A PROGRAM PACKAGE FOR LINEAR MULTIPLE CRITERIA REFERENCE POINT OPTIMIZATION SHORT USER MANUAL

A. Lewandowski

August 1982
WP-82-80

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

A PROGRAM PACKAGE FOR LINEAR MULTIPLE CRITERIA REFERENCE POINT OPTIMIZATION SHORT USER MANUAL

A. Lewandowski

1. INTRODUCTION

The reference point approach has been developed by Wierzbicki and described in series of papers and reports (e.g. Wierzbicki, 1980). This method, being the generalization of goal programming developed by Geoffrion and displaced ideal method developed by Zeleny (Ching-Lai-Hwang, 1979), joins together the best properties of both approaches, eliminating simultaneously their disadvantages. In the author's opinion, the reference point approach is one of the most suitable tools for solving multiple criteria decision problems. This approach has several desirable properties:

-- it applies to convex and nonconvex cases
-- it can easily check Pareto-optimality of a given decision
-- it can be easily supplemented by an *a posteriori* computation of trade-off coefficients for the objectives
-- it is numerically well-conditioned and easy for implementation
-- the concept of reference point optimization makes it possible to take into account the desires of a decision maker directly, without necessarily asking him questions about his preferences.

Simoultaneously, with the development of the theory of reference point optimization approach, some computational works have been performed in order to determine the practical applicability of the method. Selected results are presented in Wierzbicki's papers (Wierzbicki, 1978) and Kindler et al. (1980). However, the most extensive work has been performed by Professor Orchard-Hays during his stay at IIASA. He developed a specialized linear-programming package for reference point optimization. This package, named MULTI, has been briefly described by Kallio et al. (1980). Some rather complex problems have been solved using this package; and one of these have been described by Kallio et al.

Unfortunately, this package has two basic disadvantages:

-- it has been written in DATAMAT, the input language for SESAME, a very powerful and sophisticated LP system available only on IBM 370 computers under VM/CMS operating system

-- this system has never been sufficiently documented.

For these reasons, the author decided to develop a portable package for reference point optimization which could be used with any computer equiped with a Fortran compiler and LP package working with a standard MPSX input format. Experience with this system has shown its portability and usability; it is available on a series of computers and a number of practical problems have been solved with its aid. Thus, it seems reasonable to describe this program package as a user manual. Parts of an earlier working paper by the author, written in collaboration with Kallio and Orchard-Hays, have been adapted and extended for this purpose.

2. REFERENCE POINT OPTIMIZATION

Let A be in $R^{m\times n}$, C in $R^{p\times n}$, and b in R^m and consider the multicriteria linear program (MCLP):

(MCLP.1) $\quad Cx = q$

(MCLP.2) $\quad Ax = b$

(MCLP.3) $\quad x \geq 0$,

where the decision problem is to determine an n-vector x of decision variables satisfying (MCLP.2-3) and taking into account the p-vector q of objectives defined by (MCLP.1). We will assume that each component of q is desired to be as large as possible.

An objective vector value $q = \bar{q}$ is *attainable* if there is a feasible x for which $Cx = \bar{q}$. Let q_i^*, for $i = 1,2,\ldots, p$, be the largest attainable value for q_i; i.e., $q_i^* = \sup \{q_i | q \text{ attainable}\}$. The point $q^* \equiv (q_1^*, q_2^*, \ldots, q_3^*)^T$ is the *utopia point*. If q^* is attainable, it is a solution for the decision problem. However, usually q^* is not attainable. A point $\bar{q}$ is *strictly Pareto inferior* if there is an attainable point q for which $q > \bar{q}$. If there is an attainable q for which $q \geq \bar{q}$ and the inequality is strict at least in one component, then $\bar{q}$ is *Pareto inferior*. An attainable point $\bar{q}$ is *weakly Pareto-optimal* if it is not strictly pareto inferior and it is *Pareto-optimal* if there is no attainable point q such that $q \geq \bar{q}$ with a strict inequality for at least one component. Thus, a pareto optimal point is also weakly pareto optimal, and a weakly pareto optimal point may be pareto inferior. For brevity, we shall sometimes call a pareto optimal point a *Pareto point* and the set of all such points the *Pareto set*.

