Journal of Global Optimization **12:** 139–155, 1998.

Two-phases Method and Branch and Bound Procedures to Solve the Bi–objective Knapsack Problem

M. VISÉE, J. TEGHEM, M. PIRLOT and E.L. ULUNGU
Laboratory of Mathematics and Operational Research, Faculté Polytechnique de Mons, 9, rue de Houdain B-7000 Mons, Belgium

(Received and accepted: 2 May 1997)

Abstract. The classical 0–1 knapsack problem is considered with two objectives. Two methods of the "two–phases" type are developed to generate the set of efficient solutions. In the first phase, the set of supported efficient solutions is determined by optimizing a parameterized single-objective knapsack problem. Two versions are proposed for a second phase, determining the non-supported efficient solutions: both versions are Branch and Bound approaches, but one is "breadth first", while the other is "depth first". Extensive numerical experiments have been realized to compare the results of both methods.

Key words: Combinatorial optimization, multi-objective programming, knapsack problem.

1. Introduction

Until recently, multi-objective combinatorial optimization (MOCO) did not receive much attention in spite of its potential applications. The reason is probably due to specific difficulties of MOCO models as pointed out in a recent survey [6] concerning this field.

The aim of the present paper is to develop and compare two procedures to generate the set of efficient solutions of a particular bi-objective MOCO problem, the *Knapsack problem*. This is a basic well known combinatorial optimization problem, important for applications and as a sub-problem of more complicated ones. Moreover its mathematical structure is relatively simple and there exist efficient algorithms to solve it, like the Martello and Toth [4] method.

In a bi-objective framework, the Knapsack problem can be formulated as

$$\left\{\begin{array}{ll} \text{“max”} \; z_k(X) = \displaystyle\sum_{j=1}^{n} c_j^k x_j & k = 1, 2 \\ \displaystyle\sum_{j=1}^{n} w_j x_j \leq W & \\ x_j \in \{0, 1\} & \end{array}\right. \tag{P}$$

where coefficients c_j^k, w_j and W are non negative integers and $X = (x_1, \ldots, x_n)$; in order to avoid trivial solutions, we suppose in addition that

$$w_j \leq W \quad \forall\, j = 1, \ldots, n, \quad \text{and} \quad \sum_{j=1}^{n} w_j > W.$$

This model is suitable in various contexts including problems of media selection [2] and capital budgeting [5].

A solution $X^\star$ of problem P is *efficient* if there does not exist any other feasible solution X such that $z_k(X) \geq z_k(X^\star), \quad k = 1, 2$, with at least one strict inequality. We will denote by $E(P)$ the set of efficient solutions of problem P.

In multi-objective integer linear programming, it is necessary to distinguish two kinds of efficient solutions (cf. [6]):

– the set $SE(P)$ of *supported* efficient solutions which are optimal solutions of the parameterized single objective problem

$$\begin{cases} \max\ z_\lambda(X) = \lambda_1 z_1(X) + \lambda_2 z_2(X) \\ \displaystyle\sum_{j=1}^{n} w_j x_j \leq W \\ x_j = (0, 1) \\ \lambda_1 > 0, \lambda_2 > 0 \end{cases} \tag{P}$$

– the set $NSE(P) = E(P) \setminus SE(P)$ of *non-supported efficient solutions* which cannot be found by optimization of problem (P_λ). These non-supported efficient solutions are necessarily located in the triangles $\triangle Z_r Z_s$ generated in the objective space by two successive supported efficient solutions, as represented in Figure 1.

Recently, several studies have been devoted to the generation of $E(P)$:

– an adaptation to the multi-objective framework of the implicit enumeration method of Martello and Toth has been proposed by Ulungu and Teghem [8];
– in [7] a two-phase method is developed: in the first phase, $SE(P)$ is obtained by solving problem (P_λ); then a Branch and Bound procedure is applied to each triangle $\triangle Z_r Z_s$ to generate $NSE(P)$;
– in [9] and [1], Simulated Annealing is adapted to the multi-objective context to approximate $E(P)$ in a heuristic way.

In the following sections, we will describe two new procedures to generate $E(P)$ which are both two-phases methods.

The common first phase of these two procedures is recalled in section 2; the second phases are respectively described in sections 3 and 4.

The first procedure improves on what is done in [7], due to a better lower bound of z_λ for non supported efficient solutions in triangles $\triangle Z_r Z_s$. In the second procedure, we adapt the implicit enumeration scheme of [8] to generate the non supported efficient solutions in each triangle $\triangle Z_r Z_s$.

Extensive numerical results are given in section 5 in order to compare the two methods.

