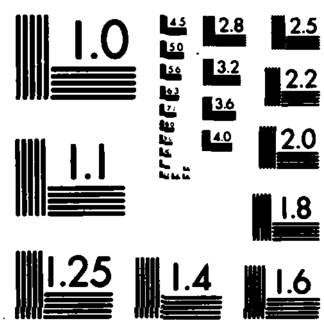
AD-A138 314 WHOLESALE PROVISIONING MODELS: MODEL OPTIMIZATION(U) 1/1
NAVAL POSTGRADUATE SCHOOL MONTEREY CA G T HOWARD
OCT 83 NPS55-83-028

UNCLASSIFIED F/G 15/5 NL

END
DATE FILMED
3 84
DTIC


1.0
4.5
5.0
2.8
2.5
3.2
2.2
3.6
4.0
2.0
1.1
1.8
1.25
1.4
1.6

2

NPS55-83-028

NAVAL POSTGRADUATE SCHOOL

Monterey, California

WHOLESALE PROVISIONING MODELS:
MODEL OPTIMIZATION

by

G. T. Howard

October 1983

Final Report for Period April 1983 - September 1983

Approved for public release; distribution unlimited.

Prepared for:

Commanding Officer
Fleet Material Support Office
Mechanicsburg, PA 17055

NAVAL POSTGRADUATE SCHOOL
Monterey, California

R. H. Shumaker
Superintendent

David A. Schrady
Provost

The work reported herein was supported with funds provided by the Navy Fleet Material Support Office, Mechanicsburg, Pennsylvania.

This report was prepared by:

G. T. Howard, Associate Professor
Department of Operations Research

Reviewed by:

Alan Washburn, Chairman
Department of Operations Research

Released by:

KNEALE T. MARSHALL
Dean of Information and Policy Sciences

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE

READ INSTRUCTIONS BEFORE COMPLETING FORM

1. REPORT NUMBER NPS55-83-028	2. GOVT ACCESSION NO.	3. RECIPIENT'S CATALOG NUMBER
4. TITLE (and Subtitle) WHOLESALE PROVISIONING MODELS: MODEL OPTIMIZATION		5. TYPE OF REPORT & PERIOD COVERED FINAL REPORT 1 April 83 - September 83
		6. PERFORMING ORG. REPORT NUMBER
7. AUTHOR(s) G. T. Howard		8. CONTRACT OR GRANT NUMBER(s)
9. PERFORMING ORGANIZATION NAME AND ADDRESS Naval Postgraduate School Monterey, CA 93943		10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS N0036782PON3901
11. CONTROLLING OFFICE NAME AND ADDRESS Commanding Officer Fleet Material Support Office Mechanicsburg, PA 17055		12. REPORT DATE October 1983
		13. NUMBER OF PAGES
14. MONITORING AGENCY NAME & ADDRESS(if different from Controlling Office)		15. SECURITY CLASS. (of this report) Unclassified
		15a. DECLASSIFICATION/DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT (of this Report)

Approved for public release; distribution unlimited.

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

18. SUPPLEMENTARY NOTES

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

Provisioning
Availability
Inventory Models
Mean Supply Response Time
Dynamic Programming

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

Wholesale provisioning models are considered. The performance measures used are: supply material availability, mean supply response time, and availability. These measures are optimized subject to a budget constraint. The performance constrained budget minimization problems are also solved for the optimal allocation. The basic method used is dynamic programming.

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE

S/N 0102-LF-014-6601

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

Abstract

Wholesale provisioning models are considered. The performance measures used are: supply material availability, mean supply response time, and availability. These measures are optimized subject to a budget constraint. The performance constrained budget minimization problems are also solved for the optimal allocation. The basic method used is dynamic programming.

I. Introduction

A. Overview

The purpose of this report is to discuss solution methods for several problems arising in inventory provisioning. Two related types of problems are considered:

1) optimization of a performance measure subject to a budget constraint
2) minimization of cost subject to a constraint on performance.

The performance measures considered are Supply Material Availability (SMA), Mean Supply Response Time (MSRT), and "Pseudo-Availability" (PA). The basic approach to these problems is dynamic programming.

First the problems are formulated and solved using one recursive technique. Then a more efficient recursion is presented and discussed in the context of maximizing MSRT subject to a budget constraint.

These same budget constrained problems are formulated, discussed and solved using a marginal analysis approach in [2] and [3]. That method, although fast, does not guarantee that optimal solutions are obtained. This report provides a method of obtaining optimal solutions and thus provides a means of evaluating heuristic methods. In addition, this report shows how the performance constrained budget minimization problems can be solved directly. By contrast, reference [2] addresses this problem using generalized Lagrange Multipliers or by solving the budget constrained performance problem repeatedly for various budget levels.

The second recursion presented here is considerably faster than the first and it also guarantees an optimal solution in some cases. It is competitive in speed with the marginal analysis method for small and medium-sized problems, but it is inefficient for problems with a large number of items or large budget values. Its virtue lies in its ability to get exact solutions to medium-sized problems.

B. Problem Formulations, budget constrained

This report considers three specific budget constrained optimization problems arising in inventory provisioning. These problems are formulated and discussed in detail elsewhere [2] and will be stated here without extensive explanation.

Following [2] we let

n = the total number of items considered for provisioning

C_i = the unit cost of item i

E_i = the essentiality code for item type i

λ_i = the demand rate for item i

T_i = the procurement leadtime for item i

S_i = the number of items of type i provided (the decision variables)

$Z_i(S_i)$ = the performance measure for item i when S_i units are stocked

$D_i(S_i) = Z_i(S_{i+1}) - Z_i(S_i)$

$p_i(x_i)$ = probability that demand for item i is x_i during the provisioning interval

$P_i(x_i)$ = cumulative probability of x_i or fewer demands during a provisioning interval

$MTTR_i$ = mean time to repair or replace item i

$MTBF_i$ = mean time between failures = $1/\lambda_i$

$MSRT_i(S_i)$ = mean supply response time when S_i units are stocked.

The three budget constrained problems considered are:

a1) maximize Supply Material Availability (SMA), defined as

$$SMA(S_1,\ldots,S_n) = \sum_{i=1}^{n} E_i \lambda_i T_i Z_i^{(1)}(S_i) / \sum_{i=1}^{n} E_i \lambda_i T_i$$

where

$$Z_i^{(1)}(S_i) = (1 - p_i(S_i)) + (S_i - \lambda_i T_i)(1 - P_i(S_i))/\lambda_i T_i .$$

b1) minimize Mean Supply Response Time (MSRT), defined as

$$MSRT(S_1,\ldots,S_n) = \sum_{i=1}^{n} E_i \lambda_i T_i Z_i^{(2)}(S_i) / \sum_{i=1}^{n} E_i \lambda_i T_i$$

where

$$Z_i^{(2)}(S_i) = (1 - P_i(S_i))(T_i/2 - S_i/\lambda_i + \frac{S_i(S_i+1)}{2\lambda_i^2 T_i}) + p_i(S_i)(\lambda_i T_i - S_i)/2\lambda_i .$$

c1) maximize Pseudo-Availability (PA), defined as

$$PA(S_1,\ldots,S_n) = \prod_{i=1}^{n} Z_i^{(3)}(S_i)$$

where

$$Z_i^{(3)}(S\ \ ) = MTBF_i/(MTBF_i + MTTR_i + Z_i^{(2)}(S_i)) .$$

In each of the budget constrained problems above the objective is to be optimized by selection of $S_i,\ldots,S_n$ subject to the constraints

$$\sum_{i=1}^{n} C_i S_i \leq B , \qquad S_i \geq 0 \text{ integer}$$

where B is the specified budget level.

C. Problem Formulations, Performance Constrained

In addition to the three problems just stated, we consider three related problems in which the cost is to be minimized subject to a constraint on performance.

a2) $$\min \sum_{i=1}^{n} C_i S_i$$

$$\text{s.t.} \quad SMA(S_1,\ldots,S_n) \geq SMA$$

$$S_i \geq 0 \text{ integer.}$$

b2) $$\min \sum_{i=1}^{n} C_i S_i$$

$$\text{s.t.} \quad MSRT(S_1,\ldots,S_n) \leq MSRT$$

$$S_i \geq 0 \text{ integer.}$$

c2) $$\min \sum_{i=1}^{n} C_i S_i$$

$$\text{s.t.} \quad PA(S_1,\ldots,S_n) \geq PA$$

$$S_i \geq 0 \text{ integer.}$$

II. Solution Method and Examples

A. Dynamic Programming Approach

The computer program used to solve these problems is DP4, a general purpose program for performing dynamic programming tabular computations. Here we will describe the general nature of that program and those elements required to tailor it for use in the problems considered in this report.

The DP4 program deals with a problem consisting of n related stages each of which is characterized as shown in figure 1.

