ABSTRACT

Title of Document: CHARACTERIZATION OF GRADIENT ESTIMATORS FOR STOCHASTIC ACTIVITY NETWORKS

Renato Mauricio Manterola, MS, 2011

Directed By: Professor Michael C. Fu
Decision, Operations and Information Technologies Department

This thesis aims to characterize the statistical properties of Monte Carlo simulation-based gradient estimation techniques for performance measures in stochastic activity networks (SANs) using the estimators' variance as the comparison criterion. When analyzing SANs, both performance measures and their sensitivities (gradient, Hessian) are important. This thesis focuses on analyzing three direct gradient estimation techniques: infinitesimal perturbation analysis, the score function or likelihood ratio method, and weak derivatives.

To investigate how statistical properties of the different gradient estimation techniques depend on characteristics of the SAN, we carry out both theoretical analyses and numerical experiments. The objective of these studies is to provide guidelines for selecting which technique to use for particular classes of SANs based on features such as complexity, size, shape and interconnectivity. The results reveal that a specific weak derivatives-based method with common random numbers outperforms the other direct techniques in nearly every network configuration tested.

CHARACTERIZATION OF GRADIENT ESTIMATORS FOR STOCHASTIC ACTIVITY NETWORKS

by

Renato Mauricio Manterola

Thesis submitted to the Faculty of the Graduate School of the
University of Maryland, College Park, in partial fulfillment
of the requirements for the degree of
Master of Science
2011

Dedication

To my family: Paulina and Alonso.

Acknowledgments

I owe my deep gratefulness to all the people who have made this thesis possible.

First, I would like to thank my advisor Professor Michael Fu, not only a bright faculty and inspiring teacher but also an extraordinary person. I was very lucky to learn from him and to be part of one of his graduate classes. He was always accessible and willing to help me, so I will always be grateful for his guidance.

I would also like to thank my thesis committee. I was an honor to have in my committee the professors I liked the most during my coursework: Professor Steve Marcus and Professor Adrian Papamarcou.

I also want to mention the great job done by Tracy Chung and Melanie Prange in the graduate studies office for their dedication and consideration in academic and personal matters. They were the kind and lovely face of the prestigious Department of Electrical and Computer Engineering.

Words cannot express the gratitude I owe my family, my wife Paulina, my son Alonso, and relatives. Especially to Patricia, who came to the United States to help us out when things got difficult. Also, I am very grateful of the constant support and help from Susana and Marcelo, who were always present even from the distance. To my grandmother Marina and my mother Lenka, who were always concerned about me and whose phone calls provided me with the strength to keep going.

I am very grateful of my dear friends and fellow graduate students here at College Park and DC: Karina, Magaly, Daniela, José, Isabel, Pablo, Eduardo, Paula, Danae, Enrique, Peter, Nadia, Nazre, Maice, Kai Sum, Hien to name a few. By sharing many difficult and also happy experiences I was able to overcome the obstacles and to share a friendship that I hope it will last forever.

Thanks

Table of Contents

List of Tables

List of Figures

List of Abbreviations

SAN	Stochastic Activity Network
IPA	Infinitesimal Perturbation Analysis
SF	Score Function
LR	Likelihood Ratio
RV	Random Variable
IID	Independent and Identically Distributed
WD	Weak Derivative
PDF	Probability density function
CDF	Cumulative density function
PMF	Probability mass function
CMF	Cumulative mass function
w.r.t.	With respect to
A-R	Acceptance-Rejection
CRN	Common random numbers
wp	With probability
CPM	Critical path method
PERT	Program evaluation and review technique

Chapter 1

Introduction

1.1 Overview

The main purpose of this work is to compare the precision of three direct gradient estimation techniques in estimating performance measures of stochastic activity networks (SANs) using Monte Carlo simulation. The approach relies mainly on critical path method (CPM), program evaluation and review technique (PERT) and gradient estimator approaches to implement algorithms. The variance of the estimates is used as the comparison criterion. Therefore, the main focus of this study is the precision of estimators and how that precision relates to certain SAN characteristics.

When dealing with SANs, it is of interest to obtain sensitivities (gradients, derivatives, or Hessians) along with the usual performance measures, such as the expected completion time and activities criticality. These sensitivities are often required to perform sensitivity analysis and optimization. Because some practical problems may arise when implementing direct gradient estimation techniques using Monte Carlo simulation, practitioners usually prefer to conduct sensitivity analysis using finite differences estimation instead. However, finite differences estimation may be less accurate, more time consuming, and less computationally efficient, particularly in the case of high dimension estimates.

It is necessary to clarify that when a closed-form expression is difficult to obtain and other numerical methods are not applicable, Monte Carlo simulation provides an alternative way to estimate such performance measures and their gradients. Because implementing direct gradient estimator techniques presents some practical implementation challenges, this work aims to provide guidelines for the selection of the most accurate gradient estimation technique depending on the characteristics of a SAN.

The main objective of this endeavor is the assessment of how well each of the direct gradient estimation methods developed in the last few decades estimate SANs sensitivities in terms of the sample variance. The gradient estimator methods which will be studied are infinitesimal perturbation analysis (IPA); score function also known as likelihood ratio (SF/LR); and weak derivatives method, also know as measure-valued differentiation based gradient estimation (WD or MVD).

We investigated the relationship between the accuracy of the estimator and the SAN features encoded by the network complexity measures. In order to uncover this relationship, theoretical analysis and experimental results from MC simulations will be examined.

The results of this study indicate that WD based gradient estimation presents a lower variance, but IPA and SF/LR may be more computationally efficient because the estimators require only a single simulation, whereas WD estimators require multiple simulations. Therefore, it can be concluded that aside from other considerations such as ease of implementation or the speed of the algorithms, WD gradient estimation is superior in terms of precision.

1.2 Statement of the problem

Because Monte Carlo simulation is a computationally costly method to implement, other techniques may be preferred, such as simpler computational methods or obtaining closed-form expressions. However, when dealing with complex and/or numerous interactions (most physical systems), closed-form expressions are almost impossible to obtain, and less sophisticated computational methods are unable to model such levels of complexity. In these cases, and particularly when dealing with high dimension systems, Monte Carlo simulation constitutes an alternative method of implementing and analyzing stochastic systems. This work focuses on stochastic activity networks (SANs) and Monte Carlo simulation as tools to model complex stochastic systems and obtain their performance measures. SANs have an ample variety of applications, from project management to analysis of communication systems [12]. In these kinds of applications, analysts usually must compute certain performance measures, such as expected completion time, task criticalities, priorities, and total time delay, among others. Additionally, the derivatives of such performance measures are also of interest to analysts, mainly to perform system optimization and sensitivity analysis. In the case of complex SANs, both performance measures and their respective derivatives can be obtained using Monte Carlo simulation. Despite the availability of direct gradient estimation methods that have been developed in the past few decades, which are more efficient than indirect techniques for estimating derivatives in stochastic simulation, some researchers still depend on resimulation for gradient estimation. This approach, known as finite differences, requires changing the input parameter of interest a small amount and looking the deviation in the output performance measure. This makes Monte Carlo simulation even more inefficient and cumbersome, especially when there is a need to obtain a gradient with respect to various differentiation parameters. Unlike indirect gradient estimation methods, which do not require knowledge of the internal functioning of a system and only estimate an approximation of the true gradient value [10], direct gradient estimation techniques require some knowledge about the internal dynamics of the system or the input distributions. As such, these estimators are model-dependent [10]. The three main methods of estimating derivatives directly are the family of perturbation analysis (IPA, SPA, PA, etc.), likelihood ratio (LR) and weak derivatives (WD). perturbation analysis and likelihood ratio methods allow researchers to directly obtain (in the same simulation run) both performance measures and their respective derivatives by exploiting the information gathered in every repetition from the original system. WD may also require resimulation but gives unbiased estimates. Because of the potential benefits of using these gradient estimation techniques, this work aims to provide researchers with some guiding strategies to determine in which cases the use of a particular estimator may be more effective and useful, depending on the types of networks and performance measures being considered. This thesis implements and compares the performance of direct gradient estimators using estimation precision as the main criterion for comparison and assessment. Theoretical and experimental analyses will be conducted to investigate possible relationships between the performance of the gradient estimators and some particular characteristics of the SANs, such as the size, distributions and/or shape of the activity networks.

1.3 Review of the literature

In this section, an examination will be presented of previous work that is relevant to the problem setting on which this study focuses, i.e., stochastic activity networks and gradient estimation using Monte Carlo, as well as the measurement of network complexity from the graph theory point of view.

1.3.1 Stochastic activity networks and gradient estimation using Monte Carlo simulation

Fu [8] provides a review of the different gradient estimation techniques, which provides the main theoretical foundations for this study. Fu [8] classifies gradient estimation techniques into two categories: direct and indirect methods. When using indirect techniques (finite differences and simultaneous perturbations) the estimate is indirectly obtained by computing the differences between the simulation outputs. Although these techniques are easier to implement than direct gradient methods, they are less efficient due to the high computational cost associated with the need to perform multiple resimulations. Consequently, obtaining accurate estimations may be very time consuming. In contrast, direct gradient techniques (perturbation analysis (PA), score function/likelihood ratio (SF/LR), and weak derivatives (WD)) may be more difficult to implement because an actual estimator needs to be designed and analytically obtained for each specific problem setting before running the simulation (off-line work). A direct estimator is used during the same simulation run to obtain gradient estimates as well as estimates of other performance measures. In other words, the simulation output of a single run (of numerous repetitions) includes both performance measure estimates and their gradient estimates. In [9], Fu developed three direct gradient estimators specialized for critical path related measurements in the particular context of stochastic activity networks (SANs). He uses activity on the arc (AoA) SAN modeling and presents some examples with exponentially distributed activity times. The main contribution of this paper to this present work is that it provides ready-to-use definitions of gradient estimators for three different performance measures: 1) the derivative of the project completion time with respect to the mean time of one or several activities of the network, 2) the gradient of the probability that the critical path of a network will surpass a threshold value, and 3) the derivative of the tail distribution with respect to the lower limit of such distribution. Following paper suggestion of conducting further theoretical and experimental analysis to compare the variance properties of the estimators he provides, this thesis builds on Fu's work to expand the understanding of the functioning of these estimators and how they behave in terms of their estimation precision. Another work conducted by Fu is also relevant to this study. In [10], the author summarizes the main direct gradient estimation techniques, refers to the applications in which these estimators are generally used, and provides an overview of some of the challenges that may arise when trying to implement these different techniques using Monte Carlo simulation. This work sheds light on when it is appropriate to use each and the precision of the estimates obtained with each method. As such, this work constitutes a starting point for the present research.

In another study relevant to this thesis, Groër and Ryals [12] building on Fu's work [9], implemented two indirect techniques (finite differences with common random numbers and with independent variates) and three direct gradient estimation methods (PA, SF/LR, and WD) to estimate two performance measures related to the longest path through a SAN and compared these techniques in relation to the variance of the estimates. The authors found that IPA performed better than other estimators and that the WD estimator with common random numbers (CRN) can be as good as IPA/SPA. Groër and Ryals also proposed the use of a combination of results from different methods to achieve variance reduction.

Heidergott et al.[13] also obtained direct gradient estimators. Unlike previous works conducted in this area, these authors derived the estimators for systems with Gaussian input distributions. The authors proposed analytical expressions for the variance of the estimator using a simplistic but mathematically manageable approach in which the output is computed as a polynomial function of a single stochastic input. They showed that the estimator based on weak derivatives outperforms IPA. Then, they also empirically compared these three estimators using a computational simulation, in a similar way to Groër and Ryals. Heidergott and colleagues concluded that WD techniques with CRN favorably compare to IPA in terms of the precision of the estimation. It is worth mentioning that the present study draws from Heidergott et al.'s work to implement the algorithms to compute the gradient estimators in Gaussian systems by means of Monte Carlo simulation.

1.3.2 Complexity measures for graphs

During the last decades, there has been interest in the field of Graph Theory in the characterization of the complexity of a graph. Many reasons have been given for this interest, including the reduction of complexity, measurements of the capacity to perform some task, description of a social network, potential use as a predictor of the effort needed to analyze, to decompose or to construct some subsystem, among others (e.g.[5]). Each of these reasons is relevant to different areas of study. The concept of complexity, although almost natural in a loose context, is vague in general, and it has been difficult to define and establish it in a formal and unique way.

There have been multiple attempts throughout the literature to tackle the problem of defining a meaningful quantity for the measurements of complexity. Also, complexity seems to have a different meaning in different scientific areas, given the rise of several operational and/or abstract definitions; examples of this include Kolmogorov complexity or Krohn-Rhodes complexity. Additionally, there is no agreement on which properties this quantity should satisfy. For instance, if there are two activity networks and they are to be connected in a serial fashion, should it be expected that the total complexity of the new network must be the sum of the individual ones? What if the networks are combined in parallel?

In the specific area of CPM/PERT, the first complexity measure proposition was made by Pascoe [20] for use in resource allocation. The coefficient of network complexity proposed by Pascoe (CNCp) is simply defined as the ratio of nodes over arcs.

$$CNC_p = A/N \tag{1.1}$$

.

This quantity takes the idea of normalization by the number of nodes, for far comparison, i.e., more arcs per node is synonymous to greater complexity in this approach. Although this idea seems sensible, its validity is debatable.

Similarly, Kaimann [14], proposed his own CNC. He wanted a quantity to measure the degree of relationship between events (nodes), which needed to be simple enough for simulation analysts to use and able to capture the non-linear increase in processing time when complexity is increased. Consequently, Kaimann defined CNC as follows:

$$CNC_k = A^2/N \tag{1.2}$$

Obviously, the same advantages are present as in Pascoe's CNC, such as simplicity; it is easy to understand and compute. The same drawbacks are also present when using this coefficient, such as shorter processing times for a network with a large CNC_k number.

The cyclomatic number (S) and the Davies [3] Coefficient of Complexity CC (also know as CNC_d) possess similar benefits of ease and elegance as well as similar drawbacks of overly simplistic coefficient.

$$S = A - N + 1 \tag{1.3}$$

This number counts the number of cycles of an undirected graph. In the case of activity networks, Elmaghraby [7], presents a very interesting interpretation of this number. Elmaghraby suggests that it is the number of operations which must be performed in order to find the critical path. For instance, in a purely serial network there are exactly $N-1$ arcs, hence $S=0$. That is correct since the single path is the critical path, so no comparisons are necessary. But for each arc added, a comparison must be made in order to find the critical path, i.e. $A-(N-1)$ operations. The cyclomatic number has some interesting features. It depends on the topology of the network, so it is not affected by the separation of joining series activities. Also, the combination of networks in a chain results in the sum of cyclomatic numbers.

Th Davies CNC is a normalized quantity which takes values between 0 and 1.

$$CNC_d = \frac{A-N+1}{(N-1)(N-2)} \tag{1.4}$$

Davies' main objective in the definition of this number was to compare different sorting criteria for resource allocation in various networks. CNC_d and other quantities were just part of a "computer-based experiment in the scheduling of multi-activity projects with limited resources".

The number of trees descriptor counts the number of trees rooted in the destination node. Remember that a tree is a connected graph without cycles. The computation of this number and the reason it is considered a complexity descriptor were developed by Temperley [23]. The computation of this number involves the obtainment of the minor D_{NN}, where D is the Laplacian matrix.

$$D = \begin{bmatrix} \sum_{j\neq 1} a_{1,j} & -a_{1,2} & \dots & -a_{1,N} \\ -a_{2,1} & \sum_{j\neq 2} a_{2,j} & \dots & -a_{2,N} \\ \vdots & \vdots & \ddots & \vdots \\ -a_{N,1} & -a_{N,2} & \dots & \sum_{j\neq N} a_{N,j} \end{bmatrix} \tag{1.5}$$

where $A = [a_{i,j}]$ is the adjacency matrix, i.e., the matrix listing the weighted connection between nodes i and j.

The last descriptor considered for the present study is the restrictiveness estimator. Originally presented by Thesen [24], this index takes values in the range $[0, 1]$, where 0 means no restrictiveness. A true parallel network has $RT = 0$. On the other hand, a pure series network is "fully restrictive", and the measure is $RT = 1$. This number is an estimator of the restrictiveness of a directed graph, a number difficult to compute because it involves the enumeration of a large number of permutations, hence a hard combinational problem. The formula is,

$$RT = \frac{\sum_{i,j} r_{i,j} - 6\,(N-1)}{(N-2)\,(N-3)} \tag{1.6}$$

where $R = [r_{i,j}]$ is the reachability matrix, which is defined as $r_{i,j} = 1$ if there exists some path from node i to j, or $r_{i,j} = 0$ otherwise.

There are multiple definitions of complexity indices not directly related to CPM/PERT. As early as in Kaimann's paper, the work of Mowshowitz is cited regarding the extraction of the information content of a graph to be considered as a complexity measure. After the seminal work of Mowshowitz [18], a whole branch of information-theoretical-inspired measures has emerged. These indices basically consist of assigning weights to elements of a graph (to nodes, arcs or paths) according to some criterion. After that, these weights are interpreted as forming a PMF. Then the author is free to apply the entropy definition formula to obtain the desired index. This way of obtaining complexity indices is used to characterize chemical reactions or biological interactions. It is also used in the area of network physics. For a thorough review of these information theoretical based methods, please refer to the work of Dehmer, Mowshowitz and Emmert-Streib [4].

In this thesis, we wish to investigate the applicability of the various complexity indices to our study.

1.4 Organization of the chapters

The rest of the thesis is organized as follows. Chapter 2 explains how the analysis will be developed, presents some background relevant to the subsequent chapters, and addresses some practical issues and steps followed to obtain the algorithms to run the Monte Carlo simulation. Chapter 3 derives analytical expressions for the variance of the estimators for some network structures. In Chapter 4, Monte Carlo simulation (MCS) is used to empirically assess the precision of the estimates for each technique. Chapter 5 discusses the main findings and suggests future directions for further research.

Chapter 2

Methodology

2.1 Overview

In order to understand the gradient estimator methods in SANs, it is useful to first uncover relationships in some relatively simple and mathematically tractable networks, and then check how reusable these discoveries are in more complex networks. This second step is performed by means of Monte Carlo simulation (MCS) in MATLAB and C over a set of SANs specially created for this study. In this step, complexity indices or measures were computed for every SAN to characterize the "intricacy" of the network for comparison with respect to (w.r.t.) the gradient estimation.

This chapter presents background information about graph theory and gradient estimation. Then it presents an explanation of how the programming code work was performed to obtain the estimates and finally how the set of complex SANs was generated.

2.2 Theoretical and conceptual background

2.2.1 Stochastic activity networks (SANs)

A SAN is a directed graph with edge weights of random values [1]. There are two ways of representing SANs. One approach is to assume the activity on the arc (AoA) and the other is to consider the activity on the node (AoN) representation. In this thesis, the focus will be on the AoA representation.

SANs are often conceptualized as a project in which nodes correspond to milestones or events, arcs or edges represent tasks to be completed, and arc weights denote the time required for task completion. The latter must obey a particular precedence order to reach a final milestone. This presumes the presence of an acyclic network as a necessary condition for a legitimate precedence order.

Another interpretation involves the consideration of the SAN as portraying alternative ways or paths to transport materials and/or resources between different geographical points. In this case, weights represent distances or costs of moving from one point to the next, as in the traveling salesperson problem. Another common interpretation involves the assumption that edges are routes for moving information packets with corresponding delays in a communication network. Although most of these interpretations are independent from the concepts discussed in here, for the purpose of this work, SANs will be understood from a project management standpoint.

Formally, let $\mathcal{G} = V(\mathcal{N}, \mathcal{A})$ be a directed acyclic graph, where $\mathcal{N}$ is the set of nodes with finite cardinality N, and $\mathcal{A}$ is the set of ordered pairs (i, j) called arcs with finite cardinality A. Ordered pairs are formed from a set $\mathcal{N}$ or a subset of $\mathcal{N}$. The first entry of each arc pair is an outgoing node and the second one is the incoming node. In other words, for arc (i, j), i is the start node and j is the end node. An example of the graphical rendering of this construction is portrayed in Figure 2.1 (example taken from [1]).

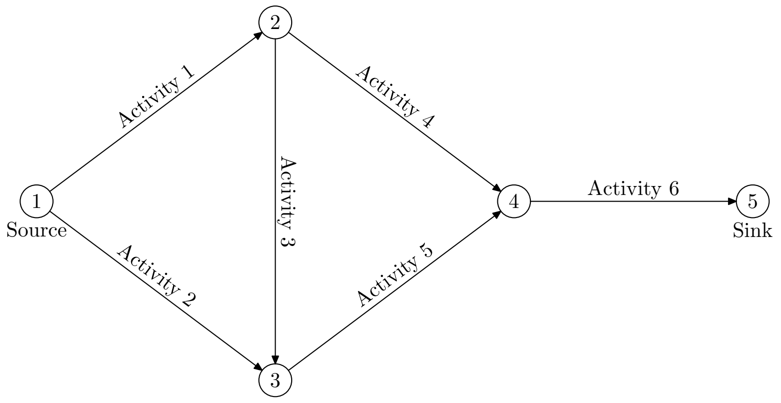


Figure 2.1: This figure depicts a most common graphical interpretation of an Activity Network.

Now let us define a **path**. A path (denoted by character P) is a sequence of arcs and the corresponding sequence of nodes. The first node of the sequence is called the source node and the last node is called the sink or destination node. If the sequence of arcs (and corresponding nodes) has no repeated arcs (and nodes), then the path is called a **simple path**. A cycle is defined as a path in which the source and the sink nodes are the same. Since this thesis is focused on acyclic directed graphs, the activity networks analyzed here will only include simple paths.

If a node can be reached from another node following a particular sequence of arcs (a path), this means that the nodes are connected. The connectivity of a pair of nodes can be defined as the number of independent paths that connect them [1].

In the field of project management, performance measures of interest usually consist of the expected completion time of the project and the probability that the critical path will exceed a certain threshold value. The critical path is one of the most relevant concepts derived from CPM/PERT techniques. When the activity times are deterministic, the determination of the critical path is equivalent to finding the longest path from the source to the sink node, because the precedence order and completion of activities in the critical path constrain and determine the start of the subsequent tasks in the network to complete the project.

However, if activity times are stochastic in nature, the previous definition of criticality falls short, because in SANs almost every path has a non-zero probability of being critical. Therefore, in these cases there is no single critical path; instead, there is a degree of criticality, which is referred to as the criticality index. Criticality index computation and estimator is a large and interesting topic by itself. Refer to [6] for further discussion of this topic or [2] for a method that can be used to estimate criticality indices.

2.2.2 Gradient estimation techniques

Consider a system in which a performance measure of interest Y (output) is a function of a vector of inputs $\mathbf{X}$. If $\mathbf{X}$ is a random vector, the output Y is also a random variable. Hence, the expected value of Y or some other statistic of Y should be of interest. From the "Law of unconscious statistician" [21]:

$$E[Y] = \int y \mathrm{d}F_Y(y) = \int Y(\mathbf{x})\, dF(\mathbf{x}) = \int Y(\mathbf{x})\, f(\mathbf{x})\, \mathrm{d}\mathbf{x} \tag{2.1}$$

where $F_Y(\cdot)$ is the CDF of Y, $F(\cdot)$ and $f(\cdot)$ are the multivariate CDF and PDF of $\mathbf{X}$, respectively. Notice that just the first integral is scalar and the other two are multiple integrals.

Taking the derivative w.r.t. some implicit parameter θ, the following can be obtained.

$$\frac{dE[Y]}{d\theta} = \frac{d}{d\theta} \int Y(\mathbf{x})\, dF(\mathbf{x}) = \int \frac{d}{d\theta} Y(\mathbf{x})\, \mathrm{d}F(\mathbf{x}) \tag{2.2}$$

Certain conditions need to be met in order to interchange the order of the integral and the derivative. These conditions are related to the concept of uniform integrability and the dominated convergence theorem which establish sufficient conditions to ensure that $\mathrm{E}[Z_n] \to \mathrm{E}[Z]$ as $n \to \infty$, where Z_n and Z are RVs such that $Z_n \underset{n\to\infty}{\to} Z$ with probability 1 (almost surely). In other words, the following interchange of expectation and the limit is required:

$$\mathrm{E}\left[\lim_{n\to\infty} Z_n\right] = \lim_{n\to\infty} \mathrm{E}[Z_n] \tag{2.3}$$

Determining whether the dominated convergence theorem conditions are met is beyond the scope of this thesis. For more details, see [10].

If we consider that the derivative parameter θ is part of the sample vector $\mathbf{X}$ by some functional dependency and that the interchange is valid, then we can write:

$$\frac{dE[Y]}{d\theta} = \int \frac{\partial Y(\mathbf{x})}{\partial \mathbf{x}} \frac{d\mathbf{x}}{d\theta} \mathrm{d}F(\mathbf{x}) \tag{2.4}$$

On the other hand, if we suppose that random vector $\mathbf{X}$ is generated by a continuously differentiable family of CDFs, parametrized by θ, we can write:

$$\frac{dE[Y]}{d\theta} = \int Y(\mathbf{x}) \frac{\partial f(\mathbf{x})}{\partial \theta} \mathrm{d}\mathbf{x} \tag{2.5}$$

Notice that equation (2.4) can be used in the Monte Carlo simulation. It is just necessary to obtain the derivative of the output Y w.r.t. $\mathbf{X}$ and the derivative of the sample path $\mathbf{X}$ w.r.t. θ, which is application dependent.

For instance, consider the case in which $Y = \min(X_1, X_2)$. Also assume that X_1 and X_2 are exponentially and independently distributed RV. Let us assume we need to find the sensitivity w.r.t. the first arc mean, i.e. $\theta = \beta_1$. Therefore, we can write:

$$X_1 \sim \exp(\beta_1), \quad X_2 \sim \exp(\beta_2), \quad Y = \min(X_1, X_2) \tag{2.6}$$

$$\frac{\partial Y}{\partial \beta_1} = \frac{\partial X_1}{\partial \beta_1} \cdot \mathbb{1}\{X_1 < X_2\} \tag{2.7}$$

The derivative $\partial X_1 / \partial \beta_1$ is not only application dependent but also representation dependent, which means that the derivative depends on the way the distributed variates are generated in the simulation (see more details in Fu & Hu [11], Chapter 1). In the particular case of exponential distribution, the most common solution for generating the samples is the inverse transform method, in which the source of randomness is a random number generator (RNG) of uniform distribution and by inversion of the CDF the random variates are obtained:

$$X_1 = F^{-1}(U;\beta_1), \quad \text{where} \quad U \sim U(0,1), \quad F(x,\beta_1) = 1 - e^{-x/\beta_1} \tag{2.8}$$

$$\Rightarrow X_1 = -\beta_1 \ln(1-U) \tag{2.9}$$

$$\Rightarrow \frac{\partial X_1}{\partial \beta_1} = -\ln(1-U) = \frac{X_1}{\beta_1} \tag{2.10}$$

In conclusion, for this construction we have:

$$\frac{\partial Y}{\partial \beta_1} = \frac{X_1}{\beta_1} \cdot \mathbb{1}\{X_1 < X_2\} = Z \tag{2.11}$$

Hence, a normal MCS can be performed and the estimation of sensitivity w.r.t. the mean of the first arc can be obtained at the same time. It is just necessary to compute $Z = X_1/\beta_1$ in each replication where $X_1 < X_2$ and $Z = 0$ otherwise. Then an estimate of the gradient can be obtained:

$$\frac{\mathrm{dE}[Y]}{\mathrm{d}\beta_1} = \mathrm{E}[X] \approx \sum_i Z_i/N \tag{2.12}$$

where Z_i are computed in each repetition by means of equation (2.11) and N is the number of replications performed.

The estimator $Z = X_1/\beta_1 \cdot \mathbb{1}\{X_1 < X_2\}$ is known as the IPA estimator w.r.t. β_1 in this specific case.

Now consider a situation in which θ is in the distribution function (equation (2.5)), and for exposition reasons also assume that the multivariate PDF $f(\mathbf{X})$ is separable w.r.t. the θ dependence. For example, take X_1 as the only RV with dependence on θ. Consequently, f is separable and can be written as:

$$f(\mathbf{X}) = f_{X_1}(x_1) f_{X_{1-}}(\mathbf{X_{1-}}) \tag{2.13}$$

and equation (2.5) can be expressed as:

$$\begin{aligned}\frac{dE[Y]}{d\theta} &= \int Y(\mathbf{X}) \frac{\partial f_{X_1}(x_1, \theta)}{\partial \theta} f(\mathbf{X_{1-}}) \mathrm{d}\mathbf{X} \\ &= \int Y(\mathbf{X}) \frac{1}{f_{X_1}(x_1)} \frac{\partial f_{X_1}(x_1, \theta)}{\partial \theta} f_{X_1}(x_1) f(\mathbf{X_{1-}}) \mathrm{d}\mathbf{X} \\ &= \int Y(\mathbf{X}) \frac{\partial}{\partial \theta} \ln\left(f_{X_1}(x_1, \theta)\right) f(\mathbf{X}) \mathrm{d}\mathbf{X} \end{aligned} \tag{2.14}$$

Equation (2.14) can now be used to estimate the gradient w.r.t. θ if we set:

$$Z = Y(\mathbf{x}) \frac{\partial}{\partial \theta} \ln\left(f_{X_1}(X_1, \theta)\right) \tag{2.15}$$

This type of estimator is called a SF/LR gradient estimator. For example, in the stochastic system described before, let us find the SF/LR estimator w.r.t. β_1. The PDF is separable if we consider X_1 and X_2 independent RVs.

$$f(\mathbf{x}) = f_{X_1}(x_1) f_{X_2}(x_2) \tag{2.16}$$

where

$$f_{X_1}(x_1) = e^{-x_1/\beta_1}/\beta_1 \tag{2.17}$$

Hence, by differentiation we get:

$$\begin{aligned}\frac{\partial f(\mathbf{x})}{\partial \theta} &= \frac{\partial f_{X_1}(x_1)}{\partial \beta_1} f_{X_2}(x_2) \\ &= \frac{e^{-x_1/\beta_1}}{\beta_1} \frac{1}{\beta_1} \left(\frac{x_1}{\beta_1} - 1\right) f_{X_2}(x_2) \\ &= \frac{1}{\beta_1} \left(\frac{x_1}{\beta_1} - 1\right) f(\mathbf{x}) \end{aligned} \tag{2.18}$$

Therefore, the SF/LR gradient estimator Z is equal to

$$Z = \frac{Y(\mathbf{X})}{\beta_1} \left(\frac{X_1}{\beta_1} - 1\right) \tag{2.19}$$

i.e., the random vector realization $\mathbf{X}$ is generated in each simulation repetition and Z is calculated each time using equation (2.19). Then the expected value is approximated by:

$$\frac{\partial \mathrm{E}\left[Y\right]}{\partial \beta_1} \approx \frac{\sum_i Z_i}{N}. \tag{2.20}$$

Next we consider the weak derivatives method (WD). In equation (2.5), assume that X_1 is the only component in $\mathbf{X}$ that has dependence on θ. If we express the density function derivative as follows

$$\frac{\partial f_{X_1}(x_1, \theta)}{\partial \beta_1} = c(\theta)\left(f_{X_1}^{(2)}(x_1, \theta) - f_{X_1}^{(1)}(x_1, \theta)\right) \tag{2.21}$$

where $f_{X_1}^{(1)}$ and $f_{X_1}^{(2)}$ are PDFs by themselves and are called the "phantoms" of f_{X_1}, then the gradient w.r.t. θ can be written as:

$$\begin{aligned}\frac{dE[Y]}{d\theta} =& c(\theta)(\int Y(\mathbf{x}) f_{X_1}^{(2)}(x_1) f_{X_{1-}}(\mathbf{x_{1-}}) \mathrm{d}\mathbf{x} \\ &- \int Y(\mathbf{x}) f_{X_1}^{(2)}(x_1) f_{X_{1-}}(\mathbf{x_{1-}}) \mathrm{d}\mathbf{x})\end{aligned} \tag{2.22}$$

obtaining the new gradient estimator:

$$c(\theta)\left(Y(X_1^{(2)}, \mathbf{X_{1-}}) - Y(X_1^{(1)}, \mathbf{X_{1-}})\right) \tag{2.23}$$

where $X_1^{(1)} \sim f_{X_1}^{(1)}$, $X_1^{(2)} \sim f_{X_1}^{(2)}$ and $\mathbf{X_{1-}} \sim f_{X_{1-}}$.

Thus, weak derivative estimators require two runs for each gradient estimation per parameter of interest, namely the one with the modified distribution $f_{X_1}^{(1)}$ instead of f_{X_1}, and another one with $f_{X_1}^{(2)}$.

In the $Y = \min\left(X_1, X_2\right)$ example, notice that the derivative of the exponential distribution can be expressed as:

$$\begin{aligned}\frac{\partial}{\partial\beta_1}\left(\frac{e^{-x_1/\beta_1}}{\beta_1}\right) &= \frac{1}{\beta_1}\left(\frac{x_1}{\beta_1^2}e^{-x_1/\beta_1} - \frac{e^{-x_1/\beta_1}}{\beta_1}\right) \\ &= \frac{1}{\beta_1}\left(f_{X_1}^{(2)}(x_1) - f_{X_1}(x_1)\right) \qquad (2.24)\end{aligned}$$

in which the expression in parentheses is the difference of an Erlang PDF and an exponential PDF, i.e., $f_{X_1}^{(2)} \sim Erl(2, \beta_1)$ and $f_{X_1} \sim exp(\beta_1)$. Because of the weak derivative used here, $f_{X_1}^{(1)} = f_{X_1}$, so the estimator is:

$$Z_i = \left(\min\left(X_1^{(2)}, X_2\right) - \min\left(X_1, X_2\right)\right)/\beta_1 \qquad (2.25)$$

and the gradient estimate can be approximated by:

$$\frac{\partial \mathrm{E}\left[Y\right]}{\partial\beta_1} \approx \frac{\sum_i Z_i}{N} \qquad (2.26)$$

These are the fundamentals of the three methods employed in this work.

2.3 Algorithms and network representation

Algorithms were developed in MATLAB and C. The random number generator (RNG) included by default in MATLAB, was used to obtain random variates needed for Monte Carlo simulation. Since MATLAB version 7, the generator uses Mersenne Twister (MT), a pseudo-random number generating algorithm developed by Makoto Matsumoto and Takuji Nishimura [17]. The RNG routines of MATLAB also provide tools for reseeding, resetting and creating multiple streams, in order to get more control over simulations. For C programs, the RngStreams package was used, which implements a combined multiple recursive RNG (CMR-RNG) proposed by L'Ecuyer in [16]. The code with implementations of the algorithms for gradient estimation is included in the appendix.

