SAMPLING SCHEMES FOR ESTIMATING

SOFTWARE RELIABILITY

A DISSERTATION
IN
Mathematics
and
Physics

Presented to the Faculty of the University
of Missouri–Kansas City in partial fulfillment of
the requirements for the degree

DOCTOR OF PHILOSOPHY

by
YOUSEF ALHARBI

B.S. Al Qassim University, 2010
M.S. Kent State University, 2017
M.S. University of Missouri-Kansas City, 2021

Kansas City, Missouri
2022

SAMPLING SCHEMES FOR ESTIMATING

SOFTWARE RELIABILITY

YOUSEF ALHARBI, Candidate for the Doctor of Philosophy Degree

University of Missouri–Kansas City, 2022

ABSTRACT

Any software system of non-trivial size cannot be easily and completely tested because the domain of all possible inputs is complex and very large. In this study, we use a technique called partition testing, in which we divide the input domain of all potential testing cases D into $K \geq 2$ non-overlapping sub-domains. Each sub-domain can therefore be tested independently from the others. We employ two methods, a fully sequential method and two-stages method, that are based on a sample of the test cases to allocate the test cases among partitions and minimize the variance of estimated software reliability when usage probabilities are random. These methods allow us to take advantages from the previous testing as we test and, as a result, dynamically improve the distribution of test cases throughout the reliability testing process. By dynamically allocating test cases to partitions, these methods aim to minimize the variance of the reliability estimation. The variance in-

curred by fully sequential method and the variance incurred by two-stages method are compared with the variance incurred by the optimal and the variance incurred by the balanced sampling method. Using theoretical results and a Monte Carlo simulation, the fully sampling method and the two-stages method perform better than the balanced sampling method and are nearly optimal.

APPROVAL PAGE

The faculty listed below, appointed by the Dean of the College of Graduate Studies, have examined a dissertation titled " "Sampling Schemes For Estimating Software Reliability," presented by YOUSEF ALHARBI, candidate for the Doctor of Philosophy degree, and certify that in their opinion it is worthy of acceptance.

Supervisory Committee

Kamel Rekab, Ph.D., Committee Chair
Department of Mathematics and Statistics

Elizabeth Stoddard, Ph.D.,Co-discipline Advisor
Department of Physics and Astronomy

Majid Bani-Yaghoub, Ph.D.
Department of Mathematics and Statistics

Rhee Noah, Ph.D.
Department of Mathematics and Statistics

Da-Ming Zhu, Ph.D.
Department of Physics and Astronomy

Contents

TABLES

ACKNOWLEDGEMENTS

I would like to show my greatest appreciation to Dr. Kamel Rekab. Without his guidance and persistent help, this dissertation would not have been possible. I sincerely appreciate the intellectual independence that he fostered in his research, and I hope to emulate his open mindedness, creativity, and scientific rigor in my own career.

I would like to express my gratitude to Dr. Elizabeth Stoddard, Dr. Majid Bani-Yaghoub, Dr. Rhee Noah, and Dr. Da-Ming Zhu for their kind agreement to be on my doctoral committee.

I cannot express my thanks to my parents for their love, prayers, and continuing support over my life. All thanks also go to my brothers and sisters. Special thanks are for Al-fuyran ,my big family, for all their support and encouragement. Thanks to everyone who helped me, even if with only an encouraging word.

I give my deepest thanks and gratitude to my wife Fatimah and my sons, Adnan, Yaqoob, and Noah for for being in my life and for their unwavering patience as I achieved this goal.

CHAPTER 1

INTRODUCTION

Software reliability is one of the most essential characteristics for a critical system that can impact the lives of humans. Reliability is the ability of the software system to preserve a specific level of efficiency when used under a specified set of circumstances. To ensure the reliability of software, developers have to make a test at the end of the development cycle. The decision makers are based on the outcome of these testing to determine whether it is the time to launch the software or not.

Software that is highly accurate profits some industries and academic organizations. So, all companies and users are looking for a reliable software to reduce both risk and cost. However,unreliable software can have disastrous consequences. In April 2018, A "simple" software upgrade that resulted in a significant banking outage led to millions of TSB Bank customers being locked out of their accounts. Although the system upgrade was planned, it seems that it wasn't done sufficiently. Customer login problems started occurring as soon as TSB turned on the new system. Details of people's accounts were revealed to others. Additionally, inaccurate credits and debts were reported.Many customers were unable to access their accounts for two weeks before being allowed access[10].

An extensive test is defined as the set of all possible system input. Since the domain of all possible inputs is complex and very large, exhaustive testing is not possible in software reliability for any non-trivial software. As a result, just a small portion of the program's input domain can be allocated for testing using some testing strategy. Numerous testing strategies have been explored and put into use, including functional testing [7], control flow coverage [1], data flow coverage [12], mutation testing [4], partition analysis [22], and others[26, **?**, 28, 29]. The strategy used in this paper's approach focuses on partition testing, which divides the input domain of all potential test cases **D** into $K \geq 2$ non-overlapping sub-domains, where each sub-domains can be tested independently from the others. If test case j is taken from partition i, then none of the other partitions will have test case j. Our goal of this paper is to minimize the variance of the overall estimated reliability of the software by allocating the test cases to the partitions.

CHAPTER 2

SOFTWARE RELIABILITY ESTIMATION FOR K PARTITIONS

2.1 Introduction

In this chapter, we start by officially defining our partition testing model as a mathematical model that can be thought of as a stratified random sampling model [3, 11] composed of:

- The partitions $\mathbf{D_1}, \mathbf{D_2}, \ldots, \mathbf{D_k}$ must cover the entire domain $\mathbf{D}$ and be mutually disjointed so that

$$\mathbf{D} = \bigcup_{\mathbf{i=1}}^{\mathbf{k}} \mathbf{D_i}$$

$$\mathbf{D_i} \cap \mathbf{D_j} = \emptyset \ , \mathbf{i} \neq \mathbf{j}$$

  where D_i sub-domain of the ith partition, D is the domain of all test cases, and k is the total number of partitions.

- The sizes of samples $n_1, n_2, \ldots, n_k$ are taken from sub-domain $\mathbf{D_1}, \mathbf{D_2}, \ldots, \mathbf{D_k}$ and N the total number of test cases

$$\sum_{i=1}^{k} n_i = N$$

Considering a simple random sample is taken from the whole domain D, we define the probability P_i as a "usage probability [27]" for each partition D_i. As a result, if we observe a sample of users chosen at random, P_i percent of their software usage is typically found in test cases of sub-domain D_i. Hence, P_i will be assumed unknown and $\sum_{i=1}^{k} P_i = 1$. Furthermore, we describe θ_i as the reliability of each sub-domain D_i. The previous definitions are used to determine the overall software reliability, η:

$$\eta = \sum_{i=1}^{k} P_i \theta_i$$

As mentioned above, any software system of non-trivial size cannot be completely tested. We use partition testing technique divide the N test cases allocated for reliability estimation across the k partitions, and then utilize the results to estimate each θ_i. Thus, the estimate of the overall software reliability η,denoted by $\hat{\eta}$ can be determined as:

$$\hat{\eta} = \sum_{i=1}^{k} P_i \hat{\theta}_i$$

where $\hat{\theta}_i$ is the estimate of θ_i after all n_i tests have been allocated to partitions i such that:

$$\hat{\theta}_i = \frac{\sum_{j=1}^{n_i} X_{ij}}{n_i}$$

where n_i is the number of test cases allocated to partitions i and X_{ij} are the outcome of the jth test taken from the ith partition modeled as Bernoulli random variable with a parameter θ_i such that:

$$X_{ij} = \begin{cases} 1 & \text{,if the jth test case of ith partition succeeds} \\ 0 & \text{,if the jth test case of ith partition fails} \end{cases}$$

We shall assume that P_i and θ_i are independents, where $i = 1, \ldots, k$. Since the objective is to allocate the test cases among partitions to determine the optimal estimated software reliability, we will approach the process with the goal of minimizing the variance of our estimated reliability as follow:

$$Var[\sum_{i=1}^{k} P_i\hat{\theta}_i] = \sum_{i=1}^{k} Var[P_i\hat{\theta}_i] + 2\sum_{j=1}^{k}\sum_{i<j} Cov(P_i\hat{\theta}_i, P_j\hat{\theta}_j) \qquad (2.1)$$

In the next section, we will show the near-optimal method that minimizes the variance of our estimated reliability.

2.2 Optimal Sampling Method

Our goal in this section is to determine the optimal estimated software reliability by minimizing the variance of our estimated reliability. This section will begin with an important lemma by Rekab [13].