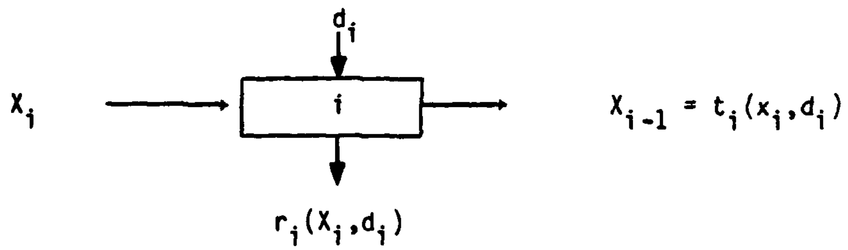


Figure 1. A single-stage decision problem.

In figure 1

X_i is the "state" variable

d_i is the decision variable

r_i is the stage return function

t_i is the stage transformation function.

In the overall problem consisting of n stages the output state variable from stage i, namely X_{i-1}, is the input to stage $i-1$. Thus, the n stage problem can be pictured as in figure 2.

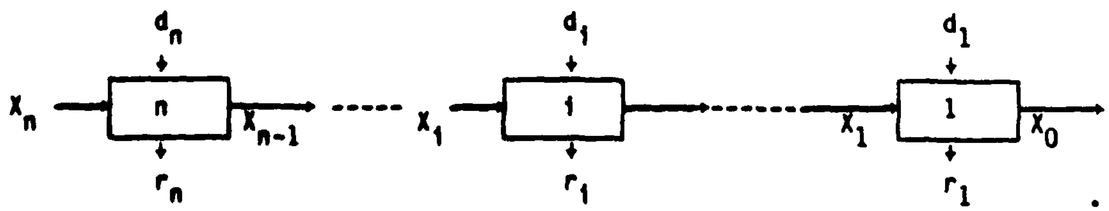


Figure 2. n-stage decision problem.

The state variable X_i can easily be understood in the context of the budget constrained problems as the amount of resource (money) remaining to be allocated to stages $i, i-1, \ldots, 1$. The stages, of course, correspond to the items in the inventory problems.

At each stage i a decision d_i must be made. The decision has two effects. First, it yields a return r_i, the performance measure associated with the current item. Second, it yields a value of X_{i-1} which serves as the input to the remainder of the decision process. The decision d_i must be made with consideration both for the immediate return r_i and the future state X_{i-1}. The overall problem is to make the series of decisions $d_n, \ldots, d_1$ to optimize some function of the individual stage returns.

In the problem (a1), where S_i is the decision variable, we can let the return functions be

$$r_i(X_i, S_i) = E_i \lambda_i T_i Z_i^{(1)}(S_i) / \sum_{i=1}^{n} E_i \lambda_i T_i \qquad i = 1, \ldots, n$$

and the stage transformation functions be

$$X_n = B$$

$$X_{i-1} = t_i(X_i, S_i) = X_i - C_i S_i \qquad i = 1, \ldots, n .$$

The overall return function is the sum of the individual return functions. Namely,

$$\text{SMA}(S_1, \ldots, S_n) = \sum_{i=1}^{n} r_i(X_i, S_i) .$$

The object remains to select $S_1, \ldots, S_n$ to optimize this return.

We let $f_i(X_i)$ = the optimal total return from stages $i, i-1, \ldots, 1$

given that we enter stage i with state variable X_i.

Then we can write the recursive equations for this optimization as

$$f_i(x_i) = \max_{S_i}\{r_i(X_i, s_i) + f_{i-1}(X_{i-1})\}$$

$$\text{s.t.} \quad X_{i-1} = X_i - C_i S_i$$

$$\text{and} \quad 0 \leq S_i \leq X_i/C_i$$

$$\text{and} \quad S_i = \text{integer}$$

for $i = 2, \ldots, n$.

The equation at stage 1 is

$$f_1(X_1) = \max_{S_1} r_1(X_1, S_1)$$

$$\text{s.t.} \quad 0 \leq S_1 \leq X_1/C_1$$

$$\text{and} \quad S_1 \text{ integer} .$$

The program DP4 performs this optimization provided the user supplies the following subroutines and data.

Required subroutines

1. STGRET - this subroutine defines the function $r_i(X_i, d_i)$
2. TRANFM - defines the stage transformation function $t_i(X_i, d_i)$
3. DLIMIT - defines the range of decision values d_i which can be considered for the particular value of X_i under consideration
4. STORE - allows the input of constants to be used in the other subroutines.

Required Data

1. n - the number of stages

2. For each stage i

XLOW - the lowest value of X_i to consider

XHIGH - the highest value of X_i to consider

DELX - the increment for X_i

XMODE - tells whether to maximize or minimize

XSTAGE - tells how this stage return relates to lower numbered stage returns (sum, product).

The methodology is essentially the same for the performance constrained problems. There the return functions $r_i(X_i, d_i) = c_i d_i$. The state variable X_i is interpreted as the portion of the performance measure to be attributed to stages $1,\ldots,i$. The stage transformation functions in problem (a2) and (b2) are

$$X_{i-1} = X_i - Z_i(S_i) .$$

In problem (c2) the stage transformation is

$$X_{i-1} = X_i / Z^{(3)}(S_i) .$$

The program DP4 and the subroutines are shown in Appendix A. The subroutines are written to solve any of the problems (a1), (b1), (c1) or (a2), (b2), (c2). Thus they involve complications not needed for solving just one of these problems.

B. Examples

Several example problems were solved to illustrate the approach discussed here. All of the problems involved $n = 10$ items and all used the data shown in table 1.

Data

n	λ	Time	Cost	MTTR	E
1	5.0	1.0	1.0	.0137	1.0
2	2.0	1.0	2.0	.0274	1.0
3	3.0	1.0	5.0	.0137	1.0
4	5.0	1.0	10.0	.0822	1.0
5	10.0	1.0	20.0	.0274	1.0
6	25.0	1.0	5.0	.0027	1.0
7	1.0	1.0	1.0	.0054	1.0
8	1.0	1.0	100.0	.0411	3.0
9	0.5	1.0	50.0	.0082	1.0
10	2.0	1.0	10.0	.1370	3.0

Table 1: Data for Examples

1. The budget constrained problems.

Tables 2, 3, and 4 summarize the solutions for the example problems (a1), (b1), and (c1). These are the budget constrained problems.

B = Budget = max SMA =	300 .750654	295 .746466	290 .741160	285 .736563	280 .732964
decision S_1 =	4	4	4	4	4
S_2 =	4	4	4	4	4
S_3 =	4	4	4	4	4
S_4 =	5	5	5	4	5
S_5 =	2	2	2	2	1
S_6 =	29	28	27	28	29
S_7 =	3	3	3	3	3
S_8 =	0	0	0	0	0
S_9 =	0	0	0	0	0
S_{10} =	3	3	3	3	3

Table 2: Solutions to example problem (a1)

B =	300	295	290	285	280
min MSRT =	.0838231	.0859625	.0878466	.0900560	.0927486
decision S_1	2	2	2	2	2
S_2	3	3	3	3	3
S_3	3	4	3	3	3
S_4	4	4	4	4	4
S_5	5	4	4	4	4
S_6	21	23	23	22	21
S_7	2	2	2	2	2
S_8	0	0	0	0	0
S_9	0	0	0	0	0
S_{10}	3	3	3	3	3

Table 3. Solutions to example problem (b1)

B = max PA =	300 .0403727	295 .0387421	290 .037005	285 .0355446	280 .0341911
decision S_1	3	3	3	3	3
S_2	4	4	4	4	4
S_3	5	4	5	5	5
S_4	5	5	5	5	5
S_5	3	3	2	2	2
S_6	26	26	26	27	26
S_7	4	4	4	4	4
S_8	0	0	0	0	0
S_9	0	0	0	0	0
S_{10}	2	2	3	2	2

Table 4. Solutions to example problem (c1)

2. The performance constrained problems.

The related performance constrained problems were also solved for illustration. For example, the problem (a2) was solved using the data from Table 1 with the restriction that SMA $\geq$ .732964 . The solution to that problem is the same as the solution shown in the last column of Table 2 since the value .732964 is the (largest) value of SMA corresponding to a budget of 280.

III. Modifications to Basic Method

A. An Alternative Recursion

An alternative and more efficient approach is available for the budget constrained problems.

The approach is based on a different recursion from that used in Section 2 and is very similar to an approach used to solve the "cargo loading problem". See for example Dreyfus [1].

The cargo loading problem, stated as a maximization, is:

$$\max \sum_{j=1}^{N} v_j d_j$$

$$\text{s.t.} \quad \sum_j c_j d_j \le B$$

$$d_j \ge 0 \text{ integer .}$$

Although many methods are available for solving this problem, the one of interest to us is based on the following recursion

$$f(b) = \max_{j \in \{1,\dots,N\}} \{v_j + f(b-c_j)\}$$

where $f(b)$ is the optimal total return that can be obtained when a budget of b is available.

