Computer-Assisted Mathematical Programming (Modelling) System: CAMPS

C. LUCAS AND G. MITRA*

Department of Mathematics and Statistics, Brunel University, Uxbridge, Middlesex, UB8 3PH

A Computer-Assisted Mathematical Programming (Modelling) System (CAMPS) is described in this paper. The system uses program-generator techniques for model creation and contrasts with earlier approaches, which use a special-purpose language to construct models. Thus no programming skill is required to formulate a model. In designing the system we have first analysed the salient components of the mathematical programming modelling activity. A mathematical programming model is usually constructed by progressive definition of dimensions, data tables, model variables, model constraints and the matrix coefficients which connect the last two entities. Computer assistance is provided to structure the data and the resulting model in the above sequence. In addition to this novel feature and the automatic documentation facility, the system is in line with recent developments, and incorporates a friendly and flexible user interface.

Received October 1986, revised March 1987

1. INTRODUCTION AND MAJOR ISSUES

During the last thirty years algorithms and computer programs for solving optimisation problems have witnessed sustained and accelerated development. Today large-scale problems can be processed robustly and successfully, yet our ability to construct models and our understanding of many facets of these models are in contrast less developed. This issue has received the attention of many practitioners. According to Geoffrion: 'Modelling as done today is a much lower productivity process than it ought to be. It takes too long to build, verify and document models. It is too hard to maintain and make evolutionary improvements.'[11] Krabek, Sjoquist and Sommer pointed out that the steps of managing data, building the model and reporting and analysing results are much more expensive than that of optimisation.[22] Furthermore, these steps prove to be a barrier to the effective use of LP. Greenberg observed that 'comprehension is the present bottleneck in using large-scale models.'[12]

During the seventies MPSX established itself as the *de facto* standard for Linear Programming and Integer Programming (LP/IP) and as a result its input format is also accepted as the standard for specifying LP/IP model input data.[16] High-level languages such as FORTRAN and PL 1 were used to generate MPSX input files: although this is model-specific and burdensome it is still widely used. During the seventies matrix generator and report writer (MGRW) systems became well established: MAGEN,[15] GAMMA3,[33] and DATAFORM[21] are among the best known of these. These introduced flexibility and productivity in creating LP applications and are still heavily used today. The next generation of tools may be broadly classed as 'Matrix languages', and these bear more resemblance to the way a modeller would describe a problem. Early 'matrix languages' include LP MODEL,[20] MGRW,[17] MGG/RWG,[31] UIMP[27] and GAMS.[1] In contrast to special-purpose programs and MGRW systems, which require considerable understanding of the input data formats, the matrix languages require only a 'limited' knowledge of these. Fourer provides a comprehensive discussion of the major issues as seen in the early eighties.[10] Since the paper by Fourer three other systems, ULP,[36] MAGIC[5] and EXPRESS LP[4] of the same genre have been developed and reported. An alternative approach to describing LP models uses the concept of flows and flow balances in networks. LOGS is perhaps the most widely known of these systems.[3]

Application of these systems to substantial models in the corporate context brings out a number of other considerations. Murphy and Stohr[28] highlight the relevance of block structuring and block connectivity of such models, and Geoffrion has addressed the question of aggregation in considerable depth.[11] Bradley and Clemence[2] report a mathematical programming implementation (LEXICON) of the structured modelling framework of Geoffrion. The most well-known implementations of LP in corporate modelling are PLATOFORM[29] and PLANETS.[7]

During the last three years personal computers have become established, and there has been a considerable upsurge of interest in teaching/training systems for Operational Research (OR) in general and LP in particular. Of these systems LINDO is the best-established teaching system.[30] For a discussion and evaluation of a number of micro-based LP optimisers readers are referred to Ref. 32. On the model-building front there has been a strong trend towards using well-known spreadsheet systems such as LOTUS 1-2-3 and SYMPHONY.[23,19,6] These entry-level systems are excellent for training and for breaking down barriers to modelling, but their use and applicability in large structured models remain questionable.

In this paper we describe a new mathematical programming modelling system called CAMPS. It is an interactive system and comprises a set of integrated 'program generation' and data-management tools which are controlled by a series of menus and screenforms. Our design objectives are broad: the system is set out to help non-expert LP users to come to grips with the task of conceptualising and describing LP models, whereas the expert LP user is also supported in his requirements to

* To whom correspondence should be addressed.

construct large and complex models. The contents of this paper are organised as follows. Section 2 describes the salient and novel features of CAMPS, an example of model construction using CAMPS is illustrated in Section 3, the logical analysis of the modelling task and the derivation of mathematical statement are set out in Section 4. The method of automated reformulation of separable and 0–1 integer programming is considered in Section 5. The problem of Section 3 is reformulated using ULP[36] and OMNI[14] in the appendix, and contrasts our approach with these well-known systems.

2. SALIENT AND NOVEL FEATURES OF CAMPS

Computer-Assisted Mathematical Programming (Modelling) System (CAMPS) is an interactive system designed to aid model formulation, matrix generation and model management. The system comprises a set of integrated 'program generation' and data-management tools which are controlled by a series of menus and screenforms. The main menu shown in Display 2.1 together with the information flow diagram Display 2.2 provide an outline of the structure and the major functions of the system. A full user specification of the system is given in Ref. 24.