Rekab's Lemma: If X and X are independent, then Y/X=Y , X/Y=X and

$$Var(XY) = E(Y)^2 Var(X) + Var(Y)E(X^2)$$

We will start our goal of minimizing the variance of our estimated reliability :

$$Var[\sum_{i=1}^{k} P_i\hat{\theta}_i] = \sum_{i=1}^{k} Var[P_i\hat{\theta}_i] + 2\sum_{j=1}^{k}\sum_{i<j} Cov(P_i\hat{\theta}_i, P_j\hat{\theta}_j) \tag{2.2}$$

Using Rekab's Lemma, the first term on the right side of equation (2.2) becomes

$$\begin{aligned} \sum_{i=1}^{k} Var[P_i\hat{\theta}_i] &= \sum_{i=1}^{k}[Var(\hat{\theta}_i)E(P_i^2) + Var(P_i)E^2(\hat{\theta}_i)] \\ &= \sum_{i=1}^{k}[\frac{\sigma_{x_i}^2}{n_i}E(P_i^2) + Var(P_i)\theta_i^2], \text{ where } \sigma_{x_i}^2 = \theta_i(1-\theta_i) \end{aligned} \tag{2.3}$$

and the second term on the left side of equation (2.2) becomes

$$2\sum_{j=1}^{k}\sum_{i<j} Cov(P_i\hat{\theta}_i, P_j\hat{\theta}_j) = 2\sum_{j=1}^{k}\sum_{i<j} \theta_i\theta_j Cov(P_i, P_j) \tag{2.4}$$

Therefore, we define the variance of our estimated reliability $\hat{\eta}$ as

$$\begin{aligned} Var[\hat{\eta}] &= \sum_{i=1}^{k} \frac{\sigma_{x_i}^2}{n_i}E(P_i^2) + C \\ \text{where} \quad C &= 2\sum_{j=1}^{k}\sum_{i<j} \theta_i\theta_j Cov(P_i, P_j) + \textstyle\sum_{i=1}^{k}[Var(P_i)\theta_i^2] \end{aligned} \tag{2.5}$$

The problem's objective can now be restated as follows: find the best sampling strategy that will choose $n_1, n_2, \ldots, n_k$ test cases so that the variance is as small as possible. The equation (2.5) can be written as

$$Var(\hat{\eta}) = \frac{\sum_{i=1}^{k}(\sqrt{E(P_i^2)}\sigma_{x_i})^2}{N} + \frac{1}{N}\sum_{i=1}^{k-1}\sum_{j=i+1}^{k}(\frac{n_i\sqrt{E(P_j^2)}\sigma_{x_j} - n_j\sqrt{E(P_i^2)}\sigma_{x_i}}{n_i n_j})^2 + C \tag{2.6}$$

Hence, the variance is bounded by:

$$Var(\hat{\eta}) \geq \frac{\sum_{i=1}^{k}(\sqrt{E(P_i^2)}\sigma_{x_i})^2}{N} + C \tag{2.7}$$

with equality if:

$$\frac{n_i}{n_j} = \frac{\sqrt{E(P_i^2)}\sigma_{x_i}}{\sqrt{E(P_j^2)}\sigma_{x_j}} \tag{2.8}$$

Therefore, the optimal allocation is determined by:

$$\frac{N}{n_j} = \frac{\sum_{i=1}^{k} \sqrt{E(P_i^2)}\sigma_{x_i}}{\sqrt{E(P_j^2)}\sigma_{x_j}}$$

$$n_j = N \frac{\sqrt{E(P_j^2)}\sigma_{x_j}}{\sum_{i=1}^{k} \sqrt{E(P_i^2)}\sigma_{x_i}}$$

where $j = 1, \ldots, k-1$

$$n_k = N - \sum_{j=1}^{k-1} n_j$$

By achieving the equality of equation (2.7), the variance explained by the optimal allocation, which is:

$$Var(\hat{\eta}_{opt}) = \frac{\sum_{i=1}^{k} (\sqrt{E(P_i^2)}\sigma_{x_i})^2}{N} + C \tag{2.9}$$

Since the optimal allocation depends on the actual reliabilities which are unknown, the optimal sampling scheme is not practical. As a result of this shortcoming, we decided to employ dynamic allocation approaches, which are discussed in chapter(4) and chapter(5).

CHAPTER 3

SOFTWARE RELIABILITY ESTIMATION FOR TWO PARTITIONS

3.1 Introduction

In this chapter, we introduce the same problem in chapter 2 when we have two partitions, $k = 2$ for one reason. The reason is we will use two partitions in Monte Carlo Simulation in chapter 6. Now, we start by considering the case in which the test domain is divided into two subdomains D_1, D_2, each with reliability θ_1 and θ_1. Samples of sizes n_1 and n_2 are drawn from subdomains D_1, D_2, respectively such that $N = n_1 + n_2$. We have one assumption which is P_i and θ_i where $i = 1, 2$ are independents. We define η as $\eta = P_1\theta_1 + P_2\theta_2$.

We have

$$E(X) = \theta_1 \quad , \quad E(Y) = \theta_2$$

$$Var(X) = \theta_1(1 - \theta_1) = \sigma_x^2 \quad , \quad Var(Y) = \theta_2(1 - \theta_2) = \sigma_y^2$$

The strategy that minimizes the variance of our estimated reliability for two partitions will be presented in the next section.

3.2 Optimal Sampling Method

We aim to minimize the variance of our estimated reliability for two partitions in this section to determine the optimal estimated software reliability. Thus, the variance of our estimated reliability for two partitions is shown as:

$$\begin{aligned} Var[\hat{\eta}] = Var[P_1\hat{\theta}_1 + P_2\hat{\theta}_2] = Var[P_1\hat{\theta}_1 + (1-P_1)\hat{\theta}_2] \\ = Var[P_1\hat{\theta}_1] + Var[(1-P_1)\hat{\theta}_2] + 2Cov(P_1\hat{\theta}_1, (1-P_1)\hat{\theta}_2) \end{aligned} \tag{3.1}$$

by using Rekab's Lemma, the first term of the right hand side of equation (3.1) becomes

$$\begin{aligned} Var[P_1\hat{\theta}_1] = Var(\hat{\theta}_1)E(P_1^2) + Var(P_1)E^2(\hat{\theta}_1) \\ = \frac{\sigma_x^2}{n_1}E(P_1^2) + Var(P_1)\theta_1^2 \end{aligned} \tag{3.2}$$

by using Rekab's Lemma, the second term of the right hand side of equation (3.1) becomes

$$\begin{aligned} Var\left[(1-P_1)\hat{\theta}_2\right] = Var(\hat{\theta}_2)E((1-P_1)^2) + Var(1-P_1)E^2(\hat{\theta}_2) \\ = \frac{\sigma_y^2}{n_2}E((1-P_1)^2) + Var(P_1)\theta_2^2 \end{aligned} \tag{3.3}$$

Since P_i and θ_i where $i = 1, 2$ are independents, the last term of the right hand side of equation (3.1) becomes

$$\begin{aligned}
Cov(P_1\hat{\theta}_1, (1-p)\hat{\theta}_2) &= Cov(P_1\hat{\theta}_1, \hat{\theta}_2) - Cov(P_1\hat{\theta}_1, P_1\hat{\theta}_2) \\
Cov(P_1\hat{\theta}_1, \hat{\theta}_2) &= E(P_1\hat{\theta}_1\hat{\theta}_2) - E(P_1\hat{\theta}_1)E(\hat{\theta}_2) \\
&= E(P_1\hat{\theta}_1)E(\hat{\theta}_2) - E(P_1\hat{\theta}_1)E(\hat{\theta}_2) = 0 \\
Cov(P_1\hat{\theta}_1, P_1\hat{\theta}_2) &= E\left(P_1^2\hat{\theta}_1\hat{\theta}_2\right) - E(P_1\hat{\theta}_1)\cdot E(P_1\hat{\theta}_2) \\
&= E\left(P_1^2\right)E(\hat{\theta}_1)E(\hat{\theta}_2) - E(P_1)E(\hat{\theta}_1)E(P_1)E(\hat{\theta}_2) \\
&= E\left(P_1^2\right)\theta_1\theta_2 - E^2(P_1)\theta_1\theta_2 = \left(E\left(P_1^2\right) - E^2(P_1)\right)\theta_1\theta_2 \\
&= Var(P_1)\theta_1\theta_2 \\
2\,\mathrm{Cov}(P_1\hat{\theta}_1, (1-P_1)\hat{\theta}_2) &= -2Var(P_1)\theta_1\theta_2
\end{aligned} \tag{3.4}$$

Substitute the equations (3.2), (3.3),and (3.4) into equation (3.1)

$$\begin{aligned}
Var[P_1\hat{\theta}_1 + (1-P_1)\hat{\theta}_2] = \frac{\sigma_x^2}{n_1}E(P_1^2) + \theta_1^2 Var(P_1) + \frac{\sigma_y^2}{n_2}E((1-P_1)^2) \\
+\theta_2^2 Var(P_1) - 2Var(P_1)\theta_1\theta_2
\end{aligned} \tag{3.5}$$

Thus, the variance of our estimated reliability will written as:

$$\mathrm{Var}[P_1\hat{\theta}_1 + P_2\hat{\theta}_2] = \frac{\sigma_x^2}{n_1}E(P_1^2) + \frac{\sigma_y^2}{n_2}E(P_2^2) + \mathrm{Var}(P_1)(\theta_1 - \theta_2)^2 \tag{3.6}$$

Now, we need to choose the optimal n_1, n_2 so that equation (3.6) is minimum. Since the third term of equation (3.6) depends on the unknown operational

profiles, and unknown software reliability estimator, so we can not handle it. In the next step, we will manipulate the rest of equation (3.6).

