University of Amsterdam

Masters Thesis

Methods to optimize rare-event Monte Carlo reliability simulations for Large Hadron Collider Protection Systems

Examiner:
Dr. Valeria Krzhizhanovskaya

Author:
Milosz Blaszkiewicz

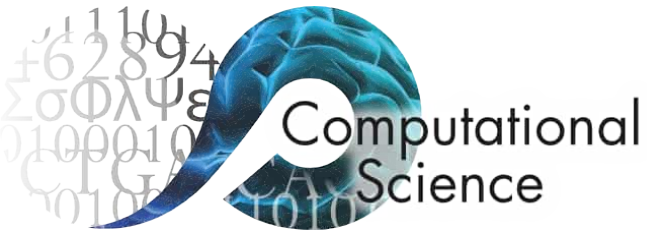
Supervisors:
Dr. Andrea Apollonio
Dr. Thomas Cartier-Michaud

A thesis submitted in partial fulfilment of the requirements for the degree of Master of Science in Computational Science

in the

Computational Science Lab
Informatics Institute

February 2022



Declaration of Authorship

I, Milosz Blaszkiewicz, declare that this thesis, entitled 'Methods to optimize rare-event Monte Carlo reliability simulations for Large Hadron Collider Protection Systems' and the work presented in it are my own. I confirm that:

- This work was done wholly or mainly while in candidature for a research degree at the University of Amsterdam.
- Where any part of this thesis has previously been submitted for a degree or any other qualification at this University or any other institution, this has been clearly stated.
- Where I have consulted the published work of others, this is always clearly attributed.
- Where I have quoted from the work of others, the source is always given. With the exception of such quotations, this thesis is entirely my own work.
- I have acknowledged all main sources of help.
- Where the thesis is based on work done by myself jointly with others, I have made clear exactly what was done by others and what I have contributed myself.

Date: 1 August 2020

UNIVERSITY OF AMSTERDAM

Abstract

Faculty of Science
Informatics Institute

Master of Science in Computational Science

Methods to optimize rare-event Monte Carlo reliability simulations for Large Hadron Collider Protection Systems

by Milosz Blaszkiewicz

Machine Protection Systems of the Large Hadron Collider (LHC) are safety-critical systems that adhere to stringent reliability and availability requirements. High reliability is required because the energy of the particles beam and the energy stored in the magnets are high enough to damage the machine beyond repair. High availability is required to maximize the amount of data collected by the experiments. Both constraints will become even more critical for LHC upgrades (High Luminosity LHC) and a possible larger accelerator (Future Circular Collider). The field of availability and reliability studies provides a variety of tools well-suited for those purposes. Among them, Monte Carlo simulation is a remarkably flexible and versatile option for in-depth analyses of complex systems.

This thesis reviews available methods for increasing computations efficiency in generic rare-event simulations framework used for reliability engineering purposes. Efforts to reduce the workload of MC simulations have been ongoing for a very long time, since the 1950s, and include methods such as importance sampling (IS), importance splitting (ISp), and randomized quasi-Monte Carlo (RQMC) method. More recent developments usually build on top of those methods. Based on a review of popular methods employed across the rare-event simulations field, ISp and RQMC methods were selected as promising solutions for our reliability and availability simulations. The empirical tests are a working proof of concept and show potential for remarkable improvements for reliability simulations of rare events with ISp and availability prediction with RQMC.

Alongside the rare-event methods, the project also involved an analysis of HL-LHC Energy Extraction system reliability, which serves as an introduction for reliability modelling and is presented in a report attached to the thesis. The project involved contributions to the AvailSim4 MC framework developed at CERN.

Acknowledgements

This project would not be possible without many people who helped in numerous aspects, ranging from organizational matters to research and implementation issues. I would like to express my sincere gratitude to everyone who was involved in this project.

First, I would like to thank the examiner of this thesis, Dr Valeria Krzhizhanovskaya, from the University of Amsterdam. The project would not even start had it not been for the helpfulness and support I got. Not to mention the significant knowledge, which, through the discussions that we held throughout the year, deeply impacted the final result of this thesis. I would also like to express my most profound appreciation to Dongwei Ye for his help, pointers to literature and numerous meetings.

I am also greatly indebted to Dr Andrea Apollonio, the supervisor and team leader, for his involvement, help and valuable remarks. My efficient induction into his team and work on this and other subjects would be impossible without his leadership. I also want to express my deepest gratitude to Dr Thomas Cartier-Michaud, who supervised me throughout this project, shared a lot of his knowledge, supported every step and provided many helpful suggestions. Without his passion, enthusiasm and determination, as well as expertise and patience, this project could never reach this stage. I could not hope for any better guidance. I also thank both my supervisors for many useful comments and suggestions to this text. Additionally, I would like to thank Lorenz Fischl for valuable suggestions regarding the quasi-Monte Carlo method.

Finally, I thank Dr Daniel Wollmann, the Controls and Beam Studies for Protection section leader, for his flexibility, supportiveness and his helpfulness. I also appreciate immeasurable amount of support from CERN colleagues, as well as staff and other students at the University of Amsterdam for their help and guidance in scientific writing sessions.

Contents

Abbreviations

CERN	European Organization for Nuclear Research
CMC	**C**rude **M**onte **C**arlo
CSL	**C**omputational **S**cience **L**ab
DES	**D**iscrete **E**vent **S**imulation
DES iteration	a single timeline created using system model, also a *model evaluation*.
EE	**E**nergy **E**xtraction system
FMECA	**F**ailure **M**ode **E**ffects and **C**riticallity **A**nalysis
HL-LHC	**H**igh **L**uminosity-**L**arge **H**adron **C**ollider
IS	**I**mportance **S**ampling
ISp	**I**mportance **S**plitting
LDS	**L**ow **D**iscrepancy **S**equence
LINAC4	**LIN**ear **AC**celerator **4**
LHC	**L**arge **H**adron **C**ollider
MC	**M**onte **C**arlo
MPS	**M**achine **P**rotection **S**ystems
MRU	**M**inimal **R**eplaceable **U**nit
MTTF	**M**ean **T**ime **t**o **F**ailure
RESTART	**REP**etitive **S**imulation **T**rials **A**fter **R**eaching **T**hresholds
RQMC	**R**andomized **Q**uasi-**M**onte **C**arlo
Simulation campaign	an aggregation of a number of *simulation runs*
Simulation offsprings	ISp restarts from the branching point, which have the same history
Simulation run	a group of *model evaluations*
UvA	**U**niversitiet **v**an **A**msterdam
QMC	**Q**uasi-**M**onte **C**arlo
QPS	**Q**uench **P**rotection **S**ystem

Chapter 1

Introduction

Mathematical modelling and computer simulation are well established methods for prediction of real-world systems behaviour. Whenever experimenting with the actual system is unfeasible, exceedingly expensive or dangerous, development of its computer model is a primary choice as a substitute for performing physical tests. Leveraging flexibility and relative ease of use, computer-based experiments offer superior range of possibilities and potential insights – due to, for instance, complete control of the environment or possibility to perform uncertainty quantification among others. The considerable difficulty of using this method, apart from creating a model which accurately describes the system, is to ensure proper understanding of the results accuracy. Therefore, validation and verification of a model are particularly relevant parts of the entire modelling process. Given all advantages, computer simulations have found many applications in fields such as natural sciences, engineering, finance, operations research and many more.

The earliest computer simulations belong to the group traditionally referred to as *Monte Carlo (MC) simulations*. Their history goes back to the early days of computing at the Manhattan Project active during World War II. Despite their long-standing history, they retain a strong presence in modern computing. The goal of those simulations is to quantify probabilities of uncertain events based on inputs defined in probabilistic way, e.g., by probability distributions. One of the most powerful advantages of the MC simulations is that their results are obtained in a somewhat intuitive fashion, and thus have a straightforward interpretation. However, such simulations require a highly intensive computational effort, which sometimes increases to unsustainable levels - as it is the case when they involve rare events.

In this thesis, we want to explore methods for increasing the efficiency of Monte Carlo simulations for rare events in the context of systems' reliability studies. Chapter 1 gives an introduction to several aspects of this work: Monte Carlo and the rare events

methods (section 1.1), CERN (section 1.2) the context of general reliability studies and their specific instance in Machine Protection studies at CERN which is the primary use case for this evaluation (section 1.3) and the research objectives set for this thesis (section 1.4).

The thesis structure follows the standard convention. Chapter 2 gives an overview of the available methods and of the current state of research. In chapter 3, details of the methods selected for evaluation are presented, together with solutions used to bridge the approaches with the reliability framework. The experimental setup and results of the performed tests are the subjects of chapter 4. Discussion of the results is provided in chapter 5. Finally, conclusions and future work are presented in chapter 6.

1.1 Rare-event Monte Carlo simulations

Monte Carlo simulations are a class of computational algorithms for quantification of the outcomes probability based on input defined using probabilistic means. Typically, they involve sampling the parameter space according to probability distributions defined as input, evaluating the defined problem function using drawn samples and performing a statistical analysis of the gathered results. One of the examples illustrating the method is the approximation of π value by performing a numerical experiment counting the number of samples evaluated to be within a circle or a similar Buffon's needle problem [1]. Their true potential started to be utilized only with the advent of computers providing means to generate pseudo-random numbers. In this thesis, we try to address one of the significant limitations of Monte Carlo simulations: the computational effort needed for simulations of rare events.

Observing rare events in Monte Carlo simulations makes it necessary to perform significant number of model evaluations, the majority of which is inconsequential to observing events of interest (e.g. a system failure). There are various approaches to mitigate those disadvantages. First is to perform computations in parallel or distributed way. The standard MC algorithm, called Crude Monte Carlo (CMC) to distinguish from more advanced methods, is a rather simple case for parallel or distributed computing algorithms, however such a solution requires appropriate computing infrastructure and resources. The other option is to use advanced methods of variance reduction to explore the problem space in a more efficient way.

[1] Buffon's needle problem, formulated in 1733, originally not meant as a method for π approximation

The research in rare event aspects of Monte Carlo simulations has drawn interest since the very beginning of the field's existence, as early computers and their limited capabilities made many use cases fall into this category. In the early days of Monte Carlo simulations, the late 40s and 50s, only the very first and strongly limited computers were deployed. The computations were severely limited, as well as time-consuming, making available computational power scarce and not sufficient even for relatively uncomplicated simulations. The advantage given by decreasing the workload through the algorithm could not have been neglected at that point. Consequently, the first variance reduction techniques started almost in parallel with the formulation of the entire Monte Carlo approach.

The constant development of individual processing units and methods for distributed computing led to a massive expansion of processing capabilities. Gradual improvements in equipment and computing infrastructure made the theoretical advances of rare-event simulations somewhat less relevant, as what was considered *rare* earlier could be simulated much easier by applying more computing power. Nevertheless, despite the growth of capabilities, the needs continued to grow at the same (if not even higher) rate. The research in this field continues to incorporate fresh ideas into state-of-the-art simulation algorithms and new use cases. In this thesis, *rare events* are understood as events which appear relatively infrequently in the CMC simulations. Following [1], they can be defined already as events of probability lower than 10^{-3}. As explained later in the thesis, the events of interest in reliability simulations can be significantly lower than 10^{-6}.

The overall goal for rare event methods is to increase accuracy of the simulation by reducing the number of samples required to observe events of interest (or achieve a predefined accuracy). The reduction is achieved through leveraging elements of knowledge of the problem or making assumptions about the expected result. The term *rare event methods* spans over all methods used to force simulations to focus on selected *interesting* parts of the problem space, otherwise unlikely visited by CMC sampling, while spending less time in regions yielding trivial or non-interesting solutions. Such biasing requires compensation in the statistical analysis of the results and can be difficult to perform efficiently. Details of available approaches to this problem in the literature are studied in detail in chapter 2.

1.2 Machine protection at CERN

With 23 member countries and many international collaborations, the European Organization for Nuclear Research (CERN) is a leading research institution established in 1954 on the French-Swiss border near Geneva. Through specialized facilities for particle

and high-energy physics, it enables scientists to seek answers to fundamental questions relating to the nature of the world and to deepen human understanding of laws governing the universe. The organization headquarters are home to the world's largest particle accelerator, the Large Hadron Collider (LHC). Machines such as this one are used to accelerate beams of subatomic particles to high energies using electromagnetic fields. The LHC is a circular collider accelerator, meaning that beams are traveling in a ring-shaped tunnel and colliding with each other in designated spots, producing events of interest. Detectors measure and record those events – tracked parameters include speed, mass and charge. The objective is to track numerous collisions happening at high energies, as those provide data to the experiments and thus contribute to experimental validation of theoretical physics models and theories.

Maintaining the beam on track requires using powerful superconducting magnets. Those are at risk of suffering from a physical phenomenon called *quench*, which is a transition from the *superconducting* to a *normal* conducting state. In such events, the loss of superconductivity is accompanied by a release of heat, which – if not correctly managed – can cause extensive damage to a magnet and other equipment in the vicinity , compromising the machine integrity. Notably, a magnet quench caused a significant incident in the first tests of the accelerator in 2008. The damage to the magnets in the tunnel resulted in a one year delay in the start of LHC operations. Further details were released in a report [2] summarising the incident. In order to counter effects of *quenches*, Machine Protection Systems (MPS) are monitoring the parameters that indicate their potential occurrence and, upon detection by the so-called Quench Detection System (QDS), initiate a procedure of energy extraction. In broad terms, the process involves the interruption of the magnet's powering circuit and the dissipation of the stored energy in suitable resistor banks. Those tasks are realized by Energy Extraction system (EES).

Given the extraordinary scale and purpose of the LHC project, ensuring safety and appropriate performance of the installation has always been a primary concern. To ensure a safe operation, the machine protection systems need to adhere to stringent reliability requirements. The accelerator complex is a world-class example of an intersection of science, technology and engineering. Reliability is of particular concern for MPS systems, given their significance in protecting personnel safety and equipment integrity inside the tunnel. Methods which are used for risk quantification include failure modes, effects and criticality analysis (FMECA), fault tree analysis and probabilistic failure simulations.

The LHC has led to many great discoveries, including the most well-known discovery of the Higgs boson in 2012, which was a cornerstone for the research planned at the facility. The device proved capable and delivered accurate measurements that eventually enabled the scientists to prove its existence. Apart from that, there have been multiple other

experiments and theories tested using observations and measurements completed using the LHC. However, the current capabilities of the accelerator are reaching their limits. Many of the more intricate theories need even larger amounts of more accurate data. To address these needs in the long-term, CERN develops plans for upgrading the current equipment. In the coming years, a vital upgrade of the LHC will be shaping the increase in discovery potential. The High Luminosity Large Hadron Collider (HL-LHC) project aims at increasing the rate of collisions by one order of magnitude. Such an improvement will enable more accurate measurements and more frequent observations of rare processes resulting from subatomic collisions.

Such an undertaking relies on implementing cutting-edge technology in various LHC systems. Primarily, it includes even more powerful focusing magnets, improved beam optics, enhanced machine protection and more. A higher number of collisions inseparably puts a strain on the equipment, making maintenance an even more relevant problem. Without doubt, probabilistic simulations of availability and reliability will provide indispensable insights to ensure safety of the upgraded machine. Furthermore, initial studies (e.g., [3]) have shown that standard maintenance strategies in the Future Circular Collider (FCC - one of the planned successors to the LHC) context would decrease its beam running time to just a small fraction of the entire operational period. State-of-the-art research, like preventive maintenance, are viewed more as a necessity than a way to optimize usage of resources.

1.3 Reliability and availability studies

Modern devices and machines are often examples of complex systems composed of hundreds or even thousands components that depend on each other to various extents. Studying their reliability is a significant undertaking that provides essential insights for all parts of the systems life-cycle: from design phase, through production, up until long-term operation. Leveraging results of a failure risk quantification may positively impact the system lifetime, assist quality assurance, diminish failures probability by recommending appropriate measures, contribute to defining manuals and policies guiding the use of the system. The need for such analysis is even stronger in cases of large, safety-critical machines, where involved costs are higher and risk has to be strictly controlled – with aircraft and nuclear power plants being common examples.

The reliability and maintainability theory provides mathematical definitions and the necessary tools in the form of probabilistic abstractions and stochastic models. Reliability can be defined as a function of success $R(T = t)$ at time t. For further details on mathematical foundations, the last chapter of Handbook on Stochastic Models [4] presents

the subject primarily from the perspective of simple statistical modelling. The chapter contains an overview of statistical notions useful in this context, mathematical definitions of probabilistic reliability models' components, derivation of several commonly used probability formulas as well as presents material on subjects such as *accelerated life testing*.

Availability is a separate metric from reliability and is defined as a fraction of time A in which a system remains operational: $A = \frac{\text{uptime}}{\text{downtime+uptime}}$. Its accurate predictions are among crucial performance indicators and play important role in design and planning. Despite the differences, it shares a common estimation approach and many methods with reliability; for that reason, the same software framework is used to simulate both quantities.

The systems' reliability field has defined a range of methods that can be applied to model system failures or to observe the dynamics of the overall reliability behaviour. When choosing the simulation method for a given problem, one of the first questions which needs to be answered is whether the problem can be solved analytically. Should it be possible, other approaches can hardly offer any superior advantages. However, the task of estimating the risk probability is not always a straightforward undertaking. Especially in cases of large, safety- or mission-critical systems, aspects such as complex inter-dependencies, maintenance policies, replacement strategies, and others create a fairly complicated system that requires stochastic modelling and statistical analysis to obtain accurate insights and estimations.

Among the classic approaches, a system made of components can be described as a fault tree or a reliability block diagram. Individual components are defined by parameters such as individual failure rate, repair time, and other properties. The problem becomes increasingly more complex when the study is required to account for more factors. Those could include aspects such as repair strategies, periodic inspections or operational phases dictating changing failure regimes. On top of that, components might be (and very often are) interdependent, i.e., their functioning might be altered by the status of groups of other components, or they may be replaced together as a part of a larger unit. All those factors, collectively describing operation and maintenance of the system, contribute to the overall reliability of the analysed system. An analytical solution to a problem formulated like that is an unfeasible option in most practical applications.

Monte Carlo simulations of availability and reliability give the advantage of easily incorporating advanced inter-dependencies between individual components and allow for modelling complex maintenance strategies (e.g., replacement of a number of components triggered by failures in some parts of the system) and additional elements of monitoring and inspections. Those elements can create a specific dynamic of the overall system

reliability, which would be overlooked in a plain analysis of the fault rates. In addition, Monte Carlo simulations allow for accurate description of various events and specific scenarios, offering superior flexibility, vastly surpassing possibilities of other methods.

Approaches based on performing stochastic simulations of the system's reliability models are very often omitted due to the rare-event nature of the sought events. For a reasonable accuracy and meaningful statistical properties, simulations must show a repeated pattern, so a realistic number of iterations proving system's reliability would require observing the rare event even 100 times or more. That translates into enormous computational power requirement, as usually the probabilities of events of interest are small. This is the main reason why in the case of such simulations, methods aiming at dealing with rare events are particularly beneficial.

AvailSim4 - Monte Carlo framework for availability and reliability simulations

In this thesis, rare event simulation methods are evaluated in the context of computationally-intensive reliability simulations of LHC Machine Protection Systems at CERN. The current approach is a result of development of consecutive iterations of the reliability Monte Carlo framework called AvailSim4.

The first version was developed at Stanford University for the availability study of the International Linear Collider (ILC) and impact of its component failures on various performance metrics. The overall aim was to provide estimations exclusively for that single study, therefore the tool was not designed with reusability or adaptability in mind. Hence, when a similar study was performed for International Fusion Materials Irradiation Facility (IFMIF), the second version of AvailSim was developed. While it extended a number of features, it was also largely contained to the object of that study. The third version was developed at the European Spallation Source and then at CERN. Approaches taken in both first iterations were promising, however other considerations made it much less attractive: a steep and demanding learning curve. It was a significant obstacle for using it, given a large and often-changing user base of such tools at CERN. For that reason, the third version of AvailSim was developed in 2016. The goal of that one was to make the analysis workflow more generic and ready to be customized by the end-users. Ultimately, not only the usage needed to be generic. For a modern software to succeed, it requires constant maintenance and development. That problem was addressed by another version, AvailSim4. The priority was put on high code quality, modern software design, development and continuous integration practices, while maintaining features and capabilities of the previous versions.

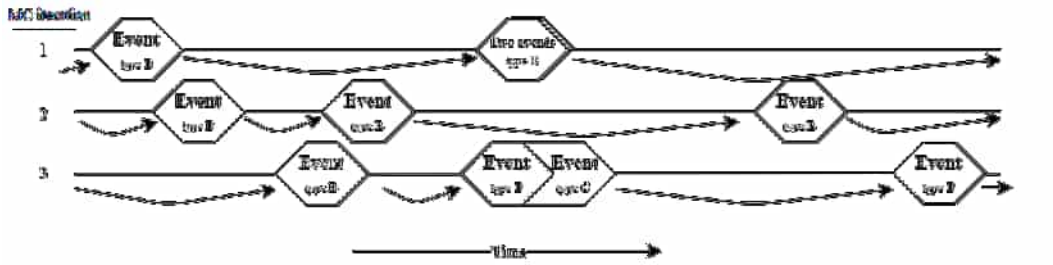


FIGURE 1.1: Illustration of three DES timelines generated in three different MC iteration of the CMC algorithm. The three phases approach is illustrated by the jumps between events.

The framework is based on a CMC simulation method driven by a discrete event simulation (DES) approach. In principle, the simulation is driven by jumps between chronologically consecutive events - in this case, usually failures of individual components. In AvailSim4, it has been decided to use the Three Phased Approach, first proposed by Tocher in his book [5] and then described in greater detail by Pidd et al. [6]. The three phases are defined as follows:

- A – searching for the next event and moving the clock to its timestamp,
- B – executing all events that are due at the clock time,
- C – executing all conditional events which conditions have been met in the B phase.

DES is repeating those three phases in a loop, monitoring time of the successive events. Evaluation ends when simulation time exceeds predefined threshold (simulation lifetime) or there are no more events to execute. The workflow is illustrated in figure 1.1. Each model is defined by a set of properties describing individual components in the system, their failure likelihood, dependence on other components, repair strategy, monitoring, etc. Special events, such as inspections or replacements of groups of components, can be defined on the system level. Particularly important for accelerator-related availability and reliability analyses is the introduction of phases: for example, the LHC may operate in different conditions depending on the mode in which it is operating in a given moment. Failures, their properties and consequences may vary, e.g. depending on the beam energy. Phase changes can be independent from failures, but they can also be triggered by failures. More details about AvailSim4 are given in chapter 4.

1.4 Research objectives

The thesis is focused on experimental work concerning methods for optimizing rare-event Monte Carlo simulations. The work includes development of the simulation algorithms for observing and quantifying the risk of occurrence of such events in a Monte Carlo framework, created originally for machine protection systems at CERN. The research questions investigated in this thesis are:

1. What methods are available to optimize Monte Carlo simulations of systems reliability and/or availability by increasing their efficiency?
2. How relevant and applicable are those methods to modelling safety-critical systems?
3. How to leverage application of those methods in context of machine protection for accelerators?

Aside from the main thesis objectives, the work at CERN also included more technical contributions made to the AvailSim4 software. They include a development of root cause analysis feature and a tool for reduction of partial results obtained in distributed execution (appendix D). An analysis of the reliability of the HL-LHC Energy Extraction system design was performed and is included in appendix C, which served as an introduction to the reliability modeling and is a good example of studies that could benefit from the rare-event simulation methods.

Chapter 2

Literature review

Choosing Monte Carlo over other methods for reliability simulations enables greater flexibility in modelling and, consequently, more detailed representation of the dynamic behavior of the studied system. Accounting for factors such as maintenance strategies, complex replacement procedures or operation phases gives an opportunity to understand the real-world failure dynamics better. However, this advantage comes at the cost of more computationally-intensive simulation task. In the attempt to alleviate this issue, rare-event Monte Carlo simulation methods have been studied by numerous researchers. In this chapter, a literature review of some selected approaches is presented, with particular emphasis on the reliability context.

The general idea behind all methods described in this chapter is to reduce the variance of the results obtained from simulations. This objective translates directly into increasing accuracy while maintaining the same number of model evaluations, or, conversely, obtaining same accuracy level while decreasing the number of simulations. The variance reduction in practice can be viewed as an attempt to use the available knowledge of the examined problem to improve the accuracy of the result estimations. Multiple options are available but each is characterized by specific requirements, potential efficiency gains and, ultimately, limitations.

Research in rare-events Monte Carlo simulations is coming from many different domains and thus new developments tend to originate from various backgrounds. A comprehensive overview of all available methods is a challenging task. An interesting take is presented in article [7], where authors provide a general survey and attempt to provide a high-level overview of for rare-event methods for static models. The main underlying objective for this article is to provide a guide through such methods for practitioners, outlining their advantages, disadvantages and suitability for specific types of simulations.

In their contribution to the Handbook on Uncertainty Quantification [1], Beck and Zuev present a more specialized overview of three rare-event Monte Carlo simulations methods of highly-reliable systems: *importance sampling*, *subset simulations* and *importance splitting*.

Every method discussed in this thesis is addressing the same problem. The goal is to estimate the probability of the modelled system to enter a configuration state that is leading to a critical failure up to time t, i.e.:

$$\mu = \mathbb{P}_t(x \in F)$$

where t is the intended lifetime of the system and set F is defined as set of system configurations which are causing a critical failure.

2.1 Crude Monte Carlo

Before moving on to more specialized methods, it is beneficial to look at the crude Monte Carlo simulations as a method for simulating rare-events. Despite the disadvantages, the plain approach offers certain other advantages and as such might also be useful to consider in certain cases (e.g., the analysis of the Energy Extraction system completed alongside this thesis used parallel crude Monte Carlo simulations; more details are available in Appendix C).

The probability of a failure of component c in crude Monte Carlo is given by the estimator $\hat{\mu_c}$:

$$\hat{\mu_c} = \frac{1}{N} \sum_{i=1}^{N} f_c(x)$$

where N is the number of iterations of the algorithm, $f_c(x)$ is a function taking independent and identically distributed samples as arguments.

The convergence rate of a crude Monte Carlo simulation, by the Central Limit Theorem, is close to $O(\sqrt{1/N})$.

As presented in [7], the relative error of the estimator $\hat{\mu_c}$ is equal to:

$$RE(\hat{\mu_c}) = \frac{\sigma_{\hat{\mu_c}}}{\mu_c} = \frac{1}{\sqrt{N}} \frac{\sqrt{\mu_c - \mu_c^2}}{\mu_c}$$

For rare events, probability μ_c is close to 0, so:

$$\lim_{\mu_c \to 0} \frac{\sigma_{\hat{\mu_c}}}{\mu_c} = \lim_{\mu_c \to 0} \frac{1}{\sqrt{N \mu_c}} = +\infty$$

The relative error of an unbiased estimator thus tends towards infinity as probability of the simulated event tends to zero. In other words, the relative error is unbounded when decreasing the events probability, showing the main drawback of using crude Monte Carlo methods in rare-events field. In practical terms, an event whose probability is 10^{-5}, will require, on average, 10^5 model evaluations to observe even a single occurrence. For a meaningful statistical value, such events should be observed at least several times. In case of AvailSim4, the general target is to observe roughly 100 events of interest (i.e. system failures) – which brings the number of model evaluations to 10^7. This is already a significant number – assuming that each evaluation takes a second, a sequential execution would take roughly 4 months, making it necessary to apply parallel or distributed computing techniques.

Advantages, however, include their simple and straight-forward implementation, which also makes results interpretation and explaining a relatively easy task. No changes in the sampling also decrease the risk of complicated, intricate code, more advanced estimators and remove the need for compensating biased results.

2.2 Importance sampling

The importance sampling approach is based on the observation that certain regions of the problem space carry greater *importance* (i.e. weight) than other for the estimated value. The method concept is relatively simple and, as such, its origins go back to the early days of the Monte Carlo method (e.g., [8], [9], at that time also known as "quota sampling").

The method is discussed broadly in textbooks and publications such as [10] or [11]. Generally, the method is based on the choice of an appropriate sampling which is supposed to decrease the variance of the model evaluation results. Rather than scattering the sampling points accordingly to the theoretical distribution throughout the problem space (as it is the idea in crude Monte Carlo), this sampling strategy shifts the distribution of samples to parts which are the most "important". How exactly variance is reduced in this case can be easily illustrated using one of the standard Monte Carlo problems, integration of a multidimensional integral I, as shown in [12]:

$$I = \int f(x)\, dx, \text{ where } x \in R^n$$

Using an importance sampling distribution $g_X(x)$ (where $\forall x \in R^n : g_X(x) > 0$), the integral can be rewritten as:

$$I = \int \frac{f(x)}{g_X(x)} g_X(x)\ dx = \mathbb{E}\left[\frac{f(X)}{g_X(X)}\right]$$

Hence, $\hat{I} = \frac{f(X)}{g_X(X)}$ is an unbiased estimator of I, and its variance is equal to:

$$var\ \hat{I} = \mathbb{E}\left[\hat{I}^2\right] - \mathbb{E}\left[\hat{I}\right]^2 = \int \frac{f^2(x)}{g_X^2(x)} g_X(x)\ dx - I^2$$

To estimate the integral, samples are drawn from the probability density function $g_X(x)$ and then used in the sample-mean formula to estimate $\mathbb{E}\left[\frac{f(X)}{g_X(X)}\right]$.

