SIMULATION PROGRAM FOR ASSESSING THE RELIABILITIES OF COMPLEX SYSTEMS (SPARCS)

By

JOHN WAYNE COOLEY

Bachelor of Science
McNeese State University
Lake Charles, Louisiana
1968

Master of Business Administration
Lamar State University
Beaumont, Texas
1970

Submitted to the Faculty of the Graduate College
of the Oklahoma State University in partial
fulfillment of the requirements
for the Degree of
DOCTOR OF PHILOSOPHY
May, 1976

Thesis
1976D
C7745s
Cop. 2

SIMULATION PROGRAM FOR ASSESSING THE RELIABILITIES OF COMPLEX SYSTEMS (SPARCS)

Thesis Adviser:

Mitchell O. Locks
Thesis Adviser

Donald W. Grace

[illegible]

[illegible]

Joe H. Mize

[illegible]

N N Durham
Dean of the Graduate College

964131

ACKNOWLEDGMENTS

The author wishes to express his appreciation to his major adviser, Dr. Mitchell O. Locks, for his guidance and assistance throughout this study. Appreciation is also expressed to the other committee members, Dr. Wayne Meinhardt, Dr. William Warde, Dr. Donald Grace, Dr. Joe Mize and Dr. James Shamblin, for their helpful and constructive comments in the preparation of the final manuscript.

A note of thanks is given to the O.S.U. computer center for their assistance in the preparation of the programming model. Thanks are also extended to Ms. Patty Barry for her assistance in typing earlier drafts of the manuscripts and to Ms. Ilze Zigurs Powers for the excellence of the final copy and her grammar and form corrections and suggestions.

Finally, a special gratitude is expressed to my wife, Belva, for her encouragement, understanding, and many sacrifices.

TABLE OF CONTENTS

Page

LIST OF TABLES

LIST OF FIGURES

CHAPTER I

INTRODUCTION

A system is a configuration of components combined to perform a particular task. The reliability of a component or system is the probability that the device will work successfully. To analyze a system, this analysis must be based upon an analysis of the components with regards to their configuration within that system.

The system reliability depends upon the reliability of the components that make up that system. In analyzing the reliability of a system or component, a confidence level is associated with each reliability since the actual reliability of an item cannot be determined precisely. This confidence level provides a measure of the quality of the reliability estimate.

SPARCS* (Simulation Program for Assessing the Reliability of Complex Systems) is a program that provides interval estimates for assessing the reliability of complex systems. The system components consist of two component types: Bernoulli components (attribute type) and Poisson process components (time-to-failure type). The model uses information about the logical configuration of a system in the form of success states or failure states and failure-history

*SPARCS was developed in conjunction with a grant from the Air Force System Command, Wright-Patterson Air Force Base, in Dayton, Ohio as project number F33615-74-C-4077. The result of that contract is technical report AFFDL-TR-75-144 [62].

data for each component as input. System mission time may also be input as an option to obtain a MTBF (mean-time-between-failure) for the system.

The system logical information is analyzed using Poincaire's theorem (the method of inclusion-exclusion) to provide a system equation as a function of the system components. Each component's failure-history data is used to provide component reliability or unreliability values for use with the system equation. This failure-history data are parameters of Bayesian conjugate prior distributions on the component reliabilities. A beta distribution is used for the Bernoulli process components and a negative-log gamma distribution is used with the Poisson process components.

SPARCS is an efficient procedure written in PL/1 that uses Monte Carlo methods to provide an "empirical" distribution on the system reliabilities (or unreliabilities) and the system MTBF. This is accomplished for a system of any logical configuration and complexity. The procedure involved is facilitated by the use of modularization which allows large systems to be broken down into smaller independent modules which may be analyzed separately and later combined.

Purpose and Scope of the Study

The purpose of this study is to provide a computerized procedure for the determination of confidence bounds and appropriate limits for the reliability (or unreliability) of a complex system of any logical configuration. The scope of this study is limited to the development of such a procedure. In particular, a system equation will be developed which is a function of the component reliabilities and their logical placement within the system. The components will be of two basic types: Bernoulli components and Poisson process components. Monte Carlo Techniques will be used, in conjunction with this equation, to provide point estimates for the system reliability. These point estimates will be ordered and analyzed statistically to provide empirical confidence bounds and limits on the (un)reliability of the system under analysis. The mean, variance and standard deviation as well as an estimated reliability for the system will be provided.

Methodology

Since there is usually no failure-history data available for the system under evaluation, failure-history data for each component is used based upon the best available data. This information is supplied to SPARCS along with information concerning the logical configuration of the system components and a "mission time" for determination of a system MTBF if desired.

Poincaire's Theorem (inclusion-exclusion) is used to generate an equation for the system as a function of the system components and their placement within the system. Component (un)reliabilities are provided for this equation based upon the historical component test information supplied by the user.

System components are of two basic types: Bernoulli (pass-fail) components and Poisson process (time-to-failure) components. The model uses the component failure-history data to provide reliability confidence assessment for a system containing any logical combination of either or both types of these components. For each component type, Bayesian analysis is used to provide the (un)reliability for that component. For Bernoulli components, the Bayesian prior is the beta with accumulated successes and failures as sufficient statistics. For Poisson process components, the Bayesian prior is the negative-log gamma with accumulated total time in tests per test unit and accumulated failures as sufficient statistics. These sufficient statistics are parameters of the beta prior and the negative-log gamma prior.

Monte Carlo techniques are used to enter the appropriate Bayesian prior distribution to provide an estimate of the reliability for that component. The Monte Carlo techniques utilize the historical test data as sufficient statistics when entering the appropriate distribution. Each component reliability is placed in the reliability equation in its proper position. This function is then evaluated to yield a point estimate for the reliability of that particular system. The Monte Carlo procedures produce a number of these point estimates which are used to provide confidence limits and statistical information on the empirical distribution of these reliability point estimates.

These system reliability point estimates are sorted in increasing order. Percentage points are provided by an analysis of these ordered values. The mean, variance and standard deviation of this empirical distribution is determined. An estimated reliability for the system is also calculated by placing the mean value of each component into the system equation. If the mean-time-between-failures (MTBF) is desired, the MTBF is presented for each percentage point value by direct conversion of that value into an MTBF.

Background

A program by J.L. Burris, called Model for the Analysis of the Probabilities of Systems (MAPS) [11] provides the system equation, using Poincaire's method, and the basic input-output format. SPARCS, Simulation Program for Assessing the Reliabilities of Complex Systems, provides the Monte Carlo techniques and statistical techniques necessary to develop and analyze the empirical distribution of system point estimates. These point estimates can be generated for a system of any logical configuration: series, parallel, or series-parallel. Complex systems may be broken down into smaller subsystems (modules) which are later combined to determine the system reliability or unreliability. This modular idea along with the use of PL-1, makes it possible to handle complex systems with a considerable saving of time and computer storage.

Chapter Organization

Chapter II discusses the pertinent literature around which the model revolves. Current methods for assessing the reliability of simple systems and Monte Carlo methods for reliability confidence assessment are discussed. Literature concerning concepts around which SPARCS revolves is presented such as Poincaire's Theorem and Bayesian reliability analysis.

Chapter III discusses the system and logical aspects used in SPARCS. Poincaire's Theorem and its development is analyzed.

Chapter IV discusses the statistical distributions used in the model. A brief discussion of the beta and negative-log gamma distributions are presented. The implementation and reason for implementation of the uniform prior is analyzed along with a discussion of the statistical aspects of confidence bounds and confidence limits used in reliability.

Chapter V describes the tests and analysis used for model validation. The duality concept, tests of the uniform prior for component (un)reliability and some simple binomial and exponential tests are analyzed. These tests show that the concepts are intact in the model and that the model does produce very good results as compared to other results in the field.

Chapter VI analyzes some of the techniques and procedures incorporated into the model. The International Mathematical and Statistical Library (IMSL) routines that provide component reliabilities are checked for inherent error. Tests of the pseudo-random number generators and the sorting routine incorporated into the model are described. Finally, a discussion of sample size determination is presented.

Chapter VII discusses the model software procedures. An analysis of the storage requirements and the purpose of each procedure is presented. The JCL aspects are also discussed to facilitate system transitions.

Chapter VIII is a documentation of SPARCS. A discussion of what the model does as a composite unit is presented. The input format is explicitly delineated to enhance user use.

Chapter IX summarizes the model methodology and test conclusions. A small section lists possible extensions of this work.

Finally, four appendices are at the end of the chapters. The first two appendices present the JCL used with the model. The next two are a program source listing of SPARCS and an Apollo Lunar Excursion Module (LEM) test run.

CHAPTER II

LITERATURE REVIEW

Introduction

In reliability confidence assessment, prediction statements are made concerning the reliability of a system from life test data accumulated for each system component. The reliability of a system is associated with a probability that shows the confidence of the reliability estimates. Estimation of the reliability of a system that provides no confidence or predictive value for the reliability of that system ignores the possibility of variability in these estimates.

This chapter reviews the literature that deals with procedures for assessing the reliability of systems. Included in this review is a historical development of techniques and statistical descriptions of procedures that are employed in systems reliability confidence assessment.

A History of Reliability Confidence Assessment

This section of the literature review traces the early development of reliability estimation and assessment techniques. Early articles and books on reliability are reviewed dealing with both component and system reliability analysis.

Early Articles

In 1953, Epstein and Sobel [23] write a classic article on reliability assessment for components, dealing with the exponential distribution. Life testing procedures are proposed for estimating the reliability of exponential type components. Their procedure contends that only r out of n component failures need to occur within a specified testing time, where $r < n$, to provide an estimate of the component reliability. Assessment is performed usint the Chi-square distribution with 2r degrees of freedom, where r is the number of failures.

In 1957, Buehler [8] and Steck [87] publish articles which consider assessing the reliability of simple systems as well as a technique for single component reliability assessment. In each case, binomial components are considered. Buehler [8] provides confidence limits on a system of two binomial independent components which are linked in a parallel configuration. A Poisson approximation to the binomial distribution is used and his analysis is specialized to small probabilities of failure and moderate sample sizes. Steck [87] proposed a more general solution to the problem. His solution requires an ordering of component test results that produces complex manipulations for all but simple systems. In each case, reliability analysis was applied to systems of components.

In 1963, Rosenblatt [76] uses a U-statistic as discussed by Hoeffding [35] to analyze a simple binomial system. This article begins to hint at analysis of systems of a more complex nature in which the components may be either series or parallel or a combination.

Early Books and Tables

The first text explicitly dealing with the subject of reliability was written by Bazovsky [3] in 1961. Bazovsky provides discussions of network analysis, component reliability assessment and simple system reliability estimations. In 1962, Lloyd and Lipow [47] write a text on reliability which used approximations such as the Poisson approximation to the binomial as developed by Buehler [8] to produce confidence bounds on the system reliability. This was used in lieu of methods which combined confidence bounds on the components to obtain confidence bounds on the system as proposed by Conner and Wales [15]. Earlier, Lipow and Riley [46] had tabled upper confidence limits on 1, 2, and 3 component serial systems.

Early Monte Carlo Techniques

In the late 50's and early 60's, system reliability analysis was approached using Monte Carlo techniques. The earlier techniques consisted of simulating the success or failure of each component as events. These component success or failure events were then combined logically to see if the system succeeded or failed. However, little information is written describing these early techniques.

Simple System Reliability Assessment

In the mid 60's, the literature begins to expand. The earlier articles on component and simple system reliability analysis are extended by use of approximations and exact expansions. However, most of the literature continues to deal with assessing the reliabilities of simple series or parallel systems of exponential and binomial type components. Since SPARCS also deals with both exponential and binomial type components, the literature interest is channeled in that direction.

Binomial Systems

Confidence limits for systems consisting of binomial type subsystems of more than two components are discussed by Madansky [52] in 1965. Madansky uses a maximum likelihood ratio test in lieu of the Poisson approximation suggested by Buehler [8]. However, his procedure did not obtain reliable values for systems with high reliabilities. The Poisson approximation of Buehler produced much better values in these cases. Consequently, his procedure is applicable only to systems with moderate reliabilities.

Since Buehler's method is developed for systems with two binomial components and Madansky's procedure does not produce good results for highly reliable systems, Harris [33] tries to devise a method to provide confidence limits for systems of more than two components which will produce adequate results for highly reliable systems. To accomplish this, Harris uses the Poisson approximation to the binomial distribution in conjunction with a uniform random variate to produce confidence limits for systems of more than two binomial components. An article by Myhre and Saunders [67] used by Harris [33], succinctly analyzes the method of Madansky.

Springer and Thompson [83] are one of the earliest to try the Bayesian approach to binomial component systems. A Bayesian prior distribution, which is uniform in the abpence of data, is applied to the system under analysis. A transform is applied to each component and the results combined to produce confidence limits on the system reliability.

In 1972, Easterling [19] develops a procedure which uses a maximum likelihood estimate of the system reliability. The maximum likelihood estimates are substituted into an incomplete beta function to obtain confidence limits on the reliability of the system of binomial components. Mann [54] produces a basic simplification of Buehler's [8] article which removes the two component restriction on system size. For systems of more than two components, the Wilson-Hilferty [93] transformation to the chi-square is used to provide a standard normal variate for system reliability confidence assessment. Winterbottom [94] provides a comparative study of exact and approximate methods for providing lower confidence limits on the reliability of binomial systems. Exact methods are methods that do not use approximations in their techniques to facilitate calculations. Approximation methods revolve around the use of approximation procedures such as chi-square approximations, normal approximations, the Wilson-Hilferty transformation and others. Thus, approximate methods are ways of approaching exact results which are used as a standard.

Exponential Systems

Confidence intervals for exponential component systems are discussed by Lentner and Buehler [44] in 1963. Life testing procedures are applied to these exponential type components as developed by Epstein and Sobel [23]. By defining fixed "mission times," gamma variates are used to provide a linear function of more than two parameters which are analyzed through the use of "similar regions" as described in Lehmann and Scheffé [42] and Lehmann [41].

In the mid 60's, El Mawaziny [21] expands the work of Lentner and Buehler [44] to produce explicit expressions for an exponential type system of any size. A linear combination of incomplete gamma functions is used to derive confidence limits on exponential systems by elaborate computer techniques. Later, El Mawaziny and Buehler [22] provide a large scale approximation to El Mawaziny's procedure. This approximation follows El Mawaziny's [21] idea of no restrictions on the number of system components with each component following exponential failure laws.

Springer and Thompson [84, 85] provide an extensive analysis of exponential type components in parallel configuration. Bayesian confidence limits are placed on redundant exponential systems from component test data in which component tests are terminated at the first failure. The analysis is for components and systems having extremely high reliabilities. Later, Thompson and Chang [89] generalize the technique of Springer and Thompson [85] to remove the restriction of the single life sample with termination at the first failure.

In 1971, Leiberman and Ross [43] expand the work of Kraemer [38] and Sarkar [77] to provide lower confidence limits on systems of two independent exponential components. Analysis of the two exponential components are shown to produce a distribution for the system reliability that approximates a Gamma distribution.

Grubbs [32] develops a process which provides a lower limit on the system reliability for systems consisting of exponential time-to-failure components using the number of component failures in specified "mission times." His method is designed to be used in lieu of methods involving Monte Carlo simulation techniques. Grubbs' method uses the first two moments of the "fiducial" distribution of the system failure rate to fit a non-central Chi-square distribution. His method requires a minimum of calculation and uses tables of standard normal deviates to obtain the system lower confidence limits.

Mann and Grubbs [56] combine the earlier methods of El Mawaziny [21], Lentner and Buehler [44] and Grubbs [32] to propose a simple method to approximate the system lower confidence limits for exponential series systems. The general results supplied by Patnaik [71] concerning the noncentral Chi-square approximation and the Wilson-Hilferty transformation [93] are used in conjunction with Fertig [27] and Cox [18] to provide these lower limits.

The "approximately optimum" method of Mann and Grubbs [56] is later simplified by Mann [55]. Essentially, the process uses a transformed chi-square probability density function and the moments of this function to provide an approximation that tends to agree within approximately a unit in the second decimal place with the method of El Mawaziny [21] which is considered an "exact" method.

Complex Systems

Complex systems are systems with other than strict series or parallel configuration in which the component types may be intermixed. Generally, systems are restricted to either all exponential or all binomial components in series or parallel configuration. Some literature intimates that their procedures may be extended easily to include complex systems but never actually follow through with such an explanation.

In the early 60's Rosenblatt [76] hints at an expansion of her method to a more logically complex system as does Mann and Grubbs [57] and Wolf [95] later. However, no formal details are presented. Mann and Grubbs propose the application of simplified approximations to a complex system of "mixed" components by finding equivalent Beta or binomial transformations for their simplified exponential computations. However, the requirement that a complex system be expressed as a series or parallel system composed of more series and parallel components restricts their calculations. Nowhere in the literature was there found an explicit analysis that purported to place confidence limits on a complex system of any logical configuration using "mixed" historical component information in any order with the exception of an article by Levy and Moore [45].

Monte Carlo Techniques

In 1961 and 1962, Burnett and Wales [10] and Bosinoff and Klion [7] proposed Monte Carlo techniques for system reliability assessment in which component life distributions are used to provide component reliabilities. These reliabilities are placed in a system reliability equation to provide interval estimates on the system reliability through repeated Monte Carlo trials. The basic assumption in each instance is that the components have exponential life distributions, are all connected in series, and are independent.

These basic simulation assumptions are still used. Generally, Monte Carlo analyses still assume simple series or parallel systems in which all components are either exponential or binomial type components with the exception of Levy and Moore [45].

Levy and Moore [45] analyze a system which is not a strict series or parallel system with either binomial or exponential type components. Their components are a mixture of Weibull, normal, lognormal, exponential and Gamma type components in a complex system of seven components. A group of values are provided for each component. These values are ordered to form an "empirical" distribution. Then, random numbers are used to enter these "empirical" distributions to obtain component reliabilities. The complex system is broken down into easily manipulated parallel or series subsystems. These subsystems are combined to form either a simple series or parallel system which can be analyzed with relative ease.

In the 70's, Mann [55] and Berkbigler and Byers [4] use Monte Carlo techniques to analyze the effect prior distributions have on the lower confidence limits of the system reliability. However, in each case, a simple series system is used to provide the analysis.

Development of the SPARCS System Reliability Assessment Model

Early Minimal State Analysis

In 1956, Moore and Shannon [64], inspired by a paper presented the same year by von Neuman [91], develop methods for producing highly reliable systems from components of low reliability. This paper set the framework for minimal state analysis of complex systems. Moore and Shannon provide bounds on the number of components needed to achieve a specified system reliability and initially develop the concept of minimal state analysis. They show that the reliability of a network consisting of independent components of equal reliability is S-shaped. In 1959, Mine [61] further expands these procedures by examining complex systems which are represented as Boolean functions.

In 1962 and 1963, Birnbaum, Esary and Saunders [6] and Esary and Proschan [24, 25], expand the work of Moore and Shannon [64]. Birnbaum, Esary and Saunders explore the reliability of complex systems in which the reliability of each composent is the same. Esary and Proschan extend this concept to systems in which the reliability of the components are not necessarily analogous. In each case, minimal paths, defined as "a smallest set of components which by functioning cause the system to function" [24], are used to provide an upper bound on the system reliability. Minimal cuts, defined as "a smallest set of components which by failing cause the structure to fail" [24], are used to furnish a lower bound on the system reliability. In 1965, Barlow and Proschan [2] also enlarge this minimal state concept by further examination of coherent systems, i.e. structures which have the property that replacing failed components with functioning components cannot cause a funcioning structure to fail.

System Reliability Equation Development Using Poincaire's Theorem

In 1971, Locks [48] uses Poincaire's Theorem, based in part on the theory of inclusion-exclusion discussed in Feller [26, pp.26ff], in conjunction with the minimal state definitions and concepts of the early articles mentioned above, to develop an exact system (un)reliability equation which is a function of the system components.

The minimal states of the system under investigation are used to obtain a polynomial that represents the reliability of unreliability of the system as a function of success states (minimal paths) or failure states (minimal cuts). The system reliability or unreliability estimates can be obtained for a system of any size and any logical configuration.

A complete description of the use of Poincaire's Theorem for developing the exact system reliability equation as a function of the system minimal states is presented by Locks [48]. Locks [49] also presents an error analysis between his exact method for providing upper and lower bounds on the (un)reliability of a system and the earlier minimal state methods for forming the upper and lower bounds.

Earlier Computer Models for System Reliability Analysis

In the late 60's, a program called SCOPE (System for Computing Operational Probability Equations) [88] was developed for the Saturn and Apollo space programs. This program was the basis for MAPS (Model for the Analysis of the Probabilities of Systems) [11], developed in 1972 by J. L. Burris. MAPS is coded in PL/1 as opposed to the FORTRAN coding of SCOPE and incorporates a modularity concept that allows large systems to be broken down into smaller subsystems.

MAPS is a computer program designed to produce a point estimate of the reliability of a complex system as a function of the reliabilities of the components that make up that system. An analysis of the system network by the user provides the minimal states for the system. These minimal states are used as input to generate an equation for the system as a function of the component reliabilities or unreliabilities. An estimate of the reliability (or unreliability, depending upon the type of analysis desired by the user) is input for each component. These component reliability (unreliability) values are then substituted into the reliability (or unreliablIity) equation to produce an estimate of the reliability (unreliability) of the system. Parts of this program were used as a base for the development of SPARCS [61].

Sample Size Determination

A formal method for sample size Determination has not been incorporated into the model. Burdick and Naylor [9] and Naylor, Balintfy, Burdick and Chu [68, pp. 335-338] discuss the sample size determination problem as one of the major problems in simulation. When the data to be analyzed lack independence and normality, an efficient method for the determination of how many observations to measure and when to begin measurement becomes very difficult. Without some knowledge of the types of distributions obtained from analysis of systems of different configurations, the sample size cannot be efficiently determined.

Consequently, the law of large numbers and the Central Limit Theorem are used to provide an estimate of the number of simulation runs necessary for a certain confidence interval about the mean. Although this basic sample size formula is provided for use with the model, it will be shown that SPARCS provides very good results with reasonably small sample sizes. These small sample size values are compared with values obtained from larger samples obtained from literature and verified using a duality check. The reason for these results with small sample sizes may revolve around the idea that conventional sample size procedures are based upon the sampling of events whereas SPARCS in fact samples reliabilities.

Statistical Development of the Model

System reliability confidence assessment may be approached through the use of three statistical procedures: Classical analysis, Bayesian analysis, and fiducial analysis. Since system reliability assessment is a function of the components that make up that system, these procedures revolve around a method for analysis of the system components.

In the classical approach, prior information is not taken into account and prediction limits are placed around an estimate of the true reliability. These limits provide a true frequency interpretation not produced by the other two methods [39, 69]. The Bayesian procedure [1, 30] and the fiducial procedure [28, 36] take into consideration prior data and knowledge plus the statistician's personal assessment of this prior knowledge. The Bayesian analysis generally uses an ignorance (uniform) prior as the basis for any resultant posterior distribution. The fiducial analysis was introduced by R. A. Fisher [28]. One of the basic differences between fiducial priors and uniform prior revolves around the idea that fiducial priors assume prior experience with this experience being used as a base. The uniform prior, in the absence of data, uses the assumption of no prior knowledge (ignorance). The difference between Classical analysis and Bayesian analysis (including fiducial analysis) is succinctly summarized by Springer and Thompson [83]. The confidence limits in the Bayesian sense are defined such that the probability of a particular estimate lying outside these limits will not exceed the specified posterior probability. In the Classical sense, as developed by Neyman [70], the confidence limits stipulate that the frequency with which prediction lies outside these confidence limits will not, in the long run, exceed the specified confidence. Consequently, limits obtained by Bayesian and fiducial procedures do not provide an exact frequency interpretation in all instances. However, these are used quite extensively in reliability analysis because standard classical procedures are unavailable for all except the simplest systems [95].

Bayesian Analysis

Bayesian priors as discussed by Locks [50, p. 115ff] are used in the model to determine the reliability of each component for assessing system reliability. Raiffa and Schlaifer [75] provide an analysis of the theory behind the Bayesian approach. Using the Bayesian approach, historical data about each component is allowed to be incorporated into an appropriate Bayesian prior distribution provided for that component. Because the resulting posterior distribution depends upon the specific prior chosen, it is evident that problems are generated because of this prior. Mann [57] analyzes the selection of prior distributions and their effect on the resulting confidence limits. She found that for an exponential series system, the Bayesian bounds, although exact in the Bayesian sense were smaller than the classical bounds in every case.

In our case, the uniform prior is used in the absence of data for each component. Lawless [39] and Sarkar [77] analyzed the use of bounds generated by uniform priors on a system of exponential components and found them to be more conservative than bounds provided by fiducial priors. In comparing the Bayesian uniform prior approach with the classical approach, Schick and Prior [78] found that the uniform prior approach produced lower confidence limits on the system reliability that were larger than that produced by exact methods. Fertig [27] analyzed a serial system composed of exponential components from both the classical and Bayesian approach. He concluded that there are no prior distributions in the absence of data that can yield the exact unbiased confidence bounds provided by the classical approach. A review of the Bayesian controversy is analyzed by Easterling [20] and Lawless [39]. A summary of the finding and results for numerous articles is found in Mann, Schafer, and Singpurwalla [58].

Although some of the literature seems to indicate the lack of an optimum prior, it is believed that the uniform prior used in SPARCS in conjunction with an exact method, Poincaire's Theorem, for determining a system reliability equation as a function of the components, does in fact produce confidence bounds which indicate that the uniform prior does produce very good results. These results are verified through the use of a duality check in which system reliabilities and unreliabilities were compared from minimal path and minimal cut analysis. The results show very accurate complementary confidence levels for system reliability and unreliability and tend to indicate that a uniform prior is perhaps the optimum prior.

CHAPTER III

SYSTEM AND LOGICAL ANALYSIS

Introduction

An estimate of the reliability of a complex system can be determined by the development of an exact equation that is a function of the component reliabilities. This equation is developed for any logical system from an analysis of the system states. These states are of two types: success states called paths and failure states called cuts.

An algorithm has been developed for combining system minimal states, say minimal paths, to provide an equation for the system reliability. This algorithm, called Poincaire's Theorem (inclusion-exclusion), uses Boolean algebra and the theory of partially ordered sets to produce a system reliability equation as a function of the components. Set concepts, as presented by Feller [26], are developed in the concept of reliability by Locks [48]. The analysis that follows closely parallels the analysis provided by Locks [48].

Once the system equation is developed, confidence assessment for the system reliability can be performed. Since SPARCS specifies that the components be either attribute or Poisson process components, Monte Carlo methods are used to provide the individual component reliabilities for the generated system equation. This is done a number of times until a resultant empirical distribution of the system reliability estimates is produced.

Mathematical Concepts

Network diagrams may be used to analyze a system to determine the ways in which the success or failure of a system can occur. In a success-type network, called a logic diagram, each mode indicates the success or non-failure of a component or specific element of the network. In this context, a path is a set of components which by functioning cause the system to function. A minimal path is a smallest set of components which by functioning cause the system to function even if all the other components fail [25].

In a failure-type network, often called a fault tree, each mode denotes a failure of non-success for a particular element or component of the network. Then, a cut is a set of components which by failing cause the system to fail. A minimal cut is a smallest set of components which by failing cause the system to fail even with all other components functioning [25].

The analysis of any element of a network is binary in nature. Either the element is a success (1) or it is a failure (0). Consequently, Boolean algebra is used to provide a mathematical representation of the system states. Following this analysis, a system, which we will call ASYS, is composed of n binary components or elements i, $i = 1, 2, \ldots, n$. A 1 denotes a success and a 0 denotes a failure. Then, any state of ASYS can be represented as a binary n-dimensioned vector

$$X = (x_1, x_2 \;, \ldots\ldots x_n)$$

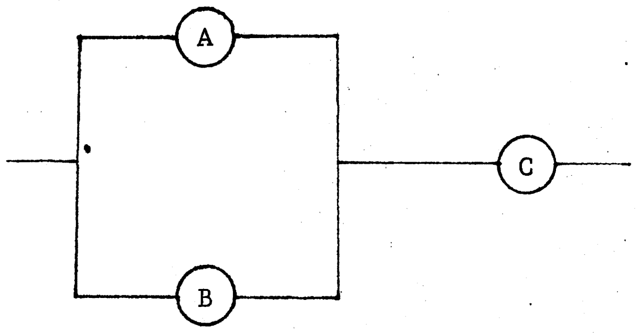
where

$x_i = 1$ is a success and

$x_i = 0$ is a failure.

The set of states $\{X\}$ that make up the network has 2^n different elements because of the binary nature of each element. States may be written as a function of X which has a value of unity, $f(X) = 1$, for those vectors which make the structure perform, (paths) and a value of 0, $f(X) = 0$, for those vectors which make the structure fail (cuts). It is assumed that all states are either paths or cuts.

In a system of the form



(1)

there are three elements. An analysis of the network can be provided by an analysis of the three elements. If each state is analyzed in order with the Boolean representation of each component (A, B, C) as

$X_n = A\ B\ C$

then the binary representations for the minimal paths are

$X_1 = 1\ 0\ 1$

$X_2 = 0\ 1\ 1$

and those for the minimal cuts are

$X_3 = 0\ 0\ 1$

$X_4 = 1\ 1\ 0.$

This provides a complete analysis of the network through an analysis of each state.

The probability of at least one of the minimal paths occurring is given as

$$R = p(X_1) + p(X_2) - p(X_1X_2) \tag{2}$$

This is the sum of the probabilities of each minimal path minus their intersections. The probability of at least one of the minimal cuts occurring is

$$\overline{R} = p(X_3) + p(X_4) - p(X_3X_4) \tag{3}$$

which is the sum of the probabilities of each minimal cut minus their intersection. This is the basis for Poincaire's Theorem which follows.

For each network component i, i = 1, 2, ..., n, the reliability r_i, $0 \leq r_i \leq 1$, is the probability of success, $x_i = 1$. Then $1 - r_i$ is the probability of failure, $x_i = 0$. Each component is assumed to be independent. Consequently, the probability of a particular state, pr(X) of X, is the reliability of the functioning components times the unreliability of the failed ones.

$$pr(X) = \prod_{i=1}^{n} \left(r_i^{\ x_i} \ (1 - r_i)^{1 - x_i} \right) \geq 0. \quad [48] \tag{4}$$

Poincaire's Theorem

By definition, V is a minimal path if it is a path and it does not include another path. This is the shortest path through a logic diagram and is so structured that the system functions even if all the other elements fail. For any path X, this may be represented as

$$f(V) = 1, \qquad V \nless X. \tag{5}$$

V is a minimal cut if it is a cut and is not included in another cut. The system fails with a minimal cut even if all the other elements are successful. This may be represented similar to (5) as

$$f(V) = 0, \qquad X \nless V. \tag{6}$$

Every path can be shown to include at least one minimal path and every cut is included in at least one minimal cut. Then the probability of the outcome of a network (success or failure) includes any given minimal state (path or cut) and is the numerical product of the probabilities of the state components (reliability or unreliability), which identify the state (path or cut). An analysis of these system minimal states leads to Poincaire's Theorem. Since there is a dual relationship between minimal-paths and minimal-cuts, Poincaire's Theorem is developed for paths and easily converted to cuts. Minimal cuts are just the minimal paths for failure [48].

Since every success state includes at least one minimal path, if there are m minimal paths $V_1, V_2, \ldots, V_m$, then the system reliability is the probability that at least one of these minimal paths are contained in a random outcome of system success.

$$R(ASYS) = pr\left(\bigcup_{j=1}^{m} (V_j \leq X)\right) \qquad (7)$$

The above expression is a combination of m expressions, an expression for each minimal path. At each step, the probability associated with that minimal path is combined with the previously combined minimal path probabilities. This is shown as

$$R_j = pr\left(\bigcup_{k=1}^{j} (V_k \leq X)\right)$$

$$R_j = pr\left(\left(\bigcup_{k=1}^{j-1} (V_k \leq X)\right) \quad v \quad (V_j \leq X)\right) \qquad (8)$$

$$R_j = R_{j-1} + h(V_j) - pr\left(\left(\bigcup_{k=1}^{j-1} (V_k \leq X)\right) \& \ (V_j \leq X)\right)$$

This combination ultimately yields an equation for the system reliability developed from an analysis of the minimal paths and a function of the component reliabilities. This expression, (8), expands very quickly as the number of minimal paths increase. The expression with 3 minimal paths, V_1, V_2, V_3, is developed in three steps.

$$R_1 = h(V_1)$$
$$R_2 = R_1 + h(V_2) - h(V_1 + V_2) \qquad (9)$$
$$R_3 = R_2 + h(V_3) - [h(V_1 + V_3) + h(V_2 + V_3) - h(V_1 + V_2 + V_3)]$$

R_3 may be expressed

$$R_3 = R_2 + h(V_3) - pr((V_1 \leq X \quad v \quad V_2 \leq X) \ \& \ (V_3 \leq X))$$

If R_3 is expanded to include m minimal paths, it becomes a prototype of the general case. For any step j, j = 1, 2, ..., m, let $\{h_2\}$

denote the set of $\binom{j}{2}$ minimal states expressed by $h(V_i + V_k)$ where $i < k < j$. Let $\{h_3\}$ denote the set of $\binom{j}{3}$ minimal states expressed by $h(V_i + V_k + V_l)$ where $i < k < l < j$. Following this expansion, the general case becomes:

$$R_j = \sum_{k=1}^{j} h(V_k) - \sum_{\{h_2\}} h(V_i + V_k) + \sum_{\{h_3\}} h(V_i + V_k + V_l) - \dots\dots + (-1)^{j-1} h(\sum_{k=1}^{j} V_k) \qquad (10)$$

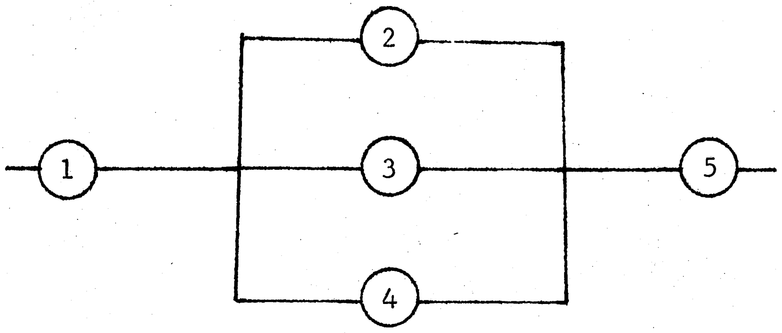
which is Poincare's Theorem.

In (10), at step j, $j = 1, 2, \dots, m$, the maximum number of terms is $2^j - 1$ [48]. Generally, the actual number of terms is some number less than $2^j - 1$ because in expanding (10), there are elements in common which can be merged with or cancelled against each other. Without these cancellations and combinations, (10) becomes cumbersome and possibly infeasible for large systems.