Variate generation is mainly done by inverse transform. First, a single stream is created and reset at the beginning of every Monte Carlo simulation. In special cases involving the weak derivatives method (also known as measure-valued differentiation), an additional and independent stream is created to generate the extra "phantom" samples. In this way, it is possible to achieve a fair comparison by obtaining synchronized estimates across different gradient estimation techniques.

Graphs are defined by their connection matrix. In this context, the connection matrix is a $N \times N$ sparse matrix type in which non-zero entries $[a_{ij}]$ represent the "weight" of the link between nodes i and j, and N represents the number of nodes.

For this implementation, the network definition routine fills the sparse connection matrix inserting "1"s as needed, which correspond to an arc in the graph. No weights are assigned at this point, meaning no activity times were yet specified. Activity times are set later in the MCS code since arc lengths are stochastic quantities.

In the actual MCS code, activity times are generated by the RNG for every edge (each non-zero entry in the connection matrix) in each repetition and transformed into a variate of the desired distribution. Every time a variate is needed, the rand() function is called. This function return a (pseudo)Random number drawn from uniform U(0,1) distribution. Random variates are then generated using inverse transform or acceptance-rejection (A-R) methods as needed, depending on the distribution of the activity times. For example, for Gaussian distributed activity times, the A-R approach was used, following the same ideas presented by Heidergott et al.[13].

Once the activity times are realized, the SAN can be considered a deterministic activity network and then the critical path needs to be determined. For this purpose, MATLAB Bioinformatics Toolbox provides some useful graph theory functions. The shortestpath function is extensively used in the implementations. Since we are interested in the longest path, the shortestpath() function is called with negative activity times and "acyclic" method type arguments in order to obtain the **critical path**. This procedure is warranted if the network contains no cycles. Note also that the shortestpath() algorithm will solve the original longest path problem in polynomial time, which is beneficial for large networks.

Once the longest path is determined, the gradient estimate is calculated using the formulas in Table 1 contained in [10]. The SF/LR estimator implementation is very straightforward, whereas IPA is bit more complicated because of the need to determine whether the activity of interest is in the critical path or not. On the other hand, implementation of the WD estimator requires the generation of additional activity times and the need to perform the longest path search for each modified activity network. In the worst case, two realizations of activity networks are needed for each sensitivity estimation desired (without counting the simulation runs needed for the original or unmodified system).

Additional coding was necessary for the computation of network complexity measures like Restrictiveness Index (RT) and Number of Trees (T). On the other hand, coefficients of network complexity (CNCp and CNCk) and cyclomatic number

(S) were calculated directly in the spreadsheet since they are very simple formulas.

Gradient estimation techniques were also implemented in C code for simple networks, where enumeration of all paths from source to destination was used to find the critical path. The C compiled code runs much faster which is particularly important for creating some parametrized curves in Chapter 4.

Precision of gradient estimator comparisons were done for certain networks selected using the following criteria.

- Very simple networks like 2 parallel arcs or serial arcs were selected for closed-form analysis. These structures were considered every time they appeared to shed some light on this study and when they are mathematically tractable.
- Networks that can be grown in a structured and easy way. Typical examples include pure series or pure parallel SANs
- Layered networks with forwards connections, with or without random cancellation of activities. These SANs were randomly selected/constructed while maintaining the underlying structure of layers. This was the solution for testing very complex and large networks generated semi-automatically.

Chapter 3

Theoretical Analysis

3.1 Gradient estimator analysis and network topologies

Test networks were chosen to address the problem of determining the best estimator and hypothesizing which effect it is reasonable to expect. The first network to analyze is the simplest one: 2 nodes (source and sink) and a single activity. This particular case is examined analytically in next sections. Next step is to consider a serial interconnection of 2 or more activities. It will be shown that this structure is just a simple extension of the single activity case. A more interesting case is the parallel connection of 2 arcs, because the longest path changes depending on the criticality index of each arc. Finally, a simple series-parallel network combination will be analytically examined. Conclusions will be drawn from these simple networks and we will hypothesize if the behavior can be extended to more complex network structures.

3.2 Single activity and series configuration

Consider the following single arc network in Figure 3.1. We are going to focus on the precision of the gradient estimator, specifically using the **variance** of gradient of longest path. Let Y be the RV which represents the total longest path time. In this very simple case the longest path is the unique path of random length X.

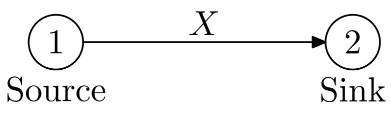


Figure 3.1: Single arc, two nodes stochastic activity network (SAN).

Arc exponentially distributed, sensitivity w.r.t. mean

In this specific network, the longest path is a random variable with time $X \sim \exp(\beta)$ where β is the expected value or mean of activity time X. The gradient to consider is computed with respect to the mean β. In other words, the parameter $\theta = \beta$ and we want to obtain an estimator of the "real" sensitivity $d\mathrm{E}[Y]/d\theta = d\mathrm{E}[X]/d\beta$.

Since Y (total completion time) is just X, we should anticipate that an infinitesimal perturbation $\delta\beta$ in the expected value of X, results in a $\delta\beta$ variation in the expected total completion time. In other words, the gradient w.r.t. β should be equal to 1.

IPA estimator

For this implementation, inverse transform method was used to generate exponential distributed samples. Consider $U \sim U(0,1)$ then,

$$\frac{dX}{d\beta} = \frac{X}{\beta} \tag{3.1}$$

From this expression, the variance of the estimator (a RV itself) can be calculated:

$$\begin{aligned}\operatorname{Var}\left(\frac{X}{\beta}\right) &= \frac{1}{\beta^2}\operatorname{Var}(X)\\ \operatorname{Var}\left(\frac{X}{\beta}\right) &= \frac{1}{\beta^2}\beta^2\\ \operatorname{Var}\left(\frac{X}{\beta}\right) &= 1\\ \operatorname{Var}\left(IPA - SINGLE_{\text{exp}}\right) &= 1 \qquad (3.2)\end{aligned}$$

Therefore, the exact variance of this estimator is fixed, i.e. independent of β, the mean of X. For this simple case, let us check that the expected gradient really equals to 1, i.e., it is unbiased.

$$\operatorname{E}\left[IPA - SINGLE_{\text{exp}}\right] = \operatorname{E}\left[\frac{X}{\beta}\right] = \frac{\operatorname{E}[X]}{\beta} = \frac{\beta}{\beta} = 1 \qquad (3.3)$$

SF/LR estimator

Differentiating the probability density function (PDF), the following can be obtained:

$$\begin{aligned}\frac{d}{d\beta}f_X(x) &= \frac{-1}{\beta^2}\exp\left(\frac{-x}{\beta}\right) + \frac{1}{\beta}\exp\left(\frac{-x}{\beta}\right)\cdot\frac{x}{\beta^2}\\ \frac{d}{d\beta}f_X(x) &= \frac{1}{\beta}\left(\frac{x}{\beta}-1\right)\frac{1}{\beta}\exp\left(\frac{-x}{\beta}\right), \qquad (3.4)\end{aligned}$$

for SF/LR method we need to get $\frac{d}{d\beta}\ln\left(f_X(x)\right) = \frac{\partial f_X(x)}{\partial\beta}/f_X(x)$, i.e., the score function, therefore,

$$\frac{d}{d\beta}\ln\left(f_X(x)\right) = \frac{1}{\beta}\left(\frac{x}{\beta}-1\right) \qquad (3.5)$$

Because of $Y = X$ in this network, the SF/LR gradient estimator for a single arc network is:

$$X \cdot \frac{1}{\beta}\left(\frac{X}{\beta} - 1\right), \tag{3.6}$$

and the variance of this gradient estimator is:

$$\begin{aligned} \mathrm{Var}\left(X \cdot \frac{1}{\beta}\left(\frac{X}{\beta} - 1\right)\right) &= \frac{1}{\beta^2}\mathrm{Var}\left(\frac{X^2}{\beta} - X\right) \\ = \frac{1}{\beta^2}&\left(\frac{1}{\beta^2}\mathrm{Var}\left(X^2\right) + \mathrm{Var}\left(X\right) - \frac{2}{\beta}\mathrm{Cov}\left(X^2, X\right)\right) \end{aligned} \tag{3.7}$$

The variance of X^2 is:

$$\mathrm{Var}\left(X^2\right) = \mathrm{E}\left[X^4\right] - \mathrm{E}\left[X^2\right]^2 = 24\beta^4 - \left(2\beta^2\right)^2 = 20\beta^4 \tag{3.8}$$

On the other hand, variance of X is β^2, and covariance of X^2 and X is:

$$\mathrm{Cov}\left(X^2, X\right) = \mathrm{E}\left[X^3\right] - \mathrm{E}\left[X^2\right] \cdot \mathrm{E}\left[X\right] = 6\beta^3 - 2\beta^2 \cdot \beta = 4\beta^3 \tag{3.9}$$

Plugging everything back to equation (3.7) :

$$\begin{aligned} \mathrm{Var}\left(X \cdot \frac{1}{\beta}\left(\frac{X}{\beta} - 1\right)\right) &= \frac{1}{\beta^2}\left(\frac{1}{\beta^2} \cdot 20\beta^4 + \beta^2 - \frac{2}{\beta} \cdot 4\beta^3\right) \\ \mathrm{Var}\left(SF/LR - SINGLE\right) &= (20 + 1 - 8) = 13 \end{aligned} \tag{3.10}$$

Again, the variance of this estimator in the single activity network is fixed, i.e. independent of the mean of X, but much larger than the variance of the IPA estimator.

WD estimator

Starting from equation (3.4), we need to rewrite the derivative of the density function as a difference of densities,

$$\frac{d}{d\beta} f_X(x) = \frac{1}{\beta}\left(\underbrace{\frac{x}{\beta^2}\exp\left(-\frac{x}{\beta}\right)}_{Erl(2,\beta)} - \underbrace{\frac{1}{\beta}\exp\left(-\frac{x}{\beta}\right)}_{\exp(\beta)}\right) \tag{3.11}$$

This way to write the derivate is not unique. Variance of the WD estimator is easy to write, since this is a extremely simple network,

$$\mathrm{Var}\left(\frac{1}{\beta}\left(X^{(2)} - X^{(1)}\right)\right) = \frac{1}{\beta^2}\left(\mathrm{Var}\left(X^{(2)}\right) + \mathrm{Var}\left(X^{(1)}\right) - 2\mathrm{Cov}\left(X^{(2)}, X^{(1)}\right)\right), \tag{3.12}$$

where $X^{(2)} \sim Erl(2,\beta)$ and $X^{(1)} = X \sim \exp(\beta)$. Variance for this kind of distributions are known:

$$\mathrm{Var}\left(X^{(2)}\right) = 2\beta^2,$$
$$\mathrm{Var}\left(X^{(1)}\right) = \mathrm{Var}\left(X\right) = \beta^2 \tag{3.13}$$

The covariance term depends on the way variates $X^{(2)}$ and X are generated. Since lower variance implies better precision of the estimator, $X^{(2)}$ and X are obtained from common random numbers (CRN) to get positive covariance. In this study, the Erlang variate $X^{(2)}$ is obtained by summing RV X and another independent exponential distributed variate. Let X^* be the other exponential distributed variable, independent of X, then

$$\begin{aligned}\operatorname{Cov}\left(X^{(2)}, X^{(1)}\right) &= \operatorname{Cov}\left(X + X^{*}, X\right) \\ = \mathrm{E}\left[\left(X + X^{*}\right) X\right] &- \underbrace{\mathrm{E}\left[X + X^{*}\right]}_{2\beta}\underbrace{\mathrm{E}\left[X\right]}_{\beta} \\ = \underbrace{\mathrm{E}\left[X^{2}\right]}_{2\beta^{2}} &+ \mathrm{E}\left[X^{*}X\right] - 2\beta^{2} \\ = 2\beta^{2} + \underbrace{\mathrm{E}\left[X^{*}\right]\mathrm{E}\left[X\right]}_{\beta\cdot\beta} &- 2\beta^{2} \\ \operatorname{Cov}\left(X^{(2)}, X^{(1)}\right) &= \beta^{2} \end{aligned} \qquad (3.14)$$

Putting these results together in equation (3.12),

$$\begin{aligned}\operatorname{Var}\left(\frac{1}{\beta}\left(X^{(2)} - X^{(1)}\right)\right) &= \frac{1}{\beta^{2}}\left(2\beta^{2} + \beta^{2} - 2\beta^{2}\right) \\ \operatorname{Var}\left(WD - SINGLE_{CRN}\right) &= 1 \end{aligned} \qquad (3.15)$$

Please note that if Common Randon Numbers (CRN) are NOT used, i.e. $X^{(2)}$ is independent of X then the variance of the WD estimator for this simple case is 3.

$$\operatorname{Var}\left(WD - SINGLE_{INDEP}\right) = 3 \qquad (3.16)$$

Also note that the these values are fixed and do not depend on the “scaling” of the RV X, i.e. it does not depends on β.

In summary, IPA and WD with CRN clearly outperform SF/LR estimator in terms of precision for this simple network (see Table 3.1).

Table 3.1: Comparison of gradient estimators for one arc exponentially distributed network

Exponential Distribution	Estimator variance Sens. w.r.t. β
IPA	1
SF/LR	13
WD w/CRN	1
WD indep	3

Activity times Gaussian distributed, sensitivity w.r.t. mean

Now assume that the parameter of interest is the mean of a Gaussian distribution.

IPA estimator

In this case IPA estimator is 1 (one), always (see [8], pag.16). Therefore the variance is always zero.

$$\mathrm{Var}\left(IPA - SINGLE\right) = 0 \tag{3.17}$$

SF/LR estimator

Since $\frac{\partial}{\partial\sigma}\ln f_X(x) = \left(\frac{x-\mu}{\sigma^2}\right)$ [8], we can compute the variance of the gradient estimator:

$$\begin{aligned}\mathrm{Var}\left(X\left(\frac{X-\mu}{\sigma^2}\right)\right) &= \frac{1}{\sigma^4}\left(\mathrm{Var}\left(X^2\right) + \mu^2\mathrm{Var}\left(X\right) - 2\mu\mathrm{Cov}\left(X^2, X\right)\right) \\ &= \frac{1}{\sigma^4}\left(2\sigma^4 + 4\mu^4\sigma^2 + \mu^2\sigma^2 - 4\mu^2\sigma^2\right) \\ \mathrm{Var}\left(SF/LR - SINGLE\right) &= 2 + \frac{\mu^2}{\sigma^2} \end{aligned} \tag{3.18}$$

WD estimator

WD estimator is given by $\frac{1}{\sqrt{2\pi}\sigma}\left(X^{(2)} - X^{(1)}\right)$, where $X^{(2)} \sim \mu + Wei\left(2, 1/2\sigma^2\right)$ and $X^{(1)} \sim \mu - Wei\left(2, 1/2\sigma^2\right)$. Variance of WD-based gradient estimator can be expressed as:

$$\mathrm{Var}\left(\frac{1}{\sqrt{2\pi}\sigma}\left(X^{(2)} - X^{(1)}\right)\right) = \frac{1}{2\pi\sigma^2}\left(\mathrm{Var}\left(X^{(2)}\right) + \mathrm{Var}\left(X^{(1)}\right) - 2\mathrm{Cov}\left(X^{(2)}, X^{(1)}\right)\right) \tag{3.19}$$

Assuming $X^{(2)}$ and $X^{(1)}$ are generated from the same Weibull distributed random variate X_{wei} using CRN, the covariance term is not zero. Let us compute the variance of the $X^{(2)}$ and $X^{(1)}$, which turns out to be the equal.

$$\begin{aligned}
\mathrm{Var}\left(X^{(2)}\right) &= \mathrm{Var}\left(\mu + X_{wei}\right) \\
\mathrm{Var}\left(X^{(2)}\right) = \mathrm{Var}\left(X_{wei}\right) &= \mathrm{Var}\left(X^{(1)}\right) \\
&= 2\left(1 - \frac{\pi}{4}\right)\sigma^2 \approx 0.4292\sigma^2,
\end{aligned} \tag{3.20}$$

where $X_{wei} \sim Wei\left(2, 1/2\sigma^2\right)$.

Also note that covariance for CRN is given by:

$$\begin{aligned}
\mathrm{Cov}\left(X^{(2)}, X^{(1)}\right) &= \mathrm{E}\left[\left(\mu + X_{wei}\right)\left(\mu - X_{wei}\right)\right] - \mathrm{E}\left[\mu + X_{wei}\right]\mathrm{E}\left[\mu - X_{wei}\right] \\
&= \mathrm{E}\left[\mu^2 - X_{wei}^2\right] - \left(\mu^2 - \mathrm{E}\left[X_{wei}\right]^2\right) \\
&= \mathrm{E}\left[X_{wei}\right]^2 - \mathrm{E}\left[X_{wei}^2\right] = -\mathrm{Var}\left(X_{wei}\right)
\end{aligned} \tag{3.21}$$

So, it is better to generate $X^{(2)}$ and $X^{(1)}$ independently. Plugging back results (3.20) and (3.21) in equation (3.19), we can obtain:

$$\begin{aligned}\mathrm{Var}\left(\frac{1}{\sqrt{2\pi}\sigma}\left(X^{(2)}-X^{(1)}\right)\right) &= \frac{1}{2\pi\sigma^2}\cdot 4\mathrm{Var}\left(X_{wei}\right)\\ \mathrm{Var}\left(WD-SINGLE_{CRN}\right) &= \frac{4}{\pi}-1\approx 0.2732 \qquad (3.22)\end{aligned}$$

Instead, if $X^{(1)}$ and $X^{(2)}$ are generated independently, then

$$\begin{aligned}\mathrm{Var}\left(\frac{1}{\sqrt{2\pi}\sigma}\left(X^{(2)}-X^{(1)}\right)\right) &= \frac{1}{2\pi\sigma^2}\cdot 2\mathrm{Var}\left(X_{wei}\right)\\ \mathrm{Var}\left(WD-SINGLE_{INDEP}\right) &= \frac{2}{\pi}-\frac{1}{2}\approx 0.1366 \qquad (3.23)\end{aligned}$$

In this case, if CRN is used the result worsen, because the covariance is negative. Again the variance is constant, i.e., no dependency of distributional parameters.

In conclusion, as summarized in Table 3.2, IPA estimator is the best estimator in terms of variance vs. the particular WD estimator considered here, and SF/LR is always greater than 2, hence the worse estimator:

Table 3.2: Comparison of gradient estimators for one arc normally distributed network

Gaussian Distribution	Estimator variance Sens. w.r.t. μ
IPA	0
SF/LR	$2+\mu^2/\sigma^2$
WD w/CRN	$4/\pi-1\approx 0.273$
WD Indep	$2/\pi-0.5\approx 0.137$

Activity times Gaussian distributed, sensitivity w.r.t. std. deviation

Now assume that the parameter of interest is the standard deviation of a Gaussian distribution.

IPA estimator

In this case, the IPA estimator is given by $\left(\frac{X-\mu}{\sigma}\right)$. Therefore, we can calculate the variance in closed form.

$$\operatorname{Var}\left(\frac{X-\mu}{\sigma}\right) = \frac{1}{\sigma^2}\operatorname{Var}(X) = \frac{\sigma^2}{\sigma^2} = 1$$

$$\operatorname{Var}(IPA - SINGLE) = 1 \tag{3.24}$$

SF/LR estimator

Since $\frac{\partial}{\partial\sigma}\ln f_X(x) = \frac{1}{\sigma}\left[\left(\frac{x-\mu}{\sigma}\right)^2 - 1\right]$, we can compute the variance of the gradient estimator:

$$\operatorname{Var}\left(\frac{X}{\sigma}\left[\left(\frac{X-\mu}{\sigma}\right)^2 - 1\right]\right) = \frac{1}{\sigma^6}\operatorname{Var}\left(X^3 - 2\mu X^2 + \left(\mu^2 - \sigma^2\right)X\right)$$

$$\operatorname{Var}(SL/LR - SINGLE) = 10 + 2\frac{\mu^2}{\sigma^2} \tag{3.25}$$

WD estimator

WD estimator is given by $\frac{1}{\sigma}\left(X^{(2)} - X^{(1)}\right)$, where $X^{(2)} \sim Mxw\left(\mu,\sigma^2\right)$ and as before $X = X^{(1)} \sim \mathcal{N}\left(\mu,\sigma^2\right)$. $Mxw\left(\mu,\sigma^2\right)$ represents a double-sided Maxwell distribution with PDF $f(x) = (x-\mu)^2/(\sqrt{2\pi}\sigma^3)\exp\left(-(x-\mu)^2/(2\sigma)\right)$.

The variance of WD-based estimator can be expressed as:

$$\begin{aligned}\mathrm{Var}\left(\frac{1}{\sigma}\left(X^{(2)}-X\right)\right) &= \frac{1}{\sigma^2}\left(\mathrm{Var}\left(X^{(2)}\right)+\mathrm{Var}\left(X\right)-2\mathrm{Cov}\left(X^{(2)},X\right)\right)\\ &= \frac{1}{\sigma^2}\left(3\sigma^2+\sigma^2-2\mathrm{Cov}\left(X^{(2)},X\right)\right) \qquad (3.26)\end{aligned}$$

If $X^{(2)}$ and X are generated independently, the covariance term is zero. CRN can be used to reduce the variance of the estimator (increasing the precision) by making the covariance term positive.

Following the implementation in [13], let $X_{StdMxw} \sim Mxw(0,1)$ (known as standard double-sided Maxwell). Samples from this distribution are obtained by A-R method. Then Mwx(μ,σ^2) and $\mathcal{N}(\mu,\sigma^2)$ distributed samples were generated using the following identities:

$$\begin{aligned}X_{Mxw} &= \sigma X_{StdMxw}+\mu\\ X_{Norm} &= \sigma X_{StdMxw}X_U+\mu\\ \Rightarrow X_{Norm} &= X_{Mxw}X_U-\mu X_U+\mu \qquad (3.27)\end{aligned}$$

where $X_U \sim \mathrm{Unif}(0,1)$ and is independent of X_{StdMxw}. Hence we have:

$$\begin{aligned}\mathrm{Cov}\left(X^{(2)},X^{(1)}\right) &= \mathrm{E}\left[X^{(2)}X^{(1)}\right]-\mathrm{E}\left[X^{(2)}\right]\mathrm{E}\left[X^{(1)}\right]\\ &= \mathrm{E}\left[X^{(2)}\left(X^{(2)}X_U-\mu X_U+\mu\right)\right]-\mu\cdot\mu\\ &= \mathrm{E}\left[\left(X^{(2)}\right)^2\right]\mathrm{E}\left[X_U\right]-\mu\mathrm{E}\left[X^{(2)}\right]\mathrm{E}\left[X_U\right]+\mu\mathrm{E}\left[X^{(2)}\right]-\mu^2\\ &= \left(3\sigma^2+\mu^2\right)\frac{1}{2}-\mu^2\frac{1}{2}+\mu^2-\mu^2=\frac{3\sigma^2}{2} \qquad (3.28)\end{aligned}$$

Plugging the covariance back to equation (3.26), next expression can be obtained:

$$\text{Var}\left(\frac{1}{\sigma}\left(X^{(2)} - X\right)\right) = \frac{1}{\sigma^2}\left(3\sigma^2 + \sigma^2 - 2\frac{3\sigma^2}{2}\right) = 1$$

$$\therefore \text{Var}\left(WD - SINGLE_{CRN}\right) = 1 \tag{3.29}$$

If no common random numbers are used, the covariance term is zero, and we get:

$$\text{Var}\left(\frac{1}{\sigma}\left(X^{(2)} - X\right)\right) = \frac{1}{\sigma^2}\left(3\sigma^2 + \sigma^2 - 2 \cdot 0\right) = 4$$

$$\therefore \text{Var}\left(WD - SINGLE_{INDEP}\right) = 4 \tag{3.30}$$

Table 3.3: Comparison of gradient estimators for one arc normally distributed network

Gaussian Distribution	Estimator variance Sens. wrt σ
IPA	1
SF/LR	$10 + 2\mu^2/\sigma^2$
WD w/CRN	1
WD Indep	4

According to Table 3.3, the comparative behavior of estimators are similar to previous instances, in where IPA and WD estimators outperform SF/LR, which in the best case is close to 10.

Pure series

In pure series configuration, we have a single path. Hence, the critical path is just the summation of arcs. Sensitivity will be calculated with respect to a change in the first activity. If more activities are perturbed, the net result is the sum of the individual perturbations.

IPA estimator

Since the IPA estimator is derived considering the parameter dependency in the samples, we need to differentiate the output RV $Y = \sum_i X_i$ with respect to a parameter in the first activity, then,

$$\frac{\partial}{\partial \theta} Y = \frac{\partial}{\partial \theta} X_1 + 0 + \ldots + 0 = \frac{\partial X_1}{\partial \theta} \tag{3.31}$$

In this method the distributions functions are untouched; thus IPA estimators for this particular instance are exactly the same as the single arc IPA estimators. This means that precision of the estimator is not degraded as the number of arcs in series increases.

SF/LR estimator

If the arc times are distributed as Gaussian, the analysis becomes easier since the sum of Gaussian RVs is also a Gaussian RV. Mathematically, if $X_i \sim \mathcal{N}(\mu_i, \sigma_i^2)$, then $Y = \sum_i X_i \sim \mathcal{N}(\sum_i \mu_i, \sum_i \sigma_i^2)$. Let $\mu = \sum_i \mu_i$ and $\sigma^2 = \sum_i \sigma_i^2$. Then we can derive an expression for the variance of the sensitivity with respect to the first mean μ_1.

$$\begin{aligned} \text{Var}\left(Y \cdot \frac{\partial \ln f_Y(y)}{\partial \mu_1}\right) &= \text{Var}\left(Y \cdot \frac{\partial \ln f_Y(y)}{\partial \mu} \cdot \underbrace{\frac{\partial \mu}{\partial \mu_1}}_{=1}\right) \\ &= 2 + \frac{\mu^2}{\sigma^2} = 2 + \frac{(\sum_{i=1}^{n} \mu_i)^2}{\sum_{i=1}^{n} \sigma_i^2} \end{aligned} \tag{3.32}$$

Equation (3.32) was obtained using equation (3.18) of previous section, considering the series connection as a single arc. Now, if we take IID arcs, then the variance of the gradient with respect to the first arc is:

$$\operatorname{Var}\left(\frac{\partial Y}{\partial \mu_1}\right)_{\text{SF/LRIID}} = 2 + \frac{(n\mu_1)^2}{n\sigma_1^2} = 2 + n\frac{\mu_1^2}{\sigma_1^2} \tag{3.33}$$

According to the last equation the variance of this estimator grows linearly as the number of activities n increases. However, this result is different to the system implemented, since the former is just a single-arc equivalent which does not represent the simulation estimator that degrades by adding variance from every arc. Although single-arc equivalent estimator deduced here has better precision and it is unbiased, in most networks and input distributions we are unable to get the equivalent distribution of the whole SAN. In fact, there is no use doing simulation if it is possible to get the distribution of the output in the first place.

On the other hand, if we consider the sensitivity with respect to the σ_1 we have.

$$\operatorname{Var}\left(Y \cdot \frac{\partial \ln f_Y(Y)}{\partial \sigma_1}\right) = \operatorname{Var}\left(Y \cdot \frac{\partial \ln f_Y(Y)}{\partial \sigma} \cdot \frac{\partial \sigma}{\partial \sigma_1}\right) \tag{3.34}$$

Since $\sigma^2 = \sum \sigma_i^2 \Rightarrow \sigma = \sqrt{\sum \sigma_i^2}$, we have,

$$\frac{\partial \sigma}{\partial \sigma_1} = \frac{1}{2\sqrt{\sigma^2}} \cdot 2\sigma_1 = \frac{\sigma_1}{\sigma} \tag{3.35}$$

Plugging back this expression in equation (3.34) and using equation (3.25) we have:

$$\operatorname{Var}\left(Y \cdot \frac{\partial \ln f_Y(Y)}{\partial \sigma} \cdot \frac{\partial \sigma}{\partial \sigma_1}\right) = \left(\frac{\sigma_1}{\sigma}\right)^2 \left(10 + 2\frac{\mu^2}{\sigma^2}\right) \tag{3.36}$$

If we choose IID random variables X_i then the following is obtained:

$$\operatorname{Var}\left(\frac{\partial Y}{\partial \sigma_1}\right)_{\text{SF/LRIID}} = \frac{\sigma_1^2}{n\sigma_1^2}\left(10 + 2\frac{n^2\mu_1^2}{n\sigma_1^2}\right) = \frac{10}{n} + 2\frac{\mu_1^2}{\sigma_1^2} \tag{3.37}$$

This result tells us that the variance decreases as the number of arcs increases, which seems to contradict simulation results obtained in the next chapter. The same reasoning from previous paragraph explains this behavior: The SF/LR estimator is different from the one obtained taking each arc individually as is done in the simulation.

Hence, let us try to find a closed-form expression for the sensitivity w.r.t. the mean in the pure series Gaussian distributed network without taking the equivalent distribution of the sum of the arcs.

When we just have one arc, the variance of the SF/LR was already calculated.

$$Y = X_1 \Rightarrow \left(\frac{\partial \hat{Y}}{\partial \mu_1}\right)_{SF/LR} = 2 + \frac{\mu^2}{\sigma^2} \tag{3.38}$$

Now, consider a network of 2 arcs in series configuration:

$$\begin{aligned} Y = X_1 + X_2 \Rightarrow \left(\frac{\partial \hat{Y}}{\partial \mu_1}\right)_{SF/LR} &= \mathrm{Var}\left((X_1 + X_2)\frac{X_1 - \mu_1}{\sigma_1^2}\right) \\ &= \mathrm{Var}(\underbrace{X_1\frac{(X_1-\mu_1)}{\sigma_1^2}}_{Z_1} + \underbrace{X_2\frac{(X_1-\mu_1)}{\sigma_1^2}}_{Z_2}) \\ &= \mathrm{Var}\left(Z_1\right) + \mathrm{Var}\left(Z_2\right) + 2\mathrm{Cov}\left(Z_1, Z_2\right) \end{aligned} \tag{3.39}$$

Notice that $\mathrm{Var}\left(Z_1\right) = 2 + \mu^2/\sigma^2$ (single arc case). For variance of Z_2 we can use the formula for the variance of a product of independent RVs.[1]

[1]The variance of multiplication of 2 independent RV can be deduced this way:

$\mathrm{Var}\left(UV\right) = \mathrm{E}\left[U^2V^2\right] - \mathrm{E}\left[UV\right]^2 \underset{indep}{=} \mathrm{E}\left[U^2\right]\mathrm{E}\left[V^2\right] - \mathrm{E}\left[U\right]^2\mathrm{E}\left[V\right]^2$

$= \mathrm{E}\left[U^2\right](\mathrm{E}\left[V^2\right] - \mathrm{E}\left[V\right]^2) + \mathrm{E}\left[V\right]^2(\mathrm{E}\left[U^2\right] - \mathrm{E}\left[U\right]^2)$

$= (\mathrm{Var}\left(U\right) + \mathrm{E}\left[U\right]^2)\mathrm{Var}\left(V\right) + \mathrm{E}\left[V\right]^2\mathrm{Var}\left(U\right)$

$= \mathrm{Var}\left(U\right)\mathrm{Var}\left(V\right) + \mathrm{E}\left[U\right]^2\mathrm{Var}\left(V\right) + \mathrm{E}\left[V\right]^2\mathrm{Var}\left(U\right)$

$\Rightarrow \mathrm{Var}\left(UV\right) = \mathrm{Var}\left(U\right)\mathrm{Var}\left(V\right) + \mathrm{E}\left[U\right]^2\mathrm{Var}\left(V\right) + \mathrm{E}\left[V\right]^2\mathrm{Var}\left(U\right)$

$$\operatorname{Var}(Z_2) = \operatorname{Var}\left(X_2 \frac{X_1-\mu_1}{\sigma_1^2}\right) = \frac{1}{\sigma_1^2}\operatorname{Var}(X_2 \underbrace{\frac{X_1-\mu_1}{\sigma_1}}_{0-mean, var=1})$$
$$= \frac{1}{\sigma_1^2}(\operatorname{Var}(X_2) + \operatorname{E}[X_2]^2 + 0) = \frac{\sigma_2^2+\mu_2^2}{\sigma_1^2} \tag{3.40}$$

Let us concentrate on the covariance between Z_1 and Z_2.

$$\operatorname{Cov}(Z_1, Z_2) = \operatorname{E}\left[X_1X_2\left(\frac{X_1-\mu_1}{\sigma_1^2}\right)^2\right] - \operatorname{E}\left[X_1\frac{(X_1-\mu_1)}{\sigma_1^2}\right]\underbrace{\operatorname{E}\left[X_2\frac{(X_1-\mu_1)}{\sigma_1^2}\right]}_{=0}$$
$$= \mu_2\operatorname{E}\left[X_1\left(\frac{X_1-\mu_1}{\sigma_1^2}\right)^2\right]$$
$$= \frac{\mu_2}{\sigma_1^4}\left(\operatorname{E}\left[X_1^3\right] - 2\mu_1\operatorname{E}\left[X_1^2\right] + \mu_1^2\operatorname{E}[X_1]\right)$$
$$= \frac{\mu_2}{\sigma_1^4}\left(\mu_1^3 + 3\mu_1\sigma_1^2 - 2\mu_1(\mu_1^2+\sigma_1^2) + \mu_1^2\mu_1\right)$$
$$= \frac{\mu_2}{\sigma_1^4}\left(\mu_1\sigma_1^2\right) = \frac{\mu_1\mu_2}{\sigma_1^2} \tag{3.41}$$

Bringing back previous results together, we can write:

$$\left(\frac{\partial\hat{Y}}{\partial\mu_1}\right)_{SF/LR} = 2 + \frac{\mu_1^2}{\sigma_1^2} + \frac{\mu_2^2+\sigma_2^2}{\sigma_1^2} + 2\frac{\mu_1\mu_2}{\sigma_1^2} \tag{3.42}$$

Now, let $Y = X_1 + X_2 + X_3$:

$$\left(\frac{\partial\hat{Y}}{\partial\mu_1}\right)_{SF/LR} = \operatorname{Var}\left(X_1\frac{X_1-\mu_1}{\sigma_1^2} + X_2\frac{X_1-\mu_1}{\sigma_1^2} + X_3\frac{X_1-\mu_1}{\sigma_1^2}\right)$$
$$= \operatorname{Var}(Z_1+Z_2+Z_3)$$
$$= \operatorname{Var}(Z_1) + \operatorname{Var}(Z_2) + \operatorname{Var}(Z_3) + 2\operatorname{Cov}(Z_1,Z_2) + 2\operatorname{Cov}(Z_1,Z_3) + 2\operatorname{Cov}(Z_2,Z_3) \tag{3.43}$$

Similarly, it can be concluded that:

$$\begin{aligned} \operatorname{Var}(Z_3) &= \frac{\sigma_3^2 + \mu_3^2}{\sigma_1^2} \\ \operatorname{Cov}(Z_1, Z_3) &= \frac{\mu_1 \mu_3}{\sigma_1^2} \end{aligned} \tag{3.44}$$

It is still needed to compute the covariance between Z_2 and Z_3:

$$\begin{aligned} \operatorname{Cov}(Z_2, Z_3) &= \mathrm{E}\left[X_2 X_3 \left(\frac{X_1 - \mu_1}{\sigma_1^2}\right)^2\right] - 0 \\ &= \frac{\mu_2 \mu_3}{\sigma_1^2} \mathrm{E}\left[\left(\frac{X_1 - \mu_1}{\sigma_1}\right)^2\right] \\ &= \frac{\mu_2 \mu_3}{\sigma_1^2}(1 + 0) = \frac{\mu_2 \mu_3}{\sigma_1^2} \end{aligned} \tag{3.45}$$

If we assume activity times IID, i.e. $X_i \sim \mathcal{N}(\mu, \sigma^2), i = 1, ..., n$, then we can express a condensed expression for the variance of this estimator:

$$\left(\frac{\partial \hat{Y}}{\partial \mu_1}\right)_{SF/LR-IID} = \left(2 + \frac{\mu^2}{\sigma^2}\right) + 2\frac{\sigma^2 + \mu^2}{\sigma^2} + 3 \cdot 2\frac{\mu^2}{\sigma^2} \tag{3.46}$$

Now we are ready to present a general formula for $Y = X_1 + ... + X_n$:

$$\left(\frac{\partial \hat{Y}}{\partial \mu_1}\right)_{SF/LR-IID} = \left(2 + \frac{\mu^2}{\sigma^2}\right) + (n-1)\frac{\sigma^2 + \mu^2}{\sigma^2} + \binom{n}{2} \cdot 2\frac{\mu^2}{\sigma^2} \tag{3.47}$$

A plot of this last equation is presented in Figure 3.2, taking $\mu = 30$ and $\sigma = 5$.

