WORKING PAPER

A DYNAMIC INTERACTIVE DECISION ANALYSIS AND SUPPORT SYSTEM (DIDASS)

USER'S GUIDE (MAY 1983)

Manfred Grauer

June 1983
WP-83-60

IIASA
International Institute
for Applied Systems Analysis

A DYNAMIC INTERACTIVE DECISION ANALYSIS AND SUPPORT SYSTEM (DIDASS)

USER'S GUIDE (MAY 1983)

Manfred Grauer

June 1983
WP-83-60

INTERNATIONAL INSTITUTE FOR APPLIED SYSTEMS ANALYSIS
A-2361 Laxenburg, Austria

A DYNAMIC INTERACTIVE DECISION ANALYSIS AND SUPPORT SYSTEM (DIDASS)

USER'S GUIDE (MAY 1983)

Manfred Grauer

System and Decision Sciences Program
International Institute for Applied Systems Analysis (IIASA)
Laxenburg, Austria

PREFACE

The Interactive Decision Analysis group at IIASA has recently developed an interactive decision support system called DIDASS (dynamic interactive decision analysis and support system). The major advantage of this system over most other computerized approaches to decision problems is that it is interactive, that is, it involves the decision maker in the decision process. It is an attempt to combine the analytical power of the "hard" computer model with the qualitative assesments of the decision maker.

DIDASS is an interactive multicriteria programming package based on the reference (aspiration) approach to multicriteria analysis, and is capable of dealing with both linear and nonlinear problems. It has been written in FORTRAN 77, avoiding the use of any operating-system-dependent statements or commands, which means that it can be transfered to almost any computer without difficulty. This guide has been prepared for users both at IIASA and at the many collaborating institutions where DIDASS is now running, and is based on the version of DIDASS available on tape from IIASA.

If you have any comments or suggestions concerning the system or this guide, we would be glad to hear from you - DIDASS is intended to be useful, useable, and used!

ANDRZEJ WIERZBIEKI

Chairman

Systems and Decision Sciences

1. INTRODUCTION

DIDASS is an interactive multicriteria programming package designed for decision support. It is able to deal with both linear and nonlinear multicriteria programming problems, and is based on the reference point approach to multicriteria analysis.

The basic idea of the reference point method is to rank multidimensional decision alternatives q, defined as points in the R^p $(p \geq 2)$, relative to a reference point $\bar{q}$ which reflects the preferences of the user.

The ranking of the decision alternatives is based on a partial ordering of the R^p:

$$q^1 \leq q^2; \quad q_i^1 \leq q_i^2 \; ; \quad i = 1,2,\ldots p \; ; \; q^1;q^2 \in R^p \qquad (1)$$

The decision problem is to determine an n-vector x of decision variables satisfying all given constraints while taking into account the p-vector of objectives. We will assume that each component of q should be as large as possible.

A ***reference point*** or ***reference objective*** is a suggestion $\bar{q}$ supplied by the user which reflects in some sense the "desired level" of the objective. An achievement scalarizing function $s(q-\bar{q})$ defined over the set of objective vectors q is then associated with each reference point $\bar{q}$ [1]. If we regard the function $s(q-\bar{q})$ as the "distance" between the points q and $\bar{q}$, then, intuitively, the problem of minimizing this distance may be interpreted as the problem of finding from within the Pareto set the point $\hat{q}$ "nearest" to the reference point $\bar{q}$. (However, the function s is not necessarily related to the usual notion of distance.) With this interpretation in mind, reference point optimization may be viewed as a way of guiding a sequence $\{ \hat{q}^k \}$ of Pareto points generated from a sequence $\{ \bar{q}^k \}$ of reference objectives. These sequences are generated through an interactive procedure and should result in a set of attainable efficient points $\{ \hat{q}^k \}$ of interest to the user. If the sequence $\{ \hat{q}^k \}$ converges, the limit may be seen as the solution to the decision problem.