First, Let $\alpha = E(P_1^2)$ and $\beta = E(P_2^2)$.

Hence the first and second terms of equation (3.6) can be rewritten such that:

$$\frac{\alpha\sigma_x^2}{n_1} + \frac{\beta\sigma_y^2}{n_2}$$

$$= \frac{n_2\ \alpha\sigma_x^2 + n_1\ \beta\ \sigma_y^2}{n_1 n_2}$$

$$= \frac{N(n_2\ \alpha\sigma_x^2 + n_1\ \beta\ \sigma_y^2)}{N\ n_1 n_2}$$

$$= \frac{(n_1 + n_2)(n_2\ \alpha\sigma_x^2 + n_1\ \beta\ \sigma_y^2)}{(n_1 + n_2)\ n_1 n_2}$$

$$= \frac{(n_1 n_2 + n_2^2)\ \alpha\sigma_x^2 + (n_1^2 + n_1 n_2)\ \beta\ \sigma_y^2}{(n_1 + n_2)\ n_1 n_2}$$

$$= \frac{n_1 n_2 \alpha\sigma_x^2 + n_2^2\ \alpha\sigma_x^2 + n_1^2\beta\ \sigma_y^2 + n_1 n_2\ \beta\ \sigma_y^2}{(n_1 + n_2)\ n_1 n_2}$$

$$= \frac{n_1 n_2 \alpha \sigma_x^2 + n_2^2\ \alpha \sigma_x^2 + n_1^2 \beta\ \sigma_y^2 + n_1 n_2\ \beta\ \sigma_y^2}{(n_1 + n_2)\ n_1 n_2} + \frac{2\sqrt{\alpha}\sqrt{\beta}\sigma_x^2\sigma_y^2 - 2\sqrt{\alpha}\sqrt{\beta}\sigma_x^2\sigma_y^2}{(n_1 + n_2)\ n_1 n_2}$$

$$= \frac{n_1 n_2 \alpha \sigma_x^2 + n_1 n_2\ \beta\ \sigma_y^2 + 2 n_1 n_2 \sqrt{\alpha}\sqrt{\beta}\sigma_x^2\sigma_y^2}{(n_1 + n_2)\ n_1 n_1 n_2} + \frac{n_2^2\ \alpha \sigma_x^2 + n_1^2 \beta\ \sigma_y^2 - 2 n_1 n_2 \sqrt{\alpha}\sqrt{\beta}\sigma_x^2\sigma_y^2}{(n_1 + n_2)\ n_1 n_2}$$

$$= \frac{(\sqrt{\alpha}\sigma_x + \sqrt{\beta}\sigma_y)^2}{(n_1 + n_2)} + \frac{(n_2\sqrt{\alpha}\sigma_x - n_1\sqrt{\beta}\sigma_y)^2}{(n_1 + n_2)\ n_1 n_2}$$

$$= \frac{(\sqrt{\alpha}\sigma_x + \sqrt{\beta}\sigma_y)^2}{(n_1 + n_2)} + \frac{(n_2\sqrt{\alpha}\sigma_x - n_1\sqrt{\beta}\sigma_y)^2}{N\ n_1 n_2}$$

Thus, the variance of overall software reliability estimator can be given by:

$$\begin{aligned} Var(\hat{\eta}) &= \mathrm{Var}(P_1)(\theta_1 - \theta_2)^2 + \frac{(\sqrt{\alpha}\sigma_x + \sqrt{\beta}\sigma_y)^2}{(n_1 + n_2)} + \frac{(n_2\sqrt{\alpha}\sigma_x - n_1\sqrt{\beta}\sigma_y)^2}{N\ n_1 n_2} \\ &= \mathrm{Var}(P_1)(\theta_1 - \theta_2)^2 + \frac{\left(\sqrt{E(P_1^2)}\sigma_x + \sqrt{E(P_2^2)}\ \sigma_y\right)^2}{N} \\ &\quad + \frac{\left(n_2\ \sqrt{E(p^2)}\sigma_x - n_1\sqrt{E(P_2^2)}\sigma_y\right)^2}{N\ n_1 n_2} \end{aligned} \tag{3.7}$$

By selection of n_1, n_2, our goal is to minimize $Var(\hat{\eta})$ in above equation. Note the first two terms of (3.7) relies on the fixed total number of test cases,

the unknown operational profiles, and unknown software reliability estimators. Since they are fixed, we cannot manipulate it.

Hence, we can see that the variance is bounded by:

$$Var(\hat{\eta}) \geq \mathrm{Var}(P_1)(\theta_1 - \theta_2)^2 + \frac{\left(\sqrt{E(P_1^2)}\sigma_x + \sqrt{E(P_2^2)}\ \sigma_y\right)^2}{N} \tag{3.8}$$

with equality if:

$$\frac{n_1}{n_2} = \frac{\sqrt{E(P_1^2)}\sigma_x}{\sqrt{E(P_2^2)}\sigma_y} \tag{3.9}$$

By achieving the equality of (3.8), we obtain the variance incurred by the optimal allocation that is:

$$Var(\hat{\eta}) = \mathrm{Var}(P_1)(\theta_1 - \theta_2)^2 + \frac{\left(\sqrt{E(P_1^2)}\sigma_x + \sqrt{E(P_2^2)}\ \sigma_y\right)^2}{N} \tag{3.10}$$

In the next step, We need to find n_1, n_2. First, add 1 to both sides of equation (3.9), Then

$$1 + \frac{n_1}{n_2} = 1 + \frac{\sqrt{E(P_1^2)}\sigma_x}{\sqrt{E(P_2^2)}\sigma_y}$$

$$\frac{N}{n_2} = \frac{\sqrt{E(P_1^2)}\sigma_x + \sqrt{E(P_2^2)}\sigma_y}{\sqrt{E(P_2^2)}\sigma_y}$$

$$n_2 = N(\frac{\sqrt{E(P_2^2)}\sigma_y}{\sqrt{E(P_1^2)}\sigma_x + \sqrt{E(P_2^2)}\sigma_y})$$

$$n_1 = N - n_2$$

The optimal sampling strategy is not practical since the optimal allocation depends on the actual reliabilities, which are unknown. According to this issue, we chose to use dynamic allocation methods, which are explained in next two chapters.

CHAPTER 4

FULLY SEQUENTIAL ESTIMATION IN SOFTWARE RELIABILITY

4.1 Introduction

We will go through one of the sequential sampling approach for testing software and estimating software reliability. This sequential allocation refines learning about the software by allocating test cases based on previous outcomes throughout the testing process. In a fully sequential sampling approach, test cases were allocated to partitions one by one in each stage, with an allocation decision made after each test outcome. The purpose of this method is to find the near optimal distribution of test cases across subdomains that minimizes the variance of the overall software reliability estimator. This fully sequential approach is predicted to perform better and be more accurate than the balanced sampling strategy, which determines the number of test cases in advance for each partition.

4.2 Fully Sequential Method

This sequential allocation refines learning about the software by allocating test cases based on previous outcomes throughout the testing process. In a fully sequential sampling approach, test cases were allocated to partitions

one by one in each stage, with an allocation decision made after each test outcome. The purpose of this method is to find the near optimal distribution of test cases across sub-domains that minimizes the variance of the overall software reliability estimator. Since the actual conditional reliabilities θ_i in equation (2.9) for each sub-domain are unknown, we need to estimate its value during the testing process and choose which sub-domain to sample from to adjust the ratio such that

$\frac{n_{l,i}}{n_{l,j}}$ is close to $\hat{C}_{i,j}(l)$

where

$$\hat{C}_{i,j}(l) = \frac{\sqrt{E(p_i^2)}\sigma x_i}{\sqrt{E((p_j^2))}\sigma x_j} \tag{4.1}$$

The fully sequential design is outlined as follows:

Step1: Distribute l test case through partitions and estimate the reliability for each partition.

Step 2: After l tests have been allocated, where $l \geq k$,we take test $l+1$ from partition i if for all j:

$$\frac{n_{l,i}}{n_{l,j}} < \hat{C}_{i,j}(l) \tag{4.2}$$

where $n_{l,i}$ are the cumulative test cases allocated to partition i after l tests have been allocated and $n_{l,j}$ are the cumulative test cases allocated to partition j after l tests have been allocated and the estimated reliability for partition i is given by:

$$\hat{\theta}_{l,i} = \sum_{m=1}^{n_{l,i}} \frac{X_{im}}{n_{l,i}}$$

Step 3: We will sequentially apply step 2 until all N tests are allocated.

