The splitting method in rare event simulation

Marnix Garvels
Faculty of Mathematical Sciences
University of Twente
P.O. Box 217
7500 AE Enschede
The Netherlands

ISBN 90-365-14320

THE SPLITTING METHOD IN RARE EVENT SIMULATION

PROEFSCHRIFT

ter verkrijging van
de graad van doctor aan de Universiteit Twente,
op gezag van de rector magnificus,
prof. dr. F.A. van Vught,
volgens besluit van het College voor Promoties
in het openbaar te verdedigen
op vrijdag 20 oktober 2000 te 13.15 uur.

door

Marnix Joseph Johann Garvels
geboren op 7 juni 1971
te Vlagtwedde

Dit proefschrift is goedgekeurd door
prof. dr. ir. J.H.A. de Smit, promotor,
prof. dr. ir. I.G.M.M. Niemegeers, co-promotor, en
dr. ir. V.F. Nicola, assistent-promotor.

Voorwoord

Het werk besloten in dit proefschrift is tot stand gekomen dankzij een grote inspanning van vele betrokkenen.

Allereerst wil ik Dirk Kroese bedanken voor het prettige samenwerkingsverband, de vele ideeën voor de splitting method zoals de entrance distribution en de duality methode. Ook zijn inwijding in Mathematica en Fortran en het opzetten van papers zijn bijzonder waardevolle toevoegingen geweest in het onderzoek.

Natuurlijk waren ook de bijdragen van Jos de Smit, Victor Nicola en Jan-Kees van Ommeren, die het onderzoek begeleid hebben gedurende het eerste jaar en de laatste twee jaren, zeer nuttig en altijd uitdagend.

Verder dank ik mijn familie, vrienden en bekenden die mij geholpen hebben bij het promotiewerk. In het bijzonder dank ik Francis voor al haar hulp en steun.

Als laatste dank ik het Centrum voor Telematica en Informatie Technologie voor het bieden van deze mogelijkheid tot het doen van dit onderzoek, en natuurlijk de rekenfaciliteiten van de Faculteit der Informatica, zonder welke deze resultaten niet mogelijk zouden zijn geweest.

Marnix Garvels *Enschede, mei 2000*

Contents

1 Introduction **1**

1.1 Rare Events in Queueing Models of Communication Networks . . 1

1.2 Statement and Objectives 2

1.3 Practical applications 3

1.4 Contributions in this thesis 3

1.5 Layout of the thesis 4

2 The splitting method **5**

2.1 Introduction 5

2.2 Basic Setting 6

2.2.1 Basic asymptotics framework 7

2.2.2 Monte Carlo simulation 8

2.2.3 Importance Sampling 9

2.2.4 Importance Splitting 10

2.3 Basic Splitting Method 11

2.3.1 Overflow probabilities 11

2.3.2 Importance Function 11

2.3.3 Stages and thresholds 12

2.3.4 Splitting 13

2.3.5 Starting and Stopping 14

2.4 Analysis 14

2.4.1 Distribution of the number of successes 15

2.4.2 Estimator 15

2.4.3 Variance 17

2.4.4 Efficiency 18

2.4.5 Complexity 20

2.4.6 Dependencies 21

2.4.7 Multiple thresholds with dependence 22

2.4.8 Convergence 27

2.5 Related Methods 29
2.6 Conclusions 33

3 Implementation Issues 35
3.1 Introduction 35
3.2 Discrete parameters 36
3.2.1 Number of Stages 36
3.2.2 Discrete State Space 36
3.2.3 Discrete Number of Trials 37
3.3 Thresholds 37
3.3.1 Choosing the thresholds 38
3.3.2 Crossing multiple thresholds 38
3.3.3 The Importance Function 41
3.4 Truncation 41
3.5 Global and Single Stepping 43
3.5.1 Fixed Effort vs. Fixed Splitting 44
3.5.2 Fixed Assignment vs. Random Assignment 45
3.6 RESTART 48
3.6.1 Truncation variant of RESTART 54
3.7 Dependency Effects 55
3.7.1 Optimal number of stages 56
3.7.2 Variation coefficients 56
3.7.3 Pilot run algorithm 57
3.8 Simulation Results 58
3.8.1 Global and Single Stepping 59
3.8.2 Truncation and RESTART methods 63
3.9 Conclusions 68

4 The entrance distribution 71
4.1 Introduction 71
4.2 The entrance distribution 72
4.3 Limiting entrance distribution 76
4.4 Simulation experiments 79
4.4.1 $M/G/1$-queue 80
4.4.2 Decay Rate Estimation 82
4.4.3 Markov-modulated $M/M/1$ queue with breakdowns . . . 82
4.4.4 Tandem queue 84
4.5 Conclusions 85

5 The importance function 87
5.1 Introduction . . . 87
5.2 The Importance Function . . . 88
5.2.1 One Dimensional State Space . . . 89
5.2.2 Multi-dimensional State Space . . . 90
5.2.3 Discrete State Space Stochastic Processes . . . 94
5.3 Implementation . . . 95
5.3.1 Defining the optimal importance function . . . 95
5.3.2 Reverse Time Analysis . . . 97
5.4 Simulation Results . . . 104
5.4.1 Two-node tandem queue . . . 105
5.4.2 Three-node tandem queue . . . 109
5.5 Conclusions . . . 110

6 Steady state splitting simulation 111
6.1 Introduction . . . 111
6.2 Simple steady state estimators . . . 111
6.2.1 Enhancements . . . 114
6.2.2 Bootstrapping equivalent . . . 117
6.3 Simulation results . . . 120
6.3.1 $M/M/1$ queue . . . 120
6.3.2 Two-node tandem queue . . . 122
6.3.3 ATM buffer switch . . . 123
6.4 Conclusions . . . 126

7 Duality in steady state splitting simulation 127
7.1 Introduction . . . 127
7.2 Problem setting . . . 128
7.2.1 Duality and first passage times . . . 128
7.2.2 Stochastic Recursions . . . 130
7.2.3 Setup for duality simulation . . . 130
7.3 Simulation results . . . 134
7.3.1 $M/M/1$ queue . . . 134
7.3.2 Two-node tandem queue . . . 135
7.3.3 Markov modulated fluid queue . . . 136
7.4 Conclusions . . . 138

A Estimating the variance for the FE method 139

B Matrix methods for the tandem queue 141
B.1 The entrance distribution for the tandem queue . . . 141
B.2 The importance function for the tandem queue . . . 143

C Markov modulated fluid queue analysis **145**
C.1 Analysis 145
C.2 Starting Distribution, original process 147
C.3 Asymptotic entrance distribution 148

D Simulation setup **151**

Bibliography **153**

Index **158**

Chapter 1

Introduction

1.1 Rare Events in Queueing Models of Communication Networks

Communication networks sparked the theory of queueing systems around the end of the nineteenth century, simply because the communication network at that time, i.e. the telephone, could not afford to connect all extensions with all other extensions. Extensions were connected to local switches, which in turn were connected with one big trunk cable to other local switches. The problem for the phone company was how to choose the capacity of the trunk cables, such that the cable cost was minimal and the availability of the phone service remained high for the consumers. The commonly used model for this problem involves a lot of randomness, inherent to the behavioural patterns of the users of the telephone service and needed a new approach for a solution. The theory that evolved became the more general theory of queueing systems, which covers processes in which customers arrive at and leave systems, consisting of a waiting room and a server, after receiving service during some time from the server. The new-born queueing theory in mathematics could only solve some simplified cases where restrictive assumptions were done about the client behaviour. Fortunately, these turned out to hold in practice.

Today the advent of the high-speed communication networks, the Internet and the mobile communication networks have caused behavioural patterns of communication services to evolve, making the old assumptions on client behaviour not holding. An classical example of this is the Internet causing people to use their phone for extended periods of time for their connection to the Internet service provider, whereas they used to use the phone for an average period of three minutes. Another example is the network traffic pattern that has changed from a fixed stream in the telephone network into bursty packetized audio/video streams in the modern high-speed communications networks.

All these new effects multiplied the number of problems which can only be solved by stochastic simulation. It is safe to say that simulation will only be used more and more in solving these problems, simply because modern computing power coupled with specialized simulation packages and languages can provide solutions in many cases.

Stochastic simulation has proven itself in practice; it is commonly used for accurate estimations in many problems. However, there are some limitations to the standard stochastic simulation method in some special cases.

An important class of problems that cannot be efficiently solved using standard simulation is that involving rare events. Since these rare events occur so seldom in a standard simulation, methods and techniques have been developed since the nineteen-fifties to solve some rare event problems. A recent boost in research in this area occurred in the nineteen-nineties when the new fiber-glass networks proved to be so reliable for transmission that the event of the loss of a light signal inside the fiber-glass has become a very rare event. The old rare event simulation methods were refined and some rediscovered, resulting in a better understanding of which methods work for certain types of problems.

1.2 Statement and Objectives

For one of the main rare event simulation techniques, called *the splitting method*, its potential for queueing networks was not clear, simply because it lacked a mathematical foundation. The intuitiveness of the method made its implementation possible without much mathematical basis.

Motivation The motivation for the work in this thesis is to solve queueing network models that are not solved adequately using standard simulation.

Problem Statement In order to solve queueing network problems we will analyse the splitting method and its potential for rare event simulation.

Objectives In order to complete the answer to the Problem Statement we can formulate a number of objectives.

- Clarify foundation of the method.

- Investigate region of asymptotic efficiency.

- Contrast with other known rare event simulation methods.
- Verify correctness of the method using analytical solutions of special models.
- Implementation of new techniques enhancing its usefulness and efficiency.

1.3 Practical applications

In modern communication networks, an important Quality of Service (QoS) parameter is the Cell Loss Ratio (CLR), which is the probability that a transmitted packet gets lost inside the network due to physical deficiencies in the fiber-glass or due to the limited buffering capacities inside the network nodes. Typically the CLR is smaller than $1 \cdot 10^{-9}$, due to the high reliability of the network infrastructure; making the packet loss a rare event. The fact that billions of packets are transmitted every second draws the problem from a rare event in the packet time-scale to a not so rare event in the human time-scale. Arguably the packet loss problem becomes a non-rare event problem when switching to a larger time-scale, but the same rare event problem persists in terms of needing a large simulation time to complete since simulating a second of network traffic will take a multiple of that in a simulation, thus making accurate estimation within a reasonable time-frame impossible.

The estimation of the CLR in telecommunications systems is a prominent example of a practical use of rare event simulation; however many other applications are possible. One can think of a flooding in a weather simulation or a bankruptcy problem for a bank or insurance company.

In this thesis we will however mainly focus on telecommunications problems which are often modelled as queueing network models.

1.4 Contributions in this thesis

This thesis extends and unifies the existing splitting methods, and introduces new splitting strategies for optimizing the simulation effort. All new techniques are tested on a variety of models for validation of the optimal strategy. New concepts are the use of the entrance distribution, the analysis of the importance function and the analysis of splitting methods in steady-state problems. Existing techniques in the splitting method such as RESTART and truncation receive due analysis and simulation testing. Explicit optimal results and known good rules of thumb for remaining problems are given for the correct implementation of an efficient splitting strategy.

1.5 Layout of the thesis

In the next chapter we give the foundation of the basic splitting method, including an analysis of its properties.

In the third chapter we look at the implementations issues involved and design choices for the algorithm. We arrive at the optimal design comparing different implementations on a variety of test models.

The fourth chapter deals with the investigation of methods that take advantage of the entrance distribution which is implicitly estimated during a simulation. Several techniques are tested that take advantage of a certain asymptotic properties of the entrance distribution.

In the fifth chapter we look at the problem of a multi-dimensional state-space. In these types of models we see that the importance function (IF) plays a vital part in guaranteeing the asymptotic optimality of the method. We derive an implicit form of the optimal IF and present an algorithm to estimate it. Again the validation is done by a series of experiments.

Chapters six and seven deal with methods that improve the method for estimation of steady-state parameters. By techniques extending the splitting method we find that we can obtain better results in chapter six. Also, when we transform the system into its dual we see that there is a potential gain because the steady-state problem is reduced to a transient one in chapter seven.

Chapter 2

The splitting method

2.1 Introduction

The splitting method is a simple simulation method for the estimation of rare event probabilities. The method is based on the idea to restart the simulation in certain system states, in order to generate more occurrences of the rare event. The first reference to this type of method in simulation is in a standard work by Kahn and Harris in [KH51], who investigated splitting particles in a physical context. The method was again introduced in 1970 by Bayes in two papers [Bay72] and [Bay70], who considered a simple discrete Markov chain and one threshold. A more recent version of the splitting method was introduced by M. and J. Villén-Altamirano in [VAVA91] named RESTART and further generalised in [VAMMGFC94], [VAVA97] and [VAVA99].

An extension providing bounded relative error called the LRE method has been investigated by Schreiber, Görg and Fuß in [SG94] and enhanced with other methods in [GF98] and [GF99]; a further extension was proposed by Below in [Bel98] and [BBK99].

A tool for stochastic Petri nets was developed by Kelling [Kel96], based on the RESTART method.

Several extensions of the RESTART/splitting methodology were investigated in [GK98], [GK99] and [GKvO00], as well as an analysis of the mathematical properties of the methods.

A more theoretical analysis of the splitting method is found in Glasserman, Heidelberger, Shahabuddin and Zajic in the papers [GHSZ98], [GHSZ99] and [GHSZ96b], where the authors prove asymptotic non-optimality for the method under some conditions.

Another different interpretation of the splitting method is found in Haraszti and Townsend's work [HT99] called DPR (Direct Probability Redistribution).

In this chapter we discuss the basic features of the splitting method. First we situate the method in the family of rare event methods and discuss the field of rare event simulation and its techniques. An asymptotics framework is introduced which is necessary in order to discuss asymptotics results. We describe the basic method and analyse the variance and the asymptotic optimality for the the standard versions.

2.2 Basic Setting

The basic problem setting is that we are observing any stochastic process $X = (X_t, t \geq 0)$ with sampling set S, a general state space E, say, and we want to estimate the probability that a rare event, $R \subset S$ say, occurs. Denote the rare event probability as $\gamma := \mathbb{P}(R)$. We assume that the stochastic process is a Markov process. Since we can incorporate the history of the process into the system state X_t, we can make any time-homogeneous stochastic process obey the Markov conditions; thus the Markov property is not really a limitation of the assumptions necessary for the proper use of the method. However, as we will later see, the fewer variables are necessary to describe the system as a Markov system, the better the method will work. By this we mean that the dimensionality of the state space E of X negatively influences the performance of the splitting method.

Denote T_A as the first time process X enters the rare event set A, and T_B another stopping time for process X. The rare event can then be expressed as $R := \{T_A < T_B\}$, the event that set A is hit before the stopping time and we are looking for the probability $\gamma := \mathbb{P}(T_A < T_B)$.

Example 2.1 Consider a particle lifetime experiment in physics. Suppose at random times particles are generated which will die after a random life time. B is now the event that the particle dies and set A is the event that the particle reaches a certain lifetime (before disappearing). The lifetime of a particle is assumed to be dependent on the path taken during its life. The interesting rare event is the event that a random particle reaches a high age, i.e. it lives for a very long time, which is then easily seen to be equal to $\gamma := \mathbb{P}(T_A < T_B)$.

We will not limit the method to the estimation of this particular class of transient simulation; we can also apply the method in steady state simulation by considering the problem of estimating $\pi(A) = \lim_{t\to\infty} \mathbb{P}(X_t \in A)$.

2.2.1 Basic asymptotics framework

In order to discuss asymptotic properties of estimators and algorithms in general, we introduce basic complexity definitions by Turing (see e.g. [LV93]). We assume that the set A is parameterized by a *rarity parameter* which we will denote as L. By this we mean that the rare event probability $\gamma(L)$ is decreasing in L.

By the *complexity* of a rare event simulation algorithm we mean the time, $T(L)$ say, that the algorithm needs in order to achieve a fixed accuracy as a function of the input parameter L. We assume that the simulation time is proportional to the number of events simulated inside a discrete event simulator performing the simulation. Since the simulation time used, $T(L)$, is usually a random variable, we extend the definition by introducing the (ϵ, δ)-complexity of the random variable $T(L)$ as $t_{\delta,\epsilon}(L)$ as in [RM98] as the smallest number for which $\mathbb{P}(|T(L) - t_{\delta,\epsilon}(L)| < \epsilon t_{\delta,\epsilon}(L)) > 1 - \delta$, holds for small $\delta > 0$, small $\epsilon > 0$ and large sample sizes, i.e. the random time a simulation takes to achieve the fixed accuracy, $T(L)$, falls for large samples with confidence $(1 - \delta)$ below $(1 + \epsilon)t_{\delta,\epsilon}(L)$. When the random variable $T(L)$, the time it takes the rare event simulation algorithm to reach a fixed accuracy, has an asymptotic normal distribution we easily find the complexity $t_{\delta,\epsilon}(L)$ as follows. For large sample size n the distribution function of $T(L)$ approximates $\Phi(\mu, \sigma^2/n)$, where $\Phi(x, y)$ is the distribution function of a normal distributed random variable with expectation x and variance y, for some μ and σ, we easily verify that $t_{\delta,\epsilon}(L) := \mu$, i.e. the (δ, ϵ)-complexity is equal to the expectation of the simulation run-time, is sufficient when the sample size exceeds $\epsilon^{-2}\mu^{-2}\sigma^2\Phi^{-2}(1 - \delta)$ and is high enough for the asymptotic normality to be valid. Note that we can also obtain $t_{\delta,\epsilon}(L)$ easily when the variance of $T(L)$ has a zero limit for large sample sizes using the Chernoff bound. In further discussions of the complexity of stochastic algorithms we will generally omit the proper involvement of $t_{\delta,\epsilon}$ and only concern ourselves with the expectation of $T(L)$ since it usually follows an asymptotic normal distribution for which the complexity was just shown to be equal to the expectation.

The distinction with complexities of estimators is that standard estimator complexity is measured as the number of samples necessary to arrive at a fixed accuracy for a given rarity parameter L; our definition also incorporates the effort it takes to acquire each sample.

The functional form of the complexity is usually simplified using the asymptotic equivalence notation $T(L) = O(h(L))$ for some simple function h, which we define to hold when

$$\limsup_{L\to\infty} \frac{T(L)}{h(L)} = c \in \mathbb{R} \quad \text{with} \quad 0 \leq c < \infty$$

When comparing two algorithms, we say that algorithm A is asymptotically more efficient than algorithm B when $\limsup_{L\to\infty} T_A(L)/T_B(L) = 0$.

Many typical queueing problems exhibit *asymptotic exponential decay*, by which we mean that $\gamma(L) = O(e^{-\theta L})$, with $\theta \in \mathbb{R}^+$. We will classify algorithms for estimating rare event probabilities into two basic categories, depending on their complexities.

- **Exponential** The time necessary for the algorithm to achieve a fixed accuracy grows exponential in L: $T(L) = O(e^{\theta\, L})$ for some $\theta > 0 \in \mathbb{R}$.

- **Polynomial** The time necessary for the algorithm to achieve a fixed accuracy grows polynomially in L: $T(L) = O(L^k)$ for some $k \in \mathbb{N}$.

For exponential decay problems we arrive at the equivalent definition in terms of γ with $T(\gamma) = O(\gamma^{-1})$ for the Exponential class and $T(\gamma) = O((-\log(\gamma))^k)$ for the Polynomial class.

We define the algorithm to be asymptotically efficient when the algorithm belongs to the Polynomial class. The inefficiency of Exponential class algorithms will be demonstrated in the next section where we discuss one such example.

2.2.2 Monte Carlo simulation

The rare event probability estimation problem is hard to do in simulation when the desired probability, which we will call γ, is extremely small. In this case the standard Monte-Carlo (MC) simulation method is very inefficient and simulation takes a very long time. The MC method consists of simulating the system without making any changes to its stochastic behaviour; as a results very few samples will actually hit the rare event. An analytic demonstration will show the inefficiency of the MC method. We have samples $(X_1, \dots, X_n)$ of the n, say, random observations in which the event A may occur; the value of a random variable X_i equals one when the rare event is seen in the i-th attempt, and equals zero otherwise. The MC estimator is simply

$$\widehat{\gamma} = n^{-1} \sum_{i=1}^{n} X_i \tag{2.1}$$

Suppose we want a relative error of at most r in our estimate $\widehat{\gamma}$. Since the variance of a MC estimator using n samples is equal to

$$\mathbb{V}\mathrm{ar}(\widehat{\gamma}) = \frac{\gamma(1-\gamma)}{n}, \tag{2.2}$$

this guarantees that we have a *relative error* of [1]

$$\mathrm{RE}(\widehat{\gamma}) := \frac{\sqrt{\mathbb{V}\mathrm{ar}(\widehat{\gamma})}}{\gamma} = \frac{\sqrt{1-\gamma}}{\sqrt{n\gamma}}. \tag{2.3}$$

Fixing this to r we see that we need at least $n^* = r^{-2}(1-\gamma)/\gamma \approx r^{-2}/\gamma$ samples. For the particular problem where γ has an asymptotic decay (see 2.2.1) we obtain $n^* = O(r^{-2}\,e^{L\theta})$ which grows exponentially fast in L and is thus a problem for large L. In other words, the algorithm is not asymptotically efficient.

Example 2.2 To demonstrate the problem numerically we assume that $n^* = cr^{-2}e^{L\,\theta}$ with $\theta = 2$, $r = 0.01$ and $c = 1$; common in many rare event problems. Furthermore, suppose that the cost in simulation time of one sample is 0.1 msec. Note in the following table 2.1 that MC simulation takes very long in practice for $L > 5$; even with a one thousand times faster supercomputer the cases where $L > 10$ are not feasible in a reasonable amount of time (one day, say).

L	γ	n^*	time
1	0.13	$7.39 \cdot 10^4$	7 sec
2	$1.83 \cdot 10^{-2}$	$5.45 \cdot 10^5$	54 sec
5	$4.54 \cdot 10^{-5}$	$2.20 \cdot 10^8$	6.12 hr
10	$2.06 \cdot 10^{-9}$	$4.85 \cdot 10^{12}$	15.4 yr

Table 2.1: Typical properties of a MC estimator

2.2.3 Importance Sampling

In order to overcome these problems involving rare events, researchers developed techniques that circumvented the MC method's breakdown by having an efficient algorithm for large L; by efficient we mean that the number of samples n^* for the algorithm is a polynomial in L, which ensures that the algorithm is still usable for very large L. We therefore call these techniques rare event simulation methods.

Two main directions have developed in this field of rare event simulation. The first technique works by changing the stochastic process $(X_t, t \geq 0)$ itself.

[1] Note that we introduce the symbol := to denote a definition.

By changing (X_t) we can get a process for which the set A has a high probability, which makes the MC estimator efficient again. To obtain an unbiased estimator we need to weigh the sample values we get to account for the change in stochastic process we are simulating. This factor is also known as the likelihood ratio (LR) because it is the ratio of the likelihoods of the events occurring of the old process and the new process.

The method is called Importance Sampling (IS) since the rare event becomes more likely and thus the important events are sampled more frequently. The well-known problem of the IS method is that it is hard to derive the optimal change: the choice of new process to simulate which will minimize the variance in the estimator. A non-optimal choice could make the behaviour of the IS estimator even worse than the MC estimator.

Another problem is that for complex systems the optimal change always exists but usually cannot be found analytically and/or adaptively. The optimal change within a predetermined twisting family may not be able to improve upon the MC efficiency, and in the worst case a change is chosen which appears to be optimal but gives an infinite variance. We will not describe the IS method and its properties in detail, the reader is referred to Asmussen and Rubinstein [AR95], e.g., where the asymptotic properties of the IS estimator are shown to be able to make the number of necessary samples independent of L in some special cases. Note that this does not automatically mean that in the complexity formulation of section 2.2.1 that $T(L) = O(1)$; the actual number of steps to take in the algorithm per sample may (and for queueing systems usually will) depend on L, since a complete path needs to be traversed leading to the rare event. When in fact the average sample simulation needs $O(L)$ time, it is easily seen that the complexity of the total algorithm becomes $O(L)$.

2.2.4 Importance Splitting

The other main rare event simulation technique, the one which is the subject of the thesis, tries to create many hits of the rare event by a divide and conquer technique. We define a simulation *path* as a realisation of the state space trajectory of the stochastic process. It does not change the system itself to generate more *important* paths, but decides while running a simulation whether it is a promising path or not. A simulation path that looks important will be re-sampled or *split* a number of times to help generate more samples of the rare event, hence the name. We also call it the splitting method since we will split a promising simulation path into a number of smaller paths. These paths are correlated since they share a common history up until the path split but they are more likely to hit the rare event set than ordinary paths. When the decisions when to split and how to split are chosen carefully, the method can be shown to be asymptotically efficient for a large class of rare event problems. We compute the estimation for γ as the number of the paths that hit set A, divided by the product of the number of splits and starting samples.

2.3 Basic Splitting Method

The basic idea of the splitting method is to consider the rare event as the intersection of a nested sequence of events. The probability of the rare event is thus the product of conditional probabilities, each of which can usually be estimated much more accurately than the rare event itself, for a given simulation effort.

2.3.1 Overflow probabilities

In this section we describe the class of problems for which we wish to use the splitting method more precisely. The basic setting is the following.

Consider a Markov process $X := (X_t, t \geq 0)$ with space E, and let f be a real-valued measurable function from E to the real numbers $\mathbb{R}$. Define $Z_t := f(X_t)$, for all $t \geq 0$. Assume for definiteness that $Z_0 \geq 0$. For any *threshold* or *level* $L \geq 0$, let T_L denote the first time that the process $Z := (Z_t, t \geq 0)$ hits the set $[L, \infty)$; and let T_B denote the first time, after 0, that Z hits the set $(-\infty, 0]$ from above. We assume that T_L and T_B are well-defined (possibly infinite) stopping times with respect to the history of X.

We are interested in the probability, γ say, of the event $D_L := \{T_L < T_B\}$, i.e., the probability that Z up-crosses level L before it down-crosses level 0. Note that γ depends on the initial distribution of X. This particular problem is often referred to as the estimation of an *overflow probability* .

2.3.2 Importance Function

The splitting method basics are simple and best described visually. We imagine the state space of the problem we are trying to solve mapped onto a real-valued parameter we will refer to as the level. The function doing the mapping, $f : E \to \mathbb{R}$ is called the Importance Function (IF) and gives us the level of the current state X_t as $Z_t = f(X_t)$. The rare event set A will be of the form $A = \{x \in E | f(x) \geq L\}$, consisting of the highest values of f. Since we completely hid the state space of the process in f, we can view every splitting method as a plot of $Z_t = f(X_t)$ versus t as in the following Figure 2.1.

The evolution of a sample path is now very similar to a ladder height or a dam model where we would watch the water level on the dam, the rare event being the flooding because the water rose higher than the dam height L.

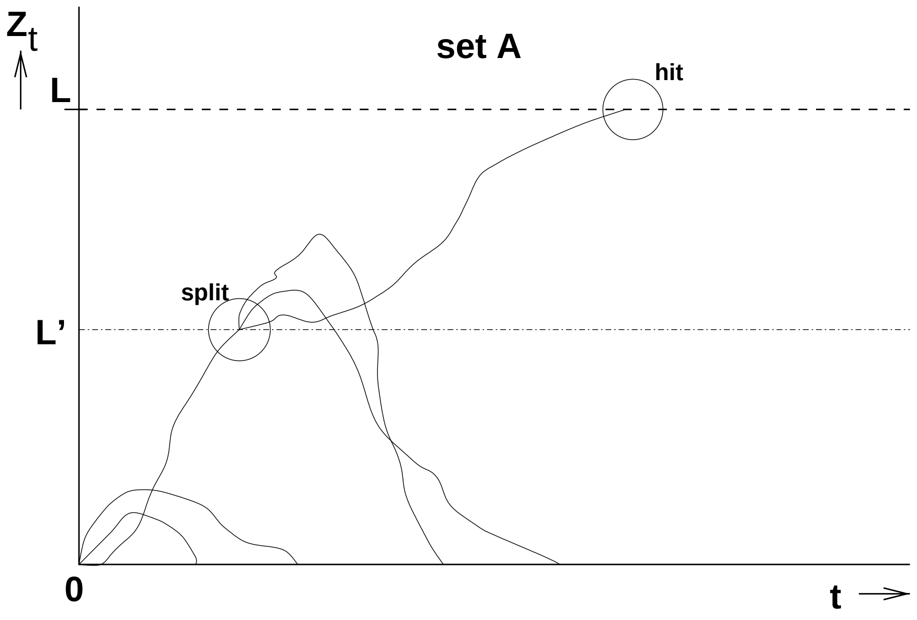

Figure 2.1: *The splitting model*

Example 2.3 As an example of the setting described above, consider a $GI/G/1$ queue. Let Z_t be the number of customers in the system at time t; and let A_t and S_t denote the remaining inter-arrival and service time (when the system is not empty) at time t, respectively. Obviously, $X := (A_t, S_t, Z_t, t \geq 0)$ is a Markov process with a continuous time parameter. Assume that at time zero the first customer arrives in an empty system. A performance measure of interest is the probability that the number of customers during the first regeneration cycle, i.e., before the system becomes empty again, exceeds some level L.

2.3.3 Stages and thresholds

Since it is hard to reach the highest level L we split the natural domain for the level variable (Z_t), $[0, L]$, into m disjunct intervals; effectively dividing the state space into m so-called stages by choosing $0 =: L_0 < L_1 < L_2 < \ldots < L_{m-1} < L_m := L$. Note that we do not count L_0 as a threshold, it is merely the stopping set. Doing so allows us to identify m thresholds and m stages. The m intervals are then given by interval number i : $I_i = [L_{i-1}, L_i]$. In the Figure 2.1 we chose $m = 2$ and $L_1 = L'$ to keep the figure as simple as possible; we present a version with multiple thresholds and a two-dimensional state-space to visualise the thresholds in the E domain in the next Figure 2.2. Note that the contours of the importance function form the thresholds for the splitting procedure. Also, the method always generates a tree: a split path behaves exactly like a tree branching into smaller twigs.

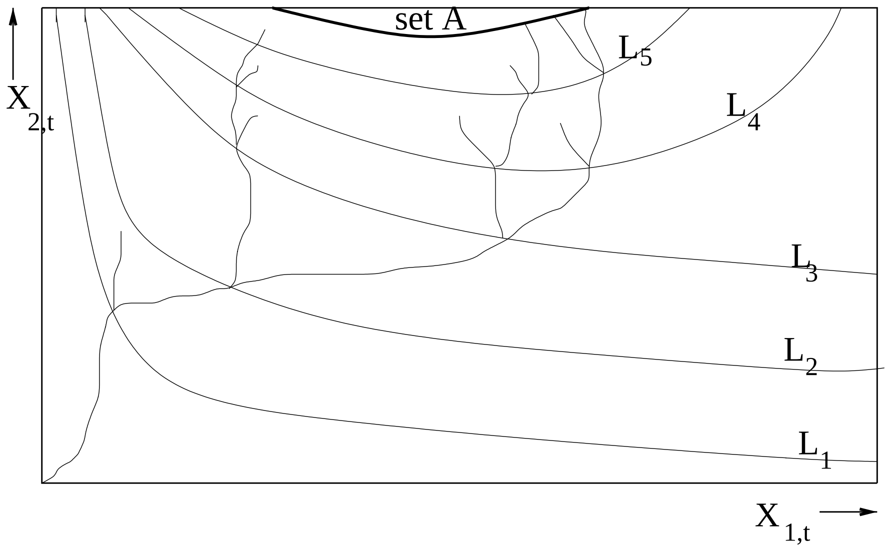

Figure 2.2: *Splitting in the state space* $E = \mathbb{R}^2$

The levels L_i are called thresholds since they govern the entry into the next stage. The number of thresholds, m, and the thresholds themselves are to be chosen such that a sample path crossing a threshold has a noticeable probability of reaching the next threshold. We will come back to the problem of choosing the splitting parameters efficiently later on, but is must be clear that going from one stage to the next must not be a rare event, or there is no benefit in using the algorithm.

2.3.4 Splitting

Upon hitting a threshold, L_i say, the state of the process has to be saved, and n_i identical copies of the saved state are made, each progressing independently after the so called "splitting" of the sample path. This is done for every next threshold that is hit; i.e. every "hit" that reaches the top level L will have split exactly $m-1$ times. Again, care has to be taken to choose the number of splits n_i appropriately; a too small number will generate few hits of the next threshold and this will cause the paths to "die-out" after a certain number of stages. A too large splitting factor n_i will cause too many hits of the next threshold and will cause the number of paths per stage to "snowball". This will result in a large total simulation time, and the highest fraction of it spent on the last stages which is not efficient: the most interesting samples are the ones which start from scratch because they are independent, unlike the samples in the later stages. We will also address the proper choice of the splitting factors in the Analysis section 2.4. In the Figure 2.1 we chose $n_1 = 3$ to keep it simple.

2.3.5 Starting and Stopping

We further assume that we start at level zero, and draw start states that generate sample paths which are independent during the first stage. In the second stage sample paths which have the same "root", i.e. which were generated from identical starting system states must be assumed to be dependent since they share a common history. Although they progress independently after the split, their stochastic properties will be influenced by the common starting state.

The stopping criterion for a sample path can be chosen freely, as long as it is independent of the sample future. Typical examples are using a fixed stopping time, or a stopping time T_B which marks the instant of the first return to the zero level.

2.4 Analysis

Define $R_i, i = 1, 2, \ldots, m$ as the number of hits of threshold i ($R_0 := 1$), and $p_i, i = 1, 2, \ldots, m$ as the probability that a sample path starting at the lower threshold of stage i will hit the upper threshold before stopping. The number of samples starting at time zero in the initial state is equal to n_0; the number of re-trials of a successful path in stage i will be $n_i, i = 1, 2, \ldots, m$.

We introduce the random variables denoting the times at which the process hits the thresholds as

$$T_i = \inf\{t \geq 0 \,|\, Z_t \geq L_i\} \quad i = 0, 1, 2, \ldots, m$$

and set $T_i := \infty$ when threshold i is not hit before T_B.

We define the entrance distribution at threshold i as the distribution of the system state at the instant that threshold i is first reached. In other words, define the entrance distribution at threshold i as the stochastic variable $S_i := X_{T_i}$.

It is obvious that the success probability of a path in stage $i+1$ will depend on the entrance state S_i; denote this as the function

$$p_{i+1}(s) := \mathbb{E}(I_{i+1}(s)) \quad , s \in \mathcal{S}(S_i)$$

where we define $\mathcal{S}(S_i)$ to be the sampling set of S_i, and $I_{i+1}(s)$ is a Bernoulli variable denoting whether a sample path in stage $i+1$, starting from state s hits the next threshold L_{i+1}. Clearly we expect the success probability to equal p_{i+1}:

$$\mathbb{E}(p_{i+1}(S_i)) = p_{i+1} \tag{2.4}$$

2.4.1 Distribution of the number of successes

In stage $i+1$ we have R_i entrance states; denote these as $S_i^{(1)}, S_i^{(2)}, \ldots, S_i^{(R_i)}$. For each of these entrance states we simulate n_i paths and watch whether they hit the next threshold. Denote the result of these observations as the vector of Bernoulli random variables $\{I_{i+1}^{(1)}(S_i^{(1)}), I_{i+1}^{(2)}(S_i^{(1)}), \ldots, I_{i+1}^{(n_i)}(S_i^{(R_i)})\}$. Now we obtain the number of hits of the next threshold as

$$R_{i+1} = \sum_{j=1}^{R_i} \sum_{k=1}^{n_i} I_{i+1}^{(k)}(S_i^{(j)})$$

The sum of the indicators per entrance state, $G_{i+1}(S_i^{(j)}) := \sum_{k=1}^{n_i} I_{i+1}^{(k)}(S_i^{(j)})$ say, can be seen to have a Binomial distribution with parameters n_i and $p_{i+1}(S_i^{(j)})$. We denote this as $G_{i+1}(S_i^{(j)}) \sim \text{Bin}(n_i, p_{i+1}(S_i^{(j)}))$. The distribution of R_{i+1} becomes

$$R_{i+1} \sim \sum_{k=1}^{R_i} \text{Bin}(n_i, p_{i+1}(S_i^{(k)})) \tag{2.5}$$

2.4.2 Estimator

The Monte Carlo estimator for the success probability in stage i is

$$\hat{p}_i = \frac{R_i}{n_{i-1}R_{i-1}} \quad \text{,if and only if} \, R_{i-1} > 0, \quad i = 1, 2, \ldots, m \tag{2.6}$$

since this is the fraction of the samples in stage i that were successful. Note that when $R_i = 0$ for some $0 < i < m$, the estimator $\hat{p}_i = 0$ and all the subsequent R_j with $j > i$ are also zero, making the estimator $\hat{p}_j$ undefined for $j > i$. Since there are no observations of the stages $j > i$ the estimator $\hat{p}_j$ does not exist for $j > i$. We will further assume that we have $R_i > 0 \quad, i = 1, 2, \ldots, m$ since the estimations will be undefined otherwise. Denote the σ-algebra associated with the stochastic process $(R_i)_{i=1}^k$ as $\mathcal{F}_k$. The estimator $\hat{p}_i$ is unbiased since by (2.5), (2.6) and (2.4)

$$\begin{aligned}\mathbb{E}(R_{i+1}|\mathcal{F}_i) &= \mathbb{E}(R_{i+1}|R_i) = \mathbb{E}_{S_i}\{\mathbb{E}(R_{i+1} \mid R_i, S_i^{(1)}, S_i^{(2)}, \ldots, S_i^{(R_i)})\} \\ &= \mathbb{E}_{S_i}\{\mathbb{E}(\sum_{k=1}^{R_i} \text{Bin}\,(n_i, p_{i+1}(S_i^{(k)})) \mid R_i, S_i^{(1)}, S_i^{(2)}, \ldots, S_i^{(R_i)})\} \\ &= \mathbb{E}_{S_i}\{\sum_{k=1}^{R_i} n_i p_{i+1}(S_i^{(k)})\} = n_i p_{i+1} R_i \qquad (2.7)\end{aligned}$$

and applying this to $\hat{p}_i$ results in

$$\mathbb{E}(\hat{p}_i \mid \mathcal{F}_{i-1}) = \mathbb{E}(\frac{R_i}{n_{i-1}R_{i-1}} \mid R_{i-1}) = \frac{\mathbb{E}(R_i \mid R_{i-1})}{n_{i-1}R_{i-1}} = \frac{n_{i-1}p_i R_{i-1}}{n_{i-1}R_{i-1}} = p_i \qquad (2.8)$$

so that we obtain unbiasedness of $\hat{p}_i$ by unconditioning (2.8) on $\mathcal{F}_{i-1}$:

$$\mathbb{E}(\hat{p}_i) = \mathbb{E}(\mathbb{E}(\hat{p}_i \mid \mathcal{F}_{i-1})) = \mathbb{E}(p_i) = p_i \qquad (2.9)$$

Set D_i as the event that threshold i is reached during a simulation

$$D_i := \{T_i < T_B\}$$

Note again that $D_0 = \mathcal{S}(X)$, the entire sample set, since $T_0 = 0 < T_B$ always holds. We see that the rare event $\{T_A < T_B\} = D_m$ and $D_{i+1} \subset D_i \quad, i = 0, 1, 2, \ldots, m-1$, so that necessarily for every j satisfying $m \geq j > i$

$$\mathbb{P}(D_j|D_i) = \frac{\mathbb{P}(D_j \cap D_i)}{\mathbb{P}(D_i)} = \frac{\mathbb{P}(D_j)}{\mathbb{P}(D_i)} = \prod_{k=i+1}^{j} p_k \qquad (2.10)$$

resulting in the equation

$$\gamma = \prod_{i=1}^{m} p_i. \qquad (2.11)$$

We introduce

$$\widehat{\gamma} = \prod_{i=1}^{m} \hat{p}_i = \prod_{i=1}^{m} \frac{R_i}{n_{i-1}R_{i-1}} = \frac{R_m}{\prod_{i=0}^{m-1} n_i} \qquad (2.12)$$

as the estimator for the probability of reaching the highest level.

We now prove that we can exchange product and expectation for the sequence of estimators $(\hat{p}_i)_{i=1}^m$. For any pair $(i,j) \in \{1,2,\dots,m\}^2, i \neq j$ (assume $i > j$ without loss of generality)

$$\mathbb{E}(\hat{p}_i\hat{p}_j) = \mathbb{E}(\mathbb{E}(\hat{p}_i\hat{p}_j \,|\, \mathcal{F}_{i-1})) = \mathbb{E}(\hat{p}_j\mathbb{E}(\hat{p}_i \,|\, \mathcal{F}_{i-1})) = \mathbb{E}(\hat{p}_j p_i) = p_i\mathbb{E}(\hat{p}_j) = p_ip_j \tag{2.13}$$

We can now prove that $\widehat{\gamma}$ is unbiased since by (2.11) and recursively applying (2.13)

$$\mathbb{E}(\widehat{\gamma}) = \mathbb{E}(\prod_{i=1}^m \hat{p}_i) = \prod_{i=1}^m \mathbb{E}(\hat{p}_i) = \prod_{i=1}^m p_i = \gamma \tag{2.14}$$

2.4.3 Variance

Number the first stage sample paths as $j = 1, \dots, n_0$. We assumed these to be independent, and thus the random variable that denotes the number of hits caused by the offspring of first stage path j, denoted by Y_j, are also independent since paths that have a different "root" or first-stage history will be independent. We have

$$\widehat{\gamma} = \frac{R_m}{\prod_{i=0}^{m-1} n_i} = \frac{\sum_{j=1}^{n_0} Y_j}{\prod_{i=0}^{m-1} n_i}$$

making

$$\mathbb{V}\text{ar}(\widehat{\gamma}) = \frac{\mathbb{V}\text{ar}(\sum_{j=1}^{n_0} Y_j)}{\prod_{i=0}^{m-1} n_i^2} = \frac{\mathbb{V}\text{ar}(Y_1)}{n_0\prod_{i=1}^{m-1} n_i^2} \tag{2.15}$$

Using the standard variance estimator

$$S_Y^2 = \frac{\sum_{j=1}^{n_0}(Y_j - \bar{Y})^2}{n_0 - 1}$$

as an unbiased estimator for $\mathbb{V}\text{ar}(Y_1)$ results in an unbiased variance estimator for $\widehat{\gamma}$ using (2.15).

It is very hard to deliver analytic expressions for $\mathbb{V}\text{ar}(Y_1)$ in a multistage simulation because of the many dependencies due to the sharing of the history. For a two-stage simulation we can. We introduce Bernoulli random variables $I_j^{(i)}$ denoting the success or failure of hitting the next threshold for path number j in stage number i. We see that we can express Y_1 as

$$Y_1 = I_1^{(1)} \sum_{i=1}^{n_1} I_i^{(2)} = \text{Bin}(1, p_1)\text{Bin}(n_1, p_2(S_1))$$

where S_1 is the random variable describing the entrance state, which determines the success probability for stage 2, p_2, and thus

$$\mathbb{P}(Y_1 = i) = \begin{cases} p_1 \mathbb{P}(\mathrm{Bin}(n_1, p_2(S_1)) = i) & , \quad i = 1, 2, \ldots, n_1 \\ (1 - p_1) + p_1 \mathbb{P}(\mathrm{Bin}(n_1, p_2(S_1)) = 0) & , \quad i = 0 \end{cases}$$

which leads to

$$\mathbb{E}(Y_1) = p_1 \mathbb{E}(\mathrm{Bin}(n_1, p_2(S_1))) = p_1\, n_1\, \mathbb{E}(p_2(S_1)) = p_1\, n_1\, p_2 \tag{2.16}$$

using (2.4) and

$$\mathbb{E}(Y_1^2) = p_1 \mathbb{E}(\mathrm{Bin}(n_1, p_2(S_1))^2) \tag{2.17}$$

Combining the results from (2.16) and (2.17) we obtain

$$\mathbb{V}\mathrm{ar}(Y_1) = p_1 \mathbb{V}\mathrm{ar}(\mathrm{Bin}(n_1, p_2(S_1))) + p_1(1 - p_1)(n_1 p_2)^2 \tag{2.18}$$

We expand the variance of the binomial as

$$\begin{aligned} & \mathbb{V}\mathrm{ar}(\mathrm{Bin}(n_1, p_2(S_1))) \\ = \; & \mathbb{E}\{\mathbb{V}\mathrm{ar}(\mathrm{Bin}(n_1, p_2(S_1))|S_1)\} + \mathbb{V}\mathrm{ar}\{\mathbb{E}(\mathrm{Bin}(n_1, p_2(S_1))|S_1)\} \\ = \; & \mathbb{E}\{n_1 p_2(S_1)(1 - p_2(S_1)\} + \mathbb{V}\mathrm{ar}\{n_1 p_2(S_1)\} \\ = \; & n_1(p_2(1 - p_2) + (n_1 - 1)\mathbb{V}\mathrm{ar}(p_2(S_1))) \end{aligned} \tag{2.19}$$

Merging (2.18) with (2.19) results in the equation

$$\mathbb{V}\mathrm{ar}(Y_1) = n_1 p_1 \{p_2(1 - p_2) + (n_1 - 1)\mathbb{V}\mathrm{ar}(p_2(S_1)) + (1 - p_1) n_1 p_2^2\} \tag{2.20}$$

and with (2.20) and (2.15) we obtain

$$\frac{n_0\, \mathbb{V}\mathrm{ar}(\widehat{\gamma})}{\gamma^2} = \frac{1 - p_1}{p_1} + \frac{1 - p_2}{n_1\, p_1\, p_2} + \frac{(1 - n_1^{-1})\, \mathbb{V}\mathrm{ar}(p_2(S_1))}{p_1\, p_2^2} \tag{2.21}$$

The variance expression for using more than two stages will be very complex but will be built of terms involving variances of the success probability over the entrance distribution of the system state into thresholds similar to the two-stage case.

2.4.4 Efficiency

We will next describe the best choice for the parameters L_i and n_i. Since the choice of L_i uniquely determines p_i, we instead use p_i in our analysis; the proper value of L_i can be determined by a trial run to fix all the L_i close to the optimal p_i.

The way we will analyse the problem for more than two stages is by assuming that the $\hat{p}_i$ are independent, i.e. the success probabilities do not depend on the entrance state into that level, since the variance calculation would otherwise become too complex for analysis. A system for which this assumption holds is for instance the $M/M/1$ queueing model because of its memoryless property.