The minimum value of $var\ \hat{I}$ is (the proof uses Cauchy-Schwartz inequality and is presented in [12]):

$$var\ \hat{I}_0 = \left(\int |f(x)|\ dx\right)^2 - I^2$$

If we take the probability density function (p.d.f.) $g_X(x) = \frac{f(x)}{I}$ and substitute into the variance formula above, we see that it leads to the optimal value of the variance $var\ \hat{(I)} = 0$. The optimal sampling p.d.f. depends on the value of the integral which the method tries to estimate at that point. However, selecting a p.d.f. which only resembles or exploits some specific peculiarities of the surveyed problem can already give a way to a successful application of the method.

The problem of how to choose a suitable p.d.f. remains the most significant obstacle to use the method for reliability predictions. Methods include "forcing" described in [13] and "balanced failure biasing" described in [14] and [15]. The former method changes the probability density function to enforce model evaluations to have failure transitions before predefined time T. The second method realizes the following principle: biasing the model evaluations by increasing the probability of transitions to failure states over the repair ones. Of course, both methods have to compensate appropriately for the created bias.

More general strategies for guiding sampling for reliability predictions are discussed in [16] and [17]. The first one makes use of the cross entropy method, which utilizes Kullback–Leibler divergence. The second one goes even further, and builds the strategy using a surrogate model.

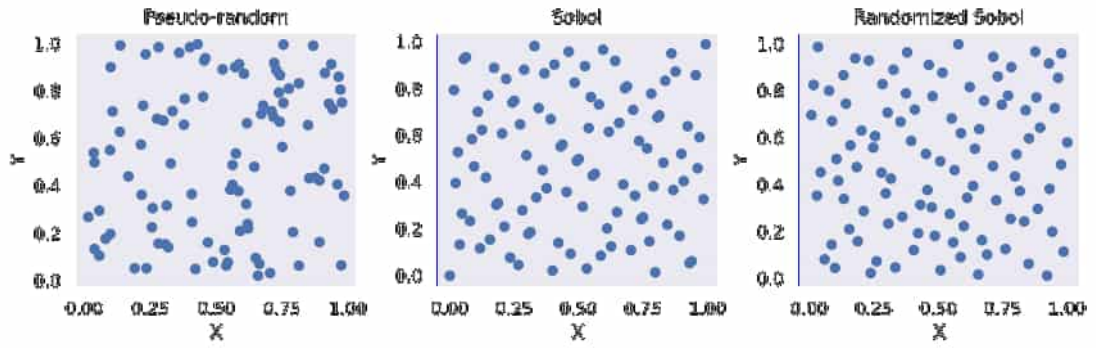


FIGURE 2.1: Scatter plots of points generated using standard pseudo-random algorithm (left), Sobol generator (center) and a randomized Sobol generator (right). Each plot contains 100 two-dimensional points, with values of [0, 1)

2.3 Quasi-Monte Carlo

A notably different approach to reducing variance is represented by the quasi-Monte Carlo technique. The goal is to increase the accuracy of simulations for a given number of points by ensuring that the points are distributed more evenly, thus preventing emergence of points' "clusterings" - that very often occur in sampling based on standard pseudo-random numbers. Quasi-Monte Carlo are different from crude Monte Carlo in this sense that the sampling is utilizing low-discrepancy sequences (LDS). In the strict meaning, the QMC method is not based on randomness at all, as the LDS sequences are deterministic – randomization can, in many cases, be introduced in sample generation, but does not have to be. Figure 2.1 displays examples of pseudo-random sampling, generating points from a Sobol sequence.

A comprehensive introduction of the quasi-Monte Carlo sampling is given by Niederreiter in the introductory chapter of his book [18] and provides a justification and the whole reasoning behind the concept. The general outline is that crude Monte Carlo error is characterized by a probabilistic error bound of $O(N^{-\frac{1}{2}})$ with N being the number of points selected at random for sampling (the mathematical derivation of this bound is presented in the mentioned chapter). Hence, producing a selection of points which diminish the said error is beneficial for the results - and that is the goal of using low discrepancy sequences, which are the very basis of the quasi-Monte Carlo approach. The mathematical formulation of the standard Monte Carlo integration problem is:

$$\int_D f(x)\ dx = \frac{1}{N} \sum_{i=1,\ x_i \in D}^{N} f(x_i)$$

where D is the integration domain, and points x_i are deterministic. The remaining question is how to achieve accuracy improvement by selecting specific points in the problem

space using a general algorithm. To that end, the proposed solution is a construction of *low-discrepancy sequences*, which are guaranteed to be evenly scattered across the problem space. Of course, discrepancy needs to be formulated in mathematical terms as well. There are many metrics that display specific properties; here – following [19] – one of the simplest of them, the star discrepancy $D*$, is defined as:

$$D_n^*(x_1, \ldots, x_n) = \sup_{a \in [0,1)^d} \left| \frac{1}{n} \sum_{i=1}^{n} 1_{0 \leq x_i < a} - |[0, a)| \right|$$

The subject of low-discrepancy sequences raised a substantial interest from researchers. Over the years, multiple such sequences were defined and examined from the theoretical and empirical perspective. The literature shows many studied examples, of which the most well-known include: van der Corput sequence ([20]) generalized in the form of Halton sequence ([21]), Sobol sequence ([22]), Faure sequence ([23]) and its Niederreiter's generalization ([24]). Of course, each of these sequences has different properties and may be suitable to certain classes of problems more than to others. Apart from that, essentially each type of sequence has already received multiple implementations and modifications, each increasing their performance in some problem sub-classes. However, the studied articles generally indicate that the superior computational results are usually displayed by Sobol' sequences. In [25], authors compare implementations of Sobol, Halton sequences and antithetic variables methods in the context of financial derivatives valuation, and conclude that Sobol sequences consistently outperform the other two methods. Harase [26] compares Sobol and Niederreiter sequences with certain additional modifications (e.g., dimension projections using a Principal Component Analysis method) and concludes that both exhibit high performance and accuracy in the studied finanical application.

Chapters 6.4 and 6.5 of the Lemieux's book [27] address the issue of using quasi-Monte Carlo sampling for simulation purposes. They conclude that low-discrepancy sequences in quasi-Monte Carlo can be considered variance reduction techniques. The author warns, however, that when the simulated region of interest is small compared to the entire sampling space, the quasi-Monte Carlo might suffer from the very same issue of not having enough samples in the region of the problem. This problem is very relevant to the studied case – individual failures occur with low frequency and their critical combinations are even more rare. One potential advantage, still, is that generating randomized low-discrepancy sequences can be less demanding task than generating sets of random points of corresponding sizes.

The worst-case error of the quasi-Monte Carlo method is of order $\frac{(\log n)^d}{n}$, as shown in [28]. Such a result may indicate potential problems when the number of dimensions d

is large. As argued in [29], the estimation of the worst case error bounds of QMC is not in line with empirical results that the method had been achieving with some high-dimensional integrals and thus they provide a *partial answer* explaining why this is the case. They do it by identifying a class of integrals in which impact of dimension d is negligible.

In a chapter of [30] devoted to numerical integration, the author states that the QMC method has *potentially* a convergence rate of $O(1/n)$, which is a significant gain in comparison with $\frac{1}{\sqrt{n}}$ convergence rate of crude Monte Carlo. The article [31] studies the approximation of integrals and defines the sufficient conditions for fast quasi-Monte Carlo convergence. It shows that in the worst case quasi-Monte Carlo convergence rate is of order $n^{-1+p(logn)^{-1/2}}$ where $p \geq 0$ is a constant that depends on the type of the integration problem.

A significant limitation still remains in the capabilities of modern implementations. There are substantial obstacles for designing highly-dimensional Sobol' sequences, extensively discussed in [32]. At the time of its publication, their most advanced attempt finished at 16,384 dimensions - this result was outperformed later with a generator capable of generating samples of 65,536 dimensions (as stated in [33]).

An interesting aspect in the view of this study purposes is presented in [34], where authors examine the possibility of a combining two or more of the approaches described in this chapter. More specifically, they found that randomized quasi-Monte Carlo used with importance sampling or splitting generally provide advantage over using those methods separately.

2.4 Importance splitting

Importance splitting is a name for a broad category of methods used to accommodate the needs of intensive rare-event Monte Carlo simulations. Similarly to importance sampling, those methods reduce variance – however the path to achieving this goal is different. Instead of biasing the sampling distribution *a priori*, the problem sampling is dynamically shifted according to the content of the model evaluations obtained so far. More specifically, values of a pre-defined importance functions are monitored and when their value crosses a defined threshold, the evaluation is branched into multiple instances. Each instance shares the same *past* from the period before branching (biasing the results), but then develops independently. This way, the simulations are more likely to return significant results: the number of evaluations which are close to the events of

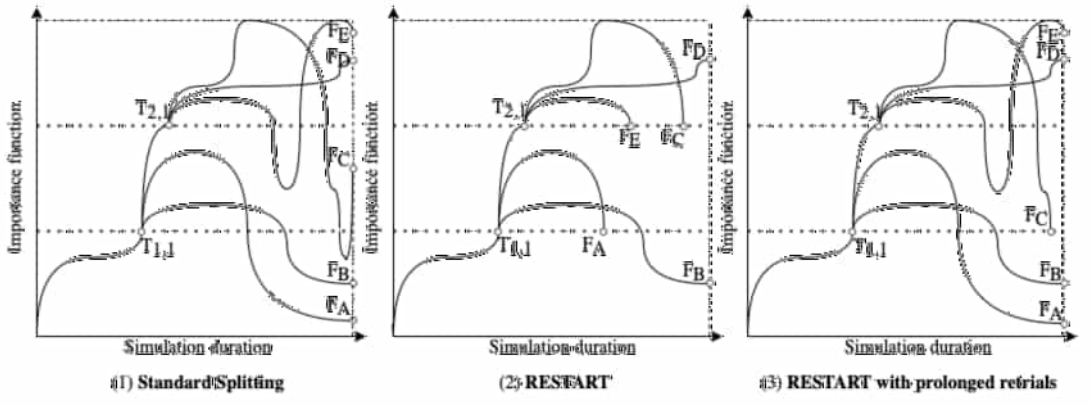


FIGURE 2.2: Illustration of three strategies for managing paths crossing the threshold

interest (and thus - more likely to reach them) are artificially inflated, while *uninteresting* are not branched and thus finish, with considerable computation time savings.

A general note for the *importance splitting* field is that the literature appears to show no consensus on the naming convention for these methods. For instance, *importance splitting* sometimes refers to the general category of methods based on branching simulation paths, while sometimes (usually by shorter *splitting*) it refers exclusively to the standard splitting method. On the other hand, *importance splitting* sometimes is used as a name for *subset simulation*, *subset sampling*, or *sequential Monte Carlo* [35] – which sometimes refer to more specific type of splitting algorithms involving Markov Chain Monte Carlo to generate conditional samples.

The standard splitting method is somewhat less well-known than importance sampling, but its history can also be traced back to beginnings of MC simulations, and has been studied extensively since – a number of publication is listed in [36].

The problem of estimating a single rare event probability is transformed into estimating more probable conditional probabilities that eventually lead, in the reliability simulation context, to a critical system failure. This can be described using a chain of conditional (probabilities [7]):

$$\mu = \mathbb{P}(X_m) = \prod_{k=0}^{m-1} P(X \in F_{k+1} | X \in F_k)$$

where $X_0 \supset X_1 \supset \cdots \supset X_m$ and F_k are all states that reach threshold k. The probability of the first $P(X \in X_0)$ can be obtained by using a crude Monte Carlo estimator of the number of evaluations reaching the first threshold. The second level restarts are slightly more complicated matter. Essentially, to estimate the probability in a level $k+1$, the algorithm should draw samples from entrance distribution G_k – however, that distribution cannot be known. To approximate drawing samples from that distribution, samples can be created by starting multiple copies of evaluations that have reached the

threshold; $P(X \in F_{k+1}|X \in F_k)$ is then equal to R_k/N_{k-1}, where R_k is the number of evaluations reaching next threshold.

The exact procedure is illustrated in the left pane of figure 2.2. In the first phase, a predefined number of independent evaluations start from the initial point. The ones that never observe any crossings of the first critical threshold finish at the end of the simulation time. All others, upon crossing the threshold, create a number of copies of themselves and continue on-wards. Depending on the simulation and modelled system, there might be many thresholds defined: then the situation from the first phase repeats for each level.

A notable modification of this approach is called RESTART, REPetitive Simulation Trials After Reaching Thresholds, introduced in [37]. Improvement is obtained by truncating repeated evaluations that again cross the critical threshold – their importance falls below the value where they would be branched, so they become less likely to reach the event of interest. The method is shown in the center pane of figure 2.2. Only a single evaluation is designated to go through cross the threshold again, in order to account for restarts in the latter parts of the evaluation (in the presented example – it is the path finishing in point F_B). The paper provides a comprehensive analysis of the method, including the analysis of potential gains, their bounds, optimal choice of parameters and a general summary of the approach.

The procedure can be summarized as follows (letters refer to the center pane (RESTART) of figure 2.2):

1. Begin model evaluation and monitor value of the importance function, just as in the standard splitting.
2. When the importance function reaches a predefined threshold, save the state of the system and run N instances of the simulation starting in the saved state.
3. Whenever any of those trials crosses the same threshold from above, stop the trial (e.g. evaluations finishing in F_A, F_E, F_C). This rule applies to all evaluations except for the last one, which continues further (e.g. path F_B).
4. If the threshold is reached again, repeat the procedure.

An example of a successful application is presented in a companion paper [38]. Authors claim increase in accuracy in that specific instance of the algorithm to be of four orders of magnitude.

Similarly to the importance sampling, the standard importance splitting and RESTART method also rely on an importance function which defines the likelihood of reaching the

event of interest from a given state of the system. One extension addressing this issue is is called automated importance splitting technique ([39]). The automation is relatively simple and realized through inverse breadth first search traversal of the tree of all states, labeling each of them with the probability of reaching the event of interest. The paper provides the theoretical discussion of the algorithm, as well as empirical results and comparison with importance functions defined in a standard way.

Researchers continued studies of the RESTART method introducing improvements to the original algorithm. The most promising one is called "RESTART with prolonged retrials", first proposed in [40] and then subsequently reproduced in [41]. In those publications, authors present the theoretical foundation and empirical results for a change in the procedure in the way retrials are managed. In standard splitting, the retrials are running from the branching point until the end of a simulation. In classic RESTART, they are ended when the importance function value falls below the threshold - and only the last retrial is allowed to continue, possibly reaching the threshold again, triggering more branches. The prolonged retrials make a point of keeping simulation branches which cross the triggering threshold and cancel them only when they drop several consecutive thresholds. A few example evaluations are shown in the right pane of figure 2.2 – points F_E and F_B would be truncated earlier without prolonged retrials, while evaluation leading to point F_C is stopped after dropping through two thresholds. In [42], author compares all three modes and concludes that even classic RESTART is outperforming standard splitting in all cases, while the prolonged retrials variant increases that advantage up to 50% with respect to standard splitting.

Another modification, referred to as *subset simulations*, was proposed in [43] and is an extension of the standard splitting method. The calculation of the overall simulated probability is also divided into a chain of conditional probabilities estimations representing steps on a path to the event of interest. The novel part is in the application of the Markov Chain Monte Carlo (MCMC) and a modified Metropolis algorithm to efficiently draw samples from entrance distribution G_k rather than using cloned copies of the evaluations that have reached them in the previous phases. Metropolis (also Metropolis-Hastings) algorithm is a method to obtain samples from distributions from which direct sampling is difficult. It uses random walk extended with accepting and rejecting samples based on a acceptance ratio set . More details can be found in the original paper [44] or in a more practical [45].

As mentioned in the beginning, there are multiple extensions and modifications of the original method. In [46], which proposes importance splitting for finite-time rare event simulations, authors point to many other works done in this area. It is worth to note

one more method which further extends the applicability of the importance splitting approach: *general splitting*, which is the subject of [47]. The main objective is to generalize the approach to static (time-independent) and non-Markovian models. Authors propose the algorithm, prove its unbiasedness, and present empirical data of its application to 3SAT problems.

Finally, there are some practical considerations involved in the implementations of the algorithm. In his PhD thesis [48], Gravels includes a discussion of two aspects and two approaches for each aspect: *fixed splitting* or *fixed effort* in one, and *single stepping* and *global stepping* in another. The difference is in the number of copied evaluations – *fixed splitting* always creates the same number of copies from each state that reached the threshold, while in *fixed effort*, the total number of created copies is fixed (i.e., more starting states means creating less copies of the same evaluation). The former approach carries two significant risks: with a fixed number of predefined restarts, their number can easily explode, or, conversely, produce too few evaluations to restart from in the next phase. However, the execution time is more predictable and avoids adding additional controls for situations when a threshold cannot be reached. On the other hand, as argued by Gravels, *fixed effort* not only removes that risk, but also shows better efficiency.

In his PhD thesis [48], Gravels differentiates between two approaches to the splitting implementation which are particularly important for parallel computations aspect. In *global stepping*, the algorithm branches simulation paths when the importance function reaches the pre-defined threshold and proceeds to compute those paths immediately. Contrary to this, *single stepping* evaluates paths in the same level concurrently. The execution is thus divided into several stages, and each stage has an assigned threshold up until which the evaluations are performed. In simplistic terms, the first approach is more straight-forward and requires less memory, as only the simulation state at the point of most recent branching needs to be stored. However, it can be problematic in distributed computing setting: after starting simulations on multiple nodes, some of them will require little time to finish simple runs which happen to produce less branches, while other nodes - which encounter more interesting paths, with multiple branches - will take significantly longer. Therefore, the second approach can be particularly suitable for parallel computations: simulations workload can be evenly balanced between stages across computing nodes without much additional effort.

The task of setting correct parameters for the importance splitting algorithm is a challenging one. [49] and [48] recommend performing initial pilot jobs to establish some general knowledge of the probabilities in splitting levels and use that information to calibrate the main simulation run.

2.5 Surrogate models

There is a rapidly growing literature on surrogate models (meta-models), which has not omitted applications to estimations of rare-events probabilities. The basic premise of those models is to replace an expensive model evaluations with an inexpensive, simplified model which is supposed to capture the behavior dynamics of the original one.

In the case of rare-event reliability simulations, the usage of surrogate modelling not always involve direct creation of a meta-model, especially since creating such in presence of rare-events is an intensive task. For that reason, in the context of this study, we only briefly discuss surrogate models that can be used to drive other rare-event methods, predominantly the importance sampling and importance splitting algorithms. A condensed overview of similar scope, with more specific references to extensive literature in the subject, is presented in chapter 10 of [7].

Creating a surrogate model and estimating probabilities using only that one might lead to inaccurate results, and to solve that problem, [50] proposes a hybrid approach using polynomial chaos methodology that combines sampling the meta-model with sampling the actual one which returns more accurate results. Support Vector Machines are reported as a method to construct the limit state functions (failure criterion) in [51]. In [52], authors observe that while the advantage of importance splitting or sampling methods can be huge, their efficient implementation is often a challenge. As a solution, they propose an adaptive importance sampling method based on Kriging meta-models. It allows for quantification of the error variance and thus select sampling points more efficiently. The method was quite successful and has even some more specialized offsprings. A modification of the algorithm, focusing on application of the method to the system reliability estimations, is presented in [53]. In [54] and [55], authors suggest further changes of the algorithm to fully utilize information from the Kriging model and present a comparison of empirical results obtained with their modified approach.

One of the major drawbacks of surrogate models is that they do not always allow for appropriate error estimation, which is a significant problem for reliability simulations.

As side note, in the reliability engineering context, meta-models are utilized with great success in simulations done for structural reliability. Essentially, such models may involve complex and particularly expensive calls to the evaluation function, making them a perfect use case for surrogate modelling. A comprehensive overview of main meta-models in the field is presented in [56].

2.6 Other methods

Other methods which could be used for rare-events to improve the probability estimations include classic variance reduction techniques such as control variates, antithetic variates, stratified sampling or Latin Hypercube sampling [7]. Their more in-depth descriptions can be found in the Ross's *Simulation*, chapters [57] and [58].

The former two methods are based on selecting samples whose co-variance or co-variance-to-variance ratio are chosen in a way that decreases the variance of the overall result. As stated in the survey paper, they cannot be applied efficiently to cases in which analytical formulation of the model evaluation function is not known.

The latter two methods, stratified sampling and Latin Hypercube sampling are more flexible. Stratified sampling is a method used often in statistical surveys, where population is not chosen at random, but according to their status, preferences or choices. This enforces more equal distribution of the samples. In cases of stratified sampling in Monte Carlo simulation, the method divides the space into discrete intervals (*stratas*) and increases the number of samples proportionally to the variance inside them to minimize the variance of the overall result. The biased sampling can be compensated in the estimator definition, making it unbiased again. Simple implementation is discounted by drawbacks such as the need of good knowledge of the evaluation function (i.e., information on where to focus samples) and increasing the error when the sampling focus is not chosen properly. The Latin Hypercube approach helps in cases where it is difficult to choose the sampling focus appropriately. Each sample is drawn from a different equidistant intervals (*stratas*) in which the parameter space is divided. Again, the implementation of the method is straight-forward, but in this case - potential gains are not particularly large.

Chapter 3

Methodology

This chapter introduces a range of considerations involved in developing and integrating selected rare-event MC methods in the existing simulations framework, AvailSim4. The review conducted in the previous chapter provided theoretical foundations for selecting and developing the approaches to be tested in the studied context. This part places the focus on an intersection of practical and theoretical factors involved in robust implementation of those methods.

A typical studied Monte Carlo system reliability simulation needs to account for a number of factors rendering the acceleration more difficult to achieve. Their outline is stated in the list below.

1. The simulations have non-uniform discrete time steps.
2. Failures are often following exponential, or, more general, Weibull distribution – but other probability distributions might be needed too. Even when exponential probabilities are used for individual components, the system typically is still non-Markovian as there are various time-dependencies between components, preventing a straight-forward application of Markov Chain methods.
3. Components have discrete statuses.
4. The cost function is almost binary – the system either fails or remains operational.
5. Models have many dimensions: each component has its own failure dynamic which conceptually is a separate dimension of the problem.
6. Non-trivial reliability dynamic introduced by dependencies created by repair strategies, inspections and other advanced modelling constructs (e.g., fail-safe behaviour).

With this perspective in mind and findings of the literature review summarized in the previous chapter, the focus of the work was concentrated on two distinctly different approaches: **randomized quasi-Monte Carlo (RQMC)** and **Importance Splitting (ISp)**.

The decision to exclude Importance Sampling (IS) method from the considered approaches was made primarily because of the complexity involved in defining complex sampling (importance) functions, tailored to the individual systems. Although a similar function is also present in ISp to decide the splitting thresholds, this one has a less intuitive meaning and translation into results. IS modifies the sampling strategy, for example forcing the evaluations to produce more failures; the created bias must be properly interpreted using a weighting function. The initial attempts to use this method led to ambiguous conclusions and thus this direction was eventually abandoned.

Surrogate models, both in their general variant as well as extensions of, e.g., IS, were also considered. Building meta-models of the entire system could offer some advantage but of a different type than the one sought in this work. Machine Protection reliability simulations rarely require extensive sensitivity analyses – more often, the sensitivity analysis explores specific discrete parameters and combinations. The typical objective is to establish the probabilistic reliability point estimation, with additional estimations completed for sets of arbitrarily selected parameters. Building confidence in results by a more structured approach to exploration of the parameter space could be beneficial, but the goal of this thesis is to create less intensive modes of simulations. For rare-events, surrogate models could have been applied to control IS, ISp or related methods: such extensions, however, are added on top of more fundamental methods and as such might be evaluated in the future.

The simple methods mentioned in the previous chapter, antithetic variables and control variates, also do not appear in the empirical evaluation of the methods due to the listed downsides. Using stratified sampling or Latin Hypercube sampling (LHS) would be of interest, but in comparison QMC offered superior potential for most foreseen use cases. It is worth noting that LHS does not limit the number of dimensions, which is an interesting aspect compared to QMC and should be considered another potential extension for models which do not meet the QMC limit.

Eventually, we chose RQMC and ISp for empirical examination. The RQMC approach on its own may not provide enough acceleration, but it does offer variance reduction and can be easily integrated with other methods. We also expect that it will provide visible improvement for use in availability simulations (i.e. for failures occurring relatively more frequently during the system observation time). The ISp method, apart from

potentially delivering two or even three orders of magnitude acceleration, has also a relatively straight-forward explanation in the reliability domain and thus seems particularly appealing.

For clarity, the thesis uses a number of generic terms which have specific meaning in this context. *Offsprings* are simulations restarted from the same point, sharing the same past up until that point. *Model evaluation* is an equivalent to *DES iteration*, a single simulated timeline of a system life. *Simulation run* is the entire group of *model evaluations* that is used for a MC study, while *simulation campaign* is an aggregation of a number of *simulation runs* (with different parameters, for example to understand how a parameter impacts the results).

This chapter provides information on practical implementation of the methods. It includes a discussion of model requirements, to better present what the methods need to account for and how the reliability simulations work in standard variant. All those elements are already a part of the AvailSim4 framework and are expected to work equally well with the rare-event methods. The second part of the chapter is divided into yet another two sections, one for each method. Practical factors and measures taken to overcome encountered obstacles are mixed with discussion of some theoretical aspects relevant to the methods implementation.

3.1 Inputs of a reliability simulation model in AvailSim4

Before explaining the rare-events methods themselves, this section presents a short overview of common elements defined in a typical simulation performed to estimate the reliability of a system. Those elements are already defined in AvailSim4 and have been used extensively in classic Monte Carlo simulations before.

Component tree

The basis of any reliability model is a structure describing components and their dependencies. In AvailSim4, it is an oriented graph (which, most of the time, is a tree) of components which defines both aspects. They have two types of nodes with corresponding specific properties:

- **basic components** are represented by nodes without outgoing edges. They correspond to actual, physical components or group of components of the system which are failing according to a given failure mode (detailed below),

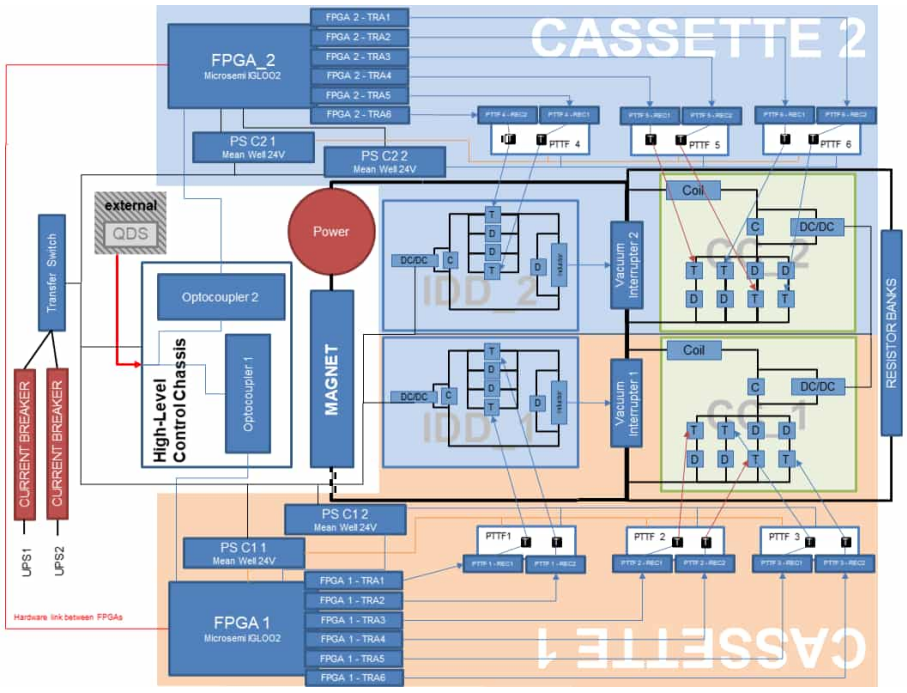


FIGURE 3.1: Block diagram of the Energy Extraction system reliability model. Explanation of the symbols used in the drawing is attached in the appendices.

- **compound components** are nodes which have outgoing edges; they aggregate other basic and compound components, creating failure dependencies. In short, all nodes to which a compound component has a connection are a part of that component.

Typical goal of a reliability simulation is to estimate a failure probability of a compound component (usually the one aggregating all other components) in a simulation where only basic components fail and statuses of the intermediary compound components are decided by those basic failures and a set of rules, depicted by the graph structure. The exact failure propagation works in the following way: basic components are grouped together as children to a more general (compound) component (and sometimes more than one component). Each component is in one of the following 7 states:

1. *Running* – fully operational, with no failures or degradation of children components.

2. *Degraded* – still operational, but some children components are not running (i.e. any status expect running).

3. *Blind degraded* – like *degraded*, but the component is not monitored and no repair action can be triggered directly. In the list of children, at least one component is blind failed or blind degraded – but the component still operates correctly.