The above procedures are exactly the same for an analysis of minimal cuts. The only difference is the system reliability R(ASYS) becomes the system unreliability $\bar{R}$(ASYS). Then, $\bar{R}$(ASYS) is a combination of the minimal cuts to provide the system unreliability as a function of the component unreliabilities. In either case, it is assumed that every path is contained only in paths and every cut contains only cuts and all components are independent.

Application of the Theorem

Assume a system of the following configuration containing five components.



The system contains three paths: 125, 135, and 145. If any path functions, the system will function. Let r_1, r_2, r_3, r_4, r_5, represent the reliability of each component, then following Poincaire's Theorem, (10), the system reliability equation becomes

$$R = r_1r_2r_5 + r_1r_3r_5 + r_1r_4r_5 - r_1r_2r_3r_5 - r_1r_2r_4r_5 - r_1r_3r_4r_5 + r_1r_2r_3r_4r_5. \tag{11}$$

Because of the complementary relationship of the system reliability and unreliability, the system unreliability may be found as

$$\bar{R} = 1 - R \tag{12}$$

If an unreliability analysis is desired, the system cuts are identified as 1, 234, and 5. Thus, the system will fail if any one of these three situations occurs even if the other components are not failed. Again following Poincaire's Theorem, (10), if $\bar{r}_1$, $\bar{r}_2$, $\bar{r}_3$, $\bar{r}_4$, $\bar{r}_5$ represents the unreliability of each component, the system unreliability equation becomes

$$\bar{R} = \bar{r}_1 + \bar{r}_2\bar{r}_3\bar{r}_4 + \bar{r}_5 - \bar{r}_1\bar{r}_2\bar{r}_3\bar{r}_4 - \bar{r}_1\bar{r}_5 - \bar{r}_2\bar{r}_3\bar{r}_4\bar{r}_5 + \bar{r}_1\bar{r}_2\bar{r}_3\bar{r}_4\bar{r}_5. \tag{13}$$

The system reliability can be obtained as the complement of the unreliability:

$$R = 1 - \bar{R} \tag{14}$$

Equation Development by MAPS

The development of the system equation as a function of the components is provided in a program by J. L. Burris [11]. This equation generating routine is used in SPARCS to provide the equations for the simulation of complex systems. The minimal paths or minimal cuts are provided by the user. EQGEN, the part of MAPS (and SPARCS) that generates the system equation, uses this information to provide the system equation as a function of the component reliabilities or unreliabilities. Component values are placed into this equation for each simulation run to provide an estimate of the system reliability or unreliability.

CHAPTER IV

STATISTICAL ASPECTS

Introduction

SPARCS provides an analysis of a complex system of any logical configuration in which the components are either Bernoulli or Poisson process components. Bayesian analysis is applied to the two component types. It provides a convenient method of incorporating sample observations with prior distributions to provide adjusted estimates of component reliabilities. These prior distributions are functions of prior data and test observations. In this way, prior knowledge and historical data can be used to provide reliability assessment.

Using two basic component types allows the use of predefined natural conjugates. These natural conjugates allow the combination of future observations with these conjugate priors to yield a posterior distribution of the same family. In both cases, the priors, in the absence of data, are defined to be uniform priors.

This chapter covers Bayesian analysis of Beta and Gamma type components. The Bernoulli and Poisson processes for providing component information are analyzed. Next follows a brief discussion of the purpose and fundamentals of the Bayesian approach with regards to reliability analysis. Finally, a detailed discussion for each type of component with a mathematical analysis for each is given.

In this discussion, a number of authors will be paralleled for a portion of the analysis. The Bayesian discussion will parallel Raiffa and Schlaifer [75, pp. 28-79], Lehman [41, pp. 10-21], Schmidtt [80], Locks [50, pp. 115129] and Mann, Schafer and Singpurwalla [58, pp. 379-404].

Bernoulli and Poisson Processes

Bernoulli Process

A Bernoulli process is a process in which the probability of success (or failure) remains constant over a series of independent trials. The probability of success is generally denoted by p and that of failure by (1-p). Thus, the probability of any outcome is the product of the probabilities of the results of the independent trials:

$$p^{r} (1-p)^{n-r} \qquad \text{where } 0 < p < 1. \tag{1}$$

This is known as the "kernel" of a binomial distribution which is the result of a Bernoulli process.

There are two basic types of Bernoulli (attributes) testing. If the number of tests are fixed such that the number of successes (failures) becomes a random variable, the binomial family is used as representative of this procedure. When the number of successes (failures) are fixed and the number of tests become random, the negative binomial family represents this procedure.

Since the reliability of a Bernoulli component is the probability of success of that component, the value of p is the unknown. Thus, assessing the reliability of a Bernoulli component is the same as assessing the value of p.

A beta distribution is used in reliability assessment for Bernoulli processes. This distribution is on the reliability p, which is a continuous random variable over the range (0,1). This function appears in the probability density function as developed from the "kernel":

$$f_{\beta}(p|a,b) = \frac{(a+b+1)!}{a!\ b!} p^{a}(1-p)^{b} \tag{2}$$

where $0 < p < 1$. Here,

$$\beta(a+1,\ b+1) = \frac{(a+b+1)!}{a!\ b!} \tag{3}$$

is known as a Beta function.

Poisson Process

For a Poisson process, the probability density function, distribution function, and reliability assessment is based upon the assumption of a constant failure rate, λ, which is independent of time. The amount of time necessary for the first failure to occur, T, is a random variable whose probability is subject to the exponential density function, $\lambda e^{-\lambda x}$. The time may be in ordinary units such as minutes, hours, etc., or in time blocks where each block represents one time unit.

The probability that the time for a failure to occur, T, is less than time, t, is given below. If

$$F(t) = pr(T \leq t)$$

$$1-F(t) = pr(T > t) \tag{4}$$

then

$$1-F(t) = \exp\left\{-\lambda t\right\} \tag{5}$$

and

$$F(t) = 1 - \exp\left\{-\lambda t\right\} . \tag{6}$$

By definition

$$f(t) = \frac{d(F(t))}{dt} \tag{7}$$

then

$$f(t) = \lambda \exp\left\{-\lambda t\right\} . \tag{8}$$

For any time, t, the probability that failure occurs before time t is the function represented by (4) and (6). The probability that failure occurs after t is the survival probability or reliability given by

$$R(t) = pr(T > t) = 1-F(t) = \exp\left\{-\lambda t\right\} . \tag{9}$$

If the analysis is expanded to analyze cases in which more than one failure occurs, the reliability is judged not by the time for a single failure to occur but by the time for n failures to occur where $n > 1$. This expansion yields a probability density function on the time to the n^{th} failure of the form

$$f(t) = \frac{\lambda^n t^{n-1} \exp\left\{-\lambda t\right\}}{(n-1)!} . \tag{10}$$

This expansion results in a Gamma probability density function in which the denominator is also known as a Gamma function, $\Gamma(n)$.

Bayesian Reliability Analysis

Bayesian analysis generally uses a continuous prior distribution on the reliability, p, over the range (0,1). Because of the inherent variability of data, the value for the reliability, p, can only be specified up to a confidence factor. The lower confidence limit on the reliability, p, at confidence level γ is the lowest value p_ℓ such that

$$\gamma = pr(p \geq p_\ell) \tag{11}$$

and

$$1-\gamma = pr(p < p_\ell). \tag{12}$$

Thus, the Bayesian analysis partitions the prior distribution into two parts: the part below the lower confidence limit p_ℓ with probability $1-\gamma$ and the proportion above with probability γ.

For a Poisson process, the prior is a Gamma distribution on the failure rate λ with total testing time and total failures as parameters. A change of variables technique is required to produce a distribution on the reliability, p. For a Bernoulli process, the prior is a Beta prior on the reliability, p, which may be used with both binomial and negative binomial data. The parameters for the Beta are total tests and total failures.

In SPARCS, the components are defined to be of two types: Bernoulli and Poisson process components. These components lend themselves to priors from the Beta and Gamma families which are acceptable prior families as defined by Raiffa and Schlaifer [75]. These priors are mathematically tractable in that a posterior distribution may be reasonably determined from a prior distribution and a given observation from the same population. Both distributions are closed in the sense that the posterior is a member of the same family as the prior. Thus, both distributions are associated with the Koopman-Pitman-Darmois [75, 41] class of distributions. In these distributions, the likelihood obtained by repeated independent trials is a function of the additive sufficient statistics observed in these trials. Thus the priors for both distributions are the natural conjugates. This guarantees that the posterior distributions are of the same form and family as the prior with parameters that are the sum of the sufficient statistics for the prior and the sufficient statistics for current data.

Component Analysis

Bernoulli Components

Bernoulli analysis is utilized for components which are placed in tests and a record kept on the number of failures observed in the tests. The conditional probability given, p, that our Bernoulli process will generate r successes and n-r failures in some specified order is

$$\Pi \left(p^{x_i} (1-p)^{1-x_i}\right) = p^r (1-p)^{n-r} \tag{13}$$

which is the likelihood of the sample observations from our population with the parameters (r,n) as sufficient statistics.

For a Bernoulli process with p as a random variable, the natural conjugate is the Beta distribution which is continuous and defined by

$$f_\beta (p|r, n) \propto p^r (1-p)^{n-r}. \tag{14}$$

Following the use of primes (') by Raiffa and Schlaifer, [75, p. 53] the Beta distribution has (n', r') as parameters which are sufficient statistics. If the sample observations also have parameters (n,r) then it can be shown that the parameters of the posterior distribution on p are

$$n'' = n' + n, \qquad r'' = r' + r,$$

and the posterior is of the same form as the prior. Then by Bayes' Theorem,

$$G'(p|r', n':r,n) \propto p^{r'}(1-p)^{n'-r'} \cdot p^{r}(1-p)^{n-r} \qquad (15)$$
$$\propto p^{r''}(1-p)^{n''-r''}$$

which is a Beta.

The kernel of the beta prior distribution has the form

$$p^{r}(1-p)^{n-r}. \qquad (16)$$

From this function, the normalizing constant, denoted as K[B] is developed such that

$$\int_0^1 G'(p\ r', n':r,n)\,dp = \int_0^1 p^{r''}(1-p)^{n''-r''}dp \qquad (17)$$
$$= K[B]\int_0^1 p^{r''}(1-p)^{n''-r''}dp = 1.$$

Let

$$a = r''$$
$$b = n'' - r''$$

then by use of successive integration by parts

$$[K(B)]^{-1} = \int_0^1 p^a(1-p)^b\,dp = \frac{a!b!}{(a+b+1)!} \qquad (18)$$

Applying this to equation (17) above gives the incomplete integral as

$$\int_0^p G'(x|a, b)\,dx = \frac{(a+b+1)!}{a!b!}\int_0^p x^a(1-x)^b\,dx \qquad (19)$$

which is the incomplete Beta function. This function (19) is of the same form as the prior function with the addition of the normalizing constant.

Since (19) is the incomplete Beta function, it is represented as

$$F_{\beta}(p|r, n-r) = \int_0^p G'(x|a, b)dx$$

$$= \int_0^p G'(x|r'', n'' - r'')\, dx \tag{20}$$

for easier analysis. For reliability-confidence assessment, p is the probability of success (reliability) of the component under analysis where

r = number of successes

n = number of trials.

Now, since the Beta distribution is continuous,

$$F_{\beta}(p_{\ell}\,|\,r, n-r) = pr(p < p_{\ell}). \tag{21}$$

For reliability-confidence assessment a lower limit is needed on the reliability and is accomplished by

$$1 - F_{\beta}(p_{\ell}\,|\,r, n-r) = pr(p \geq p_{\ell}) \tag{22}$$

where

$$\gamma = pr(p \geq p_{\ell}). \tag{23}$$

and

$$1-\gamma = F_{\beta}(p_{\ell}\,|\,r, n-r). \tag{24}$$

Then, $1-\gamma$ is defined as the confidence that the actual reliability (p) is greater than the lower confidence limit (p_{ℓ}) placed on the reliability and that p_ℓ is the $1-\gamma$ percentage point of the Bayesian posterior distribution.

Poisson Process Components

The second type of components provided for are those on which statistics have been obtained on the number of failures relative to a specific testing time. These components are analyzed in one of two ways. If a Poisson process is used, the total number of failures (r) in a specified testing period (t) may be observed or the components may be tested with regards to the total testing time (t) necessary to generate a specified number of failures (r).

In either case, the natural conjugate prior is the Gamma distribution defined by

$$f(\lambda \mid r,t) \propto \lambda^{r-1} \exp\{-\lambda t\} \ . \tag{25}$$

where λ is defined as the failure rate which is a constant independent of time.

For Poisson processes, the survival probability is the probability that a failure occurs after a specified time t and is defined by the relationship

$$R_t = \exp\{-\lambda t\} = pr(T > t) \tag{26}$$

where T is the time of the specified failure. This derives from the basic exponential density function

$$f(t \mid \lambda) = \lambda \exp\{-\lambda t\} \ . \tag{27}$$

Then

$$F(t \mid \lambda) = 1 - \exp\{-\lambda t\} \tag{28}$$

which is $pr(T \leq t)$.

Now the probability that T occurs at some time greater than t is

$$R_t = 1 - (1 - \exp\{-\lambda t\}) \qquad \text{or}$$

$$R_t = \exp\{-\lambda t\} = pr(T > t) \tag{29}$$

which provides our survival probability. Then (29) is the conditional probability that the failure time T will be greater than a specified time t, given λ.

To provide the likelihood of the sample, the joint likelihood that a process will provide r failures in a specified time period t is

$$\left(\Pi_{i=1}^{r} (\lambda \exp\{-\lambda t_i\})\right) \exp\{-\lambda t_{r+1}\} = \lambda^r \exp\left\{-\lambda \sum_{i=1}^{r+1} t_i\right\}. \tag{30}$$

If

$$t = \sum_{i=1}^{r+1} t_i \tag{31}$$

then (31) is written

$$\lambda^r \exp\{-\lambda t\} . \tag{32}$$

The time, T, for the first failure to occur is derived from the basic exponential density function. The analysis may be extended to include cases where the reliability of a system is judged by the time for n failures to occur, $n > 1$, and not by the time for a single failure [75, p. 96]. Taking this into consideration, the gamma density function is used and is defined as [75, p. 225]

$$f(\lambda \mid r,t) \propto \lambda^{r-1}\, t^r \exp\{-\lambda t\} . \tag{33}$$

The joint likelihood is defined as [75, p. 225]

$$\lambda^r\, t^r \exp\{-\lambda t\} \quad [75]. \tag{34}$$

If (34) is a sample observation from the population with (r,t) sufficient and (33) is the prior kernel with (r', t') sufficient then

$$r'' = r' + r, \qquad t'' = t' + t,$$

and the posterior will be of the same form as the prior. The posterior Gamma distribution is

$$\begin{aligned} G'(\lambda \mid r'', t'') &\propto (\lambda^{r-1} t^{r} \exp\{-\lambda t'\}) \cdot (\lambda^{r} t^{r} \exp\{-\lambda t\}) \\ &\propto \lambda^{r''-1} t^{r''} \exp\{-\lambda t''\} . \end{aligned} \tag{35}$$

which is a combination of the natural conjugate and the joint likelihood.

Following earlier analysis, the normalizing function is determined from the Gamma density function such that

$$[K(B)]^{-1} = \int_0^\infty x^{r-1} \exp\{-x\} \quad dx = (r-1)!. \tag{36}$$

The posterior distribution on the Poisson process follows in that

$$\begin{aligned} G'(\lambda \mid r'', t'') &= \lambda^{r''-1} t^{r''} \exp\{-\lambda t''\} \cdot K(B) \\ &= \frac{\lambda^{r''-1} t^{r''} \exp\{-\lambda t''\}}{(r''-1)!} \cdot \end{aligned} \tag{37}$$

For easier analysis (37) is represented as

$$F_\gamma (\lambda_\gamma \mid r,t) = \int_{\lambda_\gamma}^\infty G'(\lambda_\gamma \mid r'', t'') d\lambda . \tag{38}$$

The posterior distribution for a Poisson process is on the failure rate, λ such that

$$\gamma = \text{pr} (\lambda < \lambda_\gamma).$$

Following the reliability-confidence assessment on the Bernoulli components, the upper confidence limit on λ is the γ percentage point of the Gamma distribution. The lower confidence limit on λ is provided by

$$\gamma = 1 - F_\gamma (\lambda_\gamma \mid r,t) = pr(\lambda < \lambda_\gamma) \qquad (39)$$

$$1-\gamma = F_\gamma (\lambda_\gamma \mid r,t).$$

However, for component reliability analysis, a Gamma distribution needs to be on the component reliability, p, instead of the failure rate, λ. From earlier analysis,

$$R_t = \exp \{- \lambda t\}$$

which is the survival probability (reliability) for a specified period of time--if

$$R_t = p$$

then

$$p = \exp \{- \lambda t\} \qquad (40)$$

which provides a lower confidence limit on p such that

$$R = pr(p_\ell \leq p).$$

From (40) a conversion factor is obtained to provide a negative-log gamma on p instead of λ. Hence,

$$p = \exp \{- \lambda t\}$$

or

$$\lambda t = -\ln(p). \qquad (41)$$

If t in (40) is measured in "required" operation time (blocks) instead of minutes, hours, etc. and if t = 1, it becomes

$\lambda = -\ln p = \ln (1/p).$ [50, p. 125]

Applying this conversion factor to (37), the negative-log gamma distribution on p becomes [50, p. 125]

$$F(p|r'',t'') = \frac{t^r \ln(1/p)^{r-1} p^{t-1}}{(r-1)!}, \quad 0<p<1. \tag{42}$$

Calculations by SPARCS

SPARCS uses the incomplete Beta and Gamma distribution obtained from the International Mathematical and Statistical Library (IMSL) to provide a lower confidence limit (p_ℓ) on each component. If unreliability analysis is desired, the unreliability ($\bar{p}_u$) is obtained by the relationship:

$$\bar{p}_u = 1 - p_\ell .$$

Here, $\bar{p}_u$ is an upper confidence limit on the unreliability. This specifies that the true unreliability is between $\bar{p}_u$ and zero (0).

The reliability for each component is obtained in a series of steps. First, a uniform random number is generated which corresponds to the confidence level $(1 - \gamma)$, for both component types:

$1 - \gamma = F_\beta (p_\ell | r, n - r)$ for the Beta

and

$1 - \gamma = F_\gamma (p_\ell | r, t)$ for the negative-log gamma. (43)

Next, the historical data supplied for each component is utilized. If the component is Beta, the number of successes and failures are used as parameters in conjunction with the random number. If the component is Gamma, the number of failures and the total testing time are used as parameters.

The parameters and the random number are used with the appropriate distribution to provide a random deviate from that distribution. This deviate corresponds to a lower confidence limit (p_ℓ) for that component for a given confidence level ($1 - \gamma$). When the reliability (p_ℓ) or unreliability ($\bar{p}_u$) for each component is obtained, this value is placed in the reliability (unreliability) equation for the system (or module) to provide an estimate of the reliability (unreliability) for that system (or module).

In the absence of data, the input parameters for each distribution become zero. Consequently, the observed prior on that distribution becomes indeterminate. To alleviate this problem, a uniform prior is provided for each distribution in the absence of data by adding one (1) to each parameter. For the Beta components the parameters become:

$$n'' + 1$$

and

$$n'' - r'' + 1 \tag{44}$$

and the negative-log gamma parameters become

$$r'' + 1$$

and

$$t'' + 1 \tag{45}$$

This uniform (ignorance) prior also allows SPARCS to handle troublesome parameters. For example, in Apollo-Saturn component testing, many components have no failures for a representative period of testing time. Thus, one of the parameters is zero. This would yield an indeterminate distribution and prevent analysis. The uniform prior alleviates this problem.

CHAPTER V

MODEL VERIFICATION

Introduction

In this chapter SPARCS is extensively tested to determine that the concept employed in the model and the model results are intact and correct. This testing extended over a considerable period of time. However, to facilitate analysis, they are grouped into two basic categories.

The first portion of this chapter discusses the results of tests of some of the concepts incorporated into the model. The duality concept [48] which is a result of Poincaire's Theorem, is tested. Next, the presence of the uniform (ignorance) prior in the absence of data concept is verified.

The second portion provides tests of some simple binomial and exponential systems. The first runs consist of single component systems. Since reliability assessment for single components is available, this will verify that the statistical routines and basic concepts are implemented correctly. Next, test runs from SPARCS are compared with known non-randomized, randomized and Monte Carlo techniques for assessment of simple system reliability by Mann [54], Buehler [8], Harris [33], Berkbigler and Byers [4], and Grubbs [32]. Non-randomized bounds require techniques which do not rely on a uniform random variate from a uniform (0,1) distribution while randomized techniques **do** require a uniform variate. Monte Carlo bounds utilize simulation techniques to derive confidence bounds on the system reliability.

Duality Verification

Poincaire's Theorem contains a duality concept which describes a complementary relationship between the system reliability and unreliability. This concept states that the system reliability, R, can be obtained as the complement of the system unreliability, $\bar{R}$ [8].

$$R = 1 - \bar{R} \tag{1}$$

As discussed in Chapter III, the system reliability is a function of the reliability of the system components and is derived from the system minimal paths. The system unreliability is a function of the unreliability of the system components and is derived from the system minimal cuts. Although the system reliability and unreliability are obtained in two different ways, they still must be complementary.

In providing interval estimates for system reliability or unreliability, SPARCS uses the system minimal paths to provide a system reliability equation as a function of the component's reliabilities. The system minimal cuts are used to obtain a system unreliability equation as a function of the component unreliabilities. In either case, component failure-history data is used to enter the appropriate distribution (either beta or negative-log gamma). For the beta components, the failure-history data consists of total successes and total failures. For the negative-log gamma components, the failure-history data consists of total failures and a specified number of "mission times." In each case, the values returned from the appropriate distribution is the component reliability. Using the duality concept for each component, the component unreliability is found as 1-R. Component reliabilities are placed in the system reliability equation generated from the minimal paths to obtain a system reliability point estimate. Component unreliability values are placed in the systems unreliability equation generated from the system minimal cuts to obtain a system unreliability point estimate.

A test run was made with the simple series-parallel system of Figure 1 and the more complex system of Figure 2.

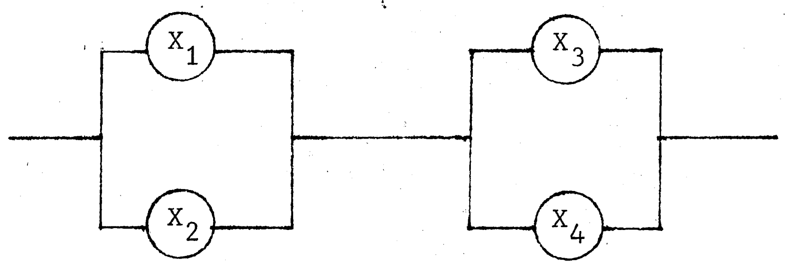


Figure 1. Series-Parallel System

On the first example, system reliability assessment was performed. The four minimal paths for Figure 1 (X_1X_3, X_1X_4, X_2X_3, X_2X_4) were combined to provide reliability interval estimates for the system. Next, system unreliability assessment was performed on the same system.

The two minimal cuts (X_1X_2, X_3X_4) were combined to provide unreliability interval estimates for the system. If the duality concept and Poincaire's Theorem are utilized correctly, these should be complements of each other.

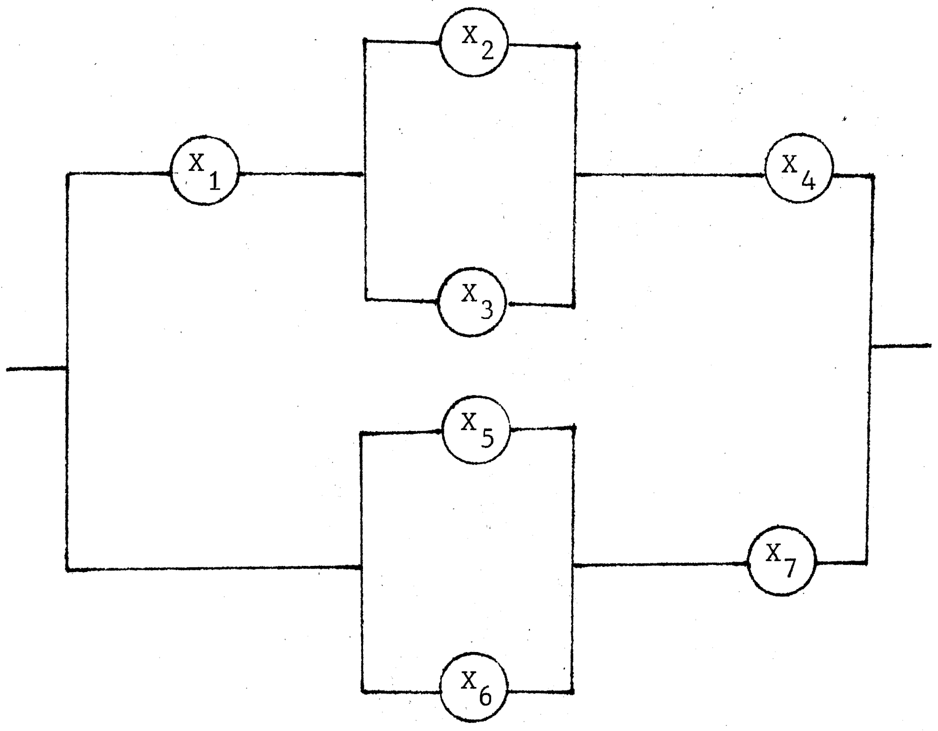


Figure 2. Seven Component Example

On the second example (Figure 2), the four minimal paths ($X_1X_2X_4$, $X_1X_3X_4$, X_5X_7, X_6X_7) were combined for the system reliability interval estimates and the six minimal cuts ($X_1X_5X_6$, X_1X_7, $X_2X_3X_5X_6$, $X_2X_3X_7$,

$X_4X_5X_6$, X_4X_7) provided system unreliability interval estimates. Again, these should be complements of each other.

The reliability and unreliability values obtained from the duality tests are shown in Table I and Table II. The results show the duality concept intact. Since

$$\bar{R} = 1 - R \tag{2}$$

then

$$R + \bar{R} = 1 \tag{3}$$

The reliability values in each case are obtained from the system minimal paths. The unreliability values are derived from the system minimal cuts. The equation generated in each case by Poincaire's Theorem is different. However, the reliability and unreliability values obtained should be complementary according to the duality concept.

For example, assume a simple parallel system as depicted in Figure 3.

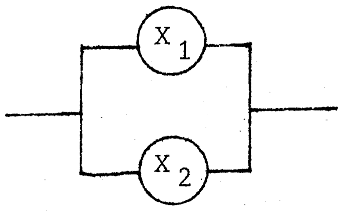


Figure 3. Parallel System

TABLE I

SERIES-PARALLEL DUALITY VERIFICATION

Percent	Reliability Lower Bounds	Unreliability Upper Bounds	Sum of Values	Variance
1	.996452	.003131	.999581	.000417
2.5	.995942	.003415	.999357	.000643
5	.995591	.003656	.999247	.000753
10	.995222	.004979	1.000201	(.000201)
20	.993751	.006088	.999839	.000161
25	.992806	.007101	.999907	.000093
50	.989835	.009908	.999743	.000257
75	.984355	.014056	.998411	.001589
80	.982509	.017350	.999859	.000141
90	.979810	.020747	1.000557	(.000557)
95	.972242	.023081	.995323	.004677
97.5	.970772	.024031	.994803	.005197
99	.965649	.025333	.990982	.009018

TABLE II

SEVEN COMPONENT DUALITY VERIFICATION

Percent	Reliability Lower Bounds	Unreliability Upper Bounds	Sum of Values	Variance
1	.999530	.000591	1.000121	(.000121)
2.5	.999263	.000678	.999941	.000059
5	.999010	.000999	1.000009	(.000009)
10	.998546	.001607	1.000153	(.000153)
20	.997917	.002277	1.000194	(.000194)
25	.997600	.002652	1.000252	(.000252)
50	.995886	.004338	1.000224	(.000224)
75	.992766	.006847	.999613	.000387
80	.991539	.007434	.998973	.001027
90	.988821	.10059	.998880	.001120
95	.986696	.012361	.999057	.000943
97.5	.983445	.015057	.998502	.001498
99	.979860	.017183	.997043	.002957

There are two components, X_1 and X_2 whose reliabilities are R_1 and R_2, respectively. The system minimal paths are (X_1 and X_2) and the system minimal cut is X_1X_2. Following Chapter III, the system reliability equation generated from the minimal paths is

$$S_r = X_1 + X_2 - X_1X_2.$$

Substituting in the component reliabilities gives

$$S_r = R_1 + R_2 - R_1R_2.$$

The system unreliability equation generated from the system minimal cut is

$$S_u = X_1X_2.$$

The component unreliabilities are $(1-R_1)$ and $(1-R_2)$ respectively. Substituting the component unreliabilities into the system unreliability equation yields

$$S_u = (1-R_1)\ (1-R_2)$$

which may be expanded to obtain

$$S_u = 1 - R_1 - R_2 + R_1R_2$$
$$S_u = 1 - (R_1 + R_2 - R_1R_2)$$

which in fact is 1 minus the system reliability as derived in the system reliability equation, S_r, obtained from the system minimal paths. By adding the system unreliability and reliability

$$S_u + S_r = 1 - (R_1 + R_2 - R_1R_2) + (R_1 + R_2 - R_1R_2)$$
$$S_u + S_r = 1.$$

Ideally, the sum of the system reliability and unreliability should equal 1 since the reliability factors cancel.

To determine the accuracy of the values being generated by SPARCS, the system reliability and unreliability values at each percentile were added. Since the reliability factors should be equal, the sum of the system reliability and unreliability values should equal 1.

The tables show that SPARCS is generating very good results. In each case, the sum of the reliability and unreliability percentile points for the system very closely approximate one. For the series-parallel example, 100 simulation runs are made which produce a maximum difference of .0090. For the seven component example, 400 runs are made with a maximum difference of .00295.

The values in each case do not exactly equal one. These values are obtained by finding percentile points from the empirical distribution generated by SPARCS. The idea that these points are determined by an interpolation procedure on an empirical distribution generated by Monte Carlo procedures can easily account for the slight differences being encountered. However, even with these small differences, the accuracy of the values is, in fact, very good, even with small sample sizes.

Test of Uniform Priors

The model was developed to handle components of two basic types: Bernoulli components and Poisson process components. For the Bernoulli components, the prior distribution on the reliability p is a beta whose probability density function is

$$f(p|r, n) = \frac{p^r (1-p)^{n-r}}{\beta(r+1, n-r+1)}, \quad r \geq 0;\ n \geq r; \quad 0 < p < 1 \qquad (4)$$

where

$$\beta(r+1,\ n-r+1) = \frac{r!\ (n-r)!}{(n+1)!} \tag{5}$$

For Poisson process components, the prior distribution on p is the negative-log gamma whose probability density function is

$$f(p|r,\ T) = \frac{p^T\ (\ln 1/p)^r\ (T+1)^{r+1}}{\Gamma(r+1)},\ r,\ T \geq 0;\ 0 < p < 1 \tag{6}$$

where

$\Gamma(r+1) = r!$ [50, p. 115-128].

In either case, a uniform (ignorance) prior is provided in the absence of data. This provision is used to remove the possibility of generating an indeterminate distribution in cases which contain components with very high reliabilities (no failures in a representative number of tests).

To test this provision, a system containing both types of components was used. No component in this system contained any value (other than zero) for its historical data (failures, successes, or testing time). If the uniform provision is intact, there should be a 50% probability for either success or failure for the system in the absence of data.

The mean system reliability and the estimated system reliability values were observed to determine how well they approximated the 50% probability for success or failure for the system. The mean system reliability is the value obtained by summing the reliability point values for the empirically generated distribution and dividing by the number of Monte Carlo simulation runs required to generate this distribution. Thus, it is the "calculated" arithmetic mean of this empirical distribution. The estimated system reliability is a value derived from the "average" reliability for each component. The "average" reliability for Bernoulli components is $\frac{r+1}{n+2}$ which is the mean of the beta distribution representing that component [50, p. 52ff]. For each beta distribution, r is successes and n is the number of tests.

For the Poisson process components, the "average" reliability is $\left(\frac{T+1}{T+2}\right)^{r+1}$ which is the mean of the gamma distribution representing that component [50, p. 159]. For each gamma distribution, T is actual testing time in mission equivalents and r is the actual failures. These "average" values for each components are placed in the proper position in the system equation to derive an estimated reliability for the system as a function of the component "average" reliabilities. Thus, from this test, the system mean system reliability was .500852 and the estimated system reliability was .492188 which compares favorably with the 50% value that was needed.

Simple System Tests

There are currently no methods other than SPARCS for reliability confidence assessment of a complex system of any logical configuration in which component types may be freely intermixed. However, there are methods to approximate the reliability of simple series or parallel systems in which all components are of the same type.

The results of the tests performed by SPARCS tend to be very good. Analysis of these two basic methods has shown that Monte Carlo bounds as suggested by Burnett and Wales [10] and Levy and Moore [45], which are approximated by Grubbs [32], tend to be slightly conservative. For example, the lower confidence bounds for system reliability assessment obtained by Monte Carlo methods tend to be slightly lower than similar non-randomized bounds as developed by Buehler [8]. Conversely, upper bounds on system unreliability tend to be greater using Monte Carlo techniques than the same non-randomized bounds [56]. If this is the case, the bounds generated by SPARCS are better bounds than the less conservative bounds. Although SPARCS provides Monte Carlo bounds, the non-randomized and randomized bounds are also presented for reference in the forthcoming tables.

Single Component Reliability Comparisons

A test was used with the IMSL routines, as incorporated in the model, to determine that they were producing correct component reliability values. Since confidence bounds can be approximated for components, these values should approximate the bounds produced by simulating values for a single component. Hand calculated lower confidence bounds were developed for a gamma and beta component.

For the gamma component, the chi-square approximation was used. The upper confidence bound of the failure rate, λ, at confidence level γ where

$$\gamma = \mathrm{pr}(\lambda \leq \lambda_\gamma) \tag{7}$$

is distributed as

$$\gamma = F_{\chi^2(2r)}(2\lambda_\gamma T). \qquad \begin{array}{l} r = \text{failures} \\ T = \text{total times in test} \end{array} \tag{8}$$

From this, the lower confidence bound on the reliability, R_γ , of a component may be found by a direct conversion:

$$R_{\gamma} = \exp \left\{-\lambda_{\gamma}\right\} \qquad [50, \text{pp. } 115\text{-}128]. \tag{9}$$

Thus, by entering chi-square tables with 2r degrees of freedom and the confidence level γ, the value $2\lambda T$ can be obtained. Since the value of T is provided as part of the component historical data, λ_{γ} can be obtained.

Then, λ_{γ} is placed in (9) to obtain a lower confidence bound on R_{γ} at confidence level γ. Table III shows the hand calculated bounds as compared to the lower confidence bounds determined by simulation.