We limited our analysis to normally distributed activity times, since even simple networks like this pure series becomes intractable when other distributions are considered. For example, if we use exponential distributions to generate the activity times, the distribution of the longest (an unique) path is called Hypo-Exponential. In other words, if $X_i \sim \exp(\beta_i)$ then $\sum_i^n X_i \sim \operatorname{HypoExp}(\beta_1, \ldots, \beta_n)$.

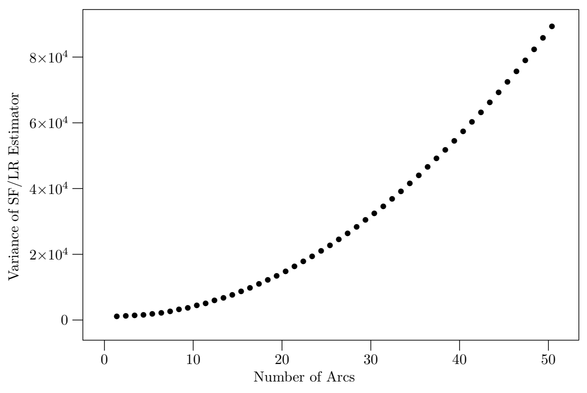


Figure 3.2: Variance of the SF/LR based estimator for a pure series stochastic activity network. Activity times normally distributed and IID with mean 30 and standard deviation 5, i.e. $X_i \sim \mathcal{N}\left(\mu = 30, \sigma = 5\right)$.

It turns out that it is not trivial to deduct the SF/LR gradient estimator for this distribution mostly due its matrix representation.

Moreover, the previous example illustrate the problem of SF/LR under increasing number of activities: the positive covariance terms plus the individual variance terms sum up, degrading the estimation precision.

WD estimator

Remember that in the weak derivative gradient estimator, the parameter θ is in the distribution function and the derivative is rewritten as a difference.

$$\begin{aligned}
\frac{\partial}{\partial\theta}\mathrm{E}\left[Y\right] &= \frac{1}{c}\left(\mathrm{E}\left[X_1^{(2)} + X_2 + \ldots + X_n\right] - \mathrm{E}\left[X_1^{(1)} + X_2 + \ldots + X_n\right]\right) \\
&= \frac{1}{c}\left(\mathrm{E}\left[X_1^{(2)}\right] - \mathrm{E}\left[X_1^{(1)}\right]\right) \\
&= \frac{\partial}{\partial\theta}\mathrm{E}\left[X_1\right] \qquad (3.48)
\end{aligned}$$

Hence, the WD estimator is the same as the single arc WD estimator. This implies that there is no degradation in the precision as the number of activities n increases.

3.3 Two parallel activities

Let's now consider the case of 2 parallel activities. An analysis of the variance of the gradient estimator will be presented.

IPA estimator

The critical (longest) path in this case is given by $Y = \max(X_1, X_2)$, where $X_1 \sim \exp(\beta_1)$ and $X_2 \sim \exp(\beta_2)$. Remember that the IPA estimator is dependent on the sampling strategy, in this case the 2 independent activity times are generated using the CDF inverse transform method, i.e., $Y = \max(-\beta_1 \ln U_1, -\beta_2 \ln U_2)$, where $U_1, U_2 \sim \text{Unif}(0,1)$. Hence, the derivative with respect to first mean β_1 can be written as:

$$Y' = \frac{\partial}{\partial \beta_1} Y = \begin{cases} -\ln U_1 & \text{if } -\beta_1 \ln U_1 > -\beta_2 \ln U_2 \\ 0 & \text{otherwise} \end{cases} \tag{3.49}$$

Please note that the condition $-\beta_1 \ln U_1 > -\beta_2 \ln U_2$ is equivalent to $U_1 < U_2^{\beta_2/\beta_1}$. Hence, the expected value and the expected value of the square can be calculated using double integrals.

$$\begin{aligned}
\mathrm{E}\left[\frac{\partial Y}{\partial \beta_1}\right] &= \int_{u_2=0}^{u_2=1}\int_{u_1=0}^{u_1=u_2^{\beta_2/\beta_1}}(-\ln u_1)du_1du_2 \\
&= \int_0^1 u_2^{\beta_2/\beta_1}du_2 - \int_0^1 u_2^{\beta_2/\beta_1}\ln\left(u_2^{\beta_2/\beta_1}\right)du_2 \\
&= \frac{1}{\beta_2/\beta_1+1}+\frac{\beta_2/\beta_1}{(\beta_2/\beta_1+1)^2} \\
&= \frac{1+2\beta_2/\beta_1}{(\beta_2/\beta_1+1)^2} \qquad (3.50)
\end{aligned}$$

Since we expect unbiased estimators, every gradient estimator method in this section should have the same expected value of Equation (3.50). This equation is also plotted in Figure 3.3.

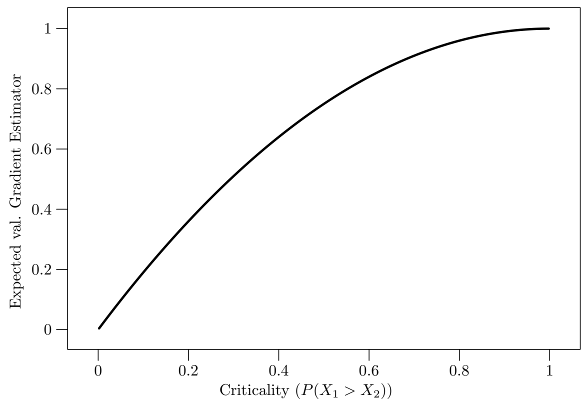


Figure 3.3: Relation between the criticality index and expected value of the gradient for a SAN of two parallel exponentially distributed Arcs. Critical path (maximum path) gradient sensitivity with respect to the mean.

$$\mathrm{E}\left[\left(\frac{\partial Y}{\partial \beta_1}\right)^2\right] = \int_{u_2=0}^{u_2=1}\int_{u_1=0}^{u_1=u_2^{\beta_2/\beta_1}} \ln^2 u_1 du_1 du_2$$

$$= \int_0^1 2u_2^{\beta_2/\beta_1} du_2 - \int_0^1 2u_2^{\beta_2/\beta_1} \ln\left(u_2^{\beta_2/\beta_1}\right) du_2 + \int_0^1 u_2^{\beta_2/\beta_1} \ln^2\left(u_2^{\beta_2/\beta_1}\right) du_2$$

$$= 2\left(\frac{1}{\beta_2/\beta_1+1} + \frac{\beta_2/\beta_1}{(\beta_2/\beta_1+1)^2} + \frac{(\beta_2/\beta_1)^2}{(\beta_2/\beta_1+1)^3}\right) \quad (3.51)$$

Now we can obtain the variance of this estimator using the equation $\mathrm{Var}(Z) = \mathrm{E}[Z^2] - \mathrm{E}[Z]^2$:

$$\mathrm{Var}\left(\frac{\partial Y}{\partial \beta_1}\right)_{\mathrm{IPA}} = \frac{6(\beta_2/\beta_1)^3 + 8(\beta_2/\beta_1)^2 + 4(\beta_2/\beta_1) + 1}{(\beta_2/\beta_1+1)^4} \quad (3.52)$$

This equation was plotted using logarithmic scale in the x-axis in Figure 3.4. Observe in this figure that the variance does not depends on the scaling of the network, i.e., if β_1 and β_2 are increased or decreased keeping constant the ratio between them, then the variance does not change.

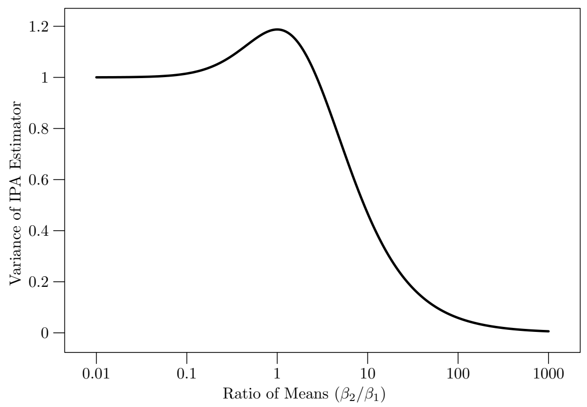


Figure 3.4: Relation between the means ratio and variance of the IPA estimator for a SAN of two parallel exponentially distributed Arcs. Critical path (maximum path) sensitivity with respect to the mean.

Plotting this same variance w.r.t. the criticality result in the curve showed in Figure 3.5 with the maximum occurring at a criticality index value of 50%.

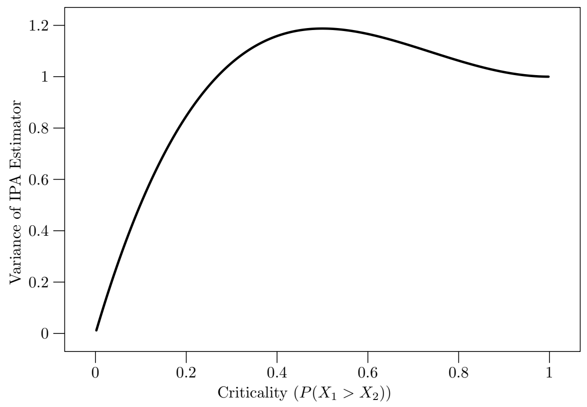


Figure 3.5: Variance of the IPA estimator for a SAN of two parallel exponentially distributed arcs with respect to the criticality index. Critical path (maximum path) sensitivity with respect to the mean.

SF/LR estimator

Since the output random variable $Y = \max(X_1, X_2)$ is simple enough, we can get a closed-form expression for the variance of this gradient estimator. $\mathrm{Var}(Z) = \mathrm{E}[Z^2] - \mathrm{E}[Z]^2$, where Z, in this case, is the SF/LR gradient estimator, given by,

$$Z = \max(X_1, X_2) \cdot \frac{1}{\beta_1}\left(\frac{X_1}{\beta_1} - 1\right) \tag{3.53}$$

Let's begin finding an expression for the expected value.

$$\mathrm{E}\left[\max\left(X_1,X_2\right)\cdot\frac{1}{\beta_1}\left(\frac{X_1}{\beta_1}-1\right)\right]=\int_{x_1=0}^{\infty}\int_{x_2=0}^{x_1}\frac{x_1}{\beta_1}\left(\frac{x_1}{\beta_1}-1\right)f_{X_1}f_{X_2}dx_2dx_1$$
$$+\int_{x_1=0}^{\infty}\int_{x_2=x_1}^{\infty}\frac{x_2}{\beta_1}\left(\frac{x_1}{\beta_1}-1\right)f_{X_1}f_{X_2}dx_2dx_1$$
$$=\frac{\beta_1}{\beta_1+\beta_2^2}\left(\beta_1+2\beta_2\right)$$
$$=\frac{1+2\beta_2/\beta_1}{\left(1+\beta_2/\beta_1\right)^2} \tag{3.54}$$

Same expected value as IPA estimator which is correct if we assume both estimators are unbiased. Now the expected value of the square.

$$\mathrm{E}\left[\left(\max\left(X_1,X_2\right)\cdot\frac{1}{\beta_1}\left(\frac{X_1}{\beta_1}-1\right)\right)^2\right]$$
$$=\int_{x_1=0}^{\infty}\int_{x_2=0}^{x_1}\left(x_1\frac{1}{\beta_1}\left(\frac{x_1}{\beta_1}-1\right)\right)^2 f_{X_1}f_{X_2}dx_2dx_1$$
$$+\int_{x_1=0}^{\infty}\int_{x_2=x_1}^{\infty}\left(x_2\frac{1}{\beta_1}\left(\frac{x_1}{\beta_1}-1\right)\right)^2 f_{X_1}f_{X_2}dx_2dx_1$$
$$=\frac{2\left(7\beta_1^6+28\beta_1^5\beta_2+42\beta_1^4\beta_2^2+30\beta_1^3\beta_2^3+6\beta_1^2\beta_2^4+4\beta_1\beta_2^5+\beta_2^6\right)}{\beta_1^2(\beta_1+\beta_2)^4} \tag{3.55}$$

The closed-form equation for the variance is:

$$\mathrm{Var}\left(\max\left(X_1,X_2\right)\cdot\frac{1}{\beta_1}\left(\frac{X_1}{\beta_1}-1\right)\right)$$
$$=\frac{13\beta_1^6+52\beta_1^5\beta_2+80\beta_1^4\beta_2^2+60\beta_1^3\beta_2^3+12\beta_1^2\beta_2^4+8\beta_1\beta_2^5+2\beta_2^6}{\beta_1^2(\beta_1+\beta_2)^4}$$
$$\frac{13+52\left(\beta_2/\beta_1\right)+80\left(\beta_2/\beta_1\right)^2+60\left(\beta_2/\beta_1\right)^3+12\left(\beta_2/\beta_1\right)^4+8\left(\beta_2/\beta_1\right)^5+2\left(\beta_2/\beta_1\right)^6}{\left(1+\beta_2/\beta_1\right)^4} \tag{3.56}$$

Notice in Figure 3.6 that curve has an asymptote at variance=13 (the limiting case of single arc), and decreasing the criticality of the first arc just worsens the variance of the estimator very quickly. This behavior differs w.r.t the behavior of the IPA estimator.

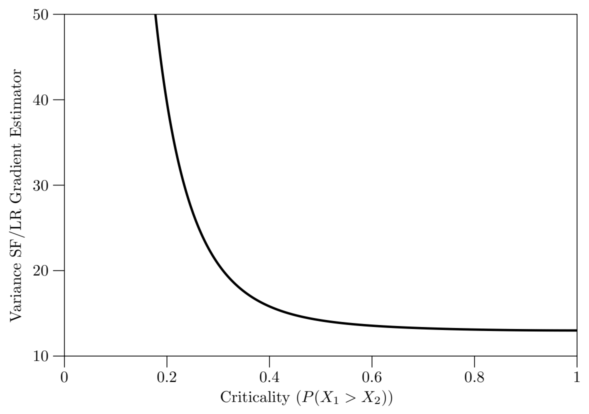


Figure 3.6: Relation between the ratio of means and variance of the SF/LR estimator for a SAN of two parallel exponentially distributed arcs. Critical path (maximum path) sensitivity with respect to the mean of the first arc.

WD estimator

The variance of the weak derivative based gradient estimator from [8], in this case is:

$$\operatorname{Var}\left(\frac{\partial Y}{\partial \beta_1}\right)_{\mathrm{WD}} = \frac{1}{\beta_1^2}\left(\operatorname{Var}\left(Y^{(2)}\right) + \operatorname{Var}\left(Y^{(1)}\right) - 2\operatorname{Cov}\left(Y^{(2)}, Y^{(1)}\right)\right) \tag{3.57}$$

where $Y^{(2)}$ and $Y^{(1)}$ are given by:

$$Y^{(2)} = \max\left(X^{(2)}, X_2)\right) \tag{3.58}$$

$$Y^{(1)} = \max\left(X^{(1)}, X_2)\right) \tag{3.59}$$

The variance of the WD estimator depends on the 'coupling' used to generate the phantom samples. In this case, it is assumed that the random Erlang variates $X^{(2)} \sim \operatorname{Erl}(2, \beta_1)$ were generated summing two independent exponential variates: X_1 and X_3, where X_1, $X_3 \sim \exp(\beta_1)$. On the other hand, X_2 is still an exponentially distributed variable, i.e. $X_2 \sim \exp(\beta_2)$. Finally $X^{(1)}$ is just X_1. Using these assumptions, we have that:

$$Y^{(2)} = \max\left(X^{(2)}, X_2\right) = \max\left(X_1 + X_3, X_2\right) \tag{3.60}$$

$$Y^{(1)} = \max\left(X^{(1)}, X_2\right) = \max\left(X_1, X_2\right) \tag{3.61}$$

Note that we can know the variance of $Y^{(1)}$ by definition using the joint distribution. Because we assumed independent RVs, we can write the joint distribution as the product of the marginal ones.

$$\mathrm{E}\left[Y^{(1)}\right] = \int_{x_2=0}^{\infty}\int_{x_1=0}^{x_2} x_2 f_{X_1} f_{X_2} dx_1 dx_2 + \int_{x_2=0}^{\infty}\int_{x_1=x_2}^{\infty} x_1 f_{X_1} f_{X_2} dx_1 dx_2$$

$$\mathrm{E}\left[Y^{(1)}\right] = \beta_2 + \frac{\beta_1^2}{\beta_1+\beta_2} \tag{3.62}$$

$$\mathrm{E}\left[\left(Y^{(1)}\right)^2\right] = \int_{x_2=0}^{\infty}\int_{x_1=0}^{x_2} x_2^2 f_{X_1} f_{X_2} dx_1 dx_2 + \int_{x_2=0}^{\infty}\int_{x_1=x_2}^{\infty} x_1^2 f_{X_1} f_{X_2} dx_1 dx_2$$

$$\mathrm{E}\left[\left(Y^{(1)}\right)^2\right] = 2\left(\beta_2^2 + \beta_1^3 \frac{\beta_1 + 2\beta_2}{\left(\beta_1+\beta_2\right)^2}\right)$$

$$\mathrm{Var}\left(Y^{(1)}\right) = \frac{\left(\beta_1^2 - \beta_1\beta_2 + \beta_2^2\right)\left(\beta_1^2 + 3\beta_1\beta_2 + \beta_2^2\right)}{\left(\beta_1+\beta_2\right)^2} \tag{3.63}$$

Now, consider the variance of $Y^{(2)}$. Notice that the expectation integral can be decomposed in three parts depending on the relative magnitudes of X_1, X_2 and X_3.

$$\mathrm{E}\left[Y^{(2)}\right] = \int_{x_2=0}^{\infty}\int_{x_1=0}^{x_2}\int_{x_3=0}^{x_2-x_1} x_2 f_{X_1} f_{X_2} f_{X_3} dx_3 dx_2 dx_1$$

$$+\int_{x_2=0}^{\infty}\int_{x_1=0}^{x_2}\int_{x_3=x_2-x_1}^{\infty} \left(x_1+x_3\right) f_{X_1} f_{X_2} f_{X_3} dx_3 dx_2 dx_1$$

$$+\int_{x_2=0}^{\infty}\int_{x_1=x_2}^{\infty}\int_{x_3=0}^{\infty} \left(x_1+x_2\right) f_{X_1} f_{X_2} f_{X_3} dx_3 dx_2 dx_1$$

$$\mathrm{E}\left[Y^{(2)}\right] = \beta_2 + \frac{\beta_1}{\left(\beta_1+\beta_2\right)^2}\left(2\beta_1+3\beta_2\right) \tag{3.64}$$

$$\mathrm{E}\left[(Y^{(2)})^2\right] = \int_{x_2=0}^{\infty}\int_{x_1=0}^{x_2}\int_{x_3=0}^{x_2-x_1} x_2^2 f_{X_1} f_{X_2} f_{X_3} dx_3 dx_2 dx_1$$

$$+\int_{x_2=0}^{\infty}\int_{x_1=0}^{x_2}\int_{x_3=x_2-x_1}^{\infty} \left(x_1+x_3\right)^2 f_{X_1} f_{X_2} f_{X_3} dx_3 dx_2 dx_1$$

$$+\int_{x_2=0}^{\infty}\int_{x_1=x_2}^{\infty}\int_{x_3=0}^{\infty} \left(x_1+x_2\right)^2 f_{X_1} f_{X_2} f_{X_3} dx_3 dx_2 dx_1$$

$$\mathrm{E}\left[(Y^{(2)})^2\right] = 2\left(\beta_2^2 + \beta_1^3\frac{3\beta_1^2+9\beta_1\beta_2+8\beta_2^2}{\left(\beta_1+\beta_2\right)^3}\right) \tag{3.65}$$

$$\tag{3.66}$$

With these results we can give an expression for the variance of $Y^{(2)}$.

$$\mathrm{Var}\left(Y^{(2)}\right) = \mathrm{E}\left[\left(Y^{(2)}\right)^2\right] - \mathrm{E}\left[Y^{(2)}\right]^2$$

$$= \frac{2\beta_1^6+8\beta_1^5\beta_2+12\beta_1^4\beta_2^2+4\beta_1^3\beta_2^3+4\beta_1\beta_2^5+\beta_2^6}{\left(\beta_1+\beta_2\right)^4} \tag{3.67}$$

Now we need to give an expression for the covariance between $Y^{(2)}$ and $Y^{(1)}$.

Let us follow the definition of covariance:

$$\mathrm{Cov}\left(Y^{(2)}, Y^{(1)}\right) = \mathrm{E}\left[Y^{(2)}Y^{(1)}\right] - \mathrm{E}\left[Y^{(2)}\right]\mathrm{E}\left[Y^{(1)}\right]$$

$$= \mathrm{E}\left[\max\left(X_1+X_3, X_2\right)\cdot\max\left(X_1, X_2\right)\right]$$

$$-\mathrm{E}\left[\max\left(X_1+X_3, X_2\right)\right]\mathrm{E}\left[\max\left(X_1, X_2\right)\right] \tag{3.68}$$

First, the expected vale of the product can be calculated using the same integration technique presented above.

$$\begin{aligned}
\mathrm{E}\left[Y^{(2)}Y^{(1)}\right] &= \int_{x_2=0}^{\infty}\int_{x_1=0}^{x_2}\int_{x_3=0}^{x_2-x_1} x_2 \cdot x_2 f_{X_1} f_{X_2} f_{X_3} dx_3 dx_1 dx_2 \\
&+ \int_{x_2=0}^{\infty}\int_{x_1=0}^{x_2}\int_{x_3=x_2-x_1}^{\infty} (x_1+x_3) \cdot x_2 f_{X_1} f_{X_2} f_{X_3} dx_3 dx_1 dx_2 \\
&+ \int_{x_2=0}^{\infty}\int_{x_1=x_2}^{\infty}\int_{x_3=0}^{\infty} (x_1+x_3) \cdot x_1 f_{X_1} f_{X_2} f_{X_3} dx_3 dx_1 dx_2 \\
\mathrm{E}\left[Y^{(2)}Y^{(1)}\right] &= 2\beta_2^2 + \beta_1^3 \frac{3\beta_1^2 + 9\beta_1\beta_2 + 8\beta_2^2}{(\beta_1+\beta_2)^3} \qquad (3.69)
\end{aligned}$$

Hence, the covariance between $Y^{(2)}$ and $Y^{(1)}$ is:

$$\mathrm{Cov}\left(Y^{(2)}, Y^{(1)}\right) = \frac{\beta_1^5 + 3\beta_1^4\beta_2 + 2\beta_1^3\beta_2^2 - \beta_1^2\beta_2^3 + 3\beta_1\beta_2^4 + \beta_2^5}{(\beta_1+\beta_2)^3} \qquad (3.70)$$

Now the variance of the WD estimator can be written as:

$$\begin{aligned}
\mathrm{Var}\left(\frac{\partial Y}{\partial \beta_1}\right)_{\mathrm{WD}} &= \frac{1}{\beta_1^2}\left(\mathrm{Var}\left(Y^{(2)}\right) + \mathrm{Var}\left(Y^{(1)}\right) - 2\mathrm{Cov}\left(Y^{(2)}, Y^{(1)}\right)\right) \\
\mathrm{Var}\left(\frac{\partial Y}{\partial \beta_1}\right)_{\mathrm{WD}} &= (1 + 2\beta_2/\beta_1) \frac{1 + 2\beta_2/\beta_1 + 2(\beta_2/\beta_1)^2}{(1+\beta_2/\beta_1)^4} \qquad (3.71)
\end{aligned}$$

Figure 3.7 shows a logarithmic graph in the horizontal axis, where we assigned the ratio of the means (β_2/β_1).

Based on the variance of these three gradient estimators, it can be concluded that WD estimator outperforms IPA and SF/LR estimator for every criticality level in this particular network setup (see Figure 3.8).

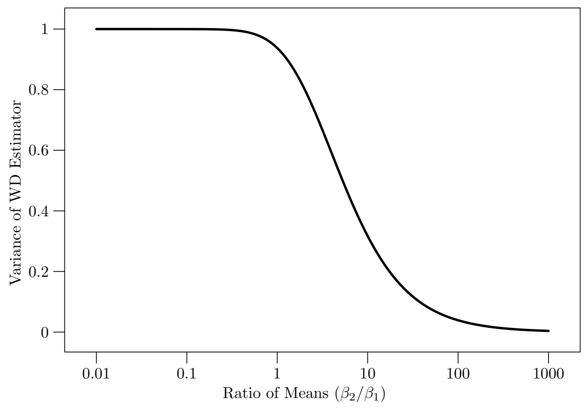


Figure 3.7: Relation between the means ratio and variance of the WD-based estimator for a SAN of two parallel exponentially distributed arcs. Critical path sensitivity with respect to mean of the first activity.

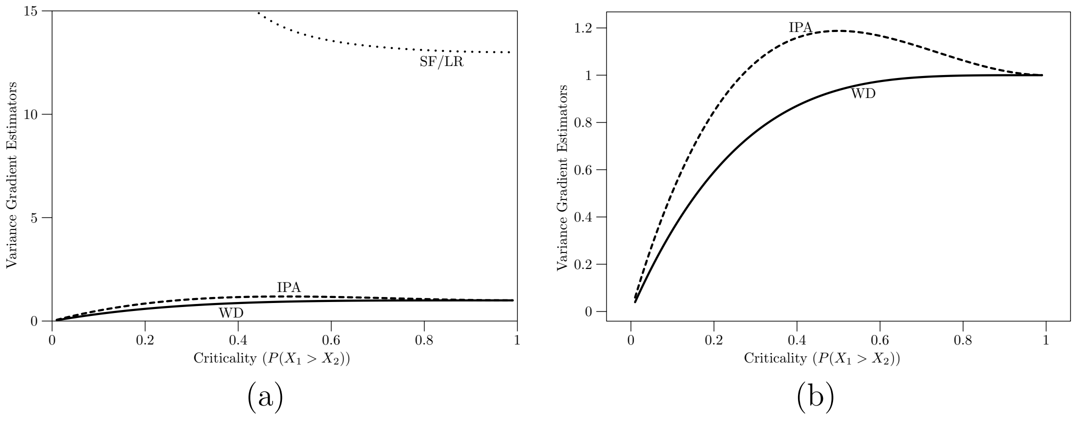


Figure 3.8: Comparison of estimator precision among IPA, SF/LR and WD methods (less is better). Two parallel activities, exponentially distributed. Critical (longest) path gradient with respect to mean of the first activity β_1. (a) IPA and WD are much more precise than SF/LR (b) Detail between IPA and WD estimators, WD outperform IPA at every criticality level

3.4 Two parallel activities, shortest path

Let's now consider the case of the **shortest path** in a network of two parallel activities. Analysis of the variance of the gradient estimator will be presented in next sub-sections. We are departing slightly from our original focus on the critical path. However, the idea is to drawn useful conclusions from this special case that are applicable to the critical path problem.

IPA estimator

The shortest path in this case is given by $Y = \min(X_1, X_2)$, where $X_1 \sim \exp(\beta_1)$ and $X_2 \sim \exp(\beta_2)$. Remember that the IPA estimator is dependent on the random variate generation strategy. Just like in the previous section, two independent activity times are generated using the CDF inverse method, i.e.,

$$Y = \min(-\beta_1 \ln U_1, -\beta_2 \ln U_2) \tag{3.72}$$

where U_1, U_2 are $U(0,1)$ IID RVs, and β_1, β_2 are the means of exponentially distributed RVs respectively. Hence, the derivative with respect to first mean β_1 can be written as:

$$Y' = \frac{\partial}{\partial \beta_1} Y = \begin{cases} -\ln U_1 & \text{if } -\beta_1 \ln U_1 < -\beta_2 \ln U_2 \\ 0 & \text{otherwise} \end{cases} \tag{3.73}$$

The condition $-\beta_1 \ln U_1 < -\beta_2 \ln U_2$ is equivalent to $U_1 > U_2^{\beta_2/\beta_1}$. Hence, expected value and variance can be calculated using the following double integrals.

$$\begin{aligned} \mathrm{E}\left[\frac{\partial Y}{\partial \beta_1}\right] &= \int_{u_2=0}^{1} \int_{u_1=u_2^{\beta_2/\beta_1}}^{1} (-\ln u_1) du_1 du_2 \\ &= \left(\frac{\beta_2/\beta_1}{1+\beta_2/\beta_1}\right)^2 \end{aligned} \tag{3.74}$$

$$\begin{aligned} \mathrm{E}\left[\left(\frac{\partial Y}{\partial \beta_1}\right)^2\right] &= \int_{u_2=0}^{1} \int_{u_1=0}^{u_2^{\beta_2/\beta_1}} (-\ln u_1)^2 \, du_1 du_2 \\ &= 2\left(\frac{\beta_2/\beta_1}{1+\beta_2/\beta_1}\right)^3 \end{aligned} \tag{3.75}$$

$$\therefore \mathrm{Var}\left(\frac{\partial Y}{\partial \beta_1}\right)_{\mathrm{IPA}} = \mathrm{E}\left[\left(\frac{\partial Y}{\partial \beta_1}\right)^2\right] - \mathrm{E}\left[\frac{\partial Y}{\partial \beta_1}\right]^2 \mathrm{Var}\left(\frac{\partial Y}{\partial \beta_1}\right)_{\mathrm{IPA}} = \frac{2+\beta_2/\beta_1}{(1+\beta_2/\beta_1)^4} (\beta_2/\beta_1)^3$$

This equation was plotted putting the ratio β_2/β_1 in the horizontal axis using a logarithmic scale. Observe in Figure 3.9 that the variance is close to zero for small values of β_1/β_2 and it is still small when $\beta_1/\beta_2 = 1$ (means of the two arcs are the same). This behavior is totally different to the critical (maximum) path behavior. Also note that the variance does not depends on the scaling of the network, i.e., if β_1 and β_2 are increased or decreased keeping the ratio between them constant, the variance does not change.

SF/LR estimator

Taking $Y = \min(X_1, X_2)$, where $X_1 \sim \exp(\beta_1)$ and $X_2 \sim \exp(\beta_2)$, the shortest path distribution for this particular setup is:

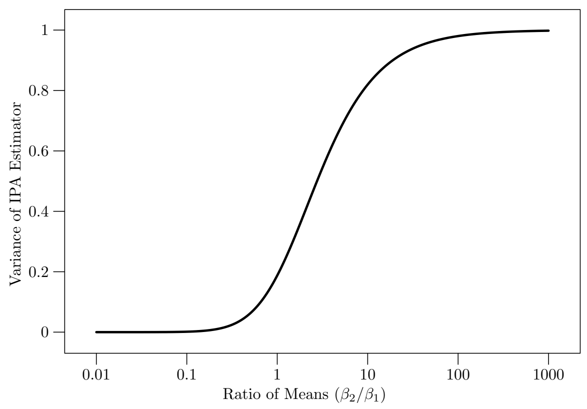


Figure 3.9: Relation between the means ratio and variance of the IPA estimator for a SAN of two parallel, minimum path and exponentially distributed arcs.

$$\begin{aligned} F_Y(y) = P(\min(X_1, X_2) < y) &= 1 - P(\min(X_1, X_2) > y) \\ &= 1 - P(X_1 > y)P(X_2 > y) \\ &= 1 - \exp(-y/\beta_1)\exp(-y/\beta_2)] \\ &= 1 - \exp(-y/\beta) \qquad (3.76) \end{aligned}$$

Hence, Y is also a exponentially distributed RV, with mean $\beta = \beta_1\beta_2/(\beta_1+\beta_2)$. Now, the SF/LR estimator is given by:

$$\begin{aligned} Y \cdot \frac{\partial}{\partial\beta_1} \ln f_Y(Y) &= Y \cdot \frac{1}{\beta}\left(\frac{Y}{\beta} - 1\right) \cdot \frac{d\beta}{d\beta_1} \\ &= Y\frac{1}{\beta}\left(\frac{Y}{\beta} - 1\right) \cdot \left(\frac{\beta_2/\beta_1}{1 + \beta_2/\beta_1}\right)^2 \qquad (3.77) \end{aligned}$$

Therefore, the problem has been reduced to the single arc with exponentially distributed time, weighted by a factor. The factor is constant for given β_1 and β_2.