2. PROBLEM FORMULATION

Let us assume that the decision problem can be clarified by analyzing a general constrained multicriteria problem in the following standard form:

$$\max_{x_{nl}, x_l} \begin{bmatrix} f_1(x_{nl}) + c_1^T x_{nl} + d_1^T x_l = q_1 \\ f_2(x_{nl}) + c_2^T x_{nl} + d_2^T x_l = q_2 \\ \cdots \quad \cdots \quad \cdots \\ f_p(x_{nl}) + c_p^T x_{nl} + d_p^T x_l = q_p \end{bmatrix} \tag{2}$$

subject to:

$$g(x_{nl}) + A_1 x_l \le b_1 \tag{3}$$

$$A_2 x_{nl} + A_3 x_l \le b_2 \tag{4}$$

$$l \le \begin{bmatrix} x_{nl} \\ x_l \end{bmatrix} \le u \tag{5}$$

where $g(x_{nl}) = \left[g_1(x_{nl}), g_2(x_{nl}), \ldots, g_m(x_{nl})\right]^T$ is a vector of nonlinear constraints and $f_1(x_{nl}), f_2(x_{nl}), \ldots, f_p(x_{nl})$ in (2) represents the nonlinear parts of the performance criteria. The decision variables are divided into two subsets: a vector of "nonlinear" variables (x_{nl}) and a vector of "linear" variables (x_l). It is clear that when vectors f and g are nonexistent, formulation (2)-(5) is identical with the standard multicriteria linear programming problem. An overview of the various ways in which the reference point approach can be used in the linear case is given in [2], while the nonlinear case is described in [3].

The current computer implementation of the decision analysis and support system DIDASS is based on a two-stage model of the decision-making process. In the first stage - the exploratory stage - the user is informed about the range of his alternatives, thus giving him an overview of the problem. In the second stage - the search stage - the user works with the system in an interactive way to analyze the efficient alternatives $\{\hat{q}^k\}$ generated by DIDASS in response to his reference objectives $\{ \bar{q}^k \}$. The initial information for the exploratory stage is provided by calculating the extreme points for each of the objectives in (2) separately. A matrix D_S which yields information on the range of numerical values of each objective is then constructed. We shall call this the *decision support matrix*.

$$D_S = \begin{bmatrix} q_1^* & q_2^1 & \cdots & q_p^1 \\ q_1^2 & q_2^* & \cdots & q_p^2 \\ \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot \\ q_1^i & q_2^i & \cdots & q_p^i \\ \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot \\ q_1^p & q_2^p & \cdots & q_p^* \end{bmatrix} \tag{6}$$

Row i corresponds to the solution vector x_i which maximizes objective q_i. The vector with elements $q_i^i = q_i^*$, i.e., the diagonal of D_S, represents the ***utopia (ideal) point***. This point is not normally attainable (if it were, it would be the solution of the proposed decision problem), but it is presented to the user as an upper guideline to the sequence $\{ \bar{q}^k \}$ of reference objectives. Let us consider column i of the matrix D_S. The maximum value in the column is q_i^*. Let q_i^n be the minimum value, where

$$\min_{1 \le j \le p} \left\{ q_i^j \right\} = q_i^n$$

We shall call this the ***nadir*** value. The vector with elements $q_1^n, q_2^n, \ldots, q_p^n$ represents the ***nadir point***, and may be seen as a lower guideline to the values of the user's objectives.

In the linear case we use the following scalarizing function $s(w)$, where minimization results in a linear programming formulation:

$$s(w) = -\min\left\{\rho \min_i w_i \; ; \; \sum_{i=1}^{p} w_i\right\} - \varepsilon\, w \tag{7}$$

Here $w_i \equiv (q_i - \bar{q}_i)/\gamma_i$, ρ is an arbitrary coefficient which is greater than or equal to p, γ_i is a scaling factor, and $\varepsilon = (\varepsilon_1, \varepsilon_2, \ldots, \varepsilon_p)$ is a nonnegative vector of parameters.

In the nonlinear version of the package the following achievement scalarizing function is used:

$$s(w) = -\frac{1}{\rho} \ln\left[\frac{1}{p}\sum_{i=1}^{p} (w_i)^{\rho}\right] \tag{8}$$

where $w_i = \gamma_i \left[(\tilde{q}_i - q_i)/(\tilde{q}_i - \bar{q}_i)\right]$, $\tilde{q}_i$ is an upper limit to the sequence of reference points, $\rho \geq 2$ is again an arbitrary coefficient greater than or equal to p, and γ_i acts here as a weighting factor. This achievement scalarizing function meets the following requirements:

- It yields scaling factors which make additional scaling of objectives unnecessary.
- It is a smoothly differentiable function that approximates the nonsmooth function $s = \max_i w_i$.
- It is strongly order-preserving and weakly order-approximating.

The resulting single-criterion programming problems are solved using the solution package MINOS [4].