TABLE III

LCB* FOR A SINGLE GAMMA COMPONENT
WITH T=51.2 and F=5

Lower Confidence Level	Chi-Square Approximation Values	SPARCS Simulation Values
.95	.83629	.824990
.90	.855529	.854416
.80	.876982	.874309
.50	.912808	.911996
.20	.941443	.942352
.10	.953601	.955653
.05	.962255	.962827

*LCB = lower confidence bound. For this comparison, the uniform prior assumption was removed from SPARCS.

For the beta component test, an approximation for the lower confidence bound on the reliability of a single component as developed by Mann [33] is used. For this approximation, the mean is

$$m = \ln(n + .5) - \ln(n-r-.5), \tag{10}$$

the variance is

$$V = (n-r-.5)^{-1} - (n+.5)^{-1}, \tag{11}$$

and the degrees of freedom are

$$f = \frac{2m^2}{V} \qquad \text{where } n = \text{total time in tests},\ r = \text{failures.} \tag{12}$$

This information is used in conjunction with the Wilson-Hilferty chi-square approximation [57] to approximate a lower confidence bound in the expression:

$$R = \exp\left\{-m(1 - (2/9f) + Z_\gamma (2/9f))^3\right\} \tag{13}$$

where Z_γ is the γth quantile of the standard normal distribution. The results are found in Table IV.

<u>Bernoulli System Tests</u>

Buehler [8], Harris [33] and Mann [54] provide methods for approximating the reliability of simple systems consisting of Bernoulli type components. Buehler [8] provides confidence intervals for a system of two binomial components with small probabilities of failure and moderate sample sizes for historical test information. His intervals are based upon a set of inequalities in conjunction with a Poisson approximation to the binomial distribution. These bounds tend to be conservative in general in that the $1-\alpha$ confidence level may be frequently exceeded [33]. Harris [33] uses a random variate from

TABLE IV

LCB* FOR A SINGLE BETA COMPONENT
WITH F=3 AND S=47

Lower Confidence Level	Values From Mann Approximation	SPARCS Simulation Values
.95	.829573	.832268
.90	.849497	.850151
.75	.879815	.873055
.50	.90897	.904778
.25	.933414	.933372
.10	.951533	.952446
.05	.960684	.957405

*LCB = lower confidence bound

a uniform (0,1) distribution to remedy the conservatism of Buehler's method. Harris also applies a Poisson approximation to obtain lower bounds on the reliability of redundant binomial systems and extends his procedure to accommodate more than two component systems. Mann [54] develops a procedure to provide bounds on series or parallel binomial systems in which the component sample sizes are large and the component failures are small. Mann's Approximately Optimum (AO) procedure can be used either with or without uniform random variates. For these bounds, the Wilson-Hilferty transformation for the approximate noncentral Chi-square distribution is used. These methods, although based on approximations, provide results which closely approximate supplied test values. Since some of the methods require extensive programming and mathematical calculations, test results were taken from several articles and compared with similar results produced by SPARCS.

The examples used for comparison were simple parallel systems containing two and three Bernoulli type components, respectively. The historical component information was the same type used in tests by each of the respective authors. The results are presented in Table V.

Exponential System Tests

As mentioned earlier, there are three basic approaches for providing confidence bounds on the reliability of simple systems. One approach was developed by Lentner and Buehler [44] and expanded by El Mawaziny [21] for simple series systems. This approach uses nonrandomized techniques for providing lower confidence bounds on the

TABLE V

COMPARISON OF UPPER 90% CONFIDENCE BOUNDS FOR SIMPLE BERNOULLI SYSTEMS WITH r FAILURES AND s SUCCESSES

Example Number	Data (r, s)	Buehler's Poisson Approx.[8]	Harris Non-Random Poisson[33]	Mann Non-Random[54]	Harris Random Poisson[33]	Mann Random[54]	SPARCS Bound
1	(3,97) (5,95)	.00412	.00486	.00420	.00416	.00417	.004477
2	(2,98) (3.97) (5,95)	.000133*	.000186	.000127	.000145	.000146	.000132

*Likelihood-ratio confidence bound substituted since confidence bounds are unavailable for k > 2.

r = number of failures
s = number of successes
k = number of components

reliability of exponential series systems. Since the procedure used by El Mawaziny, for more than two subsystems, tends to be large and tedious, approximations have been developed by Mann and Grubbs [56] and El Mawaziny and Buehler [22].

Simulation, another technique used to provide confidence bounds, has been discussed by Levy and Moore [45] and Burnett and Wales [10] and others. A mathematical technique for approximating these bounds for simple series systems has been developed by Grubbs [32] to shorten the time involved in obtaining these bounds using computer runs. Lower bounds on the system reliability determined by the simulation techniques mentioned above tend to be lower than bound provided by the non-randomized techniques.

Berkbigler and Byers [4] used the basic simulation techniques discussed earlier to provide 95% lower confidence bounds on the reliability of some simple exponential systems. The same data was used with SPARCS to compare bounds. Berkbigler and Byers [4] made 1,000 simulation runs as opposed to 400 runs by SPARCS. The results are in Table VI. Both lower confidence bounds assume a uniform prior in the absence of data.

Mann's [56] bounds are compared to the lower confidence bounds for exponential series systems of El Mawaziny [21] and El Mawaziny and Buehler [22]. In addition, the simulation bound approximation developed by Grubbs [32] is compared. The same data was used with SPARCS to provide some similar bounds. The simulation bound should approach the bound of Grubbs [32]. Table VII shows the comparison between techniques. To obtain the values from SPARCS, the uniform

TABLE VI

95% LCB ON SIMPLE EXPONENTIAL SERIES SYSTEMS
AS PER BERKBIGLER AND BYERS
AS COMPARED WITH SPARCS

Number of Systems	Data (r, T)	Berkbigler and Byers [4]	SPARCS
2	(1,100)		
	(3,140)	.921	.920151
	(2,200)		
	(3,225)		
5	(2,480)	.914	.917189
	(5,400)		
	(4,500)		

r = number of failures
T = total time in test per test unit

TABLE VII

LCB ON SIMPLE EXPONENTIAL SERIES SYSTEM COMPARING SPARCS WITH OTHER MATHEMATICAL TECHNIQUES

Number of Systems	Data (r, T)	Confidence Bounds	El Mawaziny [21]	El Mawaziny and Buehler [22]	Mann [56]	Grubbs [32]	SPARCS
3	(4,25.53) (3,56.47) (2,23.47)	.90	.700	.738	.699	.649	.647609
3	(2,35.97) (2,14.61) (2,62.54)	.90	.732	.811	.738	.693	.689471

r = number of failures
T = total time in tests per test unit

prior assumption was removed to obtain a fiducial prior for comparison purposes.

The confidence bound provided by SPARCS is lower than the other confidence bounds. However, the bounds provided by SPARCS closely approximate the bounds of Grubbs [32] as would also be expected since the bounds provided by Grubbs [32] were developed to closely approximate the bounds provided by simulation techniques.

CHAPTER VI

MISCELLANEOUS MATHEMATICAL ASPECTS

Introduction

SPARCS employs a combination of many techniques and procedures. A discussion of some selected techniques and parts is presented in this chapter.

Beta and gamma proprietary routines from the International Mathematical and Statistical Library (IMSL) [14] are used in SPARCS. These routines (MDBETI and GGTMAJ) provide component reliabilities for system reliability assessment. Analysis of these routines was performed in two steps.

An error analysis on the values generated by these routines was provided by Keun K. Lee [40] in a master's report at Oklahoma State University. The inverse beta (MDBETA) and gamma (GGTMAJ) were compared to forward routines of the same type. After the routines were incorporated into the model, SPARCS was tested to see if the routines were producing component reliabilities consistent with hand calculated component reliability values as described in Chapter V.

Two pseudo-random number generation routines are utilized by SPARCS. RANF is a routine that is coded in PL/1 and used with the beta components. GGU1 is an assembler language routine that is an

IMSL subroutine used with the gamma type components. These routines are briefly compared and analyzed.

A sorting technique developed by Donald Shell [81; 37, pp. 84-86] is used to sort the system (un)reliability point estimates into ascending order. This technique is very efficient for sorting large blocks of numbers. Since the number of simulation runs (and corresponding reliability point estimates) may tend to be large for system assessment, this technique is chosen to sort the system (un)reliability values.

Finally, sample size determination is discussed. The number of simulation runs (sample size) is left to the discretion of the user. A brief presentation of sample size determination is provided followed by a discussion of sample size versus accuracy tradeoffs discovered by SPARCS. This technique represents the "standard" sample size determination technique frequently used in simulation experiments.

Inverse Beta and Gamma Analysis

IMSL Error Analysis

Lee [40] analyzed the IMSL routines utilized in SPARCS to determine the amount of error inherent in these routines. First, the incomplete (forward) beta and gamma distributions were developed as polynomials which provided probability values for given appropriate percentage point values. These probability values were then compared with the probability values in Pearson's tables of the Incomplete Beta and the Incomplete Gamma functions [72, 73].

For the beta distribution, one hundred percentile point values (from .01 to .50 in increments of .01) were used to correspond to the values used in Pearson's tables. The incomplete (forward) beta distribution was used to obtain 50 probability values. These probability values were used as input values to the inverse beta distribution function in the IMSL routine (MDBETI) and the percentage point values compute. The difference between the input and output values of the percentage points were considered as error.

For the gamma distribution, the IMSL random gamma deviate generator, GGTMAJ, generated a set of 50 values. These random deviates (percentage point values) were used as input to the incomplete (forward) gamma to obtain probability values. Since GGTMAJ used a random number generator to produce the random gamma deviates the results of the incomplete (forward) gamma should be a uniform distribution in the range (0,1). The Kolmogorov-Smirnov (K-S) two sample goodness of fit test results were used to compare the probability values from the incomplete gamma distribution and the theoretical uniform distribution.

Table VIII shows the error analysis for the IMSL MDBETI routine. The MDBETI routine has a maximum absolute error value of .0008 from fifty tests over a range of parameter values from $a = 1$ and $b = 1$ up to $a = 100$ and $b = 100$ in various combinations.

Table IX shows the error analysis for the IMSL GGTMAJ routine. The results of fifty K-S goodness of fit tests over various combinations of parameter values from $n = 1$ and $t = 1$ up to $n = 100$ and $t = 100$ over a range of degrees of freedom from 1 to 200. The hypothesis for the K-S goodness of fit test is

TABLE VIII

MAXIMUM ERROR FROM MDBETI

a	b	Input Percentage Point	Probability	Output Percentage Point	Absolute Error
1	1	.97	.97000	.96999	.00001
20	20	.71	.99718	.70992	.00008
50	50	.66	.99948	.69996	.00004
50	40	.69	.99607	.68999	.00001
100	100	.64	.99997	.63998	.00002
100	110	.62	.99999	.61993	.00007

TABLE IX

KOLMOGOROV-SMIRNOV TEST FOR GGTMAJ

n	t	Maximum Absolute Difference*	Observed Significance Level
1	1	.0666	$p > .2$
13	4	.0528	$p > .2$
5	29	.0676	$p > .2$
50	50	.0595	$p > .2$
100	100	.0775	$p > .2$

*Difference between empirical cumulative distribution function and uniform cumulative distribution function.

H_0 $F(X) = G(X)$, for all X

H_1 $F(X) \neq G(X)$, for at least 1 value of X

where F(X) is the forward gamma and G(X) is the IMSL gamma deviate generator. The observed significance levels were greater than .2 for most of the tests on GGTMAJ.

Pseudo-Random Number Generators

RANF

RANF is a pseudo-random number generator that provides a number from a uniform distribution over the range (0,1). RANF is a composite of three multiplicative congruential generators as proposed by Maclaren and Marsaglia [51].

In 1968, Marsaglia [59] showed that the standard multiplicative congruential method used for most pseudo-random number generators produced values with nonrandom characteristics. Thus, Marsaglia and Bray [60] developed the procedures as incorporated in RANF to remove these inconsistencies.

RANF essentially uses numbers from one generator to shuffle numbers obtained from a second generator. This second generator is used to shuffle numbers from a third generator. The value obtained from the third generator is the pseudo-random number over the range 0 to 1. RANF, as a composite generator, has been subjected to tests, by von Gelder [90] and Chandler [12], which have yielded some very good results. RANF generally passes all known tests of randomness. These tests show good results even if the component generators used to provide the pseudo-random digits are not of the highest quality. However, RANF has not been exhaustively tested and, as with any such routine, there is the possibility of some nonoptimal results given favorable situations.

GGU1

GGU1 is a proprietary subroutine of the International Mathematical and Statistical Library (IMSL) [14]. It is used in conjunction with the IMSL subroutine GGTMAJ to provide random gamma deviates for use in obtaining component reliabilities for gamma type components.

GGU1 is written in Assembler language and provides a pseudo-random number from a uniform distribution over the range 0 to 1. It is a multiplicative generator that manipulates the binary digits (bits) and groupings of bits (bytes) to produce a pseudo-random number.

The working of the generator is basically simple. Initially, the lower order bytes of the double precision seed are zeroed. A logical "or" is performed against the fifth byte of the seed to ensure a nonzero number for future multiplication. This value is multiplied by a constant (which may be altered) to produce a third number. The integer part of this number is truncated to leave the fractional part which is the pseudo-random number over the range 0 to 1.

Schmidt and Taylor Tests

Both RANF and GGU1 were subjected to some simple tests for goodness of fit, randomness and autocorrelation proposed by Schmidt and Taylor [79, p. 229] and Poore [74, p. 101]. First a frequency distribution for each generator was obtained for different seed values. In each case, visual inspection showed no unusual skewness as would be expected because of initial tests on these generators. Next, the runs test and test for autocorrelation [74, p. 241] were applied to each generator. Basically, the runs test is used to test the "randomness" of a sequence of numbers. Although numbers may fit a uniform distribution, this does not guarantee "randomness" [74, p. 241]. The autocorrelation test checks for the tendency of some numbers to be followed by other numbers. Thus, the amount of autocorrelation between each value from a pseudo-random number generator is examined.

The test runs show very good results. Table X shows the results of these tests.

TABLE X

ABSOLUTE Z VALUES ($|Z|$) FOR RUNS AND AUTOCORRELATION TESTS ON RANF AND GGUI*

	RANF	GGU1
Test for Randomness	.885	.794
Autocorrelation Tests	.727	.510

*In both cases, the limiting value is 1.96.

In every case, the resulting values are well below our limit and show that each generator does produce values that show very little autocorrelation and a high degree of randomness.

The Shell Sort

There are as many sorting routines as there are sorting needs. There are insertion sorts, exchange sorts, selection sorts, special purpose sorts and many others. SPARCS needed a sorting routine that would sort a large block of numbers with an efficient use of time and core storage. The sort chosen needed to be an internal computer sort without the aid of peripheral storage devices. The Shell [81; 37, pp. 84-86; 82] sort was chosen as the best and simplest sort for our purposes.

The Shell sort is initially discussed in a paper by Donald L. Shell in 1959 [81]. It divides the record of information to be sorted into groups of diminishing size. This grouping provides each element to be sorted with the capability of moving many positions in one jump. This group size diminishes until the final sort is just a straight insertion sort. The insertion sort considers one element at a time and compares it with a previous element or a group of elements that are sorted in the desired order.

The size of the decreasing increments is very important. Although there is no "best" size for a large number of elements to be sorted, it has been determined that some group sizes are better than others. In choosing group sizes, execution time is the main factor that needs to be minimized. Execution time is determined by 5 factors: 1) size of the record, 2) number of sorting passes, 3) the number of comparisons, 4) the number of moves, and 5) the sum of the increment values or group sizes [37, pp. 84-86]. In SPARCS, the size of the record (number of simulation runs) is determined externally. Therefore, to minimize execution time, the other 4 factors will have to be kept to a minimum.

To choose the diminishing increment sizes, let

h_i = group or increment size of i^{th} group

N = record size

Then, let

$$h_1 = 1, \quad h_{i+1} = 3h_i + 1$$

and stop when

$$h_{i+2} \geq N.$$

The first increment (group size) is h_i and decreases until $h_i = h_1 = 1$. For example, if

$$N = 1000$$

then:

$$h_1 = 1$$

$$h_2 = 3(1) + 1 = 4$$

$$h_3 = 3(4) + 1 = 13$$

$$h_4 = 3(13) + 1 = 40$$

$$h_5 = 3(40) + 1 = 121$$

$$h_6 = 3(121) + 1 = 364$$

$$h_7 = 3(364) + 1 = 1093.$$

Here,

$$h_{i+2} = h_7 = 1093 > N = 1000$$

so that the first group size (increment) is

$$h_i = h_5 = 121.$$

The increment will continue to diminish on each pass until $h_i = 1$ which reduces to the straight insertion sort. In this manner, record elements will be moved closer to their correct position in large jumps before the final simple insertion sort.

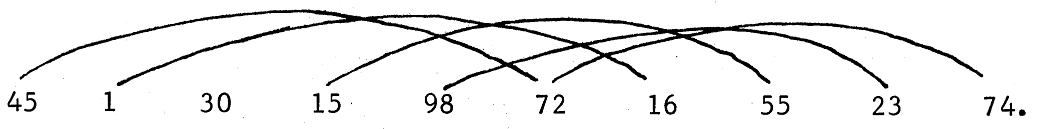
For example, if 10 items are to be sorted with the following increments

$h_1 = 1$

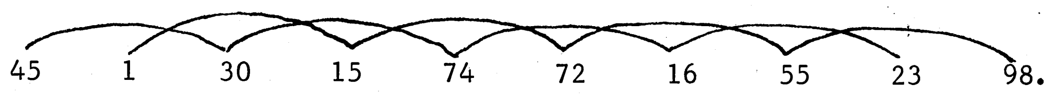
$h_2 = 2$

$h_3 = 5$

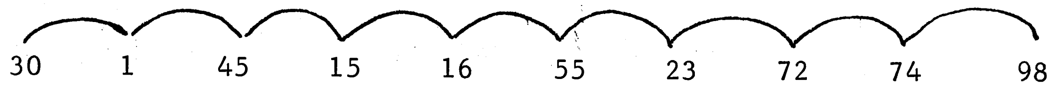
the sorting would proceed as follows.



Sorting with $h_3 = 5$ would yield



With $h_2 = 2$ yields



The final simple insertion sort yields

1 15 16 23 30 45 55 72 74 98.

For a large number of items, the Shell sort is more efficient than any of the other sort methods mentioned earlier [81]. The coding of the Shell sort does not require extensive core. Consequently, this sorting routine was chosen for use in SPARCS over other sorting methods analyzed.

Sample Size Determination

Conventional Methods

Conventional methods for sample size determination revolve around the Central Limit Theorem. This theorem, which is the basic theorem used in statistical inference, stipulates that if a universe has a mean μ and a finite standard deviation σ, then the distribution of sample means, $\bar{x}$, approaches a normal distribution with mean μ and standard deviation $\frac{\sigma}{\sqrt{n}}$ as the sample size increases [13, p. 240; 86, p. 259]. This theorem holds true regardless of the type of universe under analysis (assuming unimodality).

The Central Limit Theorem is based upon the law of large numbers. This law states that sample means are approximately centered about the universe mean. These sample means tend to become more closely clustered about the universe mean as the sample size becomes larger. This relationship is represented succintly by Tchebycheff's inequality which states that for any set of data $x_1, \ldots x_n$ and any $k \geq 1$,

$$P(|\bar{x} - \mu| \geq k\sigma) \leq 1 \ . \qquad (4)$$

Thus, the probability of selecting a randomly selected value, $\bar{x}$, which differs from the universe mean, μ, by at least k standard deviations will not exceed $\frac{1}{k^2}$ [13, p. 239].

Because of the Central Limit Theorem and the law of large numbers, interval estimates can be used to provide information about the universe mean, μ, and its relationship to a sample mean, $\bar{x}$. A probability relationship concerning the deviation of a sample mean from the universe mean is given by

$$P(\bar{x}-Z_{\alpha/2}\ \sigma_{\bar{x}} \leq \mu \leq \bar{x}-Z_{\alpha/2}\,\sigma_{\bar{x}}) = 1-\alpha \tag{5}$$

where Z is standard deviation units from a standard normal distribution, and $\sigma_{\bar{x}}$ can be estimated by

$$\sigma_{\bar{x}} = \frac{s}{\sqrt{n}} \tag{6}$$

for large samples and s is the standard deviation of the sample [79, p. 260; 74, p. 266-267].

Using the above information, the distribution of sample means can be standardized by

$$Z = \frac{\bar{x}-\mu}{\sigma/\sqrt{n}} \tag{7}$$

where Z is normally distributed with a mean of 0 and a standard deviation of 1. For the analysis in (5), the universe standard deviation must be known. When the universe standard deviation is not known and must be estimated, the Student-t distribution provides the appropriate distribution of the form

$$t = \frac{\bar{x}-\mu}{\frac{s}{\sqrt{n-1}}} \tag{8}$$

The Student-t distribution with n - 1 degrees of freedom, although not normally distributed, approaches the normal distribution as the sample size increases (where s is the sample standard deviation and n - 1 adjusts for small sample bias). Since most sample sizes greater than 30 observations are considered large, the normal approximation discussed below is used for the distribution of t [13, p. 266].

If the maximum allowable deviation of $\bar{x}$ from μ at a specified confidence level is represented as

$$\bar{x} - \mu = \delta \qquad (9)$$

then the sample size, n, can be obtained iteratively as:

$$Z = \frac{\delta}{\frac{s}{\sqrt{n}}}$$

$$\delta = \frac{Z \cdot s}{\sqrt{n}}$$

$$\sqrt{n} = \frac{Z \cdot s}{\delta}$$

$$n = \left(\frac{Z \cdot s}{\delta}\right)^2 \qquad (10)$$

where Z = standard normal statistic - N(0,1)

s = standard deviation of the sample

δ = maximum allowable deviation between $\bar{x}$ and μ.

Equation (10), then, would be the applicable formula for calculation of the required sample size for a specified confidence level for use with SPARCS.

For a system with a variance (s^2) of .000327, a standard deviation (s) of .018076, and a maximum allowable deviation (δ) of .001, the sample size, at the 95% confidence level, would be calculated as follows.

$$n = \left(\frac{Z \cdot s}{\delta}\right)^2 = \left(\frac{1.96 \cdot .018076}{.001}\right)^2$$

$$n = 1{,}255 \qquad (11)$$

Consequently, it would take 1,255 simulation runs to provide a sample mean, $\bar{x}$, that would have a maximum allowable deviation of .001 from the true population mean at a confidence level of 95%. Furthermore, the standard deviation used in the sample size calculation would probably have to be obtained as a result of a sample run.

This procedure may also be used to obtain a confidence level that may be applied to the results of any simulation run without regard for sample size. The sample mean, $\bar{x}$, and standard deviation, s, are derived for each system run by SPARCS. If the maximum allowable deviation (δ) is specified, a confidence level can be associated with the results of a specified sample run. For example, if a run of a system produces a standard deviation of .010204 in 400 runs, then for an allowable deviation of .001, the confidence level would be associated with 1.96 standard normal deviates or a 95% confidence level.

$$Z = \frac{\delta}{\frac{s}{\sqrt{n}}} = \frac{.001}{\frac{.010204}{\sqrt{400}}}$$

$$Z = 1.96 \qquad (12)$$

Thus, there is a 95% confidence that the population mean is within a maximum allowable deviation of .001 from the sample mean.

Sample Size Problems

Although the above analysis of the sample size problem seems very succinct and explicit, Burdick and Naylor [9] and Naylor, Balintfy, Burdick and Chu [68, p. 332ff.] point to sample size determination as one of the major simulation problems. The problem revolves around two basic elements: 1) how many observations to measure and 2) when to begin measurement.

In most situations, practitioners appeal to the Central Limit Theorem, as presented above, relying on the assumptions of normality and independence to provide a sample size value [68, p. 335]. However, the efficiency of this method has been questioned by Fishman [29], Graybill [31] and Cooley [17] as to the number of samples required and the slowness of normality convergence. With some knowledge of the distribution of the universe to be sampled, the sample size can be determined more efficiently in some cases [29, 31, 17]. However, to my knowledge, there is no analysis that purports to classify different distributions of reliability values (assuming they are different) obtained by analysis of different system configurations. Consequently, there is no method for efficiently determining sample size in SPARCS other than that proposed above.

Secondly, the problems of autocorrelation [29, 31, 17] steady state and startup bias, as discussed by Conway [16], Moran [65, p. 87] and Morse [66, p. 61] directly affect the problem of when to begin measurement. However, these are areas about which there is very little in-depth information and consensus as can be seen by analyzing the steady state discussion by Schmidt and Taylor [79, p. 346] and Conway [16]. Consequently, in practice, these problems tend to be arbitrarily determined or ignored.

Model Sample Size Considerations

The number of simulation runs (sample size), for each system under consideration by SPARCS, is supplied by the user. This supplied value may be calculated by equation (7) or arbitrarily assigned. Since there is no formal knowledge concerning resulting distributions from system reliability assessment, these are the only two methods currently available for sample size determination, to my knowledge.

Most of the values obtained for validation purposes in Chapter V, were obtained from 400 iterations or less. These values were compared with literature values obtained from sample sizes of 1,000 iterations up. Comparison of the values in Chapter V shows a very close correlation between answers obtained from the smaller sample sizes of SPARCS and the large sample sizes from literature. This phenomenon seems to follow for each comparison run made. Thus, it seems that reasonable accuracy can be obtained with SPARCS from smaller sample sizes.

There are no explicit reasons proposed for these results. However, there are two situations that may contribute to this phenomenon. The first possible explanation is based upon the idea that SPARCS does not simulate discrete events but instead simulates system reliability values. An empirical distribution of reliability values is the purpose and direct result of this simulation. Consequently, this type of analysis may have an effect on the sample size. Secondly, the conventional sample size determination methods discussed earlier were developed to pertain to any unimodal distribution. This encompasses a wide range of possibilities requiring a certain amount of "overkill" to accomplish its objectives. However, it seems that the empirical distributions, as generated by SPARCS, do not require as large a sample size as would be suggested by those methods to achieve adequate results. Perhaps, either one or both of these situations may be responsible for the satisfactory results obtained from SPARCS relatively small sample sizes.

CHAPTER VII

PROCEDURE DESCRIPTIONS AND JCL ASPECTS

Introduction

MAPS (Model for the Analysis of Probabilities of Systems) written by J.L. Burris [11] is the basis around which SPARCS is developed. MAPS provides an estimate of the system reliability as a function of the reliabilities of the components. Originally programmed in two parts, MAPS I and MAPS II were combined to produce a one pass version of MAPS. This version was modified to provide for simulation and other capabilities. Consequently, many of the procedure names found in MAPS are also found in SPARCS.

SPARCS contains a Shell sort, two random number generators, certain proprietary routines from the International Mathematical and Statistical Library (IMSL), percentile calculation routines, simulation capabilities, and an MTBF (mean-time-between-failure) routine not found in MAPS. SPARCS is designed to call the MDBETI and GGTMAJ routines from the IMSL library. If the facility using SPARCS does not subscribe to the IMSL library, these routines may be used as a load module.

The storage requirements of each procedure and array is presented with a discussion of the dynamic storage concept utilized by SPARCS. JCL aspects of the model are discussed with and without the load module. Appendix C contains a complete source program listing which may be used for reference during the discussion of each procedure.

Procedure Descriptions

UNITED (MAIN)

UNITED is the main PL/1 procedure. It assumes control of the program calling other procedures when necessary, controlling the simulation process, inputting and outputting information, processing modules, calculating percentiles, sorting system reliabilities and determining when to stop.

Initially UNITED reads in information needed to prepare for procedures that follow. Information about the simulation process, MTBF calculations and dynamic storage development is read first. Data for system identification, the type of analysis desired (reliability or unreliability), provision for user or program supplied component and system labels, information about the input form of minimal states (binary or hexadecimal) and an indication of whether punched output is desired follows. Next, the appropriate storage for dynamic arrays is allocated and UNITED begins its iterative calculations.

Entry point CALCUL is located in UNITED. CALCUL, entered after the simulation process is completed, provides statistical information on the arithmetic mean, standard deviation, average reliability, and MTBF for the system. The Shell sort [81] is used to sort the system reliabilities or unreliabilities in ascending order. It breaks the items to be sorted into groups which are decreased in size following each sort procedure. Information is moved between these groups until the items to be sorted are in the order desired. Tests and calculations show the Shell sort as very efficient in its use of computer time and storage when dealing with a large number of items. CONF, in conjunction with CALCUL, then provides percentiles for both the reliabilities or unreliabilities and the MTBF if desired. When this is finished, UNITED terminates the program or reads a new system to be analyzed, whichever is applicable.

FNPUT and HEXIN

FNPUT procedure is used to input the minimal states for the system and for each module. The minimal states are represented in either binary (bit string) or hexadecimal (character string) notation. A code (KODE) is used to indicate how the minimal states are represented. If hexadecimal notation is used, entry point HEXIN converts hexadecimal input to binary notation for use later in the program since binary notation is necessary to generate the probability equation(s).

HEXIN procedure uses a "table lookup" approach to convert from hexadecimal input to binary notation. The hexadecimal option is allowed to enable the user to reduce the number of characters necessary to represent a minimal state, especially for large systems or modules.

EQGEN

Probability equations are generated in EQGEN using Poincaire's method as the primary algorithm. The minimal paths (or cuts) are combined and accumulated to form the equation. If the system configuration is arranged into two or more modules, the probability equation is generated for each module, in addition to the equation generated for the system. The same computational process is used to generate both the system and module equations.

For a system or module having n minimal states, the probability equation has a maximum of $2^n - 1$ terms. Because of the cancellation of duplicate terms by EQGEN, the actual probability equation contains only a fraction of the maximum number of terms. Each minimal state is introduced and combined with the previously generated terms to form new terms. Terms that have zero coefficients are removed before the next minimal state is introduced. Terms of the probability equation are initially stored in an array called TERMS. Coefficients are stored in an array called COEF.

OUTI

Information about the system being analyzed, the probability equations for the system and each module are handled in OUTI. These procedures are handled by three major entry points. OUT1 is used to assign labels to the elements of the system and print control information concerning the system. OUT2 is used to assign labels to the elements of a module and print control information for that module. OUT3 is called to print the minimal paths and probability equation for each module and the system.

OUT1 and OUT2 are also designed to store the necessary historical information about each component for further use in the simulation. Information as to the historical number of failures (FAILS), number of successes or testing time (PORT), and type of component (TYPE) is stored. Next, the SIMULATE procedure is called to produce the results for the first simulation run. All other simulation runs are performed by entry point SIMOUT of OUTI using this historical information stored during execution of OUT1 and OUT2.

A provision allowing the user to assign labels or the computer to assign labels to the components and modules is incorporated into OUTI. If the user wishes to assign particular names, OUTI uses these names as labels. Otherwise, the labels are assigned by OUTI. Components are assigned a number in order from 1 to 128. Modules are assigned labels in order from A - Z, A1 - Z1, A2 - Z2, A3 - Z3, A4 - Z4, etc. Thus modular elements are assigned labels beginning with an alphabetic character and nonmodular elements are assigned numeric labels.

OUT II

Output concerning the reliabilities of each component, module and the system are produced by the OUT II procedure. The output is in the following general order:

1) Identification of the system or module.

A listing of:

2) the reliability (unreliability) value for each component as obtained in the first simulation run (This value is to be used as a check figure),
3) the type of analysis to be performed on each component, i.e., either Beta or Gamma,
4) historical data about the total testing time (if Gamma) or the total number of successes (if Beta) for each component,

5) the total number of historical failures observed with each component, and

6) the computed reliability and unreliability for the module.

For each additional module, steps 2 through 6 are repeated. Next,

7) a listing of reliabilities (unreliabilities) for the system consisting of both modular and nonmodular reliabilities),

is provided and is the final output produced by OUT II.

COMPUTE

The COMPUTE procedure calculates a probability value for each module and combines these to produce a system reliability. The probabilities for each module are calculated first and substituted into the system probability equation to compute the values of the system reliabilities. Each system reliability is stored in an array called RELSTO for later use.

The reliabilities for the modules and/or the system are accumulated on a term-by-term basis. A three step process is used. First, the product of the reliability (unreliability) of each element in a term, denoted by a "1" in the bit string, is found. Next, the product found in the first step is multiplied by the coefficient of the term. This computes the reliability (unreliability) attributable to that term. Finally, the reliability (unreliability) calculated in step two is added to the accumulated reliability from previous terms.

Both the system reliability and unreliability are a product of the COMPUTE procedure. If a reliability analysis is specified, the analysis uses component reliabilities to provide the system and module reliabilities. If an unreliability analysis is desired, the component unreliabilities are used to calculate the system and module unreliability. In either case, the complementary value of the specified analysis (reliability) is obtained by subtracting the result of that analysis (system reliability) from 1 (1 - system reliability = system unreliability).

SLINE, PRINTER and DLINE

The SLINE procedure is used to show continuation of a system reliability equation. If a reliability equation requires more than 120 spaces on any line, an asterisk (*) is placed at the end of that line to indicate the continuation of the equation onto the next line. The PRINTER procedure is called to place the asterisk (*) at the end of the continued line.

The DLINE procedure keeps track of the page number as each new page of output is initiated. It also provides for the printing of "**CONTINUED**" each time a new page is started.

SIMULATE

The SIMULATE procedure calculates a reliability or unreliability interval estimate for each component based on historical test information provided for that component. Each component has a reliability (or unreliability) value calculated for each simulation run. These values are generated from either a Beta or Gamma prior depending on the type of component being analyzed.

The BETASUB procedure is used with Beta type components. It has two purposes: First, it calls the MDBETI routine from the IMSL (International Mathematical and Statistical Library) library which provides interval estimates on the reliability (unreliability) of each Beta component. Next, after all the simulation runs are finished, it calculates the average reliability (unreliability) for each component to be used in the calculation of the average system reliability (unreliability).

The GAMASUB procedure is used with Gamma (time-to-failure) components. It has the same basic purpose as the BETASUB procedure except that it calls the GGTMAJ routine from the IMSL library. Both interval estimates and average reliabilities (unreliabilities) are generated by GAMASUB from historical data provided for each component.

Both BETASUB and GAMASUB procedures provide essentially the same information for their respective component types. In both instances, the historical component data is adjusted to provide uniform priors in the absence of data.

RANF

The RANF procedure provides a random number generator that is used in the Monte Carlo process. RANF, the name of the pseudo-random number generator, was provided by Dr. J.P. Chandler of Oklahoma State University. Essentially it is a composite of three multiplicative congruential random number generators. Tests have shown it to be a very good generator with few vices.

IMSL Routines

MDBETI and GGTMAJ are two IMSL (International Mathematical and Statistical Library) routines incorporated into SPARCS to be called by the SIMULATE procedure. If the IMSL library is available at the facility using SPARCS, the appropriate routines may be called directly. If the library is not available, the appropriate routines may be incorporated as load modules. If the load module is used, the facility must have FORTRAN-G, PL/1-F, and Assembler F language capabilities.

