



The LBS package - a microcomputer implementation of the Light Beam Search method for the multiple-objective non-linear mathematical programming

Jaszkiewicz, A. and Slowinski, R.

IIASA Working Paper



January 1994

Jaszkiewicz, A. and Slowinski, R. (1994) The LBS package - a microcomputer implementation of the Light Beam Search method for the multiple -objective non-linear mathematical programming. IIASA Working Paper. Copyright © 1994 by the author(s). <http://pure.iiasa.ac.at/4202/>

Working Papers on work of the International Institute for Applied Systems Analysis receive only limited review. Views or opinions expressed herein do not necessarily represent those of the Institute, its National Member Organizations, or other organizations supporting the work. All rights reserved. Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage. All copies must bear this notice and the full citation on the first page. For other purposes, to republish, to post on servers or to redistribute to lists, permission must be sought by contacting repository@iiasa.ac.at

Working Paper

**The LBS package
- a microcomputer implementation
of the Light Beam Search method
for multiple-objective non-linear
mathematical programming**

Andrzej Jaszkiewicz and Roman Słowiński

WP-94-07
January 1994



International Institute for Applied Systems Analysis □ A-2361 Laxenburg □ Austria

Telephone: +43 2236 715210 □ Telex: 079 137 iiasa a □ Telefax: +43 2236 71313

**The LBS package
- a microcomputer implementation
of the Light Beam Search method
for multiple-objective non-linear
mathematical programming**

Andrzej Jaskiewicz and Roman Słowiński

WP-94-07
January 1994

Working Papers are interim reports on work of the International Institute for Applied Systems Analysis and have received only limited review. Views or opinions expressed herein do not necessarily represent those of the Institute or of its National Member Organizations.



International Institute for Applied Systems Analysis □ A-2361 Laxenburg □ Austria

Telephone: +43 2236 715210 □ Telex: 079 137 iiasa a □ Telefax: +43 2236 71313

Foreword

The research described in this Working Paper was performed at the Institute of Computing Science, Technical University of Poznan, as a part of IIASA CSA project activities on *Methodology and Techniques of Decision Analysis*.

This Working Paper documents the LBS (Light Beam Search) package aimed at interactive definition, solution and analysis of multi-objective non-linear programming problems. The methodological background of the LBS method is provided. The detailed User's manual is augmented by an example of a real-life application. Short descriptions of the two implemented solvers are also given.

The documented software is available free of charge for non-commercial applications upon request. Those requests should be addressed to the Methodology of Decision Analysis Project.

Abstract

The paper presents the LBS package which is a microcomputer implementation of the Light Beam Search method. The software has been designed to support interactive analysis of multiple-objective continuous non-linear mathematical programming problems. At the decision phase of the interactive procedure, a sample of points, composed of the current point and a number of alternative proposals, is presented to the decision maker (DM). The sample is constructed to ensure a relatively easy evaluation of the sample by the DM. To this end an outranking relation is used as a local preference model in a neighborhood of the current point. The outranking relation is used to define a sub-region of the non-dominated set where the sample presented to the DM comes from. The DM has two possibilities to move from one sub-region to another which better fits his/her preferences. The first possibility consists in specifying a new reference point which is then projected onto the non-dominated set in order to find a better non-dominated point. The second possibility consists in shifting the current point to a selected point from the sub-region. In both cases, a new sub-region is defined around the updated current point. This technique can be compared to projecting a focused beam of light from a spotlight at the reference point onto the non-dominated set; the highlighted sub-region changes when either the reference point or the point of interest in the non-dominated set are changed.

The LBS package has been implemented in Turbo Pascal within the MS-Windows environment. The package includes two versions of the LBS executable program and a set of example problems. The LBS program is composed of three modules: the problem definition module, the solver module and the interactive analysis module. The problem definition module allows for defining multiple-objective non-linear problems in a natural text form. It supports also checking the correctness of the problem definition and compilation of a problem defined in a text form to an internal format. The solver module is exchangeable and any non-linear optimizer fitting to the specified interface can be used in this module. The two versions of the LBS program differ just by the solver used. The first one, coming from the PINOKIO package, is an implementation of the Generalized Reduced Gradient method (GRG). The second one, coming from the DIDAS-N package is an implementation of the Penalty Shifting Method. The interactive analysis module makes an extensive use of computer graphics to help in the perception of a large amount of information. The graphical windows environment allows for simultaneous presentation of different kinds of information and mixing of textual, numerical and graphical forms of presentation.

Contents

1. Methodological guide	1
1.1 Introduction	1
1.2 Problem statement and basic definitions	3
1.3 Main idea of the Light Beam Search procedure	4
1.4 General scheme of the interactive procedure	6
1.5 Detailed description of particular steps	6
2. User's manual	12
2.1 Executive summary	12
2.2 Installation	12
2.3 Main menu	13
2.3.1 File submenu	14
2.3.2 Edit submenu	14
2.3.3 Analyse submenu	15
2.3.4 Light-Beam submenu	15
2.3.5 Outranking submenu	18
2.3.6 History submenu	19
2.3.7 Options submenu	19
2.3.8 About submenu	19
2.3.9 Toolbar	19
2.4 Problem definition format	20
3. Example application of the LBS method in chemical industry	22
3.1 Multiple-objective optimization of parameters of chemical reactors	22
3.2 Formulation of the example problem	23
3.2.1 Kinetic model of isomerization of ortho-xylene over H-modernite	23
3.2.2 Formulation of a multiple-objective non-linear mathematical programming problem	24
3.3 Multiple-objective analysis of the problem	26
3.4 Conclusions	29
4. Description of non-linear solvers used in the LBS package	30
4.1 Generalized Reduced Gradient method	30
4.2 Penalty Function Shifting method	31
4.3 Interface between the LBS package and a non-linear solver	33

The LBS package - a microcomputer implementation of the Light Beam Search method for multiple-objective non-linear mathematical programming

Andrzej Jaskiewicz, Roman Słowiński*

1. Methodological guide

1.1 Introduction

In the general case of multiple-objective linear and non-linear mathematical programming, the decision problem consists in selecting the best compromise solution from an infinite, multi-dimensional set of non-dominated alternatives. It is commonly acknowledged that interactive procedures are very effective in searching over the non-dominated set for the best compromise. Procedures of this type are characterized by phases of decision alternating with phases of computation. At each computation phase, a solution, or a subset of solutions, is generated for examination in the decision phase. As a result of the examination, the DM inputs some preferential information which intends to improve the proposal(s) generated in the next computation phase.

A number of interactive procedures that present to the DM one point only at each iteration, has been proposed. This class of methods includes such well-known representatives like: STEM (Benayoun et al., 1971), interactive goal programming (see e.g. Lee and Shim, 1986), the reference point method (Wierzbicki, 1980) and Pareto Race (Korhonen and Wallenius, 1988). The presentation of one solution at each iteration, however, does not give the DM the possibility to learn much about the shape of the non-dominated set. In practical situations, the preliminary preferences of the DM are often non-realistic and his/her expectations usually exceed by far attainable ranges of objectives. The DM is 'learning' about the problem during the interactive process. Wavering, incoherence and changes of DM's preferences are typical to the process. So, the more the DM learns about the non-dominated set at each iteration, the fewer steps are necessary to find a final solution and the stronger becomes the conviction of the DM that he/she has found the best compromise. Another drawback of methods from this class is that no information about a neighborhood of the current point is presented to the DM. So, the DM can miss a possibility of improving the score on one objective at a very small expense of other objectives.

There is also a class of interactive procedures that present to the DM samples of non-dominated points at each iteration. To this class belong such methods like: the Zionts-Wallenius method (Zionts and Wallenius, 1976), the Jacquet-Lagrèze, Meziani and Słowiński method (Jacquet-Lagrèze et al., 1987), as well as the Reference Direction Approach (Narula et al., 1992) which is an extension of the VIG method (Korhonen, 1987) for the non-linear case, and the Computer Graphics-Based method (Korhonen et al., 1992) which is another extension of VIG. At decision phases of such methods, the DM is usually expected to evaluate the presented solutions and specify which one is the best or rank all the

* Technical University of Poznań, Institute of Computing Science, Poznań, Poland

solutions in the sample. Authors of these methods make the assumption that evaluation of a finite sample of non-dominated points is relatively easy for the DM.

However, it follows from practical experience and theoretical results in the field of MCDA that evaluation of an even small finite sets of alternatives can be difficult for the DM. It is rather illusory to expect from the DM an explicit and complete evaluation of the alternatives if, for example, some of them are incomparable. Instead, he/she gives some preferential information upon which a global preference model can be built.

The above mentioned procedures can fail if the DM refuses to accept a substitution between objectives. Such a situation arises when objectives are in strong conflict. In this case, the DM may be simply unable to compare alternatives that are significantly different. Another type of difficulties may appear if the values of objective functions calculated for a feasible solution are uncertain for some reasons. In this case, small differences in the values of the objective functions are meaningless for the DM and alternatives that do not differ sufficiently are indifferent.

It is usually assumed that one of the four following situations can appear while comparing two alternatives a and b (Vincke, 1990):

$a P b$ i.e. a is preferred to b ,

$b P a$ i.e. b is preferred to a ,

$a I b$ i.e. a and b are indifferent,

$a ? b$ i.e. a and b are incomparable.

The *preference* P , *indifference* I and *incomparability* $?$ relations are the sets of ordered pairs (a, b) such that $a P b$, $a I b$, $a ? b$, respectively. The relations are not assumed to be transitive.

However, in order to handle situations where the DM is unable or unwilling to make distinctions between $a P b$, $a I b$ and $a ? b$, the use is recommended of a grouped relation S called an *outranking relation* (Roy, 1985): $a S b$ means that a is at least as good as b ; $a \not S b$ and $b \not S a$ means that a and b are incomparable.

In order that each particular step of an interactive procedure makes an improvement in the search for the best compromise solution, the sample of points presented to the DM for an examination should meet some requirements. Specifically, the points in the sample should not be indifferent nor incomparable. Otherwise, difficulties in evaluation of the sample can yield additional incoherence in the preferential information supplied by the DM. Moreover, in such a case, the DM can stop the interactive procedure being unable to find a better proposal among the presented points even if the current point is far from the best compromise.

The procedure presented in this paper tries to overcome the drawbacks of the above mentioned interactive procedures. Specifically,

- it uses an outranking relation as a local preference model built in a neighborhood of a current point,
- the neighborhood of the current point is composed of non-dominated points that outrank this point, so the neighborhood includes points that are sufficiently different but comparable; the points from outside the neighborhood are either incomparable or outranked by the current point,
- the sample of non-dominated points presented to the DM in each decision phase comes from the neighborhood of the current point,
- the outranking relation used to define the interesting sub-region of the non-dominated set is based on relatively weak preferential information of an inter- and intra-criteria type,
- the scanning of the non-dominated set is organized such that the sub-region moves in result of either a change of the DM's reference point or a shift of the current point within a neighborhood of this point.