We see that (we use $\approx$ to symbolize the independence assumption)

$$\mathbb{V}\mathrm{ar}(\widehat{\gamma}) = \mathbb{V}\mathrm{ar}(\prod_{i=1}^{m} \hat{p}_i) \approx \sum_{i=1}^{m} \Big\{ \prod_{j=1, j\neq i}^{m} p_j^2 \Big\} \mathbb{V}\mathrm{ar}(\hat{p}_i)$$

using the first two terms of the following product expansion for independent random variables (X, Y) (i.e., neglecting the variance product)

$$\mathbb{V}\mathrm{ar}(X\,Y) = (\mathbb{E}Y)^2\, \mathbb{V}\mathrm{ar}(X) + (\mathbb{E}X)^2\, \mathbb{V}\mathrm{ar}(Y) + \mathbb{V}\mathrm{ar}(X)\, \mathbb{V}\mathrm{ar}(Y) \qquad (2.22)$$

Suppose the number of trials in stage i is *fixed* and given by r_i in order to arrive at an expression independent of $(R_i)_{i=1}^m$; this will entail

$$\frac{\mathbb{V}\mathrm{ar}(\widehat{\gamma})}{\gamma^2} \approx \sum_{i=1}^{m} \frac{1-p_i}{r_{i-1}p_i} \qquad (2.23)$$

Minimising the squared relative error of $\widehat{\gamma}$ with respect to the parameters p_i under the condition $\prod_{i=1}^m p_i = \gamma$ using the Lagrange method results in the optimal choice of $p_i = \gamma^{1/m} \quad, i = 1, 2, \ldots, m$.

The next question is how to choose r_i. Plugging the optimal p_i into the variance expression we get

$$\frac{\mathbb{V}\mathrm{ar}(\widehat{\gamma})}{\gamma^2} \approx (\gamma^{-1/m} - 1) \sum_{i=0}^{m-1} r_i^{-1}$$

Under the constraint that the sum of the number of sample paths $\sum_{i=0}^{m-1} r_i$ is fixed to a budget B, we see by a similar Lagrange expansion that we want $r_i = B/m = c, \quad i = 0, 1, \ldots, m-1$. In order to ensure that the real number of samples in stage i, given by $n_i R_i$, is close to the optimal r_i we have to fix

$$\begin{aligned} c &= r_i = \mathbb{E}\{\mathbb{E}(n_i R_i | \mathcal{F}_{i-1})\} = n_i \mathbb{E}\{n_{i-1} p_i \mathbb{E}(R_{i-1} | \mathcal{F}_{i-2})\} \qquad (2.24) \\ &= n_i p_i n_{i-1} \mathbb{E}\{\mathbb{E}(R_{i-1} | \mathcal{F}_{i-2})\} = n_i p_i r_{i-1} = n_i p_i c \quad, i = 1, \ldots, m-1 \end{aligned}$$

Equation (2.24) leads to the proper choice of

$$n_i := 1/p_i,\ i = 1, 2, \ldots, m-1. \quad (\text{"Balanced Growth"}) \qquad (2.25)$$

Incorporating this result into the relative error formula we obtain

$$\frac{\mathbb{V}\mathrm{ar}(\widehat{\gamma})}{\gamma^2} \approx (\gamma^{-1/m} - 1)mc^{-1} = (\gamma^{-1/m} - 1)m^2/B$$

and minimizing this with respect to m, the number of stages, we get

$$m^* = -\log(\gamma)/2 \tag{2.26}$$

and thus $p_i^* = e^{-2}$ and $n_i^* = e^2$ for $i > 0, n_0^* = B/m^* = -2B/\log(\gamma)$. Substituting these results in an optimal squared relative error of

$$\mathrm{RE}_{opt}^2 = (e^2 - 1)\frac{\log^2(\gamma)}{4B}$$

2.4.5 Complexity

For a probability $\gamma = \mathbb{P}(A(L))$ where we have an asymptotic exponential decay (see 2.2.1) with rate θ, we obtain the asymptotic relative error equality of

$$\mathrm{RE}_{opt}^2 = (e^2 - 1)\frac{O(L^2\theta^2)}{4B}$$

The budget needed to achieve a fixed relative error r is then at least

$$B^* = (e^2 - 1)\frac{O(L^2\theta^2)}{4r^2} \tag{2.27}$$

which is quadratic in the rarity parameter L.

We expect each of the samples in the budget to be approximately equal in terms of the simulation time necessary in the algorithm to complete the sample for the following reason.

For every sample in the budget we need to do a number of tasks:

- **a.** Restore the system state
- **b.** Simulate until a stopping criterion is encountered
- **c.** Save system state
- **d.** Update sample statistics

We assume that the cost for the sum of these tasks is constant. Note that in some cases items a, c or d may not apply; we therefore assume that the cost of these items is much smaller than item b and can be ignored. That the cost of item b is constant is easily verified by observing how the algorithm works: for a larger value of the rarity parameter L, there are simply more stages added to the simulation; the thresholds themselves are not moved and thus the sample paths for the simulations are independent of L, in other words $T(L) = kB(L)$, where $T(L)$ denotes the simulation time it takes to arrive at a fixed relative error r with the splitting method for a rarity parameter L, k is a conversion constant and $B(L)$ is the budget needed in terms of the total number of samples required to achieve the fixed relative error of r.The constant k will be depending on the speed of the computer and the implementation of the algorithm.

We will now give complexity properties of the splitting algorithm based on the discussion in section 2.2.1. We approximate the time the algorithm takes as the product of the number of paths B to simulate and the average time in the algorithm that each path takes. Since the simulation time in the algorithm can be assumed to be fixed for every path, from (2.27) we conclude that the splitting algorithm has a complexity of $O(L^2)$.

When we compare this $O(L^2)$ result with the results for the optimal Importance Sampling with bounded relative error, we see that the computational complexities of the methods are close: both methods are in the Polynomial family and thus asymptotically efficient.

2.4.6 Dependencies

The assumptions in Section 2.4.4, which resulted in equation (2.23), do not hold for most rare event simulation problems, since the variance from an important source is not considered : the dependence of R_{i+1} on R_i, or equivalently, the dependence of $\hat{p}_{i+1}$ on $\hat{p}_i$. The dependence occurs through the entrance distribution S_i; the larger the variation coefficient of $p_{i+1}(S_i)$ the more variance is incurred.

In equation (2.21) we already saw the effect of this extra factor on the variance of the estimator in the two-stage splitting set-up. In practical situations we will not know the entrance distribution S_i and thus cannot derive the optimal splitting parameters a priori.

Even if the entrance distribution is known, in order to optimize (2.21) we would have to know the entrance distribution for every possible intermediate threshold. This is obvious since the thresholds themselves are variables in the optimization, and changing the thresholds immediately changes the entrance distributions, and as such need to be incorporated into the optimization program. We look at a special case next.

Constant coefficient of variation

We suppose that the distribution of $p_2(S_1)$ has a constant coefficient of variation c for every threshold L ,i.e. $\mathbb{V}\mathrm{ar}(p_2(S_1)) = c\,p_2^2$ and obtain with $p_2 = \gamma/p_1$ the following minimisation problem (see (2.21)):

$$\begin{aligned} \min\{n_0\,\gamma^{-2}\,\mathbb{V}\mathrm{ar}(\widehat{\gamma})\} &= (1+c)(1-n_1^{-1})p_1^{-1} - 1 + n_1^{-1}\gamma^{-1} \\ \text{s.t.}\quad Bn_0^{-1} &= 1 + p_1 n_1 \end{aligned}$$

Now set the budget $B := 2\,n_0$; we want to spend an average of n_0 samples per stage. The problem is resolved by Lagrange multipliers as follows:

$$\begin{cases} p_1^* = \sqrt{(1+c)\,\gamma} \\ n_1^* = \sqrt{(1+c)^{-1}\gamma^{-1}}. \end{cases}$$

For the case $c = 0$, where there is no variance in the success probability, we arrive at the standard solution $p_1 = p_2 = \sqrt{\gamma}, n_1 = 1/\sqrt{\gamma}$ as we saw in Section 2.4.4. For $c > 0$, however, we see that $p_1 = (1+c)p_2$; p_1 is always greater than p_2 and n_1 is thus also decreased by a factor $\sqrt{1+c}$. For the efficiency we see that $\gamma^{-2}\mathbb{V}\mathrm{ar}(\widehat{\gamma}) = O(1/\sqrt{\gamma})$, which improves on the MC estimator but is still not asymptotically efficient. In order to achieve asymptotic efficiency we need to vary the number of thresholds, as we will see in the next section.

2.4.7 Multiple thresholds with dependence

We can now extend this model to multiple thresholds. Since the variance in the estimator is completely determined by the variance in the number of offspring of one starting sample, Y_1, (see (2.15)), we analyse this variance by considering the process of the number of hits at threshold k, (R_k) for the special case of one starting sample, i.e. $n_0 = R_0 = 1$. Note that for this special process we have $Y_1 = R_m$, i.e. R_m is the number of successes at the top threshold.

Suppose the variance of the success probability at threshold $k-1$ is fixed by the number v_k, i.e. $v_k = \mathbb{V}\mathrm{ar}(p_k(S_{k-1}))$. Note that necessarily $v_1 = 0$.

We now express the variance in R_k given R_{k-1} using (2.5) and (2.4) and introducing the vector notation $\boldsymbol{S}_{k-1} := (S_{k-1}^{(1)}, S_{k-1}^{(2)}, \ldots, S_{k-1}^{(R_{k-1})})$ and $\mathbb{E}_S, \mathbb{V}\mathrm{ar}_S$ as the expectation and variance respectively with respect to the stochastic vector

$\boldsymbol{S}_{k-1}$ for $k > 0$:

$$
\begin{aligned}
&\mathbb{V}\text{ar}(R_k|R_{k-1}) = \mathbb{E}_S\{\mathbb{V}\text{ar}(R_k \mid R_{k-1}, \boldsymbol{S}_{k-1})\} + \mathbb{V}\text{ar}_S\{\mathbb{E}(R_k \mid R_{k-1}, \boldsymbol{S}_{k-1})\} \\
&= \mathbb{E}_S\{\mathbb{V}\text{ar} \sum_{j=1}^{R_{k-1}} \text{Bin}(n_{k-1}, p_k(S_{k-1}^{(j)}))\} + \mathbb{V}\text{ar}_S\{\mathbb{E} \sum_{j=1}^{R_{k-1}} \text{Bin}(n_{k-1}, p_k(S_{k-1}^{(j)}))\} \\
&= \mathbb{E}_S\{ \sum_{j=1}^{R_{k-1}} n_{k-1}p_k(S_{k-1}^{(j)})(1 - p_k(S_{k-1}^{(j)}))\} + \mathbb{V}\text{ar}_S\{ \sum_{j=1}^{R_{k-1}} n_{k-1}p_k(S_{k-1}^{(j)})\} \\
&= R_{k-1}n_{k-1}\mathbb{E}\{p_k(S_{k-1})(1 - p_k(S_{k-1}))\} + R_{k-1}n_{k-1}^2\mathbb{V}\text{ar}\{p_k(S_{k-1})\} \\
&= R_{k-1}n_{k-1}\{p_k - p_k^2 - v_k\} + R_{k-1}n_{k-1}^2 v_k \\
&= R_{k-1}n_{k-1}(p_k(1 - p_k) + (n_{k-1} - 1)v_k) \qquad (2.28)
\end{aligned}
$$

For the expectation of R_k conditional on R_{k-1} we have the simple relation

$$
\begin{aligned}
\mathbb{E}(R_k \mid R_{k-1}) \quad &= \quad \mathbb{E}_S\{\mathbb{E}(R_k \mid R_{k-1}, S_{k-1}^{(1)}, S_{k-1}^{(2)}, \ldots, S_{k-1}^{(R_{k-1})})\} \\
&= \mathbb{E}_S\{\mathbb{E}(\sum_{j=1}^{R_{k-1}} \text{Bin}(n_{k-1}, p_k(S_{k-1}^{(j)})))\} \\
&= \mathbb{E}_S\{ \sum_{j=1}^{R_{k-1}} n_{k-1}p_k(S_{k-1}^{(j)})\} = n_{k-1}p_kR_{k-1} \qquad (2.29)
\end{aligned}
$$

Combining these results and taking the expectation with respect to R_{k-1} results in

$$
\begin{aligned}
\mathbb{E}(R_k^2) \quad &= \quad n_{k-1}^2p_k^2\mathbb{E}(R_{k-1}^2) + \{p_k(1 - p_k) + (n_{k-1} - 1)v_k\}n_{k-1}\mathbb{E}(R_{k-1}) \\
\mathbb{E}(R_k) \quad &= \quad n_{k-1}p_k\mathbb{E}(R_{k-1}) \qquad (2.30)
\end{aligned}
$$

The recurrence relation given in (2.30) can be solved giving (note that the empty product equals one, as commonly used)

$$
\mathbb{V}\text{ar}(R_k) = \sum_{i=1}^{k}\{p_i(1 - p_i) + (n_{i-1} - 1)v_i\}n_{i-1} \prod_{j=1}^{i-1} n_{j-1}p_j \prod_{j=i+1}^{k} n_{j-1}^2p_j^2
$$

We eliminate the parameters n_k from the result by posing the extra "balanced growth" restrictions $n_kp_k = 1$, $k = 1, 2, \ldots, m$:

$$
\begin{aligned}
\mathbb{V}\text{ar}(R_k) \quad &= \quad \sum_{i=1}^{k}\{p_i(1 - p_i) + (1 - p_{i-1})v_i/p_{i-1}\}p_k^2/p_i^2 \qquad (2.31) \\
&= \quad p_k^2\Big\{ \sum_{i=1}^{k} \frac{1 - p_i}{p_i} + \frac{(1 - p_{i-1})v_i}{p_i^2\, p_{i-1}}\Big\}
\end{aligned}
$$

and we define the coefficient of variance of R_m as

$$\mathrm{CV}(R_m) := \frac{\mathbb{V}\mathrm{ar}(R_m)}{(\mathbb{E}(R_m))^2} = \sum_{k=1}^{m} \frac{1-p_k}{p_k} + \frac{(1-p_{k-1})v_k}{p_k^2 p_{k-1}} = \sum_{k=1}^{m}(1+\frac{v_{k+1}}{p_{k+1}^2})\frac{1-p_k}{p_k}$$

where we define $v_{m+1} := 0$. Note that this coefficient of variance is now exactly equal to $\mathrm{RE}^2(\widehat{\gamma}) = \mathbb{V}\mathrm{ar}(\widehat{\gamma})/\gamma^2$ as in 2.15 except for the factor n_0. When the success probabilities themselves have a constant coefficient of variation, i.e. we have $v_k = c_k p_k^2$, this works out nicely giving the optimal parameters p_k^* which minimise the relative variation $\mathrm{CV}(R_m)$ under the condition $\prod_{k=1}^m p_k = \gamma$ which we again find by using Lagrange multipliers as

$$p_k^* = \frac{1+c_{k+1}}{(\prod_{i=1}^m 1+c_{i+1})^{1/m}}\gamma^{1/m} \quad ,k = 1,2,\ldots,m \tag{2.32}$$

In order to find the optimal number of stages m^* we consider the optimisation program with variable n describing the number of stage one samples since we want to contrast the variance reductions with the incurred effort increases. The expected work for one stage one path is equal to the expected amount of samples used for one path. The proper equation restricting the expected total work, $\mathbb{E}(W)$ say, to a fixed available budget, B say, is then

$$\mathbb{E}(W) = n\mathbb{E}(\sum_{k=0}^{m-1} n_k R_k) = n(1+\sum_{k=1}^{m-1} n_k p_k) = nm \leq B \tag{2.33}$$

The resulting optimisation program

$$\begin{aligned} \min \qquad & \mathrm{CV}(\widehat{\gamma}) = n^{-1}\Big\{m\gamma^{-1/m}\big(\prod_{k=2}^{m+1}(1+c_k)\big)^{1/m} - \sum_{k=2}^{m+1}(1+c_k)\Big\} \\ & \text{s.t. } n\,m \leq B \end{aligned} \tag{2.34}$$

is then solved using Lagrange multipliers for a constant coefficient of variation, i.e. $c_k = c\,, k = 2,3,\ldots,m$, giving the equations

$$\begin{cases} 0 & = \{2m + \log(\gamma(1+c))\}\{(\gamma(1+c))^{-1/m} - 1\} + c/(1-c) \\ n & = B/m \end{cases}$$

Note that for $c = 0$ we derive the same optimum $m^* = -\log(\gamma)/2$ as in (2.26) in Section 2.4.4 where we looked at the independent model. To get an idea of the dependence op the optimal m on c we present Figure 2.3 where the optimal number of stages is plotted against the coefficient of variation c for a target probability of $\gamma = e^{-20}$.

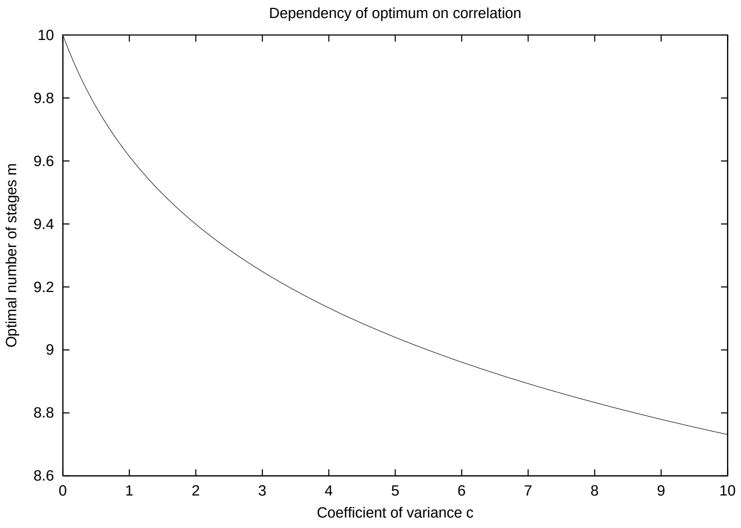

Figure 2.3: *Dependence effect on number of stages*

It is clear that for a larger variance factor c less stages are to be used. Interesting is that in spite of the intuition that more stages should be used for higher dependency systems in order to create more independence within the paths starting from one "root", the independence is simply achieved by using more starting samples n_0. We see that the effect of the variation coefficient on the number of stages is very small, moving from $c = 0$ (independence) to $c = 10$ (high dependence) we only see a shift in m of about 12%. A good rule of thumb is obtained by removing the last term in the optimisation program which simply leads us to set $m := -\log(\gamma\,(1+c))/2$. When we substitute this rule of thumb back into the efficiency equation we see that this guarantees that

$$\mathrm{RE}^2(\widehat{\gamma}) = \frac{(1+c)\,(e^2-1)\log^2(\gamma(1+c)) + 2c\log(\gamma(1+c))}{4B} \tag{2.35}$$

and we obviously still have the quadratic complexity for the asymptotic exponential decay model used in Section 2.4.5.

For a extremely high values of c, the variance coefficient, we see that the optimal number of stages m can drop to a value of one. E.g., when setting $c = e^{-2}\gamma^{-1} - 1$ we see that the rule of thumb gives us $m := 1$. In general, any $c = O(\gamma^{-1})$ gives almost unity for the optimal number of thresholds m. This will mean that there is no gain in using the splitting method, since it behaves as a Monte Carlo estimator for $m = 1$. Intuitively this is easily explained: when it is as likely to hit the next threshold as it is to hit the rare event set, i.e. $p_k \approx \gamma$, there will be no gain in using the splitting method. This translates into the model by noting that $v_k = O(\gamma)$, and we simply obtain $c_k = O(\gamma^{-1})$, for which the splitting method does not work.

We therefore must presuppose that the variance coefficient c is not of order $O(\gamma^{-1})$; when $c = O((-\log(\gamma))^n)$ the complexity of the algorithm remains polynomial with $B = O((-\log(\gamma))^{2+n})$ as is clear from (2.35).

Note that a specific model was chosen for the dependency structure in order to solve the problem explicitly; however many models will fit this pattern asymptotically.

Example 2.4 (Aging model) We give an example of a model for which we cannot do better than Monte Carlo. Also, we demonstrate an intuitive pilot run algorithm to derive the optimal number of stages, which will be refined and formalised in Chapter 3. Suppose the stochastic process $(X_t, t \geq 0)$ is given by the following

$$X_t = \begin{cases} t & t \leq T \\ 0 & t > T \end{cases}$$

for some non-negative real-valued random variable T, which is drawn at time $t = 0$. The system describes the age of a particle until its death time T. The objective is to estimate the probability that the event $\{X_t > L\}$ occurs before $(X_t, t \geq 0)$ returns to zero. Note that we are effectively estimating $\gamma = \mathbb{P}(T \geq L)$. The process clearly complies with the requirements necessary to perform splitting simulation using the system state (t, T). Since the future of the process is fixed at the starting time, as there are no random variables after $t = 0$, there will be no point in splitting since all split paths will be guaranteed to meet the same fate. We know that the splitting method will have to do a basic Monte Carlo simulation in order to be at its most efficient.

Suppose we have two stages of simulation, and thus one intermediate threshold L_1. Clearly the entrance distribution at this threshold is given by the set $\{t = L_1, T \geq L_1\}$. We now have

$$\begin{aligned} \mathbb{V}\mathrm{ar}(p_2(S_1)) &= \mathbb{V}\mathrm{ar}(I\{T \geq L\} \,|\, T \geq L_1) \\ &= \mathbb{E}(I^2\{T \geq L\} \,|\, T \geq L_1) - (\mathbb{E}(I\{T \geq L\} \,|\, T \geq L_1))^2 \\ &= p_2(1 - p_2) \end{aligned}$$

and thus $c_2 = (1 - p_2)/p_2$. Substituting this into the optimal success probability equation (2.32) we find the following solution

$$\begin{aligned} p_1^* &= \frac{p_2^{-1}}{p_2^{-1/2}}\sqrt{p_1 p_2} = \sqrt{p_1} \\ p_2^* &= \frac{1}{p_2^{-1/2}}\sqrt{p_1 p_2} = p_2\sqrt{p_1} \end{aligned}$$

This clearly leads to the optimal solution $(p_1 = 1, p_2 = \gamma)$, and thus $L_1 = 0$ and there should not be a first stage at all.

We confirm this with the formulation for the optimal number of stages under dependency (2.34): Find $m \in \mathbb{N}$ that minimizes

$$\mathrm{CV}(\widehat{\gamma}, m) = m^2\gamma^{-1/m}(\prod_{k=2}^{m+1}(1+c_k))^{1/m} - m\sum_{k=2}^{m+1}(1+c_k). \tag{2.36}$$

In our example we have $\mathrm{CV}(\widehat{\gamma}, 1) = \gamma^{-1} - 1$ and $\mathrm{CV}(\widehat{\gamma}, 2) = 4\sqrt{p_1}\gamma^{-1} - 2(1 + p_1\gamma^{-1}) = -2(\sqrt{p_1} - 1)^2\gamma^{-1} + 2(\gamma^{-1} - 1)$. We cannot conclude directly that using one stage is more efficient in general. However after one iteration of updating (p_1, p_2) according to the optimal values $(p_1^*, p_2^*) := (\sqrt{p_1}, \sqrt{p_1}p_2)$ the CV's become equal, and after two iterations we indeed obtain that the use of one stage is optimal because of its lower CV. Note that this iteration is necessary since the parameter c_2 is a variable and depends on the parameters (p_1, p_2); in the minimisation algorithm (2.36) we had to assume it is a constant, and therefore cannot draw the correct conclusion directly.

2.4.8 Convergence

We are now ready to formulate the restrictions the system must obey in order to obtain the asymptotic quadratic complexity for exponential decay models. For every threshold k we need the expectations $\mathbb{E}(p_k^2(S_{k-1}))$ and $\mathbb{E}(p_k(S_{k-1}))$ to exist, otherwise the variance of the estimator becomes infinite. This requirement is fulfilled when

$$\mathbb{E}(p_k^2(S_{k-1})) < \infty \quad, k = 1, 2, \ldots$$

For the asymptotic case with infinitely many thresholds there are some more requirements to be made in order to keep the variance buildup bounded.

$$\limsup_{k\to\infty} \mathbb{E}(p_k^2(S_{k-1})) < \infty \tag{2.37}$$

We have now arrived at the model discussed in the previous section 2.4.6, i.e. we have the variances v_k all finite. When we further assume that not only the decay has a limiting behaviour but that also the variance sequence has a limit

$$\limsup_{k\to\infty} v_k/p_k^2 = c \tag{2.38}$$

for some asymptotic coefficient of variance c, we can asymptotically approximate the system behaviour with the model with fixed variance coefficient c from the previous section, for which the quadratic complexity was already shown. We now find the requirements necessary to obtain a quadratic complexity with the splitting method involving unknown dependencies. In other words, we assume that we can find the parameters $(p_k, n_k, k = 1, 2, \ldots, m)$ approximately (via a pre-run, say), but do not know the covariance parameters $(v_k, k = 1, 2, \ldots, m)$. Using $c_k := v_k/p_k^2$, the asymptotic covariance parameter $c = \lim_{k\to\infty} c_k$ is assumed to be finite. We are interested in the requirements necessary to keep the complexity quadratic with this method.

Our objective is to show that the coefficient of variance of the estimator $\widehat{\gamma}$ needs to be of complexity order $O(m)$; from (2.33) we obtain the expected work for all the offspring of a stage one sample $\mathbb{E}(W) = m$; by combining we simply obtain by using the same argument as in Section 2.4.7 that the total number of samples necessary to obtain a fixed accuracy is of complexity $O(m^2) = O((-\log(\gamma))^2)$, which is the optimal splitting method complexity.

For quadratic complexity, we need the following requirement involving the coefficient of variance of the estimator $\widehat{\gamma}$,

$$\lim_{m\to\infty} \frac{1}{m} \sum_{k=1}^{m} (1 + c_{k+1}) \frac{1 - p_k}{p_k} < \infty \quad ,$$

which typically holds when the sequence $\{c_k\}_{k=1}^{\infty}$ converges fast enough and the success probability sequence $\{p_k\}_{k=1}^{\infty}$ remains positive, e.g. when $c_k = c + O(1/k^2)$ we see that

$$\begin{aligned} \lim_{L\to\infty} n_0^2 \operatorname{CV}(\widehat{\gamma}) &= \lim_{m\to\infty} \sum_{k=1}^{m} (1 + c_{k+1}) \frac{1 - p_k}{p_k} \leq \\ &\leq (1 + c) \lim_{m\to\infty} \sum_{k=1}^{m} \frac{1 - p_k}{p_k} + \lim_{m\to\infty} \sum_{k=1}^{m} O(k^{-2}) \frac{1 - p_k}{p_k}. \end{aligned}$$

Here the first term is clearly $O(m)$ when the individual terms are bounded by $p_k \geq p > 0$ for some p for all k, and the second term is even finite when the boundedness of the p_k holds.

When there is no upper limit for the sequence (v_k/p_k^2) it is clear that the coefficient of variance of the success probability diverges and that there cannot be an asymptotic efficient estimator. A sufficient condition for the existence of the limsup in (2.38) is the existence of a positive limiting success probability

$$\liminf_{k\to\infty} p_k = p > 0$$

since then we have $\limsup_{k\to\infty} v_k/p_k^2 = 1/p^2 \limsup_{k\to\infty} v_k$ and the limsup of the sequence v_k is finite by (2.37).

2.5 Related Methods

The splitting method has some interesting parallels with other known theories and methods. We will describe the analogies in order to enhance our understanding of the inner workings of the method.

Branching process

The series of stochastic variables describing the number of hits at each threshold, $\{R_i\}_{i=1}^{m-1}$, is a martingale [Mét82] since by (2.7)

$$\mathbb{E}(R_{i+1}|\mathcal{F}_i) = \mathbb{E}(R_{i+1}|R_i) = n_i p_{i+1} R_i$$

and we have a finite expectation for $(R_i) \quad i = 1, 2, \ldots, m-1$, since all parameters B, n_i and p_i are required to be finite.

In the case of under-sampling, $n_i < 1/p_{i+1}\,, i = 1, 2, \ldots, m-1$, $(R_i)_{i=1}^m$ is a sub-martingale, in the oversampling case, $n_i > 1/p_{i+1}\,, i = 1, 2, \ldots, m-1$, $(R_i)_{i=1}^m$ is a super-martingale and in the optimal sampling case, $n_i = 1/p_{i+1}\,, i = 1, 2, \ldots, m-1$, we have a real martingale. Note that this can only be achieved with systems in which the state-space E and the success probability function from a given point in E are continuous. Using classic martingale theory we can even derive ruin probabilities $\mathbb{P}(\lim_{n\to\infty}\{R_n\} = 0)$, for the asymptotic case $L \to \infty$, in this context the probability of a “die-out” of the simulation paths for an infinitely rare event probability.

We used a balanced growth equation ($n_k p_{k+1} = 1$) different from the growth equation ($n_k p_k = 1$) as used in Section 2.4.4 and 2.4.6. We clarify the difference in these choices. The new equations $n_k p_{k+1} = 1, k > 1$ guarantee that $\mathbb{E}(R_k)$ is constant and equal to $n_0 p_1$ for $k > 0$; the growth conditions $n_k p_k = 1, k > 0$ guarantee that $\mathbb{E}(R_k) = n_0 p_k$, i.e. the expected number of successes is different for different stages but the expected number of samples to take in stage k, $n_k R_k$, is a constant n_0. The same martingale discussion would then hold for the process $(\tilde{R}_k)_{k=0}^{m-1}$ denoting the number of starting samples in stage k with the equations $n_k p_k = 1$. Note that then simply $\tilde{R}_k := n_k R_k$. Asymptotically the two conditions are equal, under the condition $\lim_{k\to\infty} p_k = p$ for some constant p, when using a large number of stages m.

A similar view of the process $(R_n)_{n=1}^{m-1}$ comes from the field of branching processes [AN71]. Suppose the number of offspring from one stage i sample is given by the probability function $\{q_k; k = 0, 1, 2, \dots\}$ satisfying $q_k \geq 0, \sum_k q_k = 1$ giving exactly the probability that k offspring are generated from that sample. In our case we have of course $q_k := \mathbb{P}(\text{Bin}(n_i, p_{i+1}) = k)$. In the case where the success probability does not depend on the entrance state or the stage and we have $p_i = p \quad, i = 1, 2, \dots, m$, the number of re-trials $n_i = n$ is equal for all stages and $(R_n, n = 0, 1, 2, \dots)$ is a Markov chain with transition probabilities

$$P(i,j) = \mathbb{P}(R_{n+1} = j | R_n = i) = \begin{cases} q_j^{*i} & , i \geq 1, j \geq 0 \\ \delta_{0j} & , i = 0, j \geq 0 \end{cases}$$

with δ_{ij} the Kronecker delta and $\{q_k^{*i}, k = 0, 1, 2, \dots\}$ the i-fold convolution of parameters $\{q_k, k = 0, 1, 2, \dots\}$. (Recall that the i-fold convolution of q_k is the probability function of the sum of i independent variables each with probability function q_k.) This process is exactly a Galton-Watson process, see e.g. [AN71]. The process (R_n) is also investigated using Galton-Watson results in [GHSZ96b].

Bootstrapping

Bootstrapping [Efr82] is a widely applicable method to derive robust estimates based on re-sampling observations. We in fact implicitly use this method as we clarify next.

Every stage in the splitting process can be seen as an estimation procedure for the success probability p_{i+1} in that stage, $i + 1$ say. This probability is conditional; it depends on the distribution of the system state upon entering that stage. We will denote this entrance distribution by S_i. All the information about S_i we have is the sample entrance states given as $(S_i^{(1)}, S_i^{(2)}, \dots, S_i^{(R_i)})$ by the simulation procedure from stage i. In other words, we are trying to estimate

$$p_{i+1} = \mathbb{E}(I_{i+1}(S_i))$$

where I_{i+1} is the indicator function that a path starting at threshold i from entrance distribution S_i reaches threshold $i + 1$ before stopping. For every observation of S_i, s_i say, the random variable I_{i+1} has a binomial distribution with parameter $p_i(s_i)$; we can thus estimate $p_{i+1}(s_i)$ nicely using r replications with $\hat{p}_i(s_i) := r^{-1} \sum_{j=1}^{r} I_{i+1}^{(j)}(s_i)$.

Bootstrapping [Ded90] gives us a method for estimating p_i: draw n samples from an estimated distribution function of S_i, say we draw the samples $s_i^1, \ldots s_i^n$, and estimate p_{i+1} using

$$\hat{p}_{i+1} = n^{-1} \sum_{j=1}^{n} I_{i+1}^{(j)}(s_i^j).$$

In the standard bootstrapping method, the estimated distribution function is the empirical distribution function, however this is not required in the generalized bootstrap and indeed the splitting method prescribes the choice of $s_i^j := S_i^{(\lceil R_i(j/n) \rceil)}$, with $n = n_i R_i$ making $\hat{p}_{i+1} := R_i^{-1} \sum_{j=1}^{R_i} \hat{p}_{i+1}(S_i^{(j)})$. Different methods for the subproblem of drawing the entrance state will be investigated in Chapters 3 and 4. As in bootstrapping, the estimate for p_{i+1} can be extended with a confidence interval based on the quantiles of the empirical distribution of $p_{i+1}(S_i)$.

Importance Sampling

We now show two similarities between I.S. and the splitting method; note that it is not the objective to show that either of the methods is a special case of the other because this is not the case.

Stage-wise We can view the splitting process as an adaptive I.S. implementation: in m steps we estimate the rare event probability as follows:

0. Set stage $k := 1$
1. Set the rare event set $A := D_k$
2. Set probability measure $\mathbb{P}_k := \mathbb{P}(\cdot | D_{k-1})$.
3. Estimate $\mathbb{P}(A)$ using r_k samples with $\widehat{\mathbb{P}}(A) := r_k^{-1} \sum_{j=1}^{r_k} 1_A(j) \widehat{L}(j,k)$
4. Set $k := k+1$; return to step 1 if $k \leq m$.

This change of probability measure involved in the splitting method is common in I.S.; after every stage k the probability measure becomes $\mathbb{P}_k := \mathbb{P}(\cdot | D_{k-1})$ which makes the next set D_k a likely event to happen since by definition $\mathbb{P}_k(D_k) = p_k$, and p_k is relatively large. The correct likelihood ratio of the probability change for path j in stage k is then

$$L(j,k) := \frac{\mathbb{P}(D_k)}{\mathbb{P}_k(D_k)} = \mathbb{P}(D_{k-1}) = \prod_{i=1}^{k-1} p_i,$$

Since we do not know the actual L because the probabilities are unknown, we estimate it by $\widehat{L}(j,k) := \widehat{\mathbb{P}}(D_{k-1}) = \prod_{i=1}^{k-1} \hat{p}_i$; which is exactly the estimate we obtained in the previous stage. The difference is that here we here do not know the actual likelihood of the new paths, and that the likelihoods are equal for every sample in the same stage (which is not common in I.S.).

Note however that no twisting or tilting of probability distributions takes place as in I.S., the probability measures are changed but simply become conditional ones.

Another interpretation of this link with I.S. occurs in Haraszti [HT99] where the change of measure is perceived as a Direct Probability Redistribution (DPR).

Path-wise We can also view the splitting simulation as an estimation process involving $n := \prod_{k=0}^{m-1} n_k$ possible samples of the last stage of splitting. The word "possible" is used to indicate that some samples will not be available because they died out (i.e. failed to meet the next threshold) in one of the previous (before last) stages.

We attach a weight to every path, equal to the probability of seeing that path occur under normal circumstances, i.e. the original density. For each path in stage one, the weight is necessarily equal to one. When a simulation path reaches threshold k, it is divided into n_k parts, which each receive a n_k^{-1}-fraction of the original weight, since under normal circumstances it would take n_k as many starting samples to see this many paths in the area associated with the next stage. Every last-stage path has hit all the intermediate thresholds, making its weight equal to $L := \prod_{i=1}^{m-1} n_i^{-1}$.

Denote the collection of sample paths as $\{\omega_1, \omega_2, \ldots, \omega_n\}$. We can now express the rare event probability estimator from (2.12) using the introduced weight function as follows

$$\widehat{\gamma} = n_0^{-1} \sum_{i=1}^{n} I_A(\omega_i) L(\omega_i)$$

where I_A is the indicator function of hitting set A. Note that this formulation of the estimator is equal to the classic I.S. estimator, see e.g. Asmussen and Rubinstein [AR95].

Again, the comparison in artificial insofar that the average is not natural: we are summing n elements and only dividing by n_0; furthermore the likelihood factor L is in fact a constant, i.e. does not depend on ω_i. Although no change of density occurs in this formulation, the fact that the weight function L is in fact a likelihood ratio is the main analogy with I.S. for the path-wise look at the splitting method.

Bayes introduces the notion of the weights as importance factors in [Bay72].

2.6 Conclusions

The splitting algorithm is robust in the sense that it is applicable to a broad class of problems, and in many cases delivers a substantial speed-up over the Monte Carlo method. A well-tailored splitting method can be shown to need a simulation budget quadratic in the rarity parameter of the problem, making it asymptotically efficient. Note that an un-tailored implementation does not guarantee asymptotic efficiency: choosing the thresholds wrong will reduce the efficiency to the Monte Carlo level.

Essential to the method is that the process X moves slowly towards the rare event. Also, there must be a way to determine how 'important' the current state is (i.e. how close we are to the rare event) so that we can decide when to split the current path. When the process moves into the rare event using only a few events, or the probability of reaching the rare event does not vary much in the state space, the method is not useful since we cannot place useful thresholds and the efficiency gain over Monte Carlo will not be large, and there certainly is no asymptotic optimality.

Comparing the splitting method to I.S., we see that although I.S. can have slightly better complexity properties the splitting method does not suffer from the main difficulty of I.S., which is obtaining the optimal changes of measure and the possibility of infinite variance when choosing the wrong one. Also, the splitting method is more intuitive than I.S.

Chapter 3

Implementation Issues

3.1 Introduction

One of the main questions for using the splitting methods in practice is to decide on an implementation. Since there are so many parameters and choices to be set before doing an actual simulation, this is not an easy issue.

In this chapter we investigate and compare, both theoretically and empirically, different implementations of the splitting method.

We describe and analyse all the known existing methods and complement these by introducing new techniques. We find that the original splitting implementation, in which each path is split into a fixed number of copies, may not be the most efficient one in all cases.

First we look at issues stemming from the discrete nature of a large number of parameters in Section 3.2. In the next Section 3.3 we describe how to derive the thresholds, and prove that the I.F. should be chosen such that it is impossible to cross multiple thresholds at once. Next we look at a way to improve the efficiency of the splitting method called truncation in Section 3.4. We describe two basic implementations of the algorithm in detail in Section 3.5, we describe a variation on the run-time behaviour of the standard splitting method called RESTART in Section 3.6 and give the proper implementation of the algorithm in the case that there exist dependencies between the samples in Section 3.7. Finally, we implement all the aforementioned methods and compare them in Section 3.8 by means of numerical results of simulations.

3.2 Discrete parameters

In Section 2 we introduced the splitting method using a stochastic process (X_t) with a continuous time parameter as the leading example. However, this is not a necessary prerequisite for the splitting method; the reader can easily verify that the results also hold for processes $(X_n)_{n=0}^{\infty}$ with a discrete-time parameter. However, a number of issues that stem from the discrete nature of a number of parameters, must be dealt with.

3.2.1 Number of Stages

The optimal number of stages, $m^* = -\log(\gamma)/2$ derived in Section 2.4.4 for the case where the process (R_i) is uncorrelated, cannot be determined in advance and it is not a discrete value, i.e. $m^* \notin \mathbb{N}$. A trial run, typically a splitting simulation with an arbitrary (non-optimal) number of stages, can determine an estimate for γ, $\widehat{\gamma}$ say, which is used to optimize the number of stages m as done in Section 2.4.4:

$$\begin{aligned} \min \qquad & \gamma^{-2}\mathbb{V}\mathrm{ar}(\widehat{\gamma}) = (\gamma^{-1/m} - 1)m^2/B \\ \text{s.t.} \quad & m \in \mathbb{N} \end{aligned} \tag{3.1}$$

Relaxing the condition we obtained $\widehat{m}^* := -\log(\widehat{\gamma})/2$. A rule of thumb is to round off this obtained estimate to arrive at an integer number by setting $m^* = [\widehat{m}^*]$. Note that since the function is convex the minimum m^* of the original problem (3.1) is unique and easy to find. We expect little gain from the more sophisticated procedure, certainly for realistic problems where m is large.

3.2.2 Discrete State Space

Using a discrete state space will put a further limitation on the possibilities of choosing the thresholds. This in turn makes it impossible to fix the success probabilities to the optimal values of $p_i = e^{-2}$. Since the state space is discrete, the success probability function $g(x, y) = \mathbb{P}(\exists n \in \mathbb{N} | Z_n \geq x, Z_0 = y)$, equal to the probability of reaching level x given that level y has been reached, will not be continuous as we would expect in the continuous state space model. We arrive at the optimization problem from (2.23)

$$\begin{aligned} \min \qquad & \gamma^{-2}\mathbb{V}\mathrm{ar}(\widehat{\gamma}) \approx \sum_{i=1}^{m} \frac{1 - g(L_i, L_{i-1})}{r_{i-1} g(L_i, L_{i-1})} \\ \text{s.t.} \quad & L_i \in \mathbb{R} \quad , i = 1, 2, \ldots, m \\ & L_i > L_{i-1} \quad , i = 1, 2, \ldots, m \\ & m \in \mathbb{N} \end{aligned} \tag{3.2}$$

We saw in the previous Section 2.4.4 that for continuous g this is solved by choosing the L_i such that $g(L_i, L_{i-1}) = e^{-2} \quad , i = 1, 2, \ldots, m$. Using a discrete state space will make the function g a discontinuous step function and thus make the optimization problem hard since it's a non-linear integer programming problem. Although the decision variables $(L_i, i = 1, 2, \ldots, m)$ are all real-valued, the step function $g(x, y)$ makes the problem equivalent to choosing the $L_i, i = 1, 2, \ldots, m$ from the countable set of jump points of $g(x, y)$, and thus an integer programming model. The non-linearity of the target function makes it impossible to solve the model with dynamic programming algorithms, since we cannot optimize L_i recursively depending on the previous thresholds. Since also the function $g(x, y)$ is usually unknown and must be estimated in order to perform the optimization of the $(L_i, i = 1, 2, \ldots, m)$, we cannot solve the problem explicitly.

For a rule of thumb we turn to the relaxed problem with continuous g and assume the function $g(\cdot, \cdot)$ is linear and we can thus use a dynamic programming solution, which is to choose $L_i, i = 1, 2, \ldots, m$ such that $g(L_i, L_{i-1}) \approx e^{-2}$ recursively.

3.2.3 Discrete Number of Trials

Accompanying this problem is the choice of n_i. Even for ideally defined thresholds, the number of re-trials in stage i, n_i, must be a discrete value, whereas from (2.25) we see that the optimal $n_i := 1/p_i$ is a real, possibly non-integer, number. A possible solution for this problem is to use a randomized n_i, N_i as proposed in [GHSZ96a].

Say the optimal n_i would be n_i^*, non-integer. Draw N_i for every hit to threshold i from the set $\{\lfloor n_i^* \rfloor, \lceil n_i^* \rceil\}$, in such a way that $\mathbb{E}(N_i) = n_i^*$. This would alleviate the discrete number of re-trials problem, but the gain is minimal compared to the non-random case $N_i := [n_i^*]$, as reported in [GHSZ96a]. In [VAVA99] it is also not recommended to use randomization of the number of re-trials.

3.3 Thresholds

In Chapter 2 we introduced the thresholds, the point where we split the simulation paths into multiple sub-paths, as fixed. In an implementation one will have to choose both the level-determining Importance Function and the thresholds where the splitting will occur, $(L_1, L_2, \ldots, L_m)$. Last but not least the number of splits at each threshold $(n_1, n_2, \ldots, n_m)$ has to be determined.

3.3.1 Choosing the thresholds

When it is unclear how to choose the thresholds L_i, but the desired p_i is known, it is best to estimate them doing a trial run from the previous threshold. The problem can be formalised as

$$L_{i+1} = \inf\{x \in \mathbb{R} | \mathbb{P}(\exists n = 1, 2, \ldots, Z_n := f(X_n) \geq x, X_0 \sim S_i) \leq p_{i+1}\}$$

Using the stochastic variable $M_i := \sup\{Z_n, n = 1, 2, \ldots, X_0 \sim S_i\}$ denoting the maximum level obtained in a simulation of (Z_n) when starting at threshold L_i, it is clear that L_{i+1} is to be chosen as the $(1-p_{i+1})$-quantile of the distribution function of M_i. We assume that the distribution of variable M_i is unknown, and we thus have to revert to non-parametric methods to obtain an estimate for L_{i+1}. We will now present the non-parametric quantile estimator based on order statistics to estimate L_{i+1}.

Using n samples starting at random from threshold L_i, obtain the maximum level of the Importance Function reached with each sample j as $W_j, j = 1, \ldots, n$. Using the order statistics $W_{[j]}$, for which we have $W_{[1]} \leq W_{[2]} \leq \ldots \leq W_{[n]}$, the best choice for L_{i+1} is to set $L_{i+1} := W_{[n(1-p_{i+1})]}$ which is an unbiased estimator for the $1 - p_{i+1}$ -th quantile of the distribution of M_i, and we thus have an estimated success probability for this stage of p_{i+1}.

When $n(1 - p_{i+1}) \notin \mathbb{N}$, set $d = \lfloor n(1 - p_{i+1}) \rfloor$ as the integer part and $f = n(1 - p_{i+1}) - d$ as the remaining fraction and we propose the interpolation estimator $L_{i+1} := (1-f)W_{[d]} + fW_{[d+1]}$. See e.g. [Ser80] p. 74 for the statistical properties of the non-parametric quantile estimator.

This newly defined threshold can then be used to find the next thresholds recursively: first the same stage has to be redone but now with this newly found threshold to obtain samples of the system state upon crossing the new threshold; when they have been obtained a new threshold estimation stage can be entered from the new threshold to obtain the next threshold.

Care has to be taken that enough samples are used in determining the thresholds, otherwise the thresholds may become biased and force the real simulation into the wrong area. A good rule of thumb is to devote 10% of the simulation effort to a trial run which will determine all the thresholds.