GGTMAJ is used with exponential (time-to-failure) type components. It generates a gamma random deviate using a rejection method. Two other routines, GGBTA and GGU1, are called by GGTMAJ during processing, one of which (GGU1) is an IMSL pseudo-random number generator. Historical data about each exponential component is input as parameters and gamma random deviates are returned. GGTMAJ and GGBTA are in FORTRAN and GGU1 is in Assembler.

MDBETI is used with Poisson process (pass-fail) components. It generates a Beta deviate from the inverse beta probability distribution function in the exclusive range (0,1). MDBETA and UERTST are called during processing both of which are in FORTRAN the same as MDBETI. Historical data about each pass-fail component is input and a beta deviate is output in the range (0,1). Whether using the load module or calling the IMSL routines directly from the IMSL library, familiarity with JCL capabilities is a necessity.

System Size and Storage Capacity

In SPARCS, storage is dynamic and a function of the size of the system being analyzed. Storage size is determined by three items. The OS (operating system) occupies a certain amount of core. This requirement is static and cannot be affected by the programmer. Consequently, for our purposes, it is disregarded. Second, storage is required to hold the actual recorded program statements (object program). This storage is static and requires about 72,000 bytes for SPARCS. Finally, storage for variables and arrays must be assigned. The dynamic capabilities of PL/1 are used when possible to save storage in the utilization of arrays. Thus, the arrays are allowed to expand or contract as the size of the systems under analysis changes.

In discussing the system and storage size, three aspects are analyzed. The maximum system limitations for use with the program are presented. Second, the amount of core used with each procedure for statement storage is given. Finally, the amount of storage required for each array is examined. Dynamic arrays are identified and their core range specified where possible.

System Size

The system size limitations are presented in Table XI and are the same as required for the Burris program [11]. These values represent a maximum. If systems of a smaller size are used, SPARCS is designed to release unused core for use elsewhere. This is done automatically by the program and does not require any special manipulations by the user.

Procedure Storage Requirements

The core requirements to store the statements from each procedure are listed in Table XII. Since the IMSL routines are subroutines called by BETASUB and GAMASUB, their storage requirements are included in the storage requirements of these routines. The procedures vary substantially in size but OUTI is by far the largest.

TABLE XI

SPARCS-II SYSTEM LIMITS

	Maximum	Marginal Storage Required (Bytes)
Number of Modules per System	128	11
Number of Elements per Module	128	2,015
Number of Terms in Probability Equation of System or Module	2,000	28
Number of Systems per Run	No Limit	--
Number of Simulation Runs	No Limit	--

TABLE XII

PROCEDURE STORAGE REQUIREMENTS FOR SPARCS

Procedure	Storage Required (Bytes)
UNITED (MAIN)	972
CONF	268
FNPUT	268
HEXIN	540
EQGEN	408
OUTI	10,596
SLINE	364
PRINTER	240
DLINE	240
COMPUTE	348
OUTII	444
SIMULATE	336
BETASUB	292
GAMASUB	292
RANF	260
Total	15,868

Array Storage Requirements

Core information about each array is presented in Table XIII. If an array is static (does not vary), the dimension size and the storage bytes required are given. For example, the array CODED is static, dimensioned 16, and requires 16 bytes of core storage.

If arrays are dynamic they are identified as adjustable. Adjustable arrays are of two types: 1) those that have an upper limit and 2) those that are not limited. If an array has an upper limit, the maximum dimension size and core reqruiement is given. For example, DCOM is adjustable with a maximum dimension size of 128 and a maximum core size of 384 bytes. Because they are adjustable, these arrays may take on any dimension value below the maximum with an appropriate reduction in core size. Therefore, DCOM may have a dimension size from 0 to 128 items and require from 0 to 384 bytes of core.

Some arrays are indeterminate and are identified as such by two asterisks in the storage column. Indeterminate arrays are arrays with no upper limits to core size. This occurs when one or more of the dimension values for these arrays are not restricted. Generally, the use of the number of simulation runs as one of the dimension values is the primary cause for an indeterminate array. Since the number of simulation runs is not restriqted, these arrays may take on any size necessary to accomodate the required information.

TABLE XIII

ARRAY STORAGE REQUIREMENTS FOR SPARCS

Array	Description	Dimension (Maximum)	Storage (Bytes)
CODED	Set of 4 binary characters that correspond to each hex character	16	16
COEF	Coefficient of equation terms	1,500	4,500
COMPS	Labels for nonmodular elements	128	384
DCOM	Storage array for terms in module groups	Adjustable (128)	0 - 384
DERMS	Terms of the system or module probability equation	Adjustable (128, 1500)	0 - 384,000
DOEF	Coefficient of equation terms	Adjustable (128, 128)	0 - 65,536
DIERM	Terms of system or module equation	Adjustable (128)	0 - 384
FAILS	Number of component historical failures	Adjustable (128, 128)	0 - 229,376
FERMS	Terms of the system or module probability equation	(1500)	24
HEX	Table of hex characters	16	8
KOMPS	Default labels for nonmodular elements	128	384
LA	Dummy variable used in calculation of percentiles	20	80
MDESCR	Description of modular elements	128	8,960

TABLE XIII (CONTINUED)

Array	Description	Dimension (Maximum)	Storage (Bytes)
MINPTH	Minimal states of module or system	256	4,096
MODSY	Labels for modules	128	384
MODSYM	Default labels for modules	128	384
N	Variable used in random number generation	128	572
PORT	Number of component historical successes or total testing time	Adjustable (128, 128)	0 - 229,376
PREL	Intermediate storage for reliability calculations	Adjustable	0 - 1,024
R	Parameter for the gamma variate in GGTMAJ	1	4
REL	Values for element probabilities per module or element	Adjustable	0 - 1,024
RELSTO	Array used in sorting the system reliabilities	Adjustable (SIMNUM *1 as maximum)	0 - ? **
SIMCOM	Variable which holds the number of components for each module and/or system	Adjustable (128)	0 - 384
SLAB	Labels for elements of system or modules	Adjustable (128)	0 - 384
SREL	Storage array for computed module probabilities	Adjustable (128)	0 - 384
TERMS	Terms of the system or module probability equation	2,000	24,000

TABLE XIII (CONTINUED)

Array	Description	Dimension (Maximum)	Storage (Bytes)
TYPE	Specifies whether each component is a gamma or beta component	Adjustable (128, 128)	0 - 229,376
WA	Used in calculation the average reliability for a gamma component	Adjustable (total test time as maximum)	0 - ? **
Z	Used in Shell Sort routine	Adjustable (SIMNUM as *1 maximum)	0 - ? **

**Upper limit of array size indeterminate because one of the upper limits of the array has no upper limit set on it.

*1 The number of simulation runs are the upper limit for this array. There is no restrictions on the number of simulation runs.

Note: Some of these values for core are hand calculated. Consequently, they may be smaller than represented on certain systems.

JCL Aspects

The JCL aspects of SPARCS can become intricate although not overly difficult. Appendix A and B illustrate the JCL used at Oklahoma State University to execute the program. The JCL aspects of running SPARCS will be discussed both with and without the use of a load module. The JCL cards needed are discussed in general terms since the exact format of the cards used will depend upon the computer system and the facility.

Appendix A contains the JCL for referencing the IMSL package as a part of the system. If the IMSL package is referenced directly, a JCL card is needed in the LKED section to reference the IMSL object program and link it with SPARCS. The FORTRAN and Assembler libraries must be linked as in the load module in case some library functions are called.

Appendix B provides the JCL for the IMSL routines as load modules on an IBM 360-65. The load modules are composed of FORTRAN G and Assembler F routines. The FORTRAN routines are grouped and preceded by an EXEC card for FORTRAN G. This execute card need only compile the routine. Following the routines should be a SYSIN card for FORTRAN. The Assembler routine is preceded by an EXEC card for Assembler F and followed by a SYSIN card for Assembler. This routine also need only be compiled. The main PL/1 program follows the IMSL routines. The basic JCL required to run and PL/1 program is adequate except for the LKED (link-edit) step. The LKED step requires a SYSLIB card to reference the FORTRAN and Assembler libraries for the load module routine. This allows the routine to use any stored functions they may need peculiar to that language. This is only necessary if a routine calls a stored function. The GO. JCL card need only refer to the PL/1 program since all output is done there.

Examples of the cards referred to are identified with asterisks in Appendix A and B. These were used with the IBM 360-65 at Oklahoma State University in Stillwater, Oklahoma. SPARCS with respective JCL has been checked out at Phillips Petroleum in Bartlesville, Oklahoma and Wright-Patterson Air Force Base in Dayton, Ohio on IBM 370 systems.

CHAPTER VIII

DOCUMENTATION OF SPARCS

Introduction

SPARCS, Simulation Program for Assessing the Reliabilities of Complex Systems, is a computerized procedure to provide confidence limits on the reliability or unreliability and the MTBF (mean-time-between-failures) for a system of any logical configuration. The components that comprise this system may be either attribute or time-to-failure components with no restriction as to their placement in the system. Interval estimates on the system (un)reliability and the MTBF, if desired, are provided by use of Monte Carlo techniques in conjunction with Bayesian component analysis.

A PL/1 program by J. L. Burris [11] called MAPS, is used to provide a system equation as a function of the component reliabilities (unreliabilities) from analysis of the system minimal states. The basic input-output format of the Burris program, the equation generation routine, and the modularity concept developed by Burris is the basic structure around which SPARCS is developed.

In SPARCS, the component reliabilities (unreliabilities) are obtained from statistical analysis of historical data provided for each component. It is assumed that data for the attributes components is obtained from Bernoulli processes and the time-to-failure component data is obtained from Poisson processes. It is also assumed that all components succeed or fail independently.

General Description

SPARCS is designed to provide statistical information about the reliability (unreliability) and the MTBF of a complex system of any logical configuration. To use SPARCS, the system under consideration must be capable of being represented as a logical network of minimal states. A minimal system success state is called a minimal path [48], and is defined by a specified smallest set of components, which if they are all operating properly, will guarantee system success. A minimal system failure state is known as a minimal cut [48], and is defined as a specified smallest set of components which, if they are all failed, guarantee system failure. This minimal state information is provided by the user and is analyzed using Poincaire's Theorem [48, 26].

In Poincaire's Method, the system reliability, or probability of success, can be calculated from the component reliabilities if the minimal paths are known. This reliability value is the lower confidence bound on the system reliability. Likewise, the system unreliability, or probability of failure, or 1-system reliability, can be calculated from the component unreliabilities, given the minimal cuts. This unreliability is the upper confidence bound on the system unreliability. This minimal state information along with component failure data history, the number of simulation runs desired, and other information is input into SPARCS. For attributes-type components, this failure data consists of accumulated prior tests and prior failures. For time-to-failure components, this data consists of prior testing time and prior failures. Optionally, system mission time is also input to provide MTBF information for the system. SPARCS employs Monte Carlo methods to obtain component reliabilities (unreliabilities) from Bayesian prior distributions whose parameters are the component prior test data. These prior distributions are beta for attributes components, and negative-log gamma for time-to-failure components.

In this program, we have incorporated a random number generator, RANF, developed in FORTRAN by Professor J. P. Chandler of Oklahoma State University, based on an algorithm developed by Maclaren and Marsaglia [51] and recoded in PL/1 for use with SPARCS. We also employ six library routines supplied by the International Mathematical and Statistical Library (IMSL). MDBETI, the inverse beta generator and GGTMAJ, the inverse gamma generator are referenced directly. GGTMAJ calls GGU1 and GGBTA library routines while MDBETI calls MDBETA and UERTST and seems to be somewhat time consuming.

A modularity concept is employed which enables large complex systems to be broken down into smaller subsystems or modules. These subsystems are analyzed individually and later combined to provide an analysis of the system as a whole. This concept, originally developed in MAPS [11], along with other advantages of the PL/1 language, such as binary and varying bit string capabilities, makes it possible to handle large complex systems with a considerable saving of time and computer storage.

The dynamic storage capability of PL/1 is used to overcome a major storage limitation of SPARCS. Early in the development of the program, the storage requirements became greater than the IBM 360 Model 65 system could accomodate. Using the dynamic storage concept, storage is allocated only when needed and released as soon as the program no longer needs that information. This produces a saving of between 250K to 300K for a medium-large system.

Basically, SPARCS takes input information about the system supplied by the user and generates a system reliability (unreliability) equation. This system equation is a function of the reliability or unreliability of the system components. Since each component is one of two basic types, Bernoulli or Poisson process, SPARCS must generate a random number to be used with each component. This random number along with the historical prior test information about each component is used to enter the appropriate distribution and provide a reliability or unreliability estimate for each component. This estimate is placed in the correct position in the system equation to provide an interval estimate for the system reliability or unreliability. An interval estimate is determined for each simulation run desired. These estimates are then ordered and statistical information about the resulting empirical distribution of system interval estimates is provided.

Output Description

SPARCS output is broken down into four major parts. Initially, a printout is provided of the system information read in by the user (Figure 4). The system identification and information about components, modules, etc. along with the minimal states for the system, either paths or cuts, are printed. From this information, the system reliability

reliability or unreliability equation is determined and provided. Since systems may consist of subsystems, the minimal states and the system reliability or unreliability equation uses letters to indicate subsystems (modules) and numbers to indicate components. Consequently in Figure 4, R_A denotes the reliability of subsystem A.

Next follows an analysis of each component that makes up the system (Figure 5). Each component has four lines of information. Line 1 gives the reliability or unreliability for that component provided by the first simulation run. This value is provided to give the user an idea of the general reliability or unreliability of that component. Since the components of our system are defined to be of two types, Line 2 specifies the type for this particular component. A Beta component is an attributes component using the inverse Beta to provide the component lower (upper) limit on the reliability (unreliability). A Gamma component is a time-to-failure component using the inverse Gamma to provide a lower (upper) limit on the component reliabilities (unreliabilities). Finally, Lines 3 and 4 provide the prior historical data parameters that are used to enter the appropriate distribution. If a component is a Bernoulli component, Line 3 is the total number of successes, P, obtained by testing similar components. Line 4 is the total number of failures observed in these component tests. If the component is a time-to-failure component, Line 3 is the total testing time, TIME, measured in units of required testing time observed in tests of similar components. Then, Line 4 is the total number of failures observed in this testing time.

```
SPARCS: EQUATION GENERATION ROUTINE
SIMULATION PROGRAM FOR THE ANALYSIS OF THE RELIABILITY OF COMPLEX SYSTEMS
COLLEGE OF BUSINESS ADMINISTRATION, OKLAHOMA STATE UNIVERSITY

SYSTEM IDENTIFICATION ............... SERIES-PARALLEL SYSTEM  (CASE 0)

NUMBER OF MODULES ...................   0
NUMBER OF NONMODULAR COMPONENTS .....   4
TOTAL NUMBER OF SYSTEM ELEMENTS .....   4
NUMBER OF MINIMAL PATHS .............   4
PUNCHED OUTPUT OF EQUATION .......... NO
LABELS SUPPLIED BY USER ............. NO

THE   4 MINIMAL PATHS FOR THE SYSTEM FOLLOW:
<1,3>
<1,4>
<2,3>
<2,4>
```

Figure 4-A

```
SPARCS: EQUATION GENERATION ROUTINE
SIMULATION PROGRAM FOR THE ANALYSIS OF THE RELIABILITY OF COMPLEX SYSTEMS ,
COLLEGE OF BUSINESS ADMINISTRATION, OKLAHOMA STATE UNIVERSITY

SYSTEM IDENTIFICATION ............... SERIES PARALLEL SYSTEM WITH MODULES

NUMBER OF MODULES ...................   2
NUMBER OF NONMODULAR COMPONENTS .....   0
TOTAL NUMBER OF SYSTEM ELEMENTS .....   2
NUMBER OF MINIMAL PATHS .............   1
PUNCHED OUTPUT OF EQUATION .......... NO
LABELS SUPPLIED BY USER ............. NO

THE   1 MINIMAL PATHS FOR THE SYSTEM FOLLOW:
<A,B>

                               SYSTEM RELIABILITY EQUATION (   1 TERMS)

  R      = R R
   SYS      A B
```

Figure 4-B

Figure 4. System Information Printout

Figure 5-A

```
                    SYSTEM RELIABILITY EQUATION (  9 TERMS)

R    = R R  + R R  - R R R  + R R  - R R R  + R R  + R R R R  - R R R  - R R R
 SYS    1 3    1 4    1 3 4    2 3    1 2 3    2 4    1 2 3 4    1 2 4    2 3 4


SPARCS: PROBABILITY COMPUTATION ROUTINE
(THE COMPONENT AND SYSTEM RELIABILITY INFORMATION IS FOR THE FIRST ITERATION ONLY)

MODULE AND COMPONENT RELIABILITIES FOR THE SYSTEM

R   = 0.759468           R   = 0.942294           R   = 0.913139           R   = 0.981560
 1    TYPE = BETA         2    TYPE = GAMMA        3    TYPE = GAMMA        4    TYPE = BETA
      P OR TIME =   46.00      P OR TIME =   50.10      P OR TIME =   50.20      P OR TIME =  147.00
      FAILURES =    4.00       FAILURES =    4.00       FAILURES =    4.00       FAILURES =    3.00

SYSTEM RELIABILITY = 0.984540                 UNRELIABILITY = 0.015460
```

Figure 5. Printout of System and Module Equation(s) plus Component Information

Figure 5-B

```
MODULE A
NUMBER OF COMPONENTS ................   2
NUMBER OF MINIMAL PATHS .............   2

THE   2 MINIMAL PATHS FOR MODULE A   FOLLOW:
<1>
<2>
                                        SUBSYSTEM RELIABILITY EQUATION ..... MODULE A    (  3 TERMS)

  R    = R  + R  - R R
   A      1    2    1 2


SPARCS: PROBABILITY COMPUTATION ROUTINE
(THE COMPONENT AND SYSTEM RELIABILITY INFORMATION IS FOR THE FIRST ITERATION ONLY)

COMPONENT RELIABILITIES FOR MODULE A

R    = 0.755468           R    = 0.942294
 1     TYPE = BETA         2     TYPE = GAMMA
       P OR TIME =   46.00       P OR TIME =    50.10
       FAILURES =    4.00        FAILURES =     4.00

MODULE A  RELIABILITY = 0.986120              UNRELIABILITY = 0.013880


MODULE B
NUMBER OF COMPONENTS ................   2
NUMBER OF MINIMAL PATHS .............   2


THE   2 MINIMAL PATHS FOR MODULE B   FOLLOW:
<1>
<2>
                                        SUBSYSTEM RELIABILITY EQUATION ..... MODULE B    (  3 TERMS)

  R    = R  + R  - R R
   B      1    2    1 2


SPARCS: PROBABILITY COMPUTATION ROUTINE
(THE COMPONENT AND SYSTEM RELIABILITY INFORMATION IS FOR THE FIRST ITERATION ONLY)

COMPONENT RELIABILITIES FOR MODULE B

R    = 0.913139           R    = 0.981560
 1     TYPE = GAMMA        2     TYPE = BETA
       P OR TIME =   50.20       P OR TIME =   147.00
       FAILURES =    4.00        FAILURES =     3.00

MODULE B  RELIABILITY = 0.998398              UNRELIABILITY = 0.001602

MODULE AND COMPONENT RELIABILITIES FOR THE SYSTEM

R    = 0.986120           R    = 0.998398
 A     TYPE = MODULE       B     TYPE = MODULE
       P OR TIME =    0.00       P OR TIME =     0.00
       FAILURES =    0.00        FAILURES =     0.00
```

Figure 5. Printout of System and Module Equation(s) plus Component Information

If no modules are used (Figure 5-A), the component analysis is followed by an estimate of the system reliability and unreliability. This estimate is provided as a result of the first simulation run only. If the system has modules (Figure 5-B), each module is handled like a minisystem. Module information and minimal states are printed first. A subsystem (module) reliability or unreliability equation is developed and the information about each component of the module is presented. Finally, an estimate of the module reliability and unreliability is provided and stored for future use in the system equation. For systems with modules, the system component information is presented after the subsystem information along with the system reliability and unreliability interval estimates for the first simulation run.

The last part (Figure 6) presents statistical information about the empirical distribution of interval estimates provided by the Monte Carlo procedures. Initially, the mean, variance, and standard deviation is given for the resulting distribution. An estimated reliability or unreliability for the system is determined and printed using maximum likelihood estimates for the (un)reliability of each component. An analysis of the system MTBF is optionally provided. If this option is chosen, the system mission time and the estimated MTBF is printed. The estimated MTBF is a direct conversion of the estimated system reliability (unreliability). The interval estimates of the reliabilities (unreliabilities) are ordered and percentile points are provided as direct conversions from the system reliabilities.

Finally, an analysis of the frequency and cumulative frequency counts of cases is printed. This information divides the range of

```
THE MEAN RELIABILITY IS 0.988233      VARIANCE = 0.000047        STANDARD DEVIATION = 0.006862
THE ESTIMATED RELIABILITY FOR THE SYSTEM IS  0.588715

THE MISSION TIME IS     90.00  DAYS
THE ESTIMATED MTBF IS  7.60312923E+03


                          RELIABILITY               MTBF
 PERCENTILE                PERCENTILE             PERCENTILE
                             POINTS                 POINTS

   1 PERCENT                0.965649            2.57473670E+03  DAYS
 2.5 PERCENT                0.970772            3.03404661E+03  DAYS
   5 PERCENT                0.972242            3.19704546E+03  DAYS
  10 PERCENT                0.979010            4.41240246E+03  DAYS
  20 PERCENT                0.982509            5.10027128E+03  DAYS
  25 PERCENT                0.984355            5.70735406E+03  DAYS
  50 PERCENT                0.989835            8.80859645E+03  DAYS
  75 PERCENT                0.992806            1.24651109E+04  DAYS
  80 PERCENT                0.993751            1.43565459E+04  DAYS
  90 PERCENT                0.995222            1.87925672E+04  DAYS
  95 PERCENT                0.995591            2.03608142E+04  DAYS
97.5 PERCENT                0.995942            2.21329315E+04  DAYS
  99 PERCENT                0.996452            2.53233379E+04  DAYS


                     FREQUENCY AND CUMULATIVE FREQUENCY COUNTS OF CASES

 0.9620    0.9640    0.9660    0.9680    0.9700    0.9720    0.9740    0.9760    0.9780    0.9800
      0         0         1         0         0         2         2         0         1         4
      0         0         1         1         1         3         5         5         6        10

 0.9820    0.9840    0.9860    0.9860    0.9900    0.9920    0.9940    0.9960    0.9980    1.0000
      5         8         6        12        10        17        14        15         3         0
     15        23        29        41        51        68        82        97       100       100
```

Figure 6. Statistical Information, Empirical Distribution Interval Estimates and Histogram

the reliability (unreliability) interval estimates into 20 equal parts. The first line under each subdivision is a frequency count and the second line a cumulative frequency count of interval estimates. Hopefully, this information makes it easier to visualize the resulting empirical distribution of interval estimates.

Limitations

The core size increases as the size of the system under analysis increases. SPARCS can ideally handle a system of up to 128 components or subsystems. Each subsystem can contain up to 128 components. Conse quently, we can ideally handle a system of up to (128 x 128) 16,384 total components. Also, each probability equation can contain up to 2,000 terms. Then, the total number of terms for such a system would be as high as 258,000 (128 x 2,000) terms. However, it is estimated that such a system would require something over 600K to execute.

TABLE XIV

SYSTEM LIMITS

No. of modules per system	128
No. of elements per module.	128
No. of minimal states per system or module.	256
No. of terms in probability equation of system or module.	2,000
No. of systems per run.	No limit
No. of simulation runs.	No limit

Input Information

Information to be input should follow in this general order:

(1) information used to allocate and release storage,

(2) information to identify each system,

(3) control information about the components, modules, and states to be used in the system,

(4) label information about the system elements if provided by the user,

(5) component information, and

(6) minimal states for the system.

If the system has modules, then

(7) control information about the components and states to be used with appropriate module,

(8) labels for the module elements if provided by the user, and

(9) the module minimal states.

Numbers 7, 8, and 9 are repeated for each module of the system. The use of modules is left to the discretion of the user. If no modules are used, numbers 7, 8, and 9 are disregarded.

Card Input

For input information, the basic PL/1 input rules are followed. They are as follows:

(1) all nonnumeric data must be left justified within the field, and

(2) numeric data may be punched anywhere within the field.

The minimal states may appear anywhere on a card as long as each state is separated by a comma, semicolon or a blank space.

<u>Card 1--Allocation and MTBF Information</u>

Column	Parameter	Description
FF*	MAXCOM	Maximum number of components in the system <u>or</u> in any module
FF*	SIMNUM	Number of simulation runs desired
FF*	MXTERM	Maximum number of minimal states in the system <u>or</u> in any module
FF*	STIME	System mission time if MTBF analysis is desired; 0 otherwise
FF*	SUNITS	Units of system mission time placed in single quotationmarks; use 'NO' if MTBF option not used ('NO' in single quotation marks)

<u>Card 2--System Identification Card</u>

Column	Parameter	Description
1-80	SYSID	Alphameric system identification

<u>Card 3--Control Information</u>

Column	Parameter	Description
1-3	NMOD	Number of modules in system
5-7	NCOM	Number of elements in system
9-12	NPATH	Number of system minimal states
14	ATYPE	Type of analysis (R if reliability analysis, U if unreliability analysis)

---

*FF = free form. Information items in free form may be placed anywhere on a card as long as they are in the specified order and are separated by a comma, semicolon or blank space.

Card 3--Continued

Column	Parameter	Description
16-18	ALAB	Source of labels (punch YES if supplied by user, leave blank otherwise)
20-22	APUN	Punches output of probability equation desired, (punch YES if desired, leave blank otherwise)

Card 4--System Label Cards (Optional)

Column	Parameter	Description
7-9	COMPS1	Alphameric label for system elements
10-80	DESCR	Alphameric description of system element or module

Card 5--System Component Information Card

Column	Parameter	Description
FF*	TYPE	Type of component (punch 1 if Bernoulli component, 2 if time-to-failure component, 0 if system module)
FF*	PORT	Number of success if Bernoulli component or total testing units if time-to-failure components
FF*	FAILS	Number of failures observed in component tests

Card 6--System Minimal State Card

Column	Parameter	Description
FF*	MINPTH	Minimal states for the system. May be either in hexidecimal ('4A') or binary ('1010' B) representation

Cards 7, 8, 9, and 10 are used if the system under analysis has been divided into modules.

*FF = free form. Information items in free form may be placed anywhere on a card as long as they are in the specified order and are separated by a comma, semicolon or blank space.

Card 7--Module Control Card

Column	Parameter	Description
1-3	NCOM	Number of module components
5-7	NPATH	Number of module minimal states
10	KODE	Form of input of module minimal states (H if hexadecimal, leave blank otherwise)

Card 8--Module Label Cards (Optional)

Column	Parameter	Description
7-9	COMS1	Alphameric label for module component
10-80	MDESCR	Alphameric description of module components

Card 9--Module Component Information Card

Column	Parameter	Description
FF*	TYPE	Type of component (punch 1 if Bernoulli, 2 if Poisson process)
FF*	PORT	Number of observed successes if Bernoulli or total testing units if Poisson process
FF*	FAILS	Number of failures observed in component tests

*FF = free form. Information items in free form may be placed anywhere on a card as long as they are in the specified order and are separated by a comma, semicolon or blank space.

CHAPTER IX

SUMMARY, CONCLUSIONS AND EXTENSIONS

Summary

SPARCS (Simulation Program for Assessing the Reliability of Complex Systems) is a program designed to provide reliability confidence assessment for complex systems. The model uses Monte Carlo techniques to furnish confidence bounds and limits for such systems.

Work done by J. L. Burris is used as a basis around which the model is developed. Poincaire's Theorem (inclusion-exclusion) and a modular concept used in MAPS (by Burris) is found intact in SPARCS. Poincaire's Theorem develops an equation for the system as a function of the component reliabilities and their placement in the system. The modular concept allows large systems to be subdivided into smaller modules for easier analysis.

The components for system analysis are limited to two basic types: Bernoulli components and Poisson process components. For Bernoulli components, the beta distribution is used in conjunction with Bayesian analysis to provide reliability estimates for these types of components. Historical test information on accumulated successes and failures are used as sufficient statistics for the beta parameters. For Poisson process component, Bayesian analysis is used with the negative-log gamma prior distribution to provide component reliabilities. Accumulated failures and accumulated total test time in units are used as sufficient statistics for parameters of the negative-log gamma prior. For both cases, a uniform prior is generated in the absence of data.

The Monte Carlo techniques used in SPARCS generate an empirical distribution of reliability point estimates for the system under analysis. These values are ordered and analyzed statistically to provide information about the system.

Conclusion

The beta and gamma routines are tested in Chapter V. The International Mathematical and Statistical Library (IMSL) routines used to generate component reliabilities were tested and found to provide very good values. The IMSL routines have very small inherent error.

The concepts used in SPARCS are analyzed to determine whether they are intact. The duality concept is shown to be functioning properly. The uniform (ignorance) prior is shown to be implemented correctly in both IMSL routines.

Results of small system runs by SPARCS are compared to similar system analysis in the literature. The results supplied by SPARCS are consistent with the results provided in the literature. Therefore, the Monte Carlo procedures and the theory utilized in SPARCS are proven correct.

Finally, a large network model is run by SPARCS. This validated the ability of the model to handle large systems. The test network is a large complex network with randomly placed component types. The model accomodated the network very well and provided a network analysis with a reasonable amount of core usage.

Extension

Two extensions of this work are suggested by results supplied by the model. First, the model generates point estimates for the reliability of each system under analysis. These estimates are ordered to provide an empirical distribution of the system (un)reliability. If these distributions were analyzed, some information might be obtained concerning the type or family of distributions that are being generated. Perhaps, they are all the same type or all may belong to the same family. Depending upon the results of such an analysis, a reasonably simple mathematical algorithm for providing reliability information for systems of any logical configuration may be found.

Secondly, SPARCS has been found to be very efficient. Small sample sizes tend to produce very good results. A sample size determination procedure could be developed to provide a sample size significantly smaller than those obtained with conventional methods. This would result in a saving of computer time and money.

A SELECTED BIBLIOGRAPHY

(1) Aitchison, J. and D. Sculthrope. "Some Problems of Statistical Prediction." Biometrika, Vol. 52 (December, 1965), pp. 469-483.

(2) Barlow, R. E. and F. Proschan. "Multipcomponent Structures." Mathematical Theory of Reliability. New York: John Wiley and Sons, 1965, Chapter 7.

(3) Bazovsky, Igor. Reliability: Theory and Practice. Englewood Cliffs, N. J.: Prentice Hall, Inc. 1961.

(4) Berkbigler, Kathryn and James K. Byers. "System Reliability: Exact Bayesian Intervals Compared with Fiducial Intervals." I.E.E.E. Transactions on Reliability, Vol. R-24, No. 3 (August, 1975), pp. 199-200.

(5) Berry, Donald A. "Optimal Sampling Schemes for Estimating System Reliability by Testing Components - I: Fixed Sample Sizes." Journal of the American Statistical Association, Vol. 69 (1974), pp. 485-491.

(6) Birnbaum, Z. W., J. D. Esary and S. C. Saunders. "Multicomponent Systems and Structures and their Reliability." Technometrics, Vol. 3 (1961), pp. 55-77.

(7) Bosinoff, Irving and Jerome Kion. "Development of New Prediction Techniques." Proceedings 8th National Symposium on Reliability and Quality Control (1962), pp. 382-287.

(8) Buehler, Robert J. "Confidence Intervals for the Product of Two Binomial Parameters." Journal of the American Statistical Association, Vol. 52 (1957), pp. 482-493.

(9) Burdick, Donald S. and Thomas N. Naylor. "Design of Computer Simulation Experiments for Industrial Systems." Communications of the Academy of Computing Machinery, Vol. 8 (1966), pp. 329-333.

(10) Burnett, Thomas L. and Beverly A. Wales. "System Reliability Confidence Limits." Proceedings 7th National Symposium on Reliability and Quality Control (1962), pp. 382-387.

(11) Burris, James L. "Model for the Analysis of Probabilities of Systems (MAPS)." (Unpub. MBA report, Oklahoma State University, 1972.)

(12) Chandler, J. P. "RANF - Uniformly Distributed Pseudo-Random Numbers." Stillwater: Unpublished document, Oklahoma State University, University Computer Center, OSU-22, August, 1970.

(13) Clark, Charles T. and Lawrence Schade. Statistical Analysis for Administrative Decisions. Cincinnati: South-Western Publishing Co., 1974.

(14) Computer Subroutine Libraries in Mathematics and Statistics (Library Contents). Houston, Texas: International Mathematical and Statistical Libraries, Inc., GNB Building, IMSL LIB-004P, April, 1974.

(15) Conner, W. S. and W. T. Wells. "System Reliability Confidence Limits." Proceedings Seventh National Symposium on Reliability and Quality Control (1962), pp. 14-16.

(16) Conway, R. W. "Some Tactical Problems in Digital Simulation." Management Science, Vol. X (1963), p. 49.

(17) Cooley, Belva J. "Determination of Sample Size for Specified Width Confidence Intervals in Digital Simulation Experiments." (Unpub. Ph.D. dissertation, Oklahoma State University, 1976.)

(18) Cox, D. R. "Feiller's Theorem and a Generalization." Biometrika, Vol. 54 (1967), pp. 567-572.

(19) Easterling, R. G. "A Review of the Bayesian Controversy in Reliability and Statistics." I.E.E.E. Transactions on Reliability, Vol. R-12 (August, 1971), pp. 186-194.

(20) Easterling, Robert G. "Approximate Confidence Limits for System Reliability." Journal of the American Statistical Association, Vol. 67, No. 337 (March, 1972), pp. 220-222.

(21) El Mawaziny, A. H. "Chi-Square Distribution Theory With Application to Reliability Problems." (Unpub. dissertation, Iowa State University, 1965.)

(22) El Mawaziny, A. H. and R. J. Buehler. "Confidence Limits for the Reliability of Series Systems." Journal of the American Statistical Association, Vol. 62 (1967), pp. 1452-1459.

(23) Epstein, Benjamin and Milton Sobel. "Life Testing." Journal of the American Statistical Association, Vol. 58 (1963), pp. 486-502.

(24) Esary, James D. and Frank Proschan. "The Reliability of Coherent Systems." Redundancy Techniques for Computing Systems. R. C. Wilcos and W. C. Mann, editors. Washington, D.C.: Spartan, 1962, pp. 47-61.

(25) Esary, James D. and Frank Proschan. "Coherent Structures of Non-Identical Components." Technometrics, Vol. 5 (1963), pp. 191-209.

(26) Feller, W. J. An Introduction to Probability Theory. 3rd Ed. Vol 1. New York: John Wiley and Sons, 1968.

(27) Fertig, Kenneth W. "Bayesian Prior Distributions for Systems with Exponential Failure Time Data." The Annals of Mathematical Statistics, Vol. 43, No. 5 (1972), pp. 1441-1448.