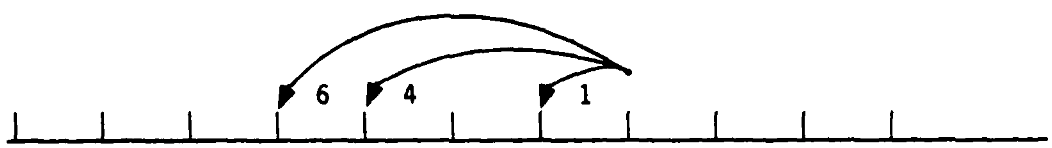
To illustrate this method consider the data in Table 5.

i	1	2	3 = N
v_i	1	4	6
c_i	1	3	4

B = 10

Table 5. Data for example using alternative recursion

The computation proceeds with increasing values of b until $b = B$ is reached. The process can be viewed as shown in figure 3 where a template representing the available items is placed over the budget value of current interest. The template points back to previously determined optimal solution. Each of these previous solution is considered for updating by including one more item of the type inducted by the template. The updated solutions are compared and the best selected as the solution for the current value of b. The illustration shows the template at the budget value of 7. The optimal solutions for $b = 0,1,\ldots,6$ have already been computed. The comparison at $B = 7$ is among the solution at 6 with an additional item 1 for a total return of 9, the solution at 4 with an additional item of type 2 for a total return of 10, and the solution at 3 with an additional item of type 3 for a return of 10. Either of the last two is chosen and recorded as an optimal solution at 7.



$b =$	0	1	2	3	4	5	6	7	8	9	10
$d_1(b) =$	0	1	2	0	0	1	2 *	0			
$d_2(b) =$	0	0	0	1	0	0	0	1			
$d_3(b) =$	0	0	0	0	1	1	1	1			
$f(B) =$	0	1	2	4	6	7	8	10			

*an alternative optimal solution at $b = 6$ is $d_1 = 0$ $d_2 = 2$ $d_3 = 0$

Figure 3: Illustration of solution method using the alternative recursion.

A very simple modification of this procedure can be used to solve the problems of the type discussed in this report. We consider

$$\max \sum_{j=1}^{n} r_j(d_j)$$

$$\text{s.t.} \sum_{j=1}^{n} c_j \cdot d_j \leq B$$

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

where the return functions $r_j(d_j)$ are concave. The procedure below is not guaranteed to give the optimal solution for all $r_j(d_j)$ but is guaranteed if the $r_j(d_j)$ are points on a concave function.

In this case we will represent the return functions $r_j(d_j)$ as the sum of the marginal values of additional items of type j.

$$r_j(d) = + \sum_{i=1}^{d} m_j(i)$$

Thus $r_j(2) = m_j(0) + m_j(1) + m_j(2)$. These marginal values form a sequence with the properties that

$$m_j(i) > 0 \qquad i$$

and

$$m_j(i) > m_j(k) \qquad i < k .$$

The same algorithm as before was applied with the modification that the value term v_j, which was formerly constant, is replaced by $m_j(i)$ for the appropriate value of i. The program which implements this algorithm was called RECUR.

It should be noted that the discussion above treats the constraint as an inequality, but the function $f(b)$ in this section is computed for the constraint

$$\sum_{j=1}^{N} c_j x_j = b .$$

For this reason we may have in a maximization problem

$$f(b_1) > f(b_2)$$

although $b_1 < b_2$. That is, if we require the equality to be met exactly, it is not necessarily true that a larger budget is better. The program prints the values of $f(b)$ for several values of b so the optimal value of $b \leq B$ can be found visually.

B. Minimum Orders

1) RECMOD

A modification was made to the program RECUR to permit the user to specify minimum packaging quantities of each item. That is, item i is assumed to be packaged with q_i items per package. The provisioning can select only whole packages of each item. This modification resulted in the program RECMOD given in the Appendix.

2) Example

To illustrate the RECMOD program, consider the example problem (a2) solved previously. The optimal solution for budget $B = 300$ is repeated in Table 2 for the case in which all $q_i = 1$.

n	1	2	3	4	5	6	7	8	9	10	Objective
q_i	1	1	1	1	1	1	1	1	1	1	
d_i	2	3	3	4	5	21	2	0	0	3	.0838231
q_i	2	1	1	5	1	1	1	3	1	1	
d_i	2	3	3	5	4	23	2	0	0	3	.0847410

Table 6. Solution to Example using RECMOD.

The optimal solution is also shown for a modified problem in which not all q_i are equal to 1 .

C. Discussion

The modified recursion just discussed has been implemented for the problem (a2) which is to minimize MSRT subject to a budget constraint. The method can also be applied to the other budget constrained problems, but this has not yet been done. All that is required is to modify the program to compute SMA or PA instead of MSRT and to maximize instead of minimize.

There is a difficulty in extending this method to problems in which the item costs are arbitrary values. The method is very effective when the costs are all integer and can be scaled so that the smallest cost is 1 . If arbitrary costs are allowed, the algorithm can become ineffective for all except small values of B . Consider for example the costs of $1.00, $1.21, $1.27 for three items. Let the budget be $25.00. The problem could be solved by scaling the costs to be 100, 120, and 125 and the budget to be 2500, but then too many values of b must be considered when many of them are not possible. Alternatively the algorithm can step to the "next possible value" and will consider the folowing sequence of values

b = 100, 121, 127, 200, 221, 227, 242, 248, 254, 300, 321, 327,... .

As the process continues, depending on the relative values of the costs, the sequence becomes more dense and may eventually include all possible values of b . This is ineffective and cumbersome for large values of B .