What we call a *reference point* or *reference objective* is a suggestion $\bar{q}$ by the decision maker (or the group of them) reflecting in some sense a "desired level" for the objectives. According to Wierzbicki (1978), we consider for a reference point $\bar{q}$ a penalty scalarizing function $s(q-\bar{q})$ defined over the set of objective vectors q. Characterization of functions s, which result in pareto optimal (or weakly pareto optimal) minimizers of s over attainable points q is given by Wierzbicki (1979).

If we regard the function $s(q-\bar{q})$ as the "distance" between the points q and $\bar{q}$, then, intuitively, the problem of finding such a minimum point means finding among the Pareto set the *nearest* point $\hat{q}$ to the reference point $\bar{q}$. (However, as it will be clear later, our function s is not necessarily related to the usual notion of distance). Having this interpretation in mind, the use of reference point optimization may be viewed as a way of guiding a sequence $\{\hat{q}^k\}$ of pareto point generated from the sequence $\{\bar{q}^k\}$ of reference objectives. These sequences will be generated in an interactive process and such interference should result in an interesting set of attainable points $\hat{q}^k$. If the sequence $\{\hat{q}^k\}$ converges, the limit point may be seen as a solution to the decision problem.

Initial information to the decision maker may be provided by maximizing all objectives separately. Let $q^i = (q^i_j)$ be the vector of objectives obtained when the i^{th} objective is maximized for all i. Then the matrix (q^i_j), i,j, = 1,..., p, yields information on the range of numerical values of objective functions, and the vector $q^* = (q^i_i)$ is the utopia point. It should be stressed, however, that such initial information is not a necessary part of the procedure and in no sense limits the freedom of the decision maker.

We denote $w \equiv q - \bar{q}$, for brevity. Then, a practical form of the penalty scalarizing function s(w), where minimization results in a linear programming formulation, is given as follows:

$$s(w) = -\min\{\rho \min_i w_i, \sum w_i\} - \varepsilon w \quad . \tag{1}$$

Here ρ is an arbitrary penalty coefficient which is greater than or equal to p and $\varepsilon = (\varepsilon_1, \varepsilon_2, \ldots, \varepsilon_p)$ is a non-negative vector of parameters. In the special case of $\rho = p$, (1) reduces to

$$s(w) = -\rho \min_i w_i - \varepsilon w \quad . \tag{2}$$

So far in our experience, form (1) of the penalty scalarizing function has proven to be most suitable. Other practical forms have been given in Wierzbicki (1979a).

For any scalar $\hat{s}$ the set $S_{\hat{s}}(\bar{q}) \equiv \{q | s(w) \geq \hat{s}, w = q - \bar{q}\}$ is called a level set. Such sets have been illustrated for function (1) in Figure 1 for $\rho = p$, for $\rho > p$ and for a very large value for ρ. In each case, if $w \not\geq 0$, then s(w) is given by (2); i.e., the functional value is proportional to the worst component of w. If $\rho = p$, the same is true for $w \geq 0$ as well. If $w > 0$, then for large enough ρ (see the case $\rho >> p$) s(w) is given by $\sum w_i$. In the general case, when $\rho > p$, the situation is shown in the middle of Figure 1. When $w \geq 0$ and its components are close enough to each other (that is, $(\rho-1)w_1 \geq w_2$ and $(\rho-1)w_2 \geq w_1$, for $\rho = 2$), then $s(w)$ is given by $\sum w_i$. Otherwise, formula (2) applies again.

For $\varepsilon = 0$, scalarizing function (1) guarantees only weak pareto optimality for its minimizer. However, as will be shown in Lemma 1 below, if $\varepsilon > 0$, then pareto optimality will be guaranteed.

The problem of minimizing $s(q-\bar{q})$ defined by (1) over the attainable points q, can be formulated as a linear programming problem. In particular, if we again denote $w = q - \bar{q} = Cx - \bar{q}$ and introduce an auxiliary decision variable y, this minimization problem can be stated as the following problem (P):