4.3 Comparison

We contrasted the balancing sampling, optimal sampling and fully sequential sampling methods using theoretical results which will be discussed in the next section and Monte Carlos simulations which will be discussed in the chapter 6. This comparison is done with a goal on minimizing the variance of the estimated overall software reliability. We show that a fully sequential method is effective even with a large number of test cases N. Furthermore, we expected that the fully sequential sampling method would outperform a balanced sampling method. This prediction of the fully sequential method is based on the allocation improvements that are feasible as we gain more knowledge of the program. We therefore have the chance to

learn from the errors made in selecting the distribution of test cases among partitions by using the results of previous tests. The distribution of test cases significantly affects the software reliability estimator's accuracy. As we mention in chapter 2, the first two terms of the next equation is fixed and independent of the partition sample size n_i and n_j.

$$\begin{aligned} Var(\hat{\eta}) = & \frac{\sum_{i=1}^{k}(\sqrt{E(P_i^2)}\sigma_{x_i})^2}{N} \\ + \frac{1}{N}\sum_{i=1}^{k-1}\sum_{j=i+1}^{k}(\frac{n_i\sqrt{E(P_j^2)}\sigma_{x_j} - n_j\sqrt{E(P_i^2)}\sigma_{x_i}}{n_i n_j})^2 + C \end{aligned} \tag{4.3}$$

The variance of a fully sequential method is obtained by selecting $n_1, n_2, \ldots, n_k$ such that the second term is forced to be zero in order to achieve the optimal.

$$Var(\hat{\eta}) \geq \frac{\sum_{i=1}^{k}(\sqrt{E(P_i^2)}\sigma_{x_i})^2}{N} + C \tag{4.4}$$

In the next section, we will present the theoretical result of comparisons between the fully sequential sampling and the optimal sampling. We also present results from Monte Carlo simulations that compare a fully sequential sampling method with a balanced sampling method in chapter 6.

4.4 Theoretical Result

We will show the theoretical result of comparisons between fully sequential sampling and the optimal sampling when N is large. Particularly, we aim to identify $Var\{\hat{\eta}_{Fully}\} - Var\{\hat{\eta}_{Optimal}\}$, where $Var\{\hat{\eta}_{Fully}\}$ is the variance incurred by the fully sequential sampling and $Var\{\hat{\eta}_{Optimal}\}$ is the variance observed by the optimal sampling. The next theorem quantifies the order of $Var\{\hat{\eta}_{Fully}\} - Var\{\hat{\eta}_{Optimal}\}$ for large N.

Theorem 1 The difference between the variance incurred by the fully Sequential Sampling Scheme and that incurred by the optimal sampling is an order of $\frac{1}{N}$.

Proof: We want to show that

$$\lim_{N\to\infty}[Var(\hat{\eta}_{fully}) - Var(\hat{\eta}_{optimal})] = 0$$

,where

$$\begin{aligned} Var(\hat{\eta}_{fully}) &= \frac{\sum_{i=1}^{k}(\sqrt{E(P_i^2)}\sigma_{x_i})^2}{N} \\ &+ \frac{1}{N}\sum_{i=1}^{k-1}\sum_{j=i+1}^{k}\left(\frac{n_i\sqrt{E(P_j^2)}\sigma_{x_j} - n_j\sqrt{E(P_i^2)}\sigma_{x_i}}{n_i n_j}\right)^2 + C \end{aligned} \tag{4.5}$$

and

$$Var(\hat{\eta}_{opt}) = \frac{\sum_{i=1}^{k}(\sqrt{E(P_i^2)}\sigma_{x_i})^2}{N} + C \tag{4.6}$$

Therefore,

$$[Var(\hat{\eta}_{fully}) - Var(\hat{\eta}_{optimal})] = \frac{1}{N}\sum_{i=1}^{k-1}\sum_{j=i+1}^{k}(\frac{n_i\sqrt{E(P_j^2)}\sigma_{x_j} - n_j\sqrt{E(P_i^2)}\sigma_{x_i}}{n_i n_j})^2 \tag{4.7}$$

We can prove the theorem by showing $equation(4.7) = 0 \qquad as \quad N \to \infty$. Hence, the proof follows if we can prove: $\frac{n_i}{n_j} = \hat{C}_{i,j}$ as $\quad N \to \infty$. To show the rest of the proof Using Rekab [15], let l be large enough and define l^i such that:

$$l^{(i)} = sup\{t < l : \frac{n_{i,t}}{n_{j,t}} < C_{i,j}(t) \quad forall \quad j \neq i\}$$

and $l^{(i)} \to \infty$ as $l \to \infty$. Then

$$\frac{n_{i,l}}{n_{j,l}} \leq \frac{n_{i,l^{(i)}} + 1}{n_{j,l^{(i)}}} \leq C_{i,j}(l^{(i)}) + \frac{1}{n_{j,l^{(i)}}}$$

. On the other hand,

$$\frac{n_{i,l}}{n_{j,l}} \geq \frac{n_{i,l^{(i)}} - 1}{n_{j,l^{(i)}}} \geq C_{i,j}(l^{(i)})(1 - \frac{1}{n_{j,l^{(i)}}})$$

The theorem follows by the strong law of large numbers.

This approach aims to develop an optimal sampling method by dynamic allocation among partitions as we get more insight into software performance. The benefit of this method is that we may get insightful information for each sub-domain and use it to improve the next test cases sequential allocation among partitions. The disadvantage of this method is that we must decide $l = N - K$ decisions to distribute the test cases and calculate our estimate. Additionally, the process should be automated because running test cases can be time-consuming and expensive. According to the disadvantage of this method, we will present the two-stages method in the next chapter.

CHAPTER 5

TWO-STAGES ESTIMATION IN SOFTWARE RELIABILITY

5.1 Introduction

We examine the same issue as in the previous chapter in this one: how test cases might be distributed among partitions. The allocation strategy's goal is to minimize the variance of the overall estimated software reliability in order to obtain the most accurate software reliability. We discussed the fully sequential sampling method in the last chapter, which calls for test cases to be run sequentially so that allocation decisions can be made after each test execution. A fully sequential sampling method can be difficult, expensive, and time-consuming in some situations. As a result, we suggested a different sequential sampling method that, in some circumstances, is more practical.

This chapter introduces a two-stage sampling strategy. In comparison to the fully sequential sampling method, the proposed method might be easier to implement. Because this two-stage sampling method is carried out in two stages, less computation is needed. According to the outcomes of the first stage, we used our method to determine the size of the second stage. We compared the outcomes of the two-stage sampling method with the balanced sampling method to show the effectiveness of the two-stage

method as determined by the variance of the overall estimated reliability.

As we know from the previous chapter, the optimal sampling method can only be theoretical and not practical because the software reliability for each partition θ_i is unknown. In a balanced sampling method, we decide in advance how to distribute test cases among partitions, so we cannot improve our distribution during testing. Due to the drawbacks of the balanced sampling method, we were motivated to use a dynamic allocation approach.

5.2 Two-stages Method

In a two-stage sampling method, we have two assumptions:

I: Total number of test cases is fixed.

$$N = n_1 + n_2 + \cdots + n_k$$

II: Each partition will be tested independently of the other partitions.

Our aim is to allocate the N test cases among k partitions to get the most accurate software reliability. This process was established by minimizing the

variance of the estimated software reliability.

$$Var[\hat{\eta}] = \sum_{i=1}^{k} Var[P_i\hat{\theta}_i] + 2\sum_{j=1}^{k}\sum_{i<j} Cov(P_i\hat{\theta}_i, P_j\hat{\theta}_j) \tag{5.1}$$

The equation above can be written as:

$$\begin{aligned} Var(\hat{\eta}) &= \frac{\sum_{i=1}^{k}(\sqrt{E(P_i^2)}\sigma_{x_i})^2}{N} \\ &+ \frac{1}{N}\sum_{i=1}^{k-1}\sum_{j=i+1}^{k}\left(\frac{n_i\sqrt{E(P_j^2)}\sigma_{x_j} - n_j\sqrt{E(P_i^2)}\sigma_{x_i}}{n_i n_j}\right)^2 + C \end{aligned} \tag{5.2}$$

From the above equations, the variance will be bounded below by:

$$Var(\hat{\eta}) \geq \frac{\sum_{i=1}^{k}(\sqrt{E(P_i^2)}\sigma_{x_i})^2}{N} + C \tag{5.3}$$

with equality if:

$$\frac{n_i}{n_j} = \frac{\sqrt{E(P_i^2)}\sigma_{x_i}}{\sqrt{E(P_j^2)}\sigma_{x_j}} \tag{5.4}$$

The optimal variance is attained when the equality of the previous equa-

tion can be satisfied. In order to minimize the variance of the software reliability estimator, we aim to determine the sample sizes for both n_i and n_j. In the following sections, we distribute the test cases into partitions by the two-stages method.

5.2.1 First Stage

In this stage, our initial test cases are chosen at random from each partition. Numerous studies have demonstrated that the anticipated initial sample size should be about $\sqrt{N}$ [14]. Let the initial sample size $L = \sqrt{N}$ be a sequence of positive numbers such that:

I: $\lim_{N\to\infty} L = \infty$.

