

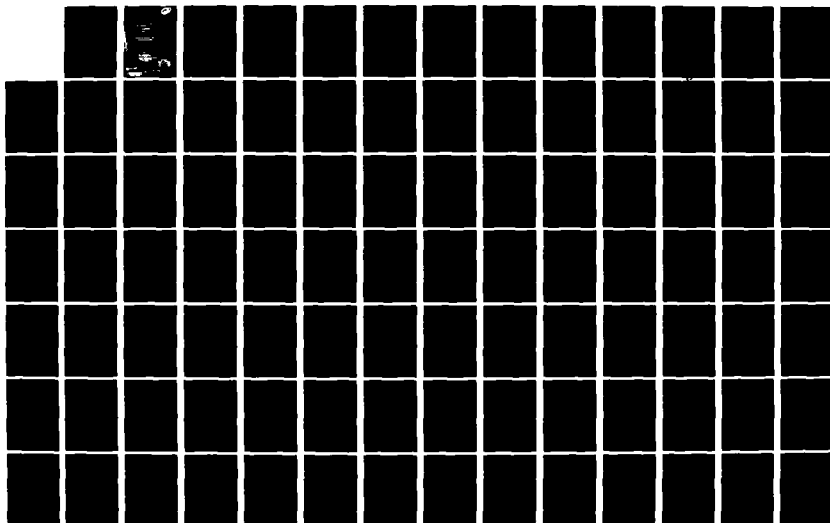
AD-A121 030

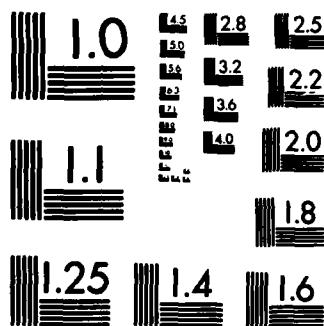
DEVELOPMENT OF A WAR READINESS SPARES KIT MODEL AND
SOLUTION TECHNIQUE(U) OKLAHOMA STATE UNIV STILLWATER
SCHOOL OF INDUSTRIAL ENGINEERI. M P TERRELL ET AL.
31 DEC 80 F33600-80-C-0423 F/G 9/2

172

UNCLASSIFIED

NL





MICROCOPY RESOLUTION TEST CHART
NATIONAL BUREAU OF STANDARDS-1963-A

ADA 121030

PREFACE

→ This final report describes work completed under Research Number F33600-80-C-423 during the period July 1980 through December 1980. This effort was performed for the United States Air Force Logistics Command at Wright-Patterson Air Force Base, Ohio.

The work was performed by the School of Industrial Engineering and Management at Oklahoma State University. Dr. M. Palmer Terrell and Dr. Philip M. Wolfe were co-principle researchers, assisted by Ph.D. students, Mr. Umit Yuceer and Mr. Shawn Yu.

This effort was undertaken to develop a new model and approach for solving the War Readiness Spares Kit (WRSK) problem. The development and evaluation of a new model and computer program, referred to as the Greedy Algorithm, is described herein.

The final report is divided into six chapters and four appendices. ←

Accession For	
NTIS GRA&I	☒
DTIC TAB	☐
Unannounced	☐
Justification	
By <u>Per Ltr. on file</u>	
Distribution/ <u>FL-88, Acq. R82-2109</u>	
Availability Codes <u>dlb 28 Sep '82</u>	
Dist	Avail and/or Special
<u>A</u>	

DTIC
COPY
INSPECTED
2

TABLE OF CONTENTS

Page No.	
PART I - Introduction	
Problem Statement	1.1
Brief Description of Current DO29 WRSK Model.	1.2
Problems With the Current Solution Methodology.	1.2
PART II - Objectives of this Research	
General	2.1
Objectives.	2.1
PART III - Measures of Supply Performance	
Introduction.	3.1
Mathematical Development of Performance Measures.	3.1
Mathematical Properties of Performance Measures	3.3
Expected Value Calculations	3.7
PART IV - Mathematical Modelling and Solution Methods for the WRSK Problem	
A Multiobjective Approach to the WRSK Problem	4.1
Potential Solution Methods.	4.5
Example Problem Solutions	4.13
Selection of Solution Method.	4.21
The Current DO29 WRSK Model	4.22
PART V - A Summary of Computational Experience	
Introduction.	5.1
Parameters b_j and m	5.1
Articulation ^j of the Weight w	5.4
Comparison of the Solutions	5.10
PART VI - Summary of Results	
REFERENCES	R.1
APPENDIX A	
A.1 Computer Code for the Primal Search Algorithm.	A.1.1
A.2 Computer Code for the Modified Algorithm of Brooks, et al.	A.2.1
A.3 Computer Code for the Saddle Point Search Algorithm	A.3.1

APPENDIX B

B.1	Detail Documentation of Greedy Algorithm	
	Computer Program	B.1.1
B.2	FORTTRAN Listing of Greedy Algorithm and Solution	
	Output	B.2.1

APPENDIX C

C.1	Notes on Conversion of the D029 Computer Program to	
	an IBM Compiler	C.1.1
C.2	FORTTRAN Listing of the IBM Version of the D029	
	Algorithm With a Solution Output	C.2.1

APPENDIX D

D.1	Listing of F4D Input Data	D.1.1
D.2	Listing of Phase I & Phase II F4D Data	D.2.1

LIST OF TABLES

Table		Page
4.1	Costs and Demand Rates for the Sample WRSK Problem . . .	4.13
5.1	Comparison of b_j Values	5.3
5.2	Objective Function $\Psi(w)$ vs. Weighting Factor W : Phase 1 Solutions for a Budget of \$3,635,906	5.6
5.3	Objective Function $\Psi(w)$ vs. Weighting Factor w : Phase 2 Solutions With a Budget of $\$7.5 \times 10^6$	5.7
5.4	Objective Function $\Psi(w)$ vs. Weighting Factor w : Phase 2 Solutions With a Budget of $\$9.0 \times 10^6$	5.7
5.5	Frequency of Differences of Kits in Phase 1	5.11
5.6	Frequency of Differences of Kits in Phase 2	5.11

LIST OF FIGURES

Figure		Page
4.1	Performance Measures vs. Various Investment Levels	4.3
4.2	Compromise Solution for a Given Weighting Factor w	4.4
4.3	Search in the Positive Quadrant	4.7
4.4	Duality Gap	4.20
5.1	Plot of $\tilde{\Psi}(w)$ Exhibited in Table 5.2	5.6
5.2	Plot of $E(\text{NORS}/X)$ vs. $E(\text{SDO}/X)$ From Table 5.2	5.8
5.3	Plot of $E(\text{SDO}/X)$ vs. $E(\text{NORS}/X)$ From Table 5.3 and 5.4 . .	5.9
5.4a	Greedy Algorithm Program-A Phase 1 Solution for Kit 5718	5.15
5.4b	Greedy Algorithm Program-A Phase 2 Solution: Budget = \$7,108,451	5.16
5.5a	IBM Version of D029 Program-A Phase 1 Solution for Kit 5718 Budget	5.17
5.5b	IBM Version of D029 Program-A Phase 2 Solution for Kit 5718 Budget	5.18
5.6a	D029-CDC Program-A Phase 1 Solution for Kit 5718 Budget	5.19
5.6b	D029 CDC Program-A Phase 2 Solution for Kit 5718 Budget	5.20
5.7a	Phase 1 Greedy Algorithm Solution: Budget of Kit 5718	5.21
5.7b	Phase 2 Greedy Algorithm Solution: Budget of \$7,562,983	5.22
5.8a	Phase 1 Greedy Algorithm Solution: Budget of Kit 5718	5.23
5.8b	Phase 2 Greedy Algorithm Solution: Budget of \$7,075,000	5.24

PART I

INTRODUCTION

Problem Statement

The management of spare parts inventories is indispensable to the logistics operations of the Air Force. Effective inventory management systems are of great importance to the Air Force since a large amount of money is spent annually on purchases of spare parts. Two systems 'METRIC and MOD-METRIC' have been implemented to assist in the management of recoverable spare parts. However, some of the logistics problems need specialized and sophisticated solution strategies and methods.

A special type of inventory is the war readiness spares kit (WRSK). This consists of a group of spare parts that are deployed with a flying unit. A WRSK system is in operation and assists in determining the spare parts that should be in a kit. The objective of this system is to determine the contents of a kit which will result in a high level of performance at a minimum cost.

An important aspect of this problem is the definition of measures of supply performance which are operationally meaningful. A thorough investigation of the problem and the underlining assumptions suggests that the NORS (Not Operationally Ready-Supply) and SDO (Stock Due Out) measures can be used to describe the performance of the kit. (See references [3] and [6].) The NORS function is non-linear and nonseparable over the nonnegative integers. This makes the problem difficult to solve. In addition, the number of items in the kit is very large in practice. Consequently, a large size non-linear integer programming needs to be solved. Dynamic programming [1] or the branch-

and-bound method [5] could be employed to solve the WRSK problem, but the computation time required is impractical.

Brief Description of Current D029 WRSK Model

The current D029 Model and its computer program is designed to solve this problem. It has two passes. The first pass tries to determine a kit which will yield a high performance for the initial six day period for a squadron of aircrafts. The second pass attempts to find the contents of the whole kit for the thirty day period. A detailed description is provided by Wright Patterson Air Force Base Logistics Command in [8].

This model employs marginal analysis to obtain a kit which will perform better than the requirement kit 57-18. The tradeoff between NORS and SDO is handled by manipulating the weight between them at each step according to some heuristic rules. In addition, at each step of the algorithm some number of units of an item are added to the kit. This number is also manipulated according to some heuristics. This is necessary because the current D029 Model uses marginal improvements which must be recalculated after one unit of an item is added to the kit due to the nonseparability of the expected NORS function, $E(NORS/X)$.

The current D029 computer program has some special subroutines to handle input, word packing and unpacking, and calculation and interpolation of probabilities. One of the subroutines provides intermediate kits at various budget levels.

Problems With Current Solution Methodology

The marginal analysis as currently used will not guarantee an optimum solution. Fox [4] states conditions when discrete marginal analysis will

result in an optimum answer. One criterion is that the functions must be separable. The model used in the WRSK system involves functions which are not separable. Consequently, the marginal analysis approach as applied to the WRSK must be categorized as a heuristic.

Another problem involves the calculation of $E(NORS/X)$ and $E(SDO/X)$ functions. These expectation formulas are truncated to ease the computations. Truncation may cause numerical errors if the upper bound of the truncation is not chosen suitably.

Relative to the current WRSK system, models involving large flying units (greater than 24 aircraft) require a large amount of memory. So much so, that portions of the resident software must be removed to provide the required storage. In order to avoid the use of excessive storage, word packing and unpacking is employed. This however creates some numerical errors throughout the computations. Some of the probability values are stored and others are interpolated when needed. This causes some probabilities to be underestimated. More importantly, this process is very time consuming. Approximately half of the execution time is spent on packing and unpacking.

PART II

OBJECTIVES OF THIS RESEARCH

General

The objective of this research has been to develop, validate, and document an improved War Readiness Spares Kit (WRSK) model and solution procedure which may be used by the United States Air Force in the management of recoverable spare parts.

This research effort has been achieved by accomplishing a series of subobjectives. These objectives are stated below, and reference is made to the pertinent parts of this report.

Objectives

Objective 1: Develop and validate an improved solution procedure for the current WRSK model.

Development and validation of such a solution method was accomplished in two stages. In the first stage, measures of supply performance and their mathematical properties were investigated. Part III of this report is devoted to the results of this investigation. The second stage was the development of a mathematical model which is reported in Part IV where a multiobjective approach is proposed for the WRSK problem. Potential solution methods were then investigated. A small example problem was solved by these algorithms in order to demonstrate how they generate solutions and how they converge to an optimum. Evaluation of these methods led to the selection of a preferred solution method which was investigated and tested using F4D data. The final section of Part IV discusses the current D029 model which has features pertinent to this research.

Objective 2: Develop procedures which will eliminate the problems caused by using expected value approximations.

Part III considers the measures of supply performance which are closely related to the stochastic behavior of a WRSK kit. Accurate expected value computations are of vital importance to the generation of high quality solutions. Truncation of an infinite sum is compared to the approximation of the sum. Proper use of the truncation procedure is explained by some examples in the last section of Part III.

Objective 3: Implement the algorithm to solve the War Readiness Spares Kit problem.

A major aspect of this research has been to implement the proposed solution method and use it to solve a WRSK problem with real data. The implementation and computational features using F4D data is discussed in Part V.

Objective 4: Compare the performance of the algorithm with the current system in terms of time and computational feasibility.

In order to establish a base for meaningful comparison, the current D029 computer program was modified for compatibility with IBM compilers. Then, performance of the two algorithms was observed in terms of time and computational feasibility. These results are discussed in Part V.

Objective 5: Document results and computer programs.

This report, in its totality, documents the research results. The detailed documentation and FORTRAN listing of the proposed solution method is given in Appendix B.1 and B.2. In addition, the FORTRAN listing of the IBM modified D029 program is provided in Appendix C.1 and C.2.

PART III

MEASURES OF SUPPLY PERFORMANCE

Introduction

This section focuses on measures of supply performance. The stochastic prediction of the behavior of a kit during the mission is of vital importance in the study of the WRSK problem. Another important aspect, when measuring the performance of a kit, is the relationship of the performance measures to the operations.

The performance of a kit can be measured in several ways; fill rate, operational rate, average back orders, average number of Not Operationally Ready Supply Aircraft, and kit cost. These can be described briefly as follows. Fill rate is the ratio of the number of parts issued over a fixed time period to the number of parts demanded over the same period. Operational rate is the probability that there will be no due outs from the base supply at any time. Back orders contain the number of stock due-outs (SDO) from base supply during a fixed period. Finally, NORS is the number of grounded aircraft due to lack of spare parts. Experience has revealed that average SDO, average NORS, and the cost of a kit relate to operations better than the other measures.

Mathematical Development of Performance Measures

The following definitions and assumptions are made in order to develop the mathematical expressions for average back orders and NORS.

- (1) X represents a kit consisting of individual items i of quantity x_i .
- (2) n items are under consideration for the kit.

- (3) The demand of an item is a Poisson distributed random variable.
- (4) The failure of any item is independent of the failure of any other item.
- (5) When a demand for an item can not be satisfied from the stock on hand, the required part is consolidated by cannibilization.

Let $F_1(\cdot)$ denote the cumulative Poisson sums for the item i . The operational rate given no more than k planes for cannibilization is expressed by the statement

$$b_k = P[\text{NORS} \leq k/X] = \prod_{i=1}^n F_i(x_i + ka_i), \quad k = 0, 1, 2, \dots, N$$

where a_i is the number of applications of the part i on an airplane, and N is the total number of aircraft in the flying unit. Consequently

$$\Delta b_k = P[\# \text{ of NORS} = k] \quad \text{for } k = 0, 1, 2, \dots, N$$

then $\Delta b_0 = P[\# \text{ of NORS} = 0] = b_0$

and $\Delta b_k = P[\# \text{ of NORS} \leq k] - P[\# \text{ of NORS} \leq k - 1]$
 $= b_k - b_{k-1} \quad \text{for } k = 1, 2, 3, \dots, N$

Finally $E(\text{NORS}/X) = \sum_{k=1}^N (b_k - b_{k-1})$

An equivalent development of the expectation of the NORS function for a given kit X goes as follows.

$$E(\text{NORS}/X) = \sum_{k=0}^{\infty} k P[\# \text{ of NORS} = k]$$

$$E(\text{NORS}/X) = \sum_{k=0}^{\infty} (1 - P[\# \text{ of NORS} \leq k])$$

$$E(\text{NORS}/X) = \sum_{k=0}^{\infty} \left(1 - \prod_{i=1}^n F_i(x_i + ka_i) \right)$$

An approximation to the above function for computational purposes was proposed by B. L. Miller in (6) as follows.

$$E(\text{NORS}/X) \approx \sum_{k=0}^m (1 - \frac{n}{n} F_1(x_1 + ka_1)) + \sum_{k=m+1}^{\infty} \sum_{i=1}^n 1 - F_i(x_1 + ka_i)$$

where m is such that $\sum_{i=1}^n F_i(x_1 + ka_i)$ is very close to 1 for all $k > m$. See Figure 3.1.

The expected back orders, SDO, for a given kit X is obtained by

$$E(\text{SDO}/X) = \sum_{i=1}^n \sum_{j=0}^{\infty} (1 - F_i(x_i + j))$$

Reference may be made to [3] and [8] for further details.

If item i costs $\$c_i$, the cost of the kit is then $\sum_{i=1}^n c_i x_i$ or in vector form, CX .

Mathematical Properties of Performance Measures

A few properties of the measures, $E(\text{NORS}/X)$ and $E(\text{SDO}/X)$, are presented below. They will be used later in the development of solution methods and algorithms.

Property 1

Both $E(\text{SDO}/X)$ and $E(\text{NORS}/X)$ are monotonically decreasing because the first differences are nonpositive.

Define $\Delta_k E(\text{SDO}/X) = E(\text{SDO}/X + e_k) - E(\text{SDO}/X)$ where e_k represents the k th unit vector. Then

$$\begin{aligned} \Delta_k E(\text{SDO}/X) &= \sum_{i=1, i \neq k}^n \sum_{j=0}^{\infty} [1 - F_i(x_i + j)] + \sum_{j=0}^{\infty} [1 - F_k(x_k + 1 + j)] - \sum_{i=1}^n \sum_{j=0}^{\infty} [1 - F_i(x_i + j)] \\ &= \sum_{j=1}^{\infty} [1 - F_k(x_k + j)] - \sum_{j=0}^{\infty} [1 - F_k(x_k + j)] \text{ which yields} \end{aligned}$$

$\Delta_k E(\text{SDO}/X) = -[1 - F_k(x_k)]$, the marginal improvement in average SDO

when a one unit of item k is added to the kit.

Likewise, the marginal improvement in average NORS is calculated as follows.

Define $\Delta_k E(\text{NORS}/X) = E(\text{NORS}/X + e_k) - E(\text{NORS}/X)$. Then

$$\begin{aligned}\Delta_k E(\text{NORS}/X) &= \sum_{j=0}^{\infty} \left[1 - \prod_{i=1}^n F_i(x_i + ja_i) \right] P_k(d_k + ja_k + 1) - \sum_{j=0}^{\infty} \left[1 - \prod_{i=1}^n F_i(x_i + ja_i) \right] P_k(x_k + ja_k) \\&= \sum_{j=0}^{\infty} \left(\left[\prod_{i=1}^n F_i(x_i + ja_i) \right] [P_k(x_k + ja_k) - P_k(x_k + ja_k + 1)] \right) \\&= - \sum_{j=0}^{\infty} P_k(x_k + ja_k + 1) \prod_{i=1, i \neq k}^n F_i(x_i + ja_i) \\&\text{where } P_k(s) = \frac{e^{-\mu_k} \mu_k^s}{s!}\end{aligned}$$

Property 2

$E(\text{SDO}/X) \geq E(\text{NORS}/X)$ for any X

Proof:

$$\begin{aligned}E(\text{SDO}/D) &= \sum_{j=1}^m \sum_{i=1}^n [1 - F_i(x_i + j)] + \sum_{j=m+1}^{\infty} \sum_{i=1}^n [1 - F_i(x_i + j)] \\E(\text{NORS}/X) &= \sum_{j=0}^{\infty} \left[1 - \prod_{i=1}^n F_i(x_i + ja_i) \right] + \sum_{j=m+1}^{\infty} \sum_{i=1}^n [1 - F_i(x_i + ja_i)]\end{aligned}$$

Define $DIF = E(SDQ/X) - E(NORS/X)$

$$= \sum_{j=0}^{\infty} \left\{ \sum_{i=1}^n [1 - F_1(x_1 + j)] - 1 + \sum_{i=1}^n F_1(x_1 + ja_i) \right\} + \sum_{j=m+1}^{\infty} \sum_{i=1}^n [F_1(x_1 + ja_i) - F_1(x_1 + j)]$$

$$= \sum_{j=0}^m \left\{ n - 1 - \sum_{i=1}^n F_1(x_1 + j) + \sum_{i=1}^n F_1(x_1 + ja_i) \right\} + \sum_{j=m+1}^{\infty} \sum_{i=1}^n [F_1(x_1 + ja_i) - F_1(x_1 + j)]$$

Since $a_i \geq 1$ for all i , then $1 \geq F_1(x_1 + ja_i) \geq F_1(x_1 + j)$ and

$$\sum_{i=1}^n F_1(x_1 + ja_i) \geq \sum_{i=1}^n F_1(x_1 + j), \text{ further}$$

$$\sum_{i=1}^n F_1(x_1 + j) - \sum_{i=1}^n F_1(x_1 + ja_i) \leq \sum_{i=1}^n F_1(x_1 + j) - \sum_{i=1}^n F_1(x_1 + j).$$

Let $\Psi(Y) = \sum_{i=1}^n y_i - \prod_{i=1}^n y_i$. The maximum of this function is $(n-1)$. The proof

goes as follows:

$$\nabla \Psi = (1 - \prod_{i=2}^n y_i, \dots, 1 - \prod_{i \neq k}^n y_i, \dots, 1 - \prod_{i=1}^{n-1} y_i)$$

$$\nabla^2 \Psi = \begin{bmatrix} 0 & a_{12} & \dots & a_{1n} \\ a_{21} & 0 & \dots & a_{2n} \\ \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot \\ a_{n1} & a_{n2} & \dots & a_{nn} \end{bmatrix} \quad \text{where } a_{kj} = - \prod_{\substack{i \neq k \\ i \neq j}}^n y_i$$

and $|a_{kj}| \leq 1$

Clearly $r^t \nabla^2 \Psi r < 0 \quad r \in R_+^n$. Hence Ψ is concave in the positive unit hypercube. Furthermore, $\nabla \Psi(y) = 0$ yields the solution that $y_i^* = 1$ for all $i=1, \dots, n$. Consequently $\Psi(i) = n-1$ is the maximum of the function.

Finally $DIF \geq 0$, since $(n-1) - [\sum_{i=1}^n F_1(x_1 + j) - \sum_{i=1}^n F_1(x_1 + ja_i)] \geq 0$

and

$$\sum_{i=1}^n [F_1(x_1 + ja_i) - F_1(x_1 + j)] \geq 0 \quad \text{for all } j \geq 0.$$

Property 3

The $E(\text{NORS}/X)$ function can be approximated by a separable function.

This is accomplished as follows (also see [3]).

$$E(\text{NORS}/X) \simeq (m+1) + \sum_{i=1}^n \sum_{k=0}^m b_k \log F_i(x_i + ka_i) - \sum_{k=m+1}^{\infty} [1 - F_i(x_i + ka_i)]$$

where $b_k = \prod_{i=1}^n F_i(\bar{x}_i + ka_i)$ for some $\bar{x} = (\bar{x}_1, \bar{x}_2, \dots, \bar{x}_n)$.

where n is such that $\prod_{i=1}^n F_i(x_i + ka_i)$ is very close to 1 for all $k > m$.

The term $\sum_{k=0}^m [1 - \prod_{i=1}^n F_i(x_i + ka_i)]$ can be rewritten as $(m+1) - \sum_{k=0}^m \prod_{i=1}^n F_i(x_i + ka_i)$.

The product $\prod_{i=1}^n F_i(x_i + ka_i)$ can be replaced by some multiple of its logarithm

in the following manner.

$$-b \sum_{i=1}^n \log F_i(x_i + ka_i) \text{ where } b_k \text{ is a parameter defined by } b_k = \prod_{i=1}^n F_i(\bar{x}_i + ka_i)$$

for some $\bar{x} = (\bar{x}_1, \bar{x}_2, \dots, \bar{x}_n)$.

Finally then, separability is accomplished as $E(\text{NORS}/X) = (m+1) +$

$$\sum_{i=1}^n \sum_{k=0}^m b_k \log F_i(x_i + ka_i) + \sum_{k=m+1}^{\infty} [1 - F_i(x_i + ka_i)].$$

This results plays an important role in the development of an algorithm which is discussed in Section IV.

Expected Value Calculations

Calculation of the expected NORS function for a given kit X ; $E(\text{NORS}/X) = \sum_{j=1}^{\infty} (1 - \prod_{i=1}^n F_i(x_i + ja_i))$ creates computational difficulties. This expectation is an infinite sum. Since it is convergent, the terms become negligibly small for sufficiently large j 's. There are two ways to handle the computation. One way is to truncate this infinite sum for a sufficiently large value M . The second way is to develop an approximation as suggested by B. L. Miller in [6]. Physically the number NORS airplanes can not exceed the number of airplanes in the squadron. This then establishes a natural upper bound for truncation. Both truncated and approximate expected NORS formulas are provided below, and the difference between them is evaluated.

a) Truncation:
$$f_T(X) = \sum_{j=0}^M (1 - \prod_{i=1}^n F_i(x_i + ja_i))$$

b) Approximation:
$$f_A(X) = \sum_{j=0}^m (1 - \prod_{i=1}^n F_i(x_i + ja_i)) + \sum_{j=m+1}^{\infty} \prod_{i=1}^n (1 - F_i(x_i + ja_i))$$

where M is the number of planes in the squadron, a_i is the number of applications of the part i on an airplane, and m is such that $\prod_{i=1}^n F_i(x_i + ja_i)$ is very close to 1 for all $j > m$.

The difference is given by the relation $\Delta = f_A(X) - f_T(X)$. If $m > M$, then

$$\Delta = \sum_{j=M+1}^m (1 - \prod_{i=1}^n F_i(x_i + ja_i)) + \sum_{j=m+1}^{\infty} \prod_{i=1}^n (1 - F_i(x_i + ja_i)).$$

If $m \leq M$, then
$$\Delta = \sum_{j=m+1}^{\infty} \prod_{i=1}^n (1 - F_i(x_i + ja_i)) - \sum_{j=m+1}^M (1 - \prod_{i=1}^n F_i(x_i + ja_i))$$

$$\text{or } \Delta = \sum_{j=m+1}^{\infty} \prod_{i=1}^n (1 - F_i(x_i + ja_i)) - \sum_{j=m+1}^M \prod_{i=1}^n (1 - F_i(x_i + ja_i))$$

because the approximation is valid for all $j > m$. Finally

$$\Delta = \sum_{j=M+1}^{\infty} \sum_{i=1}^n (1 - F_i(x_i + ja_i))$$

In either case, Δ is a positive term and Δ may not be negligibly small especially for large squadrons with some items having heavy demand rates.

The following examples illustrate the magnitude of the difference between the truncated and approximated formulas.

Example 1

Suppose that there are three planes in a squadron, and there are two essential items A, B with demand rates $\mu_A = 5$ and $\mu_B = 5.5$. The initial stock levels are determined to be $x_A = 5$ and $x_B = 6$. For simplicity, $a_A = a_B = 1$.

a) Calculation of NORS by $f_T(X)$

<u>j</u>	<u>$F_A(5+j)$</u>	<u>$F_B(6+j)$</u>	<u>Product</u>	<u>1 - Product</u>
0	.616	.686	.42258	.57742
1	.762	.809	.61646	.38354
2	.867	.894	.77510	.22490

$$f_T(x) = 1.18586$$

b) Calculation of NORS by $f_A(X)$

m is chosen as 6

<u>j</u>	<u>$F_A(5+j)$</u>	<u>$F_B(6+j)$</u>	<u>Product</u>	<u>1 - Product</u>
0	.616	.686	.42258	.57742
1	.762	.809	.61646	.38354
2	.867	.894	.77510	.22490
3	.932	.946	.88167	.11833
4	.968	.975	.94380	.05620
5	.986	.989	.97515	.02485
6	.995	.996	.99102	.00898

$$\text{Sum } 1.39422$$

<u>j</u>	<u>1 - F_A(5+j)</u>	<u>1 - F_B(6+j)</u>
7	.00202	.00160
8	.00070	.00060
9	.00022	.00020
10	.00007	.00006
11	.00002	.00001
Sum .00303		.00247 TSUM = .00550
f _A (X) = 1.39972		
Δ = 0.21385		

Example 2

Suppose there are three items A, B and C with $\mu_c = 4.5$ and $x_c = 5$
 $m = 6$ again.

a) Calculation by truncated formula

<u>j</u>	<u>F_A(5+j)</u>	<u>F_B(6+j)</u>	<u>F_C(5+j)</u>	<u>Product</u>	<u>1 - Product</u>
0	.616	.686	.720	.30426	.69574
1	.762	.809	.884	.52029	.47971
2	.867	.894	.921	.71387	.28613
f _T (X) = 1.46158					

b) Calculation by Approximated formula f_A(X)

<u>j</u>	<u>F_A(5+j)</u>	<u>F_B(6+j)</u>	<u>F_C(5+j)</u>	<u>Product</u>	<u>1 - Product</u>
0	.616	.686	.720	.30426	.69574
1	.762	.809	.844	.52029	.47971
2	.867	.894	.921	.71387	.28613
3	.932	.946	.964	.84993	.15007
4	.968	.975	.985	.92964	.07036
5	.986	.989	.994	.96930	.03070
6	.995	.996	.998	.98904	.01096
Sum = 1.72367					

<u>1</u>	<u>1 - F_A(5+j)</u>	<u>1 - F_B(6+j)</u>	<u>1 - F_C(5+j)</u>	
7	.00202	.00169	.00066	
8	.00070	.00060	.00020	
9	.00022	.00020	.00006	
10	.00007	.00006	.00002	
11	.00002	.00001	---	
			.00094	TSUM = .00644

$$f_A(X) = 1.73011$$

$$\Delta = .26853$$

These small examples show that there is an error which is not negligible. Even if the number of airplanes in the squadron is sufficiently large, one has to be careful to reduce the magnitude of this error. In addition, there will be round-off errors for large size problems even with double-precision arithmetic. It is essential to know the origin of the errors if one wishes to reduce the magnitude of the errors.

PART IV

MATHEMATICAL MODELLING AND SOLUTION METHODS FOR THE WRSK PROBLEM

A Multiobjective Approach to the WRSK Problem

In simple terms, the objective of the WRSK problem is to determine the quantity of each part in the kit which will minimize the average NORS, average SDO, and the cost of the kit simultaneously. All of these three objectives conflict with each other. If cost is minimized, then the other two measures increase. If average NORS is minimized for a fixed cost, the average SDO turns out very high. Conversely, the average NORS comes out high when the average SDO is minimized for a fixed budget level. The relationship between average NORS, SDO, and the cost was discussed in [3] and [8]. The WRSK problem is a multiobjective programming problem, and the model is stated mathematically as

$$\begin{array}{ll}\text{Min} & \{E(\text{NORS}/X), E(\text{SDO}/X), CX\} \\ \text{s.t.} & X \geq 0, \text{ integer}\end{array}$$

This model is also operationally meaningful, since all of these measures highly relate to operations. $E(\text{NORS}/X)$ measures the expected number of cannibalized airplanes. $E(\text{SDO}/X)$ is an indicator of the additional work load for repairs, and CX gives the cost of the kit.

The following solution strategy is proposed for solving this multi-objective programming problem. First of all, the problem can be considered as a parametric programming in terms of budget levels 'b' as shown below.

$$\begin{array}{ll}\text{Min} & (E(\text{NORS}/X), E(\text{SDO}/X)) \\ \text{s.t.} & CX \leq b \\ & X \geq 0, \text{ integer.}\end{array}$$

This reduced problem may be further investigated as two single-objective problems as follows:

$$\begin{aligned} (1) \quad & \text{Min} \quad E(\text{NORS}/X) \\ & \text{s.t.} \quad CX \leq b \\ & \quad \quad X \geq 0, \text{ integer} \end{aligned}$$

and

$$\begin{aligned} (2) \quad & \text{Min} \quad E(\text{SDO}/X) \\ & \text{s.t.} \quad CX \leq b \\ & \quad \quad X \geq 0, \text{ integer} \end{aligned}$$

Obviously the optimal solution of (1) is not necessarily optimal to (2), and conversely. If $\bar{X}^*$ solves (1) then $E(\text{SDO}/\bar{X}^*)$ turns out to be very high, and conversely if $\tilde{X}^*$ solves (2), then $E(\text{NORS}/\tilde{X}^*)$ is very high. This phenomenon has also been observed by WPAF Logistic Command and was stated explicitly in [8], the Preliminary Evaluation of D029 WRSK Model. Since $E(\text{SDO}/X) \geq E(\text{NORS}/X)$ for a given kit X , a best compromise solution between the $E(\text{NORS}/X)$ and $E(\text{SDO}/X)$ is sought for a specified budget level. Figure 4.1 illustrates the relation between $E(\text{SDO}/X)$ and $E(\text{NORS}/X)$ for specified budget levels.

Problems (1) and (2) can be merged into one problem (3) by the use of a weight w as follows.

$$\begin{aligned} (3) \quad & \text{Min} \quad \{E(\text{NORS}/X) + w E(\text{SDO}/X)\} \\ & \quad \quad CX \leq b \\ & \quad \quad X \geq 0, \text{ integer} \end{aligned}$$

The weight w expresses the quantity of NORS planes equivalent to each SDO. Prior assessment of the weight is required to obtain the best compromise

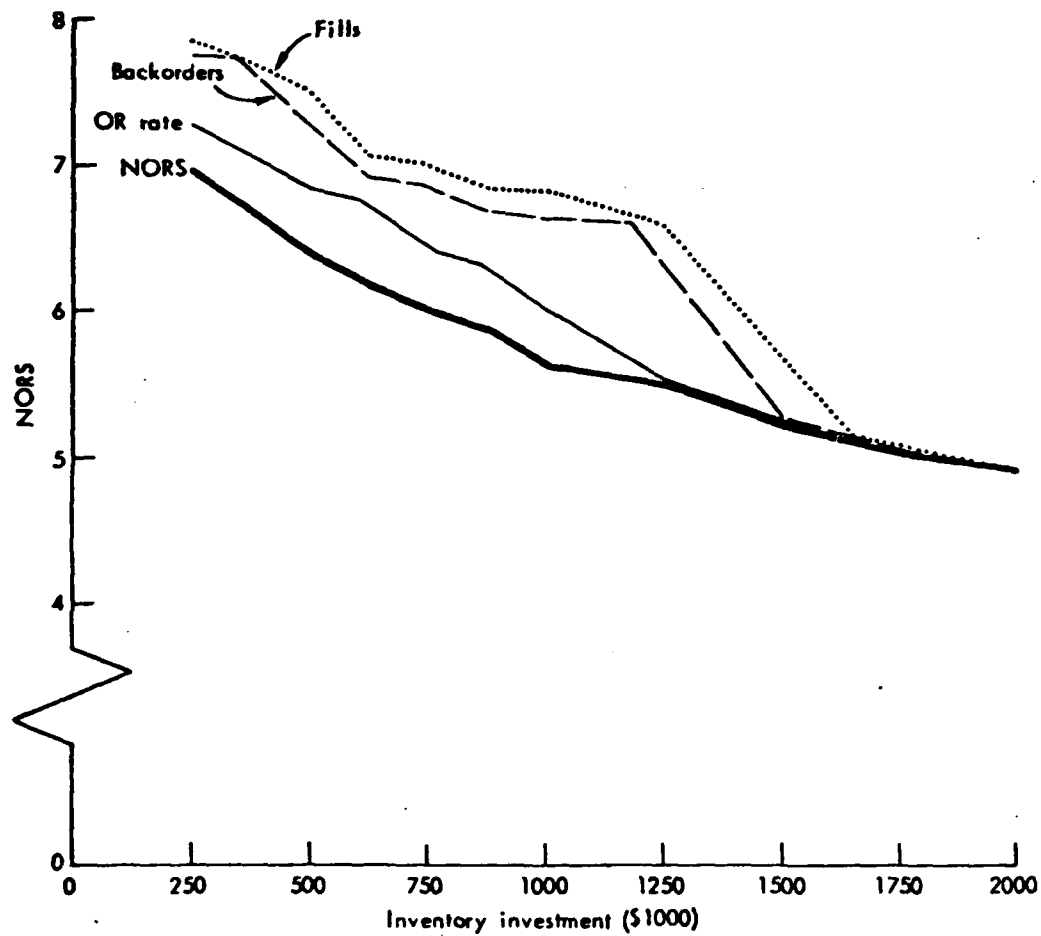


Figure 4.1. Performance Measures Vs. Various Investment Levels
(Figure taken from [3])

solution (see Figure 4.2). In the current D029 Model, the reciprocal of the weight is changed dynamically in order that both objectives converge simultaneously. It is also mentioned in [8] that $1/w = 50$ produces good results. During this research it was observed that $w = 0.025$ produced a good solution for the initial six-day kit and w between 0.015 and 0.025 produced good results for the 30-day kit. A detailed description of selecting a value for w will be provided later when we discuss in Part V the computational experience with this algorithm.

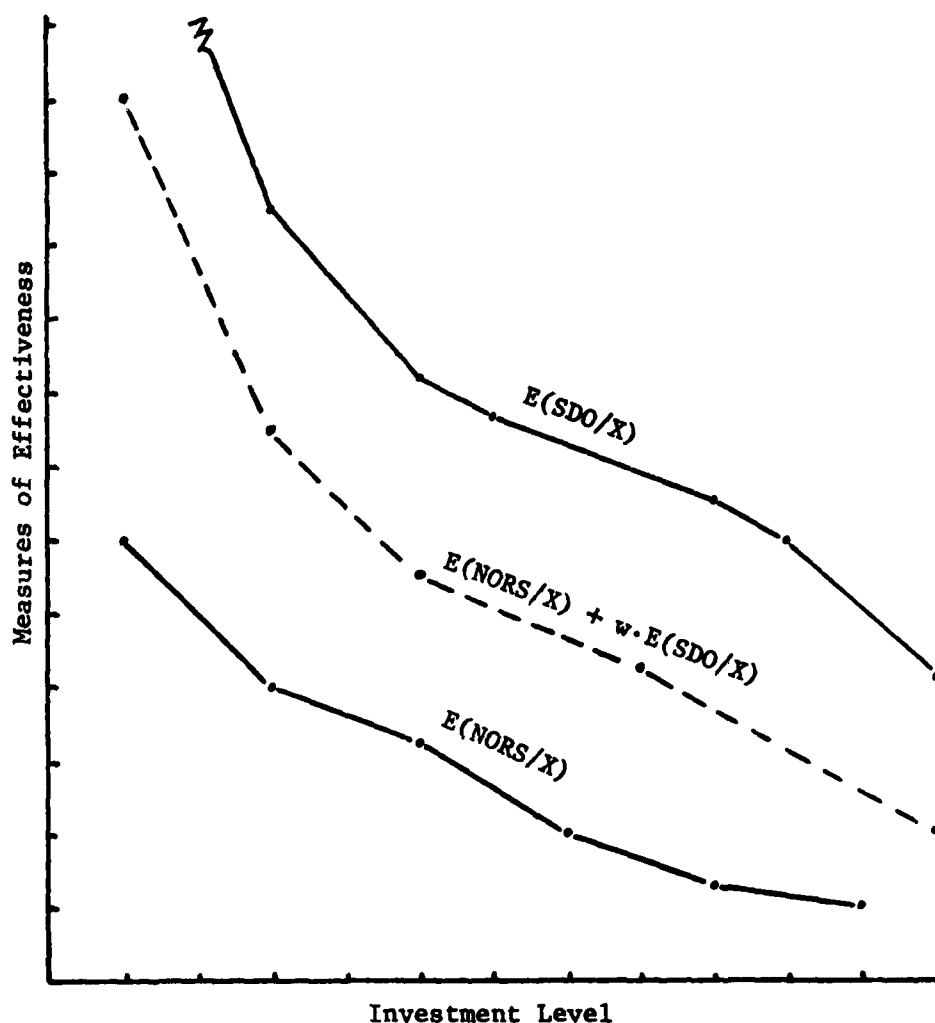


Figure 4.2. Compromise Solution For a Given Weighting Factor w

Potential Solution Methods

Many potential solution methods were investigated. Those that were found significant to the solutions of problem (1), (2), and (3) are summarized as follows:

A Primal Search Algorithm

The problem to be solved is of the form

$$\begin{aligned} \text{Min } f(X) &= E(\text{NORS}/X) \\ \text{s.t. } CX &\leq b \\ X &\geq 0 \end{aligned}$$

The discrete convexity of the (NORS/X) function was proved by B. L. Miller [6]. The Lagrangian for this problem is given by

$$L(X;\lambda) = f(X) + \lambda(CX - b)$$

and

$$\Delta_i L(X;\lambda) = L(X + e_i;\lambda) - L(X;\lambda) \quad i = 1, 2, \dots, n$$

where e_i is the i th unit vector.

This algorithm makes use of the discrete convexity (see [6] and [9]) and the monotonicity. A better feasible solution is obtained from the current solution by determining a search direction and heuristically selecting a step size.

The procedure is as follows:

Step 0. Initialization, pick a λ and p where $0 < p < 1$, and set $k = 0$. Find X^0 such that $L(X^0;\lambda) \leq L(X^0 \pm e_i;\lambda)$ for $i = 1, \dots, n$. a_i 's are assumed to be 1 for simplicity. The following reasoning can easily be extended to the cases with $a_i > 1$. In particular, the first differences for the function are given by

$$\Delta_i L(X; \lambda) = L(X + e_i; \lambda) - L(X; \lambda) \text{ for } i = 1, \dots, n \quad (3.7)$$

and

$$\Delta_i L(X; \lambda) = - \sum_{j=0}^m ((F_i(x_i + j + 1) - F_i(x_i + j))) \prod_{\substack{k=1 \\ k \neq i}}^n F_k(x_k + j) - 1 +$$

$$F_i(x_i + m + 1) + \lambda c_i$$

Since $0 \leq \prod_{k \neq i} F_k(x_k + j) \leq 1$ for $0 \leq j \leq m$, then

$$- \sum_{j=1}^m F_i(x_i + j) - F_i(x_i + m + 1) + F_i(x_i) + \sum_{j=1}^m F_i(x_i + j) - 1 +$$

$$F_i(x_i + m + 1) + \lambda c_i \leq \Delta_i L(X; \lambda)$$

and

$$F_i(x_i) - 1 + \lambda c_i \leq \Delta_i L(X; \lambda) \leq F_i(x_i + m + 1) - 1 + \lambda c_i$$

Therefore, it suffices to find an $X^0 = (x_1^0, \dots, x_n^0)$ where each x_i^0 satisfies $-1 + F_i(x_i^0 - 1) + \lambda c_i \leq 0$ and

$$-1 + F_i(x_i^0) + \lambda c_i > 0 \quad i = 1, \dots, n.$$

Step 1. Search for a better feasible solution; set $k = k + 1$.