Denote by $W \equiv \{W|-w+Cx=\bar{q},\ Ax=b,\ x\geq 0\}$ the feasible set for vector w. Then the reference point optimization problem, when the scalarizing function (1) is applied, is as follows:

$$\min_{w\in W}\{-\min\{\rho|\min_i w_i, \sum_i w_i\} - \varepsilon w\}$$

$$= \min_{w\in W}\{\max\{\max_i(-\rho w_i), -\sum_i w_i\} - \varepsilon w\}$$

$$= \min_{w\in W}\{\max\{\max_i(-\rho w_i-\varepsilon w), -\sum_i w_i - \varepsilon w\}\}$$

$$= \min_{\substack{w\in W\\ z\in R}}\{z|z\geq-\rho w_i-\varepsilon w, \text{ for all } i,\ z \geq -\sum_i w_i - \varepsilon w\}$$

$$= \min_{\substack{w\in W\\ y\in R}}\{y-\varepsilon w|-y-\rho w_i \leq 0, \text{ for all } i,\ -y-\sum_i w_i \leq 0\} \quad ,$$

where we have substituted $y = z + \varepsilon w$.

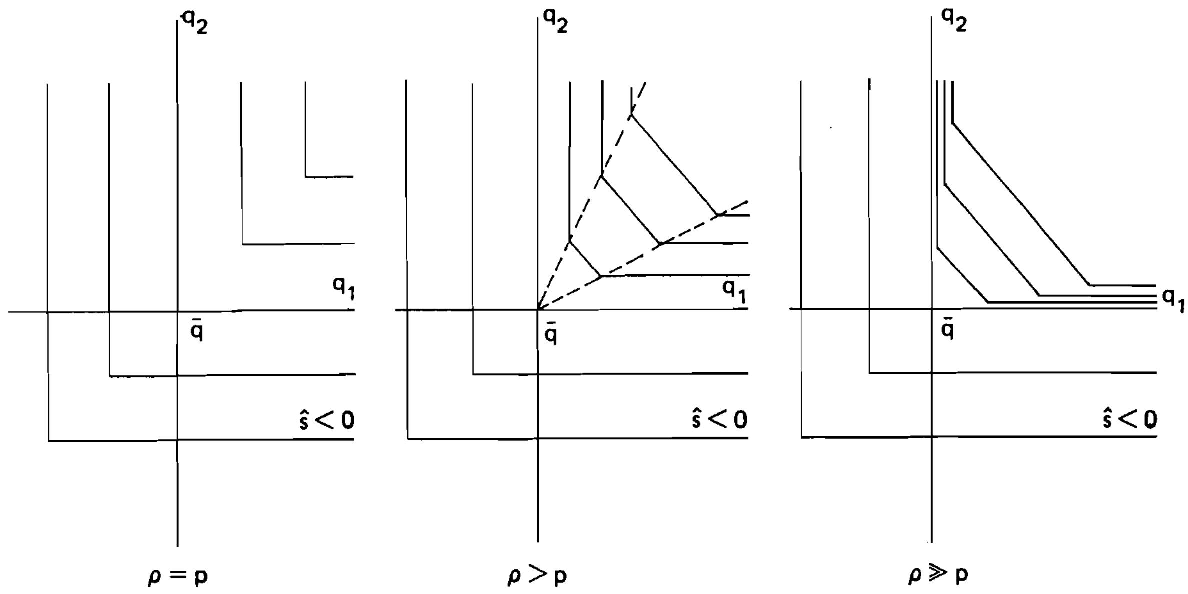

Figure 1. Level sets for penalty scalarizing functions (1) and (2) for $\varepsilon = 0$.

The optimal solution for (P) will be characterized by the following result:

LEMMA 1. *Let* $(y,w,x) = (\hat{y},\hat{w},\hat{x})$ *be an optimal solution and* δ, μ, *and* π *the corresponding dual vectors related to constraints (P.2), (P.3), and (P.4), respectively. Denote by* $\hat{q} = C\hat{x}$ *the corresponding objective vector, and by* $\hat{s} = \hat{y} - \varepsilon\hat{w}$ *the optimal value for the penalty function, and by* Q *the attainable set of objective vectors* q. *Then* $\hat{q} \in Q \cap S_{\hat{s}}(\bar{q})$ *and the hyperplane* $H = \{q \mid \mu(\hat{q}-q) = 0\}$ *separates* Q *and* $S_{\hat{s}}(\bar{q})$. *Furthermore,* $\mu \geq \varepsilon$ *and* $q = \hat{q}$ *maximizes* μq *over* $q \in Q$; *i.e.,* $\hat{q}$ *is pareto optimal if* $\varepsilon > 0$, *and* $\hat{q}$ *is weakly pareto optimal if* $\varepsilon \geq 0$.