II :L $\leq \frac{N}{k}$,where k is the number of partitions.

III: If $\lim_{N\to\infty} \frac{L}{N} = 0,$ which means that L is relatively small compared to N.

After allocating test cases for each partition, we can estimate the software reliability with the results of the small sample size by:

$$\hat{\theta}_i = \frac{\sum_{j=1}^{\sqrt{N}} X_{ij}}{\sqrt{N}}$$

Furthermore, based on the initial sample, the overall software reliability estimator is provided by:

$$\hat{\eta}_i = \sum_{i=1}^{k} p_i \hat{\theta}_i$$
$$= \sum_{i=1}^{k} [p_i \frac{\sum_{j=1}^{\sqrt{N}} X_{ij}}{\sqrt{N}}] \tag{5.5}$$

We can estimate θ_i based on the initial sample once the results of the test cases for each partition have been determined. A decision regarding how to distribute test cases in the second stage is significantly influenced by the findings of the first stage.

5.2.2 Second Stage

In this stage, the remaining test cases would be sequentially distributed among partitions using the knowledge we achieved from the first stage, to minimize the variance in overall software reliability. Since there are k subdomains, we want to know how many test cases there are for $n_1, n_2, \ldots, n_k$. So, the total of the remaining $N - k\sqrt{N}$ test cases would mathematically be such that:

$$\begin{cases} n_1 - \sqrt{N} & \text{For partition one} \\ n_2 - \sqrt{N} & \text{For partition two} \\ \dots & \\ n_k - \sqrt{N} & \text{For partition k} \end{cases}$$

The test cases for each partition will be chosen by combining the outcomes from the first stage with the optimal theoretical results. For each partition, the number of test cases n_i where $i = 1, \dots, k$ are calculated as follows:

$$n_{j,TwoStage} = N \frac{\sqrt{E(P_j^2)}\sigma_{x_j}}{\sum_{i=1}^{k} \sqrt{E(P_i^2)}\sigma_{x_i}}$$

where $j = 1, \dots, k-1$

$$n_{k,TwoStage} = N - \sum_{j=1}^{k-1} n_j$$

After we complete the second stage, we can determine the variance incurred by the two-stage sampling approach that is clarified:

$$Var[\hat{\eta}] = \sum_{i=1}^{k} Var[P_i\hat{\theta}_i] + 2\sum_{j=1}^{k}\sum_{i<j} Cov(P_i\hat{\theta}_i, P_j\hat{\theta}_j) \tag{5.6}$$

5.3 Comparison

We will compare between the balancing sampling, optimal sampling, and two-stages approaches using theoretical results, which are presented in the next section, and Monte Carlos simulations, which are covered in chapter 6. The aim of this comparison is to minimize the variance of the estimated overall software reliability. We demonstrate that a two-stage method still works well even with a large number of test cases N. Furthermore, we expected that the two-stages method would perform better than a balanced sampling method. This two-stage method prediction is predicated on allocation improvements that are doable as we learn more about stage one. As a result, we have the opportunity to learn from the mistakes made when choosing the test case distribution among the partitions in the first stage. Therefore, the distribution of test cases in the second stage significantly improve the software reliability estimator's accuracy. As we noted in chapter 2, the first two terms of equation (2.9) are fixed and independent of the partition sample size n_j where $j = 1, \ldots, k-1$ and n_k.

The variance of the two-stage method is obtained by selecting n_j and n_k such that the second term is forced to be zero in order to achieve the optimal.

$$Var(\hat{\eta}) \geq \frac{\sum_{i=1}^{k} (\sqrt{E(P_i^2)}\sigma_{x_i})^2}{N} + C \tag{5.7}$$

In the next section, we will show the theoretical result of comparisons between the two-stages method and the optimal sampling. We also will present results from Monte Carlo simulations that compare the two-stages sampling method with a balanced sampling method in the next chapter.

5.4 Theoretical Result

We will show the theoretical result of comparisons between two-stages sampling and the optimal sampling when N is large. Particularly, we aim to identify $Var\{\hat{\eta}_{2stages}\} - Var\{\hat{\eta}_{Optimal}\}$, where $Var\{\hat{\eta}_{2stages}\}$ is the variance incurred by the two-stages sampling and $Var\{\hat{\eta}_{Optimal}\}$ is the variance observed by the optimal sampling. The next theorem quantifies the order of $Var\{\hat{\eta}_{2stages}\} - Var\{\hat{\eta}_{Optimal}\}$ for large N.

Theorem 2 The difference between the variance incurred by the two-stage sampling method and that incurred by the optimal sampling is an order of $\frac{1}{N}$.

Proof: We want to show that

$$\lim_{N\to\infty}[Var(\hat{\eta}_{TwoStage}) - Var(\hat{\eta}_{optimal})] = 0$$

.

The proof will therefore follow if we demonstrate that the test cases for partition j are nearly equal to

$$\frac{\sqrt{E(P_j^2)}\sigma_{x_j}}{\sum_{i=1}^{k}\sqrt{E(P_i^2)}\sigma_{x_i}}$$

We are aware that the estimation of n_j should not be smaller than the initial sample and should not go beyond the remaining test cases for stage two, which is:

$$n_j = min\{N-(k-1)L, max\{L, n_j\}\}$$

. Next, divide both sides by N, then

$$\frac{n_j}{N} = min\{\frac{N}{N} - \frac{(k-1)L}{N}, max\{\frac{L}{N}, \frac{n_j}{N}\}\}$$

Now, using the estimate of n_j in above

$$\frac{n_j}{N} = min\{1 - \frac{(k-1)L}{N}, max\{\frac{L}{N}, \frac{\sqrt{E(P_j^2)}\sigma_{x_j}}{\sum_{i=1}^{k}\sqrt{E(P_i^2)}\sigma_{x_i}}\}\}$$

Hence, if we test a large number of test cases N, the proportion of $\frac{n_j}{N}$ reach

such as

$$\frac{\sqrt{E(P_j^2)}\sigma_{x_j}}{\sum_{i=1}^{k}\sqrt{E(P_i^2)}\sigma_{x_i}}$$

In the next chapter, we will use the Monte Carlo simulation to prove that the two-stage sampling scheme is much more effective than the balanced sampling scheme.

CHAPTER 6

MONTE CARLO SIMULATIONS

We consider the case in which the test domain is divided into two subdomains D_1, D_2, each with reliability θ_1 and θ_1. In each sub-domain we assume that the unknown usage probability P_1 and P_2 follow Beta distribution such that $P_1 \sim Beta(a, b)$, $P_2 \sim Beta(b, a)$. Before we start with a Monte Carlo Simulations, lets find from the next equation when test cases n_1 and n_2 are equal:

$$\frac{n_1}{n_2} = \frac{\sqrt{E(P_1^2)}\sigma_x}{\sqrt{E(P_2^2)}\sigma_y} = \frac{\sqrt{(a*b)/((a+b)^2*(a+b+1)) + (a/(a+b))^2}\sqrt{(\theta_1(1-\theta_1))}}{\sqrt{(a*b)/((a+b)^2*(a+b+1)) + (b/(a+b))^2}\sqrt{(\theta_2(1-\theta_2))}} \quad (6.1)$$

It is clear that $n_1 = n_2$ in the next two cases:

- $a = b$ and $\theta_1 = \theta_2$
- $a = b$ and $\theta_1 = 1 - \theta_2$

In the next two sections, we will show simulations for the fully sequential method and the two-stages method that we discussed in chapter 4 and chapter 5.

6.1 Monte Carlo Simulation For the Fully Sequential Method

In this section, we begin by showing the results of an experimental comparison between the fully allocation, the optimal allocation, and the balanced allocation in Table 1 through Table 4. In columns three, four and five of these tables, we show the estimated variance for the fully allocation sampling, the optimal allocation sampling and the balanced allocation sampling, while column six shows the ratio of the fully over the balanced. Column seven of these tables show the mean number of component 1 to be tested.

In Tables 5,6,7,and 8, we display the result of speed where

$$Speed = N * \{Var\{\hat{\eta}_{Fully}\} - Var\{\hat{\eta}_{Optimal}\}\}$$

In these tables, we show the different cases for a and b with $\theta_1 = 0.8$ and $\theta_2 = 0.9$ when N becomes very large.

Table 1 shows scenarios with uniform prior where $a = 1$ and $b = 1$. Given that this is the most typical prior parameter configuration, it can be concluded that little is actually known about the reliability of each partition. In these cases, except two cases in row 1 and row 2, the ratios less than 1 show that we still benefit from using the fully sequential method over the balanced allocation. The only two cases in which the balanced allocation is best when the reliability of partitions is $\theta_1 = \theta_2$ or $\theta_1 = 1 - \theta_2$ such as the case in row

one and row two of the Table 1. This happens simply because the optimal sampling for equal probabilities must fulfill

$$n_1 = n_2 = \frac{1}{2}N.$$

Table 2 through Table 4 represent a high expected reliability in each partition since $a = 1, 0.5, 0.1$ which is much larger than $b = 0.05, 0.01, 0.001$. This is a very likely scenario in reality and shows that high reliability in each sub-domain is assumed. The fully sequential sampling method outperformed very well over the balanced allocation under these situations, especially when the reliability for each subdomain is equal ($\theta_1 = \theta_2$).