3.3.2 Crossing multiple thresholds

We will describe an extension of the splitting model in which the restriction that only one threshold can be crossed at a time is dropped. The possibility of reaching the rare event from any threshold, which is a subset of the possibilities in crossing multiple thresholds at a time was first described in [VA98].

We have not made any assumptions about the behaviour of the level variable Z of a sample path in time. In typical cases, such as a flow model it will be continuous, but it might be possible that the process Z is not skip-free and changes from stage i ($L_i \leq Z_n < L_{i+1}$) to stage j ($L_j \leq Z_{n+1} < L_{j+1}$), with $j > i+1$ and (since we are looking at discrete event simulation we focus on the discrete time process (Z_n)) the states Z_n and Z_{n+1} belong to successive events. The correction for this is then to treat this sample as if it crossed all the intermediate thresholds at once and as such it will need $\prod_{k=i+1}^{j} n_k$ re-trials from the corresponding system state X_{n+1}.

Equivalently, if in the previous discussion we have a sample path going from any threshold into the rare event set without crossing the uppermost thresholds first, we can capture this as a multiple threshold crossing event when we regard the rare event set as the stage with index $j = m+1$.

Note that this violates the condition $D_i \subset D_{i-1}, i = 1, 2, \ldots, m$. Since it did not split after stage i its weight in the estimator should be equal to $\prod_{k=1}^{i} n_k^{-1}$, a factor $\prod_{k=i+1}^{m} n_k$ higher than the "normal" samples entering the rare event set via all the stages.

An example of such multiple thresholds crossing behaviour is found in the next Figure 3.1. Clearly we can reach set A from any threshold L_i with $i \geq 1$, making the splitting inefficient. A more "natural" choice for the I.F. would be $f(x_1, x_2) = x_2 - x_1$, since $A = \{x_2 \geq x_1 + 1\}$.

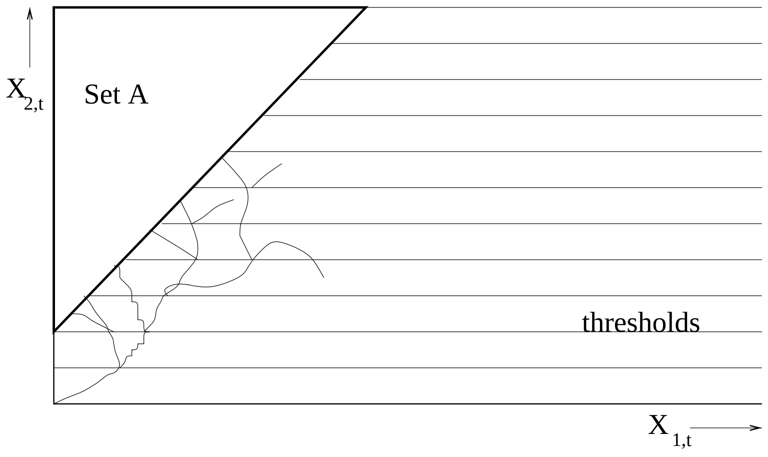

Figure 3.1: *Crossing multiple thresholds*

We will now show that the efficiency of the splitting method is bad when it is possible to reach the rare event set directly from any threshold without crossing all the higher thresholds. The algorithm then works as if we are estimating all these multi-crossing probabilities separately and summing these. We demonstrate this by introducing the sets $A_k, k = 1, \dots, m$, which denote the events that set A is hit directly from a state in stage k, i.e. for which $L_k \leq Z_n < L_{k+1}$. Since these sets are disjunct ($A_i \cap A_j = \emptyset, i \neq j$) and together fill up to A ($\cup_{k=1}^m A_k = A$) we have

$$\mathbb{P}(A) = \sum_{k=1}^{m} \mathbb{P}(A_k)$$

by Kolmogorov's probability axiom. Note that further $A_k \subset D_{k-1}, k = 1, \dots, m$ and thus

$$\gamma = \sum_{k=1}^{m} \mathbb{P}(A_k | D_{k-1}) \mathbb{P}(D_{k-1})$$

by conditional probability. We thus propose the estimator $\widehat{\gamma}$ of γ as

$$\widehat{\gamma} = \sum_{k=1}^{m} \widehat{\mathbb{P}}(A_k | D_{k-1}) \widehat{\mathbb{P}}(D_{k-1}).$$

Complexity

The complexity of the algorithm, as defined in Section 2.2.1 will be dominated by the contribution of the lowest threshold from which it is possible to hit the rare event set, since the estimator $\widehat{\mathbb{P}}(A_k | D_{k-1})$ has the worst complexity properties for the lowest k. This is easily verified by noting that since all $A_k \subset A, k = 1, 2, \dots, m$ clearly $\mathbb{P}(A_k) \leq \gamma, k = 1, 2, \dots, m$ and as such $\mathbb{P}(A_k | D_{k-1}) \leq \gamma \mathbb{P}(D_{k-1}) \leq \prod_{i=1}^{k-1} p_i^{-1} \gamma, k = 1, 2, \dots, m$. We approximate the variance in the estimator as

$$\mathbb{V}\text{ar}(\widehat{\gamma}) = \mathbb{V}\text{ar}\{\sum_{k=1}^{m} \widehat{\mathbb{P}}(A_k | D_{k-1}) \widehat{\mathbb{P}}(D_{k-1})\} \leq \sum_{k=1}^{m} (\mathbb{P}(D_{k-1}))^2 \mathbb{V}\text{ar}(\widehat{\mathbb{P}}(A_k | D_{k-1}))$$

Fixing the relative error in the estimator to r we obtain

$$\frac{\mathbb{V}\text{ar}(\widehat{\gamma})}{\gamma^2} \approx \sum_{k=1}^{m} \frac{1}{r_k \mathbb{P}(A_k | D_{k-1})} \leq r^2$$

Suppose we fix the relative contribution of stage k into the variance as α_k with $\alpha_k \geq 0, \sum_{k=1}^m \alpha_k = 1$ we see that we need a number of samples in stage k of

$$r_k = \frac{\alpha_k}{r^2 \mathbb{P}(A_k | D_{k-1})} \geq \frac{\alpha_k \prod_{i=1}^{k-1} p_i}{\gamma r^2}.$$

The sum of necessary samples is then dominated by the lowest stage k since the product in the numerator is higher for lower k. It is clear that the number of necessary samples is $O(\gamma^{-1})$ and thus also the complexity by our assumption that all samples need equal simulation time; making the splitting method not efficient by the definition in Section 2.2.1.

We see this effect intuitively when it is possible to reach the rare event without ever crossing a threshold: the complexity of the algorithm in this case is equal to the Monte Carlo method since the estimation of $\mathbb{P}(A_1|D_0) = \mathbb{P}(A_1)$ will be done using Monte Carlo.

It is essential that the importance function and the thresholds are chosen in a way that a path leading to the rare event set must cross all the thresholds, i.e. the $\{D_k\}$ subset relation condition holds.

3.3.3 The Importance Function

When using a one-dimensional state space it is natural to choose the importance function as a properly scaled state space. In a multi-dimensional state-space it is usually non-optimal to choose a scaled version of one of the state space variables as the importance function, as was shown in [GHSZ98] for the tandem queue. In Chapter 5 we will devote an entire chapter on how to find the I.F. for these cases; the main result is that the optimal importance function must be chosen in accordance with the paths leading to the rare event set.

When information about the successful paths is incorporated into the importance function the simulation efficiency is restored.

In [VAVA99] an I.F. linear in the state space variables is presented which works well for the tandem queue case shown to be non-optimal for the single-variable I.F. in [GHSZ98].

3.4 Truncation

When using a large number of stages, many sample paths will be generated in the high stages, and as such have large stopping times in the typical case where the stopping time is the time of return to the starting set B. A lot of simulation time will be spent on the simulation of these paths until they stop, which maybe quite inefficient when they are sure to have a small contribution to the final estimate when they have dropped downwards several stages. The truncation technique consists of stopping the simulation when the sample path drops a fixed number of thresholds below the threshold where it originated from a splitting path. The first reference to the truncation method is found in [GHSZ96b]. A graphical illustration of the truncation method is found in Figure 3.2 where a truncation cut-off depth of one is used.

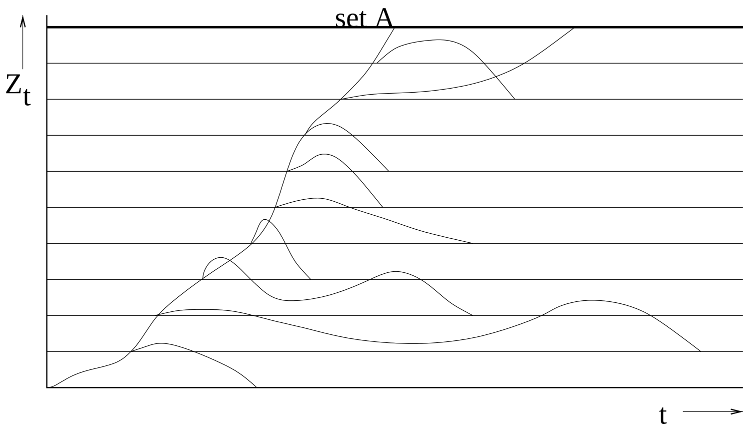

Figure 3.2: *Truncation*

A small under-bias will occur in the estimator because of the lost contribution from hits that would have occurred without cut-off, however this bias usually drops exponentially fast to zero in the cut-off depth, and it can be compensated for with an estimate of the contribution of the cut-off paths. Since this compensation will usually be unreliable, it is best to include a compensation in the relative error of equal size as the relative change in the estimate. We will try to find an expression for the incurred bias by a study of a simple example.

Denote the chosen cut-off as k. Cutting the simulations starting in stage i after crossing k thresholds downwards will result in a bias of p_i for $i > k$ of $\prod_{j=i-k}^{i} p_j$. Assuming that the success probabilities are approximately constant $p_i = p \quad, i = 1, 2, \ldots m$, which will typically hold in an exponential decay problem for large i, this results in $\mathbb{E}(p_i - \hat{p}_i) = p^k$. Since this is valid for all i we obtain a bias of

$$\begin{aligned}\mathbb{E}(1 - \frac{\hat{\gamma}}{\gamma}) &= \mathbb{E}(1 - (\prod_{i=1}^{m} \frac{\hat{p}_i}{p_i})) \\ &= 1 - (\prod_{i=1}^{m} \mathbb{E}\frac{\hat{p}_i}{p_i}) = 1 - (\prod_{i=k}^{m} (1 - p^{k-1})) \\ &\approx (m-k)p^{k-1}.\end{aligned}$$

in the first order approximation. To keep the relative bias limited to b one can obtain the appropriate k from the above formula. Since the approximation is rather crude, the b should be rather small, e.g. e^{-6}, resulting in the equation

$6 < 2(k-1) - \log(m-k)$ for the optimal $p = e^{-2}$, thus making $k = 4$ a good rule of thumb.

The efficiency gain using this cut-off technique is quite substantial and is a definite enhancement of the splitting technique.

3.5 Global and Single Stepping

The nomenclature Global and Single Stepping has been chosen in concordance with [GF98] where these different implementations are introduced.

The splitting algorithm basically works in two loops, one finishing all the consecutive stages of simulation, and one that sparks a new series of consecutive stages by starting again in the first stage in the starting state, this n_0 times. This is called the Global Step method since all the offspring of one first stage path are finished before the next first stage path is started. Another implementation works by exchanging the two loops, or in other words by doing all the simulations belonging to one stage first, and then moving to the next stage where re-trials of the saved successful runs of the previous stage are simulated. We will refer to this as the Single Step method. A graphical presentation of the simulations done in Single Step versus time is found in Figure 3.3; for the Global Step we refer to Figure 2.1.

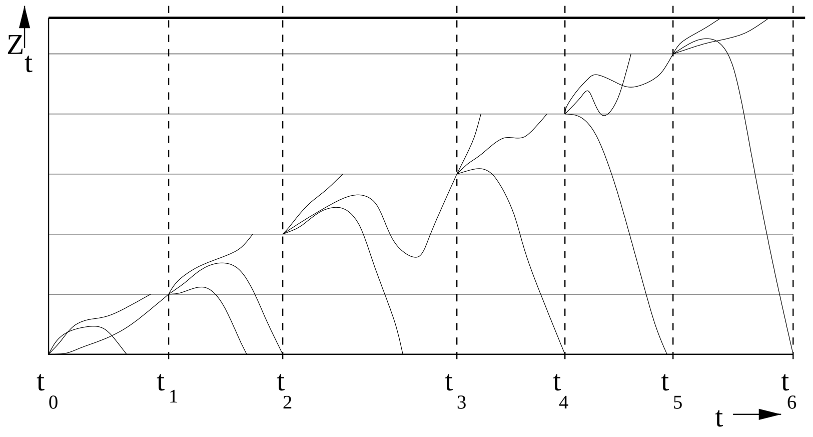

Figure 3.3: *Single Stepping*

Note that the stages are handled sequentially: between t_{i-1} and t_i all and only simulation of paths belonging to stage i takes place, whereas the Global Step does all the simulation dependent on a just-obtained saved state first.

Memory requirements Both stepping methods work well in practice, the only difference is in the memory requirements: the Global Stepping method requires a memory space of $m-1$ states (one for each intermediate threshold), and the Single Stepping method will need as much memory as there are successful hits in a stage, R_i, for saving the successful paths' entrance state for the next stage, plus the current entrance states from which the simulation is restarted, R_{i-1}. A quick calculation shows that the memory requirements measured in the number of system states to be saved (NSS_i) in stage i are

$$\mathrm{NSS} := \max_i \{\mathrm{NSS}_i\} = R_{i-1} + R_i \approx 2\mathbb{E}R_i = 2p_i n_0 \approx 2p_1 n_0$$

for the Single Step method when the $(n_i, p_i)_{i=1}^{m-1}$ are optimally (equal) chosen, and thus the only variable is the number of starting samples n_0. Denoting the virtual memory available in the simulation machine as M and the memory requirements for a single saved state as x, we see that memory limitations force the restriction $M \geq x \cdot \mathrm{NSS}$ and thus equivalently $n_0 < M/(2p_1 x)$, e.g. for the $M/G/1$-model in our simulator we[1] need $x = 10\mathrm{B}$ and have $n_0 < 3\mathrm{GB}/(2 \cdot e^{-2} \cdot 10\mathrm{B}) = 1.05 \cdot 10^9$. For an ATM model with 300 sources, where we will need storage for a floating point and an integer number per source, we have $x = 5200\mathrm{B}$, causing $n_0 < 2.02 \cdot 10^6$ which becomes restrictive. It is clear that even for modern machines these simulations become too large to handle because of memory requirements.

Comparison of both implementations This restrictive effect, limiting the achievable accuracy in the Single Step method results in choosing the Global Step method as the default algorithm. Also, in [GF98] this strategy was chosen as the best implementation. Furthermore, this method is easily extended into a multi-threading algorithm while the Single Step method does not allow for this in a simple fashion.

The Single Step approach does give control over the choice of the entrance state for the simulations, whereas the Global Step does not allow any change in implementation. We will look closer at implementations of the Single Step splitting algorithm.

3.5.1 Fixed Effort vs. Fixed Splitting

In the Single Step *Fixed Splitting* (FS) method we create at every stage a fixed number of offspring from each saved state. In the Single Step *Fixed Effort* (FE) method we create at every stage a fixed total number of offspring. Notice that

[1]We assume that 3 GB (gigabyte) of virtual memory is available. Note that B means "byte" and equals an array of 8 memory locations capable of storing a binary value.

in the latter implementation, the number of offspring per saved state depends on the total number of saved states at a certain stage.

A serious disadvantage of the FS method is that it is very sensitive to the choice of the splitting levels and the number of splits per saved state in each stage, especially when we have many stages. If we do not choose these parameters exactly right, the paths will either "die-out" or "explode", which in both cases leads to a large variance for $\hat{\gamma}$. It is therefore of utmost importance to keep the underlying branching process "critical". However, this is not an easy task. In [GHSZ99] Glasserman et al. provide a way to alleviate this problem by *randomizing* the number of splits to ensure that the number of hits will remain about the same throughout the stages, but this randomization requires a pilot run to determine the success probability in each stage in order to get the randomization working correct. The FE method however *fixes* the total number of offspring per stage and thus here we can never have any possibility of the number of successes dying out or exploding (of course it is still possible that we will not obtain any hit in some stage, causing a die-out, but this is very unlikely when the success probability is substantial and we use a large number of offspring). Note that the FE method tries to approach the critical branching process since for FE we have for the number of offspring per saved state in stage i the relation $\mathbb{E}(n_i) := r_i/R_i = 1/\hat{p}_i$, i.e. the balanced growth equation (2.25) but with the estimated success probability and with randomized number of re-trials n_i in order to cope with the non-integer reciprocal of $\hat{p}_i$. Simulation experiments show that the FE method approximates the critical branching since it not only removes the possibility of starvation or explosion but also gives better efficiency.

A possible disadvantage of the FE method is that the variance of $\hat{\gamma}$ does not have a simple form, and has to be estimated asymptotically (for a large total simulation effort B), see Appendix A. The FS method does not have such problems. The variance of $\hat{\gamma}$ can be estimated easily using the facts that the n_0 runs at the lowest level are completely independent of each other as explained in Section 2.4.3.

3.5.2 Fixed Assignment vs. Random Assignment

Assume we are using a Fixed Effort Single Step implementation. Suppose at stage k we have had R_k successful hits and wish to perform r_k new runs. We have to assign to each new run one of the R_k possible starting states. This can be done deterministically or randomly. For example, in the FS method each starting state is assigned a fixed number n_k of offspring. In the FE method however, r_k (fixed) starting states have to be chosen from R_k possible starting states. We could for example independently choose (r_k times) one of the R_k

starting positions with equal probability. This seems to be a sensible approach, because we then effectively draw the starting state from the empirical entrance distribution of stage k. We call this the *Random Assignment* (RA) approach. Notice that if we would know the *actual* entrance distribution at stage k, we should be drawing from this distribution, thus creating independent runs and reducing the variance of $\hat{\gamma}$. An alternative approach is to distribute the R_k starting states evenly (deterministically) amongst the r_k runs. We call this the *Fixed Assignment* (FA) method.

In Figure 3.4 we clarify the just proposed possible implementations of the splitting methods.

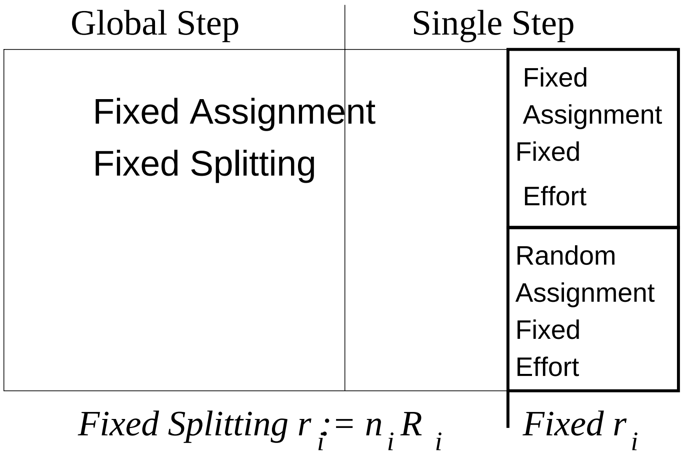
Global Step
Single Step
Fixed Assignment
Fixed Splitting
Fixed Assignment Fixed Effort
Random Assignment Fixed Effort
Fixed Splitting $r_i := n_i R_i$
Fixed r_i

Figure 3.4: *The Stepping implementations*

Note that the estimators from a Global Step and a FS-implementation of the Single Step method are statistically equivalent, since they prescribe the same algorithm but in different order.

We will analyse the variance generated in the second stage in a two-stage simulation for both methods. Suppose we have R_1 starting states $S_1, \ldots, S_{R_1}$. R_1 is assumed here to be *fixed.* We wish to start a total of $r_1 := n_1 R_1$ new runs, where n_1 is some fixed integer. The success probability from some state s will be denoted by $p_2(s)$. Let Y_i be the number of successful runs (that reach the next level) starting from state S_i. Since we have only one intermediate stage, the S_i's are independent and identically distributed, and $\mathbb{E}(p_2(S_i)) = p_2$ by

(2.4). Consequently,

$$\hat{p}_2 := \frac{1}{r_1}\sum_{i=1}^{R_1} Y_i$$

is an unbiased estimator for p_2.

Fixed Assignment
From each state we start n_1 independent paths. Then, *given* the vector $S := (S_1, \ldots, S_{R_1})$, Y_i has a Binomial distribution of size n_1 and success probability $p_2(S_i)$. Let us denote conditional expectation and variance with respect to S by $\mathbb{E}_S$ and $\mathbb{V}\mathrm{ar}_S$ respectively. We have

$$\begin{aligned}
\mathbb{V}\mathrm{ar}\,(\hat{p}_2) &= \frac{1}{r_1^2}\sum_{i=1}^{R_1}\{\mathbb{E}\mathbb{V}\mathrm{ar}_S\,(Y_i) + \mathbb{V}\mathrm{ar}\,\mathbb{E}_S\,(Y_i)\} \\
&= \frac{1}{r_1^2}\left\{\mathbb{E}\{\sum_{i=1}^{R_1} n_1 p_2(S_i)(1-p_2(S_i))\} \;+\; \mathbb{V}\mathrm{ar}\{\sum_{i=1}^{R_1} n_1\, p_2(S_i)\}\right\} \\
&= \frac{1}{r_1}\;\{\mathbb{E}\,(p_2(S_1) - p_2^2(S_1)) + n_1\,\mathbb{V}\mathrm{ar}(\,p_2(S_1))\} \\
&= \frac{1}{r_1}\{p_2(1-p_2) + (n_1-1)\mathbb{V}\mathrm{ar}\,(p_2(S_1))\}\,.
\end{aligned}$$

Random Assignment
We distribute the R_1 starting states randomly amongst the $r_1 = n_1 R_1$ new runs. Let K_i denote the number of runs that start from state S_i. Then, the vector $K := (K_1, \ldots, K_{R_1})$ has a multinomial distribution with the number of trials equal to r_1 and with equal success probabilities $1/R_1$. In particular,

$$\mathbb{E}\,(K_1^2) = \frac{r_1}{R_1}(1 - \frac{1}{R_1} + \frac{r_1}{R_1})$$

and

$$\mathbb{E}\,(K_1\,K_2) = \frac{r_1(r_1-1)}{R_1^2}.$$

Notice that the Y_i (the number of successes per successful run) are identically distributed, but not independent. Also, each pair $(Y_i, Y_j), i \neq j$ has the same joint distribution. Hence,

$$\mathbb{E}\,(\hat{p}_2^2) = \frac{1}{r_1^2}\left\{\mathbb{E}\sum_{i=1}^{R_1} Y_i^2 + \mathbb{E}\sum_{i=1}^{R_1}\sum_{j\neq i} Y_i\,Y_j\right\} = \frac{R_1}{r_1^2}\,\{\mathbb{E}\,(Y_1^2) + (R_1-1)\mathbb{E}\,(Y_1\,Y_2)\}\,.$$

Moreover, given K, the Y_i's are independent and given K *and* S, each Y_i has a Binomial distribution with size K_i and success probability $p_2(S_i)$. Finally, K and S are independent. Let us denote the conditional expectation with respect to K and S by $\mathbb{E}_{K,S}$. We have

$$\begin{aligned}
\mathbb{E}(Y_1^2) &= \mathbb{E}\mathbb{E}_{K,S}(Y_1^2) = \mathbb{E}(K_1 p_2(S_1)(1-p_2(S_1))) + \mathbb{E}(K_1^2\, p_2^2(S_1)) \\
&= \mathbb{E}(K_1)\,\mathbb{E}(p_2(S_1)) - \mathbb{E}(K_1)\,\mathbb{E}(p_2^2(S_1)) + \mathbb{E}(K_1^2)\,\mathbb{E}(p_2^2(S_1)) \\
&= \frac{r_1}{R_1}(p_2 - p_2^2 - \mathbb{V}\mathrm{ar}\,(p_2(S_1))) + (p_2^2 + \mathbb{V}\mathrm{ar}\,(p_2(S_1)))\,\mathbb{E}(K_1^2)
\end{aligned}$$

and

$$\mathbb{E}(Y_1 Y_2) = \mathbb{E}\mathbb{E}_{K,S}(Y_1\, Y_2) = \mathbb{E}\{K_1\, p_2(S_1)\, K_2\, p_2(S_2)\} = p_2^2\,\mathbb{E}(K_1 K_2).$$

Combining these results we get

$$\mathbb{V}\mathrm{ar}\,(\hat{p}_2) = \frac{1}{r_1}\left\{p_2(1-p_2) + \frac{(r_1-1)\mathbb{V}\mathrm{ar}\,(p_2(S_1))}{R_1}\right\}.$$

When we compare this with the variance for the Fixed Assignment case we see that the Random Assignment always gives a higher variance, irrespective of the unknown constant $\mathbb{V}\mathrm{ar}\, p_2(S_1)$. This is a somewhat surprising result, which has been verified empirically, both for the two-stage and the multi-stage case, see Section 3.8.

Remark 3.1 As we have remarked before, if we know the true entrance distribution at some stage, we should start our runs independently from this distribution. In general, the starting distribution should be chosen such that there is as little variability as possible in the expected success probability of hitting the next threshold over the entrance states. Note that we cannot directly choose the entrance distribution but indirectly influence it by our choice of the I.F.

For instance, for the $GI/G/1$-queue of Example 2.3, we should in general save the remaining service and arrival times. However, for an $M/G/1$- queue we only should save the remaining service time of the customer in service, and *redraw* the remaining arrival time at every restart. For an $M/M/1$-queue we should redraw both the remaining arrival and the remaining service time.

3.6 RESTART

The RESTART (Repetitive Simulation Trials After Reaching Thresholds) algorithm as first presented in [VAVA91] and enhanced in [VAMMGFC94] for multi-stage simulations works in a slightly different way with respect to the splitting decisions.

Difference with respect to Splitting

A sample path is split only for the time during which it stays above the threshold where it originated. All sample paths hitting threshold L_i are split into n_i sub-paths. The first $n_i - 1$ paths are stopped as soon as they fall back below threshold L_i; only the last path is permitted to continue after it falls back below threshold L_i. As soon as it does, the weight of that path is restored to the sample path that created it before splitting, effectively absorbing all the weight of the offspring that were stopped. This last split path thus becomes a continuation of the original path.

Of course the choice of the last path as the absorbing path is only for saving restoration costs of another system state; there is no magic involved in the choice of the surviving path.

Since this path is now effectively similar to a path that started in stage $i-1$, it has the feature that it can hit the threshold L_i again and split again. The path will be stopped when it falls below threshold L_{i-1}, unless again it is the last path of a split which occurred at threshold L_{i-1}. The same rules apply as for the split at threshold L_i: it can continue re-splitting upon hitting L_{i-1} or be absorbed at level L_{i-2}, etc.

A visual illustration of this modification of the splitting algorithm is given in Figure 3.5.

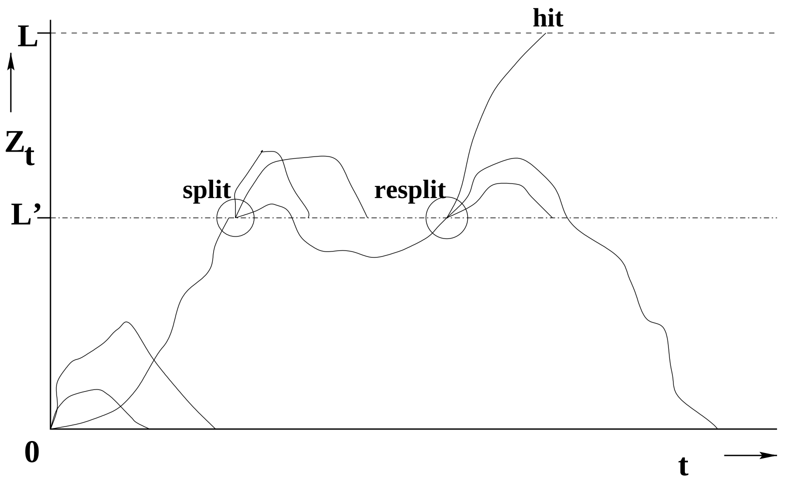

Figure 3.5: *The RESTART algorithm*

Advantages

The gain in this implementation is that out of every starting path from the starting state exactly one path will return to the zero level and stop there, in the case that all paths are only stopped upon returning to the zero level. Most sample paths which have to be simulated back to the zero level (the default stopping time) in the standard splitting method will be absorbed using this technique, thus significantly reducing the simulation time.

Drawbacks

We can recognize two drawbacks from using the RESTART run-time strategy:

- Sample paths in stage i which are offspring of a re-split have a common history up to the moment of the parent's first hit to threshold L_i. Since this is not the case for the standard splitting method, where every offspring group has a unique history in stage $i-1$, the variance in the estimator using the RESTART method will be larger than that using pure splitting. This effect is very hard to verify in simulation and analysis, it might be negligible in most cases.

- Another extra variance contribution stems from the fact that the number of splits R_i at each threshold is now much more variable due to the fact that the number of re-splits is very variable. This effect can be analysed and is shown to contribute to the variance and thus the efficiency.

Which method will be more efficient depends on the sample path dependencies which is intrinsic to the simulated process In general, independent sample paths such as those that occur in an $M/M/1$ type model will favour RESTART and highly dependent sample paths, which are common in self-similar traffic models, will favour the pure splitting method. Evidence for this behaviour will be found in Chapter 6 for the steady-state estimation and in Section 3.8 in this chapter.

Optimal parameters

The only difference with the standard splitting method is the absorbing and re-splitting, which changes the number of samples in each stage. The re-splitting hits at threshold L_i should not be regarded as regular threshold hits that count in the variable R_i. Samples originating from the re-split can be viewed as continuations of the absorbed paths from the original split; and thus this case does not necessitate a different set of optimal parameters $(m, n_i, L_i, i = 1, 2, \ldots, m)$.

Expectation

We will now proceed by proving that RESTART is in fact unbiased. Introduce the variables $S_{i,k}$ denoting the system state upon hitting threshold L_i for the k-th time before returning to zero and before hitting threshold L_{i+1}. Also, say $q_{i+1,k}$ is the probability of hitting threshold L_{i+1} from system state $S_{i,k}$ and N_i is the number of crossings of threshold L_i we see before a crossing of L_{i+1}. Note that in this formulation we obtain the number of re-splits of threshold i as $N_i - 1$; we must deduct one for the first hit of the threshold which does not count as a re-split. The system state upon the $(k-1)$-th re-split is now given by $S_{i,k}$ and the probability that one of the splits from the $(k-1)$-th re-split hits the next threshold is given by $q_{i,k}$. Clearly we have by the laws of total and conditional probability

$$\begin{aligned}
\mathbb{E}(R_{i+1} \mid R_i) &= \sum_{k=1}^{\infty} \mathbb{E}(R_{i+1} \mid R_i, N_i = k)\mathbb{P}(N_i = k) \\
&= \sum_{k=1}^{\infty} \mathbb{E}(\sum_{l=1}^{R_i} \sum_{j=1}^{k} \mathrm{Bin}(n_i, p_{i+1}(S_{i,j})) \mid R_i, N_i = k)\mathbb{P}(N_i = k) \\
&= \sum_{k=1}^{\infty} R_i \mathbb{E}(\sum_{j=1}^{k} \mathrm{Bin}(n_i, q_{i+1,j}) \mid N_i = k)\mathbb{P}(N_i = k) \\
&= \sum_{k=1}^{\infty} R_i n_i \sum_{j=1}^{k} q_{i+1,j}\mathbb{P}(N_i = k) = R_i n_i \sum_{k=1}^{\infty} q_{i+1,k}\mathbb{P}(N_i \geq k) \\
&= R_i n_i p_{i+1} \qquad (3.3)
\end{aligned}$$

which leads to unbiasedness by following the proof in Section 2.4.2.

Variance

We focus on a two-stage simulation set-up with a system state equal to the level to keep the discussion compact. Examples of such a model are the classic random walk and the $M/M/1$ queueing model. Denote q_i as the probability of hitting threshold L_i directly from threshold L_{i-1} without re-crossing threshold L_{i-1}, in other words, without leaving stage i. Denote u_i as the probability of a path starting at threshold L_{i-1} down-crossing threshold L_{i-1} up-crossing the same threshold before being stopped. This u_i is then equal to the probability of a last split causing a re-split in stage i. Combining factors we have $p_i = q_i + (1-q_i)\, u_i\, p_i$ and thus

$$p_i = \frac{q_i}{1 - u_i(1-q_i)}$$

Denote V_i as the variable describing the number of times a path from threshold i is re-split; V_i has a geometric distribution with parameter $(1-q_{i+1})u_{i+1}$. The number of hits of the next threshold is then seen to have a Binomial distribution with parameters $n_i(1+V_i)$ and q_{i+1}. We obtain unbiasedness in this specific case easily by noting that we have the same situation as in Section 3.6 but now with all the $q_{i,k} = q_i$ fixed for all k since there is no system state, and thus no dependency on k.

$$\begin{aligned}
\mathbb{E}_{\mathrm{RT}}(R_{i+1} \mid R_i) &= \mathbb{E}(\sum_{k=1}^{R_i} \mathrm{Bin}(n_i(1+V_i^{(k)}), q_{i+1}) \mid R_i) \\
&= R_i \mathbb{E}\{\mathbb{E}(\mathrm{Bin}(n_i(1+V_i), q_{i+1}) \mid V_i)\} = R_i \mathbb{E}(n_i(1+V_i)q_{i+1}) \\
&= R_i n_i q_{i+1}(1 + \frac{1}{(1-q_{i+1})u_{i+1}}) = n_i p_{i+1} R_i \qquad (3.4)
\end{aligned}$$

For our two-stage case we plug this into the variance of Y_1 (see (2.18) for the variance expression of Y_1, note that we use the unbiasedness from (3.4)):

$$\begin{aligned}
\mathbb{V}\mathrm{ar}_{\mathrm{RT}}(Y_1) &= p_1\, \mathbb{V}\mathrm{ar}(\mathrm{Bin}(n_1(1+V_1), q_2)) + p_1(1-p_1)(n_1 p_2)^2 \\
&= p_1\{\mathbb{E}(\mathbb{V}\mathrm{ar}(\mathrm{Bin}(n_1(1+V_1), q_2)|V_1)) + \\
&\quad + \mathbb{V}\mathrm{ar}(\mathbb{E}(\mathrm{Bin}(n_1(1+V_1), q_2)|V_1)\} + p_1(1-p_1)(n_1 p_2)^2 \\
&= p_1\{\mathbb{E}(n_1(1+V_1)q_2(1-q_2)) + \\
&\quad + \mathbb{V}\mathrm{ar}(n_1(1+V_1)q_2) + (1-p_1)n_1^2 p_2^2\} \\
&= p_1\{n_1(1 + \frac{(1-q_2)u_2}{(1-(1-q_2)u_2)})q_2(1-q_2) + \\
&\quad + n_1^2 q_2^2 \frac{(1-q_2)u_2}{(1-((1-q_2)u_2))^2} + (1-p_1)n_1^2 p_2^2\} \\
&= p_1\{n_1 p_2(1-q_2) + n_1^2 p_2(p_2-q_2) + n_1^2 p_2^2(1-p_1)\} \\
&= \gamma n_1((1-q_2) + n_1(p_2 - q_2 + p_2(1-p_1)))
\end{aligned}$$

As usual the abbreviation $\mathrm{Bin}(n,p)$ denotes a random variable with a Binomial distribution with parameters $n :=$ number of trials and $p :=$ success probability. The subscripts *RT* and *SP* will denote the method used, i.e. RESTART and pure splitting, respectively. For the splitting method without dependency on the system state we obtain from (2.20)

$$\mathbb{V}\mathrm{ar}_{\mathrm{SP}}(Y_1) = \gamma n_1\{(1-p_2) + n_1 p_2(1-p_1)\},$$

which makes the variance of Y_1 using the RESTART algorithm larger by an amount of

$$\mathbb{V}\mathrm{ar}_{\mathrm{RT}}(Y_1) - \mathbb{V}\mathrm{ar}_{\mathrm{SP}}(Y_1) = \gamma n_1\{(n_1-1)(p_2-q_2)\} \geq 0 \qquad (3.5)$$

since all terms are non-negative; only if all down-crossing paths never return to the re-splitting threshold, $u_2 = 0$, we can have $p_2 = q_2$ and the variance difference equals zero; in all other cases splitting has a smaller variance. The cases $\gamma = 0$ and $n_1 = 0$ are not valid set-ups; the case $n_1 = 1$ degrades both methods to the Monte Carlo method.

We were able to derive the variance difference between RESTART and the original splitting method only when we assumed the success probability to be equal for every entrance state S_1. For the general model in which we do not know the distribution of V_1 we have the following relation by (2.18) and (3.3):

$$\begin{aligned}
\mathbb{V}\mathrm{ar}_{\mathrm{RT}}(Y_1) &= p_1\,\mathbb{V}\mathrm{ar}(\sum_{k=1}^{N_1}\mathrm{Bin}(n_1, q_{2,k})) + p_1(1-p_1)(n_1p_2)^2 \qquad (3.6)\\
&= p_1\mathbb{V}\mathrm{ar}\{\mathbb{E}(\sum_{k=1}^{N_1}\mathrm{Bin}(n_1, q_{2,k})|N_1)\} +\\
&\quad +p_1\mathbb{E}\{\mathbb{V}\mathrm{ar}(\sum_{k=1}^{N_1}\mathrm{Bin}(n_1, q_{2,k})|N_1)\} + p_1(1-p_1)n_1^2p_2^2\\
&= p_1\mathbb{V}\mathrm{ar}\{n_1\sum_{k=1}^{N_1}q_{2,k}\} +\\
&\quad +p_1\mathbb{E}\{n_1\sum_{k=1}^{N_1}q_{2,k}(1-q_{2,k})\} + p_1(1-p_1)n_1^2p_2^2\\
&= p_1n_1^2\mathbb{V}\mathrm{ar}Q_2(N_1) + p_1n_1\mathbb{E}\{Q_2(N_1) - Q_2^2(N_1)\} + p_1(1-p_1)n_1^2p_2^2\\
&= p_1n_1(n_1-1)\mathbb{V}\mathrm{ar}Q_2(N_1) + p_1n_1p_2(1-p_2) + p_1(1-p_1)n_1^2p_2^2
\end{aligned}$$

where we use the shorthand summation notation $Q_i(j) = \sum_{k=1}^{j} q_{i,j}$, and the equality $\mathbb{E}Q_2(N_1) = \sum_{k=1}^{\infty} q_{2,k}\mathbb{P}(N_1 \geq k) = p_2$ as follows immediately by the unbiasedness (3.3). The difference in the variance of the number of successes, and thus the estimator for the rare event probability, between the splitting and RESTART methods for general entrance distributions can then easily be found as

$$\mathbb{V}\mathrm{ar}_{\mathrm{RT}}(Y_1) - \mathbb{V}\mathrm{ar}_{\mathrm{SP}}(Y_1) = p_1n_1(n_1-1)\mathbb{V}\mathrm{ar}Q_2(N_1) \qquad (3.7)$$

which clearly exceeds zero whenever the random variable $Q_2(N_1)$ has a non-degenerate distribution; and thus in general we can conclude that the original splitting method has a lower or equal variance than the RESTART method for the same simulation parameters. We obtain the special case (3.5) from (3.7) by noting that for the state-less system we have $Q_2(N_1) = q_2N_1$ and $N_1 - 1$ has a geometric distribution with parameter $(1 - q_2/p_2)$, leading to

$$\mathbb{V}\mathrm{ar}(Q_2(N_1)) = \mathbb{V}\mathrm{ar}(q_2 N_1) = q_2^2 \mathbb{V}\mathrm{ar}(\mathrm{Geom}(1 - \frac{q_2}{p_2})) = q_2^2 \frac{1 - \frac{q_2}{p_2}}{(\frac{q_2}{p_2})^2} = p_2(p_2 - q_2)$$

which by substitution into (3.7) indeed gives the predicted state-less variance gain (3.5). Also, for more than two stages, the variance of the RESTART method is expected to grow compared to the splitting method because of increased variability using more re-trial variables V_i. Note that in spite of the variance increase by using the RESTART implementation, it still can have a better efficiency than the splitting method because of the large reduction in simulation time. It is difficult to determine the optimal choice of implementation in advance.

3.6.1 Truncation variant of RESTART

In order to solve the variance buildup in the standard RESTART method it must behave more like the splitting method, i.e. the dependence must be lowered at the cost of simulation time. The way to do this is to introduce the notion of truncation into RESTART. Note that Villén-Altamirano , in [VAVA91], also introduced the variable-depth truncation (i.e. at a threshold lower than where the split took place), where it is called hysteresis.

In the standard RESTART the simulation paths are cut at the moment that they down-cross the threshold that they started from. In the truncation method simulation is stopped when the importance function drops one or more thresholds below the threshold at which that path originated. We therefore propose the combination of the two methods: truncate the split paths when they down-cross a fixed number of thresholds; but permit the last split path to survive. This truncation variant of RESTART method is still unbiased, since the variable truncation has no effect on the expected number of successes per stage (R_i). The split paths are just simulated longer downwards; the behaviour of absorption and re-splitting at this lower threshold is exactly the same as in the original RESTART method. Note that as a consequence we use a re-splitting factor (number of re-splits) of n_i even though the re-splitting threshold is now lower than L_{i-1}. Another consequence is that there will be no splitting when a re-split path hits a threshold lower than L_i. It is also clear that a greater truncation depth increases the direct hit probability q_i that a splitting path hits the next threshold before being absorbed, since it now has more opportunity to do so. This immediately results in a reduction of the variance difference between the RESTART and splitting methods by (3.5). This set-up could even be more efficient than the truncation-enabled splitting method since no correction for truncation is necessary, and thus it can achieve a lower variance, especially in simulations with very many thresholds. The price of having an unbiased estimator can be seen to be the simulation time spent on the re-splits, which might be more costly than correcting for the bias and thus increasing the variance as done in the truncation-enabled splitting method. Whether the efficiency will also outperform the standard splitting method with truncation is impossible to say, since one weighs the extra simulation effort of simulating the re-splits against the extra accuracy in the estimator; in section 3.8 we look at this issue with numerical results.

The optimal truncation depth (the number of thresholds to be down-crossed before the simulation is cut) will be hard to determine analytically. The rule of thumb given for the splitting method seems to work well.

An example of the truncation variant of RESTART is given in Figure 3.6.

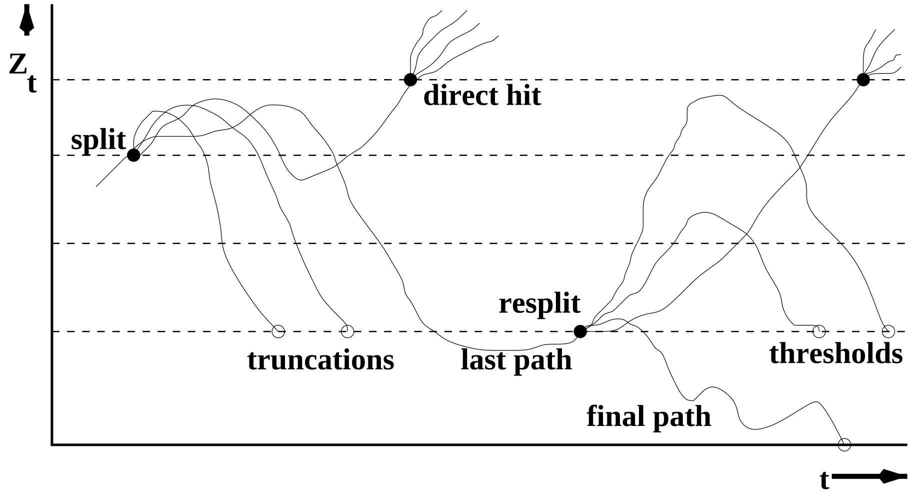

Figure 3.6: *The RESTART/truncation algorithm*

3.7 Dependency Effects

When the success probability of a Bernoulli sample depends on the entrance state, in any stage of simulation, we are dealing with dependency. This dependency will cause the standard choice of the thresholds as described in Section 3.3 to be non-optimal, as was proven in Section 2.4.7. In order to restore asymptotic optimality for the splitting algorithm, we need to change the success probabilities in each stage of simulation according to the optimal solution as given by (2.32). Since this again depends on the (unknown) number of stages m and the (unknown) variation coefficients of the stage i Bernoulli samples, $c_i, i = 2, 3, \dots, m$, we will need methods for estimating the optimal parameters $(m, c_2, c_3, \dots, c_m, p_1, p_2, \dots, p_m)$.

3.7.1 Optimal number of stages

We analyse the proper choice of the number of stages for given success probability γ and variation coefficients $c_2, c_3, \dots c_m$. Note that these are generally unknown beforehand and need to be estimated from a trial run. As shown in Section 2.4.7, from equation (2.34) we derive the optimal number of stages as the solution of

$$\min_{m\in\mathbb{N}} \quad m^2\gamma^{-1/m}\Big(\prod_{k=2}^{m+1}(1+c_k)\Big)^{1/m} - m\sum_{k=2}^{m+1}(1+c_k)$$

just as was proposed in (2.36) of Example 2.4. Since this optimization program cannot be solved analytically, we find the optimum by enumeration. Note that this is easy since we know that the number of stages under dependence is always lower than under independence, and thus the maximum number of stages to be used is the number of stages in the case of independence of the stages, being $m_{max} = -\log(\gamma)/2$. The optimal value m^* under dependence can be found by enumeration since $m^* \in \{1, 2, \dots, m_{max}\}$.

Note that a simpler asymptotically optimal choice for m, inspired by the discussion in Section 2.4.7, is given by $m^* = \log(\gamma(1+c))/2$, where c is the asymptotic value of the variation coefficient sequence $(c_2, c_3, \dots, c_m)$. An estimate for c can be found as the last value c_m of the sequence $(c_2, c_3, \dots, c_m)$ obtained by the trial run of by properly averaging the last values of the available sequence $(c_k)_{k=2}^m$.