Remark. As illustrated in Figure 2, the hyperplane H approximates the Pareto set in the neighborhood of $\hat{q}$. Thus the dual vector μ may be viewed as a vector of trade-off coefficients which tells roughly how much we have to give up in one objective in order to gain (a given small amount) in another objective. Proof of this Lemma can be found in Kallio et al (1980).

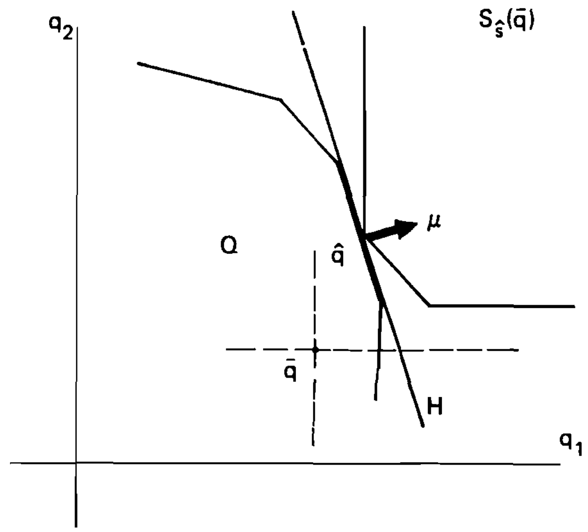

Figure 2. An illustration of Lemma 1.

3. COMPUTER IMPLEMENTATION

The basic computer implementation consists of three programs. These programs are:

-- the interactive "editor" for manipulating the reference point and the objectives (lpmod)

-- the preprocessor which converts the input model file prepared in standard MPSX format, containing the model description into its single criterion equivalent (P) (lpmulti)

-- the postprocessor, which extracts the information from the LP system output file, computes the values of the objectives and displays the necessary information (lpsol).

This concept of pre- and postprocessing of the LP problems decides, in the author's opinion, about the flexibility and portability of the system. The only machine-dependent point deals with the format of output file which differs between the different LP packages. The only adaptation needed is the modification of three FORMAT statements.

All the programs work in the interactive mode; however, the efficiency of interaction depends on the size of the LP model. The current experience shows that on the VAX with the MINOS LP system (see: MINOS System Manual), for a model of the size 150 x 100 one session*, takes about five to ten minutes, CPU time. It makes the interactive solution of rather non-trivial decision problems possible.

4.1. PROGRAM LPMOD

This program allows the user the interactive modification of the reference point components. The information flow is presented in Figure 3.

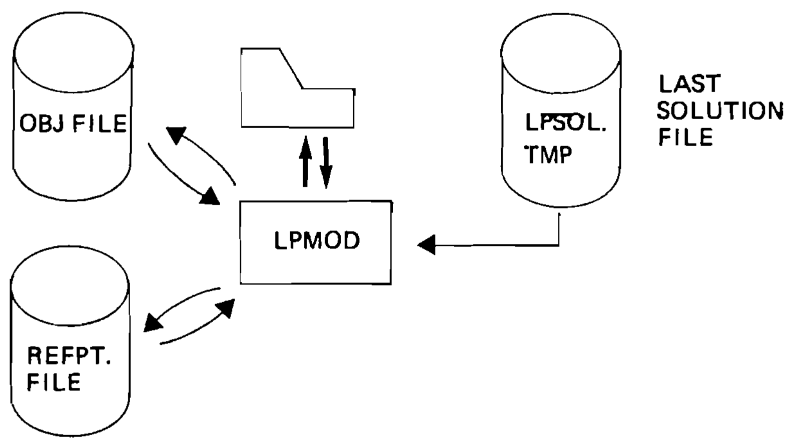

Figure 3. The information flow of lpmod program.

*By "session", we understand here the execution of lpmod, lpmulti, MINOS and lpsol programs.

In order to start work with this program on VAX under UNIX operating system, the user must prepare two files:

-- objective file containing the names of objective rows. The file format is (A4, A8, F15.0). The first four characters of each line are blanks, the next eight characters contain the name of the objective row. Each line must contain max-min indicator, +1 for maximization, -1 for minimization. The last line must contain six periods beginning in column 5.
-- reference point file, containing the reference point components, values of ρ and ε. Before the first session, this file must contain n+2 empty lines (where n - number of objectives).

