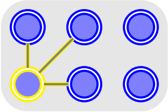
An example of play



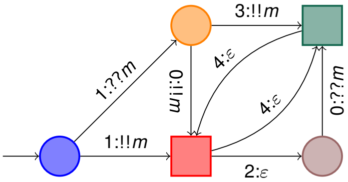


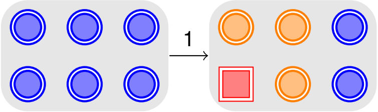
An example of play



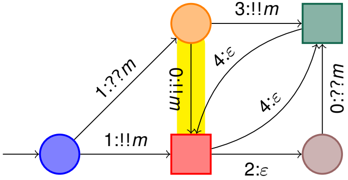


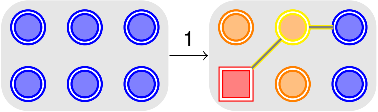
An example of play



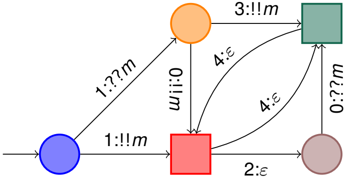


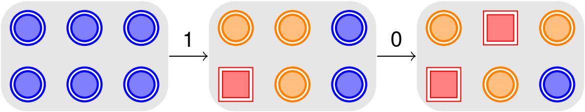
An example of play



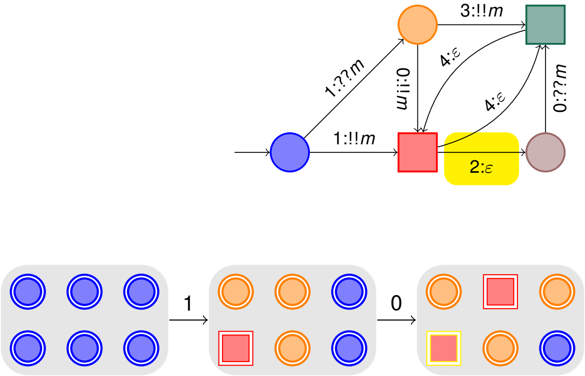


An example of play

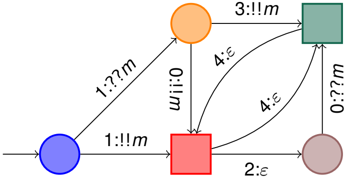


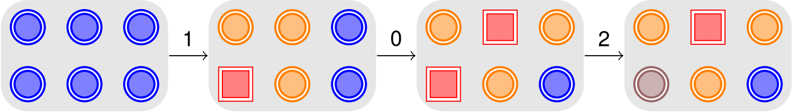


An example of play

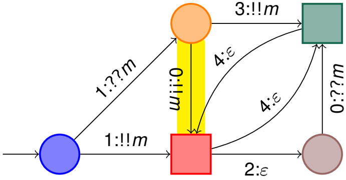


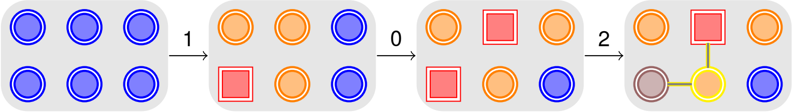
An example of play



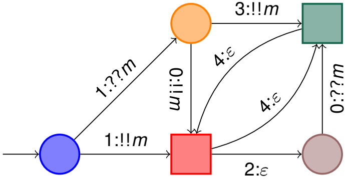


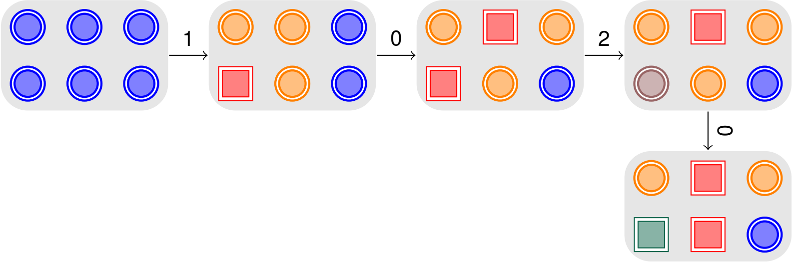
An example of play



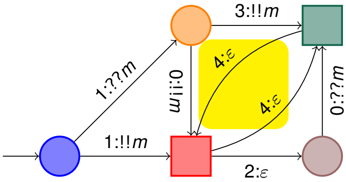


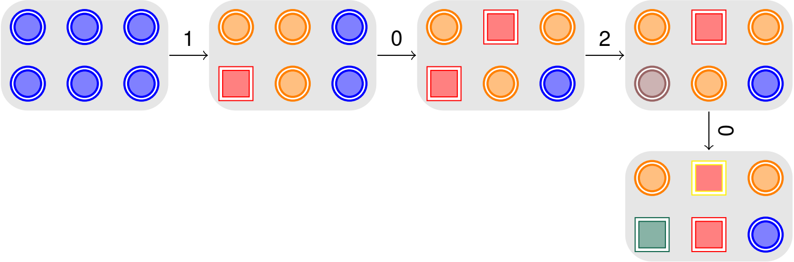
An example of play



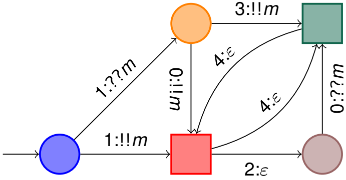


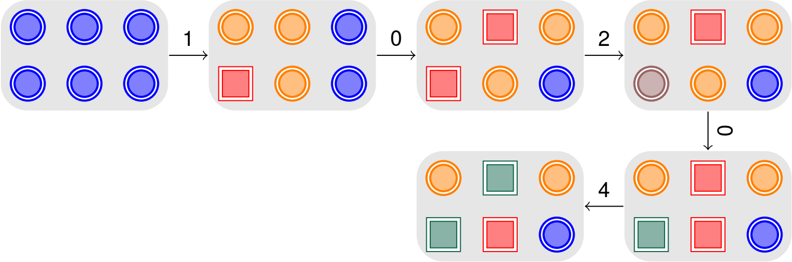
An example of play



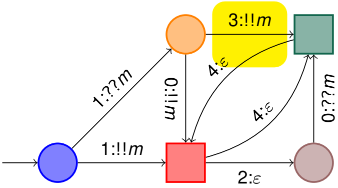


