Adaptive Stopping Algorithms Based on Concentration Inequalities

Maxime Parmentier and Axel Legay

ICTEAM, UCLouvain, 1348 Ottignies-Louvain-la-Neuve, Belgique

Abstract. Sampling is a key step of stochastic methods. In some contexts, minimizing the sample size can be of critical importance and determine by itself the viability of various approaches. Rethinking the way samples are generated can even lead to new algorithms, better suited than their alternatives to tackle specific real-world problems for which the sampling task is a costly step of the estimation process. For instance, strategies of formal verification and statistical model checking can be greatly improved by the use of adaptive stopping algorithms. Instead of initially computing the necessary sample size, those algorithms generate the samples progressively while continuously monitoring their progress along the way, allowing them to stop the sampling process as soon as possible. We present a generalization of two existing adaptive stopping algorithms for statistical model checking, and we show how this generalization can be exploited to derive tailor-made variations for specific use cases.

Keywords: Sampling · Estimation methods · Formal Verification · Statistical Model Checking · Adaptive Stopping Algorithms

1 Introduction

Increasingly complex software and systems are being built and with them, the demand for reliable and powerful methods of verification only grows. Be it for medical robots, city-wide traffic light systems, space-exploring satellites or autonomous vehicles, failing to ensure a new design meets all its (safety) requirements can lead to enormous economic losses in the best cases, and to tragedies in the worst cases.

Testing is the most widespread approach to check the correctness of a system [8], but it suffers from many limitations. The number of tests are almost always very low with respect to the size of the exploration space, because of time and resource constraints. Moreover, designing relevant tests depends on *a priori* knowledge, which can be limited or insufficient. Lastly, the outcomes of those tests can be difficult to interpret and do not always provide a path to a solution when a problem is discovered.

Formal methods can also be used to prove the correctness of a system [3]. For instance, if an executable model can be built, one can go through every possible specification or execution of the system and check if all of them meet the requirements. However, because the size of a model grows exponentially with the number of variables which are used for the representation, the so-called state explosion problem [12], complete methods of model checking are not exploitable for real-cases applications.

Statistical model checking (SMC) offers an excellent trade-off between the feasibility of testing protocols and the completeness of model checking theory. By applying the ideas of formal methods to a well-generated sample, it is indeed possible in many situations to provide strong and sufficient statistical insight without having to perform an exhaustive analysis of the system.

Therefore, more and more efforts have been deployed over the last 30 years to develop new frameworks and algorithms of SMC [1,28,29]. From basic Monte Carlo methods [34] to more advanced techniques created to deal with specific issues of formal verification, such as the problem of how to deal with rare events [22], both estimation and optimization algorithms have been developed to expand the class of systems for which SMC can effectively be used.

UPPAAL [5] and PRISM [26] are two examples of powerful tools that implement some of those frameworks and algorithms. They have been used for the verification of systems as varied as gearbox controllers [31], lip synchronization algorithms [7], network protocols [27], or large scale rail signalling systems [4].

A common feature that most stochastic algorithms for model checking share is their sampling process. Given some user-specified parameters, often related to the precision and confidence level for the estimations, they try to guess *a priori* the size of the necessary sample. Unfortunately, either for the sake of simplicity or so that they can be as universal as possible, they will generally exploit very general theoretical results to compute that sample size, which leads to overshooting [1,13]. Moreover, independently of whether the sample is completely generated at the start or in multiple steps, its generation is made agnostically, without exploiting any knowledge related to or derived from the system.

However, neglecting the importance of the sampling process can have dire consequences for a SMC algorithm. If the system under scrutiny is so complex that any single simulation asks for extensive computations, or if the verification process must be performed "online", or if the resources allocated for the verification are purposefully limited (for instance within an embedded system), not minimizing the size of the sample which needs to be exploited by the algorithm can mean that a perfectly valid and efficient algorithm will not be viable in practice.

To tackle that problem, *adaptive stopping algorithms* proceed differently. Rather than initially generating a large enough sample, then computing their output, those estimation algorithms progressively grow their sample while continuously updating their estimation and checking if their stopping criterion allows them to stop. The ongoing progress of the adaptive stopping algorithm determines whether the stopping criterion is reached, which means the algorithm takes into account the knowledge already accumulated about the system at any point to decide if more simulations must be performed. That way, to the condition that the stopping criterion was proven to have the right properties, the size of the sample can be reduced to a minimum while ensuring that the output will have the necessary statistical guarantees.

The difficulty of designing new adaptive stopping algorithms resides in the derivation of new relevant stopping critera. Our main contribution is a generalization of the proofs of two existing adaptive stopping algorithms for SMC, both based on concentration inequalities, which leads to a versatile and unifying version of those two algorithms. Moreover, we illustrate how this generalization can be exploited within the setting of estimating probabilities associated to rare events. Finally, we analyze the performance of the algorithm thus obtained and show how it can outperform a state-of-the-art general algorithm which was proven to be (quasi) optimal.

2 Mathematical Background

For SMC, any quantity that can be obtained with a simulation of a given system can be seen as a real-valued *random variable*. The goal of SMC algorithms is generally either to evaluate or optimize those quantities that describe and determine the system. Estimation algorithms are those whose objective is to approximate the (expected) value of such a quantity, while optimization algorithms try to identify the behavior of the system which will minimize or maximize that quantity.

In some cases, that parameter or key performance indicator can be seen as the probability that a specific requirement of the system is verified during a random possible execution of the system, in which case the corresponding random variable is bounded in $[0;1]$ [37]. But it can represent more concrete physical quantities as well, such as the fuel consumption of an engine [39].

Adaptive stopping algorithm, also called *stopping (rule) algorithms* (SA), are estimation algorithms which can be exploited for SMC. Their goal is to compute an estimation of the value of a quantity of interest while minimizing the size of the necessary sample. More precisely, their goal is to compute (ϵ, δ)-estimations with a minimal sample size.

Definition 1.
Let X be a random variable with distribution d_X. Let p be a parameter of d_X. Given a precision $\epsilon > 0$ and a confidence level $\delta > 0$, an (ϵ,δ)-estimation *of p is an estimation $\hat{p}$ such that $P\left((1-\epsilon)p \leq \hat{p} \leq (1+\epsilon)p\right) \geq 1-\delta$.*

As a reminder, a *random variable* X is a measurable function from a probability space $(\Omega, \mathcal{F}, P)$, with Ω being the set of possible outcomes, to a measurable space E. For most practical applications, real-valued random variables are used, i.e. $E = \mathbb{R}$. A (real-valued) random variable X is entirely characterized by its *probability distribution*, which can be expressed with the *cumulative distribution function* of X: $\mathrm{CDF}_X(x) = P(X \leq x)$. The *expected value* of X is $\mu_X = E(X) = \int_\Omega X dP$, the *variance* of X is $V_X = V(X) = E[(X - E(X))^2]$, the *standard deviation* of X is $\sigma_X = \sigma(X) = \sqrt{V(X)}$.