In order to invoke the program, it is necessary to use the shell command

lpmod objectfilename, refptfilename.

For more details about shell commands see also: An Introduction to the C shell under UNIX, by W. Foy . Immediately after starting, the program goes to the waiting status. In this status, it is possible to use the following commands:

l -- list the names of the objectives and reference point components
n -- neutral solution - zeroise the reference point
i+ -- plus infinite reference point (10^5)
i- -- minus infinite reference point (-10^5)
c -- copy solution from last session as reference point

Immediately after execution this commands program comes back to the waiting status.

The following commands are also available:

rfp-- go to the reference point definition status
eps-- go to the ε definition status
rho-- go to the ρ definition status

In order to define the new value of ρ (or ε), it is enough to type rho (or eps); the program goes to the modification status; it is now possible to type the new value of the parameter and the program comes back to the waiting status.

Redefinition of the reference point components is possible in the rfp status. In order to do it, it is necessary to type two lines -- one containing the name of the objective row, the other containing the new value of the reference point component. The only way to exit from rfp status is through l command or the command which terminates the program.

The possible flow of control in the described program has been presented in Figure 4.

Error messages reported by the program are self-explanatory.

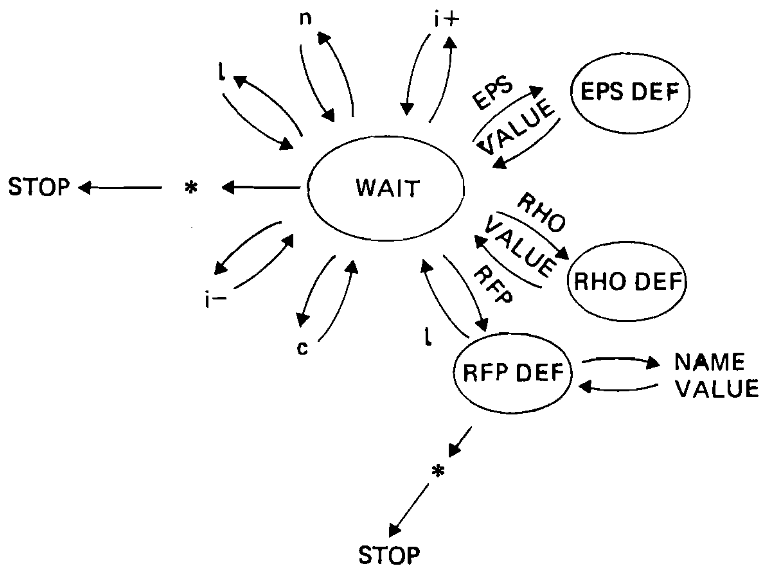

Figure 4. Control flow in lpmod program.

4.2. PROGRAM LPMULTI

This program converts the standard MPSX format input file into another file containing the single-criterion equivalent of the multiple LP problem (P). The input file should be specially prepared - the user must define the objective rows of type E (equality); they must appear as first rows in the row definition section, and in the same order as in the objective definition file (Section 4.1). The input file must contain the BOUNDS section, even if this section is empty.

The usage of this program is straightforward - it is enough to type the shell command:

lpmulti modelfilename objectfilename refptfilename

The program will ask about the name of the RHS and BOUNDS sections. As a result, it will generate a new file, named fil-9 which is the standard MINOS input file name.

The structure of information flow has been presented in Figure 5.

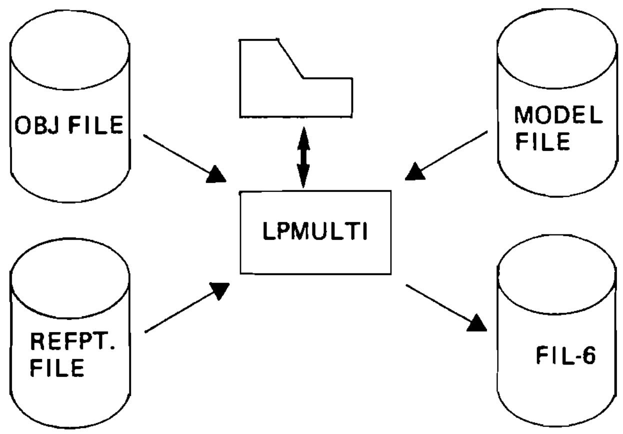

Figure 5. Information flow in lpmulti program.

