Reachability in Networks of Register Protocols under a Stochastic Scheduler

Patricia Bouyer[1] Nicolas Markey[1] Mickael Randour[2]
Arnaud Sangnier[3] Daniel Stan[1]

[1]LSV - CNRS & ENS Cachan, France

[2]ULB, Belgium

[3]IRIF - CNRS & Université Paris Diderot, France

ICALP 2016

Context: distributed systems

Goal

Study distributed systems composed of *many identical components* running concurrently.

Useful for distributed algorithms, ad-hoc networks, communication protocols, etc.

Context: distributed systems

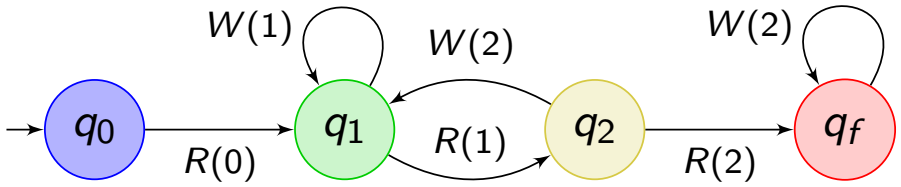
Goal

Study distributed systems composed of *many identical components* running concurrently.

Useful for distributed algorithms, ad-hoc networks, communication protocols, etc.

- Each process is (a copy of) an *automaton*, a *PDS* ...
- Different means of communication: *Rendez-vous* ([GS92]), *broadcast messages* ([EFM99, DSZ10]), *shared register* ([ABG15, EGM13])...

Our model: summary



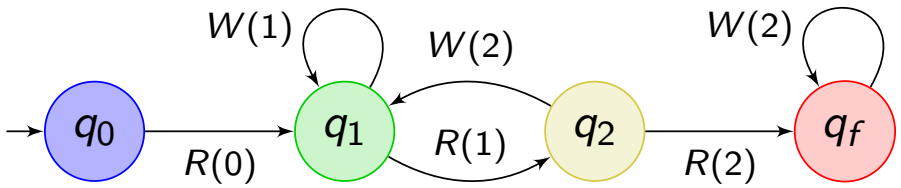
Register protocol with $D = \{0, 1, 2\}$.

Definition: register protocol

$\mathscr{P} = \langle Q, D, q_0, d_0, q_f, T \rangle$

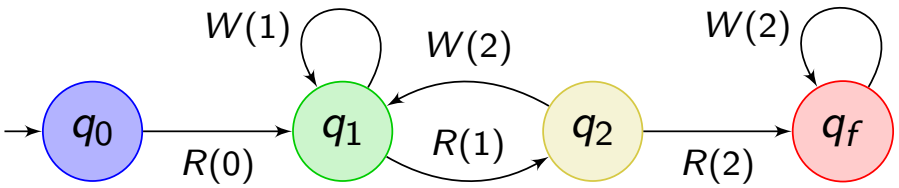
- $\langle Q, q_0, q_f, T \rangle$ is a finite state automaton;
- D finite dataset for the shared register;
- d_0 an initial value;
- $T \subseteq Q \times \{R, W\} \times D \times Q$ set of transitions, labelled by read and write operations over D.

Our model

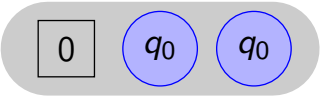


Network for two processes (self-loops omitted).

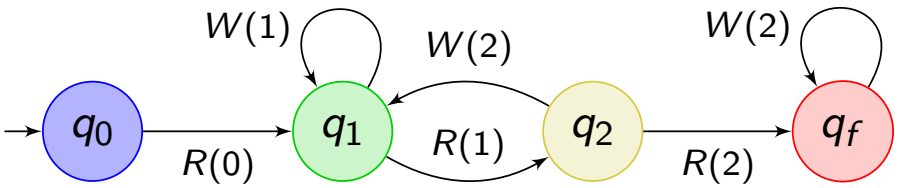
Our model



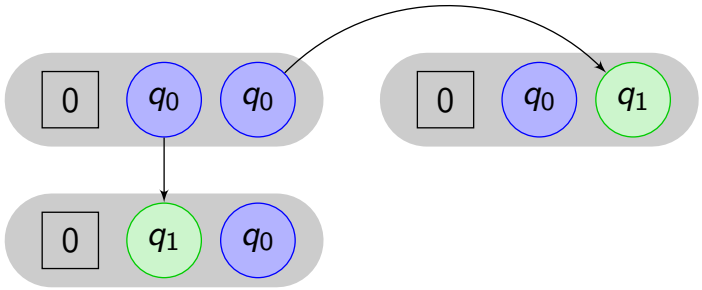
Network for two processes (self-loops omitted).



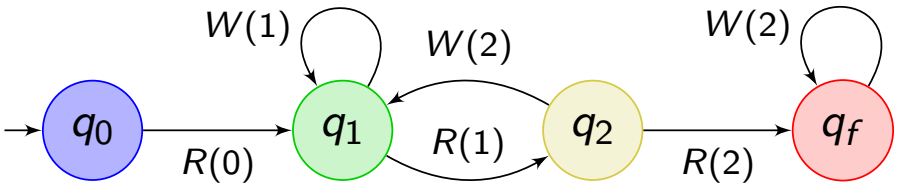
Our model



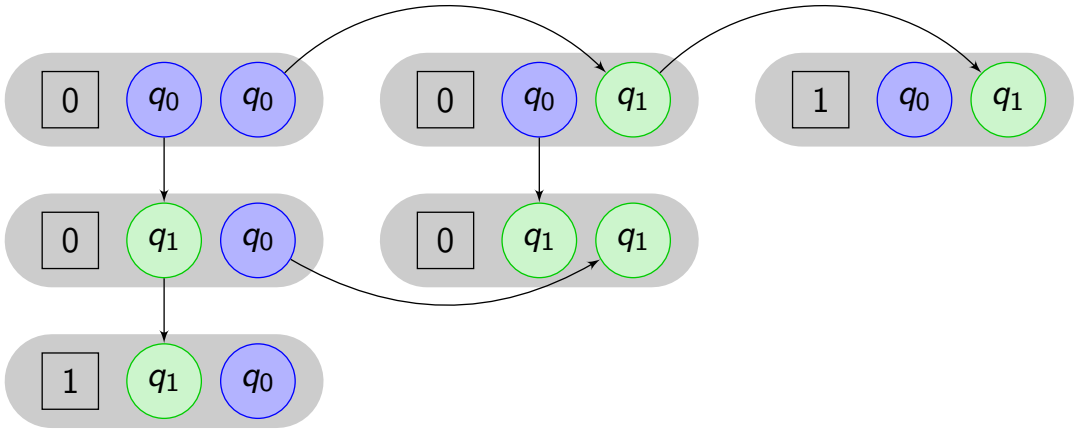
Network for two processes (self-loops omitted).



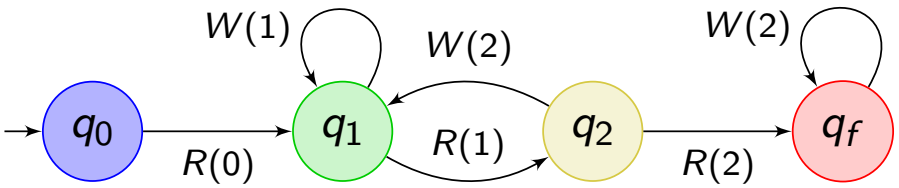
Our model



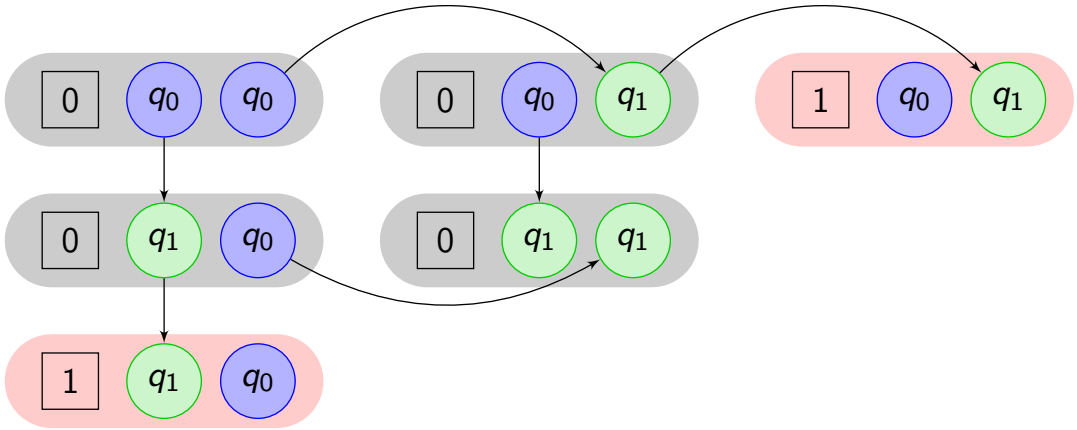
Network for two processes (self-loops omitted).