The results in Tables 2 through 4 show that the fully sequential sampling consistently performs close to optimal sampling, regardless of the reliability or usage probability for each partition. As an example, the result of the fully in row six of Table 2 is .0053062230 , which is extremely close to the optimal .005305491 while the balanced is .008182202. Inferentially, this is what is anticipated from the fully method over the balanced method given that the fully sampling scheme dynamically employs information obtained during the testing process.

For example, consider Table 4:

$$\theta_1 = 0.4 \quad , \quad \theta_2 = 0.6$$

The ratio of the variance produced by the fully allocation to the variance produced by the balanced allocation is .616519908. The fully has average sample-size of n_1, n_2:

48 mean of component 1 to be tested and

2 mean of component 2 to be tested,

whereas the balanced allocation has an average sample-size of n_1, n_2:

25 each of mean of component 1 and component 2 to be tested.

The results of various Monte Carlo simulations of a and b for the particular case where $\theta_1 = 0.8$ and $\theta_2 = 0.9$ are shown in Tables 5 to 8, giving readers an indication of the speed at which this happens. When N becomes very large in Tables 5 to 8, the speed gets close to 0. This indicates that the fully sequential sampling gets close to optimal as the total number of test cases get large. For the sample sizes that are considered, the excess variance incurred by the fully over variance incurred by optimal is of the order of $\frac{1}{N}$.

6.2 Monte Carlo Simulation For the Two-Stages Method

We begin this section by presenting the results of an experimental comparison of the two-stages allocation, the optimal allocation, and the balanced allocation in Tables 9 through 12. The estimated variance for the two-stages

allocation sampling, the optimal allocation sampling, and the balanced allocation sampling are displayed in columns three, four, and five of the tables, respectively, while the ratio of the two-stages allocation sampling to the balanced sampling is displayed in column six. The mean number of component 1 tests is displayed in column 7 of these tables.

Tables 13, 14, 15, and 16 show the outcome of speed where

$$Speed = N * \{Var\{\hat{\eta}_{Two-stages}\} - Var\{\hat{\eta}_{Optimal}\}\}$$

In these tables, we display the various scenarios for a and b with $\theta_1 = 0.8$ and $\theta_2 = 0.9$ when N increases significantly.

Table 9 displays scenarios with uniform priors where $a = 1$ and $b = 1$. It can be inferred that little is actually known about the reliability of each partition due to the most typical prior parameter configuration. Except for the two cases in rows 1 and 2, the ratios less than 1 in these cases demonstrate that we continue to gain advantages using the two-stages method over the balanced allocation. When the reliability of partitions is $\theta_1 = \theta_2$ or $\theta_1 = 1 - \theta_2$, as in the cases of row one and row two of Table 9, the only two cases in which the balanced allocation is best. This occurs simply because $n_1 = n_2 = \frac{1}{2}N$ must be met for the best sampling to yield equal probabilities.

Tables 10 through 12 show a high level of expected reliability for each partition because $a = 1, 0.5, 0.1$ is a much larger value than $b = 0.05, 0.01, 0.001$.

The assumption of high reliability in each sub-domain is demonstrated by the fact that this is a very probable scenario in reality. Under these circumstances, especially when the reliability for each subdomain is equal ($\theta_1 = \theta_2$), the two-stages sampling method performed much better than the balanced allocation.

No matter the reliability or usage probability for each partition, the results in Tables 10 through 12 demonstrate that two-stage sampling is closer to optimal sampling than balanced sampling. For instance, the outcome of the two-stages in row one of table 11 is .002225688, which is close to the optimal .002183488 while the balanced is .003553253. Since the two-stages sampling scheme employs information obtained during the first stage, this is what is expected inferentially: for the two-stages method to do better than the balanced method.

For instance, consider Table 11:

$$\theta_1 = 0.9 \quad , \quad \theta_2 = 0.99$$

The ratio of the variance produced by the two-stages allocation to the variance produced by the balanced allocation is .59006776. The two stages have an average sample size of $n1$, $n2$:

43 on average for the component 1 that will be tested, and

7 on average for the component 2 that will be tested

While the balanced allocation has an average sample size of n_1, n_2:

25 on average for component 1 and component 2 to be tested.

Tables 13 to 16 give readers an idea of the speed at which this occurs by displaying the results of various Monte Carlo simulations of a and b for the specific case where $\theta_1 = 0.8$ and $\theta_2 = 0.9$. In Tables 13 to 16, the speed approaches zero when N grows extremely large. This shows that, as the total number of test cases increases, the two-stages sampling method gets closer to optimal level. For the sample sizes taken in to consideration, the excess variance incurred by the two-stages over variance incurred by the optimal is of the order of $\frac{1}{N}$.

Although these results of the study are convincing, a tester may have other goals in addition to minimizing variance. In reality, reliability testing may only serve as a barometer to assess the reaching of a milestone or being ready for release. Knowing precisely how unreliable a system is may be useless if it is evidently insufficient under these conditions. However, the methods described here hold significant promise for situations where reliability is close to meaningful levels.

CHAPTER 7

SUMMARY AND CONCLUSION

When estimating software reliability, the optimal sampling scheme is easily determined but it is not practical. It depends on unknown parameters such as the reliability of each strata and the usage probabilities. In order to minimize the optimal, we employed two methods to allocate the test cases among partitions. By dynamically allocating test cases to partitions, this method aims to minimize the variance of the reliability estimation. we presented a fully sequential sampling method

In this dissertation, we employed two methods to estimate the reliability of software system using partition testing. We have shown by theoretical results and Monte Carlo simulations that the fully sequential sampling method and two-stages sampling method perform at least as well as balanced sampling method, which allocates test cases beforehand, and is nearly optimal.

After providing some background information about reliability in the first chapter, we presented software reliability estimation for K partitions in the second chapter. We have shown the variance of estimated reliability when usage probabilities P_i are unknown using Rekab's Lemma. Also, we determined the optimal estimated software reliability. In the same way as chapter 2, we addressed software reliability estimation for two partitions in more detail in

chapter 3 and one assumption.

In the fourth chapter, we presented the first method which is a fully sequential sampling method to estimate the reliability of software systems using partition testing. Furthermore, we introduced the second method, two-stages sampling, in the fifth chapter. According to the fully sequential sampling method, test cases are allocated one at a time, allowing decisions to be made after each test is run, while the two-stages sampling method allocates test cases in two batches. As a result, it is highly implementable, less expensive, and still produces reliable results. We have shown in both chapters by theoretical results that both methods, the fully sequential sampling method and the two-stages sampling method, perform at least as well as the balanced sampling method, which allocates test cases beforehand.

We showed the Monte Carlo simulations for two partitions in chapter 6. In the first section, comparisons have been made between the fully sequential sampling method, the optimal allocation, and the balanced method. It is indicated that the fully sequential sampling method outperforms the balanced method. In the second section, we compared the two-stages sampling method, the optimal allocation, and the balanced method. It is shown that the two-stages sampling method performs better than the balanced method.

n our future studies, we will use a fully Bayesian approach allowing the conditional reliability of each strata to be randomly distributed with inde-

pendent priors and random usage probabilities. Other sampling schemes such as multistage and accelerated sampling schemes will be compared to the best fixed sampling scheme and to the optimal. The comparisons shall be carried both theoretically and numerically.

TABLES

Table 1: Comparison of Fully, Optimal, and Balanced allocation sampling
[Uniform prior $a = 1$, $b = 1$]

[N=50 ,and 1000 replication]

θ_1	θ_2	Fully	Optimal	Balanced	Ratio	E$\{n_1\}$
.1	.1	.0024	.0024	.0024	1	25.1
.4	.6	.009774557	.009733333	.009733333	1.004235343	23.9
.5	.7	.0094551280	.00945505	.009466667	.998781092	23.68
.5	.8	.0129273500	.0129	.01296667	.99696761	23.78
.5	.9	.0177884600	.0176	.01786667	.995622575	23.6
.7	.8	.0057478630	.00574404	.005766667	.996739191	24.66
.7	.9	.0072756410	.007166364	.007333333	.992132909	24.68
.8	.9	.0041346150	.0041	.004166667	.992307521	24.9
.9	.99	.001966346	.001738995	.002007	.979743896	25.34

Table 2: Comparison of Fully, Optimal, and Balanced allocation sampling
$[a = 1\ ,\ b = 0.05]$

[N=50 ,and 1000 replication]