3.7.2 Variation coefficients

Next we need to have a method to obtain estimates for the variation coefficients $(c_2, c_3, \dots, c_m)$ from a trial run. In order to do so, we must extend the estimation procedure. Denote the probability that a sample will cross k thresholds as γ_k. The original target γ is found as γ_m. We find estimates for every γ_k by estimators analogous to the one for γ from Section 2.4.3 as

$$\widehat{\gamma_k} = \frac{R_k}{\prod_{i=0}^{k-1} n_i} = \frac{\sum_{j=1}^{n_0} Y_{j,k}}{\prod_{i=0}^{k-1} n_i} \quad , k = 1, 2, \dots, m$$

Denote the number of successful hits from offspring of the first stage path with number j of threshold L_k as $Y_{j,k}$, extending the definition in Section 2.4.3 which is the special case $Y_j := Y_{j,m}$. The variance of the estimators γ_k is then found as

$$\widehat{\mathbb{V}\mathrm{ar}}(\widehat{\gamma_k}) = S_k^2 := \frac{\sum_{j=1}^{n_0}(Y_{j,k} - \bar{Y}_k)^2}{n_0(n_0-1)\prod_{i=1}^{k-1} n_i^2} \quad , k = 1, 2, \dots, m$$

Note that we define $\bar{Y}_k := n_0^{-1}\sum_{j=1}^{n_0} Y_{j,k}$, the mean of the n_0 observations of $Y_{j,k}$. From (2.31) we obtain the real variance σ_k^2 of $\widehat{\gamma_k}$:

$$\sigma_k^2 = \mathbb{V}\mathrm{ar}(\widehat{\gamma_k}) = n_0^{-1}\sum_{i=1}^{k}(1 + c_{i+1})\frac{1-p_i}{p_i} \quad , k = 1, 2, \dots, m$$

from which we extract the desired parameter $c_{k+1}, k > 0$ as follows (define $\sigma_0^2 := 0$):

$$c_{k+1} = \frac{n_0 p_k}{1-p_k}(\sigma_k^2 - \sigma_{k-1}^2) - 1 \quad , k = 1, 2, \dots, m.$$

Note that we obtain an estimator for p_k using

$$\widehat{p_k} = \frac{\widehat{\gamma}_k}{\widehat{\gamma}_{k-1}} = \frac{R_k}{R_{k-1}n_k} \quad , k = 1, 2, \dots, m$$

and can thus create an estimator for c_{k+1} as follows

$$\widehat{c}_{k+1} = n_0(\frac{1}{1-\widehat{p_k}} - 1)(S_k^2 - S_{k-1}^2) - 1 \quad , k = 1, 2, \dots, m.$$

Note that the proposed estimator is asymptotically unbiased.

3.7.3 Pilot run algorithm

We will now describe the proper algorithm for the pilot run, which determines the optimal number of stages, the optimal thresholds and the optimal number of splits per threshold for our model in the presence of dependency.

We introduce the following vectors of the splitting parameters for reading ease.

$$\begin{aligned} \boldsymbol{n} &= (n_0, n_1, \dots, n_{m-1}) \\ \boldsymbol{L} &= (L_1, L_2, \dots, L_m) \\ \boldsymbol{p} &= (p_1, p_2, \dots, p_m) \\ \boldsymbol{c} &= (c_2, c_3, \dots, c_m) \end{aligned}$$

The proposed algorithm for the pilot run then becomes

0. Set $\hat{\boldsymbol{c}} = \mathbf{0}$, i.e. assume independence.
1. Set $\hat{m} = m(\hat{\boldsymbol{c}}, \widehat{\gamma})$, using the algorithm from Section 3.7.1.
2. Set $\hat{\boldsymbol{p}} = \boldsymbol{p}(\hat{m}, \widehat{\gamma}, \hat{\boldsymbol{c}})$, using equation (2.32).
3. Do pilot run determining $(\boldsymbol{L}, \boldsymbol{n}, \hat{\boldsymbol{c}}, \widehat{\gamma}) = g(\hat{\boldsymbol{p}}, \hat{m})$, using the algorithms from Sections 3.3.1 and 3.7.2.
4. If satisfied, stop ; else go to step 1.

The iteration in step 4 is necessary since a change in the vector $\boldsymbol{L}$, the vector op the splitting thresholds, will cause a change in vector $\boldsymbol{c}$, the vector containing the dependency factors, since the success probabilities and their variances over the entrance distributions change when the thresholds change. The algorithm should stop when all parameters have converged; note that there is no guarantee that they will, or even that they converge to the asymptotically optimal parameters if they do. However, we believe that it is the best pilot run available. Note that by removing the iteration in step four we arrive at the algorithm used in Section 3.8 to determine the thresholds whenever we assume independence; showing that the algorithm works.

3.8 Simulation Results

To study the behaviour of the different splitting method implementations we conduct a series of simulation experiments using various different models. We will compare all different presented implementations. Comparing all the Stepping implementations will be done in the first three models, for the Global Step and the Single Step methods. The run-time modification RESTART is compared with the pure splitting method in the next models; the comparison of truncation implementations in both methods is also presented there.

The splitting parameters (i.e. m, $L_1, \ldots, L_m$, $n_1, \ldots, n_m$) are chosen in accordance with the values suggested in Section 2.4.4. For example, we try to choose the levels such that success probabilities (of hitting the next level) are near the “optimal” value e^{-2}.

In all tables γ denotes the rare event probability of interest. The estimate of γ is given by $\hat{\gamma}$. For each $\hat{\gamma}$ the corresponding estimate of the *Relative Error* (RE, as defined in (2.3)) is included. As a measure of the efficiency of the estimator $\hat{\gamma}$ we use the *Relative Time Variance product* (RTV), which we define as the simulation time (in seconds) multiplied by squared (estimate of the) relative error of $\hat{\gamma}$. Notice that the RTV is equivalent to the “work-balanced variance” used in [GW92]. Once a stable estimate of the variance is reached, the

RTV becomes constant. This constant is smaller for more efficient simulation schemes. Practically, if scheme 1 gives a RTV which is half that of scheme 2, it would take twice as long to estimate γ within a certain accuracy via scheme 2 than via scheme 1. We introduce the gain of using scheme 1 rather than using scheme 2 as

$$\mathrm{Gain}(\widehat{\gamma}_1, \widehat{\gamma}_2) = \frac{\mathrm{RTV}_2}{\mathrm{RTV}_1} = \frac{\mathrm{RE}_2^2 t_2}{\mathrm{RE}_1^2 t_1} \approx \frac{\mathbb{V}\mathrm{ar}(\widehat{\gamma}_2) t_2}{\mathbb{V}\mathrm{ar}(\widehat{\gamma}_1) t_1}$$

and it is clear that the gain is now equal to the speed-up ratio in simulation time t_2/t_1 for achieving a fixed variance when using estimator $\widehat{\gamma}_1$ instead of $\widehat{\gamma}_2$.

3.8.1 Global and Single Stepping

In the next three models we compare the Stepping implementations as proposed in Section 3.5.

Example 3.1: $M/E_2/1$ **queue**

The first model is an elementary single server system. Customers arrive according to a Poisson process with rate λ. The service times are independent of each other and of the arrival process and have an Erlang$_2$ distribution with parameter μ. (Hence we can view a service time as consisting of two independent phases, each having an exponential distribution with parameter μ.) Define Z_t as the number of clients in the system at time t, and let S_t be phase of the service time of the client in service at time t. We wish to estimate the probability γ that the number of customers in the system reaches some (high) level L during a busy period. With $X := (S_t, Z_t)$ being a Markov process , we have a model that fits the basic splitting method as described in Chapter 2.

The model parameters are $\lambda = 0.76$, $\mu = 3$ and $L = 20$. In view of Remark 3.1, after each successful hit, we save the phase of the current service time rather than the remaining service time. The optimal thresholds and number of splits $n_i, i > 0$ are determined automatically by the pilot run for the GS and FS methods. For the FE method we choose the number of simulations $r_i, i > 0$ equal to n_0. For all simulations we choose $n_0 = 10^6$ in order to fix the simulation execution time at about five minutes. The exact overflow probability can be computed numerically and is included in Table 3.1 for comparison. Note that we will display the results in absolute numbers and in exponential notation, i.e. $7.2e-3 := 7.2 \cdot 10^{-3}$.

We see that both in accuracy and in execution time the FE method improves slightly on the GS and FS methods. The equivalence of the GS and FS method is apparent through the comparable RE's and RTV's. Also, the FA (Fixed

Assignment) approach is more efficient than the RA (Random Assignment) approach.

Example 3.2 : Tandem queue

The second model is a 2-node tandem queue. Customers arrive at the first queue according to a Poisson process with rate λ. The service time of a customer at the first queue is exponential with rate μ_1, independent of the input process and the other service times. The output process of the first queue forms the input process of the second queue. The service time of customer at the second queue is exponential with rate μ_2, also independent of every thing else. This model has received considerable attention in the literature for rare event probability estimation, e.g. in [GHSZ99], [PW89] and [KN99a]. We wish to estimate the probability γ of the event that the number of customers in the the second queue reaches some (high) level L, before the systems empties, starting from an empty system. See Figure 3.7 for a graphical illustration.

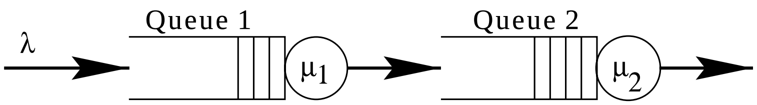

Figure 3.7: Tandem queue

Let Y_t and Z_t be number of clients in the first and the second queue at time t, respectively (including the customers in service). Then $X := (Y_t, Z_t)$ is the underlying Markov process and the model fits the basic splitting method framework.

Method	$\hat{\gamma}$	RE	RTV
Exact	4.720e-8		
GS	4.756e-8	7.2e-3	1.8e-2
FS, FA	4.762e-8	7.2e-3	1.8e-2
FE, FA	4.674e-8	7.1e-3	1.7e-2
FE, RA	4.744e-8	7.6e-3	2.0e-2

Table 3.1: Results for the $M/E_2/1$–queue. The model parameters are $\lambda = 0.76$, $\mu = 3$ and $L = 20$.

		$(\mu_1 = 4, \mu_2 = 2)$			$(\mu_1 = 4/3, \mu_2 = 2)$		
Method	L	$\hat{\gamma}$	RE	RTV	$\hat{\gamma}$	RE	RTV
Exact	20	1.27e-6			3.82e-6		
	60	1.16e-18			3.47e-18		
FE	20	1.278e-6	6.5e-3	8.5e-3	3.804e-6	6.3e-3	1.1e-2
	60	1.174e-18	3.8e-3	1.7e-1	2.909e-18	3.7e-3	1.9e-1
FS	20	1.279e-6	9.5e-3	1.2e-2	3.881e-6	1.4e-2	4.1e-2
	60	1.167e-18	7.1e-3	2.6e-1	3.502e-18	1.9e-1	3.6e2
GS	20	1.270e-6	9.5e-3	1.3e-2	3.870e-6	1.4e-2	4.3e-2
	60	1.155e-18	7.2e-3	2.9e-1	3.125e-18	1.6e-1	2.9e2

Table 3.2: Results for the tandem queue

We compare the FE approach with the FS approach of [GHSZ99]; and we do this for two different cases. In the first case the second server is the bottleneck, and in the second case the first server is the bottleneck. As service rates we use $\mu_1 = 4, \mu_2 = 2$, in the first case, and $\mu_2 = 4/3, \mu_2 = 2$, in the second case. In both cases $\lambda = 1$. Two different levels are considered: $L = 20$ and $L = 60$. The intermediate levels will be chosen automatically via a pilot run, as will be the number of splits $n_i, i > 0$.

The first rows of Table 3.2 contains the exact probabilities, obtained from [GHSZ99]. For each estimate of γ we choose n_0 large enough to obtain relative errors of about 3%; in fact we set $n_0 := 10^6$ for $L = 20$ and $n_0 := 10^7$ for $L = 60$. We use the Fixed Assignment method in all cases; for the cases $L = 60$ a truncation cut-off of five was used to reduce the simulation time.

From Table 3.2, we conclude that also in this case the FE method is more efficient than the FS method. Note that, as observed in [GHSZ99], in the second case, where the first buffer is the bottleneck, the RTV is much higher (for both the FE and FS implementation) than in the first case. The RTV seems to grow quadratically with L. We also note that the efficiency difference between the FE and FS methods in the second case is much larger than can be explained by the "die-out" of simulation paths. We have to conclude that the variance calculation method for the FE method, introduced in Appendix A, which depends on independence assumptions of the simulation stages, is not valid for this particular case.

Example 3.3 : Flow line

The third model deals with a continuous flow line consisting of three machines and two intermediate buffers. Machine $i \in \{1, 2, 3\}$ can process the continuous flow products at some maximum rate ν_i, called the *machine speed*. Moreover, the machines are prone to failure. The life and repair times of the machine i are exponentially distributed with parameters λ_i and μ_i, respectively, and are independent of each other. The buffer capacities are C_1 and C_2. The system is depicted in Figure 3.8 .

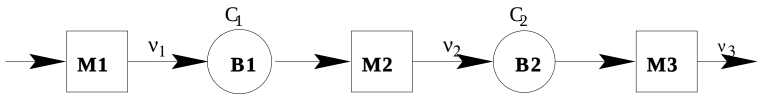

Figure 3.8: Flow line model

This model has been studied in [KN98], where an Importance Sampling procedure was described for the efficient estimation of the overflow probability γ in the second buffer 2 (i.e. the probability that buffer 2 reaches level $L := C_2$ before it empties again, starting from an empty system). The translation into the splitting set-up is the following. Let Y_t and Z_t be the level of the first and second buffer at time t, respectively; and let $M_t \in \{0, 1\}$ denote the state of the machines at time t. Then obviously $X := (M_t, Y_t, Z_t)$ is the underlying Markov process which fits the splitting method framework.

We wish to compare the performance of different splitting methods with that of the Importance Sampling (IS) method in [KN98].

The model parameters are $\nu_1 = 3$, $\nu_2 = 2$, $\nu_3 = 1$; $\lambda_1 = 5$, $\lambda_2 = 2$; $\mu_1 = 1$ $\mu_2 = 1$ and $C_1 = 1$. The third machine is assumed to be perfectly reliable. The IS estimator uses a uniformization rate of 70 (which was found to give the most accurate results) and the number of cycles (runs) is 10^5. The overflow level L is taken to be 3; The intermediate levels for the splitting methods are chosen automatically using a pilot run. The number of starting samples n_0 is always chosen as 10^6; for the standard simulation we set $n_0 = 10^7$ in order to make the simulation run-times approximately equal. The total simulation time for each experiment is about one minute.

Results for the different methods are displayed in Table 3.8.1. Comparing the efficiencies we observe in Table 3.8.1 that for $L = 3$ the Fixed Effort splitting method with Fixed Assignment outperforms the other methods. In implementing both methods, IS and splitting, we found the last method easier to implement, because it does not require parameters that need to be derived analytically, such as the tilting parameter in IS. Another advantage is that split-

Method	$\hat{\gamma}$	RE	RTV
SS	9.600e-6	1.0e-1	5.7e-1
IS	7.848e-6	8.7e-3	2.3e-2
FE,FA	7.811e-6	7.0e-3	3.5e-3
FE,RA	7.741e-6	7.6e-3	4.4e-3
FS	7.828e-6	7.4e-3	3.5e-3
GS	7.730e-6	7.4e-3	3.7e-3

Table 3.3: Results for the flow line model

ting is more robust: changing system parameters in splitting does not involve any change in the algorithm, while the IS method requires a new set of optimal tilting parameters. Note that this argument does not hold when the optimal tilting parameters can be derived within an IS algorithm. However, asymptotically (i.e. in long simulations involving much smaller probabilities) the IS method might still be the best choice, because of its asymptotic optimality.

3.8.2 Truncation and RESTART methods

We now investigate the truncation effects as described in Section 3.4 for the splitting and in Section 3.6.1 for the RESTART implementation. Also, we show that the basic results from Section 2.2 are correct for an $M/M/1$-model.

Example 3.4 : $M/M/1$-queue

We look at a simple $M/M/1$-queueing system. Customers arrive at a queue with exponential independent inter-arrival times with parameter λ. The service time has a negative exponential distribution with parameter μ. We choose $(X_t, t \geq 0)$ as the number of customers in the queue at time t and set $Z := X$ since process X is a one-dimensional Markov process . We wish to estimate the overflow probability as discussed in the basic splitting model: the probability that the number of customers in the queue exceeds level L before the system empties. We choose a load $\rho := \lambda/\mu$ of 0.7 and obtain the following results in Table 3.4.

The results are generated by using $L/5$ thresholds, evenly placed at $5i, i = 1, \ldots, m$. Re-trial parameters are $n_i = 6, i = 1, \ldots, m-1, n_0 = 10^6$, cut-off depth for the employed Global Step method (GS) is set to 4. The column entries are γ for the analytical value of the probability, $\hat{\gamma}$ for the estimate of the probability, RE for the relative error in the estimate (defined as the ratio of

		Optimal GS			Optimal RESTART		
L	γ	$\hat{\gamma}$	RE	RTV	$\hat{\gamma}$	RE	RTV
85	2.920e-14	2.953e-14	1.4e-2	8.4e-2	2.993e-14	1.7e-2	4.9e-2
90	4.907e-15	5.063e-15	1.4e-2	1.0e-1	5.014e-15	1.8e-2	5.6e-2
95	8.248e-16	8.449e-16	1.4e-2	1.1e-1	8.411e-16	1.8e-2	6.2e-2
100	1.386e-16	1.420e-16	1.4e-2	1.2e-1	1.416e-16	1.9e-2	7.0e-2

Table 3.4: Simulation results for the M/M/1 queue for optimal GS and RESTART implementations

the estimated standard deviation of the estimate and the estimate itself), and RTV for the efficiency measure of the estimation (defined as the square of the RE times the simulation run time in seconds).

The results are exactly as predicted, the RE grows linearly in L, the RTV grows quadratically in L and the method is unbiased and efficient. A plot of the RE versus the rarity parameter L can be found in Figure 3.9; a logarithmic plot of RTV versus L in Figure 3.10 shows the quadratic asymptotic efficiency in L.

Looking at the differences between the methods we observe that the GS methods always gets a lower RE as predicted in Section 3.6, but its much greater run-time makes the efficiency of the RESTART method better by a factor of almost 2. Note that the RESTART gain factor (given by RTV_{GS}/RTV_{RT}) is exactly the run-time reduction factor. The gain however is very system dependent, and we will show in Chapter 6 (for steady-state simulation) and in the next example (for the overflow probability in an ATM model) that the gain is reversed. Extension of the RESTART with the truncation method could restore its efficiency back to that of the (optimal) splitting method, as argued in Section 3.6.1.

Example 3.5 : ATM queueing model

We now try to solve a model with a high level of self-similarity, and thus high sample path dependencies to contrast the results from the previous $M/M/1$-example.

This will be accomplished by simulation of an ATM queueing model, with several sources which generate traffic in the form of "cells" as input into a large buffer. Cells are transmitted at a fixed rate on a high-speed link.

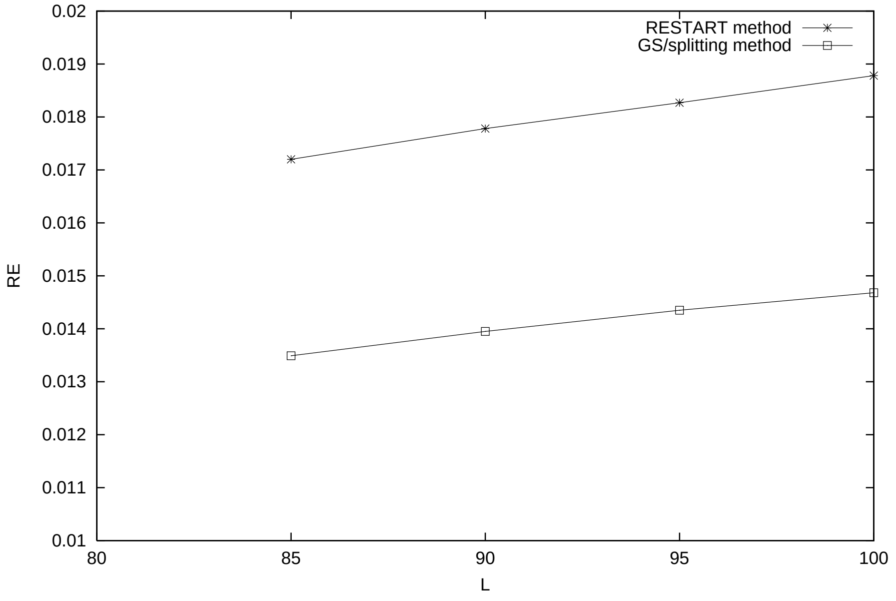

Figure 3.9: *Relative Error Comparison for an M/M/1 queue*

Source model Each source is either in the "on" or "off" state; the holding time in each is Pareto distributed with parameters λ_0 and λ_1, respectively. Note that we use '1' for the on-state and '0' for the off-state. The Pareto distributions are truncated at c_0 and c_1 (in order to make all the estimators converge); and then multiplied by a normalization factor ν_0 and ν_1, respectively. The following are the distribution functions for the on- and off-times, X_1 and X_0, respectively:

$$\mathbb{P}(X_i \leq x) = \begin{cases} \frac{1-(x/\nu_i)^{-\lambda_i}}{1-(c_i/\nu_i)^{-\lambda_i}} & , x < c_i \\ 1 & , x \geq c_i \end{cases}$$

Note that the Hurst parameters for both distributions are obtained as $H_i = (3 - \lambda_i)/2$. When c_i is infinite, the process X_i is asymptotically self-similar with Hurst parameter H_i; when c_i is finite, the autocorrelation function of the process $\{X_i\}_{i=0}^{\infty}$ drops to zero after c_i and the long-range dependency vanishes, see e.g. [GB99]. We have to make sure that c_i is large is enough to preserve adequate self-similarity in our model.

After each "on" holding time, the source will remain on and transmit another

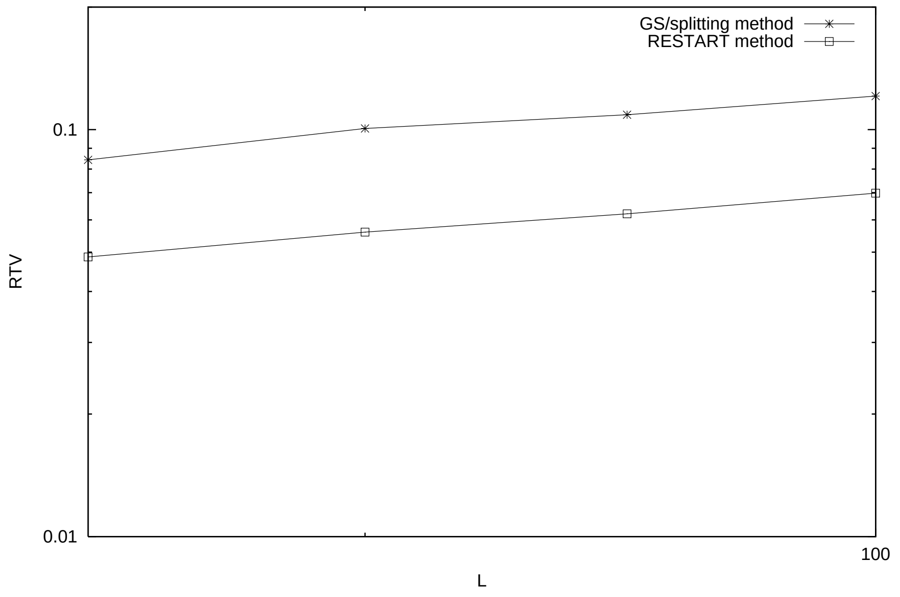

Figure 3.10: *Efficiency Comparison for an M/M/1 queue*

cell with time-independent fixed probability q; with probability $1-q$ it goes into the off state. After each "off" holding time the process will go directly to the on-state. Note that this geometric distribution of the number of cells transmitted during an "on" holding time assures that the effect of the self-similarity on the overflow probabilities is reduced. Denoting $(U_{i,t}, t \geq 0)$ as the process of the state of source i at time t, $(V_{i,t}, t \geq 0)$ as the process of the remaining waiting time in the current state for source i at time t, $(W_t, t \geq 0)$ as the remaining service time of the cell being transmitted at time t, and $(Z_t, t \geq 0)$ as the number of cells waiting in the queue at time t, we present the underlying Markov process X as $(\mathbf{U}_t, \mathbf{V}_t, W_t, Z_t; t \geq 0)$. Note that $\mathbf{U}_t$ and $\mathbf{V}_t$ are the vectors containing the variables $U_{i,t}$ and $V_{i,t}$ for all sources. A visual interpretation of the source model is then given in Figure 3.11.

The probability we will be estimating is the overflow probability of the buffer, i.e. the probability that the buffer becomes full of cells before it empties out, which fits the basic splitting framework. We choose the model parameters as : the Hurst parameters $H_0 = H_1 = H = 0.75$, $q = 0.1$, $\lambda_0 = 1$, $\lambda_1 = 1$, $c_0 = 100$, $c_1 = 100$, $\nu_0 = \nu_1 = 1$, the speed of the ATM transmitter as 100 cells/second and

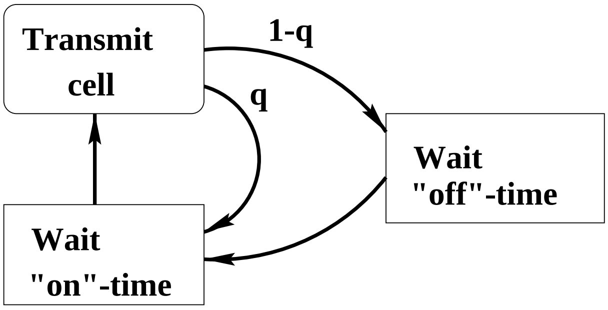

Figure 3.11: *ATM source traffic model*

the number of sources is 20 and obtain Table 3.5 with our splitting/truncation and RESTART/truncation algorithms for a buffer size $L = 50$.

Note that in the first row we denote the original methods, i.e. splitting without truncation and RESTART with the paths truncated at the original splitting threshold. The original splitting method is incorporated in the truncation enhanced splitting method by set the cut-off depth to infinity; the case of a zero cut-off depth for the truncation-enhanced splitting method is not interesting because of the huge bias. The thresholds and number of splits are determined with a pilot-run.

	Optimal GS			Optimal RESTART		
Cut-off	$\hat{\gamma}$	RE	RTV	$\hat{\gamma}$	RE	RTV
$\infty/0$	3.696e-11	1.8e-2	12.8	3.341e-11	2.6e-2	20.1
1	1.029e-12	1.4e2	1.8e7	3.545e-11	2.0e-2	14.6
2	1.341e-11	3.2e-1	306	3.736e-11	1.9e-2	14.0
3	2.498e-11	5.1e-2	24.0	3.631e-11	1.8e-2	13.0
4	3.376e-11	2.4e-2	11.9	3.704e-11	1.7e-2	13.7
5	3.854e-11	1.9e-2	10.7	3.689e-11	1.8e-2	14.5
6	3.696e-11	1.9e-2	9.9	3.754e-11	1.8e-2	15.2
7	3.785e-11	1.8e-2	10.9	3.655e-11	1.8e-2	16.8

Table 3.5: Simulation results for the ATM model

From the results in Table 3.5 we deduce that the splitting method needs a reasonable truncation cut-off depth C in order to be efficient; in this case only $C \geq 4$ produces reliable results. This effect is of course predicted in Section 3.4 ; a small cut-off gives a large bias, and thus a large variance after the bias correction since the estimate of the truncation bias has a variance of the same order as the bias.

For the RESTART implementation we observe that all results seem reliable; however for $C = 0$ (the standard RESTART) the results are the worst. We see that for every cut-off the relative error for RESTART is better than the splitting method. This is evident since no bias correction is performed for RESTART, extra re-splits will ensure the unbiasedness of the algorithm. However, we observe that the optimal efficiency for the RESTART is worse than the optimal splitting efficiency, which can only be explained by the fact that for this model the restoration costs for the system state are larger than the bias correction for simply cutting off instead of re-splitting. Note that this effect will depend on the model simulated; we saw in the previous model an example in which the RESTART algorithm beats the splitting algorithm.

3.9 Conclusions

We have compared several implementations of the splitting method and have found that the original Fixed Splitting implementation, represented by the Single Step Fixed Splitting (FS) and the Global Step (GS) methods, in which each path is split into a fixed number of copies is, in general, not the most efficient one. It is better to fix the total simulation effort at each stage of the simulation: the Fixed Effort (FE) implementation. In this way the number of paths that hit the next level will remain approximately the same, irrespective of how we chose the splitting parameters. On the other hand, the FS and GS implementations are very sensitive to the choice of the splitting parameters. If we do not choose these parameters exactly right, the paths will either "die-out" or "explode", leading to excessive simulation time for problems in which many stages are necessary. A pilot run can be used to choose the splitting parameters (n_i) approximately optimal. For both methods the "optimal" parameters are determined by making the success probabilities in each stage approximately e^{-2}, and the number of trials in each stage equal. The variance calculation of the FE method however depends on an assumption which may not always hold. The other disadvantage of the FE and FS methods is the large memory requirement for accurate estimations which causes us to choose the GS method as the default implementation of the splitting method for accurate estimations.

We also find that it is better, for a given total effort per stage (i.e., the

FE method), to restart an equal number of times from each saved state, rather than to restart each time from a randomly (in this case uniformly) chosen saved state.

The truncation extensions to the splitting and RESTART methods increases the efficiency of both the methods. The optimal value of the cut-off depth is dependent on the simulated model and cannot in general be predicted a priori. A rule of thumb given in Section 3.4 works well for the splitting method; however, for RESTART the optimum is different and the only rule of thumb is to increase the cut-off depth for higher dependency.

Comparing the splitting and RESTART methods there is no clear winner; in low dependency models (e.g., the $M/M/1$-model) the RESTART outperforms the splitting method, in high dependency models (e.g., the ATM model) the splitting model performs better.

We note that if the entrance distribution is (approximately) known, we should sample from this (approximate) distribution, thus rendering the samples independent which reduces the variance of the estimator. The advantage of this approach is investigated in the next Chapter.

Chapter 4

The entrance distribution

4.1 Introduction

In this chapter we explore the possibility of further increasing the efficiency of the method by using the so-called *entrance distributions.* Roughly speaking, the entrance distribution may be seen as the conditional distribution of the 'system state' upon entering a rare event. We will see that the entrance distribution is a natural concept in the context of splitting simulation, at least in the framework that we consider. As a by-product, we show, for certain models, a remarkable relationship between the entrance distribution and the optimal change of measure used in the Importance Sampling method, the other main rare event simulation technique.

We will study the entrance distribution mainly through examples and numerical experiments. It may be argued that the case studies are too specific and can be 'solved' numerically. However, they have served as convenient reference frames (benchmarks) for several simulation methods. For example, in [GHSZ96b], [PW89], [KN98] and [dBNvO98] we find these same cases tested for Importance Sampling estimations and various splitting implementations.

We will find that using the entrance distribution in the splitting method will usually yield a reduction in simulation effort. The standard splitting method is more robust, but is not as efficient as the new method since it does not use any information on the system we are simulating.

The chapter is organized as follows. In Section 4.2 we discuss the entrance distribution and its relevance for splitting simulation. Section 4.3 focuses on the limiting entrance distribution. A number of examples is presented for which this distribution can be easily found. An interesting link with Importance Sampling is pointed out. In Section 4.4 we investigate numerically the usefulness of the entrance distribution in splitting simulation. Conclusions are given in Section 4.5.

4.2 The entrance distribution

In this section we have a closer look at the way in which the simulation is 'restarted' at each level. Recall the following variables from the basic splitting model from Chapter 2.

X	Markov process $(X_t, t \geq 0)$.
Z	Level process $(Z_t := f(X_t), t \geq 0)$.
f	Importance function giving the level for a given system state.
m	Number of stages of splitting.
T_A	Time process Z hits rare event set A, ∞ otherwise.
T_B	Stopping time for processes X and Z; time of hitting set B.
γ	Rare event probability we want to estimate, equals $\mathbb{P}(T_A < T_B)$.
L_k	Threshold k for Z, $k = 1, 2, \dots m$.
T_k	Time of first hit of threshold k; otherwise ∞, $k = 1, 2, \dots, m$.
D_k	Event that threshold k is hit, $k = 1, 2, \dots, m$.
n_k	Number of re-trials from threshold k for every hit, $k = 0, 1, \dots, m-1$.
R_k	Random number of hits of threshold k, $k = 1, 2, \dots, m$.
r_k	Total number of samples in stage k, $k = 0, 1, \dots, m-1$.
S_k	Random system state X_{T_k} upon hitting threshold k.
$I_k^{(j)}$	Bernoulli variable denoting success of re-trial j in stage k.

We define the *entrance distribution* π_k at stage k as the conditional distribution of X_{T_k} given the event D_k, i.e.

$$\pi_k(dx) = \mathbb{P}(X_{T_k} \in dx \,|\, D_k).$$

When X is a multi-dimensional Markov process of the form $(X_t = (Y_t, Z_t), t \geq 0)$, for some process (Y_t), we will identify the entrance distribution with the conditional distribution of Y_{T_k} given D_k.

Now suppose that the entrance distributions $\{\pi_k\}$ are known. Then at each stage k we should start r_k new runs from the corresponding entrance distribution, rather than from the 'saved states' described in Section 2.3.1. Moreover, if we choose the new starting states independently from each other, the indicators $\{I_i^{(k)}\}$ are all completely independent and have expectation p_k. Consequently, we can easily determine the variance of $\hat{\gamma}$ as in Section 2.4.4.

Remark 4.1 Using the exact entrance distributions leads to a smaller variance for $\hat{\gamma}$ as illustrated in the following two-stage splitting simulation example from Section 2.4.3.

Suppose we have R_1 starting states, $S_1^{(1)}, \dots, S_1^{(R_1)}$ say, at the intermediate threshold, each observations of the random variable S_1 being the true entrance distribution. Assume for simplicity that R_1 is *fixed*. We start a total of $r_2 := n_1 R_1$ new runs; n_1 independent paths from each starting state, where $n_1 \geq 1$ is some fixed integer. The success probability of reaching the next level, starting from state s will be denoted by $p_2(s)$. Let Y_i be the number of successful runs (that reach the next level) starting from state $S_1^{(i)}$. Since we have only one intermediate stage, the $S_1^{(i)}$'s are i.i.d., causing all the Bernoullis $I_2^{(i)}$ to have expectation p_2. Consequently, $\hat{p}_2 = \sum_{i=1}^{R_1} Y_i / r_2$ is an unbiased estimator for p_2. We are interested in the variance of $\hat{p}_2$. It is not difficult to see that by (2.19) in Section 2.4.3

$$\mathbb{V}\mathrm{ar}\,(\hat{p}_2) = \frac{1}{r_2}\left\{p_2(1-p_2) + (n_1 - 1)\mathbb{V}\mathrm{ar}\,(p_2(S_1))\right\}. \tag{4.1}$$

The second term between the brackets vanishes when the runs are started from states which are drawn independently from the true entrance distribution, as we will prove now. Suppose in the second stage we start r_2 samples with independently drawn starting states from the true entrance distribution. The Bernoulli variables $\{I_2^{(i)}\}_{i=1}^{r_2}$ are thus independent and identically distributed. This results in

$$\mathbb{V}\mathrm{ar}(\hat{p}_2) = \mathbb{V}\mathrm{ar}(r_2^{-1}\sum_{i=1}^{r_2} I_2^{(i)}) = r_2^{-1}\mathbb{V}\mathrm{ar}(I_2^{(1)}).$$

Observe that $I_2^{(1)} \sim \mathrm{Bin}(1, p_2(S_1))$, the Bernoulli is distributed as a Binomial random variable with one trial and success probability equal to $p_2(S_1)$, which is seen to depend on the starting state S_1. Now we determine the variance of the Bernoulli by conditioning on the starting state and applying (2.4).

$$\begin{aligned}\mathbb{V}\mathrm{ar}(I_2^{(1)}) &= \mathbb{V}\mathrm{ar}(\mathrm{Bin}(1, p_2(S_1))) = \\ &= \mathbb{V}\mathrm{ar}\{\mathbb{E}(\mathrm{Bin}(1, p_2(S_1)) \mid S_1)\} + \mathbb{E}\{\mathbb{V}\mathrm{ar}(\mathrm{Bin}(1, p_2(S_1)) \mid S_1)\} \\ &= \mathbb{V}\mathrm{ar}(p_2(S_1)) + \mathbb{E}(p_2(S_1)(1 - p_2(S_1))) = p_2(1 - p_2) \qquad (4.2)\end{aligned}$$

The difference in variance between the two cases is thus $\frac{n_2-1}{n_2 R_1}\mathbb{V}\mathrm{ar}\,(p_2(S_1)) \approx \mathbb{V}\mathrm{ar}\,(p_2(S_1))/R_1$. This suggest that the increase in efficiency when using the entrance distribution depends very much on the variance of the success probability of reaching the next level.

Next, we give two examples for which the entrance distributions are easily established.

Example 4.1 Consider an M/M/1 queue in a random environment. This model is discussed in detail in Chapter 6 of [Neu81]. Basically, we have an ordinary M/M/1 queue whose arrival rate and service rate are determined by a continuous time Markov process (J_t) with finite state space E and infinitesimal generator (Q-matrix) Q. Specifically, when (J_t) is in state $i \in E$, customers arrive at the queue via a Poisson process with rate λ_i and are served by the server at rate μ_i. Let $\boldsymbol{\lambda}$ and $\boldsymbol{\mu}$ denote the vector of arrival rates and service rates, respectively, and let $\mathbf{q}$ denote the vector of diagonal elements of $-Q$. Let $\boldsymbol{\tau} = \mathbf{q} + \boldsymbol{\mu} + \boldsymbol{\lambda}$. We can interpret $\boldsymbol{\tau}$ as being the vector containing the total 'transition rate' out of each state. Finally, let Z_t be the number of customers in the system at time t.

For any vector $\mathbf{a}$, $\Delta(\mathbf{a})$ denotes the corresponding diagonal matrix. Starting with the buffer at level $Z_0 = z$ and the regulating process in state $J_0 = i$, define T as the first time that the queue either hits level L or becomes empty. For each fixed z, i and L, we are interested in the (entrance) distribution π defined by

$$\pi(\{j\}) = \mathbb{P}(J_T = j \mid Z_T = L,\ J_0 = i,\ Z_0 = z),\ j \in E.$$

We consider thereto the probabilities

$$\mathbb{P}(J_T = j,\ Z_T = L \mid J_0 = i,\ Z_0 = z),\ i, j \in E, \qquad (4.3)$$

which we collect (in the obvious way) into a matrix $P(z)$. To find $P(z)$ we first define the matrices $S_k, k = 1, 2, \ldots,$ such that the (i,j)th element of S_k is the probability of entering level $k+1$ at environment state j before reaching level 0, starting from level k at environment state i. Let S_0 be the 0-matrix, the reader may verify that for $k = 1, 2, \ldots$

$$\Delta(\boldsymbol{\tau})\, S_k = \Delta(\boldsymbol{\lambda}) + (Q + \Delta(\mathbf{q}))\, S_k + \Delta(\boldsymbol{\mu})\, S_{k-1} S_k,$$

It follows that S_k, $k = 1, 2, \ldots$, can be determined recursively from

$$S_k = \left(\Delta(\boldsymbol{\lambda} + \boldsymbol{\mu}) - Q - \Delta(\boldsymbol{\mu})\, S_{k-1}\right)^{-1}\, \Delta(\boldsymbol{\lambda}). \tag{4.4}$$

Moreover, the matrix $P(z)$ is given by

$$P(z) = S_z\, S_{z+1}\, \cdots\, S_{L-1}. \tag{4.5}$$

Finally, observe that the entrance distribution π is simply found from the ith row of the matrix $P(z)$, properly normalized.

Example 4.2 Next, consider as a continuous version of the previous example (also considered in [KN99b]). A fluid queue in which a reservoir, or *buffer*, is filled at rates which vary according to the current state of a continuous time Markov chain (J_t) with finite state space E and Q-matrix Q.

The chain regulates the input to the buffer in such a way that the *input rate* is $r_i \geq 0$ whenever state $i \in E$ is visited. The output rate is constant, c say. Let $\mathbf{r}$ be the vector of input rates, and let $R := \Delta(\mathbf{r})$ denote the corresponding diagonal matrix. Finally, let $E_+ := \{i \in E : r_i > c\}$, $E_- := \{i \in E : r_i < c\}$ and $E_0 := \{i \in E : r_i = c\}$. We assume that $|E_+| > 0$ (otherwise an overflow can never happen.) Let Z_t be the content of the buffer at time t.

Starting with the buffer at level $Z_0 = z$ and the regulating process in state $J_0 = i$, define T as the first time that the buffer either hits level L or becomes empty. We are interested in the probabilities of the form (4.3). which we collect, as before, in a matrix $P(z) = \{p_{ij}(z)\}$. The reader may verify that the matrix $P(z)$ satisfies the following differential equation:

$$(R - c\,I)\, P'(z) = -Q\, P(z), \quad 0 < z \leq L. \tag{4.6}$$

Note that $p_{ij}(z), i \in E_0, j \in E$ can be expressed in terms of $p_{ij}(z), i \in \{E_+ + E_-\}, j \in E$, which are obtained by solving a reduced system of differential equations, with the boundary conditions

$$p_{ij}(0+) = 0,\ i \in E_-, j \in E \ \text{ and } \ p_{ij}(L) = \delta_{ij},\ i \in E_+, j \in E,$$

where δ_{ij} denotes the Dirac function.

The entrance distribution at level L, starting from $J_0 = i$ and $Z_0 = z$, may now be found as the ith row of the matrix $P(z)$, properly normalized.

4.3 Limiting entrance distribution

The two examples in the previous section show that finding the true entrance distribution for each intermediate level may be as difficult as finding the actual overflow probabilities. However, in many cases when the sequence of levels $L_1, L_2, \ldots$ increases to infinity[1], the corresponding sequence of entrance distributions $\{\pi_k\}$ converges (in distribution) to a fixed distribution π_∞, say. Often, this *limiting entrance distribution* is much easier to calculate. Now, if the sequence $\{\pi_k\}$ indeed converges to some π_∞, then for k high enough we could start our splitting simulation from π_∞. This would presumably reduce the variance in the simulation without introducing too much bias.

In some cases we may do even better than this. For example, often the overflow probabilities have an *exponential decay*, i.e. we have

$$\lim_{L\to\infty} \frac{\log \gamma_L}{L} = -\theta, \qquad \text{for some } \theta \geq 0\,, \tag{4.7}$$

where γ_L denotes the overflow probability of level L, i.e. the probability that the process (Z_t) reaches level L before 0. The parameter θ is called the *(exponential) decay rate*; it has been the subject of numerous studies, not only because of its relevance for efficient simulation, but also because it gives important information about the asymptotics of the overflow probabilities and steady-state distributions.

To see how the decay rate is related to the limiting entrance distribution, consider the following situation. Suppose that there exists a level v such that the entrance distribution at any level $w \geq v$ is 'close' to the limiting entrance distribution π_∞. Moreover, suppose that the probability of reaching level $z = w + a$, $(a > 0)$, starting from level w and from the limiting entrance distribution, depends approximately only on a and not on w, and is given by $g(a)$, for some strictly positive function g. We then have

$$\gamma_z \approx \gamma_v\, g(z-v) \approx \gamma_v\, g(w-v)\, g(z-w).$$

This suggests that $g(a) = \mathrm{e}^{-\theta a}$, for some constant $\theta > 0$, and, consequently

$$\gamma_z \approx \gamma_v\, \mathrm{e}^{\theta v}\, \mathrm{e}^{-\theta z}.$$

Hence an accurate estimation of γ_v would automatically yield an equally accurate estimation of γ_z, provided that θ can be calculated. In cases where it is difficult or impossible to calculate θ, an estimator could be constructed using estimations for the overflow probabilities for low to moderate levels.

[1] Here we have assumed that there is no maximum level $L = L_m$.

Next, we give a number of models for which the limiting entrance distribution and the decay rate can be easily derived. These models also show a remarkable duality between the splitting parameters and the Importance Sampling parameters.

Example 4.3 (MMPP/M/1-queue) We return to the queueing system of Example 4.1. The reader may verify that for stable systems, the matrices $\{S_k\}$ converge to a fixed matrix S, say. From (4.4) it is easy to see that S must satisfy the quadratic matrix equation

$$\Delta(\boldsymbol{\mu})\, S^2 + (Q - \Delta(\boldsymbol{\lambda} + \boldsymbol{\mu}))\, S + \Delta(\boldsymbol{\lambda}) = 0. \tag{4.8}$$

We describe next how to efficiently calculate S. Notice that S is closely related to the rate matrix R of [Neu81]. In fact, S and R are *similar* (in matrix jargon).

We may determine S iteratively and then determine the dominant eigenvalue, but we may also combine these two evaluation steps via the following *power method*:

Put $S^{(0)} := \Delta(\mathbf{0})$ (0-matrix) and $\mathbf{w}^{(0)} := \mathbf{1}$. Now define recursively, for $n = 0, 1, \ldots,$

$$\begin{aligned} S^{(n+1)} &:= (\Delta(\boldsymbol{\lambda} + \boldsymbol{\mu}) - Q)^{-1} \left\{ \Delta(\boldsymbol{\mu})\, S^{(n)\,2} + \Delta(\boldsymbol{\lambda}) \right\} \\ \mathbf{u}^{(n+1)} &:= \frac{\mathbf{u}^{(n)}\, S^{(n+1)}}{||\mathbf{u}^{(n)}\, S^{(n+1)}||} \\ x^{(n+1)} &:= ||\, \mathbf{u}^{(n+1)}\, S^{(n+1)}\, ||. \end{aligned}$$

Then the sequence $\{x^{(n)}\}$ converges to the dominant eigenvalue x of S, and $\{\mathbf{u}^{(n)}\}$ to the corresponding eigenvector $\mathbf{u}$.

It turns out that the limiting entrance distribution is the *left* eigenvector corresponding to the largest[2] eigenvalue (η say) of S. Moreover, this η is exactly the *geometric*[3] *decay rate* of the overflow probabilities.

There is an interesting link with the Optimal Exponential Change of Measure (OECM) in Importance Sampling (IS) simulation for this model. Namely, using the theory of Markov additive processes, it can be shown that the OECM is completely specified by η and the *right* eigenvector of S, see [KN99b].