Our model

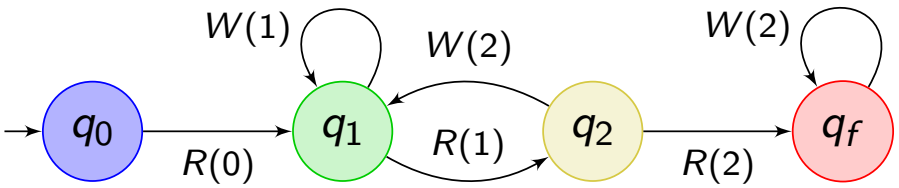


Network for two processes (self-loops omitted).

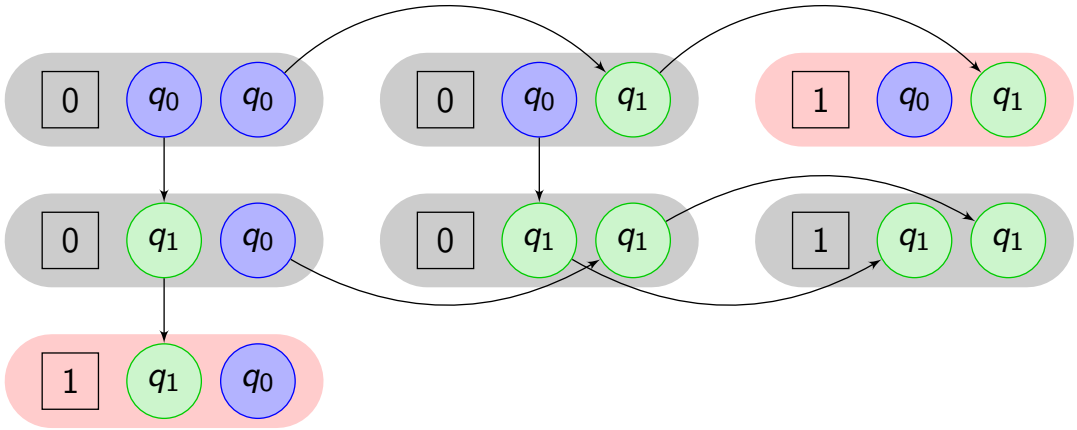


- There exist paths from there, the processes in q_0 are trapped;

Our model



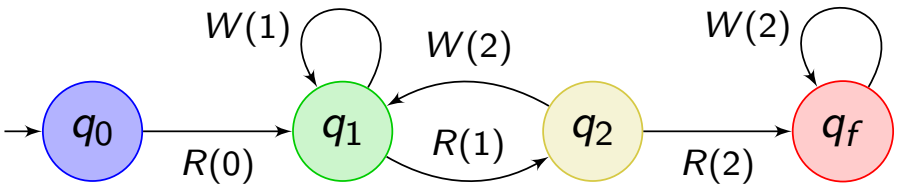
Network for two processes (self-loops omitted).



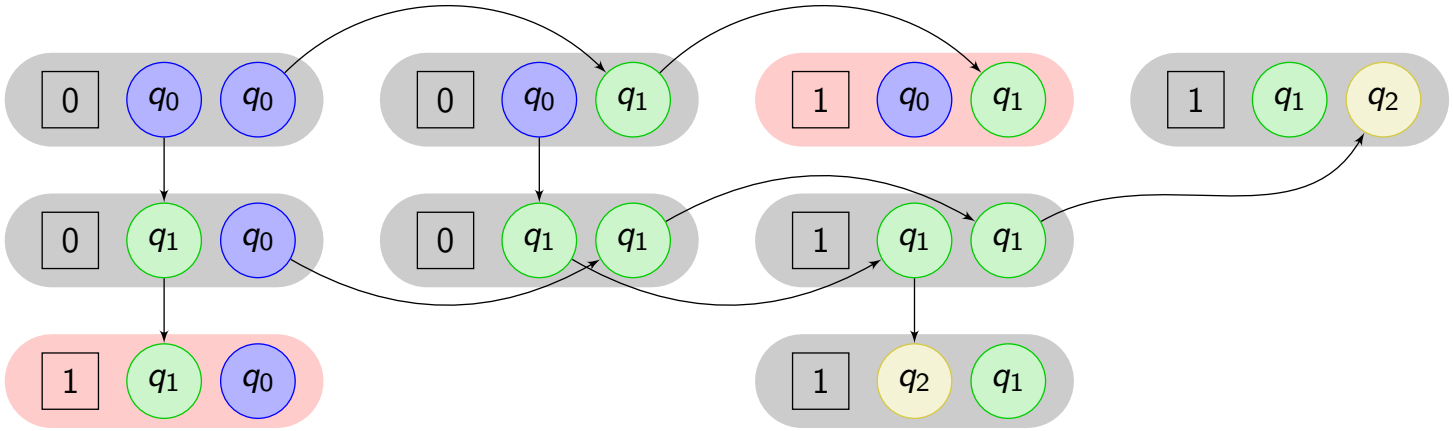
(Non-exhaustive construction)

- There exist paths from there, the processes in q_0 are trapped;

Our model



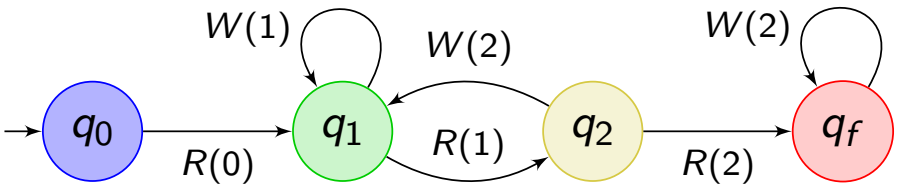
Network for two processes (self-loops omitted).



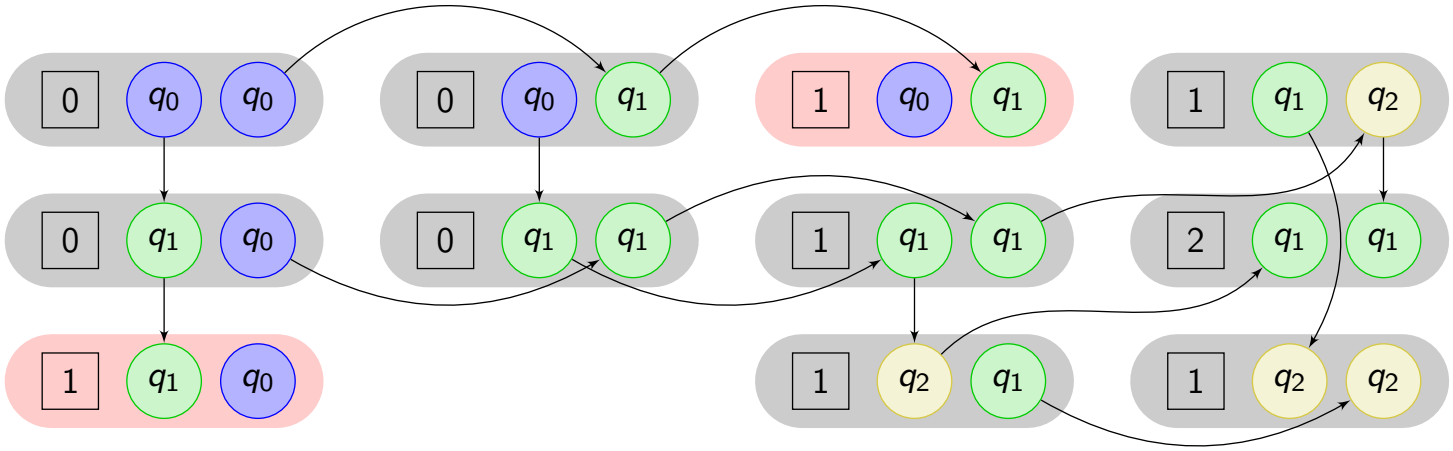
(Non-exhaustive construction)

- There exist paths from there, the processes in q_0 are trapped;

Our model



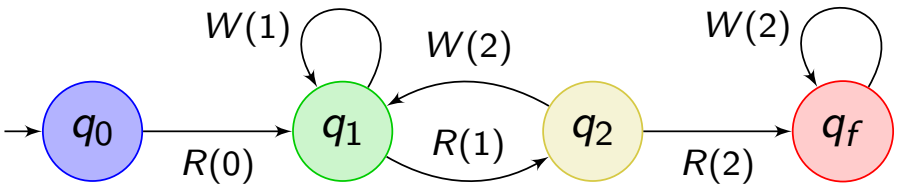
Network for two processes (self-loops omitted).



