Reachability in Networks of Register Protocols under Stochastic Schedulers

Patricia Bouyer\textsuperscript{1}  Nicolas Markey\textsuperscript{1}  Mickael Randour\textsuperscript{2}  
Arnaud Sangnier\textsuperscript{3}  Daniel Stan\textsuperscript{1}

\textsuperscript{1}LSV - CNRS & ENS Cachan, France  
\textsuperscript{2}ULB, Belgium  
\textsuperscript{3}IRIF - CNRS & Université Paris Diderot, France

April 07, 2016 - IRISA - INRIA Rennes - 68NQRT seminar
1. Networks of register protocols
2. Almost-sure reachability
3. Cut-offs: existence and decision algorithm
4. Conclusion
The talk in one slide

Networks of *arbitrarily many* identical processes:

- processes = non-deterministic automata,
- communication via a *shared register* (read and write),
- *fair* (stochastic) scheduler.

**Question:**

Is it the case that *almost-surely* one of the processes reaches a final state for a network of $N$ processes?

▷ Existence of a *cut-off property* (constant answer for large $N$).
▷ EXPSPACE algorithm based on a *symbolic graph*.
▷ *Cut-offs can be exponential.*
The talk in one slide... OK, two 😊

Goal of this talk:
- highlight the particularities of our model and their impact,
- understand typical examples,
- sketch the cornerstones of our solution.

Full paper available on arXiv [BMR+16]: abs/1602.05928
1. Networks of register protocols

2. Almost-sure reachability

3. Cut-offs: existence and decision algorithm

4. Conclusion
Context: distributed systems

Goal

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

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

⇒ Instead of fixing a bound on the number of components, we use *parameterized verification*. 
Parameterized verification

Take the number of components as a parameter and identify an infinite set of parameter values for which the system is correct, if such a set exists.

E.g., all networks of $\geq N$ components satisfy a given property.

Advantages:

- general approach covering all parameter values,
- can be more efficient than checking the system for very large values as it involves orthogonal techniques (e.g., reducing the size of the network using structural arguments).
Parameterized networks

Every process follow the same protocol (usually, a finite-state automaton).

Different means of communication $\implies$ different models.

E.g.,

- Rendez-vous communication [GS92],
- broadcast communication [EFM99, DSZ10],
- token-passing [CTTV04, AJKR14],
- message passing [BGS14],
- shared register or memory [ABG15, EGM13].

$\implies$ Minor changes in the setting can drastically change the complexity of verification problems.

See Esparza’s survey in STACS’14 [Esp14].
Our model in a nutshell

Processes

- **Protocol**: non-deterministic finite-state automaton.
- **Communication**: non-atomic read and write operations on a shared register (see [Hag11, EGM13, DEGM15]).

Some known results:

- Deciding if one process can reach a control state takes polynomial time (adapting [DSTZ12]).
- With a leader implementing a different protocol, NP-complete problem [EGM13].

Scheduler’s role

In many works, the scheduler actually helps in reaching the target state: i.e., the question is whether there exists a scheduling such that a process reaches the target.
Our model in a nutshell

Scheduler

⇒ Here, we want to get rid of this strong assumption.

⇒ Introduction of a fair scheduler.

Two flavors of fairness:

1. Temporal logic property on executions (e.g., every action available infinitely often is performed infinitely often) (e.g., [GS92, AJK16]).

2. Stochastic scheduler (w.l.o.g. uniform distribution).

⇒ The stochastic scheduler breaks regular patterns (e.g., round-robin) and considers all possible interleaving with probability one in the long run.

⇒ Important property for our approach.
Related work

In [BFS14], Bertrand et al. study networks with

- stochastic protocols,
- communication via broadcast,
- an “helping scheduler”.

One studied question is the existence of a network size and a scheduler granting almost-sure reachability of a control state: it turns out to be a coNP-complete problem.

⇒ Despite apparent similarities, the models are difficult to compare: different use of probabilities, different communication mechanism, different role of the scheduler.
Our protocols

Definition

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

Definition: register protocol

\[ \mathcal{P} = \langle Q, D, q_0, T \rangle \]

- $Q$ finite set of control locations;
- $D$ finite alphabet of data for the shared register;
- $q_0 \in Q$ initial location;
- $T \subseteq Q \times \{R, W\} \times D \times Q$ set of transitions of the protocol.

No deadlock and if $R$ then all values in $D$ can be read (omitted = self-loops).
Our protocols

Example

Imagine that our network contains a single process.