It is also not possible to apply the modified recursion for the performance constrained problems. On the other hand, the budget constrained problems are solved very rapidly and the relationship between cost and performance can easily be determined from solving the budget constrained problem using the first recursion. In fact, if MSRT is minimized for a budget of B , the solution is actually obtained for all values of b up to and including B . Those results reveal the relationship between performance and budget.

```
FILE: DPFILE    EXEC      A1   NAVAL POSTGRADUATE SCHOOL

FILEDEF 07 DISK (PERM
FILEDEF 08 DISK RUSS10 DATA (PERM
FILEDEF 06 TERM (LRECL 133 RECFM FB PERM
FILEDEF 02 DISK (PERM
```

```
FILE: FILE      FT07F001 A1   NAVAL POSTGRADUATE SCHOOL

   10        NEW
    1    10          0.          300.         1.    MIN    SUM  2
            300.                 300.
   10        OLD
            295.                 295.
   10        OLD
            290.                 290.
   10        OLD
            285.                 285.
   10        OLD
            280.                 280.
   10        OLD
            275.                 275.
```

```
FILE: RUSS10    DATA      A1   NAVAL POSTGRADUATE SCHOOL

  2   0
         1.          1.          1.          1.       .0137
         2.          2.          1.          1.       .0274
         3.          5.          1.          1.       .0137
         5.         10.          1.          1.       .0822
        10.         20.          1.          1.       .0274
        25.          5.          1.          1.       .0027
         1.          1.          1.          1.       .0054
         1.        100.          3.          1.       .0411
         .5         50.          1.          1.       .0082
         2.         10.          3.          1.       .1370
   10   300(OB,CON)0COST 1SMA 2MSRT 3AVAIL - DATA ACEBD=LAM,COST,ESS,T,M??
    2     1     1     5     1    1    1    3     1    1
 FOR RECUR THE LAST TWO LINE MUST BE PLACED FIRST INSTEAD OF THE
 CURRENT LINE.  IT IS FOR DP4
```

```
C
C      DATA DESCRIPTION FOR CARD NUMBER 1
C      --------.-----------.-----------.-------------------------------------
C      COLUMNS . JUSTIFY   . VARIBLE   . MEANING
C              .           . NAME      .
C      --------.-----------.-----------.-------------------------------------
C      1 - 5   . COL 5     . NMAX      . THE NUMBER OF STAGES IN THE PROBLEM
C      --------.-----------.-----------.-------------------------------------
C      6 - 10  . COL 10    . MTAPE     . LOGICAL TAPE NO OF THE MAIN TAPE
C              .           .           . IF THIS IS LEFT BLANK, THE COMPUTER
C              .           .           . WILL USE STANDARD SCRATCH TAPE 3
C      --------.-----------.-----------.-------------------------------------
C      11 - 15 . COL 15    . SOLVE     . =  OLD    IF THE OPTIMAL DECISION
C              .           .           .           FUNCTIONS HAVE ALREADY BEEN
C              .           .           .           CALCULATED AND ARE LOADED
C              .           .           .           ON LOGICAL TAPE MTAPE
C              .           .           .-------------------------------------
C              .           .           . =  NEW    IF THE OPTIMAL DECISION
C              .           .           .           FUNCTIONS MUST BE
C              .           .           .           CALCULATED AND STORED ON
C              .           .           .           LOGICAL TAPE MTAPE BEFORE
C              .           .           .           SOLVING THE PROBLEM
C      --------.-----------.-----------.-------------------------------------
C
C
C
C      DATA DESCRIPTION FOR THE NEXT GROUP OF CARDS
C      IF SOLVE = NEW  ON CARD NUMBER 1, THEN EACH OF THE NMAX STAGES
C      MUST BE DECRIBED BY THE CARDS DISCUSSED BELOW. YOU CAN USE 1 CARD
C      PER STAGE OR YOU CAN MAKE 1 CARD DESCRIBE MANY ADJACENT STAGES IF
C      THEY ARE SIMILAR. THE STAGES MUST BE DESCRIBED IN NUMERICAL ORDER
C      STARTING WITH STAGE 1 (I.E. STAGE 1, THEN 2, ..., THEN NMAX)
C      OMIT THIS PACK OF CARDS WHICH DESCRIBE THE STAGES IF SOLVE = OLD
C      --------.-----------.-----------.-------------------------------------
C      COLUMNS . JUSTIFY   . VARIBLE   . MEANING
C              .           . NAME      .
C      --------.-----------.-----------.-------------------------------------
C      1 - 5   . COL 5     . NSTAGE    . LOWEST NUMBERED STAGE FOR WHICH
C              .           .           . THIS CARD APPLIES
C      --------.-----------.-----------.-------------------------------------
C      6 - 10  . COL 10    . NDITTO    . HIGHEST NUMBERED STAGE FOR WHICH
C              .           .           . THIS CARD APPLIES. IF THESE COLUMNS
C              .           .           . ARE LEFT BLANK, THEN NDITTO WILL BE
C              .           .           . TAKEN AS EQUAL TO NSTAGE
C      --------.-----------.-----------.-------------------------------------
C      11 - 20 . ANY       . XLOW      . LOWEST VALUE OF XN FOR STAGE NSTAGE
C      --------.-----------.-----------.-------------------------------------
C      21 - 30 . ANY       . XHIGH     . HIGHEST VALUE OF XN FOR STAGE NSTAGE
C      --------.-----------.-----------.-------------------------------------
C      31 - 40 . ANY       . DELX      . INCREMENT IN XN FOR STAGE NSTAGE
C      --------.-----------.-----------.-------------------------------------
C      41 - 46 . COL 46    . XMODE     . =    MIN  IF STAGE NSTAGE IS TO BE
C              .           .           .           MINIMIZED
C              .           .           .-------------------------------------
C              .           .           . =    MAX  IF STAGE NSTAGE IS TO BE
C              .           .           .           MAXIMIZED
C      --------.-----------.-----------.-------------------------------------
C      47 - 52 . COL 52    . XSTAGE    . =    SUM  IF THE COMPOSITION
C              .           .           .           OPERATOR BETWEEN STAGES
C              .           .           .           NSTAGE AND NSTAGE-1 IS
C              .           .           .           ADDITION
C              .           .           .-------------------------------------
C              .           .           . =   MULT  IF THE COMPOSITION
C              .           .           .           OPERATOR BETWEEN STAGES
C              .           .           .           NSTAGE AND NSTAGE-1 IS
C              .           .           .           MULTIPLICATION
C              .           .           .-------------------------------------
C              .           .           . =MINMAX   IF THE COMPOSITION
C              .           .           .           OPERATOR BETWEEN STAGES
C              .           .           .           NSTAGE AND NSTAGE-1 IS TO
C              .           .           .           MINIMIZE THE MAXIMUM
C              .           .           .           INDIVIDUAL STAGE RETURN
```

```
C
C                .          .         .---------------------------------------
C                .          .         . =MAXMIN   IF THE COMPOSITION
C                .          .         .           OPERATOR BETWEEN STAGES
C                .          .         .           NSTAGE AND NSTAGE-1 IS TO
C                .          .         .           MAXIMIZE THE MINIMUM
C                .          .         .           INDIVIDUAL STAGE RETURN
C                .          .         .---------------------------------------
C                .          .         . =         THIS VARIABLE IS NOT USED
C                .          .         .           WHEN NSTAGE=1, THEREFORE
C                .          .         .           COLUMNS 47 TO 52 ARE
C                .          .         .           IGNORED WHEN NSTAGE=1 AND
C                .          .         .           MAY BE LEFT BLANK
C                .          .         .           HOWEVER IF NDITTO IS
C                .          .         .           GREATER THAN 1, THEN XSTAGE
C                .          .         .           MUST BE SPECIFIED IN
C                .          .         .           ACCORDANCE WITH THE
C                .          .         .           COMPOSITION OPERATOR FOR THE
C                .          .         .           REST OF STAGES DESCRIBED ON
C                .          .         .           THIS CARD
C       --------.----------.----------.---------------------------------------
C       53 - 55 . COL 55   . MODES    . = 1       IF THE FIBONACCI SEARCH IS
C                .          .         .           TO BE USED
C                .          .         .---------------------------------------
C                .          .         . = 2       IF THE FIBONACCI SEARCH IS
C                .          .         .           NOT TO BE USED
C       --------.----------.----------.---------------------------------------
C
C
C
C       DATA DESCRIPTION OF LAST CARD FOR PROBLEM
C       THE OPTIMUM DECISIONS ARE TO BE PRINTED OUT FOR A PROCESS WITH
C       NMAX STAGES, FURTHER, FN(XN) IS TO BE OPTIMIZED WITH RESPECT TO XN
C       WHEN THE PROGRAM COMPLETES THE REQUESTED PRINTOUT IT WILL RETURN
C       TO THE BEGINNING AND START A NEW PROBLEM IF THERE IS DATA, IF NO
C       DATA, IT WILL EXIT
C       --------.----------.----------.---------------------------------------