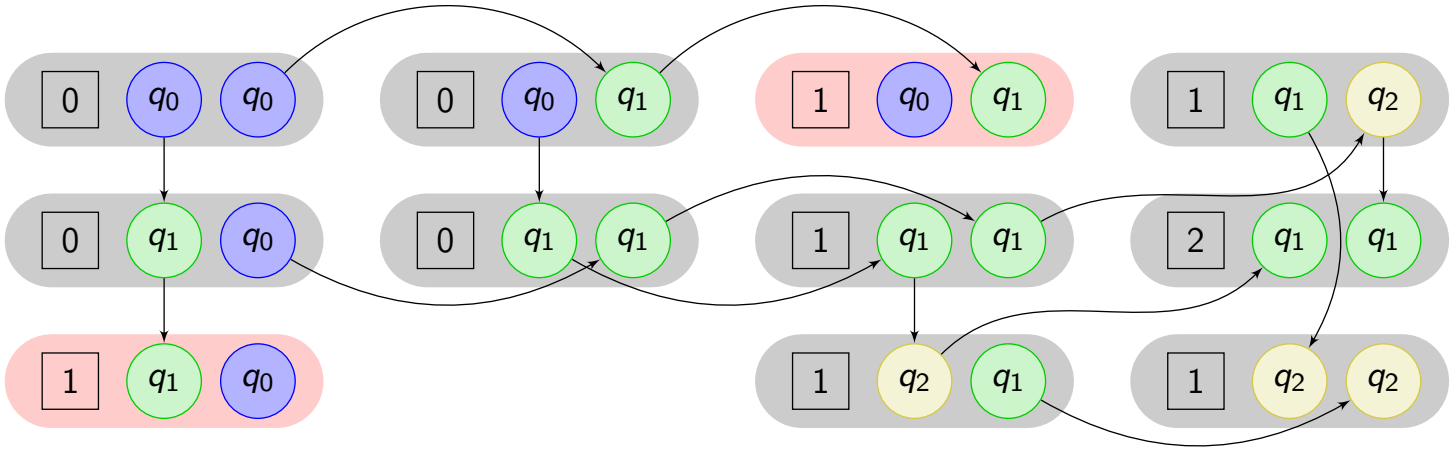
(Non-exhaustive construction)

- There exist paths from there, the processes in q_0 are trapped;

Our model



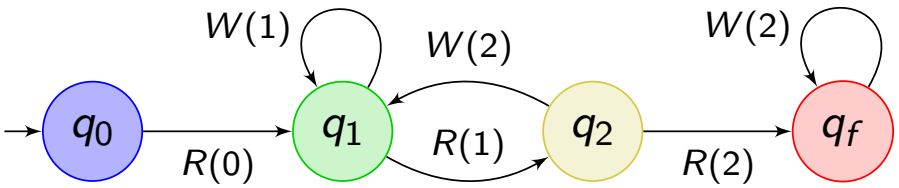
Network for two processes (self-loops omitted).



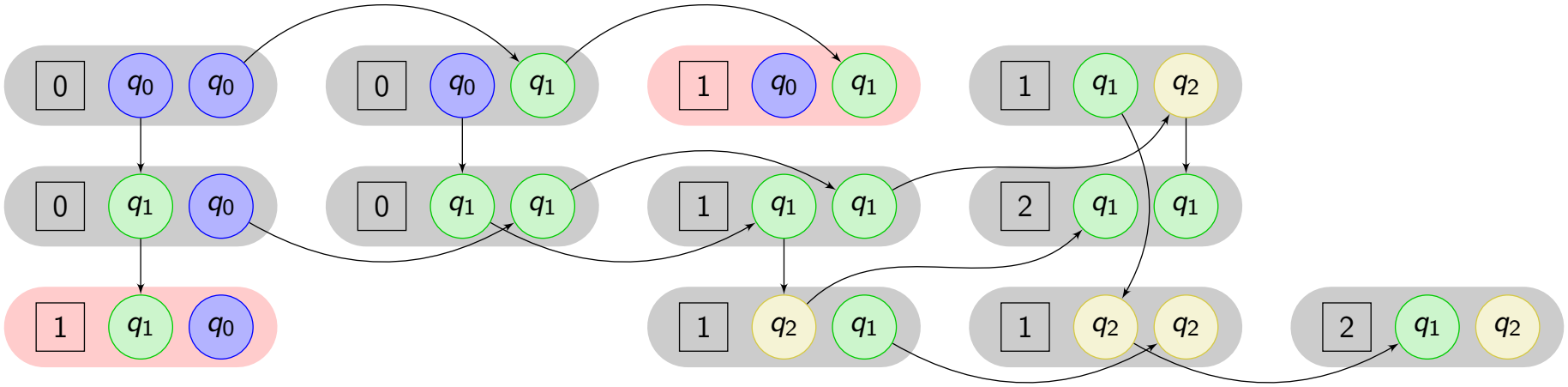
(Non-exhaustive construction)

- There exist paths from there, the processes in q_0 are trapped;

Our model



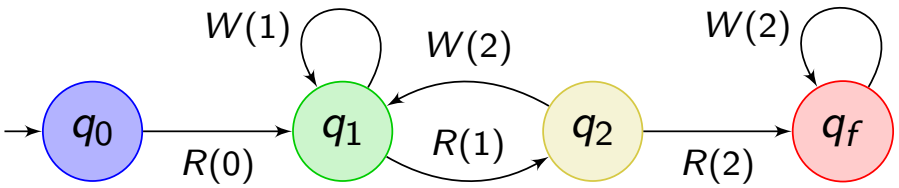
Network for two processes (self-loops omitted).



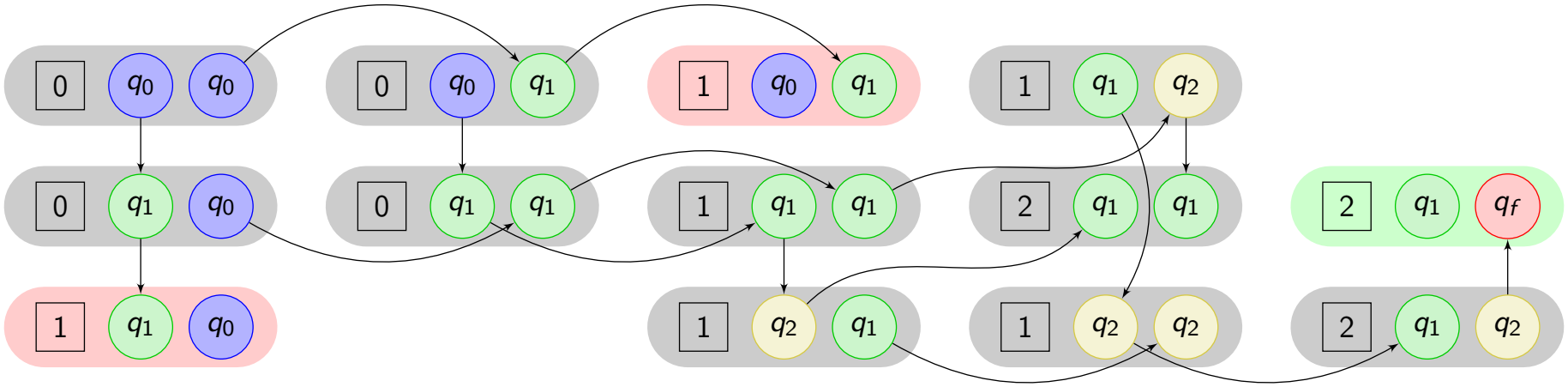
(Non-exhaustive construction)

- There exist paths from there, the processes in q_0 are trapped;

Our model



Network for two processes (self-loops omitted).



(Non-exhaustive construction)

- There exist paths from there, the processes in q_0 are trapped;
- There exist paths that reach q_f...
- ...and they require at least two processes.

- A **configuration** $\gamma \in \Gamma$ is a <u>multiset</u> of states + a <u>register value</u> $d \in D$;

- A **configuration** $\gamma \in \Gamma$ is a <u>multiset</u> of states + a <u>register value</u> $d \in D$;

- **Parameter**: once the system is started, the configuration has a **fixed** size;
- **Interleaving** semantics;
- **Non-atomic** operations (read <u>or</u> write at a time);
- Goal: reach a configuration which **covers** q_f.

- A **configuration** $\gamma \in \Gamma$ is a <u>multiset</u> of states + a <u>register value</u> $d \in D$;

- **Parameter**: once the system is started, the configuration has a **fixed** size;
- **Interleaving** semantics;
- **Non-atomic** operations (read <u>or</u> write at a time);
- Goal: reach a configuration which **covers** q_f.
- How does the **scheduler** work ?