3. INFORMATION FOR IMPLEMENTATION

The current version of the DIDASS package has been designed specifically to be portable, and it has therefore been written completely in FORTRAN 77, avoiding the use of any operating-system-dependent statements or commands.

The DIDASS source code is normally supplied by IIASA on tape (9-track, unlabeled, ebcdic, upper case, 800 bpi, block size 800 characters, record length 80 characters) under the following names:

```
list_of_files

mtibm: Tape format: upper case ebcdic
mtibm: Record length: 80. Block length: 800. Density: 800. bpi.
File:1      Records:9       Blocks:1      Filename:constr_l.f
File:2      Records:5       Blocks:1      Filename:constr_nl.f
File:3      Records:1407    Blocks:141    Filename:didass.f
File:4      Records:27      Blocks:3      Filename:list_of_files
File:5      Records:45      Blocks:5      Filename:model.l
File:6      Records:100     Blocks:10     Filename:model.nl
File:7      Records:6765    Blocks:677    Filename:nonlp.f
File:8      Records:9       Blocks:1      Filename:object_l.f
File:9      Records:103     Blocks:11     Filename:object_nl.f
File:10     Records:4       Blocks:1      Filename:rfp.l
File:11     Records:4       Blocks:1      Filename:rfp.nl
File:12     Records:22      Blocks:3      Filename:specs.l
File:13     Records:46      Blocks:5      Filename:specs.nl
```

To prepare DIDASS to solve linear problems only the user must read all files (including data and FORTRAN files) from the tape, compile, link and load the following FORTRAN files:

constr_l.f, object_l.f, didass.f, nonlp.f .

To prepare DIDASS to solve mixed linear and nonlinear problems, or nonlinear problems only, the user must read all the files (including data and FORTRAN files, excluding the files constr_l.f and object_l.f), compile, link and load the following FORTRAN files:

constr_nl.f, object_nl.f, didass.f, nonlp.f.

To support the implementation of the system, data for a linear and a nonlinear example are also given on the tape.

4. SOLVING A LINEAR PROBLEM

The solution of a linear problem is demonstrated by example I (hypoth.1):

$$\max \begin{Bmatrix} 1.5x_1 + 2x_2 - x_3 + 3x_4 + x_5 + x_7 = obj1 \\ 1.2x_1 + x_2 + x_3 + x_4 + 2.75x_5 + x_6 = obj2 \\ 2.5x_1 + x_3 + 2x_4 + 1.7x_5 - x_6 - x_7 = obj3 \end{Bmatrix}$$

subject to :

$$\begin{aligned} + x_1 + 2x_2 + x_3 + x_4 + 2x_5 + x_6 + 2x_7 &\leq 12.25 \\ - 2x_1 - x_2 + x_4 + 2x_5 + x_7 &\leq 13.75 \\ - x_1 + x_3 + 2x_5 - 2x_7 &\leq 14.00 \\ + x_2 + 2x_3 - x_4 + x_5 - x_6 - x_7 &\leq 16.5 \end{aligned}$$

$$x_i \geq 0, \; i=1,2,\dots,7$$

The MPS input file should be prepared in a certain way - the user must list objectives of type e (equalities) as the first entries in the row definition section in the same order as they appear in the reference point file (see later). The input file must contain a section on bounds, even if this section is empty. The MPS file ("model.l") and the specification file ("specs.l") for the above example are given in Appendix 1.

The reference point file ("rfp.l") has the format (2X,2A4,2X,3F12.5). The first two characters are blanks, the next eight characters contain the name of the objective, and there are then two more blanks. The first in F12.5 field gives the value of the reference point. The next field contains the max-min indicator, which is + for maximization, and - for minimization; the corresponding digits can be used as scaling factors. The last contains the values of the control coefficients, $\rho \geq p$ on the first line, and ε on the second line. The last line of the file must contain four dots (....) as characters 5-8 in the A4 field. For example I the reference point file would take the following form:

```
obj1    35.0    +1.0    3.00
obj2    15.0    +1.0    0.100e-06
obj3    25.0    +1.0
....
```

The files "constr_l.f" and "object_l.f" contain dummy subroutines which are needed if the user intends to solve linear problems only. The steps taken in the program for the linear case are explained in Figure 1.