In conclusion, the variance of the SF/LR estimator is known from previous section:

$$\operatorname{Var}\left(\frac{\partial Y}{\partial \beta_1}\right)_{\mathrm{SF/LR}} = 13\left(\frac{\beta_2/\beta_1}{1+\beta_2/\beta_1}\right)^4 \tag{3.78}$$

This equation is plotted in Figure 3.10 using the ratio (β_2/β_1).

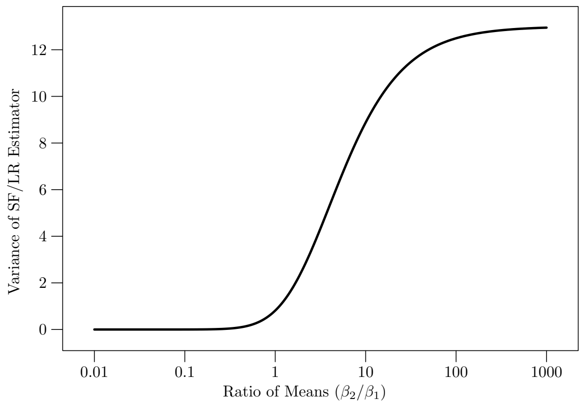


Figure 3.10: Relation between the means ratio and variance of the SF/LR estimator for a SAN of two parallel, minimum path and exponentially distributed arcs.

WD estimator

The variance of the weak derivative based gradient estimator is:

$$\operatorname{Var}\left(\frac{\partial Y}{\partial \beta_1}\right)_{\mathrm{WD}} = \frac{1}{\beta_1^2}\left(\operatorname{Var}\left(Y^{(2)}\right) + \operatorname{Var}\left(Y^{(1)}\right) - 2\operatorname{Cov}\left(Y^{(2)}, Y^{(1)}\right)\right) \tag{3.79}$$

It is needed to take into account the ‘coupling’ used to generate the phantom samples. In this case again it is assumed that the random Erlang variates $X^{(2)}$ were generated summing two independent exponential variates: X_1 and X_3, where X_1, $X_3 \sim \exp(\beta_1)$. On the other hand, X_2 is still an exponentially distributed variable, i.e. $X_2 \sim \exp(\beta_2)$, and $X^{(1)}$ is X_1. Using these assumptions, we have that:

$$Y^{(2)} = \min\left(X^{(2)}, X_2)\right) = \min\left(X_1 + X_3, X_2\right) \tag{3.80}$$

$$Y^{(1)} = \min\left(X^{(1)}, X_2)\right) = \min\left(X_1, X_2\right) \tag{3.81}$$

Note that given $Y^{(1)}$ definition, we already know that $Y^{(1)} \sim \exp(\beta)$, where $\beta = \beta_1\beta_2/\left(\beta_1 + \beta_2\right)$. Hence, we can write:

$$\mathrm{E}\left[Y^{(1)}\right] = \frac{\beta_1\beta_2}{\beta_1 + \beta_2} \tag{3.82}$$

$$\mathrm{Var}\left(Y^{(1)}\right) = \left(\frac{\beta_1\beta_2}{\beta_1 + \beta_2}\right)^2 \tag{3.83}$$

Now, let's compute some expectations of $Y^{(2)}$, needed for variance calculation. Notice that the expectation integral can be decomposed in three parts depending on the relative magnitudes of X_1, X_2 and X_3.

$$\begin{aligned}
\mathrm{E}\left[Y^{(2)}\right] = & \int_{x_2=0}^{\infty}\int_{x_1=0}^{x_2}\int_{x_3=0}^{x_2/x_3} \left(x_1 + x_2\right) f_{X_1}(x_1) f_{X_2}(x_2) f_{X_3}(x_3) dx_3 dx_2 dx_1 \\
& + \int_{x_2=0}^{\infty}\int_{x_1=0}^{x_2}\int_{x_3=x_2/x_3}^{\infty} x_2 f_{X_1}(x_1) f_{X_2}(x_2) f_{X_3}(x_3) dx_3 dx_2 dx_1 \\
& + \int_{x_2=0}^{\infty}\int_{x_1=x_2}^{\infty}\int_{x_3=0}^{\infty} x_2 f_{X_1}(x_1) f_{X_2}(x_2) f_{X_3}(x_3) dx_3 dx_2 dx_1 \\
\mathrm{E}\left[Y^{(2)}\right] = & \; 2\frac{\beta_1\beta_2^3}{\left(\beta_1 + \beta 2\right)^3} + 2\frac{\beta_1^2\beta_2^2}{\left(\beta_1 + \beta 2\right)^3} + \frac{\beta_1^2\beta_2}{\left(\beta_1 + \beta 2\right)^2} \\
= & \; \frac{\beta_1\beta_2}{\left(\beta_1 + \beta 2\right)^2}\left(\beta_1 + 2\beta_2\right)
\end{aligned} \tag{3.84}$$

$$\begin{aligned}
\mathrm{E}\left[(Y^{(2)})^2\right] &= \int_{x_2=0}^{\infty}\int_{x_1=0}^{x_2}\int_{x_3=0}^{x_2/x_3} (x_1+x_2)^2 f_{X_1}(x_1) f_{X_2}(x_2) f_{X_3}(x_3) dx_3 dx_2 dx_1 \\
&+ \int_{x_2=0}^{\infty}\int_{x_1=0}^{x_2}\int_{x_3=x_2/x_3}^{\infty} x_2^2 f_{X_1}(x_1) f_{X_2}(x_2) f_{X_3}(x_3) dx_3 dx_2 dx_1 \\
&+ \int_{x_2=0}^{\infty}\int_{x_1=x_2}^{\infty}\int_{x_3=0}^{\infty} x_2^2 f_{X_1}(x_1) f_{X_2}(x_2) f_{X_3}(x_3) dx_3 dx_2 dx_1 \\
\mathrm{E}\left[(Y^{(2)})^2\right] &= 6\frac{\beta_1^2\beta_2^4}{(\beta_1+\beta_2)^4} + 6\frac{\beta_1^3\beta_2^3}{(\beta_1+\beta_2)^4} + 2\frac{\beta_1^3\beta_2^2}{(\beta_1+\beta_2)^3} \\
&= 2\frac{\beta_1^2\beta_2^2}{(\beta_1+\beta_2)^3}(\beta_1+3\beta_2)
\end{aligned} \tag{3.85}$$

With these results we can give an expression for the variance of $Y^{(2)}$.

$$\begin{aligned}
\mathrm{Var}\left(Y^{(2)}\right) &= \mathrm{E}\left[\left(Y^{(2)}\right)^2\right] - \mathrm{E}\left[Y^{(2)}\right]^2 \\
&= \frac{\beta_1^2\beta_2^2}{(\beta_1+\beta_2)^4}\left(\beta_1^2 + 4\beta_1\beta_2 + 2\beta_2^2\right)
\end{aligned} \tag{3.86}$$

Now we need to give an expression for the Covariance between $Y^{(2)}$ and $Y^{(1)}$. Let us follow the covariance definition:

$$\begin{aligned}
\mathrm{Cov}\left(Y^{(2)}, Y^{(1)}\right) &= \mathrm{E}\left[Y^{(2)}Y^{(1)}\right] - \mathrm{E}\left[Y^{(2)}\right]\mathrm{E}\left[Y^{(1)}\right] \\
&= \mathrm{E}\left[\min(X_1+X_3, X_2)\cdot\min(X_1, X_2)\right] \\
&-\mathrm{E}\left[\min(X_1+X_3, X_2)\right]\mathrm{E}\left[\min(X_1, X_2)\right]
\end{aligned} \tag{3.87}$$

First, the expected value of the product can be calculated using the same integration technique presented above.

$$
\begin{aligned}
\mathrm{E}\left[Y^{(2)}Y^{(1)}\right] = & \int_{x_2=0}^{\infty}\int_{x_1=0}^{x_2}\int_{x_3=0}^{x_2-x_1}\left(x_1+x_3\right)x_1 f_{X_1}(x_1)f_{X_2}(x_2)f_{X_3}(x_3)dx_3dx_1dx_2 \\
& + \int_{x_2=0}^{\infty}\int_{x_1=0}^{x_2}\int_{x_3=x_2-x_1}^{\infty} x_2x_1 f_{X_1}(x_1)f_{X_2}(x_2)f_{X_3}(x_3)dx_3dx_1dx_2 \\
& + \int_{x_2=0}^{\infty}\int_{x_1=x_2}^{\infty}\int_{x_3=0}^{\infty} x_2x_2 f_{X_1}(x_1)f_{X_2}(x_2)f_{X_3}(x_3)dx_3dx_1dx_2 \\
\mathrm{E}\left[Y^{(2)}Y^{(1)}\right] = & 3\frac{\beta_1^2\beta_2^4}{\left(\beta_1+\beta_2\right)^4}+3\frac{\beta_1^3\beta_2^3}{\left(\beta_1+\beta_2\right)^4}+2\frac{\beta_1^3\beta_2^2}{\left(\beta_1+\beta_2\right)^3} \\
\mathrm{E}\left[Y^{(2)}Y^{(1)}\right] = & 3\frac{\beta_1^2\beta_2^3}{\left(\beta_1+\beta_2\right)^4}\left(2\beta_1+\beta_2\right)
\end{aligned}
\tag{3.88}
$$

Hence, the covariance between $Y^{(2)}$ and $Y^{(1)}$ is:

$$
\begin{aligned}
\mathrm{Cov}\left(Y^{(2)},Y^{(1)}\right) &= 3\frac{\beta_1^2\beta_2^3}{\left(\beta_1+\beta_2\right)^4}\left(2\beta_1+\beta_2\right)-\frac{\beta_1\beta_2}{\left(\beta_1+\beta 2\right)^2}\left(\beta_1+2\beta_2\right)\cdot\frac{\beta_1\beta_2}{\beta_1+\beta_2} \\
&= \left(\frac{\beta_1\beta_2}{\beta_1+\beta_2}\right)^2 = \beta^2
\end{aligned}
\tag{3.89}
$$

Now the variance of the WD estimator can be written as:

$$
\begin{aligned}
\therefore \mathrm{Var}\left(\frac{\partial Y}{\partial\beta_1}\right)_{\mathrm{WD}} &= \frac{1}{\beta_1^2}\left(\frac{\beta_1^2\beta_2^2}{\left(\beta_1+\beta_2\right)^4}\left(\beta_1^2+4\beta_1\beta_2+2\beta_2^2\right)+\beta^2-2\beta^2\right) \\
&= \left(\beta_2/\beta_1\right)^3\frac{2+\beta_2/\beta_1}{\left(1+\beta_2/\beta_1\right)^4}
\end{aligned}
\tag{3.90}
$$

Figure 3.11 shows a logarithmic graph in the horizontal axis, where we put the ratio of the means (β_2/β_1) and in the vertical axis is the variance of the WD estimator.

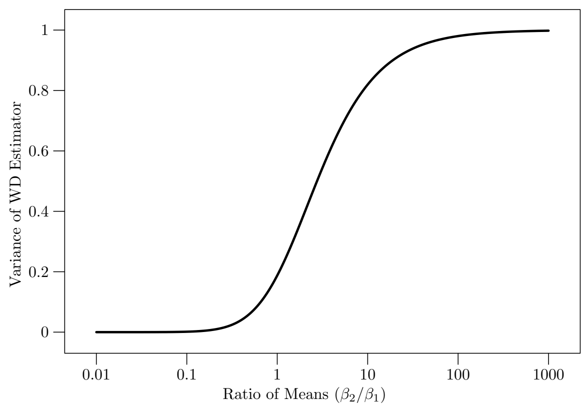


Figure 3.11: Relation between the ratio of means and variance of the weak derivative based estimator for a SAN of two parallel, minimum path and exponentially distributed arcs.

3.5 Series-parallel combination

For this section just consider the IPA estimator of the network shown in Figure 3.12. It will allow us to study the effect of the criticality and relative weight of an arc with respect to the rest of the network. The IPA estimator is mathematically tractable by computing the variance using multiple integrals to obtain the expected value of the estimator and the expected value of the square of the estimator.

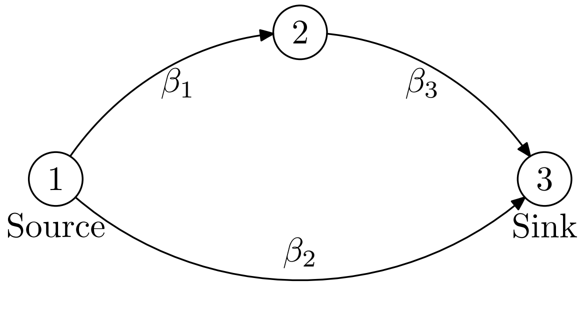


Figure 3.12: Series-parallel network configuration.

Thus, using the following definition of variance, we have:

$$\operatorname{Var}(Z) = \mathrm{E}\left[Z^2\right] - \mathrm{E}\left[Z\right]^2, \tag{3.91}$$

where Z is the IPA estimator with respect to activity 1 of $Y = \max(X_1 + X_3, X_2)$. Z is a RV by itself and it is given by:

$$Z = \begin{cases} -\ln U_1 & \text{if } -\beta_1 \ln U_1 - \beta_3 \ln U_3 > -\beta_2 \ln U_2 \\ 0 & \text{otherwise} \end{cases} \tag{3.92}$$

The last expression considers that X_1, X_2 and X_3 where generated using the inverse method, where U_1, U_2 and U_3 are all Unif(0,1) IID variables.

From here we can compute the expected value of the IPA estimator Z by integration, taking into account the ranges of U_1, U_2 and U_3 for which $X_1 + X_3 > X_2$.

$$\begin{aligned} \mathrm{E}\left[Z\right] &= \int_0^1 \int_0^1 \int_0^1 Z \mathrm{d}u_1 \mathrm{d}u_2 \mathrm{d}u_3 \\ &= \int_{u_2=0}^1 \int_{u_1=0}^{u_2^{\beta_2/\beta_1}} \int_{u_3=0}^1 -\ln U_1 \mathrm{d}u_3 \mathrm{d}u_1 \mathrm{d}u_2 \\ &+ \int_{u_2=0}^1 \int_{u_1=u_2^{\beta_2/\beta_1}}^1 \int_{u_3=0}^{u_2^{\beta_2/\beta_3}/u_1^{\beta_1/\beta_3}} -\ln U_1 \mathrm{d}u_3 \mathrm{d}u_1 \mathrm{d}u_2 \\ &= 1 - \frac{\beta_2^3}{\left(\beta_1 + \beta_2\right)^2 \left(\beta_3 + \beta_2\right)} \end{aligned} \tag{3.93}$$

Same procedure is followed for the expectation of Z squared.

$$\begin{aligned} \mathrm{E}\left[Z^2\right] &= \int_0^1 \int_0^1 \int_0^1 Z \mathrm{d}u_1 \mathrm{d}u_2 \mathrm{d}u_3 \\ &= \int_{u_2=0}^1 \int_{u_1=0}^{u_2^{\beta_2/\beta_1}} \int_{u_3=0}^1 \left(\ln U_1\right)^2 \mathrm{d}u_3 \mathrm{d}u_1 \mathrm{d}u_2 \\ &+ \int_{u_2=0}^1 \int_{u_1=u_2^{\beta_2/\beta_1}}^1 \int_{u_3=0}^{u_2^{\beta_2/\beta_3}/u_1^{\beta_1/\beta_3}} \left(\ln U_1\right)^2 \mathrm{d}u_3 \mathrm{d}u_1 \mathrm{d}u_2 \\ &= 2 - \frac{2\beta_2^4}{\left(\beta_1 + \beta_2\right)^3 \left(\beta_3 + \beta_2\right)} \end{aligned} \tag{3.94}$$

Integrating and combining these intermediate results we can get the closed-form expression of the variance of the IPA gradient estimator for this particular SAN of exponentially distributed arc times.

$$\operatorname{Var}(Z) = 1 + \frac{\beta_2^3\left(2\beta_1\left(\beta_1+\beta_2\right)\left(\beta_3+\beta_2\right)-\beta_2^3\right)}{\left(\beta_1+\beta_2\right)^4\left(\beta_3+\beta_2\right)^2} \tag{3.95}$$

This expression is not as simple as the previous ones, but it is still useful to analyze. In this part, we are interested in observe how the criticality of the first branch affects the precision of the IPA estimator. In order to achieve this objective, we need the closed-form expression for the criticality index of the first activity given the means of the three activities β_1, β_2 and β_3. The criticality index is the probability that the upper path ($Y_1 = X_1 + X_3$) is longer than the lower one (X_2).

$$\begin{aligned} P(Y_1 > X_2) &= \int_{y_1=0}^{\infty} P(X_2 < Y_1 | Y_1 = y_1) f_{Y_1}(y_1)\mathrm{d}y_1 \\ = \int_{y_1=0}^{\infty} F_{X_2}(y_1) f_{Y_1}(y_1)\mathrm{d}y_1 &= \int_{y_1=0}^{\infty}\int_{x_2=0}^{y_1} f_{X_2}(x_2)\mathrm{d}x_2 f_{Y_1}(y_1)\mathrm{d}y_1 \\ &= \int_{y_1=0}^{\infty}\int_{x_2=0}^{y_1} f_{X_2}(x_2) f_{Y_1}(y_1)\mathrm{d}x_2\mathrm{d}y_1 \end{aligned} \tag{3.96}$$

The distribution of Y_1 and X_2 are needed to calculate this last expression. As it was mentioned before, the distribution of a sum of exponentially distributed RVs is called Hypoexponential. If the means of the the two activities are exactly the same, we obtained a Erlang distribution instead.

Let's first consider the case of different means, i.e. $\beta_1 \neq \beta_3$.

$$Y_1 \sim \operatorname{HypoExp}(\beta_1, \beta_3) \iff f_{Y_1}(y) = -\underline{\alpha} e^{y\Theta}\Theta\underline{1} \tag{3.97}$$

, where

$$\underline{\alpha} = (1,0) \quad , \Theta = \begin{pmatrix} -\beta_1^{-1} & \beta_1^{-1} \\ 0 & -\beta_3^{-1} \end{pmatrix} \quad , \underline{1} = (1,1)^T$$

Hence, we get:

$$f_{Y_1}(y) = \frac{e^{-y/\beta_1} - e^{-y/\beta_3}}{\beta_1 - \beta_3} \tag{3.98}$$

On the other hand, for X_2 we already know that its PDF is:

$$f_{X_2}(x) = \frac{e^{-x/\beta_2}}{\beta_2} \tag{3.99}$$

Plugging back these results in equation (3.96) we obtain:

$$\begin{aligned} P(Y_1 > X_2) &= \int_{y_1=0}^{\infty} \int_{x_2=0}^{y_1} \frac{e^{-x_2/\beta_2}}{\beta_2} \frac{e^{-y_1/\beta_1} - e^{-y_1/\beta_3}}{\beta_1 - \beta_3} \mathrm{d}x_2 \mathrm{d}y_1 \\ P(Y_1 > X_2) &= 1 - \frac{\beta_2^2}{(\beta_1 + \beta_2)(\beta_3 + \beta_2)} \end{aligned} \tag{3.100}$$

Hence, we obtain a closed-form expression for the criticality as a function of the activity times means.

To check that this last expression is valid in the case of $\beta_1 = \beta_3$, we repeat the previous steps but using Erlang distribution for $Y_1 = X_1 + X_3$ in equation (3.96):

$$\begin{aligned} P(Y_1 > X_2) &= \int_{y_1=0}^{\infty} \int_{x_2=0}^{y_1} \frac{e^{-x_2/\beta_2}}{\beta_2} \frac{y_1 e^{-y_1/\beta_1}}{\beta_1^2} \mathrm{d}x_2 \mathrm{d}y_1 \\ P(Y_1 > X_2) &= \frac{\beta_1^2 + 2\beta_1\beta_2}{(\beta_1 + \beta_2)^2} \\ P(Y_1 > X_2) &= 1 - \frac{\beta_2^2}{(\beta_1 + \beta_2)^2} \end{aligned} \tag{3.101}$$

This expression is in accordance with equation (3.100) when $\beta_1 = \beta_3$.

Now we need to relate these closed-form equations with the concepts of criticality and relative weight. First notice that scaling (multiplying every mean parameter β by the same factor) does not alter the variance of the gradient estimator in this exponentially distributed instance. Then we can conveniently choose β_1, β_2 and β_3 to get the desired relative weight w and criticality c from the multiples ways to select β_1, β_2 and β_3. Also remember that β_1, β_2 and β_3 are greater than zero.

Hence, given the relative weight w, we fixed the relative magnitude of β_1 and β_3, because:

$$w = \frac{\beta_1}{\beta_1 + \beta_3} \tag{3.102}$$

Now, because of the invariant scale property, arbitrarily assume $\beta_1 + \beta_3 = 1$, then β_1 is the relative weight and $\beta_3 = 1 - \beta_1$.

Second, given a target criticality and with β_1 and β_3 fixed from the previous step, we can write a quadratic equation in β_2.

$$c = 1 - \frac{\beta_2^2}{(\beta_1 + \beta_2)(\beta_3 + \beta_2)}$$

$$(\beta_1 + \beta_2)(\beta_3 + \beta_2)(1 - c) = \beta_2^2$$

$$c\beta_2^2 - (\beta_1 + \beta_3)(1 - c)\beta_2 - \beta_1\beta_3(1 - c) = 0$$

$$\Rightarrow \beta_2 = \frac{(\beta_1 + \beta_3)(1 - c) \pm \sqrt{(\beta_1 + \beta_3)^2 (1 - c)^2 + 4c\beta_1\beta_3(1 - c)}}{2c} \tag{3.103}$$

Notice that the minus sign always yields negative β_2 because $4c\beta_1\beta_3(1 - c) > 0$. Thus, the plus sign must be used to obtain valid values of β_2.

Putting these last results together, we have:

$$\beta_1 = w \tag{3.104}$$

$$\beta_3 = 1 - w \tag{3.105}$$

$$\beta_2 = \frac{(1-c) + \sqrt{(1-c)^2 + 4c \cdot w\,(1-w)\,(1-c)}}{2c} \tag{3.106}$$

Figure 3.13 shows how the precision of the estimator varies with criticality of the first activity, as the relative weight is maintained constant.

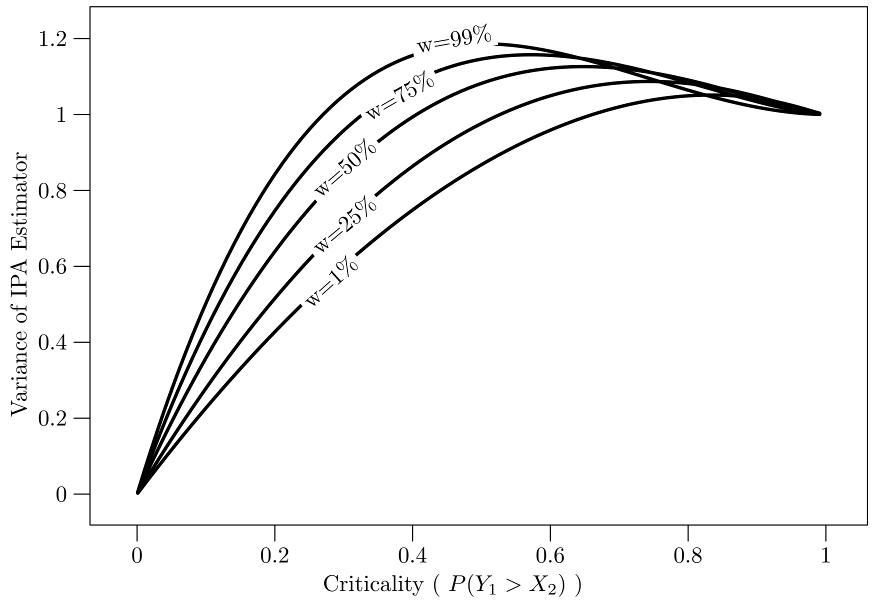


Figure 3.13: Variance curves for IPA estimator for series-parallel network configuration.

On the other hand, Figure 3.14 shows how variance varies with the relative weight as the criticality is kept constant.

Another observation is that the precision of the IPA estimator does not change if criticality and relative weight are fixed, even though the expected times of the

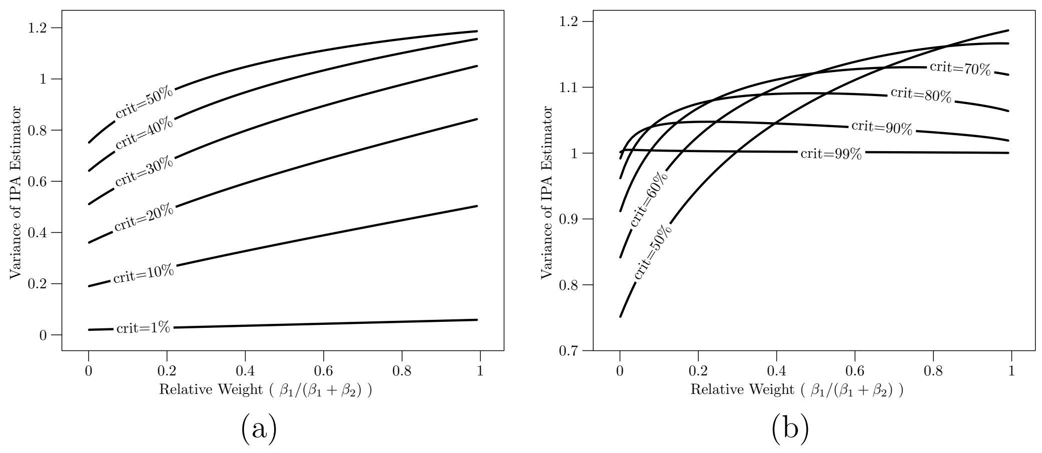


Figure 3.14: IPA estimator variance curves for series-parallel network and fixed criticality (a) 1% <crit< 50%, (b) 50% <crit< 99%.

activities were altered. This observation means that the IPA estimator is invariant with respect to the scale factor of the network. That is a behavior common to many networks analyzed so far.

One interesting conjecture to explore is: does the IPA estimator change if criticality and relative weight are kept constant in the path of interest but we change the complexity of the rest of the network?

It is also interesting that the curves are not monotone, as they have a maximum between the 50% and 100% of criticality, depending on the relative weight. This behavior suggest that at the maximum point,the variability of the total completion time is maximum since it is exists the maximum uncertainty about which path is going to be the optimum (longest) one at each realization. Accordingly, we propose to describe this uncertainty using Entropy, an information-theoretic measure introduced by Shannon [22]. Then we will find out how it compares to the variance.

3.5.1 Criticality-Weight influence in distinct SANs

This section is devoted to obtain the IPA estimator variance vs. criticality for two different SANs. This exercise is of interest because we want to know if the precision of IPA estimators is independent of the network structure for a particular and fixed combination of criticality and relative weight of the arc.

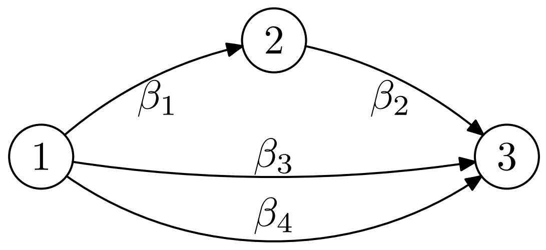


Figure 3.15: Three parallel paths Stochastic activity network, one 2 series arc path. Exponentially distributed activity times with means β_i

Let us obtain the IPA estimator variance for a SAN of 3 parallel paths and the first one consisting in two serial activities as is shown in Figure 3.15. This new network will be compared with the one used in Section 3.5 and exponentially distributed times will be considered.

Let Z be the IPA estimator (a RV by itself), defined in the following way:

$$Z = \begin{cases} \frac{X_1}{\beta_1} & \text{if } X_1 + X_2 > \max(X_3, X_4) \\ 0 & \text{otherwise} \end{cases} \tag{3.107}$$

Then we can get the expected value by integration for the exponential PDFs.

$$\begin{aligned}
\mathrm{E}\left[Z\right] &= \int_{x_1=0}^{\infty}\int_{x_2=0}^{\infty}\int_{x_3=0}^{x_1+x_2}\int_{x_4=0}^{x_1+x_2}\frac{x_1}{\beta_1}f_{X_4}(x_4)f_{X_3}(x_3)f_{X_2}(x_2)f_{X_1}(x_1)\mathrm{d}x_4\mathrm{d}x_3\mathrm{d}x_2\mathrm{d}x_1 \\
&= \int_{x_1=0}^{\infty}\int_{x_2=0}^{\infty}\frac{x_1}{\beta_1}\left(1-e^{-\frac{x_1+x_2}{\beta_3}}\right)\left(1-e^{-\frac{x_1+x_2}{\beta_4}}\right)f_{X_2}(x_2)f_{X_1}(x_1)\mathrm{d}x_2\mathrm{d}x_1 \\
&= 1-\frac{\beta_4^3}{\left(\beta_1+\beta_4\right)^2\left(\beta_2+\beta_4\right)} \\
&+\beta_3^3\left(\frac{\beta_4^3}{\left(\beta_3\beta_4+\beta_1\left(\beta_3+\beta_4\right)\right)^2\left(\beta_3\beta_4+\beta_2\left(\beta_3+\beta_4\right)\right)}-\frac{1}{\left(\beta_1+\beta_3\right)^2\left(\beta_2+\beta_3\right)}\right)
\end{aligned} \tag{3.108}$$

$$\begin{aligned}
\mathrm{E}\left[Z^2\right] &= \int_{x_1=0}^{\infty}\int_{x_2=0}^{\infty}\int_{x_3=0}^{x_1+x_2}\int_{x_4=0}^{x_1+x_2}\frac{x_1^2}{\beta_1^2}f_{X_4}(x_4)f_{X_3}(x_3)f_{X_2}(x_2)f_{X_1}(x_1)\mathrm{d}x_4\mathrm{d}x_3\mathrm{d}x_2\mathrm{d}x_1 \\
&= \int_{x_1=0}^{\infty}\int_{x_2=0}^{\infty}\frac{x_1^2}{\beta_1^2}\left(1-e^{-\frac{x_1+x_2}{\beta_3}}\right)\left(1-e^{-\frac{x_1+x_2}{\beta_4}}\right)f_{X_2}(x_2)f_{X_1}(x_1)\mathrm{d}x_2\mathrm{d}x_1 \\
&= 2-\frac{2\beta_4^4}{\left(\beta_1+\beta_4\right)^3\left(\beta_2+\beta_4\right)} \\
&+\beta_3^4\left(\frac{2\beta_4^4}{\left(\beta_3\beta_4+\beta_1\left(\beta_3+\beta_4\right)\right)^3\left(\beta_3\beta_4+\beta_2\left(\beta_3+\beta_4\right)\right)}-\frac{2}{\left(\beta_1+\beta_3\right)^3\left(\beta_2+\beta_3\right)}\right)
\end{aligned} \tag{3.109}$$

Thus, we can obtain the variance of the IPA estimator using Equation (3.91).

An expression for the relative weight in this network is just $w=\beta_1/\left(\beta_1+\beta_2\right)$, but a closed-form expression for the criticality index is needed.

$$\begin{aligned}
CI = P\left(X_1+X_2>\max\left(X_3,X_4\right)\right) &= \int_{x_1=0}^{\infty}\int_{x_2=0}^{\infty}\int_{x_3=0}^{x_1+x_2}\int_{x_4=0}^{x_1+x_2}f_{X_4}(x_4)f_{X_3}(x_3)f_{X_2}(x_2)f_{X_1}(x_1)\mathrm{d}x_4 \\
&= 1-\frac{\beta_4^3}{\left(\beta_1+\beta_4\right)^2\left(\beta_2+\beta_4\right)} \\
&+\beta_3^3\left(\frac{\beta_4^3}{\left(\beta_3\beta_4+\beta_1\left(\beta_3+\beta_4\right)\right)^2\left(\beta_3\beta_4+\beta_2\left(\beta_3+\beta_4\right)\right)}-\frac{1}{\left(\beta_1+\beta_3\right)^2\left(\beta_2+\beta_3\right)}\right)
\end{aligned} \tag{3.110}$$

Choosing β_1, β_2, β_3 and β_4 conveniently, implies apply some restrictions to get criticality and relative weight. Hence, we have:

$$\beta_1 = w$$

$$\beta_2 = 1 - w$$

$$\beta_3 = \beta_4 \tag{3.111}$$

where β_4 satisfies the following equation:

$$c\beta_4^4 + 3c\beta_4^3 - \left(2 - 2c - 2w - 5cw + 2w^2 + 5cw^2\right)\beta_4^2 - 6\left(w - cw - w^2 + cw^2\right)\beta_4$$
$$= 4w^2 - 4cw^2 - 8w^3 + 8cw^3 + 4w^4 - 4cw^4 \tag{3.112}$$

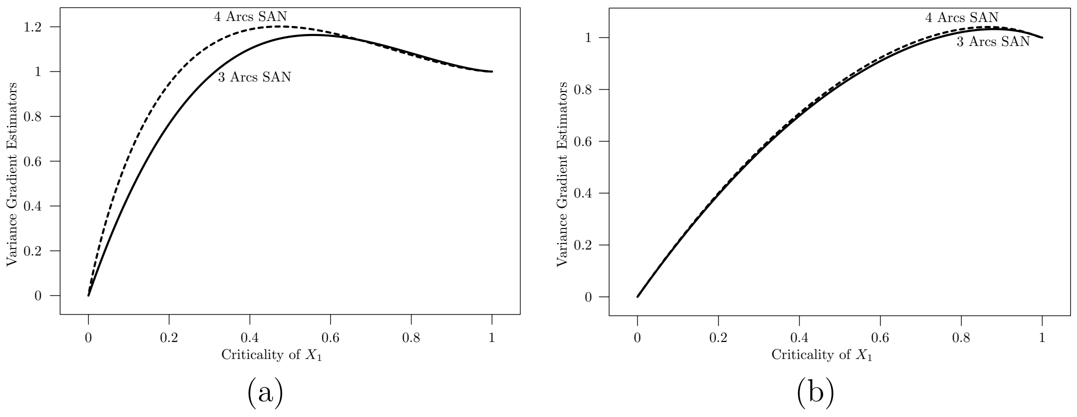


Figure 3.16: IPA estimator variance curves for different complexity networks (a) $w = 80\%$, (b) $w = 5\%$.