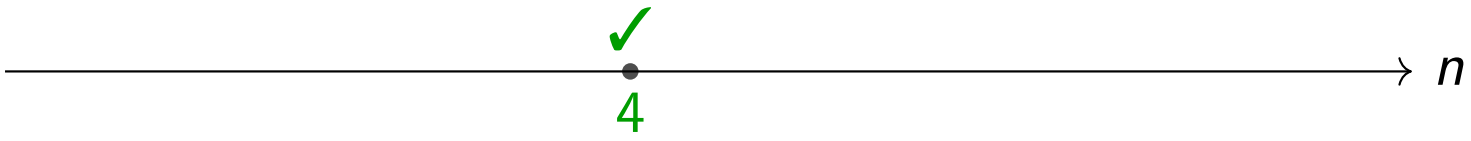
Non-deterministic Scheduler Case

- The scheduler is **helpful**;
- **Monotonicity:** if q_f is reachable with n initial processes, it is with $n+1$;

$$\longrightarrow n$$

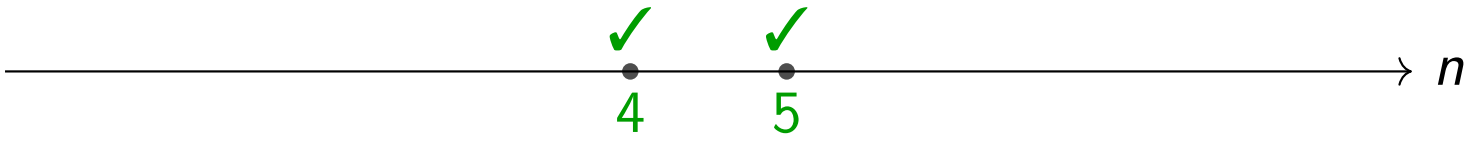
Non-deterministic Scheduler Case

- The scheduler is **helpful**;
- **Monotonicity:** if q_f is reachable with n initial processes, it is with $n+1$;



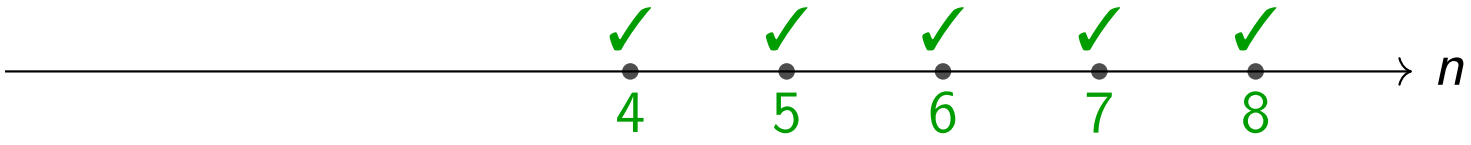
Non-deterministic Scheduler Case

- The scheduler is **helpful**;
- **Monotonicity:** if q_f is reachable with n initial processes, it is with $n+1$;



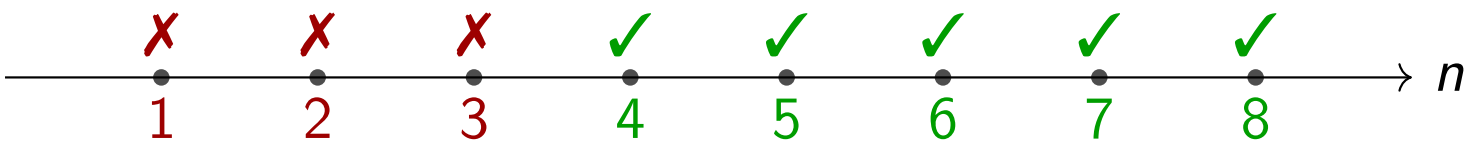
Non-deterministic Scheduler Case

- The scheduler is **helpful**;
- **Monotonicity:** if q_f is reachable with n initial processes, it is with $n+1$;



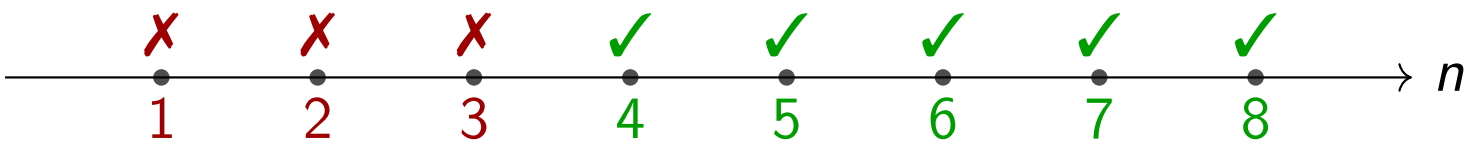
Non-deterministic Scheduler Case

- The scheduler is **helpful**;
- **Monotonicity:** if q_f is reachable with n initial processes, it is with $n+1$;



Non-deterministic Scheduler Case

- The scheduler is **helpful**;
- **Monotonicity:** if q_f is reachable with n initial processes, it is with $n+1$;
- ...and there exists a witness of polynomial size.



Theorem ([EGM13])

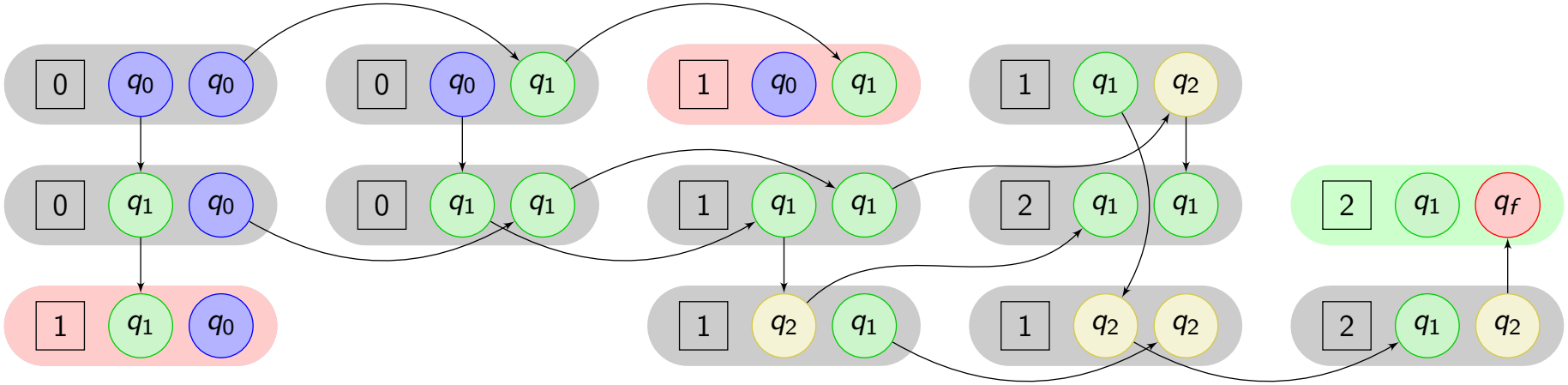
Given a protocol, one can decide in PTIME *whether there exists a parameter such that q_f is reachable.*
The problem becomes NP*-complete, when one of the initial process is distinguished from the others (leader).*

Fair, Probabilistic scheduler

- We don't control the scheduler anymore;
- Stochastic behaviour (environment);
- Finite patterns cannot be repeated infinetely often;
- We consider **almost-sure** reachability: $\mathbb{P}_n(\diamond \uparrow q_f) \stackrel{?}{=} 1$

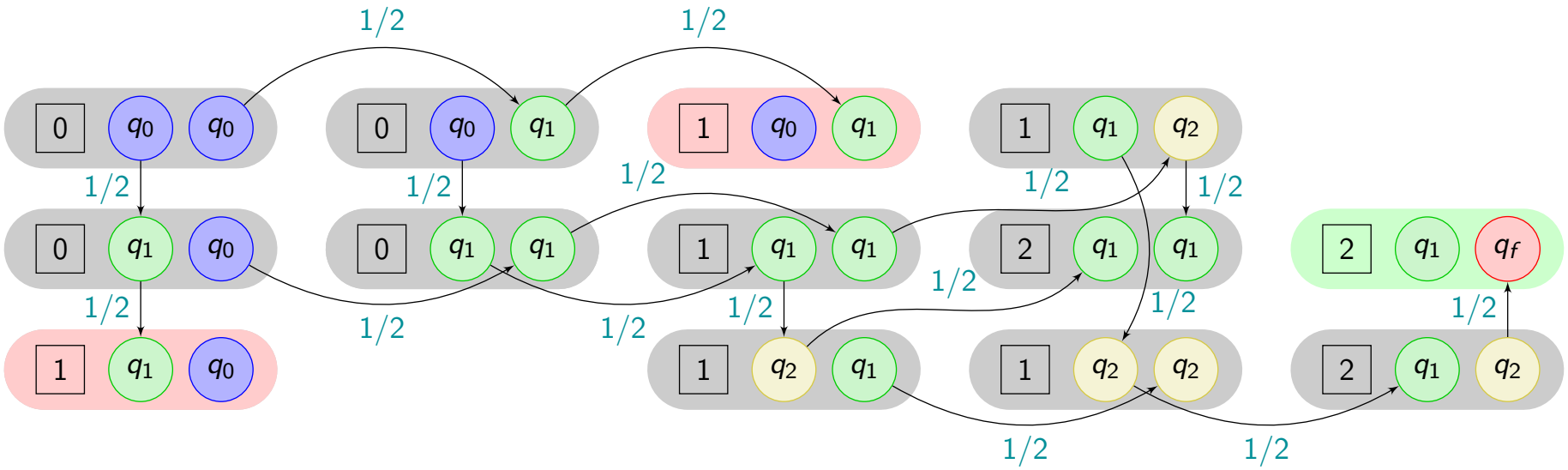
Fair, Probabilistic scheduler

- We don't control the scheduler anymore;
- **Stochastic** behaviour (environment);
- Finite patterns cannot be repeated infinetely often;
- We consider **almost-sure** reachability: $\mathbb{P}_n(\diamond \uparrow q_f) \stackrel{?}{=} 1$