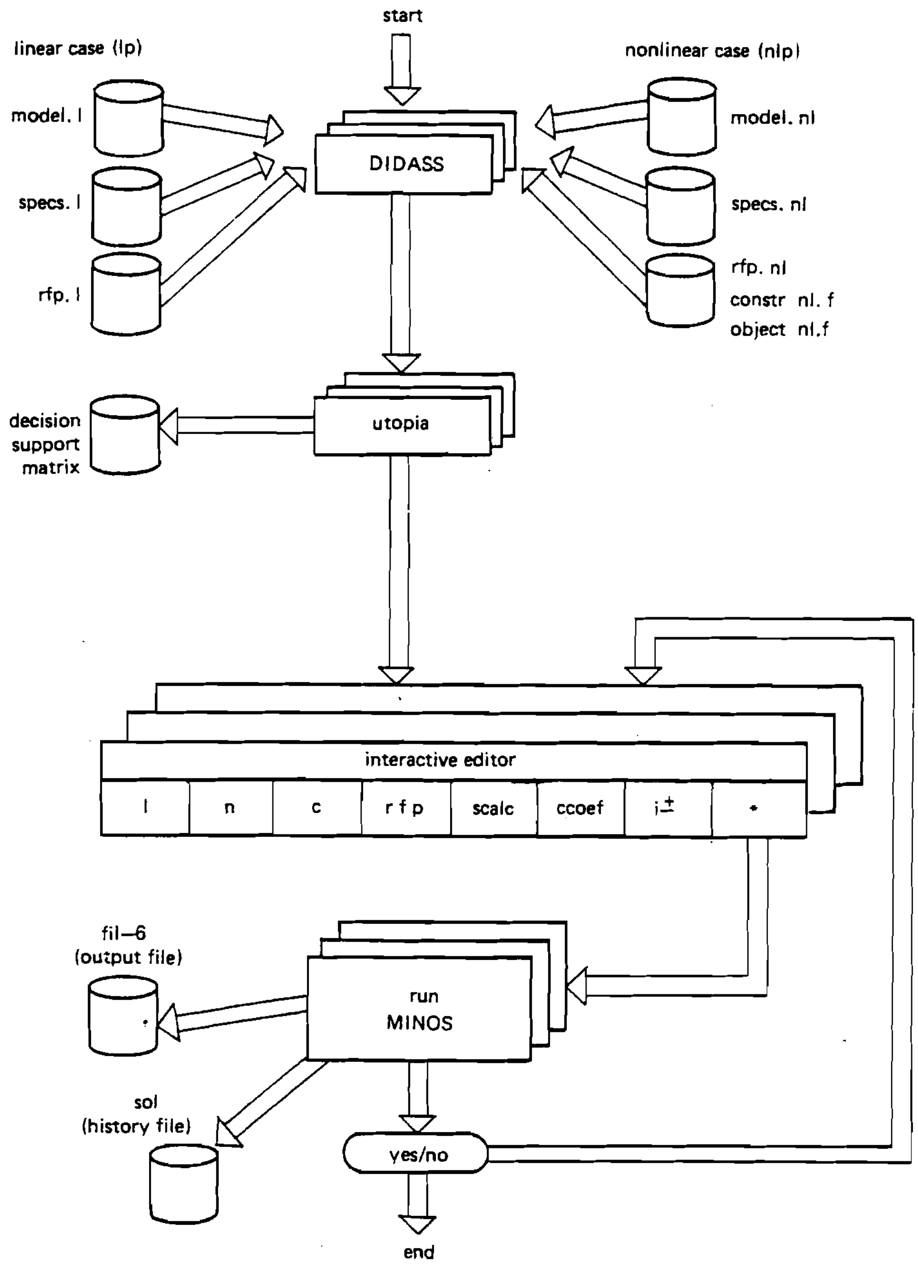

Figure 1 Structure of the DIDASS-system

The interactive procedure begins with the program asking which type of problem is to be considered (lp in this case). The program then enters the exploratory stage and informs the user of the range of values possible for each alternative. This leads into the search stage, which is split into two parts: editing and problem solving. In the editing mode the following commands may be used:

l - list the names of the objectives and the components of the reference points

n - neutral solution - zero is the reference point

i+ - positive infinite reference point ($+10^5$)

i- - negative infinite reference point (-10^5)

c - copy solution from previous session as reference point

The following commands are also available:

rfp - change to reference point definition status

scalc - change to scaling factor modification status

ccoef - change to control coefficient definition status

In order to define a new value of ρ (or ε), it is sufficient to type ccoef and then rho (or eps) ; the program moves to the corresponding definition status, the new value of the parameter may be typed in and the program returns to waiting status.