4. *Failed* – the component is not performing its functions and may trigger other events, immediate of deferred.
5. *Blind failed* – like *failed*, but component is not monitored and thus no actions can be directly triggered.
6. *Under repair* – failed components being repaired.
7. *Inspection* – the component is inspected, for keeping record.

Every compound component has an additional property: the logic determining its current failure status from the statuses of components of which it is made. In some cases, all parts of the component have to work for the overall component to function correctly (*AND* logic); in others, more than one component could be provide the needed functionality to provide redundancy (*X out of Y* logic). In addition, there might be an even more complicated setting in which only a specific subset of children is required to operate based on a set of additional, user-defined rules.

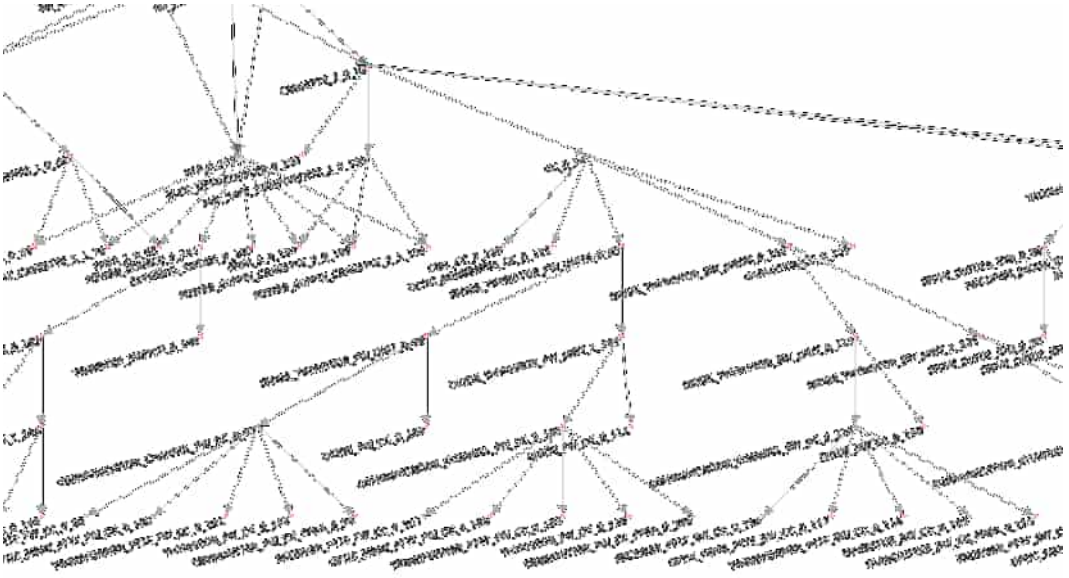


FIGURE 3.2: Part of a dependency graph created for HL-LHC Energy Extraction reliability model.

Figure 3.1 displays an example functional block diagram which needs to be translated into a graph of components for modelling purposes. The diagram has been created as a part of a reliability study performed alongside the thesis (the report can be found in Appendix C). The number of components considered in such models is usually relatively high (up to thousands of components), and functional dependencies may follow intricate rules. One example of such rules is particularly relevant to the safety-critical systems (such as Machine Protection systems): fail safe mechanisms implemented on the system's design level. Failures, instead of lingering and causing a potential problem later, may

trigger themselves a maintenance response, such as safe abort. Functional dependencies in such cases are inverted and require careful consideration in the model. The functional block diagram presented as the example translates into a very complex graph – a snippet is included in figure 3.2. The graph resembles a tree in which successive levels are consecutive abstraction levels grouping components into more and more complex aggregations: for example, groups of thyristors, transformers and other basic elements form a communication channel (*COMMUNICATION_CHANNEL_FW_CC_0_97*), which is a part of a specific unit inside the Counter Current (*CC_0_94*) subsystem located inside one of the cassettes (*CASSETTE_2_0_72*).

Failure modes

A failure mode is a characterization of potential failure event concerning a unique *basic* component or a group of components sharing the same failure mechanism. In AvailSim4, failures change the status of a basic component and start a cascade of updates to compound components of which the original one is part of. Whether and how an aggregating node is affected is decided according to the rules of assigning statuses described earlier.

The probability of a basic component's failure occurring is described using a probability density function (in reliability context, the most prevalent are: exponential, Weibull or normal; some failure modes can also occur in deterministic intervals) with appropriate parameters (e.g., scale, shape, delay, etc.). Most often, the source of those estimations for individual components are data sheets issued by their producers, however those assessments are conservative by nature and may not always apply to the operational conditions of a studied system. Instead, reliability tests are the preferred choice, but those require additional resources and effort. If neither options are available, general models such as the ones included in a widespread industry standard, MIL-HDBK-217 [59], a military handbook prepared in the 90s, are used. Failure likelihoods are expressed in terms of Mean Time to Failure (MTTF) or Failure in Time (FIT – average number of failures in 10^9 hours). When using an exponential distribution, its scale parameter $\beta = \frac{1}{\lambda}$ is the mean of the distribution function and by definition is equal to the targeted MTTF ([60]). The scale parameter is also called inverse rate parameter.

Detectability is a notable property of a failure mode. A detectable failure can trigger other actions, such as immediate or deferred repair process, while a blind failure remains unknown to the system operators – and, thus, is likely to stay in the system for longer. Speaking in terms of the physical system, *detectability* usually means that there is some kind of component's active supervision: a change in the system is visible in the monitoring system and operators can follow up with appropriate actions.

Failure modes can be defined only for basic components. They escalate further up the tree from there – changing the statuses of the components aggregating them. *Blind* statuses have priority over their standard variants since their effects are likely to be more critical as the failure is unknown for the operators.

Minimal Replaceable Units (MRU)

Replacing groups of components rather than a single failed part is another element of a maintenance policy. It is often the case that components are mounted as a part of a cassette, rack or a tray rather than separately – for example, Energy Extraction system is divided into two redundant cassettes (represented in figure 3.1). Such arrangements are in place to maximize the availability of the machine, reducing the intervention time and providing an opportunity for an in-depth failure investigation during a repair off-site. In the CERN context, it might also reduce the exposition of personnel to radiation while intervening in tunnel areas. Given that such setups are often utilized in designs and that they potentially change the failure dynamics by removing undetected blind failures, their inclusion in the reliability model is an indispensable requirement.

Inspections

Finally, reliability dynamics are impacted by regular inspections, which might be the primary countermeasure against blind failures. The system inspections can be of various nature – from visual inspection to dedicated system testing – and typically occur in regular intervals and last for a predefined time. A specific inspection may be performed only on a subset of all components. Each inspection resets the time of the component's next imminent failure drawing a new sample and repairs accumulated detectable failures. Alongside MRUs, inspections are the only way to repair components which failed undetected.

Their frequency is often an interesting aspect for further exploration within acceptable boundaries in order to optimize the life-cycle costs of the system (including operation and maintenance costs) while maximizing its reliability. Performing a study of this aspect can help to optimize the workload, also accounting for various systems requiring maintenance, and to better understand its usefulness.

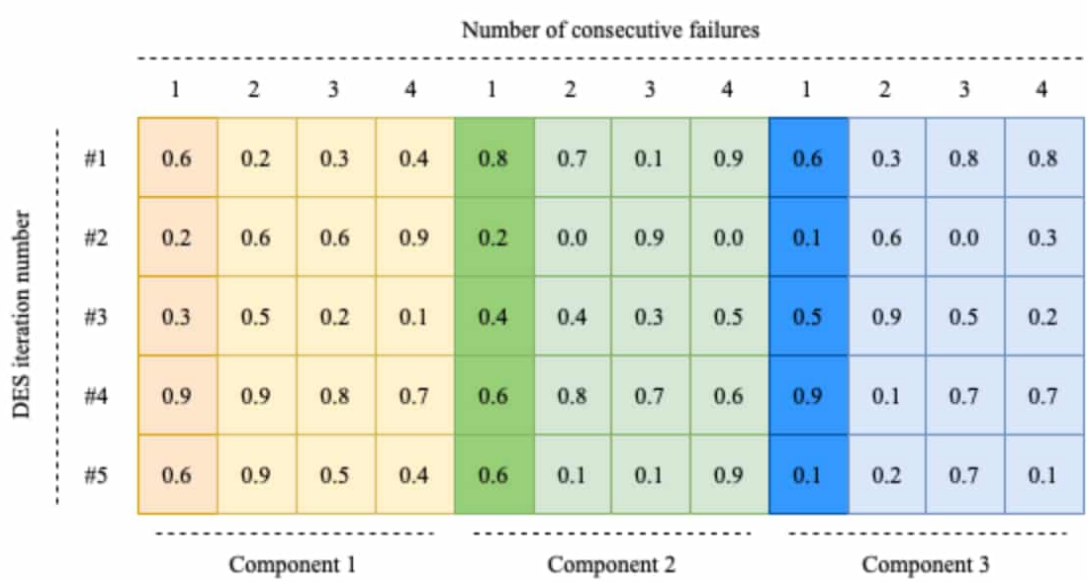


FIGURE 3.3: Matrix of random values generated using randomized Sobol sequence for maximum of o4 successive failures in 3 components over 5 model evaluations.

3.2 Randomized Quasi-Monte Carlo sampling

Generating RQMC samples is supported by several open-source libraries, of which QMCPy ([61], [62]) offered most fitting capabilities for the intended use in this study. Although flexible integration of the library with the software framework allows for using and testing various low-discrepancy sequences, we expect Sobol sequences to deliver the best results, and that is the reason of additional scrutiny targeted at them in this section too.

Although the simulation approach is conceptually very similar to the one used in CMC, it required some additional changes of the simulation implementation. In many other cases, the implementation of RQMC on top of a standard MC procedure is trivial. In the presented case, each DES timeline is defined by an unknown quantity of random numbers. Generating samples like in CMC, values one-by-one, is no longer valid, since the point of LDS is to cover equally and uniformly the entire problem space with consecutive samples. To take advantage of it, a single generated sample has to provide enough random numbers for all events in the system. If the generator were to draw one-dimensional samples one-by-one for successive events, they would be well-distributed only within that single evaluation. The overall results would not get to utilize the method's potential.

To use RQMC, the algorithm has to generate a sample for the entire evaluation – their optimal distribution properties can be then best leveraged. In this approach, all samples are multidimensional vectors rather than separate pseudo-random values. Instead

of drawing samples *on the fly* all random numbers used in a specific evaluation are generated beforehand and stored in a matrix. Figure 3.3 presents a conceptual fragment of such a matrix. The idea used in the RQMC simulation mode implementation is to treat each dimension as a separate failure of a given component. Each sample (represented by a single row) is linked to a single DES evaluation of the model. Groups made of a predefined number of columns are describing consecutive failures of each individual component (illustrated with distinct colours in the figure). In the portrayed case, sample's 1st dimension is the 1st failure of the component 1, 2nd dimension is its 2nd failure, while 5th dimension is the 1st failure of the component 2, and so on. The dimension of the matrix is then $I \times F$, with I the number of iterations needed in the RQMC algorithm, and $F = \sum_{i \in [1,n]} F_i$ being the total number of failures expected with n the number of components and F_i the number of failures for the component i. While the dimension I can easily be set by the user, the dimension F is more challenging because the number of component's failures is different in each evaluation and cannot be foreseen before, since it depends on events in the simulation and is essentially the estimated quantity. Consequently, the code should try to produce as many failure samples as model evaluations might require, but the value has to be as small as possible at the same time due to theoretical memory issues and very practical limitations of LDS generators. The number should be higher than the number of repairs, inspections or any other events triggering renewal of the failure time, so that failure times can be renewed according. The upper threshold is defined by two practical factors: the overhead of creating unnecessary samples and the number of dimensions that are possible to obtain from an LDS generator. The latter factor, a generator's maximum number of dimensions, is a stiff limit imposed by individual libraries used to generate samples. In this project, we use a well-established QMCPy library, which provides means for generation of more than 20,000 dimension Sobol sequences – for other LDS, the limit is noticeably lower. For practical purposes, the number of samples dimension is defined as the number of average occurrences of the most frequent event multiplied by a parameter set arbitrarily by users.

In case the estimation of the number of random value needed was too low (or could not be bigger due to the maximum number of dimensions), a practical solution is implemented in the software code. After the program runs out of randomized LDS samples, it switches to standard pseudo-random numbers generated *on demand*. In practical terms it means that simulations receive a randomized LDS to use – once the entire sequence fragment allocated to a component has been utilized, next samples for that component are obtained from a pseudo-random number generator. The result is a hybrid approach, efficiency of which might be diminished, but which is still able to leverage the advantage given by RQMC approach. Such a solution does not impact the advantage obtained

by the simulation part where LDS samples were used, it only switches to a different sampling strategy when no more dimensions are available.

All elements of a randomized LDS are uniformly distributed values from inside a unit hypercube. Those values can be easily transformed into other probability distributions using inverse cumulative distribution functions, which are also known as quantile functions. For example, for exponential distribution the cumulative distribution function is defined as:

$$F(x;\lambda) = 1 - e^{-\lambda x}$$

for $x \geq 0$ and where λ is the rate parameter. The quantile function is:

$$Q(p;\lambda) = \frac{-\ln 1 - p}{\lambda}$$

By substituting p with a single value from a RQMC original sample, one obtains a sample from the exponential distribution.

The QMCPy framework used in this project provides two algorithms to generate Sobol sequences, both summarized in the note [63] and discussed more extensively in [64]. The first method uses primitive polynomials and bitwise operations to obtain consecutive elements of the sequence. A more efficient - and flexible - generation method additionally leverages Gray code [65], in which each number differs by just one bit, enabling the algorithm to generate the points recursively. A significant advantage posed by the latter method is that the sequence can be generated from any given sequence point. This is important for distributed use of AvailSim4 code, since each instance has to generate only samples that are useful to it (and not all preceding elements of a sequence, which would hinder the entire process).

Another aspect, which has been vividly discussed by communities using QMC methods, is skipping the initial point of a Sobol sequence. It is extensively discussed in note [66]. Generally, omitting the first point of the Sobol sequence, which happens to be $\overrightarrow{0} = (0,\ 0,\ \ldots,\ 0)$ (visible in lower left corner of the center plot in figure 2.1), is a way to avoid problems created by the $\overrightarrow{0}$ point. For instance, transforming samples into Gaussian distribution in such case is bound to fail and return a non-numerical value. Skipping the initial point also makes sense from a practical perspective of applying those number in our DES simulations – zero point essentially produces a simulation in which every component fails immediately after each consecutive repair. Such a sample not only leads to very unlikely evaluation, but would also block the DES by preventing it to move forward, as all actions take place without any delay in time 0. This would happen provided the sample had unlimited number of dimensions; given that it is impossible,

the implemented algorithm would use all QMC samples in time 0 and then possibly proceed further with pseudo-random samples.

Those advantages, as observed by Owen, are countered by a number of other problems, with both theoretical understanding of the resulting sequence and also inferior overall performance of the method. In the studied case, however, the initial point problem does not need to be addressed explicitly, as the algorithm is used by default in its randomized version – which maintains all digital net's properties while avoiding the actual zero point. The default mode of randomization used by the QMCPy library is linear matrix scramble with digital shift, which will be used in the tests. More details on RQMC and randomization methods can be found in [67].

The last aspect is distributed processing, which is quite important for potential reliability estimations given that expected speed-up using QMC is not considered as enough. Assuming that each iteration of the QMC (evaluation of a timeline) is expected to be approximately of the same length (with certain degree of variability), the total number can be equally divided to distribute the workload across all available nodes. Each iteration has to use a specific element of that Sobol sequence. In the Gray code implementation of the Sobol sequence generator specifically, each instance can compute the selected part of the sequence, which keeps the work viable and balanced on every node. The results can be reduced afterwards exactly as in the CMC approach, for example utilizing the post-processing tool (which is one of the other contributions to the AvailSim4 code, discussed shortly in Appendix D).

3.3 Importance Splitting

Importance splitting in Monte Carlo simulations is a fairly powerful technique based on a reasonably simple concept. Its foundation is the division of the overall (very low) probability into a chain of conditional probabilities that are more likely to occur in the simulation lifetime. Through estimating conditional probabilities of increasingly likely events, simulation can eventually establish the probability of the overall rare event. The process is much more effective because model evaluations are restarted in each level separately – by drawing samples from approximations of conditional probability distributions of successive phases. The conditional probabilities in consecutive phases are higher than the likelihood of the overall rare event, and thus the simulations are more likely to observe rare events proceeding this way.

An illustration of the standard splitting method is included in figure 3.4. The plot illustrates example model evaluations in terms of their importance function across the

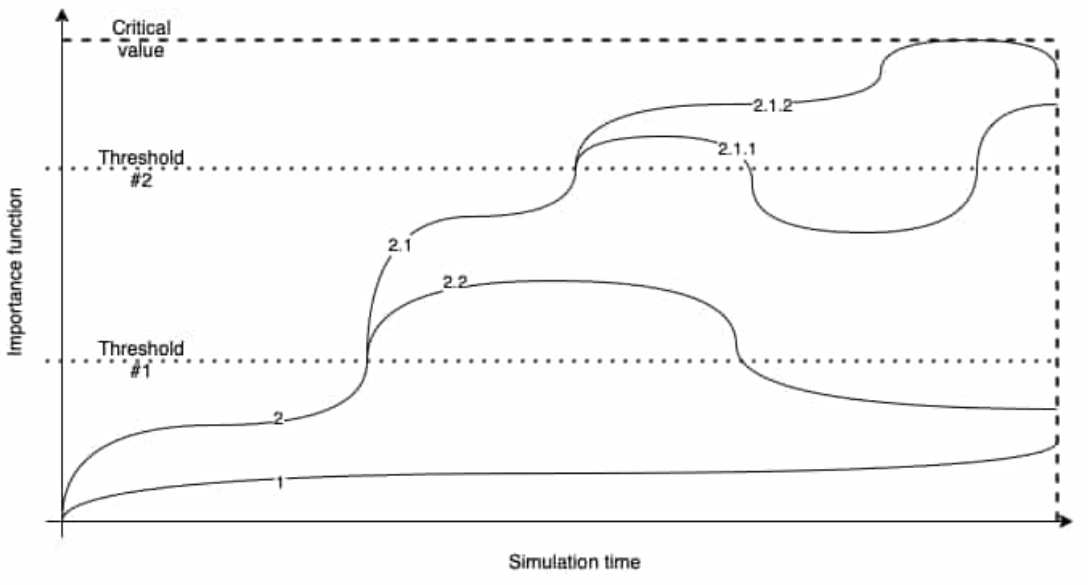


FIGURE 3.4: Example evaluations paths of in the ISp algorithm. For clarity, the method illustration is simplified to only 2 restarts per level.

entire simulated lifetime. The importance function values are divided into three fragments by thresholds – arbitrarily selected values which indicate when evaluations are to be branched. The first level evaluations (number 1 and 2, the ones starting below the threshold #1) begin in the initial point of the simulation. The first evaluation reaches the end of the simulation time without hitting the first threshold and, as a result, ends without branching at any point. The second evaluation does hit the threshold. The branching at this point produces two off-springs (simulation paths which are exactly the same until the branching moment), starting at the same state in which evaluation 2 was when it crossed the threshold. Evaluation 2.2 does not hit the next threshold but instead reaches the end of the simulation lifetime. In the meantime, it crosses the threshold #1 – since it is the standard splitting mode, it continues further. Evaluation 2.1 is more successful and does hit the threshold #2. As it was done after hitting the threshold #1, two new off-springs start at that point. In evaluation 2.1.1, the evaluation temporarily goes below the threshold – the simulation is not branched again when crossing the same level again. That one would undergo another branching only when hitting a threshold level higher than the one already encountered (in this case, a third threshold, which does not exist). Evaluation 2.1.2 hits the critical value before finishing at the end of the simulated lifetime.

The risk carried by using the method is similar to the one in importance sampling: an incorrect definition of the importance function may significantly increase variance rather than reduce it. If a specific rare event is not working well with the critical function, the

importance splitting might omit its existence entirely. This problem needs to be addressed by using appropriate methods for choosing that function and by introducing additional verification of the model results (for example, through deterministic identification of the critical paths). Furthermore, choosing appropriate splitting parameters will play an essential role in the accuracy of the results. Overall results are a product of estimations completed in each level, therefore level errors are projected and may be visible in the final value.

A side note: the RESTART approach (not implemented and thus not included in the empirical results) would have the branches truncated every time an offspring crosses a threshold from above. Such a strategy would likely fit the reliability context well. In terms of the tested weights, crossing the threshold from above is equivalent to a finished component repair. Such a case is no more interesting from that point than the one in which nothing happened. After the system goes back into a safe configuration, removing such evaluations is beneficial in terms of the overall computation time.

Generating all minimal critical failure paths of the system

The simulation aims to measure the likelihood of a system's critical state configuration occurrence. Whether a given configuration is in a critical state depends on failure logic defined on various component levels. This algorithm is supposed to provide an exhaustive list of all such configurations.

The first step of the algorithm is to identify all minimal critical failure paths, which are understood as minimal sets of components whose failures cause the overall system to fail critically. Obtained results can be used to tune the algorithm's parameters better but also constitute a valuable additional feature on its own, providing information complementary to the main estimations returned by the software framework. Most importantly, it surveys failure paths that may be overlooked when using incorrect parameters, such as rare single points of failure (those can be then subjected to a more targeted analysis). The long-term goal of identifying minimal critical failure paths is to provide an alternative, more sophisticated way to define the importance function – the proposal of the idea is outlined in the next pages.

A naïve, brute-force approach is to iterate through every potential configuration of all components states and classify them according to whether they cause a critical failure. However, this approach is inefficient and likely to fail when examinations involve a large number of components and intricate dependency rules. All configurations, assuming there are only two possible states a component might have (e.g., operating and failed) of N individual components, is equal to 2^N. Iterating through every potential combination

is a tedious task. Another issue is that a significant share of the recognized states would be redundant: system configurations leading to a critical failure, without the minimal property, contain all combinations of states not directly involved in the critical failure. The minimal set contains only information on components which combination causes the critical failure, and the states of others are disregarded. A comprehensive list of configurations contains statuses of those in the disregarded part, describing every potential configuration that returns a critical failure. Such an approach also causes a significant increase in terms of space complexity and therefore, a more efficient solution is needed to address this problem.

Algorithm

The algorithm is a depth-first graph traversal algorithm, conceptually similar post-order tree traversal algorithm. It is based on the following recursive notion: for each node, return the list of critical failure paths present in the subgraph reachable from that node. All node's outgoing edges must be visited (and return results for their subgraphs) before the node can answer for itself. The base recursive case for the method is a node without any outgoing edges, which should always return itself as a critical failure path – since the system defined by that subgraph consists only of that component. The algorithm is depicted in the block diagram attached in figure 3.5.

This algorithm produces the requested results in reasonable time. The computational complexity is $O(n)$, where n is the number of nodes, equal to the total number of components in the model. It is the case since each node of the tree is visited at most twice, which means that the total number of visits is $2n$. The result of this algorithm is an exhaustive list of system configurations which lead to the fault of a root component. As systems in AvailSim4 are not all described by trees, but rather graphs (because some components can have two or more parents), a verification step is added to the process to ensure no failure configuration is included in another one. Indeed, if one system failure configuration is included in another one, then the second one contains faults which do not necessarily lead to a critical event. The additional check eliminates this problem.

As this list might be consulted frequently, the efficiency of the distance lookup is particularly important. The concept of importance splitting effectively makes it required to assess the criticality of the system configuration whenever there is a relevant change in the system. Such checks should be performed as little as possible to save computational resources, but for the algorithm to work correctly they might be necessary every time any component changes its state to "failed". In this view, if the lookup operation adds considerable overhead, it might render the entire approach counterproductive.

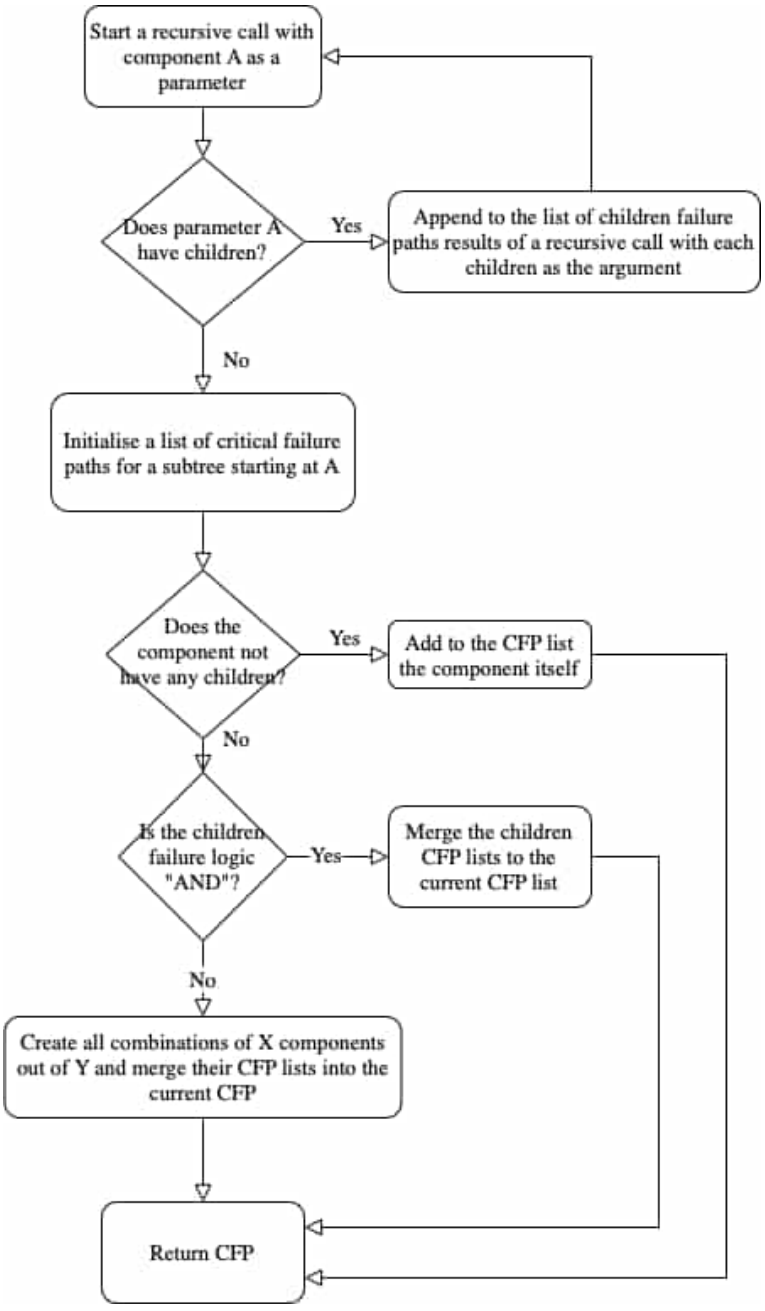


FIGURE 3.5: Block diagram of the minimal critical failure paths algorithm.

One implemented idea is to translate a list of failure configurations into a sparse matrix C, which can be used in fast calculations thanks to optimized numerical algorithms implemented in the utilized Python library. The matrix will have size $n \times m$, where n is the number of basic components in the model, and m is the number of critical failure paths identified in the previous step. Each matrix's row corresponds to a single path, where 0 stands for no specific state and 1 for a given component's failure. The distance to the critically is then just $min(C - \vec{1}c) = d$ with c a vector representing the current state of the system and $\vec{1}$ a unit vector used to transform c into a matrix of the size of C.

Another aspect of the problem is the need for appropriate likelihoods approximation of reaching a critical failure path. The straightforward approach to this issue is establishing a metric in which the distance is the minimal number of failures that yet have to occur for the system to reach its critical configuration. This idea has a considerable flaw, as it quickly leads to situations where, e.g., improbable paths with one or two components are prioritized over other paths that involve more components but which are decidedly more likely.

The idea which has been attempted is based on extending the distance metric with weights vector W that account for additional factors: $max\ W \times (C - \vec{1}c) = d$. The goal for defining weights is not to achieve a very accurate path probability estimation, as achieving it would also be a solution to the question for the entire Monte Carlo simulation. If we can establish a likelihood of reaching the critical failure path with high accuracy at any given point, we could also do that for a system in the initial state. What is sought at this stage is simply an approximation that could help diminish the importance of the less likely scenarios. The exact weight values will be used to sort the critical failure paths according to their significance and likelihood of appearing in subsequent simulations. That value is influenced by many factors, including primarily MTTF, effective MTTR (the period between failure and reparation), inspection frequency, repair strategies, and many other aspects. Often, the most critical factor is whether a failure is detectable. This optimization step is directly linked to the number of thresholds and their values at which the algorithm should split and repeat a timeline.

Splitting mechanism in DES evaluations

DES evaluation procedure must be altered to accommodate the needs of the improved algorithm. First, each model evaluation should monitor the criticality of the system configuration (via the importance function) after any of the components enters any of the *failed* statuses or returns to *running*. Second, each evaluation has to be capable of stopping itself upon registering the system in the critical region and restarting multiple copied instances from there.

