Black-Box Testing Liveness Properties of Partially Observable Stochastic Systems

Javier Esparza
Technische Universiät München, Germany

Vincent P. Grande
RWTH Aachen University, Germany

Abstract

We study black-box testing for stochastic systems and arbitrary ω-regular specifications, explicitly including liveness properties. We are given a finite-state probabilistic system that we can only execute from the initial state. We have no information on the number of reachable states, or on the probabilities; further, we can only partially observe the states. The only action we can take is to restart the system. We design restart strategies guaranteeing that, if the specification is violated with non-zero probability, then w.p.1 the number of restarts is finite, and the infinite run executed after the last restart violates the specification. This improves on previous work that required full observability. We obtain asymptotically optimal upper bounds on the expected number of steps until the last restart. We conduct experiments on a number of benchmarks, and show that our strategies allow one to find violations in Markov chains much larger than the ones considered in previous work.

2012 ACM Subject Classification Theory of computation → Verification by model checking

Keywords and phrases Partially observable Markov chains, ω-regular properties, black-box testing

Digital Object Identifier 10.4230/LIPIcs.ICALP.2023.126

Category Track B: Automata, Logic, Semantics, and Theory of Programming

Related Version *Full Version*: `https://arxiv.org/abs/2303.03292`

Supplementary Material *Software (Source Code)*:
`https://git.rwth-aachen.de/netsci/restarting-markov-chains-experiments/`

Funding *Javier Esparza*: Partially funded by the DFG within Research Training Group 2428 CONVEY.
Vincent P. Grande: Funded by the DFG (German Research Foundation) under Research Training Group 2236/2 UnRAVeL.

Acknowledgements The authors thank the anonymous reviewers for helpful feedback that improved the paper.

1 Introduction

Black-box testing is a fundamental analysis technique when the user does not have access to the design or the internal structure of a system [12, 15]. Since it only examines one run of the system at a time, it is computationally cheap, which makes it often the only applicable method for large systems.

We study the black-box testing problem for finite-state probabilistic systems and ω-regular specifications: Given an ω-regular specification, the problem consists of finding a run of the program that violates the property, assuming that such runs have nonzero probability.

Let us describe our assumptions in more detail. We do not have access to the code of the system or its internal structure, and we do not know any upper bound on the size of its state space. We can repeatedly execute the system, restarting it at any time. W.l.o.g. we assume that all runs of the system are infinite. We do not assume full observability of the states of the system, only that we can observe whether the atomic propositions of the property are currently true or false. For example, if the property states that a system variable, say x, should have a positive value infinitely often, then we only assume that at each state we can observe the sign of x; letting Σ denote the set of possible observations, we have $\Sigma = \{+, -\}$, standing for a positive and a zero or negative value, respectively (in the rest of the introduction we shorten "zero or negative" to "negative"). Every system execution induces an observation, that is, an element of Σ^ω. The violations of the property are the ω-words $V \subseteq \Sigma^\omega$ containing only finitely many occurrences of $+$.

Our goal is to find a *strategy* that decides after each step whether to abort the current run and restart the system, or continue the execution of the current run. The strategy must ensure that some run that violates the property, that is, a run whose observation belongs to V, is eventually executed. The strategy decides depending on the observations made so far. Formally, given Σ and the set of actions $A = \{\mathsf{r}, \mathsf{c}\}$ (for "restart" and "continue") a strategy for V is a mapping from $(\Sigma \times A)^*\Sigma$, the sequence of observations and actions executed so far, to A, the next decision. Our goal is to find a strategy σ satisfying the following property:

> For every finite-state program P over Σ, if $V \subseteq \Sigma^\omega$ has positive probability and the runs of P are restarted according to σ, then w.p.1 the number of restarts is finite, and the observation of the run executed after the last restart belongs to V.

Observe that it is not clear that such strategies exist. They are easy to find for safety properties, where the fact that a run violates the property is witnessed by a *finite* prefix[1], but for liveness properties there is no such prefix in general. We show that these strategies exist for every ω-regular language V. Moreover, the strategies only need to maintain a number of counters that depends only on V, and not on the program. So in order to restart P according to σ one only needs logarithmic memory in the length of the current sequence.

► **Example 1.** To give a first idea of why these strategies also exist for liveness properties, consider the property over $\Sigma = \{+, -\}$ stating that a variable x should have a positive value only finitely often. The runs violating the property are those that visit $+$-states infinitely often. Our results show that the following strategy works in detecting a run violating the property (among others):

> After the n-th restart, repeatedly execute blocks of $2n$ steps. If at some point after executing the first block *the second half* of the concatenation of the blocks executed so far contains only negative states, then restart.

For example, assume there have been 4 restarts. Then the strategy repeatedly executes blocks of 8 steps. If after executing $1, 2, 3, \ldots$ of these blocks the last $4, 8, 16, \ldots$ states are negative, then the strategy restarts for the 5th time. If that is never the case, then there are only 4 restarts. Figure 1 shows a family of Markov chains for which naive strategies do not work, but the above strategy does: almost surely the number of restarts is finite and the run after the last restart visits the rightmost state infinitely often. Observe that for every $n \geq 0$ the family exhibits executions that visit $+$ states at least n times, and executions that visit a $+$ state at most once every n steps.

[1] One can choose for σ the strategy "after the n-th reset, execute n steps; if this finite execution is not a witness, restart, otherwise continue forever." Indeed, if the shortest witness has length k, then for every $n \geq k$, after the n-th restart the strategy executes a witness with positive probability, and so it eventually executes one w.p.1.

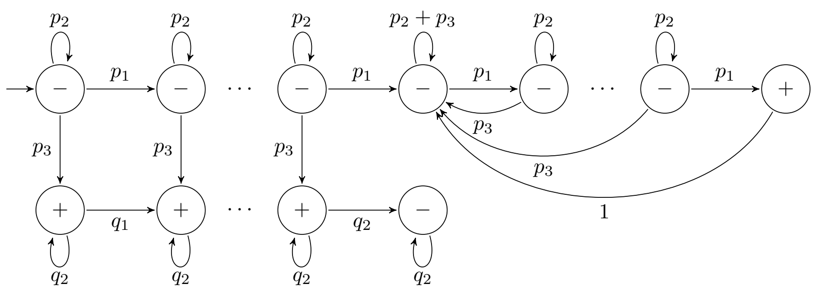

Figure 1 A family of partially observable Markov chains.

We also obtain asymptotically optimal upper bounds on the expected time until the last restart, that is, on the time until the execution of the run violating the property starts. The bounds depend on two parameters of the Markov chain associated to the program, called the *progress radius* and the *progress probability*. An important part of our contribution is the identification of these parameters as the key ones to analyze.

While our results are stated in an abstract setting, they easily translate into practice. In a practical scenario, on top of the values of the atomic propositions, we can also observe useful debugging information, like the values of some variables. We let a computer execute runs of the system for some fixed time t according to the strategy σ. If at time t we observe that the last restart took place a long time ago, then we stop testing and return the run executed since the last restart as candidate for a violation of the property. In the experimental section of our paper we use this scenario to detect errors in *population protocols*, a model of distributed computation, whose state space is too large to find them by other means.

Related work. There is a wealth of literature on black-box testing and black-box checking [12, 15], but the underlying models are not probabilistic and the methods require to know an upper bound on the number of states. Work on probabilistic model-checking assumes that (a model of) the system is known [2]. There are also works on black-box verification of probabilistic systems using statistical model checking of statistical hypothesis testing [22, 17, 18, 20, 21] (see also [11, 13] for surveys on statistical model checking). They consider a different problem: we focus on producing a counterexample run, while the goal of black-box verification is to accept or reject a hypothesis on the probability of the runs that satisfy a property. Our work is also related to the *runtime enforcement problem* [16, 3, 14, 7, 8], which also focus on identifying violations of a property. However, in these works either the setting is not probabilistic, or only a subset of the ω-regular properties close to saftey properties is considered. Finally, the paper closest to ours is [6], which considers the same problem, but for fully observable systems. In particular, in the worst case the strategies introduced in [6] require to store the full sequence of states visited along a run, and so they use linear memory in the length of the current sequence, instead of logarithmic memory, as is the case for our strategy.

Structure of the paper. The paper is organized as follows. Section 2 contains preliminaries. Section 3 introduces the black-box testing problem for arbitrary ω-regular languages with partial observability, and shows that it can be reduced to the problem for canonical languages called the Rabin languages. Section 4 presents our black-box strategies for the Rabin languages, and proves them correct. Section 5 obtains asymptotically optimal upper bounds on the time to the last restart. Section 6 reports some experimental results.