Construct a favorable direction

$$d_k = -\nabla f(X^k) - p \cdot p_k \cdot C$$

where $\nabla f(X^k) = (\Delta_1 f(X^k), \dots, \Delta_n f(X^k))$, $\Delta_1 f(X^k) =$

$$- \sum_{j=0}^m \left[F_1(x_1^k + j + 1) - F_1(x_1^k + j) \right] \prod_{\substack{\ell=1 \\ \ell \neq 1}}^n F_\ell(x_\ell^k + j) - 1 + F_1(x_1^k + m + 1)$$

and p_k is obtained by solving $(-\nabla f(X^k) - p_k C) \cdot C = 0$ and p is left as an input parameter by the user. d_k is normalized such the $\max\{|d_{ki}|; i = 1, \dots, n\} = 1$. Calculate an α by $\alpha = \left\lfloor \frac{M - C \cdot X^k}{\sqrt{n} C \cdot d_k} \right\rfloor$ where $\lfloor \cdot \rfloor$ denotes the greatest integer function. Then a point y_α is obtained as $y_\alpha = X^k + \alpha d_k$. The nearest integer point V to y_α is obtained by $v_i = \lfloor y_{\alpha i} + .5 \rfloor$ for $i = 1, \dots, n$.

- i) If V is feasible and $f(V) < f(X^k)$, set $X^{k+1} = V$, to to Step 1.
- ii) If V is feasible, but $f(V) \geq f(X^k)$, then check the points $V + e_i$ for $i = 1, \dots, n$, Figure 4.3. If there exists an i^* such that $V + e_{i^*}$ is feasible and $f(V + e_{i^*}) < f(X^k)$, set $X^{k+1} = V + e_{i^*}$, and go to Step 1. Otherwise, go to Step 2.

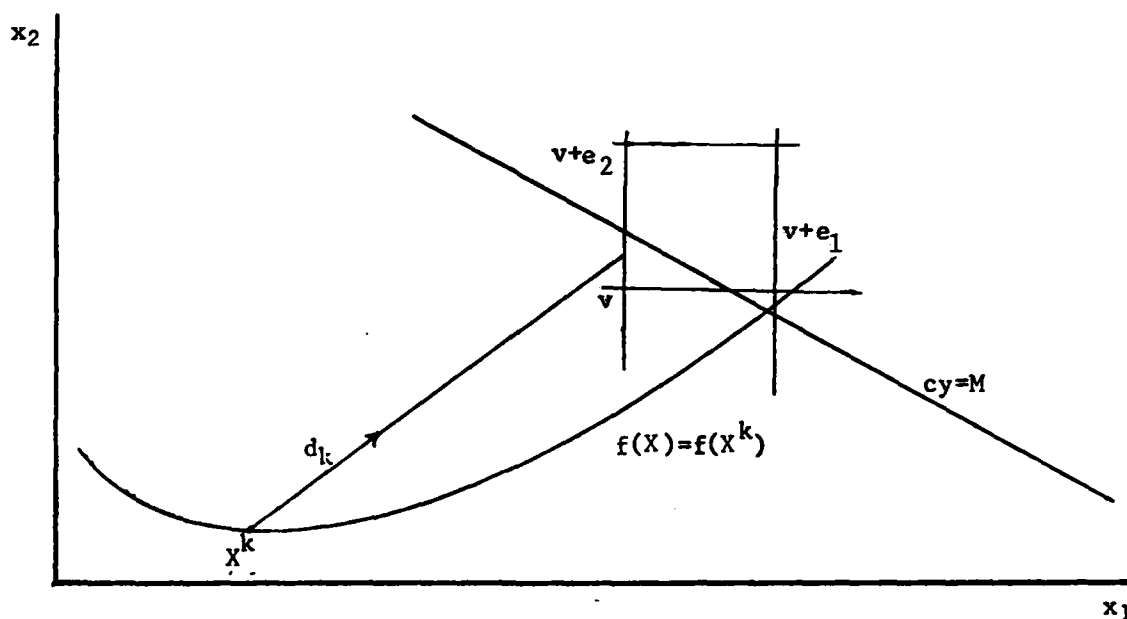


Figure 4.3. Search in the Positive Quadrant

iii) If V is not feasible, then search the points $V - e_i$ for $i = 1, \dots, n$. If there exists an i^* such that $V - e_{i^*}$ is feasible and $f(V - e_{i^*}) < f(X^1)$, set $X^{k+1} = V - e_{i^*}$ and go to Step 1. Otherwise go to Step 2.

Step 2. Test of optimality. Test the points $X^k + e_i$, $i = 1, \dots, n$. If there exists an i^* such that $X^k + e_{i^*}$ is feasible and $f(X^k + e_{i^*}) < f(X^k)$, set $X^{k+1} = X^k + e_{i^*}$ and go to Step 1, otherwise terminate.

The Modified Algorithm of Brooks et al.

This approach requires the objective function $f(X)$ be approximated by a separable function of the form

$$f(X) \approx \sum_{i=1}^n \sum_{j=0}^m b_j F_i(x_i; j)$$

where

$$b_j = \beta_j(x'_1, x'_2, \dots, x'_n) \geq 0$$

for some $X' = (x'_1, x'_2, \dots, x'_n)$, and for some function $\beta_j(\cdot)$.

Let

$$g_i(x_i) = \sum_{j=0}^m b_j F_i(x_i; j)$$

then

$$f(X) \approx \sum_{i=1}^n g_i(x_i)$$

When $g_i(x_i)$ is strictly decreasing, then the implementation of marginal analysis is justified, and leads to the following series of steps.

Step 0. Initialization, pick $b_0^0, b_1^0, \dots, b_m^0$, set $k = 1$.

Step 1. Solve the problem.

$$\min \sum_{i=1}^n g_i(x_i)$$

Subject to

$$\sum_{i=1}^m c_i x_i \leq b$$

$$x_i \geq 0 \text{ integer}$$

by the following algorithm which is known as marginal analysis [4].

Step 1 i) Initialization: $X^0 = (0, 0, \dots, 0)$, set $\ell = 0$.

Step 1 ii) $Y^{\ell+1} = Y^{\ell} + e_j$, if feasible, where j is any index for which the ratio

$$r_i = \frac{g_i(y_i + 1) - g_i(y_i)}{c_i}$$

is maximum.

Step 1 iii) Stop if $CY^{\ell+1} \geq b$, and set $X^k = Y^{\ell}$, then go to Step 2. Otherwise go to Step 1 ii.

Step 2 Calculate $b_j^k = \beta_j(X^k)$ for $j = 0, 1, \dots, m$.

Step 3 If b_j^k is very close to b_j^{k-1} for all $j = 0, 1, \dots, m$, then $X^* = X^k$, and stop. Otherwise, set $k = k + 1$, and go to 1.

This is a modified version of the algorithm proposed by R. B. S.

Brooks, C. A. Gillen, and J. Y. Lu on page 24 of [3]. The Step 1 of their

approach was quite different from the one described above. They attempted to find the saddle point solution to the problem

$$\max_{\lambda \geq 0} \min_{X \geq 0} L(X; \lambda) = \sum g_i(x_i) + \lambda (\sum c_i x_i - b).$$

Their Step 1 was as follows:

Step 1) Find $x_1^k, x_2^k, \dots, x_n^k$ at k^{th} iteration which maximizes

$$\sum_{j=0}^m b_j \sum_{i=1}^n \log F_i(x_i + j) - \lambda \sum_{i=1}^n c_i x_i$$

Since this function is separable, the optimization is carried out for each component separately. At the end of the third step, the Lagrange Multiplier is rearranged as suggested in [2], and the algorithm repeated again.

A Saddle Point Search Algorithm

This algorithm solves the problem

$$\max_{\lambda > 0} \min_{X \in \Pi^n} L(X; \lambda).$$

The algorithm in fact searches the saddle point (X^*, λ^*) of the Lagrangian function. This is, however, more complicated than the previous algorithms, and requires subalgorithms to solve the primal and dual subproblems. This was originally suggested by H. Everett [2].

The procedure is as follows:

Step 0. Initialization. Pick X^0 and λ_0 . Set $k = 0$.

Step 1. Solve the primal subproblem.

$$\min_{X \in \Pi^n} L(X; \lambda_k)$$

to obtain X^{k+1} .

Step 2. Solve the dual subproblem

$$\max_{\lambda > 0} L(X^{k+1}; \lambda)$$

to obtain λ_{k+1} .

Step 3. Stop if λ_{k+1} is very close to λ_k . Otherwise set $k = k + 1$ and go to Step 1.

B. L. Miller developed an algorithm to solve the primal subproblem for discretely convex functions. Since a detailed description of this algorithm is presented in [6], it is not going to be discussed here. This is a very complicated algorithm. The author claims it produces good solutions.

At the k^{th} iteration ($k \geq 1$), a piecewise convex approximation of $L(X; \lambda)$ can be given in the space generated by the points, $X^0, X^1, \dots, X^{k+1}$ as follows:

$$L(X; \lambda) = \sum_{t=0}^{k+1} \mu_t L(X^t; \lambda)$$

where $\sum_{t=0}^{k+1} \mu_t = 1$, $\mu_t \geq 0$ for all $t = 0, 1, \dots, k+1$.

Then the dual subproblem is restated as

$$\max \sum_{t=0}^{k+1} \mu_t L(X^t; \lambda)$$

Subject to

$$\sum_{t=0}^{k+1} \mu_t = 1$$

$$\lambda > 0, \mu_t \geq 0 \text{ for } t = 0, 1, \dots, k+1$$

Since $L(X;\lambda) = f(X) + \lambda CX - \lambda b$, then the problem

$$\max \sum_{t=0}^{k+1} \mu_t f(X^t) + \lambda \sum_{t=0}^{k+1} \mu_t CX^t - \lambda b$$

Subject to
$$\sum_{t=0}^{k+1} \mu_t = 1$$

$$\lambda > 0, \mu_t \geq 0 \text{ for } t = 0, 1, \dots, k+1$$

can be considered as the Lagrangian of the linear program

$$\min \sum_{t=0}^{k+1} \mu_t f(X^t)$$

Subject to
$$\sum_{t=0}^{k+1} \mu_t CX^t \leq b$$

$$\sum_{t=0}^{k+1} \mu_t = 1$$

$$\mu_t \geq 0 \text{ for } t = 0, 1, \dots, k+1$$

or as the Lagrangian of the dual linear program

$$\min (\lambda b + v)$$

Subject to $\lambda CX^t + v \geq f(X^t) \text{ for } t = 0, 1, \dots, k+1$

$$\lambda > 0$$

$$v \text{ unrestricted}$$

Therefore the dual linear program can be solved instead of the subproblem in Step 2. This can be done as suggested by Brooks R. B. S. and A. M. Geoffrion in [2].

Example Problem Solutions

General

A war readiness spares kit problem with five items is used to demonstrate the calculations in the algorithms. Computer program listings are given in Appendix A. The steady-state Poisson demand rates and cost of each item is given in Table 4.1.

The budget is assumed to be $M = \$25000$. In addition the value for m of Equation is chosen to be 5. Also, $a_i = 1$ for $i = 1, 2, 3, 4, 5$.

TABLE 4.1
COSTS AND DEMAND RATES FOR THE SAMPLE WRSK PROBLEM

Item No.	Cost/Unit	Demand Rate
1	2980.	2.10
2	1751.	1.50
3	462.	1.20
4	1500.	5.00
5	345.	3.50

Solution by Means of the Primal Search Method

Iteration 0. Initialization: Parameter p is chosen to be .85 and the Lagrange multiplier is specified as $\lambda = .0002$. (For further reference see [9].) This iteration seeks an X^0 which satisfies the condition of 3.11, page 4.6.

Since $-1 + F_1(1) + (.0002)(2980) = -1 + .37962 + .596 = -.02438$,
and $-1 + F_1(2) + (.0002)(2980) = -1 + .64963 + .596 = .24563$, $x_1^0 = 2$.
Repeating the same argument for each component, yields $X^0 = (2, 2, 3, 6, 6)$.

Iteration 1. Seeking a better feasible solution: The first differences, $\nabla f(X^k)$, at the point $X^0 = (2, 2, 3, 6, 6)$ are computed (by Step 1, page 4.6), and $\Delta f(2, 2, 3, 6, 6) = (-.240, -.106, -.015, -.163, -.035)$

$$p_1 = \frac{-\Delta f(2, 2, 3, 6, 6)}{C.C} .C = .00008$$

Since $p = .85$, then after normalization, the direction is

$$d_1 = (.610, -.225, -.268, 1., .190)$$

and α is evaluated to be 1. Thus a point $U = (3, 2, 3, 7, 6)$ is obtained by finding the nearest integer point to $Y = (2.610, 1.775, 2.732, 7., 6.190)$. However the point U is not feasible, $\text{cost}(U) = 26398$, hence a search in the negative direction is required. For that purpose $\Delta_1 f(2, 2, 3, 7, 6) = -.262$, $\Delta_2 f(3, 1, 3, 7, 6) = -.333$, $\Delta_3 f(3, 2, 2, 7, 6) = -.076$, $\Delta_4 f(3, 2, 3, 6, 6) = -.185$, and $\Delta_5 f(3, 2, 3, 7, 5) = -.100$ are calculated. The minimum of these values is for $i = 3$, but $(3, 2, 2, 7, 6)$ is not feasible. The second minimum value is for $i = 5$, again $(3, 2, 3, 7, 5)$ is not feasible. The third minimum is for $i = 4$, fortunately $(3, 2, 3, 6, 6)$ is feasible and $f(3, 2, 3, 6, 6) = .98571$. Therefore $X^1 = (3, 2, 3, 6, 6)$ with $CX = 24898$.

Iteration 2. As in Iteration 1, a search direction $d_2 = (-.356, .362, -.037, 1., .216)$ issuing from $(3, 2, 3, 6, 6)$ is determined. A point $U = (3, 2, 3, 7, 6)$ is obtained similarly. Since U is infeasible, and a search in the negative direction does not yield a distinct and better feasible point than $X^1 = (3, 2, 3, 6, 6)$, a search in the positive direction at X^1 is employed. All the points of the form $X^1 + e_i$ for $i = 1, 2, \dots, 5$ are infeasible. Hence $X^1 = (3, 2, 3, 6, 6)$ is optimal. This small problem takes .24 seconds of execution time.

Solution by the Modified Solution Method of Brooks et al.

Iteration 1. Choose $b_0 = b_1 = \dots = b_4 = 0$ and $b_5 = 1$. Set $x_i^0 = 0$ for $i = 1, 2, \dots, 5$, then $g_i(x_i) = b_5 \log F_1(x_i + 5)$, and calculating the

ratio $r_i = \frac{g_i(x_i + 1) - g_i(x_i)}{c_i}$ for each $i = 1, \dots, 5$ yields: $r_1 = 0.00000496$,

$r_2 = 0.00000202$, $r_3 = 0.00000270$, $r_4 = 0.00014200$, $r_5 = 0.00024952$. For instance,

$$r_1 = \frac{b_5}{c_1} \log \frac{F_1(6)}{F_1(5)} = \frac{1}{2980} \log \frac{.99413}{.97955} = 4.96 \times 10^{-6}$$

r_5 is the maximum, hence $x_5^1 = 1$. Updating r_5 yields: $r_1 = 0.00000496$, $r_2 = 0.00000202$, $r_3 = 0.00000270$, $r_4 = 0.00014200$, $r_5 = 0.00011714$. The maximum is r_4 , then $x_4^1 = 1$. Continuing in this fashion produces $X^1 = (2, 2, 2, 8, 7)$ with $\text{cost}(X^1) = 24801$, and calculating b_j by $b_j^1 = \frac{n}{\sum_{i=1}^n F_i(x_i^1 + j)}$ gives $b_0^1 = .419$, $b_1^1 = .726$, $b_2^1 = .898$, $b_3^1 = .967$, $b_4^1 = .991$, and $b_5^1 = .998$.

Iteration 2. Repeating the same steps with b_j^1 $j = 0, 1, \dots, 5$ yields $X^2 = (2, 2, 4, 7, 9)$ with $\text{cost}(X^2) = 24915$, and $b_0^2 = .450$, $b_1^2 = .728$, $b_2^2 = .891$, $b_3^2 = .962$, $b_4^2 = .988$, and $b_5^2 = .996$. Since b_j^1 's are not within .001 of b_j^2 's, go to Iteration 3.

Iteration 3. Repeating the same steps again with b_j^2 for $j = 0, 1, 2, \dots, 5$ yields $X^3 = X^2$ and $b_j^3 = b_j^2$, hence X^2 is optimal with $f(2, 2, 4, 7, 9) = .98619$.

This solution however is a near optimal solution. The primal search procedure has produced a better solution than this one. On the other hand this method is faster, the execution time to solve this small problem with this algorithm is .15 seconds.

In this algorithm b_j 's are treated as parameters. They have practical implications for real life problems. β_j is given by the product $\prod_{i=1}^n F_i(\bar{X} + j)$ for some $\bar{X}$ and $j = 0, 1, \dots, m$. This product is in fact equal to the probability of cannibalizing at most j airplanes with a kit $\bar{X}$. In other words b_j gives the operational rate given j airplanes for cannibalization for a kit $\bar{X}$. Selecting $b_0 = b_1 = \dots = b_{m-1} = 0$ and $b_m = 1$ implies that all the planes will be cannibalized and corresponds to the most pessimistic view.

Solution by the Saddle Point Search Algorithm

Iteration 1. A starting solution of $X^0 = (10, 10, 10, 10, 10)$ with $f(X^0) = .02486$ and cost $(X^0) = 70380$ and a Lagrange multiplier of $\lambda_0 = .0001$ is chosen. In the first step the subproblem $\min_{X \in \Pi^n} L(X; \lambda_0) = f(X) + \lambda_0(CX - M)$ is solved by the algorithm proposed by B. L. Miller in [6]. This algorithm produces the solution $X^1 = (2, 2, 2, 6, 6)$ with $f(X^1) = 1.28241$ and cost $(X^1) = 21456$.

Step 1. Initialization: An initial point is obtained by solving $-1 + F_i(\omega_i - 1) + \lambda C_i \leq 0$ and $-1 + F_i(\omega_i) + \lambda C_i > 0$ and setting $X_i = \max(0, \omega_i - k)$ for all $i=1, \dots, n$ where k is an input parameter and chosen as 1 in this particular example. Since $-1 + F_1(2) + (.0001)(2980) = -1 + .6496 + .2980 = -.0524$ and $-1 + F_1(3) + (.0001)(2980) = -1 + .8386 + .2980 = .1366$, then $\omega_1 = 3$, consequently $x_1^0 = 2$. Repeating the same argument for each component yields that $x_2^0 = 2$, $x_3^0 = 2$, $x_4^0 = 6$, and $x_5^0 = 6$.

Step 2. Finding A Stationary Vector: Starting from $X^0 = (2, 2, 2, 6, 6)$, a point X is sought such that $\Delta_i g(X - e_i) \leq 0$ and $\Delta_i g(X) > 0$ for all $i=1, \dots, 5$. For instance for item 5, $\Delta_5 g(2, 2, 2, 6, 6)$ is

calculated as follows; $\Delta_5 g(2, 2, 2, 6, 6) = \lambda C_5 + F_5(6 + 5 + 1) - 1.$

$$- \sum_{j=0}^5 (F_5(6 + j + 1) - F_5(6 + j)) \prod_{i=1}^4 F_i(X_i + j)$$

j	$F_1(2+j)$	$F_2(2+j)$	$F_3(2+j)$	$F_4(6+j)$	$F_5(7+j) - F_5(6+j)$	Product
0	.64963	.80885	.87949	.76218	.03855	.01358
1	.83864	.93436	.96623	.86663	.01687	.01107
2	.93787	.98142	.99225	.93191	.00656	.00558
3	.97955	.99554	.99850	.96817	.00230	.00217
4	.99414	.99907	.99975	.98630	.00073	.00072
5	.99851	.99983	.99996	.99455	.00021	.00020
SUM =						.03332

Then $\Delta_5 g(2, 2, 2, 6, 6) = .0345 + .99992 - 1. - .03332 = .00110$. Since it is positive, $\pi_5 = 6$. This argument is repeated for each item to obtain a stationary vector $X = (2, 2, 2, 6, 6)$ with $f(X) = 1.28241$, $\text{cost}(X) = 21456$, and $g(X) = 3.42801$.

Step 3. A heuristic approach to obtain improvement in "up-phase":

Let $S_i = \sum_{k=i}^n e_k - e_i$, then the ratios $\Delta_i g(X + S_i) / \Delta_i g(X)$ are ranked in ascending order, and the vectors of the form $X + U_i$ will be tested for improvement where U_i is the vector with +1 for variables ranked from 1 to i and 0 for the ones ranked i + 1 to n. Calculation of the ratios gives $\Delta_1 g(X + S_1) / \Delta_1 g(X) = -5.61$, $\Delta_2 g(X + S_2) / \Delta_2 g(X) = .15$, $\Delta_3 g(X + S_3) / \Delta_3 g(X) = 1.47$, $\Delta_4 g(X + S_4) / \Delta_4 g(X) = 1.44$ and $\Delta_5 g(X + S_5) / \Delta_5 g(X) = 1.64$. The ordering according to the magnitudes yields 1, 2, 4, 3, 5 respectively. For instance, for $i = 4$ $U_4 = (1, 1, 0, 1, 0)$ and $X + U_4 = (3, 3, 2, 7, 6)$, but this point does not yield a better solution, since $g(X + U_4) = 3.48107$ is greater than $g(X) = 3.42801$.

Repeating the same argument for each component results that the current solution is stationary.

The same argument is repeated for the "down-phase" too. This does not yield a better solution either.

Step 4. A Second Initialization and Reducing the Set of Potential Improving Points.

First the up-phase is implemented. For instance for $i = 1$, the vector $(0, 1, 1, 1, 1)$ will be checked for improvement by looking at $\Delta_1 g(2, 3, 3, 7, 7)$. Since $\Delta_1 g(2, 3, 3, 7, 7) = -.028$ is negative, the test fails. For $i = 2$, the vector $(1, 0, 1, 1, 1)$ is tested and $\Delta_2 g(3, 2, 3, 7, 7) = .005$, then the conclusion is that x_2^* can not be 3. Next the vector $(1, 0, 0, 1, 1)$ is used for the third component and $\Delta_3 g(3, 2, 2, 7, 7) = -.056$. Similarly $\Delta_4 g(3, 2, 3, 6, 7) = -.069$ and $\Delta_5 g(3, 2, 3, 7, 6) = -.022$. The test fails for $i = 1, 3, 4$, and 5. Consequently Part 1 is used. Calculation of the ratios $\Delta_1 g(2, 3, 3, 7, 7)/\Delta_1 g(2, 2, 2, 6, 6) = -5.61$, $\Delta_3 g(3, 2, 2, 7, 7)/\Delta_3 g(2, 2, 2, 6, 6) = 1.37$, $\Delta_4 g(3, 2, 3, 6, 7)/\Delta_4 g(2, 2, 2, 6, 6) = 1.33$ and finally $\Delta_5 g(3, 2, 3, 7, 6)/\Delta_5 g(2, 2, 2, 6, 6) = 1.52$ gives that the maximum ratio is for $i = 5$. By contradiction it will be proved that $s_5 = 1$ implies $s_3 = 1$ where $X + S$ is the point to be tested. Since $b_0 = -\lambda C_3$ $-\Delta_3 g(2, 2, 3, 6, 7) = .042 > 0$, S_3 can not be 1. Repeating this argument for $i = 1$ and $i = 4$ does not yield a better solution. Consequently the down phase is implemented. Down-phase does not generate a better feasible solution. Finally the conclusion is that $X = (2, 2, 2, 6, 6)$ is a stationary point.

Then the linear program

$$\min .02486 \mu_1 + 1.28241 \mu_2$$

$$\text{Subject to } 70380 \mu_1 + 21456 \mu_2 \leq 25000$$

$$\mu_1 + \mu_2 = 1$$

$$\mu_1, \mu_2 \geq 0$$

is solved to obtain $\lambda_1 = .0000257$. Since λ_1 is not within .000005 of λ_0 another iteration is performed.

Iteration 2. The subproblem $\min_{X \in \Pi^n} L(X; \lambda_1) = f(X) + \lambda_1(CX - M)$ is solved again in the first step by the same method to obtain $X^2 = (4, 4, 4, 9, 9)$ with $f(X^2) = .17727$ and $\text{cost}(X^2) = 37377$. In the second step the linear program $\min .02486 \mu_1 + 1.28261 \mu_2 + .17727 \mu_3$,

$$\text{Subject to } 70380 \mu_1 + 21456 \mu_2 + 37377 \mu_3 \leq 25000$$

$$\mu_1 + \mu_2 + \mu_3 = 1$$

$$\mu_i \geq 0 \quad i = 1, 2, 3$$

is solved to obtain $\lambda_2 = .0000694$. Since λ_2 is not within .000005 of λ_1 , another iteration is required. Continuing in this fashion produces the following results:

Iteration 3. $X^3 = (3, 3, 3, 8, 7)$ with $f(X^3) = .49955$ and $\text{cost}(X^3) = 29994$. Further $\lambda_3 = .0000917$.

Iteration 4. $X^4 = (2, 2, 3, 7, 7)$ with $f(X^4) = 1.02459$ and $\text{cost}(X^4) = 23763$. Also $\lambda_4 = .0000848$.

Iteration 5. $X^5 = (3, 2, 3, 7, 7)$ with $f(X^5) = .75556$ and $\text{cost}(X^5) = 26743$. Also $\lambda_5 = .0000903$.

Iteration 6. $X^6 = (2, 2, 3, 7, 6)$ with $f(X^6) = 1.06282$ and cost $(X^6) = 23418$ and $\lambda_6 = .0000903$.

This algorithm produced a solution with a better objective value than the previous ones, but it is an infeasible solution. In fact, this algorithm suffers from the duality gap. In the first step of this algorithm, the subproblem $\min_{X \in \Pi^n} L(X; \lambda)$ is solved. Since $L(X; \lambda)$ is discrete in $X \in \Pi^n$, there is a range for λ values which produce the same answer. From the results of previous algorithms, it was discovered that the optimal solution was $X^* = (3, 2, 3, 6, 6)$. Evidently its corresponding Lagrange multiplier should lie between $\lambda_4 = .0000848$ and $\lambda_5 = .0000903$. Unfortunately the algorithm won't generate this value, and if λ is close to λ_4 , the solution to subproblem is X^5 and if λ is close to λ_5 , then the solution is X^6 , see Figure 4.4.

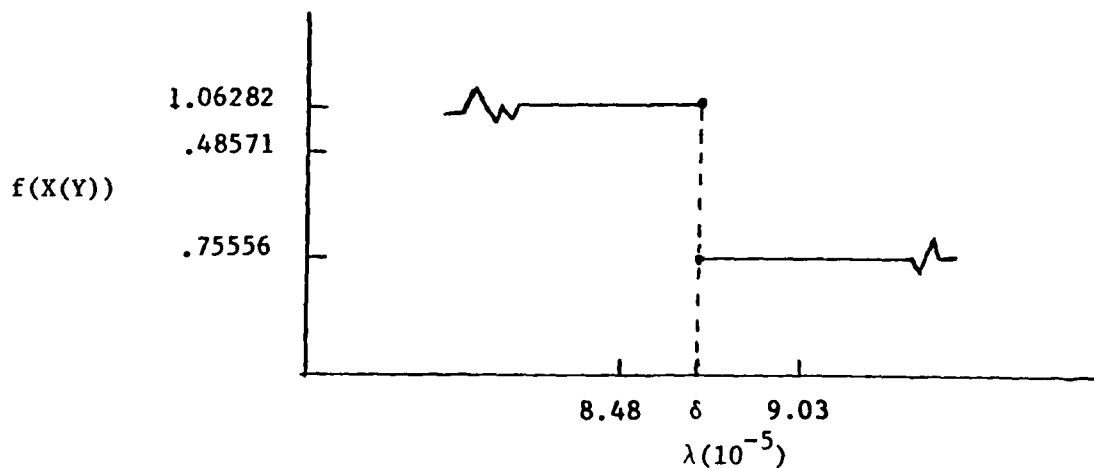


Figure 4.4. Duality Gap

Selection of Solution Method

The objective of this research is to develop and/or modify an efficient solution method to solve the WRSK problem. Therefore, these three solution methods have been under investigation to observe their computational requirements. The result of this investigation shows that the Modified Algorithm of Brooks et al. is more efficient in terms of computational time than the others. This algorithm is also going to be compared with the current D029 Model which is summarized on the next page. The result of this comparison will be discussed in Section V.

The Modified Algorithm of Brooks et al. is a greedy algorithm in nature, and will be refined to in the remainder of this report as the "Greedy Algorithm". It uses marginal analysis as a subalgorithm. It is easier to implement in solving the WRSK problem and is faster than the others. The selection of its initial parameters does not require any guesswork from the user. The parameter values obtained for the requirement kit 57-18 can serve that purpose. On the other hand, marginal analysis adds one unit of the item which will yield the highest rate of return in terms of performance measures per dollar invested in the kit. Approximation of the $E(NORS/X)$ function by a separable function [as discussed in Part III] eases the computations and in fact eliminates the need for updating all of the marginal returns whenever a unit of some item is added to the kit as is necessary in the current D029 computer program. The speed of convergence of an algorithm is very important, especially when solving large size problems, and this algorithm supplies this advantage.

The others have some particular advantages, but they are slower than this algorithm. The Saddle Point Search Algorithm is theoretically very

appealing but suffers from the duality gap as indicated by the small example of the last section. The Primal Search Algorithm uses the first differences in determining the search direction to explore better feasible solutions. All three algorithms contain some heuristic rules. Thus the solutions are not necessarily the true optimum which, for the same reason, is true for the current D029 Model. The comparison of their solutions either with each other or with the D029 solution indicates that they can generate good near optimal solutions.

The Current D029 WRSK Model

This model and its associated computer program is currently in use by WPAFLC. This solution method also makes use of marginal analysis. A detailed description of this algorithm is provided in [8]. A brief description of the algorithm is provided here. The algorithm solves the problems of the form

$$\begin{array}{ll}
 \text{Min } CX & \\
 \text{s.t.} & E(\text{NORS}/X) \leq \alpha_1 \\
 & E(\text{SDO}/X) \leq \alpha_2 \\
 & X \geq 0 \text{ integer}
 \end{array}$$

in the following manner.

Step 0. Set $U = 0$, $X^0 = 0$

Step 1. Choose a W^k and I_{Nt}^k

Step 2. Add I_{Nt}^k units to the kit by selecting the highest I_{Nt}^k ratios from

$$r_i = \frac{w^k \Delta_i E(\text{NORS}/X^k) + \Delta_i E(\text{SDO}/X^u)}{C_i}$$

$$X^{k+1} = X^k + \sum_{i \in H_{I_{Nt}}^k} e_i$$

where $H_{I_{Nt}}^k$ is the index set of the highest I_{Nt}^k ratios.

Step 3. Calculate $E(\text{NORS}/X^{k+1})$, $E(\text{SDO}/X^{k+1})$ and CX^{k+1}

Step 4. Phase 1: Terminate if $CX^{k+1} > b^*$

where b^* is 90% of the cost of Kit 57-18

Phase 2: Terminate if $E(\text{NORS}/X^{k+1}) \leq \alpha_1$ and $E(\text{SDO}/X^{k+1}) \leq \alpha_2$

Otherwise Set $k = k + 1$ and go to Step 1.

PART V

A SUMMARY OF COMPUTATIONAL EXPERIENCE

Introduction

Computational experimentation was devoted to the investigation of computational requirements, initialization of parameters for the Greedy Algorithm (Modified Algorithm of Brooks et al) and the IBM version of D029 Program. F4D Data was used in the computational analysis. Some of the subroutines remain the same in both programs in order to make meaningful comparisons. (The FORTRAN listings of the programs are in Appendix B.2 & C.2.) The Greedy Algorithm takes 16.53 seconds of CPU time, and the IBM version of D029 Program takes 24.78 seconds of CPU time on IBM 370/168 at O.S.U. The Greedy Algorithm is obviously faster on the F4D problem than the algorithm currently in use.

The Parameters b_j and m of the E(NORS/X) Approximation

The Greedy Algorithm requires that b_j and m be initialized in advance. A thorough inspection of the WRSK problem suggested that initial b^0_j values can be calculated from the requirement kit 57-18 by the formula

$$b^0_j = \sum_{i=1}^n F_i(\bar{x}_i + ja_i) \quad \begin{array}{l} j = 0, 1, 2, \dots, K = \text{Number of planes and} \\ n = \text{Total number of items in consideration} \end{array}$$

in the squadron, where $\bar{x}_i$ is the kit 57-18 quantity for item i . The initial b^0_j values for Phase 1 and Phase 2 of the WRSK Problem are displayed in Table 5.1. Each b_j is the operational rate given that there are j aircraft for cannibalization. Hence, b^0_j 's represent operational rates under the kit 57-18. These are good choices for b_j 's since the b^0_j 's may be con-

sidered as operational rate goals as a consequence of the AFLCR. Since the WRSK problem attempts to find a kit which will perform better than the kit 57-18, these b^0_j 's, which can be observed in Figures 5.7a and 5.7b on pages 23 and 24 respectively, are the target operational rates for the optimal kit. On the other hand, a natural selection for m is the number of planes in the squadron. An inspection of the Table 5.1 indicates that the b_j 's approach 1 for some lower value than m . Consequently, using Table 5.1, one can choose $m = 12$ for Phase 1 and still get the same answers.

The b^*_j 's displayed in Table 5.1 correspond to the calculated values from the optimal kit and can also be observed in Figures 5.9a and 5.9b respectively. For Phase 1 the b^*_j 's are very close to the b^0_j 's. However, a similarly close relationship between b^*_j and b^0_j does not occur in Phase 2. Note that each $b^*_j \geq b^0_j$ for all j . Since the b^0_j represent the operational rate goals, this shows that the optimal kit provides higher operational rates than the goals.

Instead of heuristically choosing the initial b^0_j 's, the Greedy program of Appendix B.2 uses the b^0_j 's of the 5718 kit. It also uses $m=24$, the number of planes in the squadron, since 24 is not a large number.

TABLE 5.1
COMPARISON OF b_j VALUES

j	Phase 1		Phase 2	
	5718 Kit b^0_j	Optimal Kit b^*_j	5718 Kit b^0_j	Optimal Kit b^*_j
0	.00000010	0.0	0.	0.0
1	.00286799	.00172540	0.	0.0
2	.06779460	.06441003	.00000001	.00000935
3	.25850234	.26115723	.00011449	.00376217
4	.48975927	.49893585	.00861967	.05835153
5	.68018747	.69276720	.07484124	.22116812
6	.81107932	.82435903	.23535679	.43810253
7	.89285344	.90472555	.44260779	.63279580
8	.94108090	.95038868	.63162230	.77581119
9	.96842135	.97499286	.77280201	.86953349
10	.98345607	.98772442	.86636330	.92667958
11	.99151443	.99410652	.92392780	.95987318
12	.99573534	.99722461	.95769561	.97850019
13	.99789912	.99871559	.97688397	.98868473
14	.99898552	.99941528	.98754667	.99413854
15	.99951989	.99973806	.99337298	.99700800
16	.99977739	.99988454	.99651327	.99849417
17	.99989891	.99994995	.99818555	.99925267
18	.99995505	.99997868	.99906608	.99963432
19	.99998044	.99999108	.99952462	.99982364
20	.99999167	.99999634	.99976077	.99991621
21	.99999653	.99999852	.99988101	.99996080
22	.99999859	.99999942	.99994153	.99998195
23	.99999944	.99999972	.99997163	.99999182
24	1.	.99999991	1.	.99999636

Articulation of the Weight w

The weighting factor w represents the relative importance of $E(\text{SDO}/X)$ function compared to the $E(\text{NORS}/X)$ function. The weighting factor is, in fact, a measure of cannibalization of any plane for each back-ordered part. Conversely, $1/w$ measures the number of consolidated parts from a cannibalized airplane. During a mission, if a plane is grounded for lack of spare parts, it may then be cannibalized to repair others by removing its good parts and installing these parts in other downed planes. Theoretically 100% cannibalized is possible. This process increases the work load during the mission. Obviously, a reduction in the work load is required by the mission command. In practice, however, there is empirical knowledge as to when to cannibalize a plane, which plane to cannibalize, and how many parts to be removed from a plane. In the D029 program, manipulation of the parameter RATIO indicates that this empirical knowledge is regularly applied (RATIO in the Greedy program has a different meaning). It has also been observed that $\text{RATIO} = 50$ produces better results than any other value, for the purpose of reaching the $E(\text{NORS})$ and $E(\text{SDO})$ goals simultaneously. The operational meaning of $\text{RATIO} = 50$ is that, on the average, 50 parts are removed from a cannibalized plane; thus RATIO corresponds to $1/w$. This would also mean that 2% of an airplane is cannibalized for each back order. It is suggested by this research that w be assessed in advance by evaluating this empirical knowledge. Tables 5.2, 5.3, 5.4 and Figures 5.1, 5.2, 5.3 show the relationship between $E(\text{NORS}/X)$ and $E(\text{SDO}/X)$ for different weights. These relationships based on limited experimentation, appear to be linear effects.

The problem is to determine the best effective weight. Computational experimentation with F4D data indicates that $w = 2.25\%$ is the effective weight for the initial six-day period, and that $1.5\% \leq w \leq 2.5\%$ gives a range for the effective weight for a thirty-day period. This means that 45 back ordered parts would be cannibalized during the initial six-day and 40 to 65 back ordered parts would be cannibalized during the 30 day mission period. This observation should be combined with the empirical knowledge to come up with an effective weight in determining a best compromise kit.

Only a limited effort was undertaken to determine the underlying relationships between the weighting factor w , budgetary levels, objective function values, and kit configuration sensitivity to these parameters and measures. Tables 5.2, 5.3, 5.4 and Figures 5.1, 5.2, 5.3 indicate various ways that relationships can be exhibited and analyzed. Analysis and study of the relationships exhibited lead to the selection of parameters and measures that are used in the next section which compares various solutions which have been computed.