Random variables can be used to formalize any random experiment, such as the generation of a possible specialization or execution of the model of a system in the context of SMC. In practice, the random variable X whose parameter needs to be estimated can be seen as any quantity associated with the system. The realizations of this random variable are individual values that the quantity X can take with different possible specifications or executions of the system, and one can obtain such realizations by running (independent) simulations of the system. Especially for SMC, the parameter of interest p is generally taken to be the expected value $\mu = E(X)$ of the random variable. In particular, it can correspond to the probability that a given property which needs to be verified holds true for a random execution of the system.

The notion of (ϵ,δ)-estimation is key for SMC algorithms. Indeed, since those algorithms do not perform exhaustive searches, their outputs must come with statistical guarantees: the precision ϵ of their output (how close the estimation should be to the true value) and the confidence associated with their output (how likely it is that the true value is at distance at most ϵ of the estimation). SMC algorithms always ask for those two parameters, either explicitly or indirectly.

Most SMC algorithms will compute their sample size *a priori* in function of ϵ and δ (*e.g.*, see [17]). A SA is instead built upon a *stopping criterion*, also called *stopping rule*. That stopping criterion, which depends on the chosen precision ϵ and confidence level δ, gives a different structure to adaptive stopping algorithms. In its most generic form (see Algorithm 1), such an algorithm is a single loop that iteratively checks if the stopping criterion has been reached, produces one individual sample, and then update the estimation and the stopping criterion.

Algorithm 1
Generic Adaptive Stopping Algorithm

Input: random variable X,
precision ϵ and confidence level δ
Output: (ϵ, δ)-estimation of $\mu = E(X)$

```
1: $t \leftarrow 0$
2: sample $\leftarrow \{\}$
3: while $\neg$ stopping criterion do
4:     $t \leftarrow t + 1$
5:     sample $\leftarrow$ sample $\cup \{x_t\}$
6:     $\hat{\mu}_t \leftarrow$ mean_estimator(sample)
7:     update stopping criterion
8: end while
9: return: $\hat{\mu}_t$
```

The stopping criterion depends on the precision ϵ and the confidence level δ, but also on the already accumulated sample at any step. This means that, if built properly, the stopping criterion can continuously take into account the knowledge which has been obtained about the system. Typically, the stopping criterion will for instance stay unverified for longer if the elements of the sample that have been produced up to a point have high variance.

Note that Algorithm 1 is fully agnostic with respect to how the samples are generated. Not only that means it can be applied to a large variety of problems, but also that it can be combined with any unique routine aimed at optimizing the process of generating a sample element and that parallelization can easily be implemented. Algorithm 1 is agnostic with respect to the actual estimation process (the estimator) as well.

We now give a brief description of the two adaptive stopping algorithms which led to our generalization.

2.1 AdaSelect and EBStop

The AdaSelect algorithm [16], essentially follows the structure of the generic Algorithm 1. It requires that X has finite range R and that $\mu = E(X) \neq 0$. It derives its stopping criterion from the *Hoeffding inequality*:

Theorem 1 (Hoeffding Inequality). *Let $X_1, ..., X_n$ be real-valued random variables that are independent and identically distributed (i.i.d.) to a random variable X with expected value $\mu = E(X)$, such that there exist $a \leq b$ with $P(a \leq X_i \leq b) = 1$ for all $i \in \{1, ..., n\}$.*

Then, for all $\epsilon > 0$: $P\left(\left|\frac{1}{n}\sum_{i=1}^{n} X_i - \mu\right| \geq \epsilon\right) \leq 2\exp\left(-\frac{2n\epsilon^2}{(b-a)^2}\right)$.

The derived stopping criterion is of the form $|\hat{\mu}_t| < c_t$, where $\hat{\mu}_t$ is the sample mean $(1/t)\sum_{i=1}^{t} x_i$ at step t, and c_t is computed as $(b-a)\sqrt{\frac{\log(t(t+1)/\delta)}{2t}}(1+1/\epsilon)$.

Moreover, the authors gave a probabilistic upper bound for the size of the sample generated by the improved version of their algorithm. If N is the size of the sample which is generated by the algorithm, they showed that there exists a constant $C > 0$ such that $P\left(N > C\frac{R^2}{\epsilon^2\mu^2}(\log(2/\delta) + \log(R/(\epsilon\mu))\right) < \delta$.

The basic form of the EBStop algorithm [32], one of the current state-of-the-art SAs, is similar to the AdaSelect algorithm. It also requires that X has finite range R and that $\mu = E(X) \neq 0$. It derives its stopping criterion from the *Bernstein inequality*:

Theorem 2 (Bernstein Inequality). *Let $X_1, ..., X_n$ be real-valued random variables that are independent and identically distributed (i.i.d.) to a random variable X with expected value $\mu = E(X)$ and variance $\sigma^2 = V(X)$, with $R > 0$ such that $P(|X - \mu| < R) = 1$.*

Then, for all $\epsilon > 0$: $P\left(\left|\frac{1}{n}\sum_{i=1}^{n} X_i - \mu\right| \geq \epsilon\right) \leq 2\exp\left(-\frac{n\epsilon^2}{\frac{2}{3}R\epsilon + 2\sigma^2}\right)$.

In the final version of EBStop, the stopping criterion is made more complex and symmetrical than in the case of AdaSelect. Nevertheless, it also relies on a sequence of numbers c_t with respect to which are compared the progressively better estimations of μ. If $\hat{\sigma}_t^2$ is the sample variance at step t, and if $(d_t)_{t\in\mathbb{N}}$ is a sequence such that $\Sigma_{t=1}^{\infty} d_t \leq \delta$, they can be computed as $\hat{\sigma}_t\sqrt{\frac{2\log(3/d_t)}{t}} + \frac{3R\log(3/d_t)}{t}$.