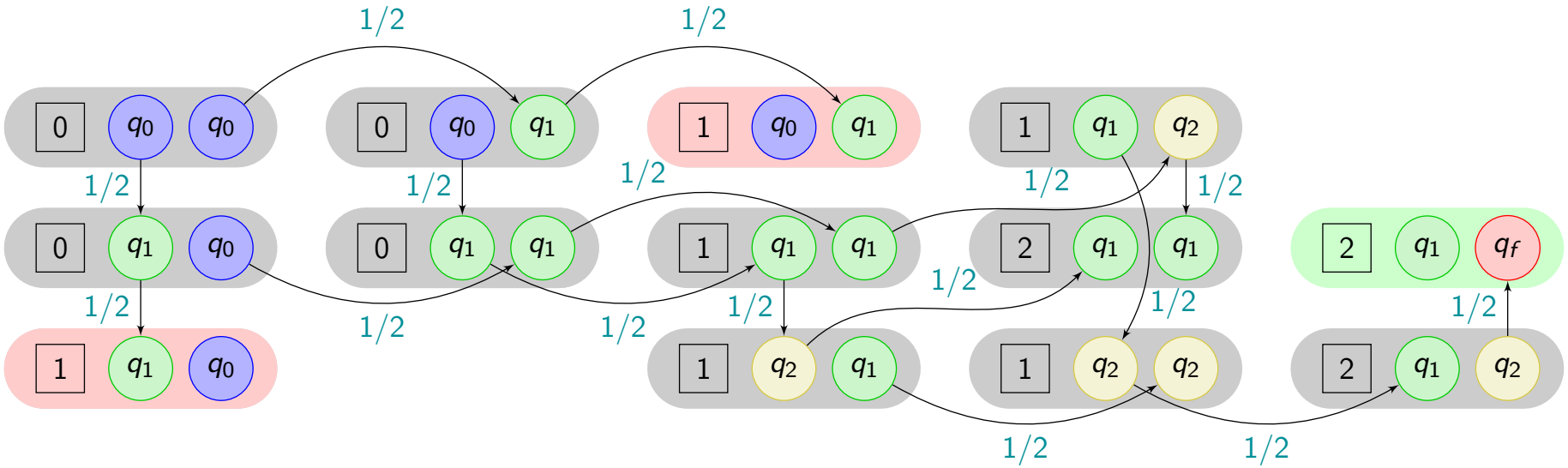
Fair, Probabilistic scheduler

- We don't control the scheduler anymore;
- **Stochastic** behaviour (environment);
- Finite patterns cannot be repeated infinetely often;
- We consider **almost-sure** reachability: $\mathbb{P}_n(\diamond \uparrow q_f) \stackrel{?}{=} 1$



Fair, Probabilistic scheduler

- We don't control the scheduler anymore;
- **Stochastic** behaviour (environment);
- Finite patterns cannot be repeated infinetely often;
- We consider **almost-sure** reachability: $\mathbb{P}_n(\diamond \uparrow q_f) \stackrel{?}{=} 1$

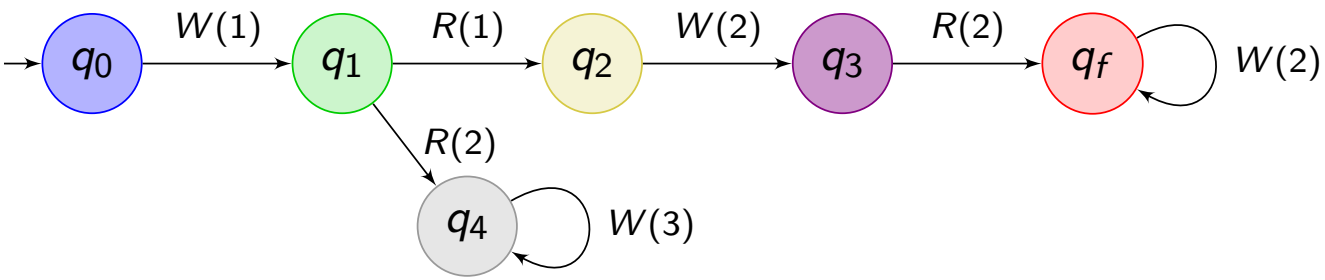


Qualitative property
+
Finite configuration space

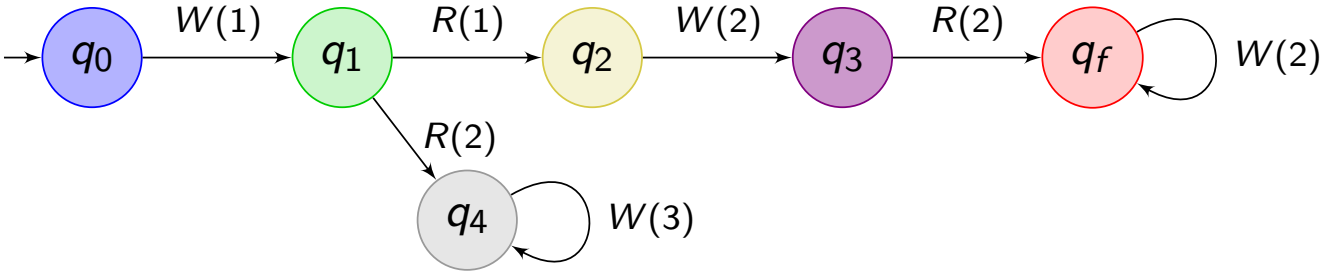
$\Longrightarrow$

Exact probability values are **not** relevant.

Lack of monotonicity

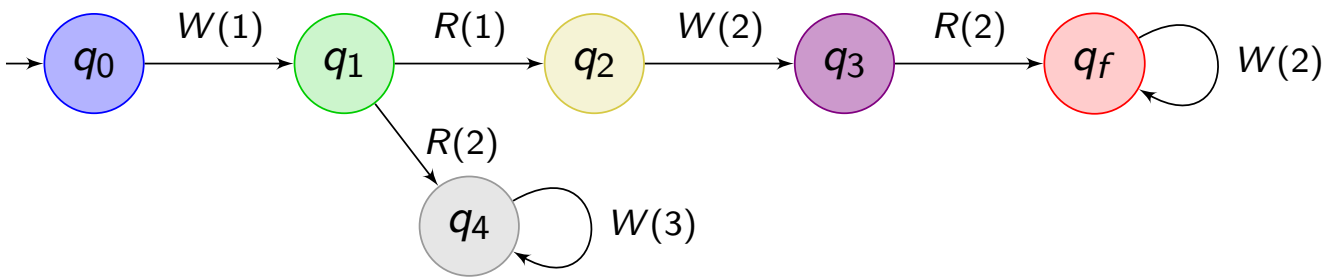


Lack of monotonicity

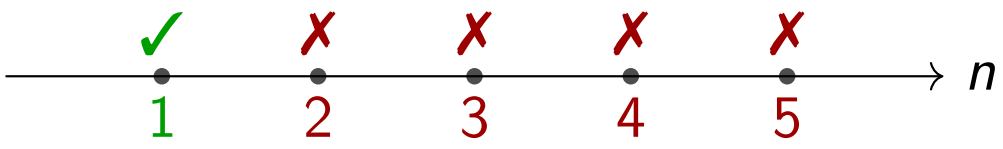


$\Longrightarrow$ **Additional processes can create new deadlocks!**

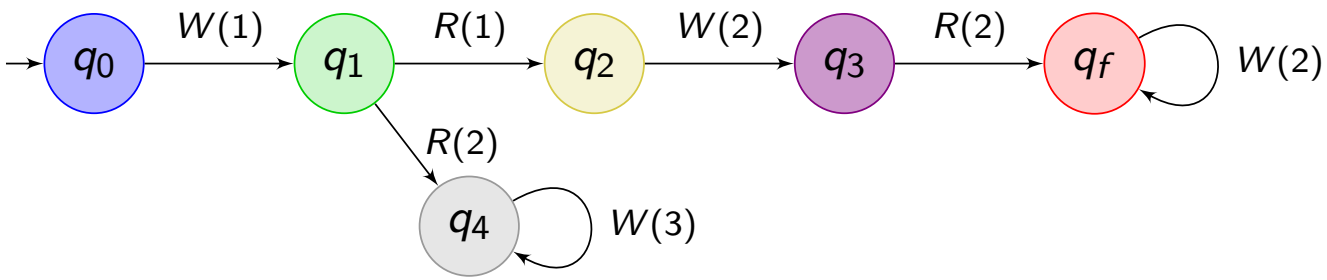
Lack of monotonicity



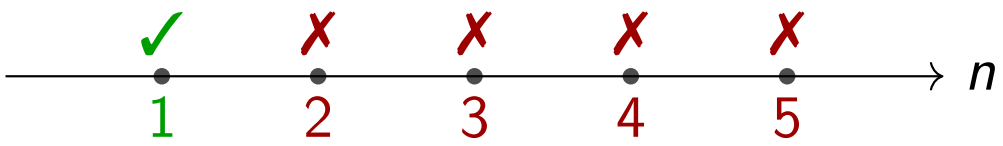
$\Longrightarrow$ **Additional processes can create new deadlocks!**