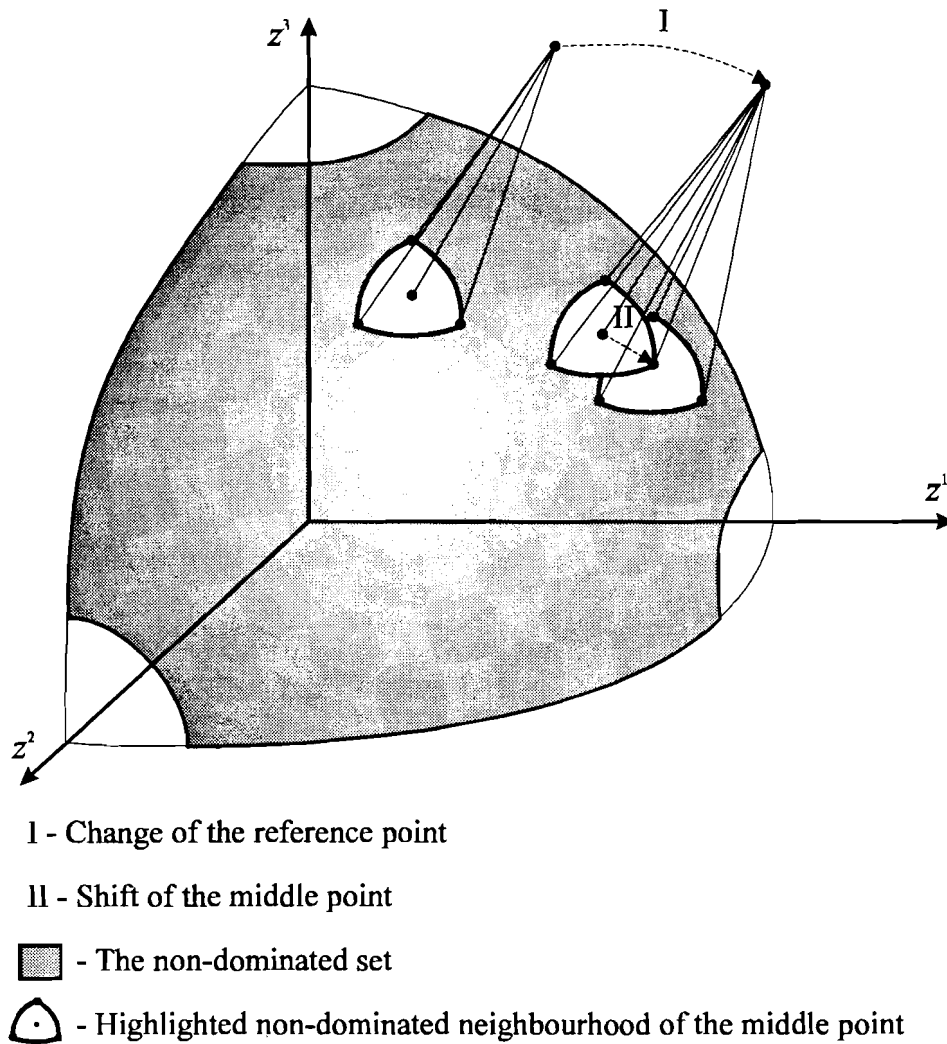


Figure 1. The Light Beam Search over a non-dominated set

The last point submits some analogy with projecting of a focused beam of light from a spotlight at the reference point onto the non-dominated set. For this reason the procedure is called the Light Beam Search or, shortly, LBS (see figure 1).

1.2 Problem statement and basic definitions

The general multiple-objective programming problem is formulated as:

$$\begin{aligned}
 & \max\{f_1(x) = z_1\} \\
 & \dots \dots \dots \\
 & \max\{f_J(x) = z_J\}
 \end{aligned}
 \tag{P1}$$

$$\text{s.t.} \\ \mathbf{x} \in D$$

where $\mathbf{x} = [x_1, \dots, x_J]$ is a vector of decision variables, functions $f_j, j=1, \dots, J$, are continuous and differentiable and condition $\mathbf{x} \in D$ can be stated using continuous and differentiable constraints.

Problem (P1) can also be stated more succinctly as:

$$\begin{aligned} & \max\{\mathbf{z}\} && \text{(P2)} \\ \text{s.t.} & \mathbf{z} \in Z \end{aligned}$$

where $\mathbf{z} = [z_1, \dots, z_J]$ is a vector of objective functions $z_j = f_j(\mathbf{x})$ and Z is an image of set D in the objective space.

Point $\mathbf{z}' \in Z$ is *non-dominated* if there is no $\mathbf{z} \in Z$ such that $z_j \geq z'_j \forall j$, and $z_i > z'_i$ for at least one i . Point $\mathbf{z}' \in Z$ is *weakly non-dominated* if there is no $\mathbf{z} \in Z$ such that $z_j > z'_j \forall j$. The set of all non-dominated points is the *non-dominated set*. For other definitions concerning non-dominance and efficiency, see e.g. Wierzbicki (1986).

The point \mathbf{z}^* composed of the best attainable objective function values is called the *ideal point*:

$$z_j^* = \max \{f_j(\mathbf{x}) \mid \mathbf{x} \in D\} \quad j=1, \dots, J.$$

Another useful definition is the *achievement scalarizing function* in the objective space:

$$s(\mathbf{z}, \Lambda, \rho) = \max_j \left\{ \lambda_j (\varepsilon_j + z_j^0 - z_j) \right\} + \rho \sum_{j=1}^J (z_j^0 - z_j) \quad (1)$$

where \mathbf{z}^0 is a reference point, $\varepsilon_j > 0$ is moderately small, $\Lambda = [\lambda_1, \dots, \lambda_J]$ is a weighting vector, $\lambda_j \geq 0$, $\sum_{j=1}^J \lambda_j = 1$ and ρ is a sufficiently small positive number.

1.3 Main idea of the Light Beam Search procedure

The LBS procedure falls into the category of interactive procedures with generation of finite samples of non-dominated points at each computation phase. A sample is composed of a current point, called the *middle point*, obtained at a previous iteration, and a number of non-dominated points from its neighborhood. In order to define the neighborhood the sample represents, an outranking relation S is used as a local preference model. Precisely, for a current middle point, the sub-region is defined as a set of non-dominated points that are not worse than the middle point, i.e. outrank the middle point. The sub-region is called the *outranking neighborhood* of the middle point. The sample is composed of points that are obtained by independent optimization of particular objectives in the outranking neighborhood, called the *characteristic neighbours* of the middle point. Moreover, the DM is able to scan more precisely the inner area of the neighborhood through the objective function trajectories between any two characteristic neighbours or between a characteristic neighbour and the middle point. Other methods for exploration of the neighborhood can also be used.

The formal expression of the conditions that must be satisfied to validate the assertion $a S b$ can be influenced by many factors. In the presented procedure, following the approaches proposed in various versions of the ELECTRE methods (Roy, 1990; Roy and Bouyssou, 1993), the following factors will be taken into account:

- the discrimination power of the DM's preferences with respect to particular objectives which will be modelled with *indifference* and *preference thresholds* (i.e. the intra-criteria information),
- the inter-criteria information which will be specified in the form of the *veto thresholds*.

Similarly to ELECTRE IV, the inter-criteria information in the form of importance coefficients, which might be too difficult to define, will not be used; it is assumed, however, that one objective is not more important than all the others together. It is worth noticing that the ratio of veto and preference thresholds of a criterion is related with its importance; the lower the ratio the greater the importance (Roy, 1980).

In the traditional preference modelling, it is assumed that every difference on a single objective z_j is significant to the DM. However, in practice, there exists an interval in which the DM does not feel any difference between two elements or refuses to accept a preference for one of the alternatives. This fact was already pointed out by Poincaré (1935 p.69), but it was Luce (1956) who introduced this fundamental feature in preference modelling. This can be modelled with the indifference threshold q_j given by the DM.

Moreover, experience shows that, usually, there is no precise value giving the limit between the indifference and preference, but there exists an intermediary region where the DM hesitates between indifference and preference or gives different answers, depending on the way he/she is questioned. This remark has led to the introduction of the preference threshold p_j . In general, the indifference and preference thresholds are functions of z_j ; moreover:

$$p_j(z_j) \geq q_j(z_j) \geq 0$$

The indifference and preference thresholds allow to distinguish between the three following preference relations with respect to z_j for any ordered pair (a, b) of alternatives:

$$\begin{aligned} a I_j b \quad \text{i.e.} \quad & a \text{ and } b \text{ are equivalent} & \Leftrightarrow & -q_j(z_j^a) \leq z_j^a - z_j^b \leq q_j(z_j^b), \\ a Q_j b \quad \text{i.e.} \quad & a \text{ is weakly preferred to } b & \Leftrightarrow & q_j(z_j^a) < z_j^a - z_j^b < p_j(z_j^b), \\ a P_j b \quad \text{i.e.} \quad & a \text{ is significantly preferred to } b & \Leftrightarrow & p_j(z_j^a) \leq z_j^a - z_j^b. \end{aligned}$$

The veto threshold v_j allows to take into account the possible difficulties of comparing the relative value of two alternatives when one is significantly better than the other on a subset of objectives, but much worse on at least one other objective. In general, the veto threshold is also a function of z_j .

The outranking relation has already been used as a preference model in the Cone Contraction Method with Visual Interaction for Multiple-Objective Non-Linear Programmes (Jaszkievicz and Słowiński, 1992a). In that method, however, it is used as global preference model. The construction of an outranking relation follows the methodology proposed for the ELECTRE III method (Roy, 1978) and the relation is built on a representative sample of non-dominated points. As the indifference, preference and veto thresholds, in general, depend on z_j , the DM should specify these thresholds in the form of mathematical functions, $q_j(z_j)$, $p_j(z_j)$ and $v_j(z_j)$. If the functions are complicated, it is practically impossible for the DM to

specify them explicitly. In the Light Beam Search procedure the outranking relation is used as a local preference model in the neighborhood of a middle point, so a single value of each threshold is sufficient for a given middle point. Of course, the DM can update the values of the thresholds for every new middle point.

The outranking relation has also been used as a local preference model in the method proposed by Lotfi et al. (1992). However, their method has been developed for multiple objective analysis of problems with finite set of alternatives only. In this case, the whole neighborhood can be generated and presented to the DM. Moreover, as the authors do not use any additional preferential information, the definition of the neighborhood seems somewhat arbitrary.

1.4 General scheme of the interactive procedure

The following is a general scheme of the proposed procedure presented in a Pascal-like form:

```

Fix the points of the best and the worst values of objectives; make the former one the
first reference point;
Ask the DM to specify the preferential information of inter- and intra-criteria type;
Find a starting middle point;
repeat
    Present the middle point to the DM;
    Calculate the characteristic neighbours of the middle point and present them to
    the DM;
    Allow the DM to scan the inner area of the current neighborhood;
    if the DM wants to store the middle point then
        Add it to the set of stored points;
    case
        The DM wants to define a new reference point:
            Ask the DM to specify the aspiration levels on particular
            objectives;
            Project the reference point onto the non-dominated set;
        The DM wants a point from the neighborhood to be the new middle
point:
            Ask the DM to select the new middle point;
        The DM wants to return to one of the stored points:
            Use the stored point as a new middle point;
        The DM wants to update the preferential information:
            Ask the DM to specify the new preferential information;
    end
until the DM feels satisfied with a point found during the interactive process;

```

1.5 Detailed description of particular steps

The procedure starts by asking the DM to specify (subjective) best and worst values of objectives, z_j^* , z_j^* ($j = 1, \dots, J$), respectively. If he/she is unable to do so, the best values are fixed at individual maxima of particular objectives (ideal point) and the worst values are set equal to minimal values of objectives at the points corresponding to the individual maxima. The point of the best values \mathbf{z}^* becomes the first reference point, \mathbf{z}^0 .

Then, the DM is asked to give the preferential information for each objective, i.e. the indifference and, optionally, the preference and veto threshold. At this stage the DM should decide if he/she wants to specify the preference and/or veto thresholds, however, he/she is able to change these settings at every step of the procedure.

In the next step, the starting middle point \mathbf{z}^c is computed. The point is obtained by projecting point \mathbf{z}^* of the best values of objectives onto the non-dominated set in the direction defined by point \mathbf{z}^* and point \mathbf{z}_* of the worst values of the objectives. The achievement scalarizing function (1) is used to this end.

Then, the *characteristic neighbours* of the middle point are computed. The characteristic neighbour, with respect to objective z_j is a point \mathbf{z}^j from the outranking neighborhood of point \mathbf{z}^c that maximizes the distance from \mathbf{z}^c in the direction of the greatest locally feasible improvement of objective z_j ($j = 1, \dots, J$). An *attainable characteristic neighbour* \mathbf{z}^j is a point obtained as result of a projection of point \mathbf{z}^j onto the non-dominated set ($j = 1, \dots, J$).