The authors showed that it is possible to prove quasi optimality for the EBStop algorithm, in the sense of the optimality which was proven for the $\mathcal{AA}$ algorithm, another state-of-the-art SA [14]. While $\mathcal{AA}$ is limited to the class of random variables which are nonnegative, they proved that if N is the size of the sample which is generated by $\mathcal{AA}$, and σ the standard deviation of X, there exists a constant $C > 0$ such that $P\left(N > C\max(\sigma^2, \epsilon\mu)(1/\epsilon^2\mu^2)\log(2/\delta)\right) < \delta$. The authors also showed that for any SA that can be applied to the same class of random variables, if N' is the size of the sample which is generated by the algorithm, there always exists a constant $C' > 0$ such that $P\left(N' < C'\max(\sigma^2, \epsilon\mu)(1/\epsilon^2\mu^2)\log(2/\delta)\right) < \delta$.

Therefore, for random variables which are bounded and nonnegative, any SA will probabilistically require a sample size that is at best smaller than the one which would have been produced by the $\mathcal{AA}$ algorithm by a fixed multiplicative factor. For the EBStop algorithm, the theoretical bound is slightly less tight than with the $\mathcal{AA}$ algorithm: if N is the size of the sample which is generated by the algorithm, there exists a constant $C > 0$ such that $P(N > C \max((\sigma^2/\epsilon^2\mu^2), R/(\epsilon|\mu|))(\log(1/\delta) + \log(R/(\epsilon|\mu|)))) < \delta$. This shows that the EBStop algorithm scales almost as well as the $\mathcal{AA}$ algorithm, without being limited in its application to nonnegative random variables. Moreover, it often does not significantly affect the actual performance of the EBStop algorithm in practice [14]. EBStop has been effectively exploited to solve subset selection problems [9], policy search problems [18], and improve reinforcement learning strategies [19].

2.2 Concentration Inequalities

The statistical results on which AdaSelect and EBStop are based are what is called concentration inequalities. A *concentration inequality* for a random variable X is a bound on the probability that realizations of X are within a chosen distance of a specific value. There exists a multitude of useful and well-known concentration inequalities which can be applied to large classes of random variables, among which are Markov's inequality, Chebyshev's inequality, and the Chernoff bound (see appendix for more details).

A particularly important collection of concentration inequalities for sampling and estimation algorithms are those that are specific to random variables $S_n = X_1 + ... + X_n$ which can be decomposed as the sum of n independent random variables, such as the random variables that follow a binomial distribution. The Hoeffding and the Bernstein inequalities belong to that group, among others such as Azuma's inequality, McDiamid's inequality or Benett's inequality.

For statistical model checking methods, the sampling can generally be assumed to be independent and always be performed following a constant distribution: simulations are executed independently of each other and the model's behaviour does not change from one simulation to another (nondeterminism put aside). If the random variables X_1, ..., X_n are independent and identically distributed (i.i.d.), additional bounds can be derived, for instance from the central limit theorem. Moreover, most classic concentration inequalities can be simplified and enhanced under the assumption that the random variables X_1, ..., X_n are i.i.d.. For those reasons, we focus on that setting and give the following definition.

Definition 2 (Concentration Inequality Function). *Let $X_1, ..., X_n$ be real-valued random variables that are independent and identically distributed (i.i.d.) to a random variable X. A function $F_X : \mathbb{N}_0^+ \times \mathbb{R}_0^+ \rightarrow \mathbb{R}$ such that, for all $n \in \mathbb{N}_0$ and for all $\epsilon > 0$:*

$$P\left(\left|\sum_{i=1}^{n} X_i - E\left(\sum_{i=1}^{n} X_i\right)\right| \geq \epsilon\right) \leq F_X(n, \epsilon) \tag{1}$$

is called a concentration inequality function *for X.*

Proposition 1 links the notions of concentration inequality function and (ϵ, δ)-estimation.

Proposition 1. *Let X be a random variable of expected value $\mu = E(X)$. Let $X_1, ..., X_n$ be random variables i.i.d to X. Let F_X be a concentration inequality function for X. Then, for all $n \in \mathbb{N}_0$ and for all $\epsilon > 0$:*

$$P\left(\left|\frac{1}{n}\sum_{i=1}^{n} X_i - \mu\right| \geq \epsilon\right) \leq F_X(n, n\epsilon)$$

Proof. Equation (1) with $\epsilon^* = n\epsilon$ gives:

$$P\left(\left|\sum_{i=1}^{n} X_i - E\left(\sum_{i=1}^{n} X_i\right)\right| \geq n\epsilon\right) \leq F_X(n, n\epsilon)$$

However, since $X_1, ..., X_n$ are i.i.d. to X:

$$P\left(\left|\sum_{i=1}^{n} X_i - E\left(\sum_{i=1}^{n} X_i\right)\right| \geq n\epsilon\right) =$$

$$P\left(\left|\frac{1}{n}\sum_{i=1}^{n} X_i - \frac{1}{n}\sum_{i=1}^{n} E(X_i)\right| \geq \epsilon\right) =$$

$$P\left(\left|\frac{1}{n}\sum_{i=1}^{n} X_i - \frac{n\mu}{n}\right| \geq \epsilon\right)$$

3 From Concentration Inequalities to Stopping Algorithms

The intuition behind our generalization of the AdaSelect and EBstop algorithms is the following. A concentration inequality that is valid for a random variable X (or a specific class of random variables) gives a bound to the probability that a sample mean deviates from the expected value of X by a quantity larger than $\epsilon > 0$ (Proposition 1). This means that for any ϵ, someone performing a Monte-Carlo estimation of $\mu = E(X)$ can directly know how likely it is that the estimation does not possess the required precision, *i.e.* the confidence level of the estimation, in function of the size of the sample which was used. Therefore, to obtain an (ϵ, δ)-approximation of μ, one can successively produce (new and independent) larger and larger samples, hoping that the values of the bounds eventually become smaller than the chosen δ. With the Hoeffding's inequality and the Bernstein's inequality, both of the corresponding concentration inequality functions, $F(n, \epsilon) = 2\exp\left(-\frac{2\epsilon^2}{\sum_{i=1}^{n}(b_i - a_i)^2}\right)$ and $F(n, \epsilon) = 2\exp\left(-\frac{\epsilon^2}{2V(S_n) + (2/3)\epsilon R}\right)$, are such that $F(n, n\epsilon)$ is (strictly) decreasing with respect to n and such that $\lim_{n\to\infty} F(n, n\epsilon) = 0$ (for any fixed ϵ). Therefore, with those two concentration inequality functions, for any δ, this process of repeating Monte Carlo estimations with larger and larger samples will always eventually terminate.