Lack of monotonicity



$\Longrightarrow$ **Additional processes can create new deadlocks!**



$\Longrightarrow$ **We need new techniques to detect such behaviors.**

Cut-off

Definition: cut-off

An integer $k \in \mathbb{N}$ is a cut-off for almost-sure reachability for $\mathscr{P}$ if one of the following two properties holds:

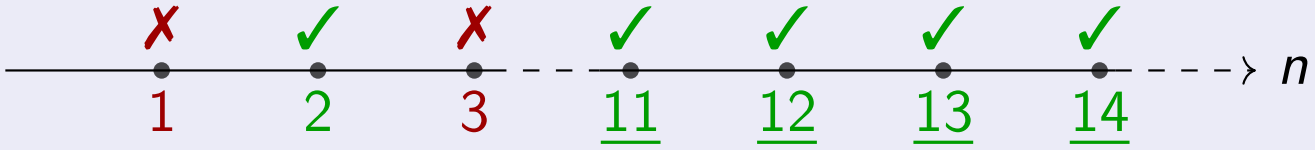
- $\forall n \geq k \ \mathbb{P}_n(\diamond \uparrow q_f) = 1$: Positive cut-off;

Cut-off

Definition: cut-off

An integer $k \in \mathbb{N}$ is a <u>cut-off for almost-sure reachability</u> for $\mathscr{P}$ if one of the following two properties holds:

- $\forall n \geq k \ \mathbb{P}_n(\diamond \uparrow q_f) = 1$: **Positive** cut-off;

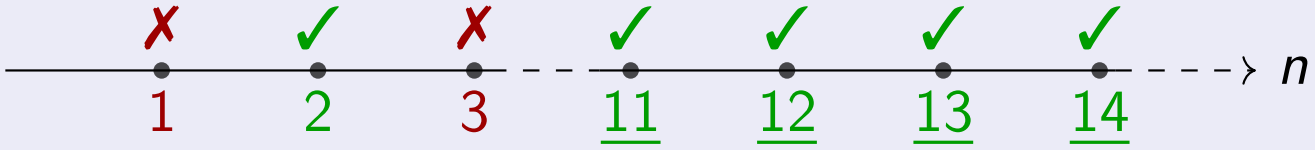


Cut-off

Definition: cut-off

An integer $k \in \mathbb{N}$ is a <u>cut-off for almost-sure reachability</u> for $\mathscr{P}$ if one of the following two properties holds:

- $\forall n \geq k \ \mathbb{P}_n(\diamond \uparrow q_f) = 1$: **Positive** cut-off;



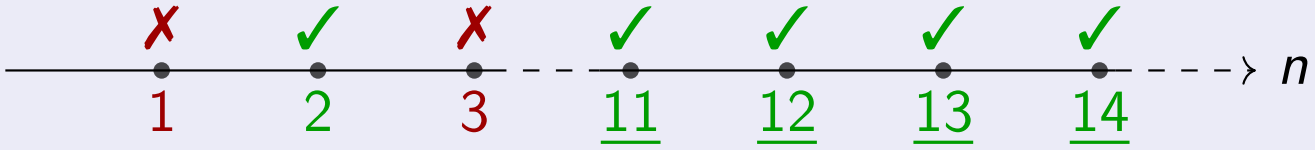
- $\forall n \geq k \ \mathbb{P}_n(\diamond \uparrow q_f) < 1$: **Negative** cut-off;

Cut-off

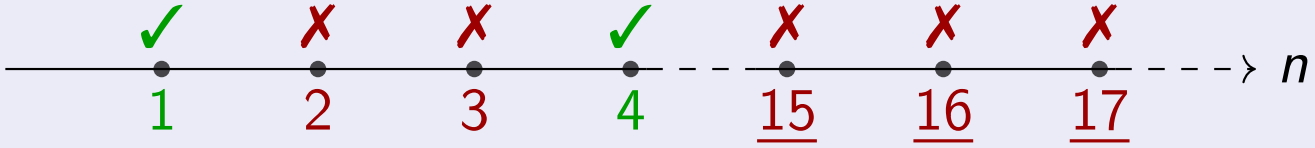
Definition: cut-off

An integer $k \in \mathbb{N}$ is a <u>cut-off for almost-sure reachability</u> for $\mathscr{P}$ if one of the following two properties holds:

- $\forall n \geq k \ \mathbb{P}_n(\diamond \uparrow q_f) = 1$: **Positive** cut-off;



- $\forall n \geq k \ \mathbb{P}_n(\diamond \uparrow q_f) < 1$: **Negative** cut-off;

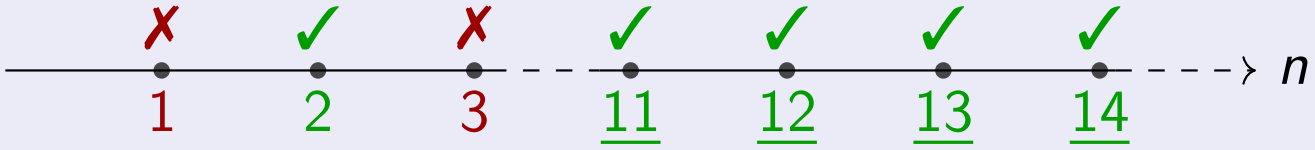


Cut-off

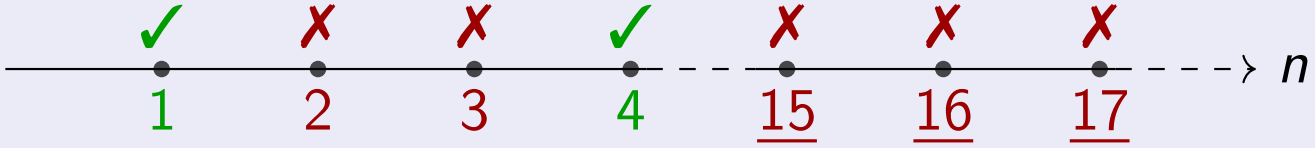
Definition: cut-off

An integer $k \in \mathbb{N}$ is a <u>cut-off for almost-sure reachability</u> for $\mathscr{P}$ if one of the following two properties holds:

- $\forall n \geq k \ \mathbb{P}_n(\diamond \uparrow q_f) = 1$: **Positive** cut-off;



- $\forall n \geq k \ \mathbb{P}_n(\diamond \uparrow q_f) < 1$: **Negative** cut-off;



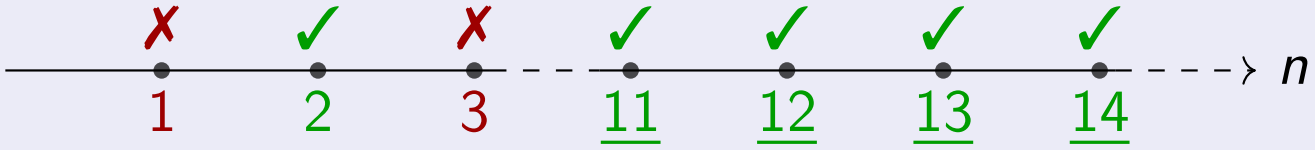
An integer k is a <u>tight</u> cut-off if it is a cut-off and $k-1$ is not.