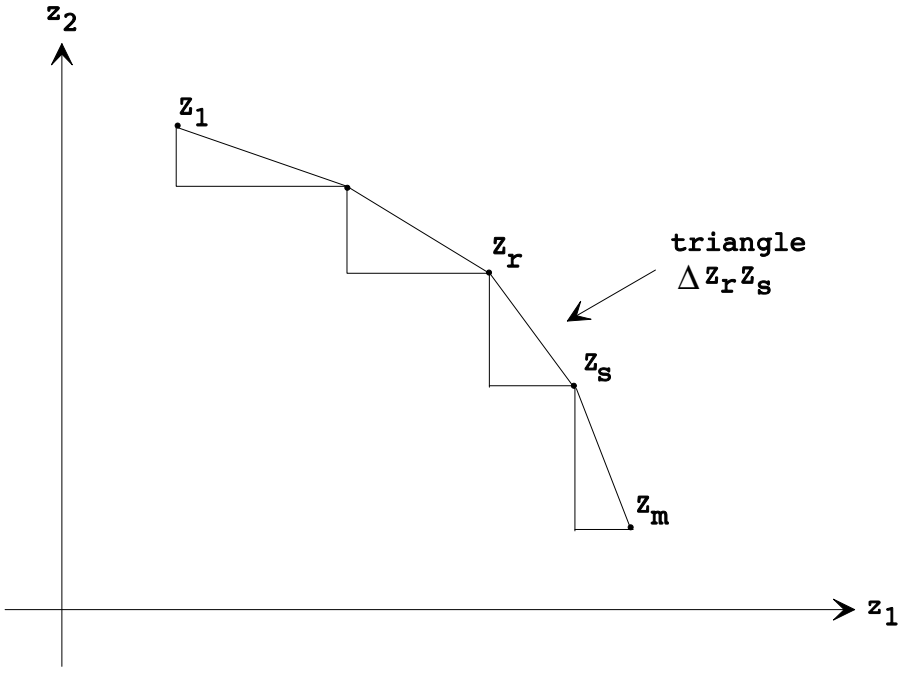

Figure 1. Supported efficient solutions $Z_1, \ldots, Z_m$ and potential regions $\triangle Z_r Z_s$ of non supported efficient solutions.

2. The First Phase: Determination of $SE(P)$

Let $S \cup S'$ be the list of supported efficient solutions already generated; S is initialized with the two efficient optimal solutions respectively of objectives z_1 and z_2, obtained by Martello and Toth's single-criterion algorithm.

Solutions of S are ordered by increasing value of criterion 1; let X^r and X^s be two consecutive solutions in S, thus with $z_{1r} < z_{1s}$ and $z_{2r} > z_{2s}$ where $z_{kl} = z_k(X^l)$.

The single-criterion problem P_λ is considered with the values $\lambda_1 = z_{2r} - z_{2s}$ and $\lambda_2 = z_{1s} - z_{1r}$ and optimized by Martello and Toth's algorithm (in fact a variant which returns *all* optimal solutions).

Let $\{X^t, t = 1, \ldots, T\}$ be the set of optimal solutions obtained in this manner and $\{Z_t, t = 1, \ldots, T\}$ their images in the objective space. The two possible cases are illustrated respectively in Figures 2 and 3.

• $\underline{\{Z_r, Z_s\} \bigcap \{Z_t, t = 1, \ldots, T\} = \emptyset}$

Solutions X^t are new supported efficient solutions. X^1 and X^T — provided $T > 1$ — are put in S and if $T > 2$, $X^2, \ldots, X^{T-1}$ are put in S'.

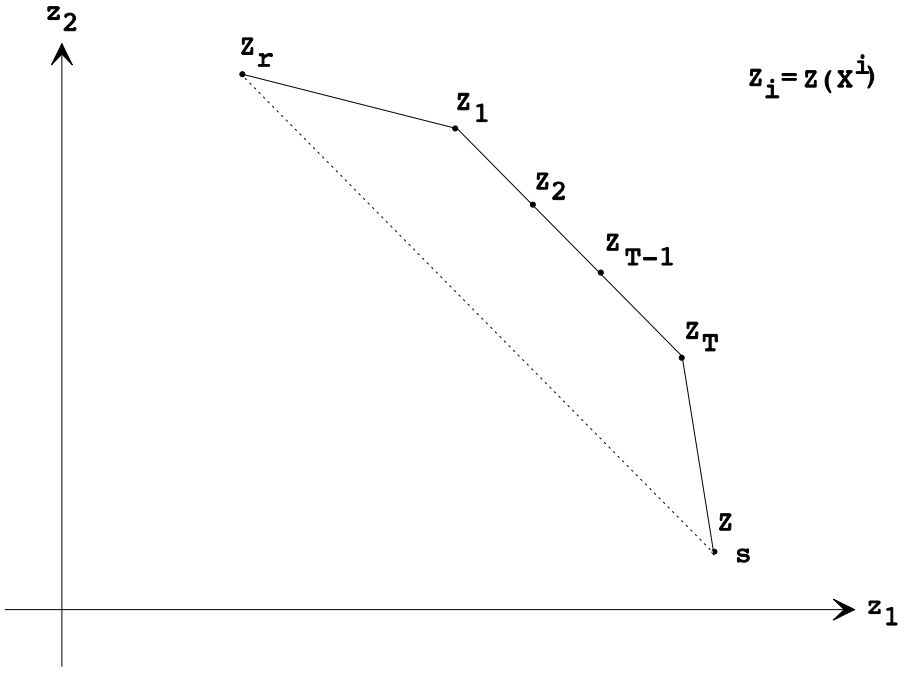

Figure 2. New supported efficient solutions in S

It will be necessary at further steps to consider the pairs (X^r, X^1) and $\left(X^T, X^s\right)$

• $\underline{\{Z_r, Z_s\} \subset \{Z_t, t = 1, \ldots, T\}}$

Solutions $\{X^t, t = 1, \ldots, T\} \setminus \{X^r, X^s\}$ are new supported efficient solutions giving the same optimal value as X^r and X^s for $z_\lambda(X)$; we put them in list S'.

This first phase is continued till all pairs (X^r, X^s) of S have been examined without extension of S.

Finally, we obtain $SE(P) = S \cup S'$ as illustrated in Figure 4

3. Second Phase: a Breadth First Branch and Bound

The purpose is to examine each triangle $\triangle Z_r Z_s$ determined by two successive solutions X^r and X^s of $SE(P)$ and to determine the possible non supported solutions whose image lies inside this triangle.