Of course, such a strategy is not only impractical, it is extremely inefficient. Therefore, instead of blindly trying larger and larger sample sizes, we consider the inverse problem of identifying a sequence of decreasing precision levels "ϵ_t" with the property that the series of the sequence $F(t, t\epsilon_t)$ reaches δ (for the sake of readability and to avoid confusion, we'll use the symbol c_t instead of ϵ_t thereafter). Once found, that sequence ensures that for any cumulative sampling with sample mean μ_t, the event $\mathcal{E} = \{|\mu_t - \mu| \leq c_t \text{ for all } t \in \mathbb{N}_0\}$ happens with a probability $1 - \delta$. The key and difficult step is then to design a stopping criterion based on those parameters c_i, one that will only put an end to the sampling process if the current sample mean μ_t is an (ϵ, δ)-estimation of μ (as soon as it is the case, and not before), given that $\mathcal{E}$ holds.

3.1 Generalized Stopping Algorithm

Our main result shows how to systematically turn a decreasing sequence of precision levels $\{c_t\}_{t\in\mathbb{N}_0}$ with the right properties into a valid stopping criterion for a given concentration inequality function.

Theorem 3. *Let X be a real-valued random variable with $\mu = E(X) \neq 0$, and let F_X be a concentration inequality function for X. Let be $\epsilon > 0$ and $\delta > 0$.*

If there exist a sequence $(c_t)_{t\in\mathbb{N}_0}$ that is decreasing, with $\lim_{t\to\infty} c_t = 0$, and such that for all $t \in \mathbb{N}_0$:

$$F(t, tc_t) = \frac{\delta}{t(t+1)}$$

Then Algorithm 2, $GSA(X, \epsilon, \delta)$, outputs an (ϵ, δ)-estimation of μ, and does so with a number of steps n that is minimal with probability $1 - \delta$.

Algorithm 2 Generalized Stopping Algorithm

Input:
random variable X, precision ϵ and confidence level δ
Output:
$\hat{\mu}$, an (ϵ, δ)-estimation of $\mu = E(X)$

```
1: $t \leftarrow 0$
2: sample $\leftarrow \{\}$
3: while $\neg\left(|\hat{\mu}_t| \geq c_t(\frac{1}{\epsilon} + 1)\right)$ do
4:     $t \leftarrow t + 1$
5:     sample $\leftarrow$ sample $\cup \{x_t\}$
6:     $\hat{\mu}_t \leftarrow \frac{1}{t}\sum_{i=1}^{t} x_i$
7:     update $c_t$
8: end while
9: return: $\hat{\mu}_t$
```

Proof. To shorten the proof and lighten the notations, we make the unneeded assumption that X is nonnegative.

The general case just involves additional steps because of necessary considerations with absolute values.

We have to bound the sum of the two following probabilities by δ:

1. The probability that the stopping criterion is reached at a step t while μ_t is not at a distance of at most $\mu\epsilon$ of μ, *i.e.* the probability of a false positive.
2. The probability that the stopping criterion is **not** reached at a step t while μ_t **is** at a distance of at most $\mu\epsilon$ of μ, *i.e.* the probability of a false negative.

Let n be the smallest integer such that:

$$c_{n-1} > \mu\epsilon \quad \text{and} \quad c_n \leq \mu\epsilon$$

$$(c_{n-1} > |\mu|\epsilon \text{ and } c_n \leq |\mu|\epsilon \text{ in the general case})$$

We want to show that the algorithm ends at step n with high probability. Note that the integer n has to exist since the sequence $(c_t)_{t\in\mathbb{N}_0}$ is decreasing, $\lim_{t\to\infty} c_t = 0$, and $\mu \neq 0$. Keep in mind that those two inequalities imply that $\frac{c_{n-1}}{\epsilon} > \mu$ and $\frac{c_n}{\epsilon} \leq \mu$.

1. First, let us show that the probability of a false postive, P_{f^+}, is smaller than δ. First, note that if P_t is the probability that the stopping criterion is reached at step t, we have:

$$P_{f^+} \leq \sum_{1\leq t<n} P_t$$

By the definition of the stopping criterion of the algorithm:

$$P_t \leq P\left(\frac{1}{t}\sum_{i=1}^{t} X_i \geq c_t\left(\frac{1}{\epsilon}+1\right)\right)$$

However, with $S_t = \sum_{i=1}^{t} X_i$:

$$P\left(\frac{S_t}{t} \geq c_t\left(\frac{1}{\epsilon}+1\right)\right) = P\left(\frac{S_t}{t} \geq c_t\left(\frac{1}{\epsilon}\right)+c_t\right)$$

Since $c_1 \geq ... \geq c_t \geq ... \geq c_{n-1} \geq \mu\epsilon$:

$$P\left(\frac{S_t}{t} \geq c_t\left(\frac{1}{\epsilon}+1\right)\right) \leq P\left(\frac{S_t}{t} \geq \mu\epsilon\left(\frac{1}{\epsilon}\right)+c_t\right)$$

Therefore:

$$P\left(\frac{S_t}{t} \geq c_t\left(\frac{1}{\epsilon}+1\right)\right) \leq P\left(\frac{S_t}{t} \geq \mu+c_t\right)$$

In conclusion:

$$P\left(\frac{S_t}{t} \geq c_t\left(\frac{1}{\epsilon}+1\right)\right) \leq P\left(\frac{S_t}{t} - \mu \geq c_t\right)$$

Now, remember that F_X is a concentration inequality function for X (Proposition 1):

$$P_{f^+} \leq \sum_{1\leq t<n} P_t \leq \sum_{1\leq t<n} P\left(\frac{1}{t}\sum_{i=1}^{t} X_i - \mu > c_t\right)$$
$$\leq \sum_{1\leq t<n} F(t, tc_t) \leq \sum_{1\leq t<n} \frac{\delta}{t(t+1)}$$

Finally:

$$P_{f^+} \leq \sum_{1\leq t<n} \frac{\delta}{t(t+1)} \leq \sum_{1\leq t} \frac{\delta}{t(t+1)} \leq \delta\left(1-\frac{1}{n}\right)$$

2. Now, let us show that the probability of a false negative, P_{f^-}, is smaller than δ. Because of the definition of the stopping criterion, we have:

$$P_{f_-} \leq P\left(\frac{S_n}{n} \leq c_n\left(\frac{1}{\epsilon}+1\right)\right)$$

However:

$$P\left(\frac{S_n}{n} \leq c_n\left(\frac{1}{\epsilon}+1\right)\right) \leq P\left(\frac{S_n}{n} \leq \frac{c_n}{\epsilon}+c_n\right)$$
$$\leq P\left(\frac{S_n}{n} \leq \mu + c_n\right) \leq P\left(\frac{S_n}{n} - \mu \leq c_n\right)$$

Since F_X is a concentration inequality function for X:

$$P_{f_-} \leq P\left(\frac{S_n}{n} - \mu \leq c_n\right) \leq F(n, nc_n) \leq \frac{\delta}{n(n+1)}$$

3. In conclusion, if $P_{f^+_-}$ is the probability to have a false positive or a false negative:

$$P_{f^+_-} \leq P_{f^+} + P_{f_-} \leq \delta\left(1-\frac{1}{n}\right) + \frac{\delta}{n(n+1)} \leq \delta$$

From the proof of the second claim of Theorem 3 directly follows Corollary 1, which is the generalization of the corresponding results bounding the size of the samples produced by AdaSelect and EBStop mentioned in Section 2.1.