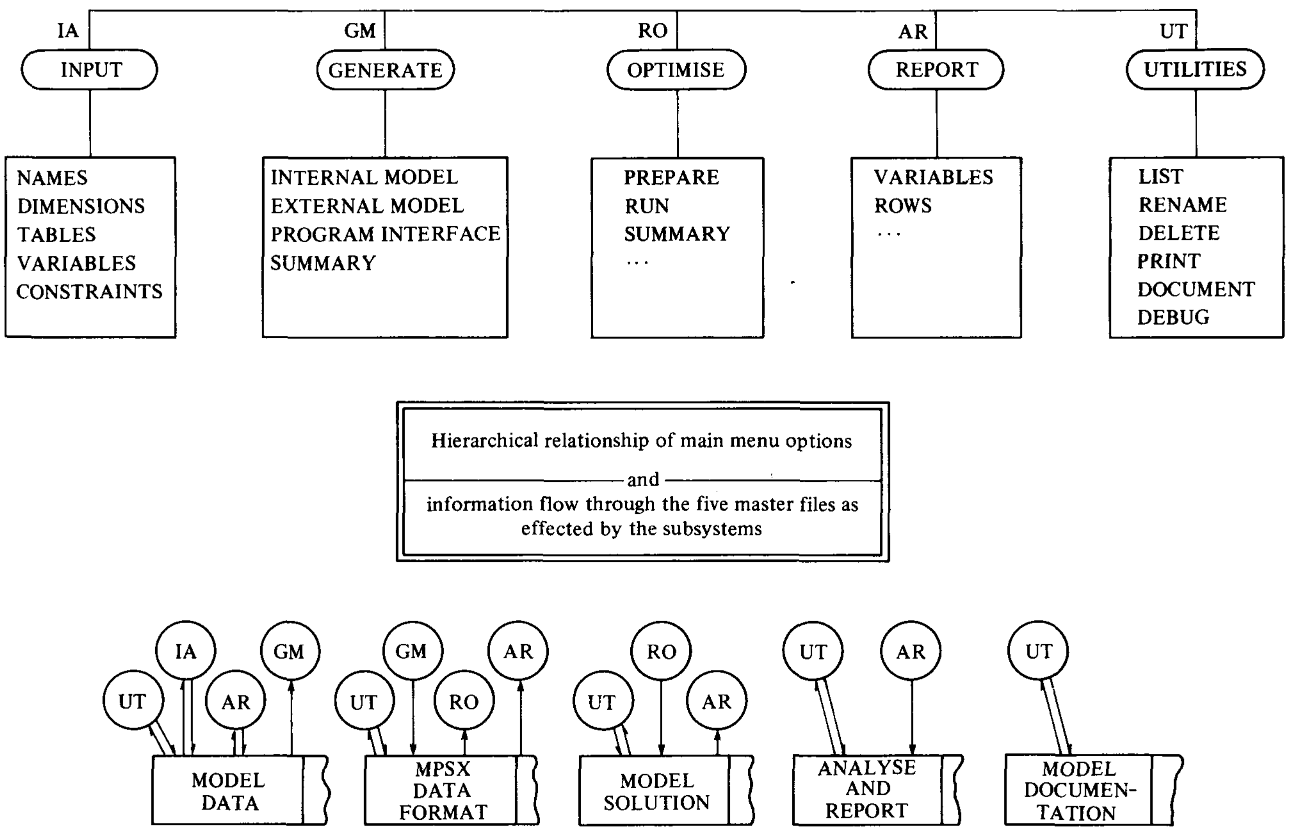
The INPUT (and AMEND) option is used to construct and/or update all aspects of a model created entirely within CAMPS. Display 2.3 illustrates the options under this subsystem and reflects the modelling methodology, which is stated succinctly as a sequence of three logical steps. Step 1: define the subscripts and their ranges (sets and dimensions). Step 2: define input data tables, model variables and model constraints, in terms of these subscripts. Step 3: specify the linear relationships in a rowwise fashion which connect the items defined in Step 2.

```
                    . . . C A M P S . . .

USER:                                                DATE:
MODEL:                                               TIME:
                       SEC: MAIN

.............................................................

                       1.INPUT
                       2.GENERATE
                       3.OPTIMISE
                       4.REPORT
                       5.UTILITIES
                       6.LOGOUT

                                     TYPE NUMBER<<    >>:
```

Display 2.1

```
                    . . . C A M P S . . .

USER:                                                DATE:
MODEL:                                               TIME:
                       SEC: INPUT

.............................................................

                       1.NAMES
                       2.DIMENSIONS
                       3.TABLES
                       4.VARIABLES
                       5.CONSTRAINTS
                       6.RETURN

                                     TYPE NUMBER<<    >>:
```

Display 2.3



Display 2.2

The subscripts correspond to 'basic entities' which are elements of 'sets', and in actual models these 'sets' could represent geographical regions, materials and time periods. This progressive approach to model definition allows us to dispense with a procedural language and replace it with an option-driven program-generator approach. The syntax of commands is captured in their context, and thus mistakes introduced by erroneous keystrokes are kept to a minimum. This is because predefined indices, sets and variables are prompted at the appropriate fields of the screenforms. For instance, at the time of defining variables and tables currently defined sets are displayed. At the time of entering the linear forms the operators (+, −, *) are prompted and a linear term is forced to comply with the dimensions of the summation indices and the row indices. We discuss this point further in the example given in Section 3.

The first four options of the main menu are designed to facilitate construction and investigation of a model whereas the fifth, the UTILITIES option, provides model-management support. In CAMPS the usual model-management functions such as DELETE, RENAME, LIST and PRINT are augmented by a further option called DOCUMENT. Tabular displays of the input data, variable (MPSX) names and row (MPSX) names, and tabulated results are essential aspects of documentation as supplied by all known systems. In addition to these a mathematical formulation of the model is also provided by CAMPS. This mathematical statement can be enhanced by textual annotations specific for a given application. These explanatory texts are introduced at the input stage.

The REPORT subsystem allows information relating to the rows, columns and reduced costs to be examined. The analysis module within REPORT is now designed to interface with the interactive model and solution analysis system ANALYZE by Greenberg.[12] For each 'basic entity' a textual annotation may be supplied and a unique two-character identifier called 'stub' is extracted out of this text.[13] This stub is used to create the 'syntax file' of ANALYZE. The OPTIMISE option uses the FORTLP system.[34] For all practical purposes this is treated as a black box, although a few algorithm control parameters can be set under this option.