Monitoring the system status in terms of the importance function requires adding a slight overhead on top of the plain DES evaluation. The loop iterating through the timeline is generally agnostic to the type or content of events – therefore, any event can be one of interest to the criticality value. Thus, calculation of the importance function value needs to occur every time there is a jump between events in DES, after completing phase B (three-phased approach, described in chapter 1). Estimation is not a particularly

complicated task, but due to its potential frequency, it might be happening very often - and thus, it is vital to optimizing the weighting function as much as possible.

Before running any model evaluations, the system's consecutive thresholds need to be established. It is a particularly sensitive part of the algorithm, likely to significantly impact the performance. In general, the critical threshold choice is strongly connected to the weights defined in the critical failure paths identification. Setting the bar too low will be triggering branching frequently – likely overwhelming the simulation. In the worst case, each initial model evaluation will trigger branching, effectively creating many simulations that do not differ much from standard Monte Carlo (although the accuracy might be slightly diminished, as parts of the simulations will be shared among multiple evaluations). On the other hand, the bar might be too high, with the worst scenario where none of the simulations triggers additional instances. In this case, the simplest solution would be to lower the threshold by a certain margin and try another batch of simulations.

This thesis considers two methods of implementing the splitting mechanism. They are known in literature as *global stepping* and *single stepping*. The first is to start a fixed number of evaluations for a given threshold and restart additional instances only after the entire initial batch has finished. It is a more well-structured way to tackle the problem, which offers more flexibility. However, it puts unnecessary strain on available memory: each evaluation entering the next critical region needs to create a snapshot of its state (i.e., save all necessary data from resuming from those instances later). When the model is large and the number of instances passing to the next level is considerable, problems may arise. The *single stepping* approach is to proceed with instances produced by branching immediately – more alike depth-first search (DFS) algorithm. It requires holding only a single state per threshold in memory. In runs depicted in figure 3.4, for the second threshold, *global stepping* would first finish running paths 2.1 and 2.2, to only then proceed to paths 2.1.1 and 2.1.2. In *single stepping*, after 2.1 reaches the threshold, the simulation engine would run 2.1.1 and 2.1.2 to return to 2.2 later.

The downside is a slightly more complicated implementation. One of the top concerns is the memory space needed to store intermediary results: given that the algorithm is supposed to deal with potentially hundreds of thousands or even millions of evaluations, all timelines cannot be stored until the end. For that reason, the algorithm keeps track of statistics for every event and updates them directly after the evaluation finishes. The number of times each instance in the level was restarted is essential to update the statistics. A solution to that problem is to summarize the results per level first and combine them across all levels. Nevertheless, in this thesis, we decided to go with the

first approach due to time constraints. It is a likely improvement to be considered in the future work on the method.

Restarting simulations at a given point requires a bit of caution in the studied discrete event simulations. In the implemented three-phased approach, each component generates a set of potential events that are characterized by the time to execution set according to the sample from a corresponding probability distribution function. In the simulation loop, the event with the smallest time value is the next imminent one. When simulation crosses the importance threshold, it is due to a status change in a singular component. Off-springs are supposed to start from that exact point in simulation time. In practice, this means re-sampling all future events starting after for the triggering one.

For this approach to work, the time is recorded whenever a component's effective time to failure is assigned. When a branching after reaching a threshold happens, each event in the queue obtains a new imminent time to failure, which is added to stored previous assignment time. However, this creates a possibility for samples to change the past, as we assume the evaluations restart from the triggering event. For that reason, we need to discriminate against those samples by discarding them and generating replacements. Given that we want to restart offsprings from exactly the triggering event (as this is the weight of simulations we can calculate), it describes precisely the situation that splitting is dealing with: the triggering event took place and, as such, no other event took place before it.

This chapter presented details of AvailSim4 models and introduced most of the practical considerations involved in implementing rare-event methods in Monte Carlo simulations. The specifics of the . In the next chapter, the both implemented methods will be tested and validated on a series of examples illustrating important aspects of reliability simulations.

Chapter 4

Experiments & Results

The primary goal of this chapter is to introduce experiments designed and performed to evaluate the rare-event Monte Carlo methods implemented in this thesis. Those efforts first and foremost attempt to assess how the discussed improvements can help Monte Carlo simulations in the system reliability and availability estimations for safety-critical systems with a high degree of redundancy (such as LHC Machine Protection systems or LINAC4). Given the complexity of the methods, their specific applications and the intricacies of the discussed use case, empirical tests and examinations of the results are indispensable to answer the questions set for this thesis, as well as help build confidence in the optimized approaches.

Alongside those goals, the experiments also aim at exploring specific aspects of the methods by comparing different parameters and options to obtain valuable insights for practitioners in the rare-event field and future users. In both studied methods, the amount of adjustable or optional parameters – whether it is type of a LDS or importance function definition – makes them very susceptible to the decisions made by users. Results presented in this section should explain some of those issues, providing guidance for future development of these methods (e.g., through automatic selection of parameters).

The first section describes shortly the two testing environments utilized for performing tests. The latter part of this chapter presents the experiments including their design, models and eventually results. Their broad discussion is conducted in the next chapter.

4.1 Testing environment

Tests performed in this thesis were executed in two computing environments:

1. PC with eight cores CPU enabling parallel processing and low-level code optimizations

2. HTCondor ([68]), a computing cluster at CERN.

Generally – as seen later – tests with Importance Splitting are significantly faster and thus do not require that much computing power, so they were executed locally on a PC. More extensive runs (above $1,000,000$ evaluations of individual timelines for a unique set of parameters – single point in a plot) were performed on a cluster, using the distributed execution tools that had been available already prior to this work (like multiple jobs submission) and improved to easily handle submissions of up to 10 000 jobs.

The computing cluster at CERN is running on HTCondor workload manager exploiting approx. 200,000 cores. It is an example of High Throughput Computing (HTC) systems, which, opposite to High-Performance Computing (HPC), prioritizes long-term reliability and performance of long-running jobs over providing outbursts of computing power for short periods of time. The primary concern is stability and performance over an extended time since this is a typical configuration of a single job. On the other hand, HPC clusters are better suited for jobs utilizing a large part of the resources for a limited time, often communicating between separate nodes.

For those reasons, on HTCondor, submitted jobs have to be sequential, i.e. single-instance applications instead of parallel programs. Different instances cannot communicate with each other, excluding factors such as latency and other network communication issues between nodes. As such, a single AvailSim4 simulation campaign has to be divided into a set of many independent jobs. Results are saved independently and subsequently combined into the final results using another dedicated program. Appendix D describes development of the post-processing application, used to combine partial results created by a distributed execution of AvailSim4.

The CERN HTCondor computing cluster is a heterogeneous cluster, which means that allocated nodes might differ in speed, performance and offered capabilities. This is a severe inconvenience for benchmarking, including in cases such as tests performed for this thesis. Ultimately, the lack of consistent execution time measurements combined with yet to-be-completed optimizations would introduce a likely 10-20% noise in the execution time measure. Results are therefore expressed in number of computations of individual timelines required for a simulation to converge. The execution time is expected on average to follow the speed-up observed with respect to the number of timelines executed.

4.2 Simulation results

The experiments and their results are presented in this section. The first part is devoted entirely to the QMC method and availability simulations, while the second – to both ISp and QMC in the context of rare-events.

4.2.1 Availability simulations

The RQMC method was tested using two separate system models:

1. **Basic model**, an artificially created configuration with 1, 3, and 5 components and varying failure frequencies.
2. **LINAC4** availability model, already developed before this project started and used as a benchmark.

The first goal is to evaluate the approach in the availability context and both models are examples of those rather than of reliability simulations (i.e. one expects a relatively high number of failures in each simulated timeline). The focus is set on exploring the optimal settings and exploration of options and parameters in a controlled environment with known theoretical solutions. The rare-events aspect is not present in the runs presented in this section; advantage in those is illustrated in the next section.

Basic model

The model in this configuration is simplified as much as possible for the detailed investigation of the benefits of QMC. It contains a single compound component aggregating other 1, 3 or 5 basic components, which are all instances of the same type of component. The system is considered unavailable when any single basic component fails during a simulation with a time duration of 1 hour (hour unit is added for clarity; generally the software operates in units of time, as everything is normalized). Performed tests cover MTTF values set to 0.5, 1.0 and 2.0 hours across all components at the same time. Results of the test with basic model with MTTF set to 2 hours are presented in figure 4.1, which shows the convergence of the availability metric (left side) and the elapsed time (right side). Figures for the other two parameters and raw data are attached in Appendix A. Additional strategies and elements (such as inspections, group replacements, phases) that could potentially be included in the model are at this stage purposefully omitted.

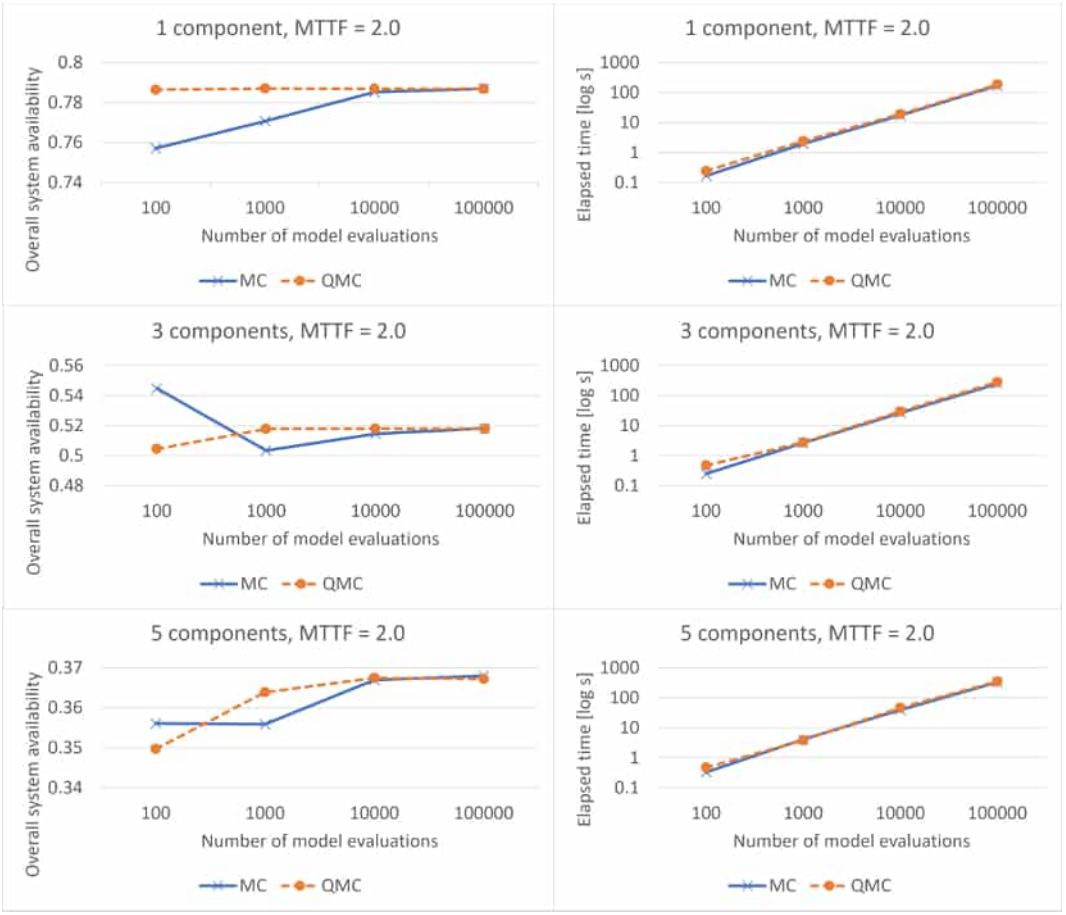


FIGURE 4.1: Comparison of results for basic model obtained with RQMC and CMC methods.

The estimation of the theoretical value of the unavailability metric ($\prod_{i=0}^{N} \frac{\text{MTTF}_i}{\text{MTTF}_i+\text{MTTR}_i}$, where N is the number of components) is not equivalent to the quantity estimated in the simulations, since it does not account for the repair time with a finite and constant duration in opposition to the random MTTF. The target value is approximated as the average between results obtained by MC and QMC methods in the most extensive run. Figure 4.2 shows differences between availability values and the said approximation.

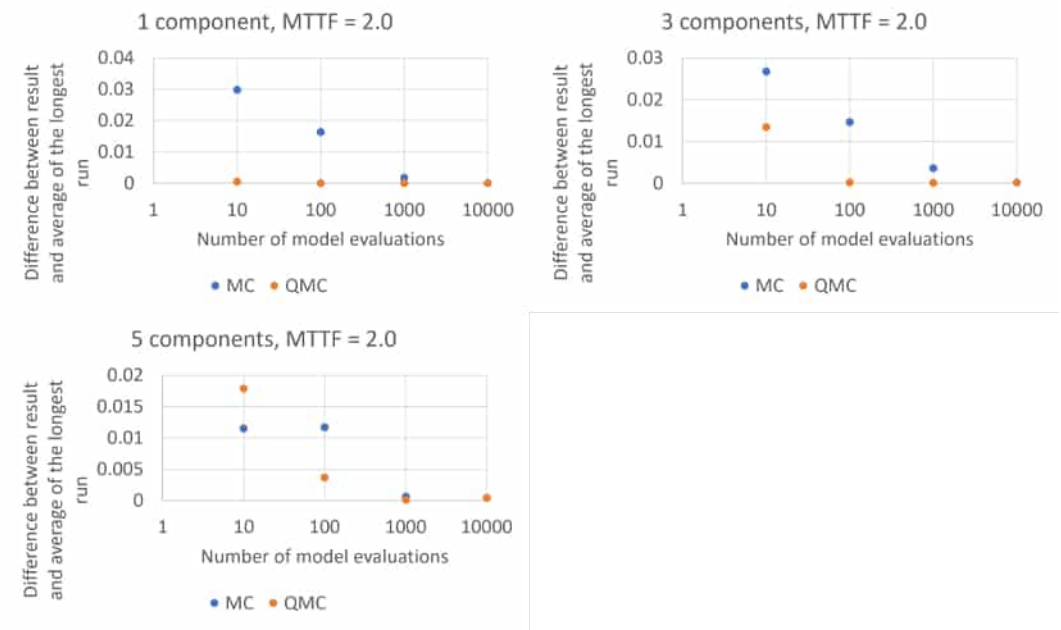


FIGURE 4.2: Error of the estimation in basic model simulations with RQMC and CMC methods.

Almost all tested cases with the basic model observe an increase in estimations accuracy when using QMC. As hypothesised, the method works best in runs which do not utilize many samples: QMC results usually provide a reasonable estimation of the results already with a really small number of evaluations, while MC is susceptible to noise. As shown in plots in the previous chapter and the Appendix A, in the case of the basic model, the results are relatively accurate already when using only 100 evaluations. Only in one case – 5 components with MTTF set to 2 – MC provided a better estimation (although already at 1000 evaluations, the QMC advantage is visible, suggesting that this may have been caused by noise). Out of the remaining eight parameter configurations of the basic model, only in one test case (of five components with MTTF set to 1), both simulation modes provide comparable results and similar convergence (and even then, there is a slight advantage for QMC). In all other cases, QMC provides a relatively accurate answer with the smallest number of evaluations, while MC needs 1,000 or even 10,000 evaluations to reach a similar accuracy level.

LINAC4 availability model

The second test case of the QMC method is the availability model of LINAC4. The model had been prepared before this project was started and can be considered part of the AvailSim4 framework. It is often used as a benchmark or test case; for the purposes of validating the results of using the QMC method. In this thesis it is re-run it for different simulation lengths (25, 250 and 2,500 hours). The method is run with

randomized Sobol sequences (other LDS are the subject of the next experiment). Such a choice of simulation lengths changes the average number of events inside evaluations. The purpose of the test is to observe how shorter and less-extensive runs benefit from infrequent sampling, while, hypothetically speaking, the advantage might diminish when samples are drawn more often.

The model consists of 12 basic elements and one compound element aggregating all of them together. Since it is the availability metric considered in this model, the components are covering all accelerator sub-systems and their infrastructures (for instance, electrical network or accelerator controls failures). Each of them has a failure mode, which expresses its probability of occurrence using exponential distributions with individually set scale parameter by the MTTF. Each failure mode has also a defined repair time which is chosen individually as deterministic rather than probabilistic value, set equal to the MTTR. The model does not contain features such as operational phases, minimum replaceable units or inspections.

Results of the test with the LINAC4 availability model are presented in figure 4.3. The availability estimations are provided on the left side, while the duration of the simulation is on the right side. There are three sets of results for, from the top, 25, 250 and 2500 hours of observation time. In the way QMC is implemented in the code, the shorter duration of the simulation makes it use less dimensions – and thus comparing different values might give interesting results also in terms of the execution time.

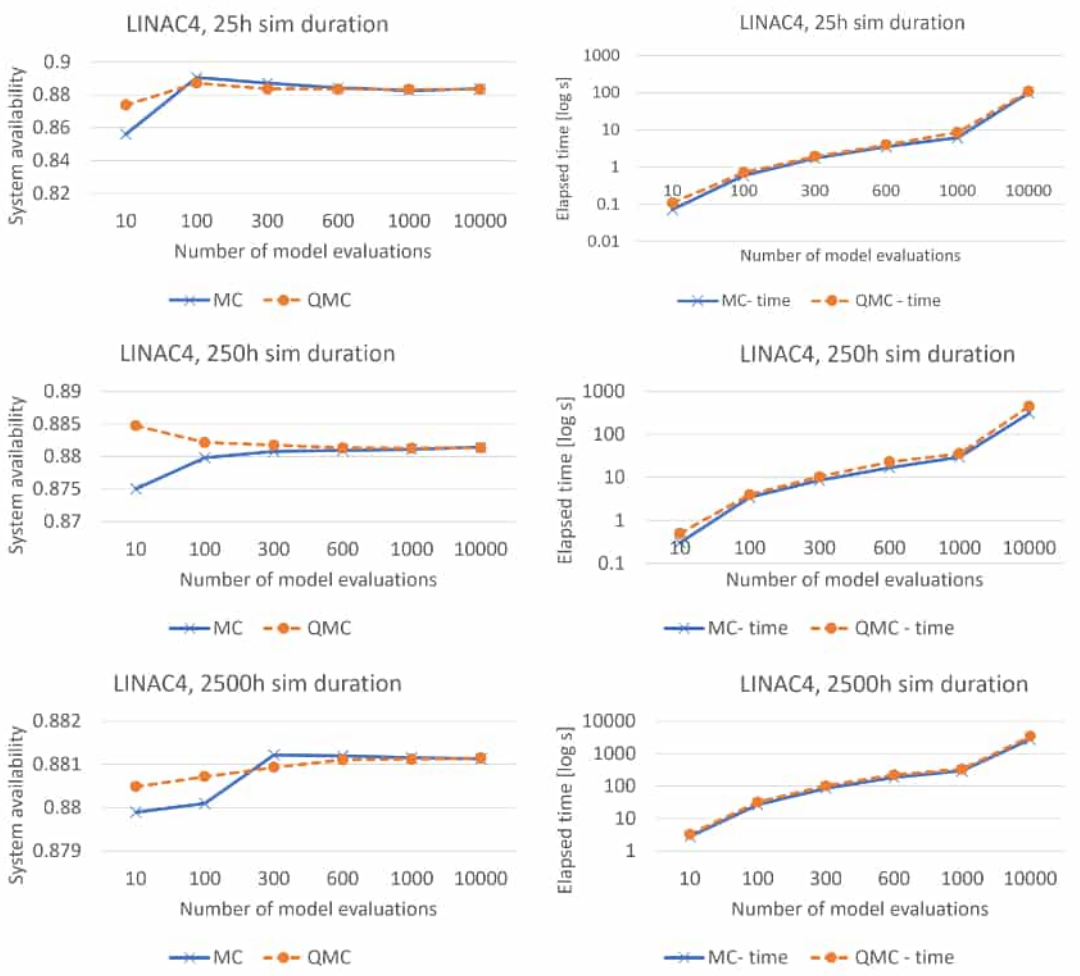


FIGURE 4.3: Results comparison for basic availability model obtained with RQMC and CMC methods.

Similarly as was the case with basic model, it is difficult to establish the exact error for both QMC and MC in those simulations. For this reason, figure 4.4 displays the error in relation to the approximated result which is taken as the average between results obtained by both methods when running largest number of evaluations (in this case, 10,000).

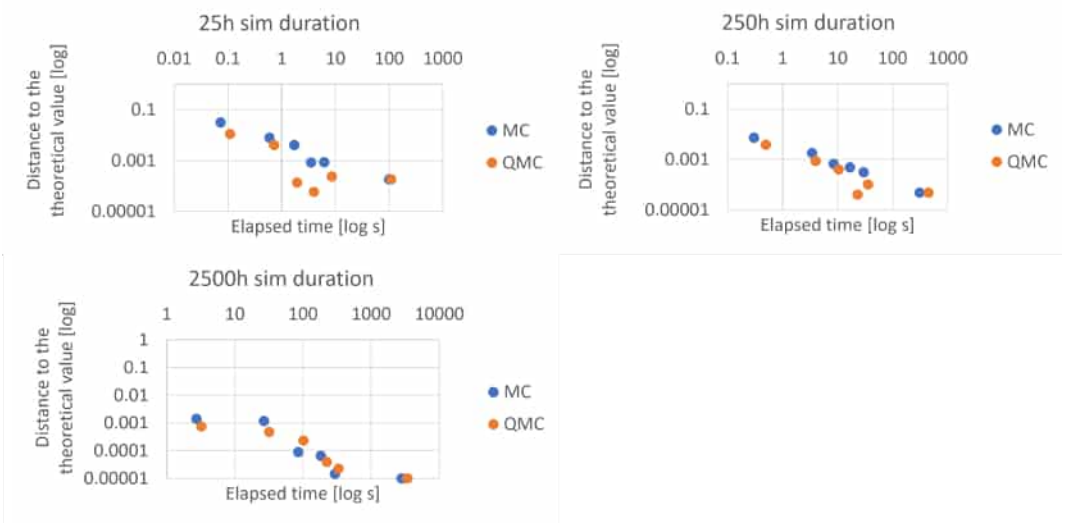


FIGURE 4.4: Relative error in simulations of LINAC4 availability model with RQMC.

In LINAC4 availability simulations, the general takeaway is the same: RQMC converges to the correct result faster, with a lesser deviation. However, the advantage appears to be much smaller. One aspect compared using the LINAC4 model is the impact of the simulation duration on the overall availability estimation. Clearly, fewer events occur in each evaluation when the simulated length is shorter. Therefore, it is expected that the comparison will be more favourable for the QMC method while performing short simulations where MC could not converge as fast. It is the case only in the smallest number of evaluations (10) – for 25 simulated hours, the error of QMC is approximately 0.01, while of CMC, almost 0.03. At the 250 hours, the differences are one order of magnitude lower. The same thing happens in simulations of 2,500 hours. The availability runs with LINAC4 model are especially difficult for the QMC mode, since even within a single evaluation there are enough events for it to return a relatively accurate value.

As seen in the execution time plots, simulations completed with QMC take more time than simulations with MC for the same number of evaluations. It could have been expected given the changes needed in the QMC approach. The literature claims that LDS samples can be generated more efficiently than CMC ones, as the transformations used in these are less demanding than those used to produce pseudo-random sequences. Another aspect favouring QMC is that a single function call performs all sampling altogether before simulations start instead of making multiple small calls from within the DES loop. Nevertheless, this advantage is mostly invisible in our simulations. The current implementation likely generates many times more individual values for QMC-based evaluations, many of which are never used. Empirical data shows that QMC runs always contain a relatively small overhead compared to CMC. The positive aspect is that the difference is not growing but remains constant when the number of evaluations

increases. If the difference were growing, the approach would be not scalable and thus problematic to use in more extensive simulations.

Low-discrepancy sequences comparison

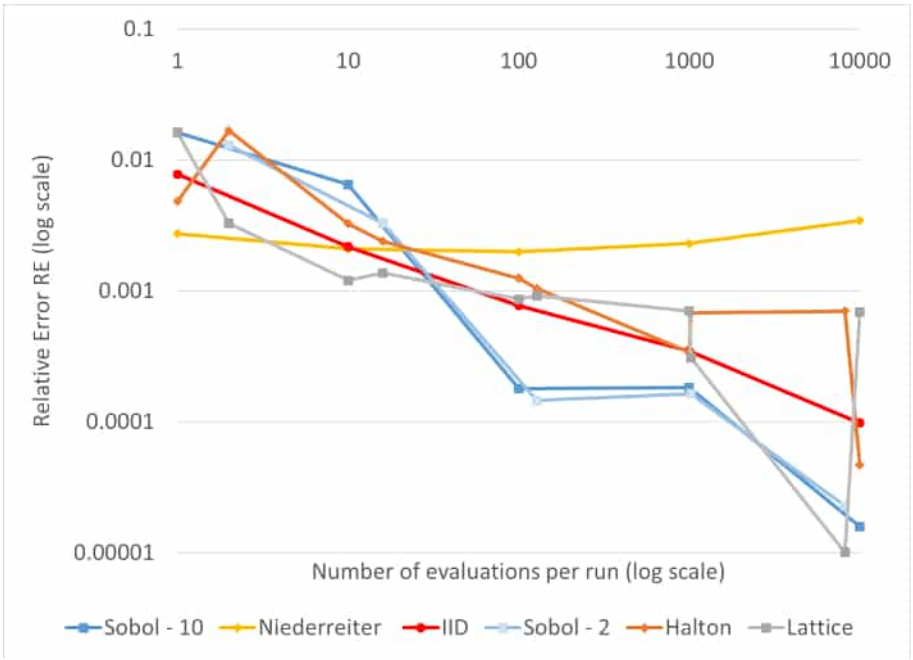


FIGURE 4.5: Relative error of simulations using Sobol, Halton, Lattice, Niederreiter and IID randomized sequences.

The last QMC test is a comparison of the popular low-discrepancy sequences: Sobol, Niederretier, lattice and Halton sequences, with a baseline sequence of independent and identically distributed samples generated in the exact same way as in the LDS approach. The experiment's model had to be altered to enable using generators of all listed LDS – the simulated time had to be decreased to limit the number of used samples due to generators' dimensionality constraints. For the Halton generator to work, the overall simulated time was chosen to 1,000 hours, so that it could provide enough randomized LDS samples in all evaluations. This also revealed a weak point of some of them and will be discussed in more detail in the next chapter. Results presented in figure 4.5 summarize evaluations completed with each LDS. The model is the LINAC4 availability model used previously to compare QMC and the original MC approaches.

Since those tests were designed to test differences between types of low-discrepancy sequences, it would be highly undesirable for testing purposes to switch to pseudo-random number generators after running out of dimensions provided by the LDS. On the other hand, some generators tested in this experiment have a very stringent upper boundary of the number of dimensions. For this reason, before the actual test runs,

the number of dimensions had to be adjusted accordingly. Another parameter, the one mentioned at the end of section 3.2 which is setting the multiplier in the formula for samples dimensions is set to 2. It is the lowest multiplier with which all tests used samples drawn exclusively from randomized LDS.

Results gathered in the figure include series of results for randomized Sobol sequence using powers of 2 as a number of samples aside from sequence involving powers of 10. Based on the literature and the method's documentation, the expectation is that those might show improvement in terms of performance. Generally speaking, Sobol sequences maintain their balance property when number of drawn samples is a 2^m, $m \in \mathbb{N}$. Halton sequences do not have this recommendation, however it has been observed that their convergence is slower ([69]).

This effort aimed at a comparison of the most popular choices of randomized LDS empirically. The evaluation proves Sobol sequences to be the most accurate in the most extensive run. However, when using a smaller number of evaluations, other methods perform better. It is also worth noting that the difference between Sobol samples drawn in powers of 2 or 10 are not particularly significant, but the former seems to be more structured and, as such, less prone to noise. Generally, we conclude that using powers of 2, as advised by the documentation and other literature, is beneficial for the simulations.

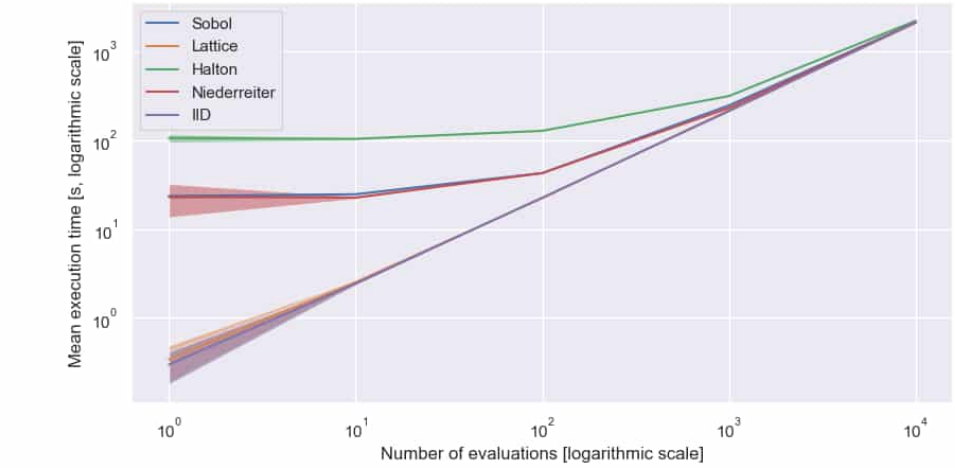


FIGURE 4.6: Average execution time (from three runs, in seconds) of simulations using Sobol, Halton, Lattice, Niederreiter and IID randomized sequences, logarithmic scale on both axes.