Corollary 1. *With the setting of Theorem 3, the GSA algorithm requires a sample size of at most n with probability $1-\delta$, with n being the smallest integer such that $c_n \leq \mu\epsilon$.*

Given a concentration inequality function, it is relatively easy to find a sequence $(c_t)_{t\in\mathbb{N}_0}$ which verifies the conditions of Theorem 3 to generate a specialized version of the GSA algorithm. As hinted by Proposition 1, it generally suffices to inverse (with respect to ϵ) the function $F(t, t\epsilon)$ around its image $\delta/(t(t+1))$, for all $t \in \mathbb{N}_0$. For instance, with Hoeffding's inequality, this is a straightforward computation which can even be completed symbolically: with R

the range of X, $\exp\left(-2(tc_t)^2/tR^2\right) = \delta/(2t(t+1))$. If we isolate c_t, we obtain the formula $c_t = R\sqrt{((1/2t)\ln((t(t+1))/\delta))}$, which is exactly the formula for the parameters of the stopping criterion of AdaSelect.

When an explicit expression for the parameters c_t can be derived, Corollary 1 can additionally be used to provide an explicit bound on the size of the generated samples, similar to those given for AdaSelect and EBStop in Section 2.1. However, having access to an explicit formula for those parameters is not absolutely needed to make use of the GSA algorithm, for they can always be computed numerically if needed (*e.g.* this would be necessary to find the parameters c_t associated with Benett's inequality). Moreover, the computation of those parameters can be performed in parallel to the generation of the simulations. Furthermore, once they have been determined (up to a certain point), they can be stored and reused afterwards (even with other values for the precision ϵ, as they only depend on δ and the properties of X).

Lastly, depending on the concentration inequality function, it might happen that the functions $F(t, t\epsilon)$ are only decreasing bijections (with respect to ϵ, for a fixed t) when restricted to a small enough interval around 0, and only for large enough t. In that case, the sequence of parameters c_t might need to start at an index $t_* > 1$ rather than with index 1. However, this purely technical problem can only arise when considering very large ϵ and computing estimations which require very small sample sizes. Initializing the SA with a sample of size t_* (rather than 1) and starting with the t_*-th iteration of the main loop is the easiest technical solution to that problem. Its only downside is to make the algorithm slightly overshoot in the case of the most trivial estimations.

3.2 Geometric Sampling and Bilateral Testing

The basic Algorithm 2 can be improved and optimized in the same way that the basic AdaSelect and the basic EBStop algorithms can be upgraded to the geometric AdaSelect and the EBGStop algorithms. The underlying optimization is based on the idea that checking if the stopping criterion has been reached at every step of the execution of the algorithm is rarely useful, and serves no purpose as long as that stopping criterion is far from being verified. A geometric progression in how the sample is grown can instead be used, with $\lceil g^t \rceil$ elements added at step t, for some $g > 1$. On one hand, choosing a large geometric parameter g can significantly speed up the algorithm, especially when large samples are needed and simulation times are low. Moreover, it can also significantly lower the number of parameters c_t which needs to be computed. On the other hand, it can have the undesirable consequence of forcing the algorithm to generate slightly too many samples, by an order of the multiplicative factor g.

Additionally, a bilateral reformulation of the stopping criterion of Algorithm 2, similar to the bilateral reformulation of the stopping criterion of EBStop can be implemented. With $\mathrm{LB}_t = \max(0, \max_{1\leq s\leq t}(|\mu_s| - c_s))$ and $\mathrm{UB}_t = \min_{1\leq s\leq t}(|\mu_s| + c_s)$, the stopping criterion can be rewritten as $(1+\epsilon)\mathrm{LB}_t \geq (1-\epsilon)\mathrm{UB}_t$ (see [32] for details). This can make it easier to deal with nonnegative random variables.

4 Experiments

To analyze the applicability and the performance of GSA (Algorithm 2), we decided to focus on the setting of estimating "extreme" probabilities, *i.e.* probabilities very close to 0 or 1. This is called the rare event problem.

In SMC, a common strategy to check whether a system verifies one of its requirements is to estimate the probability p that the property will hold true for a random possible execution or specification of the system. This can be done, for instance, by representing the system as a Markov decision process and the property as a formula of a variant of temporal logic [13,15,20,35].

Since only the success or the failure of the system for each simulation matters, the complexity of the (model of) the system does not matter for any estimation algorithm whose operation is independent of the way each of those simulations is performed and deemed a success or a failure. The verification task is thus reduced to the estimation of the expected value of a random variable X which follows a Bernoulli distribution. More precisely, we focused on the task of estimating $p = \mu = E(X)$ with $X \simeq \mathcal{B}(p)$, with absolute precision ϵ and a confidence level of δ.

The rare event problem has already been widely studied over the years. Importance sampling [23,25] and importance splitting [22,24,25,36] are the two classic approaches. Importance sampling corresponds to the modification of the distribution of the random variable whose expected value must be estimated, *i.e.* the introduction of a bias towards successful or failing simulations that is compensated later on. Importance splitting corresponds to the reformulation of the property which needs to be verified into several more likely properties, so that the probability of the rare event can eventually be estimated as the product of their intermediate probabilities.

As SAs are nothing but statistical algorithms that minimize the sample size without trying to optimize how the samples are produced or the parameters are estimated, we do not expect what we expose below to outperform those advanced strategies by itself. However, specifically because SAs are agnostic with respect to those processes, they could actually be combined with importance sampling and importance splitting to improve their effectiveness.

4.1 A Stopping Algorithm for Bernoulli Distributions

We compared a basic Monte Carlo approach and a standard sample size estimation method used in multiple SMC tools with AdaSelect, EBStop and three variants of GSA: GSA$_H$, GSA$_B$ and GSA$_C$. Those three algorithms were respectively obtained from Hoeffding's inequality, Bernstein's inequality and a specialized version of the Chernoff bound.