As previously, we note $z_\lambda(X) = \lambda_1 z_1(X) + \lambda_2 z_2(X)$, with $\lambda_1 = z_{2r} - z_{2s}$ and $\lambda_2 = z_{1s} - z_{1r}$.

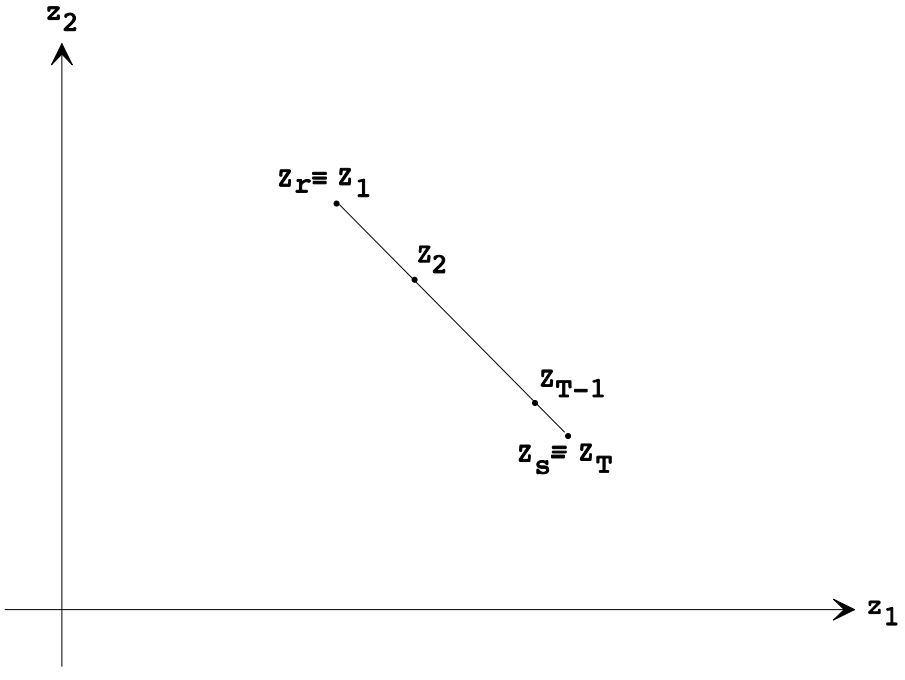

Figure 3. New supported efficient solutions in S'

3.1. LOWER BOUNDS FOR z_λ

3.1.1. *Bound* u_1

Clearly, inside the triangle $\triangle Z_r Z_s$, we must have

$$z_\lambda(X) \geq \lambda_1 z_{1r} + \lambda_2 z_{2s} = u_1$$

(see Figure 5)

3.1.2. *Bound* u_2

Let us suppose that we have now determined some feasible solutions $X_1, \ldots, X_m$ defining points $Z_1, \ldots, Z_m$ inside the triangle (see Figure 5). We note $X_0 \equiv X^r,\ X_{m+1} \equiv X^s$ and $z_k(X_i) = z_{ki},\ k = 1, 2; \quad i = 0, \ldots, m+1$.
Each non dominated point inside the triangle must satisfy (see Figure 5)

$$z_\lambda(X) \geq \max_{i=0\ldots,m} \min\left(\lambda_1 z_{10} + \lambda_2 z_{2i}, \lambda_1 z_{1i} + \lambda_2 z_{2,m+1}\right) = u_2.$$

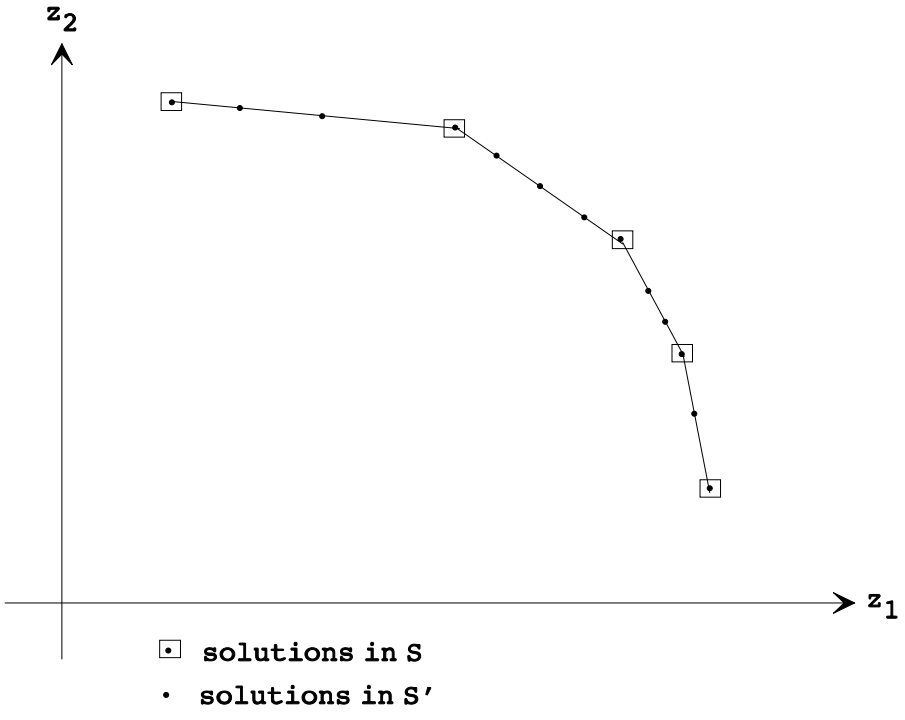

Figure 4. $SE(P) = S \cup S'$

3.1.3. *Bound* u_3

We also have for non dominated points inside the triangle (see Figure 5)

$$z_\lambda(X) \geq \min_{i=0...,m} (\lambda_1 z_{1i} + \lambda_2 z_{2,i+1}) = u_3.$$

It is easy to verify that $u_3 \geq u_2 \geq u_1$.
N.B. Of course, if $m = 0$ then $u_3 = u_2 = u_1$, and if $m = 1$, $u_3 = u_2 > u_1$.