Cut-off

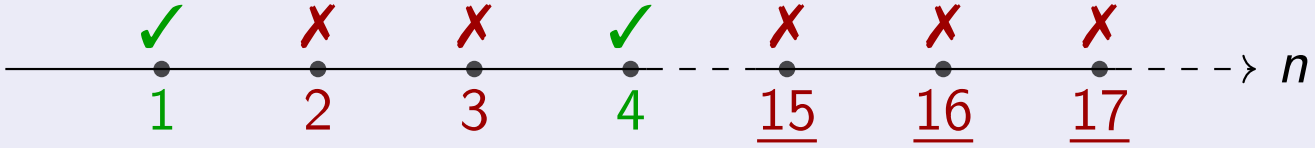
Definition: cut-off

An integer $k \in \mathbb{N}$ is a <u>cut-off for almost-sure reachability</u> for $\mathscr{P}$ if one of the following two properties holds:

- $\forall n \geq k \ \mathbb{P}_n(\diamond \uparrow q_f) = 1$: **Positive** cut-off;



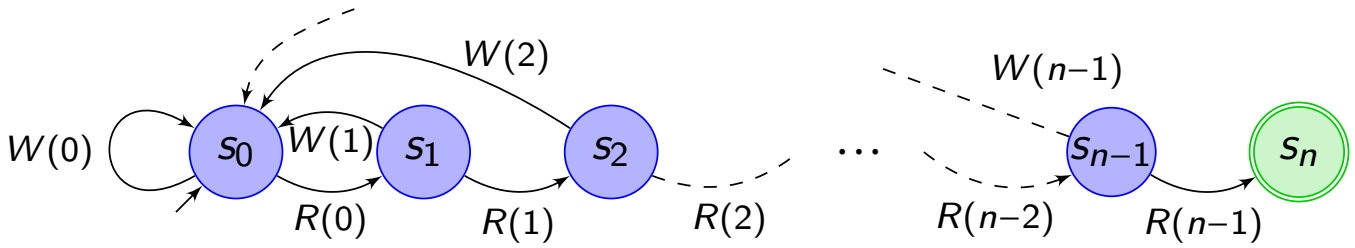
- $\forall n \geq k \ \mathbb{P}_n(\diamond \uparrow q_f) < 1$: **Negative** cut-off;



An integer k is a <u>tight</u> cut-off if it is a cut-off and $k-1$ is not.

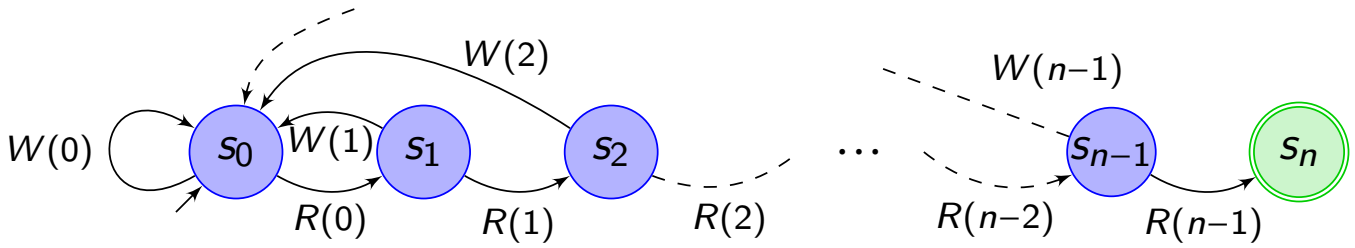
⚠ **Cut-offs need not exist *a priori* from the definition**

Examples



"Filter" protocol $\mathscr{F}_n$ for $n > 0$.

Examples



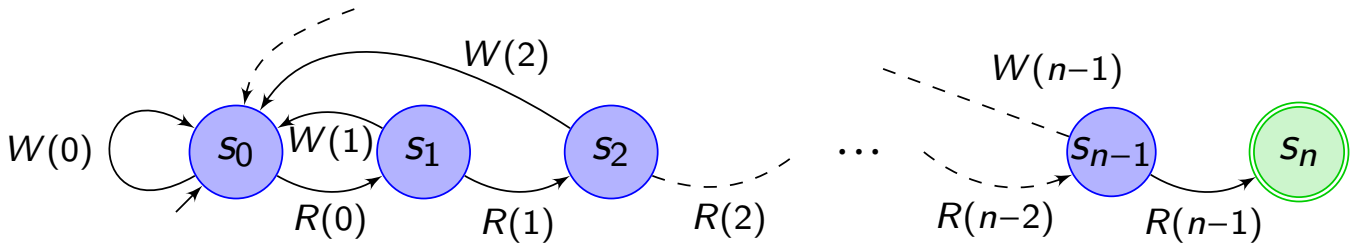
"Filter" protocol $\mathscr{F}_n$ for $n > 0$.

For protocol $\mathscr{F}_n$,

- ▷ networks of size $\geq n$ cover s_n with probability 1,
- ▷ networks of size $< n$ cannot cover s_n.

No deadlock can ever occur as all processes can always go back to the initial state.

Examples



"Filter" protocol $\mathscr{F}_n$ for $n > 0$.

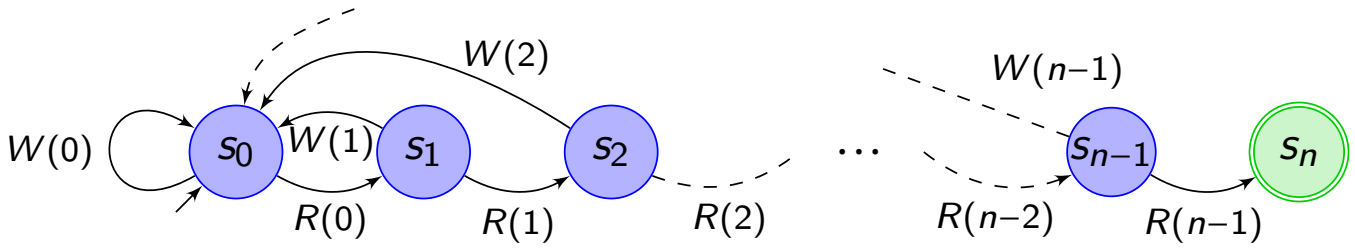
For protocol $\mathscr{F}_n$,

- ▷ networks of size $\geq n$ cover s_n with probability 1,
- ▷ networks of size $< n$ cannot cover s_n.

No deadlock can ever occur as all processes can always go back to the initial state.

$\Longrightarrow$ **Tight positive cut-off equal to n, i.e., linear in the protocol size.**

Examples



"Filter" protocol $\mathscr{F}_n$ for $n > 0$.

For protocol $\mathscr{F}_n$,

- ▷ networks of size $\geq n$ cover s_n with probability 1,
- ▷ networks of size $< n$ cannot cover s_n.

No deadlock can ever occur as all processes can always go back to the initial state.

$\Longrightarrow$ **Tight positive cut-off equal to n, i.e., linear in the protocol size.**

$\Longrightarrow$ Tight negative cut-off exponential.

Existence of a cut-off

Main result

Theorem

For any register protocol $\mathscr{P}$ **there always exists a cut-off for almost-sure reachability**, whose value is at most doubly-exponential in the size of $\mathscr{P}$. Whether it is a positive or a negative cut-off can be decided in EXPSPACE, and is PSPACE-hard.

Existence of a cut-off

Main result

Theorem

For any register protocol $\mathscr{P}$ **there always exists a cut-off for almost-sure reachability**, whose value is at most doubly-exponential in the size of $\mathscr{P}$. Whether it is a positive or a negative cut-off can be decided in EXPSPACE, and is PSPACE-hard.

⚠ **This result strongly relies on the "regularity-breaking" aspect of our stochastic scheduler and on the non-atomicity of read/write operations.**

Existence of a cut-off

Main result

Theorem

For any register protocol $\mathscr{P}$ **there always exists a cut-off for almost-sure reachability**, whose value is at most doubly-exponential in the size of $\mathscr{P}$. Whether it is a positive or a negative cut-off can be decided in EXPSPACE, and is PSPACE-hard.

⚠ **This result strongly relies on the "regularity-breaking" aspect of our stochastic scheduler and on the non-atomicity of read/write operations.**

The non-atomicity guarantees that when a process takes a transition, all processes in the same state can also take the same transition (with a non-zero probability).

$\Longrightarrow$ **Crucial to obtain a copycat lemma.**

Existence: quick sketch

- Write $\mathbf{Pre}^* \subseteq \Gamma$ the set of configurations that <u>can reach</u> q_f;
- Write $\mathbf{Post}^{*\geq n} \subseteq \Gamma$ the set of <u>reachable configurations</u> of size $\geq n$;

Existence: quick sketch

- Write $\mathbf{Pre}^* \subseteq \Gamma$ the set of configurations that <u>can reach</u> q_f;
- Write $\mathbf{Post}^{*\geq n} \subseteq \Gamma$ the set of <u>reachable configurations</u> of size $\geq n$;

Positive cut-off:
$\exists n\ \mathbf{Post}^{*\geq n} \subseteq \mathbf{Pre}^*$

Negative cut-off:
$\forall n\ \mathbf{Post}^{*=n} \not\subseteq \mathbf{Pre}^*$

Existence: quick sketch

- Write $\mathbf{Pre}^* \subseteq \Gamma$ the set of configurations that <u>can reach</u> q_f;
- Write $\mathbf{Post}^{*\geq n} \subseteq \Gamma$ the set of <u>reachable configurations</u> of size $\geq n$;

Positive cut-off:
$\exists n\ \mathbf{Post}^{*\geq n} \subseteq \mathbf{Pre}^*$

Negative cut-off:
$\forall n\ \mathbf{Post}^{*=n} \not\subseteq \mathbf{Pre}^*$

- Consider $\preceq$ point-wise order over configurations, **with state-value support** equality.