The Monte Carlo algorithm MC uses Chebyshev's inequality to determine *a priori* the sample size it needs to compute an estimation of p with an absolute precision ϵ and a confidence level δ. For a sample of size n, Chebyshev's inequality ensures that $P(|S_n/n - E(S_n/n)| \geq \epsilon) \leq \frac{V(S_n/n)}{\epsilon^2}$. Since X follows a Bernoulli distribution, $V(X) \leq 1/4$. Therefore, as $V(S_n/n) \leq 1/4n$, a naive guess for the minimal sample size that is required for the estimation is the smallest integer n such that $1/4n\epsilon^2 \leq \delta$, *i.e.* such that $1/4\delta\epsilon^2 \leq n$.

For a less naive point of comparison, we considered the sample size estimation formula that is derived from a concentration inequality which was designed for Bernoulli distributions [33]: $n \geq \ln\left(\frac{2}{\delta}\right)/(2\epsilon^2)$. It is for instance implemented in PRISM as the APMC (Approximate Probabilistic Model Checking) method [21] and used in PLASMA (Platform for Learning and Advanced Statistical Model checking Algorithms) [30] for the Smart Sampling algorithm [17,35].

GSA$_H$, GSA$_B$ are theoretically equivalent to AdaSelect and EBStop. To find a true challenger, we therefore needed to start with a concentration inequality that holds for the random variables which follow a Bernoulli distribution specifically, ideally one which depends on the variance. It turns out that the Chernoff bound can be specialized and simplified for that class of simple random variables, in multiple ways [11,17,33]. In particular, we used the following version [11] to build GSA$_C$: if $X_1, ..., X_n$ are i.i.d to X with $0 \leq |X| \leq 1$, then for any $0 \leq k \leq 2\sigma_n$: $P\left(|S_n - E(S_n)| \geq k\sigma_n\right) \leq 2\exp(-k^2/4)$ (with $S_n = X_1 + ... + X_n$) and $\sigma_n = \sigma(S_n)$). For any $\epsilon \leq 2(\sigma_n)^2$, this gives us the concentration inequality:

$$P\left(|S_n - E(S_n)| \geq \epsilon\right) \leq 2\exp(-\epsilon^2/4(\sigma_n)^2)$$

Theorem 3 applied with the associated concentration inequality function produces a stopping algorithm GSA$_C$ whose parameters c_t are of the form $(2\sigma_t/t)\sqrt{\ln(t(t+1)/\delta)}$, *i.e.* $(2\sigma(X_t)/\sqrt{t})\sqrt{\ln(t(t+1)/\delta)}$. This shows that GSA$_C$ can take advantage of a low variance like GSA$_B$ and EBStop, but its parameters decrease at a faster rate.

Finally, we added small optimizations to the implementations of the SAs. First, as the goal is to estimate extreme probabilities, we force the algorithms to generate simulations at the start until they can initialize with a sample containing at least one success and one failure. Second, we estimate in parallel $E(X)$ and $E(1-X)$ to deal with probabilities close to 0 just as well as with probabilities close to 1 without making any assumption about the extreme nature of p. This is justified by the fact that, if $\hat{p}$ is an (ϵ, δ)-estimation of p, since $p \leq 1$, we automatically have $P\left(|p - \hat{p}| < \epsilon\right) \geq 1 - \delta$.

4.2 Results

Table 1, Table 2 and Table 3 show the order of magnitude of the (average) sample sizes for six (ϵ, δ) configurations with ϵ, δ and p ranging from 10^{-3} to 10^{-6} (and down to 10^{-9} for δ). As the results for GSA$_H$ and GSA$_B$ were very similar to those of AdaSelect and EBStop, we do not report them.

Table 1. Average sample sizes for MC, AdaSelect (GSA_H), EBStop (GSA_B) and GSA_C $\delta = 0.001$, $p = 0.001$

ϵ	MC	APMC	AdaSelect	EBStop	GSA_C
10^{-3}	$2.5 \cdot 10^8$	$3.8 \cdot 10^6$	$2 \cdot 10^7$	$2.3 \cdot 10^5$	$1.2 \cdot 10^5$
10^{-4}	$2.5 \cdot 10^{10}$	$3.8 \cdot 10^8$	$2.5 \cdot 10^9$	$1 \cdot 10^7$	$1.6 \cdot 10^7$
10^{-5}	$2.5 \cdot 10^{12}$	$3.8 \cdot 10^{10}$	$> 10^{10}$	$1 \cdot 10^9$	$1.7 \cdot 10^9$

Table 2. Average sample sizes for MC, AdaSelect (GSA_H), EBStop (GSA_B) and GSA_C $\epsilon = 0.001$, $p = 0.001$

ϵ	MC	APMC	AdaSelect	EBStop	GSA_C
10^{-3}	$2.5 \cdot 10^8$	$3.8 \cdot 10^6$	$2 \cdot 10^7$	$2.3 \cdot 10^5$	$1.2 \cdot 10^5$
10^{-6}	$2.5 \cdot 10^{11}$	$7.2 \cdot 10^6$	$2.4 \cdot 10^7$	$2.7 \cdot 10^5$	$1.4 \cdot 10^5$
10^{-9}	$2.5 \cdot 10^{14}$	$1 \cdot 10^7$	$2.8 \cdot 10^7$	$3 \cdot 10^5$	$1.9 \cdot 10^5$

Table 3. Average sample sizes for MC, AdaSelect (GSA_H), EBStop (GSA_B) and GSA_C $x = \epsilon = \delta = p$

ϵ	MC	APMC	AdaSelect	EBStop	GSA_C
10^{-3}	$2.5 \cdot 10^8$	$3.8 \cdot 10^6$	$2 \cdot 10^7$	$2.3 \cdot 10^5$	$1.2 \cdot 10^5$
10^{-4}	$2.5 \cdot 10^{11}$	$5 \cdot 10^8$	$2.6 \cdot 10^9$	$2.5 \cdot 10^6$	$1.5 \cdot 10^6$
10^{-5}	$2.5 \cdot 10^{14}$	$6.1 \cdot 10^{10}$	$> 10^{10}$	$2.3 \cdot 10^7$	$1.7 \cdot 10^7$
10^{-6}	$2.5 \cdot 10^{17}$	$7.3 \cdot 10^{12}$	$> 10^{11}$	$3.5 \cdot 10^8$	$2 \cdot 10^8$

The SAs scale proportionally to $1/\epsilon$ and proportionally to $\ln(1/\delta)$. This is consistent with the theoretical bounds for AdaSelect and EBStop (see Section 2.1) and the structure of the stopping criterion of Algorithm 2. All SAs require much smaller samples than the Monte Carlo method. AdaSelect clearly suffers from not being able to exploit the small variances. GSA_C outperforms EBStop in all cases, except when the probability to estimate (and thus the variance) is relatively large with respect to ϵ. When it matters the most however, with very small ϵ, δ and p, the reduction in sample size that GSA_C offers is substantial (-43%). The parallelization of the two algorithms appears to be the optimal strategy.