3.2. REDUCTION PROCEDURE

Some variables can be eventually fixed to 0 or 1 to obtain points inside the triangle. For each of the two situations $x_j = 0$ and $x_j = 1$, upper bounds of z_1, z_2 and z_λ are determined, for instance the Martello and Toth's upper bounds (see [4] and [7]).

Let (v_1^{j0}, v_1^{j1}), (v_2^{j0}, v_2^{j1}) and $(v_\lambda^{j0}, v_\lambda^{j1})$ these pairs of upper bounds for each variable x_j.
If $v_1^{j0} \leq z_{1r}$ or $v_2^{j0} \leq z_{2s}$ or $v_\lambda^{j0} \leq u$, then it is necessary to put $x_j = 1$ to be in the triangle. Similarly, if $v_1^{j1} \leq z_{1r}$ or $v_2^{j1} \leq z_{2s}$ or $v_\lambda^{j1} \leq u$, then we fix $x_j = 0$.

So, before getting into the analysis of the triangle, the dimension of the problem can eventually be reduced.

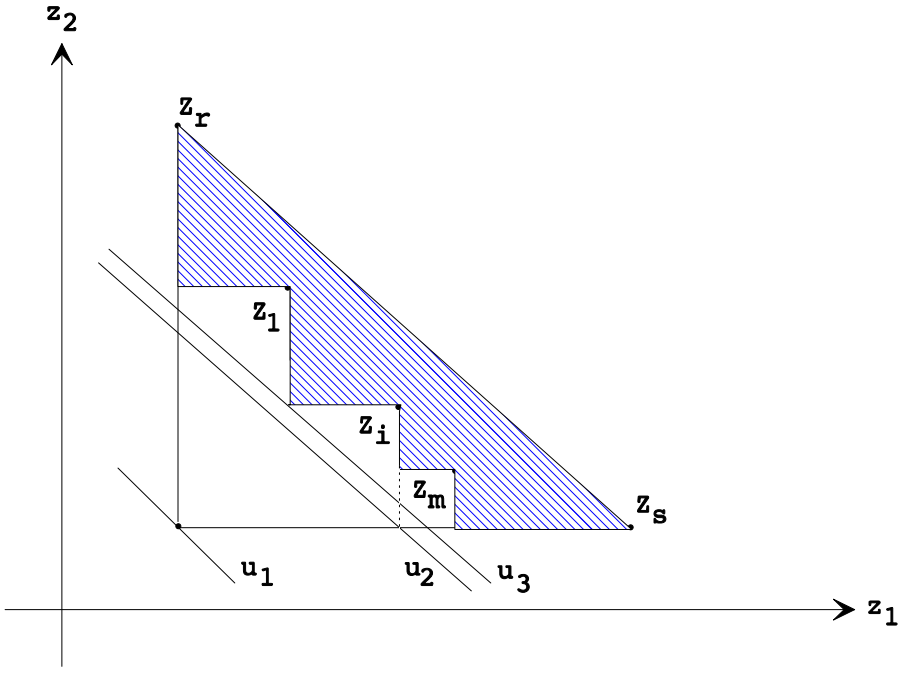

Figure 5. Lower bounds for z_λ

3.3. SEPARATION PRINCIPLE

We associate to the node S_0 of the tree, one of the solutions X^r or X^s, let us say X^s, and we note $\{j|x_j^s = 1\} = \{p_1, \ldots, p_q\}$, the set of indices of variables equal to 1 in this associated solution. We create q subnodes $S_1, \ldots, S_q$ which are characterized by one to q fixed variables in the manner illustrated in Figure 6.

Proceeding like this, the sets of admissible solutions at each subnode are disjoint. Each subnode is examined separately. A list L of solutions will contain potential non supported solutions. Initially, L is empty and at the end of the procedure L, will contain the non supported efficient solutions related to the triangle $\triangle Z_r Z_s$.

3.4. FIRST QUICK FATHOMING TESTS OF A NODE

Inside a node, we compute an upper bound, for instance Martello and Toth bounds, for objectives $z_1,\ z_2$ and z_λ.

Let $v_1,\ v_2$ and v_λ be these bounds. Three fathoming tests are considered.

Test 1: if $v_1 \leq z_{1r}$, the node is fathomed.

Test 2: if $v_2 \leq z_{2s}$, the node is fathomed.

Test 3: if $v_\lambda \leq \mu_3$, the node is fathomed.