θ_1	θ_2	Fully	Optimal	Balanced	Ratio	E$\{n_1\}$
.1	.1	.002266392	.00225832	.003432753	.660225772	45.4
.4	.6	.0069286190	.006907093	.01003891	.690176424	45.2
.5	.7	.0070178390	.007012827	.01038131	.676007074	44.88
.5	.8	.0079207200	.007919169	.01143866	.692451738	44.8
.5	.9	.0091847540	.00913253	.01291895	.710952051	44.42
.7	.8	.0053062230	.005305491	.008182202	.648507944	44.38
.7	.9	.0056853500	.005659081	.00877759	.647711957	44.02
.8	.9	.0039658150	.003961007	.006255605	.633961863	43.38
.9	.99	.0021199760	.002036239	.0035338	.599913974	41.68

Table 3: Comparison of Fully, Optimal, and Balanced allocation sampling
$[a = 0.5\ ,\ b = 0.01]$

[N=50 ,and 1000 replication]

θ_1	θ_2	Fully	Optimal	Balanced	Ratio	E$\{n_1\}$
.1	.1	.002183872	.002183488	.003553253	.614612019	47.06
.4	.6	.0063773540	.006331863	.009984569	.63872101	46.66
.5	.7	.0064907070	.006469629	.01035839	.626613499	46.68
.5	.8	.0069633030	.006961367	.0109687	.634833937	46.44
.5	.9	.0076106860	.007608121	.01182312	.64371215	46.4
.7	.8	.0050918570	.005088309	.008392	.606751311	45.94
.7	.9	.0052503830	.005244705	.008737199	.600922904	46
.8	.9	.0038261340	.003809875	.006407479	.597135629	45.36
.9	.99	.0020768860	.001993675	.003614351	.574622111	43.76

Table 4: Comparison of Fully, Optimal, and Balanced allocation sampling
$[a = 0.1\ ,\ b = 0.001]$

[N=50 ,and 1000 replication]

θ_1	θ_2	Fully	Optimal	Balanced	Ratio	E$\{n_1\}$
.1	.1	.002137929	.002136472	.003593525	.594939231	47.6
.4	.6	.006127518	.006053405	.009938881	.616519908	47.54
.5	.7	.0062479680	.00620481	.01032376	.605202756	47.5
.5	.8	.0065431240	.006531045	.01075094	.60860948	47.52
.5	.9	.0069563420	.006952974	.01134899	.612948113	47.26
.7	.8	.0049889570	.004964367	.008455926	.589995348	47.04
.7	.9	.0050460280	.005045904	.008697831	.580147855	46.88
.8	.9	.0037322700	.003723625	.006452321	.578438363	46.12
.9	.99	.00202979	.001967131	.003636804	.558124661	45.14

Table 5: Comparison of Fully and Optimal allocation sampling [$a = 1$, $b = 1$, $\theta_1 = 0.8$, $\theta_2 = 0.9$,and 1000 replication]

N	Fully	Optimal	speed
10	.0191666700	.0171666700	.0200000000
20	.0094865320	.0090000000	.0097306400
30	.0065178570	.0062777780	.0072023700
40	.0050689220	.0049166670	.0060902000
50	.0042094020	.0041000000	.0054701000
60	.0036401560	.0035555560	.0050760000
70	.0032352940	.0031666670	.0048038900
80	.0029325620	.0028750000	.0046049600
90	.0026976280	.0026481480	.0044532000
100	.0025100040	.0024666670	.0043337000
200	.001669084	.00165	.0038168000
500	.0011670450	.00116	.0035225000

Table 6: Comparison of Fully and Optimal allocation sampling [$a = 1$, $b = 0.05$, $\theta_1 = 0.8$, $\theta_2 = 0.9$,and 1000 replication]

N	Fully	Optimal	speed
10	.0199018300	.0189201300	.0098170000
20	.0100615300	.0095706780	.0098170400
30	.0067814280	.0064541940	.0098170200
40	.0050397960	.0048959520	.0057537600
50	.0040352440	.0039610070	.0037118500
60	.0033801160	.0033377110	.0025443000
70	.0029184780	.0028924990	.0018185300
80	.0025753390	.0025585900	.0013399200
90	.0023101070	.0022988830	.0010101600
100	.0020988680	.0020911170	.0007751000
200	.0011585420	.0011561720	.0004740000
500	.0005954183	.0005952048	.0001067500

Table 7: Comparison of Fully and Optimal allocation sampling [$a = 0.5$, $b = 0.01$, $\theta_1 = 0.8$, $\theta_2 = 0.9$,and 1000 replication]

N	Fully	Optimal	speed
10	.0227813200	.0185401500	.0424117000
20	.0107516400	.0093337270	.0283582600
30	.0066809120	.0062649200	.0124797600
40	.0050178680	.0047305170	.0114940400
50	.0039638220	.0038098750	.0076973500
60	.0032360050	.0031961140	.0023934600
70	.0027693260	.0027577130	.0008129100
80	.0024390740	.0024289120	.0008129600
90	.0021790810	.0021731780	.0005312700
100	.0019708100	.0019685910	.0002219000
200	.001048237	.001047949	.0000576000
500	.0004955822	.000495564	.0000091500

Table 8: Comparison of Fully and Optimal allocation sampling [$a = 0.1$, $b = 0.001$, $\theta_1 = 0.8$, $\theta_2 = 0.9$,and 1000 replication]

N	Fully	Optimal	speed
10	.0202781100	.0182619800	.0201613000
20	.0096692040	.0091755060	.0098739600
30	.0063789590	.0061466830	.0069682800
40	.0047731240	.0046322720	.0056340800
50	.0037914470	.0037236250	.0033911000
60	.0031357890	.0031178600	.0010757400
70	.0027213560	.0026851710	.0025329500
80	.0023728820	.0023606540	.0009782400
90	.0021116640	.0021082520	.0003070800
100	.0019066220	.0019063310	.0000291000
200	.000997731	.000997684	.0000093600
500	.0004524991	.000452496	.0000017000

Table 9: Comparison of Two Stages, Optimal, and Balanced allocation sampling [Uniform prior $a = 1$, $b = 1$]

[N=50 ,and 1000 replication]

θ_1	θ_2	2stages	Optimal	Balanced	Ratio	E$\{n_1\}$
.1	.1	.002405899	.0024	.0024	1.002457917	31.851
.4	.6	.009989469	.009733333	.009733333	1.026315343	22.864
.5	.7	.0094600560	.00945505	.009466667	.999301655	23.801
.5	.8	.0129380000	.0129	.01296667	.997788947	26.157
.5	.9	.0178000400	.0176	.01786667	.99627071	31.534
.7	.8	.0057571710	.00574404	.005766667	.998353295	25.675
.7	.9	.0072858240	.007166364	.007333333	.9935215	32.311
.8	.9	.0041432170	.0041	.004166667	.994372	31.87
.9	0.99	.001766679	.001738995	.002007	.880258595	41.551

Table 10: Comparison of Two Stages, Optimal, and Balanced allocation sampling [$a = 1$, $b = 0.05$]

[N=50 ,and 1000 replication]

θ_1	θ_2	2stages	Optimal	Balanced	Ratio	E$\{n_1\}$
.1	.1	.002313736	.00225832	.003432753	.674017618	39.988
.4	.6	.0073014310	.006907093	.01003891	.727313125	37.114
.5	.7	.0075048000	.007012827	.01038131	.722914545	37.474
.5	.8	.0085532520	.007919169	.01143866	.747749474	38.26
.5	.9	.0097339950	.00913253	.01291895	.753466419	39.712
.7	.8	.0057144850	.005305491	.008182202	.698404293	38.272
.7	.9	.0060807540	.005659081	.00877759	.692758946	39.886
.8	0.9	.0042097460	.003961007	.006255605	.672955853	40.102
.9	.99	.0022025940	.002036239	.0035338	.623293339	42.622

Table 11: Comparison of Two Stages, Optimal, and Balanced allocation sampling [$a = 0.5$, $b = 0.01$]

[N=50 ,and 1000 replication]

θ_1	θ_2	2stages	Optimal	Balanced	Ratio	E$\{n_1\}$
.1	.1	.002225688	.002183488	.003553253	.62638039	40.812
.4	.6	.0067431420	.006331863	.009984569	.675356342	39.108
.5	.7	.0069355380	.006469629	.01035839	.669557528	39.332
.5	.8	.0074492650	.006961367	.0109687	.679138366	39.74
.5	.9	.0081226580	.007608121	.01182312	.687014764	40.86
.7	.8	.0054382210	.005088309	.008392	.648024428	39.8
.7	.9	.0056434610	.005244705	.008737199	.645911922	40.948
.8	.9	.0040256200	.003809875	.006407479	.628268934	40.952
.9	.99	.0021327120	.001993675	.003614351	.59006776	42.776

Table 12: Comparison of Two Stages, Optimal, and Balanced allocation sampling [$a = 0.1$, $b = 0.001$]

[N=50 ,and 1000 replication]