5 Conclusion and Future Work

We showed how stopping algorithms can be valuable assets for SMC by minimizing the size of the samples which are needed for the estimation of parameters. The generalization of the AdaSelect and EBStop algorithms allowed us to provide a systematic method to generate tailor-made stopping algorithms from concentration inequalities. Our main contribution, Theorem 3, can be exploited to improve the efficiency of SMC methods for specific settings, such as the rare event problem.

We see multiple possible paths of exploration to expand this work. From a theoretical perspective, improving the structure of Algorithm 2 and optimizing its geometric version seems to be a promising prospect. If the result used to minimize the drawback of geometric sampling in the case of EBStop [2] could be generalized, this could improve the performance of the geometric version of GSA. Theorem 3 could also maybe be duplicated for the estimation of the variance of random variables, rather than their expected value. The Efron–Stein inequality could be a good starting point.

In terms of application, we want to specialize the GSA algorithm for more ambitious classes of random variables, again hoping that those specialized SAs could outclass the generally (quasi) optimal algorithms $\mathcal{AA}$ and EBStop. For instance, if the quantity X which needs to be estimated is itself the outcome of a repeated process inside a system, a normal distribution could be assumed for X, potentially allowing for stronger results. Even more ambitiously, we want to determine whether it would be possible to automatize to process of deriving a system specific concentration inequality directly from the structure of the model of that system, which would result in a tool able to produce tailor-made SAs for individual systems without any input needed from the user.

Acknowledgements

M. Parmentier is funded by a FNRS PhD Grant and by the UCLouvain.
A. Legay is funded by a FNRS PDR - T013721.

References

1. Agha, G., Palmskog, K.: A survey of statistical model checking. ACM Transactions on Modeling and Computer Simulation (TOMACS) **28**(1), 1–39 (2018)
2. Audibert, J.Y., Munos, R., Szepesvári, C.: Variance estimates and exploration function in multi-armed bandit. In: CERTIS Research Report 07–31. Citeseer (2007)
3. Baier, C., Katoen, J.P.: Principles of model checking. MIT press (2008)
4. Basile, D., ter Beek, M.H., Ferrari, A., Legay, A.: Exploring the ertms/etcs full moving block specification: an experience with formal methods. International Journal on Software Tools for Technology Transfer **24**(3), 351–370 (2022)
5. Behrmann, G., David, A., Larsen, K.G.: A tutorial on uppaal. Formal methods for the design of real-time systems pp. 200–236 (2004)
6. Boucheron, S., Lugosi, G., Bousquet, O.: Concentration inequalities. In: Summer school on machine learning, pp. 208–240. Springer (2003)
7. Bowman, H., Faconti, G., Katoen, J.P., Latella, D., Massink, M.: Automatic verification of a lip-synchronisation protocol using uppaal. Formal Aspects of Computing **10**, 550–575 (1998)
8. Broy, M., Jonsson, B., Katoen, J.P., Leucker, M., Pretschner, A.: Model-based testing of reactive systems. In: Volume 3472 of Springer LNCS. Springer (2005)
9. Busa-Fekete, R., Szorenyi, B., Cheng, W., Weng, P., Hüllermeier, E.: Top-k selection based on adaptive sampling of noisy preferences. In: International Conference on Machine Learning. pp. 1094–1102. PMLR (2013)

10. Casella, G., Berger, R.L.: Statistical inference. Cengage Learning (2021)
11. Chung Graham, F., Lu, L.: Complex graphs and networks american mathematical society (2006)
12. Clarke, E.M., Klieber, W., Nováček, M., Zuliani, P.: Model checking and the state explosion problem. In: LASER Summer School on Software Engineering, pp. 1–30. Springer (2011)
13. Clarke, E.M., Zuliani, P.: Statistical model checking for cyber-physical systems. In: International symposium on automated technology for verification and analysis. pp. 1–12. Springer (2011)
14. Dagum, P., Karp, R., Luby, M., Ross, S.: An optimal algorithm for Monte Carlo estimation. SIAM Journal on computing **29**(5), 1484–1496 (2000)
15. Ding, X., Smith, S.L., Belta, C., Rus, D.: Optimal control of Markov decision processes with linear temporal logic constraints. IEEE Transactions on Automatic Control **59**(5), 1244–1257 (2014)
16. Domingo, C., Gavalda, R., Watanabe, O.: Adaptive sampling methods for scaling up knowledge discovery algorithms. Data Mining and Knowledge Discovery **6**, 131–152 (2002)
17. D'Argenio, P., Legay, A., Sedwards, S., Traonouez, L.M.: Smart sampling for lightweight verification of Markov decision processes. International Journal on Software Tools for Technology Transfer **17**, 469–484 (2015)
18. Heidrich-Meisner, V., Igel, C.: Hoeffding and Bernstein races for selecting policies in evolutionary direct policy search. In: Proceedings of the 26th Annual International Conference on Machine Learning. pp. 401–408 (2009)
19. Heidrich-Meisner, V., Igel, C.: Neuroevolution strategies for episodic reinforcement learning. Journal of Algorithms **64**(4), 152–168 (2009)
20. Henriques, D., Martins, J.G., Zuliani, P., Platzer, A., Clarke, E.M.: Statistical model checking for Markov decision processes. In: 2012 Ninth international conference on quantitative evaluation of systems. pp. 84–93. IEEE (2012)
21. Hérault, T., Lassaigne, R., Magniette, F., Peyronnet, S.: Approximate probabilistic model checking. In: Verification, Model Checking, and Abstract Interpretation: 5th International Conference, VMCAI 2004 Venice, Italy, January 11-13, 2004 Proceedings 5. pp. 73–84. Springer (2004)
22. Jegourel, C., Legay, A., Sedwards, S.: Importance splitting for statistical model checking rare properties. In: Computer Aided Verification: 25th International Conference, CAV 2013, Saint Petersburg, Russia, July 13-19, 2013. Proceedings 25. pp. 576–591. Springer (2013)
23. Kahn, H.: Random sampling (Monte Carlo) techniques in neutron attenuation problems. i. Nucleonics (US) Ceased publication **6**(See also NSA 3-990) (1950)
24. Kahn, H., Harris, T.E.: Estimation of particle transmission by random sampling. National Bureau of Standards applied mathematics series **12**, 27–30 (1951)
25. Kahn, H., Marshall, A.W.: Methods of reducing sample size in Monte Carlo computations. Journal of the Operations Research Society of America **1**(5), 263–278 (1953)
26. Kwiatkowska, M., Norman, G., Parker, D.: Prism: Probabilistic model checking for performance and reliability analysis. ACM SIGMETRICS Performance Evaluation Review **36**(4), 40–45 (2009)
27. Kwiatkowska, M., Norman, G., Sproston, J., Wang, F.: Symbolic model checking for probabilistic timed automata. Information and Computation **205**(7), 1027–1077 (2007)