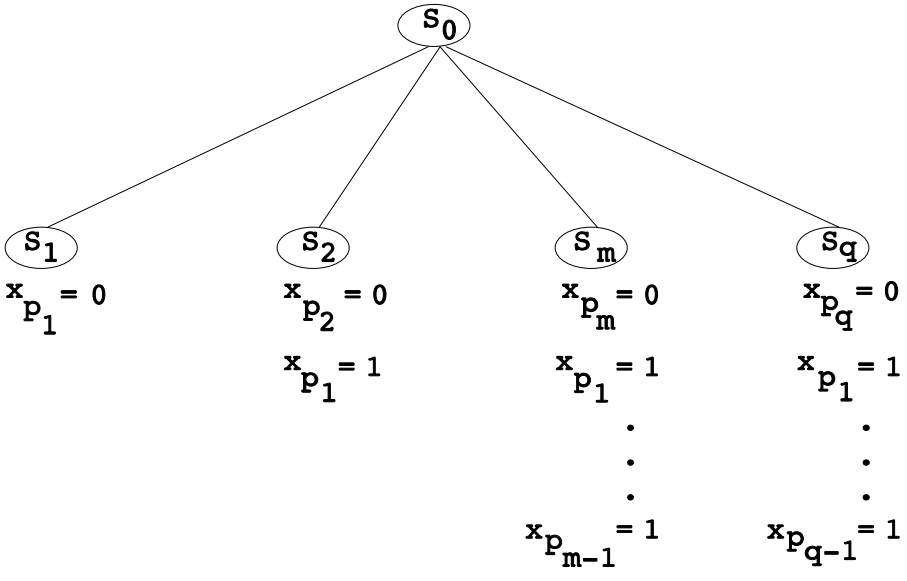

Figure 6. Separation principle

These tests guarantee indeed that the admissible solutions of this node will be outside the triangle $\triangle Z_r Z_s$ as shown in Figure 7.

Note that test 1 (resp. test 2) will be efficient only if Z_r (resp. Z_s) is close to the optimal point for z_1 (resp. for z_2).

3.5. Optimization Inside a Node

Objective $z_\lambda(X)$ is optimized on the considered node. Let $\widetilde{X}$ be the optimal solution with $\widetilde{z}_1 = z_1(\widetilde{X})$, $\widetilde{z}_2 = z_2(\widetilde{X})$ and $\widetilde{z}_\lambda = z_\lambda(\widetilde{X})$.

Four different cases can appear (see the four regions of Figure 8).

1) If $\widetilde{z}_\lambda \leq u$, then $\widetilde{X}$ is dominated and the considered subnode is fathomed (region 1).
2) and 3) If $\widetilde{z}_\lambda > u$ and either $\widetilde{z}_1 \leq z_{1r}$ (region 2) or $\widetilde{z}_2 \leq z_{2s}$ (region 3), then $\widetilde{X}$ is dominated but the subnode is not fathomed.
4) Otherwise (region 4), i.e. $\widetilde{z}_\lambda > u$, $\widetilde{z}_1 > z_{1r}$ and $\widetilde{z}_2 > z_{2s}$ simultaneously, $\widetilde{X}$ is a potential efficient point and the node is not fathomed.

– If $\widetilde{X}$ is dominated by a solution of the list L, there is no change in L.
– Otherwise
  - $\widetilde{X}$ is set in L and this list is updated by pairwise comparisons, keeping only in it non dominated solutions by $\widetilde{X}$;
  - the upper bound u (u_2 or u_3) is updated;
  - the reduction procedure (see §3.2) can eventually be repeated as u has been increased.

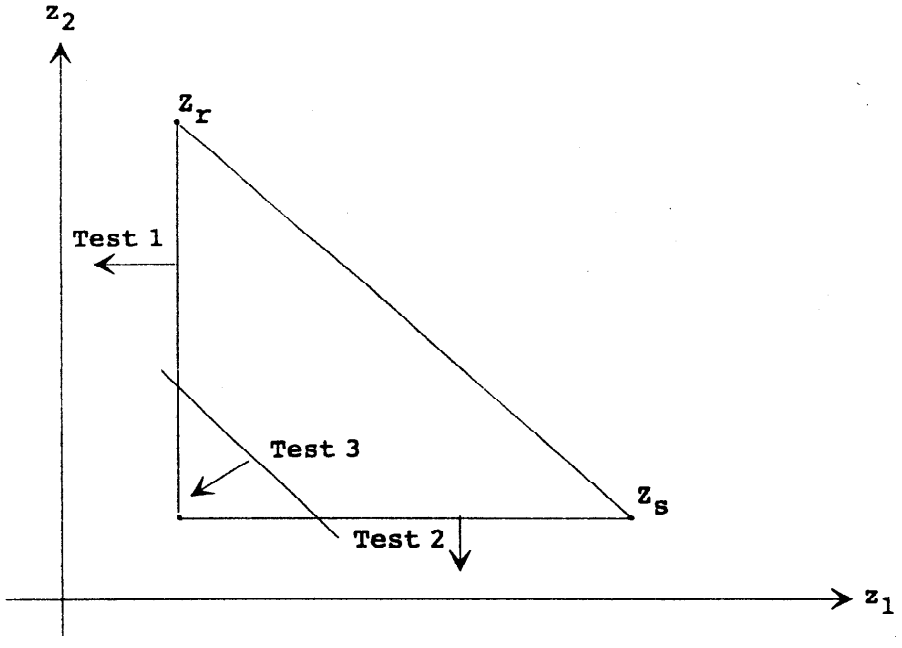

Figure 7. First fathoming tests

3.6. PROGRESSION IN THE TREE

All the subnodes are examined in a "breadth first" manner. Then, between all non fathomed nodes, we choose for being separated the node with the greatest $\widetilde{z}_\lambda$ value. The branching process is made in the same manner as in §3.3 and based on the variables equal to 1 in solution $\widetilde{X}$.

The procedure stops when all the created nodes have been fathomed. By this time, the list L contains the part of $NSE(P)$ related to triangle $\triangle Z_r Z_s$.