C       COLUMNS . JUSTIFY  . VARIBLE  . MEANING
C               .          . NAME     .
C       --------.----------.----------.---------------------------------------
C       1 - 15  . ANY      . XN1      . AT STAGE NMAX, OPTIMIZE FN(XN) FOR
C               .          .          . XN NOT LESS THAN XN1
C       --------.----------.----------.---------------------------------------
C       16 - 30 . ANY      . XN2      . AT STAGE NMAX, OPTIMIZE FN(XN) FOR
C               .          .          . XN NOT GREATER THAN XN2
C       --------.----------.----------.---------------------------------------
C
C
C
C       DESCRIPTION OF THE SUBROUTINE TRANFM
C       THE MAIN PROGRAM TRANSMITS XN,DN, AND NUMBER (THE SEQUENCE NUMBER
C       OF THE PROBLEM) TO THE SUBROUTINE AND THE SUBROUTINE CALCULATES
C       YN (THE OUTPUT OF THE STAGE I.E. YN=XN-1)
C
C
C
C       DESCRIPTION OF THE SUBROUTINE STGRET
C       THE MAIN PROGRAM TRANSMITS XN,DN,YN, AND NUMBER TO THE SUBROUTINE,
C       AND THE SUBROUTINE CALCULATES RN (THE STAGE RETURN)
C
C
C
C       DESCRIPTION OF SUBROUTINE STORE
C       THE MAIN PROGRAM CALLS THE SUBROUTINE RIGHT AFTER READING THE
C       FIRST DATA CARD AND DEFINING NUMBER. THE PURPOSE OF THIS
C       SUBROUTINE IS TO ALLOW THE STORING OF CONSTANTS IN COMMON STORAGE
C       FOR THE POSSIBLE USE OF SUBROUTINES STGRET AND TRANFM
C
C       DATA WHICH WILL BE READ FROM DATA CARDS AT OBJECT TIME BY
C       THE SUBROUTINE STORE MUST BE INSERTED BETWEEN THE FIRST DATA CARD
C       AND THE GROUP OF CARDS WHICH DESCRIBE THE INDIVIDUAL STAGES
C
C
```

```
C
C     DESCRIPTION OF SUBROUTINE DLIMIT
C     THE MAIN PROGRAM TRANSMITS N,XN, AND NUMBER TO THE SUBROUTINE,
C     AND THE SUBROUTINE CALCULATES DLOW, DHIGH, AND DELD
C     WHERE DLOW = LOWEST VALUE OF DN FOR THAT PARTICULAR VALUE OF N AND
C                  XN
C           DHIGH = HIGHEST VALUE OF DN  FOR THAT PARTICULAR VALUE OF N
C                  AND XN
C            DELD = INCREMENT IN DN FOR THAT PARTICULAR VALUE OF N AND XN
C     IF ANY OF THESE VARIBLES DO NOT CHANGE, THEY CAN BE SET IN
C     SUBROUTINE STORE AND NOT MENTIONED IN SUBROUTINE DLIMIT
C
C
C
C     IF ANY SUBROUTINE SETS NUFF TO -1  THEN THE PROGRAM IGNORES THE
C     VALUE OF DN BEING PROCESSED AT THAT MOMENT, AND IGNORES ALL LARGER
C     VALUES OF DN FOR THAT PARTICULAR VALUE OF XN AT THAT PARTICULAR
C     STAGE--THIS FEATURE IS DISABLED WHEN USING THE FIBONACCI SEARCH
C
C
C
C     THIS PROGRAM USES TAPES 3 AND 4 FOR SCRATCH
C     NSS2 ZERO FOR ENTIRE PROBLEM SOLVING AND OUTPUT
C     NSS2 NON-ZERO FOR OUTPUT FROM PREVIOUS PROBLEM
      DIMENSION FN(10001),FNM1(10001),DNOFXN(10001)
      REAL * 8 HOLMIN/'   MIN'/, HOLOLD/'   OLD'/
      REAL * 8 XMODE, XSTAGE, SOLVE
      REAL * 8 TAB(4)
      COMMON /CON/TAB
      COMMON XN,N,DN,YN,RN,SLOW,XHIGH,DELX,DLOW,DHIGH,DELD,NUMBER
      COMMON YLOW,YHIGH,DELY,NUFF,FN,JTOP,NNFLAG,NMAX,XLAM
      COMMON A,B,C,D,E,SUMELT,KODE1,KODE2
      EQUIVALENCE (FN(1),FNM1(1),DNOFXN(1))
      WRITE(6,1000) HOLMIN, HOLOLD, (TAB(I), I=1,4)
 1000 FORMAT(1X, 6A6)
      MAX=10001
      INTAPE=7
      NOUTPE=6
      CALL SEARCH(0.0,0.0,-1.0,0.0,DNBEST,BEST)
      NPAGE=0
      NUMBER=0
  240 READ(INTAPE,100,END=2000) NMAX,MTAPE, SOLVE
  100 FORMAT(2I5,A5)
      MTAP = MTAPE
      NDITTO=0
      NCALC=4
      IF(MTAPE-4) 405,501,405
  501 NCALC=3
      GO TO 172
  405 IF(MTAPE) 322,322,172
  322 MTAPE=3
  172 NTAPE=MTAPE
      REWIND MTAPE
      NUMBER=NUMBER+1
C     CALL STORE WAS MOVED FROM HERE
      IF(SOLVE-HOLOLD) 301,302,301
  302 NSS2=1
      NPAGE=NPAGE+1
      WRITE(NOUTPE,152) NPAGE
  152 FORMAT(1H1100X4HPAGEI4)
      WRITE(NOUTPE,303) MTAPE
  303 FORMAT (58H THE TABLES OF FN(XN) AND DN(XN) WHICH ARE ON LOGICAL T
     1APEI5,35H WILL BE USED TO SOLVE THIS PROBLEM)
      READ(MTAPE) NMAX
      REWIND MTAPE
      GO TO 304
  301 NSS2=0
      CALL STORE
      NPAGE=NPAGE+1
      WRITE(NOUTPE,152) NPAGE
      WRITE(NOUTPE,305) MTAPE
  305 FORMAT(89H THE COMPUTER IS TO CALCULATE TABLES OF FN(XN) AND DN(XN
```

```
     1), AND STORE THEM ON LOGICAL TAPEI5)
      REWIND NCALC
  500 WRITE(MTAPE) NMAX
  304 WRITE(NOUTPE,306) NMAX
  306 FORMAT(16H THE PROBLEM HASI5,7H STAGES)
  314 IF(NSS2) 197,109,197
  109 LINES = 6
      WRITE(NOUTPE,2006) NMAX, MTAP, SOLVE
 2006 FORMAT(/// ' DATA CARD  ', 2I7, A7)
  121 DO 123 N=1,NMAX
      NM1=N-1
      IF(NDITTO-N) 241,132,132
  241 READ(INTAPE,124) NSTAGE,NDITTO,XLOW,XHIGH,DELX,XMODE,XSTAGE,MODES
  124 FORMAT(2I5,3E10.6,2A6,I3)
      WRITE(NOUTPE,2007) NSTAGE, NDITTO, XLOW, XHIGH, DELX, XMODE,
     1             XSTAGE, MODES
 2007 FORMAT(/// ' DATA CARD  ', 2I5, 2X, 3F10.6, 2A6, I3)
      IF(LINES-41) 30,30,31
   31 NPAGE=NPAGE+1
      WRITE(NOUTPE,152) NPAGE
      LINES=1
   30 LINES = LINES + 9
      ITOP=(XHIGH-XLOW)/DELX+1.001
      IF(MODES-1) 600,601,600
  600 MODES=-1
  601 IF(NSTAGE-N) 125,245,125
  125 WRITE(NOUTPE,127)
  127 FORMAT(23H1DATA CARD OUT OF ORDER)
      GO TO 2003
  245 IF(NSTAGE-NDITTO) 246,126,126
  246 WRITE(NOUTPE,247) N,NSTAGE,NDITTO
  247 FORMAT(///46H THE DESCRIPTION WHICH APPEARS BELOW FOR STAGE,I5,17H
     1 APPLIES TO STAGE,I5,11H THRU STAGE,I5,10H INCLUSIVE)
      LINES=LINES+4
      XXMODE = XMODE - HOLMIN
      WRITE(6,1001) XXMODE
 1001 FORMAT(1X, A6)
  126 IF(XMODE-HOLMIN) 230,231,230
  231 XMODE=1.0
      WRITE(NOUTPE,232) N
  232 FORMAT(//6H STAGEI5,18H IS A MINIMIZATION)
      GO TO 520
  230 XMODE=-1.0
      WRITE(NOUTPE,234) N
  234 FORMAT(//6H STAGEI5,18H IS A MAXIMIZATION)
  520 DO 129 I=1,4
      IF(XSTAGE-TAB(I)) 129,130,129
  129 CONTINUE
  521 NNFLAG=5
      GO TO 502
  130 NNFLAG=I
  502 IF(ITOP-MAX) 316,316,112
  316 WRITE(NOUTPE,110) XLOW,XHIGH,DELX
  110 FORMAT(63H THE STATE VARIBLE IS BEING TREATED AS DISCRETE, GOING F
     1ROM XN=E14.5,7H TO XN=E14.5,13H IN STEPS OF E14.5)
      IF(MODES) 318,318,321
  318 WRITE(NOUTPE,510)
  510 FORMAT(41H TOTAL ENUMERATION IS USED FOR THIS STAGE)
      GO TO 320
  321 WRITE(NOUTPE,511)
  511 FORMAT(40H FIBONACCI SEARCH IS USED FOR THIS STAGE)
  320 GO TO (11,12,13,14,132),NNFLAG
   11 WRITE(NOUTPE,16) NM1,N
   16 FORMAT(40H THE COMPOSITION OPERATOR BETWEEN STAGESI6,4H ANDI6,12H
     1IS ADDITION)
      GO TO 132
   12 WRITE(NOUTPE,17) NM1,N
   17 FORMAT(40H THE COMPOSITION OPERATOR BETWEEN STAGESI6,4H ANDI6,18H
     1IS MULTIPLICATION)
      GO TO 132
   13 WRITE(NOUTPE,18) NM1,N
   18 FORMAT(40H THE COMPOSITION OPERATOR BETWEEN STAGESI6,4H ANDI6,34H
```

```
     1IS TO MAXIMIZE THE MINIMUM RETURN)
      GO TO 132
   14 WRITE(NOUTPE,19) NM1,N
   19 FORMAT(40H THE COMPOSITION OPERATOR BETWEEN STAGESI6,4H ANDI6,34H
     1IS TO MINIMIZE THE MAXIMUM RETURN)
  132 DO 136 I=1,ITOP
  134 BEST=1.0E+35*XMODE
      X=I-1
      XN=XLOW+X*DELX
      NUFF=1
      CALL DLIMIT
      IF(MODES) 602,602,603
  603 CALL SEARCH(DLOW,DHIGH,DELD,XMODE,DNBEST,BEST)
      GO TO 400
  602 KTOP=(DHIGH-DLOW)/DELD+1.001
  223 DO 137 J=1,KTOP
      X=J-1
      DN=DLOW+X*DELD
C     JUMP MAY BE SET TO 1 IN TRANFM TO INDICATE AN INFEASIBLE XN-1
C     JUMP=0
      CALL TRANFM
      CALL STGRET
C     XMODE=1 FOR MIN, -1 FOR MAX
C     IF(JUMP.EQ.1)RN=999999*XMODE
      IF(NUFF) 400,401,401
  401 IF(N-1) 504,504,503
  504 QN=RN
      GO TO 143
  503 K=(YN-YLOW)/DELY+1.001
      IF(K) 212,212,209
  212 WRITE(NOUTPE,210) N,XN,DN,YN
  210 FORMAT(///9H AT STAGEI5,9H WITH XN=E15.8,8H AND DN=E15.8,12H XN-1
     1EQUALSE16.8,21H AND IS OUT OF LIMITS///)
      GO TO 2003
  209 IF(K-JTOP) 205,206,212
  206 RNM1=FNM1(JTOP)
      GO TO 207
  205 X=K-1
      X=YLOW+X*DELY
      RNM1=FNM1(K)+(FNM1(K+1)-FNM1(K))*(YN-X)/DELY
  207 GO TO (21,22,23,24,248),NNFLAG
  248 WRITE(NOUTPE,249) N
  249 FORMAT(43H1COMPOSITION OPERATOR NOT DEFINED FOR STAGE,I5)
      GO TO 2003
   21 QN=RN+RNM1
      GO TO 143
   22 QN=RN*RNM1
      GO TO 143
   23 IF(RN-RNM1) 141,141,142
  141 QN=RN
      GO TO 143
  142 QN=RNM1
      GO TO 143
   24 IF(RN-RNM1) 145,145,146
  145 QN=RNM1
      GO TO 143
  146 QN=RN
  143 IF((QN-BEST)*XMODE) 144,137,137
  144 BEST=QN
      DNBEST=DN
  137 CONTINUE
  400 WRITE(NCALC) BEST,DNBEST
  136 CONTINUE
      REWIND NCALC
      WRITE(NTAPE) XLOW,XHIGH,DELX,ITOP,XMODE
      DO 404 I=1,ITOP
  404 READ(NCALC) DUM,DNOFXN(I)
      REWIND NCALC
      WRITE(NTAPE) (DNOFXN(II),II=1,ITOP)
      DO 402 I=1,ITOP
  402 READ(NCALC) FN(I),DUM
      REWIND NCALC
```

```
      YLOW=XLOW
      YHIGH=XHIGH
      DELY=DELX
      JTOP=ITOP
  123 CONTINUE
      WRITE(NTAPE) (FN(II),II=1,ITOP)
  161 END FILE MTAPE
  197 REWIND MTAPE
      READ(MTAPE) NMAX
      N=NMAX
      J=N-1
      IF(J) 407,407,201
  201 DO 522 I=1,J
      READ(MTAPE)
  522 READ(MTAPE)
  407 READ(MTAPE) XLOW,XHIGH,DELX,ITOP,XMODE
      READ(MTAPE)
      READ(MTAPE) (FN(I),I=1,ITOP)
      BACKSPACE MTAPE
      BACKSPACE MTAPE
      READ(INTAPE,162) XN1,XN2
  162 FORMAT(2E15.8)
      NPAGE=NPAGE+1
      WRITE(NOUTPE,152) NPAGE
      WRITE(NOUTPE,2008) XN1, XN2
 2008 FORMAT(/// ' DATA CARD  ', 2F20.8 ///)
      IF(XMODE) 323,323,324
  324 WRITE(NOUTPE,325) N
  325 FORMAT(29H THE PROBLEM IS TO MINIMIZE AI6,14H STAGE PROCESS)
      GO TO 327
  323 WRITE(NOUTPE,326) N
  326 FORMAT(29H THE PROBLEM IS TO MAXIMIZE AI6,14H STAGE PROCESS)
  327 WRITE(NOUTPE,328) XN1,XN2
  328 FORMAT (42H XN IS TO BE CHOOSEN OPTIMALLY BETWEEN XN=1PE14.5,8H AN
     1D XN=1PE14.5)
  171 IF(XLOW-XN1) 175,175,176