The reference point components may be redefined in rfp status. To do this, it is necessary to type two lines - one containing the name of the objective, the other the new value of the reference point component. Redefinition of scaling factors is carried out in scaling factor modification status. Here again it is necessary to type two lines - one containing the name of the objective, the other the new value of the scaling factor. The only way to leave editing status is to type an asterisk (*).

In the next phase of the interactive process the program asks the user for the names of the RHS and BOUNDS sections. (In our example these are "rhs" and "bnd".) Using this option the user can change the set of constraints between sessions, thus modifying the problem. The system then generates a new MPS file and the single-criterion LP problem (7) is solved. Finally, the necessary information from the LP output file is extracted and presented to the user in the same form as the original problem.

The user then repeats the search stage until he obtains satisfactory results.

5. SOLVING A NONLINEAR PROBLEM

The solution of a nonlinear problem is demonstrated by example II (theo):

$$\min \left\{ \begin{array}{l} (x_1-3)^2 + (x_2-2)^2 + (x_6-6)^2 + (x_7-4)^2 = obj1 \\ 0.5(x_3-4)^2 + (x_8-6)^2 + (x_9-11)^2 = obj2 \\ (x_4-1)^2 + (x_5-8)^2 + (x_{11}-4)^2 + (x_{12}-1)^2 + (x_{10}-8)^2 = obj3 \end{array} \right\}$$

subject to :

$$\begin{aligned} 2x_1+0.5x_2-x_6+x_{11} &= 5 \\ x_1+2x_2-x_7+x_{11} &= 0 \\ x_3+0.5x_6-x_8-x_{12} &= 0 \\ 0.5x_3+x_6-x_9+x_{12} &= 0 \\ x_4+0.5x_5+0.5x_8-x_{10} &= 0 \\ 2x_4+x_5-x_8-x_{11} &= 0 \\ 3x_4-x_5+x_8-x_{12} &= 0 \\ 2x_1+x_2+2x_{11} &\le 8 \\ 2x_6+3x_{12} &\le 12 \\ 5x_4+3x_5 &\le 15 \\ 3x_4+2x_5+3x_8 &\le 12 \\ 3x_1+2x_2 &\le 13 \end{aligned}$$

and

$$x_i \ge 0, \quad i=1,2,\ldots,12$$

$$x_1 \le 2\,,\; x_2 \le 6\,,\; x_3 \le 3\,,\; x_4 \le 2\,,\; x_5 \le 4\,,\; x_6 \le 4\,,\; x_8 \le 3\,,\; x_{11} \le 3\,,\; x_{12} \le 2$$

The corresponding MPS file ("model.nl"), including the linear part, is given together with the specification file ("specs.nl") in Appendix 2.

The subroutine constrn ("constr_nl.f") (see Appendix 2) may be used to compute the nonlinear constraint functions $g(x)$ (here $f(m)$) and the corresponding elements of the Jacobian matrix $\partial g / \partial x_j$ (here $g(m,n)$). As example II does not contain nonlinear constraints the subroutine is "empty" in this case.

In subroutine objectf ("object_nl.f") the user has to insert the nonlinear objective functions $f_i(x_{nl})$ under the name $obj(i)$ as indicated in Appendix 2. The gradients are calculated automatically.

The reference point file ("rfp.nl") has the format (2X,2A4,2X,3F12.5). The first two characters in each line contain blanks, the next eight characters the name of the objective, and there are then two more blanks. The first F12.5 field contains the value of the reference point, the second a variable that can be used as a weighting coefficient and the third a control variable. In example II the value of the control coefficient is ρ=24 . The last line must contain four dots (....) as characters 5-8 in the 2A4 field.

```
rfp1    25.0        1.      24.0
rfp2    50.0        1.       1.0
rfp3    45.         1.       1.0
....
```

To start the interactive procedure in the nonlinear case, having prepared the files, the user must first initialize DIDASS. The program steps correspond to those already outlined for the linear case (see Section 4 and Figure 1). Appendix 3 contains examples of the interactive use of DIDASS in the linear and nonlinear cases.

ADDITIONAL REMARKS

The (May 83) implementation of DIDASS described in this guide is still being tested and improved. We would be glad to receive any suggestions or comments you might have concerning the system.

ACKNOWLEDGMENT

DIDASS developed from work carried out by W. Orchard-Hays, M. Kallio, A. Lewandowski and M. Grauer at IIASA. The author would like to thank T. Novachkova and Z. Fortuna for their help in structuring the system.