2 Preliminaries

Directed graphs. A directed graph is a pair $G = (V, E)$, where V is the set of nodes and $E \subseteq V \times V$ is the set of edges. A path (infinite path) of G is a finite (infinite) sequence $\pi = v_0, v_1, \ldots$ of nodes such that $(v_i, v_{i+1}) \in E$ for every $i = 0, 1, \ldots$. A path consisting only of one node is *empty*. Given two vertices $v, v' \in V$, the *distance* from v to v' is the length of a shortest path from v to v', and the distance from v to a set $V' \subseteq V$ is the minimum over all $v' \in V'$ of the distance from v to v'.

A graph G is strongly connected if for every two vertices v, v' there is a path leading from v to v'. A graph $G' = (V', E')$ is a subgraph of G, denoted $G' \preceq G$, if $V' \subseteq V$ and $E' \subseteq E \cap (V' \times V')$; we write $G' \prec G$ if $G' \preceq G$ and $G' \neq G$. A graph $G' \preceq G$ is a strongly connected component (SCC) of G if it is strongly connected and no graph G'' satisfying $G' \prec G'' \preceq G$ is strongly connected. An SCC $G' = (V', E')$ of G is a bottom SCC (BSCC) if $v \in V'$ and $(v, v') \in E$ imply $v' \in V'$.

Partially observable Markov chains. Fix a finite set Σ of *observations*. A *partially observable Markov chain* is a tuple $\mathcal{M} = (S, s_{in}, \Sigma, \mathit{Obs}, \mathbf{P})$, where

- Σ is a set of *observations*;
- S is a finite set of *states* and $s_{in} \in S$ is the *initial* state;
- $\mathit{Obs}\colon S \to \Sigma$ is an *observation function* that assigns to every state an observation; and
- $\mathbf{P} : S \times S \to [0, 1]$ is the transition probability matrix, such that for every $s \in S$ it holds $\sum_{s' \in S} \mathbf{P}(s, s') = 1$,

Intuitively, $\mathit{Obs}(s)$ models the information we can observe when the chain visits s. For example, if s is the state of a program, consisting of the value of the program counter and the values of all variables, $\mathit{Obs}(s)$ could be just the values of the program counter, or the values of a subset of public variables. The graph of $\mathcal{M}$ has S as set of nodes and $\{(s, s') \mid \mathbf{P}(s, s') > 0\}$ as set of edges. Abusing language, we also use $\mathcal{M}$ to denote the graph of $\mathcal{M}$. A *run* of $\mathcal{M}$ is an infinite path $\rho = s_0 s_1 \cdots$ of $\mathcal{M}$; we let $\rho[i]$ denote the state s_i. The sequence $\mathit{Obs}(\rho) \coloneqq \mathit{Obs}(s_0)\mathit{Obs}(s_1) \cdots$ is the *observation* associated to ρ. Each path π in $\mathcal{M}$ determines the set of runs $\mathsf{Cone}(\pi)$ consisting of all runs that start with π. To $\mathcal{M}$ we assign the probability space $(\mathsf{Runs}, \mathcal{F}, \mathbb{P})$, where Runs is the set of all runs in $\mathcal{M}$, $\mathcal{F}$ is the σ-algebra generated by all $\mathsf{Cone}(\pi)$, and $\mathbb{P}$ is the unique probability measure such that $\mathbb{P}[\mathsf{Cone}(s_0 s_1 \cdots s_k)] = \mu(s_0) \cdot \prod_{i=1}^{k} \mathbf{P}(s_{i-1}, s_i)$, where the empty product equals 1. The expected value of a random variable $f\colon \mathsf{Runs} \to \mathbb{R}$ is $\mathbb{E}[f] = \int_{\mathsf{Runs}} f \, d\mathbb{P}$.

Partially Observable Markov Decision Processes. A Σ-*observable Markov Decision Process* (Σ-MDP) is a tuple $\mathsf{M} = (S, s_{in}, \Sigma, \mathit{Obs}, A, \Delta)$, where $S, s_{in}, \Sigma, \mathit{Obs}$ are as for Markov chains, A is a finite set of *actions*, and $\Delta\colon S \times A \to \mathcal{D}(S)$ is a *transition function* that for each state s and action $a \in A(s)$ yields a probability distribution over successor states. The probability of state s' in this distribution is denoted $\Delta(s, a, s')$.

Strategies. A *strategy* on Σ-MDPs with A as set of actions is a function $\sigma\colon (\Sigma \times A)^*\Sigma \to A$, which given a finite path $\pi = \ell_0 a_0 \, \ell_1 \, a_1 \ldots a_{n-1} \, \ell_n \in (\Sigma \times A)^*\Sigma$, yields the action $\sigma(\pi) \in A$ to be taken next. Notice that σ only "observes" $\mathit{Obs}(s)$, not the state s itself. Therefore, it can be applied to any Σ-MDP $\mathsf{M} = (S, s_{in}, \Sigma, \mathit{Obs}, A, \Delta)$, inducing the Markov chain $\mathsf{M}^\sigma = (S^\sigma, s_{in}, \Sigma, \mathit{Obs}, A, \mathbf{P}^\sigma)$ defined as follows: $S^\sigma = (S \times A)^* \times S$; and for every state $\pi \in S^\sigma$ of M^σ ending at a state $s \in S$ of M, the successor distribution is defined by $\mathbf{P}^\sigma(\pi, \pi \, a \, s') \coloneqq \Delta(s, a, s')$ if $\sigma(\pi) = a$ and 0 otherwise.

3 The black-box testing problem

Fix a set Σ of observations, and let r, c (for **r**estart and **c**ontinue) be two actions. We associate to a Σ-observable Markov chain $\mathcal{M} = (S, s_{in}, \Sigma, Obs, \mathbf{P})$ a *restart MDP* $\mathsf{M}_\mathsf{r} = (S, s_{in}, Obs, \{\mathsf{r}, \mathsf{c}\}, \Delta)$, where for every two states $s, s' \in S$ the transition function is given by: $\Delta(s, \mathsf{r}, s') = 1$ if $s' = s_{in}$ and 0 otherwise, and $\Delta(s, \boldsymbol{c}, s') = \mathbf{P}(s, s')$. Intuitively, at every state of M_r we have the choice between restarting the chain $\mathcal{M}$ or continuing.

We consider black-box strategies on Σ and $\{\mathsf{r}, \mathsf{c}\}$. Observe that if a run π of $\mathsf{M}_\mathsf{r}^\sigma$ contains finitely many occurrences of r, then the suffix of π after the last occurrence of r is a run of $\mathcal{M}$ (after dropping the occurrences of the continue action c). More precisely, if $\pi = \pi_0\pi'$, where π' is the longest suffix of π not containing r, then $\pi' = (\pi_0\, s_{in})\, (\pi_0\, s_{in}\, \mathsf{c}\, s_1)\, (\pi_0\, s_{in}\, \mathsf{c}\, s_1\, \mathsf{c}\, s_2) \ldots$, where $s_{in}s_1s_2\ldots$ is a run of $\mathcal{M}$. The sequence of observations of $s_{in}s_1s_2\ldots$ is an infinite word over Σ, called the *tail* of π; formally $\mathit{tail}(\pi) := Obs(s_{in})\,Obs(s_1)\,Obs(s_2)\cdots$.

▶ **Definition 2** (Black-box testing strategies). *Let $L \subseteq \Sigma^\omega$ be an ω-regular language. A* black-box strategy *σ on Σ and $\{\mathsf{r}, \mathsf{c}\}$ is a* testing strategy for *L if it satisfies the following property: for every Σ-observable Markov chain $\mathcal{M}$, if $\Pr_\mathcal{M}(L) > 0$ then w.p.1 a run of $\mathsf{M}_\mathsf{r}^\sigma$ has a finite number of restarts, and its tail belongs to L. The* black-box testing problem *for L consists of finding a black-box testing strategy for L.*

We denote by $\#\mathsf{r}(\rho) \in \mathbb{N} \cup \{\infty\}$ the number of appearances of the restart action r in ρ. Intuitively, the language L models the set of potential violations of a given liveness specifications. If we sample any finite-state Σ-observable Markov chain $\mathcal{M}$ according to a testing strategy for L, then w.p.1 we eventually stop restarting, and the tail of the run is a violation, or there exist no violations.

3.1 Canonical black-box testing problems

Using standard automata-theoretic techniques, the black-box testing problem for an arbitrary ω-regular language L can be reduced to the black-box testing problem for a canonical language. For this, we need to introduce some standard notions of the theory of automata on infinite words.