4. Second Phase: a Depth First Branch and Bound

The purpose is again to determine the non supported efficient solutions having image inside the triangle $\triangle Z_r Z_s$, for each successive solutions X^r and X^s of $SE(P)$.

The reduction procedure of §3.2 is first applied to possibly fix definitively some variables. As in section 3, a list L is created containing at each step of the procedure, the potential non supported efficient solutions already found.

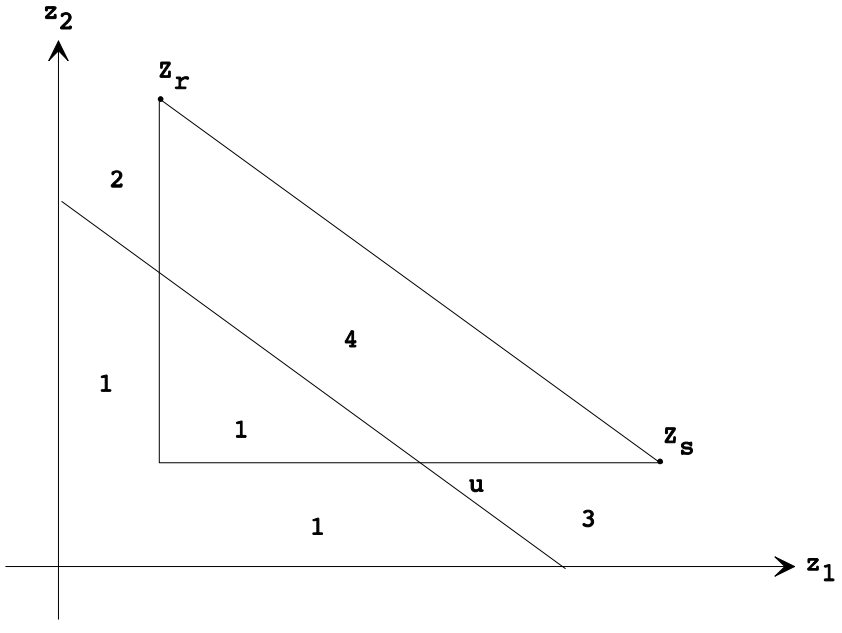

Figure 8. Optimization inside a node

4.1. SEPARATION PRINCIPLE

The variables are ordered by decreasing values of $\frac{\lambda_1 c_j^1 + \lambda_2 c_j^2}{w_j}$ as for the determination of the upper bound of objective z_λ (see [4] or [7]).

At each node S_m, the next variable with regard to this order is fixed to 1 or 0 to create two sub-nodes like in Figure 9

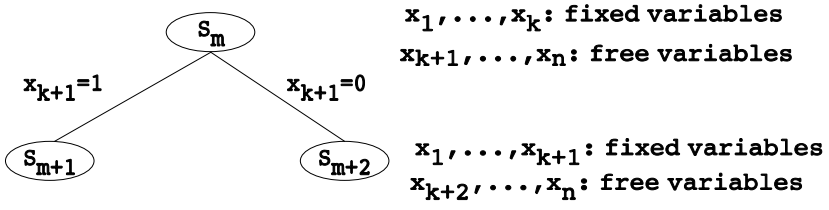

Figure 9. Separation principle

4.2. PROGRESSION IN THE TREE

The sub-nodes are examined in a "depth first" manner.

First we check whether the solution $(x_1, \ldots, x_k, 1, 0, \ldots, 0)$ satisfies the three constraints:

– $\sum_{j=1}^{n} w_j x_j \leq W$ (feasible solution);
– $\sum_{j=1}^{n} c_j^1 x_j \leq z_{1s} - 1$ (necessary condition to be in the triangle);
– $\sum_{j=1}^{n} c_j^2 x_j \leq z_{2r} - 1$ (necessary condition to be in the triangle).

If it is the case, the subnode S_{m+1} is considered. If not, S_{m+1} is fathomed and S_{m+2} is immediately considered.

4.3. Evaluation of a Node

The three fathoming tests of §3.4 are applied.

Test 1: if $v_1 \leq z_{1r}$, the node is fathomed;
Test 2: if $v_2 \leq z_{2s}$, the node is fathomed;
Test 3: if $v_\lambda \leq u$, the node is fathomed,

where u is the lower bound (u_2 or u_3) taking into account the solutions inside the list L. If the node is fathomed, the usual backtracking is applied coming back to the last node of type S_{m+2} not yet evaluated.

4.4. Terminal Nodes

The solution corresponding to a terminal node is possibly an efficient solution. For this:
– the solution must be inside the triangle $\triangle Z_r Z_s$;
– the solution must be non dominated by each solution of L.
If it is the case, this solution is set in L and the list L is updated by pairwise comparisons; the upper bound u is also updated.

5. Numerical Results

All the tests have been made on a PC Pentium 133MHz.

5.1. Impact of the Capacity of the Knapsack

It is well known that the single-objective Knapsack problem is more difficult to solve if the ratio

$$r = \frac{W}{\sum_{j=1}^{n} w_j}$$

is close to the value 0.5. Of course, this is also true in a multi-objective framework as proved by table 1 and figure 10. For r ranging from 0.1 to 0.9, we have generated

20 problems of dimension n=250. Table 1 gives the average number of efficient solutions in each case and the corresponding average time to obtain these solutions with algorithm of section 4 with the bound u_3; figure 10 represents these results graphically.