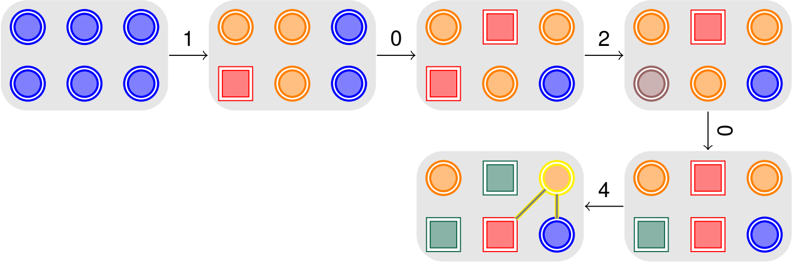
An example of play



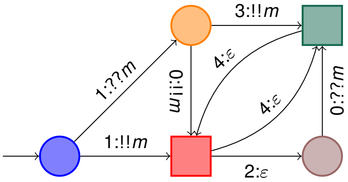


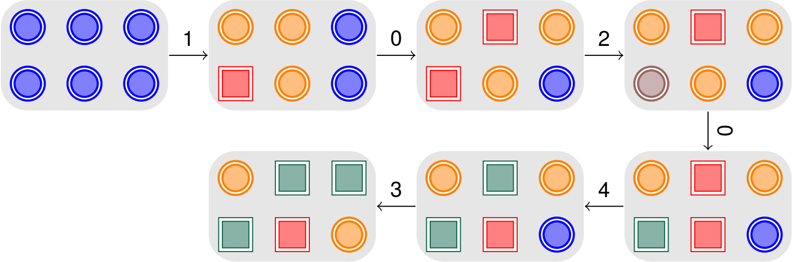
An example of play



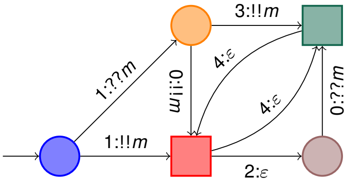


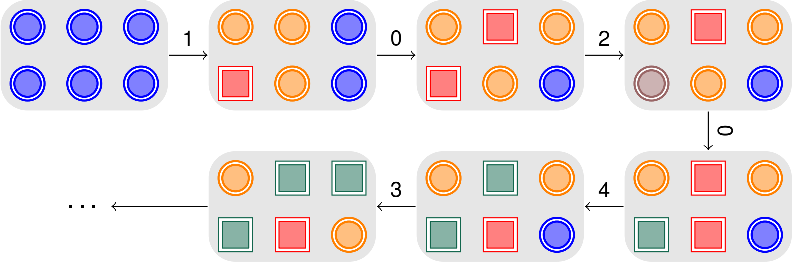
An example of play





An example of play





Parameterized game problem

Winning condition

$Win = \{\rho \texttt{ infinite play} \mid \max\{c | \#_c(\rho) = \infty\} \texttt{ is even}\}$

Parameterized game problem

Winning condition

$Win = \{\rho \texttt{ infinite play} \mid \max\{c | \#_c(\rho) = \infty\} \texttt{ is even}\}$

Does Player 1 has a winning strategy for the parity objective for some number of nodes?

Game problem for parity networks

Input: A parity protocol P
Question: Does there exists N and a strategy σ for Player 1 such that for all strategies τ for Player 2 $\rho(\sigma, \tau, N) \in Win$.

Solving games on parity networks

Two steps

1. state-based strategies for Player 2 are enough