A deterministic Rabin automaton (DRA) $\mathcal{A}$ over an alphabet Σ is a tuple $(Q, \Sigma, \gamma, q_0, Acc)$, where Q is a finite set of states, $\gamma\colon Q \times \Sigma \to Q$ is a transition function, $q_0 \in Q$ is the initial state, and $Acc \subseteq 2^Q \times 2^Q$ is the acceptance condition. The elements of Acc are called *Rabin pairs*. A word $w = a_0a_1a_2\ldots \in \Sigma^\omega$ is accepted by $\mathcal{A}$ if the unique run $q_0q_1q_2\ldots$ of $\mathcal{A}$ on w satisfies the following condition: there exists a Rabin pair $(E, F) \in Acc$ such that $a_i \in E$ for infinitely many $i \in \mathbb{N}$ and $a_i \in F$ for finitely many $i \in \mathbb{N}$. It is well known that DRAs recognize exactly the ω-regular languages (see e.g. [2]). The *Rabin index* of an ω-regular language L is the minimal number of Rabin pairs of the DRAs that recognize L.

▶ **Definition 3.** *Let $k \geq 1$, and let $M_k = \{\boldsymbol{e}_1, \ldots, \boldsymbol{e}_k, \boldsymbol{f}_1, \ldots, \boldsymbol{f}_k\}$ be a set of* markers. *The* Rabin language *$\mathcal{R}_k \subseteq (2^{M_k})^\omega$ is the language of all words $w = \alpha_0\alpha_1\cdots \in (2^{M_k})^\omega$ satisfying the following property: there exists $1 \leq j \leq k$ such that $\boldsymbol{e}_j \in \alpha_i$ for infinitely many $i \geq 0$, and $\boldsymbol{f}_j \in \alpha_i$ for at most finitely many $i \geq 0$.*

We show that the black-box testing problem for languages of Rabin index k can be reduced to the black-box testing problem for $\mathcal{R}_k$.

▶ **Lemma 4.** *There is an algorithm that, given an ω-regular language $L \subseteq \Sigma^\omega$ of index k and given a testing strategy σ_k for $\mathcal{R}_k$, effectively constructs a testing strategy σ_L for L.*

Proof. (Sketch, full proof in the Appendix.) Let $\mathcal{A} = (Q, \Sigma, \gamma, q_0, Acc)$ be a DRA recognizing $L \subseteq \Sigma^\omega$ with accepting condition $Acc = \{(E_1, F_1), \ldots, (E_k, F_k)\}$, i.e., Acc contains k Rabin pairs. Let σ_k be a black-box strategy for the Rabin language $\mathcal{R}_k$. We construct a black-box strategy σ_L for L.

Let $w = \ell_1 a_1 \ell_2 \cdots \ell_{n-1} a_n \ell_n \in (\Sigma \times \{\mathsf{r}, \mathsf{c}\})^* \Sigma$. We define the action $\sigma_L(w)$ as follows. Let $q_0 q_1 \ldots q_n$ be the unique run of $\mathcal{A}$ on the word $\ell_1 \ell_2 \ldots \ell_n \in \Sigma^*$. We then define $v = \ell'_1 a_1 \ell'_2 \cdots \ell'_{n-1} a_n \ell'_n \in (2^{M_k} \times \{\mathsf{r}, \mathsf{c}\})^* 2^{M_k}$ as the word given by: $\boldsymbol{e}_j \in \ell'_i$ iff $q_i \in E_j$, and $\boldsymbol{f}_j \in \ell'_i$ iff $q_i \in F_j$. (Intuitively, we mark with $\boldsymbol{e}_j$ the positions in the run at which the DRA visits E_j, and with $\boldsymbol{f}_j$ the positions at which the DRA visits F_j.) We set $\sigma_L(w) := \sigma_k(v)$. We show in the Appendix that σ_L is a black-box strategy for L. ◀

4 Black-box strategies for Rabin languages

We describe a family of testing strategies for the Rabin languages $\{\mathcal{R}_k \mid k \geq 1\}$. In Section 4.1 we describe our strategy in detail. In Section 4.2 we introduce the progress radius and the progress probability, two parameters of a chain needed to prove correctness and necessary for quantitative analysis in Section 5. In Section 4.3 we formally prove that our strategy works.

4.1 The strategy

Let $\mathcal{M}$ be a Markov chain with observations in 2^{M_k}, and let $\pi = s_0 s_1 s_2 \cdots s_m$ be a finite path of $\mathcal{M}$. The *length* of π is m, and its last state, denoted $\mathit{last}(\pi)$, is s_m. The *second half* of π is the path $\mathit{SecondHalf}(\pi) := s_{\lceil m/2 \rceil} \ldots s_m$. The *concatenation* of π and a finite path $\rho = r_0 r_1 \cdots r_l$ of $\mathcal{M}$ such that $s_m = r_0$ is the path $\pi \odot \rho := s_0 s_1 s_2 \cdots s_m r_1 \cdots r_l$. A path π is *i-good* if it has length 0 or there are markers $\boldsymbol{e}_i, \boldsymbol{f}_i \in M_k$ such that some state s of π satisfies $\boldsymbol{e}_i \in \mathit{Obs}(s)$ and no state s of π satisfies $\boldsymbol{f}_i \in \mathit{Obs}(s)$. Further, π is *good* if it is i-good for some $1 \leq i \leq k$.

The strategy $\mathfrak{S}[f]$, described in Figure 2, is parametrized by a function $f \colon \mathbb{N} \to \mathbb{N}$. The only requirement on f is $\limsup_{n\to\infty} f(n) = \infty$. In words, after the n-th restart the strategy

```
$n := 0$                                          ▷ number of restarts
while true do
    $\pi \leftarrow s_{in}$                                   ▷ initial state of the chain
    while $\mathit{SecondHalf}(\pi)$ is good do
        sample path $\rho$ from state $\mathit{last}(\pi)$
        of length $2 \cdot f(n)$                  ▷ even length for convenience
        $\pi \leftarrow \pi \odot \rho$
    end while                                             ▷ restart
    $n \leftarrow n + 1$
end while
```

Figure 2 Strategy $\mathfrak{S}[f]$ for the Rabin language $\mathcal{R}_k$ and a function $f \colon \mathbb{N} \to \mathbb{N}$.

keeps sampling in blocks of $2 \cdot f(n)$ steps until the *second half* of the complete path sampled so far is bad, in which case it restarts. For example, after the n-th restart the strategy samples a block $\pi_0 = \pi_{01} \odot \pi_{02}$, where $|\pi_{01}| = |\pi_{02}| = f(n)$, and checks whether π_{02} is good; if not, it restarts, otherwise it samples a block $\pi_1 = \pi_{11} \odot \pi_{12}$ starting from $\mathit{last}(\pi_{02})$, and checks whether $\pi_{11} \odot \pi_{12}$ is good; if not, it restarts; if so it samples a block $\pi_2 = \pi_{21} \odot \pi_{22}$ starting from $\mathit{last}(\pi_{12})$, and checks whether $\pi_{12} \odot \pi_{21} \odot \pi_{22}$ is good, etc. Intuitively, the growth of f controls how the strategy prioritizes deep runs into the chain over quick restarts while the number of restarts increases.

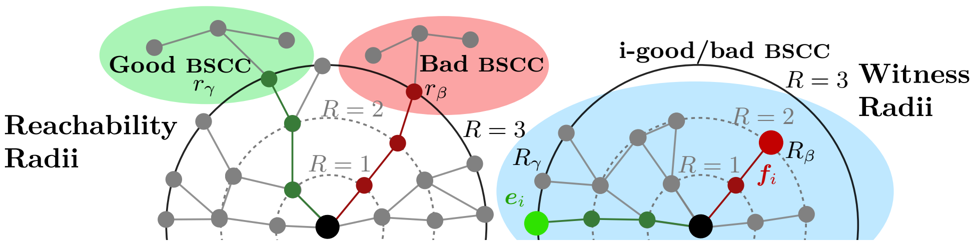

Figure 3 *Left:* Intuitively, r_γ and r_β denote an upper bound of how hard it is to reach a BSCC. *Right:* R_γ and R_β measure how hard it is to reach a state with label $\boldsymbol{e}_i/\boldsymbol{f}_i$ inside the BSCCs.

In the rest of the paper we prove that our strategy is correct, and obtain optimal upper bounds on the number of steps to the last reset. These bounds are given in terms of two parameters of the chain: the progress radius and the progress probability. We introduce the parameters in section 4.2.

4.2 Progress radius and progress probability

We define the notion of progress radius and progress probability for a Markov chain $\mathcal{M}$ with 2^{M_k} as set of observations and such that $\Pr(\mathcal{R}_k) > 0$. Intuitively, the progress radius is the smallest number of steps such that, for any state of the chain, conducting only this number of steps one can "make progress" toward producing a good run or a bad run. The progress probability gives a lower bound for the probability of the paths that make progress.