⇒ A single process cannot reach $q_f$. 
Our networks

Sketch

We study **distributed systems**:

- asynchronous composition of $k$ copies of the protocol,
- non-determinism (inside the protocol and choice of process) resolved by a stochastic scheduler (uniform).

\[ \Rightarrow \text{Markov chain over the set of configurations } \Gamma = \mathbb{N}^Q \times D \]

(multiset + data), finite if $k$ is fixed.

\[ \Rightarrow \text{No creation/deletion of processes.} \]

Notations:

- $\mathcal{S}_P$ distributed system,
- $\mathcal{S}_P^k$ distributed system of size $k$,
- $\gamma_0 \rightarrow \gamma_1 \ldots \rightarrow \gamma_n$ sequence of configurations, also $\gamma_0 \rightarrow^{*} \gamma_n$
<table>
<thead>
<tr>
<th></th>
<th>Networks of register protocols</th>
<th>Almost-sure reachability</th>
<th>Cut-offs</th>
<th>Conclusion</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Networks of register protocols</td>
<td>Almost-sure reachability</td>
<td>Cut-offs</td>
<td>Conclusion</td>
</tr>
<tr>
<td>2</td>
<td>Almost-sure reachability</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>Cut-offs: existence and decision algorithm</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>Conclusion</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Almost-sure reachability

For $q_f \in Q$:

- $[q_f] = \{\gamma \in \mathcal{G} \mid st(\gamma)(q_f) > 0\}$: configurations covering $q_f$, i.e., $\gamma$ s.t. $st(\gamma)(q_f) > 0$.

- $[\Diamond q_f] = \{\gamma_0 \rightarrow^* \gamma_n \mid \exists i \in [0; n], st(\gamma_i)(q_f) > 0\}$: paths covering $q_f$.

$\implies$ Paths covering $q_f$.

$\mathbb{P}(\gamma, [\Diamond q_f]) = \text{probability to cover } q_f \text{ starting in } \gamma$.

$\implies$ We seek cut-off properties for almost-sure reachability.
Cut-off

Definition: cut-off

An integer $k \in \mathbb{N}$ is a cut-off for almost-sure reachability for $P$, $d_0$ and $q_f$ if one of the following two properties holds:

- for all $h \geq k$, we have $P(\langle q_0^h, d_0 \rangle, \lbrack \Diamond q_f \rbrack) = 1$. In this case $k$ is a positive cut-off;
- for all $h \geq k$, we have $P(\langle q_0^h, d_0 \rangle, \lbrack \Diamond q_f \rbrack) < 1$. Then $k$ is a negative cut-off.

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

⚠️ Cut-offs need not exist from the definition and

$\nexists \ positive \ \iff \ \exists \ negative.$

↩️ We will prove that they always exist!
Back to the example

Network for two processes (self-loops omitted).

⇒ From here, the process in $q_0$ is trapped hence the other one is alone and will never reach $q_f$.

⇒ From here, non-exhaustive construction.

⇒ With $\geq 2$ processes, $q_f$ reached with probability $> 0$ but $< 1$!

⇒ $k = 1$ is a negative cut-off.
Other examples

Positive cut-off

“For Filter” protocol $\mathcal{F}_n$ for $n > 0$.

For protocol $\mathcal{F}_n$,

- 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.

$\implies$ Tight positive cut-off equal to $n$, i.e., linear in the protocol size.
Other examples
Lack of monotonicity for small network sizes

Observation
When considering an “helping scheduler” as in many models, increasing the network size is never a bad thing (as the scheduler can decide not to activate the additional processes at all).

⇒ Not true anymore with our fair scheduler!

⇒ Additional processes can create new deadlocks!

⇒ We need new techniques to detect such behaviors.
1. Networks of register protocols
2. Almost-sure reachability
3. Cut-offs: existence and decision algorithm
4. Conclusion
Existence of a cut-off

Main result

Theorem

For any register protocol \( \mathcal{P} \), any initial register value \( d_0 \) and any target location \( q_f \), there always exists a cut-off for almost-sure reachability, whose value is at most doubly-exponential in the size of \( \mathcal{P} \). 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 transition can also take the same transition (with a non-zero probability).