LP/IP models are created in MPSX format under the GENERATE subsystem. Within the GENERATE subsystem externally created models are also accepted, but REPORT and DOCUMENT options cannot be used in this case. Whereas CAMPS itself is designed for high-level interaction in the modeller's form, at the GENERATE subsystem level a programmer's interface for model generation is also available. Thus it is possible to create MPSX models using data tables and model descriptions not held within CAMPS. In this approach the system-held subroutine library for model generation is used. We note that this approach is somehow similar to the ideas put forward by Forrest.[9] We have used this approach to create set-covering models in MPSX format.[8] These models were supplied to us in a non-standard format.

In order to deal with well-known model structures or restrictive modelling situations a compendium of reserved words has been introduced in the TABLES and ROWS section of the system. A reserved table RESTRICT with appropriate dimension is created by default as an internal table of 0–1 entries. It is used subsequently to deal with undefined entries in the primary tables. NETWORK, CONVEX and REFER are reserved row names. NETWORK is used to create a compact network model with balanced flows. CONVEX and REFER are used to achieve separable programming (set type one and set type two) model reformulation within the system.[26]

3. AN ANNOTATED EXAMPLE

In this section we consider a problem taken from the book by Jensen and Barnes.[18] This example is specially chosen as it displays the typical structure of an integrated production and distribution model. The example is also adopted by Geoffrion[11] and Bradley[2] to illustrate their systems.

The Tanglewood Manufacturing Co. has four plants located around the country. The fabrication and assembly cost per chair and the minimum and maximum monthly production for each plant are shown in Table 3.1.

Table 3.1. Fabrication cost and production restrictions by plant

Plant	Cost ($)	Production Maximum	Production Minimum
Washington	5.00	500	0
Philadelphia	7.00	750	400
Denver	3.00	1000	500
Buffalo	4.00	250	250

The company obtains the 20lb of wood required to make each chair from two suppliers who have agreed to supply any amount ordered. In return, the company guarantees the purchase of at least 8 tons of wood per month from each supplier. The cost of wood is $0.10/lb from supplier 1 and $0.075/lb from supplier 2. The shipping cost in $/lb from each supplier to each plant is shown in Table 3.2.

Table 3.2. Shipping cost from source to plant (unit cost $/lb of wood)

	Plant			
Supplier	Washington	Philadelphia	Denver	Buffalo
Ontario	0.01	0.02	0.04	0.04
Quebec	0.04	0.03	0.02	0.02

The chairs are sold in New York, Houston, San Francisco and Chicago. Transportation costs in $/chair between the cities and plants are listed in Table 3.3. Finally, Table 3.4 shows the minimum demand that must be satisfied, the maximum demand that must be satisfied and the selling price for chairs in each city.

It is desired to find the optimal production and shipment so as to maximise profit. A mathematical statement of this problem is set out below.

Table 3.3. Transportation cost between plants and cities (unit cost $/chair)

	City			
Plant	New York	Houston	San Francisco	Chicago
Washington	1.00	1.00	2.00	0.00
Philadelphia	3.00	6.00	7.00	3.00
Denver	3.00	1.00	5.00	3.00
Buffalo	8.00	2.00	1.00	4.00

Table 3.4. Selling price and demand restrictions by city

City	Selling price per chair ($)	Demand Maximum	Demand Minimum
New York	20.00	2000	500
Houston	15.00	400	100
San Francisco	20.00	1500	500
Chicago	18.00	1500	500

Subscripts and dimensions

Let

$i = 1, 2$	denote the timber merchants (suppliers),
$j = 1, 2, 3, 4$	denote the wood fabrication units (plants),
$k = 1, 2, 3, 4$	denote the chair retailers (cities).

Model coefficients (descriptors)

Let

c_j	denote the cost of producing one chair at wood plant j,
n_j	denote the minimum production of chairs at wood plant j,
q_j	denote the maximum production of chairs at wood plant j
p_k	denote the selling price of chairs at chair retailer k,
l_k	denote the minimum amount of chairs required by chair retailer k,
h_k	denote the maximum amount of chairs that can be handled by chair retailer k,
t_{jk}	denote the shipment cost between wood plant j and chair retailer k,
m_{ij}	denote the shipment cost between timber merchant i and wood plant j,
s_i	denote the cost of wood at timber merchant i,
d_i	denote the minimum order amount at timber merchant i.

Model variables

Let

z_{ij}	denote the quantity of wood bought from timber merchant i and processed in wood plant j,
y_{jk}	denote the number of chairs bought by customer chair retailer k from wood plant j.

A mathematical statement of the objective function and linear constraint relations

Maximise profit

$$= \sum_{j=1}^{4}\sum_{k=1}^{4}(p_k y_{jk} - c_j y_{jk} - t_{jk} y_{jk}) - \sum_{i=1}^{2}\sum_{j=1}^{4}(m_{ij} z_{ij} + s_i z_{ij})$$

subject to the constraints:

minimum order amount of the timber merchant i,

$$\sum_{j=1}^{4} z_{ij} \geqslant d_i \quad i = 1, 2$$

production at plant j within allowable range,

$$\left.\begin{array}{l}\sum_{k=1}^{4} y_{jk} \geqslant n_j \\ \sum_{k=1}^{4} y_{jk} \leqslant q_j\end{array}\right\} j = 1, 2, 3, 4$$

meeting customer demand at k within allowable range,

$$\left.\begin{array}{l}\sum_{j=1}^{4} y_{jk} \geqslant l_k \\ \sum_{j=1}^{4} y_{jk} \leqslant h_k\end{array}\right\} k = 1, 2, 3, 4$$

stock balance at plant j,

$$\sum_{i=1}^{2} z_{ij} - \sum_{k=1}^{4} 20 y_{jk} = 0 \quad j = 1, 2, 3, 4.$$

This problem was created using CAMPS, and descriptive names for tables and variables were used instead of one-character algebraic symbols. For example, c_j is replaced by PLNTCOST(J). Displays 3.1–3.5 provide a selection of screenforms which were used to construct the model. The method of defining names and the associated text is illustrated by the table names screenform (Display 3.1). The sets, indices and their ranges are defined as shown in Display 3.2. Displays 3.3 and 3.4 illustrate how the data tables and model variables are dimensioned. A typical model equation (the objective function) is set out in Display 3.5.