(28) Fisher, R. A. Statistical Methods and Scientific Inference. New York: Hofner, 1956.

(29) Fishman, George S. "Estimating Sample Size in Computing Simulation Experiments." Management Science, Vol. VIII (1971), p. 28.

(30) Geisser, S. "The Statistical Use of Predictive Densities." Foundations of Statistics. Eds. Godambe and Scott. Toronto: Holt, Reinhart, and Winston, 1970, pp. 469-483.

(31) Graybill, Franklin A. "Determining Sample Size for a Specified Width Confidence Interval." Annals of Mathematical Statistics, Vol. XXIX (1958), p. 287.

(32) Grubbs, Frank E. "Approximate Fiducial Bounds for the Reliability of a Series System for Which Each Component has an Exponential Time-to-Failure Distribution." Technometrics, Vol. 13, No. 4 (November, 1971), pp. 865-871.

(33) Harris, Bernard. "Hypothesis Testing and Confidence Intervals for Products and Quotients of Poisson Parameters with Applications to Reliability." Journal of the American Statistical Association, Vol. 69 (1974), pp. 609-613.

(34) Harter, H. L. and A. H. Moore. "Point and Interval Estimators, Based on M Order Statistics for the Scale Parameter of a Weibull Distribution with Known Shape Parameter." Technometrics, Vol. R-14 (1965), pp. 405-422.

(35) Hoeffding, W. "A Class of Statistics with Asymptotically Normal Distribution." Annals of Mathematical Statistics, Vol. 19 (1948), pp. 293-325.

(36) Kalbfleisch, J. D. and D. A. Sprott. "Applications of Likelihood and Fiducial Probability to Sampling Finite Populations." Recent Advances in Survey Sampling. Chapel Hill, N.C.: University of North Carolina Press, 1969., pp. 227-338.

(37) Knuth, Donald E. The Art of Computer Programming. Vol. 3. Reading, Mass.: Addison-Wesley, 1969.

(38) Kraemer, H. C. "One Sided Confidence Intervals for the Quality Indices of a Complex Item." Technometrics, Vol. 5 (1963), pp. 400-403.

(39) Lawless, J. F. "On Prediction of Survival Times for Individual Systems." I.E.E.E. Transactions on Reliability, Vol. R-23, No. 4 (October, 1972), pp. 235-241.

(40) Lee, Keun K. "Error Analysis of Selected Inverse Beta and Gamma Distribution Function Routines." (Unpub. M.S. thesis, Oklahoma State University, 1976.)

(41) Lehmann, E. L. Testing Statistical Hypothesis. New York: John Wiley and Sons, Inc., 1959.

(42) Lehmann, E. L. and Henry Scheffe. "Completeness, Similar Regions, and Unbiased Estimation, Part II." Sankhya, Vol. 15 (1955), pp. 219-236.

(43) Leiberman , G. J. and Sheldon M. Ross. "Confidence Intervals for Independent Exponential Series Systems." Journal of the American Statistical Association, Vol. 66, No. 336 (1971), pp. 837-840.

(44) Lentner, M. M. and R. J. Buehler. "Some Inferences About Gamma Parameters With an Application to a Reliability Problem." Journal of the American Statistical Association, Vol. 58 (1963), pp. 670-677.

(45) Levy, Louis L. and Albert H. Moore. "A Monte Carlo Technique for Obtaining System Reliability Confidence Limits from Component Test Data." I.E.E.E. Transactions on Reliability, Vol. R-16, No. 2 (September, 1967), pp. 69-72.

(46) Lipow, M. and J. Riley. Tables of Upper Confidence Limits on Failure Probability of 1, 2, and 3 Component Serial Systems. Redondo Beach, California: Space Technology Laboratories, Vol. 1 and 2, AD-609-100, AD-636-718, 1960.

(47) Lloyd, D. K. and M. Lipow. Reliability, Management, Methods and Mathematics. Englewood Cliffs, N.J.: Prentice-Hall, 1962.

(48) Locks, Mitchell O. "Exact Minimal State System Reliability Analysis." Proceedings of Computer Science and Statistics: 5th Annual Symposium on the Interface (November, 1971), available from Western Periodicals Co., 13000 Ramer Street, North Hollywood, Calif. 91605.

(49) Locks, Mitchell O. "The Maximum Error in System Reliability Calculations by Using a Subset of the Minimal States." I.E.E.E. Transactions on Reliability, Vol. R-24, No. 4 (November, 1971), pp. 231-234.

(50) Locks, Mitchell O. Reliability, Maintainability and Reliability Assessment. Rochelle Park, New Jersey: Spartan Books, 1973.

(51) Maclaren, M. D. and G. Marsaglia. "Uniform Random Number Generators." Journal of the Association of Computing Machinery, Vol. 12 (1965), p. 83.

(52) Madansky, Albert. "Approximate Confidence Limits for the Reliability of Series and Parallel Systems." Technometrics, Vol. 7 (1965), pp. 495-503.

(53) Mann, Nancy R. "Computer-Aided Selection of Prior Distributions for Generating Monte Carlo Confidence Bounds on System Reliability." Naval Research Logistics Quarterly, Vol. 17, No. 1, NAVSO P-1278 (March, 1970), pp. 41-54.

(54) Mann, Nancy R. "Approximately Optimum Randomized and Nonrandomized Confidence Bounds on Series and Parallel System Reliability for Systems with Binomial Subsystem Data." I.E.E.E. Transactions on Reliability, Vol. R-23, No. 5 (December, 1974), pp. 295-304.

(55) Mann, Nancy R. "Simplified Expressions for Obtaining Approximate Optimal System Reliability Confidence Bounds." Journal of the American Statistical Association, Vol. 69 (1974), pp. 492-495.

(56) Mann, Nancy R. and Frank E. Grubbs. "Approximate Optimum Confidence Bounds on Series System Reliability for Exponential Time to Fail Data." Biometrika, Vol. 59 (April, 1972), pp. 191-204.

(57) Mann, Nancy R. and Frank E. Grubbs. "Approximately Optimum Confidence Bounds for System Reliability Based on Component Test Data." Technometrics, Vol. 16 (1974), pp. 335-347.

(58) Mann, Nancy R., R. E. Schafer and N. D. Singpurwalla. Methods for Statistical Analysis of Reliability and Life Data. New York: John Wiley and Sons, 1974.

(59) Marsaglia, G. "Random Numbers Fall Mainly in the Planes." Proceedings of the National Academy of Science, Vol. 61 (Sept., 1968), p. 25.

(60) Marsaglia, G. and T. A. Bray. "One Line Random Number Generators and Their Use if Combinations." Communications of the Association of Computing Machinery, Vol. 11, No. 11 (1968), pp. 757-759.

(61) Mine, Hasashi. "Reliability of Physical System." IRE Transactions, PGIT, Vol. IT-5, Special Supplement (May, 1959), pp. 138-150.

(62) Monte Carlo Bayesian System Reliability and MTBF Confidence Assessment. Wright Patterson Air Force Base, Ohio: Technical Report AFFDL-TR-75-144, Air Force Flight Dynamics Laboratory, Air Force Wright Aeronautical Laboratories, Air Force Systems Command, 1976.

(63) Mood, A. M. and F. A. Graybill. Introduction to the Theory of Statistics. New York: McGraw-Hill, 1963, pp. 225-231.

(64) Moore, E. F. and C. E. Shannon. "Reliable Circuits Using Less Reliable Relays." Journal of the Franklin Institute, Vol. 262 (1956), pp. 191-208.

(65) Moran, P. A. P. The Theory of Storage. New York: John Wiley and Sons, 1959.

(66) Morse, P. M. Queues, Inventories and Maintenance. New York: John Wiley and Sons, 1958.

(67) Myhre, J. M. and S. C. Saunders. "Comparison of Two Methods of Obtaining Approximate Confidence Intervals for System Reliability." Technometrics, Vol. 10 (1968), pp. 37-49.

(68) Naylor, Thomas H., Joseph L. Balintfy, Donald S. Burdick, and Kong Chu. Computer Simulation Techniques. New York: John Wiley and Sons, 1966.

(69) Nelson, W. B. "Two Sample Prediction." General Electric Company TIS Report 68-C-404. Schenectady, N.Y.: Corporate Research and Development, 1968.

(70) Neyman, J. "On the Problem of Confidence Intervals." Annals of Mathematical Statistics, Vol. 6 (1935), pp. 111-116.

(71) Patnaik, P. B. "The Non-Central X^2 and F Distributions and Their Applications." Biometrika, Vol. 36 (1949), pp. 202-232.

(72) Pearson, Karl. Tables of the Incomplete Beta-Function. London: Cambridge University Press, 1956.

(73) Pearson, Karl. Tables of the Incomplete Gamma-Function. London: Cambridge University Press, 1956.

(74) Poore, Jr., Jessie H. "Computational Procedures for Generating and Testing Random Numbers." Operations and Systems Analysis. Boston: Allyn and Bacon, Inc., 1974, pp. 101-126.

(75) Raiffa, H. and R. Schlaifer. Applied Statistical Decision Theory. Boston, Mass: Graduate School of Business Administration, Harvard University, 1961.

(76) Rosenblatt, Joan Raup. "Confidence Limits for the Reliability of Complex Systems." _Statistical Theory of Reliability._ Ed. Marvin Zelen.

(77) Sarkar, T. D. "An Exact Lower Confidence Bound for the Reliability of a Series System when Each Component Has an Exponential Time to Failure Distribution." Report No. 117. Palo Alto, Cal.: Department of Operations Research and Department of Statistics, Stanford University, March, 1969.

(78) Schick, G. J. and R. J. Prior. "Reliability and Confidence of Serially Connected Systems." _Proceedings of the Third Space Congress_, Cocoa Beach, Fla. (1966), pp. 352-360.

(79) Schmidt, J. W. and R. E. Taylor. _Simulations and Analysis of Industrial Systems._ Homewood, Ill.: Richard D. Irwin, Inc., 1970.

(80) Schmidtt, Samuel A. _Measuring Uncertainty: An Elementary Introduction to Bayesian Statistics._ Reading, Mass.: Addison-Wesley, 1969.

(81) Shell, Donald L. "A High Speed Sorting Procedure." _Communications of the Associaton of Computing Machinery_, Vol. 2, No. 7 (July, 1959), pp. 30-32.

(82) Shell, Donald L. "Optimizing the Polyphase Sort." _Communications of the Association of Computing Machinery_, Vol.14, No. 11 (November, 1971), pp. 713-719.

(83) Springer, M. D. and W. E. Thompson. "Bayesian Confidence Limits for the Product of n Binomial Parameters." _Biometrika_, Vol. 53 (1966), pp. 611-613.

(84) Springer, M. D. and W. E. Thompson. "Bayesian Confidence Limits for Reliability of Cascade Exponential Systems." _I.E.E.E. Transactions on Reliability_, Vol. R-16, No. 2 (September, 1967), pp. 86-89.

(85) Springer, M. D. and W. E. Thompson. "Bayesian Confidence Limits for Reliability of Redundant Systems when Tests are Terminated at the First Failure." _Technometrics_, Vol. 10, No. 1 (February, 1968), pp. 29-36.

(86) Spurr, William A. and Charles P. Bonini. _Statistical Analysis for Business Decisions._ Homewood, Ill.: Richard D. Irwin, 1967.

(87) Steck, G. P. "Upper Confidence Limits for the Failure Probability of Complex Networks." SC-4133(TR). Albuquerque, N.M.: Sandia Corporation, 1957.

(88) System for Computing Operational Probability Equation (SCOPE); Version II. North American Rockwell Corporation Space Division, Program MFS-24035. Athens, Ga.: University of Georgia, 1960.

(89) Thompson, W. E. and Eugene Y. Chang. "Bayes Confidence Limits for the Reliability of Redundant Systems." Technometrics, Vol. 17, No. 1 (February, 1975), pp. 89-93.

(90) Van Gelder, A. "Some New Results in Pseudo-Random Number Generations." Journal of the Association of Computing Machinery, Vol. 14 (October, 1967), pp. 785-792.

(91) von Neuman, J. "Probabilistic Logistics and the Synthesis of Reliable Organisms from Unreliable Components." Automa Studies, Annals of Math Studies, No. 34 (1956), pp. 43-98.

(92) Wilks, S. S. "The Large-sample Distribution of the Likelihood Ratio for Testing Composite Hypothesis." Annals of Mathematical Statistics, Vol. 9 (1938), pp. 60-62.

(93) Wilson, E. B. and M. M. Hilferty. "The Distribution of Chi-Square." Proceedings National Academy of Science, Vol. 17 (1931), pp. 684-688.

(94) Winterbottom, Alan. "Lower Confidence Limits for Series System Reliability from Binomial Subsystem Data." Journal of the American Statistical Association, Vol. 69, No. 347 (September, 1974), pp. 782-788.

(95) Wolf, James E. "Bayesian Reliability Assessment from Test Data." Proceedings 1976 Annual Reliability and Maintainability Symposium (January, 1976), pp. 75-84.

APPENDIX A

JCL FOR CALLING IMSL ROUTINES FROM THE IMSL LIBRARY

```
//JWC4      JOB (XXXXXXXXXXXXXXXXXX,9,,,,1),'JOHN COOLEY',CLASS=L,
// TIME=(0009,00)
***ROUTE PRINT HOLD
// EXEC PL1LFCLG,
// REGION.GO=275K
XXPL1L       EXEC PGM=IEMAA,PARM='LOAD,NODECK',REGION=127K              PL1L 00000600
XXSYSPRINT DD    SYSOUT=A                                                  PL1L 00000700
//PL1L.SYSLIN DD SPACE=(400,(200,200))
X/SYSLIN    DD   DSNAME=&LOADSET,DISP=(MOD,PASS),UNIT=SYSDA,              PL1L 00000800
XX               SPACE=(400,(50,20)),DCB=BLKSIZE=800                       PL1L 00000900
XXSYSUT3    DD    UNIT=SYSDA,SPACE=(80,(250,250)),DSN=&SYSUT3,            PL1L 00001000
XX               SEP=SYSPRINT,DCB=BLKSIZE=80                               PL1L 00001100
XXSYSUT1    DD    UNIT=SYSDA,SPACE=(1024,(60,60),,CONTIG),                PL1L 00001200
XX               DCB=BLKSIZE=1024                                          PL1L 00001300
//PL1L.SYSIN DD *
IEF236I ALLOC. FOR JWC4      PL1L
IEF237I 347    ALLOCATED TO SYSPRINT
IEF237I 131    ALLOCATED TO SYSLIN
IEF237I 131    ALLOCATED TO SYSUT3
IEF237I 131    ALLOCATED TO SYSUT1
IEF237I 302    ALLOCATED TO SYSIN
IEF142I - STEP WAS EXECUTED - COND CODE 0004
IEF285I    SYS75188.T075403.RV000.JWC4.LOADSET                  PASSED
IEF285I    VOL SER NOS= DISK05.
IEF285I    SYS75188.T075403.RV000.JWC4.SYSLI3                   DELETED
IEF285I    VOL SER NOS= DISK05.
IEF285I    SYS75188.T075403.RV000.JWC4.R0004773                 DELETED
IEF285I    VOL SER NOS= DISK05.
IEF373I STEP /PL1L      / START 75188.2221
IEF374I STEP /PL1L      / STOP  75188.2224 CPU    1MIN 08.00SEC MAIN 128K LCS    OK

XXLKED       EXEC PGM=IEWLF440,PARM='LIST,MAP',COND=(08,LT,PL1L),       LKED 00001400
XX               REGION=63K                                                LKED 00001500
//LKED.SYSLIB DD DSNAME=&PL1LIB,DISP=(SHR,PASS)
X/SYSLIB    DD    DSN=&PL1LIB,DISP=(SHR,PASS)                              LKED 00001600
*//             DD DSNAME=&PLISSP,DISP=(SHR,PASS)
X/          DD    DSN=&PLISSP,DISP=(SHR,PASS)                              LKED 00001700
*//             DD DSNAME=SYS1.IMSL.OBJECT,DISP=(SHR,PASS)
*//             DD DSNAME=&FORTLIB,DISP=(SHR,PASS)
XXSYSLMOD   DD    DSNAME=&GOSET(GO),DISP=(NEW,PASS),                      LKED 00001800
XX               DCB=BLKSIZE=1024,UNIT=SYSDA,SPACE=(CYL,(4,,1))           LKED 00001900
XXSYSUT1    DD    UNIT=(SYSDA,SEP=(SYSLMOD,SYSLIB)),SPACE=(1024,          LKED 00002000
XX               (200,20)),DSN=&SYSUT1,DCB=BLKSIZE=1024                    LKED 00002100
XXSYSPRINT DD     SYSOUT=A                                                 LKED 00002200
XXSYSLIN    DD    DSNAME=&LOADSET,DISP=(OLD,DELETE),DCB=(RECFM=FB,        LKED 00002300
XX               BLKSIZE=800)                                              LKED 00002400
XX          DD    DDNAME=SYSIN                                             LKED 00002500
IEF236I ALLOC. FOR JWC4      LKED
IEF237I 230    ALLOCATED TO SYSLIB
IEF237I 234    ALLOCATED TO
IEF237I 134    ALLOCATED TO
IEF237I 230    ALLOCATED TO
IEF237I 131    ALLOCATED TO SYSLMOD
IEF237I 232    ALLOCATED TO SYSUT1
IEF237I 347    ALLOCATED TO SYSPRINT
IEF237I 131    ALLOCATED TO SYSLIN
IEF142I - STEP WAS EXECUTED - COND CODE 0004
IEF285I    SYS1.PL1LIB                                          PASSED
IEF285I    VOL SER NOS= SYSRS2.
IEF285I    SYS1.PLISSP                                          PASSED
IEF285I    VOL SER NOS= DISK06.
IEF285I    SYS1.IMSL.OBJECT                                     PASSED
IEF285I    VOL SER NOS= DISK02.
IEF285I    SYS1.FORTLIB                                         PASSED



IEF285I    SYS75188.T075403.RV000.JWC4.GOSET                    PASSED
IEF285I    VOL SER NOS= DISK05.
IEF285I    SYS75188.T075403.RV000.JWC4.SYSUT1                   DELETED
IEF285I    VOL SER NOS= DISK06.
IEF285I    SYS75188.T075403.RV000.JWC4.LOADSET                  DELETED
IEF285I    VOL SER NOS= DISK05.
IEF373I STEP /LKED      / START 75188.2224
IEF374I STEP /LKED      / STOP  75188.2229 CPU    0MIN 14.02SEC MAIN  64K LCS    OK

XXGO         EXEC PGM=*.LKED.SYSLMOD,COND=((9,LT,LKED),(08,LT,          GO      00002600
XX               PL1L)),REGION=63K                                         GO      00002700
XXSYSPRINT DD    SYSOUT=A                                                  GO      00002800
//GO.PUNCH DD SYSOUT=B,DCB=BLKSIZE=80
//GO.FT06F001 DD SYSOUT=A
//GO.SYSPRINT DD SYSOUT=A
//GO.SYSIN DD *
//
IEF236I ALLOC. FOR JWC4      GO
IEF237I 131    ALLOCATED TO PGM=*.DD
IEF237I 346    ALLOCATED TO SYSPRINT
IEF237I 370    ALLOCATED TO PUNCH
IEF237I 347    ALLOCATED TO FT06F001
IEF237I 349    ALLOCATED TO SYSPRINT
IEF237I 303    ALLOCATED TO SYSIN
IEF142I - STEP WAS EXECUTED - COND CODE 1000
IEF285I    SYS75188.T075403.RV000.JWC4.GOSET                    PASSED
IEF285I    VOL SER NOS= DISK05.
IEF373I STEP /GO        / START 75188.2229
IEF374I STEP /GO        / STOP  75188.2237 CPU    0MIN 51.84SEC MAIN 264K LCS    OK


IEF285I    SYS1.PL1LIB                                          KEPT
IEF285I    VOL SER NOS= SYSRS2.
IEF285I    SYS1.PLISSP                                          KEPT
IEF285I    VOL SER NOS= DISK06.
IEF285I    SYS1.IMSL.OBJECT                                     KEPT
IEF285I    VOL SER NOS= DISK02.
IEF285I    SYS1.FORTLIB                                         KEPT
IEF285I    VOL SER NOS= SYSRS2.
IEF285I    SYS75188.T075403.RV000.JWC4.GOSET                    DELETED
IEF285I    VOL SER NOS= DISK05.
IEF375I  JOB /JWC4      / START 75188.2221
IEF376I  JOB /JWC4      / STOP  75188.2237 CPU    2MIN 13.86SEC
```

APPENDIX B

JCL FOR USING THE REQUIRED IMSL ROUTINES AS LOAD MODULES

```
  //JWC4      JOB (XXXXXXXXXXXXXXXXX,9,,,,1),'JOHA COOLEY',CLASS=L,
  // TIME=(0009,00)
  ***FORMS 9C01
* // EXEC FORTGC
  XX          PROC KP=EBCDIC                                            00000010
  XXFCRT  EXEC PGM=IEYFGRT,REGICN=99K,PARM=(&KP)                   FORT 00000020
  IEF(53I SUBSTITUTION JCL - FGM=IEYFCRT,REGICN=99K,PARM=(EBCDIC)
  XXSYSPRINT DD    SYSOUT=A                                          FORT 00000030
  XXSYSLIN    DD    DSNAME=&LOADSET,DISP=(MOD,PASS),UNIT=SYSDA,       FORT 00000040
  XX               SPACE=(800,(20C,100)),DCB=(BLKSIZE=800,LRECL=80,   FORT 00000050
  XX               RECFM=FB)                                          FORT 00000060
* //FCRT.SYSIN  DD *
  IEF236I ALLOC. FOR JWC4      FCRT
  IEF237I 347   ALLOCATED TO SYSPRINT
  IEF237I 131   ALLOCATED TO SYSLIN
  IEF237I 301   ALLOCATED TO SYSIN
  IEF142I - STEP WAS EXECUTED - COND CODE 0000
  IEF285I    SYS75207.T080104.RV000.JWC4.LOADSET           PASSED
  IEF285I    VOL SER NOS= DISK05.
  IEF373I STEP /FORT     / START 75207.1728
  IEF374I STEP /FURT     / STOP  75207.1728 CPU   0MIN 05.72SEC MAIN  88K LCS   OK
                                                                 THE PREVIOUS JOBSTEP REQUESTED  12K BYTES OF JNUSED CORE.
                                                                                          STEP COMPLETION CODE - 0000

* // EXEC ASPFC
  XXASM       EXEC PGM=IEUASM,REGION=63K,PARM='NODECK,LCAD'          ASM  00000010
  XXSYSLIB    DD    DSNAME=&MACLIB,DISP=(SHR,PASS)                   ASM  00000020
  XXSYSLT1    DD    UNIT=SYSDA,SPACE=(1700,(400,50)),DSN=&SYSUT1     ASM  00000030
  XXSYSLT2    DD    UNIT=SYSDA,SPACE=(1700,(4C0,50)),DSN=&SYSUT2     ASM  00000040
  XXSYSUT3    DD    UNIT=(SYSCA,SEP=(SYSUT2,SYSUT1,SYSLIB)),         ASM  00000050
  XX               SPACE=(1700,(400,50)),DSN=&SYSUT3                 ASM  00000060
  XXSYSPRINT DD    SYSOUT=A                                          ASM  00000070
  XXSYSGO     DD    DSNAME=&LOADSET,DISP=(MOD,PASS),UNIT=SYSDA,      ASM  00000080
  XX               SPACE=(400,(200,50)),DCB=(BLKSIZE=800,LRECL=80,   ASM  00000090
  XX               RECFM=FB)                                         ASM  00000100
* //ASM.SYSIN DD *
  IEF236I ALLOC. FOR JWC4      ASM
  IEF237I 130   ALLOCATED TO SYSLIB
  IEF237I 131   ALLOCATED TO SYSUT1
  IEF237I 131   ALLOCATED TO SYSUT2
  IEF237I 232   ALLOCATED TO SYSUT3
  IEF237I 347   ALLOCATED TO SYSPRINT
  IEF237I 131   ALLOCATED TO SYSGO
  IEF237I 3C4   ALLOCATED TO SYSIN
  IEF142I - STEP WAS EXECUTED - COND CODE 0000
  IEF285I    SYS1.MACLIB                                   PASSED
  IEF285I    VOL SER NOS= SYSRS1.
  IEF285I    SYS75207.T080104.RV000.JWC4.SYSUT1            DELETED
  IEF285I    VOL SER NOS= DISK05.
  IEF285I    SYS75207.T080104.RV000.JWC4.SYSUT2            DELETED
  IEF285I    VOL SER NOS= DISK05.
  IEF285I    SYS75207.T080104.RV000.JWC4.SYSUT3            DELETED
  IEF285I    VOL SER NOS= DISK56.
  IEF285I    SYS75207.T080104.RV000.JWC4.LOADSET           PASSED
  IEF285I    VOL SER NOS= DISK05.
  IEF373I STEP /ASM      / START 75207.1728
  IEF374I STEP /ASM      / STOP  75207.1729 CPU   0MIN 02.14SEC MAIN  64K LCS   OK
                                                                                          STEP COMPLETION CODE - 0000
  // EXEC PL1LFCLG,
  // REGICN.GC=275K
  XXPL1L      EXEC PGM=IEMAA,PARM='LOAD,NODECK',REGION=127K          PL1L 00000600
  XXSYSPRINT DD    SYSOUT=A                                          PL1L 000C0700
  //PL1L.SYSLIN DD SPACE=(40C,(2C0,200))
  X/SYSLIN    DD    DSNAME=&LOADSET,DISP=(MOD,PASS),UNIT=SYSDA,      PL1L 00000800
  XX               SPACE=(400,(50,20)),DCB=BLKSIZE=800               PL1L 00000900



 XXSYSUT3    DD    UNIT=SYSDA,SPACE=(80,(250,250)),DSN=&SYSUT3,       PL1L 00001000
 XX               SEP=SYSPRINT,DCB=BLKSIZE=80                         PL1L 00001100
 XXSYSUT1    DD    UNIT=SYSDA,SPACE=(1024,(60,60),,CONTIG),           PL1L 00001200
 XX               DCB=BLKSIZE=1024                                    PL1L 00001300
 //PL1L.SYSIN DD *
 IEF236I ALLOC. FOR JWC4      PL1L
 IEF237I 347   ALLOCATED TO SYSPRINT
 IEF237I 131   ALLOCATED TO SYSLIN
 IEF237I 131   ALLOCATED TO SYSUT3
 IEF237I 131   ALLOCATED TU SYSUT1
 IEF237I 305   ALLOCATED TC SYSIN
 IEF142I - STEP WAS EXECUTED - COND CODE 0004
 IEF285I    SYS75207.T080104.RV000.JWC4.LOADSET           PASSED
 IEF285I    VCL SER NOS= DISK05.
 IEF285I    SYS75207.T080104.RVC00.JWC4.SYSUT3            DELETED
 IEF285I    VOL SER NOS= DISKC5.
 IEF285I    SYS75207.T080104.RV000.JWC4.R0001306          DELETED
 IEF285I    VOL SER NOS= DISK05.
 IEF373I STEP /PL1L     / START 75207.1729
 IEF374I STEP /PL1L     / STOP  75207.1734 CPU   1MIN 08.09SEC MAIN 128K LCS   OK
                                                                                          STEP COMPLETION CODE - 0004
 XXLKED      EXEC PGM=IEWLF44C,PARM='LIST,MAP',COND=(08,LT,PL1L),   LKED 00001400
 XX               REGION=63K                                         LKED 00001500
*//LKED.SYSLIB  DD  DSN=SYS1.PL1LIB,DISP=SHR
 X/SYSLIB    DD    DSN=&PL1LIB,DISP=(SHR,PASS)                       LKED 00001600
*//               DD  DSN=SYS1.PL1SSP,DISP=SHR
 X/          DD    DSN=&PL1SSP,DISP=(SHR,PASS)                       LKED 00001700
 //               DD  DSN=SYS1.FORTLIB,DISP=SHR
 XXSYSLMOD   CD    DSNAME=&GOSET(GO),DISP=(NEW,PASS),                LKED 00001800
 XX               DCB=BLKSIZE=1024,UNIT=SYSDA,SPACE=(CYL,(4,,1))     LKED 00001900
 XXSYSUT1    DD    UNIT=(SYSDA,SEP=(SYSLMOD,SYSLIB)),SPACE=(1024,    LKED 00002000
 XX               (20C,20)),DSN=&SYSUT1,DCB=BLKSIZE=1024             LKED 00002100
 XXSYSPRINT CD    SYSOUT=A                                           LKED 00002200
 XXSYSLIN    CD    DSNAME=&LOADSET,DISP=(OLD,DELETE),DCB=(RECFM=FB,  LKED 00002300
 XX               BLKSIZE=800)                                       LKED 00002400
 XX          CC    DDNAME=SYSIN                                      LKED 00002500
 //LKED.SYSIN DD *
 IEF236I ALLOC. FOR JWC4      LKED
 IEF237I 230   ALLOCATED TO SYSLIB
 IEF237I 234   ALLOCATED TO
 IEF237I 23C   ALLOCATED TO
 IEF237I 131   ALLOCATED TO SYSLMOD
 IEF237I 232   ALLOCATED TO SYSUT1
 IEF237I 346   ALLOCATED TO SYSPRINT
 IEF237I 131   ALLOCATED TO SYSLIN
 IEF237I 306   ALLOCATED TO
 IEF142I - STEP WAS EXECUTED - COND CODE 0004
```

```
IEF285I   SYS1.PL1LIB                                 KEPT
IEF285I   VCL SER NOS= SYSRS2.
IEF285I   SYS1.PL1SSP                                 KEPT
IEF285I   VOL SER NOS= DISK06.
IEF285I   SYS1.FORTLIB                                KEPT
IEF285I   VCL SER NOS= SYSRS2.
IEF285I   SYS75207.T080104.RV000.JWC4.GOSET           PASSED
IEF285I   VOL SER NOS= DISK05.
IEF285I   SYS75207.T080104.RV000.JWC4.SYSUT1          DELETED
IEF285I   VOL SER NOS= DISK56.
IEF285I   SYS75207.T080104.RV000.JWC4.LOADSET         DELETED
IEF285I   VOL SER NOS= DISKC5.
IEF373I STEP /LKED    / START 75207.1734
IEF374I STEP /LKED    / STOP  75207.1735 CPU   0MIN 12.85SEC MAIN  64K LCS   OK
                                                                                                   STEP COMPLETION CODE - 0004

XXGC        EXEC PGM=*.LKED.SYSLMOD,COND=((9,LT,LKED),(08,LT,          GO   00002600
XX               PL1L)),REGION=63K                                     GO   00002700
XXSYSPRINT DD   SYSOUT=A                                               GO   00002800
//GC.FUNCH DD SYSOUT=B,DCB=BLKSIZE=80



//GC.FT06F001 DD SYSOUT=A
//GC.SYSPRINT DD SYSOUT=A
//GC.SYSIN CD *
//
IEF236I ALLOC. FOR JWC4     GC
IEF237I 131   ALLOCATED TO PGM=*.DD
IEF237I 346   ALLOCATED TO SYSPRINT
IEF237I 37C   ALLOCATED TO PUNCH
IEF237I 347   ALLOCATED TO FTG6F001
IEF237I 348   ALLOCATED TO SYSPRINT
IEF237I 307   ALLOCATED TO SYSIN
IEF142I - STEP WAS EXECUTED - CCND CODE 2000
IEF285I   SYS75207.T080104.RV0C0.JWC4.GOSET           PASSED
IEF285I   VOL SER NOS= DISK05.
IEF373I STEP /GO      / START 75207.1735
IEF374I STEP /GO      / STOP  75207.1735 CPU   GMIN 01.82SEC MAIN 268K LCS   OK
                                                                      THE PREVIOUS JOBSTEP REQUESTED   8K BYTES OF JNUSED CORE.
                                                                                                   STEP COMPLETION CODE - 2000
IEF285I   SYS1.MACLIB                                 KEPT
IEF285I   VOL SER NOS= SYSRS1.
IEF285I   SYS75207.T080104.RV000.JWC4.GOSET           DELETED
IEF285I   VOL SER NOS= DISK05.
IEF375I  JCB /JWC4    / START 75207.1728
IEF376I  JOB /JWC4    / STOP  75207.1735 CPU   1MIN 30.62SEC
```