⇒ Crucial to obtain a copycat lemma.
Existence of a cut-off

Atomic read/write $\sim$ no cut-off

$\implies$  State $q_f$ is reached with probability one if and only if the network size is odd.
Existence of a cut-off

Proof sketch (1/3)

1. Partial order \( \preceq \) over configurations s.t. \( \langle \mu, d \rangle \preceq \langle \mu', d' \rangle \) iff 
\[ d = d', \text{ the multisets have the same support and } \mu \subseteq \mu'. \]
\[ \implies \langle \Gamma, \preceq \rangle \text{ is a wqo}. \]

2. For \( k > 0 \),
\[ \mathbb{P}(\langle q_0^k, d_0 \rangle, \lbrack \Diamond q_f \rbrack) = 1 \iff \text{Post}^*(\{\langle q_0^k, d_0 \rangle\}) \subseteq \text{Pre}^*(\lbrack q_f \rbrack). \]
\[ \implies \text{Cut-off } k_0 \text{ if for all } k \geq k_0, \text{ either the inclusion is always true or it is always false}. \]

3. Copycat lemma: if \( \gamma_1 \rightarrow^* \gamma_2 \) and \( \gamma_2 \preceq \gamma'_2 \), then there exists \( \gamma'_1 \) such that \( \gamma'_1 \rightarrow^* \gamma'_2 \) and \( \gamma_1 \preceq \gamma'_1 \).
\[ \implies \text{Monotonicity property}. \]

4. \( \text{Post}^*(\uparrow\{\langle q_0, d_0 \rangle\}) \) and \( \text{Pre}^*(\lbrack q_f \rbrack) \) are upward-closed sets.
\[ \implies \text{Can be represented by minimal elements}. \]
Existence of a cut-off

Proof sketch (2/3)

5. \( \text{Post}^*(\uparrow \{ \langle q_0, d_0 \rangle \}) = \uparrow \{ \theta_1, \ldots, \theta_n \} \) and \( \text{Pre}^*(\lceil q_f \rceil) = \uparrow \{ \eta_1, \ldots, \eta_m \} \).

6. Is \( \text{Post}^*(\uparrow \{ \langle q_0, d_0 \rangle \}) \) included to \( \text{Pre}^*(\lceil q_f \rceil) \) modulo single-state incrementation?

\[ \Rightarrow \text{A bit technical...} \]

\[ \ldots \text{intuitively, the goal is to check if elements of } \text{Post}^*(\uparrow \{ \langle q_0, d_0 \rangle \}) \text{ can enter } \text{Pre}^*(\lceil q_f \rceil) \text{ by adding sufficiently many processes in a given state.} \]
Existence of a cut-off

Proof sketch (3/3)

7 If \textbf{No}, then there is a \textbf{negative cut-off}.
   \[\implies\text{For each } k \text{ sufficiently large, we can build a configuration that is in } Post^*\left(\{\langle q_0^k, d_0 \rangle\}\right) \text{ but not in } Pre^*\left([q_f]\right) \implies P(\langle q_0^k, d_0 \rangle, [\Diamond q_f]) < 1.\]

8 If \textbf{Yes}, then there is a \textbf{positive cut-off}.
   \[\implies\text{For } k \text{ sufficiently large, every configuration in } Post^*\left(\{\langle q_0^k, d_0 \rangle\}\right) \text{ is also in } Pre^*\left([q_f]\right) \implies P(\langle q_0^k, d_0 \rangle, [\Diamond q_f]) = 1.\]

\[\implies\text{There is always a cut-off!}\]

\[\implies\text{Value of the cut-off at most polynomial in the size of the minimal elements...}\]
Deciding the nature of the cut-off

Goal
Decide if the system admits a *negative* cut-off. If not, then there is a *positive* one.

Idea
Abstract *arbitrarily large* systems by a *symbolic graph* of bounded size and study this graph to conclude.

⇒ The crux is to maintain enough information!
Symbolic graph

Traditional approach: using only supports (1/2)

**Fully symbolic graph:**

- We totally abstract the number of processes in each state by keeping only *supports* of configurations.
- Sufficient abstraction in simpler models.

**Hope (soon to be crushed)**

State $q_f$ is almost-surely covered *if and only if* supports containing $q_f$ are reachable from all reachable states in the symbolic graph.
Symbolic graph
Traditional approach: using only supports (2/2)

What can we conclude from the symbolic graph?

$q_f$ is reachable from everywhere, so positive cut-off?

No! We saw that $k = 1$ is a negative cut-off!
Symbolic graph

Extending this approach

Is this graph useless?

⇒ No! One direction of the equivalence holds.

Observation

If the symbolic graph contains a deadlock (i.e., a reachable state from which $q_f$ is not reachable), then there is a negative cut-off.