Example 4.4 (Fluid queue) For the fluid queue of Example 4.2 the limiting entrance distribution is found by determining the pair $(\theta, \mathbf{u})$ that satisfies the

[2]Note that S is non-negative matrix; hence the by the Perron-Frobenius Theory S has a maximal real eigenvalue, the so-called *dominant* eigenvalue, with a positive eigenvector.

[3]As opposed to the exponential decay rate θ, given in (4.7). The two are related by $\theta = -\log \eta$.

eigenvalue equation

$$-\mathbf{u}\,Q = \theta\,\mathbf{u}\,(R - c\,I), \tag{4.9}$$

where θ is the smallest strictly positive eigenvalue. The parameter θ is the exponential decay rate of the buffer, and the corresponding left eigenvalue $\mathbf{u}$ (properly normalized) is the limiting entrance distribution.

We have a similar right/left-eigenvector duality between splitting and IS parameters as in the previous example. Namely, the OECM used in an Importance Sampling scheme for this model can be found from two parameters $(\theta, \mathbf{v})$ that satisfy the eigenvalue equation

$$-Q\,\mathbf{v} = \theta\,(R - c\,I)\,\mathbf{v},$$

where θ is the same as above, see [KN99b].

Example 4.5 (M/G/1-queue) Consider an $M/G/1$-queue with traffic intensity ρ less than 1. Let G be the distribution(function) of the service time and let B denote the corresponding the Laplace-Stieltjes transform. The arrival intensity is denoted by λ.

The entrance distribution π_k at level k is identified with the conditional distribution of the remaining service time given that the queue reaches level k before it becomes empty. The following result is from [dBNvO98]. Let $K > 1$ be the solution of the equation

$$B(\lambda(1 - z)) = z.$$

The sequence of entrance distributions $\pi_1, \pi_2, \ldots$ converges to a limiting distribution π_∞ which has probability density h given by

$$h(x) = \frac{\lambda}{1-\rho}\int_x^\infty \left\{ \mathrm{e}^{\lambda(K-1)(t-x)} - 1 \right\} G(dt),\ x \geq 0.$$

We now return to the idea discussed at the beginning of this section to start the splitting simulation from the asymptotic entrance distribution π_∞ for high enough levels, when such a distribution exists. The question is, of course, how to decide when a true entrance distribution π_k is close enough to the asymptotic entrance distribution so that only a minimum bias is introduced in the simulation. We suggest here a heuristic method which appears to work well. Let us call the smallest level from which we can ‘safely’ start our runs using the asymptotic distribution the *asymptotic level* (AL). In other words, AL is the level at which we switch from the standard splitting implementation to the entrance distribution method. In order to specify an AL we need to decide when π_k is close enough to π_∞. Generally, only π_∞ is known, but not the actual π_k's, so we need to estimate the π_k's during the simulation and test whether these empirical distributions are close enough to π_∞. This run-time test can be performed after each splitting stage, when we have collected all the entrance states, say $S^{(1)}, \dots, S^{(n)}$ for the next stage from the successful samples in the current stage. By testing whether the entrance states could have been generated from the asymptotic entrance distribution, we can determine if and when we have reached the asymptotic level.

The easiest way to implement such a test would be to use a standard χ^2-test (see e.g. [Leh97]). Specifically, we partition the state space E into m, say, subsets $E_1, \dots, E_m$ [4].

We now define the number of Observed entrance states in class E_i as

$$O_i = \sum_{j=1}^{n} I_{\{S^{(j)} \in E_i\}}, \qquad i = 1, \dots, m,$$

and the number of expected states as

$$M_i = n\,\pi_\infty(E_i), \qquad i = 1, \dots, m.$$

The test statistic

$$T = \sum_{i=1}^{m} \frac{(M_i - O_i)^2}{M_i},$$

has (asymptotically) a χ^2_{m-1} distribution when the entrance states come from an independent sample from π_∞. Choosing E_i such that $\pi_\infty(E_i) \approx \frac{1}{m}$ usually generates the best results. The value of m should be small enough such that $M_i \geq 5, i = 1, \dots, m$.

Note that this test can also be used to determine whether two consecutive splitting stages could have been generated from the same distribution, which could be another way of testing whether we have reached the asymptotic level, without knowing the actual π_∞. This would also be useful for efficient estimation of decay rates. For certain problems, the Kolmogorov-Smirnov test could also be used, see e.g. [Ser80] p. 57.

4.4 Simulation experiments

To study the effect of using the (limiting) entrance distribution on the efficiency of splitting simulations, we conduct a series of experiments. The implementation

[4] the subscript m has no reference to the one in Section 2.3.1 and in any other section of the chapter.

is described in Appendix D, the description and explanation of the tables is described in Section 3.8.

4.4.1 $M/G/1$-queue

In this section we examine how the splitting estimation of buffer overflow in an $M/G/1$ queue is effected by the use of the limiting entrance distribution. Referring to the general model in Section 2.3.1, let Z_t be the number of customers in the system at time t, and let $X_t = (Z_t, Y_t)$, where Y_t denotes the remaining service time at time t. We assume that $Z_0 = 1$ and that Y_0 is distributed according to the service time distribution function G. We want to estimate the probability γ that Z hits some level L before 0.

Let λ denote the arrival rate. We choose for the service time the Hyper-exponential-2 distribution. In other words,

$$G(x) \quad = \quad p\,(1 - e^{-\mu_1 x}) + (1-p)\,(1 - \mathrm{e}^{-\mu_2 x}).$$

From Example 4.5 we find that by using [dBNvO98] the limiting entrance distribution has a density h, with

$$h(x) = \frac{\lambda\mu_1\mu_2}{\mu_1\mu_2 - \lambda(p\mu_2 + (1-p)\mu_1)} \sum_{i=1}^{2} \frac{p_i(K-1)\lambda}{\mu_i + \lambda(1-K)}\, \mathrm{e}^{-\mu_i x},$$

where $K > 1$ is the unique solution of the equation

$$K = \sum_{i=1}^{2} \frac{p_i\mu_i}{\mu_i + \lambda(1-K)},$$

with $p_1 = p$ and $p_2 = 1-p$.

In order to test the conjecture in Section 4.2 that the effectiveness of the entrance distribution method depends on the variance of the probability of entering the next level, we compare three different sets of parameter values $(\mu_1, \mu_2, p, \lambda)$. Given a specific value of μ_1, we choose $\mu_2 = 2\mu_1$; p is chosen such that the expected service time is 1; and finally λ is chosen such that the geometric decay rate is 0.5. This ensures that the probability of reaching the next level is approximately the same for all cases; we assume that we make the thresholds equidistant.

The simulation machine used in this case was machine two. The number of simulations in each stage (i.e. r_k) is $3{\cdot}10^6$. The levels are $L_k = k{+}1, k = 1, 2, \ldots$.

The simulation results are listed in Table 4.1. We have used the χ^2-heuristic, described in the previous section, to determine from what level onwards we may

		With asympt. entr. dist.			Standard (FE) splitting		
μ_1	L	$\hat{\gamma}$	RE	RTV	$\hat{\gamma}$	RE	RTV
0.55	21	2.905e-7	3.6e-3	7.3e-3	2.854e-7	4.1e-3	9.3e-3
0.55	31	3.010e-10	4.5e-3	1.8e-2	2.858e-10	6.4e-3	3.5e-2
0.55	41	3.101e-13	5.4e-3	3.5e-2	2.902e-13	8.6e-3	8.5e-2
0.60	21	3.283e-7	3.6e-3	7.0e-3	3.280e-7	3.8e-3	7.7e-3
0.60	31	3.393e-10	4.5e-3	1.7e-2	3.390e-10	5.1e-3	2.2e-2
0.60	41	3.492e-13	5.4e-3	3.4e-2	3.525e-13	8.4e-3	7.9e-2
0.75	21	4.155e-7	3.5e-3	6.7e-3	4.166e-7	3.7e-3	7.3e-3
0.75	31	4.251e-10	4.3e-3	1.7e-2	4.281e-10	4.8e-3	1.9e-2
0.75	41	4.343e-13	5.3e-3	3.4e-2	4.354e-13	6.0e-3	4.1e-2

Table 4.1: *Simulation results for the $M/G/1$ queue*

use the asymptotic entrance distribution. A 95% confidence level was used for the test. The Asymptotic Level was in all cases found to be 15.

We note that the use of the entrance distribution gives a relatively significant reduction of simulation effort. We can quantify this by calculating the (limiting) *gain* for the new method, which we define as

$$\text{gain} = \lim_{L\to\infty} \frac{RTV_{old}(L)}{RTV_{new}(L)}.$$

The gain represents the factor by which the simulation time decreases when using the new method instead of the old one. Notice that this gain is dependent on the success probability per stage. However, for the three cases of interest we have chosen the parameters such that these probabilities are practically the same. When we estimate the gain for the three sets of parameter values, using $L = 41$ for each case, we find that the gains are 2.4, 2.3 and 1.2, respectively, i.e. the gain decreases when μ_1 increases. Now, the service time distribution is a mixture of an Exp(μ_1) and an Exp($\mu_2 = 2\mu_1$) distribution, therefore increasing μ_1 decreases the variance in the service time and thus also the variability in the entrance state. This results in less variance in the success probability of reaching the next level. The observed decrease in gain therefore confirms what we predicted in Section 4.2, namely that the reduction in simulation time increases with to the variance in the success probability of reaching the next level.

Remark 4.2 In order to quantify the advantages of the new method for a general $M/G/1$ queue, we chose the remaining service time as our 'system state'. Notice that a more efficient estimation of γ would have been possible in this case by choosing a different system state. In particular, if we choose $X_t = (Z_t, U_t)$, where (U_t) represents the current type of service time distribution ($\text{Exp}(\mu_1)$ or $\text{Exp}(\mu_2)$), we have a saved state equal to the chosen type of service. The memoryless property of the Exponential service-time distribution makes it possible to re-draw a remaining service time for every re-trial of a saved state while fixing the type of service; which gives a considerable variance reduction. In this case the advantage of the new method is considerably less significant, only a gain of a few percent.

4.4.2 Decay Rate Estimation

An alternative approach to use the knowledge of the asymptotic properties of the system into the simulator is to use the geometric decay rate η. The idea is that the overflow probability γ_L of level L is of the form

$$\gamma_L \approx C\,\eta^L = \gamma_{\text{AL}}\,\eta^{L-\text{AL}},$$

for L large enough. Here AL is the Asymptotic Level, and C and η are presumed to be unknown constants. In particular, for the estimate of η we use the estimated probability of reaching the next level starting from the AL using the (estimated) asymptotic entrance distribution. In this case η was estimated by using 20 million samples, starting from level 15 and observing the number of hits of level 16. An estimate for γ_{AL} follows from the same simulation. Our experience is that the number of simulations needed to estimate η accurately enough is substantially higher than the number of simulations in a 'ordinary' splitting stage.

Assuming that the estimators for η and γ_{AL} are independent, we find the relative error for the estimate of γ_L via standard methods. The results have been listed in Table 4.2. We have also included the gain of this method with respect to the method using the asymptotic entrance distribution. Observe that this method yields a noticeable improvement over the asymptotic entrance distribution method, especially for larger values of L.

4.4.3 Markov-modulated $M/M/1$ queue with breakdowns

In this section we consider a particular case of the $M/M/1$-queue in a random environment discussed in Example 4.1 and Example 4.3. We investigate how much we can speed up the splitting estimation of overflow probabilities by using

μ_1	L	$\hat{\gamma}$	RE	RTV	gain
0.55	21	2.912e-7	3.5e-3	8.0e-3	0.9
0.55	31	3.022e-10	4.6e-3	1.3e-2	1.3
0.55	41	3.136e-13	6.3e-3	2.5e-2	1.4
0.60	21	3.293e-7	3.3e-2	6.6e-3	1.1
0.60	31	3.392e-10	4.7e-2	1.3e-2	1.4
0.60	41	3.495e-13	6.6e-3	2.4e-2	1.4
0.75	21	4.168e-7	3.2e-3	5.7e-3	1.2
0.75	31	4.301e-10	4.6e-3	1.2e-2	1.5
0.75	41	4.440e-13	6.5e-3	2.6e-2	1.3

Table 4.2: *Simulation results with Decay Rate Estimation*

the exact entrance distributions at all levels. This gives an idea of the robustness of the normal splitting method. Note that the calculation of *all* entrance distribution is just as involved as the calculation of the overflow probability itself.

The model that we consider is a $M/M/1$ queue whose input process consists of the superposition of ten independent Poisson arrival sources, 5 of type 1 and 5 of type 2. Sources of type i have arrival rate λ_i, $i = 1, 2$. A single server serves the customers at rate μ. However, both the server and the input sources are subject to breakdowns and repairs. We assume that the corresponding up- and down-times are independent of everything else, and are exponentially distributed. The failure rate of the server is γ and the repair rate is δ. For sources of type i the failure rate is α_i and the repair rate is β_i, $i = 1, 2$. Note that this model is a special case of the one described in Example 4.1. We may take the environment process $(N_1(t), N_2(t), M(t))$, where $N_i(t)$ denotes the number of active sources of type i, $i = 1, 2$ and $M(t)$ denotes the state of the server (0 = failed, 1 = working).

The simulation machine used for these results was machine three. For the simulation we have taken the following parameters: $\alpha_1 = 3$, $\alpha_2 = 1$, $\beta_1 = 2$, $\beta_2 = 4$, $r_1 = 3$, $r_2 = 6$, $\gamma = 3$, $\delta = 4$, $\mu = 100$. We have used 53 equal sized stages, with levels $L_i = 3i + 1, i = 1, 2, \ldots, 53$. At each stage 10^6 simulations were performed, and the truncation technique was used to limit simulation time spent on simulating sample paths back to $\{Z = L_0\}$. For each run the starting state for the environment process is (0,1,0), and we start with 1 customer in the queue. We easily find, for example by using the power method from 4.3, that the geometric decay rate for this case is $\eta = 0.836993$. The results for the simulations have been listed in Table 4.3. Note that the true overflow probabilities γ may be obtained by summing the elements of the i-th row of the matrix $P(z)$ of Example 4.1.

	True	With true entr. dist.			Standard (FE) splitting		
L	γ	$\hat{\gamma}$	RE	RTV	$\hat{\gamma}$	RE	RTV
40	4.591e-4	4.62e-4	1.5e-2	4.837e-2	4.35e-4	1.5e-2	4.5e-2
80	3.721e-7	3.75e-7	1.9e-2	1.807e-1	3.65e-7	2.3e-2	2.9e-1
160	2.445e-13	2.32e-13	2.5e-2	6.993e-1	2.45e-13	2.9e-2	1.6e-0

Table 4.3: *Simulation results for the MMPP/M/1 queue with breakdowns*

Note that by using the entrance distributions at all the thresholds we only gain a little in variance, typically up to 10%. The reason for this relatively small gain in variance is that the variance in the success probability of reaching the next level is not very large compared to the success probability itself, i.e. $\mathbb{V}\text{ar}(p_i)/p_i^2 = c_i$ is small and thus also the variance difference between the two methods as given by (4.1) and (4.2). From (4.1) it is obvious that in this particular case the difference in variance between the two methods is small, resulting in this relatively small gain. This is of course very good since we can then obtain near optimal estimations using the plain splitting estimator.

In the RTV comparison we observe a more substantial reduction in simulation effort, mainly because the true entrance distribution is easier to sample and hence uses far less simulation time.

4.4.4 Tandem queue

In this section we consider a test case for many rare event studies: a system of two M/M/1 queues in tandem. Customers arrive at the first queue according to a Poisson process with parameter λ and are served by a server with service rate μ_1. When a customer is finished at the first queue he/she proceeds to the second queue. In the second queue the service rate is μ_2. The maximum number of customers in the first system is n_1 (in other words the maximal queue size is $n_1 - 1$); the second system allows maximally n_2 customers. Our 'system state' is now $X_t := (N_1(t), N_2(t))$, where $N_i(t)$ is the number of customers in the ith system, $i = 1, 2$. The process Z is simply $(N_2(t))$, and we are interested in the probability that the second buffer overflows before it becomes empty, starting with $N_1(0) = 0$ and $N_2(0) = 1$. As in the previous test case, we restart our simulations from the actual entrance distributions, as calculated in described in Appendix B.1. We remark that for this calculation n_1 needs to be finite. However, n_2 may be infinite. For the simulation we chose $\lambda = 3, \mu_1 = 2, \mu_2 = 6$ and $n_1 = 40$. Per stage 10^6 runs were simulated. The levels are $L_i = i + 1, i = 1, \dots L$. The simulation results for various values for $n_2 = L$ are given in Table 4.4. As before, the results for the standard splitting are presented in the left half, and the results for the simulations using the entrance distribution are given in the right half of the table. Note that the true overflow probabilities γ are easily obtained in Appendix B.1.

	True	With true entr.dist.			Standard (FE) splitting		
L	γ	$\hat{\gamma}$	RE	RTV	$\hat{\gamma}$	RE	RTV
10	3.148e-7	3.12e-7	9.3e-3	1.2e-2	3.23e-7	1.2e-2	1.8e-2
20	2.990e-13	2.98e-13	1.2e-2	5.0e-2	2.93e-13	1.7e-2	9.4e-2
30	4.111e-19	4.06e-19	1.4e-2	1.2e-1	4.23e-19	2.0e-2	2.1e-1

Table 4.4: *Simulation results for the tandem queue*

From this table we can determine that the gain from the use of the true entrance distribution is significant in terms of improvement in relative error; the variance growth with the rarity parameter L is considerably slower in the new method, causing a 80% efficiency improvement in the last row, judging from the RTV ratios.

4.5 Conclusions

In this chapter we have looked at several ways to improve the efficiency of the splitting method by using the (limiting) entrance distributions. We have investigated various properties of the entrance distributions and have shown that for certain models they are closely related to Importance Sampling parameters, thus pointing out a relation between the two main rare event simulation methods – splitting and IS.

We have studied a number of test cases. For the $M/G/1$ queue we observed a considerable improvement in efficiency when using the limiting entrance distribution in the splitting method. The gain in efficiency is strongly dependent on the choice of parameters; this dependence could be explained by considering the variance of the success probability of reaching the next level. The larger the variance, the larger the gain. A further improvement could be made in this model by directly estimating the geometric decay rate for the overflow probabilities.

The observed improvement over the old method was small in the second test case where the simulated system appeared to perform well using the standard splitting algorithm. It is clear that the standard splitting method performs very well, and is only beaten when the variance of the success probability over the entrance distribution is relatively large, which was not the case for the second model.

The last case involved the tandem system of $M/M/1$ queues, a well-known reference case for many rare event simulation techniques. Also in this case the simulations were restarted from the true entrance distributions. The outcome of this case was mixed. On the one hand, some increase in efficiency was observed. On the other hand, the computation and implementation of the exact entrance distributions is quite involved, whereas the standard splitting method doesn't require complicated pre-calculations.

Chapter 5

The importance function

5.1 Introduction

Many problems concerning rare events involve estimating overflow probabilities, where we are interested in the probability that some performance measure exceeds a preset limit. A method for estimating these rare overflow probabilities is the *splitting method*. The splitting method re-samples simulation paths that reach certain intermediate thresholds. The placement of these thresholds is crucial to the effectiveness of the method and is governed by the so-called Importance Function (IF). We will show that in the multi-dimensional state space models the "natural" choice of the performance measure as IF does not optimize the simulation effort. We derive expressions for the optimal IF by analysis.

Using the optimal IF we obtain a significant improvement in simulation efficiency, as we will show for the models we investigate.

We will try to enhance the splitting method in order to deal with systems having a multi-dimensional state space. While the method worked well for one dimensional state spaces such as the single server queue, its efficiency is likely to drop to that of the crude Monte Carlo method when the re-sampling thresholds are not chosen in a way that strokes with the paths leading to the rare event set. The *Importance Function* (IF) is the key to efficient simulation in these cases. It may be viewed as a measure of likeliness to hit the rare event set from the current position, and as such governs the re-sampling decision of when to re-sample. The term IF was introduced in [VAMMGFC94]. We will see that only an optimally chosen importance function guarantees an asymptotically efficient simulation strategy. We will study the importance function mainly through analysis; numerical experiments will also be provided to clarify its usefulness.

The chapter is organized as follows. In Section 5.2 we discuss the importance function and its relevance for splitting simulation. We apply the new-found optimal IF into an optimal splitting implementation in Section 5.3. In Section 5.4 we investigate empirically the usefulness of the importance function in splitting simulation. Conclusions are given in Section 5.5.

5.2 The Importance Function

In this section we have a closer look at the way in which the simulation is carried out, and try to optimize the simulation effort by choosing an optimal importance function.

We recall the following definitions from Chapter 2.

X	Markov process $(X_t, t \geq 0)$ with state space E.
Z	Level process $(Z_t := f(X_t), t \geq 0)$ with state space $\mathbb{R}$.
f	Importance function giving the level for a given system state.
m	Number of stages of splitting.
A	Rare event set.
T_A	Time process Z hits rare event set A, ∞ otherwise.
T_B	Stopping time for processes X and Z; time of hitting set B.
γ	Rare event probability we want to estimate, equals $\mathbb{P}(T_A < T_B)$.
L_k	Threshold k for Z, $k = 1, 2, \ldots m$.
T_k	Time of first hit of threshold k; otherwise ∞, $k = 1, 2, \ldots, m$.
D_k	Event that threshold k is hit, $k = 1, 2, \ldots, m$.
n_k	Number of re-trials from threshold k for every hit, $k = 0, 1, \ldots, m-1$.
R_k	Random number of hits of threshold k, $k = 1, 2, \ldots, m$.
r_k	Total number of samples in stage k, $k = 0, 1, \ldots, m-1$.
S_k	Random system state X_{T_k} upon hitting threshold k.
$I_k^{(j)}$	Bernoulli variable denoting success of re-trial j in stage k.

5.2.1 One Dimensional State Space

In a one-dimensional state space Markov process we have the overflow probability $\gamma := \mathbb{P}(T_m < T_B | X_0 = x)$ monotone increasing in x for a skip-free to the right process X. This can be seen easily by the Markov property of the system. Define the random variable M as the maximum state reached before returning to the starting set. Since we are talking here about a one-dimensional system state, we simply have $M \in \mathbb{R}$ and so we can use the binary operators $<$ and $>$ on M. Furthermore, we assume that (X_t) is skip-free to the right, i.e. every threshold is hit on a successful path. It is clear that the rare event set must be of the form $A = \{\max_{t \in [0,T_B]} X_t \geq z\}$ for some z. The overflow probability is easily seen to be equal to

$$\mathbb{P}^*(A|x) := \mathbb{P}(T_m < T_B | X_0 = x) = \mathbb{P}(M \geq z | M \geq x) = \frac{\mathbb{P}(M \geq z)}{\mathbb{P}(M \geq x)},$$

where z is the threshold of the rare event set. We introduced the shorthand notation $\mathbb{P}^*(S)$ as the probability of hitting set $S \subset E$ before the stopping time T_B. It is evident that this overflow probability is increasing monotonically in the starting state x for one-dimensional state space models, and as such the importance function should also be chosen monotonically increasing in x.

Theorem 1 *Any monotone increasing IF is optimal for a skip-free to the right process $(X_t, t \geq 0)$ with a one-dimensional state space.*

Proof. Suppose the optimal success probabilities for this problem are given by $(p_k)_{k=1}^m$ and set the optimal rare event hitting probabilities from stage k thus as $P_k := \prod_{j=k}^m p_j$. We find the optimal event sets D_k as $\{\max_{t \in [0,T_B]} X_t \geq x_k\}$ where x_k is uniquely defined by $x_k := \inf\{x \in \mathbb{R} \,|\, \mathbb{P}(\max_{t \in [0,T_B]} X_t \geq x) \leq \gamma / P_k\}$ since we just found these probabilities to be monotone decreasing in x when using the IF $f(x) = x$, and thus also continuous since we assumed $(X_t, t \geq 0)$ skip-free to the right. Using the continuity we directly obtain the x_k from the distribution function of M. It remains to be shown that we can map these optimal x_k uniquely to optimal thresholds L_k for the process $(Z_t, t \geq 0)$. The event equivalence $D_k = \{\exists 0 < t < T_B \,|\, X_t \geq x_k\} = \{\exists 0 < t < T_B \,|\, f(X_t) \geq f(x_k)\} = \{\exists 0 < t < T_B \,|\, Z_t \geq L_k\}$ is satisfied for every sequence $\{p_1, p_2, \ldots, p_m, m\}$ if and only if f is monotone increasing, since otherwise the transformation of the event set is not unique. Note that we obtain the thresholds simply by putting $L_k := f(x_k)$. $\square$

The choice of the IF only influences the location of the thresholds in the (Z_t)

process; there is no other effect on the splitting process since the thresholds in the (X_t) process remain unchanged by the importance function.

The standard choice for the importance function is $f(x) = x$ for system state $E \subset \mathbb{R}$, simply because there is no gain in another f. We now look at the location of the thresholds for rare event probabilities $\mathbb{P}(M > x)$ converging to exponential decay. Since in the optimal implementation we need to have the success probabilities equal (as discussed in Section 2.4.4), i.e. $p_i = p$ for all i, and suppose L_k is a threshold with large k, such that the exponential decay condition holds, we find the next threshold using $\mathbb{P}(M > L_{k+1}) = p\,\mathbb{P}(M > L_k)$ and thus

$$p = \frac{\mathbb{P}(M > L_{k+1})}{\mathbb{P}(M > L_k)} = \frac{c\,e^{-\theta L_{k+1}}}{c\,e^{-\theta L_k}} = e^{-\theta(L_{k+1}-L_k)}$$

and the threshold L_{k+1} is given by $L_{k+1} = L_k - \log(p)/\theta$. Recursing this equation we find $L_{k+i} = L_k - i\,\log(p)/\theta$ for all $i > 0$, i.e. the thresholds are spaced equidistantly for stages large enough, since this distance is equal to the constant $-\log(p)/\theta$. This formula also shows that for a strong decay (θ large) the thresholds will have to be close together, whereas for a weak decay (θ small) we will see distant asymptotic thresholds.

5.2.2 Multi-dimensional State Space

In the multi-dimensional state space case, usually $E \subset \mathbb{R}^k, k > 1$, the choice of the importance function is not straightforward. The intuitive approach is to describe the rare event set A as $A = \{h(x) > L | x \in E\}$, for a suitable function h, so that it is a valid importance function (as defined in Chapter 2). The function h is then also used as the importance function to determine the intermediate levels: the splitting at threshold $k, k = 1, 2, \ldots, m-1$ will occur when the process $(Z_t, t \geq 0)$ hits level L_k, and since $Z_t = h(X_t)$ it is clear that the function h effectively governs the splitting process.

Choosing the importance function and thus the intermediate levels in this naive fashion will generally not be optimal, since no information regarding the system's behaviour is used in this heuristic. For a one-dimensional state-space E we saw that this simple choice of importance function usually does not matter (see 1); however this simple choice of importance function will usually not lead to efficient simulation in a multi-dimensional state space, as will become apparent in the next example.

Example 5.1 It is easier to visualise the problem using Figure 5.1.

It is clear that we intend to have simulation runs that hit the intermediate level to increase the probability of reaching the target level L. Using

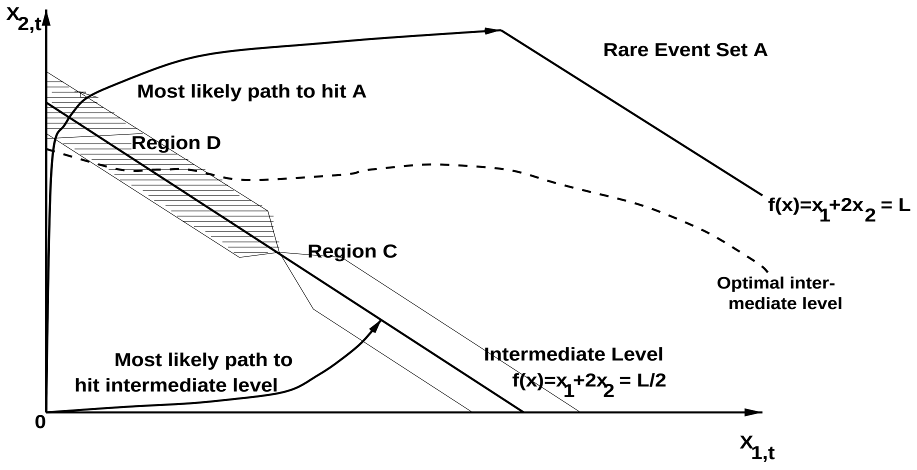

Figure 5.1: *An illustration of the importance function*

the simple importance function will create lots of runs that hit the intermediate level in Region C. However, we want many runs to cross the intermediate level in Region D, because those will have a high probability of actually reaching set A, whereas reaching A from Region C might even be impossible. Defining $\mathbb{P}^*(C)$ as the probability of reaching set C before returning to the starting set, and $\mathbb{P}^*(C|A)$ the probability of reaching set C given that we have reached set A, we analyse an example of how wrong the simple choice of importance function can be. As a numerical example, assume $\mathbb{P}^*(C) = 0.1, \mathbb{P}^*(D) = 0.01, \mathbb{P}^*(A|C) = 10^{-5}, \mathbb{P}^*(A|D) = 0.1$. We see that $\mathbb{P}^*(C|A) \approx 10^{-4}$ and $\mathbb{P}^*(D|A) \approx 10^{-1}$. Clearly region D must be much more important than region C with respect to the rare event. Not honouring this difference in importance between the regions results in having 91% of all runs reaching the intermediate level in region C, and from there almost none reach the rare event set A. To favour region D we need to change the intermediate level, and this is done by changing the importance function f. Suppose the optimal intermediate threshold is represented by the dashed line; it is clear that upon hitting region D we are in or close to the optimal intermediate level, but that it is very hard to reach the optimal intermediate level from the region C. This optimal importance function will therefore favour paths that pass through region D and spend little effort on paths crossing region C, as we would want the optimal simulation to occur. Note that the unbiasedness of the estimator is still preserved regardless of the choice of the intermediate level, or equivalently, the importance function.

It is therefore essential that we construct our simulations using information about the system's behaviour, much as it is commonplace in Importance Sampling, where a lot of attention has been focused on creating paths that are compliant with the large deviations behaviour of the system.

Glasserman et al. [GHSZ99] also look at this problem and prove that in certain cases the simple approach will not yield asymptotically efficient estimates, and will even show apparent bias with high probability. This effect is seen in our Example 5.1 by the large waste of effort on uninteresting samples, resulting in high variance and apparent bias in the estimator.

Remark 5.1 The fact that the optimal importance function leads to a smaller variance for $\hat{\gamma}$ can be illustrated by the following two-stage example from Chapter 3.
Consider a two-stage splitting simulation. Suppose we have R_1 starting states, $S_{1,1}, \ldots, S_{1,R_1}$, at the first stage. Assume for simplicity that R_1 is *fixed.* We start a total of $r_2 := n_2 R_1$ new runs; n_2 independent paths from each starting state, where $n_2 \geq 1$ is some fixed integer. Denote the success of path j originating from saved state $S_{1,i}$ by the indicator variable $I_{i,j}$. The success probability of reaching the next level, starting from state s will be denoted by $p_2(s)$. Since we have only one intermediate stage, the $S_{1,i}$'s are i.i.d., and the expectation of $I_{i,j}$ is p_2. Consequently, $\hat{p}_2 = \sum_{i=1}^{r_2} I_{i,j}/r_2$ is an unbiased estimator for p_2. We are interested in the variance of $\hat{p}_2$. It is not difficult to see ((2.19) in Section 2.4.3) that

$$\mathbb{V}\mathrm{ar}\,(\hat{p}_2) = \frac{1}{r_2}\left\{p_2(1-p_2) + (n_2-1)\mathbb{V}\mathrm{ar}\,(p_2(S_1))\right\}. \tag{5.1}$$

It is clear that the only factor in the variance equation that is dependent on the choice of the importance function is $\mathbb{V}\mathrm{ar}\,(p_2(S_1))$. Note that $\mathbb{E}(p_2(S)) = p_2$ fixed a priori by (2.4). This leads to the optimal choice for f: Choose f so that $\mathbb{P}^*(A|S_1) \approx c$ for a suitable constant c, for every entrance state S_1. Note again that $\mathbb{P}^*(A) = \mathbb{P}(T_m < T_B)$

Lemma 1 *The importance function f for which the success probability is equal over the entrance distribution is optimal in the two-stage splitting simulation in the sense that it minimizes the variance of the estimator $\hat{\gamma}$.*

Proof: Suppose $H := \{x \in E \,|\, \mathbb{P}^*(A|x) = c\}$, the set of states from which the success probability is fixed to some c. Suppose the variable S_1 denotes the entrance distribution for the second stage of splitting using importance function f; it is clear that $\mathcal{S}(S_1) \subset H$ by definition, where $\mathcal{S}(S)$ denotes the sampling set of random variable S. Suppose g is the optimal importance function with entrance state S^*. Since g is optimal, we must have $\mathbb{V}\mathrm{ar}\,(p_2(S^*)) = 0$, by (5.1), which directly gives

$$p_2(y) = \mathbb{E}p_2(S^*) = c \qquad , \qquad \forall y \in \mathcal{S}(S^*) \subset E.$$

It is also clear that $\mathcal{S}(S^*) \subset H$. We will now show that S_1 and S^* are indistinguishable. Clearly $\mathcal{S}(S^*) \subset \mathcal{S}(S_1)$ since $\mathcal{S}(S_1)$ contains all points from H which are reachable. The reverse $\mathcal{S}(S_1) \subset \mathcal{S}(S^*)$ must also be true, since otherwise there are points in $\mathcal{S}(S_1)$ not incorporated in $\mathcal{S}(S^*)$ which correspond to successful trajectories of $(X_t, t \geq 0)$, which would not hit the intermediate threshold with importance function g, contrary to the definition of the importance function (every successful path hits all the intermediate thresholds). Apparently thus the same threshold is used for both f and g, which makes the entrance distributions S_1 and S^* equivalent since they are both just the system state upon hitting the same threshold, and the behaviour of the process is the same for any importance function in the first stage (i.e. before hitting the first threshold). Varying the success probability c gives us the equivalence of the importance functions f and g. $\square$

This two–stage result can be generalised to any number of stages using induction:

Theorem 2 *The importance function f for which the success probability of reaching the highest level is equal over the entrance distribution for all intermediate levels, is optimal in the multi-stage splitting simulation in the sense that it minimizes the variance of the estimator $\hat{\gamma}$*

Proof: Suppose the theorem holds for n stages. We will prove that the case $n+1$ holds as well, proving the theorem by induction combined with Lemma 1. (We only need to prove the Theorem for $n \geq 2$ since the $n = 1$ case is the standard Monte-Carlo simulation method.) We will first show that the first $n-1$ stages are not interesting since they have a fixed success probability in each threshold by induction. The only problem then becomes the choice of the last threshold, which must be also fixing the success probability by applying Lemma 1.

Suppose we have an implementation for n stages with expected success probabilities $\mathbb{E}_{S_i}\{\mathbb{E}(\mathbb{P}^*(A \mid X_0 = S_i))\} = c_i$, keeping the success probability $\mathbb{P}^*(A|x) = c_i$ fixed over the threshold for stage i for any given $1 > c_n > c_{n-1} > \ldots > c_1 > \gamma$. We now want a method that optimizes simulation for any sequence $1 > c^*_{n+1} > c^*_n \ldots > c^*_1 > \gamma$, with the c^*_i the expected probability of hitting A before B starting from threshold i. We will prove that fixing the success probability over the thresholds is optimal.

First of all, the success probabilities must be constants from the first n thresholds. This is clear since otherwise the problem of estimating γ/c^*_{n+1} would be solved by non-constant success probabilities over the first n thresholds, which is contrary to our induction hypothesis.

Since we know c^*_{n+1}, the problem of estimating γ/c^*_{n+1} is equal to the problem of estimating γ and so the intermediate levels are equal for the optimal importance function when using the same $\{c_i\}$ parameters.

For the last threshold, observe that all the stage n paths have a success probability of c^*_n. This makes the simulation up to the n-th stage independent of the last two stages, since the estimators (the number of hits of the next threshold) do not depend on the starting states. The variance in the estimator $\hat{\gamma}$ can then be expressed as

$$\mathbb{V}\mathrm{ar}(\hat{\gamma}) = \mathbb{V}\mathrm{ar}(\widehat{\frac{\gamma}{c^*_n}}\hat{c^*_n}) = (c^*_n)^2\mathbb{V}\mathrm{ar}(\widehat{\frac{\gamma}{c^*_n}}) + (\frac{\gamma}{c^*_n})^2\mathbb{V}\mathrm{ar}(\hat{c^*_n}) + \mathbb{V}\mathrm{ar}(\widehat{\frac{\gamma}{c^*_n}})\mathbb{V}\mathrm{ar}(\hat{c^*_n}).$$

Since the constants are fixed and the estimator for γ/c^*_n has minimal variance, we obtain optimality for $\mathbb{V}\mathrm{ar}(\hat{\gamma})$ using the estimator $\hat{c^*_n}$ which has minimal variance. But now we have a simple two-stage problem again when looking at stage n and $n+1$, which is solved by putting the success probability constant across the threshold to stage $n+1$, by Lemma 1. $\square$

Note that the optimal importance function makes the stages independent as the success or failure of a stage $k+1$ path does not depend on the entrance state produced by its parent path in stage k. Using the results from Section 2.4.4 we find the optimal efficiency for any optimal importance function by (2.27), which is thus the best efficiency that can be achieved by the splitting method.

5.2.3 Discrete State Space Stochastic Processes

So far we have only considered continuous state space stochastic processes simply because then it is clear that the threshold sets $\{x \in E|\mathbb{P}^*(A|x) = c\}$ do exist, are nonempty and are hit by any sample path that reaches the rare event set A. We assume that for a discrete state space stochastic process without loss of generality that $E \subset \mathbb{N}^p$ for some dimension p. Suppose we scale the discrete space stochastic process with the target level $L \in \mathbb{N}$, which we see here as a rarity parameter because it indicates how hard it is to reach set A,

$$X_t^L = X_t/L$$

then the scaled process on sample space $E/L := \{y|L\,y \in E\}$ will usually have a fluid limit for $L \to \infty$ (see e.g. Anantharam et al. [AHT90]) which does possess the properties necessary to construct the optimal importance function. Note that $E/L \subset \mathbb{Q}^p$ [1].

For finite L in a discrete space stochastic process X however we cannot obtain the optimal importance function in an easy way. The problem in this case reduces to the assignment problem of a countable state space, assigning each state to one of the sets $A_i, i = 0, \ldots, m$ (i.e. A_i is the set of points in stage i), while minimizing the variance in the estimator. When all the probabilities are known, this could be solved by a integer program that minimizes the variance in the estimator for a given choice of decision variables x_i, the stage assignment for state i, which should be integer and between 0 and L; i.e. it is a classical Set Partitioning problem which is proven to be hard to solve.

In practice we expect the solution of the proposed program to be inefficient since it will be very complex. Furthermore, it will only work for finite state space, and the gain of the use of the importance function is largest in the asymptotic area, where we can apply the fluid limit approximation.

5.3 Implementation

In this section we pay attention to the various implementation issues that need to be resolved in order to make effective use of the implicit results found in the previous Section 5.2. We will then apply the derived methods in Section 5.4.

5.3.1 Defining the optimal importance function

So far we have only obtained an implicit definition of our optimal importance function; we now want to determine this importance function.

The optimal IF f is constructed such that for a given x

$$P_i \leq \mathbb{P}^*(A|x) \Leftrightarrow i \leq f(x) \tag{5.2}$$

where P_i is the success probability of reaching the rare event set for a path starting at threshold i, in other words:

$$P_i = \prod_{j=i}^{m} p_j.$$

[1] as usual $\mathbb{Q} = \{x/y \,|\, x, y \in \mathbb{Z}, y \neq 0\}$.

When we assume that the process does not cross more than one threshold at a time (which is true for an optimal implementation, see (3.3.2)), we arrive at the following restriction for the optimal IF:

$$P_i \leq \mathbb{P}^* (A|x) < P_{i+1} \Leftrightarrow i \leq f(x) < i + 1. \tag{5.3}$$

Since we want an explicit method for finding the optimal f we restrict it further by enforcing the following simple rules which we found as the optimal implementation techniques as presented in Chapter 2:

1. Make the success probability in each stage equal,
2. Make the number of stages equal to $m = -\log(\gamma)/2$
3. Make the number of samples per stage equal.

It is best to construct the thresholds such that the system actually hits all intermediate thresholds for optimal efficiency. Since we have

$$p_i = p, \qquad i = 1, 2, \ldots, m$$

for the optimal implementation, we obtain $P_i = p^{m-i}$, and since $P_0 = p^m = \gamma$, we obtain $\log(p) = \log(\gamma)/m$ resulting in

$$\gamma^{(m-i)/m} \leq \mathbb{P}^* (A|x) < \gamma^{(m-i+1)/m} \Leftrightarrow i \leq f(x) < i + 1$$

The definition of

$$f(x) = m\big(1 - \frac{\log(\mathbb{P}^*(A|x))}{\log(\gamma)}\big)$$

is easily seen to fulfill the requirement above, as well as making the function f continuous in the case that the success probability is continuous over the state space E. For the optimal m we obtain

$$f(x) = \frac{1}{2} \log \big(\frac{\mathbb{P}^* (A|x)}{\gamma}\big)$$

The remaining problem is that the quantities $\mathbb{P}^* (A \,|\, x)$ and γ are not known, in general, because it is as hard to solve as the original problem (substitute $x = X_0$). Therefore, we look at methods to find estimators for the unknown parameters.

5.3.2 Reverse Time Analysis

Alternatives are either to use a parametric model for the importance function or to use a clever approach to obtain estimates for the rare event probabilities for all the system states. The parametric approach is hard since we already know that the optimal importance function for the two-node tandem queue cannot be captured in a simple parametric function. We will therefore look at the implementation of a reverse time simulation scheme to generate an estimate for the optimal importance function; it is obvious that the reverse time paths from the rare event are statistically indistinguishable from forward time paths that lead to the rare event. We want to investigate whether this observation can benefit the splitting simulation by estimating a better importance function.

It is hard to derive good methods for approximating the optimal importance function because the problem is as hard as estimating the probability we are after in the first place. Therefore we focus on a specific type of system for which we can derive good estimates for the IF, which will feature in our Simulation results section 5.4. [2] In this analysis the simulation system is assumed to be a continuous time Markov chain; this condition is necessary for the following derivations to hold. Specifically, we consider a Jackson network as in [AHT90] of which many properties are known, such as the existence of an asymptotic flow. We assume the system is stable, and that we have the stationary distribution π. We also assume that the process (X_t) is defined for doubly infinite time $(-\infty < t < \infty)$; that the state space E is finite; that we can analyse the process through its discrete skeleton $(X_n, n = \ldots, -1, 0, 1, 2, \ldots)$, and that the chain is recurrent and ergodic.

Next, we will describe the properties of the reverse-time paths which we want to generate samples from. Suppose we look at the discrete time Markov chain X_n in reverse time, denoted by $\tilde{X}_n := X_{-n}$. For a stationary system X_n with stationary probability distribution π and one-step probability matrix P we know (see e.g. [Kel79], [AHT90]) that the process $\tilde{X}_n$ is also a discrete time Markov chain, with the same stationary probability distribution π and one-step probability matrix $\tilde{P}$ defined by

$$\tilde{P}_{ij} = \pi(j)\, P_{ji} \,/\, \pi(i)$$

When we have $P = \tilde{P}$ we call the process *reversible*, i.e., it is statistically indistinguishable from $(X_n)_{n=-\infty}^{\infty}$; otherwise it is just *quasi-reversible*, i.e., the process $(\tilde{X}_n)_{n=-\infty}^{\infty}$ behaves different from the original process $(X_n)_{n=-\infty}^{\infty}$, but it is still a Markov chain. Recall that the rare event set we are trying to reach is A, the starting set is B and we want an estimate of the probability that

[2] The assumptions made here are not essential, as will become apparent later. We do propose to use this framework as it gives good results.

a path starting in B passes through a point $x \in E$, then reaches the set A, all this before returning to the starting set B. Since the chain $(X_n)_{n=-\infty}^{\infty}$ is positive recurrent, we know that we can divide the chain into cycles which start by leaving set B, and that we have countably infinite many of such cycles with a.s. finite length C_i, $i \in \mathbb{N}$ the index of the current cycle. Also the behaviour of $(X_n)_{n=0}^{C_i}$ is independent of the past and of other cycles because the times at which a new cycle starts $\sum_{i=0}^{j} C_i$ are renewal points, i.e. the sequences $(X_i)_{i=C_j}^{C_{j+1}-1}$ are identically distributed and independent. The length of every cycle, $C_i, i = 1, 2, \ldots$, is thus also independent and identically distributed as C_1. We have sets $A \neq \emptyset$ and $B \neq \emptyset$, $A \cap B = \emptyset$, $\pi(A)$ very small, $\pi(B)$ large and a point $x \in E, x \notin A, x \notin B$. Suppose without loss of generality that at $n = 0$ a new cycle starts, i.e., $X_0 \in B$ and $X_1 \notin B$. Set $N_B = \inf\{n > 0 \,|\, X_n \in B\}$, $N_x = \inf\{n > 0 \,|\, X_n = x\}$ and $N_A = \inf\{n > 0 \,|\, X_n \in A\}$. All variables are finite a.s. since $\{X_n\}_{n=-\infty}^{\infty}$ recurrent. Note that $N_B < C_1$ stochastically since the first emergence from set B (at $n = C_1$) must happen after the first return to set B which occurs at $n = N_B$. The behaviour of the system in set B, $(X_n)_{n=N_B}^{C_1}$ is not interesting and we focus on the time before N_B.