In order to illustrate the method of specifying linear forms and the interactive syntactic support (of CAMPS) which ensures consistency of dimensions, consider the

```
SEC:   NAMES SECTION                     MODEL: TANGWOOD

       TABLE NAME               TEXT

..................................................


       ..........               ..................
       .PLNTCOST.               .PLANT—COST———————.
       ..........               ..................

       PLNTCOST                 PLANT COST
       PLNTMIN                  MIN PRODUCTION
       PLNTMAX                  MAX PRODUCTION
       CUSTPRCE                 CUSTOMER PRICE
       CUSTLDMD                 MIN CUST DMND
       CUSTHDMD                 MAX CUST DMND
       TCSTPTC                  TRAN COST TO CST
       TCSTPTP                  TRAN COST FR SRC

                                HIT ALPHA COMMAND<<  >>:
```

Display 3.1

```
        SEC:   INDICES SECTION                              MODEL: TANGWOOD

   SET NAME       TEXT                INDICES        LLIM    ULIM     STEP

   1. I-      TIMBER MERCHANTS   i----------------   ---1    ---2      -1
   2. J-      WOOD-PLANTS-----   j----------------   ---1    ---4      -1
   3. K-      CHAIR RETAILERS-   k----------------   ---1    ---4      -1
   4. -       ----------------   -----------------   ----    ----      --
   5. -       ----------------   -----------------   ----    ----      --
   6. -       ----------------   -----------------   ----    ----      --
   7. -       ----------------   -----------------   ----    ----      --
   8. -       ----------------   -----------------   ----    ----      --

                                               HIT ALPHA COMMAND<<  >>:
```

Display 3.2

```
     SEC:   TABLES SECTION                          MODEL: TANGWOOD

   TABLE NAME        TEXT             TYPE           INDICES

   1. PLNTCOST    PLANT-COST------    -REAL--    j---------------
   2. PLNTMIN-    MIN-PRODUCTION--    -REAL--    j---------------
   3. PLNTMAX-    MAX-PRODUCTION--    -REAL--    j---------------
   4. CUSTPRCE    CUSTOMER-PRICE--    -REAL--    k---------------
   5. CUSTLDMD    MIN-CUST-DMND---    -REAL--    k---------------
   6. CUSTHDMD    MAX-CUST-DMND---    -REAL--    k---------------
   7. TCSTPTC-    TRAN-COST-TO-CST    -REAL--    j-,k------------
   8. TCSTSTS-    TRAN-COST-FR-SRC    -REAL--    i-,j------------

                                          HIT ALPHA COMMAND<<  >>:
```

Display 3.3

```
     SEC:   VARIABLES SECTION                          MODEL: TANGWOOD

   VARIABLE NAME        TEXT            TYPE           INDICES

   1. WOFSTP--      TIMBER-SHIPPED--    -REAL--    i-,j------------
   2. CHFPTC--      CHAIRS-SOLD-----    -REAL--    j-,k------------
   3. --------      -.--------------    -------    ----------------
   4. --------      ----------------    -------    ----------------
   5. --------      ----------------    -------    ----------------
   6. --------      ----------------    -------    ----------------
   7. --------      ----------------    -------    ----------------
   8. --------      ----------------    -------    ----------------

                                          HIT ALPHA COMMAND<<  >>:
```

Display 3.4

```
        SEC:  ROWS SECTION                                  MODEL: TANGWOOD

                              ROW NAME PROFIT

  .......................................................................

  SUM OVER j ,k      -PLNTCOST(j )*CHFPTC  (j ,k )
  SUM OVER j ,k      CUSTPRCE(k )*CHFPTC  (j ,k )
  SUM OVER j ,k      -TCSTPTC (j ,k )*CHFPTC  (j ,k )
  SUM OVER i ,j      -TCSTSTS (i ,j )*WOFSTP  (i ,j )
  SUM OVER i ,j      -SCRPRCE (i )*WOFSTP  (i ,j )
```

Display 3.5

objective function (Display 3.5) which is made up of a few (5) summation terms. For each term once the (summation) indices are chosen (out of *i, j, k*) only tables (or constants) and model variables with matching indices can be chosen to construct the term.

A mathematical statement of the problem is obtained using the documentation facility of the UTILITY subsystem and is illustrated in Displays 3.6 and 3.7. This documentation is sufficiently detailed and can be used for communication between analysts. In the linear expressions for the objective row and the constraint rows each term is annotated: a feature also found in GAMS.[1]

4. CAMPS AND LP MODELLING TACTICS

The modelling support provided by CAMPS follows closely the logical steps that a modeller goes through to formulate an LP/IP application. The first task is to consider only the modelling requirements and extract the quantitative relationships which are germane to model definition. Having identified these items a compact statement of the problem is set out with only these pertinent details.