TABLE 5.2

OBJECTIVE FUNCTION $\Psi(w)$ VS. WEIGHTING FACTOR w :

PHASE 1 SOLUTIONS FOR A BUDGET OF \$3,635,906

$w(10^{-2})$	$E(NORS/X)$	$E(SDO/X)$	$\Psi(w)$	$\tilde{\Psi}(w)$
2	4.80602	31.02199	5.42646	5.4346
2.25	4.84981	29.36157	5.51045	5.5061
2.5	4.85617	29.07728	5.58315	5.579
3.	4.87135	28.57339	5.72855	5.724
4.	4.94188	26.69839	6.00982	6.013

$$\Psi(w) = E(NOR/X) + w \cdot E(SDO)$$

$$\tilde{\Psi}(w) = 28.94220 w + 4.85578$$

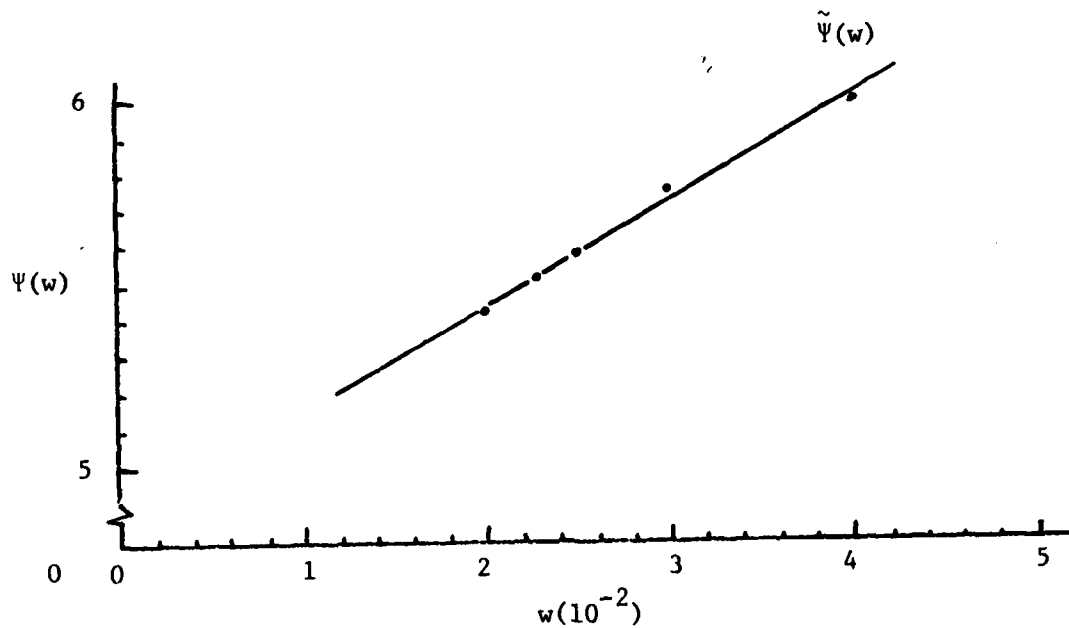
Figure 5.1. Plot of $\tilde{\Psi}(w)$ Exhibited in Table 5.2

TABLE 5.3

OBJECTIVE FUNCTION $\Psi(w)$ VS. WEIGHTING FACTOR w :
 PHASE 2 SOLUTIONS WITH A BUDGET OF $\$7.5 \times 10^6$

$w(10^{-2})$	$E(NORS/X)$	$E(SDO/X)$	$\Psi(w)$	$\tilde{\Psi}(w)$
.3	6.56880	261.25884	7.35258	7.36159
1.	6.91434	113.02191	8.04456	8.05923
1.25	7.01317	104.48303	8.31921	8.30838
1.5	7.12255	96.53860	8.57063	8.55754
2.	7.15833	94.41359	9.04660	9.05585
2.5	7.22784	90.83792	9.49879	9.55417

$$\Psi(w) = E(NOR/X) + w \cdot E(SDO)$$

$$\tilde{\Psi}(w) = 7.06260 + 99.66263w$$

TABLE 5.4

OBJECTIVE FUNCTION $\Psi(w)$ VS. WEIGHTING FACTOR w :
 PHASE 2 SOLUTIONS WITH A BUDGET OF $\$9 \times 10^6$

$w(10^{-2})$	$E(NORS/X)$	$E(SDO/X)$	$\Psi(w)$	$\tilde{\Psi}(w)$
1	4.93755	41.56367	5.35319	5.37412
1.5	5.01991	38.17230	5.59249	5.57405
1.75	5.05138	36.31621	5.68691	5.67401
2.	5.08688	34.68301	5.78054	5.77398
2.5	5.12059	33.45378	5.95693	5.97390

$$\Psi(w) = E(NORS/X) + w \cdot E(SDO)$$

$$\tilde{\Psi}(w) = 4.97427 + 39.98540(w)$$

5.00

4.98

4.96

4.94

4.92

4.90

4.88

4.86

4.84

4.82

4.80

E(NORS/X)

8.5

Table 5.2

Relationships



• w = 4%

Budget = \$3,635,906

$$\tilde{E}(NORS/X) = -0.03148E(SDO/X) + 5.77638$$

w = 2%

0

26

27

28

29

30

31

E(SDO/X)

Figure 5.2. Plot of E(NORS/X) vs. E(SDO/X) from Table 5.2

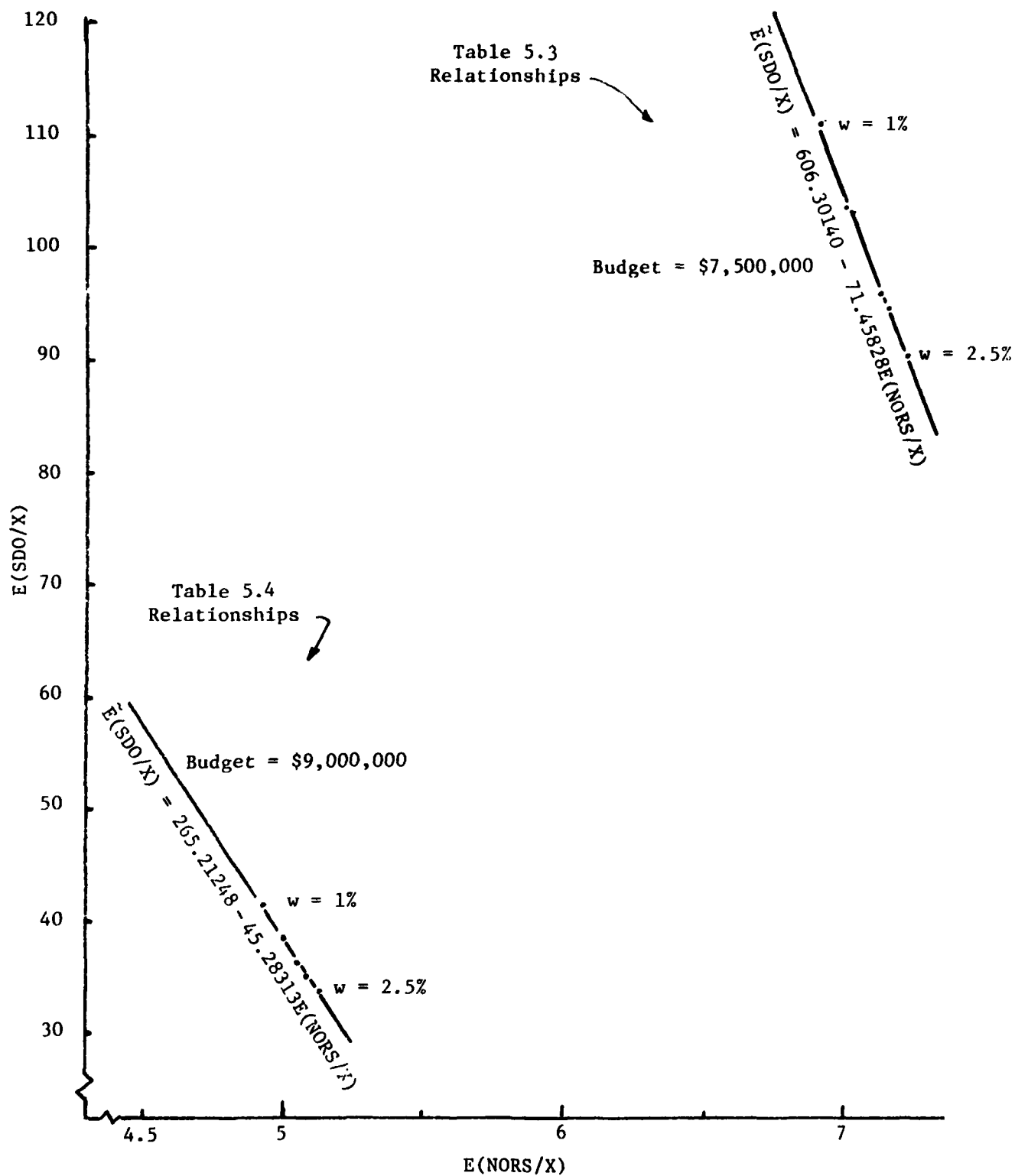


Figure 5.3. Plot of $E(SDO/X)$ vs. $E(NORS/X)$ from Tables 5.3 and 5.4

Comparison of the Solutions

Comparison of solutions was undertaken relative to the two phases of the WRSK Problem. F4D Data was used as sample data, and is provided in Appendix C. Three sets of solutions are shown in Figures 5.4 a, b, 5.5 a, b, and 5.6 a, b. They are the Phase I and Phase II output results of the Greedy Algorithm, the IBM version of the D029 Program and the D029-CDC Program of WPAFLC. The FORTRAN listings of the first two algorithms are provided in Appendix B.2 and C.2, and are the final products of this current research. It should be understood that only limited experimentation has taken place with both of these programs.

The frequency of the differences in item quantities can be of interest in discussing the kits obtained by these algorithms. In Phase I, there are 180 items under consideration. Table 5.5 indicates that 73% of the kit items (for the three comparisons) have the same quantity values with a standard deviation of 1.4%. Also the maximum difference between the individual kit items does not exceed 2. One can also note, for example, that the Greedy Algorithm kit has 40 and 43 less parts in the kit compared to the D029 solutions respectively.

In Phase II, there are 254 items under consideration. Table 5.6 shows the frequency of the discrepancy of the kits. 75% of the individual items in the kits have the same quantity values with a standard deviation of 3%. This indicates that the kits are in close agreement. In this case, the Greedy Algorithm kit has 19 and 18 less parts in the kit compared to the D029 solutions respectively. However, note the difference of nine that occurs in column 1 and column 3.

This difference occurs on item number 249 in Phase II. WPAFLC's D029-CDC Program (Figure 5.6b) yields $x_{249}^* = 1$, on the other hand the Greedy Algorithm

TABLE 5.5

FREQUENCY OF DIFFERENCES OF KITS IN PHASE I

Difference	Greedy-D029 CDC	Greedy-D029 IBM	D029 IBM-D029 CDC
-2	8	-	6
-1	31	46	14
0	134	131	129
+1	7	3	31
Net Δ	-40	-43	+5

TABLE 5.6

FREQUENCY OF DIFFERENCES OF KITS IN PHASE II

Difference	Greedy-D029 CDC	Greedy-D029 IBM	D029 IBM-D029 CDC
-3	2	-	-
-2	3	3	4
-1	34	37	33
0	196	195	183
1	18	19	30
2	-	3	3
9	1	-	1
Net Δ	-19	-18	+4

(Figure 5.4b) and the IBM Version of the D029 Program (Figure 5.5b) both produce $x_{249}^*=10$. This item has a moderate demand rate, $\mu_{249}=7.29728$, is not expensive at \$653, but has, however, a high number, 15, of applications on an aircraft (Appendix D.2.2). Since $\mu_{249}=7.29728$, probability of demand not exceeding 1 is $P(d \leq 1) = 0.0056$, but probability of demand not exceeding 10 is $P(d \leq 10) = .878$. Addition of 9 more items increases the probability considerably, thus providing more protection from back ordering item 249. The same reasoning applies to other items which have different kit values. If the difference in item quantities is small, the probabilities are close to each other. Investigation has not been undertaken to determine why the CDC version yields $X_{249}^*=1$. Different approaches to calculating the Poisson probabilities may be a contributing factor.

The Greedy Algorithm solution was obtained in 16.53 seconds of CPU time while the D029-IBM solution was obtained in 24.78 seconds of CPU time. This indicates the potential of a significant improvement in computational time if the Greedy Algorithm is coded for a CDC FORTRAN compiler and used rather than the D029-CDC program currently in use.

On Figure 5.6b it can be noted that the D029-CDC program has obtained a kit with $E(NORS) = 8.127$, $E(SDO) = 149.65$, and cost = \$7,108,451.15. This cost was used as a budget parameter for the Greedy Algorithm solution which is shown on Figure 5.4b. The solution obtained was $E(NORS) = 8.037$, $E(SDO) = 144.96$, and cost = \$7,108,477. For \$4 less expenditure of budget funds, the Greedy Algorithm obtained a kit yielding better performance measures than the D029-CDC solution.

Figure 5.5b illustrates a D029-IBM solution for the same budget for \$7,562,983 used by the D029-CDC solution (Figure 5.6b). In this solution

$E(\text{NORS}) = 8.130$, $E(\text{SDO}) = 145.30$ and cost = \$7,007,734. The budget is not utilized to its full potential. The objective of the D029 approach (Part IV, page 4.22) is to meet the NORS and SDO goals in a way that minimizes cost, and under utilization of a budget can result since the actual budget is the last item of priority that is checked in the algorithmic scheme of the D029 WRSK solution method.

The Greedy solution exhibited in Figure 5.7b for the same budget as in Figures 5.5b and 5.6b (\$7,562,983) yields $E(\text{NORS}) = 7.158$, $E(\text{SDO}) = 96.81$, and cost = \$7,562,944. All but \$39 of the available budget is utilized. These greatly improved performance measures are as expected since more money has gone into the kit.

The WRSK Model solved by the Greedy Algorithm (Part IV, page 4.1) is not the WRSK Model solved by the D029 Algorithm. For this reason it is difficult at times to make direct comparisons between D029 solutions and Greedy solutions. It should be recognized again that, in essence, the goals (constraints) to be satisfied by a D029 solution, are the elements of the objective function (which is to be minimized) of the Greedy model. In like manner, the Greedy model budget constraint, which is not to be exceeded, is the objective function to be minimized in the D029 model.

Figure 5.8b illustrates an attempt to develop a Greedy solution that is relatively comparable to the D029-IBM solution of Figure 5.5b. Observe that a budget of \$7,075,000 was used leading to a solution in which $E(\text{NORS}) = 8.173$, $E(\text{SDO}) = 145.09$, and cost = \$7,074,907 which is within 1% of the D029-IBM solution of Figure 5.5b. Since two different models are being used, answers exactly equal to each other are not be expected. However, it might be hypothesized that a closer stopping criteria on the b^*j values might continue to improve the Greedy solution.

All of the algorithms contain some heuristic rules; therefore the answers exhibited in this section are best categorized as near optimal solutions. Both algorithms yield satisfactory answers, but with differences discussed in this Part V and in Part VI.

	INITIAL VALUES OF PARAMETERS θ				
θ_1	0.30020210	3.003286799	3.26779460	0.25250234	3.46575927
θ_2	0.58345067	0.99151443	0.95573534	0.95789912	0.68018747
θ_3	0.99099167	0.99959453	0.99959459	0.95551969	0.9977739
θ_4				1.00000000	0.81107932
θ_5					0.89285344
θ_6					0.99989591
θ_7					0.9995505
θ_8					0.92100090
θ_9					0.96842135
θ_{10}					0.95998044

GREEDY ALGORITHM

PAGE 1

INITIAL KIT COMPUTATION

100 ITEMS IN COMPUTATION

KIT 5718

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

TARGETS **PERFORMANCE CF 5710 KIT**

COST CF 5710 KIT=8 4039896.00 NORSGAL= 6.92074 SOGGAL= 24.49522

BUDGET OF INITIAL KIT IS \$ 3635906.00

$PRICE(X) = 3639861.00$ $E(NORS/X) = 4.84982$ $E(SDO/X) = 29.47310$

NUMBER OF ITERATIONS 3

	B (J)	C	00000000	0.0017250	0.0644103	0.2611523	0.4989358	0.6927679	0.8243593	0.9067255	C.9503688	0.9749286
*	0.9572442	0.9941052	C.9672240	0.9987159	0.9564120	0.9067300	0.9998845	0.9599495	0.9997868	0.9599018		
b	0.9595634	0.9995982	0.9959942	0.9995677	0.9599551							

WEIGHT : ~~0.003~~ 0.0225

INITIAL KIT

00-0000
0000-00
00-1-00
0000-10
00-1-0
0-1-1-1
10000-1
000-00
0-00-10
0000-00
0000-10
1-1-0-0
1-1-0-1
0-1-0-1
0000-11
000-00
00-0-0
0000-0
0000-0
0-1-00
0000-11
000-0-0
000-0-0
0-1-00

Figure 5.4a Greedy Algorithm Program-A Phase 1 Solution for Kit 5713

INITIAL VALUES OF PARAMETERS		PARAMETERS	
α_0	0.3	0.0030001	0.0001449
α_1	0.6453770	0.5769561	0.97688197
α_2	0.0057607	0.55088191	0.99959153
β_1			0.99957163
β_2			0.20751657
β_3			0.9933729
β_4			0.27424124
β_5			0.23535479
β_6			0.9651327
β_7			0.99819555
β_8			0.44260776
β_9			0.63142239
β_{10}			0.95996509
β_{11}			0.77280201
β_{12}			0.99952452

2 36 WAD
2 36 WAD

TARGETS

NCESGAL : 0013547
SODGAL : 103092074
PRICE : 7562503.00

[illegible]

TOTAL NUMBER OF ITEMS UNDER CONSIDERATIONS 254

NUMBER OF ITERATIONS 2

NOTE: Forced Budget = \$7,108,451

$E(MCPS/X) =$	0.03740	$E(SCO/X) =$	140.56469	$PRICE(X) =$	710647.00
---------------	---------	--------------	-----------	--------------	-----------

OPTIMAL KIT

[illegible][illegible]

WGT : 9.013

Figure 5.4b Greedy Algorithm Program-A Phase 2 Solution: Budget = \$7,108,451

5.17

***** CPU = 24.789 SEC.

5.18

INITIAL D029 KIT - CUST 3 3665930.12 EXCLUDING NON-OPTIMIZED ITEMS
 LAST ITEM ADDED WAS 157, COST \$ 41566. Phase 1

ITEM	QUANTITY	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
ITEM	QUANTITY	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
ITEM	QUANTITY	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
ITEM	QUANTITY	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
ITEM	QUANTITY	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
ITEM	QUANTITY	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
ITEM	QUANTITY	126	127	128	129	130	131	132	133	134	135	136	137	138	139	140	141	142	143	144	145	146	147	148	149	150
ITEM	QUANTITY	151	152	153	154	155	156	157	158	159	160	161	162	163	164	165	166	167	168	169	170	171	172	173	174	175
ITEM	QUANTITY	176	177	178	179	180																				

CALCULATION OF EXPECTED NMCS AND SDO BY PROGRAM
 THE EXPECTED STOCK DUE OUTS ARE 29.4830498954
 THE EXPECTED NMCS ARE 4.456314621866

CALCULATION OF EXPECTED NMCS AND SDO BY ACCURATE FORMULA

PROBABILITY OF M AIRCRAFT DOWN - FORMULA (IN: PROG) 2

ITEM	QUANTITY	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
ITEM	QUANTITY	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
ITEM	QUANTITY	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
ITEM	QUANTITY	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
ITEM	QUANTITY	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
ITEM	QUANTITY	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
ITEM	QUANTITY	126	127	128	129	130	131	132	133	134	135	136	137	138	139	140	141	142	143	144	145	146	147	148	149	150
ITEM	QUANTITY	151	152	153	154	155	156	157	158	159	160	161	162	163	164	165	166	167	168	169	170	171	172	173	174	175
ITEM	QUANTITY	176	177	178	179	180																				

EXPECTED VALUES
 STOCK DUE OUTS 20.5643494194
 NMCS AIRCRAFT 4.856234164731
 SDO DEV OF NMCS 1.900014697496
 NEW NMCS 1.489170265463
 AVAILABILITY 298307449453
 NO. OF UNAVAIL. AC 16.84462121312

MRK ITEM ADDED TO KIT: ITEM 158, COST \$ 41566.
 MRR ITEM ADDED TO KIT: ITEM 142, COST \$ 13000.
 MASS 1 DONE - 17.205 SECONDS

Figure 5.6a D029-CDC Program-A Phase 1 Solution for Kit 5718 Budget

OPTIMAL D029 KIT - COST \$ 7100451.15 EXCLUDING NON-OPTIMIZED ITEMS		Phase 2	
LAST ITEM ADDED WAS 100, COST \$ 4000.			
ITEM	QUANTITY	1	2
ITEM	QUANTITY	2	3
ITEM	QUANTITY	3	4
ITEM	QUANTITY	4	5
ITEM	QUANTITY	5	6
ITEM	QUANTITY	6	7
ITEM	QUANTITY	7	8
ITEM	QUANTITY	8	9
ITEM	QUANTITY	9	10
ITEM	QUANTITY	10	11
ITEM	QUANTITY	11	12
ITEM	QUANTITY	12	13
ITEM	QUANTITY	13	14
ITEM	QUANTITY	14	15
ITEM	QUANTITY	15	16
ITEM	QUANTITY	16	17
ITEM	QUANTITY	17	18
ITEM	QUANTITY	18	19
ITEM	QUANTITY	19	20
ITEM	QUANTITY	20	21
ITEM	QUANTITY	21	22
ITEM	QUANTITY	22	23
ITEM	QUANTITY	23	24
ITEM	QUANTITY	24	25
ITEM	QUANTITY	25	26
ITEM	QUANTITY	26	27
ITEM	QUANTITY	27	28
ITEM	QUANTITY	28	29
ITEM	QUANTITY	29	30
ITEM	QUANTITY	30	31
ITEM	QUANTITY	31	32
ITEM	QUANTITY	32	33
ITEM	QUANTITY	33	34
ITEM	QUANTITY	34	35
ITEM	QUANTITY	35	36
ITEM	QUANTITY	36	37
ITEM	QUANTITY	37	38
ITEM	QUANTITY	38	39
ITEM	QUANTITY	39	40
ITEM	QUANTITY	40	41
ITEM	QUANTITY	41	42
ITEM	QUANTITY	42	43
ITEM	QUANTITY	43	44
ITEM	QUANTITY	44	45
ITEM	QUANTITY	45	46
ITEM	QUANTITY	46	47
ITEM	QUANTITY	47	48
ITEM	QUANTITY	48	49
ITEM	QUANTITY	49	50
ITEM	QUANTITY	50	51
ITEM	QUANTITY	51	52
ITEM	QUANTITY	52	53
ITEM	QUANTITY	53	54
ITEM	QUANTITY	54	55
ITEM	QUANTITY	55	56
ITEM	QUANTITY	56	57
ITEM	QUANTITY	57	58
ITEM	QUANTITY	58	59
ITEM	QUANTITY	59	60
ITEM	QUANTITY	60	61
ITEM	QUANTITY	61	62
ITEM	QUANTITY	62	63
ITEM	QUANTITY	63	64
ITEM	QUANTITY	64	65
ITEM	QUANTITY	65	66
ITEM	QUANTITY	66	67
ITEM	QUANTITY	67	68
ITEM	QUANTITY	68	69
ITEM	QUANTITY	69	70
ITEM	QUANTITY	70	71
ITEM	QUANTITY	71	72
ITEM	QUANTITY	72	73
ITEM	QUANTITY	73	74
ITEM	QUANTITY	74	75
ITEM	QUANTITY	75	76
ITEM	QUANTITY	76	77
ITEM	QUANTITY	77	78
ITEM	QUANTITY	78	79
ITEM	QUANTITY	79	80
ITEM	QUANTITY	80	81
ITEM	QUANTITY	81	82
ITEM	QUANTITY	82	83
ITEM	QUANTITY	83	84
ITEM	QUANTITY	84	85
ITEM	QUANTITY	85	86
ITEM	QUANTITY	86	87
ITEM	QUANTITY	87	88
ITEM	QUANTITY	88	89
ITEM	QUANTITY	89	90
ITEM	QUANTITY	90	91
ITEM	QUANTITY	91	92
ITEM	QUANTITY	92	93
ITEM	QUANTITY	93	94
ITEM	QUANTITY	94	95
ITEM	QUANTITY	95	96
ITEM	QUANTITY	96	97
ITEM	QUANTITY	97	98
ITEM	QUANTITY	98	99
ITEM	QUANTITY	99	100
ITEM	QUANTITY	100	101
ITEM	QUANTITY	101	102
ITEM	QUANTITY	102	103
ITEM	QUANTITY	103	104
ITEM	QUANTITY	104	105
ITEM	QUANTITY	105	106
ITEM	QUANTITY	106	107
ITEM	QUANTITY	107	108
ITEM	QUANTITY	108	109
ITEM	QUANTITY	109	110
ITEM	QUANTITY	110	111
ITEM	QUANTITY	111	112
ITEM	QUANTITY	112	113
ITEM	QUANTITY	113	114
ITEM	QUANTITY	114	115
ITEM	QUANTITY	115	116
ITEM	QUANTITY	116	117
ITEM	QUANTITY	117	118
ITEM	QUANTITY	118	119
ITEM	QUANTITY	119	120
ITEM	QUANTITY	120	121
ITEM	QUANTITY	121	122
ITEM	QUANTITY	122	123
ITEM	QUANTITY	123	124
ITEM	QUANTITY	124	125
ITEM	QUANTITY	125	126
ITEM	QUANTITY	126	127
ITEM	QUANTITY	127	128
ITEM	QUANTITY	128	129
ITEM	QUANTITY	129	130
ITEM	QUANTITY	130	131
ITEM	QUANTITY	131	132
ITEM	QUANTITY	132	133
ITEM	QUANTITY	133	134
ITEM	QUANTITY	134	135
ITEM	QUANTITY	135	136
ITEM	QUANTITY	136	137
ITEM	QUANTITY	137	138
ITEM	QUANTITY	138	139
ITEM	QUANTITY	139	140
ITEM	QUANTITY	140	141
ITEM	QUANTITY	141	142
ITEM	QUANTITY	142	143
ITEM	QUANTITY	143	144
ITEM	QUANTITY	144	145
ITEM	QUANTITY	145	146
ITEM	QUANTITY	146	147
ITEM	QUANTITY	147	148
ITEM	QUANTITY	148	149
ITEM	QUANTITY	149	150
ITEM	QUANTITY	150	151
ITEM	QUANTITY	151	152
ITEM	QUANTITY	152	153
ITEM	QUANTITY	153	154
ITEM	QUANTITY	154	155
ITEM	QUANTITY	155	156
ITEM	QUANTITY	156	157
ITEM	QUANTITY	157	158
ITEM	QUANTITY	158	159
ITEM	QUANTITY	159	160
ITEM	QUANTITY	160	161
ITEM	QUANTITY	161	162
ITEM	QUANTITY	162	163
ITEM	QUANTITY	163	164
ITEM	QUANTITY	164	165
ITEM	QUANTITY	165	166
ITEM	QUANTITY	166	167
ITEM	QUANTITY	167	168
ITEM	QUANTITY	168	169
ITEM	QUANTITY	169	170
ITEM	QUANTITY	170	171
ITEM	QUANTITY	171	172
ITEM	QUANTITY	172	173
ITEM	QUANTITY	173	174
ITEM	QUANTITY	174	175
ITEM	QUANTITY	175	176
ITEM	QUANTITY	176	177
ITEM	QUANTITY	177	178
ITEM	QUANTITY	178	179
ITEM	QUANTITY	179	180
ITEM	QUANTITY	180	181
ITEM	QUANTITY	181	182
ITEM	QUANTITY	182	183
ITEM	QUANTITY	183	184
ITEM	QUANTITY	184	185
ITEM	QUANTITY	185	186
ITEM	QUANTITY	186	187
ITEM	QUANTITY	187	188
ITEM	QUANTITY	188	189
ITEM	QUANTITY	189	190
ITEM	QUANTITY	190	191
ITEM	QUANTITY	191	192
ITEM	QUANTITY	192	193
ITEM	QUANTITY	193	194
ITEM	QUANTITY	194	195
ITEM	QUANTITY	195	196
ITEM	QUANTITY	196	197
ITEM	QUANTITY	197	198
ITEM	QUANTITY	198	199
ITEM	QUANTITY	199	200
ITEM	QUANTITY	200	201
ITEM	QUANTITY	201	202
ITEM	QUANTITY	202	203
ITEM	QUANTITY	203	204
ITEM	QUANTITY	204	205
ITEM	QUANTITY	205	206
ITEM	QUANTITY	206	207
ITEM	QUANTITY	207	208
ITEM	QUANTITY	208	209
ITEM	QUANTITY	209	210
ITEM	QUANTITY	210	211
ITEM	QUANTITY	211	212
ITEM	QUANTITY	212	213
ITEM	QUANTITY	213	214
ITEM	QUANTITY	214	215
ITEM	QUANTITY	215	216
ITEM	QUANTITY	216	217
ITEM	QUANTITY	217	218
ITEM	QUANTITY	218	219
ITEM	QUANTITY	219	220
ITEM	QUANTITY	220	221
ITEM	QUANTITY	221	222
ITEM	QUANTITY	222	223
ITEM	QUANTITY	223	224
ITEM	QUANTITY	224	225
ITEM	QUANTITY	225	226
ITEM	QUANTITY	226	227
ITEM	QUANTITY	227	228
ITEM	QUANTITY	228	229
ITEM	QUANTITY	229	230
ITEM	QUANTITY	230	231
ITEM	QUANTITY	231	232
ITEM	QUANTITY	232	233
ITEM	QUANTITY	233	234
ITEM	QUANTITY	234	235
ITEM	QUANTITY	235	236
ITEM	QUANTITY	236	237
ITEM	QUANTITY	237	238
ITEM	QUANTITY	238	239
ITEM	QUANTITY	239	240
ITEM	QUANTITY	240	241
ITEM	QUANTITY	241	242
ITEM	QUANTITY	242	243
ITEM	QUANTITY	243	244
ITEM	QUANTITY	244	245
ITEM	QUANTITY	245	246
ITEM	QUANTITY	246	247
ITEM	QUANTITY	247	248
ITEM	QUANTITY	248	249
ITEM	QUANTITY	249	250
ITEM	QUANTITY	250	251
ITEM	QUANTITY	251	252
ITEM	QUANTITY	252	253
ITEM	QUANTITY	253	254
ITEM	QUANTITY	254	255

CALCULATION OF EXPECTED NMCS AND SDO BY PROGRAM
THE EXPECTED STOCK DUE OUTS ARE 152,663,084,983
THE EXPECTED NMCS ARE 8,127,570,556,005

CALCULATION OF EXPECTED NMCS AND SDO BY ACCURATE FORMULA

PROBABILITY OF N AIRCRAFT DOWN - FORMAT (N, PROBI)
0 3.20588077519E-45
1 1.02797876265E-14
2 1.02797876265E-14
3 1.02797876265E-14
4 1.02797876265E-14
5 1.02797876265E-14
6 1.02797876265E-14
7 1.02797876265E-14
8 1.02797876265E-14
9 1.02797876265E-14
10 1.02797876265E-14
11 1.02797876265E-14
12 1.02797876265E-14
13 1.02797876265E-14
14 1.02797876265E-14
15 1.02797876265E-14
16 1.02797876265E-14
17 1.02797876265E-14
18 1.02797876265E-14
19 1.02797876265E-14
20 1.02797876265E-14
21 1.02797876265E-14
22 1.02797876265E-14
23 1.02797876265E-14
24 1.02797876265E-14
25 1.02797876265E-14
26 1.02797876265E-14
27 1.02797876265E-14
28 1.02797876265E-14
29 1.02797876265E-14
30 1.02797876265E-14
31 1.02797876265E-14
32 1.02797876265E-14
33 1.02797876265E-14
34 1.02797876265E-14
35 1.02797876265E-14
36 1.02797876265E-14
37 1.02797876265E-14
38 1.02797876265E-14
39 1.02797876265E-14
40 1.02797876265E-14
41 1.02797876265E-14
42 1.02797876265E-14
43 1.02797876265E-14
44 1.02797876265E-14
45 1.02797876265E-14
46 1.02797876265E-14
47 1.02797876265E-14
48 1.02797876265E-14
49 1.02797876265E-14
50 1.02797876265E-14
51 1.02797876265E-14
52 1.02797876265E-14
53 1.02797876265E-14
54 1.02797876265E-14
55 1.02797876265E-14
56 1.02797876265E-14
57 1.02797876265E-14
58 1.02797876265E-14
59 1.02797876265E-14
60 1.02797876265E-14
61 1.02797876265E-14
62 1.02797876265E-14
63 1.02797876265E-14
64 1.02797876265E-14
65 1.02797876265E-14
66 1.02797876265E-14
67 1.02797876265E-14
68 1.02797876265E-14
69 1.02797876265E-14
70 1.02797876265E-14
71 1.02797876265E-14
72 1.02797876265E-14
73 1.02797876265E-14
74 1.02797876265E-14
75 1.02797876265E-14
76 1.02797876265E-14
77 1.02797876265E-14
78 1.02797876265E-14
79 1.02797876265E-14
80 1.02797876265E-14
81 1.02797876265E-14
82 1.02797876265E-14
83 1.02797876265E-14
84 1.02797876265E-14
85 1.02797876265E-14
86 1.02797876265E-14
87 1.02797876265E-14
88 1.02797876265E-14
89 1.02797876265E-14
90 1.02797876265E-14
91 1.02797876265E-14
92 1.02797876265E-14
93 1.02797876265E-14
94 1.02797876265E-14
95 1.02797876265E-14
96 1.02797876265E-14
97 1.02797876265E-14
98 1.02797876265E-14
99 1.02797876265E-14
100 1.02797876265E-14
101 1.02797876265E-14
102 1.02797876265E-14
103 1.02797876265E-14
104 1.02797876265E-14
105 1.02797876265E-14
106 1.02797876265E-14
107 1.02797876265E-14
108 1.02797876265E-14
109 1.02797876265E-14
110 1.02797876265E-14
111 1.02797876265E-14
112 1.02797876265E-14
113 1.02797876265E-14
114 1.02797876265E-14
115 1.02797876265E-14
116 1.02797876265E-14
117 1.02797876265E-14
118 1.02797876265E-14
119 1.02797876265E-14
120 1.02797876265E-14
121 1.02797876265E-14
122 1.02797876265E-14
123 1.02797876265E-14
124 1.02797876265E-14
125 1.02797876265E-14
126 1.02797876265E-14
127 1.02797876265E-14
128 1.02797876265E-14
129 1.02797876265E-14
130 1.02797876265E-14
131 1.02797876265E-14
132 1.02797876265E-14
133 1.02797876265E-14
134 1.02797876265E-14
135 1.02797876265E-14
136 1.02797876265E-14
137 1.02797876265E-14
138 1.02797876265E-14
139 1.02797876265E-14
140 1.02797876265E-14
141 1.02797876265E-14
142 1.02797876265E-14
143 1.02797876265E-14
144 1.02797876265E-14
145 1.02797876265E-14
146 1.02797876265E-14
147 1.02797876265E-14
148 1.02797876265E

INITIAL VALUES OF PARAMETERS B			
α	0.0000001C	0.00266799	0.00779450
β	0.00834567	0.99151463	0.99375336
γ	0.0000001B	0.99999987	0.99999963
δ	0.25355234	0.40975927	0.61107932
ϵ	0.91789912	0.99089552	0.99977739
ζ	0.99395544	1.00003400	0.99991554
η	1.09205344	0.99990491	0.99995505
θ	0.99100090	0.99990049	0.99990049

GREEDY ALGORITHM

PHASE 1

INITIAL & IV COMPUTATION

001 08317 MY COMPUTATION

AT 45 11M

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 99 100 101 102 103 104 105 106 107 108 109 110 111 112 113 114 115 116 117 118 119 120 121 122 123 124 125 126 127 128 129 130 131 132 133 134 135 136 137 138 139 140 141 142 143 144 145 146 147 148 149 150 151 152 153 154 155 156 157 158 159 160 161 162 163 164 165 166 167 168 169 170 171 172 173 174 175 176 177 178 179 180 181 182 183 184 185 186 187 188 189 190 191 192 193 194 195 196 197 198 199 200 201 202 203 204 205 206 207 208 209 210 211 212 213 214 215 216 217 218 219 220 221 222 223 224 225 226 227 228 229 230 231 232 233 234 235 236 237 238 239 240 241 242 243 244 245 246 247 248 249 250 251 252 253 254 255 256 257 258 259 260 261 262 263 264 265 266 267 268 269 270 271 272 273 274 275 276 277 278 279 280 281 282 283 284 285 286 287 288 289 290 291 292 293 294 295 296 297 298 299 300 301 302 303 304 305 306 307 308 309 310 311 312 313 314 315 316 317 318 319 320 321 322 323 324 325 326 327 328 329 330 331 332 333 334 335 336 337 338 339 340 341 342 343 344 345 346 347 348 349 350 351 352 353 354 355 356 357 358 359 360 361 362 363 364 365 366 367 368 369 370 371 372 373 374 375 376 377 378 379 380 381 382 383 384 385 386 387 388 389 390 391 392 393 394 395 396 397 398 399 400 401 402 403 404 405 406 407 408 409 410 411 412 413 414 415 416 417 418 419 420 421 422 423 424 425 426 427 428 429 430 431 432 433 434 435 436 437 438 439 440 441 442 443 444 445 446 447 448 449 450 451 452 453 454 455 456 457 458 459 460 461 462 463 464 465 466 467 468 469 470 471 472 473 474 475 476 477 478 479 480 481 482 483 484 485 486 487 488 489 490 491 492 493 494 495 496 497 498 499 500 501 502 503 504 505 506 507 508 509 510 511 512 513 514 515 516 517 518 519 520 521 522 523 524 525 526 527 528 529 530 531 532 533 534 535 536 537 538 539 540 541 542 543 544 545 546 547 548 549 550 551 552 553 554 555 556 557 558 559 560 561 562 563 564 565 566 567 568 569 570 571 572 573 574 575 576 577 578 579 580 581 582 583 584 585 586 587 588 589 590 591 592 593 594 595 596 597 598 599 600 601 602 603 604 605 606 607 608 609 610 611 612 613 614 615 616 617 618 619 620 621 622 623 624 625 626 627 628 629 630 631 632 633 634 635 636 637 638 639 640 641 642 643 644 645 646 647 648 649 650 651 652 653 654 655 656 657 658 659 660 661 662 663 664 665 666 667 668 669 670 671 672 673 674 675 676 677 678 679 680 681 682 683 684 685 686 687 688 689 690 691 692 693 694 695 696 697 698 699 700 701 702 703 704 705 706 707 708 709 710 711 712 713 714 715 716 717 718 719 720 721 722 723 724 725 726 727 728 729 730 731 732 733 734 735 736 737 738 739 740 741 742 743 744 745 746 747 748 749 750 751 752 753 754 755 756 757 758 759 760 761 762 763 764 765 766 767 768 769 770 771 772 773 774 775 776 777 778 779 780 781 782 783 784 785 786 787 788 789 790 791 792 793 794 795 796 797 798 799 800 801 802 803 804 805 806 807 808 809 810 811 812 813 814 815 816 817 818 819 820 821 822 823 824 825 826 827 828 829 830 831 832 833 834 835 836 837 838 839 840 841 842 843 844 845 846 847 848 849 850 851 852 853 854 855 856 857 858 859 860 861 862 863 864 865 866 867 868 869 870 871 872 873 874 875 876 877 878 879 880 881 882 883 884 885 886 887 888 889 890 891 892 893 894 895 896 897 898 899 900 901 902 903 904 905 906 907 908 909 910 911 912 913 914 915 916 917 918 919 920 921 922 923 924 925 926 927 928 929 930 931 932 933 934 935 936 937 938 939 940 941 942 943 944 945 946 947 948 949 950 951 952 953 954 955 956 957 958 959 960 961 962 963 964 965 966 967 968 969 970 971 972 973 974 975 976 977 978 979 980 981 982 983 984 985 986 987 988 989 990 991 992 993 994 995 996 997 998 999 1000 1001 1002 1003 1004 1005 1006 1007 1008 1009 1010 1011 1012 1013 1014 1015 1016 1017 1018 1019 1020 1021 1022 1023 1024 1025 1026 1027 1028 1029 1030 1031 1032 1033 1034 1035 1036 1037 1038 1039 104

TARGETS PERFORMANCE OF 5718 KII

COST CF 5710 KIT=8 4039896.00 4.92074 5006JAL= 24.49522

BUDGET OF INITIAL KIT IS \$ 3635903.00

PRICE(A)=	3035861.00	E(NORS/X)=	0.84902	E(50/X)=	29.47310
-----------	------------	------------	---------	----------	----------

NUMBER OF ITERATIONS 3

a*	81.1	6.8000000	0.3617250	0.0644401	0.5811572	0.4989385	0.6927672	0.8233903	0.507255	0.503808	0.374928
b*	0.997242	0.9941652	0.9972461	0.9471559	0.994120	0.5988456	0.9997486	0.9997486	0.9999192		
	0.9999834	0.9999882	0.9999942	0.9999977	0.9999991						

WEIGHT : 0.023

INITIAL XIF

[illegible]

Figure 5.7a Phase I Greedy Algorithm Solution: Budget of Kit 5718

[illegible]

GREEDY ALGORITHM
PHASE 2

TARGETS

NORSGOAL : 0.1354
SDGOAL : 163.32674
PRICE=5
7562983.00

[illegible]

TOTAL NUMBER OF ITEMS UNDER CONSIDERATIONS :54

NUMBER OF ITERATIONS •

NOTE: Budget (ls for kit 5718) = \$7,562,983

PRICE/KG	7562944.00
21500/KG	16.61477
7.15853	

OPTIMAL K17

[illegible]

	0.0000000	0.0000035	0.0000070	0.0000105	0.0000140	0.0000175	0.0000210	0.0000245	0.0000280	0.0000315	0.0000350	0.0000385	0.0000420	0.0000455	0.0000490	0.0000525	0.0000560	0.0000595	0.0000630	0.0000665	0.0000700	0.0000735	0.0000770	0.0000805	0.0000840	0.0000875	0.0000910	0.0000945	0.0000980	0.0001015	0.0001050	0.0001085	0.0001120	0.0001155	0.0001190	0.0001225	0.0001260	0.0001295	0.0001330	0.0001365	0.0001400	0.0001435	0.0001470	0.0001505	0.0001540	0.0001575	0.0001610	0.0001645	0.0001680	0.0001715	0.0001750	0.0001785	0.0001820	0.0001855	0.0001890	0.0001925	0.0001960	0.0001995	0.0002030	0.0002065	0.0002100	0.0002135	0.0002170	0.0002205	0.0002240	0.0002275	0.0002310	0.0002345	0.0002380	0.0002415	0.0002450	0.0002485	0.0002520	0.0002555	0.0002590	0.0002625	0.0002660	0.0002695	0.0002730	0.0002765	0.0002800	0.0002835	0.0002870	0.0002905	0.0002940	0.0002975	0.0003010	0.0003045	0.0003080	0.0003115	0.0003150	0.0003185	0.0003220	0.0003255	0.0003290	0.0003325	0.0003360	0.0003395	0.0003430	0.0003465	0.0003500	0.0003535	0.0003570	0.0003605	0.0003640	0.0003675	0.0003710	0.0003745	0.0003780	0.0003815	0.0003850	0.0003885	0.0003920	0.0003955	0.0003990	0.0004025	0.0004060	0.0004095	0.0004130	0.0004165	0.0004200	0.0004235	0.0004270	0.0004305	0.0004340	0.0004375	0.0004410	0.0004445	0.0004480	0.0004515	0.0004550	0.0004585	0.0004620	0.0004655	0.0004690	0.0004725	0.0004760	0.0004795	0.0004830	0.0004865	0.0004900	0.0004935	0.0004970	0.0005005	0.0005040	0.0005075	0.0005110	0.0005145	0.0005180	0.0005215	0.0005250	0.0005285	0.0005320	0.0005355	0.0005390	0.0005425	0.0005460	0.0005495	0.0005530	0.0005565	0.0005600	0.0005635	0.0005670	0.0005705	0.0005740	0.0005775	0.0005810	0.0005845	0.0005880	0.0005915	0.0005950	0.0005985	0.0006020	0.0006055	0.0006090	0.0006125	0.0006160	0.0006195	0.0006230	0.0006265	0.0006300	0.0006335	0.0006370	0.0006405	0.0006440	0.0006475	0.0006510	0.0006545	0.0006580	0.0006615	0.0006650	0.0006685	0.0006720	0.0006755	0.0006790	0.0006825	0.0006860	0.0006895	0.0006930	0.0006965	0.0007000	0.0007035	0.0007070	0.0007105	0.0007140	0.0007175	0.0007210	0.0007245	0.0007280	0.0007315	0.0007350	0.0007385	0.0007420	0.0007455	0.0007490	0.0007525	0.0007560	0.0007595	0.0007630	0.0007665	0.0007700	0.0007735	0.0007770	0.0007805	0.0007840	0.0007875	0.0007910	0.0007945	0.0007980	0.0008015	0.0008050	0.0008085	0.0008120	0.0008155	0.0008190	0.0008225	0.0008260	0.0008295	0.0008330	0.0008365	0.0008400	0.0008435	0.0008470	0.0008505	0.0008540	0.0008575	0.0008610	0.0008645	0.0008680	0.0008715	0.0008750	0.0008785	0.0008820	0.0008855	0.0008890	0.0008925	0.0008960	0.0008995	0.0009030	0.0009065	0.0009100	0.0009135	0.0009170	0.0009205	0.0009240	0.0009275	0.0009310	0.0009345	0.0009380	0.0009415	0.0009450	0.0009485
--	-----------	-----------	-----------	-----------	-----------	-----------	-----------	-----------	-----------	-----------	-----------	-----------	-----------	-----------	-----------	-----------	-----------	-----------	-----------	-----------	-----------	-----------	-----------	-----------	-----------	-----------	-----------	-----------	-----------	-----------	-----------	-----------	-----------	-----------	-----------	-----------	-----------	-----------	-----------	-----------	-----------	-----------	-----------	-----------	-----------	-----------	-----------	-----------	-----------	-----------	-----------	-----------	-----------	-----------	-----------	-----------	-----------	-----------	-----------	-----------	-----------	-----------	-----------	-----------	-----------	-----------	-----------	-----------	-----------	-----------	-----------	-----------	-----------	-----------	-----------	-----------	-----------	-----------	-----------	-----------	-----------	-----------	-----------	-----------	-----------	-----------	-----------	-----------	-----------	-----------	-----------	-----------	-----------	-----------	-----------	-----------	-----------	-----------	-----------	-----------	-----------	-----------	-----------	-----------	-----------	-----------	-----------	-----------	-----------	-----------	-----------	-----------	-----------	-----------	-----------	-----------	-----------	-----------	-----------	-----------	-----------	-----------	-----------	-----------	-----------	-----------	-----------	-----------	-----------	-----------	-----------	-----------	-----------	-----------	-----------	-----------	-----------	-----------	-----------	-----------	-----------	-----------	-----------	-----------	-----------	-----------	-----------	-----------	-----------	-----------	-----------	-----------	-----------	-----------	-----------	-----------	-----------	-----------	-----------	-----------	-----------	-----------	-----------	-----------	-----------	-----------	-----------	-----------	-----------	-----------	-----------	-----------	-----------	-----------	-----------	-----------	-----------	-----------	-----------	-----------	-----------	-----------	-----------	-----------	-----------	-----------	-----------	-----------	-----------	-----------	-----------	-----------	-----------	-----------	-----------	-----------	-----------	-----------	-----------	-----------	-----------	-----------	-----------	-----------	-----------	-----------	-----------	-----------	-----------	-----------	-----------	-----------	-----------	-----------	-----------	-----------	-----------	-----------	-----------	-----------	-----------	-----------	-----------	-----------	-----------	-----------	-----------	-----------	-----------	-----------	-----------	-----------	-----------	-----------	-----------	-----------	-----------	-----------	-----------	-----------	-----------	-----------	-----------	-----------	-----------	-----------	-----------	-----------	-----------	-----------	-----------	-----------	-----------	-----------	-----------	-----------	-----------	-----------	-----------	-----------	-----------	-----------	-----------	-----------	-----------	-----------	-----------	-----------	-----------	-----------	-----------	-----------

WEIGHT : 0.020

Figure 5 7b Phase 2 Greedy Algorithm Solution: Budget of \$7,562,983

PART VI

SUMMARY OF RESULTS

This research provides a new and more general model, the "multiobjective model," for the WRSK problem. This multiobjective approach resulted in an improved measure of supply performance which relates $E(NORS/X)$ and $E(SDO/X)$ for a WRSK kit in a meaningful way.