We define the notions only for the case $k = 1$, which already contains all the important features. The definition for arbitrary k is more technical, and is given in the Appendix.

Good runs and good BSCCs. We extend the definition of good paths to good runs and good BSCCs of a Markov chain. A run $\rho = s_0 s_1 s_2 \ldots$ is *good* if $\boldsymbol{e}_1$ appears infinitely often in ρ and $\boldsymbol{f}_1$ finitely often, and *bad* otherwise. So a run ρ is good iff there exists a decomposition of ρ into an infinite concatenation $\rho := \pi_0 \odot \pi_1 \odot \pi_2 \odot \cdots$ of non-empty paths such that $\pi_1, \pi_2, \ldots$ are good. We let P_{good} denote the probability of the good runs of $\mathcal{M}$.

A BSCC of $\mathcal{M}$ is *good* if it contains at least one state labeled by $\boldsymbol{e}_1$ and no state labeled by $\boldsymbol{f}_1$, and bad otherwise. It is well-known that the runs of any finite-state Markov chain reach a BSCC and visit all its states infinitely often w.p.1 [2, Thm. 10.27]. It follows that good (resp. bad) runs eventually reach a good (resp. bad) BSCC w.p.1.

Progress radius. Intuitively, the progress radius R_{m} is the smallest number of steps such that, for any state s, by conducting R_{m} steps one can "make progress" toward producing a good run – by reaching a good BSCC or, if already in one, by reaching a state with observation $\boldsymbol{e}_1$ – or a bad run.

▶ **Definition 5** (Good-reachability and good-witness radii). *Let B_γ be the set of states of $\mathcal{M}$ that belong to good BSCCs and let S_γ be the set of states from which it is possible to reach B_γ, and let $s \in S_\gamma$. A non-empty path π starting at s is a* good progress path *if*

- *$s \in S_\gamma \setminus B_\gamma$, and π ends at a state of B_γ; or*
- *$s \in B_\gamma$, and π ends at a state with observation $\boldsymbol{e}_1$.*

The good-reachability radius *r_γ is the maximum, taken over every $s \in S_\gamma \setminus B_\gamma$, of the length of a shortest progress path for s. The* good-witness radius *R_γ is the same maximum, but taken over every $s \in B_\gamma$.*

The bad-reachability and bad-witness radii, denoted r_β and R_β are defined analogously. Only the notion of progress path of a states $\in B_\beta$ needs to be adapted. Loosely speaking, a bad BSCC either contains no states with observation $\boldsymbol{e}_1$, or it contains some state with observation $\boldsymbol{f}_1$. Accordingly, if no state of the BSCC of s has observation $\boldsymbol{e}_1$, then any non-empty path starting at s is a progress path, and otherwise a progress path of s is a non-empty path starting at s and ending at a state with observation $\boldsymbol{f}_1$. We illustrate the definition of the reachability and witness radii in Figure 3. We leave r_β, R_β, p_β, and P_β undefined if the chain does not contain a bad BSCC, and hence runs are good w.p.1.

▶ **Definition 6** (Progress radius). *The* progress radius R_m *of* $\mathcal{M}$ *is the maximum of* r_γ, R_γ, r_β, *and* R_β.

Progress probability. From any state of the Markov chain it is possible to "make progress" by executing a progress path of length R_m. However, the probability of such paths varies from state to state. Intuitively, the progress probability gives a lower bound on the probability of making progress.

▶ **Definition 7.** *Let* B_γ *be the set of states of* $\mathcal{M}$ *that belong to good BSCCs, let* S_γ *be the set of states from which it is possible to reach* B_γ*, and let* $s \in S_\gamma$*. The* good-reachability probability p_γ *is the minimum, taken over every* $s \in S_\gamma \setminus B_\gamma$*, of the probability that a path with length* r_γ *starting at* s *contains a good progress path. The* good-witness probability P_γ *is the same mininum, but taken over every* $s \in B_\gamma$ *with paths of length* R_γ*. The corresponding bad probabilities are defined analogously. The* progress probability P_m *is the minimum of* $p_\gamma, P_\gamma, p_\beta, P_\beta$.

4.3 Correctness proof

We prove that the strategy $\mathfrak{S}[f]$ of section 4.1 is a valid testing strategy $\mathfrak{S}[f]$ for arbitrary Markov chains $\mathcal{M}$. First, we will give an upper bound on the probability that $\mathfrak{S}[f]$ restarts "incorrectly", i.e. at a state $s \in S_\gamma$ from which a good BSCC could still be reached.

▶ **Lemma 8.** *Let* $\mathcal{M}$ *be a Markov chain, and let* $\mathsf{M}_\mathsf{r}^{\mathfrak{S}[f]}$ *be its associated Markov chain with* $\mathfrak{S}[f]$ *as restart strategy. Let* NB_n *be the set of paths of* $\mathsf{M}_\mathsf{r}^{\mathfrak{S}[f]}$ *that have at least* $n-1$ *restarts and only visit states in* S_γ *after the* $(n-1)$*-th restart. We have:*

$$\Pr[\#\mathsf{r} \geq n \mid NB_n] \leq 3(1-P_\mathrm{m})^{\lfloor f(n)/R_\mathrm{m} \rfloor - 1}$$

The technical proof of this lemma is in the Appendix. We give here the proof for a special case that illustrates most ideas. Consider the Markov chain with labels in $2^{\{\boldsymbol{e}_1, \boldsymbol{f}_1\}}$ at the top of Figure 4. The labeling function is $Obs(s_\mathrm{goal}) = \{\boldsymbol{e}_1\}$ and $Obs(s) = \emptyset$ for all other states, and $Obs(\rho) \in \mathcal{R}_1$ iff ρ visits s_goal infinitely often. The set S_γ contains all states because s_goal is reachable from every state. The only BSCC is $\mathcal{B} = \{s_1, s_\mathrm{goal}\}$, and it is a good BSCC. From the definitions of the parameters we obtain $r_\gamma = R_\gamma = 1$, $p_\gamma = p$ and $P_\gamma = q$. Further, since there are no bad BSCCs, r_β and R_β are undefined, and so $R_\mathrm{m} = 1$. So for this Markov chain Lemma 8 states $\Pr[\#\mathsf{r} \geq n \mid NB_n] \leq 3(1-P_\mathrm{m})^{f(n)-1}$. Let us see why this is the case.

Let ρ be a run of $\mathsf{M}_\mathsf{r}^{\mathfrak{S}[f]}$ such that $\#\mathsf{r}(\rho) \geq n$, i.e., ρ has at least n restarts. Since S_γ contains all states, we have $\rho \in NB_n$ iff $\#\mathsf{r}(\rho) \geq n-1$. We consider three cases. In the definition of the cases we start counting steps immediately after the $(n-1)$-th restart, and denote by $\rho[a, b]$ the fragment of ρ that starts immediately before step a, and ends immediately after step b.

$$\begin{tikzcd}
s_{\text{start}} \arrow[loop left, "1-p"] \arrow[r, "p"] & s_1 \arrow[loop right, "1-q"] \arrow[d, bend right, "q"'] \\
& s_{\text{goal}} \arrow[u, bend right, "1-q"'] \arrow[loop right, "q"]
\end{tikzcd}$$

$$s_{\text{start}} \to s_1 \to \underbrace{s_1 \to s_{\text{goal}}}_{\text{no restart}} \to \underbrace{s_1 \to s_1 \to s_1 \to s_1}_{\text{restart!}}$$

Figure 4 A Markov chain (left), and (a finite prefix of) one of its runs for $n = 2$ and $f(n) = n$ (right). After 2 steps, the run has reached a good BSCC $\mathcal{B}$. After 4 steps, σ checks whether to restart, but decides against it because of s_{goal} in step 4. After 8 steps it checks again, restarting this time. This restart is covered by the third case of the case distinction, with $k = 4$. The run must have visited s_{goal} in step 4, because otherwise the minimal k would be 3 or less.

(a) After $f(n)$ steps, ρ has not yet reached $\mathcal{B}$.
Then ρ has stayed in s_{start} for $f(n)$ consecutive steps, which, since $p = p_\gamma$, happens with probability at most $(1 - p_\gamma)^{f(n)}$.

(b) After $f(n)$ steps, ρ has already reached $\mathcal{B}$. Further, the n-th restart happens immediately after step $2f(n)$.
In this case, by the definition of the strategy, ρ does not visit s_{goal} during the interval $\rho[f(n) + 1, 2f(n)]$ (the second half of $[0, 2f(n)]$). So ρ stays in s_1 during the interval $\rho[f(n) + 1, 2f(n)]$ which, since ρ has already reached $\mathcal{B}$ by step $f(n)$, occurs with probability $(1 - P_\gamma)^{f(n)}$.