After identifying the key components of the model the next task is to discover the underlying structure in the model. This amounts to finding a way of defining categories. The following is an illustrative list of typical categories that are found in practical problems. Number of (decentralised) geographical locations, number of planning periods, number of different products, number of grades of people, number of age groups, and so on. Within CAMPS categories are called 'basic entities'.

```
                    ****************************************
                    *                                      *
                    *  Model Documentation                 *
                    *                                      *
                    *  Prepared by     ...CLucas           *
                    *                                      *
                    *  Problem name    ...TANGWOOD         *
                    *                                      *
                    *  Date            ...07/01/86         *
                    *                                      *
                    *  Time            ...11:45            *
                    *                                      *
                    ****************************************
INDICES
-------
i  =1,   2        # .. TIMBER MERCHANTS .. #
j  =1,   4        # .. WOOD PLANTS      .. #
k  =1,   4        # .. CHAIR RETAILERS  .. #

TABLES
------
PLNTCOST(j)       # PLANT COST       ..by.. WOOD PLANTS      .. #
PLNTMIN(j)        # MIN PRODUCTION   ..by.. WOOD PLANTS      .. #
PLNTMAX(j)        # MAX PRODUCTION   ..by.. WOOD PLANTS      .. #
CUSTPRCE(k)       # CUSTOMER PRICE   ..by.. CHAIR RETAILERS  .. #
CUSTLDMD(k)       # MIN CUST DMND    ..by.. CHAIR RETAILERS  .. #
CUSTHDMD(k)       # MAX CUST DMND    ..by.. CHAIR RETAILERS  .. #
TCSTPTC(j,k)      # TRAN COST TO CST ..by.. WOOD PLANTS      ..and.. CHAIR RETAILERS  .. #
TCSTSTP(i,j)      # TRAN COST FR SRC ..by.. TIMBER MERCHANTS ..and.. WOOD PLANTS      .. #
SCRPRCE(i)        # SOURCE PRICES    ..by.. TIMBER MERCHANTS .. #
SCRLDMD(i)        # SOURCE DEMANDS   ..by.. TIMBER MERCHANTS .. #

VARIABLES
---------
WOFSTP(i,j)       # TIMBER SHIPPED   ..by.. TIMBER MERCHANTS ..and.. WOOD PLANTS      .. #
CHFPTC(j,k)       # CHAIRS SOLD      ..by.. WOOD PLANTS      ..and.. CHAIR RETAILERS  .. #

ROWS
----
WMINSRC(i)        # MIN AMT SHIPPED  ..by.. TIMBER MERCHANTS .. #
MPROD(j)          # MIN AMT PRODUCED ..by.. WOOD PLANTS      .. #
XPROD(j)          # MAX AMT PRODUCED ..by.. WOOD PLANTS -    .. #
CLOW(k)           # MIN CUST DEMAND  ..by.. CHAIR RETAILERS  .. #
THIGH(k)          # MAX CUST DEMAND  ..by.. CHAIR RETAILERS  .. #
BSTOCK(j)         # STOCK BALANCE    ..by.. WOOD PLANTS      .. #
PROFIT            # MAXIMISE PROFIT  #

CONSTRAINTS
-----------
Row name WMINSRC(i)                  # MIN AMT SHIPPED ..restriction.. #
Sum over j  [ +1.000000*WOFSTP(i,j) ]
                                     #  ..for.. TIMBER SHIPPED   #

..ge..SCRLDMD(i)                     # .. SOURCE DEMANDS    .. #

                                     For all i

Row name MPROD(j)                    # MIN AMT PRODUCED..restriction.. #

Sum over k  [ +1.000000*CHFPTC(j,k) ]
                                     #  ..for.. CHAIRS SOLD      #

..ge..PLNTMIN(j)                     # .. MIN PRODUCTION    .. #

                                     For all j
```

Display 3.6

4.1 Model variables

Once the 'basic entities' are defined the model (decision) variables or the unknowns are broadly identified. An analysis of the decision variables may also suggest new 'basic entities' at this stage. This is because the model variables are generally detailed by 'basic entities'. For the purpose of illustration a number of decision variables taken from different contexts are considered below.

Production planning. The quantity X_{pm} of a certain product p manufactured on a machine m. *Distribution planning.* The quantity X_{prn} of a product p that is shipped from a source r to an outlet n. *Inventory scheduling.* The quantity X_{pt} of a product p that is kept as closing stock at the end of a period t. *Project analysis.* Whether one should invest in project p at the beginning of time period t, or not invest in this project $Y_{pt} = 1$ or 0 may be represented by this zero-one variable Y_{pt}.

4.2 Model constraints

The constraints connect the decision variables and express the physical restrictions of the problem. By and large these are also detailed by 'basic entities'. A few representative examples of these are set out below.

Material balance equation

$$XO_t + XP_t - XC_t = D_t, \quad t = 1, 2, \ldots, T.$$

In this equation XO_t represents the opening inventory, XC_t represents the closing inventory, and XP_t the

```
Row name XPROD(j)                     # MAX AMT PRODUCED..restriction.. #

Sum over k   [ +1.000000*CHFPTC(j,k) ]
                                      #  ..for.. CHAIRS SOLD      #

..le..PLNTMAX(j)                      # .. MAX PRODUCTION    .. #

                                      For all j


Row name CLOW(k)                      # MIN CUST DEMAND ..restriction.. #

Sum over j   [ +1.000000*CHFPTC(j,k) ]
                                      #  ..for.. CHAIRS SOLD      #

..ge..CUSTLDMD(k)                     # .. MIN CUST DMND     .. #

                                      For all k


Row name THIGH(k)                     # MAX CUST DEMAND ..restriction.. #

Sum over j   [ +1.000000*CHFPTC(j,k) ]
                                      #  ..for.. CHAIRS SOLD      #

..le..CUSTHDMD(k)                     # .. MAX CUST DMND     .. #

                                      For all k


Row name PROFIT                       # MAXIMISE PROFIT ..no restriction.. #

Sum over j ,k   [ -PLNTCOST(j)*CHFPTC(j,k) ]
                                      # PLANT COST        ..for.. CHAIRS SOLD       #
Sum over j ,k   [ +CUSTPRCE(k)*CHFPTC(j,k) ]
                                      # CUSTOMER PRICE    ..for.. CHAIRS SOLD       #
Sum over j ,k   [ -TCSTPTC(j,k)*CHFPTC(j,k) ]
                                      # TRAN COST TO CST  ..for.. CHAIRS SOLD       #
Sum over i ,j   [ -TCSTSTP(i,j)*WOFSTP(i,j) ]
                                      # TRAN COST FR SRC  ..for.. TIMBER SHIPPED    #
Sum over i ,j   [ -SCRPRCE(i)*WOFSTP(i,j) ]
                                      # SOURCE PRICES     ..for.. TIMBER SHIPPED    #

..fr..0


Row name BSTOCK(j)                    # STOCK BALANCE   ..restriction.. #

Sum over i   [ +1.000000*WOFSTP(i,j) ]
                                      #  ..for.. TIMBER SHIPPED    #
Sum over k   [ -20.000000*CHFPTC(j,k) ]
                                      #  ..for.. CHAIRS SOLD      #

..eq..0


                                      For all j
```

Display 3.7

quantities to be produced. They are all decision variables pertaining to the time period t. D_t represents the customer demand for the product and is an input information.