References

1. A. Wierzbicki, "A mathematical basis for satisficing decision making," pp. 465-485 in *Organizations: Multiple Agents with Multiple Criteria*, ed. J.N. Morse,Springer-Verlag, Berlin , New York (1981).

2. A. Lewandowski and M. Grauer, "The reference point optimization approach - methods of efficient implementation," WP-82-26, IIASA (1982).

3. M. Grauer, "Reference point optimization - the nonlinear case ," pp. 126-135 in *Essays and surveys on Multiple Criteria Decision Making*, ed. P. Hansen,Springer Verlag, New York (1983).

4. B.A. Murtagh and M.A. Saunders, "Minos/Augmented," Technical Report SOL-80-14, Systems Optimization Laboratory, Stanford University (1980).

APPENDIX 1

The MPS file ("model.l") for example I is:

```
name     hypoth.1
rows
 e  obj1
 e  obj2
 e  obj3
 l  const1
 l  const2
 l  const3
 l  const4
columns
    x1        obj1      1.5           obj3      2.5
    x1        obj2      1.2
    x1        const1    1.0           const2   -2.0
    x1        const3   -1.0
    x2        obj1      2.0           obj2      1.0
    x2        const1    2.0           const2   -1.0
    x2        const4    1.0
    x3        obj1     -1.0           obj2      1.0
    x3        obj3      1.0           const1    1.0
    x3        const3    1.0           const4    2.0
    x4        obj1      3.0           obj2      1.0
    x4        obj3      2.0           const1    1.0
    x4        const2    1.0           const4   -1.0
    x5        obj1      1.0           obj2      2.75
    x5        obj3      1.7
    x5        const1    2.0           const2    2.0
    x5        const3    2.0           const4    1.0
    x6        obj2      1.0           obj3     -1.0
    x6        const1    1.0           const4   -1.0
    x7        obj1      1.0           obj3     -1.0
    x7        const1    2.0           const2    1.0
    x7        const3   -2.0           const4   -1.0
rhs
    rhs       const1    12.25
    rhs       const2    13.75
    rhs       const3    14.0
    rhs       const4    16.5
bounds
 lo bnd        x1        0.0
 lo bnd        x2        0.0
 lo bnd        x3        0.0
 lo bnd        x4        0.0
 lo bnd        x5        0.0
 lo bnd        x6        0.0
 lo bnd        x7        0.0
endata
```

The specification file ("specs.1") for example I is:

```
begin  hypoth.1
 minimize
 objective          mocobj
 rhs                   rhs
 bounds                bnd
 rows                 1400
 columns              1500
 elements            11000
 aijtol           0.000001

 mps file                9

 crash option            1
 iterations           6000
 log frequency          50
 factorize frequency   100
 partial price           1
 solution              yes
 feasibility tol    1.0e-5

 problem number          0
endrun
```

APPENDIX 2

The MPS file ("model.nl"), including the linear part, for example II is:

```
name           theo
 rows
  e  gl1
  e  gl2
  e  gl3
  e  gl4
  e  gl5
  e  gl6
  e  gl7
  l  ugl1
  l  ugl2
  l  ugl3
  l  ugl4
  l  ugl5
 columns
    x1       gl1        2.0      gl2      1.0
    x1       ugl1       2.0      ugl5     3.0
    x2       gl1        0.5      gl2      2.0
    x2       ugl1       1.0      ugl5     2.0
    x3       gl3        1.0      gl4      0.5
    x4       gl5        1.0      gl6      2.0
    x4       gl7        3.0      ugl3     5.0
    x4       ugl4       3.0
    x5       gl5        0.5      gl6      1.0
    x5       gl7       -1.0
    x5       ugl3       3.0      ugl4     2.0
    x6       gl1       -1.0      gl3      0.5
    x6       gl4        1.0      ugl2     2.0
    x7       gl2       -1.0
    x8       gl3       -1.0      gl5      0.5
    x8       gl6       -1.0      gl7      1.0
    x8       ugl4       3.0
    x9       gl4       -1.0
    x10      gl5       -1.0
    x11      gl1        1.0      gl2      1.0
    x11      gl6       -1.0      ugl1     2.0
    x12      gl3       -1.0      gl4      1.0
    x12      gl7       -1.0      ugl2     3.0
 rhs
    rhs      gl1        5.0
    rhs      ugl1       8.0
    rhs      ugl2      12.0
    rhs      ugl3      15.0
    rhs      ugl4      12.0
    rhs      ugl5      13.0
 bounds
  up bnd      x1         2.0
  up bnd      x2         6.0
  up bnd      x3         3.0
  up bnd      x4         2.0
```

```
  up bnd        x5          4.0
  up bnd        x6          4.0
  up bnd        x8          3.0
  up bnd        x11         3.0
  up bnd        x12         2.0
  fx initial    x1          1.9
  fx initial    x2          1.6
  fx initial    x3          1.7
  fx initial    x4          0.4
  fx initial    x5          1.2
  fx initial    x6          0.6
  fx initial    x7          6.1
  fx initial    x8          1.0
  fx initial    x9          2.45
  fx initial    x10         1.5
  fx initial    x11         1.0
  fx initial    x12         1.0
endata
```