In order to test if a point \mathbf{z} outranks the middle point, first, the following numbers are calculated:

- $m_s(\mathbf{z}, \mathbf{z}^c)$ - the number of the objectives for which point \mathbf{z} is indifferent, or weakly or strictly preferred, to \mathbf{z}^c ,
- $m_q(\mathbf{z}^c, \mathbf{z})$ - the number of the objectives for which point \mathbf{z}^c is weakly preferred to \mathbf{z} ,
- $m_p(\mathbf{z}^c, \mathbf{z})$ - the number of the objectives for which point \mathbf{z}^c is strictly preferred to \mathbf{z} ,
- $m_v(\mathbf{z}^c, \mathbf{z})$ - the number of the objectives being in a strong opposition to the assertion $\mathbf{z} S \mathbf{z}^c$, i.e. $\text{card} \{j: z_j^c - v_j \geq z_j, j = 1, \dots, J\}$.

The construction of the outranking relation depends on the type of preferential information supplied by the DM. If the DM has specified all the thresholds, the following definition of the outranking relation is proposed:

$$\mathbf{z} S^a \mathbf{z}^c \Leftrightarrow \begin{cases} m_v(\mathbf{z}^c, \mathbf{z}) = 0 \text{ and} \\ m_p(\mathbf{z}^c, \mathbf{z}) \leq 1 \text{ and} \\ m_q(\mathbf{z}^c, \mathbf{z}) + m_p(\mathbf{z}^c, \mathbf{z}) \leq m_s(\mathbf{z}, \mathbf{z}^c) \end{cases}$$

If the DM has decided not to specify the veto thresholds, one should not assume that for every objective he/she is ready to accept any worsening of the objective even if a subset of other objectives is significantly improved, i.e. one should not assume that the veto threshold does not exist. Such a situation indicates that at the particular stage of the interactive process, the DM is unable or unwilling to specify the value of this threshold explicitly. In this case, the following definition of the outranking relation is proposed:

$$\mathbf{z} S^b \mathbf{z}^c \Leftrightarrow \begin{cases} m_p(\mathbf{z}^c, \mathbf{z}) = 0 \text{ and} \\ m_q(\mathbf{z}^c, \mathbf{z}) \leq m_s(\mathbf{z}, \mathbf{z}^c) \end{cases}$$

Let us observe that $S^b \subseteq S^a$ and that $S^b = S^a$ if $v_j = p_j \forall j$. If, for an objective z_j , the DM is ready to accept the worsening of its value greater than p_j , some points that outrank the middle point can be left outside the outranking neighborhood. However, the neighborhood will be still composed of points that are comparable with the middle point.

In a similar way one can analyze the situation when the DM has decided not to specify the preference threshold. In this case one should not assume that the DM feels no difference

between the weak and strict preferences. Such a situation indicates that the DM is unable or unwilling to specify explicitly the value allowing to distinguish between the two relations. In this case, the following definition of the outranking relation is proposed:

$$z S^c z^c \Leftrightarrow \begin{cases} m_v(z^c, z) = 0 \text{ and} \\ m_q(z^c, z) \leq 1 \end{cases}$$

Observe that $S^c \subseteq S^a$ and that $S^c = S^a$ if $p_j = q_j \forall j$.

Finally, if the DM has decided to specify the preference thresholds only, the following definition of the outranking relation is proposed:

$$z S^d z^c \Leftrightarrow m_q(z^c, z) = 0$$

Observe that $S^d \subseteq S^b \subseteq S^a$ and $S^d \subseteq S^c \subseteq S^a$, moreover, $S^d = S^a$ if $v_j = p_j = q_j \forall j$.

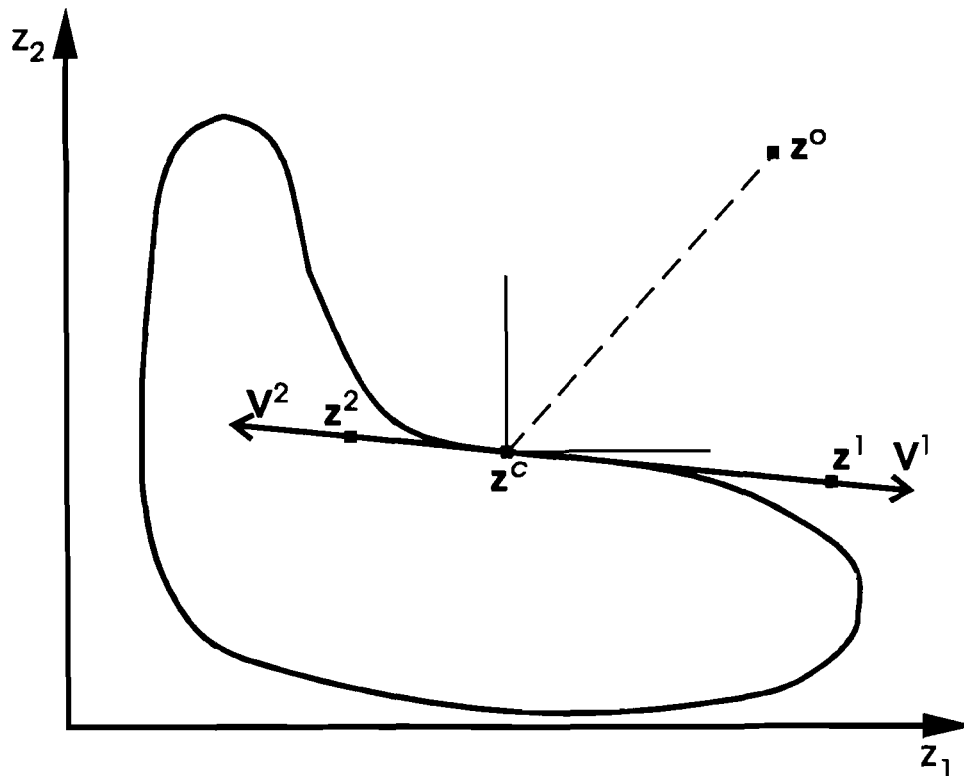


Figure 2. Characteristic neighbours found using a gradient projection onto a linear approximation of active constraints in z^c

In order to find the J characteristic neighbours, gradients of particular objectives are projected onto a linear approximation of the constraints which are active in point z^c (cf.

gradient projection methods for non-linear optimization, Rosen, 1960). Let H be the number of active constraints in point z^c . The linear constraints can be presented in a matrix form:

$$Ax = b$$

where $z_j^c = f_j(x^c)$, $j = 1, \dots, J$, $a_{hi} = \partial c_h / \partial x_i$ are elements of matrix A , t is an index of an active constraint, $h = 1, \dots, H$; $i = 1, \dots, I$. Next, the projection matrix P is calculated:

$$P = I - A^T(AA^T)^{-1}A$$

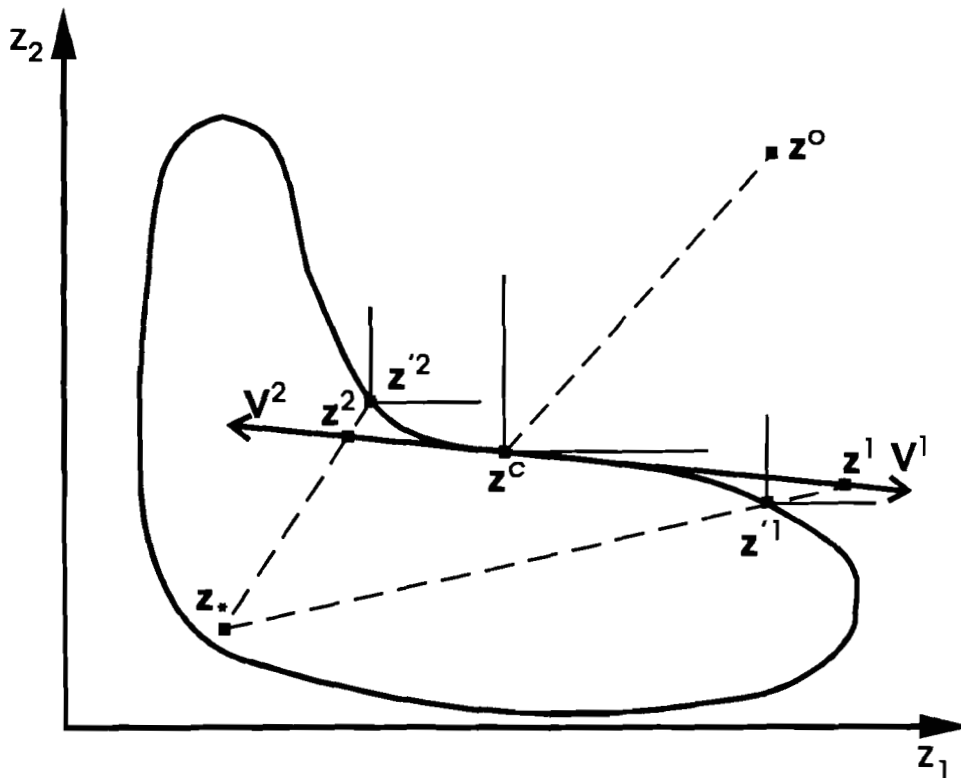


Figure 3. Characteristic neighbours found by projecting points z^1 and z^2 onto the non-dominated set

Matrix P and gradients of particular objectives $\nabla_x f_j$ are used to obtain directions Δx^j in the space of variables:

$$\Delta x^j = P \nabla_x f_j, \quad j = 1, \dots, J$$

$\Delta \mathbf{x}^j$ is the feasible direction of the greatest improvement of objective $z_j = f_j(\mathbf{x})$. Directions $\Delta \mathbf{x}^j$ are used in turn to define corresponding directions \mathbf{V}^j in the objective space:

$$\mathbf{V}^j = \left[\sum_{i=1}^I (\Delta x_i^j \partial f_1 / \partial x_i), \dots, \sum_{i=1}^I (\Delta x_i^j \partial f_j / \partial x_i) \right]$$

Then, the following mathematical programming problem is solved in order to maximize objective z_j in direction \mathbf{V}^j ($j = 1, \dots, J$):

$$\begin{array}{ll} \text{(P3)} & \max \alpha \\ & \text{s.t.} \quad \mathbf{z}^j \in \mathbf{z}^c, \quad \mathbf{z}^j = \mathbf{z}^c + \alpha \mathbf{V}^j \\ & \quad \alpha \geq 0 \end{array}$$

Problem (P3) is a small mathematical programming problem with one variable only. The points, \mathbf{z}^j ($j = 1, \dots, J$) obtained by solving the J problems (P3) give characteristic neighbours (see figure 2).

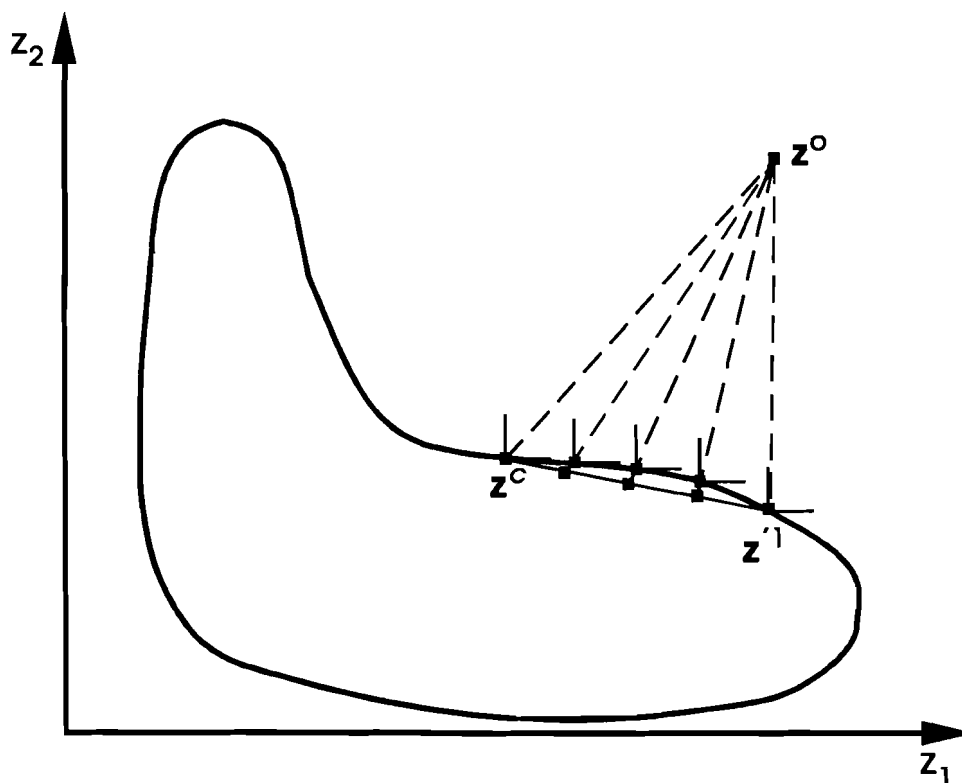


Figure 4. Finding an approximation of a profile of the non-dominated set

Attainable characteristic neighbours are obtained as result of projection of the points z^j ($j = 1, \dots, J$) onto the non-dominated set in the direction connecting z^j with point z^* (see figure 3).