The interesting event that x is hit in a cycle before A is hit in the same cycle is $D := \{0 < N_x < N_A < N_B\}$, where the point x is hit on a path from B that will hit set A before the cycle ends. We also introduce $D_x := \{N_x < N_B\}$, the event that point x is reached in a cycle. Note that $D \subset D_x$ and in this notation we have obtained

$$\mathbb{P}^*(A \,|\, x) = \mathbb{P}(D \,|\, D_x) = \frac{\mathbb{P}(D)}{\mathbb{P}(D_x)} \tag{5.4}$$

Also,

$$\mathbb{P}(D) = \mathbb{P}(N_x < N_A \wedge N_A < N_B) = \mathbb{P}(N_x < N_A \,|\, N_A < N_B)\mathbb{P}(N_A < N_B).$$

Now we try to generate a new type of cycle that starts by leaving set A and goes back in time. Again this type of points form regeneration points and countably infinite many cycles $\tilde{C}_j, j \in \mathbb{N}$. We suppose again without loss of generality that the time-reversed process $(\tilde{X}_n)_{n=-\infty}^{\infty}$ starts at time zero in a new cycle. Denote by $(Y_n)_{n=0}^{\tilde{C}}$ a cycle of $(\tilde{X}_n)_{n=-\infty}^{\infty}$ that hits set B at some time during the cycle. The idea is that $(Y_n)_{n=0}^{\tilde{C}}$ represents the successful paths of $(X_n)_{n=-\infty}^{\infty}$, i.e., $(Y_n)_{n=-\infty}^{\infty}$ visits the same states as a forward-time path from $(X_n)_{n=-\infty}^{\infty}$ which will hit set A during a cycle C_1.

For this chain we define $M_x := \inf\{n > 0 \,|\, Y_n = x\}$ and $M_B := \inf\{n > 0 \,|\, Y_n \in B\}$. Both variables are well defined and almost surely finite. The probability of hitting x in a forward-time successful path is the same as the probability of hitting x in a time-reversed successful path as we will prove now.

The probability $\mathbb{P}(D)$ is equal to the probability that both x and A are hit during a cycle C_1. In other words:

$$\mathbb{P}(D) = \int_{\Omega} I_x(\omega) I_A(\omega) \mathrm{d}P(\omega)$$

denoting the possible set of cycles as Ω. When we use the set $\Omega' := \{\omega \in \Omega \,|\, I_A(\omega) = 1\} \subset \Omega$, the set of successful paths that hit A, we can rewrite the probability $\mathbb{P}(D)$ as follows

$$\mathbb{P}(D) = \int_{\Omega'} I_x(\omega) \mathrm{d}P(\omega)$$

The same set Ω' is traversed by the realisations of the reverse time process $(Y_n)_{n=0}^{\tilde{C}}$, thus the following equation holds

$$\mathbb{P}(D) = \int_{\Omega'} I_x(\omega) \mathrm{d}P(\omega) = \int_{\Omega'} I_x(\omega) \frac{\mathrm{d}P}{\mathrm{d}P'}(\omega) \mathrm{d}P'(\omega) = \mathbb{E}_{P'}(I_x L)$$

where P is the probability measure associated with the forward-time process X and P' is the probability measure associated with the reverse-time process Y. The expression L is the so-called "likelihood ratio" of the path and is the conversion factor for the reverse-time probabilities into the forward-time probabilities, i.e. it is the Radon-Nykodym derivative of the forward-time probability measure with respect to the reverse-time probability measure. The derivative exists by the Radon-Nykodym theorem since every path ω which is impossible (i.e. has probability zero) in the reverse time, is also impossible in the forward time.

We now try to find an estimator for $\mathbb{P}(D_x)$. The same approach as before holds, now replacing A with x. The reverse-time paths Y' start in point x and are simulated back to the starting set B. Denoting $\Omega'' = \{\omega \in \Omega | I_x(\omega) = 1\}$ we immediately find the equivalent of the last expression

$$\mathbb{P}(D_x) = \int_{\Omega} I_x(\omega) \mathrm{d}P(\omega) = \int_{\Omega''} \mathrm{d}P(\omega) = \int_{\Omega''} \frac{\mathrm{d}P}{\mathrm{d}P'}(\omega) \mathrm{d}P'(\omega) = \mathbb{E}_{P'}(L')$$

Note that the likelihood ratio L' here is typically much higher than the likelihood ratio L of the paths starting in A since fewer states are visited due to the fact that x is closer to the end set B. We can combine the estimators of $\mathbb{P}(D)$ and $\mathbb{P}(D_x)$ during the same simulation run by estimating

$$\mathbb{P}^*(A|x) = \frac{\mathbb{P}(D)}{\mathbb{P}(D_x)} = \frac{\mathbb{E}_{P'}(I_x L)}{\mathbb{E}_{P'}(L')} = \frac{\mathbb{E}_{P'}(I_x F L')}{\mathbb{E}_{P'}(L')} = \frac{\mathbb{E}_{P'}(I_x F)\mathbb{E}_{P'}(L')}{\mathbb{E}_{P'}(L')} = \mathbb{E}_{P'}(I_x F)$$

where the fraction $F := L/L'$ is exactly equal to the likelihood ratio of the reverse path from A to point x. We can combine the two estimators into one because of the Markov property of the reverse-time path: the history of the process before it hits point x is not important to a simulation of L'. The likelihood ratio of reverse-time paths that pass through x can then be expressed as $L = F \cdot L'$; where F is the likelihood ratio of the path until point x is hit; since after hitting x the reverse-time process behaves as any reverse-time path that started in x, it is clear that the likelihood ratio after hitting x is just a sample of L', and that it is independent of F. Generating paths of (Y_t) thus provides us with a sample of L and L' if the path hits state x, we can thus combine the two samples into one ratio F.

Note that in this discrete case the likelihood fraction is very simple and always equal to $\pi(Y_0)/\pi(x)$ since for every step taken from i to j in the reverse direction the probability is equal to $\tilde{P}_{ij} = \pi(j)P_{ji}/\pi(i)$ and for a path $\omega = \{i_0, i_1, \ldots, i_n\}$ we have

$$\begin{aligned} F(\omega) &= \frac{\mathbb{P}_X(\{i_n, i_{n-1}, \ldots, i_1, i_0\})}{\mathbb{P}_Y(\{i_0, i_1, \ldots, i_{n-1}, i_n\})} \\ &= \frac{P_{i_n,i_{n-1}} P_{i_{n-1},i_{n-2}} \cdots P_{i_1,i_0}}{\tilde{P}_{i_0,i_1} \tilde{P}_{i_1,i_2} \cdots \tilde{P}_{i_{n-1},i_n}} \\ &= \prod_{j=0}^{n-1} \frac{P_{i_j,i_{j+1}}}{\tilde{P}_{i_{j+1},i_j}} = \prod_{j=0}^{n-1} \frac{\pi(i_j)}{\pi(i_{j+1})} = \frac{\pi(i_0)}{\pi(i_n)} = \frac{\pi(Y_0)}{\pi(x)} \end{aligned}$$

Note that this requires the stationary distribution π, which we assumed to be available. This expression gives rise to a simulation scheme which will estimate $\mathbb{P}^*(A|x)$ for all x in E as follows:

0. Set Total Likelihood to zero: $\text{TLR}(x) = 0 \qquad ,\forall x \in E$.

1. Generate N samples of $(Y_n)_{n=0}^{\tilde{C}}$;
   After each step n, where $n = 0, \ldots, \tilde{C}_i)$, of every sample path i, with $i = 1, \ldots, N$,
   increase the total likelihood of the current state $Y_n^{(i)}$, $\text{TLR}(Y_n^{(i)})$, with the likelihood ratio $F(\{Y_k^{(i)}\}_{k=0}^{n})$.

2. Afterwards the estimator for the probability is simply the mean

$$\widehat{\mathbb{P}^*}(A|x) = N^{-1} \sum_{i=1}^{N} F_i(x) I_i(M_x < M_B) = N^{-1}\text{TLR}(x) \quad ,\forall x \in E$$

   Note that we introduced $F_i(x), i = 1, 2, \ldots, N$ as the likelihood ratio obtained in sample path i for point $x \in E$, and also $I_i(S), i = 1, 2, \ldots, N$ is simply the indicator that event S occurs in sample path i.

3. Use this estimator in other procedures as described in Section 5.3 to obtain the importance function value for x.

The whole procedure gives fast estimates for the importance function in the Jackson networks we simulate in Section 5.4. It would be possible to obtain an estimate for γ itself by weighing the estimates generated for the points in set B. In general, this estimate is biased, even for Jackson networks, since the reverse-time procedure needs a finite state space, and a stationary sequence $(X_n)_{n=-\infty}^{\infty}$ in order for the analysis in this Section to hold, whereas in typical problems we are looking at transient probabilities. The bias in the estimator because of the assumption of stationarity not holding is usually small however, and does not affect the estimated importance function much. Note that the actual estimation of γ will be done using this estimated importance function, and might be slightly less optimal due the bias affecting the estimated importance function; however the actual estimation does not suffer from any bias. Another source of bias is incurred by the unknown distribution of the starting state Y_0, or equivalently X_{N_A}, for which a proxy has to be used.

Alternative implementation

Another way of implementing the reverse time path generation algorithm is more complicated but we will describe it briefly. We can also analyse $\mathbb{P}(D)$ as follows

$$\begin{aligned}\mathbb{P}(D) &= \mathbb{P}(N_x < N_A < N_B) = \mathbb{P}(N_x < N_A \,|\, N_A < N_B)\mathbb{P}(N_A < N_B) = \\ &= \gamma\mathbb{P}(N_x < N_A \,|\, N_A < N_B) \qquad (5.5)\end{aligned}$$

i.e. $\mathbb{P}(D)$ can be found as the probability of reaching set A in a cycle, γ, multiplied by the probability that x lies on a successful path. The last factor can then be made equal to the probability that x lies on a reverse-time path

$$\mathbb{P}(N_x < N_A \,|\, N_A < N_B) = \mathbb{P}(M_x < M_B)$$

since for every forward-time path $\omega \in \Omega'$ we have

$$\begin{aligned}N_A(\omega) > n &\Leftrightarrow X_i(\omega) \notin A, i = 0, \dots, n \\ &\Leftrightarrow Y_i(\omega) \notin B, i = 0, \dots, n \\ &\Leftrightarrow M_B(\omega) > n\end{aligned}$$

and thus M_B has the same distribution as N_A on Ω'. Also, the event $\{N_x < N_A\} = \{M_x < M_B\}$ for every $\omega \in \Omega'$, since

$$\begin{aligned}
I_x(\omega) &= I(\exists n < N_A(\omega) \,|\, X_n = x) \\
&= I(N_x(\omega) < N_A(\omega)) \\
&= I(\inf(n > 0 \,|\, X_n = x) < N_A) \\
&= I(\exists n \in [0, N_A] \,|\, X_n = x) \\
&= I(\inf(n > 0 \,|\, Y_n = x) < N_A) \\
&= I(M_x < M_B)
\end{aligned}$$

The reverse paths can thus be used to generate estimations of the probabilities $\mathbb{P}(N_x < N_A \,|\, N_A < N_B)$, and can be combined with standard estimators for γ and $\mathbb{P}(D_x)$ to generate an estimator for $\mathbb{P}^*(A \,|\, x)$ using (5.4) and (5.5).

An estimator for γ is superfluous as we will see in the next section where the probabilities $\mathbb{P}^*(A \,|\, x)$ are divided again by γ when we plug them into the expressions for the optimal importance function. The probability $\mathbb{P}(D_x)$ that x is reached in a cycle can be derived using the identity

$$\pi(x) = \frac{\mathbb{P}(D_x)(1 + \mathbb{E}R_x)}{\mathbb{E}(C_1)}$$

where R_x is the number of returns to point x after the first visit, and C_1 the length of an arbitrary cycle. Both these variables are easy to estimate in forward time, and thus the quantity $\mathbb{P}(D_x)$ can be estimated using only standard methods.

However, we do not propose to use this type of estimator since our previously described estimator is much more elegant and efficient.

The method is limited to discrete Markov chains; a method of implementing it for problems involving a non-countable state space such as most continuous time Markov chains is by "discretising" the state space into a countable number of disjunct sets, using the proposed method to estimate the IF for every such set and then using this set-averaged IF as a proxy for the IF for every point inside the set.

The reverse density

In the description of the reverse-time estimation method it is not necessary that the chosen probabilities are equal to the model's real reverse-time probabilities. Any choice of the probability measure on these paths will generate a correct estimator for $\mathbb{P}^*(A \,|\, x)$, provided that all paths that are possible in the forward time can also be achieved in the reverse model (i.e. no probabilities should be changed to zero). Another choice of probabilities will probably make the estimator better, especially if one fixes the x a priori and then determines the optimal probability change. However, we have not looked at this method since we wanted a general method that requires no further parameter input.

On the positive side, the use of another probability measure for the reverse paths would also alleviate the necessity of the knowledge of the steady-state distribution of X, which is now only used to calculate the reverse-time probabilities. A predetermined or pre-estimated measure could perform well for models for which the steady-state distribution is unknown or too complex.

Example 5.2 We conclude the analysis with an example to visualise the procedure. Say we have a Markov chain consisting of five points in the following Figure 5.2 :

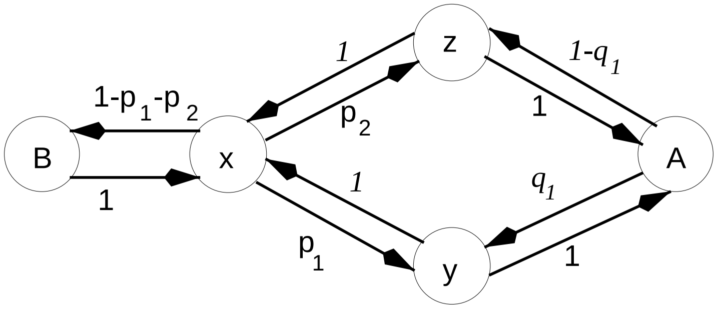

Figure 5.2: *Example Markov chain*

The numbers in sans-serif font denote the forward-time probabilities; the numbers in italic font denote the reverse-time probabilities. It is clear that our goal is to estimate $\mathbb{P}^*(A) = p_1 + p_2$. We observe that there are two paths possible that lead to the rare event, we will denote these as $\omega_1 = (B, x, y, A)$ and $\omega_2 = (B, x, z, A)$. We have $F(\omega_1) = \mathbb{P}_X((x, y, A))/\mathbb{P}_Y(A, y, x) = p_1/q_1$ and $F(\omega_2) = \mathbb{P}_X(x, z, A)/\mathbb{P}_Y(A, z, x) = p_2/(1 - q_1)$. A realisation of the reverse time path will be determined by whether ω_1 or ω_2 is chosen. Say that the number of times we choose ω_1 out of n sample paths is denoted by the random variable Z, which then has a Binomial distribution with parameters n and q_1. The estimator then becomes

$$\begin{aligned}\widehat{\mathbb{P}^*}(A\,|\,x) &= n^{-1}\sum_{i=1}^{N} F(\omega_i) = n^{-1}(Z\frac{p_1}{q_1} + (n - Z)\frac{p_2}{1 - q_1}) \\ &= \frac{p_2}{1 - q_1} + n^{-1}Z(\frac{p_1}{q_1} - \frac{p_2}{1 - q_1})\end{aligned}$$

The expectation of the above expression yields

$$\mathbb{E}(\widehat{\mathbb{P}^*}(A \mid x)) = \frac{p_2}{1-q_1} + q_1(\frac{p_1}{q_1} - \frac{p_2}{1-q_1}) = p_1 + p_2$$

and the algorithm works correctly.

Distribution of Y_0

The last problem we have in the algorithm presented is that we do not always know where to start, i.e. what to choose for the starting state in a sample path $(Y_n^{(i)})_{n=0}^{\tilde{C}_i}$. A sample of $(Y_n)_{n=0}^{\tilde{C}}$ should by definition start from a starting state according to the distribution of $Y_0 = X_{N_A}$, the system state upon entering set A. Any failure to comply to the actual entrance distribution will lead to a bias in the generated estimates. We usually do not know the real distribution of this variable since this is hard to obtain in general. However this is asymptotically not a problem since we know from the existence of an asymptotic flow [AHT90] in the cases we are studying that X_{N_A} has a degenerated asymptotic distribution, i.e. there will be a fixed Y_0 for large values of the rarity parameter.

For real (and finite rarity) problems we propose to choose Y_0 from the steady state probabilities of the border of A, ∂A say, as follows:

$$\mathbb{P}(Y_0 = k) = \frac{\pi(k)}{\pi(\partial A)}, k \in \partial A$$

Another possibility is to generate actual samples of X_{N_A} and use these in a bootstrapping manner. Such samples may be obtained using any splitting method, even for inefficient splitting methods we can usually obtain a few hits of the rare event set.

The fact that this entrance distribution is unknown in general makes the algorithm unfit for estimation of γ in the pilot run because an unknown bias will be introduced by sampling from another distribution than the true entrance distribution.

We will try to find out how close this heuristic is to the optimal function for a number of systems in the following section.

5.4 Simulation Results

See Appendix D for a description of the simulation setup, and Section 3.8 for an explanation of the tables.

The problems we will be looking at are those that have been traditionally ”bad” for the splitting method in the sense that usually the method could not get a reliable estimate.

5.4.1 Two-node tandem queue

A well-known example for this class is the two-node tandem Jackson network which has also been hard to solve using the importance sampling method. (see e.g. [GHSZ99], [GHSZ96b]). The reason for this is obvious: for large buffer sizes we need to create sample paths that concur with the large deviation behaviour of the simulated system. In Appendix B.2 we use a numerical method to obtain the optimal importance function, and we will describe our results with this importance function first. Note that in Figure 5.3 we depict the optimal IF graphically; the contour levels for multiples of five of the IF only are drawn for clarity. These represent the optimal thresholds for the splitting method.

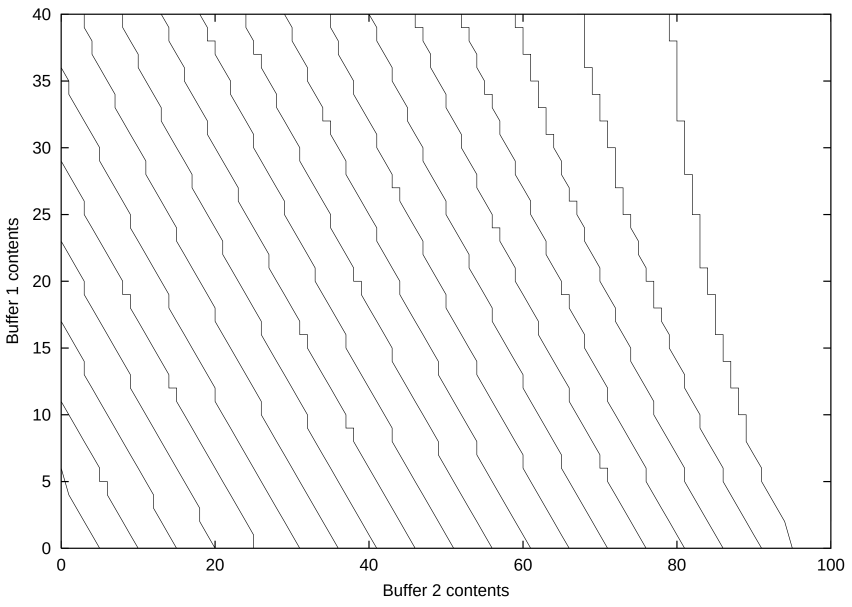

Figure 5.3: *The optimal importance function for the two-node tandem queue*

In [AHT90] Anantharam et al. look specifically at the tandem Jackson network and derive the limiting ”flow” that the successful paths take through reverse–time analysis.

The results prove that a typical path with the first buffer being the bottleneck, leading to overflow of the second buffer, starts with queue 1 building up on its own, and then emptying until buffer 2 hits the overflow level and buffer 1 is empty.

Generating a sample path that has these characteristics is very hard to do in splitting, first the thresholds would have to be linear in the queue 1 contents and later they would have to be compatible with queue 1 emptying into queue 2. A graphical description of these heuristic splitting thresholds can be found in Figure 5.4. The most likely path to hit the rare event set is obtained analytically by reverse-time analysis [AHT90]; the levels have been created by drawing lines perpendicular to the most likely path. It is clear that we can distinguish two different phases in the most likely path and thus also in the splitting optimal thresholds.

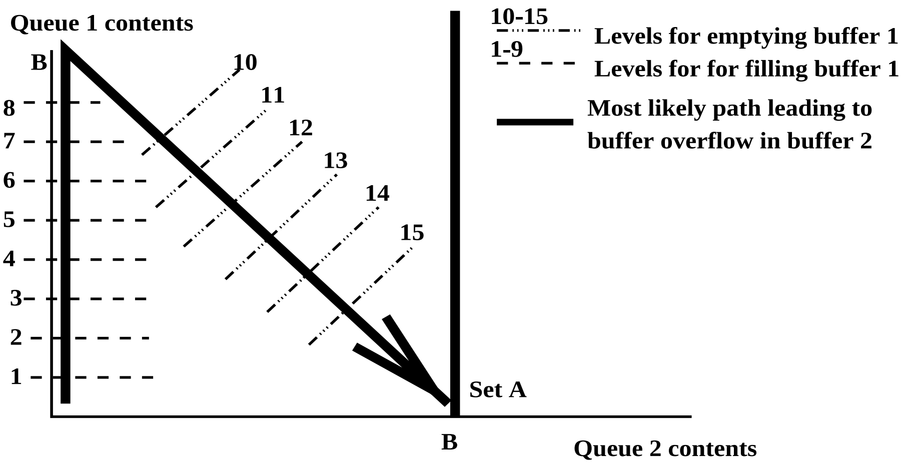
Queue 1 contents
B
8
7
6
5
4
3
2
1
10
11
12
13
14
15
Set A
B
Queue 2 contents
10-15
Levels for emptying buffer 1
1-9
Levels for for filling buffer 1
Most likely path leading to buffer overflow in buffer 2

Figure 5.4: *Asymptotic path to the rare event in the two-node tandem queue*

We derive the optimal importance function numerically for our test case depicted in Figure 5.5. Customers arrive at the first buffer according to a Poisson process with parameter λ. The first buffer has a fixed capacity C, the size of the second buffer is a variable (the rarity parameter L). The servers operate at Poisson rates μ_1 and μ_2 respectively. The underlying Markov process is given by $X = (X_t, t \geq 0)$ where $X_t = (X_{1,t}, X_{2,t})$, in which the stochastic variable $X_{i,t}$ denotes the number of clients in buffer number i at time t. In our simulations we will use $X_0 = (1, 0)$, $C = 40, \lambda = 1$, and $\mu_2 = 3$, leaving μ_1 as a variable. The variable L is the overflow level of the second buffer, the probability estimated is that of the event that the second buffer overflows before it empties.

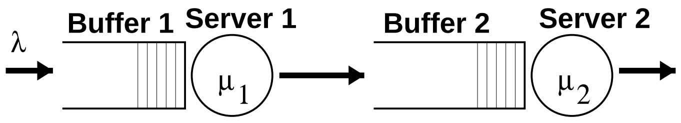

Figure 5.5: *The tandem queue*

First, we look at the results for the estimated importance function as found by the method described in 5.3.2 compared to the standard Global Step splitting results in Table 5.1.

		Estimated IF splitting			Standard (GS) splitting		
μ_1	L	$\hat{\gamma}$	RE	RTV	$\hat{\gamma}$	RE	RTV
2	50	1.050e-26	2.7e-2	1.5e+1	1.166e-26	2.0e-1	2.8e+3
2	75	1.510e-39	1.1e-1	4.0e+2	1.417e-39	1.5e-1	2.7e+3
2	100	2.669e-52	1.0e-1	5.0e+2	2.383e-52	3.9e-1	3.5e+4
3	50	7.185e-25	1.1e-2	2.7e+0	7.035-25	4.0e-2	1.5e+2
3	75	7.208e-37	2.4e-2	2.0e+1	1.023e-36	3.5e-1	2.3e+4
3	100	7.744e-49	6.2e-2	1.4e+2	5.541e-49	1.0e-1	1.9e+3
4	50	1.786e-24	5.5e-3	6.4e-1	1.785e-24	2.7e-2	2.1e+1
4	75	2.134e-36	1.3e-2	7.8e+0	2.005e-36	8.4e-2	4.5e+2
4	100	2.635e-48	2.1e-2	1.1e+1	2.165e-48	4.0e-2	7.8e+1

Table 5.1: *Simulation results for the two-node tandem queue using the estimated IF obtained by time-reversal*

Next, in Table 5.2, we look at the results using the numerical method found in Appendix B.2 for constructing the importance function, and compare these to the known optimal results in order to see how much efficiency we lose in a typical case when using an optimal importance function with splitting method compared to an optimal Importance Sampling scheme as presented in [KN99a].

We interpret these results by looking at the efficiency gain which is the quotient of the RTV's. The efficiency gain is then equal to the ratio of the time

		Optimal IF splitting			Optimal IS		
μ_1	N	$\hat{\gamma}$	RE	RTV	$\hat{\gamma}$	RE	RTV
2	50	1.009e-26	3.0e-2	6.0e+0	1.020e-26	2.7e-2	5.5e-1
2	75	1.399e-39	3.2e-2	1.2e+1	1.416e-39	2.3e-2	6.2e-1
2	100	2.123e-52	5.6e-2	5.1e+1	2.262e-52	3.1e-2	1.4e+0
3	50	7.145e-25	1.6e-2	1.3e+0	7.067e-25	2.7e-3	5.9e-3
3	75	6.951e-37	2.0e-2	2.5e+0	6.867e-37	3.3e-3	1.2e-2
3	100	7.217e-49	2.9e-2	6.0e+0	7.038e-49	3.4e-3	1.8e-2
4	50	1.772e-24	1.1e-2	3.5e-1	1.759e-24	8.8e-4	5.6e-4
4	75	2.070e-36	1.4e-2	1.0e+0	2.060e-36	8.0e-4	6.9e-4
4	100	2.494e-48	2.0e-2	2.4e+0	2.428e-48	7.9e-4	9.1e-4

Table 5.2: *Simulation results for the two-node tandem queue using the optimal IF obtained numerically*

needed by method b to reach a certain fixed accuracy and the time needed by method a to reach that same accuracy. It is thus the simulation time speedup factor of method a over method b. We see that the efficiency gain of the numerically obtained optimal importance function compared to the standard splitting typically lies between 2 to 200 for the simulated tandem queue. Comparing the proposed time-reversal method for the importance function and the standard splitting we observe gains ranging from 2 to about 100, and the gain using the numerically obtained IF over the IF obtained by the time-reversal method is also usually between 2 and 10. This causes us to believe that the proposed estimation method works well but could be improved upon to obtain the optimal gain. One such method could be the diversion of more simulation effort towards the estimation of the optimal importance function; in our test beds this has typically been 10% of the total effort. Comparing the optimal IF to the IS results, we see that the IS gain over the optimal splitting typically ranges from 1 to 10, indicating that this IS approach does a better job. Note however that the RTV seems to grow linearly in the buffer capacity for all cases, indicating that the complexity properties of both methods are equal. A last conclusion is that the RTV decreases dramatically for the IS method for an increasing μ_1, whereas the splitting RTV is very constant for all the optimal splitting implementations. We expected this behaviour for the splitting method as in all cases the optimal efficiency (2.27) is obtained; the remaining variability is caused by the rounding effects of the thresholds and the different costs of simulation for different μ_1. This insensitivity of the efficiency of the optimal splitting method with respect to the model parameters can be seen as a sign of robustness of the optimal splitting method, as defined in [Hub81] page 5.

5.4.2 Three-node tandem queue

The next system we would like to look at is an even harder to solve queueing problem: the problem with three $M/M/1$ queues in series (see Figure 5.6). Since this problem is not solvable with our numeric method, and we do not know of any Importance Sampling methods to generate reliable results, we cannot compare our reverse-time method to another method except the standard Global Step splitting.

The underlying Markov process in this case is $X = (X_t, t \geq 0)$ with a three-dimensional state space: $X_t = (X_{1,t}, X_{2,t}, X_{3,t})$ in which again $X_{i,t}$ denotes the number of clients present in queue number i at time t. We will use the following parameters for the buffer sizes: starting state $X_0 = (1, 1, 0)$, buffer capacities $C_1 = 40$ and $C_2 = 20$. The arrival rate at the first buffer is $\lambda = 1$; the service rates are $\mu_1 = 2, \mu_2 = 2$ and $\mu_3 = 4$. The event we are interested in is the event that the third buffer fills up to at least L customers before it empties out. The simulation setup is again available in Appendix D.

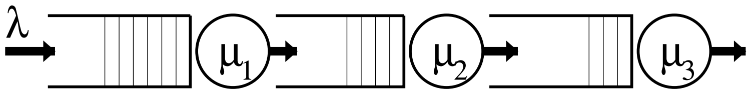

Figure 5.6: *Three node tandem queue*

The state space of this model is much bigger than the previous model, which results in a greater variance of the estimates of the importance function. This increased variance in turn can cause the level process $(Z_t := f(X_t), t \geq 0)$ to jump erratically, which can create a bias in the estimate In order to overcome this effect we used a smoothing technique that reduces the variance in the IF estimates.

The results of the simulations carried out with this system are given in Table 5.3. From these simulations we see that the new method generates more efficient results, the efficiency gain ranges between 10 and 100, just as in the previous simulations with the two-node tandem queue. Typically the gain increases with increased rarity of the rare event set. This behaviour is exactly what we aimed for, since we devised our method especially for those extremely small probabilities. Note that the last result ($L = 20$, standard splitting) does not produce

	With est. import. func.			Standard splitting		
L	$\hat{\gamma}$	RE	RTV	$\hat{\gamma}$	RE	RTV
10	1.188e-7	4.2e-3	5.4e-2	1.182e-7	5.2e-3	4.3e-1
15	2.338e-11	2.9e-2	3.4e-1	2.338e-11	2.9e-2	2.8e+1
20	5.310e-15	2.0e-2	3.8e+0	5.481e-15	1.6e-1	1.4e+3

Table 5.3: *Simulation results for the 3-node tandem queue using the proposed time-reversal method for the IF*

reliable results because the RTV is so large.

5.5 Conclusions

In this chapter we have looked at the optimization of the splitting method with respect to the decision of when to split. This is done by choosing an importance function that gives a certain weight to every system state. The exact optimum is hard to find, since it involves exact knowledge of the quantities to be estimated. A time-reversal method is proposed and evaluated that works well for Jackson networks; efficiency gains typically ranging from 2 to 200 are achieved with this method, shortening the simulation run–time by the same factor. The methodology is simple and adaptable to a broader range of systems; other heuristics for estimating a good importance function may even perform better. Future research will need to determine such heuristics.

Knowledge about the behaviour of the system leading to the rare event is necessary for finding the optimal importance function. Interestingly, the same information is necessary when implementing an optimal Importance Sampling strategy. The optimal importance function will change the complexity properties from a worst case scenario of Monte Carlo complexity of $O(\gamma^{-1})$ (in the case of extremely high dependencies, see Example 2.4) to a complexity of $O((-\log(\gamma))^2)$ (in the case of independence, see Section 2.4.4), making the efficiency of the splitting method a comparable to Importance Sampling. The robustness of the splitting simulation method combined with its complexity properties make it a very attractive rare event simulation method.

Chapter 6

Steady state splitting simulation

6.1 Introduction

The splitting method has mainly been used for the estimation of transient probabilities. Exceptions are found in the work of Kelling [Kel96] concerning Petri nets and the work of Görg [GF99] concerning the tail of delay time distribution of packets in complex communication networks. Steady state simulation offers new possibilities for speed–up especially since the splitting method can be seen as a method only to reach the rare event set; the actual estimation of the time spent there is separated from the splitting procedure itself.

In this chapter we explore the possibility of further increasing the usefulness and efficiency of the method by analysing the so-called *steady state* properties of the investigated system.

We will introduce new estimators for steady state splitting estimation based on the existing optimal transient splitting estimators, and we study their efficiency properties.

The chapter is organised as follows. In Section 6.2 we discuss extending the transient method and its relevance for splitting simulation. We give simulation results obtained with the new techniques in Section 6.3. Conclusions are given in Section 6.4.

6.2 Simple steady state estimators

We recall the following definitions from Chapter 2.

X	Markov process $(X_t, t \geq 0)$ with state space E.
Z	Level process $(Z_t := f(X_t), t \geq 0)$ with state space $\mathbb{R}$.
f	Importance function giving the level for a given system state.
A	Rare event set, equals $\{f(X_t) \geq L\}$ for some $L \in \mathbb{R}$.
γ	Rare event probability we want to estimate, equals $\lim_{t\to\infty} \mathbb{P}(X_t \in A)$.

We will try to estimate the rare event probability γ. To do this we can use the splitting method which can estimate the transient rare event probability that A is hit before some stopping time T_B. We are interested in the possible modifications of the splitting method that will enable the estimation of steady-state rare event probabilities.

For regenerative Markov systems $X = (X_t, t \geq 0)$ we can easily obtain an estimator for the steady-state probability that the level process $(Z_t := f(X_t), t \geq 0)$ is above threshold L, which we denote as the quantity $\gamma_S(L)$, by using the main theorem of regenerative processes :

$$\gamma_S(L) = \lim_{t\to\infty} \mathbb{P}(Z_t > L) = \frac{\mathbb{E}\int_0^C I(Z_t > L)\mathrm{d}t}{\mathbb{E}C} \tag{6.1}$$

where C is the regenerative cycle. The numerator is easily obtained through one of the well-known transient estimators of the probability that threshold L is reached within a regeneration cycle, and the time T_L spent above threshold L, given that we have reached the threshold, as we simply have

$$\begin{aligned} \mathbb{E}\int_0^C I(Z_t > L)\mathrm{d}t &= \mathbb{P}(T_L > 0)\mathbb{E}(T_L \mid T_L > 0) \qquad (6.2) \\ &= \gamma_T(L)\mathbb{E}(T_L \mid T_L > 0) \end{aligned}$$

We used the notation $\gamma_T(L)$ for the estimator of the probability of reaching an importance level L in a cycle C. We visualise the regenerative simulation in Figure 6.1. Note that the splitting, which occurs before threshold L is reached, is omitted since it is only used to “force” a sample path of Z_t towards L and does not interfere with the estimation of T_L.

Denote the estimator for the expected time spent in the rare event set by $\widehat{TC_L}$, and denote the estimator for $\mathbb{E}(T_L|T_L > 0)$ by $\widehat{T_L}$. We find the numerator for our estimator of $\gamma_S(L)$ by (6.1) and (6.2) as

$$\widehat{TC_L} = \widehat{\gamma_T}(L)\,\widehat{T}_L.$$

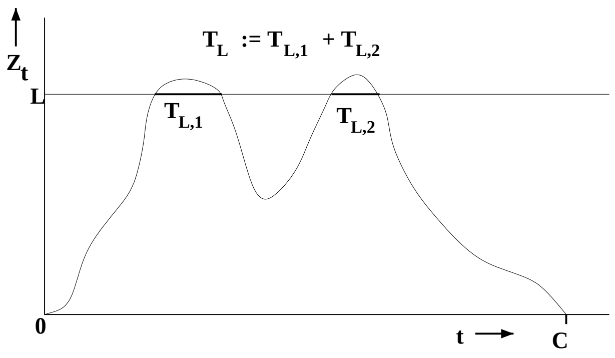

Figure 6.1: *The simple regenerative steady-state estimator*

It is easily seen that this estimator consists of the regular transient estimator $\widehat{\gamma}_T(L)$, multiplied by the time actually spent in the interesting area, exactly the same way as the transient estimator $\widehat{\gamma}_T(L)$ consists of multiplications of conditional estimators. We can therefore view this new estimator $\widehat{TC_L}$ as a transient splitting estimator, just with one more simulation stage in which the time spent above the upper threshold is measured instead of just stopping when hitting the upper threshold as would happen during ordinary transient splitting.

The denominator can be estimated by computing the mean of n samples of crude Monte-Carlo regeneration times. We denote the so-derived estimator as $\bar{C}_n$. Note that the regeneration cycles obtained during the estimation of $\widehat{TC_L}$ are not drawn from the same distribution as the independent regeneration cycles due to the splitting performed. We therefore need to generate these regeneration cycles separately and independently. We propose the following unbiased estimator $\widehat{\gamma_S}(L)$ of $\gamma_S(L)$ which is then equal to

$$\widehat{\gamma_S}(L) := \frac{\widehat{TC_L}}{\bar{C}_n}$$

Note that this estimator is nothing new; it has been proposed in hundreds of books and papers on simulation. We will now prove that this proposed estimator is indeed unbiased and derive its variance. Note that these are known in simulation literature; we will use a slightly different derivation method to arrive at a more direct applicable result. Since we have independence between numerator and denominator we only need to prove that

$$\mathbb{E} \lim_{n\to\infty} \frac{1}{\bar{C}_n} = \frac{1}{\mathbb{E}C}$$

to obtain asymptotic unbiasedness for the steady state estimator. This will of course hold for non-periodic distributions of C since by the Central Limit Theorem we have

$$\bar{C_n} \to N(\mathbb{E}C, \frac{\mathbb{V}\text{ar}(C)}{n})$$

and using [Ser80] p. 118 we obtain

$$\frac{1}{\bar{C_n}} \to N(\frac{1}{\mathbb{E}C}, \frac{\mathbb{V}\text{ar}(C)}{n(\mathbb{E}C)^4})$$

where we have used the symbol $\to$ in the lazy notational sense of $A_n \to N(B_n, C_n^2)$ is equivalent to $\lim_{n\to\infty} \mathbb{P}((A_n - B_n)/C_n \leq x) = \Phi(x), \quad \text{for all}\, x \in \mathbb{R}$, where Φ denotes the distribution function of a standard normal random variable; or in words: the distribution function of $(A_n - B_n)/C_n$ converges point wise to the standard normal distribution.

Combining this with the following theorem for independent asymptotically normal distributed random sequences A_n, B_n from [Ser80] p. 124,

$$A_n \to N(\mu, \frac{\sigma^2}{n}), B_n \to N(\nu, \frac{\tau^2}{n}) \Rightarrow A_nB_n \to N(\mu\nu, \frac{\nu^2\sigma^2 + \mu^2\tau^2}{n}),$$

gives the following expression for the asymptotic relative error for $\widehat{\gamma_S}(L)$:

$$\text{RE}^2(\widehat{\gamma_S}(L)) = \frac{\mathbb{V}\text{ar}(\widehat{\gamma_S}(L))}{\gamma_S^2(L)} = \frac{\mathbb{V}\text{ar}(\widehat{TC_L})}{(\mathbb{E}TC_L)^2} + \frac{\mathbb{V}\text{ar}(\bar{C})}{(\mathbb{E}C)^2} + O(\frac{1}{n}) \qquad (6.3)$$

in which we can use the variance estimators obtained in the simulations of TC_L and C to obtain the accuracy of the final estimator. The order term consists of cross products; we define the order term as usual: $f(x) = O(g(x))$ if and only if $\limsup_{x\to\infty} f(x)/g(x)$ exists.

6.2.1 Enhancements

The simple estimator as presented in the previous section can be improved upon simply by treating the simulation of the time spent above threshold L as a new splitting stage. In other words, we used to have only one observation of T_L for every hit of the threshold, but we can just as easily split the path that hit the threshold into multiple splits, and obtain more paths in the interesting area. Suppose we had observations $\{T^{(1)}, \dots, T^{(n)}\}$ of T_L, then we now replace each of these $T^{(i)}$ with a series of k observations $\{T_1^{(i)}, T_2^{(i)}, \dots, T_k^{(i)}\}$. We visualise this proposed method in Figure 6.2 for $k = 3$.

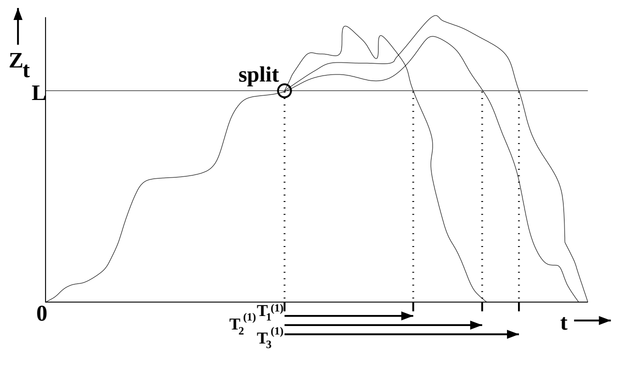

Figure 6.2: *The enhanced regenerative steady-state estimator*

Observe that $\mathbb{E}(T_L) = \mathbb{E}(T_L(S))$, the mean time spent in the rare event depends on the distribution of the entrance state, which we denote here by S. We assume to have obtained S_1 through S_n using ordinary splitting. Note that n, the number of available samples, equals $n = R_m$ in the standard splitting notation of Chapter 2, since this is the number of hits of the uppermost threshold. We now want to optimise the choice of the number of splits upon hitting the rare event set by minimizing the variance of the estimator $\widehat{TC_L}$ for a given effort or simulation budget B, which is linear in n for the standard splitting component contribution and linear in $(n\,k)$ for the contribution of the splitting upon entering the rare event. Denote the random variable describing the time spent in the rare event for a sample with starting (entrance) state S_i as $T(S_i)$; the actual k observations takes from this variable as $\{T_1(S_i), T_2(S_i), \ldots, T_k(S_i)\}$. The variance of the estimator $\widehat{TC_L}$ can be seen to be

$$\begin{aligned}
\mathbb{V}\text{ar}(\widehat{TC_L}) &= \mathbb{V}\text{ar}\Big(\frac{\sum_{i=1}^{n}\sum_{j=1}^{k} T_j(S_i))}{n\,k}\Big) \\
&= \frac{\mathbb{V}\text{ar}(\sum_{j=1}^{k} T_j(S_1))}{n\,k^2} \\
&= \frac{k\mathbb{V}\text{ar}(T_1(S_1)) + k(k-1)\mathbb{C}\text{ov}(T_1(S_1), T_2(S_1))}{n\,k^2} \\
&= \frac{\mathbb{V}\text{ar}(T_1(S_1))}{n\,k} + \frac{(k-1)\mathbb{C}\text{ov}(T_1(S_1), T_2(S_1))}{n\,k} \\
&= \frac{\sigma_T^2}{n\,k} + \frac{(k-1)\sigma_{11}}{n\,k} \qquad (6.4)
\end{aligned}$$

Note that we introduced $\sigma_T^2 := \mathbb{V}\text{ar}(T_1(S_1))$ as the variance of any observation and $\sigma_{11} := \mathbb{C}\mathrm{ov}(T_1(S_1), T_2(S_1))$ as the covariance of observations belonging to the same starting (entrance) state. We introduce C_R as the cost of obtaining one sample of the entrance distribution S, and C_T as the cost of obtaining one observation of the time spent in the rare event. The necessary budget constraint is then that the total cost, consisting of the cost of obtaining samples of S and the cost of obtaining samples of T, must be less or equal to budget B. The problem can then be approximated as

$$\text{Min} \quad \mathbb{V}\mathrm{ar}(\widehat{TC_L}) \quad = \quad \frac{\sigma_T^2}{n\,k} + \frac{\sigma_{11}}{n} \tag{6.5}$$

$$\text{s.t.} \begin{cases} C_R\, n + C_T\, n\, k \le B \\ n \ge 1, n \in \mathbb{N} \\ k \ge 1, k \in \mathbb{N} \end{cases}$$

The optimisation problem can only be solved using numerical algorithms. In order to show an approximation of the optimal solution we demonstrate the results from a Lagrange relaxation of the problem in which we drop the last two constraints. The resulting problem is solved using Lagrange multipliers by putting

$$\begin{cases} k^* &= \sqrt{\dfrac{\sigma_T^2 C_R}{\sigma_{11} C_T}} \\ n^* &= \dfrac{B}{k^*\, C_T + C_R} \end{cases}$$

To obtain an estimate of the optimal k for the original problem (6.5), one could search for the optimum near the solution k^* of the relaxed problem. Note that k^* is independent of the budget B. However, it does depend on L, the rarity parameter because the constants C_R and C_T do. In an optimal implementation, the amount of work to obtain one entrance state into set A will be linear in L (see Section 2.4.4) simply because the number of stages will be linear in L. This translates as $C_R = L\, c_r$, with c_r a constant cost per entrance state per stage. The amount of work to obtain one observation of the time spent in the rare event will be fixed however, thus making $k^* = k_s \sqrt{L}$, where $k_s = \sqrt{\sigma_T^2\, c_r / \sigma_{11} C_T}$ is a system dependent constant. Note that in an optimal implementation the covariance σ_{11} and variance σ_T^2 are constants. Since the approximation for k^* is only valid for large k (we assumed $k \gg 1$) we will look at a special case where we know $k^* = 1$.

Example 6.1 Suppose the time to be spent in the rare event is fixed upon entering the rare event set, in other words $T(S)$ is deterministic in S. This translates as $\sigma_{11} = \sigma_T^2$, and the approximation would give $k^* = \sqrt{C_R/C_T}$, whilst common sense suggests that $k^* = 1$ (there cannot be any gain by repeating the simulation of T when it is fixed). When substituting $\sigma_{11} = \sigma_T^2$ into (6.4), we obtain

$$\mathbb{V}\mathrm{ar}(\widehat{TC_L}) = \frac{\sigma_T^2}{n}$$

which is independent of k.

In general however, we have entrance states $(S_i)_{i=1}^n$ which are correlated since the simulation paths responsible for those paths usually share a common history which can affect the system state. This causes the vector of random variables $\{T_j(S_i)\}$ to be correlated not only within the groups belonging to the same starting state S_i but with other groups as well. We denote this covariance as $\sigma_{12} := \mathbb{C}\mathrm{ov}(T_1(S_1), T_1(S_2))$. This would lead to an extra variance term for the estimator $\widehat{TC_L}$ of

$$\frac{\mathbb{C}\mathrm{ov}(T(S_1), T(S_2))(n-1)}{n\,k} = \frac{(n-1)\sigma_{12}}{n\,k}$$

which leads to a quadratic system for the optimal solution of the Lagrange-relaxed problem as follows (assuming n large so $n - 1 \approx n$):

$$\begin{cases} 0 = & \{\sigma_{12}(1 - C_T)\}(n^*)^2 + \quad \{\sigma_T^2 + C_R\sigma_{11}\} n^* - B\sigma_{11} \\ k^* = & \dfrac{B}{C_T n^*} + \dfrac{C_R}{C_T} \end{cases}$$

This example serves to show that for realistic problems k^* does depend on the available budget B: clearly n^* grows linearly in $\sqrt{B}$, making k^* also linear in $\sqrt{B}$.