Extension to this test are execution time measurements. It is important to test LDS generators from this angle, as the overhead introduced by the methods might render the approach useless. The measurements were performed on a PC in the same environment. Measured results are presented in figure 4.6 and table A.3 in the Appendix A.

It is observed that generators of digital nets LDS (Sobol, Niederreiter, Halton) take time to initialize their operation. The overhead is more or less stable and thus becomes negligible when performing substantial numbers of evaluations.

4.2.2 Importance Splitting

Results presented in this section were completed primarily to illustrate the Importance Splitting approach, but they feature also RQMC as an improved benchmark comparison for the rare-event setting.

Importance function & weights

For ISp performance and robustness, the most important factor is the definition of the importance function which quantifies the criticality of the system state at any given moment. Generally, the function is describing a metric which establishes a distance from any system state to the critical failure state $d : X^N \longrightarrow \mathbb{R}$ according to the probability of transitioning between the two. As explained in chapter 3, the ISp thresholds are set in relation to that function. Observing that the current criticality value exceeds a threshold is signal to the simulation engine that the current evaluation is more likely to reach the critical region, and thus should be repeated to increase the number or interesting rare-events. In other words, the density of evaluations around system configurations that are closest to the critical region is increased as a result.

Important elements for measuring the system's criticality are weights assigned to each component. They are used to measure the contribution to the importance function of individual elements and, to maintain generic approach, are user-definable. Generally, their objective is to describe the likelihood and severity of complications induced by the component failure. The exploration of more advanced definitions of those in the reliability context was an important part of the methods development. Weights can be based on an arbitrarily selected number of parameters, such as whether a failure is detectable (detectable failures typically do not pose large critical issue), mean time to failure (e.g., to minimize the number of frequent events and focus on the ones seen less often), effective mean time to repair (usually it is the time when failure is undetected or not repaired that is the most critical). The precise definition and adjustments depend on the specific instance of a problem, and as such might be defined by the user.

Instead, the ISp implementation is tested using a simplified importance function metric, based on a single property of failure modes: their detectability. Figure 4.7 displays changes of the importance function in fifty evaluations (each color represents a different

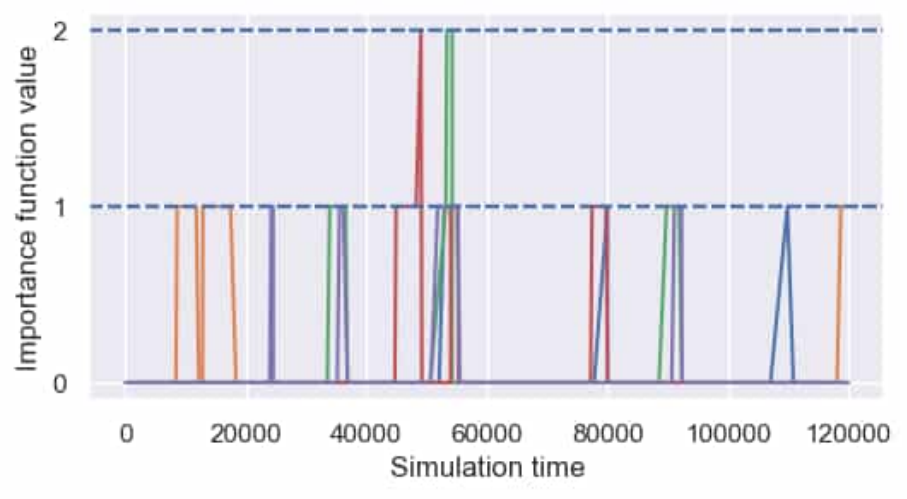


FIGURE 4.7: Changes of the importance function values in 5 evaluations of the HL-LHC Energy Extraction system model.

evaluation of the model). Those runs are not performed in ISp mode (no branching occurs when crossing the thresholds). Most of the blind failures appear in the system and are repaired shortly after (jumps between values 0 and 1 of the importance function). In two cases, however, another failure makes an appearance before repairing, and thus the function value goes to 2. A particularly positive finding about the importance function is that even with the simplest weights, the algorithm has a way to establish a criticality measure.

Using this importance function, all failures are of the same importance and the criticality metric on the system level changes according to the number of *active* undetectable failures in the system. The ISp weights (defined in chapter 3) assigned to individual failure modes have only one of the two values: 0 or 1, depending on whether a failure is detectable. The thresholds are thus consecutive numbers of failures observed in a single model evaluation.

To illustrate this approach, the Energy Extraction model (subject of a detailed study using CMC approach, described in Appendix C) provides a very straightforward example. Given the stakes involved in countering the effects of quenches, the EE systems are redundant to a high degree. Each instance consists of two twin switches, with only one necessary to break the circuit. There is also additional redundancy present within a single cassette. On the high level, this translates to a model with two compound cassettes (i.e., *cassettes*) inside which some undetectable single points of failure can be found, such as rectifier diodes. Whenever such a diode fails, the entire cassette cannot operate. For a critical failure of the entire system, both cassettes have to stop operating at the same time and an energy extraction request needs to arrive. In this situation, branching

simulations whenever such a rectifier diode fails, as the simulations are more likely to observe interesting events in time after such a failure. On the other hand, inside cassettes there are also components which have detectable failures, such as capacitors. Whenever they fail, they not only have a fail-safe mechanism extracting energy immediately, but also trigger a replacement of the entire cassette afterwards.

The applicability of this approach is limited and might need to be adjusted with more detailed function definitions in systems in which components fail routinely. In this one, every failure is assumed to have a similar likelihood of eventually reaching a critical configuration. Because of its simplicity, this approach is used in all other tests of the ISp method in this thesis, but any applicability to specific cases has to be examined on individual basis in each model: in simulations in which components fail fairly often, the number of branches might quickly grow, failing to deliver any potential improvements.

Test case: 2oo4 logic

The first test case is a model which is conceived to leverage the potential advantage of the method. It is composed of 1 compound component whose failure probability is the sought value, and it consists of 4 identical basic components. Their MTTF is set to 100 hours. For the system to operate, at least 2 of the basic components have to work. Upon third failure the system fails critically – the probability of such an occurrence is the estimated value. All failures in the system are blind and not repaired until the end of the simulation, i.e., when one component fails, it will not be repaired at any point. With such a definition of the problem, it is expected that ISp using simple weights based on the number of failures will branch evaluations at exactly 2 thresholds (when seeing 1 failure and then when seeing 2 failures - 3 failures are already the event of interest). The system lifetime is relatively short, set to just 1 hour. Given that the failures are described by the exponential distribution, we can use the formula for cumulative distribution function to obtain the probability of occurrence of the event of interest within time t:

$$p = 1 - e^{-\lambda t}$$

where λ is the scale parameter of the distribution (i.e., 100 hours, MTTF).

With this knowledge, we can establish the probability of k components failing using the analytical formula:

$$P(X = k) = \binom{n}{k} p^k (1-p)^{n-k}$$

where X is a random value for the number of failures, n is the total number of components and p is probability of a failure of a single component in time.

For the system to fail when j failures are tolerated, the sought value is the following sum:

$$P(X > j) = \sum_{k=j}^{n} \binom{n}{k} p^k (1-p)^{n-k}$$

After substituting the chosen values, it is seen that the theoretical probability of a system failure is approximately 3.91×10^{-6}, well within rare-event simulation range. In an average case, MC simulations would see only four failures in 10^6 simulations.

Threshold	0	1	2
Multiplier	40,000	50	150
Total trials	40,000	80,150	183,600
Missed threshold	38,397	78,926	183,600
Reached threshold	1,603	1,224	1,143
Ratio of reaching threshold	0.04	0.015	0.006

TABLE 4.1: Example – details of restarts in a single run with 40,000 initial paths, and multipliers 50 and 150 in second and third levels. Overall result: 3.6×10^{-6}.

Table 4.1 is an example of a result producing one point in figure 4.9. Its purpose is to illustrate how the splitting and branching mechanism works in an actual run. Since we are discussing scenario with 3 tolerated faults, the algorithm recognizes 3 thresholds. In each level, evaluations which observe less failures than a certain number are stopped (below threshold 0, simulations without any failure reach the end without branching, in level 1, evaluations with just one failure stop, and so on). Total number of trails for each threshold is decided using a formula: $r \times m$, where r is the number of instances hitting the previous threshold and m is the threshold's multiplier. In the last phase, for evaluations which were branched at the third threshold (i.e., have observed 3 failures), all evaluations are stopped at the end. The value of interest – the fourth failure – can occur in those, there is no need for further branching.

Results of a number of simulations are presented in figure 4.8. It has been decided to show the raw data instead of average or other approximation of the exact results, since such a representation highlights important information about the method and optimization of its usage. The plot includes a 10% relative error threshold set in relation to the theoretical value derived above, deemed to be an acceptable error level in most reliability simulations.

The results provide appropriate acceleration. First results reaching the acceptable relative error are two orders of magnitude faster than QMC and 4 orders of magnitude than CMC. The problem, however, remains that this result is heavily dependent on selecting appropriate parameters for spitting and branching. A number of different estimations generated in this approach for similar number of evaluations in caused by the different

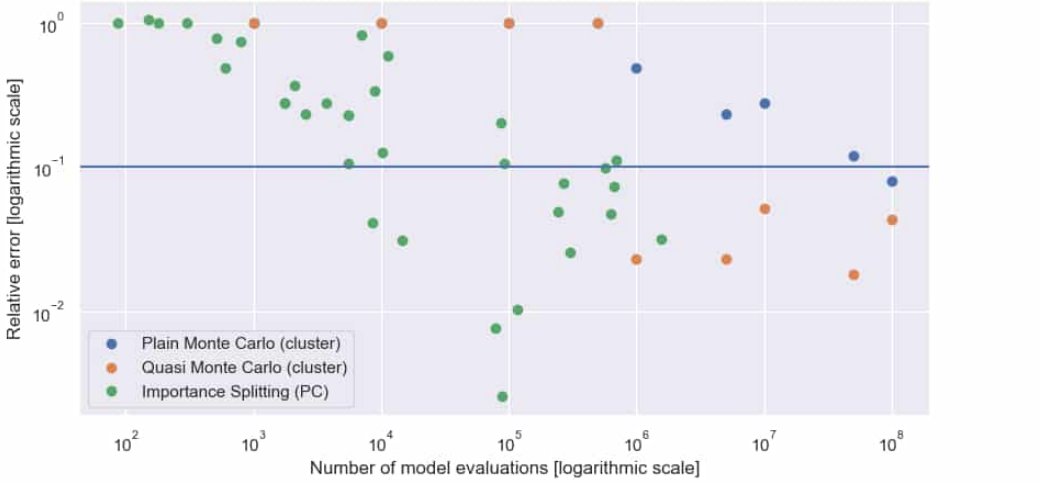


FIGURE 4.8: Relative error of CMC, RQMC and ISp estimations of the failure probability within the systems' lifetime, configuration 2oo4. Blue horizontal line indicates 10% RE threshold

number of instances passing thresholds and the number of restarted copies. With a less optimal selection of parameters (e.g., same number of restarts in each level, including the very first one), the ISp performs comparably to QMC (still two orders of magnitude ahead of CMC). What is clear from a more detailed analysis of results, is that ISp observes many more critical situations in its evaluations, however the estimation may experience poor accuracy due to errors in estimating probabilities of passing to next level (estimations of those, within a single level, are done using a standard CMC approach). Making sure that levels set by thresholds are accurately estimated (through observing statistically significant number of instances reaching successive thresholds) needs to be a priority when using this method.

On the other hand, QMC performs according to expectations. With 1,000,000 DES iterations observes just singular rare event instances – but due to well-distributed samples, it is close to the theoretical value.

Test case: 3oo4 logic

In the second case, we consider a very similar system to see the impact of introducing another threshold level in the ISp method. The system composition has not changed compared to the first case, but the critical failure will occur only when all 4 basic components fail. This leads to another splitting level, which is equivalent to a situation when 3 components are failed.

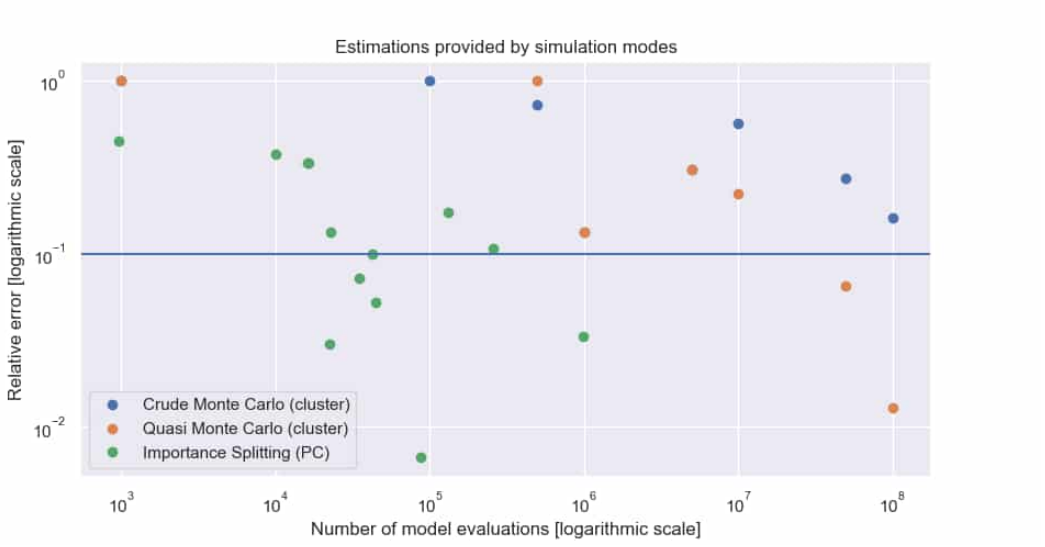


FIGURE 4.9: Relative error of MC, RQMC and ISp estimations of the failure probability within the systems' lifetime, configuration 3oo4

Another modification is a shortened MTTF of each component, which essentially causes them to fail more often. This was done to limit the rarity of the critical failure, as the comparison is done with CMC and QMC methods. The theoretical value calculated using the formula presented for the previous system is 1.16×10^{-6}.

Threshold	0	1	2	3
Multiplier	2,000	40	50	100
Total trials	2,000	9,480	23,250	52,000
Missed threshold	1,763	9,015	22,730	52,000
Reached threshold	237	465	520	459
Ratio of reaching threshold	0.119	0.049	0.022	0.009

TABLE 4.2: Example – details of restarts in a single run with 2,000 initial iterations, and multipliers 40, 50 and 100 in second, third and fourth levels. Overall result: 1.15×10^{-6}

Similarly to the other test case, table 4.2 shows details of a single run, which provides essential insights for adjusting the parameter values.

Probability estimations using increasing number of evaluation are presented in figure 4.9. The estimations obtained with ISp approach are significantly better, overtaking CMC by two to four orders of magnitude. On the other hand, again, the estimations vary a lot even when the number of evaluations is similar. This is due to the choice of specific parameters as the numbers of restarted copies in each level.

Chapter 5

Discussion

The experiments described in the previous chapter provided interesting empirical insights into the effectiveness of quasi-Monte Carlo and Importance Splitting methods, in both availability and reliability applications. This chapter presents a more broad discussion of the results in context of systems' reliability simulations.

The scope of the included considerations covers both factors of quantitative nature (such as accuracy or execution time) as well as qualitative ones (such as generality, ease of use or results interpretation). The work described in the previous chapters provided plenty of information about the methods. Completing the literature review was the first step into gaining insights about them. It gave a crucial theoretical overview of available options and their applications to various domains. The second step, described in chapter 3, focused on practical development of a maintainable implementation that works well within the adopted simulation framework, while being compatible with existing solutions. The experiments carried out and described in chapter 4 eventually show the produced edge over CMC simulations of rare-events, as well as provide recommendations for the use of the methods and identify their limitations.

5.1 Quasi-Monte Carlo experiment results

The comparison of the probability estimations shows that QMC methods perform well both in availability and reliability simulations, delivering increased accuracy to the expected extent. The convergence rate is empirically seen to be better than in CMC in both tested availability models: basic system and LINAC4. In rare-events setting of reliability simulations, the method does decreases the number of DES iterations by one or two orders of magnitude and, more importantly, improves the accuracy when minimal average number of iterations has been performed.

An important conclusion with reference to the studied reliability simulations is made about the usage of LDS regarding the limitation on the number of sample dimensions. The current implementation makes use of highly-dimensional samples, multiplying the number of components by the average number of *regenerations* and by an arbitrarily selected multiplier. In the example mentioned in this thesis, the Energy Extraction reliability study, such an approach makes the sample dimensions reach values $\geq 15,000$. The strict limitation in the used library is approximately 21,000 dimensions (only for Sobol sequences – for other LDS, the limit is even more strict), while – to our best knowledge – the most advanced generators can produce samples with around 65,000 dimensions. Those levels could be reached rather quickly in the discussed approach when the examined system contains many components, the simulated lifetime is long, and many events renew components' effective time to failure values. Still, with a minimal effort, the 20 year lifetime of the Energy Extraction system can be subdivided into smaller periods where the initial conditions are known, after an inspection, and then analytically generalize results to 20 years of operation. In case such a simplification is not possible, a precaution is taken in this implementation (switching to standard sampling using pseudo-random samples when there are no more QMC samples available) which enables simulations to finish and realize a hybrid QMC/MC approach. The potential advantage of QMC is, of course, diminished, but – provided that the part of the simulation without randomized LDS samples is not too large – is still present and potentially visible. This limitation is also the reason why Sobol sequences are the default choice for our methods, independently from performance experiments results. Among all other sequences and implementations considered, generators of other randomized LDS provide significantly stricter requirements in terms of the maximal number of dimensions.

The analysis of the execution time leads to positive conclusions too. It is expected that there will be a slight improvement in performance, as similarly to CMC, RQMC does not need to monitor any additional values (like the criticality metric for ISp), and generates all samples in a single call to a fast LDS generator library. However, the problem is that the algorithm cannot precisely estimate the number of samples needed for each component – and therefore has to generate them with a reasonable margin. The need for additional samples takes more time, and thus the QMC is adding a slight overhead over MC execution time. Already in the current implementation the overheads are limited and globally allow for substantial speed improvement. Additional work on optimizing the code and creating a more tailored number of samples is needed and likely will decrease the QMC overhead further. For example, estimation of the needed number of samples might be performed per component in a pilot job – possibly producing the need for fewer samples than when all components need the number that is calculated for an event that occurs most often.

Overall, results obtained in the experiments with RQMC method lead to recommending using this simulation mode in most cases in place of CMC. The method converges to the theoretical value more efficiently, is simple to use (from a users perspective – needs no additional inputs or parameters) and – although it requires additional initialization time – it allows to save time in terms of accuracy to execution time ratio. The problem with large number of sample dimensions used in reliability simulations might be relevant in some simulations and this might be a limiting factor. Further work on decreasing the number of sample dimensions, e.g. through better definition of the number of sample dimensions and more efficient use of the generated ones, might help alleviate this issue.

Nevertheless, using quasi-Monte Carlo methods for rare-event simulations is not sufficient to address the challenges posed by MC reliability simulations involving rare-events entirely. They were proven to decrease variance and give more accurate availability and reliability tests estimates. However, the reliability tests show that the advantage is visible only when performing a similar number of DES model evaluations as in CMC. The desired effect of the rare-events method is that the number of iterations would be decreased much further – which is addressed in the next method, Importance Splitting.

5.2 Importance Sampling experiment results

The key finding is that the method performed well in reliability simulation scenarios prepared for testing, delivering accurate estimations, compared to analytical calculations, already in runs smaller by several orders of magnitude in simplified examples. The observed efficiency gains are substantial: evaluations that otherwise take more than a couple of days to finish or require multi-core multi-node infrastructure can be completed in minutes on a local machine.

The *2oo4* and *3oo4* test cases described in chapter 4 display the capabilities of the method. The acceptable level of the relative error (RE) is defined at 10%, which is considered sufficient in real-world conditions. In both tests, this level is achieved with approximately just 10^4 evaluations, which is a workload easily executed on a local machine. CMC and RQMC methods do not even encounter critical errors until running 10^6 iterations. For the acceptable accuracy in terms of RE, MC needs in the order of 10^8 iterations (10^2 times more than the minimum to observe on average one event, which means that simulations are seeing a hundred occurrences in total on average). RQMC does better, providing relatively accurate estimations already above 10^6 iterations.

As indicated by large ranges of ISp estimations, the observed performance depends heavily on the selected parameters of splitting and branching. Its aim is to reward (and

multiply) evaluations in which risky situations occur, while keeping the overall number of evaluations reasonably limited. The threshold choice is a very sensitive parameter – events and states in the studied reliability evaluations are of discrete nature and often do not require many intermediate steps for the event of interest to happen. Hence, it prevents uniform control of all critical failure paths and branching at the exact same level of criticality. The total number of restarts in a level t is $R_t = h \times r$, where h is the number of hits of the threshold $t-1$ and r is the number of restarts from a single recorded instance. The performance shows to be be sensitive also to the parameter r, which should be chosen carefully too. A small number might prevent the runs to observe enough paths hitting the next threshold, while a large number can produce very high number of iterations to complete.

Important aspect is that the choice of the importance function value deeply impacts the method behaviour and results, changing the splitting points and the number of restarted instances in each level. For example, in the importance value plot of 50 runs (figure 4.7) of HL-LHC Energy Extraction, it can be seen that failures occur approximately at least once per evaluation. In that case – assuming simple weights and thresholds used in the other tests – the evaluations would be stopped and branched in every single one, which does not indicate any potential for improvements. One solution is to considered starting to branch simulations only after observing the second failure. Either way, it is very important for ISp to understand the importance function dynamics and appropriately decide on method parameters. A simplification of this process, mentioned as a potential future work, is a different approach to the definition of those parameters:

1. Adaptation of definable parameters. Users decide on the number of observed instances reaching threshold in each of the levels, rather than the number of evaluations and the number of copies of the same instance in the next level. The simulation engine would need to dynamically decide on the number of evaluations, depending on the results it has obtained so far.

2. Automatic parameters selection. The idea is to run a test ISp algorithm with generic parameters, not to directly observe final results but only approximate probabilities of reaching consecutive thresholds (i.e., making sure there is enough evaluations by adaptively increasing the number of evaluations). Data provides same results as in example tables presented for both 2oo4 and 3oo4 test cases in chapter 4. Using that information, the ISp algorithm could set number of restarts accordingly to obtain accurate results in actual runs.

In terms of a single model evaluation time, the method performs as it was expected. ISp evaluations take longer than MC ones, as the discrete event simulation loop is augmented with the part that keeps track of up-to-date importance function values. Its size depends on the complexity of that function (and, e.g., weights). The overhead accumulates quickly, as the calculations are performed in every iteration of the DES loop – in other words, whenever something changes in the simulation. Nevertheless, the overhead remains the same and becomes negligible when using many iterations. In addition, ISp evaluations are expected to give similarly accurate results using significantly less evaluations, and thus additional constant overhead of the observed size is not a particularly problematic aspect.

It is important to note that at this stage of the project, we have not applied any code or tailored optimizations to the methods. In the course of the development, we have observed several potential points of improvement, which can be implemented and tested as future work. The discussion above alone shows at least several potential improvements:

1. Using RQMC sampling within ISp approach.
2. ISp can calculate the importance function only in cases of a subset of the events, or at certain intervals.
3. Restarting split simulations in branching points by *reusing* events calculated in the base evaluation rather than *replaying* events in every created branch from the beginning.
4. Another pilot DES runs. Except for appropriate selection of ISp parameters, it is equally important to make sure that the weights correctly express the criticality of a specific failures. A considered generic approach assumes running a limited number of standard CMC simulations to gather insights on effective mean time to failure and similar aspects.

Especially the third point is likely to lead to shorter timelines simulated by reused, shared parts of paths and thus – the potential simulation completion speed increase is significant.

5.3 Additional considerations on rare-event simulations

In general, one could argue that increasing the parallelization is the ultimate solution to the problem of rare-event simulations. This applies to the studied case rather well,

especially since Monte Carlo simulations are examples of *embarrassingly parallel* problems. However, with the so called *Moore's law* [70] breaking down in recent years, we see that this conclusion might be considered overly optimistic. Transistor integration does no longer grow at the rate it used to, which makes placing trust in ever-expanding computing power not realistic (at least in the short-term perspective). Of course, this issue can be addressed by building even more powerful computing clusters, where limitations are dictated by practical and economical terms rather than state-of-the-art production developments. However, access to clusters is limited and – typically – problematic. Furthermore, the demand grows too due to increasing amount of computationally-intensive workloads coming from all domains of economy and science (e.g., such as ever growing Big Data field).

The problem of access to cluster computing is partially alleviated by the emergence of Cloud Computing, which has already dominated computational needs of the industry, while making also continuously more extensive advances into the field of scientific computations. Nevertheless, in these environments, end users are the ones tasked and charged with the utilized computing power. It is therefore of great interest to use advanced algorithms and design simulations in the possibly most efficient way. In the entire course of this study, which includes experimental testing of the methods, but mainly iterative runs of the EES reliability model, the usage of the computing cluster is estimated to be of approximately 94 CPU years. Pricing based on measurements of the actual use make it very beneficial to consider applying the most efficient solutions.

Another argument for highly-efficient simulation algorithms is the execution time factor. Perhaps in terms of reliability simulations for HL-LHC those requirements are less strict, however in many other real-world applications - and especially in financial engineering, the time constraints between obtaining input and producing results can be very challenging. In the case of reliability simulations, the ability to produce the results relatively quickly could provide a significant edge. The development of a model is very often an iterative process, that requires inputs from domain experts. Decreasing the time interval between finalizing the model and presenting results might significantly improve the workflow, making the iterations faster and more focused on the specific problems. Longer intervals, on the other hand, often lengthen the process not only by their own duration, but also through the organizational aspects, such as the need to re-introduce stakeholders to the model or more demanding long-term planning.

Even when the financial aspect plays no role in planning the simulations, there are also environmental factors such as energy waste that should be taken in consideration. In the modern world, awareness of risks posed by climate change and power consumption

has already led to increased pressure on green energy and efficient use of existing resources. According to the report produced by International Energy Agency [71], power consumption of data centers amounts to even 1% of the global electricity usage. Obviously, simulations are not the largest contributors to those, however it is clear that designing efficient algorithms that use less resources is of paramount importance . In the same report, experts state that even switching to renewable energy sources will not address all problems generated by this industry. Making sure that the usage is as effective as possible, should be also one of the priorities for those reasons.

Chapter 6

Conclusions

To conclude, this chapter presents a short summary of the entire thesis project and potential future work, which builds on the results and insights of the discussed work on rare-event MC methods.

6.1 Summary

This thesis focuses on the subject of Monte Carlo methods for rare-events simulations of systems reliability in the specific context of Machine Protection systems at CERN. The thesis addresses the three research questions defined at the beginning of the thesis project in section 1.4.

The completed literature review is a survey of the existing approaches and their state-of-the-art extensions, including methods such as: importance sampling, importance splitting, randomized quasi-Monte Carlo, surrogate models and other variance reduction techniques. This part of the thesis outlines available methods for Monte Carlo optimization for rare-events.

Upon theoretical evaluation of their relevance, through a review of possibilities and assessment of their applicability, the two most promising approaches for modelling safety critical systems reliability are chosen: Importance Splitting and quasi-Monte Carlo. Subsequently, they were implemented in the existing simulation framework and tested on availability and reliability models. Empirical testing was designed to capture essential aspects of models used in actual availability and reliability studies (basic and LINAC4 availability models and 3oo4 and 4oo4 reliability models). The discussion of their application and potential from the machine protection for accelerators perspective is given in chapter 5. Both theoretical and empirical evaluations in this thesis present

the methods relevance for the modelling of safety-critical systems reliability, as they always incorporate this specific perspective in the considerations.

A significant part of the experiments is devoted to the practical usage of the methods in the reliability and availability modelling. It includes aspects such as selection of optimal parameters for the methods – as it is shown, ISp and its estimations heavily depend on the parameters. To a lesser extent, QMC methods also have some additional considerations that have to be addressed before using them, with the number of sample dimensions and appropriate generator choice being the leading elements. Sobol sequences were shown to be the most flexible and efficient choice for the discussed application. All those factors are valuable insights that enable leveraging those methods in practical use for reliability and availability of machine protection systems for accelerators.

The code is incorporated into a rare-events Monte Carlo availability and reliability simulations framework developed at CERN (which is expected to become open source in the near future). The development process involved not only efficiency considerations, but also aspects such as high code quality and reusability, which is important for software maintainability in an environment of fast-changing development teams. Apart from contributions in the rare-events simulation field specifically, other extensions were implemented alongside this thesis project, including a post-processing tool for aggregating results created in distributed execution of the code on a High-Throughput Computing cluster and a Monte Carlo feature for investigating root cause analysis of failure events of interest (both described in Appendix D). The framework was used for the reliability analysis of the HL-LHC Energy Extraction system (Appendix C), using standard methods - crude Monte Carlo and distributed computing. This work was highly relevant for the validation of the system design for system experts, and it also served as an excellent introduction to the reliability modeling and Monte Carlo simulations.