(c) After $f(n)$ steps, ρ has already reached $\mathcal{B}$. Further, the n-th restart does not happen before step $2f(n) + 1$.
Since the n-th restart happens at some point, and not before step $2f(n) + 1$, by the definition of the strategy there is a smallest number $k \geq f(n)$ such that ρ does not visit s_{goal} during the interval $\rho[k + 1, 2k]$. Because we assume that the n-th restart happens after step $2f(n)$, we even have $k > f(n)$. By the minimality of k, the run ρ does visit s_{goal} during the interval $\rho[k, 2k - 2]$. So ρ moves to s_{goal} at step k, and then stays in s_1 for k steps. The probability of the runs that eventually move to s_{goal} and then move to stay in s_1 for k steps is $P_\gamma(1 - P_\gamma)^k$.

Figure 4 shows at the bottom an example of a run, and how the stratgy handles it. Since (a)-(c) are mutually exclusive events, $\Pr[\#\mathsf{r} \geq n \mid NB_n]$ is bounded by the sum of their probabilities, where in case (c) we sum over all possible values of k. This yields:

$$Pr[\#\mathsf{r} \geq n \mid NB_n] \leq \; (1-p_\gamma)^{f(n)} + (1-P_\gamma)^{f(n)} + \sum_{k=f(n)+1}^{\infty} P_\gamma(1-P_\gamma)^k \;\leq\; 3(1-P_{\mathrm{m}})^{f(n)}.$$

The proof for arbitrary Markov chains given in the Appendix has the same structure, and in particular the same split into three different events. Applying Lemma 8 we now easily obtain (see the Appendix for a detailed proof) an upper bound for the probability to restart an n-th time. Note that this bound captures the "correct" as well as the "incorrect" restarts:

▶ **Lemma 9** (Restarting probability). *Let $\mathcal{M}$ be a Markov chain, and let $\mathsf{M}_\mathsf{r}^{\mathfrak{S}[f]}$ be its associated Markov chain with $\mathfrak{S}[f]$ as restart strategy. The probability that a run restarts again after $n - 1$ restarts satisfies:*

$$\Pr[\#\mathsf{r} \geq n \mid \#\mathsf{r} \geq n-1] \leq 1 - P_{good}\left(1 - 3(1 - P_{\mathrm{m}})^{\lfloor f(n)/R_{\mathrm{m}} \rfloor - 1}\right)$$

Proof. Let NB_n be the set of paths of $\mathsf{M}_\mathsf{r}^{\mathfrak{S}[f]}$ that have at least $n - 1$ restarts and only visit states in S_γ after the $(n - 1)$-th restart and $\overline{NB}_n$ its complement. We have

$$\begin{aligned}\Pr[\#\mathsf{r} \geq n \mid \#\mathsf{r} \geq n-1] = &\Pr[\#\mathsf{r} \geq n \mid NB_n] \cdot \Pr[NB_n \mid \#\mathsf{r} \geq n-1] + \\ &\Pr[\#\mathsf{r} \geq n \mid \overline{NB}_n] \cdot \Pr[\overline{NB}_n \mid \#\mathsf{r} \geq n-1].\end{aligned}$$

Applying Lemma 8 and $\Pr[\#\mathsf{r} \geq n \mid \overline{NB}_n] \leq 1$, we get

$$\Pr[\#\mathsf{r} \geq n \mid \#\mathsf{r} \geq n-1] \leq \left(3(1-P_{\mathrm{m}})^{\lfloor f(n)/R_{\mathrm{m}} \rfloor -1}\right) \Pr[NB_n \mid \#\mathsf{r} \geq n-1] + \Pr[\overline{NB}_n \mid \#\mathsf{r} \geq n-1]$$

W.p.1, good runs of $\mathcal{M}$ only visit states of S_γ and so $\Pr[NB_n \mid \#\mathsf{r} \geq n-1] \geq P_{\mathrm{good}}$ and thus $\Pr[\overline{NB}_n \mid \#\mathsf{r} \geq n-1] \leq 1 - P_{\mathrm{good}}$, which completes the proof. ◀

Finally, we show that $\mathfrak{S}[f]$ is a correct testing strategy. Further, we show that the condition $\limsup_{n\to\infty} f(n) = \infty$ is not ony sufficient, but also necessary. The previous lemma gives an upper bound on the probability for a restart that, for increasing $f(n)$, drops below 1. If $f(n)$ is above that threshold for infinitely many n, it suffices to show that the strategy $\mathfrak{S}[f]$ restarts every bad run:

▶ **Theorem 10.** *$\mathfrak{S}[f]$ is a testing strategy for the Rabin language $\mathcal{R}_k$ iff the function f satisfies $\limsup_{n\to\infty} f(n) = \infty$.*

Proof.
(⇒): We prove the contrapositive. If $\limsup_{n\to\infty} f(n) < \infty$ then there is a bound b such that $f(n) \leq b$ for every $n \geq 0$. Consider a Markov chain over 2^{M_1} consisting of a path of $2b+1$ states, with the last state leading to itself with probability 1; the last state is labeled with $\boldsymbol{e}_1$, and no state is labeled with $\boldsymbol{f}_1$. Then the chain has a unique run that goes from the initial to the last state of the path and stays there forever, and its observation is a word of $\mathcal{R}_1$; therefore, $\Pr(\mathcal{R}_1) = 1$. However, since $2f(n) \leq 2b+1$, $\mathfrak{S}[f]$ always restarts the chain before reaching the last state.
(⇐): By the previous lemma, we can bound the restart probability after $n-1$ restarts by $1 - \left(1 - 3(1-P_{\mathrm{m}})^{\lfloor f(n)/R_{\mathrm{m}} \rfloor -1}\right) P_{\mathrm{good}}$. Because $0 < P_{\mathrm{m}} \leq 1$ and and $P_{\mathrm{good}} > 0$, for large enough $f(n)$ this is smaller than $1 - P_{\mathrm{good}}/2 < 1$. Because of $\limsup_{n\to\infty} f(n) = \infty$, we have that the probability to restart the run another time is at most $1 - P_{\mathrm{good}}/2$ for infinitely many n, and hence the total number of restarts is finite with probability 1. A bad run would enter a bad BSCC B w.p.1 and would then go on to visit a set consisting of all the f_i corresponding to B infinitely often. Thus, $\mathfrak{S}[f]$ would restart this run and hence reached a good run when it does not restart. ◀

5 Quantitative analysis

The quality of a testing strategy is given by the expected number of steps until the last restart, because this is the overhead spent until a violation starts to be executed. As in [6], given a labeled Markov chain $\mathcal{M}$ and a testing strategy σ, we define the number of steps to the last restart as random variables over the Markov chain $\mathsf{M}^\sigma_\mathsf{r}$:

▶ **Definition 11** ($S(\rho)$ and $S_n(\rho)$)**.** *Let ρ be a run of $\mathsf{M}^\sigma_\mathsf{r}$. We define: $S(\rho)$ is equal to 0 if r does not occur in ρ; it is equal to the length of the longest prefix of ρ ending in r, if r occurs at least once and finitely often in ρ; and it is equal to ∞ otherwise. Further, for every $n \geq 1$ we define $S_n(\rho)$ to be equal to 0 if r occurs less than n times in ρ; and equal to the length of the segment between the $(n-1)$-th (or the beginning of ρ) and the n-th occurrence of r.*

In this section we investigate the dependence of $\mathbb{E}[S]$ on the function $f(n)$. A priori it is unclear whether $f(n)$ should grow fast or slow. Consider the case in which all BSCCs of the chain, good or bad, have size 1, and a run eventually reaches a good BSCC with probability p. In this case the strategy restarts the chain until a sample reaches a good BSCC for the first time. If $f(n)$ grows fast, then after a few restarts, say r, every subsequent run reaches a BSCC of the chain with large probability, and so the expected number of restarts is small, at most $r + (1/p)$. However, the number of steps executed during these few restarts is large, because $f(n)$ grows fast; indeed, only the run after the penultimate restart executes already at least $2f(r + (1/p) - 1)$ steps.

In a first step we show that $\mathbb{E}[S] = \infty$ holds for every function $f(n) \in 2^{\Omega(n)}$.