Another possible enhancement of the re-sampling scheme is by allowing $k \in \mathbb{N}$ to be a random variable. From the previous optimizations it is logical to choose the expectation as $\mathbb{E}k = k^*$. a possible choice for the distribution of random variable k itself would then be in the set $\{\lfloor \mathbb{E}k \rfloor, \lceil \mathbb{E}k \rceil\}$, making sure that k remains close to the optimal k^*, and thus generating as little variance as possible.

6.2.2 Bootstrapping equivalent

The problem of estimating $\theta = \mathbb{E}(T_L(S))$ with only given (and dependent) observations $S_1, \ldots, S_n$ is a classical bootstrapping problem [Ded90]. Note that since S is the entrance distribution for the successful paths we in fact have

$\theta = \mathbb{E}(T_L(S)) = \mathbb{E}(T_L \mid T_L > 0)$. As in generalized bootstrapping we estimate θ by drawing samples from the empirical entrance distribution of S, which we will denote as $\hat{F}_n$. The proposed method draws each entrance state exactly k times; the bootstrap method allows for more random sampling as we explain below.

Random assignment of entrance state

Instead of fixing the entrance state an alternative is to draw the entrance state $(n\,k)$ times independently from $\hat{F}_n$, as is customary in the classical bootstrap. We will show that this method has a larger variance than the proposed method of distributing the samples equally over the entrance states, just like we found this to be true for the Single Step implementation of the basic splitting algorithm in section 3.5.1. We will denote the previously described method where the starting states are assigned in a fixed manner as the *Fixed Assignment* (FA) method; this bootstrapping method where the starting states are assigned randomly as the *Random Assignment* (RA) method. Denoting $Z_i, i = 1, \dots, n$ as the number of samples of T drawn from S_i, observe that the vector Z has a multinomial distribution with the parameters probability vector equal to $(1/n, \dots, 1/n)$ and number of samples equal to (nk), making $\mathbb{E}(Z_i) = k$, $\mathbb{V}\mathrm{ar}(Z_i) = k(1 - n^{-1})$ and $\mathbb{C}\mathrm{ov}(Z_i, Z_j) = -kn^{-1} \quad (i \neq j)$. We will as before use the symbols $\sigma_T^2 = \mathbb{V}\mathrm{ar}(T_1(S_1))$, $\sigma_{11} = \mathbb{C}\mathrm{ov}(T_1(S_1), T_2(S_1))$ and $\sigma_{12} = \mathbb{C}\mathrm{ov}(T_1(S_1), T_1(S_2))$.

The estimator for $\theta = \mathbb{E}(T_L \mid T_L > 0) = \mathbb{E}(T_L(S))$ then becomes

$$\hat{\theta} = (nk)^{-1} \sum_{i=1}^{n} \sum_{j=1}^{Z_i} T_j(S_i).$$

Aggregate the observations per entrance state as $T(S_i, Z_i) := \sum_{j=1}^{Z_i} T_j(S_i)$; we see the variance in the estimator is equal to

$$\begin{aligned}(nk)^2\, \mathbb{V}\mathrm{ar}(\hat{\theta}) &= \mathbb{V}\mathrm{ar} \sum_{i=1}^{n} \sum_{j=1}^{Z_i} T_j(S_i) = \mathbb{V}\mathrm{ar} \sum_{i=1}^{n} T(S_i, Z_i) \\ &= n\mathbb{V}\mathrm{ar}(T(S_1, Z_1)) + n(n-1)\mathbb{C}\mathrm{ov}(T(S_1, Z_1), T(S_2, Z_2))\end{aligned}$$

and we have

$$\begin{aligned}\mathbb{V}\mathrm{ar}(T(S_1, Z_1)) &= \mathbb{E}_Z(\mathbb{V}\mathrm{ar}(T(S_1, Z_1)|Z_1)) + \mathbb{V}\mathrm{ar}_Z(\mathbb{E}(T(S_1, Z_1)|Z_1)) \\ &= \mathbb{E}_Z\{Z_1 \mathbb{V}\mathrm{ar}(T_1(S_1)) + Z_1(Z_1 - 1)\mathbb{C}\mathrm{ov}(T_1(S_1), T_2(S_1))\} + \\ &\quad + \mathbb{V}\mathrm{ar}_Z(Z_1\theta) \\ &= k\sigma_T^2 + k(k - n^{-1})\sigma_{11} + k(1 - n^{-1})\theta^2\end{aligned}$$

and

$$\begin{aligned}
\mathbb{E}(T(S_1,Z_1)) &= \mathbb{E}_Z(\mathbb{E}(T(S_1,Z_1)|Z_1)) = \mathbb{E}_Z Z_1\theta = k\theta \\
\mathbb{E}(T(S_1,Z_1)T(S_2,Z_2)) &= \mathbb{E}_Z(\mathbb{E}(T(S_1,Z_1)T(S_2,Z_2)|Z_1,Z_2)) \\
&= \mathbb{E}_Z\mathbb{E}\Big\{\sum_{i=1}^{Z_1}\sum_{j=1}^{Z_2} T_i(S_1)T_j(S_2)\,|\,Z_1,Z_2\Big\} \\
&= \mathbb{E}_Z(Z_1Z_2)\mathbb{E}(T_1(S_1)T_1(S_2)) \\
&= (k^2-kn^{-1})(\theta^2+\sigma_{12}) \\
\mathbb{C}\mathrm{ov}(T(S_1,Z_1),T(S_2,Z_2)) &= \mathbb{E}T(S_1,Z_1)T(S_2,Z_2) - (\mathbb{E}(T(S_1,Z_1))^2 \\
&= (k^2-kn^{-1})\sigma_{12} - kn^{-1}\theta^2
\end{aligned}$$

Combining all results we obtain for the Random Assignment (RA)

$$\begin{aligned}
\mathbb{V}\mathrm{ar}(\hat\theta_{RA}) &= (nk)^{-2}\{n(k\sigma_T^2 + k(k-n^{-1})\sigma_{11} + k(1-n^{-1})\theta^2) \\
&\quad + n(n-1)((k^2-kn^{-1})\sigma_{12} - kn^{-1}\theta^2)\} \\
&= (nk)^{-1}\{\sigma_T^2 + (k-n^{-1})\sigma_{11} + (1-n^{-1})\theta^2 + \\
&\quad + (n-1)((k-n^{-1})\sigma_{12} - n^{-1}\theta^2)\} \\
&= (nk)^{-1}\{\sigma_T^2 + (k-n^{-1})\sigma_{11} + (n-1)(k-n^{-1})\sigma_{12}\}
\end{aligned}$$

For the standard Fixed Assignment (FA) $Z_i := k \quad, i = 1,2,\ldots,n$, we have a variance of

$$\begin{aligned}
\mathbb{V}\mathrm{ar}(\hat\theta_{FA}) &= (nk)^{-2}\mathbb{V}\mathrm{ar}(\sum_{i=1}^{n}\sum_{j=1}^{k} T_j(S_i)) \\
&= (nk)^{-2}(nk\sigma_T^2 + nk(k-1)\sigma_{11} + n(n-1)k^2\sigma_{12}) \\
&= (nk)^{-1}(\sigma_T^2 + (k-1)\sigma_{11} + (n-1)k\sigma_{12})
\end{aligned}$$

thus for the variance difference we have

$$\mathbb{V}\mathrm{ar}(\hat\theta_{RA}) - \mathbb{V}\mathrm{ar}(\hat\theta_{FA}) = (nk)^{-1}(1-n^{-1})(\sigma_{11}-\sigma_{12}) \geq 0$$

since we have $k \geq 1, n \geq 1$ and $\sigma_{11} \geq 0$ as $\sigma_{11} = \mathbb{C}\mathrm{ov}(T_1(S_1), T_2(S_1)) = \mathbb{E}_{S_1}T_1(S_1)T_2(S_1) - (\mathbb{E}_{S_1}T_1(S_1))^2 = \mathbb{V}\mathrm{ar}_{S_1}(\mathbb{E}(T_1(S_1)\,|\,S_1))$. Also, since we have

$$k^{-2}\mathbb{V}\mathrm{ar}(\sum_{i=1}^{k}\{T_i(S_1) - T_i(S_2)\}) = 2k^{-1}\sigma_T^2 + 2(1-k^{-1})\sigma_{11} - 2\sigma_{12} \geq 0$$

and letting $k \to \infty$ in this equation we must conclude that $\sigma_{11} \geq \sigma_{12}$ and thus that the Fixed Assignment superior to the Random Assignment.

6.3 Simulation results

In order to quantify the benefits of our proposed methods and techniques we checked the efficiency of old and new methods on various examples. The simulation setup is further described in Appendix D. For the explanation of the tables we refer to Section 3.8.

6.3.1 $M/M/1$ queue

The first example is mainly for verification of the implemented algorithms. We consider a typical $M/M/1$ system, with Poisson arrivals at a rate $\lambda = 0.7$ and service time (exponentially distributed) with parameter $\mu = 1$. The underlying Markov process is $X = (X_t, t \geq 0)$ where X_t denotes the number of customers in the queue at time t. For this system we have $\mathbb{P}(X \geq L) = (\lambda/\mu)^L$, which is easily obtained from the balance equations for the embedded Markov chain. In our simulation we chose $L = 85$ so that for this system we have $\gamma \approx 6.812 \cdot 10^{-14}$. We will demonstrate the efficiency of both the RESTART and splitting methods for various values of the number of re-trials k, and the efficiency gains and losses (denoted as negative gains) between the various implementations in Table 6.1. The estimates themselves are superfluous for the efficiency comparison but are necessary to verify the correctness of the implementation.

	Standard RESTART			
k	$\hat{\gamma}$	RE	RTV	% gain
1	6.968e-14	1.817e-2	9.258e-2	-
2	7.067e-14	1.774e-2	9.169e-2	0.96
3	6.901e-14	1.728e-2	9.438e-2	-1.94
4	6.888e-14	1.769e-2	1.009e-1	-8.99
5	6.781e-14	1.773e-2	1.026e-1	-10.8
6	6.773e-14	1.780e-2	1.076e-1	-16.2
7	6.900e-14	1.775e-2	1.110e-1	-19.9
8	6.830e-14	1.746e-2	1.100e-1	-18.8
9	6.908e-14	1.762e-2	1.169e-1	-26.3
10	6.879e-14	1.764e-2	1.213e-1	-31.0

Table 6.1: *Simulation results for RESTART simulation of the M/M/1 queue*

	Standard FS			
k	$\hat{\gamma}$	RE	RTV	% gain
1	6.982e-14	1.469e-2	1.760e-1	-
2	7.017e-14	1.421e-2	1.655e-1	5.97
3	6.971e-14	1.409e-2	1.651e-1	6.19
4	6.878e-14	1.410e-2	1.730e-1	1.70
5	6.935e-14	1.391e-2	1.693e-1	3.81
6	6.838e-14	1.396e-2	1.677e-1	4.72
7	6.821e-14	1.398e-2	1.723e-1	2.10
8	6.823e-14	1.389e-2	1.737e-1	1.31
9	6.924e-14	1.382e-2	1.753e-1	0.40
10	7.046e-14	1.375e-2	1.771e-1	-0.63

Table 6.2: *Simulation results for the Fixed Splitting simulation of the M/M/1 queue*

The gain here is in simulation wall-clock time to arrive at a fixed relative error compared to the standard method without re-trials (given in the row $k = 1$). We observe that the gain seems to peak at $k = 2$ and deteriorates after that, even though the relative error continues to drop with k, since more samples are used. From equations (6.4) and (6.3) we can express the relative error as a function of k: $\text{RE}(\widehat{\gamma_S}(L)) \approx \sqrt{ck^{-1} + d}$, for some $c > 0$ and $d > 0$ (possibly dependent on L), which clearly demonstrates the monotone decrease in RE when increasing k. The factor d in this equation originates from the first simulation phase, being the splitting in order to generate the entrance states $S_1, \dots, S_n$; the factor c denotes the variance contribution of a sample in the second phase where we observe the variables $\{T_1(S_1), T_2(S_1), \dots, T_k(S_n)\}$.

The impact of these extra samples is almost negligible, which is evident from the relatively small drop in relative error. The error in the estimation of the entrance distribution caused by the splitting algorithm must be relatively large compared to the error in the estimated time spent in the rare event set. The total effect is that the simulation time does increase with k but the relative error does not decrease very much, resulting in a RTV increase with k. The smallest non-trivial k is $k = 2$, and this choice gives us the optimal; comparing it to the standard implementation, we get 1 to 6 % gain in efficiency using the

RESTART and splitting methods respectively. The gain is larger in the splitting method since here we have larger costs to generate one sample of the entrance distribution into the rare event set. Another remark from the tables is that the RESTART method is more efficient than the splitting method for this $M/M/1$ model; this is explained in Section 3.6: the system states are independent. However, in other models this may not always the case.

6.3.2 Two-node tandem queue

The second example is a two-node tandem queue with a Poisson arrival process and two servers each having an infinite buffer, as visually depicted in Figure 6.3.

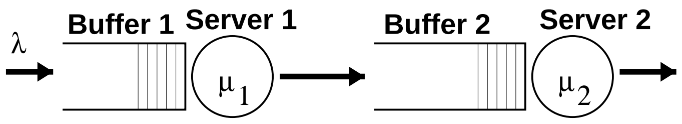

Figure 6.3: *The two-node tandem queue*

The underlying Markov chain is $X = (X_t, t \geq 0)$ with $X_t = (X_{1,t}, X_{2,t})$ and $X_{i,t}$ denoting the number of customers in queue number i at time t. Our rare event is the event that the number of clients in the second buffer exceeds a certain level L. The steady state distribution of this system is given by:

$$\lim_{t\to\infty} \mathbb{P}(X_t = (x,y)) = \pi(x,y) = (1-\rho_1)(1-\rho_2)\rho_1^x\rho_2^y$$

where ρ_i is the load of server i, defined as λ/μ_i. This easily gives us the following expression for the rare event probability, γ:

$$\gamma = \pi(A) = \rho_2^L$$

Choosing our parameters as $\lambda = 1, \mu_1 = 3$ and $\mu_2 = 2$ we see that the analytical values for γ are 2^{-L} and we generate the following Table 6.3 from simulation results using standard splitting methods.

We extend these results for the standard simulation methods with the proposed re-trials upon entering the rare event set. The optimal choice of the number of re-trials is empirically determined and presented in Table 6.4, for both the RESTART and Fixed Splitting method.

The results show that the use of the re-sampling scheme does not provide any gain; all the RTV's are larger than those found with the standard method. When we look at the difference in efficiency between the standard splitting

		Standard RESTART			Standard Fixed Splitting		
L	γ	$\hat{\gamma}$	RE	RTV	$\hat{\gamma}$	RE	RTV
10	9.765e-4	9.756e-4	2.1e-3	7.2e-3	9.791e-4	1.7e-3	6.1e-3
20	9.537e-7	9.495e-7	4.6e-3	6.4e-2	9.542e-7	3.2e-3	6.0e-2
40	9.095e-13	8.956e-13	8.3e-3	3.9e-1	9.046e-13	7.8e-3	8.0e-1

Table 6.3: *Simulation results for the two-node tandem queue*

		Optimal RESTART			Optimal Fixed Splitting		
L, k^*	γ	$\hat{\gamma}$	RE	RTV	$\hat{\gamma}$	RE	RTV
10,2	9.765e-4	9.757e-4	2.0e-3	7.7e-3	9.828e-4	1.6e-3	6.3e-3
20,3	9.537e-7	9.590e-7	4.7e-3	8.0e-2	9.565e-7	3.3e-3	7.1e-2
40,3	9.095e-13	8.988e-13	8.5e-3	4.2e-1	9.090e-13	8.0e-3	8.8e-1

Table 6.4: *Simulation results for the two-node tandem queue with re-trials at the rare event set*

(here denoted as FS for the Global-Step Fixed Splitting implementation) and pure RESTART methods we observe that for relatively large probabilities they are comparable, but for the rare events the pure RESTART seems to be more than twice as efficient as the FS method. This must be due to the relatively independent structure of the simulated tandem model: the pure RESTART takes advantage of independence, as argued in Section 3.6. At the same time the variance in the pure RESTART method is always larger than the pure FS, this is common as explained in the same Section 3.6: RESTART always reduces simulation run-time, but increases variance. In the next Section we consider a model of the ATM buffer switch with a high dependency structure.

6.3.3 ATM buffer switch

The system we consider here is more complex in the sense that we do not have an explicit analytic or numerical solution. Packets from one user arrive at a network switch following an inter-arrival time distribution $A(t)$. Traffic of N such users is multiplexed and offered to the switch which will need a fixed time C to process a packet. When the send-queue contains L packets it is full and a fresh and expensive buffer will have to be used for any arriving packets. The total cost incurred for the switching system will be determined by the fraction of the time spent using the fresh new buffer. A graphical model of this problem is given in Figure 6.4.

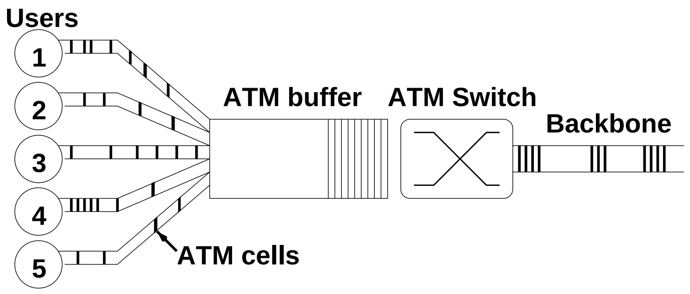

Figure 6.4: *The ATM switch model*

Traffic model

We model the user traffic as an on-off source, with on and off periods Pareto-distributed. Recall that the distribution function of a Pareto variable X is found as

$$\mathbb{P}(X > x) = (\frac{x}{b})^{-a};$$

we also call $H := 1 - a/2$ the Hurst parameter, which governs the degree of self-similarity in a sequence of identically distributed Pareto variables $\{X_1, X_2, \dots\}$. The parameter b is simply a scaling parameter. In the on period a geometrically distributed number of packets is sent, the time between two sent packets in one on-period is again Pareto-distributed. Since the user behaviour determines the buffer level, it is important that the users are in their steady state before simulation starts. We accomplish this by drawing a steady-state user state and an accompanying remaining time until the next user state change. Figure 6.5 visually depicts the model and the parameters.

Parameter setup

We set the input process Hurst parameter to 0.75 to create moderate self-similar traffic. Another choice for the model is the mean on/off time ratio of the user input traffic, we choose this to be 1 to create moderately bursty traffic. We set the server to send 100 packets per second, the average number of packets per

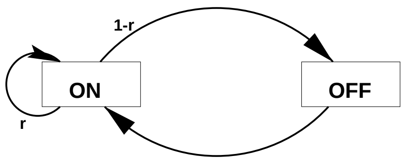

Figure 6.5: *The user model*

user on-time is 10; the Pareto distribution for the on- and off-times for the user are scaled up by a factor $b = 2$. The Pareto distribution will have to be cut-off at a certain threshold because it will otherwise cause infinite expectations for the holding times and thus an infinite variance in our estimator. The buffer size is 100 packets (or *cells* as they are usually referred to in ATM models). We will however vary the number of users N. The rare event probability we are interested in in this case is the steady state probability that the buffer is full. We accomplish this by simulating during a regeneration cycle of this system; we start and stop when the buffer is empty and the users are in their steady state. The results for the standard methods are given in Table 6.5.

	Standard RESTART			Standard FS		
N	$\hat{\gamma}$	RE	RTV	$\hat{\gamma}$	RE	RTV
20	3.823e-20	3.6e-2	64	3.720e-20	2.5e-2	41
25	2.266e-7	3.0e-2	1.0	2.261e-7	2.4e-2	1.0
30	6.221e-1	2.8e-2	0.79	6.019e-1	2.8e-2	0.75

Table 6.5: *Simulation results for ATM buffer switch model using the standard methods*

Now the same setup using the proposed re-trial method for improving the simulation efficiency in Table 6.6.

From these tables we see that the gain from using the proposed method is negative for relatively large probabilities, but increases with the rarity of the overflow event. This is also intuitively clear since the cost to obtain a sample that hits the (not so) rare event set in this case is very low, making the optimal number of re-samplings equal to one (i.e. no re-sampling).

Another conclusion is that the RESTART method is very much worse than the standard splitting (FS), as is clear from the 50% better efficiency for FS. This effect can be explained by the high correlation between the simulation

		Optimal RESTART			Optimal FS		
N	k^*	$\hat{\gamma}$	RE	RTV	$\hat{\gamma}$	RE	RTV
20	3	3.591e-20	3.6e-2	59	3.604e-20	2.4e-2	39
25	2	2.224e-7	3.0e-2	1.1	2.252e-7	2.4e-2	1.1
30	2	6.102e-1	2.6e-2	0.98	6.059e-1	2.5e-2	0.91

Table 6.6: *Simulation results for ATM buffer switch model using re-trials at the rare event set*

paths which results in a high variance for the RESTART estimate. Comparing the two tables we see that the proposed method gives a modest gain of about 5% for FS and 8% for RESTART for $N = 20$.

6.4 Conclusions

In this chapter we have introduced methods and techniques for steady state estimators using the splitting and RESTART methods. Previous existing regenerative techniques such as introduced by Kelling [Kel96] did work but could be altered by introducing a new splitting stage on top of the standard splitting method, effectively now doing re-trials at the rare event set. We have shown in Section 6.2 that this new method is giving better results as shown in Section 6.3 for the ATM and $M/M/1$ examples. The gain from using the proposed method is relatively modest; in the optimal implementation we typically get a 1 to 8% gain. We have also compared (analytically) the proposed method to an alternative, randomized method as suggested by the standard bootstrapping technique, and have shown that the latter is not better. Two different splitting algorithms were also compared, the standard splitting and the standard RESTART algorithm. Note that in Chapter 3 we also did a comparison of these two methods dealing with transient estimation, however it is interesting to see here how they compare in steady state estimation problems. There is no clear superior method, as already concluded in Chapter 3 for transient estimation problems, it depends on the simulated system which method is the better performer.

Chapter 7

Duality in steady state splitting simulation

7.1 Introduction

The splitting method has so far mainly been used in transient estimation; the steady state simulation offers new possibilities for speed–up especially when combined with "duality" as presented in Asmussen [Asm95].

The typical transient simulation method of splitting can be modified to accommodate steady state parameter estimations by using non-regenerative simulation as described in [Kel96] and [GF99] and regenerative methods as described in Chapter 6. A not yet investigated method we look at is transforming the problem of estimating steady state parameters of the studied system through the method of duality as proposed in [Asm95].

The duality principle has a long history in queueing theory and related fields, see e.g. [Sie76], [Pra65], [Asm95] and [AP92]. The basic idea is that we can analyse steady state properties of a system by looking at transient properties of another system, called the dual system. Estimation of steady state properties of the original system can then be done simply by doing standard splitting simulation of the dual system's transient properties. Note that it is not always possible to simulate a dual system, only in specific types of problems will the method be applicable.

We will show that the duality approach fits into the splitting method easily and recreates a transient estimation problem for which the standard splitting method can be used.

This chapter is organised as follows. Section 7.2 will outline the problem setting. In Section 7.2.3 we discuss the method based on the duality principle. In Section 7.3 we investigate the usefulness of the steady state splitting simulation using the duality principle compared to the standard regenerative steady-state splitting method as described in Chapter 6, by experimenting with various examples. Conclusions are given in Section 7.4.

7.2 Problem setting

The problem we will be looking at is the estimation of steady-state probabilities. Let $X = (X_t, t \geq 0)$ be a stochastic process with state space E, we are interested in parameters

$$\theta = \lim_{t\to\infty} \mathbb{E}(f(X_t) \in A) = \lim_{t\to\infty} \mathbb{P}(Z_t \in A) = \pi_Z(A), \quad A \subset \mathbb{R}$$

Examples of such problems are a reliability setup where highly reliable parts fail, or an inventory process that creates high losses only when there is no stock (stock-out). Other examples are the probability of loss with an Asian option with a positive drift or the probability of depletion of a capital due to claims as typically investigated in risk theory. We call $(Z_t := f(X_t), t \geq 0)$ the level process, indicating the importance of the current state X_t. Note that we arrive at the standard steady state problem of estimating $\pi(A)$ by setting $f(x) = x$ for a one-dimensional state space $E \subset \mathbb{R}$. For the standard splitting setting, the rare event set must be equal to $A = [L, \infty]$ for some overflow level L.

We will now look at the proposed method for estimating θ.

7.2.1 Duality and first passage times

Duality between stationary distributions and first passage times has been investigated in literature, see e.g. [Sie76], [Pra65], [Asm95], [Asm87] and [AP92]. We will explain the origin of the duality method as it greatly clarifies its purpose.

Prabhu [Pra65] researched the link between the stationary waiting time and the first passage time. We consider queueing models of the $GI/G/1$ type to describe the methodology; the following example is adapted from [Pra65]. The arrival process is defined by i.i.d. inter-arrival times governed by distribution function $A(t)$. Service times are i.i.d. and governed by the distribution function $B(t)$. Say we have the waiting time of the n-th customer described by W_n, his service time requirement B_n, the inter-arrival time elapsed since the previous customer entered the system as A_n, and we define the first passage time $T(u) := \min(n \,|\, W_n = 0; W_0 = u)$. Note that this is the first time the system becomes empty starting with an amount of work u is present in the system. Let $X_n := B_n - A_n$, the change in waiting time, and define the sequence $(S_n)_{n=0}^{\infty}$ with

$S_0 := 0, S_n := S_{n-1} + X_{n-1}, n \geq 1$ as the virtual waiting time. Let $K(x) = \mathbb{P}(X_n \leq x)$ the distribution function of X_n. Note that $W_{n+1} = [W_n + X_n]^+$ (use the usual notation $[x]^+ = \max\{0, x\}$). Define $g_n(u) = \mathbb{P}(T(u) = n), n \geq 1$, we have $g_1(u) = K(-u)$ and $g_{n+1}(u) = \int_{0+}^{\infty} \mathrm{d}K(x-u) g_n(x) \quad n \geq 1$, leading to $g(u) := \sum_{n=1}^{\infty} g_n(u) = \mathbb{P}(T(u) < \infty)$.

We can create a dual process by exchanging inter-arrival and service time distributions. Let $F(x) = \lim_{t\to\infty} \mathbb{P}(W_t \leq x)$ be the distribution function of the stationary waiting time and $g(x) = \mathbb{P}(T(x) < \infty)$ in the dual system. We then have

$$F(x) = 1 - g(x) \tag{7.1}$$

since $\lim_{n\to\infty} W_n$ is distributed as $\max_{n>0} S_n$ (see e.g., [Pra65] p. 134 or [Asm87] p. 281): recursing $W_{n+1} = \max\{0, W_n + X_n\}$ gives the equivalent $W_n = \max\{0, X_{n-1} + X_{n-2} + \ldots + X_{n-r}(r = 1, 2, \ldots, n-1), u + S_n\}$; clearly this is distributed as $\max\{S_r(r = 0, 1, \ldots, n-1), u + S_n\}$ when $W_0 = u$. Using this we find $F(x) = \mathbb{P}(S_n < x, n \geq 0)$. The dual system is then introduced as the system with interchanged arrival- and service-times. Its inter-arrival-times are given as $A'_n = B_n$ and $B'_n = A_n$, Introducing this system's virtual waiting time $S'_0 := u, S'_n = S'_{n-1} + B'_n - A'_n$ we find that $S'_n = -S_n$ and thus $F(x) = \mathbb{P}(x + S'_n > 0, n \geq 0)$ and $T(x) = \min(n \,|\, x + S'_n \leq 0)$, proving (7.1).

This simple relation for a single-server queue can be generalized as follows. Let $(X_n)_{n=0}^{\infty}$ be a stationary sequence, and $S_0 := x, S_n := S_{n-1} + X_{n-1}$. Introducing $M := \max(0, S_1, S_2, S_3, \ldots)$, and $\tau(x) := \inf\{n \,|\, S_n > x\}$, we see the equality of the events $\{M \geq x\} = \{\tau(x) < \infty\}$.

The dual process, (R_n) say, now becomes $R_0 := x, R_n := [R_0 - X_{n\wedge\tau(x)}]^+$. Observe that now $\tau(x) = \inf\{n > 0 \,|\, R_n < 0\}$.

Siegmund duality

Siegmund [Sie76] developed an even more general framework for duality; we present the main result adapted to our framework as:

Let $\{W_t\}$ be a Markov process with state space $[0, \infty)$ with discrete or continuous time. Then the existence of a Markov process R_t on state space $[0, \infty)$ with the property that

$$\mathbb{P}(W_t \geq u \,|\, W_0 = x) = \mathbb{P}(R_t \leq x \,|\, R_0 = u) \tag{7.2}$$

is equivalent to (i) $\{W_t\}$ is stochastically monotone; (ii) $\mathbb{P}(W_t \geq u \,|\, W_0 = x)$ is a right-continuous function of x for fixed u and t.

Substituting $x = 0$, setting $\tau(x) := \inf\{t \,|\, R_t = 0\}$ and taking the limit $t \to \infty$ in (7.2) we arrive at

$$\mathbb{P}(W \geq u) = \mathbb{P}(\tau(u) < \infty)$$

for all u. The stochastic process (R_t) has two absorbing states; the state $\{0\}$ and the state $\{\infty\}$. The realization of $\tau(x)$ is thus only known upon entering either of the absorbing states. We will try to make accurate estimations of $\mathbb{P}(\tau(x) < \infty)$, to obtain the steady-state probability $\mathbb{P}(W \geq x)$.

7.2.2 Stochastic Recursions

We will now describe a class of random sequences that fit the duality framework and for which we can explicitly find the dual sequence. Note that in this section the time parameter n is discrete. We define a stochastic recursion on $[0, \infty)$ to be a random process $(W_n)_{n=0}^{\infty}$ of the form

$$W_{n+1} = r(W_n, X_n), n = 0, 1, 2, \dots,$$

where $(X_n)_{n=-\infty}^{\infty}$ is a sequence of random variables with a state space E, and r is a function $[0, \infty) \times E \to [0, \infty)$. The dual function of $r(w, x)$ is a function $s(w, x)$ defined as

$$s(w, x) = \inf\{y \mid r(y, x) \geq w\} \tag{7.3}$$

and s is now a function $[0, \infty] \times E \to [0, \infty]$. The dual process of the stochastic recursion is now the random process $\{R_n\}$ with

$$R_{n+1} = s(R_n, X_{-n}), n = 0, 1, 2, \dots. \tag{7.4}$$

7.2.3 Setup for duality simulation

In this setup we focus on a specific problem which fits into the general splitting simulation setup as described in Section 7.2.

Algorithms have been proposed to solve this type of problem for a number of systems, such as the $GI/G/1$ system in [Sen90] , the Markovian fluid model in [Asm95] and the two-barrier problem in [AP92].

A number of assumptions are necessary in order to arrive at a model where we can apply our splitting method.

- We assume that the stochastic process $(X_t, t \geq 0)$ is a Markov process. The level process is $Z_t = f(X_t) \in \mathbb{R}$, making $(Z_t, t \geq 0)$ a real-valued Markov modulated process and the process (X_t, Z_t) Markov again by the fact that Z_t depends only on X_t and is thus Markov. We assume that X_t has a stationary distribution given by π_X, as such also Z_t must have a stationary distribution given by $\pi_Z = \pi_X \circ f^{-1}$ since

$$\begin{aligned}
\pi_Z(B) &= \lim_{t\to\infty} \mathbb{P}(Z_t \in B) = \lim_{t\to\infty} \mathbb{P}(f(X_t) \in B) = \\
&= \lim_{t\to\infty} \mathbb{P}(X_t \in f^{-1}(B)) = \pi_X(f^{-1}(B))
\end{aligned}$$

for any $B \subset \mathbb{R}$ with $f^{-1}(B)$ given by $\{x \in E \,|\, f(x) \in B\}$. The function $(f \circ g)(x)$ denotes as usual $f(g(x))$.

- We further assume the process $(X_t, t \geq 0)$ to be of the *separable* type which we define as $X_t = (Z_t, U_t) \in \mathbb{R} \times E$ where $(U_t, t \geq 0) \in E$ is just a driving process for the drift of $(Z_t, t \geq 0)$. We can in effect separate the important splitting parameter from X_t, leaving just a stationary process $(U_t, t \geq 0)$ for which we have the stationary distribution given. We refer to $(U_t, t \geq 0)$ as the driving process when $(U_t, t \geq 0)$ determines the drift $r(x), r : E \to \mathbb{R}$ of the process $(Z_t, t \geq 0)$ as follows:

$$\frac{\mathrm{d}Z_t}{\mathrm{d}t} = r(X_t) = r(Z_t, U_t)$$

  for a continuous-time process $(X_t, t \geq 0)$, or

$$Z_{n+1} = r(X_n) = r(Z_n, U_n)$$

  for a discrete-time set-up.

- We assume that the process $(U_t, t \in \mathbb{R})$ is stationary and has doubly infinite time. This assumption guarantees that the process $(Z_t, t \in \mathbb{R})$ or chain $(Z_n, n \in \mathbb{N})$ is also stationary with doubly infinite time. Note that since we still have the steady-state distribution of Z, given by $\pi_Z = \pi_X \circ f^{-1}$, but it might not be easy to characterise π_Z by means of the steady state distribution of the driving process $(U_t, t \in \mathbb{R})$, π_U because of the complicated dependency structure of (Z_t) on (U_t). This is the main point of the chapter, the process (U_t) is simple and its steady-state distribution is easily found, but the steady-state distribution of (Z_t) cannot be found using conventional methods since the drift function r is complicated.

- We will assume that $r(z, u)$ is continuous and non-decreasing in z for every $u \in E$, which typically holds for many systems. Note that we now have a system that fits the stochastic recursion formulation from Section 7.2.2.

The dual function of r can be found as the inverse s by (7.3); the dual function s is now left-continuous, non-decreasing and strictly increasing on $\{x \,|\, 0 < s(z, u) < \infty\}$ (see [Asm95]). For the discrete case we now obtain the dual process by (7.4).For the continuous time case we have

$$\frac{\mathrm{d}R_t}{\mathrm{d}t} = s(R_t, U_{-t}) \quad , t \geq 0,$$

defining the dual process (R_t).

This definition guarantees that the events $\{Z_t \geq x, Z_0 = y\}$ and $\{R_t \leq y, R_0 = x\}$ are equal. The equality of the events $\{R_t(x) = 0\} = \{\tau(x) \leq t\}$ holds since state zero is absorbing for (R_t); making

$$\mathbb{P}(Z \geq x) = \lim_{t \to \infty} \mathbb{P}(Z_t \geq x) = \mathbb{P}(\tau(x) < \infty).$$

Example 7.1 Markov-Modulated Fluid Queue
Markov-modulated fluid queues are incorporated in the continuous time model by considering a stationary Markov driving process U_t governing the the fluid rates $r^*(U_t)$ of input into a buffer with level process Z_t as follows: $r(Z_t, U_t) = I(Z_t > 0 \vee (Z_t = 0 \wedge r^*(U_t) > 0)) \cdot r^*(U_t)$; a reflection at zero and when $Z_v > 0, \quad$ for all $v \in [0, t]$ we have simply $Z_t = \int_0^t r^*(U_v) \mathrm{d}v$ (this readily holds for the non-reflective variant). This leads to the evolving of R_t as the reverse-time process of Z_t since the rates are given by the reverse-time process of U_t :$s(R_t, U_t) = -r^*(U_t) I(R_t > 0)$.

Example 7.2 Random Walk
For the discrete variant we arrive at a random walk when choosing the (U_n) an i.i.d. random sequence in $\mathbb{R}$ and the function r as the sum operator, and as usual $Z_n = r(Z_{n-1}, U_{n-1})$. A reflecting random walk can be obtained by choosing r as the Lindley recursion $r(x, y) = [x + y]^+$, a double reflecting random walk by choosing a maximum b and using $r(x, y) = \min(b, [x + y]^+)$. The dual function is given by $s(x, y) = I(x > 0)[x - y]^+$ and thus the dual process is given by $R_{n+1} = I(R_n > 0)[R_n - U_n]^+$. Increments for Z_n are equal decrements for R_n, as in the previous example.

Example 7.3 Single Server Queue
The single server $GI/GI/1$-queue is readily incorporated into the random walk by setting $U_n := A_n - B_n$, the arrivals in the n-th interval minus the served customers in the n-th interval, and r is the Lindley recursion $r(x, y) = [x + y]^+$. Note that this is not the same view of the single server queue as in Section 7.2.1; we are looking here at the process Z_n denoting the number of customers in the queue, here defined by $Z_n = [Z_{n-1} + A_{n-1} - B_{n-1}]^+$.

The proof for the dual process characteristics was given in [Asm95] for the discrete case and for the Markov-modulated process with a finite chain.

Splitting simulation in the duality set-up

Because of the equality of the events $\{\tau(L) < \infty\} = \{Z \geq L\}$, if the last event is rare as assumed, then the first event is also a rare event and we need a special rare event simulation technique to make estimation of the probability efficient. The choice of rare event technique to use is the splitting method as described in Chapter 2.

A problem with this simulation setup exists when estimating the probability of the event $\{\lim_{n\to\infty} R_n = \infty\}$, necessary for the stop criterion in the simulation. Since such an event will take infinite simulation time, it is not feasible in a simulator:the event will not occur in a simulator since it would take infinite time to achieve. We therefore constrain our set of problems here to the double-barrier queues where simulation will also reflect when X_t hits a finite level B. Any second barrier B will introduce a bias in the estimator; since the simulation is stopped whenever B is hit an error will be made for those samples that will return to the zero state after hitting B. The bias will be negligible however when we choose $B \gg L$. Based on our experiments we recommend the rule of thumb $B = 2L$.

As described we have $R_0 = L$ for all our simulations and estimate the transient probability $\gamma = \mathbb{P}(T_0(R) < T_B(R))$, i.e. that state zero is hit before B. Note that we introduce the random variables $T_x(Y)$ as the first time that level x is hit by stochastic process $(Y_n, n > 0)$, i.e. $T_x(Y) := \min\{n > 0 \,|\, Y_n = x\}$. The process (R_n) has an upward drift since γ is a rare event. In order to obtain a simulation with a negative drift and starting in state zero we introduce the variable $R'_n := L - R_n$. The probability we are interested in becomes $\gamma = \mathbb{P}(T_L(R') < T_{L-B}(R'))$, i.e. the probability that L is hit before $L - B$. The process (R'_n) behaves similar to the original process (Z_n), upward jumps of Z_n became downward jumps of R_n but are upward jumps again for (R'_n); only the driving process will have to start in the steady state for (R'_n) and the zero barrier is removed for (R'_n), making the variable more attractive analytically and easier to simulate.

We summarize the advantages of dual simulation as follows

- Loss of the reflective barrier makes the simulation easier; the event handler can be constructed simpler.

- Transient nature of the probability removes need for simulation of the continuous time Markov chain, the discrete skeleton process can be used and is faster; this is obvious since the random variables to be drawn for the continuous time simulator are more complex than the variables for the discrete chain simulator, and as such the speed of the discrete simulator is always higher.

- No regenerativity of the system is necessary. Systems for which the regeneration cycle is very long can now be simulated efficiently, since there is no need to wait until the cycle ends.

As a drawback we see that we need to simulate many paths back to the level $(L - B)$, which lies very low and could hurt the efficiency.

Simulation of the probability that process R'_t hits level L before $L - B$ can then be done using the standard splitting method.

7.3 Simulation results

In order to evaluate the efficiency of the proposed method,we compare it with other methods using various examples. The simulation setup is further described in Appendix D. We refer to Section 3.8 for an explanation of the tables.

7.3.1 $M/M/1$ queue

The first example will be the $M/M/1$ queueing system as described in Section 6.3.1. We will demonstrate the efficiency of both the standard and dual methods, and the gains of the various implementations in Table 7.1. For this system we have a discrete chain with $Z_n = [Z_{n-1} + U_{n-1}]^+$, with the driving sequence a binomial variable with $\mathbb{P}(U_n = 1) = \lambda/(\lambda + \mu)$ and $\mathbb{P}(U_n = -1) = \mu/(\lambda + \mu)$. The dual random variable is given by $R_{n+1} = s(R_n, U_n)$ where $s(x, u)$ is the dual of the stochastic recursion function for (Z_n), $r(y, u) = [y + u]^+$. The recursion function r satisfies the duality criteria since it is continuous and non-decreasing in y for every u. The dual function is easily found as

$$s(x,u) = \begin{cases} \infty & , x \geq B \\ x - u & , 0 < x < B \\ 0 & , x \leq 0, \end{cases}$$

where we have included the cut-off of the dual system when it hits B. From this dual function we see that we can create the dual process simply by switching the service and arrival processes. Since the original system (Z_n) was stable, the dual system (R_n) is unstable and has a positive drift. Transformation of the process as $R'_n := L - R_n$ yields a system similar to the original process (Z_n), with the exception that the boundary at zero is removed.

We conclude from the simulation results for the $M/M/1$ queue, as presented in Table 7.1, that the standard splitting method is still more efficient than the simulation of the dual system; however the difference is almost negligible.

	Analytical	Standard splitting			Duality splitting		
L	γ	$\hat{\gamma}$	RE	RTV	$\hat{\gamma}$	RE	RTV
80	4.054e-13	4.067e-13	1.8e-2	9.8e-2	4.028e-13	1.4e-2	1.1e-1
85	6.813e-14	6.811e-14	1.8e-2	1.1e-1	6.781e-14	1.4e-2	1.1e-1
90	1.145e-14	1.156e-14	1.9e-2	1.2e-1	1.129e-14	1.5e-2	1.2e-1
95	1.924e-15	1.933e-15	1.9e-2	1.3e-1	1.910e-15	1.5e-2	1.4e-1
100	3.234e-16	3.215e-16	2.0e-2	1.3e-1	3.182e-16	1.6e-2	1.5e-1

Table 7.1: *Simulation results for the M/M/1 queue*

7.3.2 Two-node tandem queue

The second example will be the two-node tandem queue, as detailed in Section 6.3.2; the same notation will be used here.

The discrete Markov chain $Z = (Z_n, n \geq 0)$ with $Z_n := X_{2,n}$ can now be modeled as $Z_n = r(Z_{n-1}, U_n, U_{n-1})$ whereby the sequence U_n is now a little more complicated than in the $M/M/1$ case, it consists of two variables: the state of queue 1 (i.e. the number of customers in the first buffer) $U_{1,n}$ and a variable $U_{2,n}$ describing the effects of the next state change on the process Z, i.e.,

$$\begin{cases} \mathbb{P}(U_{2,n}=0) = \mathbb{P}(\text{"arrival"}) = \lambda/(\lambda+\mu_1+\mu_2), \\ \mathbb{P}(U_{2,n}=1) = \mathbb{P}(\text{"service completion Q1"}) = \mu_1/(\lambda+\mu_1+\mu_2), \\ \mathbb{P}(U_{2,n}=2) = \mathbb{P}(\text{"service completion Q2"}) = \mu_2/(\lambda+\mu_1+\mu_2), \\ U_{1,n} - U_{1,n-1} = I(U_{2,n}=0) - I(U_{1,n-1}>0)I(U_{2,n}=1), \quad \text{and} \\ Z_n - Z_{n-1} = I(U_{1,n-1}>0)I(U_{2,n}=1) - I(Z_{n-1}>0)I(U_{2,n}=2) \end{cases} \tag{7.5}$$

We find the dual $s(x,\boldsymbol{u})$ of the stochastic recursion function $r(y,\boldsymbol{u})$ (just defined in (7.5) as $r(y,\boldsymbol{u}) = y + I(u_{2,1}>0)I(u_{1,2}=1) - I(y>0)I(u_{1,2}=2)$; note the introduction of the (2×2) matrix variable $\boldsymbol{u}$ which contains U_n and U_{n-1} on the first and second rows, respectively), as follows:

$$s(x,u) = \begin{cases} x & u_{1,2}=0, 0\leq x<B \\ 0 & u_{1,2}=1, x\leq 0 \\ x & u_{1,2}=1, 0<x<B, u_{2,1}=0 \\ x-1 & u_{1,2}=1, 0<x<B, u_{2,1}>0 \\ 0 & u_{1,2}=2, x\leq 0 \\ x+1 & u_{1,2}=2, 0<x<B \\ \infty & x\geq B, \end{cases}$$

where the cut-off of the dual system at level B is incorporated. Note that $s(x, \boldsymbol{u})$ is properly defined since $r(y, \boldsymbol{u})$ is non-decreasing in y. Choosing our parameters as $\lambda = 1, \mu_1 = 3$ and $\mu_2 = 2$, we see that the analytical values for γ are 2^{-L} and we generate Table 7.2 with simulation results from standard splitting and duality splitting methods.

	Analytical	Standard splitting			Duality splitting		
L	γ	$\hat{\gamma}$	RE	RTV	$\hat{\gamma}$	RE	RTV
10	9.765e-4	9.756e-4	2.1e-3	7.2e-3	9.765e-4	1.8e-3	1.0e-2
20	9.537e-7	9.495e-7	4.6e-3	6.4e-2	9.499e-7	3.9e-3	7.0e-2
40	9.095e-13	8.956e-13	8.3e-3	3.9e-1	8.850e-13	1.0e-2	3.4e-1

Table 7.2: *Simulation results for the two-node tandem queue*

We observe from the results in Table 7.2 that duality splitting is slightly better for large L.

7.3.3 Markov modulated fluid queue

For our last example we look at a continuous time Markov-modulated process. A similar model was considered in [KN99b]. Consider a queueing model with a Markov modulated input process: the environmental driving process $\boldsymbol{U_t}$ is a n-dimensional on-off process with exponential holding times: a parameter vector $\boldsymbol{\lambda}$ for the on-time and a parameter vector parameter $\boldsymbol{\mu}$ for the off-times. The state of the server is also incorporated into the environmental state; this because we assume the server is unreliable and alternates between the 'up' and 'down' states with exponential holding times. The driving process determines the influx into a buffer; the rate of influx will be the inner product of the vector $\boldsymbol{U_t}$ and a influx vector $\boldsymbol{c}$. The vector $\boldsymbol{c}$ will have only one negative entry corresponding to the server process draining the buffer. The interesting event will be the steady-state probability that the buffer content X_t exceeds a pre-specified level L. We have the following model

$$\frac{\mathrm{d}X_t}{\mathrm{d}t} = \boldsymbol{c} \cdot \boldsymbol{U_t} \quad I((X_t > 0) \vee ((X_t = 0) \wedge (\boldsymbol{c} \cdot \boldsymbol{U_t} > 0)))$$

The factor $I(\dots)$ describes the reflective boundary of the buffer at $X_t = 0$. Note that the "dot" signifies the inner vector product of the vectors $\boldsymbol{c}$ and $\boldsymbol{U_t}$. The model describes n users generating traffic into a broadband ATM switch, where the packets are so small and the network speed is so high that a fluid approximation holds. Each user generates traffic according to an 'on'-'off' source with exponential holding times. The rate at which each user generates traffic is constant in the 'on'-state, and zero in the 'off'-state. We visualise the model in Figure 7.1.