The specification file ("specs.nl") is:

```
begin     theo
 minimize
  nonlinear constraints     0
  nonlinear jacobian vars  12
  nonlinear objectiv vars  12

  objective = object
  problem no.               1

  bounds                  bnd
  rhs                     rhs
  rows                     20
  columns                  20
  elements                100
  aijtol             0.000001

 mps file                  9

  crash option              1
  iterations             1000
  solution                yes
  feasibility tol      1.0e-5
  lower bound              0.
  completion             full
  jacobian              dense

  lagrangian              yes
 major iterations          10
 minor iterations          20
  penalty parameter       0.1
```

```
   dj tolerance        1.0e-6
   row tolerance       1.0e-6
   radius of conver     0.01
   superbasics            12
   hessian dimension      12

   linesearch toler      0.1
   print level (jflxi)   101

   derivative level        2
   difference interval   1.0e-06

   call function routines when optimal

end
```

The subroutine constrn for example II is:

```
      subroutine constrn( mode,m,n,njac,x,f,g,nstate,nprob )
      implicit real*8(a-h,o-z)
      real*8 x(n),f(m),g(m,n)
      return
      end
```

The subroutine objectf for example II is:

```
      subroutine objectf( mode,n,x,f,g,nstate,nprob)
      implicit real*8(a-h,o-z)
      real*8 x(n),g(n)
      logical ityp
      character*4objnam,ipoint
      common/tables/nc,objnam(2,100),sig(100),ityp(100)
      common/err/ierr
      common/rfp/rfp(100)
      common/gamma/gam(100),obj(100),dif(100)
      common/gamma1/gam1(100),rfp1(100),sig1(100)
      common/gammau/obju(10000),objmin(100),objmax(100),w(100)
      data ipoint/'....'/
      if (nstate .ne. 1) go to 741
c
c     first entry
c
      nc=0
c     repeat
23041    continue
        nc=nc+1
        read(11,290)objnam(1,nc),rfp(nc),gam(nc),sig(nc)
        write(6,290)objnam(1,nc),rfp(nc),gam(nc),sig(nc)
        ityp(nc) = .false.
        if(.not.(objnam(1,nc).eq.ipoint)) goto 23044
```

```
      goto 280
23044     continue
23042     goto 23041
280   nc=nc-1
290   format(2x,a4,6x,3f12.5)
741   continue
c
c     normal entry
c
c     Insert the criteria functions as
c     FORTRAN-statements.
c
c     This is the example II ("theo") with quadratic
c     criteria functions and linear constraints.
c
      obj(1)=((x(1)-3)**2+(x(2)-2)**2+(x(6)-6)**2+(x(7)-4)**2)
      obj(2)=(0.5*(x(3)-4)**2+(x(8)-6)**2+(x(9)-11)**2)
      obj(3)=((x(4)-1)**2+(x(5)-8)**2+(x(11)-4)**2
     *+(x(12)-1)**2+(x(10)-8)**2)
      if (nprob .ne. 1 ) go to 720
c
c     A quadratic scalarizing function is used to
c     calculate the decision support matrix.
c
      f=0.0
      do 751 k=1,nc
         f=(gam(k)*(rfp(k)-obj(k))/sig(k))*
     :     (gam(k)*(rfp(k)-obj(k))/sig(k))+f
  751 continue
      go to 714
  720 continue
      if (nprob .ne. 2 )go to 714
      if (nstate .ne. 1)go to 103
c
c     The decision support matrix is stored for
c     future sessions.
c
295   format(f12.5)
      do 100 i=1,nc
      read (7,295)objmin(i)
100   continue
103   continue
c
c     The automatically scaled achievement variables
c     are calculated.
c
      if (nstate .ne. 1) goto 8013
      do 101 i=1,nc
      if (rfp(i) .le. objmin(i)) goto 8011
      obju(i)=.5*objmin(i)
  101 continue
      go to 8013
 8011 continue
      do 8012 i=1,nc
```

```
      obju(i)=.5*rfp(i)
 8012 continue
c
c     The logarithmic scalarizing function
c     is used.
c
 8013 continue
      rho=sig(1)
      f=.0
      s=.0
      do 102 i=1,nc
      w(i)=((obju(i)-obj(i))/(obju(i)-rfp(i)))*gam(i)
      s=s+w(i)**rho
  102 continue
      s=s/nc
      f=+(dlog(s))/rho
  714 continue
      if (nstate .ne. 2 ) return
c
c     Final entry
c
      do 802 k=1,nc
      write (10,801) obj(k)
  802 continue
  801 format (2x,f12.5)
  800 continue
      return
      end
```