4.3. RUNNING THE MINOS

After creating the single criterion LP problem, it is possible to start the LP program. Every LP package can be used, in IIASA this is currently the MINOS developed at Stanford University (Murtagh, 1980).

In order to run MINOS, it is necessary to prepare the problem specification file. This can be done in a standard way according to the MINOS User Manual. It is necessary, however, to remember the following:

-- LPMULTI adds n + 1 new columns and n + 2 new rows

-- LPMULTI creates new objective row called MOCOBJ; this name must be used in the specification file (see MINOS User Manual) and the objective must be minimized.

-- LPMULTI modifies (or creates) the BOUNDS section, the appropriate name must be used in the specification file. The same deals with the RHS section.

After running the MINOS, the standard output file FIL-6 is generated (Figure 6).

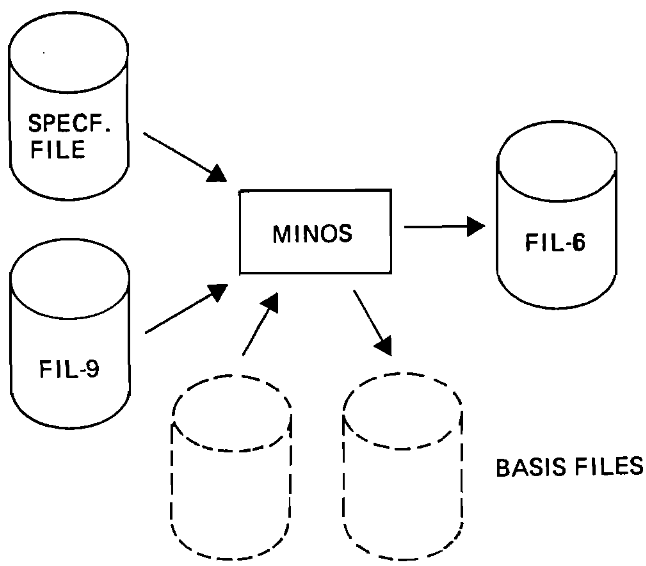

Figure 6. Information flow in MINOS program.

4.4. PROGRAM LPSOL

This program is a postprocessor which extracts all the necessary information from FIL-6 file (the standard MINOS output file) and computes the values of the objectives. The program can be activated by using the shell command:

lpsol objfilename refptfilename

The program displays the objective row names, the corresponding reference point component, the difference between the objective value and reference point component, the value of the objective and the dual variables (μ vector component according to the terminology used in Section 2). The output information is placed on the end of the file named lpoutm; this file contains the history of the session. The current solution is stored in lpsol.tmp file; this file is utilized by the lpmod program (in a case of execution of "c" command). The information flow for lpsol has been presented in Figure 7.

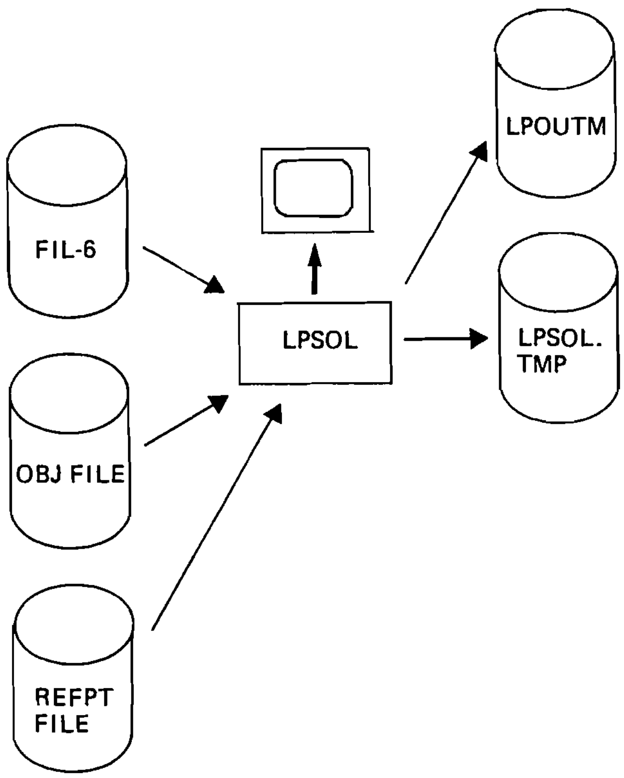

Figure 7. Information flow in lpsol program.

