A SUBGRADIENT ALGORITHM FOR THE MULTIDIMENSIONAL 0-1 KNAPSACK PROBLEM(*)

Luiz Antonio N. Lorena - INPE/MCT
Acioli Antonio de Olivo - INPE/MCT
Paulo Roberto Oliveira - IM/COPPE/UFRJ

ABSTRACT

A subgradient method is applied to approximate a solution for the multidimensional 0-1 knapsack problem (MKP). At each iteration a Lagrangean problem with a cut constraint for the objective function of (MKP) is solved.

In computational tests with problems taken from the literature, this algorithm provided bounds that were very close to the optimal solution (less than 1% error), a performance that is comparable with the best known algorithms. Tests made with random generated problems also presented excellent results.

RESUMO

Aplica-se um método de subgradientes para aproximar a solução do problema multidimensional da mochila 0-1 (PMM). A cada iteração resolve-se um problema Lagrangeano com restrição de corte na função objetivo do problema (PMM).

Em testes computacionais com problemas de literatura, o algoritmo apresentou limitantes muito próximos do valor ótimo do (PMM) (menos de 1% de erro), uma performance comparável aos melhores algoritmos conhecidos. Testes realizados com problemas gerados aleatoriamente também apresentarem resultados excelentes.

*This paper has been accepted for presentation at the 11th Triennial Conference on Operations Research, 10-14 August, 1987, Buenos Aires, Argentina.

1. INTRODUCTION

The multidimensional knapsack problem can be stated as

$$(P) \quad \begin{array}{ll} v = \max & cx \\ \text{subj. to} & Ax \le b \\ & x \in \{0,1\}^n, \end{array}$$

for $c \in N_*^n$, $b \in N_*^m$, and $A \in R_+^{m \times n}$.

As problem (P) is NP-hard [M. Garey; D. Johnson; 1979] it is very unlikely the existence of "good" algorithms for its solution. So, many heuristical methods had been proposed to approximate the value v in (P), like those in [A. Freville; G. Plateau, 1986], [M.J. Magazine; O. Oguz; 1984], [E. Balas; C.M. Martin, 1980], [R. Loulou; E. Michaelides, 1979], [G.A. Kochenberger; B.A. McKarl; F.P. Wyman, 1974], [F.S. Hillier, 1969] and [S. Senju; Y. Toyoda, 1968].

In this paper we propose a new subgradient algorithm to approximate the value v in (P), using as a subproblem at each iteration the following Lagrangean problem, with a cut constraint for the objective function of (P)

$$(LP) \quad \begin{array}{ll} \max & \{cx - \lambda^{(k)} (Ax-b)\} \\ \text{subj. to} & cx \le I^{(k)} - 1 \\ & x \in \{0,1\}^n, \end{array}$$

where k means the algorithm iteration, $\lambda^{(k)} \ge 0$, $I^{(k)} = cx^{(k-1)}$, $k \ge 1$ and $I^{(0)}$ is any upper bound on v.

Next we present the algorithm and some computational results.

2. THE METHOD

The algorithm can be stated as follows:

Algorithm

Initialization: given $I^{(0)}$, $\lambda^{(0)}$,

set $\bar{x} = 0$,
CONT=0,
$\ell = (m+n)/2$,
$k = 1$.

Step 1: Solve the (LP) problem obtaining $x^{(k)}$.

Step 2: If $x^{(k)}$ is feasible for problem (P) then

If $c\bar{x} \geq cx^{(k)}$ then $x^{(k)} = \bar{x}$,
CONT = CONT+1,
$I^{(k)} = I^{(0)}$;

Else $\bar{x} = x^{(k)}$,
$I^{(k)} = I^{(0)}$,
CONT = CONT+1;

Else set $\lambda_i^{(k+1)} = \max\left[0, \lambda_i^{(k)} + \dfrac{(Ax^{(k)} - b)_i}{(\ell+1).b_i}\right]$,

for $i=1, \ldots, m$,
$I^{(k+1)} = I^{(k)} - 1$;

Step 3: If CONT≥10 stop with the feasible solution $x^{(k)}$,

Else, set $k = k+1$ and return to step 1.