Table I. Average number of efficient solutions

r	time (sec.)	Solutions
0.1	33.34	168
0.2	94.81	316
0.3	197.3	487
0.4	231.58	564
0.5	246.76	605
0.6	201.87	583
0.7	155.01	504
0.8	70.33	328
0.9	15.65	146

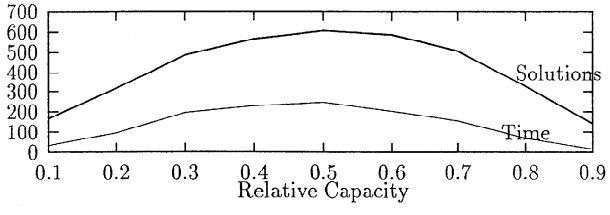

Figure 10. Impact of the capacity

In the following tests, we have always chosen $r = 0.5$.

5.2. Comparison of Algorithms and Bounds

For the dimension n of the problem varying from $n = 10$ till $n = 500$ by steps of 10, we have randomly generated 20 problems for each dimension n. These problems have been solved by the "breadth first" and "depth first" algorithms; to analyse the impact of the bound u, the two algorithms have been applied successively with $u = u_2$ and $u = u_3$. Table 2 gives the mean number of supported and non supported efficient solutions for each dimension n and Figure 11 represents these results graphically.

Clearly, the number of supported efficient solutions increases very slowly compared with the number of non supported efficient solutions. This yields a clear justification for investigating $NSE(P)$.

Table II. Number of solutions

r	Total	Sup.	Non Sup.
10	4	2	1
20	8	4	4
30	17	6	11
40	28	8	20
50	39	10	28
60	53	12	41
70	80	14	55
80	96	17	79
90	106	17	88
100	134	18	116
110	152	20	131
120	183	23	160
130	218	24	194
140	232	26	205
150	259	27	231
160	279	29	249
170	305	30	274
180	351	34	317
190	368	36	332
200	410	36	374
210	437	39	398
220	469	40	428
230	504	42	462
240	565	43	521
250	605	47	558
260	600	47	553
270	659	46	613
280	688	50	637
290	688	49	639
300	834	55	778
310	876	56	820
320	847	55	792
330	966	60	905
340	967	60	907
350	1017	62	954
360	1063	63	1000
370	1151	66	1084
380	1153	66	1087
390	1231	66	1164
400	1198	69	1128
410	1327	74	1253
420	1403	73	1330
430	1382	76	1306
440	1525	77	1447
450	1516	78	1438
460	1567	80	1487
470	1581	80	1501
480	1785	83	1701
490	1724	83	1641
500	1778	86	1691

Table III. Computing time

n	t1	t2	t3	t4	t5
10	0.00	0.00	0.00	0.00	0.00
20	0.00	0.04	0.03	0.03	0.02
30	0.02	0.23	0.16	0.19	0.13
40	0.10	0.85	0.57	0.54	0.38
50	0.21	3.48	2.22	1.18	0.85
60	0.37	10.32	6.26	2.76	1.93
70	0.50	41.37	23.55	5.19	3.23
80	0.76	67.20	38.35	6.97	4.37
90	0.92	69.15	41.40	8.68	5.43
100	1.26	765.88	394.75	16.60	9.69
110	1.66	1533.44	616.08	22.40	13.19
120	2.24	1766.65	967.85	30.31	17.05
130	2.54			48.29	27.13
140	3.25			51.59	28.16
150	3.82			68.50	35.70
160	5.03			88.54	46.53
170	5.32			108.48	56.11
180	6.73			135.89	71.11
190	7.90			179.68	93.31
200	8.34			240.06	111.44
210	9.72			224.05	117.78
220	11.05			282.50	142.53
230	13.58			365.68	186.79
240	13.65			411.84	191.32
250	15.86			494.24	246.76
n	t1	t2	t3	t4	t5
260	17.41			533.67	258.03
270	18.53			649.18	300.87
280	22.41			778.74	338.96
290	20.51			748.60	339.12
300	25.99			1085.69	490.25
310	31.25			1300.65	545.21
320	31.51			1203.40	554.60
330	36.84			1837.61	718.51
340	36.28			1813.46	761.89
350	41.04			2104.81	828.95
360	42.47			2322.86	966.00
370	47.54			2460.16	1019.36
380	48.66			2680.20	1119.61
390	51.87			3236.04	1376.35
400	58.18			3146.48	1279.71
410	65.05				1621.67
420	68.18				1737.17
430	73.46				1647.87
440	71.38				1955.72
450	78.99				2067.49
460	82.01				2244.20
470	88.81				2303.04
480	94.31				2790.25
490	99.57				2678.21
500	105.29				3222.60

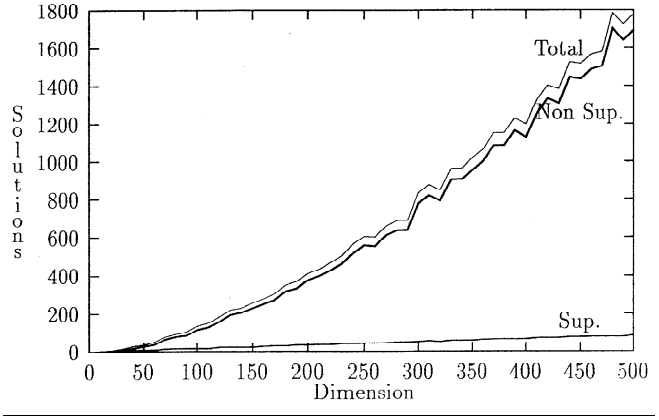

Figure 11. Number of efficient solutions

Table 3 indicates the average computing time to obtain these solutions; the five columns correspond respectively to
t_1: the first phase,
t_2: the second phase with "breadth first BB" and bound u_2,
t_3: the second phase with "breadth first BB" and bound u_3,
t_4: the second phase with "depth first BB" and bound u_2,
t_5: the second phase with "depth first BB" and bound u_3.