Now we can plot the variance curve of this SAN together with the variance obtained in Section 3.5 in page 66 to compare both results. Figure 3.16 present curves two relative weights. Note how curves approximate when relative is weight is very small. In the limit, when relative weight goes to zero, we have the case of a

Bernoulli mixture of continuous RVs, in which the Bernoulli trial is independent of the continuous RVs.

In the limiting case in which the relative weight of the arc of interest is very small, we can approximate the IPA estimator Z by:

$$Z = \frac{X_1}{\beta_1} \cdot \mathbb{1}\{S = 1\} \tag{3.113}$$

where $S \sim Bern(c)$ i.e., a Bernoulli RV of parameter c and independent of X_1 which is exponentially distributed. Then we can write:

$$\begin{aligned} \mathrm{Var}(Z) &= \mathrm{Var}(\mathrm{E}[Z|S]) + \mathrm{E}[\mathrm{Var}(Z|S)] \\ &= \mathrm{Var}(Bern(c)) + \mathrm{E}[Bern(c)] = c(1-c) + c = 2c - c^2 \end{aligned} \tag{3.114}$$

This last expression is independent of the structure of the network and it only depends on the criticality index of the first arc.

3.5.2 IPA estimator entropy of two parallel SAN and series-parallel combination SAN

In this section, let us analyze the uncertainty of the IPA estimator distribution, by the means of the entropy and differential entropy. The idea is to identify how the entropy of the estimator relates with the input distributions and the shape of the network.

Using definitions taken from work of Nair et.al. [19], the entropy of a mixed RV X can be obtained.

$$H(X) = -\sum_i p(x_i) \log(p(x_i)) - \int_A f(\xi) \log(f(\xi))\, \mathrm{d}\xi \tag{3.115}$$

where X takes the discrete values x_i w.p. $p(x_i)$ or takes values from some interval $A \in \mathbb{R}$ w.p. $P(X \in B \subseteq A) = \int_B f(\xi)\mathrm{d}\xi$ and

$$\sum_i p(x_i) + \int_A f(\xi)\mathrm{d}\xi = 1 \tag{3.116}$$

In this particular case, our mixed RV Z is the IPA estimator for the two parallel arcs, defined in the following way:

$$Z = \begin{cases} \frac{X_1}{\beta_1} & \text{if } X_1 > X_2 \\ 0 & \text{otherwise} \end{cases} \tag{3.117}$$

Let us start getting the distribution of Z:

$$\begin{aligned} P(Z < z) &= P(Z = 0) + P(0 < Z < z) \\ &= (1-c) + P(Z < z | X_1 > X_2) P(X_1 > X_2) \\ &= (1-c) + P(X_1 < z\beta_1, X_1 > X_2) \\ &= (1-c) + \int_{x_2=0}^{z\beta_1} \int_{x_1=x_2}^{z\beta_1} f_{X_1}(x_1)\mathrm{d}x_1 f_{X_2}(x_2)\mathrm{d}x_2 \\ &= (1-c) + \frac{\beta_1\left(1-e^{-z}\right) + \beta_2\left(e^{-\frac{\beta_1+\beta_2}{\beta_2}z} - e^{-z}\right)}{\beta_1+\beta_2} \\ &= (1-c) + c\left(1-e^{-z}\right) + (1-c)\left(1 - e^{-z} - \left(1 - e^{\frac{-z}{1-c}}\right)\right) \\ &= F_{\beta=1}(z) + (1-c)\left(1 - F_{\beta=(1-c)}(z)\right) \end{aligned} \tag{3.118}$$

where $c = \beta_1/(\beta_1+\beta_2)$ and $F_\beta(x) = 1 - e^{-x/\beta}$.

Hence, Z has PMF $p(Z=0) = 1-c$ and PDF:

$$f_Z(z) = f_{\beta=1}(z) - (1-c)f_{\beta=1-c}(z) = e^{-z} - e^{\frac{-z}{1-c}} = e^{-z}\left(1 - e^{-\frac{c}{1-c}z}\right) \tag{3.119}$$

The entropy can be computed from this last result.

$$\begin{aligned}
H(Z) =& -(1-c)\log(1-c) \\
&- \int_0^{\infty} e^{-\xi}\left(1-e^{-\frac{c}{1-c}\xi}\right)\log\left(e^{-\xi}\left(1-e^{-\frac{c}{1-c}\xi}\right)\right)\mathrm{d}\xi \\
H(Z) =& -(1-c)\log(1-c) + c\left(\frac{1}{1-c} + H_{(1/c)-2}\right) \qquad (3.120)
\end{aligned}$$

where $H_n = \sum_{i=1}^{n} 1/i$.

Figure 3.17 shows a plot of this last function. It is important to observe that the maximum is reached at a criticality index value of 68% approximately, unlike the variance, which reaches the maximum at 50%.

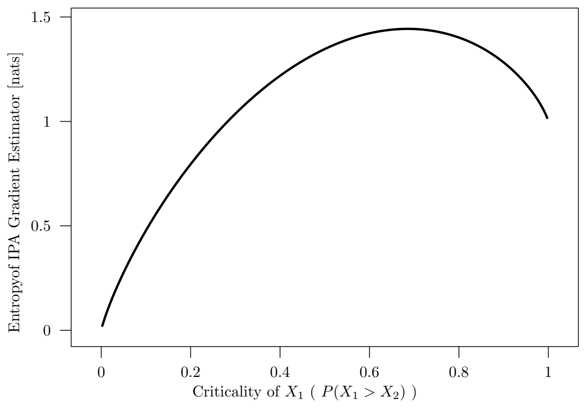


Figure 3.17: Entropy of IPA estimator for a Two parallel arc SAN.

Let us compute a closed-form expression for the entropy of the IPA gradient estimator of series-parallel combination network of 3 arcs presented in Section 3.5(Figure 3.12).

$$\begin{aligned}
P(Z<z) = & P(Z=0)+P(0<Z<z) \\
= & (1-c)+P(0<Z<z|X_1+X_2>X_3)P(X_1+X_2>X_3) \\
= & (1-c)+P(X_1<z\beta_1, X_1>X_3-X_2) \\
= & (1-c)+\int_{x_3=0}^{z\beta_1}\int_{x_2=0}^{x_3} P(x_3-x_2<X_1<z\beta_1)f_{X_2}(x_2)f_{X_3}(x_3)\mathrm{d}x_2\mathrm{d}x_3 \\
& +\int_{x_3=0}^{z\beta_1}\int_{x_2=x_3}^{\infty} P(0<X_1<z\beta_1)f_{X_2}(x_2)f_{X_3}(x_3)\mathrm{d}x_2\mathrm{d}x_3 \\
& +\int_{x_3=z\beta_1}^{\infty}\int_{x_2=x_3-z\beta_1}^{x_3} P(x_3-x_2<X_1<z\beta_1)f_{X_2}(x_2)f_{X_3}(x_3)\mathrm{d}x_2\mathrm{d}x_3 \\
& +\int_{x_3=z\beta_1}^{\infty}\int_{x_2=x_3}^{\infty} P(0<X_1<z\beta_1)f_{X_2}(x_2)f_{X_3}(x_3)\mathrm{d}x_2\mathrm{d}x_3 \\
= & (1-c)+\frac{\beta_1\beta_2+\beta_1\beta_3+\beta_2\beta_3}{(\beta_1+\beta_3)(\beta_2+\beta_3)}\left(1-e^{-z}\right) \\
& -\frac{\beta_3^2}{(\beta_1+\beta_3)(\beta_2+\beta_3)}\left(e^{-z}-e^{-\frac{\beta_1+\beta_3}{\beta_3}z}\right) \\
= & (1-c)+c\left(1-e^{-z}\right)+(1-c)\left(1-e^{-z}-\left(1-e^{-\frac{\beta_1+\beta_3}{\beta_3}z}\right)\right) \\
= & (1-c)+\left(1-e^{-z}\right)-(1-c)\left(1-e^{-\frac{\beta_1+\beta_3}{\beta_3}z}\right) \\
= & (1-c)+F_{\beta=1}(z)-(1-c)F_{\beta=\frac{\beta_3}{\beta_3+\beta_1}}(z) \qquad (3.121)
\end{aligned}$$

where in this case criticality is $c=1-\beta_3^2/(\beta_1+\beta_3)(\beta_2+\beta_3)$ and $F_\beta(z)=1-e^{-z/\beta}$, i.e. $F_\beta(z)$ is again the CDF of an exponentially distributed RV.

Therefore, the IPA estimator Z for this particular network is a RV with a probability mass of $(1-c)$ at $z=0$. Also, by differentiation we obtain the PDF[2] of

[2]Remember that the above mentioned PDF does not add to 1 since part of the probability mass is already at z=0. Thus, the integral of the this PDF sums up to c (see equation (3.116)).

Z as follows:

$$
\begin{aligned}
f_Z(z) =& f_{\beta=1}(z) - (1-c) f_{\beta=\frac{\beta_3}{\beta_3+\beta_1}}(z) \\
=& e^{-z} - (1-c)\left(\frac{\beta_3+\beta_1}{\beta_3} e^{-\frac{\beta_3+\beta_1}{\beta_3}z}\right) \\
=& e^{-z}\left(1-(1-c)\left(1+\frac{\beta_1}{\beta_3}\right)e^{-\frac{\beta_1}{\beta_3}z}\right)
\end{aligned} \tag{3.122}
$$

where $f_\beta(z)$ is the PDF of an exponentially distributed RV.

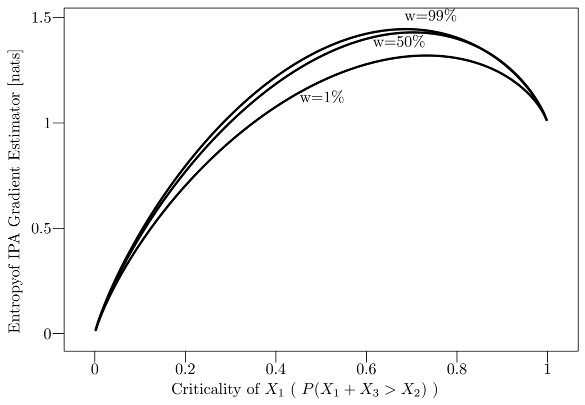


Figure 3.18: Curve of entropy of IPA estimator vs. criticality for 3 fixed relative weights. Series-parallel network with exponentially distributed activities

Now, let us use the formula for entropy for mixed RVs, in the previously deduced distribution.

$$
H(Z) = -(1-c)\log(1-c)
$$
$$
-\int_0^\infty e^{-\xi}\left(1-(1-c)\left(1+\frac{\beta_1}{\beta_3}\right)e^{-\frac{\beta_1}{\beta_3}\xi}\right)\log\left(e^{-\xi}\left(1-(1-c)\left(1+\frac{\beta_1}{\beta_3}\right)e^{-\frac{\beta_1}{\beta_3}\xi}\right)\right)\mathrm{d}\xi
$$

$$
\begin{aligned}
H(Z) = \big(&\beta_1\beta_3\left(\beta_1+\beta_3\right) + \left(\beta_1^2+\beta_1\beta_3+\beta_3^2\right)\beta_2 \\
&- \beta_1^2\left(\beta_2+\beta_3\right){}_2\mathrm{F}_1\left(1, \beta_3/\beta_1, (\beta_1+\beta_3)/\beta_1, \beta_3/(\beta_2+\beta_3)\right) \\
&- 2\beta_3^3\log\left(\beta_3\right) - \beta_3\left(\beta_1\beta_3+\left(\beta_1+\beta_3\right)\beta_2\right)\log\left(\beta_2/(\beta_2+\beta_3)\right) \\
&+\beta_3^3\log\left(\left(\beta_1+\beta_3\right)\left(\beta_2+\beta_3\right)\right)\big) / \left(\beta_3\left(\beta_1+\beta_3\right)\left(\beta_2+\beta_3\right)\right) \qquad (3.123)
\end{aligned}
$$

where ${}_2\mathrm{F}_1$ is the so called Hypergeometric function, defined by the following hypergeometric series:

$$
{}_2\mathrm{F}_1(a,b,c,x) = 1+\frac{ab}{c}\frac{x}{1!}+\frac{a(a+1)b(b+1)}{c(c+1)}\frac{z^2}{2!}+\frac{a(a+1)(a+2)b(b+1)(b+2)}{c(c+1)(c+2)}\frac{z^3}{3!}+\dots \qquad (3.124)
$$

Figure 3.18 shows the entropies curves for different relative weights.

Chapter 4

Experimental Results

4.1 Overview

This section shows the result of experiments conducted to reveal how the structure, size, distributions and parameters of the SAN affects the precision of the computed gradient derivatives.

First, this study explored the effect of the number of activities in a very simple serial structured network taking into account the 3 main methods of direct gradient estimation. We will also check with this experiment the influence of scaling of the network in the precision of the results.

Second, a network with just 2 nodes and different number of parallel arcs was used. Different distributions of activity times with independent random variables are considered to check the influence of criticality. This test continued adding more and more arcs in parallel.

As we expected, the criticality of arcs with respect to other arcs affects the final result, we explore the influence in the variance of this criticality index. It is also studied the effect of the relative 'length' of the arc with respect to the total length of the path.

Finally, a set of networks was tested w.r.t. several complexity index proposed in the graph theory literature. These indices try to measure how intricate the SAN is. In the present study, we try to relate this intricacy with the precision of a given estimator. The following complexity indexes were considered for the this work:

- Number of arcs
- Coefficient of network complexity (CNC_k) proposed by Kaimann
- Coefficient of network complexity (CNC_p) proposed by Pascoe
- Cyclomatic number (S)
- Restrictiveness estimator (RT)
- Number of trees in a graph (T)

4.2 Pure serial configuration

Let us start analyzing the behavior of several direct gradient estimators in two simple configurations: Serial and Parallel.

First, consider a network with N nodes with only one arc connecting each node in a serial fashion as is shown in Figure 4.1. Also, consider IID activity times as well as exponential distributed random variables with mean equal to 1.

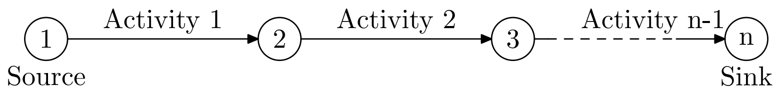


Figure 4.1: Series configuration of stochastic activity network (SAN).

4.2.1 Single activity

We started the experiment with just two nodes and a single arc and increased it up to 100 activities. The gradient is computed with respect to the first activity. Therefore, we expect to obtain a gradient equal to 1 in every configuration.

Let us first compare the gradient values and variance of IPA, SF/LR and WD estimates for the first iteration (two nodes and one arc).

Table 4.1: Comparison of gradient estimator for one arc network Exp($\beta = 1$)

Method	Estimate	Variance	Conf. Interval
IPA	1.0033	1.0066	(0.995, 1.012)
SF/LR	1.0099	13.2631	(0.978, 1.042)
WD	1.0003	0.9990	(0.992, 1.009)

From Table 4.1 we can see in the second column that every method yields the correct (unbiased) value for the gradient, but precision differs greatly between estimators (see third column).

The score function/likelihood ratio (SF/LR) method is the clearly the worst. These results are in complete agreement with the findings of the previous chapter.

Since SF/LR estimates explicitly make use of the length of longest path, the experiment was repeated with a different scaling to check if the variance is affected by this change. We modified the mean in every arc. Table 4.2 shows the results for mean activity times $\beta = 0.01$ and $\beta = 100$.

There were no changes in the results, which is expected since this was already proved theoretically in the previous chapter.

Table 4.2: Comparison of gradient estimator for one arc network, exponential dist., $\beta = 0.01$ and $\beta = 100$

Method	Variance	Conf. Interval
IPA	1.0066	(0.995, 1.012)
SF/LR	13.2631	(0.978, 1.042)
WD	0.9990	(0.992, 1.009)

(a) Arc times mean $\beta = 0.01$

Method	Variance	Conf. Interval
IPA	1.0066	(0.995, 1.012)
SF/LR	13.2631	(0.978, 1.042)
WD	0.9990	(0.992, 1.009)

(b) Arc times mean $\beta = 100$

4.2.2 Multiple activities in series

If we increase the number of arcs to observe its effect, we obtained the results that are presented in Figure 4.2. IID Gaussian distribution was used to generate the activity times for comparison with the theoretical results, i.e., $X_i \sim \mathcal{N}(\mu, \sigma^2)$, $\mu = 30, \sigma = 5$ and considering the gradient w.r.t. the standard deviation of the first arc. Please note how SF/LR estimates degrade when the series network grows in contrast with IPA and WD estimates. This result makes sense if we carefully observe the way SF/LR estimator works: taking into account every arc to build up the final solution, increasing the variance as a consequence.

The experiment was repeated using exponential distribution for every arcs, i.e. $X_i \sim \exp(\beta_i)$, $\beta_i = 1$.

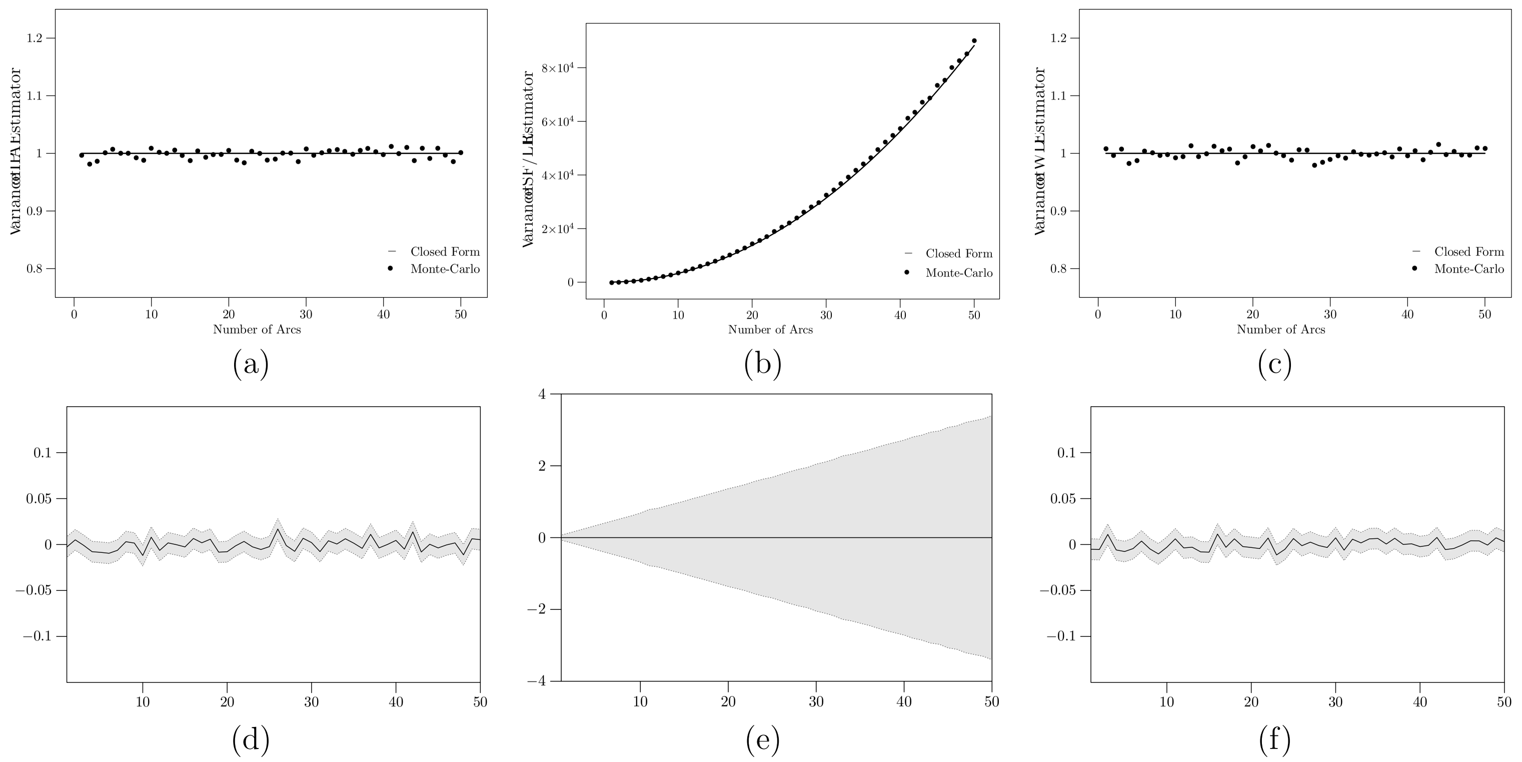


Figure 4.2: Comparison of gradient estimates variances for normally distributed arcs. The second row shows the gradient estimates and the confidence interval. (a)(d) IPA, (b)(e) SF/LR and (c)(f) WD estimators.

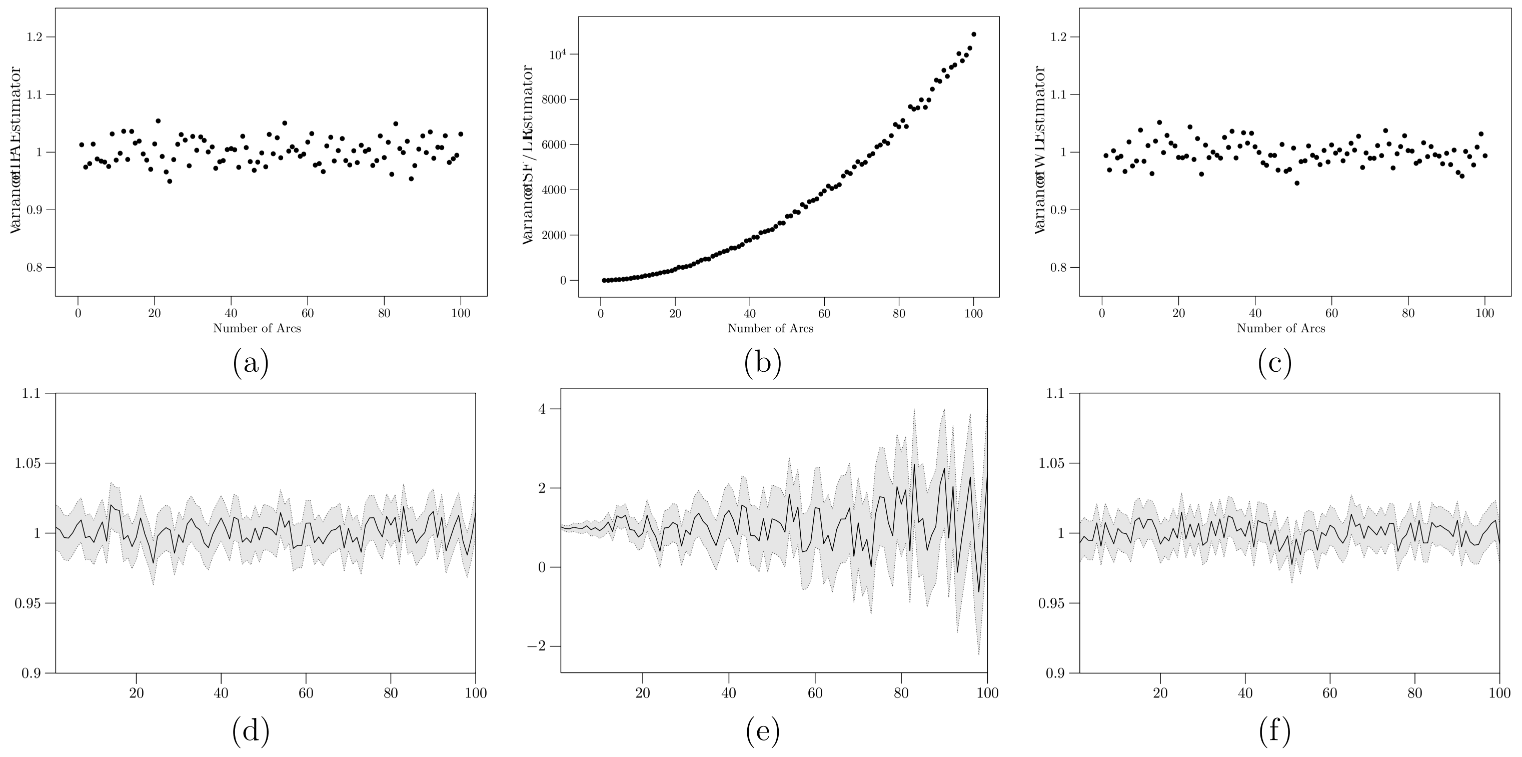


Figure 4.3: Comparison of gradient estimates variances for exponentially distributed arcs. The second row shows the gradient estimates and the confidence interval. (a)(d) IPA, (b)(e) SF/LR and (c)(f) WD estimators.

4.3 Pure parallel configuration

4.3.1 Two parallel arcs

Let us first consider a simple network of just two nodes and 2 activities in parallel (Figure 4.4).

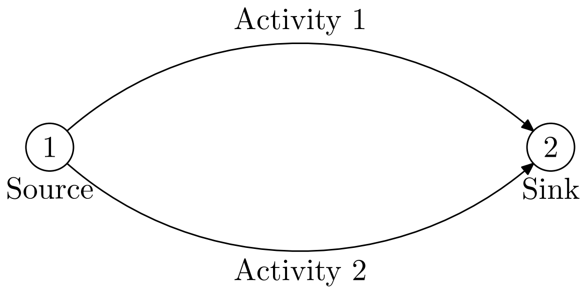


Figure 4.4: Parallel configuration of Stochastic Activity Network (SAN).

We want to know which gradient estimation scheme are more precise and how the criticality index of the first arc and distributions impact each estimator. For **exponentially distributed** arcs, we change the mean activity time β_2. The activity 1 criticality index in this particular network is given by crit= $P(X_1 > X_2) = \beta_1/(\beta_1 + \beta_2)$. The relationship between these quantities was already obtained in closed form for exponential case (see Figures 3.4, 3.6 and 3.7).

Consider Table 4.3 as a crosscheck. It shows the results obtained by Monte Carlo Simulation (MCS). As before, it can be observed that SF/LR estimator is the worst, but it improves slightly as the criticality index increases. IPA and WD gradient estimators variance are very close, but WD estimator is beaten by IPA estimator as criticality increases.

Table 4.3: Comparison of gradient estimator for two arcs parallel network, exponential dist.

Method	Mean	Variance	Conf. Interval
IPA	0.7520	1.1833	(0.7370,0.7671)
SF/LR	0.7513	14.124	(0.6992,0.0834)
WD	0.7536	0.9510	(0.7401,0.7671)
Mean=1.5055, Var=1.2455, Crit=50%			

(a) Arc Means: $\beta_1 = 1$ and $\beta_2 = 1$

Method	Mean	Variance	Conf. Interval
IPA	0.8933	1.1294	(0.8786,0.9081)
SF/LR	0.8878	13.286	(0.8373,0.9383)
WD	0.8931	1.0075	(0.8792,0.9070)
Mean=1.1712, Var=0.9095, Crit=67%			

(b) Arc Means: $\beta_1 = 1$ and $\beta_2 = 0.5$

Method	Mean	Variance	Conf. Interval
IPA	0.9783	1.0374	(0.9641,0.9924)
SF/LR	0.9702	13.019	(0.9202,1.0202)
WD	0.9748	1.0201	(0.9608,0.9888)
Mean=1.0386, Var=0.9491, Crit=84%			

(c) Arc Means: $\beta_1 = 1$ and $\beta_2 = 0.2$

Method	Mean	Variance	Conf. Interval
IPA	1.0052	0.9929	(0.9914,1.0190)
SF/LR	0.9979	12.949	(0.9481,1.0478)
WD	1.0027	1.0203	(0.9807,1.0167)
Mean=1.0054, Var=0.9926, Crit=99%			

(d) Arc Means: $\beta_1 = 1$ and $\beta_2 = 0.01$

Figure 4.5 shows the curves obtained by MCS for the case of exponentially distributed arcs. In the first row x-axes are assigned to criticality whereas in the second row x-axes are logarithmic and they represent the ratio of the means of the arcs 1 and 2. Also in Figure 4.6 IPA and WD estimator variance are plotted together for easy comparison.

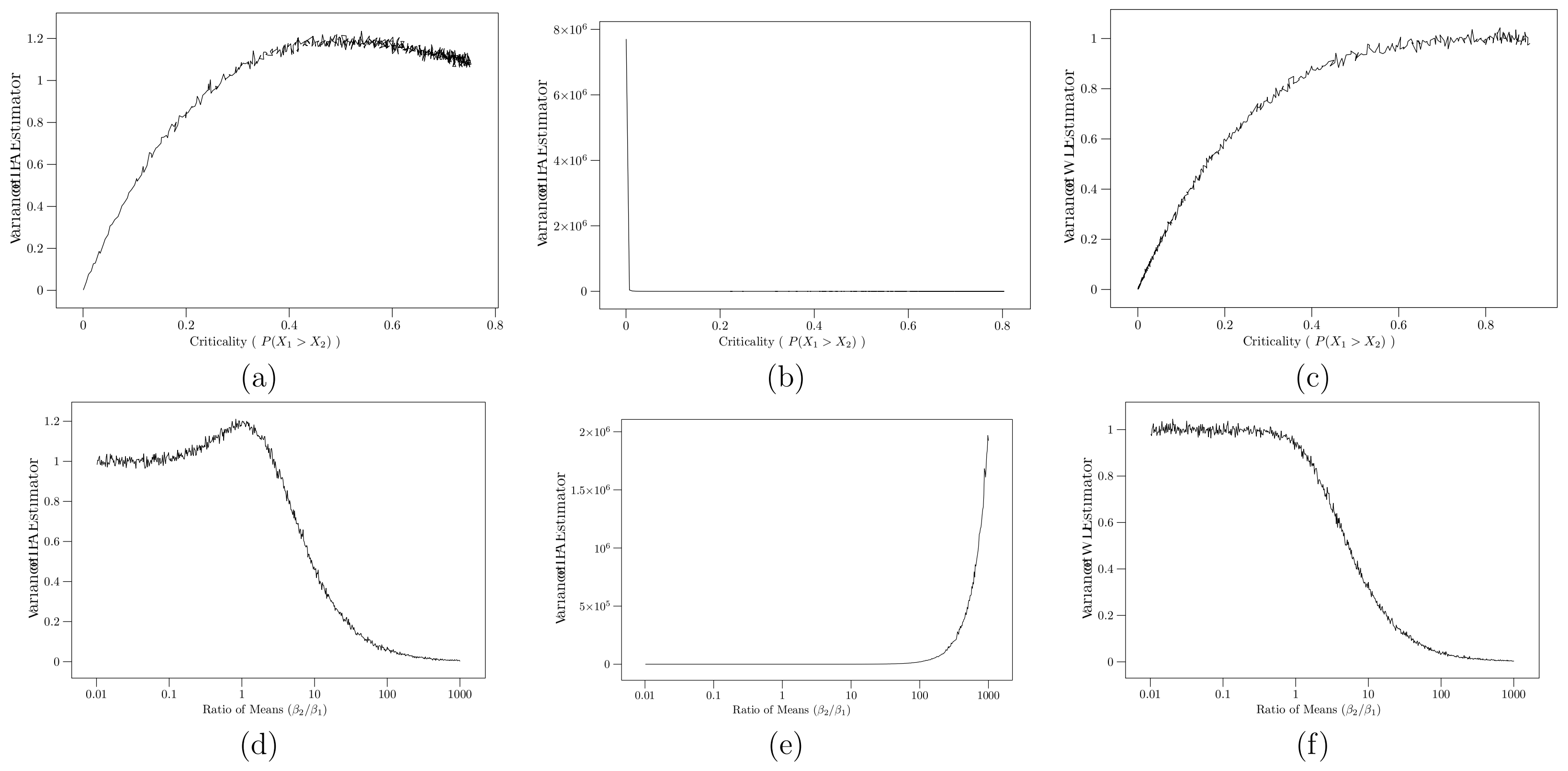


Figure 4.5: Comparison of gradient estimates variances for exponentially distributed arcs for Two parallel arcs. Plots (a)(b)(c) shows the precision dependency with the criticality. The second row plots (d)(e)(f) show the gradient variances with respect to the ratio of means β_2/β_1. (a)(d) IPA, (b)(e) SF/LR and (c)(f) WD estimators.

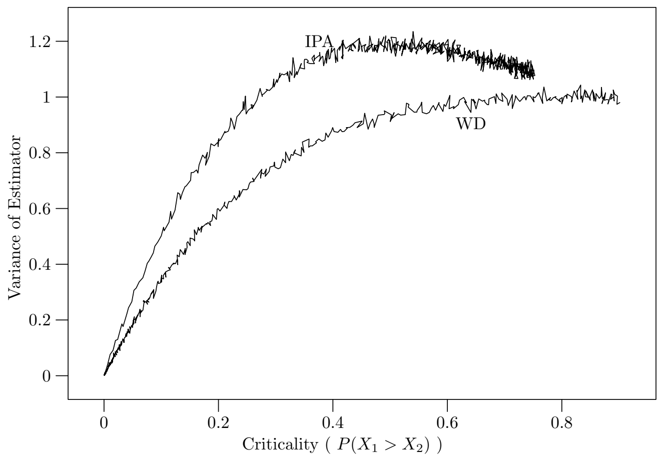


Figure 4.6: Two parallel arcs SAN, variance of IPA and WD comparison for exponentially distributed arcs .

Consider now **Gaussian** distributions and sensitivity w.r.t. the standard deviation (σ) of the first arc distribution. The MCS was performed with 30000 repetitions and the Gaussian and Weibull samples by A-R method as it is explained in [13]. Table 4.4 shows very interesting results based of networks with $\bar{X}_1 = \bar{X}_2 = \mu$ and $\mathrm{Var}(X_1) = \mathrm{Var}(X_2) = \sigma^2$.

Just like the single arc network (equations (3.24), (3.25) and (3.29)), the pure parallel network shows that the precision of the gradient estimator is independent of the parameters of the Gaussian distribution. The exception is the SF/LR estimator that depends on the ratio μ/σ.

The previous experiment was performed using the same mean in both activities. We can change the relative relation between means to observe the criticality index effect in the gradient estimation variance. Figure 4.7 was obtained varying the means to obtain criticality indices ranging from zero to one.