This holds because from any run in the symbolic graph, one can build a mimicking one in the real system given a sufficient number of processes.

⇒ To obtain the other direction, we need to add information in the symbolic graph.

⇒ We introduce a concrete part to track precisely the behavior of a bounded number of processes.
Symbolic graph

Adding a concrete part

Definition: symbolic graph of index $k$

\[ G = \langle V, v_0, E \rangle \text{ where} \]

- \[ V = \mathbb{N}^Q_k \times 2^Q \times D: \text{concrete part keeping track of a fixed set of } k \text{ processes, abstract part encoding the arbitrarily many remaining processes, data;} \]
- \[ v_0 = \langle q_0^k, \{q_0\}, \{d_0\}\rangle; \]
- \[ \langle \mu, S, d \rangle \rightarrow \langle \mu', S', d' \rangle \text{ for each } (q, O, d'', q') \in T \text{ such that } d = d' = d'' \text{ if } O = R \text{ and } d' = d'' \text{ if } O = W, \text{ and one of the following two conditions holds:} \]
  - either \( S' = S \) and \( q \sqsubseteq \mu \) and \( \mu' = \mu \oplus q \oplus q' \);
  - or \( \mu = \mu' \) and \( q \in S \) and \( S' \in \{S \setminus \{q\} \cup \{q'\}, S \cup \{q'\}\} \).

\[ \rightarrow \text{ Transitions either impact the concrete part or the symbolic part, not both (i.e., no exchange of processes).} \]
Symbolic graph
Toward a correct and complete algorithm

Recall that $\text{Pre}^*([q_f]) = \{\eta_i | 1 \leq i \leq m\}$. We show that the symbolic graph abstraction is complete for $k = K \cdot |Q|$, where $K = \max\{st(\eta_i)(q) | q \in Q, 1 \leq i \leq m\}$.

$\implies$ Intuitively, the concrete part must be large enough to capture executions involving minimal elements of $\text{Pre}^*([q_f])$.

**Theorem**

There is a negative cut-off for $\mathcal{P}$, $d_0$ and $q_f$ if, and only if, there is a node in the symbolic graph of index $K \cdot |Q|$ that is reachable from $\langle q_0^{K \cdot |Q|}, \{q_0\}, d_0 \rangle$ but from which no configuration involving $q_f$ is reachable.
Complexity (1/2)

Upper bounds

- Using results by Rackoff on the coverability problem in VAS [Rac78, DJLL13], we bound $K$ (hence the size of the graph since we use multisets and not vectors) by a double-exponential in the size of the protocol.

- Reachability in NLOGSPACE [Sip97] w.r.t. the graph $\Rightarrow$ NEXPSPACE w.r.t. the protocol $\Rightarrow$ EXPSPACE by Savitch’s theorem [Sip97].

- Doubly-exponential upper bounds on cut-off values.
Complexity (2/2)

Lower bounds

- **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 $q_{\text{halt}}$.

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

  $\Rightarrow$ Huge gap!

- Best **lower bound for negative cut-offs so far**: exponential.

  $\Rightarrow$ Shares ideas with PSPACE-hardness proof. Let’s discuss it now.
Exponential negative cut-off

Different parts: simulating a counter over $n$ bits, producing tokens needed for the simulation, filter protocol, $d_0 = \#$, target $q_f$. 

Reachability in Networks of Register Protocols... Bouyer, Markey, Randour, Sangnier, Stan 34 / 37
Exponential negative cut-off

Claim: \( \exists N > 2^n \text{ s.t. } \mathbb{P}(\langle \text{init}^N, \# \rangle, [\lozenge q_f]) < 1 \) while \( \mathbb{P}(\langle \text{init}^{2^n}, \# \rangle, [\lozenge q_f]) = 1 \).

\( \implies \) Exponential tight negative cut-off.
Exponential negative cut-off

Three phases: initialization, simulation, counting.
**Exponential negative cut-off**

**Phase 1: initialization.** Processes move to $a_i$ and $tok$ until some process in $tok$ writes 1 in the register (or until someone reaches $q_f$ by reading $\#$ from $a_i$).
**Exponential negative cut-off**

Phase 2: simulation. If all the processes are in tok, they will eventually reach $q_f$. So we assume that there is at least one process in a state $a_i$. 
Exponential negative cut-off

If some $a_i$ is empty, then $d_n$ cannot be reached and we cannot enter the counting phase $\implies$ some process will eventually reach $q_f$. 
Exponential negative cut-off

Thus, assume there is at least one process in each state $a_i$. We can prove that $d_i$ is reachable when at the start of the simulation phase, at least $2^i$ processes are in tok (we need to produce an exponential number of tokens).
Exponential negative cut-off

Reaching \( s_0 \) thus requires \( 2^n \) processes in \( tok \). If we want to avoid reaching \( q_f \), the counting phase must never contain more than \( n \) processes (because we have an \((n + 1)\) filter). So we assume each \( a_i \) has exactly one process at the start of the simulation.
Exponential negative cut-off

To avoid reaching $q_f$, we need $n$ processes in states $a_i$ and at least $2^n$ processes in $tok$.