Comments: $I^{(0)}$ is an upper bound on v, for example, the value of the objective function in the linear programming relaxation of (P). The values of $\lambda^{(0)}$ used in computational tests were the m-vectors $(0.1, 0.1, \ldots, 0.1)^T$, $(0.2, 0.2, \ldots, 0.2)^T$ and $(0.5, 0.5, \ldots, 0.5)^T$. $\bar{x}$ is an intermediate feasible solution (note that $\bar{x}=0$ is feasible for (P)) and CONT is the number of feasible solutions found (some of them may be repeated).

Hence the algorithm acts as follows: for any iteration k (step 1) we solve (LP) obtaining $x^{(k)}$. If $x^{(k)}$ is feasible for (P), we update the intermediate feasible solution $\bar{x}$, otherwise the variable $\lambda^{(k)}$ is updated (step 2). The process can be repeated (depending on CONT) with the current $\lambda^{(k)}$ and the initial $I^{(0)}$ in an attempt to find a better feasible solution, $cx^{(k)} > c\bar{x}$ (steps 2 and 3).

The success of this algorithm depends on finding a "good" feasible solution to (P) as the result of its application. The subgradient part of the algorithm plays an important role in this context.

Next we prove that the algorithm always finds a feasible solution to (P).

Proposition: The algorithm always finds a feasible solution to (P).

Proof: At any iteration k in step 1 results $x^{(k)}$, and $cx^{(k)} \leq cx^{(k-1)} - 1$, i.e., this cut removes the $x^{(k-1)}$ solution and other solutions, feasibles or not. Therefore, as the process is finite, in the worst case, $x^{(k)}=0$ is found in step 2 (a feasible solution to (P) as we know). ∎

Next, we'll see how the algorithm finds a "good" feasible solution to (P).

At any iteration k, for $i=1,\ldots,m$, we have

$$\lambda_i^{(k)} = \max \left[0, \lambda_i^{(k-1)} + \frac{(Ax^{(k-1)}-b)_i}{(\ell+1).b_i} \right],$$

$$\lambda_i^{(k-1)} = \max \left[0, \lambda_i^{(k-2)} + \frac{(Ax^{(k-2)}-b)_i}{(\ell+1).b_i} \right],$$

$$\vdots \qquad\qquad \vdots \qquad\qquad \vdots$$

$$\lambda_i^{(1)} = \max \left[0, \lambda_i^{(0)} + \frac{(Ax^{(0)}-b)_i}{(\ell+1).b_i} \right],$$

and then,

$$\lambda_i^{(k)} = \max \left\{ 0, \lambda_i^{(j)} + \frac{1}{(\ell+1).b_i} \left[(Ax^{(j)}-b)_i + \ldots + (Ax^{(k-1)}-b)_i \right] \right\}.$$

Where j is such that $\lambda_i^{(l)} > 0$ for $l=j+1,\ldots,k-1$.

The constant $[(\ell+1).b_i]^{-1}$ realizes only the scaling of the sum in brackets, in order to avoid jumps in $\lambda_i^{(0)}$ changes. Then, for example, if $\lambda_i^{(0)} = 0.1$, the sum in brackets multiplied by the constant will be of a centesimal order.

The subgradients $(Ax^{(j)}-b)_i$, $0\le j\le k-1$, are non-positive when the i-th constraint of (P) is feasible, and positive otherwise. Then, if the i-th constraint is feasible for each $x^{(j)}$, $0\le j\le k-1$, probably $\lambda_i^{(k)}\to 0$ (remember that if $\lambda_i^{(k)}\le 0$, then we set $\lambda_i^{(k)}=0$). This may happen even with some positive subgradients in the interval $0\le j\le k-1$.

In the same way $\lambda_i^{(k)}$ may be greater than $\lambda_i^{(0)}$, obviously with the contribution of many positive subgradients $(Ax^{(j)}-b)_i$, $0\le j\le k-1$.

Therefore, in general $\lambda_i^{(k)}$, $i=1,\ldots,m$, will be always non-negative and remains of the same order of $\lambda_i^{(0)}$, $i=1,\ldots, m$.

Now, consider the objective function of problem (LP):

$$\sum_{j=1}^{n} c_j x_j - \sum_{i=1}^{m} \lambda_i^{(k)} \left(\sum_{j=1}^{n} a_{ij} x_j - b_i \right).$$

We note that:

(a) If $\lambda_i^{(k)}=0$, $1\le i\le m$, the i-th constraint of (P) doesn't make any influence in the objective function of (LP), and according to the previous observations, the i-th constraint was feasible for some solutions $x^{(j)}$, $0\le j\le k-1$;

(b) If $\lambda_i^{(k)}>0$, the i-th constraint of (P) contributes to the objective function of (LP). As (LP) is a maximization problem, the feasibility of the i-th constraint is desirable, i.e., $\sum_{j=1}^{n} a_{ij} x_j - b_i\le 0$. Several solutions of (P) are eliminated by the cut $cx\le I^{(k)}-1$, and possibly some feasible solutions among them.