Three potential solution algorithms for this new "multiobjective model" were investigated. The Saddle Point Algorithm was the most general, but suffered from the duality gap. The Primal Search Algorithm made special use of the properties of the objective function, but lacked flexibility to adjust the search direction in an effective way.

A third method proposed by R. B. S. Brooks in [3] and subsequently modified during this research lead to the development of the "Greedy Algorithm" as the solution procedure. This algorithm requires that the objective function be approximated by a separable function. This research has accomplished the required separability of the multiobjective model. This allows the use of marginal analysis in a manner that achieves the highest rate of return per unit of performance measure per dollar invested in the kit as items are added to the kit.

Computational experimentation has shown that the Greedy Algorithm is faster than the other two potential algorithms. An important result of this research has been that the Greedy Algorithm program is faster and provides a better solution than the current D029 using F4D data.

The Greedy Algorithm has several advantages. It requires only one ratio of marginal return be updated whenever one unit of an item is added

to a kit. This is an immediate consequence of having achieved a separable approximation of the objective function of the new multiobjective model. This contributes to the ability of the Greedy Algorithm to yield an improved kit compared to the D029-CDC kit, and in shorter time when properly compared to the D029-IBM program.

A disadvantage of the Greedy Algorithm is that it is constructed on the convergence of the b_j parameters. The generation of a new set of parameter values from a previous set takes one full iteration. The parameter values of this new set are compared to the previous set. Such iterations and comparisons are repeated until the parameters are sufficiently close, at which time the algorithm terminates. Therefore, the last iteration is used only to conclude that the optimal solution is as good as the next to the last solution. Thus there is always an extra iteration that does not improve the kit configuration. Another disadvantage might be that kits for intermediate budgets might be desired, but must await determination of the optimal b_j parameters. Also the weighting factor w must be determined in advance, and additional experimentation may be required to determine the appropriate value.

Another product of this research has been the modification of the current D029 program for compatibility with IBM FORTRAN compilers. This effort has provided some insight into the problems with the current D029 program. The word packing and unpacking required by the CDC compilers, in order to save memory, consumes a great deal of time. The storing of some probability values, and the interpolation of others, when needed, does not appear to be a good policy. Interpolation underestimates the probabilities; hence, the marginal rates of return are less accurate and lead to kit solutions not as good as those obtained by the Greedy Algorithm.

The storing of probabilities, and interpolation of others, does appear to possibly contribute to errors in a kit configuration. The Greedy Algorithm and the D029-IBM program produces $X_{249}^* = 10$, but the D029-CDC program produces $X_{249}^* = 1$. The source of this error should be understood and eliminated from the current D029-CDC program.

Comparison of kit results, indicates that the kits obtained by the Greedy Algorithm or the D029 program are similar. Approximately 75% of the kit items are present in the same amounts and the remainings deviate only by 1 or 2. This is confirmation of the recognition that the current D029 model is acceptable. The Greedy Algorithm appears to offer the potential of substantial time savings, and a slightly better kit configuration.

REFERENCES

1. Bellman, R. E. and S. E. Dreyfus, Applied Dynamic Programming, Princeton, New Jersey, Princeton University Press, 1962.
2. Brooks, R. B. S. and A. Geoffrion, "Finding Everett's Lagrange Multipliers by Linear Programming," Operations Research, Vol. 14, 1966 , pp. 1149-1152.
3. Brooks, R. B. S., C. A. Gillen, and J. Y. Lu, "Alternative Measures of Supply Performance," The RAND Corporation, RM-6094-PR, August, 1969.
4. Fox, B., "Discrete Optimization Via Marginal Analysis," Management Science, Vol. 13, 1966, pp. 210-216.
5. Garfinkel, R. S. and G. L Nemhauser, Integer Programming, New York, New York: John Wiley and Sons, Inc., 1972.
6. Miller, B. L., "Unconstrained Optimization in the Integers," RAND Corporation, RM-6165-PR, January, 1970.
7. Sherbrooke, C. C., "Metric: A Multi-Echelon Technique for Recoverable Item Control," Operations Research, Vol. 16, 1968, pp. 122-141.
8. W.P.A.F.L.C., Preliminary Evaluation of D029 WRSK Model, Technical Report, 1980.
9. Yuceer, U., "Optimization of Single Resource Allocation Problems with Nonseparable Objective Functions," Ph.D. dissertation, Oklahoma State University, December, 1980.

APPENDIX A

- A.1 Computer Code for the Primal Search Algorithm**
- A.2 Computer Code for the Modified Algorithm of Brooks, et al.**
- A.3 Computer Code for the Saddle Point Search Algorithm**

APPENDIX A.1

COMPUTER CODE FOR THE

PRIMAL SEARCH

ALGORITHM

A.1.1

A List of the Variables in the Computer Program

ALFA	Stepsize
BASLA	The Subroutine that finds an initial point
BETA	Product of the parameters p (user determined) and p_k
BUTCE	Budget
C(100)	Cost of item i
CBETA	The parameter p_k
CDOTD	The dot product of the cost and direction vectors
CDOTX	The dot product of the cost vector and the current solution, also the cost of the kit
CKARE	The dot product the cost vector with itself
COST	Cost of the current solution
DENOM	The dot product of the Cost vector the vector of first differences at a solution
DIF	The first difference
DIREC	The direction vector
ENORS	$E(NORS/X)$
EPS	Parameter for testing the sufficiently closeness
IOX	Integer Optimal Solution
ITETA	Indicator of the search direction. If $ITETA = 1$, the search is in the positive direction, if $ITETA = -1$, it is in the negative direction.
IX(100)	Current integer solution
IY(100)	Initial solution
KODE	Indicator of the result of the search, "1" means success, "0" means failure
N	Total number of items in the problem
NEWP	New integer point
NITER	Number of iterations already performed

NNORS	Number of NORS function evaluations
NSTEP (NITER)	Number of steps within iteration NITER
M	Upper bound
PAR	Parameter p (user determined)
PDFOP	Function routine which calculates the individual poisson terms
POISON (40,100)	The matrix which contains the cumulative Poisson sums
SIRALA (X,IR)	The subroutine which ranks the array X in descending order. IR(k) gives the rank of X(k).
TABLE (POISON)	The subroutine which fills in the matrix POISON
UCOS	Cost of NEWP
UNORS	$E(NORS/NEWP)$
VNORS	Function Routine to calculate $E(NORS/X)$ funtion
XLAG	Lagrange parameter (user determined)
XLAM(I)	Poisson demand rate of item I.

FORTRAN Listing of the Computer Program

```

1JCB TIME=05
C *****
C *
C *
C *   A PRIMAL SEARCH ALGORITHM
C *
C *
C *****
C
C   MAIN PROGRAM: TO READ IN THE DATA
C                 TO WRITE OUT THE SOLUTIONS
C
1  DIMENSION IX(100),IOX(100),IY(100),DELTA(100)
2  COMMON N,M/BLOCK1/C(100)/BLOCK2/XLAM(100)/BLOCK3/BUTCE,PAR/BLOCK4.
3  *NITER,NSTEP(500)/BLOCK5/POISON(40,100)/COUNT/NNORS
4  EPS=.07005
C  INPUT THE NECESSARY INFORMATION
C
4  NNORS=0
5  READ(5,90) NOPR
6  READ(5,100) N,M,BUTCE
7  READ(5,110) (C(I),I=1,N)
8  READ(5,110) (XLAM(I),I=1,N)
9  READ(5,120) PAR,XLAG
C
C  CALCULATE THE PROBABILITIES
C
10 CALL TABLE(POISON)
C
C  FIND AN INITIAL SOLUTION
11 CALL BASLA(XLAG,IY)
12 DO 10 I=1,N
13 10 IX(I)=IY(I)
C
C  START THE OPTIMIZATION
C
14 CALL OPTIM(IX,IOX,ENORS,COST)
C
C  PRINT THE RESULTS
C
15 WRITE(6,400)
16 WRITE(6,405) NOPR
17 WRITE(6,205) N,BUTCE
18 WRITE(6,200)
19 DO 20 I=1,N
20 20 WRITE(6,210) I,C(I),XLAM(I),IY(I),IOX(I)
21 WRITE(6,201) ENORS,COST
22 WRITE(6,205) NNORS
23 WRITE(6,220) NITER,INSTEP(1),I=1,NITER)
24 WRITE(6,250) PAR,XLAG
25 90 FORMAT(14)
26 100 FORMAT(214,F10.0)
27 110 FORMAT(10F8.0)
28 120 FORMAT(2F10.0)
29 200 FORMAT(///,38X,'INITIAL',5X,'OPTIMAL',/,5X,'ITEM',3X,'COST/UNIT',6
30 1X,'DEMAND',4X,'SOLUTION',4X,'SOLUTION',/)
31 201 FORMAT(///,5X,10E(10NORS/X)=,F10.5,5X,9HCOST(X)=,F10.2)
32 205 FORMAT(///,5X,'# OF NORS EVALUATIONS=',14)
33 209 FORMAT(///,5X,11H# OF ITEMS=,14,3X,8HBUDGET=,F10.2)
210 FORMAT(5X,'#',13,2(2X,F10.2),2(8X,14))

```

```

34      220 FORMAT(//,5X,'# OF ITERATIONS=',I4,/,5X,'# OF STEPS IN EACH ITERA
      *ION:',(5X,20I4))
35      250 FORMAT(//,5X,'LAMBDA=',F10.2,5X,'LAGRANGE MULTIPLIER=',F10.8,/)
36      400 FORMAT(1H1,//,10X,'A PRIMAL SEARCH METHOD')
37      405 FORMAT(//,5X,'PROBLEM NUMBER',I4)
38      STOP
39      END

40      SUBROUTINE OPTIM(IX,IOX,ENORS,COST)
      C *
      C * OPTIMIZATION ROUTINE
      C *
41      DIMENSION IX(100),IOX(100),DIREC(100),DELTA(100),NEWP(100),
      1XNEW(100),NEIGH(100),IY(100),IRANK(100),XDEL(100)
42      COMMON N,M,BLOCK1/C(100)/BLOCK2/XLAM(100)/BLOCK3/BUTCE,PAR/BLOCK4
      *NITER,NSTEP(500)
      C
43      NITER=1
44      ENORS=VNORS(IX)
45      COST=PROD(IX)
46      CKARE=DOTPR(C)
      C
      C EVALUATE THE FIRST DIFFERENCES
47      200 CALL TJREV(IX,DELTA)
48      NSTEP(NITER)=0
49      DENOM=DOTPR(DELTA)
50      CBETA=DENOM/CKARE
      C
      C EVALUATE THE DIRECTION PARAMETER PK
51      BETA=PAR*CBETA
      C
      C DETERMINE THE SEARCH DIRECTION
52      DO 10 I=1,N
53      DIREC(I)=DELTA(I)-BETA*C(I)
54      WSA=ABS(DIREC(I))
55      IF(I.EQ.1) GO TO 15
56      IF(WSA.LE.XSI) GO TO 10
57      15 XSI=WSA
58      10 CONTINUE
      C
      C NORMALIZE THE DIRECTION
59      DO 11 I=1,N
60      11 DIREC(I)=DIREC(I)/XSI
      C CALCULATE THE STEP SIZE ALPHA
61      CDOTD=DOTPR(DIREC)
62      CDOTX=PROD(IX)
63      XN=SQRT(FLOAT(N))
64      AMAX=(BUTCE-CDOTX)/(CDOTD*XN)
65      IF(AMAX.LT.1.0) AMAX=1.0
66      MMAX=IFIX(AMAX+.5)
67      AMAX=FLOAT(MMAX)
68      ALFA=AMAX
      C
      C OBTAIN A NEW POINT
69      100 DO 20 I=1,N
70      WN=FLOAT(IX(I))+ALFA*DIREC(I)+.5
71      IF(WN.GE.1.) GO TO 20
72      WN=1.0
73      20 NEWP(I)=IFIX(WN)
74      NSTEP(NITER)=NSTEP(NITER)+1

```

```

75 C CALCULATE ITS COST $ E(NORS/X)
76 UCOS=PROD(NEWP)
UNORS=VNORS(NEWP)
77 C IF INFEASIBLE , PERFORM A SEARCH IN THE NEGATIVE DIRECTION
IF(UCOS.GT.BUTCE) GO TO 30
78 C IF A BETTER FEASIBLE POINT, REPEAT THE STEPS
IF(UNORS.LT.ENORS) GO TO 500
79 ITETA=1
80 GO TO 40
81 30 ITETA=-1
82 40 CALL SEARCH(ITETA,UCOS,UNORS,ENORS,NEWP,KODE)
83 IF(KODE.EQ.1) GO TO 500
84 IF(ALFA.EQ.1.) GO TO 900
85 ALFA=ALFA-1.
86 GO TO 100
87 900 UCOS=-COST
UNORS=ENORS
88 CALL SEARCH(1,UCOS,UNORS,ENORS,IX,KODE)
89 IF(KODE) GO TO 1000,600
90 500 DO 45 I=1,N
91 45 IX(I)=NEWP(I)
92 600 COST=UCOS
ENORS=UNORS
93 NITER=NITER+1
94 GO TO 200
95 1000 DO 50 I=1,N
96 50 IOX(I)=IX(I)
97 RETURN
98 END
99
100
101 SUBROUTINE SEARCH(ITETA,UCOS,UNORS,ENORS,NEWP,KODE)
C
C THIS SUBROUTINE PERFORMS SEARCHES EITHER IN POSITIVE OR NEGATIVE DI
C
102 DIMENSION NEWP(100),NEIGH(100),IRANK(100),DELTA(100),INSET(100),D
* L(100)
103 COMMON N,M/BLOCK1/C(100)/BLOCK2/XLAMI(100)/BLOCK3/BUTCE,PAR/BLOCK4
* NITER,NSTEP(500)
C
104 NSTEP(NITER)=NSTEP(NITER)+1
105 IF(ITETA.EQ.1) GO TO 9
C
C CALCULATE THE FIRST DIFFERENCES FOR NEGATIVE DIRECTION
106 CALL TURNO(NEWP,DELTA)
107 GO TO 11
C
C CALCULATE THE FIRST DIFFERENCES
108 9 CALL TUREV(NEWP,DELTA)
109 11 DO 10 I=1,N
110 DELTA(I)=FLOAT(ITETA)*DELTA(I)
111 10 DEL(I)=DELTA(I)
C
C RANK THE FIRST DIFFERENCES
112 CALL SIRALA(DEL,IRANK)
113 DO 20 I=1,N
114 DO 15 J=1,N
115 JV=J
116 IF(IRANK(J).EQ.1) GO TO 17
117 15 CONTINUE
118 17 INSET(I)=JV
119 20 CONTINUE

```

```

C
C CHECK WHETHER THERE IS A BETTER FEASIBLE POINT
C
120 DO 25 I=1,N
121 JVAL=INSET(I)
122 COSJ=UCOS+FLOAT(ITETA)*C(JVAL)
123 IF(COSJ.GT.BUTCE) GO TO 25
124 DO 35 JJ=1,N
125 35 NEIGH(JJ)=NEWP(JJ)
126 IF(NEWP(JVAL).EQ.0.AND.ITETA.EQ.-1) GO TO 25
127 NEIGH(JVAL)=NEWP(JVAL)+ITETA
128 XNORS=UNORS-DELTA(JVAL)
129 IF(XNORS-ENORS) 100,120,110
130 25 CONTINUE
131 GO TO 120
132 110 DO 40 I=1,N
133 40 NEWP(I)=NEIGH(I)
134 UNORS=XNORS
135 UCOS=COSJ
C IF NO BETTER FEASIBLE POINT, FAILURE & KODE=0
136 120 KODE=0
137 RETURN
C
C IF A BETTER FEASIBLE POINT, SUCCESS & KODE=1
138 100 KODE=1
139 DO 70 I=1,N
140 70 NEWP(I)=NEIGH(I)
141 UNORS=XNORS
142 UCOS=COSJ
143 RETURN
144 END
145 FUNCTION PROD(IX)
C
C TO CALCULATE THE DOT PRODUCT C.IX
C
146 DIMENSION IX(100)
C
147 COMMON N,M/BLOCK1/C(100)
148 SUM=0.0
149 DO 10 I=1,N
150 IXI=IX(I)
151 10 SUM=SUM+FLOAT(IXI)*C(I)
152 PROD=SUM
153 RETURN
154 END
155 FUNCTION DOTPR(X)
C
C TO CALCULATE THE DOT PRODUCT C.X
C
156 DIMENSION X(100)
157 COMMON N,M/BLOCK1/C(100)
158 SUM=0.0
159 DO 10 J=1,N
160 SUM=SUM+C(J)*X(J)
161 10 CONTINUE
162 DOTPR=SUM
163 RETURN
164 END

```

```

165      FUNCTION VNORS(IX)
      C
      C      TO EVALUATE THE E(NORS/X)
      C
166      DIMENSION IX(100),PM(100,20),PRJ(20)
167      COMMON N,M/BLOCKS/POISON(40,100)/COUNT/NNORS
168      DOUBLE PRECISION SUM,PRO,TOPL,PM,XW
      C
169      EPS=.00001
170      EPSQ=.00000001
171      NNORS=NNORS+1
172      M1=M+1
173      SUM=0.0+00
174      DO 20 JJ=1,M1
175      PRJ(JJ)=1.0+00
176      DO 26 II=1,N
177      IXJ=IX(II)+JJ-1
178      IF(IXJ.GT.0) GO TO 24
179      XX=PDFOP(IXJ,II)
180      GO TO 25
181      24 IF(IXJ.GT.40) IXJ=40
182      XX=POISON(IXJ,II)
183      25 IF(XX.LT.EPSQ) XX=0.0
184      IF(PRO(JJ).LT.EPSQ) GO TO 20
185      PRO(JJ)=PRJ(JJ)*XX
186      26 CONTINUE
187      20 SUM=SUM+1.-PRO(JJ)
188      TOPL=0.0+00
189      DO 30 I=1,N
190      K=J
191      40 MXK=IX(I)+M+2+K
192      WX=PDFOP(MXK,I)
193      WW=FLOAT(K+1)*WX
194      IF(WX.LT.EPS) GO TO 30
195      TOPL=TOPL+WW
196      K=K+1
197      GO TO 40
198      30 CONTINUE
199      VNORS=SUM+TOPL
200      RETURN
201      END

202      SUBROUTINE TUREV(IX,DELTA)
      C
      C      TO EVALUATE THE FIRST DIFFERENCES
      C
203      DIMENSION IX(100),DELTA(100)
204      COMMON N,M
      C
205      DO 10 K=1,N
206      DELTA(K)=DIF(IX,K)
207      10 CONTINUE
208      RETURN
209      END

210      SUBROUTINE TURND(NEWP,DELTA)
      C
      C      TO EVALUATE THE FIRST DIFFERENCES IN NEGATIVE DIRECTION
      C
211      DIMENSION NEWP(100),NEIGH(100),DELTA(100)

```

```

212      COMMON N,M
213      C
214      DO 10 I=1,N
215      DO 20 J=1,M
216      IF(J.EQ.1) GO TO 25
217      NEIGH(J)=NEWP(J)
218      GO TO 20
219      25 NEIGH(J)=NEWP(J)-1
220      IF(NEIGH(J).LT.0) NEIGH(J)=0
221      20 CONTINUE
222      DELTA(I)=DIF(NEIGH,I)
223      10 CONTINUE
224      RETURN
225      END

225      FUNCTION DIF(IX,K)
226      C
227      C TO COMPUTE THE FIRST DIFFERENCE OF ITEM K
228      C
229      DIMENSION IX(100)
230      COMMON N,M/BLOCK5/POISCN(40,100)
231      DOUBLE PRECISION PRO,SUM,EP SQ
232      C
233      EP SQ=1.D-08
234      M1=M+1
235      SUM=C.D+00
236      DO 20 J1=1,M1
237      PRO=1.D+00
238      J=J1-1
239      DO 30 I=1,N
240      IXJ=IX(I)+J
241      IF(I.EQ.K) GO TO 25
242      IF(IXJ.EQ.0) GO TO 23
243      IF(IXJ.GT.39) IXJ=39
244      PRO=PRO*POISCN(IXJ,I)
245      GO TO 30
246      23 IXJ1=IXJ
247      GO TO 26
248      25 IXJ1=IXJ+1
249      26 IF(PRO.LT.EP SQ) GO TO 20
250      PRO=PRO*PCFOP(IXJ1,I)
251      30 CONTINUE
252      SUM=SUM+PRO
253      IXM=IX(K)+M+1
254      IF(IXM.GT.40) IXM=40
255      DIF=SUM+1.-POISCN(IXM,K)
256      RETURN
257      END

254      SUBROUTINE TABLE(POISON)
255      C
256      C TO FILL IN THE ARRAY FOR CUMULATIVE POISSON SUMS
257      C
258      DIMENSION POISCN(40,100)
259      COMMON N,M/BLOCK2/XLAM(100)
260      DOUBLE PRECISION EPS,TERM,SUMX
261      C
262      EPS=1.D-06
263      DO 10 J=1,N
264      TERM=1.D+00

```

```

261      SUMX=1.D+00
262      DO 20 I=1,40
263      TERM=TERM*XLAM(J)/FLOAT(I)
264      IX=I
265      IF (TERM.LT.EPS) GO TO 25
266      SUMX=SUMX+TERM
267      POISON(I,J)=EXP(-XLAM(J))*SUMX
268      IF (POISON(I,J).GT.1.0) POISON(I,J)=1.0
269      20 CONTINUE
270      25 IF (IX.EQ.40) GO TO 10
271      DO 30 I=IX,40
272      30 POISON(I,J)=1.0
273      10 CONTINUE
274      RETURN
275      END

276      FUNCTION PDFOP(K,I)
C
C      TO CALCULATE INDIVIDUAL TERMS OF THE POISSON DISTRIBUTION
C
277      COMMON /BLOCK2/XLAM(100)
278      DOUBLE PRECISION SUM,WW
C
279      SUM=0.D+00
280      IF (K.LT.2) GO TO 10
281      DO 15 J=2,K
282      SUM=SUM+ALOG(FLOAT(J))
283      15 CONTINUE
284      10 WW=-XLAM(I)+K*ALOG(XLAM(I))-SUM
285      IF (WW.LT.-15.0) GO TO 20
286      PDFOP=DEXP(WW)
287      GO TO 25
288      20 PDFOP=0.0
289      25 RETURN
290      END

291      SUBROUTINE SIRALA(X,IR)
C
C      THIS SUBROUTINE RANKS AN ARRAY OF ELEMENTS IN DESCENDING ORDER
C      IR(K) IS THE RANK OF ITEM K IN THIS ORDERING
C
292      COMMON N,M
293      DIMENSION X(100),IR(100),ISEQ(100),Y(100)
C
294      DO 9 I=1,N
295      9 ISEQ(I)=I
296      DO 10 I=1,N
297      XMIN=X(I)
298      IMINJ=I
299      IMIN=ISEQ(I)
300      N1=N-I+1
301      DO 20 J=1,N1
302      IF (X(J).GT.XMIN) GO TO 21
303      GO TO 20
304      21 XMIN=X(J)
305      IMIN=ISEQ(J)
306      IMINJ=J
307      20 CONTINUE
308      IR(IMIN)=I
309      IF (N1.EQ.1) GO TO 10

```

```

310      N2=N1-1
311      DO 30 J=1,N2
312      IF(J.LT.IMINJ) GO TO 33
313      J1=J+1
314      ISEQ(J)=ISEQ(J1)
315      Y(J)=X(J1)
316      GO TO 29
317      33 Y(J)=X(J)
318      29 X(J)=Y(J)
319      30 CONTINUE
320      10 CONTINUE
321      RETURN
322      END

323      SUBROUTINE BASLA(XLAG,IX)
      C
      C      THIS SUBROUTIN FINDS AN INITIAL POINT WITH A PREDETERMINED
      C      LAGRANGE PARAMETER
      C
324      DIMENSION IX(100)
325      COMMON N,M/BLOCK1/C(100)/BLOCK5/POISON(40,100)
      C
326      DO 10 I=1,N
327      IW=0
328      A=XLAG*C(1)+PDFOP(IW,I)-1.
329      IF(A.GE.0.0) GO TO 10
330      15 IW=IW+1
331      B=A+PDFOP(IW,I)
332      IF(A.LE.0.0.AND.B.GT.0.0) GO TO 10
333      A=B
334      GO TO 15
335      10 IX(I)=IW
336      RETURN
337      END

      SENTRY

```

A PRIMAL SEARCH METHOD

PROBLEM NUMBER 1

OF ITEMS= 5 BUDGET=\$ 25000.00

ITEM	COST/UNIT	DEMAND	INITIAL SOLUTION	OPTIMAL SOLUTION
# 1	2980.00	2.10	2	3
# 2	1751.00	1.50	2	2
# 3	462.00	1.20	3	3
# 4	1500.00	5.00	6	6
# 5	345.00	3.50	6	6

E(NORS/X)= 0.98571 COST(X)=\$ 24898.00

OF MOPS EVALUATIONS= 3

OF ITERATIONS= 2
OF STEPS IN EACH ITERATION: 2 3

LAMBOA= 0.05 LAGRANGE MULTIPLIER=0.00020000

STATEMENTS EXECUTED= 13876

CORE USAGE OBJECT CODE= 15256 BYTES,ARRAY AREA= 43028 BYTES,TOTAL AREA AVAILABLE= 145408 BYTES

DIAGNOSTICS NUMBER OF ERRORS= 0, NUMBER OF WARNINGS= 0, NUMBER OF EXTENSIONS= 0

CCMPLE TIME= 0.27 SEC,EXECUTION TIME= 0.24 SEC, 12.35.52 MONDAY 4 AUG 80 MATFIV - JUN 1977

APPENDIX A.2

COMPUTER CODE FOR THE MODIFIED ALGORITHM OF BROOKS ET AL.

A.2.1

A List of the Variables in the Computer Program

BO	Coefficients $b_0, b_1, \dots, b_m$
C(I)	Cost of item I
COST	Cost of the current solution X
ENORS	$E(NORS/X)$
EPS	Parameter for testing sufficiently closeness
F(100,50)	Matrix for the cumulative poisson sums
ICOUN	Counter for the number of iterations
IMAX	The index I for which $RATIO(I)$ is maximum
IX(100)	Current solution
M	An upper bound
N	The total number of items under consideration
NOPR	Problem number
NSTEP (ICOUN)	Number of steps in iteration ICOUN
OPTIM	The optimization routine
PDFOP	Function routine which calculates the individual poisson terms
RATIO (100)	Ratio of return per dollar investment
RMAX	Ratio (IMAX)
TABLE	The subroutine to fill in the matrix F(100,50)
XLAM (100)	Poisson demand rates
XINVT	Budget
VNORS	Function routine to calculate $E(NORS/X)$

The FORTRAN Listing of the Computer Program

```

C
C *****
C
C   MODIFIED SOLUTION METHOD OF BROOKS ET AL
C
C *****
C
1  DIMENSION IX(100),BO(100)
2  COMMON N,M/BLOCK1/C(100),XINVT/BLOCK2/XLAM(100)/BLOCK4/F(100,50)/
   *LOCK5/ICOUN,NSTEP(50)
C
3  EPS=.001
4  EPSQ=EPS**2
C
C   INPUT THE DATA
C
5  READ(5,90) NOPR
6  READ(5,100) N,M,XINVT
7  READ(5,110) (C(I),I=1,N)
8  READ(5,110) (XLAM(I),I=1,N)
9  WRITE(6,400)
10 WRITE(6,405) NOPR
11 WRITE(6,105) N,XINVT
12 M1=M+1
C
C   INITIALIZE THE PARAMETERS B
C
13 DO 10 J=1,M
14   BO(J)=0.0
15   BO(M1)=1.0
C
C   CALCULATE THE PROBABILITIES
16   CALL TABLE(F)
17   ICOUN=1
C
C   PERFORM THE OPTIMIZATION
C
18   1000 CALL OPTIM(BO,IX,COST)
C
C   CHECK IF B-NEW IS SUFFICIENTLY CLOSE TO B-OLD
C
19   SUMSQ=0.0
20   DO 20 J1=1,M1
21   PRO=1.0
22   DO 40 I5=1,N
23   IXJ=IX(I5)+J1
24   IF(IXJ.GT.50) GO TO 40
25   PRO=PRO*F(I5,IXJ)
26   40 CONTINUE
27   IF(PRO.LT.EPSQ) PRO=0.0
28   SUMSQ=SUMSQ+((BO(J1)-PRO)**2)
29   BO(J1)=PRO
30   20 CONTINUE
C
C   IF SO, TERMINATE AND PRINT THE RESULTS
C
31   IF(SUMSQ.LT.EPS) GO TO 500
32   C   OTHERWISE REPEAT THE STEPS
   ICOUN=ICOUN+1

```

```

33      GO TO 1000
34 500 WRITE(6,240)
35      DO 30 I=1,N
36      30 WRITE(6,250) I,C(I),XLAM(I),IX(I)
37      ENORS=VNORS(IX)
38      WRITE(6,201) ENORS,COST
39      WRITE(6,300) (BO(J),J=1,M1)
40      WRITE(6,301) ICOUN,(NSTEP(I),I=1,ICOUN)
41      90 FORMAT(14)
42      100 FORMAT(2I4,F10.0)
43      105 FORMAT(//,5X,'N=',I4,5X,' BUDGET=$',F10.2)
44      110 FORMAT(10F8.0)
45      201 FORMAT(//,5X,10E(NORS/X)=,F10.5,2X,9HCOST(X)=$,F10.2)
46      240 FORMAT(//,38X,'OPTIMAL',/,5X,'ITEM',3X,'COST/JNIT',5X,'DEMAND',4X
47      1,'SOLUTION',//)
48      250 FORMAT(5X,'#',I3,2(2X,F10.2),8X,I4)
49      300 FORMAT(//,5X,'B:',10F10.3)
50      301 FORMAT(//,5X,16H# OF ITERATIONS=,I4,//,(5X,'# OF STEPS IN EACH I
51      *ERATION:',20I4))
52      400 FORMAT(1H1,//,5X,'THE MODIFIED SOLUTION METHOD OF BROOKS ET AL',/,
53      *)
54      405 FORMAT(//,5X,'PROBLEM NUMBER',I4)
55      STOP
56      END
57
58      SUBROUTINE OPTIM(BO,IX,COST)
59
60      C
61      C OPTIMIZATION ROUTINE PERFORMS THE MARGINAL ANALYSIS
62      C
63      DIMENSION IX(100),BO(100),RATIO(100),MR(100)
64      COMMON N,M/BLOCK1/C(100),XINVT/BLOCK2/F(100,50)/BLOCK5/ICOUN,NSTEP
65      *(50)
66
67      C
68      M1=M+1
69      COST=0.0
70      DO 10 I=1,N
71      IX(I)=0
72
73      C
74      C CALCULATE THE RATIO OF RETURN PER EACH DOLLAR INVESTED FOR EACH ITE
75      C
76      SUM=0.0
77      DO 20 J=1,M1
78      JJ=J+1
79      WW=F(I,JJ)/F(I,J)
80      SUM=SUM+BO(J)*ALOG(WW)
81      20 SUM=SUM+BO(J)*ALOG(WW)
82      10 RATIO(I)=SUM/C(I)
83      NSTEP(ICOUN)=1
84      200 NOR=0
85      NSTEP(ICOUN)=NSTEP(ICOUN)+1
86      DO 21 I=1,N
87      21 MR(I)=-1
88      100 IMR=0
89
90      C
91      C FIND THE MAXIMUM RATIO & ADD ONE UNIT OF THIS ITEM TO THE KIT
92      C
93      DO 20 I=1,N
94      IF(MR(I).EQ.1) GO TO 25
95      IMR=IMR+1
96      IF(IMR.EQ.1) GO TO 26
97      IF(RATIO(I).LE.RMAX) GO TO 25
98      26 RMAX=RATIO(I)

```

```

79      IMAX=I
80      25 CONTINUE
81      MR(IMAX)=1
      C
      C CALCULATE THE COST OF THE NEW KIT
82      XDCST=COST+C(IMAX)
      C
      C IF COST EXCEEDS THE BUDGET, FIND THE ITEM WHICH WILL YIELD MAXIMUM
      C WITHOUT EXCEEDING THE BUDGET, THEN RETURN THE MAIN PROG.
      C
83      IF(XDCST.GT.XINVT) GO TO 30
84      COST=XDCST
85      IX(IMAX)=IX(IMAX)+1
86      SUM=0.0
87      DO 35 J=1,M1
88      IXJ=IX(IMAX)+J
89      IXJ1=IXJ+1
90      IF(IXJ.GE.50) GO TO 35
91      WK=F(IMAX,IXJ1)/F(IMAX,IXJ)
92      SUM=SUM+BC(J)*ALOG(WK)
93      35 CONTINUE
      C OTHERWISE UPDATE THE RATIO(IMAX), REPEAT THE STEPS
94      RATIO(IMAX)=SUM/C(IMAX)
95      GO TO 200
96      30 NDR=NDR+1
97      IF(NDR.LT.N) GO TO 100
98      RETURN
99      END

100     SUBROUTINE TABLE(F)
      C
      C CALCULATING CUMULATIVE POISSON SUMS
      C
101     DIMENSION F(100,50)
102     COMMON N,M,BLOCK1/C(100),XINVT/BLOCK2/XLAM(100)
103     DOUBLE PRECISION EPS,TERM,SUM
      C
104     EPS=1.0D-06
105     DO 10 I=1,N
106     F(I,1)=PDFOP(0,I)
107     TERM=1.0D+00
108     SUM=1.0D+00
109     DO 30 J=2,50
110     TERM=TERM*XLAM(I)/FLOAT(J-1)
111     JX=J
112     IF(TERM.LT.EPS) GO TO 25
113     SUM=SUM+TERM
114     F(I,J)=EXP(-XLAM(I))*SUM
115     IF(F(I,J).GT.1.0) F(I,J)=1.0
116     30 CONTINUE
117     25 IF(JX.EQ.50) GO TO 10
118     DO 35 J=JX,50
119     F(I,J)=1.0
120     35 CONTINUE
121     RETURN
122     END

123     FUNCTION PDFOP(K,I)
      C
      C CALCULATING THE INDIVIDUAL POISSON TERMS

```

```

124      C      COMMON /BLOCK2/XLAM(100)
125      C      DOUBLE PRECISION SUM,WW

126      SUM=0.0+00
127      IF (K.LT.2) GO TO 10
128      DO 15 J=2,K
129      SUM=SUM+ALOG(FLOAT(J))
130      15 CONTINUE
131      10 WW=-XLAM(I)+K*ALOG(XLAM(I))-SUM
132      IF (WW.LT.-15.0) GO TO 20
133      PDFOP=DEXP(WW)
134      GO TO 25
135      20 PDFOP=0.0
136      25 RETURN
137      END

138      FUNCTION VNORS(IX)
139      C
140      C      CALCULATING THE E(NORS/XI) FUNCTION
141      C
142      C      DIMENSION IX(100),PRO(20)
143      C      COMMON N,M/BLOCK4/F(100,50)
144      C      DOUBLE PRECISION SUM,PRO,TOPL
145      C
146      EPS=.00001
147      M1=M+1
148      SUM=0.0+00
149      DO 10 J=1,M1
150      PRO(J)=1.0+00
151      DO 20 I=1,N
152      IXJ=IX(I)+J
153      20 PRO(J)=PRO(J)*F(I,IXJ)
154      10 SUM=SUM+1.-PRO(J)
155      TOPL=0.0+00
156      DO 30 I=1,N
157      K=0
158      40 MAX=IX(I)+M+2+K
159      WX=PDFOP(PXK,I)
160      WW=FLOAT(K+1)*WX
161      IF (WX.LT.EPS) GO TO 30
162      TOPL=TOPL+WW
163      K=K+1
164      GO TO 40
165      30 CONTINUE
166      VNORS=SUM+TOPL
167      RETURN
168      END

```

ENTRY

THE MODIFIED SOLUTION METHOD OF BROOKS ET AL

PROBLEM NUMBER 1

N= 5 BUDGET=\$ 25000.00

ITEM	COST/UNIT	DEMAND	OPTIMAL SOLUTION
# 1	2980.00	2.10	2
# 2	1751.00	1.50	2
# 3	462.00	1.20	4
# 4	1500.00	5.00	7
# 5	345.00	3.50	9

E(NGRS/X)= 0.98619 COST(X)=\$ 24915.00

B: 3.450 0.728 0.891 0.562 0.988 0.996

OF ITERATIONS= 3

OF STEPS IN EACH ITERATION: 23 26 26

STATEMENTS EXECUTED= 8963

CORE USAGE OBJECT CODE= 7320 BYTES, ARRAY AREA= 22800 BYTES, TOTAL AREA AVAILABLE= 145608 BYTES

DIAGNOSTICS NUMBER OF ERRORS= 0, NUMBER OF WARNINGS= 0, NUMBER OF EXTENSIONS= 0

COMPILE TIME= 0.13 SEC, EXECUTION TIME= 0.14 SEC, 12.11.57 MONDAY 4 AUG 80 MATFIV - JUN 1977

APPENDIX A.3

COMPUTER CODE FOR THE SADDLE POINT

SEARCH ALGORITHM

A.3.1

A List of the Variables in the Computer Program

A(50)	The cost of the current solution for subroutine DUALPR
AXM	Minimum of A(I) which exceed BUTCE; A(IXM)
AXP	Maximum of A(I) which are less than BUTCE; A(IXP)
BASLA	Subroutine which finds an initial point
BUTCE	Budget
C(100)	Cost of the item
COST	Total cost of the current solution
CX(50)	E(NORS/X) (used only in DUALPR)
CXM	CX(IXM)
CXP	CX(IXP)
DIF	Function routine for calculating first differences
DOTPR	The dot product of two vectors
DUALPR	Subroutine which solves the dual problem to obtain the Lagrange multiplier
ENORS	E(NORS/X)
INPUT	The subroutine which reads in the necessary data and initializes the arrays
ISTAR	Starting solution X^0
IX(100)	Current solution
IX0(100)	Solution obtained in the Subroutine BASLA
IX1(100)	Solution obtained in the Subroutine SEARCH
IX2(100)	Solution obtained in the Subroutine TARAMA
1X0(100)	Final and/or optimal solution
IY(100)	Initial solution
KPAR	Parameter for finding an initial point in BASLA
M	An upper bound

N	Total number of items under consideration
NARA	Number of times Subroutine ARAMA called
NBAS	Number of time Subroutine BASLA called
NITER	Number of iterations
NOPG	Number of points generated by the Saddle Point Search Algorithm
NOPR	Problem number
NPA1	Number of times Part 1 called
NPA2	Number of times Part 2 called
NPA3	Number of times Part 3 called
NSEA	Number of times SEARCH called
NVNOR	Number of E(NORS/X) function evaluations
OPTIM	Optimization Routine
PART1	The subroutine which performs a test by calculating an upper bound to the linear program in determining whether a component can be increased by one
PART2	The subroutine which calculates an upper bound for the sum of first differences
PART3	The subroutine which performs the same test as in PART1 but sets up the linear program explicitly
PDFOP	Function routine for calculating individual poisson terms
PLAG	Current Lagrange multiplier
POISON	Matrix for the cumulative poisson terms
POLD	Previous Lagrange multiplier
SEARCH	The subroutine that makes a one dimensional search for each component separately
SIRALA (X,IR)	The subroutine that ranks the array X in descending order
TABLE	The subroutine which fills in the matrix POISON

TARAMA	The subroutine which makes an attempt for improvement heuristically
VNORS	The function routine to calculate $E(\text{NORS}/X) + \lambda CX$
VX0	$E(\text{NORS}/IX0)$
VX1	$E(\text{NORS}/IX1)$
VX2	$E(\text{NORS}/IX2)$
XLAM(100)	Poisson demand rates
XLPRO	The Linear Programming Routine which solves problems of the type $\max CX$ subject to $AX \leq b \quad X \geq 0$.