2. decidability of the existence of an infinite cycle in reconfigurable broadcast networks (*i.e.* networks of 1-player games)

State-based strategies for Player 2

- only depend on the control state labeling the active node
- there are finitely many
- given a fixed state-based strategy for Player 2, one obtains a reconfigurable broadcast network

Step 1: Restricting to state-based strategies

Proposition

Player 1 has a winning strategy against any **state-based** strategy of Player 2

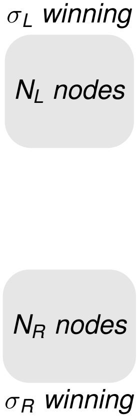
$$\Leftrightarrow$$

Player 1 has a winning strategy against any strategy of Player 2

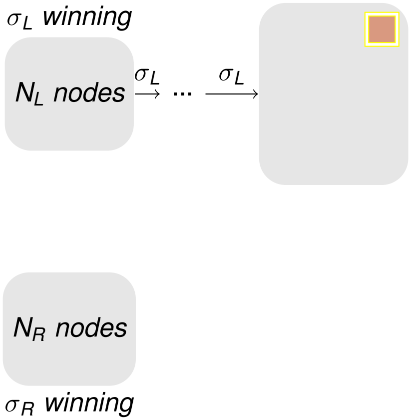
Proof by induction of the number of states of Player 2

- For the induction step, isolate a Player 2 state with two possible internal actions ε_L and ε_R
- By induction, if edge ε_R is deleted, Player 1 has a winning strategy σ_L for N_L nodes, and symmetrically
- A winning strategy is obtained combining σ_L and σ_R on $N_L + N_R$ nodes