Capacity restrictions

$$\sum_{p=1}^{P} X_{pm}\, t_{pm} \leqslant A_m, \quad m = 1, 2, \ldots, M.$$

Here $p = 1, 2, \ldots, P$ indicates the range of products which are manufactured on machines $m = 1, 2, \ldots, M$. The rate of production is indicated by t_{pm}, that is, the time taken to produce one unit of product p on machine m. A_m indicates the number of hours that machine m is available. X_{pm} is the production variable, and the constraints express the capacity of production for the machine m as limited by the number of hours of its availability.

Blending requirement

$$\sum_{c=1}^{C} X_{cp}\, b_{cr} \begin{Bmatrix} \leqslant \\ = \\ \geqslant \end{Bmatrix} Q_{pr} \begin{matrix} p = 1, \ldots, P \\ r = 1, \ldots, R \end{matrix}$$

In this case $c = 1, \ldots, C$ indicates the number of components which are used to blend $p = 1, \ldots, P$ products. The components for instance could be different crudes and products could be different types of gasoline. The range of the index $r = 1, \ldots, R$ indicates quality requirements. Typical requirements are maximum vapour pressure, minimum volatility index, etc. Thus b_{cr}, Q_{pr} are input information pertaining to linear blending rates and quality requirements respectively. X_{cp} is the decision variable indicating fractions (by volume or weight) of component c that are blended to derive product p. Thus

$$\sum_{c=1}^{C} X_{cp} = 1, \quad p = 1, \ldots, P.$$

In the discussion of the model variables and model constraints the subscripts p, m, n, c, r, t which have been introduced indicate 'basic entities', which are meaningful in the context of the model. This highlights why it is first necessary to define these 'basic entities' and then define model variables and restrictions.

5. SUPPORT FOR SEPARABLE AND LOGICAL PROGRAMMING REFORMULATION[26]

CAMPS has been designed to provide support for reformulating separable and logical (integer and fuzzy) programming problems. For instance special table types, variable types (to define special ordered sets of type 1 and type 2 variables) and row names (CONVEX*, REFER*) are used to construct separable programming problems. These facilities have been used to reformulate ten representative nonlinear optimisation problems; our investigations are reported in Ref. 25. In Ref. 26 we have shown how the analysis of bounds for linear forms plays a key role in reformulating mixed-integer, separable and fuzzy programming problems. For instance, the algebraic relations which are used to separate variables are also used to derive bounds[35] on the new variables introduced in the reformulation. These bounds are essential for piecewise linear approximation. The bounds on linear forms are also used in transforming propositions (which take logical forms) to equivalent mixed-integer linear forms. Computer support in these areas offers increasing scope and applicability of mathematical programming.

6. DISCUSSIONS

CAMPS and its underlying modelling methodology have been presented in this paper. A number of other modelling systems have command and syntax structure whereby the model description follows closely the mathematical statement of the LP. The motivation behind this approach is to force the modeller to communicate his model in a form that serves also as a full documentation. Whereas model documentation is essential, we believe it is unnecessary to tie the method by which the modeller communicates his model to the documentation requirements. In CAMPS the model is communicated and updated using menus and screenforms, and documentation is obtained under a separate option. In our experience CAMPS menus and screenforms capture a model in far fewer keystrokes than by using a modelling language. Errors introduced due to mistyping are also reduced. Our experimentations with the system suggest that reformulation support and programmer's interface are important features which should be part of any complete modeling system.

Acknowledgements

C. Lucas has developed CAMPS in part fulfilment of his Ph.D. research, in which he was supported initially by an SERC studentship and subsequently by an assistantship (grant no. GR/D/72075). We are grateful to K. Darby-Dowman and M. A. Laughton for their careful reading of earlier versions of this report and their comments. We have also greatly benefited from the comments made by H. Greenberg, who visited us during June 1986 supported by a NATO fellowship.

REFERENCES

1. J. Bisschop and A. Meeraus, *On the Development of a General Algebraic Modelling System in Strategic Planning Environment*. Mathematical Programming Study 20. North-Holland, Amsterdam (1982).
2. G. H. Bradley and R. Clemence, Implementation of a structured modeling language for optimization. Paper presented at the 12th International Symposium on Mathematical Programming, M.I.T. (1985).
3. R. W. Brown, W. D. Northup and J. F. Shapiro, LOGS: a modelling and optimization system for business planning. In *Computer Assisted Decision Making*, edited by G. Mitra. North-Holland, Amsterdam (1986).
4. Dash Associates, *XPRESS-LP* (1986).
5. R. E. Day and H. P. Williams, *MAGIC: The Design and Use of an Interactive Modelling Language for Mathematical Programming*, IMA Mathematics in Management **1** (1), 53–65 (1987).