Table 4.4: Gradient estimators for two arc parallel Gaussian network.

Parameters of Distrib.	Mean IPA Var. IPA	Mean SF/LR Var. SF/LR	Mean WD Var. WD
$\mu = 100$ $\sigma = 100$	0.292 0.428	0.319 12.30	0.286 0.378
$\mu = 100$ $\sigma = 10$	0.292 0.428	0.454 253.7	0.286 0.378
$\mu = 100$ $\sigma = 1$	0.292 0.428	1.798 20996	0.286 0.378
$\mu = 10$ $\sigma = 1$	0.292 0.428	0.454 253.7	0.286 0.378
$\mu = 1$ $\sigma = 1$	0.292 0.428	0.319 12.30	0.286 0.378
$\mu = 0.1$ $\sigma = 1$	0.292 0.428	0.306 6.478	0.286 0.378

The same independence of the parameters than in the Gaussian case can be detected when **Uniformly** distributed arcs are used. Table 4.5 exhibit this behavior. Notice, that SF/LR estimator does not exist for uniform distribution. The Uniform distributions used in this case are of the type $U(0, \theta)$, and $\bar{X}_1 = \bar{X}_2 = \theta/2$.

Table 4.5: Gradient estimators for two arc parallel uniform dist. network.

Parameters of Distrib.	Mean IPA Var. IPA	Mean WD Var. WD
$\theta = 1$	0.331 0.138	0.334 0.056
$\theta = 2$	0.331 0.138	0.334 0.056
$\theta = 200$	0.331 0.138	0.334 0.056

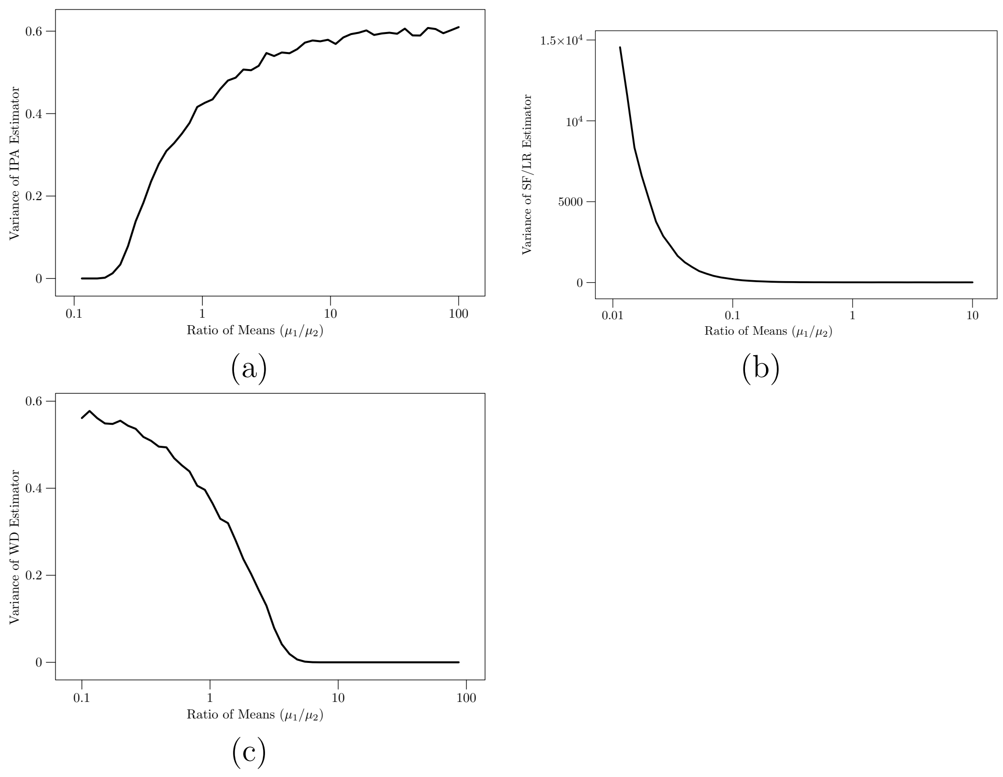
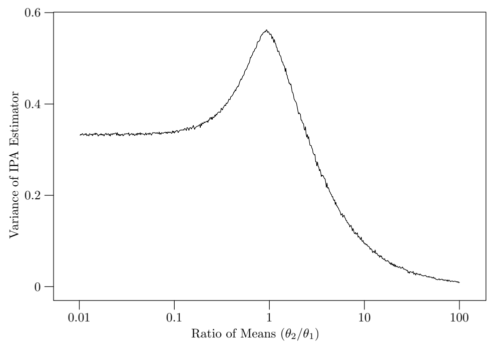
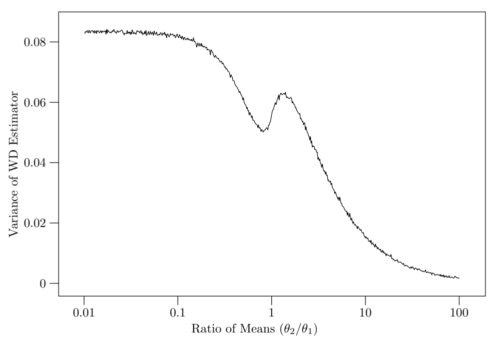


Figure 4.7: Precision of gradient estimators for parallel SAN, Gaussian distributions (a) IPA, (b) SF/LR and (c) WD

If we change the criticality index in this uniformly distributed network, then Figure 4.8 is obtained by MCS. What it is interesting about this figure is the shape of the IPA curve. Notice that the worst precision is obtained when criticality index is 50%. This same shape pattern was obtained for Figure 3.4. Again we have found similarities between $U(0,\theta)$ and $Exp(\beta)$ distribution. These similarities seem related with the fact that the sensitivities were computed w.r.t. scale parameters in both cases (IPA estimators are of the form X/θ).



(a)



(b)

Figure 4.8: Precision of gradient estimators for parallel SAN, activities distributed Unif$(0, 2\theta)$ (a) IPA and (b) WD

4.3.2 Multiple parallel arcs

The next experiment is another way to control the criticality of the first arc. It consists in adding more and more parallel arcs between the initial and final nodes as it is shown in Figure 4.9. The results are presented in Figure 4.10, which was obtained repeating the simulation and adding and extra arc every time up to 50 parallel arcs.

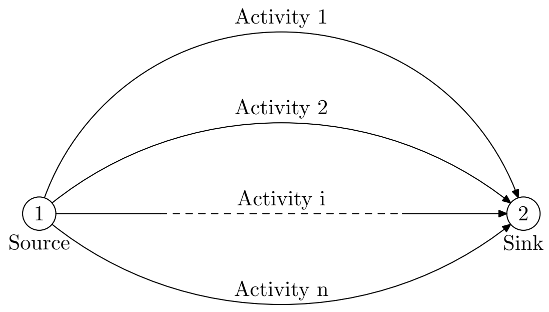
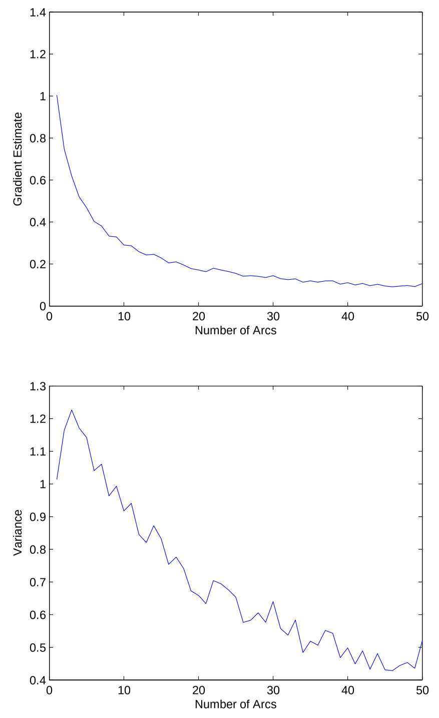


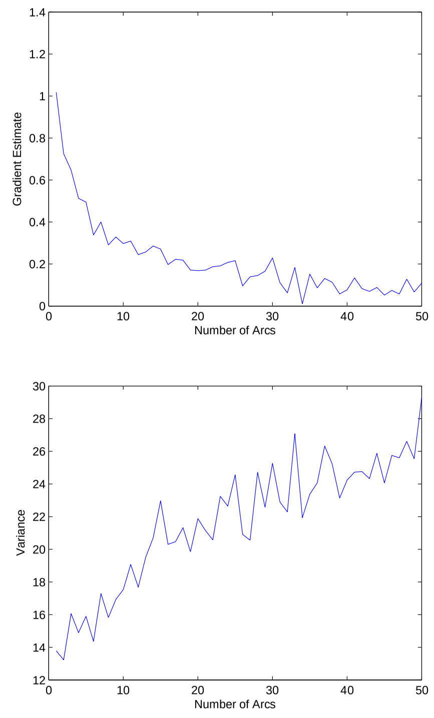
Figure 4.9: Parallel configuration of SAN

From this experiment we can conclude that both the number of arcs and the criticality have influence on the precision of the estimators. Also, we can appreciate how the form of estimator expression influence the results: a similar form makes the curves and behavior of the variance similar too. For example, uniform $\mathrm{U}(0, \theta)$ is similar to exponential $\mathrm{Exp}(\theta)$. Additionally, we can see how the scaling factor of the network does not affect the precision of the gradient estimator in every distribution tested here.

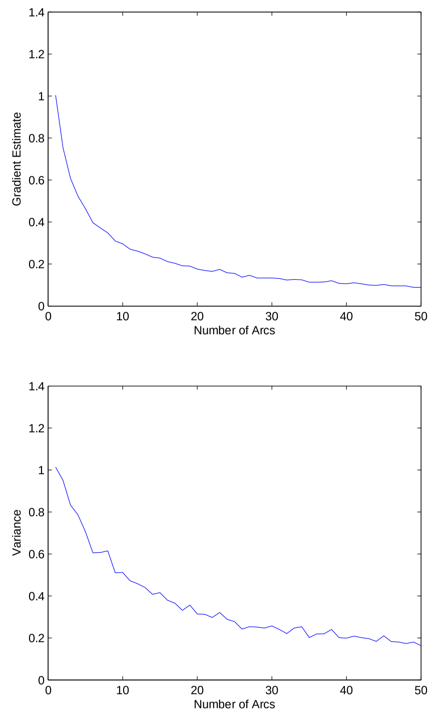
The next set of experiments were designed to try to isolate the effect of criticality in the variance of the gradient estimators.



(a)



(b)



(c)

Figure 4.10: Comparison of gradient estimates variances for parallel configuration with increasing number of arcs (a) IPA, (b) SF/LR and (c) WD estimators.

4.4 Relative weight and criticality

The relative importance of the change of a parameter with respect to rest of the network is expected to have a relevancy in the variance of the gradient estimator. From this point of view, the following network tests were used to examine the behavior of gradient estimators,first in a simple series-parallel network and then in more complex structures.

4.4.1 Series-parallel combination network

Gradient estimates were computed with respect to the first arc. The SAN has exponentially distributed arcs. Nine different combination where tested changing the relative weight of arc 1 w.r.t. the total expected path length (see Figure 4.11) and criticality index of the path containing arc 1.

Monte Carlo simulation was used to get gradient estimates by IPA, SF/LR and WD methods. Note from Table 4.6 the behavior of the SF/LR estimator: best case is obtained when the first arc has both high criticality and weight (best precision i.e. low variance is marked with a star). Conversely, IPA and WD estimator behaves better when the arc 1 has low criticality and low weight. Something similar was observed in the two parallel network analyzed in the previous chapter, however in this experiment we have added the relative weight behavior.

Plots are displayed in Figure 4.12 and show the curves relating the precision of the gradient estimates w.r.t. and the criticality for exponentially distributed arcs. Again, we can find a lot of similarities between this plots and the ones obtained for

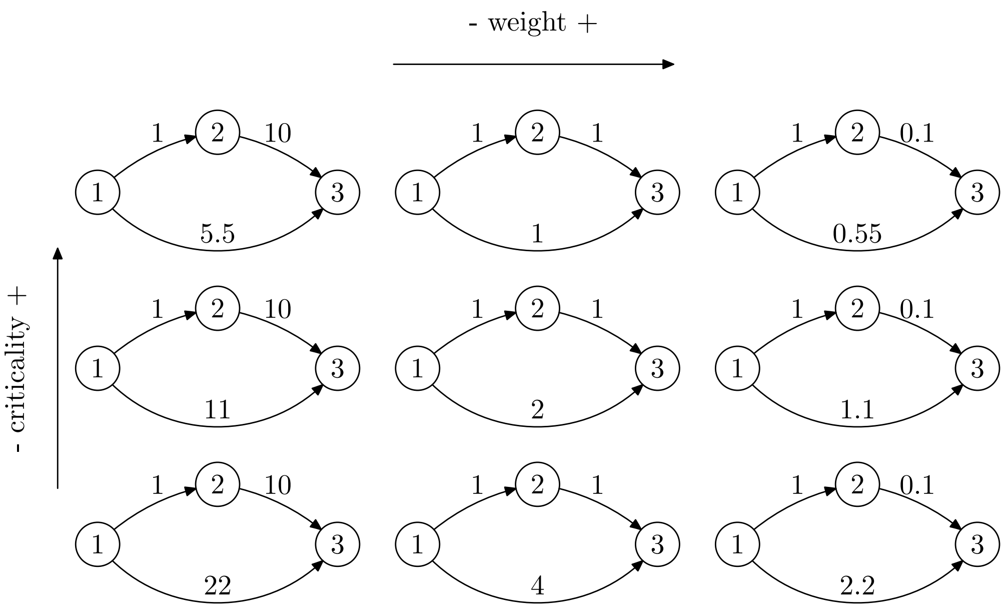


Figure 4.11: Networks used to compare criticality and relative importance of arc 1. Numerical values shown are the mean of exponentially distributed activity times.

the more simple network of two parallel arcs. It can be said that the relative weight does affect the shape of the criticality vs. variance curve, but it does not affect the behavior in general.

4.4.2 Complex SANs with constant relative weight

The next experiment was designed to discover how the rest of the network affects the precision results when criticality and relative weight is kept constant in the first activity. In other words, we are interested in compute the gradient w.r.t. the first arc, in very different network structures. But these networks have in common the criticality index and relative weight of arc 1 (activity of interest).

From results in Section 3.5.1 on page 68 (specifically Figure 3.16 in page 70), it

Table 4.6: Comparison of gradient estimators for two paths network varying criticality and relative weight.

Method	Grad.Variance
IPA	1.006
SF/LR	286.8
WD	0.940
weight=0.1, crit.=66.7%	

Method	Grad.Variance
IPA	1.128
SF/LR	21.84
WD	1.009
weight=1, crit.=66.7%	

Method	Grad.Variance
IPA	1.134
SF/LR	14.74 $\star$
WD	1.016
weight=10, crit.=66.7%	

Method	Grad.Variance
IPA	0.864
SF/LR	437.0
WD	0.786
weight=0.1, crit.=50.0%	

Method	Grad.Variance
IPA	1.161
SF/LR	23.87
WD	0.922
weight=1, crit.=50.0%	

Method	Grad.Variance
IPA	1.201
SF/LR	15.61
WD	0.964
weight=10, crit.=50.0%	

Method	Grad.Variance
IPA	0.606 $\star$
SF/LR	1139
WD	0.566 $\star$
weight=0.1, crit.=33.3%	

Method	Grad.Variance
IPA	0.906
SF/LR	37.23
WD	0.662
weight=1, crit.=33.3%	

Method	Grad.Variance
IPA	1.100
SF/LR	20.22
WD	0.800
weight=10, crit.=33.3%	

is known that for exponential networks IPA gradient estimation precision IS affected by the rest of the network which does not include the path which arc 1 belongs to. Despite this fact, for low relative weight activity times, the gradient RV tends to behave as a generalized Bernoulli RV in which the criticality is almost independent of the actual realization times. When this is the case, precision of the IPA estimation is independent of the rest of the subnetwork if the criticality of the arc of interest is kept fixed and relative weight is low. This was observed for exponential arcs in the previous chapter (see Figure 3.16).

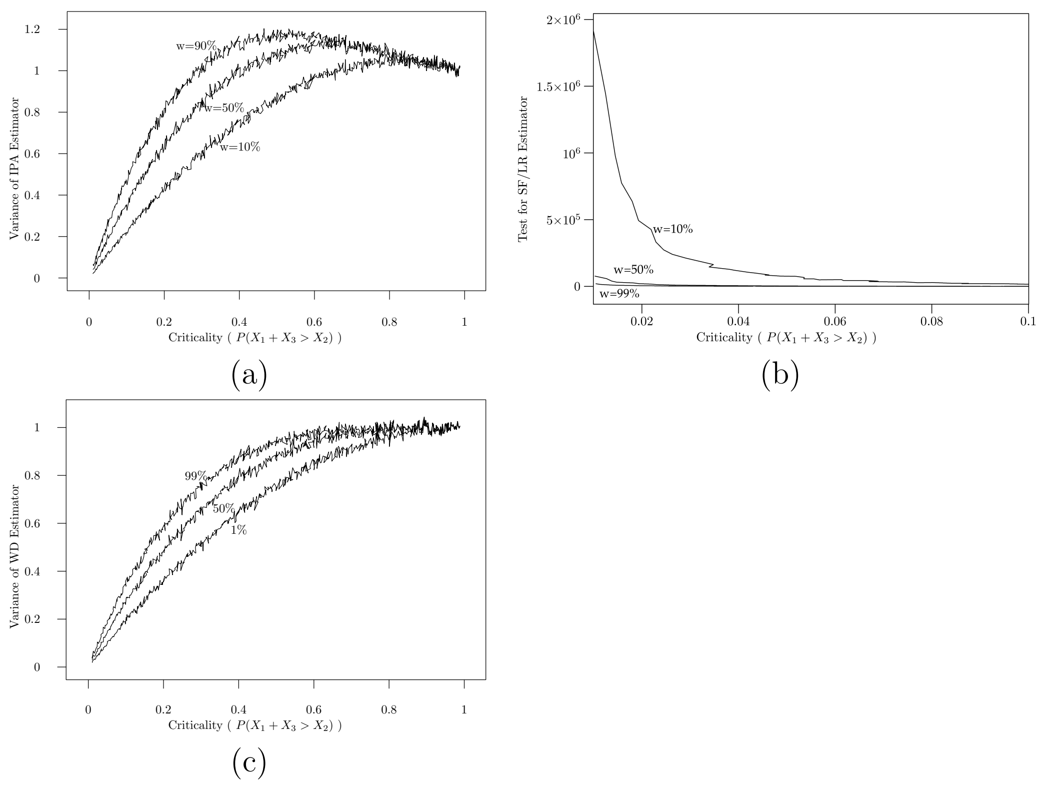


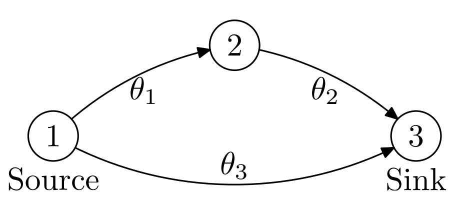
Figure 4.12: Precision of gradient estimators for Series-parallel combination SAN. (a) IPA, (b) SF/LR and (c) WD

Other distributions will be checked in this subsection. The idea is to perform verification of similar behavior of IPA gradient estimator.

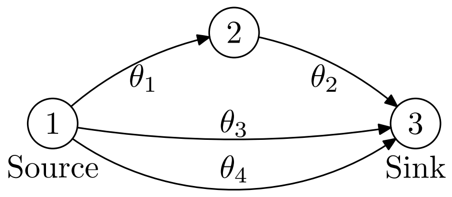
Uniformly distributed arcs, $X_i \sim U(0, 2\theta)$

Consider Uniform distributed activity times, relative weight of 50% were used for the experiment and sensitivity w.r.t. mean θ_1. Network structures to consider are the familiar series-parallel network and the two series plus two parallel network (see Figure 4.13).

Now let us repeat the simulation with relative weight $w = 1\%$, which should



(a) Series-parallel network



(b) Two series two parallel

Figure 4.13: This two networks are analyze by MCS. In each simulation relative weight and criticality index of arc 1 is kept fixed.

be a realistic value for very large networks.

Figure 4.14 confirms the intuition concerning the uniformly and exponentially distributed networks will behave likewise since they share similar derivatives formulas: X/θ.

Weibull distributed arcs, $X_i \sim Wei(\alpha, \theta)$

This experiment was included to check whether the behavior of the IPA estimator variance of exponentially distributed arcs is similar or not to the one obtained after a change in the activity times distributions. Figure 4.14 (c) and (d) show the criticality vs. variance of IPA estimator for Weibull distributed activity times.

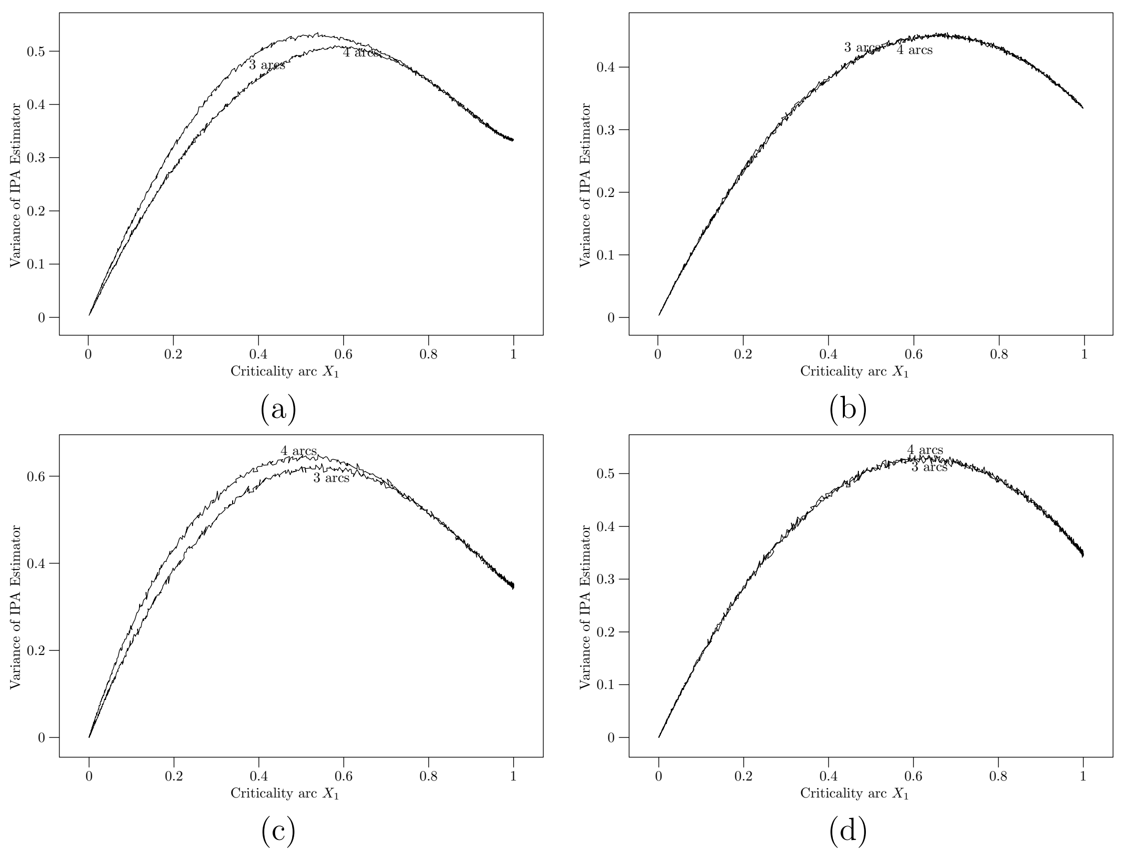


Figure 4.14: Comparison of IPA gradient estimates variances for different network structures and distributions. Uniform distributed arcs. (a)Relative weight 50%, uniform distribution. (b)Relative weight 5%, uniform distribution. (c)Relative weight 5%, Weibull distribution. (d)Relative weight 5%, Weibull distribution

4.5 Complexity coefficients

This section is intended to prove the hypothesis that more complex activity network implies more degradation in the precision of every estimator. The complexity is measured using several complexity indices found in the CPM/PERT literature. Several studies have shown how simplistic and inappropriate some of this measures are and how they fail to capture the feeling of complexity of a network. This is particularly clear in CNC measures, where it is easy to give examples of networks with different complexity but with the same CNC.

According to Elmaghraby [7], a complexity index must be defined according to the use this number is given. This way the index measures the complexity w.r.t. the operations to be applied to the graph. For example, the cyclomatic number counts the number of cycles in an undirected graph, or the number of binary decisions in a directed one, which can be a meaningful metric of complexity in the field of software testing, but it has no meaning if the graph represents a road system.

The first set of networks is very simple. It consists of networks with constant number of trees and cyclomatic number, but increasing number of nodes and arcs (see Figure 4.15).

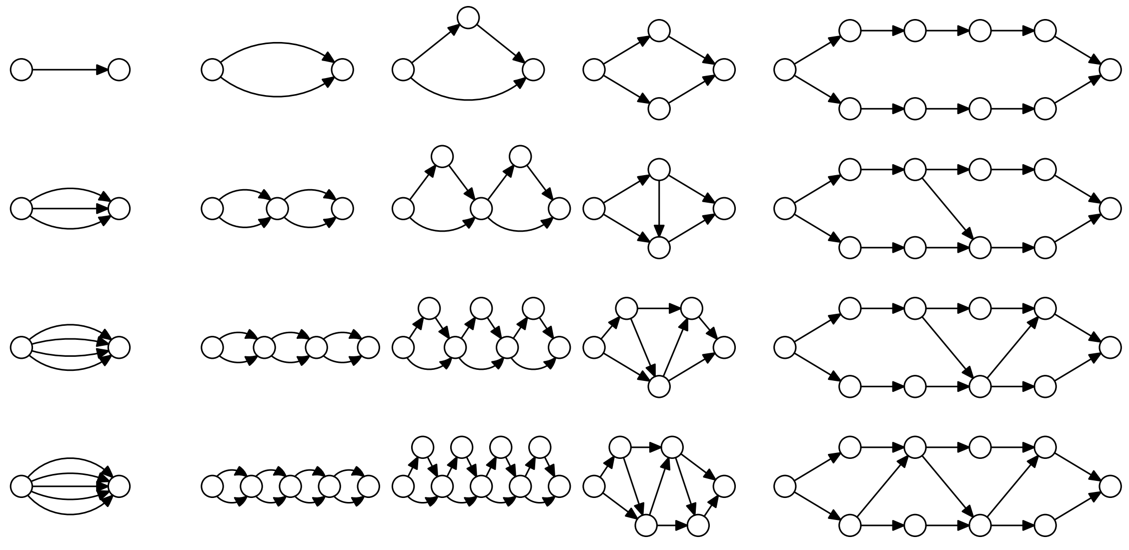


Figure 4.15: Set of small SANs for complexity measures comparison.

Then, more complex SANs were generated partially random. They maintain the layered structure presented in the original Kaimann's paper [14] and used again in the review of complexity measures by Latva-Koivisto [15]. These SANs are characterized by a layered structure with forward only connections and a skeleton substructure presented in Figure 4.16. The random part are the "inter-branch" connections represented by dashed lines. Different number of layers and nodes per layer were selected to create the SAN test set. Another parameter of constraint in the random inter-branch connections was maximum layer distance. In other words, how long the inter-branch are allowed to be.

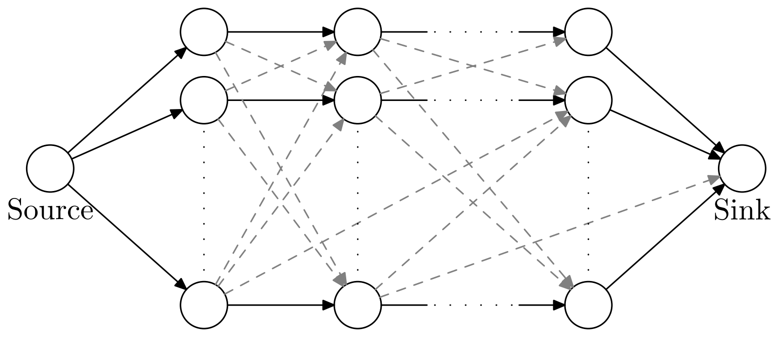
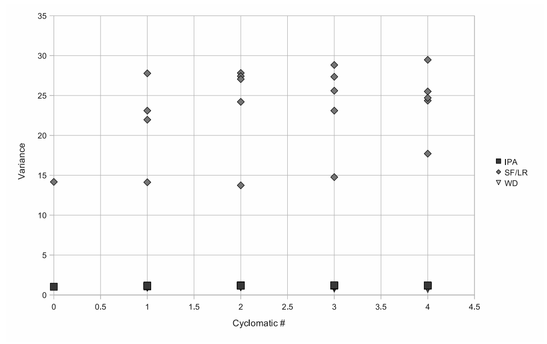


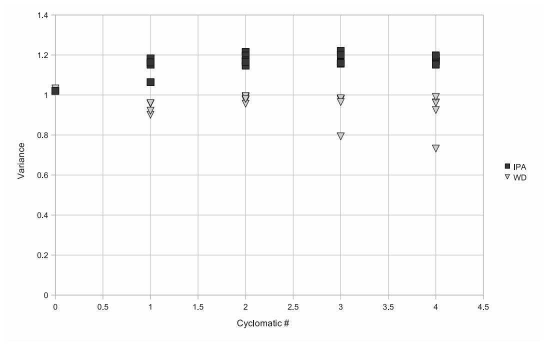
Figure 4.16: Typical structure of Stochastic Activity Networks test set. Filled lines represent fixed or skeleton activities and dashed lines are inter-branch connections added for increasing complexity.

4.5.1 Manually created set of SANs

Starting with manually created more simple network, we can notice that for a given cyclomatic number, the SF/LR estimator can have large variabilities of values. This responds to the way the network set were created: please notice in Figure 4.15 that each row features SANs with the same cyclomatic number (for example in the third row every SAN has $S = 3$). Figure 4.17 highlights the deficiencies of the



(a)

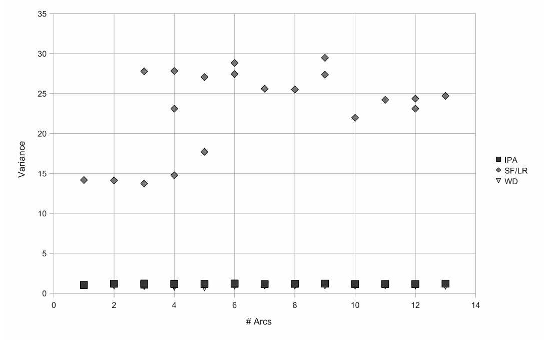


(b)

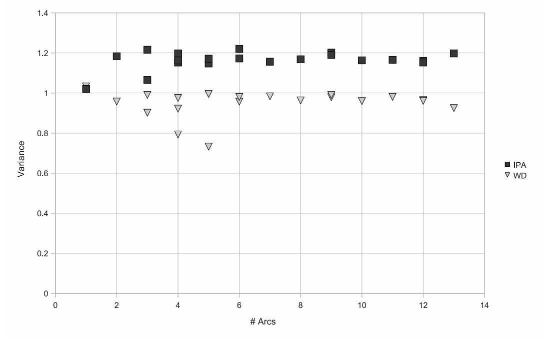
Figure 4.17: Variance of gradient estimators vs. cyclomatic number. (a)The three estimators are displayed. (b)The IPA and WD estimator are shown.

cyclomatic number for our particular purposes, namely, to explain the variability of the precision via this index.

Figure 4.18 shows the dependency of the variance w.r.t. the number of arcs. It can be observed that IPA and WD estimators present some variability but it



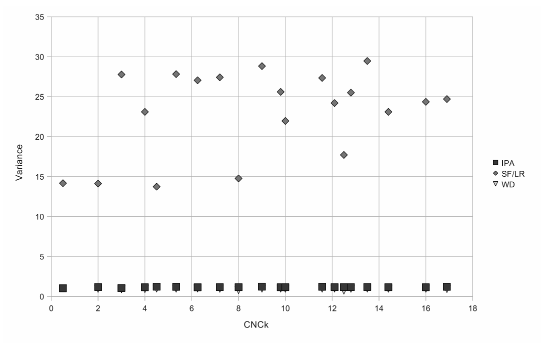
(a)



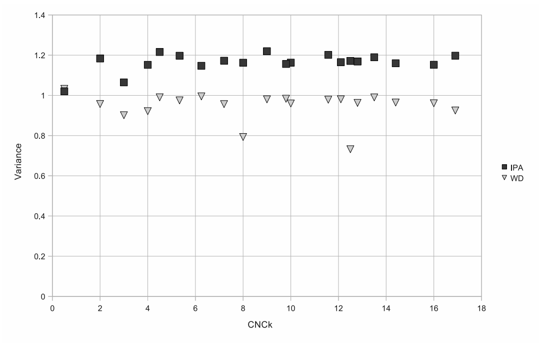
(b)

Figure 4.18: Variance of gradient estimators vs. number of arcs. (a)The three estimators are displayed. (b)The IPA and WD estimator are shown.

cannot be due to the number of activities.



(a)



(b)

Figure 4.19: Variance of gradient estimators vs. CNCk. (a)The three estimators are displayed. (b)The IPA and WD estimator are shown.

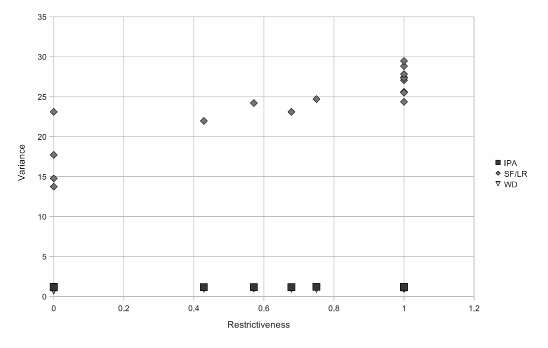
4.5.2 Complex semi-random SANs

For this section we generate networks, limiting certain features. In this instance, we first restrict the networks to be complete skeletons, the number of nodes per layer and the number of layers were changed. Figure 4.23 shows a couple of SANs that belong to the first group.