A significant advantage of Monte Carlo simulations is that – aside from the intuitive statistical formulation – they are relatively straight-forward conceptually. In the context of most scientific or engineering simulations, simplicity and explainability are important factors, since the results usually have to be viewed, validated or accepted by other experts, colleagues or other relevant stakeholders. Variance reduction techniques naturally add to the complexity of the simulations, but also to their interpretability. Ripley, in his book [72], writes: *"Even when variance reduction is achieved it may be at the cost of a much more complicated analysis of the resulting experiment because of the correlations induced. Variance reduction obscures the essential simplicity of simulation, which may help explain its lack of use"*. While the simplicity of a standard Monte Carlo approach remains unparalleled, there are plenty of advantages for the use rare-event methods

in Monte Carlo, often overcoming the advantage of plain MC method by a significant margins.

However, there are a few obstacles as it has been seen in the course of this project. Rare-event methods, despite their long history, are not easy to apply. Many requirements defined for each approach severely limit their applicability and efficient operation depends on parameters that are specific to each problem. An incorrect usage can be a significant setback. Ripley wrote in the same book ([72]): *"Variance reduction swindles*[1] *quite frequently do not work, especially when more than one idea is tried simultaneously."* Since the time he wrote those words in 1987 there have been many developments in the field. More recent advances, like adaptive importance sampling, or additions to importance splitting give new and fresh perspective on this matter. The main concern remains the proper usage of the methods, and that aspect has been improving through automated and adaptive methods for optimized parameter identification.

6.2 Future work

In terms of the future work, there is a number of potential options for improvements which could be further investigated. First and foremost, combining the two methods prepared in the process of developing this thesis would be very interesting – and, as mentioned earlier, such a method has been already successfully used elsewhere [34]. The advantage deriving from using importance splitting with quasi-Monte Carlo is somewhat implied by the results obtained in so far: ISp accelerates the simulations by orders of magnitude, while QMC is making them more accurate. Of course, there are numerous other considerations that have to be addressed beforehand – like an efficient approach to the number of dimensions and more effective samples generation.

Building on top of the importance splitting implementation, one could imagine testing more recent developments of the method presented in literature. *RESTART* [37] and *RESTART with prolonged retrials* [41] could probably increase the efficiency even further. In case of typical reliability simulations, particularly RESTART could be an interesting addition, since truncating evaluations that return to the *as good as new state* is intuitively justified – there is no reason to run multiple copies of the same evaluation to the very end, if their criticality function indicates the same risk level as a *new system*.

Another interesting aspect of the splitting approach is the use of alternative weights for defining the thresholds. The foundation for such attempts was defined within this project and is described in section 3. Finding the most efficient way to express the criticality

[1] As the book author explains shortly before, the term *swindles* is a part of the "colorful language of statistics"

of a model evaluation might be a non-trivial task, which might be very model-specific. Another major challenge is to define completely automatic thresholds. Number of splits heavily impacts the accuracy of the method and is tough for users to approximate. Problem could be solved, e.g., through running pilot jobs – more details on this subject were presented in chapter 5.

Apart from the splitting approach, there are many alternative methods that could be tested in the context of rare-event simulations. Particularly interesting is the adaptive sampling using Kriging meta-models, which seems to have raised a lot of interest from the rare-event simulations community. The specific context of systems reliability analysis makes them more complicated to apply, but some work in this direction has already been done and published (e.g., [54]).

Finally, another optimization could be considered at the programming level. Predominantly, using compiled code rather than an interpreted one at least for the most critical parts of the simulations, is likely to provide additional speed-up. Further profiling and code optimizations are also a potential further development.

The project has been a great learning opportunity, which resulted in direct contributions to the existing rare-event MC framework. Optimizations which have been implemented and tested gave promising results and will be used in the future CERN projects related to safety-critical machine protection systems to produce more efficient results.

Appendix A

Additional Quasi-Monte Carlo results

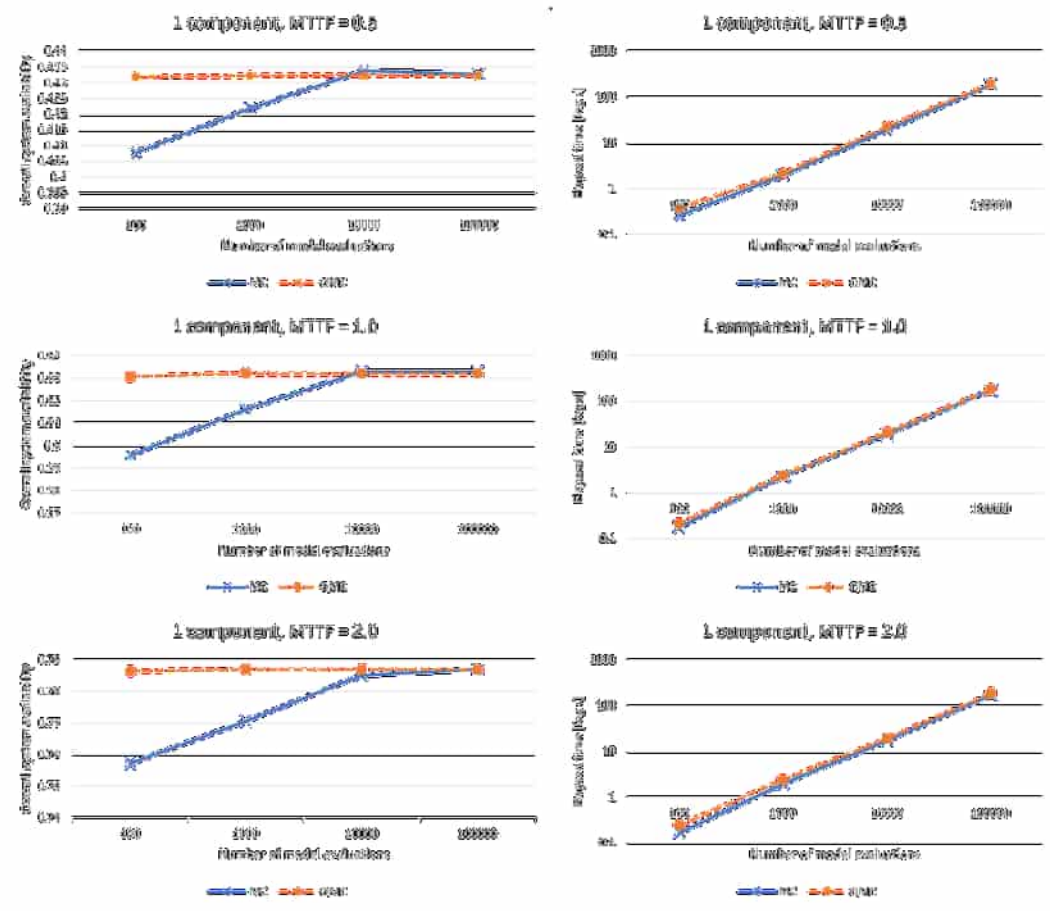


FIGURE A.1: Results for QMC simulations of a basic model using 1 component

	MC	MC- time	QMC	QMC - time	MC d	QMC d
10	0.855938	0.072442	0.873789	0.109445	0.031197	0.010992571
100	0.890458	0.591923	0.88714	0.717199	0.007875	0.004118848
300	0.887073	1.716995	0.883621	1.924926	0.004043	0.000136101
600	0.884245	3.541377	0.883448	3.952036	0.000843	5.98132E-05
1000	0.882742	6.180109	0.883294	8.600751	0.000859	0.000234459
10000	0.883663	99.60906	0.883339	109.7905	0.000183	0.00018339

TABLE A.1: Results obtained for Monte Carlo and quasi-Monte Carlo simulations - 25h

	1	10	100	1000	10000
Sobol	0.016214	0.006515	0.00016	0.000202	3.56E-06
Halton	0.004806	0.003241	0.001231	0.00032	2.76E-05
Lattice	0.015536	0.001918	0.000147	1.28E-05	2.41E-05
Niederreiter	0.002717	0.002076	0.001966	0.002286	0.00346
IID	0.007806	0.002194	0.000792	0.000366	0.000117

TABLE A.2: Relative error of simulations using Sobol, Halton, Lattice, Niederreiter and IID randomized sequences

	1	10	100	1000	10000
Sobol	22.73351	23.93981	42.57928	231.6627	2115.905
Halton	102.2772	100.3569	119.3728	307.6741	2181.123
Lattice	0.272834	2.69605	22.51829	207.8665	2078.715
Niederreiter	20.9904	22.02734	41.14323	231.5949	2066.584
IID	0.255846	2.462721	22.15722	208.843	2059.763

TABLE A.3: Execution time (seconds) of simulations using Sobol, Halton, Lattice, Niederreiter and IID randomized sequences

	1	10	100	1000	10000
Sobol A	22.73351	23.93981	42.57928	231.6627	2115.905
Halton A	102.2772	100.3569	119.3728	307.6741	2181.123
Lattice A	0.272834	2.69605	22.51829	207.8665	2078.715
Niederreiter A	20.9904	22.02734	41.14323	231.5949	2066.584
IID A	0.255846	2.462721	22.15722	208.843	2059.763
Sobol B	23.41	24.26	43.9	262.98	2135.32
Halton B	102.44	101.21	121.71	318.24	2237.97
Lattice B	0.32	2.28	22.35	220.55	2141.24
Niederreiter B	19.51	21.81	40.66	231.76	2127.5
IID B	0.27	2.26	21.27	222.68	2131.27
Sobol C	24.4733	25.86733	41.72871	245.2912	2128.462
Halton C	112.6353	110.0369	141.9478	317.6035	2226.677
Lattice C	0.424742	2.61144	21.80618	224.5374	2061.898
Niederreiter C	28.49912	24.12048	46.15362	228.621	2200.019
IID C	0.368776	2.566455	24.35152	206.6296	2198.324

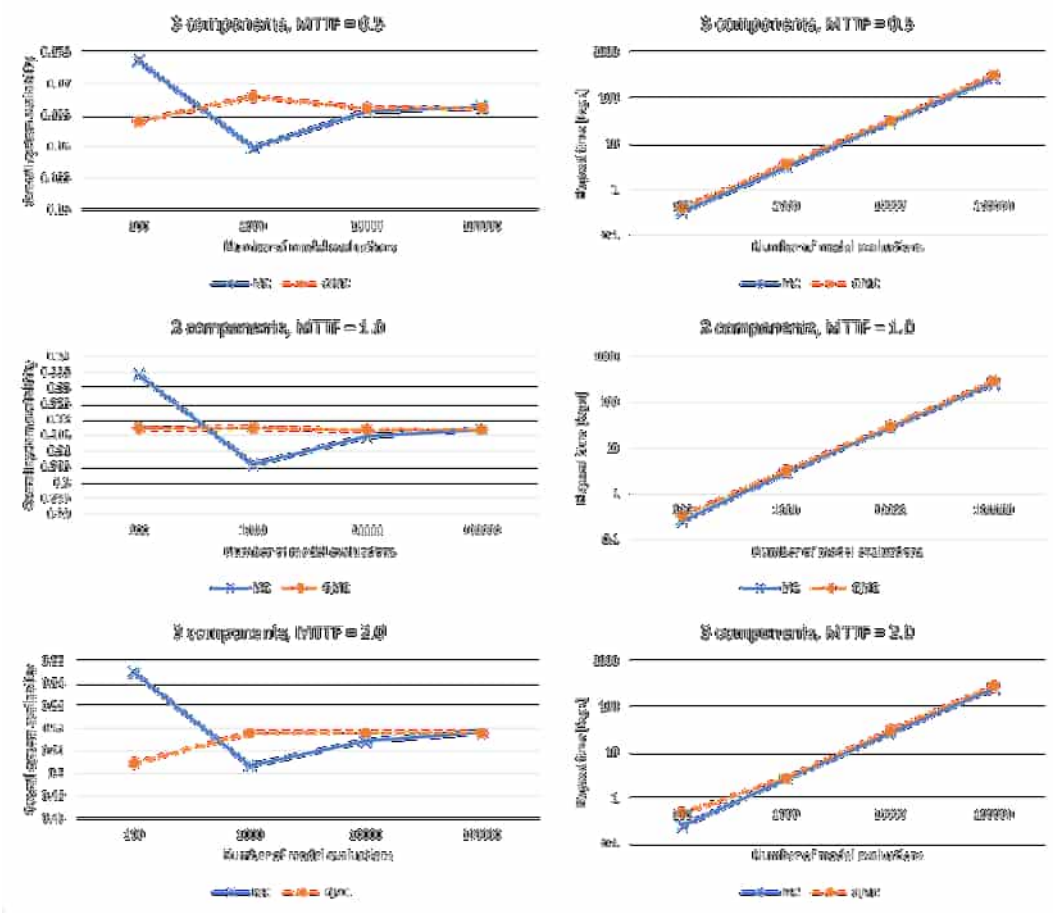


FIGURE A.2: Results for QMC simulations of a basic model using 3 components

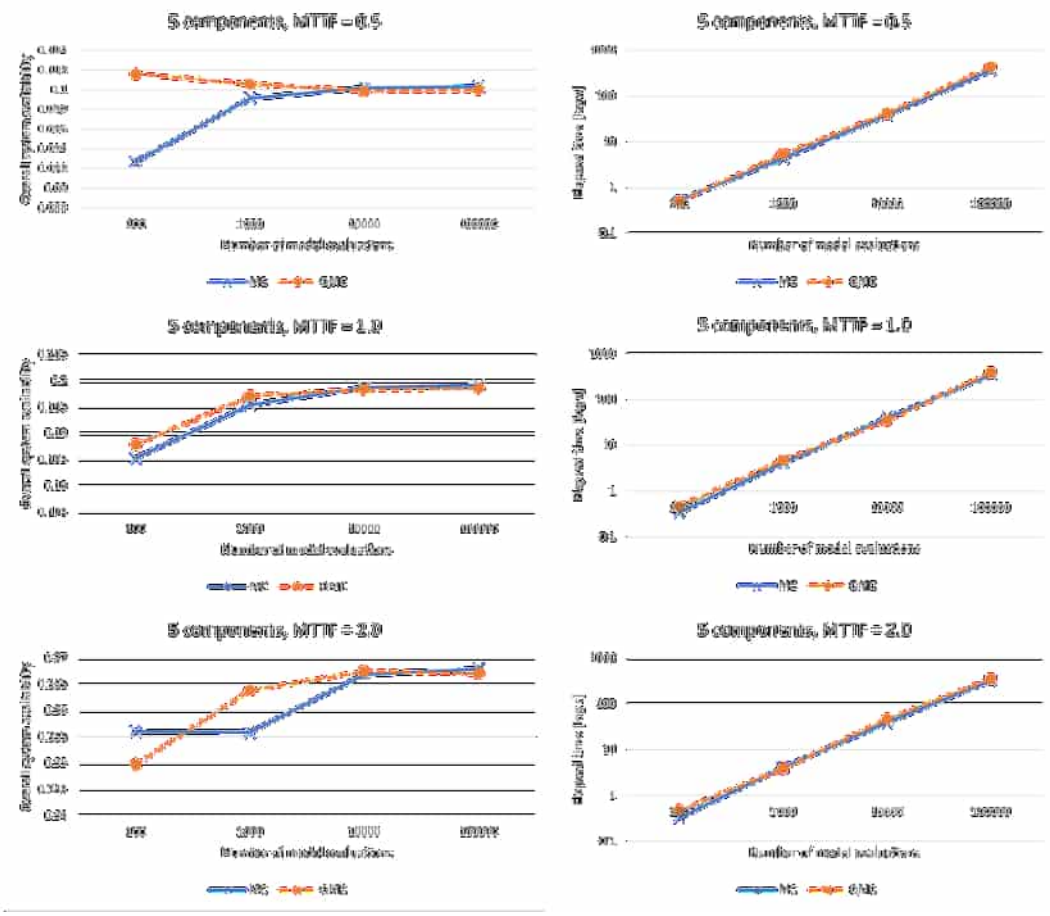


FIGURE A.3: Results for QMC simulations of a basic model using 5 components

Appendix B

Additional ISp results

MC Run	Mean
1.00E+05	0
5.00E+05	0
1.00E+06	2.00E-06
5.00E+06	3.00E-06
1.00E+07	5.00E-06
5.00E+07	4.38E-06
1.00E+08	4.22E-06

Table B.1: Raw data for 2oo4 test case using CMC

QMC Run	Mean
1.00E+05	0.00E+00
5.00E+05	0.00E+00
1.00E+06	4.00E-06
5.00E+06	4.00E-06
1.00E+07	3.71E-06
5.00E+07	3.84E-06
1.00E+08	4.08E-06

Table B.2: Raw data for 2oo4 test case using QMC

Splitting runs	Mean
87	0.00E+00
150	8.00E-06
180	0
300	0.00E+00
520	6.944E-06
600	2.00E-06
800	1.00E-06
1750	5.00E-06
1750	5.00E-06
2100	5.333E-06
2580	3.00E-06
3700	5.00E-06
5500	4.80E-06
5500	0.0000048
5520	3.50E-06
7000	7.13E-06
8500	3.75E-06
8950	0.0000026
10200	4.40E-06
11230	6.222E-06
14500	4.03E-06
78200	3.88E-06
87000	4.70E-06
89150	3.92E-06
92750	4.32E-06
117300	3.95E-06
242550	3.72E-06
268800	4.212E-06
303750	3.81E-06
572600	4.296E-06
635000	4.09E-06
671000	4.20E-06
696000	4.35E-06
1578800	4.033E-06

TABLE B.3: Raw data for 2oo4 test case using ISp

Splitting runs	Estimation
969	6.4E-07
16237	7.7E-07
16400	7.7E-07
22600	1.19E-06
22922	1.00E-06
35202	1.07E-06
42360	1.27E-06
44390	1.1E-06
63888	7.2E-07
86730	1.15E-06
130620	1.36E-06
256840	1.28E-06
994519	1.12E-06

TABLE B.4: Raw data for 3oo4 test case using ISp

MC runs	Estimation
1.00E+03	0
1.00E+05	0
5.00E+05	2.00E-06
1.00E+06	1.00E-06
5.00E+06	8E-07
1.00E+07	5E-07
5.00E+07	8.4E-07
1.00E+08	9.7E-07

TABLE B.5: Raw data for 3oo4 test case using CMC

QMC runs	Estimation
1.00E+03	0
5.00E+05	0
1.00E+06	1.00E-06
5.00E+06	8E-07
1.00E+07	9E-07
5.00E+07	1.08E-06
1.00E+08	1.17E-06

TABLE B.6: Raw data for 3oo4 test case using QMC

Appendix C

Energy Extraction Reliability Study Report

Alongside the thesis work, a study of the HL-LHC Energy Extraction systems was completed. It is was a separate effort that served as great introduction to the Monte Carlo reliability simulations. In the following pages, the final report of that study is presented.

The work comprised elements of all parts of the modelling process. It included gathering data about components, establishing information from past experience through mining databases, communicating with the system experts, working on consecutive iterations of the reliability model and finally preparing the technical report.

The events of interest are relatively low, therefore estimating the failure probabilities was a massive computational task. Computing cluster usage statistics indicate that throughout the entire duration of the project, the cluster used more than 94 CPU years. An average task would take from 6 to 12 hours and produce a single file with results. Usually, for a single set of parameters (i.e., for asingle point on the probability plots), there would be more than a 1,000 individual jobs.

Reliability Study Report: HL-LHC Vacuum Energy Extraction System

Date: 2021-10-13

Document prepared by: Andrea Apollonio, Milosz Blaszkiewicz, Thomas Cartier-Michaud

Abstract: The purpose of this study is to investigate the reliability of the HL-LHC Energy Extraction systems used, alongside other systems, to protect magnets from adverse effects of magnetic quenches. The High Luminosity update includes a substantial redesign of those systems, changing the breaker technology from mechanical high-speed DC breakers to “maintenance-free” vacuum interrupters. The new design is expected to improve the response time (more than tenfold) and increase reliability while decreasing maintenance needs at the same time. This study quantifies the risk of a failure which prevents correct protection of a magnet and offers insight into the most critical components of the design. Estimations presented in this report are based on the system's model developed in AvailSim4 software. The model accounts for failure rates declared by the components' manufacturers, planned maintenance strategies and repair procedures. Statistical analysis of the results attests that the system meets the reliability expectations set according to the standards formulated in the risk matrix for the LHC.

Introduction

The report presents the results of a reliability study of the HL-LHC vacuum energy extraction systems.

Glossary of Terms

CC	Counter Current generator (part of the EE system)
CERN	European Organisation for Nuclear Research
EE	Energy Extraction
FIT	Failure in Time, number of failures in 10^9 hours
FPGA	Field-Programmable Gate Array
HLCC	High-Level Control Chassis
HL	High-Luminosity
LHC	Large Hadron Collider
IDD	Inductive Dynamic Drive (part of the EE system)
MTTF	Mean Time to Failure
PTTF	Pulse Thyristor Train unit
QDS	Quench Detection System
QPS	Quench Protection System
UPS	Uninterruptible Power Supply
YETS	Year-End Technical Stop

Reliability Study Report: HL-LHC Vacuum Energy Extraction System

HL-LHC Energy Extraction systems

Purpose of the system

The Energy Extraction systems are elements of the quench protection chain for the LHC magnets. Each system is responsible for extracting the energy stored in the corresponding superconducting circuit upon receiving a signal from the Quench Detection System (QDS).

Energy extraction is an essential protection mechanism in the event of a magnet quench. After a magnet's resistive transition from superconducting to a normal-conducing state, energy rapidly transforms into heat which can damage part of the circuit. Quenches in magnets are unavoidable (either training or beam-induced quenches) and are therefore an accepted failure mode. The only solution is to detect these phenomena and instantaneously take the countermeasures (e.g., activate energy extraction for the circuits of interest in this note). The systems react in a time span of several milliseconds after receiving the triggering signal. First, the Energy Extraction system breaks the powering circuit and redirects the current into resistor banks while, second, it extinguishes an arc that may have formed in the short period after the contacts were separated.

The triggering signal is obtained from the Quench Detection System. QDS is considered an external system in the discussed reliability model and its failure modes are not included in this note.

The new design created for HL-LHC is conceived to offer an even more resilient, reliable, and almost maintenance-free solution that will cover the increased needs of the HL-LHC. The most relevant change is related to the circuit breaker technology: new systems will use vacuum interrupters instead of previously used mechanical DC breakers. Those interrupters offer maintenance-free operation with higher reaction speed than the alternatives. However, no previous experience makes them particularly interesting point for this study as they require additional scrutiny from the reliability perspective, which is addressed by performing a dedicated sensitivity analysis, as reported in the next sections.

Reliability Study Report: HL-LHC Vacuum Energy Extraction System

The planned lifetime of the system is 20 years. Within each year, the LHC will be active for approximately 250 days. Based on those values, the simulated lifetime is assumed to be 120,000 hours of operation.

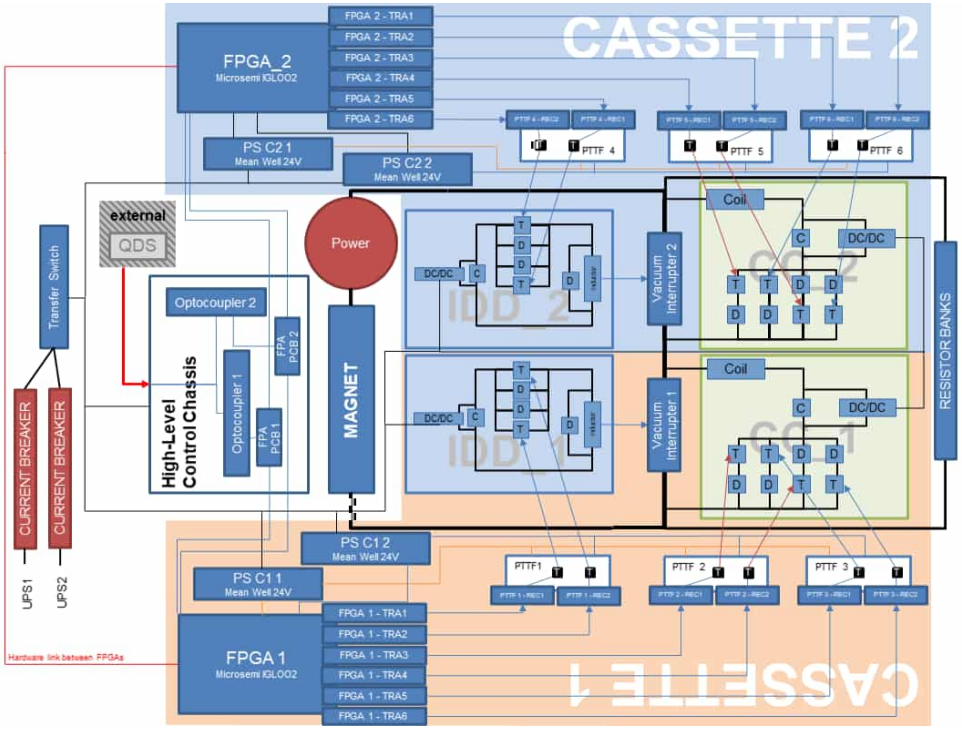


Figure 1 HL-LHC Energy Extraction system's functional block diagram

Design of the HL-LHC Energy Extraction systems

From the high-level perspective, a single EE system consists of two redundant switches connected in series to the powering circuit of the magnet. Each switch is contained in a physical cassette, bundled together with an FPGA and PTTF units. In Figure 1, the scopes of separate cassettes are indicated in orange and blue shapes behind components being part of them. Since those are physically integral units, any failure detected inside them will be repaired by replacing the entire unit.

A single switch consists of a part which controls the interrupter, Inductive Dynamic Drive (IDD), and of a part responsible for extinguishing the arc, Counter Current (CC) generator. Entry point to the system is the High-Level Control Chassis, which receives an active-high signals from QDS. When a drop in the signal is received, it is transferred by each of the two optocouplers to both cassettes. Since the signal is active-high, the logical false state (default mode, no need for extraction) is represented by the high state of the signal. A signal drop stands for logical true, meaning an extraction request. Therefore, all disruptions or failures involving a signal rupture open the switches. Moreover, optocouplers are connected in series, meaning that signal is doubled from the very entry point to the system.

To open the switch, the FPGA sends a signal to Pulse Train Thyristor Firing (PTTF) units through optic fiber connections. Each PTTF receives the signal from two independent channels (with only one of them needed) and forwards it to the corresponding thyristor inside the IDD or CC. After the triggering signal reaches the IDD or CC, the interrupter opens and the counter-current extinguishes the arc.

Power for the system is supplied by two UPS devices. Each has a separate current breaker, which is then connected to a single transfer switch for the entire system. When one UPS stops providing energy, the other one takes over within milliseconds — in the meantime, the energy stored in several capacitors in PTTF provide sufficient energy for 2-3 seconds to extract energy also in case of a power outage.

The transfer switch distributes energy to the power sources of FPGA units, the power sources of PTTFs, as well as to the DC/DC converters charging the capacitors in IDD and CC. In principle, when there is no power, interlocking design is in place to immediately open the switches, so there is no risk of critical failure caused by a sudden loss of power.

Redundant elements in the design

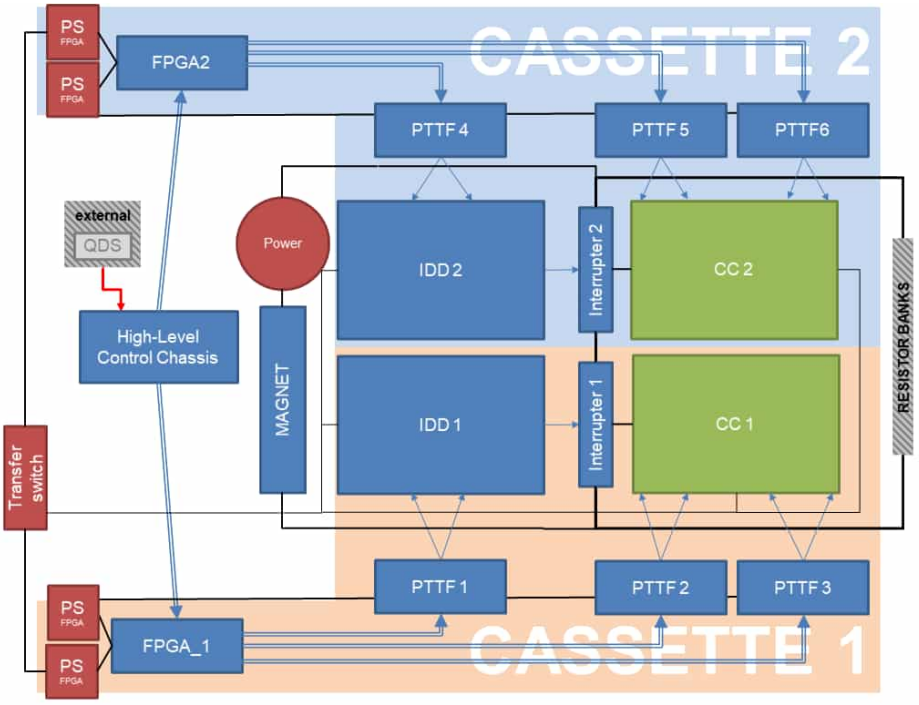


Figure 2 Redundancy overview of the system

Given the stringent reliability requirements, the EE systems are highly redundant. As seen in Figure 2, each instance consists of two twin switches, with only one necessary to break the circuit. However, the cassette needs to operate fully – i.e., extraction taking place with IDD of one cassette and CC of the other will result in a critical failure.