DEVELOPMENT OF A WAR READINESS SPARES KIT MODEL AND
SOLUTION TECHNIQUE(U) OKLAHOMA STATE UNIV STILLWATER
SCHOOL OF INDUSTRIAL ENGINEER. M P TERRELL ET AL.
31 DEC 80 F33600-80-C-0423 F/G 9/2

144

UNCLASSIFIED

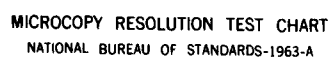
31 DEC 88 F33600-80-C-0423

F/G 9/2

NL

END

FILMED
18
DTIC



FORTRAN Listing of the Computer Program

```

      SJOB TIME=05
      C *****
      C *
      C *
      C * A SADDLE POINT SEARCH ALGORITHM *
      C *
      C *
      C *****
      C
      C MAIN PROGRAM
      C
1      DIMENSION IX(100),IY(100)
      C
      C INPUT THE DATA
2      CALL INPUT(NOPR,KPAR,PLAG,IY)
      C
      C PERFORM THE OPTIMIZATION
      C
3      CALL OPTIM(PLAG,KPAR,IY,ENORS,IX,COST)
      C
      C PRINT THE RESULTS
      C
4      CALL OUTPUT(PLAG,NOPR,IY,ENORS,IX,COST)
      C
5      STOP
6      END
      C
7      SUBROUTINE INPUT(NOPR,KPAR,PLAG,IY)
      C
      C TO READ IN THE DATA
      C
8      DIMENSION IY(100)
9      COMMON N,M/BLOCK1/C(100)/BLOCK2/XLAM(100)/BLOCK3/BUTCE
10     COMMON /COUNT/NITER,NBAS(50),NSEA(50),NTAR(50),NARA(50),NPA1(50),
      *NP A2(50),NPA3(50),NVNOR/TAB/POISSON(50,100)
      C
11     READ(5,90) NOPR
12     READ(5,100) N,M,BUTCE
13     READ(5,110) (C(I),I=1,N)
14     READ(5,110) (XLAM(I),I=1,N)
15     READ(5,150) KPAR,ISTAR,PLAG
      C
      C INITIALIZE THE COUNTERS
16     NITER=1
17     NVNOR=0
18     DO 19 I=1,50
19     NBAS(I)=0
20     NSEA(I)=0
21     NTAR(I)=0
22     NARA(I)=0
23     NPA1(I)=0
24     NPA2(I)=0
25     NPA3(I)=0
26     19 CONTINUE
27     DO 10 I=1,N
28     IY(I)=ISTAR
      C
      C CALCULATE THE CUMULATIVE POISSON SUMS
      C
29     CALL TABLE(POISSON)

```

```

30      90 FORMAT(14)
31      100 FORMAT(2 I4,F10.0)
32      110 FORMAT(10F8.0)
33      150 FORMAT(2 I4,F10.0)
34      RETURN
35      END

36      SUBROUTINE OUTPUT(PLAG,NOPR,IY,ENORS,IX,COST)
C
C      WRITING OUT THE RESULTS
C
37      DIMENSION IY(100),IX(100)
38      COMMON N,M/BLOCK1/C(100)/BLOCK2/XLAM(100)/BLOCK3/BUTCE
39      COMMON /COUNT/NITER,NBAS(50),NSEA(50),NTAR(50),NARA(50),NPA1(50),
      *NPA2(50),NPA3(50),NVNOR/TAB/POISON(5,100)
C
40      WRITE(6,300)
41      WRITE(6,405) NOPR
42      WRITE(6,205) N,BUTCE
43      WRITE(6,220)
44      DO 20 I=1,N
45      20 WRITE(6,225) I,C(I),XLAM(I),IY(I),IX(I)
46      WRITE(6,210) ENORS,COST
47      WRITE(6,230) PLAG
48      WRITE(6,240) NITER,NVNOR
49      WRITE(6,245)
50      DO 30 I=1,NITER
51      30 WRITE(6,250) I,NBAS(I),NSEA(I),NTAR(I),NARA(I),NPA1(I),NPA2(I),NPA3(I)
52      205 FORMAT(5X,'NUMBER OF ITEMS=',I4,2X,'BUDGET=',F10.2)
53      220 FORMAT(///,38X,'INITIAL',5X,'OPTIMAL',/,5X,'ITEM',3X,'COST/UNIT',6
      1X,'DEMAND',4X,'SOLUTION',4X,'SOLUTION',/)
54      225 FORMAT(5X,'#',I3,2(2X,F10.2),2(8X,I4))
55      210 FORMAT(//,5X,'E(NORS/X)=',F10.5,5X,'COST(X)=',F10.2)
56      230 FORMAT(/,5X,'LAGRANGE MULTIPLIER=',F10.6)
57      240 FORMAT(//,5X,'# OF ITERATIONS=',I4,/,5X,'# OF NORS EVALUATIONS=',
      1I4)
58      245 FORMAT(//,5X,'# OF STEPS IN EACH ITERATION',/,5X,'ITERATION #',4
      *X,'STEP1 STEP2 STEP3 STEP4 PART1 PART2 PART3',/)
59      250 FORMAT(12X,I4,2X,7(3X,I4))
60      300 FORMAT(11H,/,5X,'THE SADDLE POINT SEARCH ALGORITHM',/)
61      405 FORMAT(///,5X,'PROBLEM NUMBER',I4,/)
62      RETURN
63      END

64      SUBROUTINE OPTIM(PLAG,KPAR,IY,VXA,IXO,COST)
C
C      OPTIMIZATION ROUTINE
C
65      DIMENSION IX0(100),IX1(100),IXO(100),IY0(100),CX(50),A(50),IX2(100
      *)
66      COMMON N,M/BLOCK3/BUTCE/BLOCK4/NOPG/BLOCK1/C(100)
67      COMMON /COUNT/NITER,NBAS(50),NSEA(50),NTAR(50),NARA(50),NPA1(50),
      *NPA2(50),NPA3(50),NVNOR
C
68      EPS=.000002
69      NOPG=1
70      ZZ=0.0
71      CX(NOPG)=VNORS(ZZ,IY0)
72      A(NOPG)=DOTPR(IY0)

```

```

C
C STEP1: FIND AN INITIAL POINT
C
73 500 CALL BASLA(PLAG,KPAR,IX0)
74 IF(NITER.GT.1) GO TO 20
75 DO 25 I=1,N
76 25 IX0(I)=IX0(I)
C
C STEP2: FIND A STATIONARY POINT
C
77 20 CALL SEARCH(PLAG,IX0,IX1)
78 VX1=VNORS(PLAG,IX1)
C
C STEP3: A HEURISTIC IMPROVEMENT TEST
C
79 CALL TARAMA(PLAG,IX1,VX1,IX2,VX2)
C
C COMPARE THE NEW POINT WITH THE INCUMBENT POINT
80 IF(VX2-VX1) 14,12,12
81 14 DO 13 I=1,N
82 13 IX0(I)=IX2(I)
83 GO TO 20
C
C STEP4: REDUCTION OF POTENTIAL CANDIDATES BY CONTRADICTION
C
84 12 CALL ARAMA(PLAG,IX2,VX2,IX0,VX0)
85 IF(VX0-VX1) 10,15,15
86 10 DO 11 I=1,N
87 11 IX0(I)=IX0(I)
88 GO TO 20
89 15 NOPG=NOPG+1
C
C READJUST THE COEFFICIENT OF THE LINEAR PROGRAM
C
90 A(NOPG)=DOTPR(IX0)
91 COST=A(NOPG)
92 VXA=VNORS(IZZ,IX0)
93 CX(NOPG)=VXA
94 POLD=PLAG
C
C SOLVE THE DUAL LP TO OBTAIN THE LAGRANGE MULTIPLIER
C
95 CALL DJALPR(CX,A,PLAG)
C
C IF THE NEW MULTIPLIER IS SUFFICIENTLY CLOSE TO OLD ONE, TERMINATE
C OTHERWISE REPEAT THE ITERATIONS
C
96 FARK=ABS(POLD-PLAG)
97 IF(FARK.LT.EPS) GO TO 1000
98 NITER=NITER+1
99 GO TO 500
100 1000 RETURN
101 END
C
102 SUBROUTINE DUALPR(CX,A,X)
C
C SOLVE THE DUAL TO OBTAIN A LAGRANGE MULTIPLIER
C
103 DIMENSION CX(50),A(50)
104 COMMON /BLOCK3/BUTCE/BLOCK4/NOPG

```

```

105      IX=0
106      IY=0
107      DO 10 I=1,NOPG
108      IF(A(I)-BUTCE) 20,20,30
109      20 IY=IY+1
110      IF(IY.EQ.1) GO TO 25
111      IF(A(I)-AXP) 10,10,25
112      25 AXP=A(I)
113      CXP=CX(I)
114      GO TO 10
115      30 IX=IX+1
116      IF(IX.EQ.1) GO TO 35
117      IF(A(I)-AXM) 35,10,10
118      35 AXM=A(I)
119      CXM=CX(I)
120      10 CONTINUE
121      X=(CXP-CXM)/(AXM-AXP)
122      RETURN
123      END

124      SUBROUTINE BASLA(XLAG,KPAR,IX)
      C
      C FINDS AN INITIAL POINT
      C
125      DIMENSION IX(100)
126      COMMON N,M,BLOCK1/C(1,0)
127      COMMON /COUNT/NITER,NBAS(50),NSEA(50),NTAR(50),NARA(50),NPA1(50)
      *NPA2(50),NPA3(50),NVNOR
128      NBAS(NITER)=NBAS(NITER)+1
129      DO 10 I=1,N
130      IW=0
131      A=XLAG*C(I)+PDFOP(IW,I)-1.
132      IF(A.GE.0.) GO TO 11
133      15 IW=IW+1
134      B=A+PDFOP(IW,I)
135      IF(A.LE.0.0.AND.B.GT.0.0) GO TO 11
136      A=B
137      GO TO 15
138      11 IW=IW-KPAR
139      IF(IW.LT.0) IW=0
140      10 IX(I)=IW
141      RETURN
142      END

143      SUBROUTINE SEARCH(XLAG,IX,IXO)
      C
      C PERFORMS ONE DIMENSIONAL SEARCH FOR EACH COMPONENT SEPARATELY
      C
144      DIMENSION IX(100),IXO(100),IY(100)
145      COMMON N,M
146      COMMON /COUNT/NITER,NBAS(50),NSEA(50),NTAR(50),NARA(50),NPA1(50)
      *NPA2(50),NPA3(50),NVNOR
      C
147      NSEA(NITER)=NSEA(NITER)+1
148      DO 10 I=1,N
      C
      C CALCULATE THE FIRST DIFFERENCE
149      XI=DIF(XLAG,IX,I)
      C

```

```

C IF IT IS POSITIVE , THE POINT X(I) IS STATIONARY, OTHERWISE
C FIND X(I) SUCH THAT ITS FIRST DIFFERENCE IS POSITIVE
C
150 IF(X1) 20,25,25
151 20 DO 11 J=1,N
152 IF(I.EQ.J) GO TO 12
153 IY(J)=IX(J)
154 GO TO 11
155 12 IY(J)=IX(J)+1
156 11 CONTINUE
157 Y1=DIF(XLAG,IY,I)
158 IF(Y1) 30,36,36
159 30 DO 13 J=1,N
160 IF(I-J) 13,14,13
161 14 IX(J)=IX(J)+1
162 13 CONTINUE
163 X1=Y1
164 GO TO 20
165 25 DO 15 J=1,N
166 IF(I.EQ.J) GO TO 16
167 IY(J)=IX(J)
168 GO TO 15
169 16 IY(J)=IX(J)-1
170 IF(IY(J).LT.0) GO TO 36
171 15 CONTINUE
172 Y2=DIF(XLAG,IY,I)
173 IF(Y2) 36,36,40
174 40 DO 60 J=1,N
175 60 IX(J)=IY(J)
176 X1=Y2
177 GO TO 25
178 36 DO 65 J=1,N
179 65 IX(J)=IX(J)
180 10 CONTINUE
181 RETURN
182 END

183 SUBROUTINE TARAMA(XLAG,IX,VX,IXO,VXO)
C
C OBTAINS IMPROVEMENT HEURISTICALLY
C
184 DIMENSION IX(100),IY(100),RATIO(100),IRANK(100),IXO(100)
185 COMMON N,M
186 COMMON /COUNT/NITER,NBAS(50),NSEA(50),NTAR(50),NARA(50),NPA1(50),
* NPA2(50),NPA3(50),VVNR
C
187 VTAR(NITER)=NTAR(NITER)+1
188 ITETA=1
189 100 DO 10 I=1,N
190 DO 20 J=1,N
191 IF(J.EQ.I) GO TO 25
192 IF(ITETA.EQ.-1.AND.IX(J).EQ.0) GO TO 25
193 IY(J)=IX(J)+ITETA
194 GO TO 20
195 25 IY(J)=IX(J)
196 20 CONTINUE
197 ALFB=DIF(XLAG,IY,I)
198 ALFA=DIF(XLAG,IX,I)
199 RATIO(I)=ALFB/ALFA
200 10 CONTINUE

```

```

C
C RANK THE RATIO OF FIRST DIFFERENCES OF EACH COMPONENT
C
201 CALL SIRALA(RATIO, IRANK)
202 DO 30 I=1,A
203 DO 35 J=1,N
204 IF (IRANK(J).GT.1) GO TO 38
205 IF (ITETA.EQ.-1.AND.IX(J).EQ.0) GO TO 38
206 IY(J)=IX(J)+ITETA
207 GO TO 35
208 38 IY(J)=IX(J)
209 35 CONTINUE

C
C CALCULATE ITS E(NORS/X)
210 VY=VNORS(XLAG, IY)

C
C COMPARE WITH THE INCUMBENT SOLUTION
211 IF (VY.LT.VX) GO TO 50
212 30 CONTINUE
213 IF (ITETA.EQ.-1) GO TO 90
214 ITETA=-1
215 GO TO 100
216 50 DO 60 I=1,N
217 60 IX(I)=IY(I)
218 VX=VY
219 GO TO 110
220 90 DO 70 I=1,N
221 70 IX(I)=IX(I)
222 VX=VX
223 110 RETJRN
224 END

225 SUBROUTINE ARAMA(XLAG, IX, VX, IXO, VXO)
C
C REDUCES THE SET OF POTENTIAL CANDIDATES BY CONTRADICTION
C
226 DIMENSION IX(100), IXO(100), IY(100), IZ(100), ISIR(100), JARR(100), MO
* C(100)
227 COMMON N,M/BLOCK5/ITETA/BLOCK8/NOZE,JARR/BLOCK6/WLAM
228 COMMON /COUNT/NITER,NBAS(50),NSEA(50),NTAR(50),NARA(50),
*NPA2(50),NPA3(50),NVNOR
229 NARA(NITER)=NARA(NITER)+1

C
230 WLAM=XLAG
231 ITETA=1
232 500 ICOUN=0
233 NOZE=0
234 IKODE=0
235 400 DO 10 I=1,N
236 IF (NOZE.EQ.0) GO TO 19
237 DO 15 II=1,NOZE
238 IF (JARR(II).EQ.1) GO TO 10
239 15 CONTINUE
240 19 DO 20 J=1,N
241 IF (J.EQ.1) GO TO 25
242 IF (ITETA.EQ.-1.AND.IX(J).EQ.0) GO TO 25
243 IF (NOZE.EQ.0) GO TO 29
244 DO 28 JJ=1,NOZE
245 IF (JARR(JJ).EQ.J) GO TO 25
246 28 CONTINUE

```

```

247 29 IY(J)=IX(J)+ITETA
248 GO TO 20
249 25 IY(J)=IX(J)
250 20 CONTINUE
251 ALFA2=DIF(XLAG,IY,I)
252 IF(ALFA2) 11,12,12
253 11 ICOUN=ICOUN+1
254 ALFA1=DIF(XLAG,IX,I)
255 RATIO=ALFA2/ALFA1
256 ISIR(ICOUN)=I
257 IF(ICOUN.EQ.1) GO TO 41
258 IF(RATIO.LT.RMAX) GO TO 10
259 41 KMAX=RATIO
260 JJ=I
261 GO TO 10
262 12 NOZE=NOZE+1
263 JARR(NZE)=I
264 10 CONTINUE
265 IF(ICODE.EQ.1) GO TO 110
266 IF(ICOUN.EQ.0) GO TO 110
267 DO 30 I=1,N
268 IF(I.EQ.JJ) GO TO 33
269 GO TO 35
270 33 IF(ITETA.EQ.-1.AND.IX(I).EQ.0) GO TO 35
271 IY(I)=IX(I)+ITETA
272 GO TO 30
273 35 IY(I)=IX(I)
274 30 CONTINUE
275 IF(ICOUN.EQ.1) GO TO 60
276 NJC=0
277 DO 50 J=1,ICOUN
278 IF(ISIR(J).EQ.JJ) GO TO 50
279 JL=ISIR(J)
C
C PROVE BY CONTRADICTION THAT JL-COMPONENT CAN'T BE 1
280 CALL PART1(IY,JJ,JL,IZ,ICODE)
281 IF(ICODE.EQ.0) GO TO 55
282 DO 59 J1=1,N
283 59 IY(J1)=IZ(J1)
284 GO TO 50
285 55 NJC=NJC+1
286 MOUC(NJC)=J
287 50 CONTINUE
288 IF(NJC.EQ.J) GO TO 60
289 IF(NJC.GT.1) GO TO 40
290 JUC=MOUC(NJC)
291 IF(ITETA.EQ.-1.AND.IY(JUC).EQ.0) GO TO 60
292 IY(JUC)=IY(JUC)+ITETA
293 GO TO 60
294 40 CALL PART2(NUC,MOUC,IY,XLB)
295 IF(XLB.GE.VX) GO TO 100
296 CALL PART3(IY,VX,JJ,NJC,MOUC,IZ)
297 DO 61 J1=1,N
298 61 IY(J1)=IZ(J1)
299 60 VY=VNCRS(XLAG,IY)
300 IF(VY.GE.VX) GO TO 100
301 VX0=VY
302 DO 65 I=1,N
303 65 IX0(I)=IY(I)
304 GO TO 300

```

```

305      100 ICOUN=ICOUN-1
306          IF( ICOUN.EQ.0) GO TO 110
307          NOZE=NOZE+1
308          JARR(NOZE)=JJ
309          IKODE=1
310          GO TO 400
311      110 IF(ITETA.EQ.-1) GO TO 200
312          ITETA=-1
313          GO TO 500
314      200 DO 70 I=1,N
315          70 IX(I)=IX(I)
316          VXO=VX
317      300 RETURN
318          END

319      SUBROUTINE PART1(IX,JJ,JL,IY,ICODE)
C
C      PERFORMS A TEST BY SIMPLY EVALUATING AN UPPER BOUND FOR THE LINEAR
C      IF ICODE=1 , TEST SUCCEEDED
C      IF ICODE=0 TEST FAILED
C
320      DIMENSION IX(100),IY(100),IZ(100),XC(50),XA(50),JARR(100)
321      COMMON N,M/BLOCKS/I TETA/BLOCK6/WLAM/BLOCK8/NOZE,JARR
322      *NPA2(50),NPA3(50),NVNOR/TAB/POISON(50,100)
C
323      NPA1(NITER)=NPA1(NITER)+1
324      M1=M+1
325      XLAG=WLAM
326      B0=-DIF(XLAG,IX,JL)
327      IF(B0.GE.0.0) GO TO 50
328      DO 10 I=1,N
329      10 IF(I.EQ.JJ) GO TO 11
330          IZ(I)=IX(I)
331          GO TO 10
332      11 IZ(I)=IX(I)-ITETA
333      10 CONTINUE
334      B1=-DIF(XLAG,IZ,JJ)
335      DO 20 I=1,M1
336      20 IXS=IX(JL)+I
337          T1=PDFOP(IXS,JL)
338          T2=POISON(IXS,JL)
339          XC(I)=-T1/T2
340          IXP=IX(JJ)+I+1
341          P1=PDFOP(IXP,JJ)
342          P2=POISON(IXP,JJ)
343          XA(I)=-P1/P2
344      20 CONTINUE
345      TER1=1.
346      TER2=1.
347      IF(B1.LT.0.0) GO TO 22
348      W1=0.0
349      GO TO 23
350      22 W1=(1.1)*B0/B1
351      23 TERM=W1*B1
352      DO 30 I=1,M1
353          W1=XC(I)-W1*XA(I)
354          IF(W1) 30,30,35
355      30 DO 40 K=1,N
356          DO 37 KK=1,NOZE

```

```

357       IF (JARR(KK).EQ.K) GO TO 36
358 37 CONTINUE
359       IXK=IX(K)+1+1
360       TER1=TER1+POISON(IXK,K)
361       GO TO 38
362 36 IXK=IX(K)+1
363       TER1=TER1+POISON(IXK,K)
364 38 IF (K.EQ.JJ) GO TO 39
365       IYK=IX(K)+1-1
366       GO TO 41
367 39 IYK=IX(K)+1+1
368 41 TER2=TER2+POISON(IYK,K)
369 40 CONTINUE
370       UI=TER1-TER2
371       TERM=TERM+UI*WI
372 30 CONTINUE
373       WM=TERM-BO
374       ICODE=0
375       IF (WM) 50,55,55
376 50 ICODE=1
377       IF (ITETA.EQ.-1.AND.IX(JL).EQ.0) GO TO 55
378       IX(JL)=IX(JL)+ITETA
379 55 DO 60 I=1,N
380 60 IY(I)=IX(I)
381       RETURN
382       END

383       SUBROUTINE PART2(INJC,MOUC,IY,XLB)
C
C IF PART1 FAILS, THIS FINDS AN UPPER BOUND FOR THE SUM OF FIRST
C DIFFERENCES OF THE UNDETERMINED COEFFICIENTS
C
384       DIMENSION IY(100),MOUC(100),IZ(100)
385       COMMON N,M/BLOCK5/ITETA/BLOCK6/WLAM
386       COMMON /COUNT/ITER,NBAS(50),NSEA(50),NTAR(50),NARA(50),
+NPAA2(50),NPAA3(50),VNDJR
C
387       NPAA2(ITER)=NPAA2(ITER)+1
388       SUM=0.0
389       DO 10 I=1,NUC
390       IXK=MOUC(I)
391       IXX1=IXK-1
392       IF (IXK1.EQ.0) GO TO 15
393       DO 20 J=1,IXK1
394       IF (ITETA.EQ.-1.AND.IY(J).EQ.0) GO TO 25
395       IZ(J)=IY(J)+ITETA
396       GO TO 20
397 25 IZ(J)=IY(J)
398 20 CONTINUE
399       NXX=IXK
400       GO TO 35
401 15 NXX=1
402 35 DO 30 J=NXX,N
403 30 IZ(J)=IY(J)
404       TERM=DIF(WLAM,IZ,IXK)
405       IF (TERM.LT.0.0) GO TO 40
406       TERM=0.0
407 40 SUM=SUM+TERM
408 10 CONTINUE
409       XLB=SUM+VNORS(WLAM,IY)

```

```

410      RETURN
411      END

412      SUBROUTINE PART3(IX,VX,JJ,NUC,MOUC,IY)
C
C   PERFORMS THE SAME TEST AS PART1, BUT SETS UP THE LINEAR PROGRAM EX
C
413      DIMENSION IX(100), IY(100), IZ(100), JARR(100), MOUC(100), KOUC(100),
*10,10)
414      COMMON N,M/BLOCK5/ITETA/BLOCK8/NOZE,JARR/BLOCK9/NROW,NCOL/BLOCK6
*1,A*
415      COMMON /COUNT/NITER,NBAS(50),NSEA(50),NTAR(50),NARA(50),NPA1(50)
*PA2(50),NPA3(50),NVNOR/TAB/POISON(50,100)
416      NPA3(NITER)=NPA3(NITER)+1
C
417      NUCB=NUC
418      XLAG=WLAM
419      NROW=M+2
420      NCOL=M+3
421      NCOL1=NCOL-1
422      DO 9 I=1,N
423      IF(I.EQ.JJ) GO TO 8
424      IZ(I)=IX(I)
425      GO TO 9
426      8 IZ(I)=IX(I)-ITETA
427      9 CONTINUE
428      DO 10 ISAY=1,NUCB
429      JL=MOJC(ISAY)
430      Y(1,1)=0.0
431      DO 11 J=2,NCOL1
432      TER1=1.0
433      TER2=1.0
434      DO 40 K=1,N
435      DO 37 KK=1,NOZE
436      IF(JARR(KK).EQ.K) GO TO 36
437      37 CONTINUE
438      IXK=IZ(K)+J
439      GO TO 38
440      36 IXK=IZ(K)+J-1
441      38 TER1=TER1+POISON(IXK,K)
442      IF(K.EQ.JJ) GO TO 39
443      IYK=IZ(K)+J-1
444      GO TO 41
445      39 IYK=IZ(K)+J
446      41 TER2=TER2+POISON(IYK,K)
447      40 CONTINUE
448      11 Y(1,J)=TER1-TER2
449      Y(1,NCOL)=-DIF(XLAG,IZ,JJ)
450      DO 15 I=2,NROW
451      IXS=IX(JL)+I-1
452      Y(I,1)=PDFOP(IXS,JL)/POISON(IXS,JL)
453      IXP=IX(JJ)+I-2
454      IXR=IX(JJ)+I-1
455      15 Y(I,NCOL)=PDFOP(IXP,JJ)/POISON(IXP,JJ)
456      DO 25 I=2,NROW
457      DO 25 J=2,NCOL1
458      IF(J.EQ.1) GO TO 27
459      Y(I,J)=0.0
460      GO TO 25
461      27 Y(I,J)=-1.0

```

```

462      25 CONTINUE
463      BQ=-DIF(XLAG,IX,JL)
464      DO 210 I=1,NROW
465      210 CONTINUE
466      CALL XLPRO(Y,VGBJ)
467      FARK=VORJ-BQ
468      IF(FARK.GT.0.0) GO TO 10
469      IF(IX(JL).EQ.0.AND.ITETA.EQ.-1) GO TO 50
470      IX(JL)=IX(JL)+ITETA
471      50 KUC=N*IC-1
472      IF(KUC.EQ.0) GO TO 10
473      NKUC=0
474      DO 55 J=1,NUC
475      IF(MOUC(J).EQ.JL) GO TO 56
476      IF(NKUC.EQ.1) GO TO 57
477      KJUC(J)=MOUC(J)
478      GO TO 55
479      56 NKUC=1
480      GO TO 55
481      57 J1=J-1
482      KJUC(J1)=MOUC(J)
483      55 CONTINUE
484      CALL PART2(KUC,KJUC,IX,XLB)
485      IF(XLB.GE.VX) GO TO 500
486      NUC=KUC
487      DO 60 I=1,NUC
488      60 MOUC(I)=KJUC(I)
489      10 CONTINUE
490      500 DO 70 I=1,N
491      70 IY(I)=IX(I)
492      RETURN
493      END

494      FUNCTION DIF(XLAG,IX,K)
C
C      CALCULATES THE FIRST DIFFERENCES
C
495      DIMENSION IX(100)
496      COMMON N,M/BLOCK1/C(100)/TAB/POISON(50,100)
497      DOUBLE PRECISION PRO,SUM
498      M1=M+1
499      SUM=0.0+00
500      DO 20 J1=1,M1
501      PRO=1.0+00
502      DO 30 I=1,N
503      IXJ=IX(I)+J1
504      IF(I.EQ.K) GO TO 25
505      PRO=PRO*POISON(IXJ,I)
506      GO TO 30
507      25 PRO=PRO*PDFOP(IXJ,I)
508      30 CONTINUE
509      20 SJM=SJM+PRO
510      IXM=IX(K)+M+2
511      DIF=-SUM-1.+POISON(IXM,K)+XLAG*C(K)
512      RETURN
513      END

514      FUNCTION VNORS(XLAG,IX)
C
C      EVALUATES THE FUNCTION E(NORS/X)+L.C.X

```

```

515      C      DIMENSION IX(100),PH(100,20),PRO(20)
516      COMMON N,M
517      COMMON /COUNT/NITER,NBAS(50),NSEA(50),NTAR(50),NARA(50),NPA1(50)
518      *NPA2(50),NPA3(50),NVNOR/TAB/POISON(50,100)
519      DOUBLE PRECISION SJM,PRO,TOPL,P4,XW
520      NVNOR=NVNOR+1
521      EPS=.00001
522      M1=M+1
523      SUM=0.0+00
524      DO 20 JJ=1,M1
525      PRO(JJ)=1.0+00
526      DO 25 II=1,N
527      IXJ=IX(II)+JJ-1
528      IF(IXJ.GT.0) GO TO 24
529      XX=PDFOP(IXJ,II)
530      GO TO 25
531      24 XX=POISON(IXJ,II)
532      25 PRO(JJ)=PRO(JJ)+XX
533      20 SJM=SJM+1.-PRO(JJ)
534      TOPL=0.0+00
535      DO 30 I=1,N
536      K=0
537      40 MXK=IX(I)+M+2+K
538      WX=PDFOP(MXK,I)
539      W=FLOAT(K+1)*WX
540      IF(WX.LT.EPS) GO TO 30
541      TOPL=TOPL+W
542      K=K+1
543      GO TO 40
544      30 CONTINUE
545      VNORS=SUM+TOPL+XLAG*DJTPR(IX)
546      RETURN
547      END

547      SUBROUTINE TABLE(POISON)
548      C
549      C      CALCULATES THE CUMULATIVE POISSON SUMS
550      C
551      DIMENSION POISON(50,100)
552      COMMON N,M/BLOCK2/XLAM(100)
553      DOUBLE PRECISION EPS,TERM,SUMX
554      EPS=1.0-06
555      DO 10 J=1,N
556      TERM=1.0+00
557      SJMX=1.0+00
558      DO 20 I=1,50
559      TERM=TERM*XLAM(J)/FLOAT(I)
560      IX=I
561      IF(TERM.LT.EPS) GO TO 25
562      SUMX=SJMX+TERM
563      POISON(I,J)=EXP(-XLAM(J))*SUMX
564      IF(POISON(I,J).GT.1.0) POISON(I,J)=1.0
565      20 CONTINUE
566      10 IF(IX.EQ.50) GO TO 10
567      DO 30 I=.X,50
568      POISON(I,J)=1.0
569      30 CONTINUE
570      RETURN
571      END

```

```

569      FUNCTION PDFOP(K,I)
      C
      C      CALCULATES THE INDIVIDUAL POISSON TERMS
      C
570      COMMON /BLOCK2/XLAM(100)
571      DOUBLE PRECISION SUM,WM
572      SUM=0.0+0J
573      IF (K.LT.2) GO TO 10
574      DO 15 J=2,K
575      SUM=SUM+ALOG(FLOAT(J))
576      15 CONTINUE
577      WM=-XLAM(I)+K*ALOG(XLAM(I))-SUM
578      IF(WM.LT.-15.0) GO TO 20
579      PDFOP=DEXP(WM)
580      GO TO 25
581      20 PDFOP=.0
582      25 RETURN
583      END

584      FUNCTION DOTPR(IX)
      C
      C      CALCULATES THE DOT PRODUCT C.X
      C
585      DIMENSION IX(100)
586      COMMON N,M/BLOCK1/C(100)
587      SUM=0.0
588      DO 10 I=1,N
589      XX=FLOAT(IX(I))
590      10 SUM=SUM+XX*C(I)
591      DOTPR=SUM
592      RETURN
593      END

594      SUBROUTINE SIRALA(X,IR)
      C
      C      THIS ARRANGES AN ARRAY OF ELEMENTS IN ASCENDING ORDER
      C      IR(K) IS THE RANK OF ITEM K IN THE LIST
      C
595      DIMENSION X(100), IR(100), ISEQ(100), Y(100)
596      COMMON N,M
597      DO 9 I=1,N
598      9 ISEQ(I)=I
599      DO 10 I=1,N
600      XMIN=X(I)
601      IMINJ=I
602      IMIN=ISEQ(I)
603      N1=N-I+1
604      DO 20 J=1,N1
605      IF(X(J).LT.XMIN) GO TO 21
606      GO TO 20
607      21 XMIN=X(J)
608      IMIN=ISEQ(J)
609      IMINJ=J
610      20 CONTINUE
611      IR(IMIN)=I
612      IF(N1.EQ.1) GO TO 10
613      N2=N1-1
614      DO 30 J=1,N2
615      IF(J.LT.IMINJ) GO TO 33
616      J1=J+1

```

```

617      ISEQ(J)=ISEQ(J1)
618      Y(J)=X(J1)
619      GO TO 29
620 33 Y(J)=X(J)
621 29 X(J)=Y(J)
622 33 CONTINUE
623 17 CONTINUE
624      RETURN
625      END

626      SUBROUTINE XLPRO(Y,V0BJ)
C
C      SOLVES A LINEAR PROGRAM OF THE TYPE
C      MAX CX ST AX<B,X>0
C
627      DIMENSION Y(10,20),X(10,20),ISNB(10),ISBV(10),MROW(10)
628      COMMON /BLOCK9/NROW,NCOL
C
629      ISBV(1)=0
630      ISNB(1)=NCOL+1
631      DO 10 I=2,NROW
632 10 ISBV(I)=I-2+NCOL
633      DO 11 J=2,NCOL
634 11 ISNB(J)=J-1
635      NROW1=NROW-1
636      NCOL1=NCOL-1
637      NCOL2=NCOL+1
638      NCOL3=NCOL+NROW1
639      DO 12 I=1,NROW
640      DO 12 J=NCOL2,NCOL3
641      IM=I-1
642      JM=J-NCOL
643      IF(JM.EQ.IM) GO TO 13
644      Y(I,J)=0.0
645      GO TO 12
646 13 Y(I,J)=1.0
647 12 CONTINUE
648 100 NVC=0
649      NPV=0
650      NPC=0
651      NVV=0
652      DO 15 J=2,NCOL
653      IF(Y(1,J).LT.0.0) GO TO 16
654      NNC=NNC+1
655      IF(NNC.EQ.NCOL1) GO TO 500
656      GO TO 15
657 16 NPC=NPC+1
658      IF(NPC.EQ.1) GO TO 17
659      IF(Y(1,J).GT.YMIN) GO TO 15
660 17 YMIN=Y(1,J)
661      JMIN=J
662 15 CONTINUE
663      IEV=ISNB(JMIN)
664      NIVA=1
665      DO 20 I=2,NROW
666      IF(Y(I,JMIN).LE.0.0) GO TO 25
667      NVV=NVV+1
668      TETA=Y(I,1)/Y(I,JMIN)
669      IF(NVV.EQ.1) GO TO 23
670      IF(TETA-TMIN) 23,20,20

```

```

671      23 TMIN=TETA
672      IMIN=I
673      NMVA(NMVA)=IMIN
674      GO TO 22
675      25 NPV=NPV+1
676      IF (NPV.EQ.NROW) GO TO 450
677      20 CONTINUE
678      DO 26 I=2,NROW
679      IF (Y(I,JMIN).LE.0.0) GO TO 26
680      TETA=Y(I,1)/Y(I,JMIN)
681      IF (TETA.GT.TMIN) GO TO 26
682      IF (I.EQ.IMIN) GO TO 26
683      NMVA=NMVA+1
684      NROW(NMVA)=I
685      26 CONTINUE
686      IF (NMVA.EQ.1) GO TO 200
687      DO 80 I=1,NMVA
688      IK=NROW(I)
689      JPDS=J
690      DO 80 J=NCOL2,NCOL3
691      X(IK,J)=Y(IK,J)/Y(IK,JMIN)
692      IF (X(IK,J).GT.0.0) GO TO 85
693      GO TO 80
694      85 JPDS=JPDS+1
695      IF (JPDS.GT.1) GO TO 80
696      IF (IK.GT.NROW(1)) GO TO 87
697      GO TO 93
698      87 IF (J-JKMIN) 80,90,93
699      90 IF (X(IMIN,JKMIN).LE.X(IK,J)) GO TO 80
700      93 IMIN=IK
701      JKMIN=J
702      80 CONTINUE
703      200 IDV=ISBV(IMIN)
704      ISNB(JMIN)=IDV
705      ISBV(IMIN)=IEV
706      DO 40 I=1,NROW
707      DO 40 J=1,NCOL3
708      IF (I.EQ.IMIN) GO TO 55
709      IF (J.EQ.JMIN) GO TO 60
710      X(I,J)=Y(I,J)-(Y(I,JMIN)*Y(IMIN,J)/Y(IMIN,JMIN))
711      GO TO 40
712      55 IF (J.EQ.JMIN) GO TO 57
713      X(I,J)=Y(I,J)/Y(IMIN,JMIN)
714      GO TO 40
715      57 X(I,J)=1./Y(I,J)
716      GO TO 40
717      60 X(I,J)=-Y(I,J)/Y(IMIN,JMIN)
718      40 CONTINUE
719      DO 70 I=1,NROW
720      DO 70 J=1,NCOL3
721      70 Y(I,J)=X(I,J)
722      GO TO 100
723      450 WRITE(6,250)
724      250 FORMAT(5X,'UNBOUNDED SOLUTION')
725      500 RETURN
726      END

```

SENTRY

THE SADDLE POINT SEARCH ALGORITHM

PROBLEM NUMBER 1

NUMBER OF ITEMS= 5 BUDGET=\$ 25000.00

ITEM	COST/UNIT	DEMAND	INITIAL SOLUTION	OPTIMAL SOLUTION
# 1	2980.00	2.10	2	2
# 2	1751.00	1.50	2	2
# 3	462.00	1.20	2	3
# 4	1500.00	5.00	6	7
# 5	345.00	3.50	6	6

E(NORS/X)= 1.06282 , COST(X)=\$ 23418.00

LAGRANGE MULTIPLIER= 0.000090

OF ITERATIONS= 6

OF NORS EVALUATIONS= 141

OF STEPS IN EACH ITERATION

ITERATION #	STEP1	STEP2	STEP3	STEP4	PART1	PART2	PART3
1	1	1	1	1	5	0	0
2	1	5	5	2	0	0	0
3	1	4	4	2	3	0	0
4	1	2	2	1	1	0	0
5	1	2	2	1	0	0	0
6	1	2	2	1	2	0	0

STATEMENTS EXECUTED= 273427

CORE USAGE OBJECT CODE= 31504 BYTES, ARRAY AREA= 48572 BYTES, TOTAL

DIAGNOSTICS NUMBER OF ERRORS= 0, NUMBER OF WARNINGS= 0, P

COMPILE TIME= 0.52 SEC, EXECUTION TIME= 5.02 SEC, 12.27.33 TUE

C\$STOP

APPENDIX B

B.1 Detail Documentation of Greedy Algorithm Computer Program

B.2 FORTRAN Listing of Greedy Algorithm Computer Program and Solution Output

APPENDIX B.1

DETAIL DOCUMENTATION OF

GREEDY ALGORITHM

COMPUTER PROGRAM

B.1.1

APPENDIX B.1

DETAIL DOCUMENTATION OF GREEDY ALGORITHM

COMPUTER PROGRAM

General

In addition to the explanatory comment statements that are provided in the program listing of the algorithm (See Appendix B.2), formulations of computational formulas, and detailed statement by statement interpretations are provided in this appendix for several critical parts of the program. Since some of the subroutines are either essentially the same as those of the D029 program, or rather simple and straight forward, the effort below is devoted mainly to three subroutines: TABLE, OPTIM, and MARGAN. A flow chart describing the relationship between OPTIM and MARGAN is also provided.

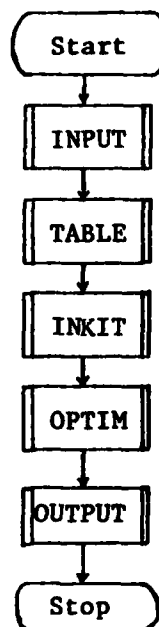
Terminology and Notations Used in Notes on Subprograms

$a_i = \text{NAP}(I)$	number of applications of item i
$b_i = \text{BO}(I+1)$	conditional operational rate, given i planes available for cannibalization
$q_i = \text{KIT}(I)$	quantity of item i in the kit
w	weighting factor of $\Delta E(\text{SDO})$ used in r_i calculation
$F_i(x)$	probability of demand $\leq x$ for item i
ITOT	total number of items considered
MUP	upper bound of subscript of b_i
$\text{NAP}(I) = a_i$	
NMAX_i	maximum possible number of demands for item i
$\text{NOPS} = \text{UE}$	number of planes in squadran
$\text{NORS} = \text{NMCS}$	number of Not Operationally Ready Supply aircraft
$\text{POISON}(I,J)$	cumulative Poisson demand $F_i(j-1)$ for item i

$r_i = \text{RATIO}(I)$ Weighted average improvement of $E(\text{NORS})$ & $E(\text{SDO})$
 per dollar invested
 SDO number of Stock Due Outs, or backorders
 $S(N)$ probability of $\text{NORS} \leq N-1$
 $\text{UE} = \text{NOPS}$

Notes on Main Program

This main program contains the various subroutine calling statements shown below.