APPENDIX C

SOURCE PROGRAM LISTING OF SPARCS

```
(SUBRG,STRG,SIZE):
   UNITED:PROCEDURE OPTIONS (MAIN);
                  /*  DECLARE STATEMENTS  */
        DCL A(20) REAL FIXEC DEC(6,0) CONTROLLED;
        DCL ALAB CHAR(3);
        DCL APUN CHAR(3);
        CCL ATYPE CHAR(1);
        DCL ATYPE1 CHAR(13) VARYING EXT;
        DCL AV REAL FIXED DEC(3) EXT;
        CCL AVAL REAL FLOAT DEC(14) EXT;
        DCL AVREL REAL FLCAT DEC(14) EXT;
        DCL BTYPE REAL FIXED DEC(1) EXT;
        DCL BVAL REAL FIXED DEC(6,0) EXT;
        DCL CA(20) REAL FIXED DEC(6,0) CONTROLLED;
        DCL CD REAL FIXED CEC(3) EXT;
        DCL CE REAL FIXED DEC(3) EXT;
        CCL (CP, CCP) REAL FIXED DEC(6);
        DCL (CS, CR, CT, CPC) REAL FIXED DEC(5);
        DCL CVAL REAL FLCAT DEC(14) EXT;
        DCL (COMPS(128), SLAB(128)) CHAR(3) VAR EXT STATIC;
        CCL DCCM(MAXEL) REAL FIXED DEC(3) CONTROLLED EXT;
        CCL DERMS(MAXEL,MAXEL) BIT (128) CONTRCLLED VAR EXT;
        DCL DMOD REAL FIXED DEC (3) EXT;
        CCL COEF(MAXEL,MAXEL) REAL FIXED DEC(4) CONTROLLED EXT;
        DCL DTERM(MAXEL) REAL FIXED DEC(3) CONTROLLED EXT;
        DCL (DVAL, EVAL) REAL FIXED DEC(3,2);
        DCL FERMS (1500) BIT (128) VAR EXT;
        DCL FVAL REAL FLOAT DEC(6);
        DCL GAMMA REAL FLOAT DEC(14) EXT;
        DCL KV REAL FIXED DEC ( 3) EXT;
        DCL (KS,KT,KU,KX,KY) REAL FIXED DEC (3) EXT;
        DCL KCDE CHAR(1) VAR EXT;
        DCL LABELS REAL FIXED DEC (1) EXT;
        DCL LA(20) REAL FIXED DEC(6,5) CONTROLLED;
        CCL MEAN REAL FLOAT CEC(14);
        DCL MTBF REAL FLGAT DEC(10);
        DCL MXTERM REAL FIXED DEC(3);
        CCL NARG FIXED BINARY(31,0) EXT;
        DCL NCCM REAL FIXEC DEC (3) EXT;
        DCL NMOD REAL FIXED DEC (3) EXT;
        DCL NPATH REAL FIXED DEC (3) EXT;
        DCL PREL(MAXCCM) REAL FLOAT DEC(14) CONTROLLED EXT;
        DCL PUNOUT REAL FIXED DEC (1) EXT;
        DCL REL(MAXCOM) REAL FLOAT DEC(14) CONTRCLLED EXT;
        DCL RELSTO(SIMNUM) REAL FLOAT DEC(14) CONTROLLED EXT;
        DCL RELVAL(SIMNUM) REAL FLOAT DEC(14) CONTROLLED EXT;
        DCL RELSCRT FLCAT CEC(14);
        DCL SEED REAL FLOAT DEC(16) EXT;
        DCL (SCOM, STERM,SMOD ) REAL FIXED DEC (3) EXT;
        DCL SIMCOM(MAXEL) REAL FIXED DEC(3) CCNTROLLED EXT;
        DCL SIMNUM REAL FIXED DEC(6) EXT;
        DCL SN REAL FIXED DEC(6,0) EXT;
        DCL SORTVAL FIXED CEC(6,0);
        DCL SREL(NMOD) REAL FLOAT DEC(14) CCNTROLLED EXT;
        DCL STDEV REAL FLOAT DEC(14);
        DCL STIME REAL FLOAT DEC(8);
        DCL SUNITS CHAR (2C) VAR;
        DCL SYSID CHAR (80) VAR EXT;
        DCL SYSIN FILE STREAM INPUT;
        DCL SYSPRINT FILE STREAM OUTPUT PRINT;
        DCL TERMS (1500) BIT (128) VAR EXT;
        DCL (TYPE(MAXEL,MAXCCM), PORT(MAXEL,MAXCOM),
            FAILS(MAXEL,MAXCOM)) REAL FLOAT DEC(14) CONTROLLED EXT;
        DCL VAR REAL FLOAT CEC(14);
        DCL Z(11) FIXED DEC(6,0) INITIAL (4,13,40,121,364,1093,3280,
             9841,29524,88573,265720);
        CPEN FILE(SYSIN), FILE(SYSPRINT);
        ON ENDFILE(SYSIN) STOP;
        NARG = 7;
        GAMMA = RANF(NARG);
        NARG = 0;
        SEED = .75;
    L1:
        GET FILE (SYSIN) LIST (MAXCOM,SIMNUM,MXTERM,STIME,SUNITS);
        GET FILE (SYSIN) EDIT (SYSID) (COL(1), A(80));
        GET FILE (SYSIN) LIST (NMOD,NCOM,NPATH);
        GET FILE (SYSIN) ECIT (ATYPE,ALAB,APUN,KODE)
             (COLUMN(14),A(1),X(1),A(3),X(1),A(3),X(1),
                A(1));
        SN = 1;
        AV = 1;
        MXTERM = 2**MXTERM - 1;
        IF NMOD = NCOM THEN MAXEL = NCOM + 1;
        ELSE MAXEL = NCCM;
        IF NMOD = 0 THEN ALLOCATE TYPE(1,MAXCCM), PORT(1,MAXCOM),
           FAILS(1,MAXCOM);
        ELSE ALLOCATE TYPE(MAXEL,MAXCOM), PORT(MAXEL,MAXCOM),
           FAILS(MAXEL,MAXCOM);
        ALLOCATE REL(MAXCOM);
        ALLOCATE SIMCOM(MAXEL);
        ALLOCATE PREL(MAXCOM);
        ALLOCATE RELSTO(SIMNUM);
        ALLOCATE SREL(NMOD);
        ALLOCATE DTERM(MAXEL), DOEF(MAXEL,MXTERM), DERMS(MAXEL,MXTERM),
             DCOM(MAXEL);
        SIMCOM(1) = NCOM;
    SL1:
        DMOD=0;
        SMOD=0;  SCOM=0;  STERM=0;
        SCOM = NCOM;
        DMOD = NMOD;
        SMOD= NMOD;
        CE = 1;
        IF SN > 1 THEN DO;
        NCOM = SIMCOM(1);
             CALL SIMOUT;
             GO TO SL2;
             END;
        ELSE;
        IF APUN='YES' THEN PUNOUT=1;
             ELSE PUNOUT=0;
        IF ALAB='YES' THEN LABELS=1;
             ELSE LABELS=0;
        IF ATYPE = 'U' THEN BTYPE = 1;
             ELSE BTYPE = 0;
        CALL OUT1;
        CALL FNPUT2;
        CALL ECGEN;
        CALL OUT3;
    SL2:
        IF DMOD=0 THEN GO TO R1;
                  /*  PROCESS MODULES  */
        KV=1;
        KX=0;
     O1:IF KX=SMOD THEN GOTO L3;
        IF SN > 1 THEN DO;
             NCOM = SIMCOM(KX+2);
             CE = KX + 2;
             CALL SIMULATE;
             CALL COMPUTE;
             GO TO SL3;
             END;
        ELSE
         CE = KX + 2;
        CALL FNPUT1;
        SIMCOM(KX+2) = NCOM;
        CALL ECGEN;
        CALL OUT3;
        CALL SOUT1;
        CALL COMPUTE;
        CALL SOUT2;
    SL3:
        KX=KX+1;
        GOTO O1;
     L3:DO KS=1 TO SMOD;
        REL(KS)=SREL(KS);
        COMPS(KS)=SLAB(KS);
        END L3;
        IF SCOM-SMOD=0 THEN GOTO W1;
        DO KY=1 TO SCOM-SMOD;
         REL(KY+SMOD)=PREL(KY+SMOD);
         COMPS(KY+SMOD)=SLAB(KY+SMOD);
        END;
     W1:DMOD=0;
        CE = 1;
        CALL COMPUTE;
        IF SN = 1 THEN CALL SOUT2;
        ELSE;
        IF SN < SIMNUM THEN DO;
             SN = SN + 1;
             GO TO SL1;
             END;
        ELSE DO;
        IF AV = 1 THEN DO;
             AV = 2;
             GO TO SL1;
             END;
        ELSE;
        FREE TYPE,PORT,FAILS,REL,SIMCOM,PREL,SREL;
        FREE DTERM, DCEF, DERMS, DCOM;
        GO TO CALCUL;
        END;
     R1:KV=1;
        IF SN > 1 THEN DO;
             CALL COMPUTE;
             SN = SN + 1;
             IF SN <= SIMNUM THEN GO TO SL1;
             ELSE DO;
        IF AV = 1 THEN DO;
             AV = 2;
             GO TO SL1;
             END;
        ELSE;
             FREE TYPE, PORT, FAILS, REL, SIMCOM, PREL, SREL;
             FREE DTERM, DOEF, DERMS, DCOM;
             GO TO CALCUL;
             END;
             END;
        ELSE
        CALL SOUT1;
        CALL COMPUTE;
        CALL SOUT2;
        IF SN < SIMNUM THEN DO;
             SN = SN + 1;
             GO TO SL1;
             END;
        ELSE DO;
        FREE TYPE,PORT,FAILS,REL,SIMCOM,PREL,SREL;
        FREE DTERM, DOEF, DERMS, DCOM;
```

```
                END;
      CALCUL:
                SUMVAL = 0;
                SUMSC = 0;
      CAL:      DC IX = 1 TO SIMNUM;
                SUMVAL = SUMVAL + RELSTO(IX);
                SUMSQ = SUMSQ + RELSTO(IX)**2;
                END CAL;

      CQAA:     DC CS = 3 TO 11;
                IF Z(CS) >= SIMNUM THEN DO;
                CR = CS-2;
                GC TO CQA;
                END;
                ELSE IF CS = 11 THEN DO;
                CR = 10;
                GO TO CQA;
                END;
                ELSE;
                END CQAA;
      CQA:      DO CS = CR TO 1 BY -1;
                SORTVAL = SIMNUM / Z(CS);
      CQB:      DO CP = 1 TO SIMNUM BY 1 WHILE (SORTVAL+CP <= SIMNUM);
                IF RELSTC(CP) > RELSTC(SORTVAL+CP) THEN DO;
                     RELSORT = RELSTO(CP);
                     RELSTO(CP) = RELSTC(SORTVAL+CP);
                     RELSTO(SORTVAL+CP) = RELSCRT;
                     END;
                ELSE;
                END CQB;
                END CQA;
      CQC:      DO CS = 1 TO SIMNUM-1;
                IF RELSTC(CS) > RELSTO(CS+1) THEN DO;
                CR = CS;
                RELSORT = RELSTO(CS);
                RELSTO(CS) = RELSTO(CS+1);
                RELSTO(CS+1) = RELSCRT;
                IF CS = 1 THEN GO TO CQE;
                ELSE;
      CQD:      CO CP = CR TO 2 BY -1;
                IF RELSTC(CP) < RELSTO(CP-1) THEN DO;
                     RELSORT = RELSTO(CP-1);
                     RELSTO(CP-1) = RELSTO(CP);
                     RELSTO(CP) = RELSORT;
                     END;
                ELSE GO TO CQE;
                END CQD;
      CQE:      END CQC;

                PUT FILE (SYSPRINT) EDIT ('ORDERED VALUES OF THE SYSTEM ',
                'RELIABILITIES AND UNRELIABILITIES')  (SKIP(3),A,A);
      CN:       DO CP = 1 TO SIMNUM;
                PUT FILE (SYSPRINT) EDIT (RELSTO(CP), 1-RELSTO(CP))
                     (COL(23), F(8,6), COL(62), F(8,6));
                END CN;
                MEAN = SUMVAL / SIMNUM;
                VAR = SUMSQ/SIMNUM - (SUMVAL/SIMNUM)**2;
                STDEV = SQRT(VAR);
                PUT PAGE;
                PUT FILE (SYSPRINT) EDIT ('THE MEAN ', ATYPE1, ' IS ', MEAN,
                  'VARIANCE = ', VAR, 'STANDARD DEVIATION = ', STDEV)
                (SKIP(2),X(5),A,A,A,F(8,6),X(5),A,F(8,6),X(5),A,F(8,6));
                PUT FILE (SYSPRINT) EDIT ('THE ESTIMATED ',ATYPE1,
                 ' FOR THE SYSTEM IS  ', AVREL)
                (COL(6),A,A,A,F(8,6));
                IF STIME ¬= 0 THEN DC;
                PUT FILE (SYSPRINT) EDIT ('THE MISSION TIME IS ',STIME,SUNITS)
                (SKIP(2),COL(6),A,F(8,2),X(2),A);
                MTBF = STIME / -(LOG(MEAN));
                PUT FILE (SYSPRINT) EDIT ('THE ESTIMATED MTBF IS ',MTBF)
                (COL(6),A,E(15,8));
                END;
                ELSE;
                IF STIME ¬= 0 THEN DC;
                PUT FILE (SYSPRINT) EDIT (ATYPE1, 'MTBF')
                  (SKIP(3),COL(25),A,COL(49),A);
                PUT FILE (SYSPRINT) EDIT ('PERCENTILE', 'PERCENTILE',
                'PERCENTILE')
                  (COL(4),A,COL(25),A,COL(46),A);
                PUT FILE (SYSPRINT) EDIT ('POINTS', 'POINTS')
                  (COL(27),A,CCL(48),A);
                END;
                ELSE DO;
                PUT FILE (SYSPRINT) EDIT ('PERCENTILE', 'PERCENTILE')
                  (SKIP(3),COL(4),A,CCL(26),A);
                PUT FILE (SYSPRINT) EDIT ('POINTS')

                  (COL(27),A);
                END;
                AVAL = SIMNUM * .01;
                IF AVAL < 1 THEN GO TO SL4;
                ELSE CALL CONF;
                IF STIME ¬= 0 THEN
                PUT FILE (SYSPRINT) EDIT ('    1 PERCENT', AVAL,MTBF,SUNITS)
                  (SKIP(2),COL(2),A,COL(26),F(8,6),COL(43),E(15,8),X(2),A);
                ELSE
                PUT FILE (SYSPRINT) EDIT ('    1 PERCENT',AVAL)
                  (SKIP(2), COL(2), A, COL(26), F(8,6));
      SL4:      AVAL = SIMNUM * .025;
                IF AVAL < 1 THEN GC TO SL5;
                ELSE CALL CONF;
```

```
             IF STIME ¬= 0 THEN
             PUT FILE (SYSPRINT) EDIT (' 2.5 PERCENT', AVAL,MTBF,SUNITS)
               (COL(2),A,COL(26),F(8,6),COL(43),E(15,8),X(2),A);
             ELSE
             PUT FILE (SYSPRINT) EDIT (' 2.5 PERCENT', AVAL)
               (COL(2), A, COL(26), F(8,6));
    SL5:     AVAL = SIMNUM * .05;
             IF AVAL < 1 THEN GO TO SL6;
             ELSE CALL CONF;
             IF STIME ¬= 0 THEN
             PUT FILE (SYSPRINT) EDIT ('   5 PERCENT', AVAL,MTBF,SUNITS)
               (COL(2),A,COL(26),F(8,6),COL(43),E(15,8),X(2),A);
             ELSE
             PUT FILE (SYSPRINT) EDIT ('   5 PERCENT', AVAL)
                (COL(2), A, COL(26), F(8,6));
    SL6:     AVAL = SIMNUM * .10;
             CALL CONF;
             IF STIME ¬= 0 THEN
             PUT FILE (SYSPRINT) EDIT ('  10 PERCENT', AVAL,MTBF,SUNITS)
               (COL(2),A,COL(26),F(8,6),COL(43),E(15,8),X(2),A);
             ELSE
             PUT FILE (SYSPRINT) EDIT ('  10 PERCENT', AVAL)
                (COL(2), A, COL(26), F(8,6));
             AVAL = SIMNUM * .20;
             CALL CONF;
             IF STIME ¬= 0 THEN
             PUT FILE (SYSPRINT) EDIT ('  20 PERCENT', AVAL,MTBF,SUNITS)
               (COL(2),A,COL(26),F(8,6),COL(43),E(15,8),X(2),A);
             ELSE
             PUT FILE (SYSPRINT) EDIT ('  20 PERCENT', AVAL)
                  (COL(2), A, COL(26), F(8,6));
              AVAL = SIMNUM * .25;
             CALL CONF;
             IF STIME ¬= 0 THEN
             PUT FILE (SYSPRINT) EDIT ('  25 PERCENT', AVAL,MTBF,SUNITS)
               (COL(2),A,COL(26),F(8,6),COL(43),E(15,8),X(2),A);
             ELSE
             PUT FILE (SYSPRINT) EDIT ('  25 PERCENT', AVAL)
                  (COL(2), A, COL(26), F(8,6));
             AVAL = SIMNUM * .50;
             CALL CONF;
             IF STIME ¬= 0 THEN
             PUT FILE (SYSPRINT) EDIT ('  50 PERCENT', AVAL,MTBF,SUNITS)
               (COL(2),A,COL(26),F(8,6),COL(43),E(15,8),X(2),A);
             ELSE
             PUT FILE (SYSPRINT) EDIT ('  50 PERCENT', AVAL)
                  (COL(2), A, COL(26), F(8,6));
             AVAL = SIMNUM * .75;
             CALL CONF;
             IF STIME ¬= 0 THEN
             PUT FILE (SYSPRINT) EDIT ('  75 PERCENT', AVAL,MTBF,SUNITS)
               (COL(2),A,COL(26),F(8,6),COL(43),E(15,8),X(2),A);
             ELSE
             PUT FILE (SYSPRINT) EDIT ('  75 PERCENT', AVAL)
                 (COL(2), A, COL(26), F(8,6));
             AVAL = SIMNUM * .80;
             CALL CONF;
             IF STIME ¬= 0 THEN
             PUT FILE (SYSPRINT) EDIT ('  80 PERCENT', AVAL,MTBF,SUNITS)
               (COL(2),A,COL(26),F(8,6),COL(43),E(15,8),X(2),A);
             ELSE
             PUT FILE (SYSPRINT) EDIT ('  80 PERCENT', AVAL)
                (COL(2), A, COL(26), F(8,6));
             AVAL = SIMNUM * .90;
             CALL CONF;
             IF STIME ¬= 0 THEN
             PUT FILE (SYSPRINT) EDIT ('  90 PERCENT', AVAL,MTBF,SUNITS)
               (COL(2),A,COL(26),F(8,6),COL(43),E(15,8),X(2),A);
             ELSE
             PUT FILE (SYSPRINT) EDIT ('  90 PERCENT', AVAL)
                (COL(2), A, COL(26), F(8,6));
             AVAL = SIMNUM * .95;
             CALL CONF;
             IF STIME ¬= 0 THEN
             PUT FILE (SYSPRINT) EDIT ('  95 PERCENT', AVAL,MTBF,SUNITS)
               (COL(2),A,COL(26),F(8,6),COL(43),E(15,8),X(2),A);
             ELSE
             PUT FILE (SYSPRINT) EDIT ('  95 PERCENT', AVAL)
                  (COL(2), A, COL(26), F(8,6));
             AVAL = SIMNUM * .975;
             CALL CONF;
             IF STIME ¬= 0 THEN
             PUT FILE (SYSPRINT) EDIT ('97.5 PERCENT', AVAL,MTBF,SUNITS)
               (COL(2),A,COL(26),F(8,6),COL(43),E(15,8),X(2),A);
             ELSE
             PUT FILE (SYSPRINT) EDIT ('97.5 PERCENT', AVAL)
               (COL(2), A, COL(26), F(8,6));
             AVAL = SIMNUM * .99;
             CALL CONF;
             IF STIME ¬= 0 THEN
             PUT FILE (SYSPRINT) EDIT ('  99 PERCENT', AVAL,MTBF,SUNITS)
               (COL(2),A,COL(26),F(8,6),COL(43),E(15,8),X(2),A);
             ELSE
             PUT FILE (SYSPRINT) EDIT ('  99 PERCENT', AVAL)
               (COL(2), A, COL(26), F(8,6));
             ALLOCATE LA, CA, A;
             DO CS = 1 TO 20;
             A(CS) = 0;
             END;
             DVAL = RELSTO(1);
```

```
                 EVAL = RELSTO(SIMNUM) + .01;
                 FVAL = (EVAL - DVAL) / 20;
                 DO CR = 1 TO 20;
                 LA(CR) = DVAL + CR * FVAL;
                 END;
        CO:      DC CCP = 1 TO SIMNUM;
        CO1:     DO CPC = 1 TO 20;
                 IF RELSTO(CCP) <= LA(CPC) THEN DO;
                      A(CPC) = A(CPC) + 1;
                      GO TO CQ;
                      END;
                 ELSE;
                 END CO1;
        CQ:      END CC;
                 CA(1) = A(1);
                 CO CS = 2 TO 20;
                 CA(CS) = CA(CS-1) + A(CS);
                 END;

                 PUT FILE (SYSPRINT) EDIT ('FREQUENCY AND CUMULATIVE FREQUENCY',
                 ' COUNTS OF CASES') (SKIP(3),COL(25),A,A);
                 PUT FILE (SYSPRINT) EDIT ((LA(I) DO I = 1 TO 10)) (SKIP(2),
                      CCL(5), 1C(F(6,4), X(3)));
                 PUT FILE (SYSPRINT) EDIT ((A(I) DO I = 1 TO 10)) (COL(5),
                      10(F(6,0), X(3)));
                 PUT FILE (SYSPPINT) EDIT ((CA(I) DO I = 1 TO 10)) (COL(5),
                      10(F(6,0), X(3)));
                 PUT FILE (SYSPRINT) EDIT ((LA(I) DO I = 11 TO 20)) (SKIP(3),
                      COL(5), 10(F(6,4), X(3)));
                 PUT FILE (SYSPRINT) EDIT ((A(I) DO I = 11 TO 20)) (COL(5),
                      10(F(6,0), X(3)));
                 PUT FILE (SYSPRINT) EDIT ((CA(I) DO I = 11 TO 20)) (COL(5),
                      10(F(6,0), X(3)));
                 FREE LA, CA, A;
                 FREE RELSTO;
                 GO TO L1;
        CONF:    PROCEDURE;
                 BVAL = AVAL;
                 IF BVAL ¬= AVAL THEN DO;
                      AVAL = AVAL - BVAL;
                      CVAL = RELSTC(BVAL+1) - RELSTO(BVAL);
                      AVAL = CVAL * AVAL;
                      AVAL = RELSTO(BVAL) + AVAL;
                      END;
                 ELSE AVAL = RELSTC(BVAL);
                 IF STIME ¬= 0 THEN DO;
                      IF BTYPE = 1 THEN MTBF = STIME / -(LOG(1-AVAL));
                      ELSE MTBF = STIME / -(LOG(AVAL));
                 END;
                 ELSE;
                 RETURN;
                 END CONF;
          FNPUT:PROCEDURE;
                           /*  DECLARE STATEMENTS  */
                 DCL CE REAL FIXED DEC(3) EXT;
                 CCL I REAL FIXED DEC (3);
                 DCL KODE CHAR (1) VAR EXT;
                 DCL MINPTH (256) BIT (128) VAR EXT;
                 DCL NCCM REAL FIXEC DEC (3) EXT;
                 DCL NPATH REAL FIXED DEC (3) EXT;
                           /*  ENTRY POINT FOR MODULES  */
          FNPUT1:ENTRY;
                 GET FILE (SYSIN) LIST (NCOM,NPATH);
                 GET FILE (SYSIN) EDIT (KODE)  (COL(10),A(1));
                 CALL OUT2;
                           /*  ENTRY POINT FOR THE SYSTEM  */
          FNPUT2:ENTRY;
                           /*  CHECK KODE TO DETERMINE IF MINIMAL
                               STATES ARE TO BE INPUT IN BINARY
                               OR HEXADECIMAL NOTATION  */
                 IF KODE='H' THEN CALL HEXIN;
                 ELSE GET FILE (SYSIN) LIST ((MINPTH(I) DO I=1 TO NPATH));
                 RETURN;
                 END FNPUT;
         HEXIN:  PROCEDURE;
                           /*  CECLARE STATEMENTS  */
                 DCL CCDE CHAR(32) VAR;
                 DCL CODED(16) CHAR(1) INIT('A','B','C','D','E','F',
       '0','1','2','3','4','5','6','7','8','9');
                 CCL HEX(16) BIT(4) INIT('1010'B,'1011'B,'1100'B,
       '1101'B,'1110'B,'1111'B,'0000'B,'0001'B,'0010'B,'0011'B,
       '0100'B,'0101'B,'011C'B,'0111'B,'1000'B,'1001'B);
                 CCL TEMP1 CHAR(1);
                 DCL NPATH REAL FIXED DEC (3) EXT;
                 DCL NCOM REAL FIXED DEC (3) EXT;
                 DCL MPTH BIT(128) VARYING;
                 DCL MINPTH(256) BIT(128) VARYING EXTERNAL;
                 DCL (J1,J3,J4,JJ) REAL FIXED DEC (3);
                 DO JJ=1 TO 256;
                 MINPTH (JJ) = ''B;
                 END;
          HEXL:  DO J4=1 TO NPATH;
                 MPTH=''B;
                 GET FILE (SYSIN) LIST (CCDE);
       DECODE:   DC J1=1 TO (NCOM+3)/4;
                 TEMP1=SUBSTR(CODE,J1,1);
       SEARCH:   DO J3=1 TO 16;
                 IF TEMP1=CODEC(J3) THEN DO;
                 MPTH=MPTH||HEX(J3);
                 GO TO NEXT;     END;
          END SEARCH;
          PUT FILE (SYSPRINT) LIST('JOB TERMINATED - INVALID CHARACTER',
'ENCOUNTERED IN MINIMAL STATE ',J4);
          STOP;
    NEXT: END DECODE;
          MINPTH(J4)=MINPTH(J4)||SUBSTR(MPTH,1,NCOM);
          END HEXL;
          END HEXIN;
   EQGEN: PROCEDURE;
                  /*  DECLARE STATEMENTS  */
          DCL COEF (1500) REAL FIXED DEC (4) EXT;
          DCL (I1, NSUB,I2, NDUP, INC2, I3, I4, I5, KK1, I6, KDUP, I7)
             REAL FIXED DEC (3);
          DCL MINPTH(256) BIT(128) VARYING EXTERNAL;
          DCL (NCOM, NPATH,NTERM) REAL FIXED DEC (3) EXT;
          DCL TERMS (1500) BIT (128) VAR EXT;
                  /*  FIRST THREE TERMS  */
                  /*  INITIALIZE PROBABILITY EQUATION  */
          TERMS(1)=MINPTH(1); COEF(1)=1;
          IF NPATH=1 THEN DO;
              NTERM=1;  RETURN;  END;
          TERMS(2)=MINPTH(2); COEF(2)=1;
          TERMS(3)=MINPTH(1) | MINPTH(2);
          COEF(3)=-1;
          NTERM=3;
          IF NPATH=2 THEN GO TO ENDEQ;
          NSUB=4;
                  /*  REMAINING TERMS  */
   LOOP1: DO I1 = 3 TO NPATH;
          TERMS(NSUB)=MINPTH(I1);
          COEF(NSUB)=1;
          NSUB=NSUB+1;
   LOOP2: DO I2 = 1 TO NTERM;
          TERMS(NSUB)=MINPTH(I1) | TERMS(I2);
                  /*  DETERMINE COEFFICIENT  */
          COEF(NSUB)=-COEF(I2);
          NSUB=NSUB+1;
    ENDA: END LOOP2;
                  /*  ACCUMULATE DUPLICATE TERMS  */
          NDUP=0;
          INC2=NSUB-1;
          DO I3=NTERM+2 TO INC2;
          DO I4=3 TO I3-1-NDUP;
          IF TERMS(I4)¬=TERMS(I3-NDUP) THEN GO TO ENDI4;
          COEF(I4)=COEF(I4) + COEF(I3-NDUP);
          IF I3-NDUP=NSUB-1 THEN GO TO SUB;
              DO I5=I3-NDUP TO INC2-1-NDUP;
              TERMS(I5)=TERMS(I5+1);
              COEF(I5)=COEF(I5+1);
              END;
     SUB: NSUB=NSUB-1;
          NDUP=NDUP+1;
          GO TO ENDI3;
   ENDI4: END;
   ENDI3: END;
                 /* REMOVE TERMS WITH ZERO COEFFICIENTS */
          KDUP=0;
          KK1=NSUB-1;
          DO I6=3 TO KK1;
          IF COEF(I6-KDUP)¬=0 THEN GO TO ENDI6;
          IF I6-KDUP=NSUB-1 THEN GO TO KSUB;
              DO I7=I6-KDUP TO KK1-1-KDUP;
              TERMS(I7)=TERMS(I7+1);
              COEF(I7)=COEF(I7+1);
              END;
    KSUB: NSUB=NSUB-1;
          KDUP=KDUP+1;
   ENDI6: END;
          NTERM=NSUB-1;
          END LOOP1;
   ENDEQ: END EQGEN;
    OUTI: PROCEDURE;
          DCL BTYPE REAL FIXED DEC(1) EXT;
          DCL CE REAL FIXED DEC(3) EXT;
          DCL CHAR1 CHAR(120) VAR EXT;
          DCL CHAR2 CHAR(120) VAR EXT;
          DCL CHECK1 REAL FIXED DEC (3);
          DCL COEF (1500) REAL FIXED DEC (4) EXT;
          DCL COMPS1 CHAR (3);
          DCL COMPS2 CHAR(1);
          DCL COEFF CHAR(3) VARYING;
          DCL (COMPS(128), SLAB(128)) CHAR(3) VAR EXT STATIC;
          DCL CTYPE CHAR(13) VARYING STATIC;
          DCL DESCR  CHAR(74);
          DCL DCOM(MAXEL) REAL FIXED DEC(3) CONTROLLED EXT;
          DCL DERMS(MAXEL,MAXEL) BIT (128) CONTROLLED VAR EXT;
          DCL DCEF(MAXEL,MAXEL) REAL FIXED DEC(4) CONTROLLED EXT;
          DCL DTERM(MAXEL) REAL FIXED DEC(3) CONTROLLED EXT;
          DCL GAMMA REAL FLOAT DEC(14) EXT;
          DCL JEN REAL FIXED DEC (3) EXT;
          DCL JMOD REAL FIXED DEC (3) EXT;
          DCL KOMPS(128) CHAR(3) VARYING INITIAL('1','2','3',
'4','5','6','7','8','9','10','11','12','13','14','15','16',
'17','18','19','20','21','22','23','24','25','26','27',
'28','29','30','31','32','33','34','35','36','37','38',
'39','40','41','42','43','44','45','46','47','48','49',
'50','51','52','53','54','55','56','57','58','59','60',
'61','62','63','64','65','66','67','68','69','70','71',
'72','73','74','75','76','77','78','79','80','81','82',
'83','84','85','86','87','88','89','90','91','92','93',
'94','95','96','97','98','99','100','101','102','103',
'104','105','106','107','108','109','110','111','112',
'113','114','115','116','117','118','119','120','121',
'122','123','124','125','126','127','128') STATIC;
          DCL HEAD1 CHAR(60) VARYING;
          DCL      (MINPL,NMIN) REAL FIXED DEC (3);
          DCL K6 REAL FIXED DEC (3) EXT;
          DCL K7 REAL FIXED DEC (3) EXT;
          DCL KX REAL FIXED DEC(3) EXT;
          DCL LABELS REAL FIXED DEC (1) EXT;
          DCL MDESCR(128) CHAR(70);
          DCL MINP CHAR(500) VARYING;
          DCL MINPTH(256) BIT(128) VARYING EXTERNAL;
          DCL MODSY(128) CHAR(3) VARYING INIT('A','B','C',
'D','E','F','G','H','I','J','K','L','M','N','O','P','Q',
'R','S','T','U','V','W','X','Y','Z','A1','B1','C1','D1',
'E1','F1','G1','H1','I1','J1','K1','L1','M1','N1','O1',
'P1','Q1','R1','S1','T1','U1','V1','W1','X1','Y1','Z1',
'A2','B2','C2','D2','E2','F2','G2','H2','I2','J2','K2',
'L2','M2','N2','O2','P2','Q2','R2','S2','T2','U2','V2',
'W2','X2','Y2','Z2','A3','B3','C3','D3','E3','F3','G3',
'H3','I3','J3','K3','L3','M3','N3','O3','P3','Q3','R3',
'S3','T3','U3','V3','W3','X3','Y3','Z3','A4','B4','C4',
'D4','E4','F4','G4','H4','I4','J4','K4','L4','M4','N4',
'O4','P4','Q4','R4','S4','T4','U4','V4','W4','X4') STATIC;
          DCL MODSYM (128) CHAR (3) EXT;
          DCL LEN REAL FIXED DEC (3) EXT;
          DCL (NCOM,NMOD,NPATH,NTERM) REAL FIXED DEC (3) EXT;
          DCL NARG FIXED BINARY(31,0);
          DCL NPAGE REAL FIXED DEC (4) EXT;
          DCL (JI,K1,K2,K3,K4,K5,JS,LC,K10,K15,K16,LEN6,JL,K13,I,K11)
                REAL FIXED DEC (3);
          DCL PREL(MAXCOM) REAL FLOAT DEC(14) CONTROLLED EXT;
          DCL PUNCH FILE STREAM OUTPUT;
          DCL PUNOUT REAL FIXED DEC (1) EXT;
          DCL REL(MAXCOM) REAL FLOAT DEC(14) CONTROLLED EXT;
          DCL STATE CHAR(5) VARYING STATIC;
          DCL SYSID CHAR(80) VAR EXT;
          DCL TERMS (1500) BIT (128) VAR EXT;
          DCL (TYPE(MAXEL,MAXCOM),PORT(MAXEL,MAXCOM),
                FAILS(MAXEL,MAXCOM)) REAL FLOAT DEC(14) CONTROLLED EXT;
          DCL XCCMPS CHAR(3) VARYING;
          OPEN FILE(PUNCH);
                    /*  ENTRY POINT TO PRINT CONTROL DATA,
                        AND PROCESS LABELS FOR THE SYSTEM  */
    OUT1: ENTRY;
                    /*  SET LABELS DEPENDING ON THE TYPE OF
                        ANALYSIS PERFORMED  */
          IF BTYPE = 0 THEN CTYPE = 'RELIABILITY';
             ELSE CTYPE = 'UNRELIABILITY';
          IF BTYPE = 0 THEN STATE = 'PATHS';
             ELSE STATE = 'CUTS';
                    /*  PUNCH SYSTEM IDENTIFICATION, AND
                        NUMBER OF MODULES  */
          IF PUNOUT=0 THEN GO TO L20;
          PUT FILE(PUNCH) EDIT(SYSID) (A(80));
          PUT FILE (PUNCH) EDIT (NMOD, BTYPE)
             (F(3),X(1),F(1));
      L20:DO JI=1 TO 128;
          MODSYM (JI) = MODSY (JI);
          COMPS (JI) = KOMPS (JI);
          END;
          JEN=0;
          JMOD=0;
          NPAGE=1;
                    /*  PRINT SYSTEM CONTROL INFORMATION  */
          PUT FILE (SYSPRINT) EDIT ('SPARCS: EQUATION GENERATION ',
          'ROUTINE','PAGE',NPAGE)  (PAGE,A,A,COL(111),A,F(4));
          PUT FILE (SYSPRINT) EDIT ('SIMULATION PROGRAM FOR THE ',
          'ANALYSIS OF THE RELIABILITY OF COMPLEX SYSTEMS')  (SKIP(1),A,
          A);
          PUT FILE (SYSPRINT) EDIT ('COLLEGE OF BUSINESS ',
          'ADMINISTRATION, OKLAHOMA STATE UNIVERSITY')  (SKIP(1),A,A);
          PUT FILE (SYSPRINT) EDIT ('SYSTEM IDENTIFICATION ............',
          '... ',SYSID)  (SKIP(2),A,A,A);
          PUT FILE (SYSPRINT) EDIT ('NUMBER OF MODULES ...............',
          '.... ',NMOD)  (SKIP(2),A,A,F(3));
          PUT FILE (SYSPRINT) EDIT ('NUMBER OF NONMODULAR COMPONENTS ',
          '..... ',NCOM - NMOD)  (SKIP(1),A,A,F(3));
          PUT FILE (SYSPRINT) EDIT ('TOTAL NUMBER OF SYSTEM ELEMENTS ..',
          '... ',NCOM)  (SKIP(1),A,A,F(3));
          PUT FILE (SYSPRINT) EDIT ('NUMBER OF MINIMAL ',STATE,'.......',
          '...... ',NPATH)  (SKIP(1),A,A,COL(25),A,A,F(3));
          PUT FILE (SYSPRINT) EDIT ('PUNCHED OUTPUT OF EQUATION .......'
          ,'.... ')  (SKIP(1),A,A);
          IF PUNOUT=0 THEN PUT FILE (SYSPRINT) EDIT (' NO') (A(3));
             ELSE PUT FILE (SYSPRINT) EDIT ('YES') (A(3));
          PUT FILE (SYSPRINT) EDIT ('LABELS SUPPLIED BY USER ..........',
          '... ') (SKIP(1),A,A,SKIP(1));
          JEN=JEN+12;
          IF NMOD=0 THEN GO TO L12;
          DO K10=1 TO NMOD;
          COMPS(K10)=MODSY(K10);
          END;
          IF NMOD=NCOM THEN GO TO L12;
                    /*  HANDLE NONMODULAR ELEMENTS OF THE
                        SYSTEM  */
          DO K15=1 TO NCOM-NMOD;
          COMPS(K15+NMOD)=KCMPS(K15);     END;
     L12: IF LABELS = 0 THEN IF NMOD = 0 | NMOD = NCOM THEN
```

```
                BA: DO;
                 PUT FILE (SYSPRINT) EDIT (' NO')  (A(3));
                BB:  DO KJ = 1 TC NCOM;
            GET FILE (SYSIN) LIST (TYPE(1,KJ), PORT(1,KJ), FAILS(1,KJ));
            IF NMOD ¬= 0 THEN SLAB(KJ) = MODSY(KJ);
            ELSE;
                  END BB;
            CALL SIMULATE;
                 RETURN;
                  END BA;
            ELSE
            CA: DC;
                 PUT FILE (SYSPRINT) EDIT (' NO')  (A(3));
            CB: DO KL = 1 TO NCOM;
            GET FILE (SYSIN) LIST (TYPE(1,KL), PORT(1,KL), FAILS(1,KL));
                END CB;
            CALL SIMULATE;
            CC: DC KM = NMOD + 1 TO NCOM;
                PREL(KM) = REL(KM);
                END CC;
      CF:   DO KN = 1 TC NMOD;
            SLAB(KN) = MODSY(KN);
            END CF;
      CG:   DC KC = NMOD+1 TO NCOM;
            SLAB(KO) = KCMPS(KC-NMOC);
            END CG;
                RETURN;
                END CA;
               ELSE PUT FILE (SYSPRINT) EDIT ('YES') (A(3));
            PUT FILE (SYSPRINT) EDIT ('LABEL INFORMATION FOR THE SYSTEM')
                 (SKIP(5),CCLUMN(20),A);
            DO LC=1 TO 4; CALL DLINE; END;
            PUT FILE (SYSPRINT) EDIT ('LABEL','DESCRIPTION')
                 (SKIP(2),COLUMN(10),A,COLUMN(45),A,SKIP(2));
            DC LC=1 TO 2; CALL CLINE; END;
            IF NMOD=0 THEN GOTO LL2;
        L9: DO K1=1 TO NMOD;
            GET FILE (SYSIN) ECIT (COMPS1,MDESCR(K1))
            (COL(7),A(3),CCL(10),A(70));
            SLAB(K1)=COMPS1;
            XCCMPS='';
            DO K10=1 TO 3;
            COMPS2=SUBSTR(COMPS1,K10,1);
            IF COMPS2=' ' THEN GO TO L22;
            XCOMPS=XCOMPS||COMPS2;
       L22: END;
            IF XCOMPS¬='' THEN COMPS(K1),MODSYM(K1)=XCOMPS;
            PUT FILE (SYSPRINT) EDIT ('MODULE    ',COMPS(K1),(20)'.',' ',
                MDESCR(K1))  (SKIP,A,A,COL(15),A,A,A);
            CALL DLINE;
            END L9;
       LL2: IF NCCM=NMOD THEN RETURN;
       LL3: DO K2=NMCD+1 TO NCOM;
            GET FILE (SYSIN) LIST (TYPE(1,K2), PORT(1,K2), FAILS(1,K2));
            CALL SIMULATE;
            FREL(K2)=REL(K2);
            GET FILE (SYSIN) EDIT (COMPS1,DESCR)
            (COL(7),A(3),COL(10),A(70));
            SLAB(K2)=COMPS1;
            XCOMPS='';
            DO K11=1 TO 3;
            COMPS2=SUBSTR(COMPS1,K11,1);
            IF CCMPS2=' ' THEN GO TC L23;
            XCOMPS=XCOMPS||COMPS2;
       L23: END;
            IF XCOMPS¬='' THEN COMPS(K2)=XCOMPS;
                 ELSE XCOMPS=COMPS(K2);
            PUT FILE (SYSPRINT) EDIT ('COMPONENT ',XCOMPS,(20)'.',' ',
                DESCR)  (SKIP(1),A,A,COL(15),A,A,A);
            CALL DLINE;
            END LL3;
            RETURN;
                 /* ENTRY POINT FOR SIMULATION */
    SIMOUT: ENTRY;
            IF LABELS = 0 THEN IF NMOD = 0 | NMOD = NCOM THEN
             BA1: DO;
                 CALL SIMULATE;
                 RETURN;
                 END BA1;
             ELSE DO;
                 CALL SIMULATE;
             CA1:
                 DO KM = NMCD+1 TC NCOM;
                 PREL(KM) = REL(KM);
                 END CA1;
                 RETURN;
                 END;
                      /*  ENTRY POINT TO PRINT CONTROL DATA,
                          AND PROCESS LABELS FOR MODULES  */
      OUT2: ENTRY;
            DO JL=1 TO 128;
            MODSYM(JL)=MCDSY(JL);
            COMPS(JL)=KOMPS(JL);
            END;
            JMOD=JMOD+1;
            CALL DLINE;
             IF LABELS=0 THEN GOTO JC;
             DO JS=1 TC NMCD;
             MODSYM(JS)=SLAB(JS);
```

```
         END;
                  /*  PRINT CONTROL INFORMATICN FOR THE
                      MODULE  */
     JC:IF LABELS=0 THEN PUT FILE (SYSPRINT) ECIT ('MODULE ',
                         MCDSYM(JMOD)) (SKIP(4),A,A); ELSE
        PUT FILE (SYSPRINT) EDIT ('MODULE ',MODSYM(JMOD),' ',(13)'.',
            ' ',MCDESCR(JMOD))  (SKIP(4),A,A,A,A,A,A);
        CALL DLINE;
        PUT FILE (SYSPRINT) EDIT ('NUMBER CF COMPCNENTS ',(16)'.',' ',
            NCOM)  (SKIP(1),A,A,A,F(3));
        CALL DLINE;
        PUT FILE (SYSPRINT) EDIT ('NUMBER OF MINIMAL ',STATE,' ',(13)
            '.',' ',NPATH) (SKIP(1),A,A,A,A,A,F(3));
        DO LC=1 TO 2; CALL DLINE; END;
         IF LABELS = 0 THEN
            LA: DO;
            LB: DO K3 = 1 TC NCOM;
        GET FILE (SYSIN) LIST (TYPE(CE,K3), PORT(CE,K3), FAILS(CE,K3));
              END LB;
        CALL SIMULATE;
              RETURN;
              END LA;
            ELSE PUT FILE (SYSPRINT) EDIT (' LABEL INFORMATION FOR ',
                     'MODULE ',MODSYM(JMOD)) (SKIP(5),COL(20),A,A,A);
        PUT FILE (SYSPRINT) EDIT ('LABEL','DESCRIPTION')
             (SKIP(2),COLUMN(10),A,COLUMN(45),A);
        DO LC=1 TO 7; CALL DLINE; END;
   LL4: DO K3=1 TO NCCM;
        GET FILE (SYSIN) LIST (TYPE(CE,K3), PORT(CE,K3), FAILS(CE,K3));
        GET FILE (SYSIN) EDIT (COMPS1,DESCR)
        (COL(7),A(3),CCL(10),A(70));
        XCOMPS='';
        DO K13=1 TO 3;
        COMPS2=SUBSTR(COMPS1,K13,1);
        IF COMPS2=' ' THEN GO TC L24;
        XCOMPS=XCOMPS||COMPS2;
   L24: END;
        IF SUBSTR(XCOMPS,1,1)¬=''  THEN COMPS(K3)=XCOMPS;
             ELSE XCOMPS=COMPS(K3);
        PUT FILE (SYSPRINT) EDIT ('COMPONENT ',XCOMPS,(20)'.',' ',
            DESCR)  (SKIP(1),A,A,COL(15),A,A,A);
        CALL DLINE;
        END LL4;
        CALL SIMULATE;
        RETURN;
                  /*  ENTRY POINT TO PRINT MINIMAL STATES,
                      AND PROBABILITY EQUATIONS  */
  OLT3: ENTRY;
        DTERM(CE) = 0;
        DCOM(CE) = 0;
        DCOM(CE) = NCCM;
        DTERM(CE) = NTERM;
        DO I=1 TO NTERM;
        DCEF(CE,I) = CCEF(I);
        DERMS(CE,I) = TERMS(I);
        END;
                  /*  PUNCH NUMBER OF COMPONENTS, NUMBER OF
                      TERMS, LABELS AND THE TERMS   */
        IF PUNOUT=0 THEN GO TO L21;
        IF JMCD=0 THEN PUT FILE(PUNCH) EDIT(NCOM,NTERM)
             (CCLUMN(1),F(3),X(5),F(3));
        ELSE PUT FILE(PUNCH) LIST(MODSYM(JMOD),NCOM,NTERM);
        PUT FILE(PUNCH) SKIP LIST((COMPS(I) DO I=1
         TO NCOM));
        PUT FILE(PUNCH) SKIP LIST((COEF(I) DO I=1
         TO NTERM));
        PUT FILE(PUNCH) SKIP LIST((TERMS(I) DO I=1
         TO NTERM));
                  /*  PRINT MINIMAL STATES  */
   L21:CALL DLINE;
        IF JMOD=0 THEN PUT FILE (SYSPRINT) EDIT('THE ',NPATH,
                          ' MINIMAL ',STATE,' FOR THE SYSTEM FOLLOW:')
        (SKIP(2),A,F(3),A,A,A,SKIP(2));   ELSE
        PUT FILE (SYSPRINT) EDIT ('THE ',NPATH,'MINIMAL ',STATE,
            ' FOR MODULE ',MODSYM(JMOD),' FOLLOW:')
             (SKIP(2),A,F(3),X(1),A,A,A,A,A,SKIP(2));
        DO LC=1 TO 2; CALL DLINE; END;
    L5: DO K4=1 TO NPATH;
        MINP='';
        MINP='<';
        MINPL=1;
    L6: DC K5=1 TO NCOM;
        IF SUBSTR(MINPTH(K4),K5,1)='1'B
                         THEN MINP=MINP||CCMPS(K5)||',';
        CHECK1=LENGTH(MINP)/MINPL;
        IF CHECK1>128 THEN CO;
             MINPL=MINPL+1;
             DO I=1 TO 132-CHECK1;
             MINP=MINP || ' ';
             END;
        END;
        END L6;
        NMIN=LENGTH(MINP);
        SUBSTR(MINP,NMIN,1)='>';
        PUT FILE (SYSPRINT) EDIT (MINP)  (SKIP(1),COL(1),A);
        CALL DLINE;
        END L5;
        IF JMOD=0 THEN
             HEAD1='SYSTEM '||CTYPE||' EQUATION (';
             ELSE HEAD1='SUBSYSTEM '||CTYPE||' EQUATION '
            ||'..... MODULE '|| MODSYM(JMOD) || '  (';
          PUT FILE (SYSPRINT) EDIT (HEAD1,NTERM,' TERMS)')
              (SKIP(2),COL(32),A,F(3),A,SKIP(2));
          DO LC=1 TO 2; CALL OLINE; END;
                        /*  DETERMINATION OF COMPONENT SYMBOLS
                            FOR OUTPUT  */
          CHAR1='';
          CHAR2='R';
          IF JMOD=0 THEN CHAR1=CHAR1||'SYS     ';      ELSE
               CHAR1=CHAR1||MODSYM(JMOD)||'      ';
          IF JMOD=0 THEN CHAR2=CHAR2||'     = ';   ELSE DO;
                     DO K10=1 TO LENGTH(MODSYM(JMOD))+1;
               CHAR2=CHAR2||' ';    END;
               CHAR2=CHAR2||'= ';       END;
            K6=0;
      L7: IF K6=NTERM THEN GO TO KK2;
        IF COEF(K6+1)>0 THEN COEFF=' + ';      ELSE DO;
           COEF(K6+1)=-COEF(K6+1);  COEFF=' - ';   END;
             KI6=K6+1;
            IF KI6¬=1 THEN DO;
          CHAR2=CHAR2||COEFF;
          CHAR1=CHAR1||'    ';
                END;
            IF COEF(K6+1)¬=1 THEN DO;
          CHAR1=CHAR1||' ';
          CHAR2=CHAR2||KOMPS(COEF(K6+1));
          END;
             K7=0;
    AG: IF K7=NCOM THEN GO TO KC1;
          IF SUBSTR(TERMS(K6+1),K7+1,1)='0'B THEN GO TO L50;
          CHAR1=CHAR1||COMPS(K7+1)||' ';
                        /*  DETERMINE R-STRING FOR OUTPUT  */
          LEN6=LENGTH(COMPS(K7+1));
          IF LEN6=1 THEN GO TO GO1;
          IF LEN6=2 THEN GO TO GO2;
               ELSE GO TO GO3;
          GO1: CHAR2=CHAR2||'R ';
            GO TO L11;
          GO2: CHAR2=CHAR2||'R  ';
            GO TO L11;
          GO3: CHAR2=CHAR2||'R   ';
    L11:LEN=LENGTH(CHAR2);
          CALL SLINE;
    L50:K7=K7+1;
          GOTO AG;
    KC1:K6=K6+1;
          GOTO L7;
    KK2:CALL SLINE;
          CLOSE FILE(PUNCH);
          END OUTI;
  SLINE:PROCEDURE;
          DCL (LEN,NCOM,NTERM) REAL FIXED DEC (3) EXT;
          IF LEN>112 & LEN <120 THEN DO;
          DCL CHAR2 CHAR(120) VAR EXT;
          DCL (K6,K7) REAL FIXED DEC (3) EXT;
            IF K7+1¬=NCOM THEN CHAR2=CHAR2||'*';
              CALL PRINTER;
              RETURN;
            END;
          ELSE DO;
            IF K6=NTERM THEN CALL PRINTER;
                   RETURN;
          END;
          END SLINE;
  PRINTER: PROCEDURE;
          DCL (CHAR1,CHAR2) CHAR(120) VAR EXT;
          DCL LEN REAL FIXED DEC (3) EXT;
          CALL OLINE;
          PUT FILE (SYSPRINT) EDIT (CHAR2) (SKIP(2),COL(3),A);
          PUT FILE (SYSPRINT) EDIT (CHAR1) (COL(4),A);
          CALL OLINE; CALL OLINE;
          LEN=0;
          CHAR1='';
          CHAR2='';
          RETURN;
  END PRINTER;
  OLINE:PROCEDURE;
          DCL JEN REAL FIXED DEC (3) EXT;
          DCL NPAGE REAL FIXED DEC (4) EXT;
          JEN=JEN+1;
          IF JEN<53 THEN GO TO OVER;
          PUT FILE (SYSPRINT) EDIT ('** CONTINUED **')
              (SKIP(3),COL(60),A);
          NPAGE=NPAGE+1;
          PUT FILE (SYSPRINT) EDIT ('PAGE',NPAGE)
              (PAGE,COL(111),A,X(1),F(4),SKIP(2));
          JEN=0;
      OVER: RETURN;
          END OLINE;
COMPUTE: PROCEDURE;
          DCL A REAL FLOAT DEC (14);
          DCL AV REAL FIXED DEC(3) EXT;
          DCL AVREL REAL FLOAT DEC(14) EXT;
          DCL BROD REAL FLOAT DEC (14);
           DCL CE REAL FIXED DEC(3) EXT;
          DCL DCOM(MAXEL) REAL FIXED DEC(3) CONTROLLED EXT;
          DCL DMOD REAL FIXED DEC (3) EXT;
          DCL DERMS(MAXEL,MAXEL) BIT (128) CONTROLLED VAR EXT;
          DCL DOEF(MAXEL,MAXEL) REAL FIXED DEC(4) CONTROLLED EXT;
          DCL DTERM(MAXEL) REAL FIXED DEC(3) CONTROLLED EXT;
          DCL KA REAL FIXED DEC (3);
          DCL KB REAL FIXED DEC (3);
          DCL KV REAL FIXED DEC (3) EXT;
          DCL MODREL REAL FLOAT DEC (14) EXT;
          DCL PMB REAL FLOAT DEC (14);
          DCL REL(MAXCOM) REAL FLOAT DEC(14) CONTROLLED EXT;
          DCL RELSTO(SIMNUM) REAL FLOAT DEC(14) CONTROLLED EXT;
          DCL SN REAL FIXED DEC(6,0) EXT;
          DCL SREL(NMOD) REAL FLOAT DEC(14) CONTROLLED EXT;
          DCL SYSREL REAL FLOAT DEC (14) EXT;
          DCL ZUM REAL FLOAT DEC (14);
          ZUM=0.0;
          DO KA = 1 TO DTERM(CE);
          PMB=0.0 ;
          BROD=1.0;
          A=0.0   ;
          DO KB = 1 TO DCOM(CE);
          IF SUBSTR(DERMS(CE,KA), KB, 1) ¬= '1'B THEN GO TO FIN;
          A=BROD*REL(KB);
          BROD=A;
          A=0.0;
     FIN: END;
          PMB = BROD * DOEF(CE,KA);
          ZUM=ZUM+PMB;
          END;
          IF DMOD¬=0 THEN DO;
          MODREL=ZUM;
          SREL(KV)=ZUM;
          KV=KV+1;
          END;
          ELSE DO;
          SYSREL=ZUM;
          IF AV = 2 THEN AVREL = SYSREL;
          ELSE
          RELSTO(SN) = SYSREL;
           CE = 1;
          END;
          END COMPUTE;
  CUTII: PROCEDURE;
                    /*  DECLARE STATEMENTS  */
          DCL ATYPE1 CHAR(13) VARYING EXT;
          DCL ATYPE2 CHAR(15) VARYING;
          DCL ATYPE3 CHAR(13) VARYING;
          DCL BEGA(0:2) CHAR(7) INITIAL ('MODULE ',
          'BETA   ', 'GAMMA  ');
          DCL BTYPE REAL FIXED DEC(1) EXT;
          DCL (C1,C2,I,LC) REAL FIXED DEC (3);
          DCL CE REAL FIXED DEC(3) EXT;
          DCL DCOM(MAXEL) REAL FIXED DEC(3) CONTROLLED EXT;
          DCL DMOD REAL FIXED DEC (3) EXT;
          DCL JMOD REAL FIXED DEC (3) EXT;
          DCL KQ REAL FIXED DEC (3);
          DCL COMPS(128) CHAR(3) VAR EXT STATIC;
          DCL MODREL REAL FLOAT DEC (14) EXT;
          DCL MODSYM (128) CHAR (3) EXT;
          DCL REL(MAXCOM) REAL FLOAT DEC(14) CONTROLLED EXT;
          DCL SYSREL REAL FLOAT DEC (14) EXT;
                    /*  ENTRY POINT TO PRINT HEADINGS  */
  SOUT1: ENTRY;
          PUT FILE (SYSPRINT) EDIT ('SPARCS: PROBABILITY COMPUTATION ',
          'ROUTINE')  (SKIP(5),A,A);
          PUT FILE (SYSPRINT) EDIT ('(THE COMPONENT AND SYSTEM ',
          'RELIABILITY INFORMATION IS FOR THE FIRST ITERATION ONLY)')
          (SKIP(1),A,A);
          DO LC=1 TO 5; CALL DLINE; END;
          RETURN;
                    /*  ENTRY POINT TO PRINT PROBABILITIES */
  SOUT2: ENTRY;
          IF BTYPE = 0 THEN DO;
               ATYPE1='RELIABILITY';
               ATYPE2='RELIABILITIES';
               ATYPE3='UNRELIABILITY';  END;    ELSE DO;
               ATYPE1='UNRELIABILITY';
               ATYPE2='UNRELIABILITIES';
               ATYPE3='RELIABILITY';   END;
          IF DMOD ¬= 0 THEN
          PUT FILE(SYSPRINT) EDIT('COMPONENT ',ATYPE2,' FOR MODULE ',
          MODSYM(JMOD))  (SKIP(3),COL(1),A,A,A,A);     ELSE
          PUT FILE(SYSPRINT) EDIT('MODULE AND COMPONENT ',ATYPE2,
          ' FOR THE SYSTEM')
              (SKIP(3),COLUMN(1),A,A,A);
          DO LC=1 TO 3; CALL DLINE; END;
          C1=1;
          IF DCOM(CE) < 4 THEN C2 = DCOM(CE);
           ELSE C2 = 4;
  WW:     DO KQ = 1 TO DCOM(CE)/4 + .9;
           PUT FILE(SYSPRINT) EDIT (('R   = ', REL(I) DO I=C1 TO C2))
           (SKIP(2), COL(1), (5) (A(6), F(8,6), X(12)));
          PUT FILE (SYSPRINT) EDIT ((COMPS(I), 'TYPE = ',
          BEGA(TYPE(CE,I)) DO I = C1 TO C2))
          (COL(2), (5)(A(3),X(2),A,A,X(7)));
          PUT FILE (SYSPRINT) EDIT (('P CR TIME = ', PORT(CE,I) DO I =
          C1 TO C2)) (COL(7), (5) (A, F(7,2), X(7)));
          PUT FILE (SYSPRINT) EDIT (('FAILURES = ', FAILS(CE,I) DO I =
          C1 TO C2)) (COL(7), (5) (A, F(7,2), X(8)));
          DO LC=1 TO 3; CALL DLINE; END;
           C1 = C1 + 4;
          IF C2+4 > DCOM(CE) THEN C2 = DCOM(CE);
           ELSE C2 = C2 + 4;
          END WW;
          IF DMOD¬=0 THEN
          PUT FILE (SYSPRINT) EDIT ('MODULE ',MODSYM(JMOD), ATYPE1,
                ' = ',MODREL, ATYPE3,' = ',1.-MODREL)
                (SKIP(2),COL(1),A,A,A,A,F(8,6),X(15),A,A,F(8,6)); ELSE
                 PUT FILE (SYSPRINT) EDIT ('SYSTEM ', ATYPE1, ' = ', SYSREL,
                 ATYPE3, ' = ', 1.-SYSREL)
                 (SKIP(2), COL(1), A, A, A, A, F(8,6),X(15), A, A, F(8,6));
                DO LC=1 TO 2; CALL DLINE; END;
                RETURN;
                        /* ENTRY POINT FOR SIMULATION OUTPUT */
      SIMOT:  ENTRY;
                IF BTYPE = 0 THEN
                   PUT FILE (SYSPRINT) EDIT (SYSREL, 1.-SYSREL)
                      (COL(22), F(8,6), X(31), F(8,6));
                ELSE
                   PUT FILE (SYSPRINT) EDIT (SYSREL, 1.-SYSREL)
                      (COL(24), F(8,6), X(29), F(8,6));
                DO LC = 1 TO 2;
                CALL DLINE;
                END;
                RETURN;
                END OUTII;
      SIMULATE: PROCEDURE;
             DCL AV REAL FIXED DEC(3) EXT;
             DCL BTYPE REAL FIXED DEC(1) EXT;
             DCL CD REAL FIXED DEC(3) EXT;
             DCL CE REAL FIXED DEC(3) EXT;
             DECLARE FLAG EXTERNAL;
             DCL GAMMA REAL FLOAT DEC(14) EXT;
                DCL NARG FIXED BINARY(31,0) EXT;
             DCL NCOM REAL FIXED DEC(3) EXT;
             DCL NMOD REAL FIXED DEC(3) EXT;
             DCL NVAL FIXED (7,6) EXT;
             DCL REL(MAXCOM) REAL FLOAT DEC(14) CONTROLLED EXT;
             DCL SEED REAL FLOAT DEC(16) EXT;
             DCL (TYPE(MAXEL,MAXCOM), PORT(MAXEL,MAXCOM), FAILS(MAXEL,MAXCOM))
                REAL FLOAT DEC(14) CONTROLLED EXT;
      REED: DO CD = 1 TO NCOM;
             IF TYPE (CE,CD) = 0 THEN DO;
                REL(CD) = 0;
                GO TO ER;
                END;
              ELSE
             IF TYPE (CE,CD) = 1 THEN GO TO BETAVAL;
             ELSE IF TYPE(CE,CD) = 2 THEN GO TO GAMMAVAL;
             PUT SKIP LIST (TYPE(CE,CD));
             PUT LIST ('TYPE DESIGNATED IN ERROR');
             GO TO WRITEM;
      BETAVAL: CALL BETASUB;
             GO TO ER;
      GAMMAVAL: CALL GAMASUB;
             GO TO ER;
      WRITEM: PUT SKIP(2) EDIT (GAMMA,FLAG,NVAL,PORT,FAILS,TYPE,PSUBL)
               (COLUMN(2),F(7,5), COLUMN(12),F(2,0), COLUMN(18),F(9,6),
               COLUMN(29),F(9,2), COLUMN(41),F(4,0), COLUMN(49),F(2,0),
               COLUMN(55),F(8,6));
             RETURN;
      ER:
             IF BTYPE = 1 THEN REL(CD) = 1 - REL(CD);
             ELSE;
             IF TYPE(CE,CD) = 1 THEN GAMMA = RANF(NARG);
             ELSE;
             END REED;
             RETURN;
      BETASUB: PROCEDURE;
             DCL AV REAL FIXED DEC(3) EXT;
             DCL CD REAL FIXED DEC(3) EXT;
             DCL CE REAL FIXED DEC(3) EXT;
             DCL GAMMA REAL FLOAT DEC(14) EXT;
              DCL IER REAL FIXED BIN(31);
             DECLARE LAMDA FIXED(12,10);
             DCL NCOM REAL FIXED DEC(3) EXT;
             DCL NVAL FIXED (7,6) EXT;
              DCL (P,AA,B,X) REAL FLOAT DEC;
             DCL REL(MAXCOM) REAL FLOAT DEC(14) CONTROLLED EXT;
             DCL (TYPE(MAXEL,MAXCOM), PORT(MAXEL,MAXCOM), FAILS(MAXEL,MAXCOM))
                REAL FLOAT DEC(14) CONTROLLED EXT;
             AA = PORT(CE,CD) + 1;
             B = FAILS(CE,CD) + 1;
              P = GAMMA;
             IF AV = 2 THEN DO;
                  REL(CD) = AA / (AA+B);
                  GO TO BETA1;
                  END;
             ELSE;
              CALL MDBETI (P,AA,B,X,IER);
              REL(CD) = X;
      BETA1: RETURN;
             END BETASUB;
      GAMASUB: PROCEDURE;
              DCL (AA,B,R(1)) REAL FLOAT DEC;
             DCL AV REAL FIXED DEC(3) EXT;
             DCL CD REAL FIXED DEC(3) EXT;
             DCL CE REAL FIXED DEC(3) EXT;
             DCL GAMMA REAL FLOAT DEC(14) EXT;
             DCL NI REAL FIXED BIN(31) INITIAL(1);
             DCL NCOM REAL FIXED DEC(3) EXT;
             DCL NVAL FIXED (7,6) EXT;
             DCL REL(MAXCOM) REAL FLOAT DEC(14) CONTROLLED EXT;
             DCL SEED REAL FLOAT DEC(16) EXT;
             DCL (TYPE(MAXEL,MAXCOM), PORT(MAXEL,MAXCOM), FAILS(MAXEL,MAXCOM))
                REAL FLOAT DEC(14) CONTROLLED EXT;
              DCL WA(MA) REAL FLOAT DEC CONTROLLED;
              AA = FAILS(CE,CD) + 1;
              B = 1 / (PORT(CE,CD) + 1);
              IF AV = 2 THEN DO;
                   REL(CD) = ((1/B) / (1/B+1)) ** AA;
                   GO TO GAM;
                   END;
              ELSE;
               ALLOCATE WA(AA+1);
              CALL GGTMAJ (SEED,AA,B,NI,WA(1),R(1));
              FREE WA;
               REL(CD) = EXP(-R(1));
         GAM: RETURN;
              END GAMASUB;
         FINIS: END SIMULATE;
         (NOFIXEDOVERFLOW):                                                     RANF0010
         RANF:   PROC(NARG) RETURNS(FLOAT BINARY);                              RANF0020
            /*                                                                  RANF0030
            THIS FUNCTION GENERATES PSEUDO-RANDOM NUMBERS, UNIFORMLY            RANF0040
            DISTRIBUTED ON (0,1).  THIS VERSION IS FOR THE IBM 360.             RANF0050
            J. P. CHANDLER, COMPUTER SCIENCE DEPT., OKLAHOMA STATE UNIVERSITY.  RANF0060
                                                                                RANF0070
            METHOD... COMPOSITE OF THREE MULTIPLICATIVE CONGRUENTIAL GENERATORS RANF0080
               G. MARSAGLIA AND T. A. BRAY, COMM. ACM 11 (1968) 757.            RANF0090
                                                                                RANF0100
            IF RANF IS CALLED WITH NARG=0, THE NEXT RANDOM NUMBER IS RETURNED.  RANF0110
            IF RANF IS CALLED WITH NARG¬=0, THE GENERATOR IS RE-INITIALIZED     RANF0120
            USING IABS(2*NARG+1) AND THE FIRST RANDOM NUMBER FROM THE NEW       RANF0130
            SEQUENCE IS RETURNED.                                               RANF0140
            */                                                                  RANF0150
                 DCL J FIXED BINARY(15,0) STATIC;                               RANF0160
                 DCL JRAN BASED(P_RAN) FIXED BINARY(31,0);                      RANF0170
                 DCL K FIXED BINARY(31,0) INITIAL(7654321) STATIC;              RANF0180
                 DCL KLM FIXED BINARY(31,0) STATIC;                             RANF0190
                 DCL L FIXED BINARY(31,0) INITIAL(7654321) STATIC;              RANF0200
                 DCL M FIXED BINARY(31,0) INITIAL(7654321) STATIC;              RANF0210
                 DCL MK FIXED BINARY(31,0) STATIC INITIAL(282629);              RANF0220
                 DCL ML FIXED BINARY(31,0) STATIC INITIAL(34821);               RANF0230
                 DCL MM FIXED BINARY(31,0) STATIC INITIAL(65541);               RANF0240
                 DCL N(128) FIXED BINARY(31,0) STATIC;                          RANF0250
                 DCL NARG FIXED BINARY(31,0);                                   RANF0260
                 DCL NDIV FIXED BINARY(31,0) STATIC;                            RANF0270
                 DCL NFIRST BIT(1) STATIC INITIAL('1'B);                        RANF0280
                 DCL NR FIXED BINARY(31,0) STATIC;                              RANF0290
                 DCL P_RAN POINTER STATIC;                                      RANF0300
                 DCL RAN FLOAT BINARY STATIC;                                   RANF0310
                 DCL RDIV FLOAT BINARY STATIC;                                  RANF0320
                 IF NARG ¬= 0 THEN                                              RANF0330
                   DO;                                                          RANF0340
                   /*                                                           RANF0350
                   RE-INITIALIZE USING NARG.                                    RANF0360
                   */                                                           RANF0370
                      KLM = ABS(2 * NARG + 1);                                  RANF0380
                      K,L,M = KLM;                                              RANF0390
                   END;                                                         RANF0400
                 ELSE                                                           RANF0410
                   DO;                                                          RANF0420
                      IF ¬ NFIRST THEN GO TO SKIP;                              RANF0430
                   END;                                                         RANF0440
            /*                                                                  RANF0450
            INITIALIZE THE ROUTINE.                                             RANF0460
            */                                                                  RANF0470
                 P_RAN = ADDR(RAN);                                             RANF0480
                 NFIRST = '0'B;                                                 RANF0490
                 NDIV = 16777216;                                               RANF0500
                 RDIV = 32768.0 * 65536.0;                                      RANF0510
            /*                                                                  RANF0520
            FILL THE TABLE.                                                     RANF0530
            */                                                                  RANF0540
                 DO J = 1 TO 128;                                               RANF0550
                      K = K * MK;                                               RANF0560
                      N(J) = K;                                                 RANF0570
                   END;                                                         RANF0580
            /*                                                                  RANF0590
            COMPUTE THE NEXT RANDOM NUMBER.                                     RANF0600
            */                                                                  RANF0610
         SKIP:   L = L * ML;                                                    RANF0620
                 J = 1 + ABS(L) / NDIV;                                         RANF0630
                 M = M * MM;                                                    RANF0640
                 NR = ABS(N(J) + L + M);                                        RANF0650
                 RAN = FLOAT(NR) / RDIV;                                        RANF0660
            /*                                                                  RANF0670
            FIXUP THE LEAST SIGNIFICANT BIT.                                    RANF0680
            */                                                                  RANF0690
                 IF J > 64  &  RAN < 1.0 THEN JRAN = JRAN + 1;                  RANF0700
            /*                                                                  RANF0710
            REFILL THE J-TH PLACE IN THE TABLE.                                 RANF0720
            */                                                                  RANF0730
                 K = K * MK;                                                    RANF0740
                 N(J) = K;                                                      RANF0750
                 RETURN(RAN);                                                   RANF0760
                 END RANF;                                                      RANF0770


                 CLOSE FILE(SYSIN), FILE(SYSPRINT);
                 END UNITED;
```