In the decision phase, the middle point and its characteristic neighbours are presented to the DM. Both numerical and graphical forms of presentation should be used to help the DM in evaluating large amounts of information. Moreover, the DM is able to scan more precisely the region between any two characteristic neighbours or between a characteristic neighbour and the middle point. For this purpose, the line segment connecting the points in the objective space is projected onto the non-dominated set. The obtained subset of the non-dominated points is called the *profile* of the neighborhood. As in the non-linear case getting a continuous profile is practically impossible, a finite numbers of points lying on the line segment is chosen and they are projected onto the non-dominated set (see figure 4). The points resulting from the projection are then presented to the DM. A similar technique of scanning a sub-region of the non-dominated set has been used in Jaszkievicz and Słowiński (1992a). Some other techniques of local characterization of the non-dominated set can also be used at this step.

The procedure stops if one of the presented points is satisfactory to the DM on all objectives. Otherwise, he/she can continue the scanning using two degrees of freedom. The first degree consists in modifying the aspiration levels, i.e. the reference point. The new reference point is then projected onto the non-dominated set in order to find the new middle point. The second degree of freedom consists in selecting one of the points from the neighborhood to be the new middle point for the same reference point. Then a new outranking neighborhood is generated (see figure 1).

Before continuing the scanning, the DM can store the current middle point. He/she is allowed to go back to any of the stored points at any time.

Finally, the DM is able to modify the preferential information given for each objective, i.e. the indifference, preference and veto thresholds. He/she can also change the type of the outranking relation. It influences the construction of the outranking relation and the size of the new outranking neighborhood.

2. User's manual

2.1 Executive summary

The LBS package is a full implementation of the Light Beam search method for the MS-Windows environment. It supports the following general functions:

- the definition and edition of a source model in the form of a multiple-objective non-linear programming problem,
- interactive analysis of the problem, with a user-friendly graphical and numerical representation of generated solutions.

There are two versions of the LBS program. They differ by the non-linear solver used in the solver module. The first one is a solver developed in the Institute of Computing Science, Technical University of Poznań, which is an improved version of the solver used in the PINOKIO package (Jaszkievicz and Słowiński, 1992b). The solver implements the Generalized Reduced Gradient method (GRG) (Abadie, 1977). This version of the program is contained in the LBS.EXE file. The second version uses a solver developed for the DIDAS-N package (Kręglewski et al., 1991) in the Institute of Automatic Control, Warsaw University of Technology. The solver implements the Penalty Shifting Method (Wierzbicki, 1971). This version of the program is contained in the LBSD.EXE file.

The hardware requirements of the LBS package are the same as the requirements of MS-Windows 3.x. LBS will run on any PC that can run MS-Windows.

LBS is a standard Windows application and working with it is similar to working with other windows applications. In the user's manual it is assumed that the user is familiar with working under MS-Windows.

2.2 Installation

MS-Windows 3.x must be installed before running LBS. The LBS package can be run from a floppy disk. It is advised, however, to install it on a hard disk (network drive). To install LBS on a hard disk make the following steps:

- create a new directory, e.g. LBS,
- copy all the files from the distribution floppy disk to this directory; if you are not interested in the version using the GRG solver do not copy file LBS.EXE; if you are not interested in the version using the DIDAS-N solver do not copy file LBSD.EXE,
- under PROGRAM MANAGER select command FILE|NEW; select PROGRAM GROUP radio button and press button OK; the PROGRAM GROUP DESCRIPTION dialog appears on the screen; type LBS in the DESCRIPTION field of this dialog and press button OK; a new program group called LBS appears on the screen;
- under PROGRAM MANAGER select command FILE|NEW again; select PROGRAM ITEM radio button and press button OK; the PROGRAM GROUP DESCRIPTION dialog appears on the screen; type LBS in the DESCRIPTION field of this dialog and full path to the LBS.EXE or LBSD.EXE file in the COMMAND LINE field of this dialog; the LBS icon appears in the LBS program group.

To run LBS doubly click on the LBS icon under PROGRAM MANAGER. After running it the invitation screen is displayed (see figure 5).

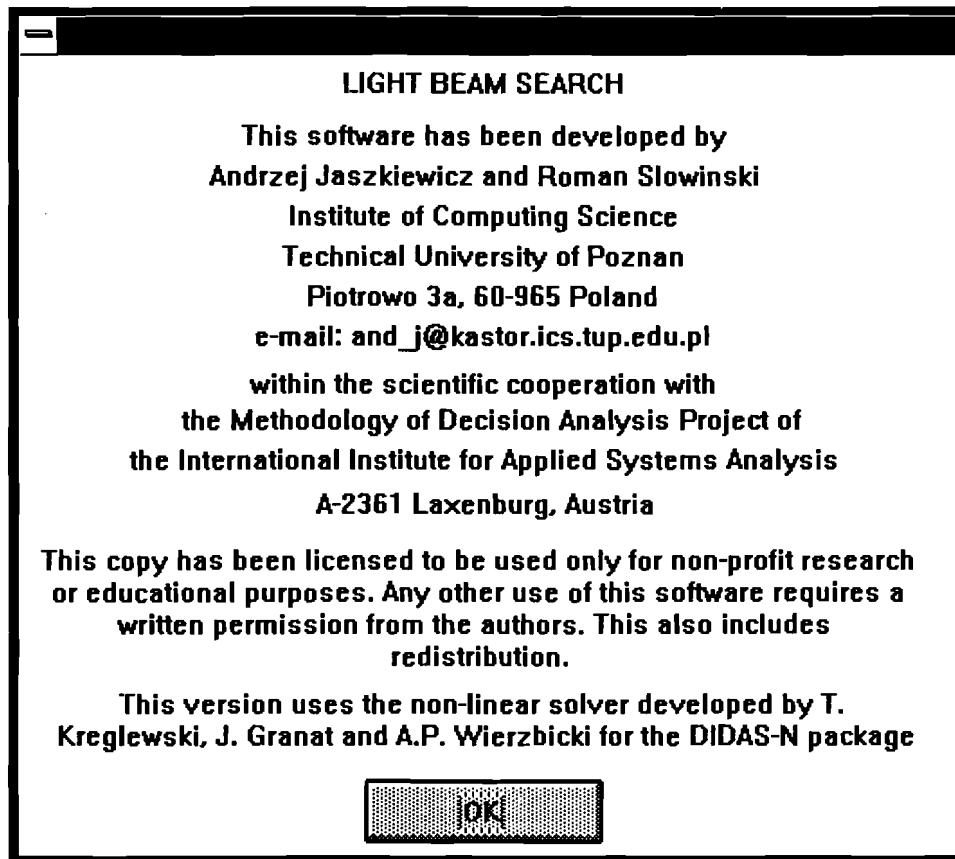


Figure 5. Invitation screen

2.3 Main menu

The main menu of LBS is presented in figure 6.

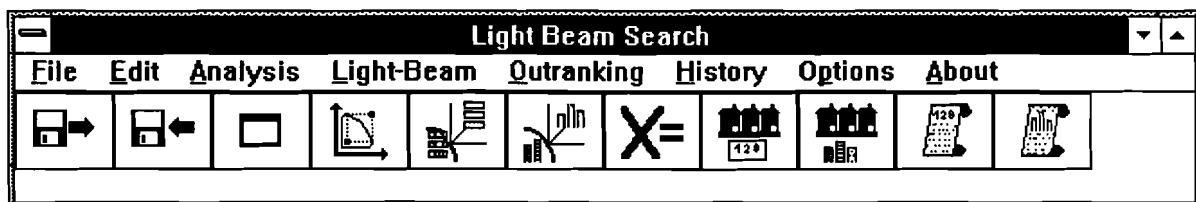


Figure 6. Main menu

Under the main menu a toolbar is placed which allows for a quick access to the most frequently used functions. The main menu is composed of the following submenus:

File

Includes commands for saving and loading problem definitions, printing a report about current solution and exiting LBS.

Edit

Contains the standard editing functions typical for windows programs, functions for exchanging data via clipboard and for searching and replacing text.

Analysis

Includes commands for compiling problem definition and for finding the best attainable values of particular objectives.

Light-Beam

This menu provides basic function of the Light Beam Search procedure - displaying the middle point, defining the reference point, moving the middle point, displaying characteristic neighbours and displaying profiles of a neighborhood.

Outranking

Allows the DM to select the type of outranking relation and to define the values of particular thresholds.

History

Allows for displaying of previously saved non-dominated points.

Options

Contains functions for setting some parameters of the software.

About

Displays information about the software.

2.3.1 File submenu**New**

Creates a new window in which a problem definition is edited. If another window with a problem definition is already open, it is closed before the new window is created.

Open

Opens a text file with a problem definition. After selecting this command a standard windows FILE OPEN dialog appears on the screen. The default extensions of a file to be open is *.TSK.

Save

Saves the problem being edited to a text file. If the problem has no given name, a standard windows SAVE FILE AS dialog appears on the screen. The default extensions of a file to be saved is *.TSK.

Save as

Saves the problem being edited under a given name. After selecting this command, a standard windows SAVE FILE AS dialog appears on the screen. The default extensions of a file to be saved is *.TSK.

Report

Prints a report about the current solution on a printer. The report includes current values at the current point, values of decision variables, values of definition and the numbers of active constraints.

Exit

Exits the LBS.

2.3.2 Edit submenu**Undo**

Undoes the recently made editing operation in problem definition window.

Cut, Copy, Paste, Delete, Clear All

These commands perform typical windows editing functions: deleting the selected text, placing the selected text in the clipboard, placing the text from clipboard in the edit window, deleting and placing the selected text in the clipboard and clearing the whole problem definition, respectively.

Find

Finds a text specified by the user in the problem definition window.

Replace

Replaces a text specified by the user by another text in the problem definition window.

Next

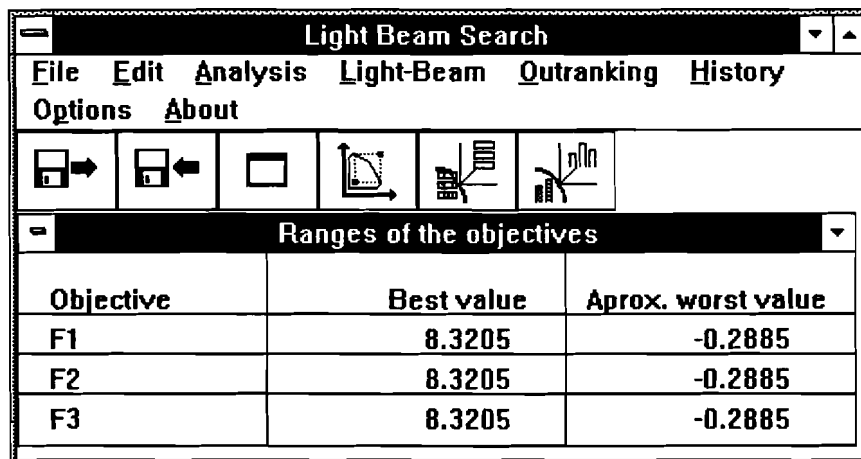
Repeats the recently performed **Find** or **Replace** operation.