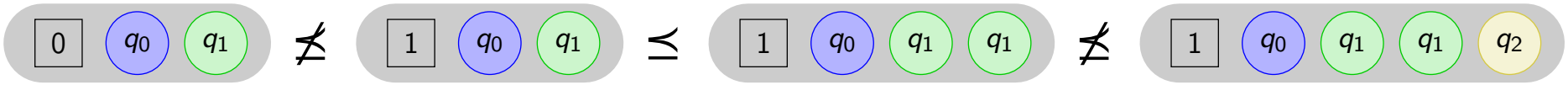
Existence: quick sketch

- Write $\mathbf{Pre}^* \subseteq \Gamma$ the set of configurations that <u>can reach</u> q_f;
- Write $\mathbf{Post}^{*\geq n} \subseteq \Gamma$ the set of <u>reachable configurations</u> of size $\geq n$;

Positive cut-off:
$\exists n\ \mathbf{Post}^{*\geq n} \subseteq \mathbf{Pre}^*$

Negative cut-off:
$\forall n\ \mathbf{Post}^{*=n} \not\subset \mathbf{Pre}^*$

- Consider $\preceq$ point-wise order over configurations, **with state-value support** equality.



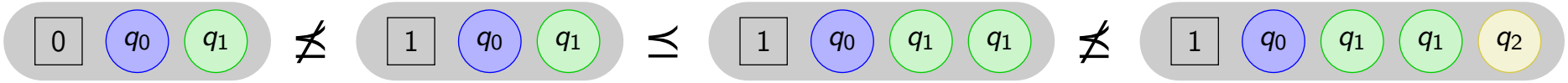
Existence: quick sketch

- Write $\mathbf{Pre}^* \subseteq \Gamma$ the set of configurations that <u>can reach</u> q_f;
- Write $\mathbf{Post}^{*\geq n} \subseteq \Gamma$ the set of <u>reachable configurations</u> of size $\geq n$;

Positive cut-off:
$\exists n\ \mathbf{Post}^{*\geq n} \subseteq \mathbf{Pre}^*$

Negative cut-off:
$\forall n\ \mathbf{Post}^{*=n} \not\subset \mathbf{Pre}^*$

- Consider $\preceq$ point-wise order over configurations, **with state-value support** equality.



- $(\Gamma, \preceq)$ is a well-quasi-ordered set;
- $\mathbf{Pre}^*$ and $\mathbf{Post}^{*\geq n}$ are upward-closed for any n;

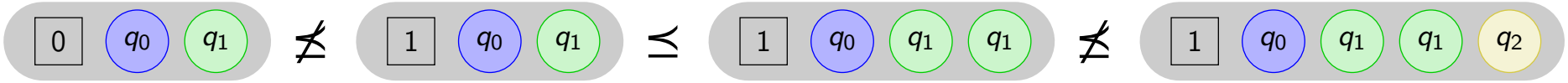
Existence: quick sketch

- Write $\mathbf{Pre}^* \subseteq \Gamma$ the set of configurations that <u>can reach</u> q_f;
- Write $\mathbf{Post}^{*\geq n} \subseteq \Gamma$ the set of <u>reachable configurations</u> of size $\geq n$;

Positive cut-off:
$\exists n\ \mathbf{Post}^{*\geq n} \subseteq \mathbf{Pre}^*$

Negative cut-off:
$\forall n\ \mathbf{Post}^{*=n} \not\subset \mathbf{Pre}^*$

- Consider $\preceq$ point-wise order over configurations, **with state-value support** equality.



- $(\Gamma, \preceq)$ is a well-quasi-ordered set;
- $\mathbf{Pre}^*$ and $\mathbf{Post}^{*\geq n}$ are upward-closed for any n;

. . .

Algorithm

Naive approach: compute **min Post**$^{*\geq 1}$ and **min Pre***.

- Minimal elements **can be large**;

Algorithm

Naive approach: compute **min Post**$^{*\geq 1}$ and **min Pre***.

- Minimal elements **can be large**;
- Using results by Rackoff on the coverability problem in VAS [Rac78, DJLL13], we bound $K = \max\{|\gamma| \mid \gamma \in \mathbf{min\ Pre}^{*}\}$ by a double-exponential in the size of the protocol;
- No similar result for **Post**$^{*\geq 1}$.

Algorithm

Naive approach: compute **min Post**$^{*\geq 1}$ and **min Pre***.

- Minimal elements **can be large**;
- Using results by Rackoff on the coverability problem in VAS [Rac78, DJLL13], we bound $K = \max\{|\gamma| \mid \gamma \in \mathbf{min\ Pre}^*\}$ by a double-exponential in the size of the protocol;
- No similar result for **Post**$^{*\geq 1}$.

Idea: abstract the system up to the first K processes.

- Almost-sure reachability of q_f **is preserved**;
- Underlying graph still of double-exponential size (multiset representation);
- Reachability in NLOGSPACE [Sip97] w.r.t. the graph $\implies$ NEXPSPACE w.r.t. the protocol $\implies$ EXPSPACE by Savitch's theorem [Sip97].

Complexity

Lower bounds

- Best lower bound for positive cut-offs so far: linear (cf. "filter" protocol).

Complexity

Lower bounds

- Best lower bound for positive cut-offs so far: linear (cf. "filter" protocol).
- Best lower bound for negative cut-offs so far: exponential.

$\Longrightarrow$ **Huge gap!**

Complexity

Lower bounds

- Best lower bound for positive cut-offs so far: linear (cf. "filter" protocol).
- Best lower bound for negative cut-offs so far: exponential.

$\Longrightarrow$ **Huge gap!**

- PSPACE-hardness via linear-bounded Turing machine [Sip97]: we build a protocol for which there is a negative cut-off iff the machine reaches its final state q_{halt}.

Shares idea with the exponential counter

Summary

Our model:

- register protocols,
- non-atomic read/write operations,
- fairness via stochastic scheduler.

Summary

Our model:

- register protocols,
- non-atomic read/write operations,
- fairness via stochastic scheduler.

Some differences with classical models:

- lack of monotonicity in general,
- complexity (PSPACE-hardness while many problems are polynomial or in NP/coNP),
- cut-offs may be exponential (most models admit polynomial cut-offs).

$\Longrightarrow$ **Slight changes in the setting induce important changes in complexity.**

Future work

Extensions still monotonous:

- Leader process;
- More registers;
- Process identifiers;

Future work

Extensions still monotonous:

- Leader process;
- More registers;
- Process identifiers;

Many open questions:

- Closing the gaps (complexity, cut-off bounds),
- Atomic read/write operations,
- Pushdown Systems ?
- Other objectives (e.g., liveness, limit-sure),
- Quantitative questions,
- Synthesis of local strategies.

Future work

Extensions still monotonous:

- Leader process;
- More registers;
- Process identifiers;

Many open questions:

- Closing the gaps (complexity, cut-off bounds),
- Atomic read/write operations,
- Pushdown Systems ?
- Other objectives (e.g., liveness, limit-sure),
- Quantitative questions,
- Synthesis of local strategies.

Many thanks! Any question?

References I

[ABG15] C. Aiswarya, Benedikt Bollig, and Paul Gastin. An automata-theoretic approach to the verification of distributed algorithms. In Luca Aceto and David de Frutos-Escrig, editors, Proceedings of the 26th International Conference on Concurrency Theory (CONCUR'15), volume 42 of Leibniz International Proceedings in Informatics, pages 340–353. Leibniz-Zentrum für Informatik, September 2015.

[DJLL13] Stéphane Demri, Marcin Jurdziński, Oded Lachish, and Ranko Lazić. The covering and boundedness problems for branching vector addition systems. Journal of Computer and System Sciences, 79(1):23–38, February 2013.

[DSZ10] Giorgio Delzanno, Arnaud Sangnier, and Gianluigi Zavattaro. Parameterized verification of ad hoc networks. In Paul Gastin and François Laroussinie, editors, Proceedings of the 21st International Conference on Concurrency Theory (CONCUR'10), volume 6269 of Lecture Notes in Computer Science, pages 313–327. Springer-Verlag, September 2010.

[EFM99] Javier Esparza, Alain Finkel, and Richard Mayr. On the verification of broadcast protocols. In Proceedings of the 14th Annual Symposium on Logic in Computer Science (LICS'99), pages 352–359. IEEE Comp. Soc. Press, July 1999.

[EGM13] Javier Esparza, Pierre Ganty, and Rupak Majumdar. Parameterized verification of asynchronous shared-memory systems. In Natasha Sharygina and Helmut Veith, editors, Proceedings of the 25th International Conference on Computer Aided Verification (CAV'13), volume 8044 of Lecture Notes in Computer Science, pages 124–140. Springer-Verlag, July 2013.

[GS92] Steven M. German and A. Prasad Sistla. Reasoning about systems with many processes. Journal of the ACM, 39(3):675–735, July 1992.

[Rac78] Charles Rackoff. The covering and boundedness problems for vector addition systems. Theoretical Computer Science, 6:223–231, 1978.

[Sip97] Michael Sipser. Introduction to the theory of computation. PWS Publishing Company, 1997.