6. Eastern Software Products, Inc., *LP88* (1984).
7. J. Lucas, Expert system/Mathematical programming applied to strategic decisions. Paper presented at TIMS XXVII, Gold Coast, Australia (1986) and runner up Franz Edelman award for Management Science achievement, TIMS (1986).
8. E. El-Darzi and G. Mitra, A collection of set covering and set partitioning test problems. Internal report, Brunel University, in preparation, 1987.
9. J. J. H. Forrest, A minimalist approach to a modelling language. Paper presented at TIMS/ORSA Joint National Meeting, Los Angeles (1986).
10. R. Fourer, Modeling languages versus matrix generators for linear programming. *ACM Trans. Math. Soft.* **9** (2), 143–183 (1983).
11. A. M. Geoffrion, *Structured Modeling*. Western Management Science Institute, Graduate School of Management, University of California, Los Angeles, CA 90024 (1985).
12. H. J. Greenberg, A functional description of ANALYZE: a computer-assisted analysis system for linear programming models. *ACM Trans. Math. Soft.* **9** (1), 18–56 (1983).
13. H. J. Greenberg, C. Lucas and G. Mitra, Computer-assisted modelling and analysis of linear programming problems: towards a unified framework. Paper presented at ORSA/TIMS, Miami (1986).
14. Haverly Systems, *Omni Linear Programming System: User and Operating Manual*, first edition (1976).
15. Haverly Systems, *MAGEN Reference Manual* (1977).
16. IBM Corporation, *Mathematical Programming System Extended, MPSX/370*, Reference Manual, SM*19-1095-1* (1976).
17. IBM Corporation, *MGRW Program Reference Manual, Program SH19-5014* (1977).
18. P. A. Jensen and J. W. Barnes, *Network Flow Programming. Wiley, New York* (1980).
19. P. A. Jensen, *MICROSOLVE/Operations Research*. Department of Mechanical Engineering, University of Texas at Austin, U.S.A. (1986).
20. S. Katz, L. J. Risman and M. Rodeh, A system for constructing linear programming models. *IBM Systems Journal* **19**, 505–520 (1980).
21. Ketron Inc., *MPSIII DATAFORM User Manual* (1980).
22. C. B. Krabek, R. J. Sjoquist and D. C. Sommer. The Apex systems: past and future. *SIGMAP Bulletin* **29**, 3–23 (1980).
23. Lotus Development Corporation, 161 First Street, Cambridge, MA 02142, *Symphony Reference Manual* (1984).
24. C. Lucas and G. Mitra, *Computer Assisted Mathematical Programming (Modelling) System: CAMPS, User Reference Manual*. Brunel University (1985).
25. C. Lucas and G. Mitra, Reformulation of nonlinear programming test problems to separable programming problems. Brunel University (1986).
26. C. Lucas, G. Mitra, J. Yadegar and K. Darby-Dowman, Linear, integer, separable and fuzzy programming problems: a unified approach towards reformulation. (To be published in the *JORS, UK*.)
27. G. Mitra and E. F. D. Ellison, User interface to mathematical programming: UIMP. *ACM Trans. Math. Soft.* **8** (3), 229–255 (1982).
28. H. M. Murphy and E. A. Stohr, *An Intelligent System for Formulating Linear Programs*. Center for Research on Information Systems, Computer Applications and Information Systems Area, Graduate School of Business Administration, New York University (1985).
29. K. H. Palmer, N. K. Boudwin, H. A. Patton, A. J. Rowland, J. D. Sammes and D. M. Smith, *A Model Management Framework for Mathematical Programming*. An Exxon monograph. John Wiley, New York (1984).
30. L. Schrage, *Linear Programming Models with LINDO*. The Scientific Press, Palo Alto, Calif. (1981).
31. Scicon Computer Services, *MGG User Guide, RWG User Guide* (1975).
32. R. Sharda, Linear programming on micro computers: a survey. *Interfaces* **14** (6), 27–38 (1984).
33. Sperry Univac, *GAMMA3, User Manual for UNIVAC 1108 Computers* (1978).
34. M. Tamiz, G. Mitra and J. Yadegar, *FORTLP: A Linear, Integer and Nonlinear Programming System, User Manual*. Brunel University (1985).
35. H. P. Williams, A reduction procedure for linear and integer programming models. In *Redundancy in Mathematical Programming*, edited S. Zionts *et al.*, pp. 87–109. Springer-Verlag, Heidelberg (1983).
36. C. Witzgall and M. McClain, *Problem and Data Specification for Linear Programs*. U.S. Department of Commerce, National Bureau of Standards, Report NBSIR 85-3125 (1985).

Two special issues of the IMA Journal of Mathematics in Management, edited by G. Mitra will be devoted to the topic of Mathematical Programming Modelling Systems. Four of the references, namely, 4, 13, 28, 36 are to appear in these volumes.