2.3.3 Analyse submenu**Compile**

This command checks the correctness of the problem definition. If there is an error in the problem definition, the text cursor is placed at the error position and a short description of the error is displayed on the screen. Otherwise, a window containing the number of definitions, the number of objectives, the number of variables and the number of constraints is displayed. If the problem definition is correct the definition is translated into an internal form which accelerates the further calculations.

Ranges of the objectives

This command performs independent optimization of particular objectives. This command displays on the screen RANGES OF THE OBJECTIVES window (see figure 7). In column BEST VALUE, the best values of particular objectives are displayed. In column APPROXIMATED WORST VALUES, the worst values of particular objectives found during the independent optimization are displayed.



Objective	Best value	Aprox. worst value
F1	8.3205	-0.2885
F2	8.3205	-0.2885
F3	8.3205	-0.2885

Figure 7. RANGES OF THE OBJECTIVES window

2.3.4 Light-Beam submenu**Numerical**

This command displays the LIGHT BEAM SEARCH (NUMERICAL) window (see figure 8). In this window, the middle point and the reference point are presented in the numerical form. Values in column REFERENCE POINT can be changed by the user. By pressing button GO the user projects the reference point onto the non-dominated set. By pressing button SAVE the user can save the middle point (adds it to the set of stored points).

Button M->R makes the middle point the new reference point (places values from column MIDDLE POINT in column REFERENCE POINT).

All changes in LIGHT BEAM SEARCH (NUMERICAL) window are simultaneously made in LIGHT BEAM SEARCH (GRAPHICAL) window.

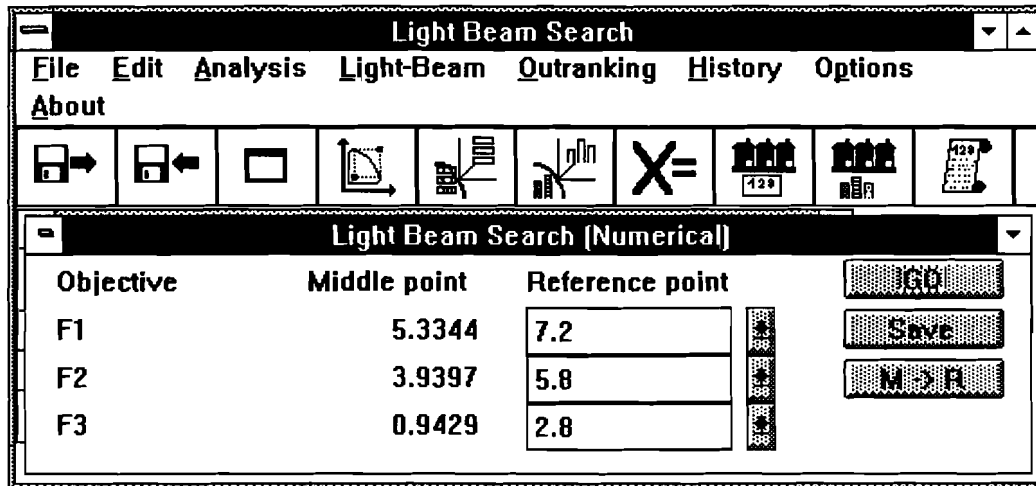


Figure 8. LIGHT BEAM SEARCH (NUMERICAL) window

Graphical

This command displays the LIGHT BEAM SEARCH (GRAPHICAL) window (see figure 9). In this window, the middle point and the reference point are presented in the graphical form. The thinner filled bars represent the middle point. The wider empty bars represent the reference point. The height of the empty bars can be changed by the user. In this way the values of the aspiration levels are changed. The aspiration levels can be changed pressing the left mouse button and dragging the top of a bar to desired position. While dragging the top of the bar, the corresponding value of the aspiration level in the LIGHT BEAM SEARCH (NUMERICAL) window is changed automatically.

By pressing button GO the user projects the reference point onto the non-dominated set. By pressing button SAVE the user can save the middle point (adds it to the set of stored points). Button M->R makes the middle point the new reference point.

All changes in the LIGHT BEAM SEARCH (GRAPHICAL) window are simultaneously made in the LIGHT BEAM SEARCH (NUMERICAL) window.

Solution details

This command displays the DETAILED DESCRIPTION OF THE MIDDLE POINT window. In this window the values of the definitions and variables as well as the numbers of active constraints are displayed.

Neighborhood | Numerical

This command displays the NEIGHBORHOOD OF THE MIDDLE POINT (NUMERICAL) window (see figure 10). This window presents the middle point and its characteristic neighbours with respect to particular objectives in a numerical form.

By pressing the left mouse button the user can select (deselect) one or more points in the window (see point MAX:F1 in figure 10). When one point is selected the MIDDLE button appears on the left side of the window. By pressing this button the user can make the selected point a new middle point. When two points are selected the PROFILE button appears on the

left side of the window. By pressing this button the user can display on the screen the PROFILE (NUMERICAL) window which presents the profile of the neighborhood between the two selected points. The PROFILE (NUMERICAL) window looks and behaves similar to the NEIGHBORHOOD OF THE MIDDLE POINT (NUMERICAL) window.

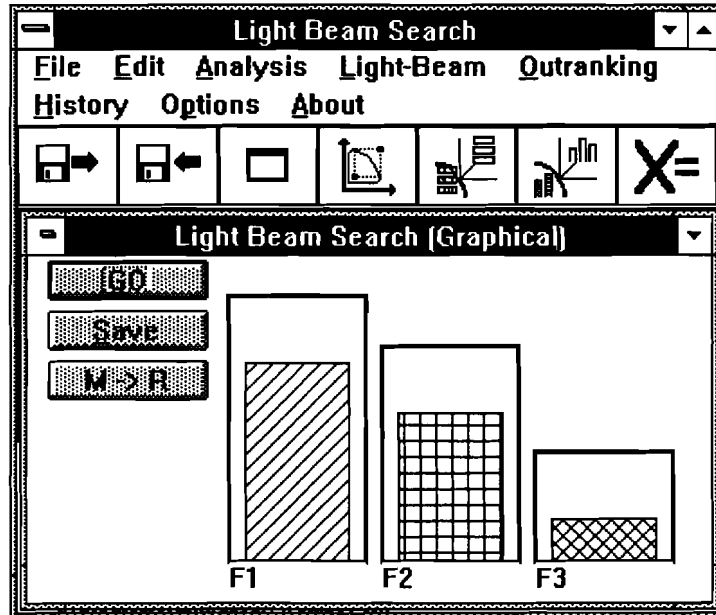


Figure 9. LIGHT BEAM SEARCH (GRAPHICAL) window

Middle	Middle point	max : F1	max : F2	max : F3
F1	5.3344	6.6561	3.2344	4.901
F2	3.9397	2.4247	6.085	3.6144
F3	0.9429	0.64	0.6193	2.0626

Figure 10. NEIGHBORHOOD OF THE MIDDLE POINT (NUMERICAL) window

Neighborhood | Graphical

This command displays the NEIGHBORHOOD OF THE MIDDLE POINT (GRAPHICAL) window (see figure 11). This window presents the middle point and its characteristic neighbours with respect to particular objectives in a graphical form.

By pressing the left mouse button the user can select (deselect) one or more points in the window (see points MAX:F2 and MAX:F3 in figure 11). When one point is selected the MIDDLE button appears on the left side of the window. By pressing this button the user can make selected point a new middle point. When two points are selected the PROFILE button appears on the left side of the window. By pressing this button the user can display on the screen the PROFILE (GRAPHICAL) window which presents the profile of the neighborhood between the two selected points. The PROFILE (GRAPHICAL) window looks and behaves similar to the NEIGHBORHOOD OF THE MIDDLE POINT (GRAPHICAL) window.

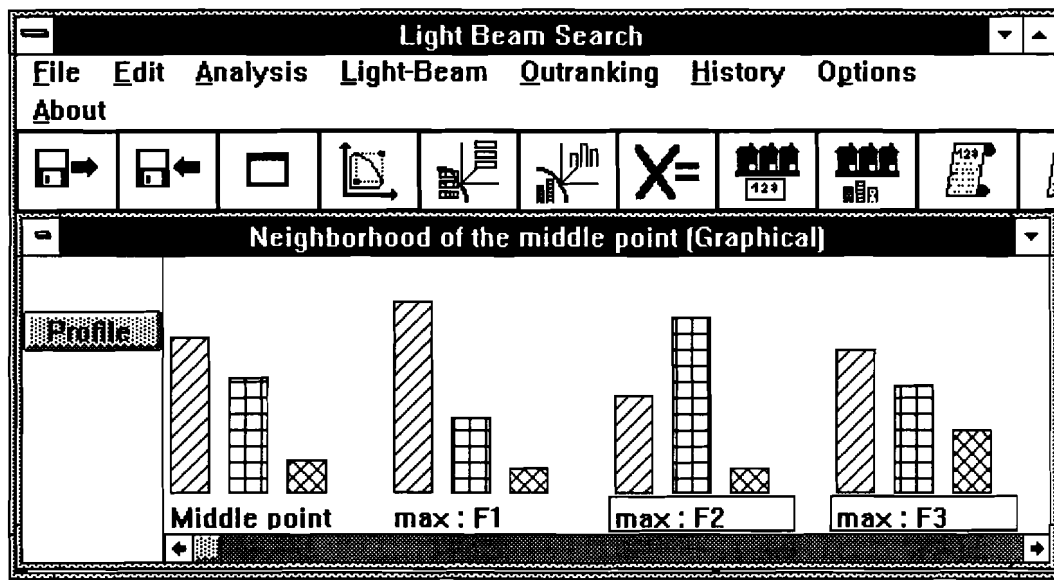


Figure 11. NEIGHBORHOOD OF THE MIDDLE POINT (GRAPHICAL) window

2.3.5 Outranking submenu

Outranking type

This submenu allows for selection of the type of preference information that the user is able to specify. The following options are available:

q p v - indifference, preference and veto thresholds (default),

q v - indifference and veto thresholds,

q p - indifference and preference thresholds,

q - indifference thresholds.

Thresholds

This command opens PREFERENCE INFORMATION dialog which allows for specification of values of particular thresholds for each objective.

2.3.6 History submenu

Numerical

This command displays the HISTORY (NUMERICAL) window. This window presents all the previously stored point. This window looks and behaves similar to the NEIGHBORHOOD OF THE MIDDLE POINT (NUMERICAL) window (see figure 10). When one or more points are selected the DELETE button appears on the left side of the

window. By pressing this button the user can delete the selected points from the set of stored points.

Graphical

This command displays the HISTORY (GRAPHICAL) window. This window presents all the previously stored point. This window looks and behaves similar to the NEIGHBORHOOD OF THE MIDDLE POINT (GRAPHICAL) window (see figure 11). When one or more points are selected the DELETE button appears on the left side of the window. By pressing this button the user can delete the selected points from the set of stored points.

2.3.7 Options submenu

Solver

This command allows to set the following parameters of non-linear solver: the precision of calculations and the maximal feasible numerical value.

Scale

This command allows to set the minimal and maximal values for particular objectives used in the graphical form of presentation. By default, the minimal value is equal to the minimal value found during the independent optimization of particular objectives while the maximal value is equal to the maximal value found during the independent optimization of particular objectives.

2.3.8 About submenu

About

This command displays a dialog containing some basic information about the software (see figure 5).