(c) Hence there exists a compromise solution to (LP), that results in the maximization of $\sum_{i=1}^{n} c_i x_i$ subject to the cut $cx \leq I^{(k)}-1$ and a search for the feasibility of (P).

A solution with $\sum_{j=1}^{n} a_{ij} x_j - b_i << 0$ imply that only as few x_j's will be 1 (one), and this makes opposition to the maximization of $\sum_{j=1}^{n} c_j x_j$. Hence, a minimal complementary slackness solution is seeked, and because of the formulation of problem (P), this search might provide a "good" feasible solution for (P).

3. COMPUTATIONAL RESULTS

The algorithm was coded in FORTRAN IV and a lot of computational tests were carried out on a Burroughs B-6800 Computer. For the solution of the (PL) problem, we use the very efficient algorithm of D. Fayard and G. Plateau [D. Fayard; G. Plateau, 1982].

Table 1 presents the results obtained by our algorithm and ten algorithms of the literature. The problems tested are of Petersen [C.C. Petersen, 1967], Weingartner [H.M. Weingartner; D.N. Ness, 1967] and Senju & Toyoda [S. Senju; Y. Toyoda, 1968]; and are of size (mxn) 5x39, 5x50, 2x28, 2x105, 30x60, 30x60, respectively. Numbers in brackets are the percentages of the optimal value founded by the corresponding algorithm. The results of Magazine's algorithm were obtained from his work [M. Magazine; O. Oguz, 1984] and the other results appear in A. Freville and G. Plateau [A. Freville; G. Plateau, 1986].

Note that our algorithm had a performance that is comparable with the best known algorithms.

Our algorithm was also tested in some randomly generated problems as showed in table 2. These problems were obtained as follows: the A and c coefficients were randomly generated in the interval [0,100], and b coefficients obtained

TABLE 1

RESULTS FOR PROBLEMS TAKEN FROM THE LITERATURE

	PETERSEN 5 X 39	PETERSEN(*) 5 X 30	WEINGARTER 2 X 28	WEINGARTER 2 X 105	SENJU & TOYODA (*) 30 X 60	SENJU & TOYODA(*) 30 X 60
Lorena, Blivo, Oliveira (1987)	18547 (99.33)	16436 (99.38)	141278 (100)	1894992 (99.95)	2706 (99.15)	8716 (99.93)
Mogazino, Oguz (1984)	18354 (97.51)	16261 (98.33)	139418 (98.68)	1892971 (99.77)	7719 (99.31)	8623 (98.86)
Freville, Plateau (Agnes 1, 1986)	10552 (99.37)	16499 (99.77)	148568 (99.49)	1895445 (100)		
Freville, Plateau (Agnes 2, 1986)	18618 (100)	16447 (99.45)	141278 (100)	1094806 (99.94)		
Martin, Oalas (1988)	18588 (99.71)	16499 (99.77)				
Senju, Toyoda (1968)	18518 (99.03)	16337 (98.79)	138878 (98.3)	1094611 (99.92)		
Kochenberguer (1974)	18313 (97.12)	16831 (96.94)	148618 (99.53)	1095332 (99.98)		
Hillior (1969)	8255 (72.24)	16499 (99.77)	127244 (90.06)	1088122 (99.33)		
Toyoda (1975)	9888 (93.12)	15897 (96.12)	139278 (98.58)	998337 (98.48)		
Louiou, Michaelides (1979)	18199 (96.05)	15897 (96.12)	141278 (100)	991921 (90.54)		
Guinard (1972)	10427 (98.28)	16274 (98.40)				

*Optimal values: 16537 , 7772 and 8722.