Notes on Subroutine INPUT

This subroutine reads input data, calculates average demand rates, and obtains KIT 5718 and its target value. The code is essentially the same as the code for the D029 algorithm. LRU-RR, LRU-RRR, SRU-RR, and SRU-RRR are differentiated and processed in different manners as required.

Notes on Subroutine TABLE

This subroutine calculates the individual cumulative Poisson probabilities for each item.

Poisson probability is $P_I(x) = \frac{\lambda_I^x e^{-\lambda_I}}{x!}$, $x = 0, 1, 2, \dots$ and $I = 1, 2, \dots, ITOT$

hence, $P_I(x+1) = \frac{\lambda_I}{x+1} P_I(x)$

e.g. $P_I(0) = e^{-\lambda_I}$, $P_I(1) = \lambda_I \cdot P_I(0)$, $P_I(2) = \frac{\lambda}{2} P_I(1), \dots$ etc.

Then the cumulative Poisson probability, $F_I(x)$, can be expressed as follows:

$$F_I(x) = \sum_{j=0}^x \frac{\lambda_I^j e^{-\lambda_I}}{j!} = F_I(x-1) + P_I(x)$$

By employing the two dimensional array POISON (500,60), $F_I(j)$ of item i can be stored in POISON (I, j+1), since POISON (I, 1) = $F_I(0)$ for item I .

The main segment of code for this subroutine is shown below.

```

ISN 0007      TERM=5.0-12
ISN 0008      DO 10 I=1,ITOT
ISN 0009      WWW=DBLE(XMU(I))
ISN 0010      TERM=DEXP(-WWW)
ISN 0011      POISON(I,1)=TERM
ISN 0012      DO 20 J=2,60
ISN 0013      IF( TERM.LT. TERL.AND. POISON(I,J-1).GT.9.0-01) GO TO 30
ISN 0015      TERM=TERM*WWW/DFLOAT(J-1)
ISN 0016      POISON(I,J)=POISON(I,J-1)+TERM
ISN 0017      20 CONTINUE
ISN 0018      GO TO 10
ISN 0019      30 DO 40 JX=J,60
ISN 0020      40 POISON(I,JX)=1.0+0J
ISN 0021      10 CONTINUE
ISN 0022      RETURN
ISN 0023      END

```

Line 7 gives the criteria for stopping further computations of the upper-tail of the Poisson distribution. At lines 10 and 11, POISON (I, 1) = $F_I(0) = e^{-\lambda_I}$. Line 13 checks if the stopping criteria is met. Lines 15 and 16 calculate

all $P_I(x)$ and $F_I(x)$ for each item. Lines 19 and 20 assign to all the remaining cumulative probability terms the value of 1.

Notes on Subroutine INKIT

This subroutine is essentially the same as that of D029. It appropriately increases KIT quantities to ensure that the probability of any item being the cause of half a squadron being down is less than 0.50.

Notes on Subroutine OPTIM

This routine calls marginal analysis subroutine, MARGAN, and updates the parameters b_i 's. It terminates if the new set of b_i 's are sufficiently close to the values of the previous set.

$$BO(J) = b_{j-1} = \sum_{\ell=1}^{ITOT} \pi_{\ell} F_{\ell} [q_{\ell} * + a_{\ell} (j-1)]$$

$$= \sum_{\ell=1}^{ITOT} \pi_{\ell} \text{POISON} [KIT(\ell) + NAP(\ell) * (J-1) + 1]$$

The measure of closeness between two sets of b_i 's and b_i' 's is the sum of squares = $\sum_i (b_i - b_i')^2$.

Referring to the code of the main body of this subroutine on the next page and the combined flow chart that follows, the algorithm can be clearly explained.

```

C
ISN 0017      100 CALL MARGAN(KIT,PRICE)
C
ISN 0018      NITER=NITER+1
C
C            STEP 2
C            UPDATE THE PARAMETERS B
C
ISN 0019      SUMSQ=0.0+00
ISN 0020      DO 10 J=1,MUP
ISN 0021      PRU=1.0+00
ISN 0022      DO 20 I=1,ITOT
ISN 0023      KITPJ=KIT(I)+(J-1)*NAP(I)+1
ISN 0024      IF(KITPJ.GT.60) GO TO 20
ISN 0026      PRU=PRU*POISSON(I,KITPJ)
ISN 0027      IF( PRU.LT.EPSQ) GO TO 15
ISN 0029      20 CONTINUE
ISN 0030      15 SUMSQ=SUMSQ+((BO(J)-PRU)**2)
ISN 0031      BO(J)=PRU
ISN 0032      10 CONTINUE
C
C            STEP 3
C            COMPARE THE NEW VALUES OF B WITH PREVIOUS VALUES
C            TERMINATE IF THEY ARE SUFFICIENTLY CLOSE
C
ISN 0033      IF( SUMSQ.LT.EPSQ) GO TO 30
ISN 0035      GO TO 100
C
ISN 0036      30 CALL CALC(KIT,ENDRS,SCD)
ISN 0037      RETURN
ISN 0038      END

```

Line 17 calls the marginal analysis subroutine in order to find the optimum kit under the current b_i parameters. Once this is done, it comes back and starts to update the b_i 's.

Line 24 provides a safeguard against running out of the limit of the Poisson array. At line 26, b_i 's are calculated by multiplying all pertinent cumulative probability terms together. Line 27 gives a lower bound check on b_i to save further calculations, since when $b_i \leq \text{EPSQ} = 5 \times 10^{-11}$, we can treat it as zero without practically affecting actual results. Line 30 gives a measure of the difference between the latest two consecutive sets of b_i 's. Line 33 checks to see if the difference is small enough to stop. If not, the

algorithm continues iterations until the above stopping criteria is satisfied. Finally, before returning to the main program, the performance of the optimal kit is evaluated.

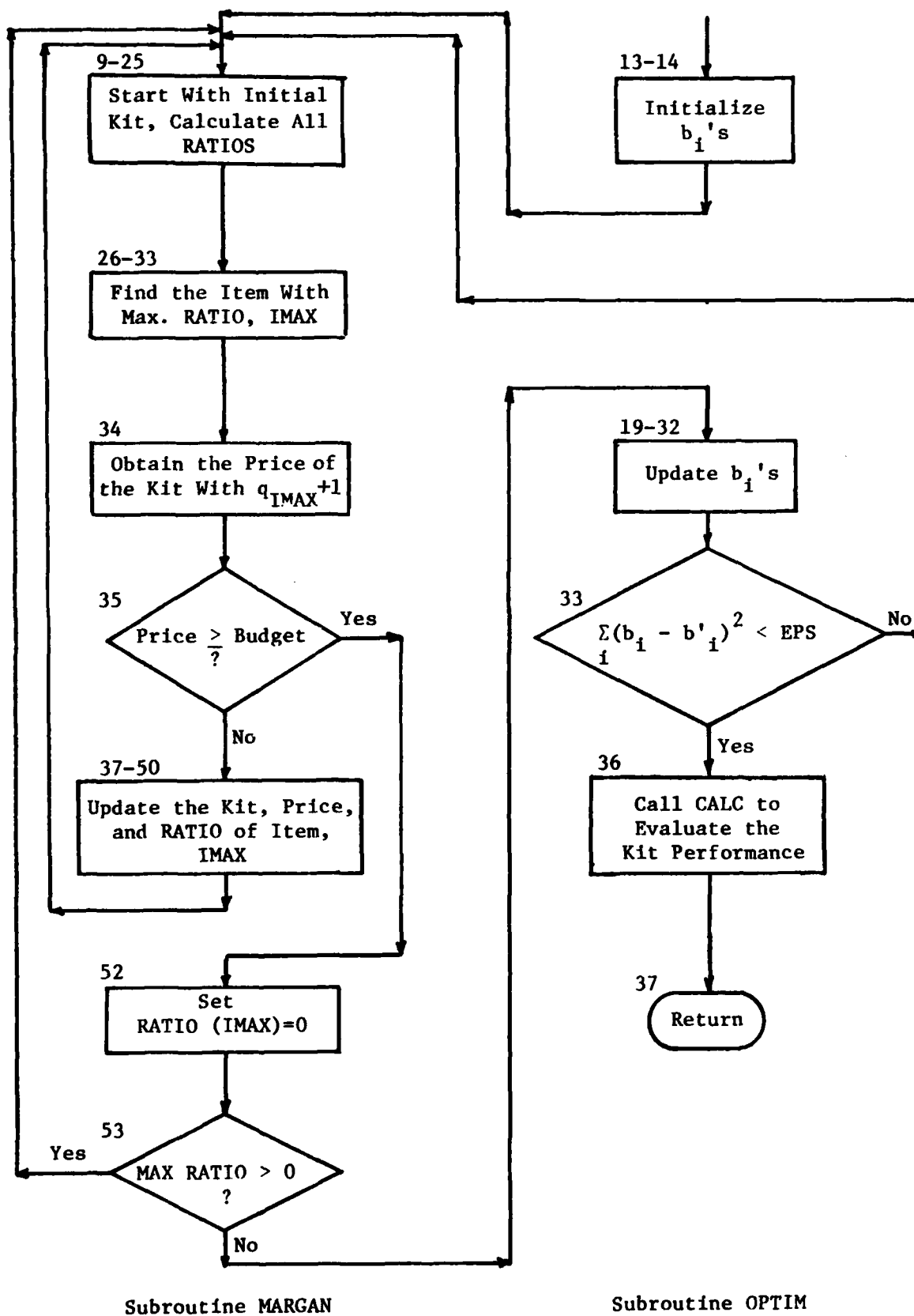


Figure B.1. Flowcharts for Subroutines MARGAN and OPTIM

Notes on Subroutine MARGAN

This subroutine performs the marginal analysis. It starts with the initial kit and calculates the RATIO of return, (r_i), per dollar invested for each item i . This ratio is the weighted average of the improvements of $E(\text{NORS})$ and $E(\text{SDO})$.

$$r_i = \frac{\Delta_i E(\text{NORS}) + w \Delta_i E(\text{SDO})}{c_i}$$

$$\text{where } \Delta_i E(\text{NORS}) = \left[\sum_{\ell=1}^{\text{ITOT}} (\text{MUP}_{\ell} + 1) - \sum_{j=0}^{\text{MUP}} b_j \sum_{\ell=1}^{\text{ITOT}} \ell n F_{\ell}(q_{\ell} + ja_{\ell}) \right] \\ - \left[\sum_{\ell=1}^{\text{ITOT}} (\text{MUP}_{\ell} + 1) - \sum_{j=0}^{\text{MUP}} b_j \sum_{\ell=1}^{\text{ITOT}} \ell n F_{\ell}(q'_{\ell} + ja_{\ell}) \right]$$

$$\text{where } q'_{\ell} = q_{\ell} + 1 \text{ when } \ell = i$$

$$q'_{\ell} = q_{\ell} \text{ when } \ell \neq i$$

$$\Delta_i E(\text{NORS}) = \sum_{j=0}^{\text{MUP}} b_j \left[\ell n F_i(q_i + 1 + ja_i) - \ell n F_i(q_i + ja_i) \right] \\ = \sum_{j=0}^{\text{MUP}} b_j \ell n \frac{F_i(q_i + ja_i + 1)}{F_i(q_i + ja_i)} \\ = \sum_{j=0}^{\text{MUP}} b_j \ell n \frac{\text{POISON}(i, q_i + ja_i + 2)}{\text{POISON}(i, q_i + ja_i + 1)}$$

$$\text{Since } \text{POISON}(i, 1) = F_i(0)$$

$$E(\text{SDO}) = \sum_{\ell=1}^{\text{ITOT}} \sum_{x=q_{\ell}}^{\text{NMAX}_{\ell}-1} [1 - F_{\ell}(x)] = \sum_{\ell=1}^{\text{ITOT}} \sum_{j=0}^{\text{JUP}_{\ell}} [1 - F_{\ell}(q_{\ell} + j)]$$

$$\text{where } \text{JUP}_{\ell} = (\text{NMAX} - 1) - q_{\ell}$$

$$= [\text{KIT}(\ell) + (\text{NOPS}-1) * \text{NAP}(\ell) + 1] - 1 - \text{KIT}(\ell)$$

$$= (\text{NOPS}-1) * \text{NAP}(\ell) + 1$$

$$\text{Then } \Delta_1 E(\text{SDO}) = \sum_{\ell=1}^{\text{ITOT}} \sum_{j=0}^{\text{JUP}_{\ell}} [1 - F_{\ell}(q_{\ell} + j)] - \sum_{\ell=1}^{\text{ITOT}} \sum_{j=0}^{\text{JUP}_{\ell}} [1 - F_{\ell}(q_{\ell} + j)]$$

$$\text{where } q_{\ell} = q_{\ell} + 1 \quad \text{when } \ell = i$$

$$q_{\ell} = q_{\ell} \quad \text{when } \ell \neq i$$

$$= \sum_{j=0}^{\text{JUP}_i} \{1 - F_i(q_i + j) - [1 - F_i(q_i + 1 + j)]\}$$

$$\text{i.e. } \Delta_1 E(\text{SDO}) = \sum_{j=0}^{\text{JUP}_i} [F_i(q_i + 1 + j) - F_i(q_i + j)]$$

$$= F_i(q_i + 1 + \text{JUP}_i) - F_i(q_i)$$

$$\text{or} = \text{POISON}(i, q_i + 1 + \text{NMAX}_i - 1 - q_i + 1) - \text{POISON}(i, q_i + 1)$$

$$= \text{POISON}(i, \text{NMAX}_i + 1) - \text{POISON}(i, \text{KIT}(i) + 1)$$

After comparing each ratio r_i and obtaining the maximum one, we increase its kit quantity by one. Termination occurs if the kit's price is greater than budget, or the procedure is repeated to update the kit's value and the ratio until the budget is exceeded.

Ratio Calculation Segment:

```

C          START WITH THE INITIAL KIT AND CALCULATE THE RATIO
C          PER DOLLAR INVESTED FOR EACH ITEM
C
ISN 0009      PRICE=PRC IN
ISN 0010      DO 10 I=1,ITOT
ISN 0011      KIT(I)=KITIN(I)
ISN 0012      NMAX1=KIT(I)+(NOPS-1)*NAP(I)+2
ISN 0013      IF(NMAX1.GE.60) NMAX1=60
ISN 0015      IXS=KIT(I)+1
ISN 0016      SUM=DBLE(PAR)*(POISON(I,NMAX1)-POISON(I,IXS))
ISN 0017      DO 20 J=1,MUP
ISN 0018      IXJ=KIT(I)+(J-1)*NAP(I)+1
ISN 0019      IF(IXJ.GE.60) GO TO 21
ISN 0021      NN=POISON(I,IXJ+1)/POISON(I,IXJ)
ISN 0022      GO TO 20
ISN 0023      21 NN=1.0+00
ISN 0024      20 SUM=SUM+DBLE(J)*DLG(NN)
ISN 0025      10 RATIO(I)=SUM/DBLE(COST(I))

```

Line 9 and 11 start with the initial kit and its price. Line 15 increases KIT quantity by 1, once for every item. Line 16 gives the $w \cdot \Delta E(\text{SDO})$, which is

the weighted improvement of $E(SDO)$. Line 19 provides a safeguard against running out of the POISON array limits. Line 21 and 24 calculate $\Delta E(NORS)$, improvements of $E(NORS)$, and add it to $w \cdot \Delta E(SDO)$. Line 25 computes the performance improvement measurement, $RATIO$.

The Remaining Segments:

```

C
C      FIND THE MAXIMUM RATIO
C
ISN 0026      100 DO 25 I=1,ITOT
ISN 0027          IF( I.EQ.1) GO TO 3J
ISN 0029          IF( RATIO(I).LT.RMAX) GO TO 25
ISN 0031      30 IMAX=I
ISN 0032          RMAX=RATIO(I)
ISN 0033      25 CONTINUE
ISN 0034          XPRICE=PRICE+COST(IMAX)

C
C      TERMINATE IF THE TARGETS ARE ACHIEVED
C
ISN 0035          IF( XPRICE.GT.BUDGET) GO TO 5C

C
C      IF NOT, UPDATE THE KIT VALUE AND THE RATIO
C
ISN 0037          KIT(IMAX)=KIT(IMAX)+1
ISN 0038          PRICE=XPRICE
ISN 0039          NMAX1=KIT(IMAX)+(NJPS-1)*NAP(IMAX)+2
ISN 0040          IF( NMAX1.GT.60) NMAX1=60
ISN 0042          IXS=KIT(IMAX)+1
ISN 0043          SUM=DOUBLE(PAR)*(POISON(IMAX,NMAX1)-POISON(IMAX,IXS))
ISN 0044          DO 40 J=1,MUP
ISN 0045              IXJ=KIT(IMAX)+(J-1)*NAP(IMAX)+1
ISN 0046              IF( IXJ.GE.60) GO TO 45
ISN 0048              WW=POISON(IMAX,IXJ+1)/POISON(IMAX,IXJ)
ISN 0049          40 SUM=SUM+BOI(J)*DLOG(WW)
ISN 0050          45 RATIO(IMAX)=SUM/DOUBLE(COST(IMAX))
ISN 0051              GO TO 100
ISN 0052          50 RATIO(IMAX)=0.D+00

C
ISN 0053          IF( RMAX.GT.0.D+00) GO TO 1C0
ISN 0055          RETURN
ISN 0056          END

```

Lines 26 to 33 find the maximum $RATIO$ value. Line 34 then calculates the corresponding price. At line 35, the budget is compared with kit price. If the kit price is exceeded, continue to check the feasibility of

other possible kits. If not exceeded, the kit value and the RATIO are updated. Lines 37 to 50 are similar to previous segments, lines 9 to 25, except this time calculation of RATIO is for the most promising item only.

With the updated RATIO value, this item enters competition with all other items again. If the kit with $q_{imax} + 1$ is infeasible, then line 52 sets its ratio to zero in order to eliminate $q_{imax} + 1$ from further investigation. Line 53 then checks if any other possibilities for consideration are left out. If all possibilities are exhausted, then $imax = 0$; hence the program exits from MARGAN.

Notes on Subroutine CALC

This subroutine calculates the expected NORS and expected STOCK DUE OUT values.

$$E(NORS) = \sum_{Y=1}^{UE} YP(NORS = Y) = \sum_{Y=1}^{UE} Y [P(NORS \leq Y) - P(NORS \leq Y-1)]$$

$$= UE \cdot P(NORS \leq UE) - \sum_{j=0}^{UE-1} P(NORS \leq j)$$

$$= UE \cdot \prod_{i=1}^{ITOT} F_i(q_i + UE \cdot a_i) - \sum_{j=0}^{UE-1} \prod_{i=1}^{ITOT} F_i(q_i + j a_i)$$

$$= UE \cdot S(UE + 1) - \sum_{j=1}^{UE} S(j)$$

$$\text{where } S(N) = P(NORS \leq N - 1)$$

$$= \prod_{i=1}^{ITOT} F_i(q_i + (N-1) \cdot a_i)$$

$$\text{or } = \prod_{i=1}^{ITOT} \text{POISON}(i, q_i + (N-1) \cdot a_i + 1)$$

$$\text{i.e. } E(\text{NORS}) = \text{NOPS} * S(\text{NOPS} + 1) - \sum_{j=1}^{\text{NOPS}} S(j)$$

where NOPS=UE=number of planes in the squadron.

$$E(\text{SDO}) = \sum_{\ell=1}^{\text{ITOT}} \sum_{x=q_{\ell}}^{\text{NMAX}_{\ell}-1} [1-F_{\ell}(x)] = \sum_{\ell=1}^{\text{ITOT}} \sum_{j=0}^{\text{JUP}_{\ell}} [1-F_{\ell}(q_{\ell} + j)]$$

$$\begin{aligned} \text{where } \text{JUP}_{\ell} &= (\text{NMAX}_{\ell} - 1) - q_{\ell} \\ &= [(q_{\ell} + (\text{NOPS}-1)*a_{\ell} + 1) - 1] - q_{\ell} \\ &= (\text{NOPS} - 1)*a_{\ell} + 1 \end{aligned}$$

$$E(\text{SDO}) = \sum_{\ell=1}^{\text{ITOT}} \sum_{j=0}^{\text{JUP}_{\ell}} [1-\text{POISON}(\ell, q_{\ell} + j)]$$

E(NORS) Computation Segment:

```

ISN 0009          EPS=5.0-10
                  C
                  C      E(NORS/X) COMPUTATION
                  C
ISN 0010          SUM=0.0+00
ISN 0011          S(NOPS+1)=1.00
ISN 0012          DO 10 K=1,NOPS
ISN 0013          S(K)=1.0+00
ISN 0014          DO 20 I=1,ITOT
ISN 0015          IXJ=KIT(I)+(K-1)*NAP(I)+1
ISN 0016          IF(IXJ.GT.60) GO TO 20
ISN 0017          S(K)=S(K)*POISON(I,IXJ)
ISN 0018          IF(S(K).LT.EPS) GO TO 15
ISN 0019          20 CONTINUE
ISN 0020          SUM=SUM+S(K)
ISN 0021          GO TO 10
ISN 0022          15 S(K)=0.0+00
ISN 0023          10 CONTINUE
ISN 0024          VNORS=DFLOAT(NOPS)*S(NOPS+1)-SUM
ISN 0025          ENORS=VNORS

```

Line 18 gives the probability of $\text{NORS} \leq N-1$, $S(N)$. Line 19 provides a check for very small $S(N)$ values to save further computational effort. Line 24 just sets any small value ($\leq 5 \times 10^{-10}$) to zero for practical purposes. At line 26, $E(\text{NORS})$ is calculated by using the formula shown previously.

E(SDO) Computation Segment:

```
      C      E(SDO/X) COMPUTATION
      C
ISN 0028      SDOP=0.0+00
ISN 0029      DO 50 I=1,ITOT
ISN 0030      JUP=(NOPS-1)*NAP(I)+1
ISN 0031      DO 60 J=1,JUP
ISN 0032      IXJ=KIT(I)+J
ISN 0033      IF(IXJ.GT.60) GO TO 50
ISN 0035      60 SDOP=SDOP+1.0+00-PJ ISDN(I,IXJ)
ISN 0036      50 CONTINUE
ISN 0037      SDO=SDOP
ISN 0038      RETURN
ISN 0039      END
```

Line 35 gives the calculation of E(SDO), which is also formulated in the above section. At lines 28 and 35, SDOP is used instead of SDO to avoid alternating the current value stored in SDO. Line 37 then updates the SDO value.

Notes on Subroutine OUTPUT

This subroutine prints out the final results of the optimization. It includes: the optimal kit, its price and performance indices E(NORS) and E(SDO); parameters set b_i 's; weighting factor of RATIO computation; kit 57-18 and its relevant values. Number of iterations is also printed.

APPENDIX B.2

FORTRAN LISTING OF
GREEDY ALGORITHM
COMPUTER PROGRAM
AND SOLUTION OUTPUT

```

C *****
C *
C *
C *      GREEDY  ALGORITHM
C *
C *
C *****
C
ISN 0002      DIMENSION KIT(500)
C
ISN 0003      CALL INPUT1(KIT,PRICE)
ISN 0004      CALL TABLE
ISN 0005      CALL INKIT(KIT,PRICE)
ISN 0006      CALL OPTIM(KIT,PRICE,ENORS,SCO)
ISN 0007      CALL OUTPUT1 (KIT,PRICE,ENORS,SCO)
C
ISN 0008      CALL INPUT2(KIT,PRICE)
ISN 0009      CALL TABLE
ISN 0010      CALL INKIT(KIT,PRICE)
ISN 0011      CALL OPTIM(KIT,PRICE,ENORS,SCO)
ISN 0012      CALL OUTPUT2 (KIT,PRICE,ENORS,SCO)
ISN 0013      STOP
ISN 0014      END

ISN 0002      SUBROUTINE INPUT1(KIT,PRICE)
ISN 0003      DIMENSION KIT(500),DFT(30)
ISN 0004      COMMON PUISON(500,60),BO(50)
ISN 0005      COMMON /BLOCK1/ITOT,NITER,NOPS,MUP,BUDGET,SKF,PAR
ISN 0006      COMMON /BLOCK2/ NAP(500),COST(500),XNU(500)
ISN 0007      COMMON /BLOCK4/ KT5718(500),LGOAL,SGOAL,BUDKT
ISN 0008      COMMON /PHASE/ IPHASE
ISN 0009      COMMON /INPUT/NDAYS,DFF,NSUD
ISN 0010      DOUBLE PRECISION PJISON,BO
ISN 0011      DATA XRU/'LRU'/,RRR/'RRR'/,DFT/7*1.15,.53,22*.52/
C
ISN 0012      IPHASE=1
ISN 0013      NITER=0
ISN 0014      PRICE=0.0
ISN 0015      ITEM=0
C
ISN 0016      READ(5,109) PAR
C
ISN 0017      READ(5,101) SFPD,NJAYS, DFF,NOPS,NSUD,SKF
C
ISN 0018      MUP=NOPS+1
ISN 0019      10 ITEM=ITEM+1
C
ISN 0020      20 READ(1,102,END=50) CAP,NRC,NAP(ITEM),TOR,COST(ITEM),TYP
C
C      ONLY LRU ITEMS WILL BE PROCESSED
C
ISN 0021      IF(TYP.EQ.XRU) GO TO 30
ISN 0023      GO TO 20
ISN 0024      30 XNU(ITEM)=0.0
ISN 0025      LENT=NSUD
C
ISN 0026      IF(CAP.EQ.RRR) LENT=NSUD+NRC
C
C      DEMAND AND KIT VALUES
C
ISN 0028      DO 40 IDAY=1,LENT
ISN 0029      40 XNU(ITEM)=XNU(ITEM)+FLOAT(NAP(ITEM))*TOR*DFT(IDAY)
ISN 0030      KIT(ITEM)=0
ISN 0031      GO TO 10
ISN 0032      50 ITOT=ITEM-1

```

```

C
C      COMPUTATION OF THE REQUIREMENT KIT 5718 & TARGET VALUES
C
ISN 0033      BUDGET=0.0
ISN 0034      DO 60 I=1,ITOT
ISN 0035      KT5718(I)=IFIX(XMUM I)*SKF+.5)
ISN 0036      IF(KT5718(I).LT.1) KT5718(I)=1
ISN 0038      BUDGET=BUDGET+COST(I)*FLOAT(KT5718(I))
ISN 0039      60 CONTINUE
C
ISN 0040      BUDKT=BUDGET
ISN 0041      BUDGET=.9*BUDGET
C
ISN 0042      101 FORMAT(F5.2,1X,I2,1X,F5.2,1X,I2,T25,I2,T28,F6.4)
ISN 0043      102 FORMAT(T19,A3,I2,1X,I2,T32,F5.4,T46,F8.2,A3)
ISN 0044      105 FORMAT(I4)
ISN 0045      109 FORMAT(F10.0)
ISN 0046      110 FORMAT(1008.0)
ISN 0047      RETURN
ISN 0048      END
C
ISN 0002      SUBROUTINE OUTPUT(IKIT,PRICE,ENGRS,SDO)
ISN 0003      DIMENSION KIT(500),SN(4)
ISN 0004      COMMON POISON(500,60),BO(5)
ISN 0005      COMMON /BLOCK1/ITOT,NITER,NOPS,MUF,BUDGET,SKF,FAR
ISN 0006      COMMON /BLOCK2/ MAP(500),COST(500),XMUM(500)
ISN 0007      COMMON /BLOCK4/ KT5718(500),VGOAL,SGOAL,BUDKT
ISN 0008      COMMON /PHASE/ IPHASE
C
ISN 0009      DOUBLE PRECISION PJISJN,BO
C
ISN 0010      ITEM=0
C
ISN 0011      WRITE(6,200)
ISN 0012      WRITE(6,210) ITOT
C
C      PRINT OUT THE TARGET VALUES
C
ISN 0013      WRITE(6,205) (KT5718(I),I=1,ITOT)
ISN 0014      WRITE(6,220) BUDKT,VGOAL,SGOAL
ISN 0015      WRITE(6,230) BUDGET
C
ISN 0016      WRITE(6,100) PRICE,ENGRS,SDO
ISN 0017      WRITE(6,110) NITER,(BO(J),J=1,MUF)
ISN 0018      WRITE(6,109) PAR
ISN 0019      WRITE(6,105)
ISN 0020      WRITE(6,120) (KIT(I),I=1,ITOT)
C
ISN 0021      RE=IND 1
C
ISN 0022      100 FORMAT(5X,'PRICE(X)=',F15.2,5X,'ENGRS/X)=',F10.5,5X,'E(SDO/X)=',
      &F10.5,/)
ISN 0023      105 FORMAT(//,5X,'INITIAL KIT',/)
ISN 0024      109 FORMAT(//,5X,'WEIGHT ',F10.3,/)
ISN 0025      110 FORMAT(//,15X,'NUMBER OF ITERATIONS',I4,/,5X,'X(J)',I4(2X,F10.8))
ISN 0026      120 FORMAT(5X,25I4)
ISN 0027      200 FORMAT(//,15X,'GREEDY ALGORITHM',/,15X,'PHASE 1',/)
ISN 0028      205 FORMAT(5X,25I4)
ISN 0029      210 FORMAT(//,15X,'INITIAL KIT COMPUTATION',/,5X,I4,' ITEMS IN COMPUTATION
      &ITATION',/,20X,'KIT 5718',/)
ISN 0030      220 FORMAT(//,15X,'TARGETS',5X,'PERFORMANCE OF 5718 KIT',/,5X,'COST
      &1 OF 5718 KIT=',F15.2,5X,'NORSGOAL=',F10.5,5X,'SDOGOAL=',F10.5,/)
ISN 0031      230 FORMAT(5X,'BUDGET OF INITIAL KIT IS ',F15.2,/)
C
ISN 0032      90 IPHASE=2
ISN 0033      RETURN
ISN 0034      END

```

```

ISN 0002      SUBROUTINE INPUT2(KIT,PRICE)
C
C      READ IN THE PERTINENT DATA FROM THE TAPES AND CARDS
C
ISN 0003      DIMENSION KIT(500),SN(4),TM(4),DPGM(30),DMDS(30),RMT(30),RPRS(30)
ISN 0004      COMMON PUISON(500,60),BO(50)
ISN 0005      COMMON /BLOCK 1/ ITOT,NITER,NOPS,MUF,BUDGET,SKF,FAK
ISN 0006      COMMON /BLOCK 2/ NAP(500),COST(500),XNU(500)
ISN 0007      COMMON /BLOCK 4/ KT57,KT(500),VGOAL,SGOAL,BUDKT
ISN 0008      COMMON /PHASE/ IPHASE
ISN 0009      COMMON /INPUT/NDAYS,OFF,NSUD
ISN 0010      DOUBLE PRECISION PUISON,BO
ISN 0011      DATA XRW,'LRU',SRU,'SRU',RRR,'RRR' /
ISN 0012      DATA DPGM/7*1,15,53,22*52/

C
ISN 0013      NITER=0
ISN 0014      PRICE=0.0
ISN 0015      ITEM=0

C
ISN 0016      READ(5,109) PAR

C
ISN 0017      10 ITEM=ITEM+1

C
ISN 0018      20 READ(1,111,END=200) (SN(I),I=1,4),CAP, NRC,NAP(ITEM),TOR,DDR,COST
ISN 0019      * (ITEM),TYP
      BRR=TOR-DDR

C
C      ONLY LRU & SRU ITEMS WILL BE PROCESSED
C
ISN 0020      IF(TYP.EQ.SRU) GO TO 30
ISN 0022      IF(TYP.NE.XRU) GO TO 20
ISN 0024      INU=KIT(ITEM)
ISN 0025      GO TO 17
ISN 0026      30 INU=0
ISN 0027      17 TOTRU=0.0
ISN 0028      DO 110 IDAY=1,NDAYS
ISN 0029      110 DMDS(IDAY)=FLOAT(NAP(ITEM))*TOR+DPGM(IDAY)

C
C      RRR PROCESSING
C
ISN 0030      IF(CAP.NE.RRR) GO TO 15
ISN 0032      IF(TYP.EQ.XRU.AND.(INU.LT.1)) INU=1
ISN 0034      KTOT=NSUD+NRC
ISN 0035      IF(KTOT.GT.30) KTOT=30
ISN 0037      DO 120 IDAY=1,KTOT
ISN 0038      120 RPRS(IDAY)=0.0
ISN 0039      RPRHOL=0.0
ISN 0040      IF(KTOT.GE.30) GO TO 132
ISN 0042      KTOT=NSUD+1
ISN 0043      DO 125 I=1,KTOT
ISN 0044      125 RPRHOL=RPRHOL+FLOAT(NAP(ITEM))*BRR+DPGM(I)
ISN 0045      RPRS(NSUD+NRC+1)=RPRHOL
ISN 0046      JTOT=NSUD+NRC+2
ISN 0047      DO 130 IDAY=JTOT,NDAYS
ISN 0048      130 RPRS(IDAY)=FLOAT(NAP(ITEM))*BRR+DPGM(IDAY-NRC)
ISN 0049      132 RMT(1)=DMDS(1)-RPRS(1)
ISN 0050      DO 135 IDAY=2,NDAYS
ISN 0051      135 RMT(IDAY)=RMT(IDAY-1)+DMDS(IDAY)-RPRS(IDAY)
ISN 0052      TOTRU=RMT(1)
ISN 0053      DO 140 IDAY=2,NDAYS
ISN 0054      IF(RMT(IDAY).GT.TOTRU) TOTRU=RMT(IDAY)
ISN 0056      140 CONTINUE
ISN 0057      GO TO 70

```

```

C
C   RR PROCESSING
C
ISN 0058   15 00 160 IDAY=1,NDAYS
ISN 0059   160 TOTRQ=TOTRQ+DMDS(IDAY)
C
C   DEMAND AND KIT VALUES
C
ISN 0060   70 XMU(ITEM)=TOTRQ
ISN 0061     KIT(ITEM)=INU
ISN 0062     PRICE=PRICE+COST(ITEM)+FLOAT(INQ)
ISN 0063     GO TO 10
C
ISN 0064   200 ITOT=ITEM-1
C
C   COMPUTATION OF THE REQUIREMENT KIT 5718 & TARGET VALUES
C
ISN 0065     BUDGET=0.0
ISN 0066     00 50 I=1,ITOT
ISN 0067       KT5718(I)=IFIX(XMU(I)+SKF+.5)
ISN 0068       IF(KT5718(I).LT.1) KT5718(I)=1
ISN 0069       BUDGET=BUDGET+COST(I)+FLOAT(KT5718(I))
ISN 0071     50 CONTINUE
ISN 0072     BUDKT=BUDGET
C
ISN 0073   102 FORMAT(5X,4A4,5X,I4)
ISN 0074   105 FORMAT(I4)
ISN 0075   108 FORMAT(1J08.0)
ISN 0076   109 FORMAT(F10.0)
ISN 0077   111 FORMAT(4A4,T19,A3,1X,I1,1X,I2,T32,F5.4,1X,F5.4,T46,F8.2,A3)
ISN 0078     RETURN
ISN 0079     END
C
ISN 0082     SUBROUTINE OUTPUT2(KIT,PRICE,ENORS,SDQ)
C
C   THIS SUBROUTINE PRINT THE FINAL RESULTS OF THE OPTIMIZATION
C
ISN 0083     DIMENSION KIT(500)
ISN 0084     COMMON POISON(500,50),80(5)
ISN 0085     COMMON /BLOCK1/ITOT,NITER,AOPS,MUF,BUDGET,SKF,PAR
ISN 0086     COMMON /BLOCK2/ NAF(500),COST(500),XMU(500)
ISN 0087     COMMON /BLOCK4/ KT5718(500),VGOAL,SGOAL,BUDKT
C
ISN 0088     DOUBLE PRECISION POISON,80
C
ISN 0089     WRITE(6,99)
C
C   PRINT OUT THE TARGET VALUES
C
ISN 0090     WRITE(6,950) VGOAL,SGOAL,BUDKT
ISN 0091     WRITE(6,990) (KT5718(I),I=1,ITOT)
C
ISN 0092     WRITE(6,50) ITOT,NITER
ISN 0093     WRITE(6,100) ENORS,SDQ,PRICE
ISN 0094     WRITE(6,200) (KIT(I),I=1,ITOT)
ISN 0095     WRITE(6,300) (BUDJ),J=1,MUF)
ISN 0096     WRITE(6,555) PAR
C
ISN 0097     50 FORMAT(//,5X,'TOTAL NUMBER OF ITEMS UNDER CONSIDERATIONS',I4,/,/,
+5X,'NUMBER OF ITERATIONS',I4,/,/)
ISN 0098     99 FORMAT(//,15X,'GREEDY ALGORITHM',//,15X,'PHASE 2',/)
ISN 0099     100 FORMAT(5X,'ENORS/X'=',F10.5,5X,'E(SDQ/X)'=',F10.5,5X,'PRICE(X)'=',
+5X,/,//,15X,'OPTIMAL KIT',/)
ISN 0100     200 FORMAT(5X,25I4)
ISN 0101     300 FORMAT(//,5X,'8',/,/(10(2X,F10.8)))
ISN 0102     555 FORMAT(//,5X,'WEIGHT 1',F10.3,/)
ISN 0103     950 FORMAT(//,5X,'TARGETS',//,5X,'NORSGOAL 1',F10.5,5X,'SDGOAL 1',F10.
+5X,5X,'PRICE=5',F15.2,/)
ISN 0104     990 FORMAT(5X,25I4)
ISN 0105     RETURN
ISN 0106     END

```

```

ISN 0002      SUBROUTINE TABLE
C
C      COMPUTATION OF THE CUMULATIVE SUMS OF POISSON DISTRIBUTION
C
ISN 0003      COMMON POISON(500,60),MU(5)
ISN 0004      COMMON /BLOCK1/ ITOT, NITER, NOPS, MUF, BUDGET, SKF, FAR
ISN 0005      COMMON /BLOCK2/ NAF(500), COST(500), XMU(500)
C
ISN 0006      DOUBLE PRECISION POISON, TERM, SUM, TERL
C
ISN 0007      TERL=9.0-12
ISN 0008      DO 10 I=1, ITOT
ISN 0009      SUM=0.0
ISN 0010      TERM=EXP(-SUM)
ISN 0011      POISON(I,1)=TERM
ISN 0012      DO 20 J=2,60
ISN 0013      IF( TERM.LT. TERL.AND. POISON(I,J-1).GT.9.0-01) GO TO 30
ISN 0014      TERM=TERM+SUM/DFLOAT(J-1)
ISN 0015      POISON(I,J)=POISON(I,J-1)+TERM
ISN 0016      20 CONTINUE
ISN 0017      GO TO 10
ISN 0018      30 DO 40 JX=J,60
ISN 0019      40 POISON(I,JX)=1.0+0.0
ISN 0020      10 CONTINUE
ISN 0021      RETURN
ISN 0022      END
ISN 0023

```

```

ISN 0002      SUBROUTINE INKIT(KIT,PRICE)
C
C      INCREASE KIT VALUES TO ENSURE THAT THE PROBABILITY OF ANY ITEM
C      WHICH WILL HAVE HALF OF THE FLEET DOWN IS LESS THAN 50%
C
ISN 0003      DIMENSION KIT(500)
ISN 0004      COMMON POISON(500,60),MU(5)
ISN 0005      COMMON /BLOCK1/ ITOT, NITER, NOPS, MUF, BUDGET, SKF, FAR
ISN 0006      COMMON /BLOCK2/ NAF(500), COST(500), XMU(500)
ISN 0007      COMMON /BLOCK3/ KITIN(500), PRGIN
C
ISN 0008      DOUBLE PRECISION POISON, MU
C
ISN 0009      NEPS=(NOPS+1)/2
ISN 0010      DO 10 ITEM=1, ITOT
ISN 0011      20 NUM=NEPS+NAF(ITEM)+KIT(ITEM)
ISN 0012      IF(NUM.GT.60) GO TO 10
ISN 0013      IF(POISON(ITEM,NUM).GT.0.5) GO TO 10
ISN 0014      KIT(ITEM)=KIT(ITEM)+1
ISN 0015      PRICE=PRICE+COST(ITEM)
ISN 0016      GO TO 20
ISN 0017      10 KITIN(ITEM)=KIT(ITEM)
ISN 0018      PRGIN=PRICE
C
ISN 0019      RETURN
ISN 0020      END
ISN 0021
ISN 0022

```

```

ISN 0002      SUBROUTINE OPTIM KIT,PRICE,ENORS,SCD)
C
C      MAIN OPTIMIZATION ROUTINE
C
ISN 0003      DIMENSION KIT(500),S(50)
ISN 0004      COMMON POISON(500,60),BO(50)
ISN 0005      COMMON /BLOCK 1/ ITOT,NITER,AOPS,MUP,BUDGET,SKF,PAR
ISN 0006      COMMON /BLOCK 2/ NAP(500),CCST(500),XMU(500)
ISN 0007      COMMON /BLOCK 4/ KT5718(500),VGOAL,SGOAL,BUDKT
ISN 0008      COMMON /PHASE/ IPHASE
C
ISN 0009      DOUBLE PRECISION POISON,BO,SLMSQ,EPSU,PRO,EPS,S
ISN 0010      COMMON /S1/S
C
ISN 0011      DATA EPS,EPSU/1.D-07,5.D-11/
C
C      COMPUTE THE PERFORMANCE OF KIT 57-18 AND INITIATE PARAMETERS
C      B
C
ISN 0012      CALL CALC(KT5718,VGOAL,SGOAL)
ISN 0013      DO 1 J=1,MUP
ISN 0014      1 BO(J)=S(J)
ISN 0015      WRITE(6,300) (BO(J),J=1,MUP)
ISN 0016      300 FORMAT('10/19, INITIAL VALUES OF PARAMETERS B'/(10(2X,F10.8)))
C
C      START OPTIMIZATION
C
C      STEP 1
C
ISN 0017      100 CALL MARGAN(KIT,PRICE)
C
ISN 0018      NITER=NITER+1
C
C      STEP 2
C      UPDATE THE PARAMETERS B
C
ISN 0019      SUMSQ=0.D+00
ISN 0020      DO 10 J=1,MUP
ISN 0021      PRO=1.D+00
ISN 0022      DO 20 I=1,ITOT
ISN 0023      KITPJ=KIT(I)+(J-1)*NAP(I)+J
ISN 0024      IF(KITPJ.GT.60) GO TO 20
ISN 0025      PRO=PRO+POISON I,KITPJ)
ISN 0026      IF( PRO.LT.EPSU) GO TO 15
ISN 0027      20 CONTINUE
ISN 0028      15 SUMSQ=SUMSQ+((BO(J)-PRO)**2)
ISN 0029      BO(J)=PRO
ISN 0030      10 CONTINUE
C
C      STEP 3
C      COMPARE THE NEW VALUES OF B WITH PREVIOUS VALUES
C      TERMINATE IF THEY ARE SUFFICIENTLY CLOSE
C
ISN 0031      IF( SUMSQ.LT.EPS) GO TO 30
ISN 0032      GO TO 100
C
ISN 0033      30 CALL CALC(KIT,ENORS,SCD)
ISN 0034      RETURN
ISN 0035      END