▶ **Proposition 12.** *Let $f \in 2^{\Omega(n)}$. Then there exists a Markov chain such that the testing strategy of Figure 2 satisfies $\mathbb{E}(S) = \infty$.*

Proof. Let f be in $2^{\Omega(n)}$. Then there exists some $k > 0$ such that we have $\limsup_{n\to\infty} f(n) \cdot (1/2)^{n/k} > 0$. Consider a Markov chain with $P_{\text{good}} = 1 - (1/2)^{1/k}$. Then we have $\mathbb{E}(S) = \sum_{n=0}^{\infty} \mathbb{E}(S_n \mid \#\mathsf{r} \geq n-1) P(\#\mathsf{r} \geq n-1)$. We have that $P(\#\mathsf{r} \geq n-1 \mid \#\mathsf{r} \geq n-2) \geq 1 - P_{\text{good}}$ because only good runs will not be restarted. We also have that $\mathbb{E}(S_n \mid \#\mathsf{r} \geq n-1) \geq f(n)(1 - P_{\text{good}})$ because of the same reason. Thus

$$\mathbb{E}[S] = \sum_{n=0}^{\infty} \mathbb{E}(S_n \mid \#\mathsf{r} \geq n-1) P(\#\mathsf{r} \geq n-1) \geq \sum_{n=0}^{\infty} f(n)(1 - P_{\text{good}}) \cdot (1 - P_{\text{good}})^n$$

and hence $\mathbb{E}[S] \geq \sum_{n=0}^{\infty} f(n)(1/2)^{n/k} = \infty$. ◀

It follows that (if we limit ourselves to monotonic functions, which is no restriction in practice), we only need to consider functions $f(n)$ satisfying $f(n) \in \omega(1) \cap 2^{o(n)}$. In the rest of the section we study the strategies corresponding to polynomial functions $f(n) = n^c$ for $c \in \mathbb{N}_+$, and obtain an upper bound as a function of the parameters $R_{\text{m}}/P_{\text{m}}$, P_{good}, and c. The study of subexponential but superpolynomial functions is beyond the scope of this paper.

5.1 Quantitative analysis of strategies with $f(n) = n^c$

We give an upper bound on $\mathbb{E}(S)$, the expected total number of steps before the last restart. Our starting point is Lemma 9, which bounds the probability to restart for the n-th time, if $(n-1)$ restarts have already happened. When the number n of restarts is small, the value of the right-hand-side is above 1, and so the bound is not useful. We first obtain a value X such that after X restarts the right-hand-side drops below 1.

▶ **Lemma 13.** *Let $X = \sqrt[c]{R_{\text{m}}\left(2 + \ln(1/6)/\ln(1 - P_{\text{m}})\right)}$. For all $n \geq X$, we have*

$$\Pr[\#\mathsf{r} \geq n \mid \#\mathsf{r} \geq n-1] \leq 1 - P_{good}/2$$

when restarting according to $\mathfrak{S}[n \mapsto n^c]$.

Proof. Follows immediately from Lemma 9, the fact that the restart probability decreases with n, the definition of X, and some calculations. We recall the statement of Lemma 9:

$$\Pr[\#\mathsf{r} \geq n \mid \#\mathsf{r} \geq n-1] \leq 1 - P_{\text{good}}\left(1 - 3(1 - P_{\text{m}})^{\lfloor f(n)/R_{\text{m}} \rfloor - 1}\right)$$

Plugging in an $n \geq X$ validates the claim. ◀

We now try to find a bound for $\mathbb{E}[S]$: By linearity of expectation, we have $\mathbb{E}[S] = \sum_{i=1}^{\infty} \mathbb{E}[S_n]$. We split the sum into two parts: for $n < X$, and for $n \geq X$. For $n < X$ we just approximate $\Pr[\#\mathsf{r} \geq n-1]$ by 1. For $n > X$ we can say more thanks to Lemma 13:

$$\begin{aligned}\Pr[\#\mathsf{r} \geq n-1] &= \Pr[\#\mathsf{r} \geq n-1 | \#\mathsf{r} \geq n-2] \cdots \Pr[\#\mathsf{r} \geq X+1 | \#\mathsf{r} \geq X] \cdot \Pr[\#\mathsf{r} \geq X] \\ &\leq \prod_{k=\lceil X \rceil}^{n} \Pr[\#\mathsf{r} \geq k | R \geq k-1] \leq (1 - P_{\text{good}}/2)^{n-X}\end{aligned}$$

This yields:

$$\begin{aligned}\mathbb{E}[S] &= \sum_{n=0}^{\infty} \mathbb{E}[S_n \mid \#\mathsf{r} \geq n-1] \Pr[\#\mathsf{r} \geq n-1] \\ &\leq \sum_{n=0}^{X} \mathbb{E}[S_n \mid \#\mathsf{r} \geq n-1] + \sum_{n=X}^{\infty} \mathbb{E}[S_n \mid \#\mathsf{r} \geq n-1] \cdot (1 - P_{\text{good}}/2)^{n-X} \qquad (1)\end{aligned}$$

It remains to bound the expected number of steps between two restarts $\mathbb{E}[S_n \mid \#\mathsf{r} \geq n-1]$, which is done in Lemma 14 below. The proof can be found in the Appendix. The proof first observes that the expected number of steps it takes to reach a good or a bad BSCC is r_γ/p_γ resp. r_β/p_β. Then we give a bound on the expected number of steps it takes to perform a progress path inside a bad BSCC for the first time, or to not perform a progress path inside a good BSCC for an entire second half of a run at some point after the $(n-1)$-st restart; the bound is also in terms of R_m/P_m and $R_\text{m}/P_\text{m}(1-P_\gamma)$. The term $2f(n)$ comes from the fact that the strategy always executes at least $2f(n)$ steps. The term $2R_\text{m}$ is an artifact due to the "granularity" of the analysis, where we divide runs in blocks of R_m steps.

▶ **Lemma 14** (Expected number of steps in a fragment). *For the strategy $\mathfrak{S}[n \mapsto n^c]$ we have:*

$$\mathbb{E}[S_n \mid \#\mathsf{r} \geq n-1] \leq 2(R_\text{m} + f(n)) + 9\left(\frac{R_\text{m}}{P_\text{m}(1-P_\gamma)}\right). \qquad (1)$$

Plugging Lemma 14 into (1), we finally obtain (see the Appendix):

▶ **Theorem 15** (Expected number of total steps). *For the strategy $\mathfrak{S}[n \mapsto n^c]$ we have:*

$$\mathbb{E}[S] \in \mathbb{O}\left((c+1)! \cdot 2^c \cdot \left(\frac{R_\text{m}}{P_\text{m}}\right)^{1+1/c} + \frac{2^c(c+1)!}{P_{good}^{c+1}} + (c+1)!(2c)^{c+1}\right).$$

If we fix a value c, we obtain a much simpler statement:

▶ **Corollary 16.** *For a fixed c, the strategy $\mathfrak{S}[n \mapsto n^c]$ satisfies:*

$$\mathbb{E}[S] \in \mathbb{O}\left(\left(\frac{R_\text{m}}{P_\text{m}}\right)^{1+1/c} + \frac{1}{P_{good}^{c+1}}\right).$$

Thus the bound on the total number of steps depends on two quantities, R_m/P_m and P_good. A small c favours the effect of R_m/P_m on the bound, a larger c the effect of P_good. In Section 6 we will see that this closely matches the performance of the algorithms for different values of c on synthetic Markov chains and on Markov chains from the PRISM benchmark set.

5.2 Optimality of the Strategy $f(n) = n^c$

We will prove the following optimality guarantee for our strategies.

▶ **Theorem 17.** *For every $c \in \mathbb{N}_+$ there is a family of Markov chains such that our bound of Corollary 16 on $\mathfrak{S}[n \mapsto n^c]$ is asymptotically optimal, i.e., no other black-box testing strategy is in a better asymptotic complexity class.*

This proves two points: first, our bounds cannot be substantially improved. Second, one necessarily needs information on $\frac{R_\mathrm{m}}{P_\mathrm{m}}$ and P_good to pick an optimal value for c; without any information every value is equally good.