2.3.9 Toolbar

Some commands are available via the toolbar (the set of icon under main menu). The commands corresponding to particular icons are listed from left to right (see figure 6): FILE | OPEN, FILE | SAVE, FILE | NEW, ANALYSE | RANGES OF THE OBJECTIVES, LIGHT-BEAM | NUMERICAL, LIGHT-BEAM | GRAPHICAL, LIGHT-BEAM | SOLUTIONS DETAILS, LIGHT-BEAM | NEIGHBORHOOD | NUMERICAL, LIGHT-BEAM | NEIGHBORHOOD | GRAPHICAL, HISTORY | NUMERICAL and HISTORY | GRAPHICAL.

2.4 Problem definition format

The LBS allows to define multiple-objective non-linear programming problems in a natural text form. The problem definitions are stored in a standard text files. The text files can be prepared under text editor included in the package or under any text editor that produces text files in the ASCII format.

Problem definition consists of five parts. In the definition part, the user can define some macrodefinitions. A macrodefinition is composed of its name and of a mathematical expression. The macrodefinition names in below lines of the problem are automatically replaced by the appropriate expressions. In the next part, objective functions are defined. Each objective can be either minimized or maximized. The third part contains constraints. In

the fourth part, the user can define bounds on some decision variables. In the last part, a feasible starting solution should be defined. For each decision variable its starting value should be defined. If starting value is not defined for some variables, value 1 is used by default.

The exact input form of the problem definition is determined by the syntax presented below. The syntax is defined with the notation of Modified Backus-Naur Form. The meaning of meta-symbols is as follows:

- = denotes a definition,
- | separates alternative options within the clause,
- "..." terminal symbols are quoted,
- (...) exactly one of the enclosed alternatives must be selected,
- [...] denotes zero or one occurrence of the enclosed subclause,
- {...} denotes zero or any number of occurrences of the enclosed subclause.

The syntax of the problem definition is as follows:

```

problem =      [definition_part]
                objective_part
                constraint_part
                [bound_part]
                start_part

definition_part =  definition {definition}

definition =      definition_name "=" expression ";"

definition_name = name

objective_part =  objective {objective}

objective =       objective_type ":" objective_name "=" expression ";"

objective_type =  "MIN" | "MAX"

objective_name =  name

constraint_part = "CONSTR" {constraint}

constraint =       expression operator expression ";"

bound_part =      "BOUNDS" {bound}

bound =           variable "[" number "," number "]"

start_part =      "START" {start}

start =           variable "=" number ";"

operator =        "=" | "<=" | "<" | ">=" | ">"

expression =      exprs ("+" | "-") expression | exprs

```

exprs = exprp ("*" | "/") exprs | exprb

exprp = exprb "^" exprp | exprb

exprb = fun "(" expression ")" | "(" expression)" | variable | number |
 definition_name

variable = name

number = digit {digit} ["." digit {digit}] |
 digit {digit} ["." digit {digit}] "e" digit {digit}

fun = "SIN" | "COS" | "TNG" | "CTNG" | "LN" | "LOG" | "SQRT"

name = letter {letter}

letter = "a".."z" | "A".."Z" | "_"

3. Example application of the LBS method in chemical industry

3.1 Multiple-objective optimization of parameters of chemical reactors

One of the most important problems in designing industrial chemical installations is the design of chemical reactors in which particular reactions, necessary for obtaining the desired products, will be performed. There are several types of chemical reactors, but flow reactors which assure continuous production are most often used (see figure 12). A design of a flow reactor consists in setting of the following parameters: volume of the reactor, temperature, pressure, flow rate of reactants and catalyst weight. The parameters should be set to ensure the best compromise between some conflicting criteria. The criteria include maximization of concentrations and/or mass productions of desired products, minimization of concentrations and/or mass productions of undesired products, minimization of the temperature, minimization of the pressure, minimization of the catalyst weight and minimization of the volume of the reactor.



Figure 12. Flow reactor

To express the criteria as functions of the parameters, the kinetic model of processes in the reactor has to be known. The model is composed of a set of differential equations. In some simple cases it is possible to solve the set of differential equations analytically. In more complicated cases, numerical methods should be used to this end. The basic set of differential equation describes the rates of disappearance of particular substrates as functions of concentrations of reactants. For uncatalyzed processes the basic set of equations is as follows:

$$-\frac{dC_j}{d\theta} = f_j(C_1, \dots, C_J) \quad j=1, \dots, J$$

while for a catalyzed process it is as follows:

$$-\frac{dN_j}{dW_{ct}} = h_j(C_1, \dots, C_J) \quad j=1, \dots, J, \quad (2)$$

where: C_j - mole concentration [mol/m³], N_j - mole flow rate [mol/s], θ - process time [s], W_{ct} - catalyst weight [kg].

The parameters of a reactor are usually fixed in two phases. In the first phase, during a thermodynamic analysis, feasible ranges of such parameters as the temperature and pressure, are defined. The aim of the analysis is to reduce share of side reactions. In the second phase

exact values of all parameters are fixed taking into account the above mentioned criteria. Traditionally, the parameters are defined in a trial-and-error manner. However, interactive methods for multiple-objective mathematical programming can be used at this phase. As the kinetic model of processes is non-linear, such a method must be applicable for non-linear problems.

3.2. Formulation of the example problem

3.2.1 Kinetic model of isomerization of ortho-xylene over H-modernite

One of the important substrates in chemical technology is para-xylene. It is used as an intermediate in production of various plastics such as: polyesters and polyamides. This compound is usually obtained by isomerization of ortho-xylene over a catalyst. Applications of several catalysts in this reaction were tested. Hansford and Ward (1969) reported high activity of H-modernite catalyst. Hopper and Shigemura (1973) developed a kinetic model of this reaction over H-modernite. These authors also selected the values of the pressure - $2.76 \cdot 10^5$ [Pa], and temperature - 505 [K] at which the share of side reactions is lower than 1%. Their kinetic model is used as a basis for the mathematical programming problem describing the design of a flow reactor.

The reaction scheme is presented in figure 13.

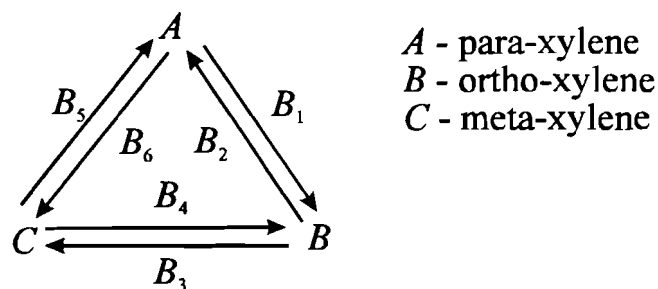


Figure 13. Scheme of the reaction of isomerization of xylene

The kinetic model assumes a first-order reversible reaction among three isomers of xylene - para-xylene, ortho-xylene and meta-xylene. For this reaction the set of equations (2) has the following form:

$$\begin{aligned}
 -\frac{dN_A}{dW_{ct}} &= B_1 C_A - B_2 C_B + B_6 C_A - B_5 C_C \\
 -\frac{dN_B}{dW_{ct}} &= B_2 C_B - B_1 C_A + B_3 C_B - B_4 C_C \\
 -\frac{dN_C}{dW_{ct}} &= B_4 C_C - B_3 C_B + B_5 C_C - B_6 C_A
 \end{aligned}$$

where: B_k - reaction rate constants [$\text{m}^3/\text{kg s}$], $k=1, \dots, 6$.

Since there is no change in the number of moles during the reaction, these equations can be written in the following form:

$$\begin{aligned}
-\phi \frac{dX_A}{d\tau} &= B_1 X_A - B_2 X_B + B_6 X_A - B_3 X_C \\
-\phi \frac{dX_B}{d\tau} &= B_2 X_B - B_1 X_A + B_3 X_B - B_4 X_C, \\
-\phi \frac{dX_C}{d\tau} &= B_4 X_C - B_3 X_B + B_5 X_C - B_6 X_A
\end{aligned} \tag{3}$$

where: X_j - mole fraction of isomer j , $j = A, \dots, C$; τ - space time [kg-CAT s/kg-FEED], ϕ - 1/average density [m^3/kg] is defined in the following way:

$$\phi = \frac{W_A}{\rho_A} + \frac{W_B}{\rho_B} + \frac{W_C}{\rho_C} + \frac{W_D}{\rho_D},$$

where: ρ_j - density of compound j [kg/m^3], W_j - weight fraction, D - symbol of a dissolvent used in the reaction (toluene).

The relationship between weight fractions W_j and mole fractions X_j is described by the following expression:

$$W_j = X_B^0 X_j \rho_j \phi^0,$$

where: X_B^0 - mole fraction of ortho-xylene in the reactor feed, ϕ^0 - 1/average density of the reactor feed.

Hopper and Shigemura (1973) reported also that the influence of the process time θ and space time τ on the reaction rate constants B_k can be described by the following expressions:

$$B_k = B_{0k} e^{-\alpha_k (\theta/\tau)}, \quad k = 1, \dots, 6. \tag{4}$$

Basing on experimental data they also found values of constants B_{0k} and α_k .

Substituting reaction rate constants B_k for expressions (4) in set of equations (3) and solving the set of equations with a numerical method one defines mole fractions X_j as functions of the process time θ and space time τ :

$$X_j = X_j(\theta, \tau), \quad j = A, \dots, C.$$

The analytical form of the above functions is unknown. Their values, however, can be found with a numerical method for given θ and τ .

3.2.2 Formulation of a multiple-objective non-linear mathematical programming problem

It is assumed that there are three decision variables:

- V - reactor volume [m^3],
- \dot{V} - feed flow rate [m^3/s],
- W_{ct} - catalyst weight [kg].

The first constraint defines the lowest, technologically feasible value of FEED/CATALYST ratio $\frac{\theta}{\tau}$:

$$\frac{\theta}{\tau} \leq 30,$$

where: $\theta = \frac{V}{\dot{V}}$, $\tau = \frac{W_{ct}\theta\phi}{V}$.

The second constraint defines the greatest feasible value of FEED/CATALYST ratio $\frac{\theta}{\tau}$ at which the presented above kinetic model describes the chemical process accurately:

$$\frac{\theta}{\tau} \geq 10.$$

Other constraints define feasible ranges of particular variables:

$$15 \leq V \leq 40,$$

$$0.01 \leq \dot{V} \leq 0.05,$$

$$200 \leq W_{ct} \leq 1500.$$

The following criteria are taken into account in the problem:

- V - reactor volume [m^3],
- W_{ct} - catalyst weight [kg],
- P_A - production of para-xylene [kg/h],
- CR - concentration ratio between para-xylene and ortho-xylene.

The first criterion - V , which is minimized, represents the designer's attitude to reduce size of the chemical installation. It influences both investment and operating cost of the reactor.

The second criterion - W_{ct} , expresses the designer's aspirations to reduce the weight of the H-modernite used in the reactor. This criterion is also minimized.

The mass production of para-xylene - P_A , depends on feed flow rate \dot{V} and weight fraction of para-xylene in the reaction products - W_A :

$$P_A = \frac{\dot{V}W_A}{\phi}.$$

The mass production is maximized.

The last criterion, concentration ratio between para-xylene and ortho-xylene - CR , is maximized. It is described by the following expression:

$$CR = \frac{X_A(\theta, \tau)}{X_B(\theta, \tau)}.$$

This criterion is correlated with the quality of the final product and with the level of transformation of ortho-xylene.

3.3 Multiple-objective analysis of the problem

The presented above multiple-objective non-linear mathematical programming problem is solved with the LBS method. The stages of the computational experiment are presented below.