APPENDIX D

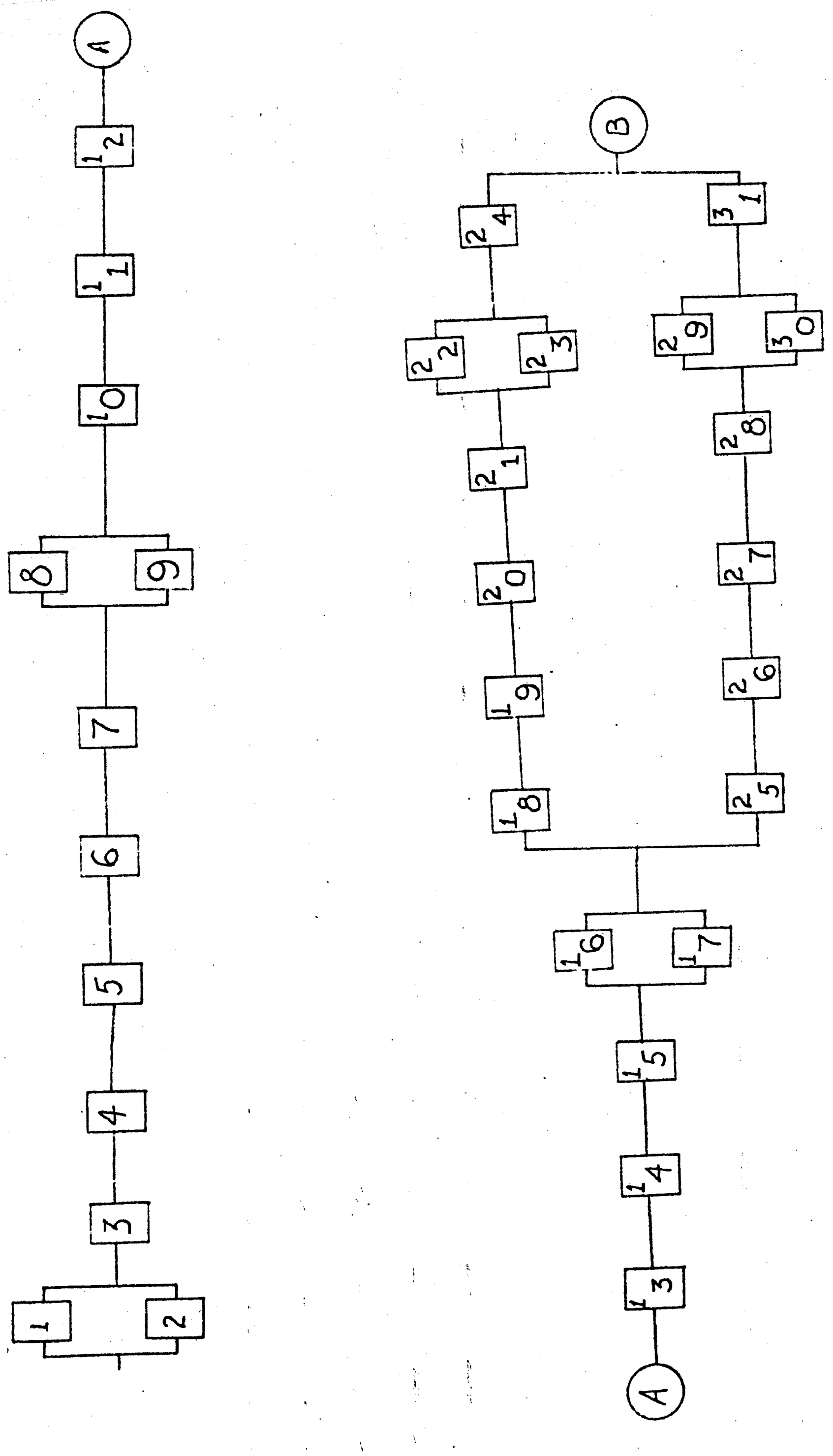
APOLLO-SATURN LUNAR EXCURSION MODULE (LEM) LARGE SAMPLE RUN FROM PRESSURIZATION THROUGH POWERED ASCENT