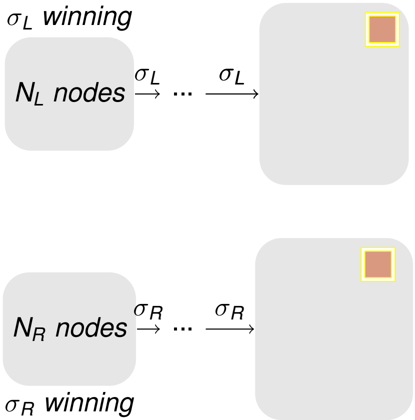
Building a strategy using σ_L and σ_R



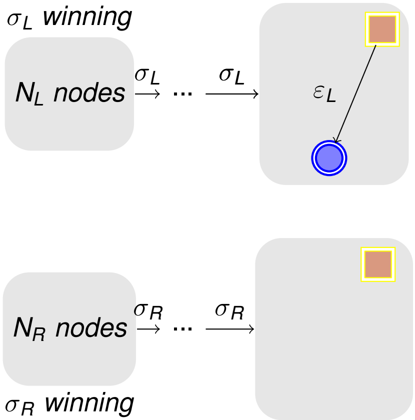
Building a strategy using σ_L and σ_R



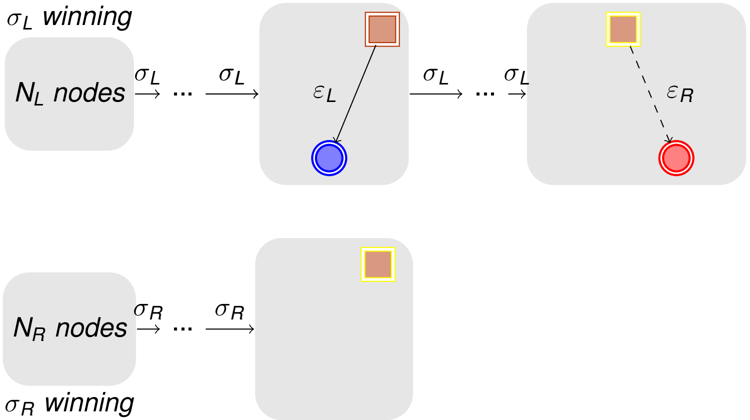
Building a strategy using σ_L and σ_R



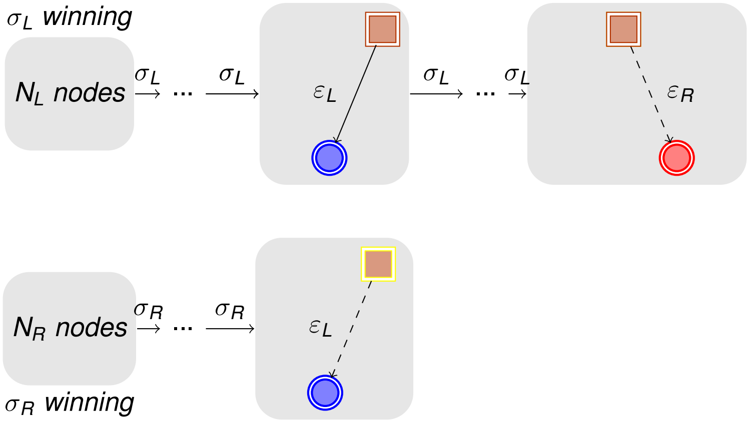
Building a strategy using σ_L and σ_R



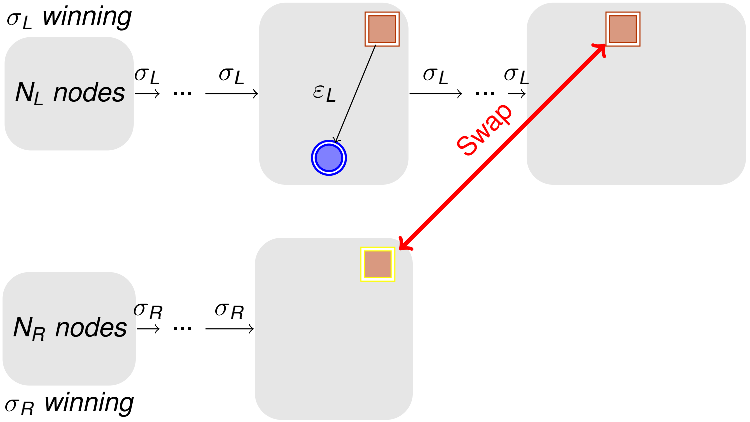
Building a strategy using σ_L and σ_R



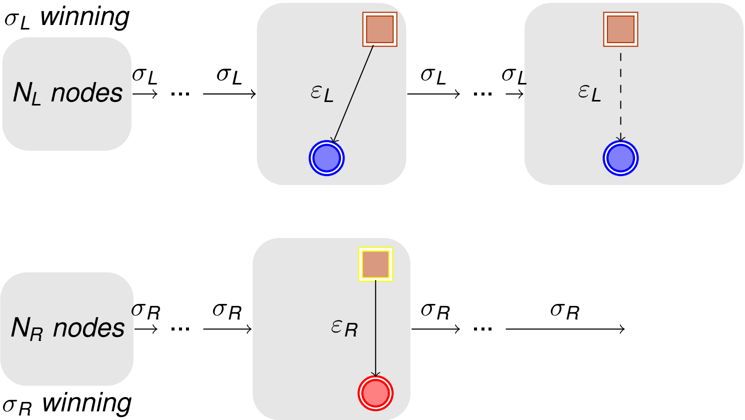
Building a strategy using σ_L and σ_R



Building a strategy using σ_L and σ_R

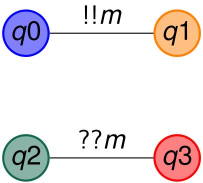


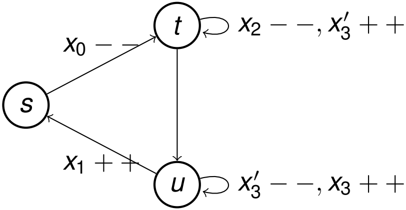
Building a strategy using σ_L and σ_R



Step 2: Detecting infinite paths

- For a fixed state-based strategy for Player 2, one obtains a reconfigurable broadcast network
- An infinite path corresponds to a positive cycle in a VASS