The procedure starts by fixing the points of the best and the worst values of objectives, z^* and z_* , respectively. Point z^* is as follows:

$$\begin{aligned} V &= 15 \text{ [m}^3\text{]} \\ W_{ct} &= 348 \text{ [kg]} \\ P_A &= 7630 \text{ [kg/h]} \\ CR &= 95.7 \text{ [\%]}, \end{aligned}$$

while point z_* is as follows:

$$\begin{aligned} V &= 40 \text{ [m}^3\text{]} \\ W_{ct} &= 1500 \text{ [kg]} \\ P_A &= 1883 \text{ [kg/h]} \\ CR &= 21 \text{ [\%]}. \end{aligned}$$

Then the DM is asked to decide what kind of preferential information he wants to specify. The DM decides to specify the indifference and veto thresholds and gives the following values:

Objective	q_j	v_j
$V \text{ [m}^3\text{]}$	5	15
$W_{ct} \text{ [kg]}$	30	300
$P_A \text{ [kg/h]}$	210	700
$CR \text{ [\%]}$	5	10

The procedure finds the starting middle point z^c :

$$\begin{aligned} V &= 22.9 \text{ [m}^3\text{]} \\ W_{ct} &= 802 \text{ [kg]} \\ P_A &= 3710 \text{ [kg/h]} \\ CR &= 35.6 \text{ [\%]}. \end{aligned}$$

An outranking neighborhood is constructed around the middle point. The attainable characteristic neighbours coming from the neighborhood are calculated :

Objective	z^{V}	$z^{W_{ct}}$	z^{P_A}	z^{CR}
$V [m^3]$	16	20.7	22.9	22.9
$W_{ct} [kg]$	829	724	802	802
$P_A [kg/h]$	3633	3437	4011	3444
$CR [\%]$	35	30.6	25.6	49.1

The points are presented to the DM. The DM has an idea about desired values of objectives and decides to specify a reference point z^0 . As he/she would like to obtain better values of the production and concentrations ratio while he/she is ready to accept worse values of the reactor volume and catalyst mass he/she gives the following point:

$$\begin{aligned} V &= 30 [m^3] \\ W_{ct} &= 1100 [kg] \\ P_A &= 5600 [kg/h] \\ CR &= 75 [\%]. \end{aligned}$$

The point given by the DM is non-attainable. The procedure projects it onto the non-dominated set. The new middle point z^c :

$$\begin{aligned} V &= 31.3 [m^3] \\ W_{ct} &= 1245 [kg] \\ P_A &= 4879 [kg/h] \\ CR &= 63.9 [\%], \end{aligned}$$

is a result of the projection. The attainable characteristic neighbours found for the new middle points are as follows:

Objective	z^{V}	$z^{W_{ct}}$	z^{P_A}	z^{CR}
$V [m^3]$	18	28.9	31.4	30.3
$W_{ct} [kg]$	1257	1143	1245	1247
$P_A [kg/h]$	4480	4655	5194	4263
$CR [\%]$	55.9	58	53.8	81.5

The DM feels that at this point the values of some thresholds are not appropriate and decides to change the preferential information. This time he/she gives the following values of the thresholds:

Objective	q_j	v_j
$V [m^3]$	4	7
$W_{ct} [kg]$	20	100
$P_A [kg/h]$	210	350
$CR [%]$	3	5

With the new preferential information the procedure constructs a new outranking neighborhood and calculates new attainable characteristic neighbours:

Objective	z^{iV}	$z^{iW_{ct}}$	z^{iP_A}	z^{iCR}
$V [m^3]$	18	29.4	31.4	30.7
$W_{ct} [kg]$	1257	1155	1245	1246
$P_A [kg/h]$	4480	4676	5040	4522
$CR [%]$	55.9	58.7	58.9	74.5

The DM thinks that point z^{iP_A} is better than the middle point and selects it to be the new middle point. New attainable characteristic neighbours are found and presented to the DM:

Objective	z^{iV}	$z^{iW_{ct}}$	z^{iP_A}	z^{iCR}
$V [m^3]$	18.1	29.7	31.3	30.9
$W_{ct} [kg]$	1259	1152	1245	1246
$P_A [kg/h]$	4592	4816	5194	4683
$CR [%]$	51.3	53.7	53.9	69.9

The DM decides to scan more precisely the profile between points $z^{iW_{ct}}$ and z^{iP_A} . The profile is constructed by projection of the line segment that connects point $z^{iW_{ct}}$ with z^{iP_A} . The sample of the non-dominated points from the profile is presented to the DM:

Objective	w ¹	w ²	w ³	w ⁴	w ⁵
V [m ³]	30.1	30.2	30.5	30.8	31.2
W_{ct} [kg]	1153	1169	1179	1209	1219
P_A [kg/h]	4837	4921	4942	5082	5103
CR [%]	53.7	53.7	53.2	53.8	53.3

The DM feels that point w^4 is satisfactory on all objectives. Thus it gives the best compromise. In the space of variables the point correspond to the following solution:

$$\begin{aligned}
 V &= 30.88 \text{ [m}^3\text{]} \\
 \dot{V} &= 0.0153 \text{ [m}^3\text{/s]} \\
 W_{ct} &= 1209.
 \end{aligned}$$

At this point the process time θ is equal to 2015 [s] and the FEED/CATALYST ratio $\frac{\theta}{\tau}$ is equal to 17.7 [kg FEED/kg CAT.].

3.4 Conclusions

The design process of a chemical reactor has been formulated as a multiple-criteria decision making problem. A detailed formulation of the problem in terms of multiple-objective mathematical programming for the case of isomerization of ortho-xylene over H-modernite has been presented. Its main task is to ensure the best compromise between such criteria as: reactor volume, catalyst weight, production of para-xylene and concentration ratio between para-xylene and ortho-xylene.

The exemplary problem has been solved with the LBS method. The results of the computational experiment proved that the method is an effective tool to optimize parameters of chemical reactors. It allows the DM to scan freely the set of efficient solutions in searching for the best compromise between conflicting criteria. At each step a number of alternative proposals coming from the outranking neighborhood of the current point are presented to the DM. It helps him/her to achieve better understanding of the problem and systematic improvement of the design of the reactor. The DM controls the interactive process by either specifying his/her aspirations levels on reactor volume, catalyst weight, production of para-xylene and concentration ratio between para-xylene and ortho-xylene or by selecting the best solution from a sample of alternative proposals.

4. Description of non-linear solvers used in the LBS package

At the computational phases of the LBS method, single-objective non-linear problems have to be solved in order to find the middle point and its characteristic neighbours. Because in interactive procedures the duration of the computational phases should be minimized, an efficient non-linear solver has to be used in the implementation. Two solvers have been integrated with the system. One coming from the PINOKIO package (Jaskiewicz and Słowiński, 1992b) is an implementation of the Generalized Reduced Gradient method (GRG) (Abadie, 1977). The second one coming from the DIDAS-N package (Kręglewski et al., 1991) is an implementation of the Penalty Shifting Method (Wierzbicki, 1971).

4.1 Generalized Reduced Gradient method

The GRG method allows to solve non-linear problems defined in the following form:

$$\begin{aligned} & \min g(\mathbf{X}) && \text{(P4)} \\ \text{s.t.} & && \\ & \mathbf{e}(\mathbf{X}) = \mathbf{0} && \\ & \mathbf{D} \leq \mathbf{X} \leq \mathbf{W} && \end{aligned}$$

where:

- $\mathbf{X} = [x_1, \dots, x_n]$ - vector of decision variables,
- $\mathbf{D} = [d_1, \dots, d_n]$ - vector of lower bounds on the decision variables,
- $\mathbf{W} = [w_1, \dots, w_n]$ - vector of upper bounds on the decision variables,
- $g: \mathbf{R}^n \rightarrow \mathbf{R}$ - objective function,
- $\mathbf{e}: \mathbf{R}^n \rightarrow \mathbf{R}^m$ - constraints.

The objective function and constraints are assumed to be continuous and differentiable. Every non-linear problem can be transformed to the above form by adding a number of slack variables if there are any inequalities among the constraints.

The method is based on a linear approximation of the objective function and constraints in a neighborhood of a current solution \mathbf{X} . Vector \mathbf{X} can be divided into two vectors $\mathbf{Y} \in \mathbf{R}^m$ and $\mathbf{Z} \in \mathbf{R}^{n-m}$. Vector \mathbf{Z} is called the vector of independent variables while vector \mathbf{Y} is called the vector of dependent variables. Elements of vector \mathbf{Y} have to be strictly within bounds. Vector \mathbf{Y} can be stated as a function of vector \mathbf{Z} - $\mathbf{Y}(\mathbf{Z})$. Substituting $\mathbf{Y}(\mathbf{Z})$ for \mathbf{Y} we obtain so called *reduced objective function*:

$$g'(\mathbf{Y}(\mathbf{Z}), \mathbf{Z})$$

A gradient of this function is equal to

$$\nabla_{\mathbf{z}} g' = - \left(\frac{\partial \mathbf{e}}{\partial \mathbf{y}} \right)^{-1} \frac{\partial \mathbf{e}}{\partial \mathbf{z}} \frac{\partial g}{\partial \mathbf{z}} + \frac{\partial g}{\partial \mathbf{z}}$$

The above expression is called the *reduced gradient* of the objective function.

The algorithm of the method is the following:

Step 1

Select an available starting solution \mathbf{X}^0 , $k := 0$.

Step 2

Divide vector \mathbf{X} into two vectors \mathbf{Y} i \mathbf{Z} such that

$$\mathbf{D}_y < \mathbf{Y} < \mathbf{W}_y$$

Step 3

Calculate the reduced gradient of the objective function at point \mathbf{X}^0 and conjugated direction $\Delta\mathbf{Z}$:

$$\Delta z_i' = \begin{cases} -\nabla_z g_i' & \text{if } d_i < z_i < w_i \\ 0 & \text{if } (z_i = d_i \text{ and } -\nabla_z g_i' < 0) \text{ or } (z_i = w_i \text{ and } -\nabla_z g_i' > 0) \end{cases}$$

$$\Delta z_j = \Delta z_j' + a \Delta z_j^{k-1}, \quad j = 1, \dots, n$$

where:

$\Delta\mathbf{Z}^{k-1}$ - direction found in the previous iteration,

a - a small constant used to obtain the conjugated direction.

Step 4

Find the optimal step length α^* , in direction $\Delta\mathbf{Z}$ by solving the following one-dimensional problem:

$$\min g(\mathbf{Y}(\mathbf{X}^k + \alpha \Delta\mathbf{Z}), \mathbf{X}^k + \alpha \Delta\mathbf{Z}).$$

Make the step in direction $\Delta\mathbf{Z}$

$$\mathbf{Z}^{k+1} := \mathbf{Z}^k + \alpha^* \Delta\mathbf{Z}.$$

Calculate vector $\mathbf{Y} = \mathbf{Y}(\mathbf{Z})$.

If the difference between \mathbf{X}^k a \mathbf{X}^{k+1} is smaller than a given threshold then stop the procedure. Otherwise, $k := k + 1$ and return to step 2.

Function $\mathbf{Y}(\mathbf{Z})$ can be calculated by solving the set of equations $\mathbf{e}(\mathbf{Y}(\mathbf{Z}), \mathbf{Z}) = \mathbf{0}$. To this end a numerical method for solving sets of non-linear equations, e.g. the Newton method, should be used.

4.2 Penalty Function Shifting method

The method allows to solve problems defined in the following form:

$$\begin{aligned} & \min g(\mathbf{X}) && \text{(P5)} \\ \text{s.t.} & && \\ & \mathbf{e}(\mathbf{X}) \leq \mathbf{0} && \end{aligned}$$

where:

$\mathbf{X} = [x_1, \dots, x_n]$ - vector of decision variables,

$g : \mathbf{R}^n \rightarrow \mathbf{R}$ - objective function,

$\mathbf{e} : \mathbf{R}^n \rightarrow \mathbf{R}^m$ - constraints.

Every non-linear problem can be transformed to the above form. Let \mathbf{E}^0 be the set of feasible solutions.