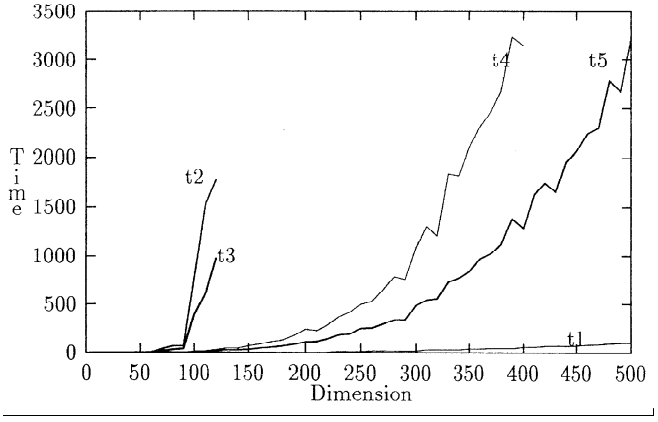

Figure 12. Computing time

Figure 12 is the graphical representation of these times and Figure 13 shows the time-ratio $\frac{t_1}{t_5}$. It appears from these results that

– the main computing time effort must be devoted to generate $NSE(P)$;
– the bound u_3 is definitely more efficient than bound u_2;

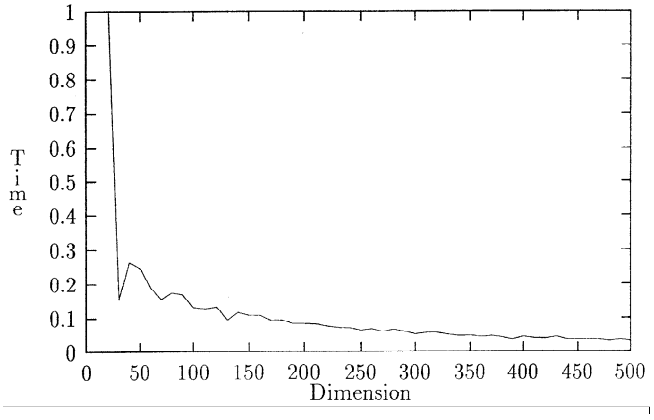

Figure 13. Time proportion: First phase/ Second phase

– the second phase "depth first BB" (section 4) is much faster than the "breadth first BB" (section 3).

The reasons of the better performance of the "depth first BB" second phase seems to be that, contrary to the other algorithm,

– it doesn't call frequently the single-objective algorithm;

– it doesn't generate solutions outside the triangles $\triangle Z_r Z_s$.

Nevertheless, the "breadth first BB" is more general and can be applied to any bi-objective combinatorial optimization problem for which an efficient single-objective algorithm is available. Our current research in this field is oriented in four directions:

– the separation principle (section 3.3) suggests that the algorithm can be efficiently implemented in a parallel environment (see for instance the recent book [3]).

– some heuristics are developed to approximate $SE(P)$ for very large dimension problems. One of them is an adaptation of Simulated Annealing to a multi-objective framework and another one is based on polyhedral theory.

– for practical applications, it is not necessary to generate $SE(P)$ but to determine a "good compromise" in regard of the preferences of the decision-maker. For this purpose, we are also developing interactive methods.

– Finally other Multi-Objective Combinatorial Optimization problems are also analysed.

References

1. Czyzak, P. and Jaszkiewicz, A. (1995), Pareto Simulated annealing – a Meta-heuristic Technique for Multiple Objective Combinatorial Optimization, Technical report, Institute of Computing Science, Poznan University of Technology. Submitted for publication.
2. Dyer, R., Foreman, E. and Mustafa, M. (1988), Decision support for media selection, Technical report, Georges Washington University. Presented at the Third International Summer School on MCDM held at Estoril (Portugal).

3. Ferreira, H. and Pardalos, P. (1996), *Solving Combinatorial Optimization Problems in Parallel*, Vol. 1054 of *Lecture notes in computer science*, Springer-Verlag.
4. Martello, S. and Toth, P. (1990), *Knapsack Problems: Algorithm and Computer Implementation*, Wiley.
5. Rosenblatt, M. and Sinuany-Stern, Z. (1989), Generating the discrete efficient frontier to the capital budgeting problem, *Operations Research* **37**(3), 384–394.
6. Ulungu, E. and Teghem, J. (1994), Multi-objective combinatorial optimization: A survey, *Journal of Multi-Criteria Decision Analysis* **3**(2), 83–104.
7. Ulungu, E. and Teghem, J. (1995), The two-phases method: An efficient procedure to solve bi-objective combinatorial optimization problems, *Foundations of Computing and Decision Sciences* **20**(2), 149–165.
8. Ulungu, E. and Teghem, J. (1997), *Solving multi-objective knapsack problems by a branch-and-bound procedure* in Multicriteria Analysis, J.N. Climaco (ed.), 269–278 Springer-Verlag.
9. Ulungu, E., Teghem, J. and Fortemps, Ph. (1995), Heuristics for multi-objective combinatorial optimization by simulated annealing, in *Multiple Criteria Decision Making: Theory and Application*. Proceedings of the 6th National Conference on MCDM, Beijing (China).