$\implies q_f$ is almost-surely reached in systems with strictly less than $n + 2^n$ processes.
Exponential negative cut-off

It remains to show that for \( N \geq n + 2^n \), \( q_f \) cannot be reached almost-surely.

\[ \implies \text{Exhibit a finite execution having no continuation reaching } q_f. \]
Exponential negative cut-off

**Execution:** during initialization, put one process in each $a_i$ and all others in $tok$. One of them writes $1$. 
Exponential negative cut-off

The $n$ processes in states $a_i$ then simulate the incrementations of the counter, consuming tokens at each step, until reaching $d_n$. 
Exponential negative cut-off

All processes in tok move to sent and the process in $d_n$ writes halt and moves to $s_0$. Other processes in the simulation phase move to $s_0$ and processes in sent move to sink.
Exponential negative cut-off

We are left with $n$ processes in $s_0$ and all the others in $sink$. Since we have an $(n + 1)$ filter, $q_f$ cannot be reached.

$$\implies \Pr(\langle \text{init}^N, \# \rangle, [\Diamond q_f]) < 1 \text{ for } N = n + 2^n.$$
**Exponential negative cut-off**

We have proved a tight negative cut-off of exponential size.
1. Networks of register protocols
2. Almost-sure reachability
3. Cut-offs: existence and decision algorithm
4. Conclusion
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).

⇒ Slight changes in the setting induce important changes in complexity.
Future work

Many open questions:

- closing the gaps (complexity, cut-off bounds),
- other objectives (e.g., liveness),
- quantitative questions,
- atomic read/write operations,
- synthesis of local strategies.

Many thanks! Any question?
References I

C. Aiswarya, Benedikt Bollig, and Paul Gastin.
An automata-theoretic approach to the verification of distributed algorithms.

Simon Außerlechner, Swen Jacobs, and Ayrat Khalimov.
Tight cutoffs for guarded protocols with fairness.

Benjamin Aminof, Swen Jacobs, Ayrat Khalimov, and Sasha Rubin.
Parametrized model checking of token-passing systems.

Nathalie Bertrand, Paulin Fournier, and Arnaud Sangnier.
Playing with probabilities in reconfigurable broadcast networks.

Benedikt Bollig, Paul Gastin, and Len Schubert.
Parameterized verification of communicating automata under context bounds.
Patricia Bouyer, Nicolas Markey, Mickael Randour, Arnaud Sangnier, and Daniel Stan.
Reachability in networks of register protocols under stochastic schedulers.

Edmund M. Clarke, Muralidhar Talupur, Tayssir Touili, and Helmut Veith.
Verification by network decomposition.

Antoine Durand-Gasselin, Javier Esparza, Pierre Ganty, and Rupak Majumdar.
Model checking parameterized asynchronous shared-memory systems.

Stéphane Demri, Marcin Jurdziński, Oded Lachish, and Ranko Lazić.
The covering and boundedness problems for branching vector addition systems.

Giorgio Delzanno, Arnaud Sangnier, Riccardo Traverso, and Gianluigi Zavattaro.
On the complexity of parameterized reachability in reconfigurable broadcast networks.
Giorgio Delzanno, Arnaud Sangnier, and Gianluigi Zavattaro.
Parameterized verification of ad hoc networks.

Javier Esparza, Alain Finkel, and Richard Mayr.
On the verification of broadcast protocols.

Javier Esparza, Pierre Ganty, and Rupak Majumdar.
Parameterized verification of asynchronous shared-memory systems.

Javier Esparza.
Keeping a crowd safe: On the complexity of parameterized verification (invited talk).

Steven M. German and A. Prasad Sistla.
Reasoning about systems with many processes.
References IV

Matthew Hague.
Parameterised pushdown systems with non-atomic writes.

Charles Rackoff.
The covering and boundedness problems for vector addition systems.

Michael Sipser.
*Introduction to the theory of computation.*