[Kosaraju & Sullivan 1988]

Detecting positive cycles in VASS can be done in PTIME

Deciding the game problem for parity networks

Theorem

The game problem for parity RBN is in coNP.

Proof idea:

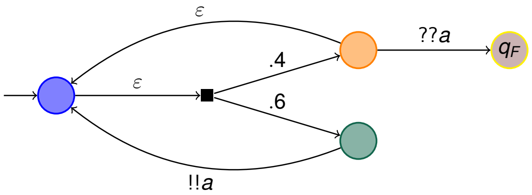
- Guess a state-based strategy τ for Player 2
- Check whether it is winning for any number of nodes and against any strategies for Player 1
  - If the VASS has a positive cycle, τ it is not winning
  - Can be decided in PTIME
- If the state-based strategy τ is winning, then return NO

Outline

1. Probabilistic reconfigurable broadcast networks
2. Parity reconfigurable broadcast networks
3. **Solving probabilistic networks via parity networks**

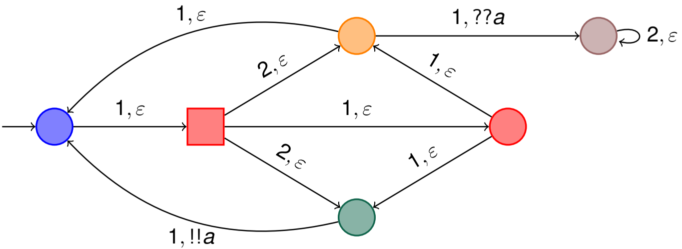
Solving $\textsc{Reach}^{=1}_{\max}$: $\exists N,\ \exists \pi,\ \mathbb{P}^{N}_{\pi}(\Diamond q_F) = 1$