Proof. Consider the family of Markov chains at the top of Figure 5. We take an arbitrary $k > 1$ and set $M = k^{c-1}$ and $p = q = 1/k$. With this choice we have $P_\mathrm{good} = P_\mathrm{m} = 1/k$, and $R_\mathrm{m} = k^{c-1}$. By Lemma 15, the strategy $\mathfrak{S}[n \mapsto n^c]$. satisfies $\mathbb{E}[S] \in \mathbb{O}((R_\mathrm{m}/P_\mathrm{m})^{1+1/c} + (1/P_\mathrm{good})^{c+1}) = \mathbb{O}(k^{c+1})$

We compare this with the optimal number of expected steps before the final restart. Since runs that visit s_goal at least once are good w.p.1, any optimal strategy stops restarting exactly after the visit to s_goal. We claim that every such strategy satisfies $\mathbb{E}[S] \geq R_\mathrm{m}/(P_\mathrm{good}P_\mathrm{m})(1 - P_\mathrm{good})$. For this, we make four observations. First, the probability of a good run is P_good. Second, the expected number of steps of a good run until the first visit to s_goal is $R_\mathrm{m}/P_\mathrm{m}$. Third, the smallest number of steps required to distinguish a bad run, i.e. being in the left BSCC, from a good run is equal to R_m, because until R_m steps are executed, all states visited carry the same label. Hence, every strategy takes $R_\mathrm{m}/(P_\mathrm{good}P_\mathrm{m})$ steps on average before reaching the state s_goal for the first time. Fourth, on average $1/P_\mathrm{good}$ tries are required to have one try result in a good run. Hence, on average at least $\frac{1/P_\mathrm{good}-1}{1/P_\mathrm{good}}$ of the $R_\mathrm{m}/(P_\mathrm{good}P_\mathrm{m})$ steps happen before the last restart. Since $\frac{1/P_\mathrm{good}-1}{1/P_\mathrm{good}} = (1 - P_\mathrm{good})$, this proves the claim. Now $R_\mathrm{m}/(P_\mathrm{good}P_\mathrm{m})(1 - P_\mathrm{good}) = k^{c+1} - k^c \in \Theta(k^{c+1})$ and we are done. ◀

6 Experiments

We report on experiments on three kinds of systems. First, we conduct experiments on two synthetic families of Markov Chains. Second, we repeat the experiments of [6] on models from the standard PRISM Benchmark Suite [10] using our black-box strategies. Finally, we conduct experiments on population protocols from the benchmark suite of the Peregrine tool [4, 5].

Synthetic Experiments. Consider the two (families of) labeled Markov chains at the top of Figure 5. The labels are a and b. In the top chain, state s_{goal} is labeled by a, all others by b. In the bottom chain, the states s_2 to s_M and s_goal are labeled by $\{a, s_\mathrm{start}\}$ and s_sink by b. The language L is the set of words containing infinitely many occurrences of a. In the top chain at the initial state we go right or left with probability q and $(1-q)$, respectively. Runs that go left are bad, and runs that go right are good w.p.1. It follows $P_\mathrm{good} = q$, $R_\mathrm{m} = M$, and $P_\mathrm{m} = \min(p, q)$. In our experiments we fix $q = 1/2$. By controlling M and p, we obtain chains with different values of R_m and P_m for fixed $P_\mathrm{good} = 1/2$. In the bottom chain, $R_\beta = R_\gamma = 1$, $R_\mathrm{m} = r_\gamma = M$, $p_\gamma = p^M$, $p_\beta = (1-p)$ and $P_\mathrm{m} = \min(p^M, 1-p)$ and $P_\mathrm{good} = p^M$.

Recall that the bound obtained in the last section is $\mathbb{E}[S] \leq f(c)(R_\mathrm{m}/P_\mathrm{m})^{1+1/c} + g(c)(1/P_\mathrm{good})^{c+1}$ where $f(c)$ and $g(c)$ are fast-growing functions of c. If P_good and $R_\mathrm{m}/P_\mathrm{m}$ are small, then $f(c)$ and $g(c)$ dominate the number of steps, and hence strategies with small c should perform better. The data confirms this prediction. Further, for fixed P_good, the

$$\begin{tikzcd}
 & & & & \mathbf{s}_{\text{goal}} \arrow[lld, bend right, "1"'] & \\
s_{\text{sink}} \arrow[loop left] & s_{\text{start}} \arrow[l, "1-q"'] \arrow[r, "q"] & s_1 \arrow[r, "1"] & \cdots \arrow[r, "1"] & s_{M-1} \arrow[u, "p"'] \arrow[r, "1-p"] & s_M \arrow[lll, bend left, "1"]
\end{tikzcd}$$

$$\begin{tikzcd}
s_{\text{sink}} \arrow[loop left] & s_{\text{start}} \arrow[l, "1-p"'] \arrow[r, "p"] & s_2 \arrow[r, "p"] \arrow[ll, bend left, "1-p"] & \cdots \arrow[r, "p"] \arrow[lll, bend left, "1-p"] & s_M \arrow[r, "p"] \arrow[llll, bend left, "1-p"] & s_{\text{goal}} \arrow[loop right]
\end{tikzcd}$$

Figure 5 Two families of Markov chains. The initial state is s_{start}. The good runs are those that visit s_{goal} infinitely often. For the top chains, $P_{\text{good}} = q$, $R_{\text{m}} = M$, and $P_{\text{m}} = p$. For the bottom chains, $P_{\text{good}} = p^M$, $R_{\text{m}} = M$, $P_{\text{m}} = p^M$.

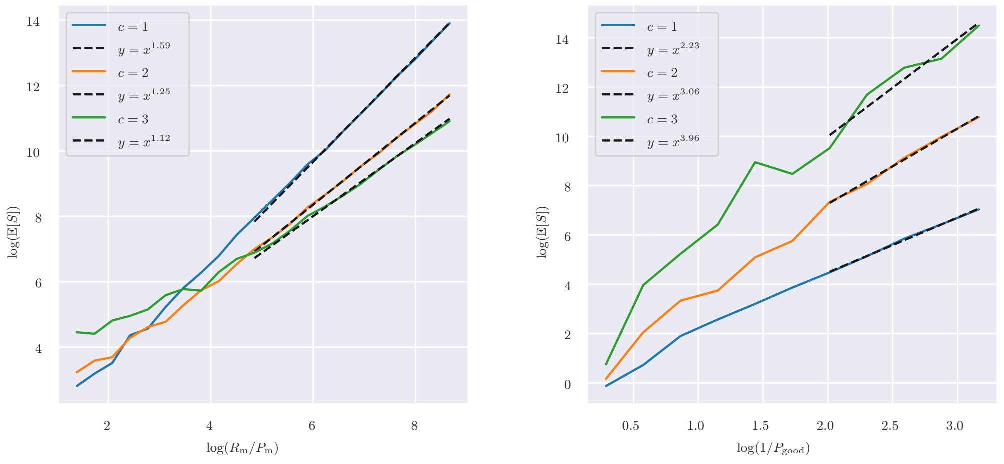

Figure 6 On the left, double-logarithmic plot of the expected total number of steps before the last restart $\mathbb{E}(S)$ for the chain at the top of Figure 5 as a function of $R_{\text{m}}/P_{\text{m}}$ for strategies (2) with $f(n) = n^c$ for varying c. On the right, same for the bottom chain as a function of $1/P_{\text{good}}$. The plots also show linear regressions. The leading exponent can be taken from the legend.

bound predicts $\mathbb{E}[S] \in O((R_{\text{m}}/P_{\text{m}})^{1+1/c})$, and so for growing $R_{\text{m}}/P_{\text{m}}$ strategies with large c should perform better. The left diagram confirms this. Also, the graphs become straight lines in the double logarithmic plot, confirming the predicted polynomial growth. Finally, for $R_{\text{m}}/P_{\text{m}}$ and $1/P_{\text{good}}$ growing roughly at the same speed as in the lower Markov chain, the bound predicts $\mathbb{E}[S] \in O(1/P_{\text{good}}^{c+1})$ for $c = 2, 3$ and $\mathbb{E}[S] \in O(M^2/P_{\text{good}}^{c+1})$ for $c = 1$, and hence for growing P_{good} and $R_{\text{m}}/P_{\text{m}}$, strategies with small c perform better. Again, the right diagram confirms this.

Experiments on the PRISM data set. We evaluate the performance of our black-box testing strategies for different values of c on discrete time Markov chain benchmarks from the PRISM Benchmark suite [10], and compare them with the strategies of [6] for fully observable systems. Table 1 shows the results. The properties checked are of the form $\mathbf{GF}$, $(\mathbf{GF} \to \mathbf{FG})$, or their negations. We add a gridworld example[2] denoted $\overline{\texttt{GW}}$, with larger values of the parameters, to increase the number of states to $\sim 5 \cdot 10^8$. When trying to construct the corresponding Markov chain, Storm experienced a timeout. Runs are sampled using the simulator of the STORM model checker [9] and the python extension STORMPY. We abort a run after 10^6 (Up to $3 \cdot 10^7$ for the gridworld examples $\texttt{gw}$, $\overline{\texttt{gw}}$, and $\overline{\texttt{GW}}$) steps without a restart. The probability of another restart is negligibly small.