  176 WRITE(NOUTPE,177) XLOW,XHIGH
  177 FORMAT(47H0THE PROGRAM ONLY HAS INFORMATION ON XN BETWEENE16.8,4H
     1ANDE16.8)
      GO TO 240
  175 IF(XN2-XHIGH) 181,181,176
  181 IF(XN1-XN2) 178,178,176
  178 IX1=(XN1-XLOW)/DELX+1.001
      IX2=(XN2-XLOW)/DELX+1.001
      BEST=1.0E+35*XMODE
      DO 182 J=IX1,IX2
      IF((FN(J)-BEST)*XMODE) 183,182,182
  183 BEST=FN(J)
      JSAVE=J
  182 CONTINUE
      X=JSAVE-1
      XN=XLOW+X*DELX
      WRITE(NOUTPE,184) XN,BEST
  184 FORMAT(12H OPTIMAL XN=1PE14.5,16H OPTIMAL RETURN=1PE14.5)
      READ(MTAPE) (DNOFXN(II),II=1,ITOP)
  186 DN=DNOFXN(JSAVE)
      CALL TRANFM
  204 WRITE(NOUTPE,188)
  188 FORMAT(7H0      N18X2HXN18X2HDN16X4HXN-1)
      NN=0
  203 WRITE(NOUTPE,189) N,XN,DN,YN
  189 FORMAT(I7,1P3E20.5)
      NN=NN+1
      N=N-1
      IF(N-1) 244,193,193
  244 GO TO 240
  193 DO 406 I=1,4
  406 BACKSPACE MTAPE
      READ(MTAPE) XLOW,XHIGH,DELX,ITOP,XMODE
      READ(MTAPE) (DNOFXN(II),II=1,ITOP)
      L=(YN-XLOW)/DELX+1.001
      IF(L) 214,214,215
```

```
  214 L=1
  219 NP1=N+1
      WRITE(NOUTPE,210) NP1,XN,DN,YN
      NN=NN+7
  215 IF(L-ITOP) 218,216,217
  216 DN=DNOFXN(L)
      XN=YN
      GO TO 196
  217 L=ITOP
      GO TO 219
  218 XN=YN
  194 X=L-1
      X=XLOW+X*DELX
      DN=DNOFXN(L)+(DNOFXN(L+1)-DNOFXN(L))*(XN-X)/DELX
  196 CALL TRANFM
      IF(NN-50) 203,203,220
  220 NPAGE=NPAGE+1
      WRITE(NOUTPE,152) NPAGE
      GO TO 204
  112 WRITE(NOUTPE,113) MAX,ITOP
  113 FORMAT(72H THIS PROGRAM LIMITS THE NUMBER OF DISCRETE STEPS OF THE
     1STATE VARIBLE TOI6,26H AND THIS PROBLEM REQUIRESI6)
 2002 FORMAT('1',' ERROR HALT')
 2003 WRITE(6,2002)
      STOP
 2000 WRITE(6,2001)
 2001 FORMAT('1','  END OF DATA FILE')
      STOP
      END
CSEARCH          DISCRETE FIBONACCI SEARCH SUBROUTINE
      SUBROUTINE SEARCH(AA,BB,DELYY,XXMODE,YYBEST,BEST)
      DIMENSION F(150)
C
C     OPTIMIZE WITH RESPECT TO Y BETWEEN AA AND BB IN STEPS OF DELYY
C     STORE OPTIMUM Y IN YYBEST AND THE OPTIMUM VALUE OF THE OBJECTIVE
C     FUNCTION IN BEST
C     XXMODE=-1 FOR MAXIMIZE
C     XXMODE=1  FOR MINIMIZE
C     SUBROUTINE MUST BE INITIALIZED BY CALLING IT WITH DELYY=-1.0 AT
C     LEAST ONCE BEFORE IT IS USED FOR A SEARCH
      DELY=DELYY
      A=AA
      B=BB
      XMODE=XXMODE
      IF(DELY) 100,100,101
  100 F(1)=1.0
      F(2)=2.0
      DO 1 I=3,150
      F(I)=F(I-1)+F(I-2)+1.0
      IF(F(I)-1.0E+35) 1,2,2
    2 II=I
      RETURN
    1 CONTINUE
      II=150
      RETURN
  101 YHI=B
      YLO=A
      FNO=(YHI-YLO)/DELY+1.0
      DO 5 N=1,II
      IF(FNO-F(N)) 6,6,5
    5 CONTINUE
      WRITE(6,7)
    7 FORMAT(38H1ERROR, TOO MANY POINTS TO BE SEARCHED)
 2002 FORMAT('1',' ERROR HALT')
 2003 WRITE(6,2002)
      STOP
    6 IF(N-2) 42,41,40
   40 Y1=F(N-2)*DELY+YLO
      CALL FUNCTN(Y1,GY1)
      Y2=F(N-1)*DELY+YLO
      CALL FUNCTN(Y2,GY2)
   16 N=N-1
```

```
      IF(N-1) 102,102,103
  103 IF((GY2-GY1)*XMODE) 19,19,20
   19 YLO=Y1+DELY
      Y1=Y2
      GY1=GY2
      Y2=F(N-1)*DELY+YLO
      IF(Y2-B) 104,104,105
  105 GY2=1.0E+35*XMODE
      GO TO 16
  104 CALL FUNCTN(Y2,GY2)
      GO TO 16
   20 YHI=Y2-DELY
      Y2=Y1
      GY2=GY1
      IF(N-2) 34,34,35
   34 FNM2=0.0
      GO TO 36
   35 FNM2=F(N-2)
   36 Y1=FNM2*DELY+YLO
      CALL FUNCTN(Y1,GY1)
      GO TO 16
  102 IF((GY2-GY1)*XMODE) 110,111,111
  110 YYBEST=Y2
      BEST=GY2
      RETURN
  111 YYBEST=Y1
      BEST=GY1
      RETURN
   41 Y1=A
      CALL FUNCTN(Y1,GY1)
      Y2=B
      CALL FUNCTN(Y2,GY2)
      GO TO 102
   42 Y1=A
      CALL FUNCTN(Y1,GY1)
      GO TO 111
      END
CFUNCTN
      SUBROUTINE FUNCTN(Y,GY)
      DIMENSION FN(10001),FNM1(10001),DNOFXN(10001)
      REAL * 8 TAB(4)
      COMMON /CON/TAB
      COMMON XN,N,DN,YN,RN,SLOW,XHIGH,DELX,DLOW,DHIGH,DELD,NUMBER
      COMMON YLOW,YHIGH,DELY,NUFF,FN,JTOP,NNFLAG,NMAX,XLAM
      COMMON A,B,C,D,E,SUMELT,KODE
      EQUIVALENCE (FN(1),FNM1(1),DNOFXN(1))
      DN=Y
      NOUTPE=6
C     JUMP MAY BE SET TO 1 IN TRANFM TO INDICATE AN INFEASIBLE XN-1
C     JUMP=0
      CALL TRANFM
      CALL STGRET
C     XMODE=1 FOR MIN, -1 FOR MAX
C     IF(JUMP.EQ.1)RN=999999*XMODE
      IF(N-1) 146,146,401
  401 K=(YN-YLOW)/DELY+1.001
      IF(K) 212,212,209
  212 WRITE(NOUTPE,210) N,XN,DN,YN
  210 FORMAT(///9H AT STAGEI5,9H WITH XN=E15.8,8H AND DN=E15.8,12H XN-1
     1EQUALSE16.8,21H AND IS OUT OF LIMITS///)
 2002 FORMAT('1',' ERROR HALT')
 2003 WRITE(NOUTPE,2002)
      STOP
  209 IF(K-JTOP) 205,206,212
  206 RNM1=FNM1(JTOP)
      GO TO 207
  205 X=K-1
      X=YLOW+X*DELY
      RNM1=FNM1(K)+(FNM1(K+1)-FNM1(K))*(YN-X)/DELY
  207 GO TO (21,22,23,24),NNFLAG
   21 QN=RN+RNM1
      GO TO 137
```

```
   22 QN=RN*RNM1
      GO TO 137
   23 IF(RN-RNM1) 141,141,142
  141 QN=RN
      GO TO 137
  142 QN=RNM1
      GO TO 137
   24 IF(RN-RNM1) 145,145,146
  145 QN=RNM1
      GO TO 137
  146 QN=RN
  137 GY=QN
      RETURN
      END
```

```
CSTORE
      SUBROUTINE STORE
      DIMENSION FN(10001),FNM1(10001),DNOFXN(10001),A(101),B(101),C(101)
      DIMENSION D(101),E(101)
      REAL * 8 TAB(4)
      COMMON /CON/TAB
      COMMON XN,N,DN,YN,RN,SLOW,XHIGH,DELX,DLOW,DHIGH,DELD,NUMBER
      COMMON YLOW,YHIGH,DELY,NUFF,FN,JTOP,NNFLAG,NMAX,XLAM
      COMMON A,B,C,D,E,SUMELT,KODE1,KODE2
      EQUIVALENCE (FN(1),FNM1(1),DNOFXN(1))
      INDATA=8
      NOUTPE=6
      READ (INDATA,808) KODE1,KODE2
  808 FORMAT (2I3)
      IF(KODE1.EQ.0) WRITE(NOUTPE,110)
      IF(KODE1.EQ.1) WRITE(NOUTPE,101)
      IF(KODE1.EQ.2) WRITE(NOUTPE,202)
      IF(KODE1.EQ.3) WRITE(NOUTPE,303)
  110 FORMAT (30H THE OBJECTIVE IS COST         )
  101 FORMAT (30H THE OBJECTIVE IS SMA          )
  202 FORMAT (30H THE OBJECTIVE IS MSRT         )
  303 FORMAT (30H THE OBJECTIVE IS AVAILABILITY)
      IF(KODE2.EQ.0) WRITE(NOUTPE,900)
      IF(KODE2.EQ.1) WRITE(NOUTPE,901)
      IF(KODE2.EQ.2) WRITE(NOUTPE,902)
      IF(KODE2.EQ.3) WRITE(NOUTPE,903)
  900 FORMAT (30H CONSTRAINT ON COST            )
  901 FORMAT (30H CONSTRAINT ON SMA             )
  902 FORMAT (30H CONSTRAINT ON MSRT            )
  903 FORMAT (30H CONSTRAINT ON AVAILABILITY    )
      DO 333 K=1,NMAX
      READ (INDATA,999) A(K),C(K),E(K),B(K),D(K)
  999 FORMAT (5F10.4)
      WRITE(NOUTPE,999) A(K),B(K),C(K),D(K),E(K)
  333 CONTINUE
      SUMELT=0.
      DO 30 I=1,NMAX
   30 SUMELT=SUMELT+E(I)*A(I)*B(I)
      WRITE(6,765)SUMELT
  765 FORMAT(F10.3)
      RETURN
      END
```

```
CSTGRET
      SUBROUTINE STGRET
      DIMENSION FN(10001),FNM1(10001),DNOFXN(10001),A(101),B(101),C(101)
      DIMENSION D(101),E(101)
      REAL * 8 TAB(4)
      COMMON /CON/TAB
      COMMON XN,N,DN,YN,RN,SLOW,XHIGH,DELX,DLOW,DHIGH,DELD,NUMBER
      COMMON YLOW,YHIGH,DELY,NUFF,FN,JTOP,VNFLAG,NMAX,XLAM
      COMMON A,B,C,D,E,SUMELT,KODE1,KODE2
      EQUIVALENCE (FN(1),FNM1(1),DNOFXN(1))
      IF(KODE1.EQ.0)GO TO 77
      IDN=INT(DN)
      AB=A(N)*B(N)
      IS=IDN
      TERM=0.
      TEMP=0.
      IF(IS.LT.0) GO TO 11
      TERM=EXP(-AB)
      TEMP=TERM
      IF(IS.EQ.0) GO TO 11
      DO 10 I=1,IS
      TEMP=TEMP*AB/I
      IF(TERM.GE..99999) GO TO 11
C     IF(TEMP.LE..00001) GO TO 11
   10 TERM=TERM +TEMP
   11 CDF=TERM
   20 CONTINUE
      IF(KODE1.NE.1)GO TO 40
C FOLLOWING CHANGED 120183
C     SMA=(AB*(1.-TEMP)+(IDN-AB)*(1.-CDF))/AB
      SMA=(1.-TEMP)+(IDN-AB)*(1.-CDF)/AB
      RN=E(N)*AB*SMA/SUMELT
      RETURN
   40 TWUS=(1.-CDF)*(AB*AB-2.*AB*IDN+IDN*(IDN+1))/(2.*A(N))
     X+TEMP*B(N)*(AB-IDN)/2.
      IF(KODE1.EQ.3)GO TO 50
      RN=E(N)*TWUS/SUMELT
      RETURN
   50 CONTINUE
      AMSRT=TWUS/AB
      AMTBF=1./A(N)
      AMTTR=D(N)
      RN=(AMTBF)/(AMTBF+AMTTR+AMSRT)
      RETURN
   77 RN=C(N)*DN
C     IF(KODE2.EQ.1.AND.N.EQ.1.AND.YN.LT.XN)RN=999999
      RETURN
      END
```

```
CDLIMIT
      SUBROUTINE DLIMIT
      DIMENSION FN(10001),FNM1(10001),DNOFXN(10001),A(101),B(101),C(101)
      DIMENSION D(101),E(101)
      REAL * 8 TAB(4)
      COMMON /CON/TAB
      COMMON XN,N,DN,YN,RN,SLOW,XHIGH,DELX,DLOW,DHIGH,DELD,NUMBER
      COMMON YLOW,YHIGH,DELY,NUFF,FN,JTOP,NNFLAG,NMAX,XLAM
      COMMON A,B,C,D,E,SUMELT,KODE1,KODE2
      EQUIVALENCE (FN(1),FNM1(1),DNOFXN(1))
      IF(KODE2.NE.0)GO TO 37
      DLOW=0.
      DELD=1.
      DHIGH=AMIN1(XN/C(N),50.)
      RETURN
   37 AB=A(N)*B(N)
      DO 555 ID=1,30
      IS=ID-1
      TERM=EXP(-AB)
      TEMP=TERM
      IF(IS.EQ.0) GO TO 11
      DO 10 I=1,IS
      TEMP=TEMP*AB/I
      IF(TERM.GE..99999) GO TO 11
C     IF(TEMP.LE..00001) GO TO 11
   10 TERM=TERM +TEMP
   11 CDF=TERM
   40 TWUS=(1.-CDF)*(AB*AB-2.*AB*IS+IS*(IS+1))/(2.*A(N))
     X+TEMP*B(N)*(AB-IS)/2.
      IF(KODE2.EQ.2.AND.E(N)*TWUS/SUMELT.LE.XN)GO TO 556
C   CHANGED FOLLOWING TO 555 FROM 556
      IF(KODE2.EQ.2)GO TO 555
      IF(KODE2.EQ.3)GO TO 554
      SSS=(1.-TEMP)+(IS-AB)*(1.-CDF)/AB
      RR=E(N)*AB*SSS/SUMELT
      Y=XN-RR
      DLOW=0.
C THE NUMBER IN THE NEXT LINE HAS A BIG EFFECT. IT SHOULD NOT.
      IF(N.EQ.1)DLOW=1000.
      IF(N.EQ.1.AND.Y.GT.0..AND.ID.LT.30)GO TO 555
      IF(N.EQ.1.AND.Y.LE.0.)DLOW=IS
  136 DHIGH=30.+DLOW
      IF(DLOW.EQ.1000.)DHIGH=1000.
      DELD=1.
      RETURN
  554 AMSRT=TWUS/AB
      AMTBF=1./A(N)
      AMTTR=D(N)
      AV=AMTBF/(AMTBF+AMTTR+AMSRT)
      IF(AV.GE.XN) GO TO 556
  555 CONTINUE
      IS=30.
  556 DLOW=IS
      DELD=1.
      DHIGH=50.
      RETURN
      END
      BLOCK DATA
      COMMON /CON/TAB
      REAL * 8 TAB(4) /'   SUM','  MULT','MAXMIN','MINMAX'/
      END
```