5. CURRENT COMPUTATIONAL EXPERIENCE

A number of problems have been solved using the package described in this paper e.g. optimization of development of the Polish chemical industry, modeling of food and agriculture section. From these experiments follows that both the approach and its implementation are sufficiently flexible and powerful to solve relatively complex, practical problems of multicriteria decision-making. From current experience, it follows that the system is very portable and can be implemented on every computer equipped with a Fortran compiler.

6. OTHER APPROACHES AND PROGRAMS

There are also some other programs being developed, or developed but not sufficiently tested, based on the modified reference point approach. One of these approaches allows the decision-maker to force or "amplify" his preferences using the penalty function technique. Namely, if the decision-maker would like to prevent the essential changes of the value of the objective in a wrong direction (too large in a case of minimization or too small in a case of maximization), he can add a penalty function to the scalarizing function.

Let J be a set of objectives for which the penalty term has been added. The modified (or nonsymmetric) has the following form (taking as a basis (r) for simplicity)

$$s(w) = -\rho \min_i w_i - \varepsilon w + \max_{i \in J} (0, -\rho_i w_i) \qquad (3)$$

This problem can be transformed to the equivalent LP

$$\min s(w) = \min_{w \in W} \{\max_i (-\rho w_i) - \varepsilon w + \max_{i \in J} (0, -\rho_i v_i)\} =$$

$$\min_{\substack{w \in W \\ y, p \in R}} \{y - \varepsilon w + p \mid y \geq -\rho w_i, \ p \geq -\rho_j w_j, \ p \geq 0, \ j \in J\} .$$

A similar collection of programs has been prepared for the solution of such problems. These programs are lqmod, lqmulti and lqsol. The only difference between the lp... and lq... package consists of a new command in the lqmod program. This is a command "pfk" which puts the program into the penalty coefficient modification status. In this status, the user can modify the ρ_i coefficient in (3), expressing the same "power" of his wishes to keep the constraints

$$v_i \geqslant 0$$

unviolated. In other cases, it is necessary to introduce the both-sided constraints for the selected objectives. Such a problem arises frequently in a case of trajectory optimization when we want to ensure the tracing of the desired (reference) trajectory. The scalarizing function has, in this case, the following form:

$$s(w) = -\rho \min_i w_i - \varepsilon w + \max_{i \in J} (0, -\rho_i w_i) + \max_{i \in M} (0, -\rho_i v_i) + \max_{i \in M} (0, \rho_i w_i),$$

where M is the set of objectives for which the both-sided constraints have been introduced. Transformation of this function into the equivalent LP problem is straightforward. This kind of problem can be solved using the lptodor program together with lqmod and lqsol (in the existing implementation set J must be empty). Some experience with these programs exist but further works on their development and testing must be performed.

REFERENCES

Ching-Lai Hwong., Abu Syad Md. Masnd. 1979. Multiple Objective Decision Making - Methods and Applications, Lecture Notes in Economics and Mathematical Systems. Vol. 164., Springer Verlog, Berlin.

Foy, W. 1980. An Introduction to the C Shell. Computer Science Division, University of California, Berkeley.

Kallio, M., Lewandowski, A., Orchard-Hays, W. 1980. An Implement of the Reference Point Approach for Multiobjective Optimization, WP-80-35. Laxenburg, Austria: International Institute for Applied Systems Analysis.

Kindler, J., Zielinski, P., de Maré, L. An Interactive Procedure for Multiobjective Analysis of Water Resources Allocation. WP-80-35. Laxenburg, Austria: International Institute for Applied Systems Analysis.

Murtagh, B.A., Saunders, M.A. 1980. MINOS/AUGMENTED, Users Manual. Systems Optimization Laboratory, Department of Operations Research, Stanford University.

Wierzbicki, A.P. 1978. On the Use of Penalty Functions in Multiobjective Optimization. Institute of Automatics, Technical University of Warsaw.

Wierzbicki, A.P. 1980. A Mathematical Basis for Satisficing Decision Making. WP-80-90. Laxenburg, Austria: International Institute for Applied Systems Analysis.