The method consists in converting the constrained problem into an unconstrained one, using a shifted penalty function. The function penalizes the original objective when the constraints are approached or violated. It has the following form:

$$G(\mathbf{X}, \mathbf{V}, \mathbf{P}) = g(\mathbf{X}) + \frac{1}{2} \sum_{j \in \mathbf{J}} p_j (e_j(\mathbf{X}) + v_j) \max(0, e_j(\mathbf{X}) + v_j)$$

Initially, $\mathbf{P} = [p_1, \dots, p_j, \dots, p_m]$ is set to a given value $\mathbf{P}_s > \mathbf{0}$, $\mathbf{V} = [v_1, \dots, v_j, \dots, v_m]$ is set to $\mathbf{0}$, \mathbf{J} is set to $\{1, \dots, j, \dots, m\}$. Given a starting solution $\mathbf{X} \in \mathbf{E}^0$, an arbitrary chosen computational method of unconstrained optimization is applied, and an approximation $\hat{\mathbf{X}}^i$ of the final solution is found ($\hat{\mathbf{X}}^i$ denotes the approximation of $\hat{\mathbf{X}}$ at the iteration i).

Set \mathbf{E}_c^i of admissible solutions at the iteration i is defined as:

$$\mathbf{E}_c^i = \{\mathbf{X} \in \mathbf{R}^n : e_j(\mathbf{X}) \leq c^i, j=1, \dots, m\}$$

where $c^1 > 0$ is given, and c^j is decreased at each iteration by at least a given factor of the convergence rate $r \in (0, 1)$. If $\hat{\mathbf{X}}^i \in \mathbf{E}_c^i$ the iteration is considered to be successful and the basic algorithm of penalty shift is applied:

$$\mathbf{V}^{i+1} = \max\left(0, \mathbf{V}^i + e\left(\hat{\mathbf{X}}^i\right)\right)$$

Clearly, $\mathbf{V}^i > \mathbf{0}$ for all i . From \mathbf{V}^{i+1} , the set of 'strongly active' constraint indices is determined:

$$\tilde{\mathbf{J}}_j^{i+1} = \{j : v_j^{i+1} > 0\}$$

Because it may happen in some iterations that the approximations is forced too deep into interior of \mathbf{E}^0 by the penalty shifts v_j (a constraint can be only 'temporarily active', i.e. drop out of the set \mathbf{J} in an iteration), the following condition is checked:

$$e_j\left(\hat{\mathbf{X}}^i\right) \geq -c^i \quad \text{for } j \in \tilde{\mathbf{J}}^{i+1}$$

If the condition is satisfied, the constant c^{i+1} for the next iteration is determined by:

$$c^{i+1} = r \max_{j \in \tilde{\mathbf{J}}^{i+1}} \left| e_j\left(\hat{\mathbf{X}}^i\right) \right|$$

Clearly, c^{i+1} satisfies the inequality:

$$c^{i+1} \leq (r)^i c^1$$

where $(r)^i$ denotes the i -th of the assumed convergence rate r . It remains to determine the set of 'possibly active' constraint indices:

$$\bar{J}^{i+1} = \left\{ j \in J \mid e_j(\hat{X}^i) \geq -\frac{1}{r} c^{i+1}, j \notin \tilde{J}^{i+1} \right\}$$

to set $J^{i+1} = \bar{J}^{i+1} \cup \tilde{J}^{i+1}$ and to check the stopping rule. Then the iteration, called large iteration, is completed.

If $\hat{X}^i \notin E_c^i$, it is assumed that the penalty coefficients \mathbf{P} are too small, and they are multiplied by a given factor $k > 1$; at the same time the penalty shifts \mathbf{V} are divided by k , and the minimization of G is repeated. Such a repetition is called a small iteration.

4.3 Interface between the LBS package and a non-linear solver

Two solvers coming from different packages have been integrated with the microcomputer implementation of the Light Beam Search method. This task was relatively easy because both the modules were written in Turbo Pascal programming language. However, in both cases, some difficulties have been met. The solvers were not written as independent modules communicating with other parts of applications through a precisely defined interface. In spite of this, the modules were strongly connected with other parts of the systems. The interface between a non-linear solver and other parts of a system for multiple-objective analysis of non-linear problems should meet the following requirements:

- The solvers should call a module which calculates the values of the objective functions and the constraints as well as their gradients.
- It should give the possibility of displaying some temporary results found during the optimization.
- It should give the possibility of stopping the optimization.

We propose a standard interface which meets the above requirements. It has been used in the implementation of the Light Beam Search method. Both the solvers have been transformed to a form compatible with the interface. The interface requires the following form of the problem definition:

$$\begin{aligned} & \max/\min o_1(\mathbf{X}) \\ \text{s.t.} & \\ & d_j^x \leq x_j \leq w_j^x, \quad i = 1, \dots, n \\ & d_j^y \leq o_j(\mathbf{X}) \leq w_j^y, \quad j = 1, \dots, m, \end{aligned}$$

where:

$$\begin{aligned} \mathbf{X} &= [x_1, \dots, x_n] - \text{vector of decision variables,} \\ \mathbf{O} &= [o_1, \dots, o_m] - \text{vector of outcomes,} \\ \mathbf{D}^x &= [d_1^x, \dots, d_n^x] - \text{vector of lower bounds on the decision variables,} \\ \mathbf{W}^x &= [w_1^x, \dots, w_n^x] - \text{vector of upper bounds on the decision variables,} \\ \mathbf{D}^y &= [d_1^y, \dots, d_m^y] - \text{vector of lower bounds on the outcomes,} \\ \mathbf{W}^y &= [w_1^y, \dots, w_m^y] - \text{vector of upper bounds on the outcomes.} \end{aligned}$$

This form is equivalent to forms (P4) and (P5).

We use the tools provided by the object oriented programming to implement the interface. A solver is defined as an object type of the following form:

TSolver =

```

object (TObject)
  v_f      : DoubleArrayPtr;
  TD       : DoubleTablePtr;
  procedure CalAll (bWithGrad : Boolean;
                   CDimCon : Integer;
                   CCons : IntegerArrayPtr;
                   CDimVar : Integer;
                   CVars : IntegerArrayPtr);
                   virtual;
  procedure NewSolution; virtual;
  function CheckStop : Boolean;
  constructor Init (ADimX, ADimY : Word;
                  AMaxValue : Extended);
  function Run (Amy_XLo, Amy_XUp, Amy_YLo, Amy_YUp :
               DoubleArrayPtr;
               AAccuracy, AViolation : Double;
               Amy_X : DoubleArrayPtr;
               ADirUto : Integer) : TError;
  destructor Done; virtual;
private
  ...
end;

```

Constructor `Init` is responsible for allocating the dynamic memory necessary to run the solver and destructor `Done` is responsible for freeing the memory. Method `Run` performs the main optimization procedure. Method `CallAll` should calculate the values of the objective function and the constraints as well as their gradients. Parameters of the method allow to specify if the gradients should be calculated and to specify the constraints and variables for which the gradients should be calculated. Method `NewSolution` should be called when a new feasible (or nearly feasible) solution is found. It can be used to display the temporary results of the optimization. Method `CheckStop` should give the user the possibility of stopping the optimization (e.g. by pressing a key on the keyboard). The method should be called at various stages of the optimization procedure and the procedure should be stopped when it returns value `True`. All other variables and methods necessary in a solver should be declared in `private` part of the object. The author of a solver should define the methods `CallAll`, `NewSolution` and `CheckStop` as abstract (empty) methods. The definition of the methods is overwritten in the LBS package in classes that are specialization of class `TSolver`.

REFERENCES

- Abadie J., The GRG method for Nonlinear Programming. In Greenberg H.J. *Design and Implementation of Optimisation Software*. Sijthoff & Noordhoff, Alphen aan den Rijn, The Netherlands (1977).
- Benayoun R., de Montgolfier J., Tergny J. and Larichev O., *Linear programming with multiple objective functions: step method (STEM)*. Operational Research Quarterly, 24: 65-77, 1971.
- Hansford R.C. and Ward J.W. (1969). The nature of Active Sites of Zeolites. *AIChE Journal*, 13, 316-322.
- Hopper J.R. and Shigemura D.S. (1973). Kinetics of liquid phase xylene isomerization over H-modernite. *AIChE Journal*, 19, no. 5, 1025-1032.
- Jacquet-Lagrèze E., Meziani R. and Słowiński R., *MOLP with an interactive assessment of a piecewise-linear utility function*. Eur. J. Oper. Res., 31: 350-357, 1987.
- Jaszkievicz A. and Słowiński R., *Cone Contraction Method with Visual Interaction for Multiple-Objective Non-Linear Programmes*. Journal of Multi-Criteria Decision Analysis, 1: 29-46, 1992a.
- Jaszkievicz A. and Słowiński R., Light Beam Search over a Non-Dominated Surface of a Multiple Objective Programming Problem. Proceedings of the Tenth International Conference on Multiple Criteria Decision Making, Taipei, vol.3, pp. 439-448, Taipei, 1992b.
- Korhonen P., *VIG - a visual interactive support system for multiple criteria decision making*. Belg. J. Ops Res. Statist. Comput. Sci., 27: 3-15, 1987.
- Korhonen P. and Wallenius J., *A Pareto race*. Naval Research Logistics, 35: 615-623, 1988.
- Korhonen P., Wallenius J. and Zionts S., *A Computer Graphics-Based Decision Support System for Multiple Objective Linear Programming*. Eur. J. Oper. Res., 60: 280-286, 1992.
- Kreglewski T., Granat J. and Wierzbicki A.P., *IAC-DIDAS-N: A Dynamic Interactive Decision Analysis and Support System for Multicriteria Analysis of Nonlinear Models. Version 4.0*. Collaborative Paper CP-91-10. International Institute for Applied Systems Analysis, Laxenburg, Austria (1991).
- Lee S.M. and Shim J.P., *Interactive goal programming on the microcomputer to establish priorities for small business*. Journal of the Operational Research Society, 37: 571-577, 1986.
- Lotfi V., Stewart T.J. and Zionts S., *An aspiration-level interactive model for multiple criteria decision making*. Comput. Oper. Res., in press.
- Luce D., *Semiordeers and a theory of utility discrimination*. Econometrica, 24: 178-191, 1956.
- Narula S.C., Kirilov L. and Vassilev V., *Reference direction approach for solving multiple objective nonlinear programming problems*. Proceedings of the Tenth International Conference on Multiple Criteria Decision Making, Taipei, 2: 355-362, 1992.
- Poincaré H., *La valeur de la Science*, Paris: Flammarion, 1935.
- Rosen J.B., *The gradient projection method for nonlinear programming. Part I: Linear constraints*. SIAM J. Appl. Math., 8, 1960.

- Roy B., *ELECTRE III - un algorithme de classement fondé sur une représentation floue des préférences en présence de critères multiples*. Cahiers du CERO, 20: 3-24, 1978.
- Roy B., *Méthodologie Multicritère d'Aide à la Décision*, Paris: Economica, 1985.
- Roy B., *The outranking approach and the foundations of ELECTRE methods*. In: Bana e Costa C.A. (ed.) *Readings in Multiple Criteria Decision Aid*, Berlin: Springer-Verlag, 155-183, 1990.
- Roy B. and Bouyssou D., *Aide Multicritère à la Décision: Méthodes et Cas*. Paris, Economica, 1993.
- Vincke P., *Basic concepts of preference modelling*. In: Bana e Costa C.A. (ed.) *Readings in Multiple Criteria Decision Aid*, Berlin: Springer-Verlag, 101-118, 1990.
- Wierzbicki A.P., *A Penalty Shifting Method in Constrained Static Optimisation and its Convergence Properties*. *Archiwum Automatyki i Telemekhaniki*, 16 (1971), pp. 395-416.
- Wierzbicki A.P., *The use of reference objective in Multiobjective Optimization*. In: Fandel G. and Gal T. (eds.) *Multiple Criteria Decision Making, Theory and Application*, Berlin: Springer-Verlag, 468-486, 1980.
- Wierzbicki A.P., *On the completeness and constructiveness of parametric characterizations to vector optimization problems*. *OR Spectrum*, 8: 73-87, 1986.
- Zionts S. and Wallenius J., *An interactive programming method for solving the multiple criteria problem*. *Management Science*, 22: 652-663, 1976.