The Cautious$_{10}$- and the Bold$_{0.1}$-strategy of [6] store the complete sequence of states observed, and so need linear memory in the length of the sample. Our strategies use at most a logarithmic amount of memory, at none or little cost in the number of steps to the last restart. Our strategies never timeout and, surprisingly, often require *fewer* steps than fully-observable ones. In particular, the strategies for fully observable systems cannot handle $\overline{\text{gridworlds}}$, and only the bold strategy handles gridworld. One reason for this difference is our strategies' ability to adapt to the size of the chain automatically by increasing values of $f(n)$ as n grows. In two cases (nand and bluetooth) the fully observing strategies perform better by a factor of ~ 2 to ~ 3. In comparison to the improvement by a factor of ~ 50 in scale$_{10}$ and a factor of ~ 90 in gridworld of the newly presented black-box strategies over the whitebox strategies, this is negligible.

Table 1 Average number of steps before the final restart, averaged over 300 (100 for Herman and $\overline{\texttt{GW}}$) runs. Results for our strategies for $c = 1, 2, 3$, and the bold and cautious strategies of [6].

	nand	bluetooth	scale$_{10}$	crowds	herman	gw	$\overline{\texttt{gw}}$	$\overline{\texttt{GW}}$
# states	$7 \cdot 10^7$	143 291	121	$1 \cdot 10^7$	$5 \cdot 10^5$	309 327	309 327	$5 \cdot 10^8$
$c = 1$	31 246	4 428	116	**44**	**2**	486	171 219	8 082 659
$c = 2$	18 827	4 548	**75**	61	**1**	404	**152 127**	4 883 449
$c = 3$	32 777	7 615	179	99	**1**	**293**	579 896	**4 252 263**
Bold$_{0.1}$	10 583	4 637	14 528	199	**0**	TO	TO	
Cautious$_{10}$	**6 900**	**2 425**	3 670	101	TO	26 361	TO	

Table 2 Testing population protocols with the strategies $\mathfrak{S}[n \mapsto n^c]$. Experiments were run 100 times, averaging the number of steps to the last restart with a restart threshold of 250 for Average and Conquer (AvC) and 10 000 for the Majority Protocol.

	AvC$_{17,8}$(faulty)		Maj$_{\leq 12}$(faulty)		AvC$_{17,8}$		Maj$_{5,6}$	
$c = 1$	13 645	ce	872	ce	126 294	true	4 264 508	ge
$c = 2$	181 746	ce	4 763	ce	10 485 163	true	11 878 533	ge
Peregrine		TO		ce		TO		true

Experiments on population Protocols. Population protocols are consensus protocols in which a crowd of indistinguishable agents decide a property of their initial configuration by reaching a stable consensus [1, 4]. The specification states that for each initial configuration

[2] Unfortunately, the experimental setup of [6] cannot be applied to this example [19].

the agents eventually reach the right consensus (property holds/does not hold). We have tested our strategies on several protocols from the benchmark suite of Peregrine, the state-of-the-art model checker for population protocols [4, 5]. The first protocol of Table 2 is faulty, but Peregrine cannot prove it; our strategy finds initial configurations for which the protocol exhibits a fault. For the second protocol both our strategies and Peregrine find faulty configurations. The third protocol is correct; Peregrine fails to prove it, and our strategies correctly fail to find counterexamples. The last protocol is correct, but in expectation consensus is reached only after an exponential number of steps in the parameters; we complement the specification, and search for a run that achieves consensus. Thanks to the logarithmic memory requirements, our strategies can run deep into the Markov chain and find the run.

7 Conclusions

We have studied the problem of testing partially observable stochastic systems against ω-regular specifications in a black-box setting where testers can only restart the system, have no information on size or probabilities, and cannot observe the states of the system, only its outputs. We have shown that, despite these limitations, black-box testing strategies exist. We have obtained asymptotically optimal bounds on the number of steps to the last restart. Surprisingly, our strategies never require many more steps than the strategies for fully observable systems of [6], and often even less. Sometimes, the improvement is by a large factor (up to ~ 90 in our experiments) or the black-box strategies are able to solve instances where the strategies of [6] time out.

References

1 Dana Angluin, James Aspnes, Zoë Diamadi, Michael J. Fischer, and René Peralta. Computation in networks of passively mobile finite-state sensors. *Distributed Comput.*, 18(4):235–253, 2006.

2 Christel Baier and Joost-Pieter Katoen. *Principles of model checking.* MIT Press, Cambridge, Massachusetts, 2008.

3 David A. Basin, Vincent Jugé, Felix Klaedtke, and Eugen Zalinescu. Enforceable security policies revisited. *ACM Trans. Inf. Syst. Secur.*, 16(1):3:1–3:26, 2013.

4 Michael Blondin, Javier Esparza, and Stefan Jaax. Peregrine: A tool for the analysis of population protocols. In *CAV (1)*, volume 10981 of *Lecture Notes in Computer Science*, pages 604–611. Springer, 2018.

5 Javier Esparza, Martin Helfrich, Stefan Jaax, and Philipp J. Meyer. Peregrine 2.0: Explaining correctness of population protocols through stage graphs. In *ATVA*, volume 12302 of *Lecture Notes in Computer Science*, pages 550–556. Springer, 2020.

6 Javier Esparza, Stefan Kiefer, Jan Kretínský, and Maximilian Weininger. Enforcing ω-regular properties in markov chains by restarting. In *CONCUR*, volume 203 of *LIPIcs*, pages 5:1–5:22. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2021.

7 Yliès Falcone, Laurent Mounier, Jean-Claude Fernandez, and Jean-Luc Richier. Runtime enforcement monitors: composition, synthesis, and enforcement abilities. *Formal Methods Syst. Des.*, 38(3):223–262, 2011.

8 Yliès Falcone and Srinivas Pinisetty. On the runtime enforcement of timed properties. In *RV*, volume 11757 of *Lecture Notes in Computer Science*, pages 48–69. Springer, 2019.

9 Christian Hensel, Sebastian Junges, Joost-Pieter Katoen, Tim Quatmann, and Matthias Volk. The probabilistic model checker storm. *CoRR*, abs/2002.07080, 2020. `arXiv:2002.07080`.

10 Marta Z. Kwiatkowska, Gethin Norman, and David Parker. The PRISM benchmark suite. In *QEST*, pages 203–204. IEEE Computer Society, 2012.

11 Kim G. Larsen and Axel Legay. On the power of statistical model checking. In *ISoLA (2)*, volume 9953 of *Lecture Notes in Computer Science*, pages 843–862, 2016.
12 David Lee and Mihalis Yannakakis. Principles and methods of testing finite state machines-a survey. *Proc. IEEE*, 84(8):1090–1123, 1996.
13 Axel Legay, Benoît Delahaye, and Saddek Bensalem. Statistical model checking: An overview. In *RV*, volume 6418 of *Lecture Notes in Computer Science*, pages 122–135. Springer, 2010.
14 Jay Ligatti, Lujo Bauer, and David Walker. Run-time enforcement of nonsafety policies. *ACM Trans. Inf. Syst. Secur.*, 12(3):19:1–19:41, 2009.
15 Doron A. Peled, Moshe Y. Vardi, and Mihalis Yannakakis. Black box checking. *J. Autom. Lang. Comb.*, 7(2):225–246, 2002.
16 Fred B. Schneider. Enforceable security policies. *ACM Trans. Inf. Syst. Secur.*, 3(1):30–50, 2000.
17 Koushik Sen, Mahesh Viswanathan, and Gul Agha. Statistical model checking of black-box probabilistic systems. In *CAV*, pages 202–215, 2004. doi:10.1007/978-3-540-27813-9_16.
18 Koushik Sen, Mahesh Viswanathan, and Gul Agha. On statistical model checking of stochastic systems. In *CAV*, volume 3576 of *Lecture Notes in Computer Science*, pages 266–280. Springer, 2005.
19 Maximilian Weininger. Personal communication, 2022.
20 Håkan L. S. Younes. Probabilistic verification for “black-box” systems. In *CAV*, volume 3576 of *Lecture Notes in Computer Science*, pages 253–265. Springer, 2005.
21 Håkan L. S. Younes, Edmund M. Clarke, and Paolo Zuliani. Statistical verification of probabilistic properties with unbounded until. In *SBMF*, volume 6527 of *Lecture Notes in Computer Science*, pages 144–160. Springer, 2010.
22 Håkan L. S. Younes and Reid G. Simmons. Probabilistic verification of discrete event systems using acceptance sampling. In *CAV*, volume 2404 of *Lecture Notes in Computer Science*, pages 223–235. Springer, 2002.

A Notes

The appendix can be found in the extended version at `https://arxiv.org/abs/2303.03292`.