TABLE 2

RESULTS FOR RANDOM GENERATED PROBLEMS

PROBLEM	SIZE (MXN)	Y	LINEAR PROGRAMMING	BRANCH & BOUND	TIME B & B (SEC.)	LORENA OLIVO OLIVEIRA	TIME (sec.)
1	20x20	.1	159.06	99	20	99	4
2	20x50	.1	440.6	386	210	386	7
3	20x75	.1	671.27	568	720(*)	626	13
4	20x100	.1	902.31	790	720(*)	843	14
5	20x20	.5	716.26	679	41	679	2
6	20x50	.5	1878.9	1828	720(*)	1843	20
7	20x75	.5	2837.45	2770	720(*)	2770	15
8	20x100	.5	3544.62	3119	720(*)	3494	8

(*) Best value in 12 minutes of CPU (Burroughs 6800).

adding the corresponding A rows coefficients and multiplying the result by $\gamma=0.1$ or $\gamma=0.5$ (as in Lorena e Olivo [L.A.N. Lorena; A.A. de Olivo, 1986]). This tests also presented excellents results, comparing with a branch-and-bound algorithm (MPS Tempo of Burroughs).

4. CONCLUSION

Although we didn't present a formal prof that our algorithm finds a "good" feasible solution to (P), the computational results showed it's very satisfactory performance. The algorithm can be extended to other zero-one optimization problems.

Note that the method presented may indicate that one solves at each iteration, the "dual" problem

$$(DP)\quad \min_{\lambda \geq 0} \left\{ \begin{array}{ll} \max\limits_{x \in \{0,1\}^n} & cx - \lambda(Ax-b) \\ \text{Subj. to} & cx \leq I^{(k)} - 1 \end{array} \right\}$$

This is not true, because the $I^{(k)}$ bounds are updated together with $\lambda^{(k)}$. Hence, at each iteration, the Lagrangean problem is modified in its cut constraint.

The subgradient algorithm used is also different of the traditional ones [M. Fisher, 1981], due mainly to the λ's updating and the stopping rule.

As a final comment we observe that all these algorithms (of Table 1) may have a poor performance when the c_j's are very small in (P). This is so because a small difference relative to the optimal value implies in a high percentual error, for the optimal value is low.

5. REFERENCES

BALAS, E.; MARTIN, C.M. "Pivot and complement - a heuristic for 0-1 programming". Management Science Research, 26(1):86-96, 1980.

FAYARD, D.; PLATEAU, G. "An algorithm for the solution of the 0-1 knapsack problem". Computing, 28:269-287, 1982.

FISHER, M. "The Lagrangean Relaxation method for solving Integer Programming Problems". Management Science, 17:581-615, 1981.

FREVILLE, A.; PLATEAU, G. "Heuristics and reduction methods for multiple constraints 0-1 linear programming problems". European Journal of Operational Research, 24:206-215, 1986.

GAREY, M.; JOHNSON, D. "Computers and Intractability: a guide to the theory of NP-completeness, Freeman, San Francisco, 1979.

GUIGNARD, M. "Methods heuristics de résolution d'un système d'inégalités linéaires en variables entières ou bivalentes". Report 32, Laboratoire de Calcu de Lille, January 1972.

HILLIER, F.S. "Efficient heuristic procedures for integer linear programming with an interior path". Operations Research, 17: 600-636, 1969.

KOCHENBERGER, G.A.; Mc KARL, B.A.; WYMAN, F.P. "A heuristic for general integer programming". Decision Sciences, 5(1):36-44, 1974.

LORENA, L.A.N.; OLIVO, A.A. "Uso de uma função penalidade exata para eliminação do 'gap' de dualidade em problemas de programação linear inteira zero-um". Presented in III CLAIO (Congreso Latino-Iberoamericano de Investigacion Operativa e Ingenieria de Sistemas). Santiago, Chile, agosto 1986.

LOULOU, R.; MICHAELIDES, E. "New greedy-like heuristics for the multidimensional 0-1 knapsack problem". Operations Research, 27(6):1101-1114, 1979.

MAGAZINE, M.J.; OGUZ, O. "A heuristic algorithm for the multidimensional zero-one knapsack problem". European Journal of Operational Research, 16:319-326 (1984).

PETERSEN, C.C. "Computational experience with variants of the Balas algorithm applied to the selection of R and D projects". Management Science, 13(9):736-750, 1967.

SENJU, S.; TOYODA, Y. "An approach to linear programming with 0-1 variables". Management Science, 15(4):196-207, 1968.

TOYODA, Y. "A simplified algorithm for obtaining approximate solutions to 0-1 programming problems". Management Science, 21(12):1417-1427, 1975.

WEINGARTNER, H.M.; NESS, D.N. "Methods for the solution of the multidimensional 0-1 knapsack problem". Operations Research, 15(1):83-103, 1967.