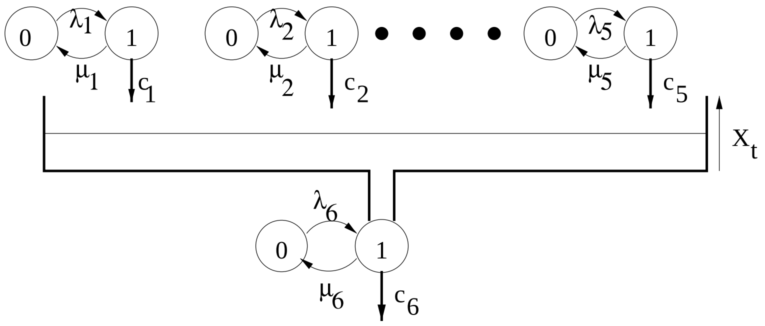

Figure 7.1: *The fluid queue*

It is clear that this model indeed satisfies the assumptions of the duality principle, since the function $r(x, \boldsymbol{u}) = \boldsymbol{c} \cdot \boldsymbol{u}\, I((x > 0) \vee ((x = 0) \wedge (\boldsymbol{c} \cdot \boldsymbol{u} > 0)))$ is continuous and non-decreasing in x for every $\boldsymbol{u}$. The dual function is found to be $s(x, \boldsymbol{u}) = -\boldsymbol{c} \cdot \boldsymbol{u}\ I(x > 0)$, and the dual process (R_t) thus moves in the opposite direction as the original process and the reflecting barrier at zero has been replaced with an absorbing one.

The chosen parameters are $n = 6$, $\boldsymbol{\lambda} = (2, 2, 2, 1, 1, 0.1), \boldsymbol{\mu} = (3, 3, 3, 4, 4, 5)$, and $\boldsymbol{c} = (1, 1, 1, 4, 4, -10)$. The simulation results of both the duality splitting and regenerative steady state splitting methods for this problem can be found in Table 7.3. For the regenerative steady state splitting method we start and stop whenever the buffer is empty. Analytical results for this model can be derived using Appendix C.

	Analytical	Standard splitting			Duality splitting		
L	γ	$\hat{\gamma}$	RE	RTV	$\hat{\gamma}$	RE	RTV
10	9.4503e-4	9.439e-4	4.2e-3	1.3e-2	9.484e-4	3.3e-3	9.3e-2
20	6.3214e-6	6.299e-6	5.2e-3	5.9e-2	6.319e-6	4.4e-3	2.6e-1
40	2.8285e-10	2.856e-10	6.1e-3	3.5e-1	2.847e-10	5.5e-3	4.4e-1

Table 7.3: *Simulation results for the fluid queue*

We again see that the standard method is more efficient than the duality-based implementation.

7.4 Conclusions

In this chapter we have introduced methods and techniques for steady state estimators using the splitting method. Previous existing techniques as introduced by Kelling [Kel96] and in Chapter 6 provided simple methods and extensions of the standard splitting method. In 7.2.3 we presented a duality-based approach which transformed the problem into a transient estimation problem and fit the requirements for the standard splitting method.

We have found the dual method to work well in all investigated problems, giving correct and efficient results. Comparing it to the the optimal standard splitting method we found it to be lower in efficiency caused by the on average longer stopping times of the paths generated by duality splitting method, which creates extra simulation cost. The unsuccessful paths in the duality splitting method will need to be simulated from their splitting threshold until they hit the cut-off level B, which is far away.

Although for the presented models we did not achieve an improvement using duality splitting over the standard regenerative steady-state splitting estimator, we can say that their efficiencies are comparable, certainly for large L. This causes us to expect the duality splitting method to be more efficient in non-regenerative systems, since for such systems we can only compare duality splitting to non-regenerative steady-state estimators, which are less efficient than the regenerative ones.

Appendix A

Estimating the variance for the FE method

In this Appendix we explain how to estimate the variance of the Fixed Effort estimator. Although the variance calculation of Section 2 provides us with valuable information on the choice of the splitting parameters, we usually cannot assume that the indicators $\{I_i^{(k)}\}$ are independent of each other, and hence we have to estimate the variance of $\hat{\gamma}$ in another way for the Fixed Effort method.

We recall a few definitions.

γ	:	probability of interest, with estimator $\hat{\gamma}$.
m	:	number of stages.
L_k	:	intermediate level, $k = 0, \dots, m$ ($L_0 = 0\,, L_m = L$).
n_k	:	number of splits per state in stage k.
p_k	:	probability of hitting L_k, starting from L_{k-1}.
R_k	:	number of successful hits of level L_k.
r_k	:	total number of restarts from level L_k.

First off, we note that the estimator for the FE method is equal to the product of the success probabilities:

$$\widehat{\gamma} = \prod_{i=1}^{m} \hat{p}_i = \prod_{i=1}^{m} \frac{R_i}{r_i} = \frac{\prod_{i=1}^{m} R_i}{\prod_{i=1}^{m} r_i}$$

which does not lend itself for the easy variance expression for the FS and Global-Step as given in Section 2.4.3.

To investigate the variance of $\hat{\gamma} = \hat{p}_1 \cdots \hat{p}_m$, let us assume that the total simulation effort in every stage is large, so that $\hat{p}_k$ is approximately distributed as $p_k + \sigma_k V_k$, where σ_k is the (small) standard deviation of $\hat{p}_k$ and V_k has a standard normal distribution. Because the σ_k's are assumed to be small, we have

$$\mathbb{V}\mathrm{ar} \prod_{k=1}^{m} (p_k + \sigma_k V_k) \approx \mathbb{V}\mathrm{ar}\, \gamma \sum_{k=1}^{m} \frac{\sigma_k V_k}{p_k} = \gamma^2\, \mathbb{V}\mathrm{ar} \sum_{k=1}^{m} \frac{\hat{p}_k}{p_k}.$$

This and the fact that the $\hat{p}_k$'s are uncorrelated, yields

$$\mathbb{V}\mathrm{ar}\, \hat{\gamma} \approx \gamma^2 \sum_{k=1}^{m} \mathbb{V}\mathrm{ar} \left(\frac{\hat{p}_k}{p_k}\right).$$

It remains to estimate the variance of each $\hat{p}_k$. Recall that at each stage k we have R_{k-1} possible states from which we can restart. Let Y_i denote the total number of paths that hit level L_k of all the paths that start from the ith initial state, $i = 1, \dots, R_{k-1}$. If we use the Fixed Assignment method, we distribute the possible starting positions "cyclicly" amongst the r_k paths, so that to each starting state $\lfloor r_k/R_{k-1} \rfloor$ or $\lceil r_k/R_{k-1} \rceil$ paths are assigned. The initial state of this cyclic scheme is chosen uniformly amongst the R_{k-1} possible states. In this way, the $\{Y_i\}$ and also $\{(Y_i, Y_j)\}_{i \neq j}$ are identically distributed. Let us denote the conditional expectation, variance and covariance with respect to some random variable W by $\mathbb{E}_W$, $\mathbb{V}\mathrm{ar}_W$ and $\mathbb{C}\mathrm{ov}_W$, respectively. Since $\mathbb{E}_{R_{k-1}}\, \hat{p}_k = p_k$, we have

$$\begin{aligned}
\mathbb{V}\mathrm{ar}\, \hat{p}_k &= \mathbb{E}\, \mathbb{V}\mathrm{ar}_{R_{k-1}}\, \hat{p}_k + \mathbb{V}\mathrm{ar}\, \mathbb{E}_{R_{k-1}}\, \hat{p}_k = \mathbb{E}\, \mathbb{V}\mathrm{ar}_{R_{k-1}}\, \hat{p}_k \\
&= \frac{1}{r_k^2}\, \mathbb{E}\, \mathbb{V}\mathrm{ar}_{R_{k-1}} (Y_1 + \cdots + Y_{R_{k-1}}) \\
&= \frac{1}{r_k^2}\, \mathbb{E}\, \left\{ R_{k-1}\, \mathbb{V}\mathrm{ar}_{R_{k-1}}(Y_1) + R_{k-1}(R_{k-1} - 1)\, \mathbb{C}\mathrm{ov}_{R_{k-1}}(Y_1, Y_2) \right\} \\
&= \frac{p_k(1 - p_k)}{r_k} + \frac{\mathbb{E}\left\{ R_{k-1}(R_{k-1} - 1) \left(Y_1 Y_2 - (R_k/R_{k-1})^2 \right) \right\}}{r_k^2}
\end{aligned}$$

Hence, we take as an unbiased estimator for $\mathbb{V}\mathrm{ar}\, \hat{p}_k$

$$\widehat{\mathbb{V}\mathrm{ar}}\, \hat{p}_k := \frac{1}{r_k^2} \left\{ R_k(1 - R_k/r_k) + \sum_{i \neq j}\sum Y_i Y_j - R_k^2 (R_{k-1} - 1)/R_{k-1} \right\}.$$

Appendix B

Matrix methods for the tandem queue

B.1 The entrance distribution for the tandem queue

In this Appendix we explain how the entrance distributions (and indeed the actual overflow probabilities) can be computed for the tandem queue. Consider the discrete time Markov process $X := \{X_1(n), X_2(n)\}$, where $X_i(n)$ is the number of customers in system i, $i = 1, 2$, at the time of the n-th transition, $n = 0, 1, 2, \ldots$. The state space of the Markov chain is $E := \{0, \ldots, n_1\} \times \{0, \ldots, n_2\}$. Here n_2 may be infinite. To simplify the definitions below, let us define the events

- a : system 1 not empty or full,
- b : system 1 empty,
- c : system 1 full.

Given the fact that system 2 is not empty or full and event a has occurred, one of the following three possible transitions will happen:

- *1* : service completion in the second queue,
- *2* : arrival in the first queue,
- *3* : service completion in the first queue.

We denote the corresponding (conditional) probabilities by a_1, a_2, a_3. However, when event b has occurred (again, system 2 is not empty or full), we have only two possible transitions (1 and 2 of the list above). The corresponding probabilities are b_1, b_2. Lastly, when event c has happened, system 1 is full and transition 2 is impossible; leaving the probabilities c_1, c_3. We thus have, in the obvious notation,

$$\begin{aligned}
\{a_1, a_2, a_3\} &:= \{\mu_2/(\mu_1+\mu_2+\lambda), \lambda/(\mu_1+\mu_2+\lambda), \mu_1/(\mu_1+\mu_2+\lambda)\},\\
\{b_1, b_2\} &:= \{\mu_2/(\mu_2+\lambda), \lambda/(\mu_2+\lambda)\}\\
\{c_1, c_3\} &:= \{\mu_2/(\mu_1+\mu_2), \mu_1/(\mu_1+\mu_2)\}
\end{aligned}$$

Now, for $i \in \{1,2,3\}$, let A_i be square matrix of dimension n_1+1 whose (j,k)th element contains the probability that transition i occurs (as defined in the list above) and the number of customers in the first system changes from j to k, assuming that the second system is not full or empty.

Thus, A_1 is the diagonal matrix with $A_1(i,i) = a_1, i = 1, \ldots, n_1 - 1$, $A_1(0,0) = b_1$ and $A_1(n_1, n_1) = c_1$. Moreover, A_2 is the "upper-diagonal" matrix with $A_2(i, i+1) = a_2$, $i = 1, \ldots, n_1 - 1$ and $A_2(0,1) = b_2$; all other entries are 0. Finally, A_3 is the "lower-diagonal" matrix with $A_3(i, i-1) = a_3$, $i = 1, \ldots, n_1 - 1$ and $A_2(n_1, n_1 - 1) = c_3$; all other entries are 0.

Let $P_{k,l}(i,j)$ be the probability that X enters (for the first time) level $(*, l)$ at state (j, l) before it reaches level $(*, 0)$, (that is, before the second server becomes idle), starting from state (i, k). We have

$$P_{1,2} = \begin{pmatrix}
b_2a_3 & b_2a_2a_3 & b_2a_2^2a_3 & \cdots & b_2a_2^{n_1-2}a_3 & b_2a_2^{n_1-1}c_3 & 0\\
a_3 & a_2a_3 & a_2^2a_3 & \cdots & a_2^{n_1-2}a_3 & a_2^{n_1-1}c_3 & \vdots\\
0 & a_3 & a_2a_3 & \cdots & a_2^{n_1-3}a_3 & a_2^{n_1-2}c_3 & \vdots\\
\vdots & 0 & a_3 & \ddots & & \vdots & \vdots\\
\vdots & & \ddots & \ddots & \ddots & \vdots & \vdots\\
\vdots & & & 0 & a_3 & a_2c_3 & 0\\
0 & \cdots & \cdots & \cdots & 0 & c_3 & 0
\end{pmatrix},$$

and

$$P_{k,k+1} = A_1 P_{k-1,k} P_{k,k+1} + A_2 P_{k,k+1} + A_3.$$

Hence, if we define $Q_k := P_{k,k+1}$, then $Q_1 = P_{1,2}$, and

$$Q_k = (I - A_1 Q_{k-1} - A_2)^{-1} A_3.$$

Finally, we have

$$P_{1,k} = Q_1 Q_2 \cdots Q_{k-1}.$$

The entrance distribution for level k, starting from state $(i, 1)$, may be found as the ith row of $P_{1,k}$, properly normalized. The actual overflow probability of level k is the sum of all the elements in the ith row of $P_{1,k}$.

B.2 The importance function for the tandem queue

Extending the results obtained in B.1 we find that the overflow probability of reaching level L starting from state (i, j) is equal to

$$\mathbb{P}(A(L) \,|\, (i, j)) = e_i \cdot P_{j,L} \cdot e$$

where we have used e_i as the row vector for which element $j := \delta_{ij}$ (Kronecker's delta) and e the column vector of all 1's.

The optimal importance function is then easily found using the method described in Section 5.3. We find the overflow probability γ itself by substituting the starting state.

Appendix C

Markov modulated fluid queue analysis

In this Appendix we look at a specific Markov modulated fluid model; namely the one described in Section 7.3.3. We assume that there are $n := n_1 + n_2 + 1$ Markov model elements, n_1 of which of source Type I, n_2 of source Type II and one server. All model elements have exponential up- and down-times λ_i and μ_i, respectively, and process fluid at a rate r_i whenever they are in the "up"-state. The processing direction is into the buffer for the sources and out of the buffer for the server.

In order to simulate the original and dual processes for this specific Markov-modulated fluid model we need to determine certain distributions. The necessary distributions are the steady-state distribution of the driving process and the distribution of the driving process at a regeneration point. Also we find the asymptotic entrance distribution, i.e. the distribution of the driving process upon hitting a very high threshold.

C.1 Analysis

Many authors have analysed this problem and have found the explicit steady-state distribution. We obtain the analytic steady-state overflow probabilities for this model by following Scheinhardt (see [Sch98]): Denote the sets of indices belonging to Type i as $\mathcal{T}_i := \{k = 1, 2, \dots, n \,|\, \text{element '}k\text{' is of Type } i\}$. We have $(\lambda_i, \mu_i, r_i) = (2, 3, 1)$ for every $i \in \mathcal{T}_1$ and $(\lambda_i, \mu_i, r_i) = (1, 4, 4)$ for every $i \in \mathcal{T}_2$, respectively, with $n_1 = 3$ sources of Type I and $n_2 = 2$ sources of Type II. For the server we see that $(\lambda_i, \mu_i, r_i) = (0.1, 5, -10)$ for $i = n_1 + n_2 + 1$.

The complete process U_t now consists of three independent birth-death processes, one for each source Type and one for the server, say $(u_i(t), i = 1, 2, 3)$, describing the number of active elements in the particular groups, with a state space of $|E| = (n_1 + 1) \cdot (n_2 + 1) \cdot 2$ states, and input rates given by $c = r \cdot u$. Denote $\mathcal{N}_+ = \{i \in E \,|\, c_i > 0\}, \mathcal{N}_- = \{i \in E \,|\, c_i \leq 0\} = E \setminus \mathcal{N}_+$ Since the process is a continuous-time Markov chain, we can describe its behaviour by its rate-matrix Q.

Denoting the distribution function of (Z_t, U_t) as

$$F_i(z, t) = \mathbb{P}(Z_t \leq z, U_t = i) \quad i \in E, z \geq 0$$

it is clear that

$$\frac{\partial F_i(z,t)}{\partial t} = c_i \frac{\partial F_i(z,t)}{\partial z} + \sum_{j=1}^{|E|} Q_{ji} F_j(z,t) \quad i \in E, z \geq 0$$

or, in matrix form, where $\mathbf{F}(z) = \lim_{t\to\infty}(F_1(z,t), \ldots, F_{|E|}(z,t))$ and C is the diagonal matrix with elements $(c_1, \ldots, c_{|E|})$, for the steady-state distribution

$$\frac{\mathrm{d}\mathbf{F}(z)}{\mathrm{d}z} = -C^{-1} Q^T \mathbf{F}(z) \quad z \geq 0$$

and it is clear that its behaviour is determined by the matrix $-C^{-1}Q^T$. Denoting that matrix' eigenvalues and orthonormalized right eigenvectors in matrix form as Ξ and V respectively, we arrive at the solution

$$\mathbf{F}(z) = V \exp(\Xi z) V^{-1} k = \pi + \sum_{i=0}^{N_+ - 1} k_i \mathbf{v}_i \exp(\xi_i z)$$

using $N_+ = |\mathcal{N}_+|$ as the number of states with positive fluid rates. The eigenvalues can be shown to be all real and simple. The coefficients k_i belonging to positive eigenvalues ξ_i are necessarily zero due to the boundary conditions that the probabilities remain smaller than one for all z and can be omitted from the equation. One of the eigenvalues is zero because Q is rank deficient and has an eigenvector equal to $\mathbf{F}(\infty)$ since the other terms vanish in the limit. Also, we find $\mathbf{F}(\infty)$ by the boundary condition $\lim_{z\to\infty}(\mathbb{P}(Z \leq z, U = i))_{i\in E} = \boldsymbol{\pi}$, $\boldsymbol{\pi}$ being given by the steady state distribution of (U_t). The number of negative eigenvalues can easily be seen to be equal to the number of states with positive net flow rates. We see that

$$\begin{aligned}
\gamma &= \mathbb{P}(Z > L) = 1 - \mathbf{e}^T \mathbf{F}(L) = 1 - \mathbf{e}^T(\boldsymbol{\pi} + \sum_{i=0}^{N_+ - 1} k_i \mathbf{v}_i \exp(\xi_i L)) \\
&= \sum_{i=0}^{N_+ - 1} -k_i (\mathbf{e}^T \mathbf{v}_i) \exp(\xi_i L),
\end{aligned}$$

where $\mathbf{e}^T$ is the row vector consisting of all ones, necessary for the summing over the states. The vector of constants k can be derived from the boundary condition that having a positive net rate and an empty buffer cannot occur in the steady state

$$\begin{aligned} 0 &= \mathbb{P}(Z = 0, U = i) = \mathbb{P}(Z \leq 0, U = i) = F_i(0) \\ &= \pi_i + \sum_{j=0}^{N_+ - 1} k_j v_{ij}, \quad \forall i \in \mathcal{N}_+ \end{aligned}$$

In matrix notation this translates as

$$0 = \tilde{\pi} + \tilde{V}\tilde{k} \Longleftrightarrow \tilde{k} = -\tilde{V}^{-1}\tilde{\pi}$$

where $\tilde{V}$ is the reduced matrix of eigenvectors with only columns corresponding to negative eigenvalues and rows corresponding to positive net rates; $\tilde{\pi}$ is the reduced vector of steady state probabilities with components corresponding only to positive net rates and $\tilde{k}$ the vector of constants k_i corresponding only to negative eigenvalues ξ_i. The matrix $\tilde{V}$ is square since the number of positive net rates equals the number of negative eigenvalues.

C.2 Starting Distribution, original process

For the steady-state fluid queue we start the simulation using the steady state distribution of the process (Z, U) at the instants $\{t > 0 \mid Z_t = 0 \wedge Z'_t > 0\}$, where it just starts to fill the buffer. The end of a simulation cycle is determined by the first next event $\{Z_t = 0 \wedge Z'_t > 0\}$, i.e. we are simulating a busy cycle. In order to perform correct simulations we need to derive the distribution of U_t at a start of a new cycle moment. We derive the steady-state distribution of U_t, U_∞ say, when $Z_t = 0$ easily as

$$\mathbb{P}(U_\infty = i, Z_\infty = 0) = F_i(0) = \pi_i + \sum_{j=0}^{N_+ - 1} k_j v_{ij}, \quad \forall i \in \mathcal{N}_-$$

and thus the conditional distribution as

$$q_i^{(0)} = \mathbb{P}(U_\infty = i \mid Z_\infty = 0) = \frac{\pi_i + \sum_{j=0}^{N_+ - 1} k_j v_{ij}}{\sum_{j \in \mathcal{N}_-} (\pi_j + \sum_{l=0}^{N_+ - 1} k_l v_{jl})} \quad \forall i \in \mathcal{N}_-$$

The distribution of U_t at the start of a new cycle is then derived by drawing U_t from the given steady-state distribution and finding the next instant for which $r \cdot U_t > 0$, i.e. when it just starts to fill the buffer again.

This is achieved by deriving the discrete Markov chain conditional upon the event $\{Z_n = 0\}$ and modifying it to make the states for which a new cycle starts, i.e. those with $r \cdot U_t > 0$, absorbing as follows (denote the original jump chain matrix by P)

$$h_i = \begin{cases} p_i & \forall i \in \mathcal{N}_- \\ e_i & \forall i \in \mathcal{N}_+ \end{cases}$$

where h_i denotes the column vector i of the P-matrix for the conditional process on $Z_t = 0$, p_i denotes the column vector of the unconditional process' P-matrix and e_i the unit vector in dimension i to make that state absorbing. The distribution of U_t at the start of a new cycle is then found as q determined by

$$\begin{aligned} q^{(n+1)} &= q^{(n)}H \quad ,n \geq 0 \\ q &= \lim_{n\to\infty} q^{(n)} \end{aligned}$$

C.3 Asymptotic entrance distribution

The asymptotic entrance distribution in vector form, $\mathbf{q}$ say, can be found as the distribution of the original process given that the event $\{Z_t \geq L\}$ occurs, as follows:

$$\begin{aligned} q_i &= \mathbb{P}(U_t = i \,|\, Z_t \geq L) = \gamma^{-1} \sum_{j=0}^{N_+-1} -k_j \mathbf{v}_j \exp(\xi_j L) \\ &= \gamma^{-1} \sum_{j=0}^{N_+-1} -k_j v_{ij} \exp(\xi_j L) \quad \forall i \in E \end{aligned}$$

It is clear that the steady state overflow probability γ has an exponential decay governed by the largest negative eigenvalue ξ_0 of $-C^{-1}Q^T$ by first looking at the expression

$$\gamma = \sum_{i=0}^{N_+-1} q_i = \sum_{i=0}^{N_+-1} -k_i \mathbf{e}^T \mathbf{v}_i \exp(\xi_i L).$$

For large L this expression is close to $-k_0(\mathbf{e}^T\mathbf{v}_0)\exp(\xi_0 L)$ since γ has an exponential decay for large L with parameter ξ_0

$$\lim_{L\to\infty} \gamma \exp(-\xi_0 L) = \lim_{L\to\infty} \sum_{j=0}^{N_+-1} -k_j(\mathbf{e}^T\mathbf{v}_j)\exp((\xi_j - \xi_0)L) = -k_0(\mathbf{e}^T\mathbf{v}_0)$$

assuming without loss of generality that the negative eigenvalues are ordered $\xi_0 > \xi_1 > \ldots > \xi_{N_+-1}$. For large L we see that we can substitute the asymptotic distribution of U_t, which is now easily found to be the normalized dominant eigenvector of $-C^{-1}Q^T$ by

$$\begin{aligned}
\lim_{L\to\infty} \mathbf{q} &= \lim_{L\to\infty} -\gamma^{-1} \sum_{j=0}^{N_+-1} k_j \mathbf{v_j} \exp(\xi_j L) = \lim_{L\to\infty} -\gamma^{-1} k_0 \mathbf{v}_0 \exp(\xi_0 L) \\
&= (\mathbf{e}^T \mathbf{v}_0)^{-1} \mathbf{v}_0
\end{aligned}$$

Appendix D

Simulation setup

Note that the use of a different simulation machine only affects the RTV's in the tables since the actual simulation time will change. The RTV ratios or gains will be approximately independent of the simulation machine since we expect that no architecture- or compiler-dependent speed-ups are involved. We explain the machine configurations here in order to make the tables in this thesis reproducible.

The simulation machine in almost all cases was the Sun compute server with DNS name *"demeter.cs.utwente.nl"* running the Solaris operating system version 2.6, equipped with 6 UltraSparcII 336 MHz processors online and 3 GB of RAM. We refer to this machine as Machine 1. The simulation tool used was the simulation package developed by the author at the University of Twente, compiled with the Sun Workshop C compiler cc version 4.2 with optimization options *-mt -xO5 -fsimple -fast -xparallel -xautopar -xreduction*, running multi-threaded to fully occupy all the available processors.

For the splitting simulations a pilot run is performed with the Fixed Effort method to determine the optimal splitting parameters. The effort used for this trial run is typically 10% of the effort used in achieving the rare events in the actual simulations. The effort involved in the pre-run is not counted in the total effort for the simulation since these are small and almost equal for different implementations.

The pure splitting is done using the truncation technique to stop unpromising simulation paths, thereby optimising the total simulation effort. This creates a bias which is corrected inside the simulator.

The number of simulations run in each example is typically 10^7, all other splitting parameters are chosen in an optimal fashion by the pilot run.

Two other simulation machines were used in Chapter 4 which we will describe briefly.

Machine 2

This machine is a PC with 128MB memory, processor Intel Celeron 300A CPU at 450Mhz, running operating system Linux-2.2.0pre7; compiler GNU gcc version 2.7.2.3 with optimization level 3.

Machine 3

This is a PC with 32MB memory running a Linux-2.0.33 kernel on an Intel Pentium 150MHz processor, using the GNU C compiler gcc version 2.7.2.1 with optimization level 3 (-O3).

Bibliography

[AHT90] V. Anantharam, P. Heidelberger, and P. Tsoucas. Analysis of rare events in continuous time Markov chains via time reversal and fluid approximation. *IBM Research Report*, 16280, 1990.

[AN71] K.B. Athreya and P.E. Ney. Branching processes. In *Die Grundlehren der Mathematischen Wissenschaften in Einzeldarstellungen*, volume 196. Springer Verlag, 1971.

[AP92] S. Asmussen and D. Perry. On cycle maxima, first passage problems and extreme value theory for queues. *Stochastic Models*, 8:421–458, 1992.

[AR95] S. Asmussen and R.Y. Rubinstein. Steady state rare events simulation in queueing models and its complexity properties. In *Advances in Queueing: Theory, Methods and Open Problems*, pages 429–461. CRC Press, 1995.

[Asm87] S. Asmussen. *Applied probability and queues.* John Wiley and Sons, 1987.

[Asm95] S. Asmussen. Stationary distributions via first passage times. In J.H. Dshalalow, editor, *Advances in Queueing: Theory, Methods and Open Problems*, pages 79–102. CRC Press, New York, 1995.

[Bay70] A.J. Bayes. Statistical techniques for simulation models. *The Australian Computer Journal*, 2(4):180–184, 1970.

[Bay72] A.J. Bayes. A minimum variance technique for simulation models. *Journal of the Association for Computing Machinery*, 19:734–741, 1972.

[BBK99] K. Below, L. Battaglia, and U. Killat. RESTART/LRE simulation : The reliability issue. In *Proceedings of the RESIM Workshop 11-12 March 1999*, pages 89–98. University of Twente, The Netherlands, 1999.

[Bel98] K. Below. Rare-event simulation in ATM networks: The RESTART/LRE method, implementation and improvement. *Digital Communication Systems*, pages 1–128, 1998.

[dBNvO98] P.T. de Boer, V.F. Nicola, and J.C.W. van Ommeren. The remaining service time upon hitting a high level in M/G/1 queues. CTIT Technical Report 98-12, University of Twente, 1998. Submitted for publication in Queueing Systems: Theory and Applications.

[Ded90] E.J. Dedewicz. The generalized bootstrap. In G. Rothe K.-H. Jöckel and W. Sendler, editors, *Bootstrapping and Related Techniques, Proceedings*, pages 31–37, Trier FRG, 1990.

[Efr82] B. Efron. The jackknife, the bootstrap and other resampling plans. In *Regional Conference Series in Applied Mathematics*, volume 38. Society for Industrial and Applied Mathematics, 1982.

[GB99] M. Grossglauser and J.-C. Bolot. On the relevance of long-range dependence in network traffic. *IEEE/ACM Transactions on Networking*, 7(5):629–640, 1999.

[GF98] C. Görg and O. Fuß. Comparison and optimization of RESTART run time strategies. *AEÜ*, 52(3):197–204, 1998.

[GF99] C. Görg and O. Fuß. Simulating rare event details of ATM delay time distributions with RESTART/LRE. In *Proceedings of the RESIM Workshop 11-12 March 1999*, pages 41–54. University of Twente, the Netherlands, 1999.

[GHSZ96a] P. Glasserman, P. Heidelberger, P. Shahabuddin, and T. Zajic. A look at multilevel splitting. In H. Niederreiter, editor, *Monte Carlo and Quasi Monte Carlo Methods 1996, Lecture Notes in Statistics*, volume 127, pages 99–108. Springer Verlag, 1996.

[GHSZ96b] P. Glasserman, P. Heidelberger, P. Shahabuddin, and T. Zajic. Splitting for rare event simulation: Analysis of simple cases. In *Proceedings of the 1996 Winter Simulation Conference*, pages

302–308, San Diego, California, 1996. IEEE Computer Society Press.

[GHSZ98] P. Glasserman, P. Heidelberger, P. Shahabuddin, and T. Zajic. A large deviations perspective on the efficiency of multilevel splitting. *IEEE Transactions on Automatic Control*, 43(12):1666–1679, 1998.

[GHSZ99] P. Glasserman, P. Heidelberger, P. Shahabuddin, and T. Zajic. Multilevel splitting for estimating rare event probabilities. *Operations Research*, 47(4):585–600, 1999.

[GK98] M.J.J. Garvels and D.P. Kroese. A comparison of RESTART implementations. In *Proceedings of the 1998 Winter Simulation Conference*, pages 601–609, Washington, DC, 1998.

[GK99] M.J.J. Garvels and D.P. Kroese. On the entrance distribution in RESTART simulation. In *Proceedings of the RESIM'99 workshop*, pages 65–88. University of Twente, 11-12 March 1999, 1999.

[GKvO00] M.J.J. Garvels, D.P. Kroese, and J.C.W. van Ommeren. On the importance function in splitting simulation. In *Proceedings of the 2000 Symposium on Performance Evaluation of Computer and Telecommunications Systems*, pages 131–138. Vancouver BC, Canada, 16-20 July, 2000.

[GW92] P.W. Glynn and W. Whitt. The asymptotic efficiency of simulation estimators. *Operations Research*, 40:505–520, 1992.

[HT99] Z. Haraszti and J. Townsend. Rare event simulation of delay in packet switching networks using DPR-based splitting. In *Proceedings of the RESIM Workshop, 11-12 March 1999*, pages 185–190. University of Twente, the Netherlands, 1999.

[Hub81] Peter J. Huber. *Robust Statistics.* John Wiley and Sons, Inc., 1981.

[Kel79] F.P. Kelly. *Reversibility and Stochastic Networks.* Wiley Statistics Series, 1979.

[Kel96] C. Kelling. A framework for rare event simulation of stochastic Petri nets using "RESTART". In *Proceedings of the 1996 Winter Simulation Conference*, pages 317–323, Coronado, California, 1996.

[KH51] H. Kahn and T.E. Harris. *Estimation of Particle Transmission by Random Sampling.* National Bureau of Standards Applied Mathematics Series, 1951.

[KN98] D.P. Kroese and V.F. Nicola. Efficient simulation of backlogs in fluid flow lines. *AEÜ Int. J. Electron. Commun.*, 52:165–172, 1998.

[KN99a] D.P. Kroese and V.F. Nicola. Efficient simulation of a tandem Jackson network. In *Proceedings of the Second Workshop on Rare Event Simulation (RESIM'99)*, pages 197–211. University of Twente, Enschede, the Netherlands, 1999.

[KN99b] D.P. Kroese and V.F. Nicola. Efficient simulation of overflow probabilities in queues with breakdowns. *Performance Evaluation*, 36–37:471–484, 1999.

[Leh97] E.L. Lehmann. *Testing statistical hypotheses.* Springer Verlag, 1997.

[LV93] M. Li and P. Vitányi. *An Introduction to Kolmogorov complexity and its applications.* Springer Verlag, 1993.

[Mét82] M. Métivier. *Semimartingales.* De Gruyter Studies in Mathematics, 1982.

[Neu81] M.F. Neuts. *Matrix-geometric solutions in stochastic models.* Johns Hopkins University Press, Baltimore, 1981.

[Pra65] N.U. Prabhu. *Queues and Inventories.* Wiley Series in Probability and Mathematical Statistics, 1965.

[PW89] S. Parekh and J. Walrand. A quick simulation method for excessive backlogs in networks of queues. *IEEE Transactions on Automatic Control*, 34:54–66, 1989.

[RM98] R.Y. Rubinstein and B. Melamed. *Modern simulation and modeling.* Wiley series in probability and Statistics, 1998.

[Sch98] W.R.W. Scheinhardt. *Markov-modulated and feedback fluid queues.* PhD thesis, University of Twente, the Netherlands, 1998.

[Sen90] B. Sengupta. The semi-Markov queue. *Stochastic Models*, 6:383–413, 1990.

[Ser80] R.J. Serfling. *Approximation Theorems of Mathematical Statistics.* Wiley Statistics Series, 1980.

[SG94] F. Schreiber and C. Görg. Rare event simulation: a modified RESTART method using the LRE-algorithm. In *Proceedings of the 14th International Teletraffic Congress*, pages 787–796. North-Holland, 1994.

[Sie76] D. Siegmund. The equivalence of absorbing and reflecting barrier problems for stochastically monotone Markov processes. *The Annals of Probability*, 4:914–924, 1976.

[VA98] J. Villén-Altamirano. RESTART method for the case where rare events can occur in re-trials from any threshold. *AEÜ International Journal of Electronics and Communications*, 3(52):183–189, 1998.

[VAMMGFC94] M. Villén-Altamirano, A. Martínez-Marrón, J. Gamo, and F. Fernández-Cuesta. Enhancement of the accelerated simulation method RESTART by considering multiple thresholds. In J. Labetouille and J.W. Roberts, editors, *Proceedings of the 14th International Teletraffic Congress, The Fundamental Role Of Teletraffic in the Evolution of Telecommunications Networks*, pages 797–810, 1994.

[VAVA91] M. Villén-Altamirano and J. Villén-Altamirano. RESTART: A method for accelerating rare event simulations. In J.W. Cohen and C.D. Pack, editors, *Proceedings of the 13th International Teletraffic Congress, Queueing, Performance and Control in ATM*, pages 71–76, 1991.

[VAVA97] M. Villén-Altamirano and J. Villén-Altamirano. RESTART: An efficient and general method for fast simulation of rare events. Technical report, Departamentado de Matemática Aplicada - Universidad Politécnica de Madrid, 1997.

[VAVA99] M. Villén-Altamirano and J. Villén-Altamirano. About the efficiency of RESTART. In *Proceedings of the RESIM'99 Workshop*, pages 99–128. University of Twente, the Netherlands, 1999.

Index

apparent bias, 92
Asian option, 128
asymptotic distribution, 78, 104
asymptotic efficiency, 64
asymptotic level, 78, 81
asymptotic optimality, 63
ATM, 64
ATM switch, 124
author, 165

Bayes, 5
Bernoulli, 17
bias, 42
Binomial, 47, 48, 52
Binomial distribution, 103
bootstrapping method, 31, 104
bootstrapping problem, 117
bottleneck, 61
branching process, 30, 45
byte, 44

cells, 64
change of measure, 31, 33, 77
complexity, 21, 33, 40
complexity properties, 110
conditional, 47, 48
contour levels, 105
contours, 13
correction, 54
cut-off, 41

dam model, 11
decay rate, 77, 80, 82, 86
dependency, 50
die-out, 29, 45
differential equations, 75
Dirac function, 75
discrete parameters, 36
discrete skeleton, 97
discrete state space, 94
double-barrier queue, 133
DPR, 32
driving process, 136
driving sequence, 131
dual process, 129
duality method, 4
dynamic programming, 37

efficiency gain, 108
eigenvalue equation, 78
empirical entrance distribution, 118
entrance distribution, 4, 46, 48, 71
entrance state, 44
Erlang distribution, 59
exponential decay, 42, 76, 90

fiber-glass, 2
first passage time, 128
fixed assignment, 118
fluid limit, 95
fluid queue, 75

gain, 59, 81
Galton-Watson process, 30
generator matrix, 74
geometric distribution, 52

GI/G/1 queue, 12
Global Step, 43, 58

Hurst parameter, 124
hysteresis, 54

importance factors, 32
importance function, 4, 11, 35, 41, 54
importance sampling, 10, 21, 31, 33, 62, 71, 77, 92, 105, 108, 110
importance splitting, 10
induction, 93
integer programming problem, 37
intermediate thresholds, 39
Internet, 1
intuitive, 33

ladder height, 11
Lagrange relaxation, 116
Laplace-Stieltjes transform, 78
large deviations, 92, 105
likelihood, 99
likelihood ratio, 10
Lindley recursion, 132
LRE method, 5

M/M/1 queue, 19
machines, 62
Markov, 5
Markov process, 11, 59, 60, 63, 66, 74, 120
Markov-modulated process, 136
martingale, 29
memory requirements, 44
MMPP queue, 77
Monte Carlo, 8, 33, 53, 113
multi-dimensional state space, 87
multi-threading, 44
multinomial, 47

offspring, 17
order statistics, 38
overflow probabilities, 87
overflow probability, 11

parallels, 29
parametric importance function, 97
Pareto distribution, 65, 124
particles, 5, 6
Petri nets, 5
Poisson process, 59, 74
probability measure, 31

quantile, 38
quasi-reversible, 97
queueing theory, 1

Radon-Nykodym derivative, 99
random assignment, 46, 118
Random Walk, 132
randomization, 37
rare event, 5
rare event simulation, 86
rare events, 2
rarity parameter, 33, 64
re-split, 49, 50
regeneration cycle, 112
regeneration points, 98
regenerative processes, 112
Relative Error, 58
relative error, 9, 61
Relative Time Variance product, 58
remaining service time, 48, 66
RESTART, 35, 48, 63, 64
RESTART method, 5, 122
restoration costs, 49
reverse time simulation scheme, 97
ruin probability, 29
rule of thumb, 38, 43, 55

sample path, 11
saved state, 73
self-similarity, 64
Set Partitioning, 95

single server queue, 87, 132
Single Step, 43
skip-free, 39
skip-free to the right, 89
speed-up, 59
splitting method, 5, 10
stages, 12
starting state distribution, 101
stationary distribution, 97
steady state, 4
stochastic recursion, 130
success probability, 73
surviving path, 49

tandem Jackson network, 105
tandem queue, 60, 84
telephone, 1
test statistic, 79
threshold, 11
thresholds, 35, 37
tilting parameters, 63
time-reversed process, 98
trial run, 38
truncation, 41, 54, 63
two-stage splitting simulation, 92

unbiased, 54
uniformization, 62

variance contribution, 50
variance estimation, 45
variance reduction, 82

weight, 39, 49
work-balanced variance, 58

Samenvatting

De afgelopen vijftig jaar is het vakgebied van de schatting der kansen op zeldzame gebeurtenissen enorm ontwikkeld, mede dankzij de groei aan kracht van rekenmachines. Omdat het nog steeds niet mogelijk is en ook niet mogelijk zal worden om de kans op zeldzame gebeurtenissen direct efficiënt te schatten is een veelvoud aan technieken ontwikkeld in de literatuur en opgenomen in simulatiepakketten. De belangrijkste twee methoden berusten op eenvoudige principes.

- **Importance Sampling**
  Verander het systeem zodanig dat de te schatten kansen groot worden. Men kan dan de oorspronkelijke kansen terugkrijgen doordat men de uitgevoerde transformatie verrekent in de geschatte kansen.

- **Importance Splitting**
  Verander de simulatiepaden zodanig dat veelbelovende paden splitsen in meerdere kleine paden. Op deze wijze verkrijgt men meer activiteit in het veelbelovende gebied zodat men vaker de zeldzame gebeurtenis zal zien.

Beide technieken zijn al circa vijftig jaar oud en hebben zich ontwikkeld in velerlei richtingen. In dit proefschrift hebben wij ons beperkt tot het onderzoek aan de laatste methode. De “splitting” methode is in de laatste vijftig jaar een aantal keren “herontdekt” in de literatuur en in de laatste incarnatie in de jaren negentig pas echt volwassen geworden. Daarvoor was het slechts mogelijk zeer beperkte modellen efficiënt te simuleren. In dit proefschrift wordt een wiskundige basis gelegd voor de methodiek, de efficiëntie en complexiteit van het algoritme. Nieuwe technieken worden ontwikkeld en worden vergeleken op basis van efficiëntie op een scala aan referentiemodellen. We zien dat alle bekende problemen met de methode bestreden kunnen worden met behulp van technieken binnen de methode. De combinatie van de analytische benadering van de gevonden methoden en de praktische toetsing levert een strategie op waarvan de efficiëntie optimaal is voor een zeer breed scala aan modellen en problemen. Het geheel aan methoden is verzameld in een simulatieprogramma dat is toegespitst op deze methode. De praktische toepassing van het onderzoek waarvan verslag is gedaan in dit proefschrift is te vinden in moderne communicatienetwerken waarin men geïnteresseerd is in de kwaliteit van diensten die geleverd worden aan de klant. De (hopelijk) zeldzame gebeurtenissen betreffen dan het verlies van gegevens die de klant wenst te versturen over het netwerk. Natuurlijk is de methodiek toepasbaar op elk probleem waarin men geïnteresseerd is in de kans op het plaatsvinden van een zeldzame gebeurtenis, maar de meest voorkomende praktische problemen met het schatten van zeldzame gebeurtenissen komen voor in de telecommunicatie.

Summary

The past fifty years the field of the estimation of rare event probabilities has grown considerably, partly because of the enormous growth in computing power during this period. Because it still is and will not ever be possible to estimate these probabilities efficiently using standard techniques a multitude of methods has been developed and described in the literature and incorporated into simulation packages. The single two most important techniques are based on simple principles.

- **Importance Sampling**
  Change the system in a way that makes the probabilities to estimate become large, so that standard methods can be applied again. One can get the original probabilities back by accounting for the system transformation used.

- **Importance Splitting**
  Change the paths traversed in the simulation in a way that the promising paths are split into a multitude of lightweight paths. In this manner one obtains more activity in the interesting area and one will see the rare event happening more frequently, making the estimates better.

Both techniques are about fifty years old and have evolved in several directions. Here we have limited the research to the latter method. The splitting method has been reinvented a number of times in the literature and it has only reached maturity during the last decade. Before, the method was limited to simple models and it was not asymptotically efficient. In this thesis a mathematical foundation is developed for the used methods; the efficiency and complexity of the basic algorithm are also derived. New techniques are developed and compared based on efficiency measures on a broad range of reference models. We see that all known problems and limitations can be dealt with using new techniques that enrich the splitting method. The combination of the analytical approach of the proposed methods and the validation in practice produces a strategy whose

efficiency is optimal for a broad class of models and problems. The collection of methods and techniques is gathered in a tool designed explicitly to solve a broad range of rare event problems. The practical application of the work reported here in this thesis can be found in modern communication networks where one is interested in the quality of service delivered to the customer. The (hopefully) rare event in such a setting is the probability of loss of data which the customer wishes to transfer over the network. The presented method will apply to many rare event problems; the currently most common practical use for the method is probably the telecommunications area.

Over de auteur

Marnix Garvels werd geboren op 7 juni 1971 te Ter Apel, Nederland. Hij bezocht de R.K. basisschool Bonifatius te Ter Apel en het Atheneum op de Rijks Scholen Gemeenschap te Ter Apel, alwaar hij in 1989 zijn V.W.O. diploma ontving. Hij vervolgde zijn opleiding aan de Universiteit Twente aan de faculteit der Toegepaste Wiskunde en behaalde daar in 1995 zijn ingenieursdiploma met het afstudeerwerk ”Semiparametrische Dichtheidsschatting” in de vakgroep der Statistiek. Hij besloot zijn vakkennis te vergroten door een baan aan te nemen als AIO bij het Centrum voor Telematica en InformatieTechnologie in 1996 met als opdracht ”Simulation Techniques in Broadband Telecommunication Networks”, en de vrucht van het daaropvolgende vierjarig onderzoek is bevangen in dit proefschrift. De komende tijd zal hij werken aan een nieuw project genaamd “Cross Entropy and RESTART” wat een vervolg is van het hier gepresenteerde werk.

About the author

Marnix Garvels was born on June 7th, 1971 in the town of Ter Apel in the Netherlands. He attended the elementary school Bonifatius in his home-town and the Atheneum on the Rijks Scholen Gemeenschap also in Ter Apel, whence he received his V.W.O diploma in 1989. He then extended his education at the University of Twente in the Faculty of Applied Mathematics, and received his Engineering diploma in 1995 with the work ”Semi-parametric Density Estimation” in the Department of Statistics. He then decided to improve his research skills by accepting a position as a PhD student at the Center for Telematics and Information Technology in 1996 with the assignment ”Simulation Techniques in Broadband Communication Networks”, the fruit of which is comprised in this dissertation. The upcoming time he will spend working on a new project called “Cross Entropy and RESTART”, which is a continuation of the work presented in this thesis.