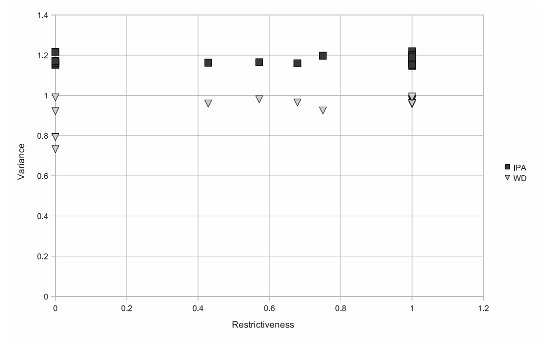
Second, included in the set are full-connected between layers networks. The arcs among layers are allowed to connect contiguous layers only. See Figure 4.24 for examples of this kind of networks. Then networks are form of arcs that are allow to jump and move forward without passing for every layer. For jump of length 2 and 3 see Figure 4.25.

Finally, inter-branch connections are selected according to a Bernoulli RV. See Figure 4.26.

Next Figures 4.27 to 4.32 show the variance of gradient estimators w.r.t. the complexity measures. In addition, variances were also plotted vs. the index of



(a)



(b)

Figure 4.20: Variance of gradient estimators vs. restrictiveness estimator (a)The three estimators are displayed. (b)The IPA and WD estimator are shown.

dispersion (Var/Mean), to try to compensate for the effect of the size of the estimate ($\mathrm{E}\left[Y(\mathbf{X})\right]$). Those plots are displayed in Figures 4.33 to 4.38.

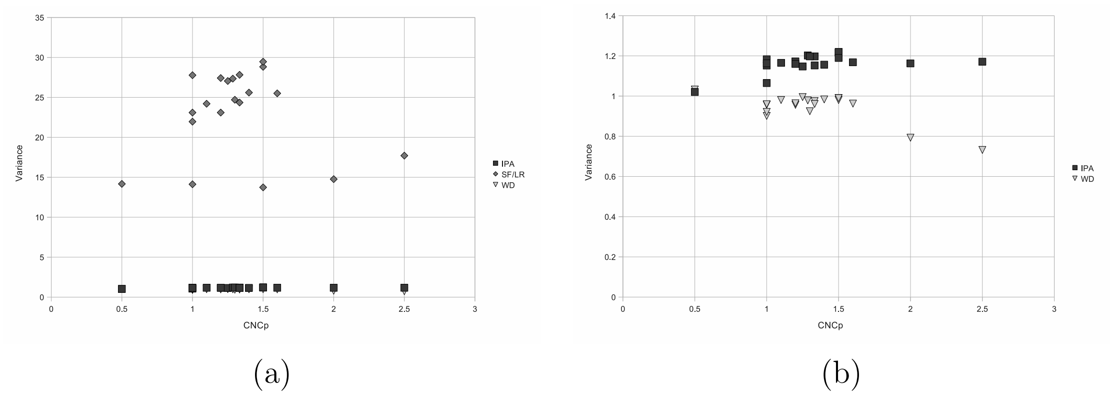


Figure 4.21: Variance of gradient estimators vs. CNCp. (a)The three estimators are displayed. (b)The IPA and WD estimator are shown.

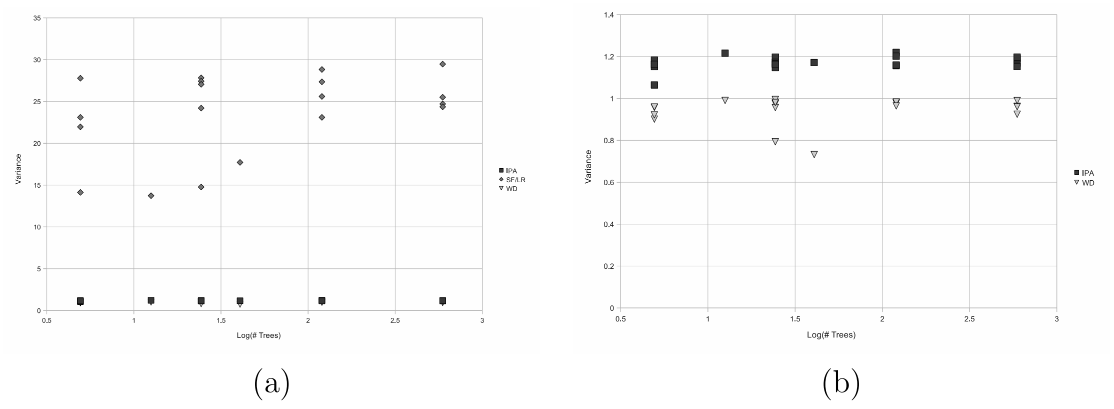
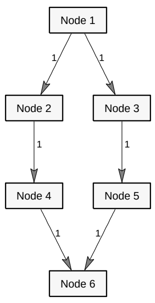


Figure 4.22: Variance of gradient estimators vs. Log(T). (a)The three estimators are displayed. (b)The IPA and WD estimator are shown.


(a)

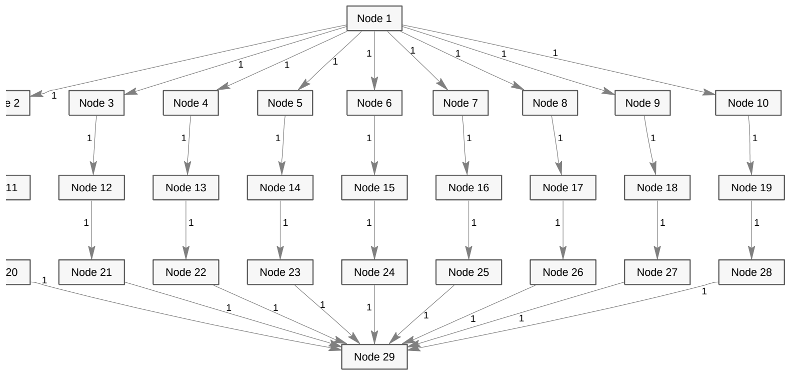
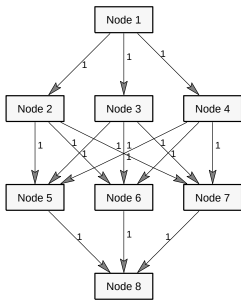

(b)

Figure 4.23: Network examples for complexity index experiments. (a)Skeletal 2x2. (b)Skeletal 3x9.


(a)

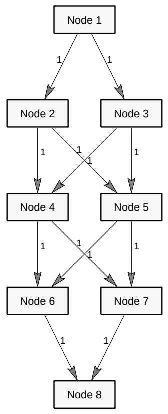

(b)

Figure 4.24: Network examples for complexity index experiments. (a)Full 2x3. (b)Full 3x2.

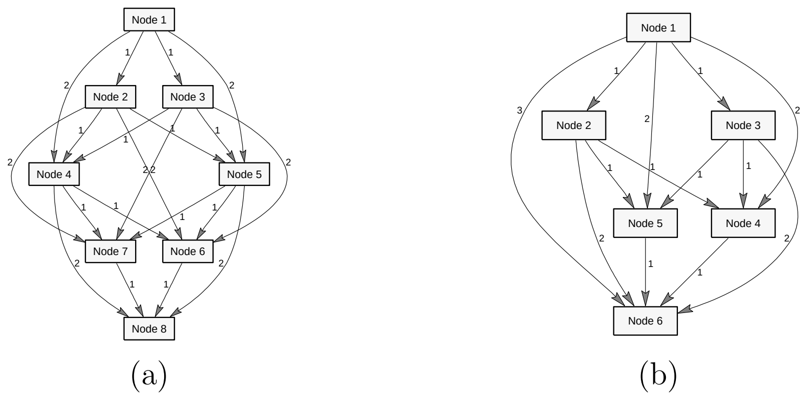


Figure 4.25: Full network example with jumps of length 2 and 3.

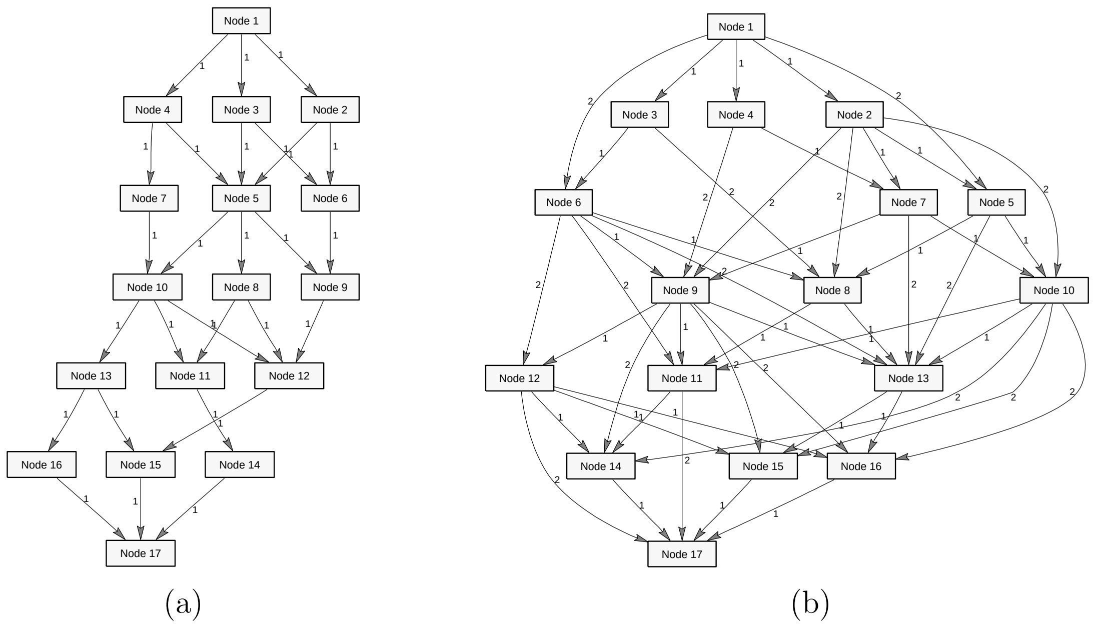
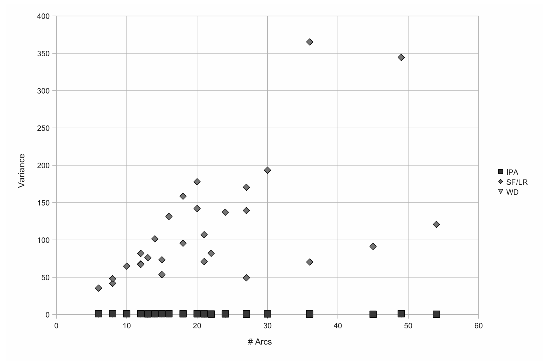
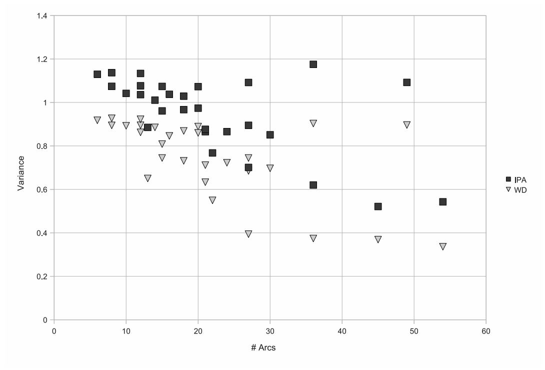


Figure 4.26: Network example with random inter-brach arcs.

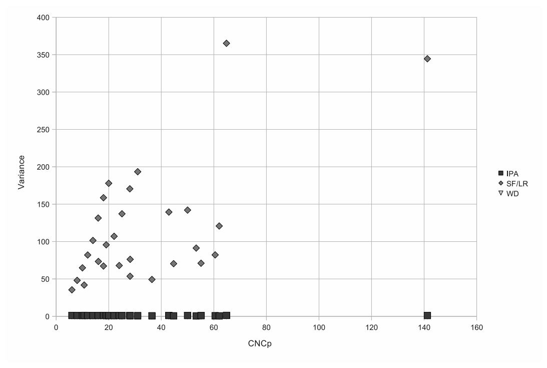


(a)

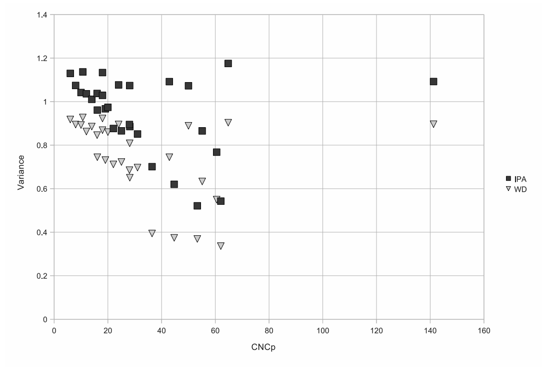


(b)

Figure 4.27: Variance of gradient estimators vs. number of arcs(N). (a)The three estimators are displayed. (b)The IPA and WD estimator are shown.

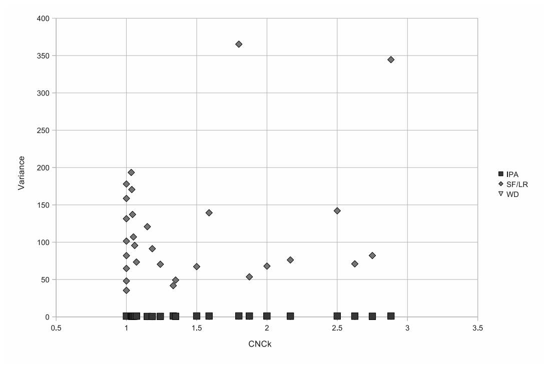


(a)

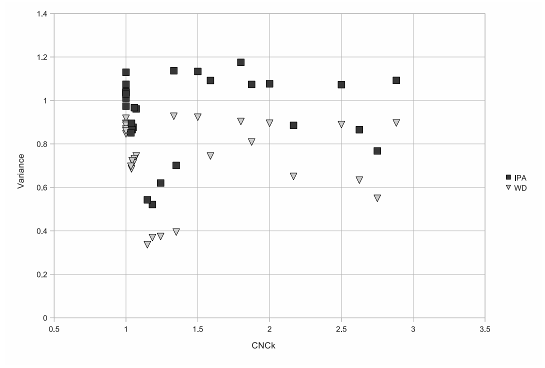


(b)

Figure 4.28: Variance of gradient estimators vs. CNCp. (a)The three estimators are displayed. (b)The IPA and WD estimator are shown.

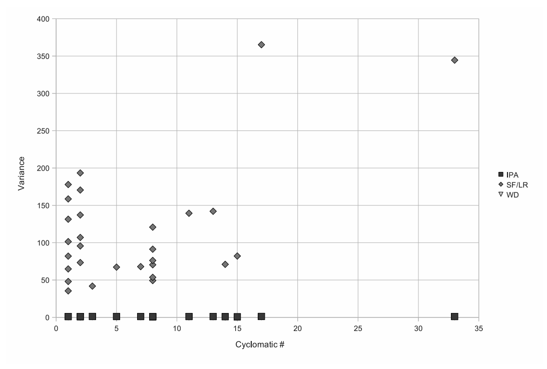


(a)

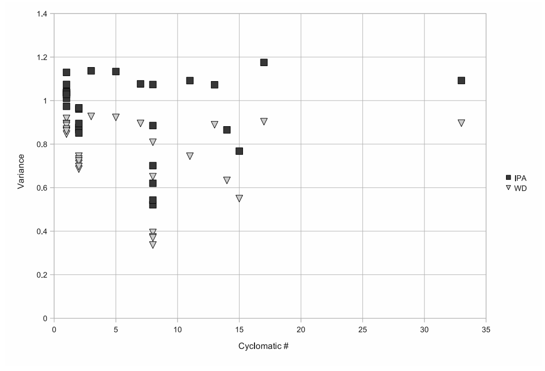


(b)

Figure 4.29: Variance of gradient estimators vs. CNCk. (a)The three estimators are displayed. (b)The IPA and WD estimator are shown.

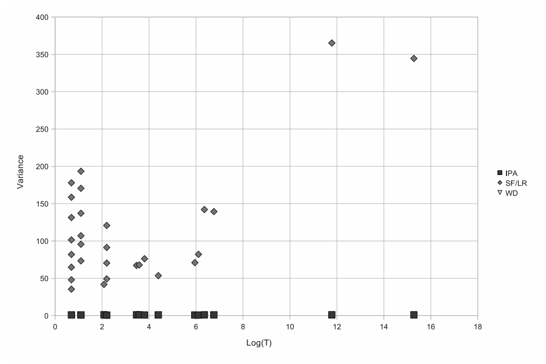


(a)

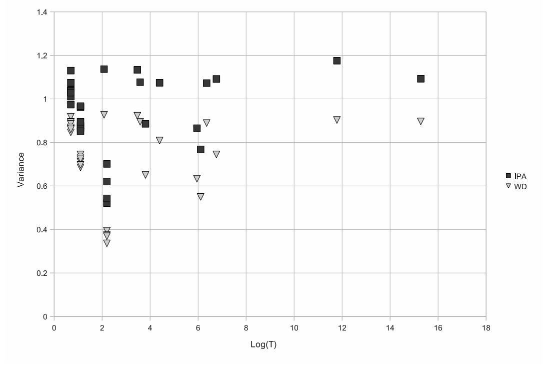


(b)

Figure 4.30: Variance of gradient estimators vs. cyclomatic number (S). (a)The three estimators are displayed. (b)The IPA and WD estimator are shown.

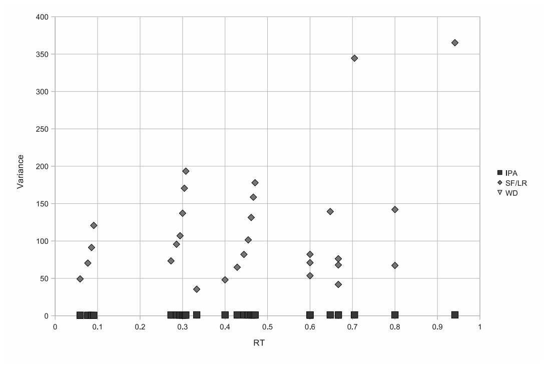


(a)

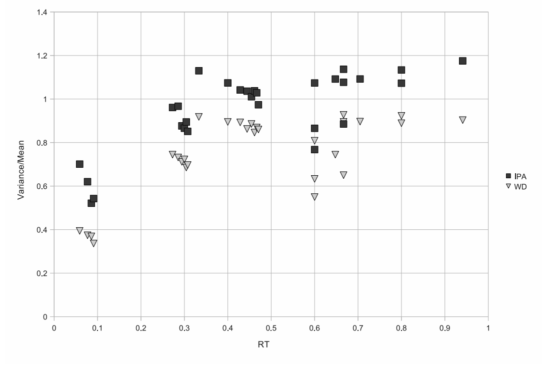


(b)

Figure 4.31: Variance of gradient estimators vs. Log(# Trees). (a)The three estimators are displayed. (b)The IPA and WD estimator are shown.

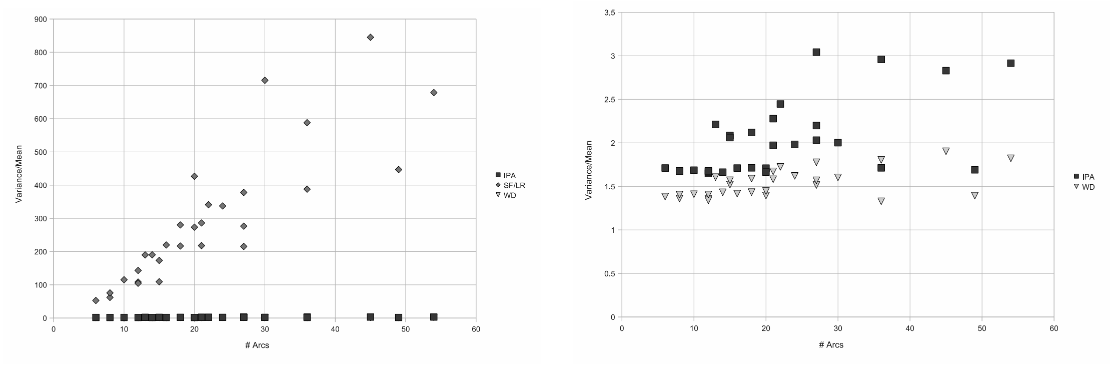


(a)



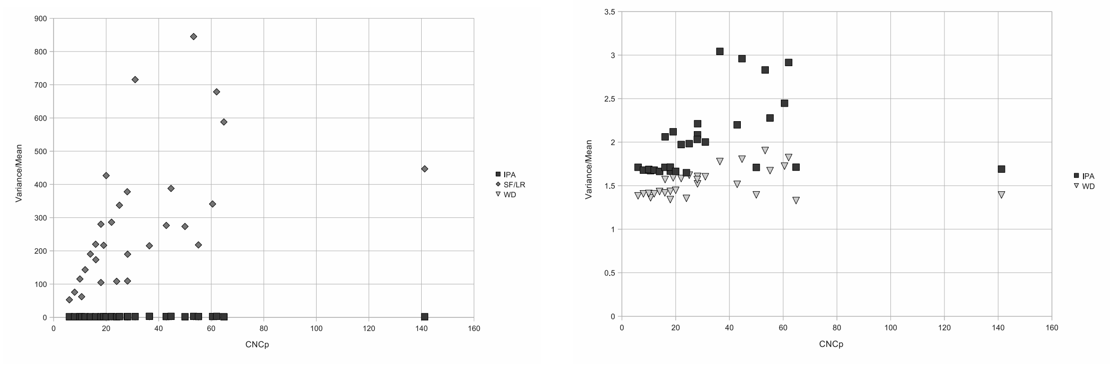
(b)

Figure 4.32: Variance of gradient estimators vs. restrictiveness estimator (RT). (a)The three estimators are displayed. (b)The IPA and WD estimator are shown.



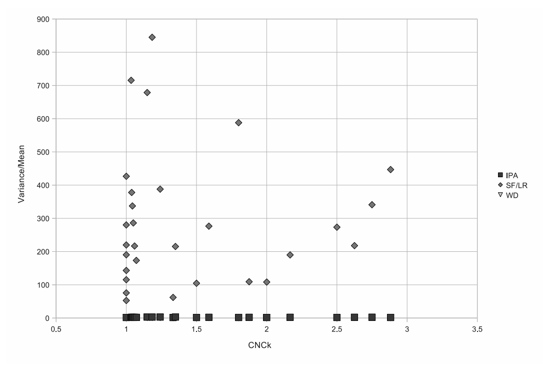
(a) (b)

Figure 4.33: Coefficient of dispersion of gradient estimators vs. number of arcs (N). (a)The three estimators are displayed. (b)The IPA and WD estimator are shown.

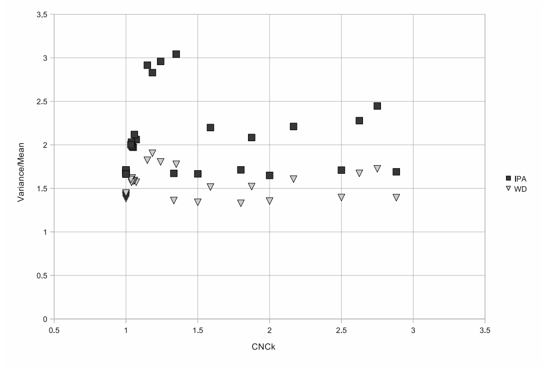


(a) (b)

Figure 4.34: Coefficient of dispersion of gradient estimators vs. CNCp. (a)The three estimators are displayed. (b)The IPA and WD estimator are shown.

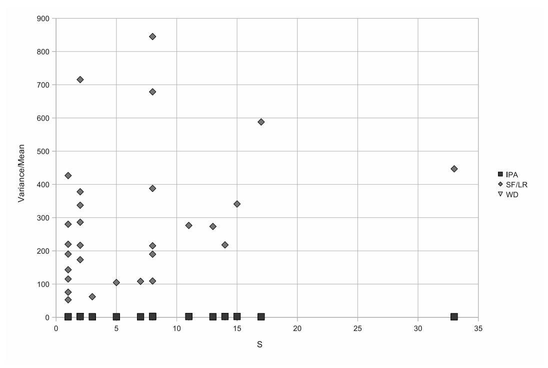


(a)

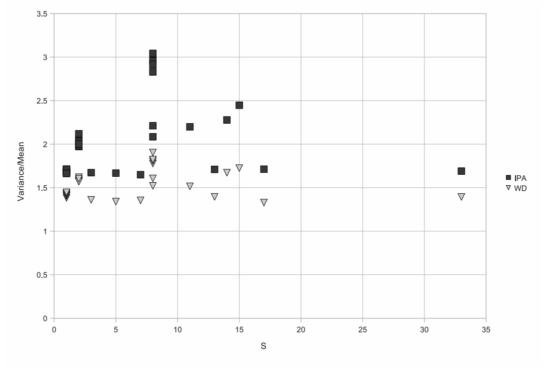


(b)

Figure 4.35: Coefficient of dispersion of gradient estimators vs. CNCk. (a)The three estimators are displayed. (b)The IPA and WD estimator are shown.

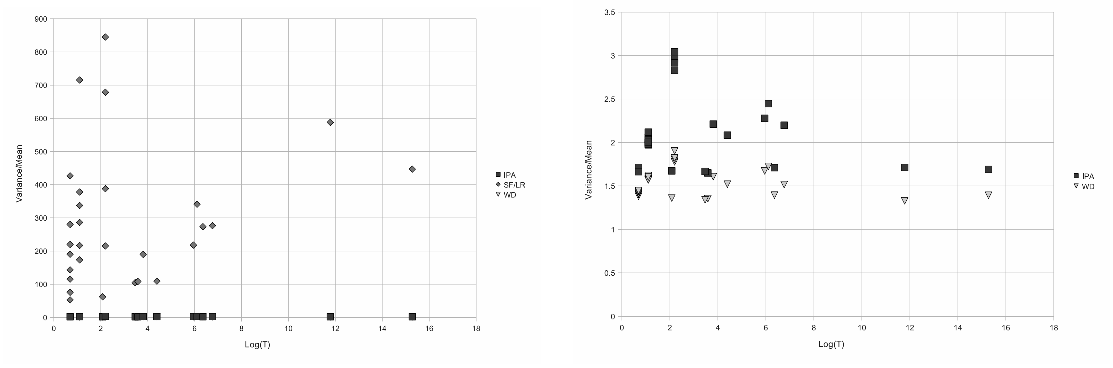


(a)



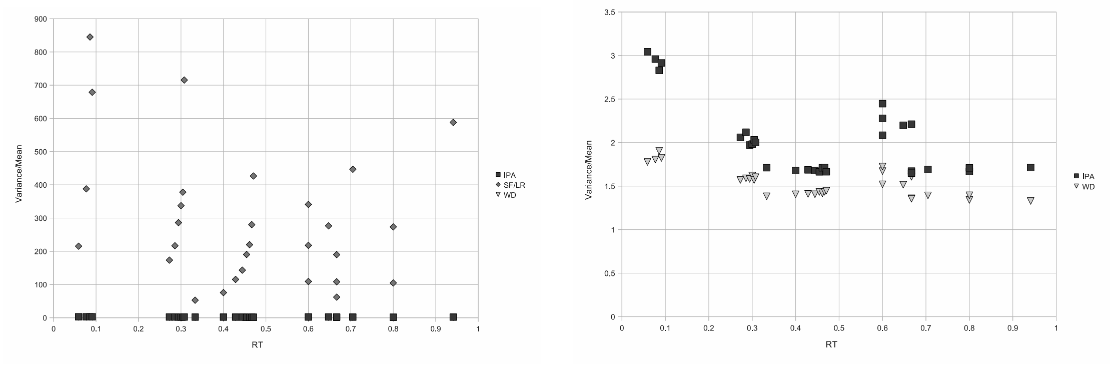
(b)

Figure 4.36: Coefficient of dispersion of gradient estimators vs. cyclomatic number (S). (a)The three estimators are displayed. (b)The IPA and WD estimator are shown.



(a) (b)

Figure 4.37: Coefficient of dispersion of gradient estimators vs. Log(# Trees). (a)The three estimators are displayed. (b)The IPA and WD estimator are shown.



(a) (b)

Figure 4.38: Coefficient of dispersion of gradient estimators vs. restrictiveness estimator (RT). (a)The three estimators are displayed. (b)The IPA and WD estimator are shown.

Chapter 5

Discussion and Conclusions

5.1 Discussion

It is evident throughout this work that the gradient estimators based on weak derivatives (WD) presented in this thesis (with CRN) outperform IPA estimation in nearly every SAN in terms of precision. At the same time, WD and IPA estimators display lower variance than crude SF/LR methods by a very large margin. This phenomenon is true for both the simplest network (single arc with exponential arc times) where IPA and WD presented variance of 1 vs. SF/LR variance of 13, and for the very intricate networks discussed in chapter 4, where charts need to be displayed with two different Y-axis scales to present the results. WD gradient estimation is not free of disadvantages. Algorithm implementation is more difficult, it consumes more simulation time and it is decomposition dependent (which is not unique), but if precision is the main issue in a particular application, WD gradient estimation is the best option according to the evidence presented here.

Looking in detail at the closed-form expressions obtained in chapter 3, it can be observed (for example, in the single path, pure series SAN) that IPA estimator variance is mostly due to the variance of activity time of the arc of interest. In contrast, SF/LR estimator variance consists of a combination of the variances and covariances of the whole SAN. Specifically, the main contribution to the variance is the high degree terms of the input activitity times. High degree terms come from the multiplication of critical path (a sum of activity times) and the input activity time of interest. It seems plausible that some specific dependency between arcs may positively affect the precision of this estimator if the inter-activity time covariances are negative.

These high degree terms do not appear in the WD estimation. This fact plus a careful generation of the variates to achieve positive covariance between phantom activity times makes this method the most precise. In the single arc case, as much as two thirds of the original variance was reduced by using common random numbers.

An interesting counterintuitive result that was discovered is that SF/LR estimation in Gaussian distributed SANs can be improved by increasing the variance of individual activity times. In contrast, IPA and WD estimators do not show dependency with respect to the distributional parameters when activity times follow Gaussian distributions.

Another important behavior is the dependence of SF/LR estimator variance on the size of the network. The importance of this effect on the variance can be observed in the experiment with multiple arcs in series. In that case, the variance of the estimator quickly increases as the number of arcs increases (see Figure 3.2 on page 43).

In more complex networks, similar behavior can be observed. For example, notice that the number of arcs is one the best indices capturing the variation of the SF/LR estimator variance (Figure 4.33). This is particularly true when the index of dispersion (σ^2/μ) is considerer to assess the precision. For the SF/LR estimator, the number of arcs is even better than more sophisticated complexity indices at capturing the differences in accuracy.

On the other hand, two techniques (IPA and WD) do not show a clear link to the number of arcs or any other complexity index. The better indices are the Pascoe CNC and the restrictiveness index (RT). The restrictiveness index shows some prospective patterns which could be a subject of future research about this complexity measure. But in general, the complexity indices did not appear to capture relevant information from the structure of the SAN related to the precision of the gradient estimators.

The IPA estimators are mainly affected by the number of times that the arc of interest is part of the critical path. In other words, IPA should be affected by the criticality index ($P(Arc \in P^{\star})$). In order to uncover this relationship, mathematical expressions were deduced for the case of exponential distribution. The curve obtained for the IPA estimator variance equation is very interesting since the criticality vs. variance curve is not monotone and it presents a maximum at 50% criticality.

The WD variance curve for the estimator used in this thesis is more predictable; it is a smooth and monotone curve from 0 at zero criticality to 1 when the longest path is always the first arc. The SF/LR curve shows a totally different shape, since precision degrades when criticality decreases and approaches zero.

From the numerical experiments, we can conclude that the IPA estimation variance **does** depend on the rest of the network structure not included in the path of the arc of interest. When the relative weight of the arc of interest is low, the IPA estimator can be approximated by a generalized Bernoulli mixed random variable, in which the Bernoulli RV and the continuous RV are independent.

The entropy of the SAN of two parallel arcs and the tree arcs series-parallel were computed in a "brute force" fashion, which did not result in a convenient closed-form expression. It is difficult to recognize any characteristic parameter of the network in such a convoluted expression (equation (3.123)). The original idea was to use the closed-form expression of the entropy of the simple network to obtain an understanding of how parameters of the SANs should appear in the entropy expression. This way it would be possible to propose a scheme to assign probability mass to the nodes or paths of the SAN and define an information theoretical complexity index.

5.2 Conclusions

The most important conclusions of this study are:

- The WD estimator with CRN used in this thesis outperforms IPA and SF/LR method in almost every network, but it involves an additional computational burden, which increases when the parameter of interest is in many arcs.
- SF/LR estimator performed the worst in every topology considered.
- SF/LR estimator variance grows when the network grows.
- Criticality index is the main aspect that determines the variance of the gradient estimator in the case of IPA.

- Common complexity measures used in the CPM/PERT literature for SANs were not able to capture the characteristics of gradient estimation variance.

5.3 Future research

The clustering observed in the charts of large network in Chapter 4 (complex networks section) suggests that some underlying descriptor of the SAN should account for the variation across family of networks. This is most evident in the restrictiveness estimator (RT) chart. Further research is needed to find such a descriptor.

Similarly, an information-theoretic index or descriptor could be defined to accomplish the objective of capturing the information embedded in the network in terms useful to the CPM/PERT field of study (for example, in the effort needed to find the critical path or the precision of a gradient estimator).

Future research could also involve an investigation of the influence of non-independent arcs in the variance of the gradient estimators. It might be possible to reduce variance by smart coupling of the arcs or through other variance reduction technique, such as importance sampling.