APPENDIX: A COMPARISON OF CAMPS WITH OTHER SYSTEMS

Using the sample problem of Section 3, a comparison of CAMPS' problem specification method with those of ULP and OMNI is presented here. ULP is a recently developed modelling language and incorporates many ideas also found in CAMPS. Thus the data entry which is separate from model definition follows the logical sequence whereby the sets are first defined and then the data tables. The model is then conceived in the equation form and generated using row statements. OMNI is a well-established matrix-generator system in which the linear program is specified in column sequence. The problem formulations in ULP and OMNI have not been tested but were developed by reading user manuals; however, the CAMPS formulation has been tested and the resulting model optimised.

```
TANGLEWOOD - ULP

*RANGES

MERCHANTS:ONTARIO,QUEBEC;
PLANTS:WASHINGTON,PHILADELPHIA,DENVER,BUFFALO;
RETAILERS:NEW YORK,HOUSTON,SAN FRANCISCO,CHICAGO;

*TABLES

PLANT COSTS(PLANTS): 5 7 3 4;
```

```
MIN PROD(PLANTS): 0 400 500 250;
MAX PROD(PLANTS): 500 750 1000 250;
SELL PRCE(RETAILERS):20 15 20 18;
MIN CUST DMND(RETAILERS): 500 100 500 500;
MAX CUST DMND(RETAILERS): 2000 400 1500 1500;
TRAN COST CUST(PLANTS,RETAILERS): 1.0 1.0 2.0 0.0
                                  3.0 6.0 7.0 3.0
                                  3.0 1.0 5.0 3.0
                                  8.0 2.0 1.0 4.0;
TRAN COST DLR(MERCHANTS,PLANTS): 0.01 0.02 0.04 0.04
                                 0.04 0.03 0.02 0.02;
SCR PRCE(MERCHANTS): 0.1 0.075;
SCR DMND(MERCHANTS): 8 8

UNKNOWN (X(MERCHANTS,PLANTS),Y(PLANTS,RETAILERS))
COMMENT (X(MERCHANTS,PLANTS)=AMOUNT TIMBER FROM MERCHANT TO
           PLANT)
COMMENT (Y(PLANTS,RETAILERS)=AMOUNT CHAIRS FROM PLANT TO
          RETAILER)

LPMAX  (SELL PRCE(RETAILERS)*Y(PLANTS,RETAILERS)
        -PLANT COSTS(PLANTS)*Y(PLANTS,RETAILERS)
        -TRAN COST CUST(PLANTS,RETAILERS)*Y(PLANTS,RETAILERS)
        -TRAN COST DLR(MERCHANTS,PLANTS)*X(MERCHANTS,PLANTS)
        -SCR PRCE(MERCHANTS)*X(MERCHANTS,PLANTS))

CONSTRAIN  (PLANTS:X(MERCHANTS,PLANTS)>SCR DMND(MERCHANTS))
CONSTRAIN  (RETAILERS:Y(PLANTS,RETAILERS)>MIN PROD(PLANTS))
CONSTRAIN  (RETAILERS:Y(PLANTS,RETAILERS)<MAX PROD(PLANTS))
CONSTRAIN  (PLANTS:Y(PLANTS,RETAILERS)>MIN CUST DMND(RETAILERS))
CONSTRAIN  (PLANTS:Y(PLANTS,RETAILERS)<MAX CUST DMND(RETAILERS))
CONSTRAIN  (MERCHANTS,RETAILERS:Y(PLANTS,RETAILERS)
             -20*X(MERCHANTS,PLANTS)=0)


TANGLEWOOD - OMNI

DICTIONARY

    CLASS MER          Set of timber merchants:
      ONT                Ontario
      QUE                Quebec

    CLASS PLA          Set of plants:
      WAS                Washington
      PHI                Philadelphia
      DEN                Denver
      BUF                Buffalo

    CLASS RET           Set of retailers:
      NEW                New York
      HOU                Houston
      SAN                San Francisco
      CHI                Chicago


DATA

    TABLE A             Plant costs for production of
*                        CHAIRS
          COSTS
    WAS     5
    PHI     7
    DEN     3
    BUF     4


    TABLE B              Minimum production level at each plant
          MIN
    WAS     0
    PHI   400
    DEN   500
    BUF   250
```

```
 TABLE C              Maximum production level at each plant
       MAX
 WAS   500
 PHI   750
 DEN  1000
 BUF   250


 TABLE D              Selling prices to retailers
       PRC
 NEW    20
 HOU    15
 SAN    20
 CHI    18


 TABLE E              Minimum retailer demands
       MIN
 NEW   500
 HOU   100
 SAN   500
 CHI   500


 TABLE F              Maximum retailer demands
       MAX
 NEW  2000
 HOU   400
 SAN  1500
 CHI  1500


 TABLE G                  Cost of transport from each plant to
*                                     each retailer
       NEW  HOU  SAN  CHI
 WAS   1.0  1.0  2.0  0.0
 PHI   3.0  6.0  7.0  3.0
 DEN   3.0  1.0  5.0  3.0
 BUF   8.0  2.0  1.0  4.0


 TABLE H                   Costs of transport from each
*                               merchant to each plant
       WAS  PHI  DEN  BUF
 ONT   0.01 0.02 0.04 0.04
 QUE   0.04 0.03 0.02 0.02


 TABLE I                   Costs of timber at each timber
*                                 merchant
       PCE
 ONT   0.1
 QUE   0.075


 TABLE J                    Minimum demand at each timber
*                                     merchant
       MIN
 ONT     8
 QUE     8
```

```
FORM ROW ID
*Maximise operating profit
        OBJ=OBJ
*Satisfy minimum production at plants limit
        PLN(PLA)=MIN
*Satisfy maximum production at plants limit
        PLX(PLA)=MAX
*Satisfy minimum order quantity
        MEN(PLA)=MIN
*Satisfy minimum customer demand limit
        CUN(RET)=MIN
*Satisfy maximum customer demand limit
        CUX(RET)=MAX
*Satisfy balance of wood stock at each plant
        WOB(PLA)=FIX

COLUMNS

*Shipping activity for wood from merchants
FORM VECTOR X(MER)(PLA)
*The amount of timber bought from merchant
        MEN(PLA)=1
*The amount of wood consumed in making chairs
        WOB(PLA)=-20
*The cost of buying and shipping timber
*Shipping activity for chairs from plants to retailers
FORM VECTOR Y(PLA)(RET)
*The amount of chairs produced at the plant
        PLN(PLA)=1
*The amount of chairs produced at plant
        PLX(PLA)=1
*The amount of chairs retailer buys
        CUN(RET)=1
*The amount of chairs retailer buys
        CUX(RET)=1
*Amount of chairs produced at plant
        WOB(PLA)=1
*The effective profit of selling chairs
        OBJ=TABLE D (PRC,(RET)) - TABLE A (COSTS,(PLA))
                  -TABLE G ((RET),(PLA))
RHS
FORM VECTOR RHSIDE
*Minimum plant production
        PLN(PLA)=TABLE B (MIN,(PLA))
*Maximum plant production
        PLX(PLA)=TABLE C (MAX,(PLA))
*Minimum order amount
        MEN(PLA)=TABLE J (MIN,(MER))
*Minimum customer demand
        CUN(RET)=TABLE E (MIN,(RET))
*Maximim customer demand
        CUX(RET)=TABLE F (MAX,(RET))
*Note the right hand sides for the balance rows and
*objective are zero

ENDATA
```