Solving $\text{REACH}^{=1}_{\max}$: $\exists N,\ \exists \pi,\ \mathbb{P}^N_\pi(\Diamond q_F) = 1$



Idea of the reduction:

- Player 2 decides the outcome of probabilistic choices
- Fairness is ensured using parities



Correctness of the reduction for $\text{REACH}^{=1}_{\max}$

configurations in prob. network $\equiv$ configurations in parity network

schedulers $\equiv$ Player 1 strategies

Correctness of the reduction for $\textsc{Reach}_{max}^{=1}$

configurations in prob. network $\equiv$ configurations in parity network
schedulers $\equiv$ Player 1 strategies

Key: $\textsc{Reach}_{max}^{=1}$ iff from every reachable configuration there is a path to a target configuration

Correctness of the reduction for $\textsc{Reach}^{=1}_{\max}$

configurations in prob. network $\equiv$ configurations in parity network
schedulers $\equiv$ Player 1 strategies

Key: $\textsc{Reach}^{=1}_{\max}$ iff from every reachable configuration there is a path to a target configuration

Proof idea:

- If Player 1 has a winning strategy
  - **case 1** Player 2 always decides the outcome of probabilistic choices; corresponds to paths in null measure set
  - **case 2** Player 2 eventually always leave decision to Player 1; from each reachable configuration, there is a path to the target
- If Player 1 has no winning strategy
  For every σ, Player 2 eventually lets Player 1 decide the outcome of probabilistic choices;
  there exists a configuration from which target is not reachable

Complexity of almost-sure reachability

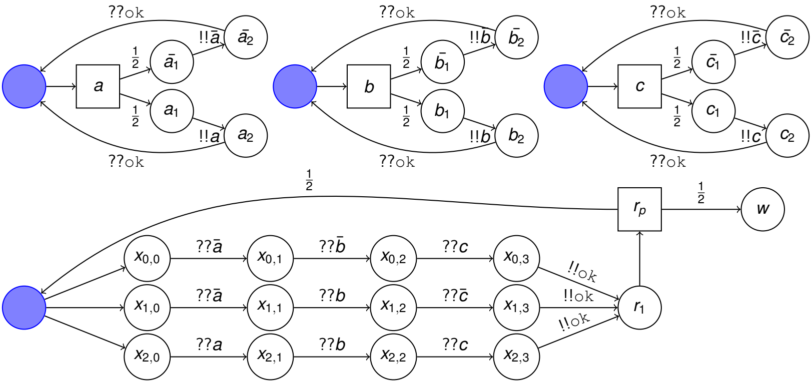
Theorem

$\textsc{Reach}^{=1}_{\text{max}}$ is $\textsc{co}$NP-complete.

- membership in $\textsc{co}$NP by reduction to games on parity networks
- $\textsc{co}$NP-hardness is obtained by reducing UNSAT

coNP-hardness of almost-sure reachability

$$\varphi = (a \vee b \vee \bar{c}) \wedge (a \vee \bar{b} \vee c) \wedge (\bar{a} \vee \bar{b} \vee \bar{c})$$



If φ is UNSAT, for any assignment, choose a clause so that the probability to reach w is .5.

Conclusion

Summary

- **model**: probabilistic selective broadcast networks
- **properties**: parameterized qualitative reachability questions
- **resolution**: via parity networks, yet another new model
- **complexities**: PTIME or coNP-complete

Conclusion

Summary

- **model**: probabilistic selective broadcast networks
- **properties**: parameterized qualitative reachability questions
- **resolution**: via parity networks, yet another new model
- **complexities**: PTIME or CONP-complete

Perspectives

- move to quantitative questions
- beyond reachability
- consider other communication means
- schedulers taking into account processes local view