The entry point to the system, the High-Level Control Chassis (HLCC) is responsible for receiving, multiplying, and forwarding signals from QDS to the FPGA units in separate cassettes. The redundant

structure of the input to the system is explained as follows. An active-high signal provided from the QDS goes to two printed circuit boards connected in series. Each has an optocoupler, which transfers the signal to both FPGA units. The HLCC has also other, non-redundant elements (like buttons, etc.), however they are not essential for transferring the signal and, for this reason, are not considered in this model.

Redundancy is implemented also on the level of individual electronic components. In IDD and CC subsystems, thyristors and diodes are doubled and connected in parallel so that a failure of a single element does not incapacitate the entire cassette (although with exceptions, as not all elements have such a back-up component).

FPGA units are connected by a hardware link. FPGAs can use it to exchange information about their status. In case of a failure of one FPGA, the other FPGA can immediately trigger an extraction in its corresponding switch immediately, minimizing the risk of a critical failure later.

Communication links are also doubled between FPGA and each type of PTTF units (two links from FPGA to a PTTF responsible for triggering thyristors in one direction, two in other direction, another two to the one responsible for IDD). The logical communication channel starting at the FPGA transmitters consists of said transmitters, fibers, receivers, transformers (part of PTTF) and thyristors. All the components in the channel must be operational for the channel to work, but each channel is redundant (i.e., there are two thyristors - with independent communication channels - performing the same function).

Monitoring

Given the redundancy of various components, it is possible that some of them would fail and not cause any problems to the overall system. The simulation software distinguishes between “detectable” and “blind” failures. The former ones are visible as they happen and, as such, system experts can initiate actions to repair them immediately - ensuring that the system retains maximal redundancy. Some loss of redundancy could be used to trigger energy extraction and thus stop of the machine, depending on the final monitoring strategy. The latter ones, “blind”, are not visible, and the risk is that impacted components will remain incapacitated for an undefined period until their action is needed and cannot be fulfilled. There is a third monitoring strategy, “blind with passive monitoring”, which is assumed to be detectable, and thus repairable, after each energy extraction. Such detection is facilitated through analysis software of the data produced during the extraction. The data produced would be stored in the Post Mortem database . In the discussed reliability model, this type of monitoring would be applicable on the level of entire switches, as each interrupter has a dedicated sensor indicating whether it opened or not. Lack of opening may be a result of a failures anywhere in between the FPGA and the interrupter. Recognizing such failure before the other cassette fails critically significantly improves reliability, as the entire failed cassette, including all types of failures, can be replaced, and subsequently investigated.

Repairs

A large part of the Energy Extraction system is physically contained in two cassettes. Such a cassette has an IDD, CC, FPGA and three PTTF units inside, together with a vacuum interrupter. Since each cassette forms an integral unit that can be mounted and unmounted together, the repairs will be usually carried out by replacing it with a spare one instead of repairing it on the spot. The cause of the failure will be subsequently investigated in a laboratory.

Reliability Study Report: HL-LHC Vacuum Energy Extraction System

This strategy of repairs means that all blind failures in a cassette will be parasitically removed once it is replaced which increase the reliability, but which is mainly used to decrease the reparation time inside the tunnel, thus maximizing the availability of the LHC.

Alternative implementations

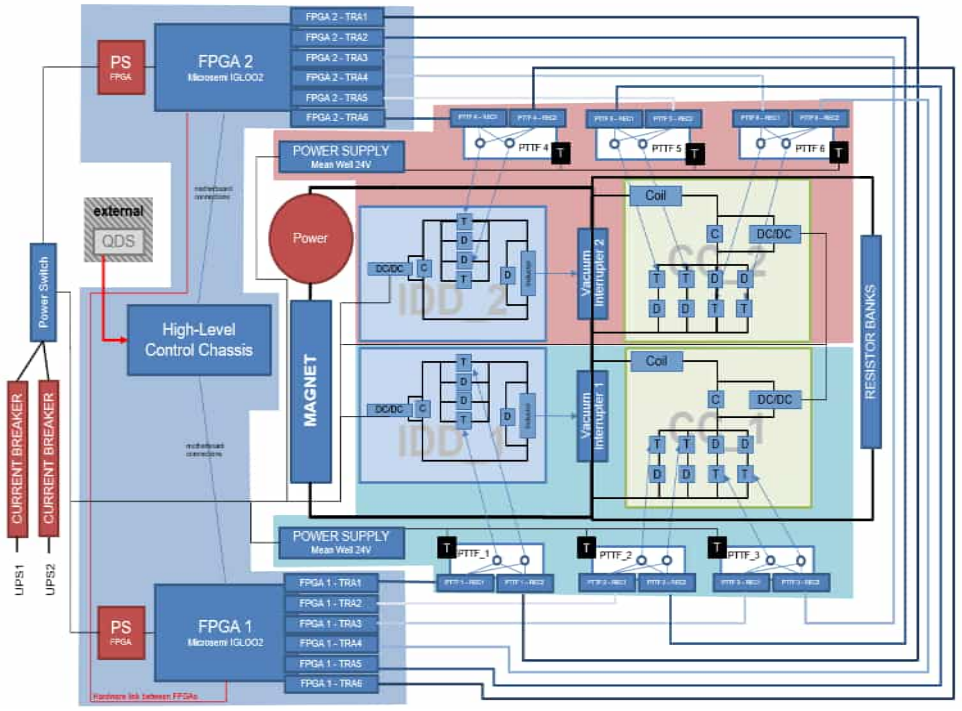


Figure 3 Functional block diagram of an alternative system design

All the simulations described in this document relate to the system in its current, base variant, as described in section 3.2. Further modifications under evaluation, which may be introduced in the final version of the system, and which will impact reliability in distinct ways. Depending on which and how they are going to be implemented in the system, the results will either improve (e.g., increased redundancy) or deteriorate (e.g., through a separation of some components that earlier would trigger replacement of the entire cassette).

Figure 3 shows one such alternative system designs. The most prominent change is the detachment of FPGA units from their corresponding cassettes and attaching them to High Level Control Chassis instead. Such a solution would allow both FPGAs to control both switches at the same time, increasing the redundancy further. However, at the same time, other reliability factors must be considered: FPGAs are one of the few components monitored – their failures predominantly decrease the risk of a critical failure occurrence, since they are immediately known and trigger equipment replacements. In addition, the optic cables leading from FPGAs to PTTFs will inevitably become much longer and thus more prone to failures.

Reliability Study Report: HL-LHC Vacuum Energy Extraction System

Reliability target

LHC reliability context

The target reliability for the examined Energy Extraction systems - as for all LHC subsystems - is derived from the general reliability risk matrix of the LHC presented in Figure 4. The matrix defines acceptable failure frequency intervals in relation to the downtime produced by such a failure. Since the risk matrix sets those values in reference to the entire LHC, the derived target must be adjusted accordingly, so that the multiple instances of the EE system installed in the LHC are all considered. Their combined risk should comply with the overall acceptable failure rate. The EE systems for HL-LHC will exist in two versions: 2kA and 600A. The 2kA systems will protect the correctors of Q1 (two MCBXFB), of Q2 (two MCBXFB) and of Q3 (two MCBXFA). The 600A systems will protect the correctors of D2 (four MCBRD) and one of the correctors of SF HO (one MQSFX). Both 600A and 2kA EE system are based on similar hardware, except for the vacuum interrupter and the resistor banks used to dissipate the extracted energy

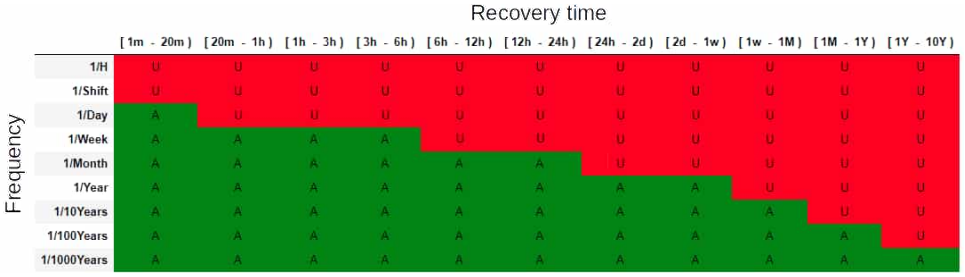


Figure 4 LHC risk matrix

Expected minimal reliability threshold

The systems operate continuously, as they are expected to always provide protection for their corresponding magnets. However, failures occurring within the system usually have no immediate impact on magnets or the accelerator. In this study, the focus is entirely on critical failures which are defined as lack of protection upon demand. For instance, if a combination of single components' failures prevents the nominal energy extraction from happening, then the system fails critically, and the magnet is at risk. Degraded state of the system is not considered a failure in this view.

To derive the acceptable failure frequency, first the recovery time needs to be assessed. Should a magnet quench without an operational Energy Extraction system, extensive damage to the magnet becomes likely. In the worst case, the magnet can be irreversibly damaged and require replacement. Such an operation will require a stop of accelerator operation for several months. This recovery time falls into 'one month to one year' [1M-1Y] interval and defines the acceptable failure frequency as 1 in 100 years.

An additional margin is taken by defining the target as follows: after 100 years of operation, the probability that any system failed should be below 10%, i.e., the expected overall reliability is 90% after one hundred years. Since the total lifetime is estimated to be of 20 years, the expected minimal reliability value must be adjusted accordingly. Target R_M in 20 years is defined as follows:

$$R_M = r^{\frac{l}{b}} = 0.9^{\frac{20}{100}} \approx 0.97914$$

where r is expected reliability, l the systems' lifetime and b the baseline lifetime. Another consideration is the number of systems. The failure rate calculated in this study refers to an individual instance of the

Reliability Study Report: HL-LHC Vacuum Energy Extraction System

system, while the target is derived based on a magnet recovery time. Hence, the reliability threshold for an individual system must comply with the overall target when combined for all protection systems for HL-LHC magnets altogether. Total number is assumed to be 76, as explained by the numbers in Table 1.

Magnets	Protection	Number
Inner Triplet Quadrupole	CLIQ + QH	6×4
2kA orbit correctors	EE	6×4
600 A and 200A high order correctors	EE	5×4
D1, D2	QH	2×4
TOTAL		$\mathbf{19 \times 4 = 76}$

Table 1 Number of protection systems considered in this study to derive the target

Overall target R_S for a single system is defined as

$$R_S = {R_M}^{\frac{1}{n}} \approx 0.97914^{\frac{1}{76}} \approx 0.999723$$

where n is the number of considered systems. Conversely, this value means that the maximal acceptable unreliability is 2.77×10^{-4}.

Individual component reliability estimations

[This is a redacted version of the report. For sections detailing individual component reliability estimations, consult original version available in CERN EDMS 2684812 v1]

Boundary condition of operations

Table 2 aggregates additional parameters which had to be assumed, including the duration of the system's lifetime, or assumptions for sensitivity analysis parameters (aspects that are uncertain or might change over time).

Parameter	**Value**
Duration of the simulation [hours]	$20 \times 250 \times 24 = 120{,}000$
Periodic inspections	1 year or 3 years interval
Mean Time to External Order (MTTEO) [hours]	320, 640, 1200, 2232, 6000

Table 2 Simulation parameter - assumptions

As detailed in the next section, reliability estimations strongly depend on the frequency of extractions. At the same time, the rate of those discharges cannot be fixed to a single value as it changes between periods (hardware commissioning vs operation) and specific magnets — and currently there is only little knowledge on the quenching frequency of the magnets that will be used in HL-LHC. Records stored in Post Mortem database contain information on the Energy Extraction openings in the past, for other circuits than the HL-LHC magnet. This data has been studied in order to give an estimate of what might be expected in the future. More specifically, for the purposes of this study, we examined period between January 2015 and December 2018. Details of the approach, including description of the

methods used to fetch the data of interest, and an overview of the obtained insights are available in Appendix A (p. 29).

As the maintenance strategy, it can be assumed that each system will be opened at least once a year for testing purposes (every 6,000 hours in the simulation). The minimal time between energy extractions is more complex to foresee. Based on information provided by the experts, we can expect 4-10 openings per EE circuit in hardware commissioning campaigns, and between 1 to 6 openings per year throughout the normal operation. Assuming a 4-month long commissioning campaign and evenly distributed 10 extractions per system, we obtain 288 hours as the minimal interval and almost 6,000 hours as the maximal (once per year).

Periodic inspections

The simulations feature periodic inspections which assumes checks performed on the systems' components in regular intervals. The model captures them as individual events, occurring at user-defined times throughout entire simulation lifetime. Failures are assumed to be selectively removed following an inspection.

In the Energy Extraction system, there is defined one type of periodic inspection, whose scope is the entire list of failure modes, detectable and undetectable alike. It effectively means that blind failures, undetectable otherwise, are recognized and repaired – preventing losses of redundancy when those failures accumulate over time.

In the considered simulation scenarios, inspection intervals are set to 1 year and 3 years. The former corresponds to Year-End Technical Stops (YETS) and the latter to the duration of a LHC run.

Remaining model aspects

In addition to the assumptions described above, there are several elements that are not considered in the model and thus ought to be treated separately.

The basic variant of the model does not include the spring failure mode. The alternative version includes spring, where the mean time to fail of the interrupter is a very pessimistic estimation made from tests of the unit performed at the institute participating in the design process of the system.

Coil and inductor have no failure modes assigned. Transfer switch and UPS are not failing in the models, as lack of power is considered to cause fail-safe behavior.

Resistors and Quench Detection System is not included in the current estimations, they can be added analytically, based on estimates derived from past experience. Optical fibers are also not assigned any failure modes, as according to the experts, their failures should not be examined quantitatively but only qualitatively.

Reliability estimations

The model of the system described above was tested in a massive simulation campaign and provided insights described in this section. Two versions of the system have been studied, the "default" one and an alternative one explicitly accounting for the "spring failure mode". The two only differ by the addition of a failure mode on the spring present in the vacuum switch: in the first case no failure leading to a miss extraction is assume while in the second the possibility that the spring blocks the interrupter is modelled.

Reliability Study Report: HL-LHC Vacuum Energy Extraction System

Base scenario with default parameters

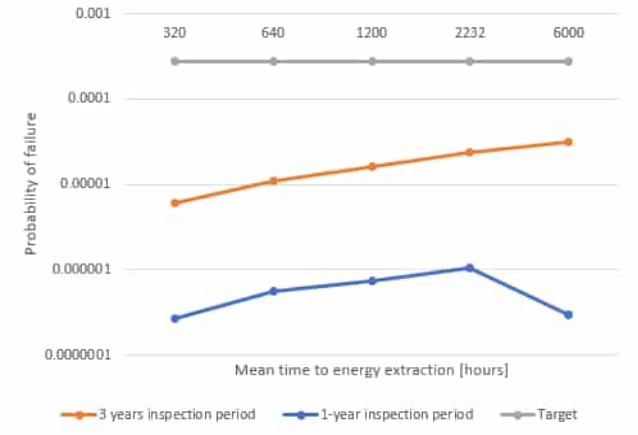


Figure 5 Probability of a critical failure occurring in 20 years of operation

This reliability model prepared with the default design, failure rate values in section 5.1 and monitoring assumptions for the current version of the Energy Extraction systems. The time to fail for the vacuum switch is assumed to be 428 million hours — without explicit mechanical spring failures considered in this approach. To present the reliability in different circumstances that the systems might encounter, the results account for changing time between extractions: from 320 hours to 6,000 hours mean time to external order. Another examined angle is the inspection frequency. An inspection in the model removes all failures (blind and detectable alike). In the sensitivity analysis performed, the frequency of the inspection has been set to 1 and 3 years as it would be possible to inspect systems at the Year-End Technical Stop or at every Long Shutdown.

The system reliability meets the target with a margin of one order of magnitude even in the less frequent inspection strategy (every three years), and by more than two orders on magnitude at all points of more favorable one (inspections conducted annually).

General results are presented in Figure 5. The reliability in the case of inspections occurring every year displays an interesting dynamic. First, the reliability decreases with increasing interval between extractions. This might be counter intuitive as it means that with more extractions performed i.e., there are more chances for a critical failure to occur, the probability of a failure still decreases. This is caused by the redundant design of the system and the monitoring strategy: each extraction essentially represents an ‘inspection’ of the system. If the other cassette functions correctly, critical failures of a single cassette can be identified and repaired without any direct danger of leaving a magnet unprotected. When the extractions are performed less often, such test are performed less often thus blind failures can accumulate for a longer period of time, possibly up to the point of losing the redundancies. However, eventually this dynamic ceases to guide the overall reliability; when energy extractions occur on average every 6000 hours (once per year), the probability of a critical failure decreases again. This can be attributed to the fact that at this value, there is simply not enough extractions in the system to critically fail and if no extraction occurred in one year, still the system is inspected at the end of the year. As for the periodic inspections happening every three years, only the first dynamic is visible.

Reliability Study Report: HL-LHC Vacuum Energy Extraction System

As a general conclusion, when there are many discharges, the system is monitored very closely, and failures can be removed before other failures incapacitate the redundant design. As system discharges become sparse, the system is more likely to fail critically on demand. Failures in various components may linger and eventually a combination of fatal failures renders the entire system impaired. A similar behavior was observed when studying the 11T protection reliability and the IT protection reliability.

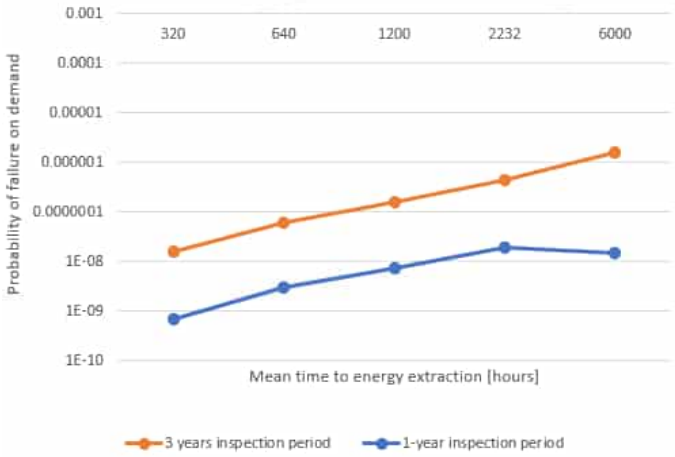


Figure 6 Probability of failure on demand

The model accounts for fail-safe mechanisms included in the system's design: in the example above, the first cassette would proceed to interrupt the circuit and dissipate arc due to active-high signal and capacitors providing appropriate amount of power. Same is the case when one FPGA unit or its power

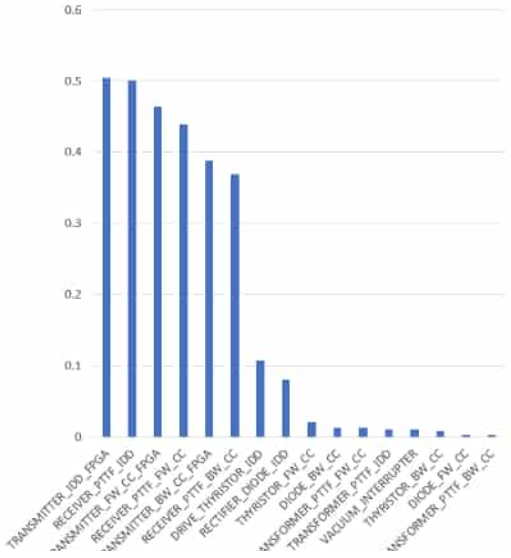


Figure 7 Components' contributions to the critical failures (fraction of critical failures which involve a given component) in all runs without spring failure mode, with mean time to external order between 640 and 2232 hours

supply fails – information about failure is propagated through hardware link between the two FPGAs. The functioning one is expected to trigger the extraction to avoid risk of protection with only one cassette.

Another estimated metric is the probability of failure on demand, displayed inFigure 6. For tested extraction intervals, the probability value is included in the range $[10^{-10}, 10^{-5}]$. This value is a ratio of unsuccessful extractions to the total number of system discharges. This probability of failure metric increases with extending the interval between extractions.

Root cause analysis of critical failures offers additional insights into the results. Figure 7 contains a breakdown of factors contributing to critical failures in all simulation runs with extraction interval between 640 and 6,000 hours and annual inspections. Overall, there were 23 critical failures in almost 600 million simulated years (overall $7 * 10^{-8}$ probability of a single failure). All of them were caused by errors inside cassettes. In almost all these cases, the extraction was triggered by an external order. Only in one case the extraction was triggered by a power supply cassette failure (in one cassette the rectifier diode failed, in the other — a communication channel to the IDD thyristor).

System with spring failures

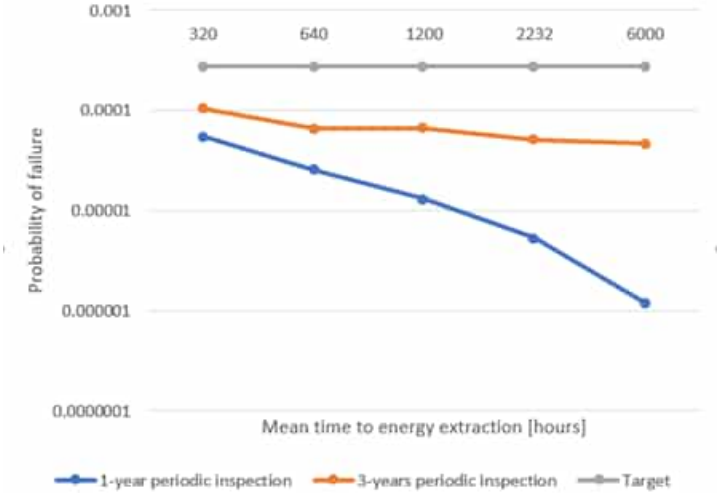


Figure 8 Probability of a critical failure in 20 years in a scenario with critical spring failures

The same simulations reported in the previous paragraph were performed for a system where the MTTF of the vacuum interrupter is set to the pessimistic assumption based on 20,000 cycles performed in a reliability test without seeing one failure and taking a 90% confidence level to derive the reliability figures of this component. For this model, the following assumption is made: the vacuum interrupter would always fail in a critical way, i.e., blocking the switch from opening because of the spring.

As explained earlier, the estimation of mean time to failure value is based on performed tests which are expressed in terms of successfully completed cycles. Therefore, a conversion to time-dependent units (such as MTTF) should also consider the number of cycles. In this case, the MTTF of a vacuum interrupter depends on the mean time to external order sent to the system and is simulated as such.

Results still meet the target (Figure 8), although in some environments they are in the same order of magnitude as the reliability threshold. The reliability curves for this scenario express probability of a

single critical failure occurring in 20 years of operation. Periodic inspections taking place more often help with the reliability, but the decreased failure rate is visible in comparison to the scenario in which more optimistic assumptions are made about the vacuum interrupter.

The probability of a critical failure occurring in 20 years of operation is decreasing with the number of external orders received from QDS. This translates into less failures when there is less extractions (or, equivalently, less chances to fail critically). The probability decreases at somewhat faster rate when inspections take place annually.

Probability of failure on demand is in the range $\in (10^{-8}, 10^{-5})$, as shown in Figure 9.

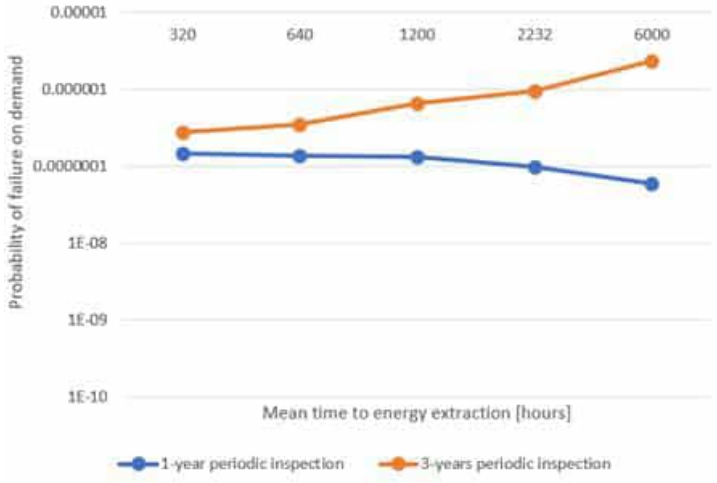


Figure 9 Probability of a critical failure in 20 years in a scenario with critical spring failures

Conclusions

Simulations of the EE system reliability show that the current design meets the identified reliability target (the probability to have at least one failure out of one EE system in 20 years is below 3.13×10^{-5} which is below the target 2.77×10^{-4}). The adopted failure rates are pessimistic (manufacturer, etc.). The design implements redundancy of the entire system parts, starting at the input point, which ensures high reliability. In addition, active monitoring of some components and a fail-safe logic in place for many simpler problems further enhance system reliability.

One of the observations made in the simulations is that monitoring of the optical connections (or results of signals passed through them, e.g., of IDD statuses) could increase the reliability value estimated through simulations. This is a relatively straight-forward conclusion, as optical communication components are the ones assigned the lowest mean time to failure values.

However, those low values might also serve as an important reminder to consider system elements that are not described quantitatively. Spring has not been assigned a failure mode in the baseline scenario, as it is considered to be fail-safe. Damage of the spring should lead to the energy extraction (however, simulations for scenario where spring failures prevent successful extraction have also been studied in this report). The transfer switch and UPS are also excluded from the model as they too exhibit fail-safe behaviour. Failures from the Quench Detection Systems are not modelled in this study (however, their contribution can be added analytically). Finally, coil and inductors in IDD and CC were also omitted

based on expert's feedback. Optical fiber connections are not represented by individual failure modes, since according to the opinion of the experts, they can be only analyzed qualitatively. On the other hand, experts designing the system expect possible problems with optical connections. Therefore, increased failure rates for end points might fittingly illustrate important point about the system.

The redundant construction of the system makes critical failures of a single cassette relatively harmless - provided that the other cassette is functional. As a matter of fact, observation of such a failure during operation (i.e., upon the trigger of an energy extraction) increases the overall system reliability, since it triggers the replacement of all components that are in the cassette, potentially preventing hidden blind failures.

This mechanism makes system tests an important part of the maintenance strategy. Making sure that the system extracts energy from time to time can decrease the likelihood of failures accumulating to a critical set of failures.

Reliability Study Report: HL-LHC Vacuum Energy Extraction System

Report Appendices

Appendix A. Exploratory Data Analysis of the past Energy Extraction operations

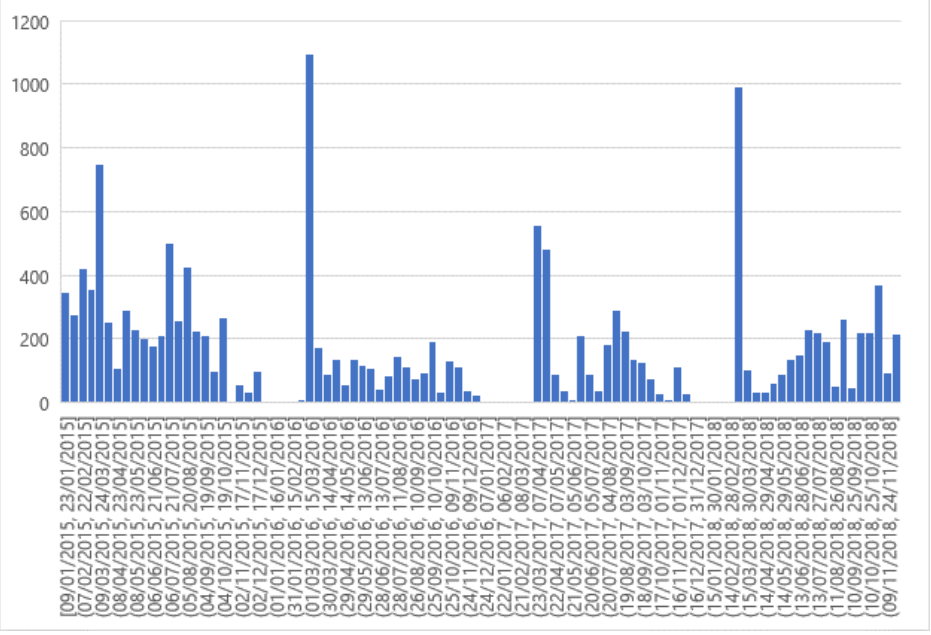
From the technical point of view, access to the Post Mortem database is possible through Sigmon API, an in-house project which is a part of LHC Signal Monitoring framework. API calls are possible directly from notebooks running on SWAN , and some examples of EE systems data exploration is available in repositories .The steps for identifying events of interest and related data can be summarized as follows:

1. Define the period of interest and split into smaller parts of duration around 1 day – since the database has not been upgraded yet, such queries are most stable given the hardware limitations. Therefore, each query needs to be divided into multiple smaller queries.
2. Define circuit types and names.
3. Search for EE events for selected circuits.
4. For each EE event:
    1. Search for a corresponding PC event with the same circuit name and within ±100ms seconds of the EE event.
    2. Query I_MEAS (absolute current) value which is nearest to the PC timestamp.
    3. Query additional digital signals (like flags ST_FPA_REC_0 – fast power abort, ST_SW_A_OPEN – switch A opening, ST_SYST_FAIL_0 – system failure, etc.).