APPENDIX 3

```
         didass
Enter the problem type
linear(enter lp) or nonlinear(enter nlp)
lp

         Each line of the matrix gives the
         results of the selfish optimizations.
         The diagonal represents the utopia point.

                        obj( 1)     obj( 2)     obj( 3)

extreme obj( 1)        36.75000    12.25000    24.50000

extreme obj( 2)         6.12500    16.84300    10.41200

extreme obj( 3)        18.37500    14.70000    30.62500

 You can now:
 list the names of the obj. and components of ref.pts.(enter l),
 ask for neutral solution(enter n),
 ask for plus infinite reference point(enter i+),
 ask for minus infinite reference point(enter i-),
 copy solution from last session as ref.p.(enter c),
 change the values of scal.coef.(enter scalc),
 change the values of control coef.(enter ccoef),
 change the values of ref. point components (enter rfp).
 If you wish to make no more changes,type *
 to exit from editing status

l
 obj.name  refpt.value  scal.coef.  contr.coef.
  obj1        20.0         4.00        3.00
  obj2        30.0         1.000       0.100e-06
  obj3        28.0         1.000
*
 enter name of rhs set
rhs
    3  objectives
 eps    0.100e-06
 rho      3.00
 enter name of bounds set
bnd

 eps    0.100e-06
 rho      3.00

    objective    utopia    reference    efficient        dual        scale
      names       point      point       point
   --------------------------------------------------------------------------
   obj( 1)       36.8        20.0         8.55            0.          4.00
   obj( 2)       16.8        30.0        16.4            2.72         1.000
   obj( 3)       30.6        28.0        14.4            0.288        1.000
 Do you want to run the program once more with edited input data(enter yes)
 or terminate the session (enter no)?
no
```

```
       didass
Enter the problem type
linear(enter lp) or nonlinear(enter nlp)
nlp

          Each line of the matrix gives the
          results of the selfish optimizations.
          The diagonal represents the utopia point.

                          obj( 1)      obj( 2)      obj( 3)

extreme obj( 1)          24.01900     89.69800     92.13700

extreme obj( 2)          38.56200     38.31200    108.99000

extreme obj( 3)          42.50000    128.00301     48.86200

  You can now:
  list the names of the obj. and  components of  ref.pts.(enter 1),
  ask for neutral solution(enter n),
  ask for plus infinite reference point(enter i+),
  ask for minus infinite reference point(enter i-),
  copy solution from last session as ref.p.(enter c),
  change the values of scal.coef.(enter scalc),
  change the values of control coef.(enter ccoef),
  change the values of ref. point components (enter rfp).
  If you wish to make no more changes,type *
  to exit from editing status

1
  obj.name  refpt.value  scal.coef.  contr.coef.
   obj1         25.0        1.000        24.0
   obj2         50.0        1.000       1.000
   obj3         45.0        1.000       1.000
*
  enter name of rhs set
rhs
  enter name of bounds set
bnd

      objective   utopia    reference    efficient    nadir
        names     point      point        point       point
     -------------------------------------------------------


     obj( 1)        24.0         25.0         31.6         42.5

     obj( 2)        38.3         50.0         63.7         128.

     obj( 3)        48.9         45.0         59.5         109.
  Do you want to run the program  once more with edited input data(enter yes)
  or terminate the session (enter no)?
no
```