```

```

ISN 0002      SUBROUTINE MARGAN(KIT,PRICE)
C
C      THIS SUBROUTINE PERFORMS THE MARGINAL ANALYSIS
C
ISN 0003      DIMENSION RATIO(500),KIT(500)
ISN 0004      COMMON POISON(500,60),BO(500)
ISN 0005      COMMON /BLOCK1/ ITOT,NIETER,NOPS,MUP,BUDGET,SKF,PAR
ISN 0006      COMMON /BLOCK2/ NAP(500),CCS(500),XMU(500)
ISN 0007      COMMON /BLOCK3/ KITIN(500),PRCIN
C
ISN 0008      DOUBLE PRECISION PJISON,RATIO,BO,SUM,WM,RMAX
C
C      START WITH THE INITIAL KIT AND CALCULATE THE RATIO OF RETURN
C      PER DOLLAR INVESTED FOR EACH ITEM
C
ISN 0009      PRICE=PRCIN
ISN 0010      DO 10 I=1,ITOT
ISN 0011      KIT(I)=KITIN(I)
ISN 0012      NMAX1=KIT(I)+(NOPS-1)*NAP(I)+2
ISN 0013      IF(NMAX1.GE.60) NMAX1=60
ISN 0014      IXS=KIT(I)+1
ISN 0015      SUM=DBLE(PAR)*(POISON(I,NMAX1)-POISON(I,IXS))
ISN 0016      DO 20 J=1,MUP
ISN 0017      IXJ=KIT(I)+(J-1)*NAP(I)+1
ISN 0018      IF(IXJ.GE.60) GO TO 21
ISN 0019      WM=POISON(I,IXJ+1)/POISON(I,IXJ)
ISN 0020      GO TO 20
ISN 0021      21 WM=1.0+00
ISN 0022      20 SUM=SUM+BO(J)*DLOG(WM)
ISN 0023      10 RATIO(I)=SUM/DBLE(COST(I))
ISN 0024
ISN 0025      C
C      FIND THE MAXIMUM RATIO
C
ISN 0026      100 DO 25 I=1,ITOT
ISN 0027      IF(I.EQ.1) GO TO 30
ISN 0028      IF(RATIO(I).LT,RMAX) GO TO 25
ISN 0029      30 IMAX=I
ISN 0030      RMAX=RATIO(I)
ISN 0031      25 CONTINUE
ISN 0032      XPRICE=PRICE+COST(IMAX)
ISN 0033
ISN 0034      C
C      TERMINATE IF THE TARGETS ARE ACHIEVED
C
ISN 0035      IF(XPRICE.GT,BUDGET) GO TO 50
C
C      IF NOT, UPDATE THE KIT VALUE AND THE RATIO
C
ISN 0037      KIT(IMAX)=KIT(IMAX)+1
ISN 0038      PRICE=XPRICE
ISN 0039      NMAX1=KIT(IMAX)+(NOPS-1)*NAP(IMAX)+2
ISN 0040      IF(NMAX1.GT.60) NMAX1=60
ISN 0041      IXS=KIT(IMAX)+1
ISN 0042      SUM=DBLE(PAR)*(POISON(IMAX,NMAX1)-POISON(IMAX,IXS))
ISN 0043      DO 40 J=1,MUP
ISN 0044      IXJ=KIT(IMAX)+(J-1)*NAP(IMAX)+1
ISN 0045      IF(IXJ.GE.60) GO TO 45
ISN 0046      WM=POISON(IMAX,IXJ+1)/POISON(IMAX,IXJ)
ISN 0047      40 SUM=SUM+BO(J)*DLOG(WM)
ISN 0048      45 RATIO(IMAX)=SUM/DBLE(COST(IMAX))
ISN 0049      GO TO 100
ISN 0050      50 RATIO(IMAX)=0.0+00
ISN 0051
ISN 0052      C
ISN 0053      IF(RMAX.GT.0.0+00) GO TO 100
ISN 0054      RETURN
ISN 0055
ISN 0056      END

```

```

ISN 0002      SUBROUTINE CALC(KIT,ENORS,SDC)
C
C      THIS SUBROUTINE COMPUTES THE E(NORS/K) AND E(SDC/K)
C
ISN 0003      DIMENSION S(50),KIT(500)
ISN 0004      COMMON POISON(500,0),BD(S(
ISN 0005      COMMON /BLOCK1/ITOT,NITEN,NOPS,MUF,BUDGET,SKF,PAR
ISN 0006      COMMON /BLOCK2/ NAP(50),CCST(500),XNU(500)
ISN 0007      COMMON /S1/S
ISN 0008      DOUBLE PRECISION POISON,S,SUP,VNORS,SDOP,EP S

C
C
ISN 0009      EPS=5.0-10

C
C      E(NORS/K) COMPUTATION
C
ISN 0010      SUM=0.0+00
ISN 0011      S(NOPS+1)=1.00
ISN 0012      DO 10 K=1,NOPS
ISN 0013      S(K)=1.0+00
ISN 0014      DO 20 I=1,ITOT
ISN 0015      IXJ=KIT(I)+(K-1)*NAP(I)+1
ISN 0016      IF(IXJ.GT.60) GO TO 20
ISN 0017      S(K)=S(K)+POISON I, IXJ)
ISN 0018      IF(S(K).LT.EPS) GO TO 15
ISN 0019      20 CONTINUE
ISN 0020      SUM=SUM+S(K)
ISN 0021      GO TO 10
ISN 0022      15 S(K)=0.0+00
ISN 0023      10 CONTINUE
ISN 0024      VNORS=DFLOAT(NOPS)+S(NOPS+1)-SUM
ISN 0025      ENORS=VNORS
ISN 0026
ISN 0027
C
C      E(SDC/K) COMPUTATION
C
ISN 0028      SDOP=0.0+00
ISN 0029      DO 50 I=1,ITOT
ISN 0030      JUP=(NOPS-1)*NAP(I)+1
ISN 0031      DO 60 J=1,JUP
ISN 0032      IXJ=KIT(I)+J
ISN 0033      IF(IXJ.GT.60) GO TO 50
ISN 0034      60 SDOP=SDOP+1.0+00-POISON(I,IXJ)
ISN 0035      50 CONTINUE
ISN 0036      SDC=SDOP
ISN 0037      RETURN
ISN 0038      ENJ
ISN 0039

```

[illegible]

PHASE 1

180 ITEMS IN COMPUTATION

[illegible]

COST	OF	5710	KIT=3	4039896.00	NORSCAL=	4.92074	SDGOAL=	24.49922
------	----	------	-------	------------	----------	---------	---------	----------

PRICE(1)=3 3635861.00 E(MORS/X)= 4.84982 E(SD0/X)= 29.47310

81,1)	0.0000000	0.00172540	0.06444003	0.76113723	0.49893585	0.69276720	0.82435903	0.90472555	0.95030868	0.97499286
0.9877242	0.981652	0.9972261	0.99871559	0.99941528	0.99973106	0.9998454	0.99994995	0.99997868	0.99999168	
0.99999634	0.99999852	0.99999942	0.99999977	0.99999991						

WEIGHT : 0.023

[illegible]

GREEDY ALGORITHM
PHASE 2

NORSGOAL :	6.1354U	SDGOAL :	163.92674	PRICE=	7562983.00
------------	---------	----------	-----------	--------	------------

[illegible]

NUMBER OF ITERATIONS 4

REINFORCING	7.15053	216.61477	PRICE/MT=	7562944.00
-------------	---------	-----------	-----------	------------

OPTIMAL KTY

[illegible][illegible]

WEIGHT : 0.020

APPENDIX C

- C.1 Notes on Conversion of the D029 Computer Program to an IBM Compiler
- C.2 FORTRAN Listing of the IBM Version of the D029 Algorithm with a
Solution Output

APPENDIX C.1

NOTES ON CONVERSION OF THE DO29 COMPUTER PROGRAM
TO AN IBM COMPILER

C.1.1

APPENDIX C.1

NOTES ON THE CONVERSION OF THE CURRENT D029 COMPUTER PROGRAM TO AN IBM COMPILER

IBM and CDC FORTRAN compilers are not exactly the same. Any computer code compatible with a CDC FORTRAN compiler is not compatible with an IBM FORTRAN compiler and vice versa. Because of this, the D029-CDC code has been converted into an IBM compatible code in order that appropriate comparisons could be made between the current D029 program and the Greedy Algorithm program.

CDC FORTRAN compilers accept variable names of seven characters, but an IBM FORTRAN compiler can accept variable names of only six characters. Therefore, variable names with seven characters were truncated to six characters. In addition, the following statements are not accepted by an IBM FORTRAN compiler.

```
IF(IMAIN.EQ.3HRRR). . .
```

```
and READ(1,107). . .  
IF(EOF(1).NE.0) GO TO 200
```

The first statement is converted by defining an alphanumeric variable RRR in the data statement as follows.

```
DATA RRR/'RRR'/  
.  
.  
.  
IF(IMAIN.EQ.RRR). . .
```

The second statement checks for the end of file in the data set from a tape. The conversion is accomplished by combining the two lines together as

```
READ(1,107,END=200). . .
```

The word size of a CDC computer is 64 bits, but on an IBM 370, a word size of 32 bits is utilized. This 64 bit word size causes the memory space on a CDC to be expensive and memory saving measures are frequently devised. Consequently, special subroutines for word packing and unpacking are included in the current D029 program.

It is mentioned on page 11 of the Preliminary Evaluation of D029 WRSK Model [8] that the QPA for each item is packed into six bits or 1/10 of a word. The cumulative probability of demand for an item is packed into 20 bits, or 1/3 of a word. The probability values for quantities K , $K+QPA$, $K+2*QPA$, . . . , $K+UE*QPA$ are stored when QPA is greater than one. When a probability value for a quantity between $K+j*QPA$ and $K+(j+1)*QPA$ is needed, this probability is estimated by interpolation. On the IBM 370/168 at Oklahoma State University, sufficient memory is available such that word packing and unpacking is not needed. However, double precision is employed to increase the accuracy in calculating the cumulative probability values for $E(NORS/X)$ and $E(SDO/X)$.

The IBM version of the D029 computer program is summarized as follows. The input section of the current D029 has been adapted as before with some necessary modifications due to machine differences. The optimization routine has not been changed. Subroutine ADD has also been preserved with some modifications. Subroutine TABLE of the IBM version calculates the cumulative Poisson sums and stores the values in the array Poisson (500,60). Poisson sums are retrieved from this array when required. The FORTRAN listing of the IBM version of the current D029 program is provided in Appendix C.2.

APPENDIX C.2

FORTRAN LISTING OF THE IBM VERSION OF THE D029 ALGORITHM
WITH A SOLUTION OUTPUT

C.2.1

```

C *****
C *
C *      0029      ALGORITHM
C *      U S AIR FORCE
C *
C *
C *****
ISN 0002      DIMENSION KIT(500)
ISN 0003      DOUBLE PRECISION E(100),SDO
ISN 0004      COMMON /BLK0/ ISUM,IPASS
ISN 0005      COMMON /PRINT/MINPUT,MXMU,MITER

C
C      DETAILED PRINT OUT OPTION, ALL DEFAULT TO ZERO.
C
C      MINPUT=1 WILL PRINT OUT INPUT DATA
C      MXMU=1 WILL PRINT OUT AVERAGE DEMAND MXMU(I), AND NFP0(I) FOR EACH
C      ITEM
C      MITER=1 WILL PRINT OUT CPU TIME AND ITEMS ADDED FOR EACH ITERATION
C
C ***
ISN 0006      CALL TIME
ISN 0007      1 CALL INPUT1( PRICE )
ISN 0008      CALL TABLE
ISN 0009      CALL INKIT(KIT,PRICE)
ISN 0010      CALL OPTIM(KIT,ENORS,SDO,PRICE)
ISN 0011      CALL OUTPUT(KIT,ENORS,SDO,PRICE)
ISN 0012      IF(IPASS.EQ.2) GO TO 1
ISN 0013      STOP
ISN 0014      END

ISN 0002      BLOCK DATA
C
C      BLOCK DATA SUBPROGRAM TO INITIALIZE VARIABLES IN LABELED COMMON
C
ISN 0003      DIMENSION BINTER(10)
ISN 0004      COMMON /BLK6/BINTER,BLK0/ISUM,IPASS
ISN 0005      COMMON /BLK8/VPT,SDJT,VPG0AL,SDGCL
ISN 0006      COMMON /COUNT/ICJUT,IADD(200)
ISN 0007      COMMON /PRINT/MINPUT,MXMU,MITER
ISN 0008      DOUBLE PRECISION VPGJAL,SCGOL
ISN 0009      DATA BINTER/1.5E6, 3.0E6, 4.5E6,6.0E6, 7.5E6,9.0E6,10.0E6,
*      3*0.0/,IPASS/1/,VPT,SDJT/2*0./,ISUM/0,ICJUT/0,
*      MINPUT,MXMU,MITER/3*0/
ISN 0010      END

ISN 0002      SUBROUTINE CALC(KIT,VP,SDO)
C
C      THIS SUBROUTINE COMPUTES THE E(ENORS/X) AND E(SDO/X)
C
ISN 0003      DIMENSION S(150),KIT(500)
ISN 0004      COMMON PJISDN(300,100)
ISN 0005      COMMON /BLK0/ ISUM,IPASS
ISN 0006      COMMON /BLK1/ ITOT,4UP,BUDGET,BLK2/XMU(500),NAP(500),COST(500),NFP0
*      (1500)/BLK3/NUPS,SKF,IUEFF,IKK
ISN 0007      COMMON /BLK7/S1
ISN 0008      DOUBLE PRECISION PJISJN,S1,VP,SDO ,EPS
ISN 0009      DATA EPS/1.0-12/

```

```

C
ISN 0010      S11 MUP)=1.0+00
ISN 0011      SDD=0.3+00
ISN 0012      DO 10 I=1,ITOT
ISN 0013      JUP=(NOPS-1)+NAP(I)+1
ISN 0014      DO 11 J=1,JUP
ISN 0015      IXJ=KIFA(I)+J
ISN 0016      IF(IXJ.GE.NFPO(I)) GO TO 1C
ISN 0018      11 SDD=SDD+1.0+00-POISON(I,IXJ)
ISN 0019      10 CONTINUE

C
ISN 0020      VP=0.0+00
ISN 0021      IU=IUFFFF
ISN 0022      IKKK=IKK
ISN 0023      DO 25 N2=1,IU
ISN 0024      N=IKKK-N2
ISN 0025      S11 N)=1.0+00
ISN 0026      DO 20 I=1,ITOT
ISN 0027      IXJ=KIFA(I)+(N-1)+NAP(I)+1
ISN 0028      IF(IXJ.GE.NFPO(I)) GO TO 2C
ISN 0030      S11 N)=S11 N)+POISON(I,IXJ)
ISN 0031      IF(S11 N).LE.EPS) GO TO 22
ISN 0033      20 CONTINUE
ISN 0034      21 VP=VP+FLOAT(N)*(S11 N+1)-S11 N)
ISN 0035      IF(S11 V).EQ.0.0+00) GO TO 30
ISN 0037      GO TO 25

C
ISN 0038      22 S11 N)=0.00
ISN 0039      GO TO 21
ISN 0040      25 CONTINUE
ISN 0041      RETURN
ISN 0042      30 DO 35 N2=1,N
ISN 0043      35 S11 N2)=0.0+00
ISN 0044      RETURN
ISN 0045      END

ISN 0002      SUBROUTINE TIME

C
C
C
C
C
C
THIS SUBROUTINE GIVES CPU TIMES

ELAPSE(I) IS AN ASSEMBLY LANGUAGE SUBROUTINE WHICH TIMES THE
EXECUTION OF A PORTION OF THE CALLING PROGRAM

COMMON /BLK0/ ISUM,IPASS
DATA K/0/
IF(K.EQ.0) GO TO 3J
CALL ELAPSE(ICPU)
ISUM=ISUM+ICPU
TCPU=FLOAT(ICPU)/1000.
TSUM=FLOAT(ISUM)/1000.
WRITE(6,10) TSUM,TCPU
10 FORMAT (1X,10(' '),2X,'CPU =',F7.3,' SEC.','5X,' INCREMENT =',
+ F7.3,2X,10(' '))
RETURN
30 CALL ELAPSE(ICPU)
K=1
RETURN
END
ISN 0013
ISN 0014
ISN 0015
ISN 0016
ISN 0017

```

```

ISN 0002      SUBROUTINE INPUT1( PRICE)
C
C      READ IN THE PERTINENT, DATA FROM THE TAPES AND CARDS
C
ISN 0003      DIMENSION DPGM(30),DNDS(30),RPHS(30),RUMT(30),NSN(4)
ISN 0004      COMMON /BLK1/ITOT,MUP,BUDGET/BLK2/XMU(500),NAP(500),COST(500),NFPD
+1500/BLK3/NOPS,SKF,IUEFF,IKK
ISN 0005      COMMON /BLK0/ISUM,IPASS
ISN 0006      COMMON /BLK12/ITOT1,ITOT2
ISN 0007      COMMON /PRINT/MINPUT,AMU,PITER
ISN 0008      INTEGER DAYS,UPAI,J,RC,UPA
ISN 0009      INTEGER XRU,RRR,SRU,END
ISN 0010      DATA XRU/'LRU',RRR/'RRR',DPGM/ 7*1,15, .53, 22*.52/
ISN 0011      DATA SRU/'SRU',END/'END'

C
ISN 0012      1 FORMAT('1'///10I'0',2X,'PHASE ',I1,' CALCULATIONS',2X,10I'0')
ISN 0013      101 FORMAT ( F5.2,1X,I2,1X,F5.2,1X,I2,T25,I2,T28,F6.4)
ISN 0014      102 FORMAT (1X,F5.2,1X,I2,1X,F5.2,1X,I2,T25,I2,T28,F6.4)
ISN 0015      107 FORMAT(4A4,T19,A3,I2,1X,I2,T32,F5.4,1X,F5.4,T46,F8.2,A3)
ISN 0016      177 FORMAT(15,2X,4A4,2X,A3,I2,1X,I2,2X,F8.4,1X,F8.4,2X,F8.2,2X,A3)

C
ISN 0017      WRITE (6,1) IPASS
ISN 0018      100 IF( IPASS.EQ.1)
+HEAD(5,101) PGMI=DAYS,PGM,NOPS,D1,SKF
ISN 0020      IF( IPASS.EQ.1)
+WRITE(6,102)PGMI,DAYS,PGM,NOPS,D1,SKF
ISN 0022      MUP=NOPS+1
ISN 0023      IKK=MUP
ISN 0024      IUEFF=NOPS
ISN 0025      ITM=0
ISN 0026      MM=0
ISN 0027      IF (D1.GT.DAYS) D1=DAYS
ISN 0029      PGMI=0.
ISN 0030      DO 8888 JJ=1,D1
ISN 0031      8888 PGMI=PGMI+DPGM(JJ)
ISN 0032      IF (IPASS.EQ.1) PGM=PGMI
ISN 0034      103 ITM=ITM+1

C
ISN 0035      105 READ (1,107,END=20) NSN,IMAIN,RC,UPAI,TDR,DOR,COST(ITM),LRU
ISN 0036      IF (IPASS.EQ.2) GO TO 321
ISN 0038      MM=MM+1

C
ISN 0039      IF(MINPUT.EQ.1)
+WRITE(6,177)MM,NSN,IMAIN, RC,UPAI,TDR,DOR,COST(ITM), LRU
ISN 0041      321 NAP(ITM) = UPAI
ISN 0042      BRN=TDR+DDR

C
C      ONLY LRU & SRU ITEMS WILL BE PROCESSED
C
C
ISN 0043      IF(LRU.LT.XRU) GO TO 105
ISN 0045      IF (LRU.EQ.XRU) GO TO 108
ISN 0047      IF(LRU.EQ.SRU) GO TO (105,108),IPASS
ISN 0049      GO TO 105
ISN 0050      108 TOTRU=J.
ISN 0051      I1=01
ISN 0052      IF (IPASS.EQ.2) GO TO 109
ISN 0054      IF( IMAIN.EQ.RRR)I1=01+RC
ISN 0056      109 DO 110 I=1,I1
ISN 0057      110 DNDS(I)=UPAI+TDR+DPGMI(I)
ISN 0058      IF (IPASS.EQ.1) GO TO 145
ISN 0060      J=0 I+1
ISN 0061      DO 115 I=J,DAYS
ISN 0062      115 DNDS(I)=UPAI+TDR+DPGMI(I)
ISN 0063      IF (IMAIN.EQ.RRR) GO TO 155

C
C      LRU RRR PROCESSING
C

```

```

ISN 0065      K=D1+RC
ISN 0066      IF (K.GT.30) K=30
ISN 0068      DO 120 I=1,K
ISN 0069      120 RPRS(I)=0.
ISN 0070      RPRHO=0.
ISN 0071      IF (K.GE.30) GO TO 132
ISN 0073      K=01+1
ISN 0074      DO 125 I=1,K
ISN 0075      125 RPRHO = RPRHO + FLOAT(IPAI)*BRR*DPGM(I)
ISN 0076      RPRS(D1+RC+1) = RPRHO
ISN 0077      J=01+RC+2
ISN 0078      IF (J.GT.30) GO TO 132
ISN 0080      DO 130 I=J,DAYS
ISN 0081      130 RPRS(I) = FLOAT(IPAI)*BRR*DPGM(I-RC)
ISN 0082      132 RQMT(I) = DMDS(1) - RPRS(I)
ISN 0083      DO 135 I=2,DAYS
ISN 0084      135 RQMT(I) = RQMT(I-1) + 0.405(I) - RPRS(I)
ISN 0085      TOTRU=RQMT(1)
ISN 0086      DO 140 I=2,DAYS
ISN 0087      140 IF (RQMT(I).GT.TOTRU) TOTRU=RQMT(I)
ISN 0089      GO TO 165
ISN 0090      145 DO 150 I=1,I1
ISN 0091      150 TOTRU=TOTRU+DMDS(I)

C
C      RR PROCESSING
C
ISN 0092      GO TO 165
ISN 0093      155 DO 160 I=1,DAYS
ISN 0094      TOTRU=TOTRU+DMDS(I)
ISN 0095      160 IF (IMAIN.EQ.0) RRR) RQMT(I)=TOTRU
ISN 0097      165 XMU(ITN)=TOTRU
ISN 0098      GO TO 103
ISN 0099      200 ITOT=ITN-1

C
ISN 0100      IF (IPASS.EQ.1) ITOT1=ITOT
ISN 0102      IF (IPASS.EQ.2) ITOT2=ITOT
ISN 0104      IF (IPASS.EQ.1) PRICE=0.0
ISN 0106      WRITE(6,193) ITOT
ISN 0107      193 FORMAT(//,5X,'TOTAL NUMBER OF ITEMS UNDER CONSIDERATION IS',I4,/)
ISN 0108      RETURN
ISN 0109      END

ISN 0002      SUBROUTINE DTPUT(KIT,ENORS,SUD,PRICE)

C
C      THIS SUBROUTINE PRINT THE FINAL RESULTS OF THE OPTIMIZATION
C
ISN 0003      COMMON /BLK1/ ITOT,MUP,BUDGET/BLK2/XMU(500),NAP(500),COST(500),NFPD
ISN 0004      +(500)/BLK3/NOPS,SKF,IUEFF,IKK/BLK5/LOOPC
ISN 0005      COMMON /BLK0/ ISUM,IPASS
ISN 0006      DIMENSION KIT(500),SN(4)
ISN 0007      DOUBLE PRECISION ENORS,SUD

C
ISN 0007      ITEM=0
ISN 0008      WRITE(6,100) PRICE,ENORS,SUD
ISN 0009      100 FORMAT(//,5X,'PRICE(X)=',F15.2,5X,'E(ENORS/X)=',F10.5,5X,'E(SUD/X)
ISN 0010      =',F10.5,/)
ISN 0011      WRITE(6,110) LOOPC
ISN 0012      110 FORMAT(//,5X,'NUMBER OF ITERATIONS IS',I4,/)
ISN 0013      ***
ISN 0014      CALL TIME
ISN 0015      REWIND 1
ISN 0016      IF (IPASS.EQ.2) STJP
ISN 0017      IPASS=IPASS+1
ISN 0018      50 RETURN
ISN 0019      END

```

ISN 0002

SUBROUTINE TABLE

C
C
C
C
C
C
C

COMPUTATION OF THE CUMULATIVE SUMS OF POISSON DISTRIBUTION

POISSON(I,K) = CUMULATIVE PROBABILITY OF DEMANDS OF QUANTITY K-1
FOR ITEM I
FOR K GREATER THAN NFPO(I), POISSON(I,K) = 1

```
ISN 0003      COMMON POISSON(300,100)
ISN 0004      COMMON /BLK1/ ITOT,MUP,BUDGET/BLK2/XMU(500),NAP(500),COST(500),NFPO
+ (500)/BLK3/NOPS,SKF,IUEEFF,IKK
ISN 0005      COMMON /BLK0/ ISUM,IPASS
ISN 0006      COMMON /PRINT/INPUT,IXMU,MITER
ISN 0007      DOUBLE PRECISION POISSON,TERM,WWW,SUM,TERL
ISN 0008      TERL=5.0-11
ISN 0009      DO 10 I=1,ITOT
ISN 0010      WWW=0BLE(XMU(I))
ISN 0011      TERM=DEXP(-WWW)
ISN 0012      POISSON(I,1)=TERM
ISN 0013      DO 20 J=2,100
ISN 0014      IF( TERM.LT. TERL.AND. POISSON(I,J-1).GT.9.0-01) GO TO 30
ISN 0015      TERM=TERM+WWW/DFLOAT(J-1)
ISN 0016      POISSON(I,J)=POISSON(I,J-1)+TERM
ISN 0017      20 CONTINUE
ISN 0018      NFPO(I)=100
ISN 0019      GO TO 10
ISN 0020      30 NFPO(I)=J
ISN 0021      DO 40 JX=J,100
ISN 0022      40 POISSON(I,JX)=1.0+0J
ISN 0023      10 IF( MXMU.E0.1)
ISN 0024      +WRITE(6,*) I,NFPO(I), NAP(I), XMU(I)
ISN 0025      RETURN
ISN 0026      END
```

ISN 0002

SUBROUTINE INKIT(KIT,PRICE)

C
C
C
C

INCREASE KIT VALUES TO ENSURE THAT THE PROBABILITY OF ANY ITEM
WHICH WILL HAVE HALF OF THE FLEET DOWN IS LESS THAN 50%

```
ISN 0003      DIMENSION KIT(500),KIT0(500)
ISN 0004      COMMON POISSON(300,100)
ISN 0005      COMMON /BLK1/ ITOT,MUP,BUDGET/BLK2/XMU(500),NAP(500),COST(500),NFPO
+ (500)/BLK3/NOPS,SKF,IUEEFF,IKK/ BLK9/VPZERO,S00ZRO
ISN 0006      COMMON /BLK0/ ISUM,IPASS
ISN 0007      COMMON /BLK12/ITOT1,ITOT2
ISN 0008      DOUBLE PRECISION POISSON,VPZERO,S00ZRO,VP,SOC
ISN 0009      DATA KIT0/500*0/
ISN 0010      IF( IPASS.NE.1) GO TO 3
ISN 0011      DO 1 I=1,ITOT
ISN 0012      1 KIT(I)=0
ISN 0013      2 CALL CALC(KIT0,VPZERO,S00ZRO)
ISN 0014      WRITE(6,991)(KIT(I), I=1,ITOT)
ISN 0015      991 FORMAT (//15X, 'EMPTY KIT'/(3X,25(2X,I2)))
ISN 0016      WRITE(6,992) VPZERO,S00ZRO,IUEEFF
ISN 0017      992 FORMAT( / 15X,'VP', F10.5,5X, 'S00',F10.5,5X, 'IUEEFF',I4)
ISN 0018      NEPS=MUP/2
ISN 0019      DO 10 ITEM=1,ITOT
ISN 0020      10 NUM=NEPS+NAP(ITEM)+KIT(ITEM)+1
ISN 0021      IF( NUM.GT.NFPO(ITEM)) GO TO 10
ISN 0022      IF( POISSON(ITEM,NUM).GT.5) GO TO 10
ISN 0023      KIT(ITEM)=KIT(ITEM)+1
ISN 0024      PRICE=PRICE+COST(ITEM)
ISN 0025      GO TO 20
ISN 0026      20 CONTINUE
```

```

ISN 0030      CALL CALC(KIT,VP,SUO)
ISN 0031      WRITE (6,91) ( KIT(I), I=1,ITOT)
ISN 0032      91 FORMAT(///15X,'INITIAL KIT',(3X,25(2X,I2)))
ISN 0033      WRITE (6,994) VP ,SCO ,IUEEFF ,PRICE
ISN 0034      994 FORMAT( / 15X,'VP', F10.5,5X, 'SUO',F10.5,5X,'IUEEFF',I4,5X,
+ 'PRICE', F12.2)
ISN 0035      RETURN
ISN 0036      3 IF(ITOT1.EQ.ITOT2) GO TO 2
ISN 0038      L1=ITOT1+1
ISN 0039      DO 4 M=L1,ITOT2
ISN 0040      4 KIT(M)=0
ISN 0041      GO TO 2
ISN 0042      END

ISN 0052      SUBROUTINE OPTIM(KIT,VP,SDO,PRICE)
C
C      MAIN OPTIMIZATION ROUTINE
C
ISN 0003      DIMENSION KIT(500),KT5718(500)
ISN 0004      DIMENSION BINTER(10)
ISN 0005      COMMON /BLK0/ISUM,IPASS
ISN 0006      COMMON /BLK1/ITOT,4UP,BUDGET/BLK2/XMU(500),NAP(500),COST(500),NFP0
+ (500)/BLK3/NOPS,SKF,IUEEFF,IAK/BLK4/IFLAG,INT,RATI)/BLK5/LOOPC
ISN 0007      COMMON /BLK6/BINTER /BLK9/VPZERO,SDOZRO
ISN 0008      COMMON /BLK8/VPT, VPGOAL,SDOGOL
ISN 0009      COMMON /COUNT/ICOUNT, IADD(200)
ISN 0010      COMMON /PRINT/INPUT,MAXMU,ITER
ISN 0011      DOUBLE PRECISION VP,VPGOAL,VPOELT,VPTST,SUO,SDOGOL,SDODEL,SUTES
+ ,VPZERO,SDOZRO,VPT.
C
ISN 0012      990 FORMAT(/// 15X,'KIT 5718' // (3X,25(2X,I2)))
ISN 0013      994 FORMAT(3,'ITERATION',I4,5X, 'VP =' ,F10.5, 3X, 'SUO =' ,F10.5, 3X,
+ 'PRICE =' , F12.2,3X, 'TOTAL ITEMS ADDED =' ,I3,/(I7,19(15)))
ISN 0014      999 FORMAT(5X,'TARGETS TO BE ATTAINED',/(5X,' COST=' ,F10.2,5X,'NOPS
+ GOAL=' ,F10.5,5X,'SUO GOAL=' ,F10.5, //)
C
ISN 0015      IFLAG=0
ISN 0016      LOOPC=0
ISN 0017      BUDGET=0.0
C
C      KIT 5718 CALCULATION
C
ISN 0018      DO 10 ITEM=1,ITOT
ISN 0019      KT5718(ITEM)=IFIX(XMU(ITEM)*SKF+.5)
ISN 0020      IF(KT5718(ITEM).LT.1) KT5718(ITEM)=1
ISN 0022      BUDGET=BUDGET+COST(ITEM)*FLOAT(KT5718(ITEM))
ISN 0023      10 CONTINUE
C
ISN 0024      CALL CALC(KT5718,VPGOAL,SDOGOL)
ISN 0025      WRITE(6,990) (KT5718(IX),IX=1,ITOT)
ISN 0026      WRITE(6,999) BUDGET,VPGOAL,SDOGOL
ISN 0027      IF (IPASS.EQ.2) VPT=.1+VPGOAL
ISN 0029      IF(IPASS.EQ.1) BUDGET=BUDGET+.9
C
ISN 0031      VPOELT=(VPZERO-VPGOAL)/5.D0
ISN 0032      VPTST=VPZERO-VPOELT
ISN 0033      IVP=0
ISN 0034      SDODEL=(SDOZRO-SDOGOL)/5.D0
ISN 0035      SUTES=SDOZRO-SDODEL
ISN 0036      VP=VPZERO
ISN 0037      SDO=SDOZRO
ISN 0038      ISDO=1
ISN 0039      IDelta=1
ISN 0040      GO TO 223
ISN 0041      210 IF(VP.LE.VPTST) GO TO 220
ISN 0043      IF(SDO.LE.SUTES) GO TO 222
ISN 0045      LOOPC=LOOPC+1
C

```

```

ISN 0046      215 CALL ADD(KIT,VP,SD,PRICE)
ISN 0047      IF(MITER,EU,1)
+             WRITE(6,994) LJOPC,VP,SD,PRICE,ICOUNT,(IACO(I),I=1,ICOUNT)
ISN 0049      DO 216 I=1,ICOUNT.
ISN 0050      216 IADD(I)=0
ISN 0051      ICOUNT=0
ISN 0052      C ***
              IF(MITER,EU,1)
+             CALL TIMEI
ISN 0054      C
              IF(IFLAG,EU,1) GO TO 350
ISN 0056      GO TO 210
ISN 0057      220 VPTEST=VPTEST-VPDEL
ISN 0058      IVP=IVP+1
ISN 0059      IF(SD,GT,SDUTES) GO TO 223
ISN 0061      222 SDUTES=SDUTES-SDDEL
ISN 0062      C
ISN 0063      C
ISN 0064      C
ISN 0066      SELECT PROPER 'RATIO' AND 'INT'
              ISD=ISD+1
ISN 0067      223 IBRAN=6+IVP+ISD
              IF(IBRAN,GT,36) IBRAN=36
              GO TO (304,305,306,307,308,309,310,311,312,313,314,315,316,317,318,319,320,321,322,323,324,325,326,327,328,329,330,331,332,333,334,335,336,337,338,339,340,341,342,343,344,345,346,347,348,349,350,351,352,353,354,355,356,357,358,359,360,361,362,363,364,365,366,367,368,369,370,371,372,373,374,375,376,377,378,379,380,381,382,383,384,385,386,387,388,389,390,391,392,393,394,395,396,397,398,399,400,401,402,403,404,405,406,407,408,409,410,411,412,413,414,415,416,417,418,419,420,421,422,423,424,425,426,427,428,429,430,431,432,433,434,435,436,437,438,439,440,441,442,443,444,445,446,447,448,449,450,451,452,453,454,455,456,457,458,459,460,461,462,463,464,465,466,467,468,469,470,471,472,473,474,475,476,477,478,479,480,481,482,483,484,485,486,487,488,489,490,491,492,493,494,495,496,497,498,499,500,501,502,503,504,505,506,507,508,509,510,511,512,513,514,515,516,517,518,519,520,521,522,523,524,525,526,527,528,529,530,531,532,533,534,535,536,537,538,539,540,541,542,543,544,545,546,547,548,549,550,551,552,553,554,555,556,557,558,559,560,561,562,563,564,565,566,567,568,569,570,571,572,573,574,575,576,577,578,579,580,581,582,583,584,585,586,587,588,589,590,591,592,593,594,595,596,597,598,599,600,601,602,603,604,605,606,607,608,609,610,611,612,613,614,615,616,617,618,619,620,621,622,623,624,625,626,627,628,629,630,631,632,633,634,635,636,637,638,639,640,641,642,643,644,645,646,647,648,649,650,651,652,653,654,655,656,657,658,659,660,661,662,663,664,665,666,667,668,669,670,671,672,673,674,675,676,677,678,679,680,681,682,683,684,685,686,687,688,689,690,691,692,693,694,695,696,697,698,699,700,701,702,703,704,705,706,707,708,709,710,711,712,713,714,715,716,717,718,719,720,721,722,723,724,725,726,727,728,729,730,731,732,733,734,735,736,737,738,739,740,741,742,743,744,745,746,747,748,749,750,751,752,753,754,755,756,757,758,759,760,761,762,763,764,765,766,767,768,769,770,771,772,773,774,775,776,777,778,779,780,781,782,783,784,785,786,787,788,789,790,791,792,793,794,795,796,797,798,799,800,801,802,803,804,805,806,807,808,809,810,811,812,813,814,815,816,817,818,819,820,821,822,823,824,825,826,827,828,829,830,831,832,833,834,835,836,837,838,839,840,841,842,843,844,845,846,847,848,849,850,851,852,853,854,855,856,857,858,859,860,861,862,863,864,865,866,867,868,869,870,871,872,873,874,875,876,877,878,879,880,881,882,883,884,885,886,887,888,889,890,891,892,893,894,895,896,897,898,899,900,901,902,903,904,905,906,907,908,909,910,911,912,913,914,915,916,917,918,919,920,921,922,923,924,925,926,927,928,929,930,931,932,933,934,935,936,937,938,939,940,941,942,943,944,945,946,947,948,949,950,951,952,953,954,955,956,957,958,959,960,961,962,963,964,965,966,967,968,969,970,971,972,973,974,975,976,977,978,979,980,981,982,983,984,985,986,987,988,989,990,991,992,993,994,995,996,997,998,999,1000) IBRAN
ISN 0068      301 VPTEST=0.0+00
ISN 0069      C
ISN 0070      C
ISN 0071      C
ISN 0072      302 RATIO=10.
ISN 0073      GO TO 250
ISN 0074      303 RATIO=25.
ISN 0075      GO TO 250
ISN 0076      304 RATIO=50.
ISN 0077      GO TO 250
ISN 0078      305 RATIO=100.
ISN 0079      GO TO 250
ISN 0080      306 RATIO=500.
ISN 0081      GO TO 250
ISN 0082      307 RATIO=1000.
ISN 0083      SDUTES=0.0+00
ISN 0084      GO TO 250
ISN 0085      C
ISN 0086      311 INT=ITOT/4+1
ISN 0087      GO TO 210
ISN 0088      312 INT=ITOT/8+1
ISN 0089      GO TO 210
ISN 0090      313 INT=ITOT/16+1
ISN 0091      GO TO 210
ISN 0092      314 INT=ITOT/32+1
ISN 0093      GO TO 210
ISN 0094      315 INT=ITOT/64+1
ISN 0095      GO TO 210
ISN 0096      316 INT=1
ISN 0097      GO TO 210
ISN 0098      333 RATIO=50.
ISN 0099      IF(BUDGET-PRICE).GE.40000.) GO TO 334
ISN 0100      INT=1
ISN 0101      GO TO 215
ISN 0102      334 IF(BUDGET-PRICE).GE.20000.) GO TO 335
ISN 0103      INT=ITOT/64+1
ISN 0104      GO TO 215
ISN 0105      335 IF(BUDGET-PRICE).GE.10000.) GO TO 336
ISN 0106      INT=ITOT/16+1
ISN 0107      GO TO 215
ISN 0108      336 INT=ITOT/8+1
ISN 0109      GO TO 215
ISN 0110      350 IFLAG=0
ISN 0111      RETURN
              END

```

```

ISN 0002      SUBROUTINE ADDKIT,VP,SDO,PRICE)
C
C      THIS SUBROUTINE ADDS INT DIFFERENT ITEMS TO THE KIT BY
C      MARGINAL ANALYSIS
C
ISN 0003      DIMENSION TS(500),KIT(500)
ISN 0004      DIMENSION BINT(10),S1(50),SS1(50),SIDEL(50)
ISN 0005      COMMON POISON(300,100)
ISN 0006      COMMON /BLK0/ISUM,IPASS
ISN 0007      COMMON /BLK1/ITOT,AUP,BUDGET/BLK2/XMU(500),NAP(500),COST(500),NFPD
+ (500)/BLK3/NOPS,SKF,IUEEFF,IKK/BLK4/IFLAG,INT,RATIO
COMMON /BLK6/BINTER
ISN 0008      COMMON /BLK7/S1/BLK8/VPT,      VPGJAL,SDJGUL
ISN 0009      COMMON /COUNT/ICOUNT,IAADD(200)
ISN 0010      DOUBLE PRECISION PJISJN,TS,DIF,VP,A,SDO,SDOP,T,
+ S1,SS1,S1TEMP,SSITEM,SISUM,SSISUM,SIDEL
ISN 0012      DOUBLE PRECISION JPGJAL,SDGJL,EP5,VPT
ISN 0013      DATA INTER/1/,MFLAG/0/,EPS/1.0-12/
C
ISN 0014      NOPS1=NOPS+1
ISN 0015      IF(IPASS.EQ.2) BTEMP=BINTER(INTER)
ISN 0017      IF (IPASS.EQ.1) BTEMP=BUDGET
ISN 0019      IF( IPASS.EQ.1.AND.BINTER(INTER).LT.BUDGET) BTEMP=BINTER(INTER)
ISN 0021      SITEMP=0.00
C
ISN 0022      DO 30 JS1=1,NOPS
ISN 0023      30 SITEMP=SITEMP+S1(JS1)
ISN 0024      SISUM=NOPS+S1(NOPS1)-SITEMP
C
ISN 0025      DO 99 ITM=1,ITOT
ISN 0026      IXL=KIT(ITM)+1
ISN 0027      IXM=KIT(ITM)+(NOPS-1)+NAP(ITM)+2
ISN 0028      IF(IAM.GE. NFPD(ITM)) IXM=NFPD(ITM)
ISN 0030      IF( IXL.GE. NFPD(ITM)) IXL=NFPD(ITM)
ISN 0032      DIF= POISON(ITM,IXM)-POISON(ITM,IXL)
C
ISN 0033      DO 20 K=1,NOPS1
ISN 0034      KJ=KIT(ITM)+(K-1)+NAP(ITM)+1
ISN 0035      KJ1=KJ+1
ISN 0036      IF (KJ.GE. NFPD(ITM)) GO TO 333
ISN 0038      IF( POISON(ITM,KJ).EQ.0.00) GO TO 222
ISN 0040      SS1(K)=S1(K)=POISON(ITM,KJ1)/POISON(ITM,KJ)
ISN 0041      GO TO 20
ISN 0042      222 SS1(K)=1.00
ISN 0043      DO 223 KI=1,ITOT
ISN 0044      KI1=KIT(KI)+(K-1)+NAP(KI)+1
ISN 0045      IF(KI.EQ.ITM) KI1=KI+1
ISN 0047      IF (KI1.GE. NFPD(KI)) GO TO 223
ISN 0049      SS1(K)=SS1(K)+POISON(KI,KI1)
ISN 0050      IF(SS1(K).LE.EPS) GO TO 321
ISN 0052      223 CONTINUE
C
ISN 0053      GO TO 20
ISN 0054      321 SS1(K)=0.00
ISN 0055      GO TO 20
ISN 0056      333 SS1(K)=S1(K)
ISN 0057      20 CONTINUE
C
ISN 0058      SSITEM=0.00
ISN 0059      DO 40 JS1=1,NOPS
ISN 0060      40 SSITEM=SSITEM+SS1(JS1)
ISN 0061      SSISUM=NOPS+SS1(NOPS1)-SSITEM
ISN 0062      A=SSISUM
ISN 0063      TS(ITM)=(VP-A)*DOUBLE(RATIO)+DIF/DOUBLE(COST(ITM))
ISN 0064      99 CONTINUE
C