LEM Large System Test

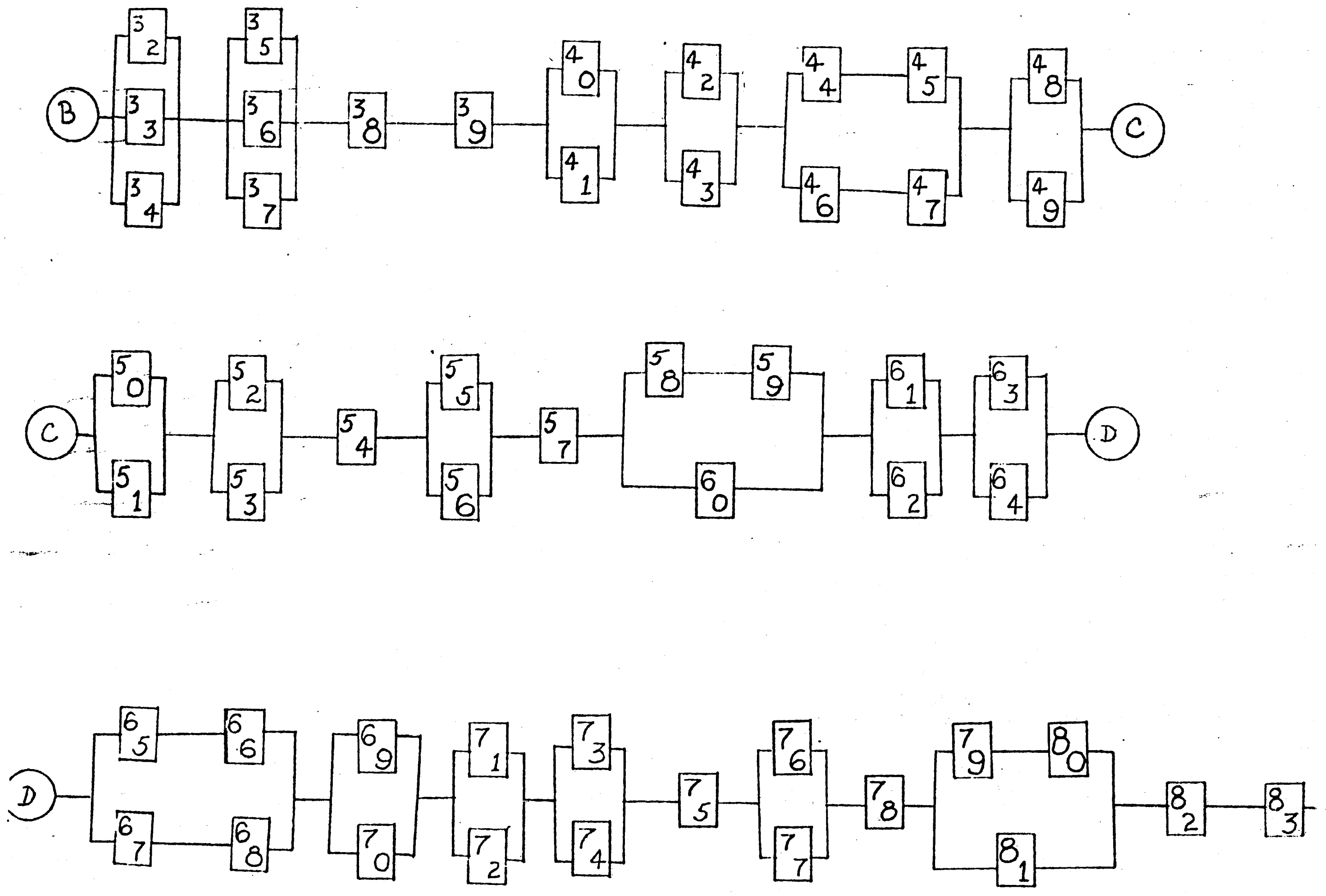
SPARCS was tested using a large network consisting of both beta and gamma components placed throughout the network in an arbitrary pattern. A network diagram was obtained for the Apollo Lunar Excursion Module (LEM) from pressurization through powered ascent. Although no data was provided with the network, component test data from previous Apollo-Saturn tests was found and arbitrarily placed in the LEM network.

The network is a logically complex network consisting of both series and parallel components. It is subdivided into 13 modules each of which contains a varying number of components. Beta and gamma components are arbitrarily dispersed throughout the network. Thus to assess this system, the module would have to handle a large complex network, using the modularity concept, with the two component types being randomly interspersed.

The run results and system assessment are presented in this appendix. Due to the size of the network, only 50 simulation runs were made. These runs took approximately 15 minutes on the IBM 360/65 at Oklahoma State and almost 2 minutes on the IBM 370/124 at Phillips in Bartlesville, Oklahoma. The results showed that SPARCS could adequately handle a system of any reasonable size and configuration.



Apollo Lunar Excursion Module (LEM) from
Pressurization Through Powered Ascent


B
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
C
C
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
D
D
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83

```
MAPS-I: EQUATION GENERATION ROUTINE
MODEL FOR THE ANALYSIS OF PROBABILITIES OF SYSTEMS
COLLEGE OF BUSINESS ADMINISTRATION, OKLAHOMA STATE UNIVERSITY

SYSTEM IDENTIFICATION ...............  LEM RELIABILITY INTEGRATED ASCENT PRESSURIZATION & FEED SYSTEM

NUMBER OF MODULES ...................  13
NUMBER OF COMPONENTS ................   2
TOTAL NUMBER OF SYSTEM ELEMENTS .....  15
NUMBER OF MINIMAL PATHS .............   1
PUNCHED OUTPUT OF EQUATION ..........  NO
LABELES SUPPLIED BY USER ............  NO

THE   1 MINIMAL PATHS FOR THE SYSTEM FOLLOW:
<A,B,C,D,E,F,G,H,I,J,K,L,M,1,2>

                              SYSTEM RELIABILITY EQUATION (  1 TERMS)

 R     = R R R R R R R R R R R R R R R
  SYS     A B C D E F G H I J K L M 1 2


MODULE A
NUMBER OF COMPONENTS ................   7
NUMBER OF MINIMAL PATHS .............   2

THE   2 MINIMAL PATHS FOR MODULE A   FOLLOW:
<1,3,4,5,6,7>
<2,3,4,5,6,7>

                              SUBSYSTEM RELIABILITY EQUATION ..... MODULE A    (  3 TERMS)

 R     = R R R R R R  + R R R R R R  - R R R R R R R
  A       1 3 4 5 6 7    2 3 4 5 6 7    1 2 3 4 5 6 7


MAPS-II: PROBABILITY COMPUTATION ROUTINE

COMPONENT RELIABILITIES FOR MODULE A

R    = 0.994544         R    = 0.997938         R    = 0.998947         R    = 0.996224
 1     TYPE =     2.00   2     TYPE =     2.00   3     TYPE =     2.00   4     TYPE =     2.00
       P OR TIME =  250.00     P OR TIME =  256.10     P OR TIME =  309.20     P OR TIME =  315.00
       FAILURES =     0.00     FAILURES =     0.00     FAILURES =     0.00     FAILURES =     0.00

R    = 0.987663         R    = 0.998626         R    = 0.998492
 5     TYPE =     2.00   6     TYPE =     2.00   7     TYPE =     2.00
       P OR TIME =  310.00     P OR TIME =  325.20     P OR TIME =  275.80
       FAILURES =     1.00     FAILURES =     1.00     FAILURES =     0.00

MODULE A  RELIABILITY = 0.980056              UNRELIABILITY = 0.019944

                                                         ** CONTINUED **


MODULE B
NUMBER OF COMPONENTS ................  10
NUMBER OF MINIMAL PATHS .............   4

THE   4 MINIMAL PATHS FOR MODULE B   FOLLOW:
<1,3,4,5,6,7,8,9>
<1,3,4,5,6,7,8,10>
<2,3,4,5,6,7,8,9>
<2,3,4,5,6,7,8,10>

                              SUBSYSTEM RELIABILITY EQUATION ..... MODULE B    (  9 TERMS)

 R     = R R R R R R R R  + R R R R R R R R   - R R R R R R R R R   + R R R R R R R R  - R R R R R R R R R  + R R R *
  B       1 3 4 5 6 7 8 9    1 3 4 5 6 7 8 10    1 3 4 5 6 7 8 9 10    2 3 4 5 6 7 8 9    1 2 3 4 5 6 7 8 9    2 3 4

 R R R R R   + R R R R R R R R R R  - R R R R R R R R R  - R R R R R R R R R
  5 6 7 8 10    1 2 3 4 5 6 7 8 9 10   1 2 3 4 5 6 7 8 10   2 3 4 5 6 7 8 9 10


MAPS-II: PROBABILITY COMPUTATION ROUTINE

COMPONENT RELIABILITIES FOR MODULE B

R    = 0.971001         R    = 0.995998         R    = 0.996695         R    = 0.999081
 1     TYPE =     1.00   2     TYPE =     1.00   3     TYPE =     2.00   4     TYPE =     2.00
       P OR TIME =  275.00     P OR TIME =  275.00     P OR TIME =  302.00     P OR TIME =  300.00
       FAILURES =     1.00     FAILURES =     1.00     FAILURES =     0.00     FAILURES =     0.00

R    = 0.994775         R    = 0.997664         R    = 0.994522         R    = 0.993460
 5     TYPE =     2.00   6     TYPE =     2.00   7     TYPE =     2.00   8     TYPE =     2.00
       P OR TIME =  252.00     P OR TIME =  325.00     P OR TIME =  325.00     P OR TIME =  302.00
       FAILURES =     0.00     FAILURES =     1.00     FAILURES =     1.00     FAILURES =     0.00

R    = 0.986553         R    = 0.992491
 9     TYPE =     1.00  10     TYPE =     1.00
       P OR TIME =  249.00     P OR TIME =  251.00
       FAILURES =     1.00     FAILURES =     1.00

MODULE B  RELIABILITY = 0.976210              UNRELIABILITY = 0.023790
MODULE C
NUMBER OF COMPONENTS ................  14
NUMBER OF MINIMAL PATHS .............   4

THE   4 MINIMAL PATHS FOR MODULE C   FOLLOW:
<1,2,3,4,5,7>
<1,2,3,4,6,7>
<8,9,10,11,12,14>
<8,9,10,11,13,14>

                                  SUBSYSTEM RELIABILITY EQUATION ..... MODULE C    ( 15 TERMS)

 R    = R R R R R R  + R R R R R R  - R R R R R R R  + R R R  R  R  R   - R R R R R R R R R  R  R  R   - R R R R R *
  C      1 2 3 4 5 7    1 2 3 4 6 7    1 2 3 4 5 6 7    8 9 10 11 12 14    1 2 3 4 5 7 8 9 10 11 12 14    1 2 3 4 6

 R R R R  R  R  R   + R R R R R R R R R  R  R  R   + R R R  R  R  R   - R R R R R R R R R  R  R  R   - R R R R R *
  7 8 9 10 11 12 14    1 2 3 4 5 6 7 8 9 10 11 12 14    8 9 10 11 13 14    1 2 3 4 5 7 8 9 10 11 13 14    1 2 3 4 6

 R R R R  R  R  R   + R R R R R R R R R  R  R  R   - R R R  R  R  R  R   + R R R R R R R R  R  R  R  R   + R R *
  7 8 9 10 11 13 14    1 2 3 4 5 6 7 8 9 10 11 13 14    8 9 10 11 12 13 14    1 2 3 4 5 7 8 9 10 11 12 13 14    1 2

 R R R R R R R  R  R  R  R   - R R R R R R R R R  R  R  R  R
  3 4 6 7 8 9 10 11 12 13 14    1 2 3 4 5 6 7 8 9 10 11 12 13 14


MAPS-II: PROBABILITY COMPUTATION ROUTINE


COMPONENT RELIABILITIES FOR MODULE C

R    = 0.985782            R    = 0.995716            R    = 0.981222            R    = 0.989727
 1     TYPE =     2.00      2     TYPE =     2.00      3     TYPE =     2.00      4     TYPE =     2.00
       P OR TIME =  242.60        P OR TIME =  275.70        P OR TIME =  231.60        P OR TIME =  209.00
       FAILURES =     1.00        FAILURES =     1.00        FAILURES =     1.00        FAILURES =     0.00

R    = 0.996925            R    = 0.986412            R    = 0.996692            R    = 0.991958
 5     TYPE =     1.00      6     TYPE =     1.00      7     TYPE =     2.00      8     TYPE =     2.00
       P OR TIME =  223.00        P OR TIME =  252.00        P OR TIME =  305.20        P OR TIME =  300.10
       FAILURES =     1.00        FAILURES =     1.00        FAILURES =     1.00        FAILURES =     1.00

R    = 0.992307            R    = 0.985021            R    = 0.983919            R    = 0.995136
 9     TYPE =     2.00     10     TYPE =     2.00     11     TYPE =     2.00     12     TYPE =     1.00
       P OR TIME =  310.20        P OR TIME =  325.20        P OR TIME =  251.50        P OR TIME =  220.00
       FAILURES =     1.00        FAILURES =     1.00        FAILURES =     0.00        FAILURES =     1.00

R    = 0.995567            R    = 0.989237
13     TYPE =     1.00     14     TYPE =     2.00
       P OR TIME =  206.00        P OR TIME =  220.40
       FAILURES =     1.00        FAILURES =     1.00

MODULE C  RELIABILITY = 0.997187              UNRELIABILITY = 0.002813


MODULE D
NUMBER OF COMPONENTS ................   3
NUMBER OF MINIMAL PATHS .............   3

THE   3 MINIMAL PATHS FOR MODULE D   FOLLOW:
<1>
<2>
<3>

                                  SUBSYSTEM RELIABILITY EQUATION ..... MODULE D    (  7 TERMS)

 R     = R  + R  - R R  + R  - R R  - R R  + R R R
  D       1    2    1 2    3    1 3    2 3    1 2 3

MAPS-II: PROBABILITY COMPUTATION ROUTINE

COMPONENT RELIABILITIES FOR MODULE D

R    = 0.996975            R    = 0.992268            R    = 0.997890
 1     TYPE =     1.00      2     TYPE =     1.00      3     TYPE =     1.00
       P OR TIME =  245.00        P OR TIME =  230.00        P OR TIME =  240.00
       FAILURES =     1.00        FAILURES =     1.00        FAILURES =     1.00

MODULE D  RELIABILITY = 1.000000              UNRELIABILITY = 0.000000
MODULE E
NUMBER OF COMPONENTS ................   3
NUMBER OF MINIMAL PATHS .............   3

THE   3 MINIMAL PATHS FOR MODULE E   FOLLOW:
<1>
<2>
<3>

                              SUBSYSTEM RELIABILITY EQUATION ..... MODULE E    (  7 TERMS)

  R    = R  + R  - R R  + R  - R R  - R R  + R R R
   E      1    2    1 2    3    1 3    2 3    1 2 3


MAPS-II: PROBABILITY COMPUTATION ROUTINE


COMPONENT RELIABILITIES FOR MODULE E

R   = 0.984993          R   = 0.987080          R   = 0.997087
 1     TYPE =     1.00   2     TYPE =     1.00   3     TYPE =     1.00
       P OR TIME =  225.00     P OR TIME =  240.00     P OR TIME =  232.00
       FAILURES =    1.00      FAILURES =    1.00      FAILURES =    1.00

MODULE E  RELIABILITY = 0.999999              UNRELIABILITY = 0.000001


MODULE F
NUMBER OF COMPONENTS ................   6
NUMBER OF MINIMAL PATHS .............   4

THE   4 MINIMAL PATHS FOR MODULE F   FOLLOW:
<1,2,3,5>
<1,2,3,6>
<1,2,4,5>
<1,2,4,6>

                              SUBSYSTEM RELIABILITY EQUATION ..... MODULE F    (  9 TERMS)

                                                                                  PAGE    5
                                                ** CONTINUED **

R    = R R R R  + R R R R  - R R R R R  + R R R R  - R R R R R  + R R R R  + R R R R R R  - R R R R R  - R R R R *
 F      1 2 3 5    1 2 3 6    1 2 3 5 6    1 2 4 5    1 2 3 4 5    1 2 4 6    1 2 3 4 5 6    1 2 3 4 6    1 2 4 5

R
 6


MAPS-II: PROBABILITY COMPUTATION ROUTINE


COMPONENT RELIABILITIES FOR MODULE F

R   = 0.996050          R   = 0.998690          R   = 0.997227          R   = 0.989677
 1     TYPE =     2.00   2     TYPE =     2.00   3     TYPE =     1.00   4     TYPE =     1.00
       P OR TIME =  250.90     P OR TIME =  272.30     P OR TIME =  249.00     P OR TIME =  262.00
       FAILURES =    1.00      FAILURES =    1.00      FAILURES =    1.00      FAILURES =    1.00

R   = 0.998559          R   = 0.995052
 5     TYPE =     2.00   6     TYPE =     2.00
       P OR TIME =  279.40     P OR TIMF =  220.90
       FAILURES =    1.00      FAILURES =    0.00

MODULE F  RELIABILITY = 0.994709              UNRELIABILITY = 0.005291


MODULE G
NUMBER OF COMPONENTS ................   6
NUMBER OF MINIMAL PATHS .............   4

THE   4 MINIMAL PATHS FOR MODULE G   FOLLOW:
<1,2,5>
<1,2,6>
<3,4,5>
<3,4,6>

                              SUBSYSTEM RELIABILITY EQUATION ..... MODULE G    (  9 TERMS)

  R    = R R R  + R R R  - R R R R  + R R R  - R R R R R  + R R R  + R R R R R R  - R R R R R  - R R R R
   G      1 2 5    1 2 6    1 2 5 6    3 4 5    1 2 3 4 5    3 4 6    1 2 3 4 5 6    1 2 3 4 6    3 4 5 6


MAPS-II: PROBABILITY COMPUTATION ROUTINE


COMPONENT RELIABILITIES FOR MODULE G

R   = 0.985847          R   = 0.990353          R   = 0.995867          R   = 0.982296
 1     TYPE =     1.00   2     TYPE =     1.00   3     TYPE =     1.00   4     TYPE =     1.00
       P OR TIME =  205.00     P OR TIME =  249.00     P OR TIME =  179.00     P OR TIME =  200.00
       FAILURES =    1.00      FAILURES =    2.00      FAILURES =    1.00      FAILURES =    1.00

R   = 0.997993          R   = 0.990351
 5     TYPE =     1.00   6     TYPE =     1.00
```

```
      P OR TIME =  225.00       P OR TIME =  250.00
      FAILURES =     1.00       FAILURES =     2.00

MODULE G  RELIABILITY = 0.999466                UNRELIABILITY = 0.000534

MODULE H
NUMBER OF COMPONENTS ................   5
NUMBER OF MINIMAL PATHS .............   4

THE   4 MINIMAL PATHS FOR MODULE H   FOLLOW:
<1,3,5>
<1,4,5>
<2,3,5>
<2,4,5>

                              SUBSYSTEM RELIABILITY EQUATION ..... MODULE H    ( 9 TERMS)

 R    = R R R  + R R R  - R R R R  + R R R  - R R R R  + R R R  + R R R R R  - R R R R  - R R R R
  H      1 3 5    1 4 5    1 3 4 5    2 3 5    1 2 3 5    2 4 5    1 2 3 4 5    1 2 4 5    2 3 4 5


MAPS-II: PROBABILITY COMPUTATION ROUTINE


COMPONENT RELIABILITIES FOR MODULE H

R    = 0.998158          R    = 0.995086          R    = 0.995702          R    = 0.998578
 1     TYPE =     2.00    2     TYPE =     2.00    3     TYPE =     2.00    4     TYPE =     2.00
       P OR TIME =  252.90        P OR TIME =  222.20        P OR TIME =  195.20        P OR TIME =  195.20
       FAILURES =     1.00        FAILURES =     1.00        FAILURES =     0.00        FAILURES =     0.00

R    = 0.990379
 5     TYPE =     2.00
       P OR TIME =  209.10
       FAILURES =     1.00

MODULE H  RELIABILITY = 0.990364                UNRELIABILITY = 0.009636


MODULE I
NUMBER OF COMPONENTS ................   6
NUMBER OF MINIMAL PATHS .............   4

THE   4 MINIMAL PATHS FOR MODULE I   FOLLOW:
<1,3,4,5>
<1,3,6>
<2,3,4,5>
<2,3,6>

                              SUBSYSTEM RELIABILITY EQUATION ..... MODULE I    ( 9 TERMS)

 R    = R R R R  + R R R  - R R R R R  + R R R R  - R R R R R  + R R R  + R R R R R R  - R R R R  - R R R R R
  I      1 3 4 5    1 3 6    1 3 4 5 6    2 3 4 5    1 2 3 4 5    2 3 6    1 2 3 4 5 6    1 2 3 6    2 3 4 5 6


MAPS-II: PROBABILITY COMPUTATION ROUTINE

COMPONENT RELIABILITIES FOR MODULE I

R    = 0.981406          R    = 0.990043          R    = 0.989489          R    = 0.982108
 1     TYPE =     2.00    2     TYPE =     2.00    3     TYPE =     2.00    4     TYPE =     2.00
       P OR TIME =  200.20        P OR TIME =  200.20        P OR TIME =  232.50        P OR TIME =  242.60
       FAILURES =     2.00        FAILURES =     2.00        FAILURES =     1.00        FAILURES =     2.00

R    = 0.985117          R    = 0.989208
 5     TYPE =     2.00    6     TYPE =     2.00
       P OR TIME =  250.20        P OR TIME =  198.10
       FAILURES =     2.00        FAILURES =     1.00

MODULE I  RELIABILITY = 0.988959                UNRELIABILITY = 0.011041


MODULE J
NUMBER OF COMPONENTS ................   4
NUMBER OF MINIMAL PATHS .............   4

THE   4 MINIMAL PATHS FOR MODULE J   FOLLOW:
<1,3>
<1,4>
<2,3>
<2,4>

                              SUBSYSTEM RELIABILITY EQUATION ..... MODULE J    ( 9 TERMS)

 R    = R R  + R R  - R R R  + R R  - R R R  + R R  + R R R R  - R R R  - R R R
  J      1 3    1 4    1 3 4    2 3    1 2 3    2 4    1 2 3 4    1 2 4    2 3 4


MAPS-II: PROBABILITY COMPUTATION ROUTINE


COMPONENT RELIABILITIES FOR MODULE J

R    = 0.990919          R    = 0.991159          R    = 0.976723          R    = 0.994587
```

```
 1    TYPE =     2.00        2    TYPE =     2.00        3    TYPE =     1.00        4    TYPE =     1.00
      P OR TIME =  199.90         P OR TIME =  202.40         P OR TIME =  249.00         P OR TIME =  269.00
      FAILURES =    1.00          FAILURES =    1.00          FAILURES =    2.00          FAILURES =    2.00

MODULE J  RELIABILITY = 0.999794               UNRELIABILITY = 0.000206


MODULE K
NUMBER OF COMPONENTS ................    6
NUMBER OF MINIMAL PATHS .............    4

THE   4 MINIMAL PATHS FOR MODULE K   FOLLOW:
<1,2,5>
<1,2,6>
<3,4,5>
<3,4,6>

                                  SUBSYSTEM RELIABILITY EQUATION ..... MODULE K    ( 9 TERMS)

  R    = R R R  + R R R  - R R R R  + R R R  - R R R R R  + R R R  + R R R R R R  - R R R R R  - R R R R
   K      1 2 5    1 2 6    1 2 5 6    3 4 5    1 2 3 4 5    3 4 6    1 2 3 4 5 6    1 2 3 4 6    3 4 5 6


MAPS-II: PROBABILITY COMPUTATION ROUTINE

COMPONENT RELIABILITIES FOR MODULE K

R    = 0.991581           R    = 0.981691           R    = 0.997981           R    = 0.999155
 1    TYPE =     2.00      2    TYPE =     2.00      3    TYPE =     2.00      4    TYPE =     2.00
      P OR TIME =  158.10       P OR TIME =  199.10       P OR TIME =  206.00       P OR TIME =  210.00
      FAILURES =    2.00        FAILURES =    1.00        FAILURES =    0.00        FAILURES =    0.00

R    = 0.992313           R    = 0.999391
 5    TYPE =     2.00      6    TYPE =     2.00
      P OR TIME =  210.00       P OR TIME =  215.10
      FAILURES =    0.00        FAILURES =    0.00

MODULE K  RELIABILITY = 0.999919               UNRELIABILITY = 0.000081


MODULE L
NUMBER OF COMPONENTS ................    5
NUMBER OF MINIMAL PATHS .............    4

THE   4 MINIMAL PATHS FOR MODULE L   FOLLOW:
<1,3,5>
<1,4,5>
<2,3,5>
<2,4,5>

                                  SUBSYSTEM RELIABILITY EQUATION ..... MODULE L    ( 9 TERMS)

  R    = R R R  + R R R  - R R R R  + R R R  - R R R R  + R R R  + R R R R R  - R R R R  - R R R R
   L      1 3 5    1 4 5    1 3 4 5    2 3 5    1 2 3 5    2 4 5    1 2 3 4 5    1 2 4 5    2 3 4 5


MAPS-II: PROBABILITY COMPUTATION ROUTINE

COMPONENT RELIABILITIES FOR MODULE L

R    = 0.997664           R    = 0.997453           R    = 0.992979           R    = 0.984631
 1    TYPE =     2.00      2    TYPE =     2.00      3    TYPE =     1.00      4    TYPE =     1.00
      P OR TIME =  182.60       P OR TIME =  189.60       P OR TIME =  249.00       P OR TIME =  251.00
      FAILURES =    0.00        FAILURES =    0.00        FAILURES =    2.00        FAILURES =    2.00

R    = 0.994559
 5    TYPE =     2.00
      P OR TIME =  260.90
      FAILURES =    1.00

MODULE L  RELIABILITY = 0.994445               UNRELIABILITY = 0.005555


MODULE M
NUMBER OF COMPONENTS ................    6
NUMBER OF MINIMAL PATHS .............    4

THE   4 MINIMAL PATHS FOR MODULE M   FOLLOW:
<1,3,4,5>
<1,3,6>
<2,3,4,5>
<2,3,6>
                                  SUBSYSTEM RELIABILITY EQUATION ..... MODULE M    ( 9 TERMS)

  R    = R R R R  + R R R  - R R R R R  + R R R R  - R R R R R  + R R R  + R R R R R R  - R R R R  - R R R R R
   M      1 3 4 5    1 3 6    1 3 4 5 6    2 3 4 5    1 2 3 4 5    2 3 6    1 2 3 4 5 6    1 2 3 6    2 3 4 5 6


MAPS-II: PROBABILITY COMPUTATION ROUTINE
```

```
COMPONENT RELIABILITIES FOR MODULE M

R    = 0.986934          R    = 0.987026          R    = 0.993753          R    = 0.995785
 1     TYPE =    2.00     2     TYPE =    2.00     3     TYPE =    2.00     4     TYPE =    1.00
       P OR TIME =  242.60        P OR TIME =  201.50        P OR TIME =  200.10        P OR TIME =  249.00
       FAILURES =    2.00         FAILURES =    1.00         FAILURES =    0.00         FAILURES =    2.00

R    = 0.982185          R    = 0.990429
 5     TYPE =    1.00     6     TYPE =    1.00
       P OR TIME =  250.00        P OR TIME =  240.00
       FAILURES =    2.00         FAILURES =    1.00

MODULE M  RELIABILITY = 0.993376                UNRELIABILITY = 0.006624


MODULE AND COMPONENT RELIABILITIES FOR THE SYSTEM

R    = 0.980056          R    = 0.976210          R    = 0.997187          R    = 1.000000
 A     TYPE =    0.00     B     TYPE =    0.00     C     TYPE =    0.00     D     TYPE =    0.00
       P OR TIME =    0.00        P OR TIME =    0.00        P OR TIME =    0.00        P OR TIME =    0.00
       FAILURES =    0.00         FAILURES =    0.00         FAILURES =    0.00         FAILURES =    0.00

R    = 0.999999          R    = 0.994709          R    = 0.999466          R    = 0.990364
 E     TYPE =    0.00     F     TYPE =    0.00     G     TYPE =    0.00     H     TYPE =    0.00
       P OR TIME =    0.00        P OR TIME =    0.00        P OR TIME =    0.00        P OR TIME =    0.00
       FAILURES =    0.00         FAILURES =    0.00         FAILURES =    0.00         FAILURES =    0.00

R    = 0.988959          R    = 0.999794          R    = 0.999919          R    = 0.994445
 I     TYPE =    0.00     J     TYPE =    0.00     K     TYPE =    0.00     L     TYPE =    0.00
       P OR TIME =    0.00        P OR TIME =    0.00        P OR TIME =    0.00        P OR TIME =    0.00
       FAILURES =    0.00         FAILURES =    0.00         FAILURES =    0.00         FAILURES =    0.00

R    = 0.993376          R    = 0.999111          R    = 0.997819
 M     TYPE =    0.00     1     TYPE =    2.00     2     TYPE =    2.00
       P OR TIME =    0.00        P OR TIME =  318.50        P OR TIME =  256.10
       FAILURES =    0.00         FAILURES =    0.00         FAILURES =    0.00

 SYSTEM RELIABILITY = 0.914624                UNRELIABILITY = 0.085376
                      0.838367                                0.161633
                      0.876300                                0.123700
                      0.877376                                0.122624
                      0.884099                                0.115901
                      0.885395                                0.114605
                      0.885404                                0.114596
                      0.885706                                0.114294
                      0.889522                                0.110478
                      0.889554                                0.110446
                      0.893957                                0.106043
                      0.894438                                0.105562
                      0.896596                                0.103404
                      0.897614                                0.102386
                      0.900380                                0.099620
                      0.901833                                0.098167
                      0.904009                                0.095991
                      0.911390                                0.088610
                      0.911959                                0.088041
                      0.914624                                0.085376
                      0.928642                                0.071358



    THE MEAN RELIABILITY IS 0.893357     VARIANCE = 0.000327      STANDARD DEVIATION = 0.018076
    THE ESTIMATED RELIABILITY FOR THE SYSTEM IS  0.895148

    THE MISSION TIME IS     90.00  DAYS
    THE ESTIMATED MTBF IS  7.98091965E+02


                          RELIABILITY              MTBF
  PERCENTILE              PERCENTILE            PERCENTILE
                            POINTS                POINTS
   5 PERCENT               0.838367          5.10497067E+02  DAYS
  10 PERCENT               0.876300          6.81574433E+02  DAYS
  20 PERCENT               0.884099          7.30601605E+02  DAYS
  25 PERCENT               0.885395          7.39393492E+02  DAYS
  50 PERCENT               0.893957          8.02867758E+02  DAYS
  75 PERCENT               0.901833          8.71027822E+02  DAYS
  80 PERCENT               0.904009          8.91829229E+02  DAYS
  90 PERCENT               0.911959          9.76554280E+02  DAYS
  95 PERCENT               0.914624          1.00849020E+03  DAYS
97.5 PERCENT               0.921633          1.10282636E+03  DAYS
  99 PERCENT               0.925838          1.16798142E+03  DAYS

                          FREQUENCY AND CUMULATIVE FREQUENCY COUNTS OF CASES

   0.8350    0.8400    0.8450    0.8500    0.8550    0.8600    0.8650    0.8700    0.8750    0.8800
        0         1         0         0         0         0         0         0         0         2
        0         1         1         1         1         1         1         1         1         3

   0.8850    0.8900    0.8950    0.9000    0.9050    0.9100    0.9150    0.9200    0.9250    0.9300
        1         5         2         2         3         0         3         0         0         1
        4         9        11        13        16        16        19        19        19        20
```

2

VITA

John Wayne Cooley

Candidate for the Degree of

Doctor of Philosophy

Thesis: SIMULATION PROGRAM FOR ASSESSING THE RELIABILITIES OF COMPLEX SYSTEMS (SPARCS)

Major Field: Business Administration

Biographical:

Personal Data: Born in Lake Charles, Louisiana, January 21, 1947, the son of Mr. and Mrs. R. J. Cooley.

Educational: Graduated from Sulphur High School, Sulphur, Louisiana, in May, 1965; received Bachelor of Science degree in Accounting from McNeese State University in 1968; received Master of Business Administration degree from Lamar State University in 1970; enrolled in doctoral program at Oklahoma State University, 1972-75; completed requirements for the Doctor of Philosophy degree at Oklahoma State University in May, 1976.

Professional Experience: Data processing assistant, 1966-68; junior accountant for Theriot, Milford and Dunn, CPA's, 1968-69; Internal Auditor for First National Bank of Lake Charles, 1969; teaching fellow at Lamar State University, 1969-70; instructor at the Stephen F. Austin State University in Nacogdoches, Texas, 1970-72; instructor at Oklahoma State University in Stillwater, Oklahoma, 1972-75; assistant professor at the University of Nebraska at Omaha, 1975-present.