θ_1	θ_2	2stages	Optimal	Balanced	Ratio	E$\{n_1\}$
.1	.1	.00223718	.002136472	.003593525	.62255863	40.856
.4	.6	.0066047490	.006053405	.009938881	.664536481	39.104
.5	.7	.0068141200	.00620481	.01032376	.660042465	39.352
.5	.8	.0071495260	.006531045	.01075094	.665014036	39.86
.5	.9	.0075374000	.006952974	.01134899	.664147206	40.996
.7	.8	.0054519840	.004964367	.008455926	.644753041	39.888
.7	.9	.0054905850	.005045904	.008697831	.631259104	40.892
.8	.9	.0039845970	.003723625	.006452321	.617544756	40.98
.9	.99	.0021001770	.001967131	.003636804	.577478742	42.724

Table 13: Comparison of Two Stages and Optimal allocation sampling [$a = 1$, $b = 1$, $\theta_1 = 0.8$, $\theta_2 = 0.9$,and 1000 replication]

N	2stages	Optimal	speed
10	.0174709100	.0171666700	.0030424000
20	.0092228890	.0090000000	.0044577800
30	.0065382790	.0062777780	.0078150300
40	.0051650980	.0049166670	.0099372400
50	.0044042220	.0041000000	.0152111000
60	.0038128540	.0035555560	.0154378800
70	.0034132140	.0031666670	.0172582900
80	.0031199520	.0028750000	.0195961600
90	.0028596910	.0026481480	.0190388700
100	.0026500900	.0024666670	.0183423000
200	.001736136	.00165	.0172272000
500	.0011887280	.00116	.0143640000

Table 14: Comparison of Two Stages and Optimal allocation sampling [$a = 1$, $b = 0.05$, $\theta_1 = 0.8$, $\theta_2 = 0.9$,and 1000 replication]

N	2stages	Optimal	speed
10	.0219423900	.0189201300	.0302226000
20	.0105000200	.0095706780	.0185868400
30	.0069086120	.0064541940	.0136325400
40	.0051865770	.0048959520	.0116250000
50	.0041522310	.0039610070	.0095612000
60	.0034956930	.0033377110	.0094789200
70	.0030123300	.0028924990	.0083881700
80	.0026624020	.0025585900	.0083049600
90	.0023840250	.0022988830	.0076627800
100	.0021594910	.0020911170	.0068374000
200	.0011785730	.0011561720	.0044802000
500	.0006003431	.0005952048	.0025691500

Table 15: Comparison of Two Stages and Optimal allocation sampling [$a = 0.5$, $b = 0.01$, $\theta_1 = 0.8$, $\theta_2 = 0.9$,and 1000 replication]

N	2stages	Optimal	speed
10	.0228064600	.0185401500	.0426631000
20	.0106595600	.0093337270	.0265166600
30	.0069550240	.0062649200	.0207031200
40	.0051744560	.0047305170	.0177575600
50	.0041364680	.0038098750	.0163296500
60	.0034081150	.0031961140	.0127200600
70	.0029098930	.0027577130	.0106526000
80	.0025422040	.0024289120	.0090633600
90	.0022661620	.0021731780	.0083685600
100	.0020444080	.0019685910	.0075817000
200	.001074416	.001047949	.0052934000
500	.0005038655	.000495564	.0041507500

Table 16: Comparison of Two Stages and Optimal allocation sampling [$a = 0.1$, $b = 0.001$, $\theta_1 = 0.8$, $\theta_2 = 0.9$,and 1000 replication]

N	2stages	Optimal	speed
10	.0248309800	.0182619800	.0656900000
20	.0106126800	.0091755060	.0287434800
30	.0069409220	.0061466830	.0238271700
40	.0050887550	.0046322720	.0182593200
50	.0040436420	.0037236250	.0160008500
60	.0033481380	.0031178600	.0138166800
70	.0028789780	.0026851710	.0135664900
80	.0025019390	.0023606540	.0113028000
90	.0022230910	.0021082520	.0103355100
100	.0019910810	.0019063310	.0084750000
200	.00102504	.000997684	.0054712000
500	.0004602199	.000452496	.0038619500

REFERENCE LIST

[1] Adrion, W. R., Branstad, M. A., and Cherniavsky, J. C. Validation, verification, and testing of computer software. *ACM Computing Surveys (CSUR) 14*, 2 (1982), 159–192.

[2] Al-Maati, S. A., and Rekab, K. Dynamic test allocation model for software reliability. In *Third International Conference on Quality Software, 2003. Proceedings.* (2003), IEEE, pp. 26–31.

[3] Cochran, W. G. Sampling Techniques, 3rd Ed. New York, NY (USA) Wiley, 1977.

[4] De Millo, R. A., Lipton, R. J., and Sayward, F. G. Hints on test data selection: Help for the practicing programmer. *Computer 11*, 4 (1978), 34–41.

[5] Frankl, P. G., and Weyuker, E. J. A formal analysis of the fault-detecting ability of testing methods. *IEEE Transactions on Software Engineering 19*, 3 (1993), 202–213.

[6] Hardwick, J. P., S. O. F. Optimal allocation for estimating the mean of a bivariate polynomial. *Sequential Analysis 15*, 2-3 (1996), 71–90.

[7] Howden, W. E. Functional program testing. *IEEE Transactions on Software Engineering*, 2 (1980), 162–169.

[8] Jesse H. Poore, H. D. M., and Mutchler., D. Planning and certifying software system reliability. *IEEE Software 10*, 1 (1993), 88–99.

[9] Kaner, C., et al. The impossibility of complete testing. *Software QA 4*, 4 (1997), 28.

[10] Nead, N. 10 examples of software development failure. ReadWrite, 2020.

[11] Neyman, J. On the two different aspects of the representative method: the method of stratified sampling and the method of purposive selection. In *Breakthroughs in Statistics*. Springer, 1992, pp. 123–150.

[12] Rapps, S., and Weyuker, E. J. Selecting software test data using data flow information. *IEEE Transactions on Software Engineering*, 4 (1985), 367–375.

[13] Rekab, K. Asymptotic efficiency in sequential designs for estimation. In *Sequential Analysis* (1989), Taylor Francis, pp. 269–280.

[14] Rekab, K. A nearly optimal two stage procedure. *Communications in Statistics - Theory and Methods 21*, 1 (1992), 197–201.

[15] Rekab, K. Sequential allocation for estimation problem. *Stochastic Analysis and Applications 10*, 4 (1992), 465–470.

[16] Rekab, K. A sampling scheme for estimating the reliability of a series system. *IEEE transactions on reliability 42*, 2 (1993), 287–290.

[17] Rekab, K., and Cheng, Y. An accelerated sequential sampling for estimating the reliability of N-parallel systems. *International Journal of Reliability and Applications 14*, 2 (2013), 71–78.

[18] Rekab, K., and Tahir, M. A two-stage sequential allocation scheme for estimating the product of several means. *Stochastic Analysis and Applications 18*, 2 (2000), 289–298.

[19] Rekab, K., Thompson, H., and Wu, W. A fully sequential test allocation for software reliability estimation. *Journal of Applied Statistical Science 20*, 1 (2012), 63.

[20] Rekab, K., Thompson, H., and Wu, W. An efficient test allocation for software reliability estimation. *Applied Mathematics and Computation 220* (2013), 94–103.

[21] Rekab, K., Thompson, H., and Wu, W. A multistage sequential test allocation for software reliability estimation. *IEEE Transactions on Reliability 62*, 2 (2013), 424–433.

[22] Richardson, D. J., and Clarke, L. A. A partition analysis method to increase program reliability. In *Proceedings of the 5th International Conference on Software Engineering* (1981), IEEE Press, pp. 244–253.

[23] Sayre, K., and Poore, J. H. Partition testing with usage models. In *Proceedings. Science and Engineering for Software Development: A Recognition of Harlin D. Mills Legacy (Cat. No. PR00010)* (1999), IEEE, pp. 24–30.

[24] Shapiro, C. Allocation schemes for estimating the product of positive parameters. *Journal of the American Statistical Association 80*, 390 (1985), 449–454.

[25] Shapiro, C., and Robert, W. Simultaneous bayesian sequential estimation. *Journal of the American Statistical Association 75*, 370 (1980), 359–365.

[26] Walter, G. J. Optimal test distributions for software failure cost estimation. *IEEE Trans. Software Engineering 21*, 3 (1995), 219–228.

[27] Walton, G. H., Poore, J. H., and Trammell, C. J. Statistical testing of software based on a usage model. *Software: Practice and Experience 25*, 1 (1995), 97–108.

[28] Weyuker, E. J. O. T. J. Theories of program testing and the application of revealing subdomains. *IEEE Trans. Software Eng SE-6* (1980), 236–245.

[29] Whitaker J.A., T. M. A Markov chain model for statistical software testing. *IEEE Trans. Software Eng* (1994), 812–824.

[30] Zarzour, N., and Rekab, K. Sequential procedure for software reliability estimation. *Applied Mathematics and Computation 402* (2021), 126116.

VITA

YOUSEF ALHARBI was born on April 24, 1987, in Saudi Arabia. On the 15th of May 2010, he received a B.S. in Mathematics from Qassim University in Saudi Arabia. He received a M.S. degree in Applied Mathematics from Kent State University in 2017. He received a M.S. degree in Statistics from the University of Missouri-Kansas City in 2021.