```

```

C      ADD 2 OF THE INT4 ITEMS WITH THE HIGHEST IMPROVEMENT RATIOS
C      ADD 1 OF THE NEXT HIGHEST INT-INT/4 ITEMS
C
ISN 0065      INT4=INT/4
ISN 0066      DO 102 I=1,INT
ISN 0067      T=TS(I)
ISN 0068      ITM=1
ISN 0069      DO 101 J=2,ITOT
ISN 0070      IF(TS(J).LT.T) GO TO 101
ISN 0072      T=TS(J)
ISN 0073      ITM=J
ISN 0074      101 CONTINUE
ISN 0075      KIT(ITM)=KIT(ITM)+1
ISN 0076      ICOUNT=ICOUNT+1
ISN 0077      IADD(ICOUNT)=ITM
ISN 0078      IF(MFLAG.EQ.1) CALL CALC(KIT,VP,SDO)
ISN 0080      PRICE=PRICE+COST(ITM)
ISN 0081      TS(ITM)=J.D+30
ISN 0082      IF(PRICE.GT.BTEMP.JR.(MFLAG.EQ.1 .AND. (VP.LE.VPGJAL.AND.
+      SDO.LE.SDOGOL))) GO TO 105
ISN 0084      IF(INT4.EQ.0) GO TO 102
ISN 0086      KIT(ITM)=KIT(ITM)+1
ISN 0087      ICOUNT=ICOUNT+1
ISN 0088      IADD(ICOUNT)=ITM
ISN 0089      IF(MFLAG.EQ.1) CALL CALC(KIT,VP,SDO)
ISN 0091      PRICE=PRICE+COST(ITM)
ISN 0092      IF(PRICE.GT.BTEMP.JR.(MFLAG.EQ.1 .AND. (VP.LE.VPGJAL.AND.
+      SDO.LE.SDOGOL))) GO TO 105
ISN 0094      INT4=INT4-1
ISN 0095      102 CONTINUE
C
ISN 0096      CALL CALC(KIT,VP,SDO)
ISN 0097      IF(VP.LE.VPT) MFLAG=1
ISN 0099      RETURN
C
ISN 0100      105 IF((IPASS.EQ.1).AND.BTEMP.EQ.BUDGET) .OR.
+      (MFLAG.EQ.1 .AND. (VP.LE.VPGJAL.AND.
+      SDO.LE.SDOGOL))) IFLAG=1
ISN 0102      CALL CALC(KIT,VP,SDO)
ISN 0103      IF(VP.LE.VPT) MFLAG=1
ISN 0105      IF(IFLAG.NE.1) WRITE(6,995) BTEMP
ISN 0107      IF(IFLAG.EQ.1) WRITE(6,994) PRICE
ISN 0109      WRITE(6,991) ((I(I), I=1,ITOT)
ISN 0110      WRITE(6,993) VP,SDO,IUEFF,PRICE,ITM,COST(ITM)
ISN 0111      S1DEL(1)=S1(1)
ISN 0112      DO 95 J=2,NOPS
ISN 0113      95 S1DEL(J)=S1(J)-S1(J-1)
ISN 0114      WRITE(6,996) (S1DEL(I),I=1,NOPS)
C
ISN 0115      IF(IFLAG.NE.1) INTER=INTER+1
ISN 0117      RETURN
C
ISN 0118      991 FORMAT( /15X,'KIT',/,(1X,25(2X,12)))
ISN 0119      993 FORMAT( /15X,'V',F10.5,5X,'SDO',F10.5,5X,'IUEFF',14,5X,
+      'PRICE',F12.2,5X,'LAST ITEM ADDED',15,5X,'ITEM COST',F10.2,
+      //)
ISN 0120      995 FORMAT( /// 2X, 'BUDGET S=',F12.2)
ISN 0121      996 FORMAT( 51 4X,E20.12)
ISN 0122      994 FORMAT( ///2X, 'PRICE' OF OPTIMAL KIT =', F12.2)
ISN 0123      END

```


BUDGET 1500000.00

VP	1057847	SDD	727562	LUUEFF	24	PRICE	1501613.00	LAST ITEM ADDED	165	ITEM COST=
0	0	0	0	0	0	0	0	0	0	0
1	0	0	0	0	0	0	0	0	0	0
1	1	0	0	0	0	0	0	0	0	0
1	1	0	0	0	0	0	0	0	0	0
0	2	2	1	1	0	0	0	0	0	0
2	2	1	1	0	0	0	0	0	0	0
0	1	1	1	1	1	1	1	1	1	1
1	2	1	0	0	0	0	0	0	0	0
1	2	1	0	0	0	0	0	0	0	0

0.0

0.182921308220-02	0.0	0.14389467160-08	0.9900516499340-06	0.8745083051200-04
0.182316263150-04	0.125352360890-01	0.1145240189550-00	0.1675350275970-00	0.1675350275970-00
0.240673501370-01	0.1191102090610-00	0.7927940958900-01	0.4859363766790-01	0.4859363766790-01
0.1101213005810-02	0.430417765420-02	0.4327347560730-02	0.2205303898560-02	0.2205303898560-02
	0.2589903524600-03	0.1219409749000-03		

BUDGET 3700000.00

VP	630619	SDD	3593377	LUUEFF	24	PRICE	333970.00	LAST ITEM ADDED	166	ITEM COST=
0	0	0	0	0	0	0	0	0	0	0
1	0	0	0	0	0	0	0	0	0	0
1	1	0	0	0	0	0	0	0	0	0
1	1	0	0	0	0	0	0	0	0	0
0	2	2	1	1	0	0	0	0	0	0
2	2	1	1	1	0	0	0	0	0	0
0	1	1	1	1	1	1	1	1	1	1
2	2	1	1	1	1	1	1	1	1	1
2	2	1	1	1	1	1	1	1	1	1

0.0

0.1048807815590-10	0.2998048127500-04	0.4312166343100-22	0.4208478085150-01	0.1299807085520-00
0.200862998230-00	0.2048660617190-00	0.1616959239710-00	0.1089943992610-00	0.6658344450270-01
0.3012453659800-01	0.2084041154930-01	0.1098865322740-01	0.5623937825970-02	0.2805254432950-02
0.1367763457800-02	0.0531641035930-03	0.3058748189240-03	0.1405646150140-03	0.6340417659370-04
0.280688855800-04	0.1219168901860-04	0.4193436343750-05	0.2168753733430-05	

●●●●●●●●

TOTAL NUMBER OF ITEMS UNDER CONSIDERATION IS 250

[illegible][illegible]

PRICE \$280115.00

FILE # 20

500 693.47 625

[illegible]

TARGETS TO BE ATTAINED

COST=97502983.00 MARGINAL: 8.13340 SUBGOAL: 163.92675

BUDGET s= 4500000.00

ALT	0	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	129	130	131	132	133	134	135	136	137	138	139	140	141	142	143	144	145	146	147	148	149	150	151	152	153	154	155	156	157	158	159	160	161	162	163	164	165	166	167	168	169	170	171	172	173	174	175	176	177	178	179	180	181	182	183	184	185	186	187	188	189	190	191	192	193	194	195	196	197	198	199	200	201	202	203	204	205	206	207	208	209	210	211	212	213	214	215	216	217	218	219	220	221	222	223	224	225	226	227	228	229	230	231	232	233	234	235	236	237	238	239	240	241	242	243	244	245	246	247	248	249	250	251	252	253	254	255	256	257	258	259	260	261	262	263	264	265	266	267	268	269	270	271	272	273	274	275	276	277	278	279	280	281	282	283	284	285	286	287	288	289	290	291	292	293	294	295	296	297	298	299	300	301	302	303	304	305	306	307	308	309	310	311	312	313	314	315	316	317	318	319	320	321	322	323	324	325	326	327	328	329	330	331	332	333	334	335	336	337	338	339	340	341	342	343	344	345	346	347	348	349	350	351	352	353	354	355	356	357	358	359	360	361	362	363	364	365	366	367	368	369	370	371	372	373	374	375	376	377	378	379	380	381	382	383	384	385	386	387	388	389	390	391	392	393	394	395	396	397	398	399	400	401	402	403	404	405	406	407	408	409	410	411	412	413	414	415	416	417	418	419	420	421	422	423	424	425	426	427	428	429	430	431	432	433	434	435	436	437	438	439	440	441	442	443	444	445	446	447	448	449	450	451	452	453	454	455	456	457	458	459	460	461	462	463	464	465	466	467	468	469	470	471	472	473	474	475	476	477	478	479	480	481	482	483	484	485	486	487	488	489	490	491	492	493	494	495	496	497	498	499	500	501	502	503	504	505	506	507	508	509	510	511	512	513	514	515	516	517	518	519	520	521	522	523	524	525	526	527	528	529	530	531	532	533	534	535	536	537	538	539	540	541	542	543	544	545	546	547	548	549	550	551	552	553	554	555	556	557	558	559	560	561	562	563	564	565	566	567	568	569	570	571	572	573	574	575	576	577	578	579	580	581	582	583	584	585	586	587	588	589	590	591	592	593	594	595	596	597	598	599	600	601	602	603	604	605	606	607	608	609	610	611	612	613	614	615	616	617	618	619	620	621	622	623	624	625	626	627	628	629	630	631	632	633	634	635	636	637	638	639	640	641	642	643	644	645	646	647	648	649	650	651	652	653	654	655	656	657	658	659	660	661	662	663	664	665	666	667	668	669	670	671	672	673	674	675	676	677	678	679	680	681	682	683	684	685	686	687	688	689	690	691	692	693	694	695	696	697	698	699	700	701	702	703	704	705	706	707	708	709	710	711	712	713	714	715	716	717	718	719	720	721	722	723	724	725	726	727	728	729	730	731	732	733	734	735	736	737	738	739	740	741	742	743	744	745	746	747	748	749	750	751	752	753	754	755	756	757	758	759	760	761	762	763	764	765	766	767	768	769	770	771	772	773	774	775	776	777	778	779	780	781	782	783	784	785	786	787	788	789	790	791	792	793	794	795	796	797	798	799	800	801	802	803	804	805	806	807	808	809	810	811	812	813	814	815	816	817	818	819	820	821	822	823	824	825	826	827	828	829	830	831	832	833	834	835	836	837	838	839	840	841	842	843	844	845	846	847	848	849	850	851	852	853	854	855	856	857	858	859	860	861	862	863	864	865	866	867	868	869	870	871	872	873	874	875	876	877	878	879	880	881	882	883	884	885	886	887	888	889	890	891	892	893	894	895	896	897	898	899	900	901	902	903	904	905	906	907	908	909	910	911	912	913	914	915	916	917	918	919	920	921	922	923	924	925	926	927	928	929	930	931	932	933	934	935	936	937	938	939	940	941	942	943	944	945	946	947	948	949	950	951	952	953	954	955	956	957	958	959	960	961	962	963	964	965	966	967	968	969	970	971	972	973	974	975	976	977	978	979	980	981	982	983	984	985	986	987	988	989	990	991	992	993	994	995	996	997	998	999	1000	1001	1002	1003	1004	1005	1006	1007	1008	1009	1010	1011	1012	1013	1014	1015	1016	1017	1018	1019	1020	1021	1022	1023	1024	1025	1026	1027	1028	1029	1030	1031	1032	1033	1034	1035	1036	1037	1038	1039	1040	1041	1042	1043	1044	1045	1046	1047	1048	1049	1050	1051	1052	1053	1054	1055	1056	1057	1058	1059	1060	1061	1062	1063	1064	1065	1066	1067	1068	1069	1070	1071	1072	1073	1074	1075	1076	1077	1078	1079	1080	1081	1082	1083	1084	1085	1086	1087	1088	1089	1090	1091	1092	1093	1094	1095	1096	1097	1098	1099	1100	1101	1102	1103	1104	1105	1106	1107	1108	1109	1110	1111	1112	1113	1114	1115	1116	1117	1118	1119	1120	1121	1122	1123	1124	1125	1126	1127	1128	1129	1130	1131	1132	1133	1134	1135	1136	1137	1138	1139	1140	1141	1142	1143	1144	1145	1146	1147	1148	1149	1150	1151	1152	1153	1154	1155	1156	1157	1158	1159	1160	1161	1162	1163	1164	1165	1166	1167	1168	1169	1170	1171	1172	1173	1174	1175	1176	1177	1178	1179	1180	1181	1182	1183	1184	1185	1186	1187	1188	1189	1190	1191	1192	1193	1194	1195	1196	1197	1198	1199	1200	1201	1202	1203	1204	1205	1206	1207	1208	1209	1210	1211	1212	1213	1214	1215	1216	1217	1218	1219	1220	1221	1222	1223	1224	1225	1226	1227	1228	1229	1230	1231	1232	1233	1234	1235	1236	1237	1238	1239	1240	1241	1242	1243	1244	1245	1246	1247	1248	1249	1250	1251	1252	1253	1254	1255	1256	1257	1258	1259	1260	1261	1262	1263	1264	1265	1266	1267	1268	1269	1270	1271	1272	1273	1274	1275	1276	1277	1278	1279	1280	1281	1282	1283	1284	1285	1286	1287	1288	1289	1290	1291	1292	1293	1294	1295	1296	1297	1298	1299	1300	1301	1302	1303	1304	1305	1306	1307	1308	1309	1310	1311	1312	1313	1314	1315	1316	1317	1318	1319	1320	1321	1322	1323	1324	1325	1326	1327	1328	1329	1330	1331	1332	1333	1334	1335	1336	1337	1338	1339	1340	1341	1342	1343	1344	1345	1346	1347	1348	1349	1350	1351	1352	1353	1354	1355	1356	1357	1358	1359	1360	1361	1362	1363	1364	1365	1366	1367	1368	1369	1370	1371	1372	1373	1374	1375	1376	1377	1378	1379	1380	1381	1382	1383	1384	1385	1386	1387	1388	1389	1390	1391	1392	1393	1394	1395	1396	1397	1398	1399	1400	1401	1402	1403	1404	1405	1406	1407	1408	1409	1410	1411	1412	1413	1414	1415	1416	1417	1418	1419	1420	1421	1422	1423	1424	1425	1426	1427	1428	1429	1430	1431	1432	1433	1434	1435	1436	1437	1438	1439	1440	1441	1442	1443	1444	1445	1446	1447	1448	1449	1450	1451	1452	1453	1454	1455	1456	1457	1458	1459	1460	1461	1462	1463	1464	1465	1466	1467	1468	1469	1470	1471	1472	1473	1474	1475</
-----	---	---	---	---	---	---	---	---	---	---	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	--------

APPENDIX D

D.1 Listing of F4D Input Data

D.2 Listing of Phase 1 and Phase 2 F4D Input Data

APPENDIX D.1

**LISTING OF
F4D INPUT DATA**

D.1.1

DO29 SYSTEM CONTROL/INPUT CARD FORMAT

I. "A" CARD : PROGRAM CONTROL CARD

<u>Field</u>	<u>Column</u>	<u>Type</u> *	<u>Description</u>
1	1 - 5	I5	Initial Program in Flying Hours.
2	7 - 8	I2	WRSK Support Period in Days.
3	10 -14	I5	Total Program in Flying Hours.
4	16 -17	I2	Unit of Equipment.
5	19 -20	A2	Run Sequence Number.
6	22 -23	I2	Number of Kit Authorizations.
7	25 -26	I2	Number of Set Up Days.
8	54	A1	A (Card Indicator).

33- STOCKAGE
FACTOR

II. "B" CARD : KIT SUMMARY CONTROL CARD

<u>Field</u>	<u>Column</u>	<u>Type</u> *	<u>Description</u>
1	1 -10	A10	Kit Serial Number.
	54	A1	B (Card Identifier).

III. "C" CARD : ITEM DATA CARD

<u>Field</u>	<u>Column</u>	<u>Type</u> *	<u>Description</u>
1	1 -15	A15	National Stock Number (Left Justified).
2	17	I1	Note Code.
3	19 -21	A3	RR/RRR Maintenance Concept (Left Justified).
4	22 -23	I2	Repair Cycle Time in Days.
5	25 -26	I2	Quantity Per Assembly.
6	28 -30	A3	Application Code.
7	32 -36	F54	Total OIM Demand Rate.
8	38 -42	F54	Depot Demand Rate.
9	44	A1	Routing Identifier.
10	46 -53	F8.2	Unit Cost.
11	54 -56	A3	LRU/SRU/EOQ/NOP Type Item.
12	58 -62	A5	Work Unit Code (Left Justified).
13	73 -80	A8	Noun.

* <u>Type</u>	<u>Explanation</u>
I	Numeric input only, leading zeros as required, right justified.
A	Alphabetic or numeric input.
F	Numeric Input, decimal assigned by system, leading zeros as required, right justified.

[illegible]

D.1.6

CARLOS
C210
C211
C227
C229
C226
C227
C228
C230
C231
C232
C234
C235
C236
C237
C238
C239
C240
C241
C242
C243
C244
C245
C246
C247
C248
C249
C250
C251
C252
C253
C254
C255
C256
C257
C258
C259
C260
C261
C262
C263
C264
C265
C266
C267
C268
C269
C270
C271

APPENDIX D.2

LISTING OF

PHASE 1 AND PHASE 2 F4D INPUT DATA

D.2.1

Phase 1 Data

1	1	25927.00	0.65423	RR	LRU
2	1	3054.00	0.01932	RR	LRU
3	1	600.00	0.03737	RR	LRU
4	1	8166.00	0.04807	RR	LRU
5	1	11107.00	0.02875	RR	LRU
6	1	3162.00	0.03737	RR	LRU
7	1	13900.00	0.01276	RR	LRU
8	1	2332.00	0.04025	RR	LRU
9	1	2868.00	0.03599	RR	LRU
10	1	3550.00	0.03599	RR	LRU
11	1	8784.00	0.07348	RR	LRU
12	1	12366.00	0.04025	RR	LRU
13	1	1627.00	0.25553	RR	LRU
14	1	2369.00	0.16422	RR	LRU
15	1	5258.00	0.11396	RR	LRU
16	1	1325.00	0.03231	RR	LRU
17	1	2975.00	0.12086	RR	LRU
18	1	2520.00	0.10879	RR	LRU
19	1	303.24	0.13075	RR	LRU
20	1	799.00	0.05048	RR	LRU
21	1	1424.00	0.06348	RR	LRU
22	1	1056.00	0.05773	RR	LRU
23	1	845.00	0.04025	RR	LRU
24	1	919.80	0.04726	RR	LRU
25	1	717.70	0.03749	RR	LRU
26	1	927.00	0.09522	RR	LRU
27	1	3126.00	0.10223	RR	LRU
28	1	2122.00	0.06049	RR	LRU
29	1	6891.00	0.21332	RR	LRU
30	1	582.00	0.11569	RR	LRU
31	2	3745.90	0.10948	RR	LRU
32	1	692.70	0.05198	RR	LRU
33	1	1333.00	0.07383	RR	LRU
34	1	1333.00	0.08889	RR	LRU
35	1	3202.00	0.22859	RR	LRU
36	1	516.00	0.71875	RR	LRU
37	1	1241.00	0.29003	RR	LRU
38	1	2105.00	0.05750	RR	LRU
39	1	824.50	0.07820	RR	LRU
40	1	1342.00	0.06014	RR	LRU
41	2	3914.00	0.17480	RR	LRU
42	2	891.73	0.17227	RR	LRU
43	2	933.80	0.04692	RR	LRU
44	2	2058.00	0.19343	RR	LRU
45	2	349.10	0.04853	RR	LRU
46	2	1242.00	0.06923	RR	LRU
47	2	471.70	0.38180	RR	LRU
48	5	304.30	0.27370	RR	LRU
49	1	1317.00	0.04393	RR	LRU
50	2	2083.00	0.05865	RR	LRU
51	2	1044.00	0.15502	RR	LRU
52	2	637.90	0.13156	RR	LRU
53	2	487.90	0.38312	RR	LRU
54	2	9901.00	0.32798	RR	LRU
55	2	275.30	0.06624	RR	LRU
56	1	2975.00	0.17043	RR	LRU
57	4	1130.00	0.96370	RR	LRU
58	5	886.00	0.07624	RR	LRU
59	1	320.30	0.14087	RR	LRU
60	1	1300.00	0.04174	RR	LRU
61	1	1300.00	0.05566	RR	LRU
62	1	456.90	0.06175	RR	LRU
63	1	309.70	0.04335	RR	LRU
64	1	945.50	0.20493	RR	LRU
65	1	288.00	0.12121	RR	LRU

66	2	978.00	1.10078	RR	LRU
67	2	708.00	0.72450	RR	LRU
68	1	4000.00	0.420412	RR	LRU
69	2	412.00	0.01035	RR	LRU
70	2	262.10	0.06992	RR	LRU
71	2	1358.00	0.64111	RR	LRU
72	1	1874.00	0.17284	RR	LRU
73	1	286.00	0.16422	RR	LRU
74	1	3658.00	0.06382	RR	LRU
75	1	335.00	0.03579	RR	LRU
76	2	2129.00	0.37237	RR	LRU
77	1	2166.00	0.14973	RR	LRU
78	1	1158.00	0.12788	RR	LRU
79	1	3550.00	0.19251	RR	LRU
80	1	3290.00	0.43493	RR	LRU
81	1	802.40	0.14715	RR	LRU
82	1	1115.00	0.25392	RR	LRU
83	2	267.80	0.23000	RR	LRU
84	1	1051.00	0.54383	RR	LRU
85	1	875.50	0.13524	RR	LRU
86	2	786.50	0.24978	RR	LRU
87	1	1401.00	0.14846	RR	LRU
88	1	1585.00	0.06290	RR	LRU
89	1	6180.00	0.84628	RR	LRU
90	1	2553.00	0.05186	RR	LRU
91	2	2292.00	0.11707	RR	LRU
92	1	831.20	0.05681	RR	LRU
93	1	641.69	0.11638	RR	LRU
94	1	1421.00	0.11500	RR	LRU
95	1	1656.00	0.19090	RR	LRU
96	1	165.00	0.13570	RR	LRU
97	1	1656.00	0.07003	RR	LRU
98	1	591.20	0.06359	RR	LRU
99	1	21858.00	0.66447	RR	LRU
100	2	1275.00	0.12144	RR	LRU
101	1	2086.00	0.08820	RR	LRU
102	1	253.60	0.03772	RR	LRU
103	1	605.64	0.48567	RR	LRU
104	1	1308.00	0.21689	RR	LRU
105	2	457.30	0.41814	RR	LRU
106	1	1224.00	0.27174	RR	LRU
107	1	1359.00	0.10718	RR	LRU
108	1	4000.00	2.44950	RRR	LRU
109	1	2560.00	0.64124	RRR	LRU
110	1	12789.00	0.27715	RR	LRU
111	2	430.50	0.08027	RR	LRU
112	1	1082.00	0.06382	RR	LRU
113	1	892.00	0.07682	RR	LRU
114	2	1429.00	0.06003	RR	LRU
115	2	1400.00	0.16882	RR	LRU
116	2	250.00	0.06118	RR	LRU
117	1	471.70	0.10154	RR	LRU
118	1	719.70	0.51589	RR	LRU
119	1	1686.00	0.08096	RR	LRU
120	1	1449.00	0.25001	RR	LRU
121	2	970.50	0.09177	RR	LRU
122	1	1278.00	0.12776	RR	LRU
123	1	886.00	0.11178	RR	LRU
124	3	659.40	0.08901	RR	LRU
125	1	1115.00	0.08636	RR	LRU
126	2	702.60	0.48323	RR	LRU
127	1	5954.00	1.02649	RR	LRU
128	1	2380.00	0.52808	RR	LRU
129	1	650.00	0.09016	RR	LRU
130	1	480.00	0.01035	RR	LRU
131	1	840.00	0.07371	RR	LRU

132	1	5000.00	0.62732	RR	LRU
133	1	1010.00	0.10718	RR	LRU
134	1	718.75	0.10430	RR	LRU
135	2	279.00	0.26818	RR	LRU
136	1	129.00	0.07728	RR	LRU
137	1	3097.00	0.71875	RR	LRU
138	2	696.00	0.20332	RR	LRU
139	1	3971.00	0.70081	RR	LRU
140	1	2760.00	0.48288	RR	LRU
141	1	2711.00	1.04075	RR	LRU
142	1	9674.00	1.46027	RR	LRU
143	1	3275.00	0.08119	RR	LRU
144	2	1615.00	0.87929	RR	LRU
145	1	236.10	0.03875	RR	LRU
146	1	1674.00	0.04611	RR	LRU
147	2	436.00	0.20102	RR	LRU
148	1	35511.00	1.19232	RR	LRU
149	1	1472.00	0.12960	RR	LRU
150	1	2730.00	0.20838	RR	LRU
151	1	17694.00	0.99969	RR	LRU
152	1	3914.00	0.34258	RR	LRU
153	1	3223.00	0.20642	RR	LRU
154	1	1183.00	0.20044	RR	LRU
155	1	1471.00	0.09085	RR	LRU
156	1	1984.00	0.74658	RRR	LRU
157	1	41566.00	2.91916	RRR	LRU
158	1	41566.00	0.23000	RRR	LRU
159	1	3668.00	1.74018	RRR	LRU
160	1	1200.00	1.01338	RRR	LRU
161	1	13000.00	1.73880	RRR	LRU
162	1	13000.00	1.18174	RRR	LRU
163	1	51886.00	16.33919	RRR	LRU
164	1	39000.00	6.78960	RRR	LRU
165	1	60485.00	8.50310	RRR	LRU
166	1	40102.00	9.07718	RRR	LRU
167	1	5951.00	2.91548	RRR	LRU
168	1	2200.00	3.87596	RRR	LRU
169	1	5352.00	1.12884	RRR	LRU
170	1	4687.00	1.91958	RRR	LRU
171	1	45581.00	11.02481	RRR	LRU
172	1	29011.00	0.24426	RRR	LRU
173	1	13721.00	4.60046	RRR	LRU
174	1	11520.00	4.38978	RRR	LRU
175	1	32182.00	5.55864	RRR	LRU
176	1	566.50	0.29566	RRR	LRU
177	1	1200.00	0.71001	RRR	LRU
178	1	4496.00	1.00567	RRR	LRU
179	1	21265.00	2.98540	RRR	LRU
180	1	40606.00	11.13084	RRR	LRU

Phase 2 Data

1	0	1	11.38937	25927.00	RR	LNU
2	0	1	0.33633	3054.00	RR	LNU
3	0	1	0.65065	600.00	RR	LNU
4	0	1	0.83683	8166.00	RR	LNU
5	0	1	0.50050	11107.00	RR	LNU
6	0	1	0.65065	3162.00	RR	LNU
7	0	1	0.22222	13900.00	RR	LNU
8	0	1	0.70070	2332.00	RR	LNU
9	0	1	0.62662	2668.00	RR	LNU
10	0	1	0.62662	3550.00	RR	LNU
11	0	1	1.27928	8784.00	RR	LNU
12	0	1	0.70070	12366.00	RR	LNU
13	1	1	4.44843	1627.00	RR	LNU
14	0	1	2.65285	2369.00	RR	LNU
15	0	1	1.98398	5258.00	RR	LNU
16	0	1	0.56256	1325.00	RR	LNU
17	0	1	2.10409	2575.00	RR	LNU
18	0	1	1.89388	2520.00	RR	LNU
19	1	1	2.27625	303.24	RR	LNU
20	0	1	0.87888	799.00	RR	LNU
21	0	1	1.10510	1424.00	RR	LNU
22	0	1	1.00500	1056.00	RR	LNU
23	0	1	0.70070	645.00	RR	LNU
24	0	1	0.82282	919.80	RR	LNU
25	0	1	0.65265	717.70	RR	LNU
26	1	1	1.65765	927.00	RR	LNU
27	0	1	1.77977	3126.00	RR	LNU
28	0	1	1.05305	2122.00	RR	LNU
29	0	1	3.71369	6891.00	RR	LNU
30	1	1	2.01401	582.00	RR	LNU
31	0	2	1.90589	3745.00	RR	LNU
32	0	1	0.90490	692.70	RR	LNU
33	0	1	1.28528	1333.00	RR	LNU
34	0	1	1.54754	1333.00	RR	LNU
35	0	1	0.97797	3202.00	RR	LNU
36	0	1	12.51249	516.00	RR	LNU
37	1	1	5.04903	1241.00	RR	LNU
38	0	1	1.00100	2105.00	RR	LNU
39	1	1	1.36135	824.50	RR	LNU
40	0	1	1.04704	1342.00	RR	LNU
41	0	2	3.04303	3514.00	RR	LNU
42	1	2	2.99898	891.73	RR	LNU
43	0	2	0.81682	933.80	RR	LNU
44	0	2	3.30735	2058.00	RR	LNU
45	1	2	0.84484	349.10	RR	LNU
46	0	2	1.20520	1242.00	RR	LNU
47	2	20	6.64663	471.70	RR	LNU
48	2	5	4.76474	304.30	RR	LNU
49	0	1	0.76476	1317.00	RR	LNU
50	0	2	1.02102	2083.00	RR	LNU
51	1	2	2.09867	1044.00	RR	LNU
52	1	8	2.29027	637.50	RR	LNU
53	2	2	6.67065	487.90	RR	LNU
54	0	2	5.70970	9901.00	RR	LNU
55	1	2	1.15315	275.30	RR	LNU
56	0	1	2.96695	2575.00	RR	LNU
57	2	4	16.77672	1130.00	RR	LNU
58	0	1	1.32732	886.00	RR	LNU
59	1	5	2.45244	320.30	RR	LNU
60	0	1	0.72672	1300.00	RR	LNU
61	0	1	0.96897	1300.00	RR	LNU
62	1	1	1.07507	456.90	RR	LNU
63	1	1	0.75475	309.70	RR	LNU
64	1	1	3.56756	945.50	RR	LNU
65	1	1	2.11010	288.00	RR	LNU

66	3	2	19.16312	578.00	RR	LRU
67	2	2	12.61259	708.00	RR	LRU
68	0	1	3.55354	4000.00	RR	LRU
69	0	2	0.18018	412.00	RR	LRU
70	1	2	1.21721	262.10	RR	LRU
71	2	2	14.64261	1358.00	RR	LRU
72	0	1	3.00899	1874.00	RR	LRU
73	1	1	2.85885	280.00	RR	LRU
74	0	1	1.11111	3658.00	RR	LRU
75	1	1	1.66766	335.00	RR	LRU
76	1	2	6.48247	2129.00	RR	LRU
77	0	1	2.60659	2166.00	RR	LRU
78	1	1	2.22621	1158.00	RR	LRU
79	0	1	3.35134	3550.00	RR	LRU
80	1	1	7.57156	3290.00	RR	LRU
81	0	1	0.62082	802.40	RR	LRU
82	1	1	4.42041	1115.00	RR	LRU
83	2	2	4.00398	267.80	RR	LRU
84	2	1	9.46745	1051.00	RR	LRU
85	1	1	2.35434	875.50	RR	LRU
86	1	2	4.34833	786.50	RR	LRU
87	1	1	2.58458	1401.00	RR	LRU
88	0	1	1.09509	1585.00	RR	LRU
89	1	1	14.73270	6180.00	RR	LRU
90	0	1	0.90290	2553.00	RR	LRU
91	0	2	2.03803	2292.00	RR	LRU
92	0	1	0.68899	831.20	RR	LRU
93	1	1	2.02602	641.69	RR	LRU
94	0	1	2.00198	1421.00	RR	LRU
95	1	1	3.32332	1656.00	RR	LRU
96	1	1	2.36235	165.00	RR	LRU
97	0	1	1.21921	1656.00	RR	LRU
98	1	1	1.10710	591.20	RR	LRU
99	0	1	11.56753	21853.00	RR	LRU
100	0	2	2.11410	1275.00	RR	LRU
101	0	1	1.53553	2086.00	RR	LRU
102	1	1	0.65666	253.60	RR	LRU
103	2	1	8.45844	605.64	RR	LRU
104	1	1	3.77575	1308.00	RR	LRU
105	2	2	7.27926	457.30	RR	LRU
106	1	1	4.73071	1224.00	RR	LRU
107	0	1	1.86586	1359.00	RR	LRU
108	4	1	9.28536	4000.00	RRR	LRU
109	1	1	0.64124	2560.00	RRR	LRU
110	0	1	4.82480	12389.00	RR	LRU
111	1	2	1.35739	430.50	RR	LRU
112	0	1	1.11111	1082.00	RR	LRU
113	0	1	1.33733	892.00	RR	LRU
114	0	2	1.04504	1429.00	RR	LRU
115	1	2	2.53893	1400.00	RR	LRU
116	1	2	1.06506	250.00	RR	LRU
117	1	1	1.76776	471.70	RR	LRU
118	2	1	8.48096	719.70	RR	LRU
119	0	1	1.40940	1686.00	RR	LRU
120	1	1	4.35234	1449.00	RR	LRU
121	1	2	1.59758	570.50	RR	LRU
122	1	1	2.22421	1278.00	RR	LRU
123	1	1	1.94594	886.00	RR	LRU
124	1	3	1.54955	659.40	RR	LRU
125	0	1	1.50349	1119.00	RR	LRU
126	2	2	8.41240	702.60	RR	LRU
127	1	1	17.86978	5954.00	RR	LRU
128	1	1	9.19317	2380.00	RR	LRU
129	1	1	1.56956	650.00	RR	LRU
130	0	1	0.18018	480.00	RR	LRU
131	0	1	1.28328	840.00	RR	LRU

132	1	1	10.92090	5000.00	RR	LRU
133	1	1	1.86566	1010.00	RR	LRU
134	1	1	1.81581	718.75	RR	LRU
135	2	2	4.66865	275.00	RR	LRU
136	1	1	1.34534	129.00	RR	LRU
137	1	1	12.51249	3097.00	RR	LRU
138	1	2	3.53951	656.00	RR	LRU
139	1	1	12.20018	3571.00	RR	LRU
140	1	1	8.40638	2760.00	RR	LRU
141	2	1	18.11806	2711.00	RR	LRU
142	1	1	25.42134	9674.00	RR	LRU
143	0	1	1.41341	3275.00	RR	LRU
144	2	2	15.30728	1615.00	RR	LRU
145	1	1	0.67467	236.10	RR	LRU
146	0	1	0.80280	1674.00	RR	LRU
147	1	2	3.45949	436.00	RR	LRU
148	0	1	20.75662	35511.00	RR	LRU
149	0	1	2.25624	1472.00	RR	LRU
150	0	1	3.62762	2730.00	RR	LRU
151	0	1	17.40334	17694.00	RR	LRU
152	0	1	5.96394	3514.00	RR	LRU
153	0	1	3.59359	3223.00	RR	LRU
154	1	1	3.48947	1183.00	RR	LRU
155	0	1	1.58157	1471.00	RR	LRU
156	2	1	1.69859	1584.00	RRR	LRU
157	2	1	3.56697	41566.00	RRR	LRU
158	1	1	0.23000	41566.00	RRR	LRU
159	3	1	1.74018	3668.00	RRR	LRU
160	2	1	1.01338	1200.00	RRR	LRU
161	1	1	1.73880	13000.00	RRR	LRU
162	1	1	1.18174	13000.00	RRR	LRU
163	17	1	16.33919	51886.00	RRR	LRU
164	6	1	6.78560	39000.00	RRR	LRU
165	8	1	8.50310	60485.00	RRR	LRU
166	9	1	9.07718	40102.00	RRR	LRU
167	4	1	2.91548	5551.00	RRR	LRU
168	7	1	3.87596	2200.00	RRR	LRU
169	1	1	1.12884	5352.00	RRR	LRU
170	3	1	1.91958	4687.00	RRR	LRU
171	11	1	11.02481	45581.00	RRR	LRU
172	1	1	0.57387	29011.00	RRR	LRU
173	6	1	4.60046	13721.00	RRR	LRU
174	5	1	4.38978	11520.00	RRR	LRU
175	5	1	12.59189	32182.00	RRR	LRU
176	1	1	3.29566	566.50	RRR	LRU
177	2	1	0.71001	1200.00	RRR	LRU
178	1	1	2.89770	4496.00	RRR	LRU
179	3	1	2.58540	21265.00	RRR	LRU
180	12	1	11.13084	40606.00	RRR	LRU
181	0	1	1.94994	1715.00	RR	SRU
182	0	1	0.62662	13245.00	RR	SRU
183	0	1	1.42743	13260.00	RR	SRU
184	0	2	1.23120	3539.70	RR	SRU
185	0	1	0.08609	6761.00	RR	SRU
186	0	1	3.57858	5300.00	RR	SRU
187	0	1	0.62662	7300.00	RR	SRU
188	0	1	0.71471	2162.00	RR	SRU
189	0	1	1.00100	2554.00	RR	SRU
190	0	1	0.45846	2100.00	RR	SRU
191	0	1	1.13914	2550.00	RR	SRU
192	0	1	3.79275	1250.00	RR	SRU
193	0	1	4.25224	2500.00	RR	SRU
194	0	1	5.38737	2200.00	RR	SRU
195	0	1	1.13713	839.30	RR	SRU
196	0	1	1.28728	758.00	RR	SRU
197	0	1	0.80080	4166.00	RR	SRU

198	0	1	0.21621	891.90	RR	SRU
199	0	2	3.76374	1246.00	RR	SRU
200	0	1	15.19716	4000.00	RR	SRU
201	0	1	1.01501	800.00	RR	SRU
202	0	2	3.22721	640.00	RR	SRU
203	0	1	3.32931	535.00	RR	SRU
204	0	1	5.30529	630.60	RR	SRU
205	0	1	1.52151	415.90	RR	SRU
206	0	1	1.43944	547.90	RR	SRU
207	0	1	3.24923	574.00	RR	SRU
208	0	2	4.51689	1174.00	RR	SRU
209	0	1	4.77476	1123.00	RR	SRU
210	0	1	0.92092	526.30	RR	SRU
211	0	1	6.83081	1674.00	RR	SRU
212	0	1	1.68768	434.00	RR	SRU
213	0	1	0.63463	668.50	RR	SRU
214	0	1	1.60760	190.60	RR	SRU
215	0	1	1.30529	345.00	RR	SRU
216	0	1	1.00100	772.50	RR	SRU
217	0	1	2.07005	2450.00	RR	SRU
218	0	1	4.95294	3846.00	RR	SRU
219	0	1	0.55095	3710.00	RR	SRU
220	0	1	2.99497	1654.00	RR	SRU
221	0	1	4.68866	1693.00	RR	SRU
222	0	1	3.56555	2563.00	RR	SRU
223	0	2	0.06807	556.10	RR	SRU
224	0	2	0.13614	685.00	RR	SRU
225	0	1	9.82379	6026.00	RR	SRU
226	0	1	1.36536	1972.00	RR	SRU
227	0	1	10.01598	3250.00	RR	SRU
228	0	1	2.17817	571.90	RR	SRU
229	0	1	2.17817	2208.00	RR	SRU
230	0	1	2.17817	217.60	RR	SRU
231	0	1	2.00198	2052.00	RR	SRU
232	0	1	1.25925	292.50	RR	SRU
233	0	1	3.90389	490.30	RR	SRU
234	0	1	1.80179	321.40	RR	SRU
235	0	1	3.88586	327.50	RR	SRU
236	0	1	2.20219	734.40	RR	SRU
237	0	1	2.55293	322.40	RR	SRU
238	0	1	2.85885	1740.00	RR	SRU
239	0	1	3.00499	854.50	RR	SRU
240	0	1	15.17715	8310.00	RR	SRU
241	0	1	1.80179	348.10	RR	SRU
242	0	1	2.12812	381.10	RR	SRU
243	0	1	0.00399	3033.00	RR	SRU
244	0	1	2.66665	473.80	RR	SRU
245	0	1	1.53953	922.40	RR	SRU
246	0	4	5.00498	690.70	RR	SRU
247	0	1	1.42141	1639.00	RR	SRU
248	0	8	3.63563	372.90	RR	SRU
249	0	15	7.29728	653.00	RR	SRU
250	0	1	2.27026	1114.00	RR	SRU
251	0	1	1.66766	736.70	RR	SRU
252	0	1	0.94094	1484.00	RR	SRU
253	0	1	10.13811	7694.00	RR	SRU
254	0	1	3.65564	794.30	RR	SRU