28. Larsen, K.G., Legay, A.: Statistical model checking: past, present, and future. In: Leveraging Applications of Formal Methods, Verification and Validation: Foundational Techniques: 7th International Symposium, ISoLA 2016, Imperial, Corfu, Greece, October 10–14, 2016, Proceedings, Part I 7. pp. 3–15. Springer (2016)
29. Legay, A., Lukina, A., Traonouez, L.M., Yang, J., Smolka, S.A., Grosu, R.: Statistical model checking. In: Computing and software science: state of the art and perspectives, pp. 478–504. Springer (2019)
30. Legay, A., Sedwards, S., Traonouez, L.M.: Plasma lab: a modular statistical model checking platform. In: International Symposium on Leveraging Applications of Formal Methods. pp. 77–93. Springer (2016)
31. Lindahl, M., Pettersson, P., Yi, W.: Formal design and analysis of a gear controller. In: International Conference on Tools and Algorithms for the Construction and Analysis of Systems. pp. 281–297. Springer (1998)
32. Mnih, V., Szepesvári, C., Audibert, J.Y.: Empirical Bernstein stopping. In: Proceedings of the 25th international conference on Machine learning. pp. 672–679 (2008)
33. Okamoto, M.: Some inequalities relating to the partial sum of binomial probabilities. Annals of the institute of Statistical Mathematics **10**, 29–35 (1959)
34. Pappagallo, A., Massini, A., Tronci, E.: Monte Carlo based statistical model checking of cyber-physical systems: A review. Information **11**(12), 588 (2020)
35. Parmentier, M., Legay, A., Chenoy, F.: Optimized smart sampling. In: International Conference on Bridging the Gap between AI and Reality. pp. 171–187. Springer (2023)
36. Rosenbluth, M.N., Rosenbluth, A.W.: Monte Carlo calculation of the average extension of molecular chains. The Journal of Chemical Physics **23**(2), 356–359 (1955)
37. Sen, K., Viswanathan, M., Agha, G.: On statistical model checking of stochastic systems. In: Computer Aided Verification: 17th International Conference, CAV 2005, Edinburgh, Scotland, UK, July 6-10, 2005. Proceedings 17. pp. 266–280. Springer (2005)
38. Wasserman, L.: All of statistics: a concise course in statistical inference. Springer Science & Business Media (2004)
39. Zuliani, P., Platzer, A., Clarke, E.M.: Bayesian statistical model checking with application to simulink/stateflow verification. In: Proceedings of the 13th ACM international conference on Hybrid systems: computation and control. pp. 243–252 (2010)

Appendix: Concentration Inequalities

This is a nonexhaustive list of some of the most well-known and useful concentration inequalities. Most of them are regarded as relatively elementary results in statistics [6,38,10].

Let X be a random variable.

- Markov's inequality (X must be almost surely nonnegative):

$$\forall \epsilon > 0 : P(X \geq \epsilon) \leq \frac{E(X)}{\epsilon}$$

- Chebyshev's inequality ($E(X)$ and $V(X)$ must be finite):

$$\forall \epsilon > 0 : P(|X - E(X)| \geq \epsilon) \leq \frac{V(X)}{\epsilon^2}$$

- Chernoff bound (the moment generating function of X, $M_X(t) = E(e^{tX})$, must be well-defined and finite):

$$\forall \epsilon > 0 : P(X \geq \epsilon) \leq \frac{E(e^{tX})}{e^{t\epsilon}}$$

Now let $X = S_n = X_1 + ... + X_n$ be the sum of n independent random variables, with each X_i almost surely bounded within $[a_i, b_i]$. Let $R_i = b_i - a_i$ be the range of X_i. Let be $R = \max\limits_{1 \leq i \leq n} (R_i)$.

- Hoeffding's inequality:

$$\forall \epsilon > 0 : P(|S_n - E(S_n)| \geq \epsilon) \leq 2 \exp\left(-\frac{2\epsilon^2}{\sum_{i=1}^n (b_i - a_i)^2}\right)$$

- Bernstein's inequality:

$$\forall \epsilon > 0 : P(|S_n - E(S_n)| \geq \epsilon) \leq 2 \exp\left(-\frac{\epsilon^2}{2V(S_n) + (2/3)\epsilon R}\right)$$

- Both Azuma's Inequality and McDiamid's inequality are equivalent to Hoeffding's inequality with those assumptions:

$$\forall \epsilon > 0 : P(|S_n - E(S_n)| \geq \epsilon) \leq 2 \exp\left(-\frac{2\epsilon^2}{\sum_{i=1}^n (b_i - a_i)^2}\right)$$

- Benett's inequality (with $f(x) = (1 + x)\log(1 + x) - x$):

$$\forall \epsilon > 0 : P(|S_n - E(S_n)| \geq \epsilon) \leq 2 \exp\left(-\frac{V(S_n)}{R^2} f\left(\frac{\epsilon R}{V(S_n)}\right)\right)$$

Additionally, if the random variables X_1, ..., X_n are independent and identically distributed (i.i.d.), the central limit theorem itself can be interpreted as a concentration inequality: as n grows, $S_n - E(S_n)$ can be approximated with $Z \sim \mathcal{N}(0, V(S_n))$, which implies:

$$\forall \epsilon > 0 : P(|S_n - E(S_n)| \geq \epsilon) \simeq P(|Z| \geq \epsilon) = 2 \int_{\epsilon}^{\infty} \frac{1}{\sqrt{2\pi V(S_n)}} \exp\left(-\frac{x^2}{2V(S_n)}\right) dx$$