In the current configuration of LHC, there have been two types of EE systems: 600A and 13kA:

- 202 600A circuits,
- 32 13kA type (8 RB EVEN, 8 RB ODD, 16 RQ).

There are some subtle technical differences between queries for both types, thus there are two separate scripts and two sets of results, one for each type.



Reliability Study Report: HL-LHC Vacuum Energy Extraction System

Figure 10 Histogram of openings of 202 600A switches between January 2015 and December 2018

Database queries for 600A systems returned 15,290 EE events with corresponding Power Converter (PC) events, of which 2/3 were performed with absolute current value above 10A, while 1/3 – below. Absolute current values are illustrated in Figure 13.

The openings are clearly more frequent in testing periods after technical stops, as depicted in Figure 10. In 2016, the peak was several times higher than the average number of openings in any given month of that year. Generally, in every examined year, the number of openings reaches the highest level in the first months (the testing periods) and then decreases to a lower – but fluctuating – value in the following months.

Figure 14 presents the varying number of openings in different switches. In the period between the beginning of 2015 and the end of 2018, some switches were triggered less than 40 times, while some others – more than 150. This translates to minimal interval between openings of 150 hours and maximal of 600 hours for an individual switch.

However, it is worth noting that such minimal and maximal estimations of intervals are slightly skewed, as the identified events are not distributed evenly. As shown in Figure 10, most system discharges happen in early parts of year, in the time of tests and commissioning runs. Those tests have also the highest absolute current, and a somewhat repeatable pattern. Discharges in other parts of the year are more distributed, although they also tend to cluster around technical stops during the year (June, September, November).

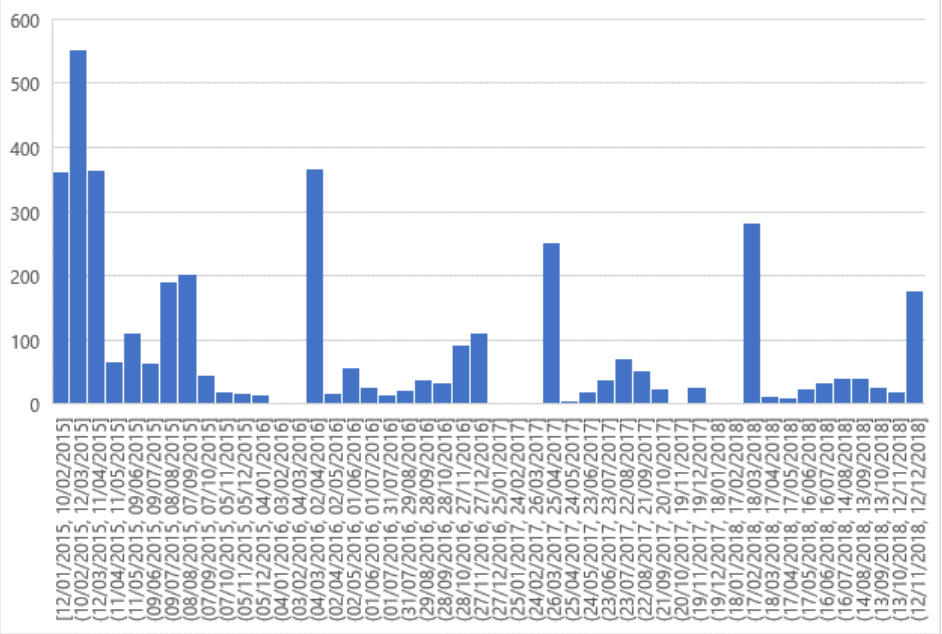


Figure 11 Histogram of openings of 13kA switches between January 2015 and December 2018

Similar patterns are visible in data gathered for 13kA switches, as seen in Figure 11 and Figure 12. Early months of the year usually result in many discharges, while in other months their number decreases, with singular openings per switch per month. Another consideration is the total number of

switches. Even though in total numbers 600A are used more often, generally there are more openings per switch of the 13kA (123 on average, compared to 75 in 600A).

Figure 12 Absolute current at PC timestamp in 13kA circuits between January 2015 and December 2018

Reliability Study Report: HL-LHC Vacuum Energy Extraction System

Figure 13 Absolute current values measured at the PC event timestamp in the circuit for each EE event matched with PC event for 600A switches

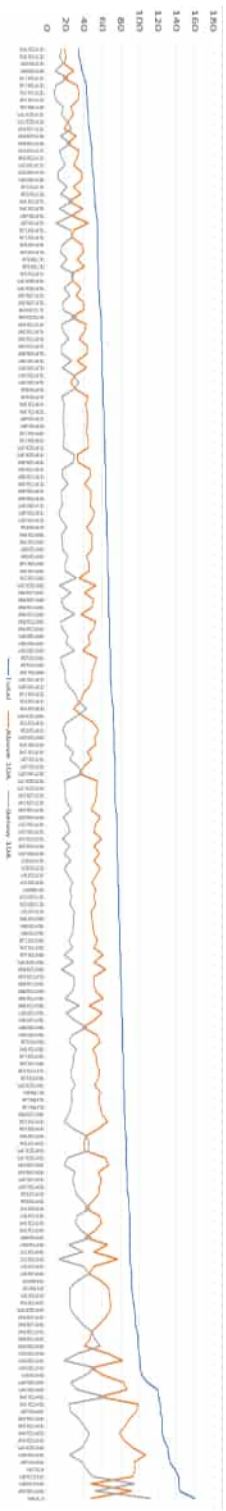


Figure 14 Number of EE events with corresponding PC events of 600A switches; horizontal axis represents different switches (a large size figure is attached in the appendix)

Appendix D

Contributions to AvailSim4

Throughout the duration of the project, various contributions to AvailSim4 software have been made. The two main ones are:

- a post-processing tool for aggregating partial results of distributed cluster execution,
- a root cause analysis feature.

D.1 Support for distributed processing in AvailSim4

Distributed processing in AvailSim4 had been supported through specialized module responsible for starting multiple instances using a cluster infrastructure, HTCondor, which is a solution available at CERN. The solution is an example of the High Throughput Computing (HTC) rather than High Performance Computing (HPC).

In practical terms, this means that even though applications can run in parallel, any communication between separate instances is impossible. Applications which require a classic HPC infrastructure can run using SLURM workload manager, however resources assigned to it are much more limited.

Given the fact that plain Monte Carlo can be parallelized in a trivial way, the distributed computing takes place in the simplest form: a starting script launches distinct copies of the program with the same input data, differing only in the value of the seed used to initialize the pseudo-random generator in each instance. The nature of the problem makes it well balanced, as on average each instance has to compute approximately the same share of the overall workload. The only problematic part is

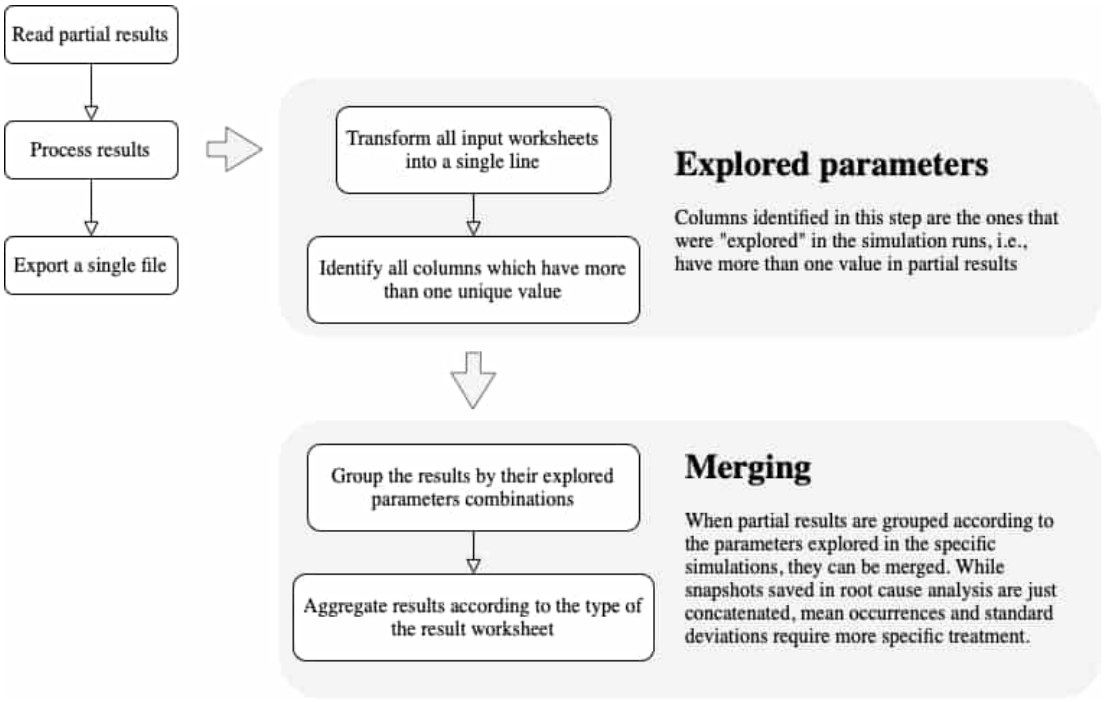


FIGURE D.1: Block diagram of the post-processing tool

reduction of the results – highly parallelized execution can produce thousands or even tens of thousands files, as it had been the case in the HL-LHC EES study.

As a part of this project, we developed a tool to combine those partial results into a single file, that is as easy to use as a result of a single-core execution. In a HPC environment this tool could be integrated into a single application, which would take care of the reduction phase, aggregating the results calculated in separate nodes. However, since the HTC environment is the primary one, the tool is a standalone extension that can be run on the cluster and on a local machine alike.

The algorithm is rather straightforward, as presented in figure D.1. The only more interesting part is combining the standard deviations of based on information gathered in the partial results.

The tool has been used extensively to process the results of the HL-LHC EES reliability study, as extensive reliability runs lasting usually around 10-12 hours each produced tens of thousands of partial result files.

D.2 Root Cause Analysis feature

Another feature developed alongside this project is the root cause analysis (RCA). The notion behind it is rather elementary: to understand the root cause of any specific

event in the model evaluations, we would like to know what were the statuses of individual components in the entire system. The default information which used to be passed with the change of the first component status which was the change that was causing the cascade of status changes in the more high-level components. The problem is that this information does not provide enough information about the failure. For instance, when there is a failure in the system that occurs relatively frequently, however causes significant problems only when some other components are also failed, there is no way of knowing which components are behind those failures most often.

The analysis feature enables users to define triggers via selecting specific components and their statuses. Whenever any of the specified components enters status listed by the user, a global snapshot of all components' statuses is taken and stored for saving in the output file later.

Bibliography

[1] James L. Beck and Konstantin M. Zuev. Rare event simulation, 2015.

[2] CERN/AT/PhL. Interim summary report on the analysis of the 19 september 2008 incident at the lhc, 2008. URL https://edms.cern.ch/ui/file/973073/1/Report_on_080919_incident_at_LHC__2_.pdf.

[3] Andrea Apollonio, Markus Brugger, Lucio Rossi, Ruediger Schmidt, Benjamin Todd, Daniel Wollmann, and Markus Zerlauth. Roadmap towards High Accelerator Availability for the CERN HL-LHC Era. page TUPTY053. 4 p, 2015. URL https://cds.cern.ch/record/2141848.

[4] Moshe Shaked and J. George Shanthikumar. Chapter 13 reliability and maintainability. In *Stochastic Models*, volume 2 of *Handbooks in Operations Research and Management Science*, pages 653–713. Elsevier, 1990. doi: https://doi.org/10.1016/S0927-0507(05)80177-X. URL https://www.sciencedirect.com/science/article/pii/S092705070580177X.

[5] K. D. Tocher. *The art of simulation*. English Universities Press London, 1963.

[6] M. Pidd and Ricardo Cassel. Three phase simulation in java. pages 367–371 vol.1, 01 1999. ISBN 0-7803-5133-9. doi: 10.1109/WSC.1998.745010.

[7] Jérôme Morio, Mathieu Balesdent, Damien Jacquemart, and Christelle Vergé. A survey of rare event simulation methods for static input–output models. *Simulation Modelling Practice and Theory*, 49:287–304, December 2014. doi: 10.1016/j.simpat.2014.10.007. URL https://hal.archives-ouvertes.fr/hal-01081888.

[8] Herman Kahn. Modification of the Monte Carlo method. *Semindar on Scientific Computations*, 1950. C. C. Hurd ed., November 16-18, 1949, IBM.

[9] Herman Kahn Gerald Goertzel. Monte Carlo Methods for Shield Computation. 1951. U.S. Atomic Energy Commission, Technical Information Division.

[10] George S. Fishman. *Monte Carlo. Concepts, Algorithms, and Applications*. Springer, 1999.

[11] Mark Denny. Introduction to importance sampling in rare-event simulations. *European Journal of Physics*, 22(4):403–411, jul 2001. doi: 10.1088/0143-0807/22/4/315. URL https://doi.org/10.1088/0143-0807/22/4/315.

[12] Reuven Y. Rubinstein. *Simulation and the Monte Carlo Method*. John Wiley & Sons, 1981.

[13] E.E. Lewis and Franz Böhm. Monte carlo simulation of markov unreliability models. *Nuclear Engineering and Design*, 77(1):49–62, 1984. ISSN 0029-5493. doi: https://doi.org/10.1016/0029-5493(84)90060-8. URL https://www.sciencedirect.com/science/article/pii/0029549384900608.

[14] Perwez Shahabuddin. *Simulation and analysis of highly reliable systems*. dissertation, Stanford University, 1990.

[15] Philip Heidelberger. Fast simulation of rare events in queueing and reliability models. *ACM Trans. Model. Comput. Simul.*, 5(1):43–85, January 1995. ISSN 1049-3301. doi: 10.1145/203091.203094. URL https://doi.org/10.1145/203091.203094.

[16] Ad Ridder. Importance sampling simulations of markovian reliability systems using cross-entropy. *Annals of Operations Research*, 134, 03 2004. doi: 10.1007/s10479-005-5727-9.

[17] Vincent Dubourg, François Deheeger, and Bruno Sudret. Metamodel-based importance sampling for the simulation of rare events. *arXiv: Methodology*, 2011.

[18] Harald Niederreiter. *Random Number Generation and Quasi-Monte Carlo Methods*. Society for Industrial and Applied Mathematics, USA, 1992. ISBN 0898712955.

[19] Art B. Owen. Quasi-monte carlo sampling.

[20] Johannes van der Corput. Verteilungsfunktionen I and Verteilungsfunktionen II. volume 38, page 813–821, 1935. Proceedings of the Koninklijke Akademie van Wetenschappen Te Amsterdam.

[21] John H. Halton. On the efficiency of certain quasi-random sequences of points in evaluating multi-dimensional integrals. *Numerische Mathematik*, 2:84–90, 1960. URL http://eudml.org/doc/131448.

[22] Ilya M. Sobol'. On the distribution of points in a cube and the approximate evaluation of integrals. *USSR Computational Mathematics and Mathematical Physics*, 7(4):86–112, 1967. ISSN 0041-5553. doi: https://doi.org/10.1016/0041-5553(67)90144-9. URL https://www.sciencedirect.com/science/article/pii/0041555367901449.

[23] Henri Faure. Discrépances de suites associées à un système de numération (en dimension un). *Bulletin de la Société Mathématique de France*, 109:143–182, 1981. doi: 10.24033/bsmf.1935. URL http://www.numdam.org/articles/10.24033/bsmf.1935/.

[24] Harald Niederreiter. Point sets and sequences with small discrepancy. *Monatshefte für Mathematik*, 104:273–338, 1987. URL http://eudml.org/doc/178356.

[25] Spassimir H. Paskov and Joseph F. Traub. Faster valuation of financial derivatives. *The Journal of Portfolio Management*, 22(1):113–123, 1995. ISSN 0095-4918. doi: 10.3905/jpm.1995.409541. URL https://jpm.pm-research.com/content/22/1/113.

[26] Shin Harase. Comparison of sobol' sequences in financial applications. *Monte Carlo Methods and Applications*, 25(1):61–74, Mar 2019. ISSN 1569-3961. doi: 10.1515/mcma-2019-2029. URL http://dx.doi.org/10.1515/mcma-2019-2029.

[27] Christiane Lemieux. *Monte Carlo and Quasi-Monte Carlo Sampling*. 01 2009. ISBN 978-0-387-78164-8. doi: 10.1007/978-0-387-78165-5.

[28] William J. Morokoff and Russel E. Caflisch. Quasi-monte carlo integration. *Journal of Computational Physics*, 122(2):218–230, 1995. ISSN 0021-9991. doi: https://doi.org/10.1006/jcph.1995.1209. URL https://www.sciencedirect.com/science/article/pii/S0021999185712090.

[29] Ian H Sloan and Henryk Woźniakowski. When are quasi-monte carlo algorithms efficient for high dimensional integrals? *Journal of Complexity*, 14(1):1–33, 1998. ISSN 0885-064X. doi: https://doi.org/10.1006/jcom.1997.0463. URL https://www.sciencedirect.com/science/article/pii/S0885064X97904635.

[30] Søren Asmussen and Peter W. Glynn. Part B Algorithms for Special Models, Chapter IX Numerical Integration. In *Stochastic Simulation: Algorithms and Analysis*, Stochastic Modelling and Applied Probability, pages 260–273. Springer, 2007. doi: https://doi.org/10.1007/978-0-387-69033-9.

[31] A. Papageorgiou. Sufficient conditions for fast quasi-monte carlo convergence. *Journal of Complexity*, 19(3):332–351, 2003. ISSN 0885-064X. doi: https://doi.org/10.1016/S0885-064X(02)00004-3. URL https://www.sciencedirect.com/science/article/pii/S0885064X02000043. Oberwolfach Special Issue.

[32] Ilya M. Sobol', Danil Asotsky, Alexander Kreinin, and Sergei Kucherenko. Construction and comparison of high-dimensional sobol' generators. *Wilmott*, 2011 (56):64–79, 2011. doi: https://doi.org/10.1002/wilm.10056. URL https://onlinelibrary.wiley.com/doi/abs/10.1002/wilm.10056.

[33] Stefano Scoleri, Marco Bianchetti, and Sergei S. Kucherenko. Application of quasi monte carlo and global sensitivity analysis to option pricing and greeks. *ERN: Options (Topic)*, 2017.

[34] Pierre L'Ecuyer, Valérie Demers, and Bruno Tuffin. Rare events, splitting, and quasi-monte carlo. *ACM Trans. Model. Comput. Simul.*, 17(2):9–es, April 2007. ISSN 1049-3301. doi: 10.1145/1225275.1225280. URL https://doi.org/10.1145/1225275.1225280.

[35] M. Balesdent, J. Morio, C. Vergé, and R. Pastel. 5 - simulation techniques. In Jérôme Morio and Mathieu Balesdent, editors, *Estimation of Rare Event Probabilities in Complex Aerospace and Other Systems*, pages 45–75. Woodhead Publishing, 2016. ISBN 978-0-08-100091-5. doi: https://doi.org/10.1016/B978-0-08-100091-5.00005-8. URL https://www.sciencedirect.com/science/article/pii/B9780081000915000058.

[36] Pierre L'Ecuyer, Valerie Demers, and Bruno Tuffin. Splitting for rare-event simulation. In *Proceedings of the 2006 Winter Simulation Conference*, pages 137–148, 2006. doi: 10.1109/WSC.2006.323046.

[37] Manuel Villén-Altamirano and José Villén-Altamirano. Restart: A method for accelerating rare event simulations. 3, 01 1991.

[38] Julian Andrade, Wojciech Burakowski, and Manuel Villen-Altamirano. Characterization of cell traffic generated by an atm source. pages 545–550, 1991.

[39] Carlos Budde, Pedro D'Argenio, and Holger Hermanns. Rare event simulation with fully automated importance splitting. volume 9272, pages 275–290, 08 2015. ISBN 978-3-319-23266-9. doi: 10.1007/978-3-319-23267-6_18.

[40] José Villén-Altamirano. An improved variant of the rare event simulation method restart using prolonged retrials. *Operations Research Perspectives*, 6:100108, 2019. ISSN 2214-7160. doi: https://doi.org/10.1016/j.orp.2019.100108. URL https://www.sciencedirect.com/science/article/pii/S2214716018303427.

[41] Carlos E. Budde and Arnd Hartmanns. *Replicating RESTART with Prolonged Retrials: An Experimental Report*, pages 373–380. Lecture Notes in Computer Science. Springer, Netherlands, March 2021. ISBN 978-3-030-72012-4. doi: 10.1007/978-3-030-72013-1_21. 27th International Conference on Tools and Algorithms for the Construction and Analysis of Systems, TACAS 2021, TACAS 2021 ; Conference date: 27-03-2021 Through 01-04-2021.

[42] José Villén-Altamirano. Restart vs splitting: A comparative study. *Performance Evaluation*, 121-122:38–47, 2018. ISSN 0166-5316. doi: https://doi.org/10.1016/

j.peva.2018.02.002. URL https://www.sciencedirect.com/science/article/pii/S0166531616300839.

[43] Siu-Kui Au and James L. Beck. Estimation of small failure probabilities in high dimensions by subset simulation. *Probabilistic Engineering Mechanics*, 16(4):263–277, 2001. ISSN 0266-8920. doi: https://doi.org/10.1016/S0266-8920(01)00019-4. URL https://www.sciencedirect.com/science/article/pii/S0266892001000194.

[44] W. K. Hastings. Monte carlo sampling methods using markov chains and their applications. *Biometrika*, 57(1):97–109, 1970. ISSN 00063444. URL http://www.jstor.org/stable/2334940.

[45] Understanding the metropolis-hastings algorithm. *The American Statistician*, 49(4):327–335, 1995. ISSN 00031305. URL http://www.jstor.org/stable/2684568.

[46] Guangxin Jiang and Michael C. Fu. Importance splitting for finite-time rare event simulation. *IEEE Transactions on Automatic Control*, 63(6):1760–1767, 2018. doi: 10.1109/TAC.2017.2758171.

[47] Zdravko Botev and Dirk Kroese. Efficient monte carlo simulation via the generalized splitting method. *Statistics and Computing*, 22:1–16, 01 2012. doi: 10.1007/s11222-010-9201-4.

[48] Marnix Joseph Johann Garvels. *The splitting method in rare event simulation*. PhD thesis, 2000. URL http://purl.utwente.nl/publications/29637.

[49] Paul Glasserman, Philip Heidelberger, Perwez Shahabuddin, and Tim Zajic. Multilevel splitting for estimating rare event probabilities. *Operations Research*, 47(4): 585–600, 1999. doi: 10.1287/opre.47.4.585.

[50] Jing Li and Dongbin Xiu. Evaluation of failure probability via surrogate models. *Journal of Computational Physics*, 229:8966–8980, 11 2010. doi: 10.1016/j.jcp.2010.08.022.

[51] Anirban Basudhar, Samy Missoum, and Antonio Harrison Sanchez. Limit state function identification using support vector machines for discontinuous responses and disjoint failure domains. *Probabilistic Engineering Mechanics*, 23(1):1–11, 2008. ISSN 0266-8920. doi: https://doi.org/10.1016/j.probengmech.2007.08.004. URL https://www.sciencedirect.com/science/article/pii/S026689200700029X.

[52] Mathieu Balesdent, Jérôme Morio, and Julien Marzat. Kriging-based adaptive importance sampling algorithms for rare event estimation. *Structural Safety*, 44:1–10,

2013. ISSN 0167-4730. doi: https://doi.org/10.1016/j.strusafe.2013.04.001. URL https://www.sciencedirect.com/science/article/pii/S0167473013000350.

[53] W. Fauriat and N. Gayton. AK-SYS: An adaptation of the AK-MCS method for system reliability. *Reliability Engineering and System Safety*, 123(C):137–144, 2014. doi: 10.1016/j.ress.2013.10.01. URL https://ideas.repec.org/a/eee/reensy/v123y2014icp137-144.html.

[54] Zhifu Zhu and Xiaoping Du. Reliability Analysis With Monte Carlo Simulation and Dependent Kriging Predictions. *Journal of Mechanical Design*, 138(12), 09 2016. ISSN 1050-0472. doi: 10.1115/1.4034219. URL https://doi.org/10.1115/1.4034219. 121403.

[55] F. Cadini, F. Santos, and E. Zio. An improved adaptive kriging-based importance technique for sampling multiple failure regions of low probability. *Reliability Engineering & System Safety*, 131:109–117, 2014. ISSN 0951-8320. doi: https://doi.org/10.1016/j.ress.2014.06.023. URL https://www.sciencedirect.com/science/article/pii/S0951832014001537.

[56] Rui Teixeira, Maria Nogal, and Alan O'Connor. Adaptive approaches in metamodel-based reliability analysis: A review. *Structural Safety*, 89:102019, 2021. ISSN 0167-4730. doi: https://doi.org/10.1016/j.strusafe.2020.102019. URL https://www.sciencedirect.com/science/article/pii/S0167473020300989.

[57] Sheldon Ross. Chapter 9 - variance reduction techniques. In Sheldon Ross, editor, *Simulation (Fifth Edition)*, pages 153–231. Academic Press, fifth edition edition, 2013. ISBN 978-0-12-415825-2. doi: https://doi.org/10.1016/B978-0-12-415825-2.00009-7. URL https://www.sciencedirect.com/science/article/pii/B9780124158252000097.

[58] Sheldon Ross. Chapter 10 - additional variance reduction techniques. In Sheldon Ross, editor, *Simulation (Fifth Edition)*, pages 233–246. Academic Press, fifth edition edition, 2013. ISBN 978-0-12-415825-2. doi: https://doi.org/10.1016/B978-0-12-415825-2.00010-3. URL https://www.sciencedirect.com/science/article/pii/B9780124158252000103.

[59] *MIL-217 (1991) Military Handbook: Reliability Prediction of Electronic Equipment: MIL-HDBK-217F*. U.S. Department of Defense, Washington DC, 1991.

[60] N. Heckert, James Filliben, C Croarkin, B Hembree, William Guthrie, P Tobias, and J Prinz. Handbook 151: Nist/sematech e-handbook of statistical methods, 2002-11-01 00:11:00 2002.

[61] S.-C. T. Choi, F. J. Hickernell, M. McCourt, and A. Sorokin. QMCPy: A quasi-Monte Carlo Python library, 2020+. URL https://github.com/QMCSoftware/QMCSoftware.

[62] Sou-Cheng T. Choi, Fred J. Hickernell, R. Jagadeeswaran, Michael J. McCourt, and Aleksei G. Sorokin. Quasi-monte carlo software. *CoRR*, abs/2102.07833, 2021. URL https://arxiv.org/abs/2102.07833.

[63] Stephen Joe and Frances Y. Kuo. Notes on generating sobol' sequences, 2008. URL https://web.maths.unsw.edu.au/~fkuo/sobol/joe-kuo-notes.pdf.

[64] Paul Bratley and Bennett L. Fox. Algorithm 659: Implementing sobol's quasirandom sequence generator. *ACM Trans. Math. Softw.*, 14(1):88–100, mar 1988. ISSN 0098-3500. doi: 10.1145/42288.214372. URL https://doi.org/10.1145/42288.214372.

[65] I.A. Antonov and V.M. Saleev. An economic method of computing lp-sequences. *USSR Computational Mathematics and Mathematical Physics*, 19(1):252–256, 1979. ISSN 0041-5553. doi: https://doi.org/10.1016/0041-5553(79)90085-5. URL https://www.sciencedirect.com/science/article/pii/0041555379900855.

[66] Art B. Owen. On dropping the first sobol' point. *ArXiv*, abs/2008.08051, 2020. URL https://statweb.stanford.edu/~owen/reports/dropzero.pdf.

[67] Pierre L'Ecuyer. *Randomized Quasi-Monte Carlo: An Introduction for Practitioners*, pages 29–52. 07 2018. ISBN 978-3-319-91435-0. doi: 10.1007/978-3-319-91436-7_2.

[68] Douglas Thain, Todd Tannenbaum, and Miron Livny. Distributed computing in practice: the condor experience. *Concurrency - Practice and Experience*, 17(2-4): 323–356, 2005.

[69] The SciPy community. SciPy documentation: Statistics (scipy.stats), 2008-2021. URL https://docs.scipy.org/doc/scipy/reference/tutorial/stats.html#introduction.

[70] Gordon E. Moore. Cramming more components onto integrated circuits, 1965. URL https://newsroom.intel.com/wp-content/uploads/sites/11/2018/05/moores-law-electronics.pdf.

[71] International Energy Agency. Data centres and data transmission networks, 2021. URL https://www.iea.org/reports/data-centres-and-data-transmission-networks.

[72] Brian D. Ripley. *Stochastic Simulation.* John Wiley & Sons, Inc., USA, 1987. ISBN 0471818844.