Appendix A

Code

A.1 MATLAB Code

A.1.1 IPA exp(θ)

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Title:    IPA exponential
% Date:     02/10/2011
% By:       Mauricio Manterola
% Desc:
% Monte Carlo Simulation of Stoch. Activity Networks in Matlab
% Compute IPA Gradient Estimation of a SAN wrt a variation
% in the mean of the Exponential distributions.
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%Init Random Generator
s1=RandStream('mt19937ar');
RandStream.setDefaultStream(s1);
s2=RandStream('mt19937ar');
%reset(s);

%Init some constants and arrays
NumRep =        20000;              %Number of repetitions
LongPathTime =  zeros(NumRep,1);    %Crit.Path storage for ea.rep.
Gradient =      zeros(NumRep,1);    %Grad. storage for each rep.
Contador =      0;

for Rep=1:NumRep
    %Generate random arc times
    UnifDist=rand(NumArcs,1);               %Generate unif. distr. variate
    ArcTimes=-MeanArcTimes.*log(UnifDist);  %Inverse transform method for exp

    %Overwrite determistic arcs
    ArcTimes=(DetermMask.*(DetermMask~=MagicNumber)+(ArcTimes.*(DetermMask==↩
        MagicNumber)));

    %Find longest path
    [dist, path, pred]=graphshortestpath(AN, SrcNode, DstNode, 'Directed', 'true'↩
        , 'Method', 'Acyclic','Weights', -ArcTimes);

    %Store time longest path
    LongPathTime(Rep)=-dist;

    %GRADIENT COMPUTATION
     %%%%%%%% %%%%%%%%%%%
    %Find if arcs SensiArcs are in the longest path
    for i=1:NumSensiArcs
        Arcindex=SensiArc(i);
        Check1=find(path==Pairs(Arcindex,1));
        Check2=find(path==Pairs(Arcindex,2));
        if(Check2-Check1==1)    %Check if arc_i is in the crit.path
            Xi=ArcTimes(Arcindex);
            XiMean=MeanArcTimes(Arcindex);
            Grad=Xi/XiMean;     %Gradient estimator
            Gradient(Rep)=Gradient(Rep)+Grad;
            Contador=Contador+1; %Counter for Crit.Index
        end;
    end;
end;

%Display results and compute results
Aver=mean(LongPathTime)     %Expected
Vari=var(LongPathTime)      %Variance
AverGrad=mean(Gradient)     %Expected sensitivity
VariGrad=var(Gradient)      %Variance of sensitivity
%hist(Gradient, 40);        %Histogram
HalfLength=1.960*sqrt(VariGrad/NumRep); %Compute H.F.
IntervBegin=AverGrad-HalfLength         %CI for sensitivity
IntervBegin=AverGrad+HalfLength         %CI for sensitivity
Criticality=Contador/NumRep             %Criticality index
```

Listing A.1: ipaexp.m IPA estimator for exponential distribution

A.1.2 SF/LR N(μ, θ)

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Title:     SF/LR Gaussian
% Date:      03/15/2011
% By:        Mauricio Manterola
% Desc:
% Monte Carlo Simulation of Stoch. Activity Networks in Matlab
% Compute SF/LR Gradient Estimation of a SAN wrt a variation
% in the Std. deviation of Gaussian distributions.
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%Init Random Generator
s=RandStream('mt19937ar');
RandStream.setDefaultStream(s);
reset(s);

%Init some constants
NumRep =        35000;            %Number of repetitions
LongPathTime =  zeros(NumRep,1); %Crit.Path storage for ea.rep.
Gradient =      zeros(NumRep,1); %Grad. storage for each rep.

for Rep=1:NumRep
     %Generate random arc times (one at a tie for A-R Method)
    for j=1:NumArcs
        MeanValue=MeanArcTimes(j);
        MwlNormGenerator;
        ArcTimes(j,1)=NormalVar;
    end;

    %Overwrite determistic arcs
    ArcTimes=(DetermMask.*(DetermMask~=MagicNumber)+(ArcTimes.*(DetermMask==↩
        MagicNumber)));

    %Find longest path
    [dist, path, pred]=graphshortestpath(AN, SrcNode, DstNode, 'Directed', 'true'↩
        , 'Method', 'Acyclic','Weights', -ArcTimes);

    %Store time longest path
    LongPathTime(Rep)=-dist;

    %GRADIENT COMPUTATION
     %%%%%%%% %%%%%%%%%%%
    %Not need to find if arcs SensiArcs are in the longest path
    for i=1:NumSensiArcs
        Arcindex=SensiArc(i);
        Xi=ArcTimes(Arcindex);
        XiMean=MeanArcTimes(Arcindex);
        Zetai=(Xi-XiMean)/StdDev;
        Grad=((Zetai*Zetai)-1)/StdDev;
        Gradient(Rep)=Gradient(Rep)+Grad;
    end;
    Gradient(Rep)=Gradient(Rep)*LongPathTime(Rep);
end;

%Display results and compute results
Aver=mean(LongPathTime)     %Expected
Vari=var(LongPathTime)      %Variance
AverGrad=mean(Gradient)     %Expected sensitivity
VariGrad=var(Gradient)      %Variance of sensitivity
%hist(Gradient, 40);        %Histogram
HalfLength=1.960*sqrt(VariGrad/NumRep); %Compute H.F.
IntervBegin=AverGrad-HalfLength         %CI for sensitivity
IntervBegin=AverGrad+HalfLength         %CI for sensitivity
Criticality=Contador/NumRep             %Criticality index
```

Listing A.2: sflrgauss.m SF/LR estimator for Gaussian distribution

A.1.3 WD exp(θ)

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Title:    WD exponential
% Date:     02/10/2011
% By:       Mauricio Manterola
% Desc:
% Monte Carlo Simulation of Stoch. Activity Networks in Matlab
% Compute WD Gradient Estimation of a SAN wrt a variation
% in the mean of the Exponential distributions.
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%Init Random Generator
s1=RandStream('mt19937ar');
s2=RandStream('mt19937ar');

%RandStream.setDefaultStream(s);
reset(s1);
reset(s2);

%Pedestrian way to move forward 2nd stream (1191 is a magic no.)
for jj=1:1191
    dummy=rand(s2,1,1);
end;

%Init some constants
NumRep =        20000;                %Number of repetitions
LongPathTime =  zeros(NumRep,1);      %Crit.Path storage for ea.rep.
Gradient =      zeros(NumRep,1);      %Grad. storage for each rep.

%Set aditional variables for Weak Derivatives
ArcTimesWd =     zeros(NumArcs,NumSensiArcs); %Arc times from mod'ed distr.
distWd =         zeros(NumSensiArcs,1); %Array 4 longest path times of mod. distr.

for Rep=1:NumRep
    %Generate random arc times
    UnifDist=rand(s1,NumArcs,1);             %Generate unif. distr. variate
    ArcTimes=-MeanArcTimes.*log(UnifDist);  %Inverse transform method for exp
    %Generate random arc times for Gradient Estimation
    for i=1:NumSensiArcs
        ArcTimesWd(:,i)=ArcTimes;                %First just copy same times ↩
            generated previously
        UnifDist2=rand(s2,1,1);                  %Generate second unif. ↩
            distributed variate
        ArcIndex=SensiArc(i);                    %Get arc index to be modified (↩
            sensiarc)
        ModArcTime=-MeanArcTimes(ArcIndex)*log(UnifDist(ArcIndex)*UnifDist2); %↩
            Compute modified random "variate" with CRN
        ArcTimesWd(ArcIndex,i)=ModArcTime;       %Store Arc time modified in the ↩
            correct position in the matrix
    end;

    %Overwrite Deterministic Arc times (useful for zero duration arcs)
    ArcTimes=(DetermMask.*(DetermMask~=MagicNumber)+(ArcTimes.*(DetermMask==↩
        MagicNumber)));
    for i=1:NumSensiArcs
        ArcTimesWd(:,i)=(DetermMask.*(DetermMask~=MagicNumber)+(ArcTimesWd(:,i)↩
            .*(DetermMask==MagicNumber)));
    end;

    %Find longest path
    [dist, path, pred]=graphshortestpath(AN, SrcNode, DstNode, 'Directed', 'true'↩
        , 'Method', 'Acyclic','Weights', -ArcTimes);


    %Find longest path with modified distributions
    for i=1:NumSensiArcs
        [distWd(i), pathWd, pred]=graphshortestpath(AN, SrcNode, DstNode, '↩
            Directed', 'true', 'Method', 'Acyclic','Weights', -ArcTimesWd(:,i));
    end;
    distWd=-distWd;

    %Store time longest path
    LongPathTime(Rep)=-dist;

    %GRADIENT COMPUTATION
    %%%%%%%%% %%%%%%%%%%%
    for i=1:NumSensiArcs
        ArcIndex=SensiArc(i);
        XiMean=MeanArcTimes(ArcIndex);
        Grad=(distWd(i)-LongPathTime(Rep))/XiMean;
        Gradient(Rep)=Gradient(Rep)+Grad;
    end;
end;


%Display results and compute results
Aver=mean(LongPathTime)            %Expected
Vari=var(LongPathTime)             %Variance
AverGrad=mean(Gradient)            %Expected sensitivity
VariGrad=var(Gradient)             %Variance of sensitivity
%hist(Gradient, 40);               %Histogram
HalfLength=1.960*sqrt(VariGrad/NumRep); %Compute H.F.
IntervBegin=AverGrad-HalfLength         %CI for sensitivity
IntervBegin=AverGrad+HalfLength         %CI for sensitivity
```

Listing A.3: wdexp.m WD estimator for exponential distribution

A.1.4 SAN definition

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Title:    Two parallel arcs description Network generator
% Date:     03/22/2011
% By:       Mauricio Manterola
% Desc:
% Monte Carlo Simulation of Stoch. Activity Networks in Matlab
% Defines a SAN of two parallel arcs. Then it performs a scan thru
% different means to change the means ratio beta2/beta1 and it calls
% the ipa, sflr or wd procedure for gradient estimation.
% The scan is performed with logarithmic steps, in order to achieve
% proper plotting with logarithmic x-axis scale.
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% DEFINITION OF THE ACTIVITY NETWORK
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
clear all;

%Init Random Generator
s1=RandStream('mt19937ar');
RandStream.setDefaultStream(s1);
%reset(s);

%Init some constants and arrays
NumRep =        20000;     %Number of repetitions
LongPathTime =  zeros(NumRep,1);
MagicNumber=24.1174;
NumNodes=3;

%Set Src and Dest nodes
SrcNode=1;
DstNode=3;

%Set Network Activities
Pairs=[ 1 2;    %1-perturbed
        1 3;    %2
        2 3];   %3

%Number of Activities
[NumArcs, dummy]=size(Pairs);

%Parameters of the scan
slices=30;
inic=0.01;
final=10;
salto=(final/inic)^(1/slices)

%Aux
contad=0;   %aux counter
done=0;     %flag to exit loop

%Create array to store gradient variances
Varis = zeros(slices,2);

%Define non-random (deterministic) arcs
DetermArcs=[3]';

%Number of deterministic activities
[NumDetermArcs, dummy]=size(DetermArcs);

%Set Arcs wrt compute derivatives
SensiArc=[1]';

%Number of activities perturbated
[NumSensiArcs, dummy]=size(SensiArc);

%Define the connection matrix using from the arcs defined previously
```

```
AN=sparse(Pairs(:,1),Pairs(:,2),ones(NumArcs,1), NumNodes, NumNodes)

%Loop to scan thru the ratios
while(~done)

    %Write to screen current step
    contad

    %Compute mean
    MeanComptd=inic*(salto^contad)

    %Parameters for activities distributions
    MeanArcTimes=[1 MeanComptd 0]';

    %Clear gradient array from previous iteration
    Gradient=zeros(NumRep,1);

    %Construct mask to overwrite deterministic arcs
    DetermMask=MagicNumber*ones(NumArcs,1);

    %Overwrite deterministic arcs
    for i=1:NumDetermArcs
        indexdummy=DetermArcs(i,1);
        DetermMask(indexdummy,1)=MeanArcTimes(indexdummy,1);
    end; %endfor

    %Call routine to perform MCS
    ipa2parallelrep;
    %sflrparallelrep;
    %wdparallelrep;

    %Update counter
    contad=contad+1;

    %Store results from current iteration
    Varis(contad,1)=MeanComptd  %Ratio=MeanComptd/1
    Varis(contad,2)=VariGrad

    %Update done: TRUE if counter reach number of 'slices'
    done=(contad==slices);
end; %endwhile
```

Listing A.4: nwdef_2par_scan.m SAN definition code and scan for variable ratio of means (β_2/β_1)

A.1.5 Number of trees computation

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Title:    NTRESS PROCEDURE
% Date:     11/01/2011
% By:       Mauricio Manterola
% Desc:
% This procedure computes the number of trees rooted in the
% destination node. This number can be used as complexity
% measure of activity networks.
% This procedure takes the sparse type connection
% matrix (AN3) and computes the number of trees.
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Obtain the size of the connection matrix
[dime,dummy]=size(AN3);

% Convert the sparse type AN3 in a regular matrix BN
```

```
BN=full(AN3);

% Construct the Laplacian matrix
D=diag(sum(BN'))-BN;

% Obtain the the minor matrix D_NN
D(dime,:)=[];
D(:,dime)=[];

% Compute the determinant of the minor
Numtrees=det(D)
Numtreeslog=log(Numtrees)
```

Listing A.5: ntrees3.m Compute the number of trees rooted in the destination node

A.1.6 Restrictiveness estimator

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Title:    RESTRICTIVENESS PROCEDURE
% Date:     11/01/2011
% By:       Mauricio Manterola
% Desc:
% This procedure computes the Restrictiveness estimator (RT)
% of a network. This number can be used as complexity
% measure of activity networks.
% This procedure takes the sparse type connection
% matrix (AN3) and computes the RT number.
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Obtain the size of the connection matrix
[dime,dummy]=size(AN3);

% Copy input matrix AN3
RMatriz=AN3;

%Scan the matrix looking for '1's
for i=1:dime
    for j=1:dime
        if (RMatriz(i,j)==1)
            RMatriz(i,:)=or(RMatriz(i,:),RMatriz(j,:));
        end;
    end;
end;

%Construct modified matrix
RMMod=RMatriz+diag(ones(dime,1))

%Restrictiveness estimator formula
RTIndex=(2*sum(sum(RMMod))-6*(dime-1))/((dime-2)*(dime-3))
```

Listing A.6: rtindex.m Compute the restrictiveness index estimator

A.2 C code

A.2.1 IPA $U(0, 2\theta)$ two parallel arcs

```
/*************************************************************/
/* Title:   IPA Uniform(0,2*theta)                           */
/* Date:    06/22/2011                                       */
/* By:      Mauricio Manterola                               */
/* Desc:                                                     */
/* Monte Carlo Simulation of Stoch. Activity Networks in C   */
/* Compute IPA Gradient Estimates for a two parallel arc SAN  */
/* wrt a variation in the mean of the first arc.             */
/* It also perform an scan from INIMEAN to FINMEAN to account */
/* for a variation in the criticality. An plain text (columns */
/* separated by tabs) is generated to be used in Metapost.    */
/*************************************************************/

/************************/
/* Includes             */
/************************/
#include "RngStream.h"
#include <stdio.h>
#include <stdlib.h>
#include <math.h>

/************************/
/* Macros               */
/************************/
#define NUMREP      32000
#define SLICES      600
#define MEANA2      10.0f
#define INIMEAN     0.01f
#define FINMEAN     190.0f

/************************/
/* Global Variables     */
/************************/
RngStream   RandomGen;
float       Mean1;
float       Mean2;
float       Critical;
float       GradiMean;
float       GradiVar;

/************************/
/* Declarations         */
/************************/
float   UnifRNG(float MeanValue);
void    IPAGrad(void);

/************************/
/* Main Function        */
/************************/
int main(void)
{
    //Variables
    int         k;
    float       jumps;
    float       Critlity[SLICES]; //Crit.index storage
    float       Varis[SLICES];    //Variaces storage
    FILE*       Datad;            //File pointer


    //Info to output (screen)
    printf("MCS Gradient Estimation\n");
    printf("IPA Estimator, two parallel arcs, Unif(0,2*theta)\n");
    printf("Performs scan from INIMEAN to FINMEAN\n\n");

    //Create Random Stream
    RandomGen=RngStream_CreateStream("a");

    //Create report file
    Datad = fopen("dipa2parunif.d", "w");
    if (Datad==NULL)
    {
        printf("Error: Could not create file dataipa2parunif.d\n");
        return(1);
    }

    //Compute jump
    jumps=(FINMEAN-INIMEAN)/(SLICES);

    //Prepare some variables for iterations
    Mean2 =     MEANA2;

    for (k=0;k<SLICES;k++)
    {
        //Compute mean for Arc 1
        Mean1=INIMEAN+(jumps*k);

        //Call IPA computation function
        IPAGrad();

        //Save output of IPA for later
        Critlity[k]=Critical;
        Varis[k]=GradiVar;
    }

    //Save Criticality and Gradient Variances to output file
    for (k=0;k<SLICES;k++)
        fprintf(Datad, "%10.4f %10.4f\n",Critlity[k],Varis[k]);

    //Exit nicely
    return(0);
}

/************************/
/* IPA Comp. Function   */
/************************/
void IPAGrad(void)
{
    int         Rep;        //Repetion
    int         conta;      //Counter 4 crit. computation
    float       ArcTime1;   //1st arc time
    float       ArcTime2;   //2nd arc time
    float       meanaux;    //Aux. var for mean
    float       varaux;     //Aux. var for variance
    float       Gradi;      //Aux. var for gradient

    //Reset vars
    conta=0;
    meanaux=0.0f;
    varaux=0.0f;

    //Iteration for MCS
    for (Rep=0; Rep<NUMREP; Rep++)
    {
        //Generate ArcTime1 and ArcTime2 unif. distributed
        ArcTime1=UnifRNG(Mean1);
        ArcTime2=UnifRNG(Mean2);

        //Obtain longest path
        if (ArcTime1>ArcTime2)
        {
            conta++;                 //Increm. counter for crit.
            Gradi=(ArcTime1/Mean1); //Sensitivity sample
        }
        else
        {
            Gradi=0.0f;              //Sensitivity sample
        }
        meanaux =   meanaux+Gradi; //Accum. 4 mean
        varaux =    varaux+(Gradi*Gradi); //Accumm. 4 var
    }

    //Compute Mean and Variance
    GradiMean = meanaux/NUMREP; //Gradient expected val.
    varaux =    varaux/NUMREP;
    GradiVar =  varaux-(GradiMean*GradiMean); //Gradient variance

    //Compute Criticality
    Critical =  ((float)conta)/((float)NUMREP);
}

/***********************/
/* Unif(0,theta) RNG    */
/***********************/
float UnifRNG(float MeanValue)
{
    float   UnifDist1;
    float   UnifTheta;

    //Generate unif distrib.variates in (0,1)
    UnifDist1=RngStream_RandU01(RandomGen);

    //Compute Unif(0,theta)
    UnifTheta=UnifDist1*2*MeanValue;

    return UnifTheta;
}
```

Listing A.7: ipaparallelunif.c: IPA estimator for uniform distribution

A.2.2 SF/LR Gaussian pure series SANs

```
/************************************************************/
/* Title:   SF/LR Gaussian incremental pure series           */
/* Date:    08/23/2011                                       */
/* By:      Mauricio Manterola                               */
/* Desc:                                                     */
/* Monte Carlo Simulation of Stoch. Activity Networks in C   */
/* Computes SF/LR Gradient Estimates for pure series Gaussian */
/* SANs wrt a variation in the mean of the first arc.        */
/* It add a new arc at the end of the network still MAXARCS is */
/* reached. A plain text (columns separated by tabs) is      */
/* generated to be used in Metapost.                         */
/************************************************************/
/***********************/
/* Includes            */
/***********************/
#include "RngStream.h"
#include <stdio.h>
#include <stdlib.h>
#include <math.h>
```

```
/************************/
/* Macros               */
/************************/
#define NUMREP      30000
#define MAXARCS     50
#define STD_DEV     5.0f
#define MEAN_VAL    30.0f

/************************/
/* Global Variables     */
/************************/
RngStream   RandomGen;
float       Mean1;      //Mean for every arc
float       StaDev1;    //Standard deviation
int         NumArcs;    //No. of arcs
float       GradiMean;  //Sensitivity mean (expd. val.)
float       GradiVar;   //Sensitivity variance

/************************/
/* Declarations         */
/************************/
float GaussRNG(float MeanValue, float StdDev);
void SFLRGrad(void);


/************************/
/* Main Function        */
/************************/
int main(void)
{
    //Variables
    int         k;
    float       Varis[MAXARCS]; //Storage of variances for each SAN
    FILE*       Datad;          //Ptr. to file


    //Info to output (screen)
    printf("MCS Gradient Estimation\n");
    printf("SF/LR Estimator, pure series arcs\n\n");

    //Create Random Stream
    RandomGen=RngStream_CreateStream("a");

    //Create report file
    Datad = fopen("datasflrpureseriesgauss.d", "w");
    if (Datad==NULL)
    {
        printf("Error: Could not create file datasflrpureseriesgauss.d\n");
        return(1);
    }

    //Prepare some variables for iterations
    StaDev1 =   STD_DEV;
    Mean1 =     MEAN_VAL;

    for (k=0;k<MAXARCS;k++)
    {
        //Add a new arc
        NumArcs=k+1;

        //Call IPA computation function
        SFLRGrad();

        //Save output of IPA for later
        Varis[k]=GradiVar;
    }

    //Save no. of arcs and variance of sensitivity
    for (k=0; k<MAXARCS; k++)
        fprintf(Datad, "%d %10.4f\n",(k+1),Varis[k]);

    //Exit nicely
    return(0);
}

/***********************/
/* SF/LR Comp. Function */
/* two parallel arcs    */
/* N(theta,sigma) distr.*/
/***********************/
void SFLRGrad(void)
{
    int         i;          //aux index for array
    int         Rep;        //Index for repetitions
    float*      ArcTime;    //Ptr to array of arc times
    float       meanaux;    //Aux for mean
    float       varaux;     //Aux for variance
    float       Gradi;      //Estim. val at each rep.
    float       Longpath;   //Total completion time

    //Reset vars
    meanaux=0.0f;
    varaux=0.0f;

    //Allocate memory for arc times.
    ArcTime=(float*)malloc(NumArcs*sizeof(float));

    //Iteration for MCS
    for (Rep=0; Rep<NUMREP; Rep++)
    {
        //Generate ArcTimes Gaussian distributed
        for (i=0; i<NumArcs; i++)
            ArcTime[i]=GaussRNG(Mean1, StaDev1);

        //Obtain longest path
        Longpath=0.0f;
        for (i=0; i<NumArcs; i++)
            Longpath=Longpath+(ArcTime[i]); //Just sum of arcs

        //Obtain Grad Estim sample
        Gradi=Longpath*(((ArcTime[0]-Mean1)/StaDev1)*((ArcTime[0]-Mean1)/StaDev1)↩
            -1)/StaDev1;

        //Accumulate samples for mean and var computation
        meanaux =   meanaux+Gradi;
        varaux =    varaux+(Gradi*Gradi);
    }

    //Computes Mean and Variance
    GradiMean = meanaux/NUMREP;
    varaux =    varaux/NUMREP;
    GradiVar =  varaux-(GradiMean*GradiMean);

    //Frees allocated memory
    free(ArcTime);
}

/***********************/
/* Gauss RNG Function   */
/***********************/
float GaussRNG(float MeanValue, float StdDev)
{
    char    notDone;    //Flag for A-R method
    float   NormalVar;  //Normal RV
    float   MaxwellVar; //Maxwell RV
    float   UnifDist1;  //Unif RV
    float   UnifDist2;  //Unif RV
    float   UnifDist3;  //Unif RV
    float   UnifDist4;  //Unif RV
    float   WeiVar;     //Weibull RV

    //Init vars
    notDone=1;

    //Generate Weibull number by Accept-Reject method
    while(notDone)
    {
        //Generate two unif distrib.variates in (0,1)
        UnifDist1=RngStream_RandU01(RandomGen);
        UnifDist2=RngStream_RandU01(RandomGen);

        //Compute Weibull dist. Variable
        WeiVar=sqrt(-3.0*log(UnifDist1));

        //Accept of Reject
        notDone=( UnifDist2>(0.951889669*WeiVar*sqrt(UnifDist1)) );
    }

    //Generate more unif distrib.variates in (0,1) for Maxwell and Gaussian
    UnifDist3=RngStream_RandU01(RandomGen);
    UnifDist4=RngStream_RandU01(RandomGen);

    //Generate double size Maxwell and Gaussian Variates
    if ( UnifDist3 > 0.5)   //Right size RV
    {
        MaxwellVar  =   MeanValue+(WeiVar*StdDev);
        NormalVar   =   MeanValue+(WeiVar*UnifDist4*StdDev);
    }
    else                    //Left size RV
    {
        MaxwellVar  =   MeanValue-(WeiVar*StdDev);
        NormalVar   =   MeanValue-(WeiVar*UnifDist4*StdDev);
    }

    return NormalVar;
}
```

Listing A.8: sflrpureseriesgaussian.c: SF/LR estimator for Gaussian distribution

A.2.3 WD exponential combo SAN

```
/***************************************************************/
/* Title:   WD exponential(theta)                              */
/* Date:    08/24/2011                                         */
/* By:      Mauricio Manterola                                 */
/* Desc:                                                       */
/* Monte Carlo Simulation of Stoch. Activity Networks in C     */
/* Compute WD Gradient Estimates for a combination SANs        */
/* wrt a variation in the mean of the first arc.               */
/* It also perform an scan from INIMEAN to FINMEAN to account  */
/* for a variation in the criticality (from 0 to 1). A plain   */
/* text (columns separated by tabs) is generated to be used in */
/* Metapost.                                                   */
/***************************************************************/
/***********************/
/* Includes            */
/***********************/
#include "RngStream.h"
#include <stdio.h>
```

```
#include <stdlib.h>
#include <math.h>

/************************/
/* Macros               */
/************************/
#define NUMREP      32000
#define SLICES      500
#define INICRIT     0.01f
#define FINCRIT     0.99f
#define WEIG        0.99f

/************************/
/* Global Variables     */
/************************/
RngStream   RandomGen;  //Lecuyer RNG
float       Mean1;      //Mean arc 1
float       Mean2;      //Mean arc 2
float       Mean3;      //Mean arc 3
float       Critical;   //Crit.index
float       GradiMean;  //Sensitivity mean (expd. val.)
float       GradiVar;   //Sensitivity variance

float       ExpTheta;   //Exponential(theta) RV
float       ErlTheta;   //Erlang(2,theta) RV
/************************/
/* Declarations         */
/************************/
void    ExpRNG(float MeanValue);
void    WDGrad(void);

/************************/
/* Main Function        */
/************************/
int main(void)
{
    //Variables
    int         k;                  //For scan thru criticalities
    float       jumps;              //Step size
    float       Critlity[SLICES];   //Crit.index storage
    float       Varis[SLICES];      //Variance storage
    float       CritTgt;            //Crit.index target
    FILE*       Datad;              //Ptr to output file


    //Info to output (screen)
    printf("MCS Gradient Estimation\n");
    printf("WD Estimator, Combo network, Exp(theta)\n\n");

    //Create Random Stream
    RandomGen=RngStream_CreateStream("a");

    //Create report file
    Datad = fopen("datawdcomboexp.d", "w");
    if (Datad==NULL)
    {
        printf("Error: Could not create file datawdcomboexp.d\n");
        return(1);
    }

    //Compute jump
    jumps=(FINCRIT-INICRIT)/(SLICES);

    //Prepare some variables for iterations
    Mean1=WEIG;
    Mean3=1-(WEIG);

    for (k=0;k<SLICES;k++)
    {
        //Compute target criticality
        CritTgt=INICRIT+(jumps*k);

        //Compute mean for Arc 2 accordingly
        Mean2=((1-CritTgt)+sqrt(((1-CritTgt)*(1-CritTgt))+(4*CritTgt*(1-CritTgt)*↩
            Mean1*Mean3)))/(2*CritTgt);

        //Call WD computation function
        WDGrad();

        //Save output from WD for later
        Critlity[k]=Critical;
        Varis[k]=GradiVar;
    }

    //Save arrays of crit.index and grad. variance to output file
    for (k=0;k<SLICES;k++)
        fprintf(Datad, "%10.4f %10.4f\n",Critlity[k],Varis[k]);

    //Exit nicely
    return(0);
}

/************************/
/* IPA Comp. Function   */
/************************/
void WDGrad(void)
{
    int         Rep;        //Index for repetitions
    int         conta;      //Counter for crit.
    float       ArcTime1;   //Realization 1st arc
    float       ArcTime2;   //Realization 2nd arc
    float       ArcTime3;   //Realization 3rd arc
    float       meanaux;    //Aux for mean computation
    float       varaux;     //Aux for vatiance computation
    float       Gradi;      //Gradient sample

    float       Longpath;   //Completion time sample

    float       ArcTime1WD; //Realization arc1 modified dist.
    float       LongpathWD; //Completion time w/mod.distr.

    //Reset vars
    conta=0;
    meanaux=0.0f;
    varaux=0.0f;

    //Iteration for MCS
    for (Rep=0; Rep<NUMREP; Rep++)
    {
        //Generate ArcTime1, ArcTime2, ArcTime3 exponentially
        // distributed and ArcTime1WD Erl(2,theta) 4 WD
        ExpRNG(Mean1);
        ArcTime1=ExpTheta;
        ArcTime1WD=ErlTheta;
        ExpRNG(Mean2);
        ArcTime2=ExpTheta;
        ExpRNG(Mean3);
        ArcTime3=ExpTheta;

        //Obtain longest path
        if ((ArcTime1+ArcTime3)>ArcTime2)
        {
            conta++;
            Longpath=ArcTime1+ArcTime3;
        }
        else
        {
            Longpath=ArcTime2;
        }

        //Obtain longest path modified distrib (erlang)
        if ((ArcTime1WD+ArcTime3)>ArcTime2)
        {
            LongpathWD=ArcTime1WD+ArcTime3;
        }
        else
        {
            LongpathWD=ArcTime2;
        }

        //Compute Gradient
        Gradi=(LongpathWD-Longpath)/Mean1;

        //Accum. 4 mean and variance computation
        meanaux =   meanaux+Gradi;
        varaux =    varaux+(Gradi*Gradi);
    }


    //Compute Mean and Variance
    GradiMean = meanaux/NUMREP;
    varaux =    varaux/NUMREP;
    GradiVar =  varaux-(GradiMean*GradiMean);

    //Compute Criticality
    Critical =  ((float)conta)/((float)NUMREP);
}

/************************/
/* Exp(theta) RNG       */
/************************/
void ExpRNG(float MeanValue)
{
    float   UnifDist1;
    float   UnifDist2;

    //Generate unif distrib.variates in (0,1)
    UnifDist1=RngStream_RandU01(RandomGen);
    UnifDist2=RngStream_RandU01(RandomGen);

    //Compute Exp(theta)
    ExpTheta=-log(UnifDist1)*MeanValue;

    //Compute Erl(theta)
    ErlTheta=-log(UnifDist2*UnifDist1)*MeanValue;
}
```

Listing A.9: wdcomboexpo.c: WD estimator for exponential distribution

Bibliography

[1] Dimitri P. Bertsekas. *Network Optimization: Continuous and Discrete Models.* Athena Scientific, May 1998.

[2] J. -G. Cho and B. -J. Yum. Functional estimation of activity criticality indices and sensitivity analysis of expected project completion time. *The Journal of the Operational Research Society*, 55(8):850–859, 2004.

[3] E. M. Davies. An experimental investigation of resource allocation in multiactivity projects. *Operational Research Quarterly (1970-1977)*, 24(4):pp. 587–591, 1973.

[4] Matthias Dehmer, Abbe Mowshowitz, and Frank Emmert-Streib. Connections between classical and parametric network entropies. *PLoS ONE*, 6(1):8, 2011.

[5] D. Easley and J. Kleinberg. *Networks, Crowds, and Markets: Reasoning About a Highly Connected World.* Cambridge University Press, 2010.

[6] Salah E. Elmaghraby. On criticality and sensitivity in activity networks. *European Journal of Operational Research*, 127(2):220 – 238, 2000.

[7] Salah E. Elmaghraby and Willy S. Herroelen. On the measurement of complexity in activity networks. *European Journal of Operational Research*, 5(4):223–234, October 1980.

[8] Michael C. Fu. Chapter 19 gradient estimation. In Shane G. Henderson and Barry L. Nelson, editors, *Simulation*, volume 13 of *Handbooks in Operations Research and Management Science*, pages 575 – 616. Elsevier, 2006.

[9] Michael C. Fu. Sensitivity analysis in monte carlo simulation of stochastic activity networks. In *Perspectives in Operations Research*, pages 351–366. 2006.

[10] Michael C. Fu. What you should know about simulation and derivatives. *Naval Research Logistics*, 55(8):723–736, 2008.

[11] Michael C. Fu and Jian-Qiang Hu. *Conditional Monte Carlo: Gradient Estimation and Optimization Applications.* Springer, March 1997.

[12] Chris Groër and Ken Ryals. Sensitivity analysis in simulation of stochastic activity networks: a computational study. In Edward K. Baker, Anito Joseph, Anuj Mehrotra, and Michael A. Trick, editors, *Extending the Horizons: Advances in Computing, Optimization, and Decision Technologies*, volume 37, pages 183–2000. Springer US, 2007.

[13] Bernd Heidergott, Felisa J. Vazquez-Abad, and Warren Volk-Makarewicz. Sensitivity estimation for gaussian systems. *European Journal of Operational Research*, 187(1):193–207, May 2008.

[14] Richard A. Kaimann. Coefficient of network complexity. *Management Science*, 21(2):172–177, October 1974.

[15] Antti Latva-Koivisto. Finding a complexity measure for business process models, 2001.

[16] Pierre L'Ecuyer. Good parameters and implementations for combined multiple recursive random number generators. *Operations Research*, 47(1):159 –164, January 1999.

[17] Makoto Matsumoto and Takuji Nishimura. Mersenne twister: a 623-dimensionally equidistributed uniform pseudo-random number generator. *ACM Trans. Model. Comput. Simul.*, 8:3–30, January 1998.

[18] Abbe Mowshowitz. Entropy and the complexity of graphs: I. an index of the relative complexity of a graph. *Bulletin of Mathematical Biology*, 30:175–204, 1968.

[19] Chandra Nair, Balaji Prabhakar, and Devavrat Shah. On entropy for mixtures of discrete and continuous variables. *Computing Research Repository (CoRR)*, abs/cs/0607075, 2006.

[20] T. Lawrence Pascoe. Allocation of resources CPM. *Revue Française de Recherche Opérationnelle*, 38:31–38, 1966.

[21] S.M. Ross. *Stochastic processes*. Wiley series in probability and mathematical statistics. Probability and mathematical statistics. Wiley, 1983.

[22] Claude E. Shannon. A mathematical theory of communication. *Bell System Technical Journal*, 27(3, 4):379–423, 623–656, July , October 1948.

[23] H. N. V. Temperley. *Graph Theory and Applications*. Ellis Horwood Limited, 1981.

[24] A. Thesen. Measures of the restrictiveness of project networks. *Networks*, 7(3):193–208, 1977.