FILE: TRANFM   FORTRAN A1 NAVAL POSTGRADUATE SCHOOL

```
CTRANFM
      SUBROUTINE TRANFM
      DIMENSION FN(10001),FNM1(10001),DNOFXN(10001),A(101),B(101),C(101)
      DIMENSION D(101),E(101)
      REAL * 8 TAB(4)
      COMMON /CON/TAB
      COMMON XN,N,DN,YN,RN,SLOW,XHIGH,DELX,DLOW,DHIGH,DELD,NUMBER
      COMMON YLOW,YHIGH,DELY,NUFF,FN,JTOP,NNFLAG,NMAX,XLAM
      COMMON A,B,C,D,E,SUMELT,KODE1,KODE2
      EQUIVALENCE (FN(1),FNM1(1),DNOFXN(1))
      IF(KODE2.NE.0)GO TO 12
      YN=XN-C(N)*DN
      RETURN
   12 IDN=INT(DN)
      AB=A(N)*B(N)
      IS=IDN
      TERM=0.
      TEMP=0.
      IF(IS.LT.0) GO TO 11
      TERM=EXP(-AB)
      TEMP=TERM
      IF(IS.EQ.0) GO TO 11
      DO 10 I=1,IS
      TEMP=TEMP*AB/I
      IF(TERM.GE..99999) GO TO 11
C     IF(TEMP.LE..00001) GO TO 11
   10 TERM=TERM +TEMP
   11 CDF=TERM
   40 TWUS=(1.-CDF)*(AB*AB-2.*AB*IDN+IDN*(IDN+1))/(2.*A(N))
     X+TEMP*B(N)*(AB-IDN)/2.
      IF(KODE2.EQ.2) GO TO 50
      IF(KODE2.EQ.3) GO TO 51
      SMA=(1.-TEMP)+(IDN-AB)*(1.-CDF)/AB
      YN=XN-E(N)*AB*SMA/SUMELT

      IF(YN.LT.0) YN=0.0

C     IF(N.NE.1.AND.YN.LT.0.OR.N.NE.1.AND.YN.GT.1.)JUMP=1
C     IF(N.EQ.1.AND.YN.GT.0)JUMP=1
C     IF(JUMP.EQ.1)YN=0.
      RETURN
   51 AMSRT=TWUS/AB
      AMTBF=1./A(N)
      AMTTR=D(N)
      AV=(AMTBF)/(AMTBF+AMTTR+AMSRT)
      YN=XN/AV
C     AV IS AVAIL
      RETURN
   50 YN=XN-E(N)*TWUS/SUMELT
      IF(YN.LE.0.)YN=0.
      RETURN
      END
```

```
      DIMENSION A(20),B(20),C(20),D(20),E(20),MINQ(20)
      DIMENSION T(10,30),F(1200),DX(10,1200),TN(10,31)
      INTEGER BUDGET,BB,BBP1,DX
      REAL*8 TIME,CTIME
      INDATA=8
      NOUTPE=6
      READ(INDATA,888) NMAX,BUDGET
  888 FORMAT(2I5)
      READ (INDATA,777) (MINQ(I),I=1,NMAX)
  777 FORMAT(20I5)
      DO 333 K=1,NMAX
      READ (INDATA,999) A(K),C(K),E(K),B(K),D(K)
  999 FORMAT (5F10.4)
C     WRITE(NOUTPE,999) A(K),B(K),C(K),D(K),E(K)
  333 CONTINUE
      TIME=CTIME(1)
      SUMELT=0.
      DO 30 I=1,NMAX
   30 SUMELT=SUMELT+E(I)*A(I)*B(I)

      DO 10 I=1,NMAX
      DX(I,1)=0
      DO 110 KP1=1,31
      K=KP1-1
      AB=A(I)*B(I)
      TERM=EXP(-AB)
      TEMP=TERM
      IF(K.EQ.0) GO TO 11
      DO 14 JI=1,K
      TEMP=TEMP*AB/JI
C     IF(TERM.GE..999999) GO TO 11
C     IF(TEMP.LE..0000001) GO TO 11
   14 TERM=TERM +TEMP
   11 CDF=TERM

   40 TWUS=(1.-CDF)*(AB*AB-2.*AB*K+K*(K+1))/(2.*A(I))
     X+TEMP*B(I)*(AB-K)/2.
  110 TN(I,KP1)=E(I)*TWUS/SUMELT
      DO 70 K=1,30
   70 T(I,K)=TN(I,K+1)-TN(I,K)
   10 CONTINUE

C     DO 456 IJK=1,10
C 456 WRITE(6,555)(TN(KI,IJK),KI=1,NMAX)
C 555 FORMAT(5F12.7)


      DO 80 I=1,NMAX
      LIM=30/MINQ(I)
      DO 85 K=1,LIM
      TMOD=0.
      MM=MINQ(I)
      DO 90 J=1,MM
   90 TMOD=TMOD+T(I,(K-1)*MINQ(I)+J)
      T(I,K)=TMOD
   85 CONTINUE
      C(I)=MINQ(I)*C(I)
   80 CONTINUE

          IBD=BUDGET+1
      DO 100 BBP1=1,IBD
          BB=BBP1-1
          XMIN=0.
          ISTAR=0
          TRY=0.
          DO 210 I=1,NMAX
          IF(BB-C(I).LT.0) GO TO 210
          IF(BB-C(I).GT.0) GO TO 211
          TRY=AMIN1(TRY,T(I,1))
          GO TO 212
```

```
  211 TRY=T(I,DX(I,BBP1-C(I))+1)+F(BBP1-C(I))
  212 IF (TRY.GT.XMIN) GO TO 210
      XMIN=TRY
      ISTAR=I
  210 CONTINUE
      F(BBP1)=XMIN
   19 DO 21 I=1,NMAX
      IF (ISTAR.EQ.0.OR.BB-C(ISTAR).LE.0) DX(I,BBP1)=0
      IF (ISTAR.EQ.0.OR.BB-C(ISTAR).LE.0) GO TO 21
      DX(I,BBP1)=DX(I,BBP1-C(ISTAR))
   21 CONTINUE
      IF(ISTAR.EQ.0)GO TO 100
      IF(BBP1-C(ISTAR).LE.1) DX(ISTAR,BB+1)=1
      IF(BBP1-C(ISTAR).LE.1) GO TO 100
      DX(ISTAR,BBP1)=AMINO(DX(ISTAR,BBP1-C(ISTAR))+1,30)
  100 CONTINUE
      KLOW=IBD-5
      KHIGH=IBD
      DO 101 KBD=KLOW,KHIGH
      TIME=TIME-CTIME(1)
C     WRITE(6,3456)TIME
 3456 FORMAT(' TIME=',F12.9)
      WRITE(6,1000)(DX(I,KBD),I=1,NMAX)
      WRITE(6,1000)(MINQ(I),I=1,NMAX)
 1000 FORMAT(10I5)
      AMSRT=0.
      DO 88 I=1,NMAX
   88 AMSRT=AMSRT+TN(I,MINQ(I)*DX(I,KBD)+1)
      KBDM1=KBD-1
      WRITE (6,2000) AMSRT,KBDM1
 2000 FORMAT(1X,'MSRT=  ',F10.8,'   BUDGET=  ',I7)
  101 CONTINUE
      RETURN
      END
      FUNCTION CTIME(I)
      REAL*8 CTIME
      DATA IFL/0/
      IF( IFL.NE.0 ) GO TO 10
      IFL = 1
      CTIME = 0.0D0
      CALL SETIME
      RETURN
   10 CALL GETIME(ITIME)
      CTIME = -ITIME*0.0000026D0
      RETURN
      END
```

References

1. Dreyfus, S. E., and Law, A. M., "The Art and Theory of Dynamic Programming," Academic Press, New York, 1977.

2. Richards, F. R., and McMasters, A. W., "Wholesale Provisioning Models: Model Development," Naval Postgraduage School Technical Report NPS55-83-026, September 1983.

3. Kang, Sung Jin, "Analysis of Inventory Models with Budget Constraints" Masters Thesis, Naval Postgraduate School, September 1983.

DISTRIBUTION LIST

	No. of Copies
Defense Technical Information Center Cameron Station Alexandria, VA 22314	2
Defense Logistics Studies Information Exchange U. S. Army Logistics Management Center Fort Lee, VA 23801	1
Library, Code 0142 Naval Postgraduate School Monterey, CA 93943	2
Library, Code 55 Department of Operations Research Naval Postgraduate School Monterey, CA 93943	1
Associate Professor G. T. Howard Code 55 Hk, Department of Operations Research Naval Postgraduate School Monterey, CA 93943	5
Associate Professor F. Russell Richards Code 55Rh, Department of Operations Research Naval Postgraduate School Monterey, CA 93943	5
Associate Professor Alan W. McMasters, Code 54Mg Department of Administrative Sciences Naval Postgraduate School Monterey, CA 93943	5
Commanding Officer Navy Fleet Material Support Office Attn: Code 93 Mechanicsburg, PA 17055	5
Commanding Officer Naval Supply Systems Command Attn: SUP 04A Washington, DC 20376	5
Commanding Officer Navy Ships Parts Control Center Attn: Code 0412 Mechanicsburg, PA 17055	5
Commanding Officer Navy Aviation Supply Office Attn: Code SD84-A Philadelphia, PA 19111	5

END

DATE
FILMED

3-84

DTIC