World Academy of Science, Engineering and Technology
International Journal of Mathematical and Computational Sciences
Vol:4, No:7, 2010

A New heuristic approach for large size Zero-One Multi Knapsack Problem using intercept matrix

K. Krishna Veni and S.Raja Balachandar

Abstract—This paper presents a heuristic to solve large size 0-1 Multi constrained Knapsack problem (01MKP) which is NP-hard. Many researchers are used heuristic operator to identify the redundant constraints of Linear Programming Problem before applying the regular procedure to solve it. We use the intercept matrix to identify the zero valued variables of 01MKP which is known as redundant variables. In this heuristic, first the dominance property of the intercept matrix of constraints is exploited to reduce the search space to find the optimal or near optimal solutions of 01MKP, second, we improve the solution by using the pseudo-utility ratio based on surrogate constraint of 01MKP. This heuristic is tested for benchmark problems of sizes upto 2500, taken from literature and the results are compared with optimum solutions. Space and computational complexity of solving 01MKP using this approach are also presented. The encouraging results especially for relatively large size test problems indicate that this heuristic can successfully be used for finding good solutions for highly constrained NP-hard problems.

Keywords—0-1 Multi constrained Knapsack problem, heuristic, computational complexity, NP-Hard problems.

I. Introduction

The multi constraints 0-1 knapsack problem (01MKP) has varied applications in various fields, e.g. economy: Consider a set of projects (variables) $(j = 1, 2, 3..., n)$ and a set of resources (constraints) $(i = 1, 2, 3, ..., m)$. Each project has assigned a profit c_j and resource consumption values a_{ij}. The problem is to find a subset of all projects leading to the highest possible profit and not exceeding given resource limits b_i

The 0-1 multi-constrained knapsack problem (01MKP) is a well known NP-Hard combinatorial optimization problem [10] which can be formulated as follows

Maximize

$$f(x_1, x_2, ..., x_n) = \sum_{j=1}^{n} c_j x_j \tag{1}$$

subject to the constraints

$$\sum_{j=1}^{n} a_{ij} x_j \leq b_j, i = 1, 2, ..., m \tag{2}$$

$$x_j \in \{0, 1\}, j = 1, 2, 3, ..., n \tag{3}$$

$$c_j > 0, a_{ij} \geq 0, b_j \geq 0 \tag{4}$$

K. Krishna Veni (corresponding author) is with the Department of Mathematics, SASTRA University,Thanjavur,INDIA, e-mail: kkveni@maths.sastra.edu

S.Raja Balachandar is with the Department of Mathematics, SASTRA University,Thanjavur,INDIA, e-mail: srbala@maths.sastra.edu

The objective function $f(x_1, x_2, ..., x_n)$ should be maximized subject to the constraints given by (2) . For 01MKP problems the variablex_j can take only two values 0 and 1. Here in a 01MKP, it is necessary that a_{ij} are positive. This necessary condition paves a way for better heuristics to obtain optimal or near optimal solutions.

The popularity of knapsack problems stems from the fact that it has attracted researchers from both camps: the Theoreticians as wells as the Practicians [23] enjoy the fact that these simple structured problems can be used as sub problems to solve more complicated ones. Practicians on the other hand, enjoy the fact that these problems can model many industrial opportunities such as cutting stock, cargo loading, and processor allocation in distributed systems. The special case of 01MKP with $m = 1$ is the classical knapsack problem (01KP) , whose usual statement is the following. Given a knapsack of capacity b and n objects, each being associated a profit a volume occupation , one wants to select k $(k <= n$ and k not fixed) objects such that the total profit is maximized and the capacity of the knapsack is not exceeded . It is well known that 01KP is strongly NP-Hard because there are polynomial approximation algorithms to solve it. This is not the case for the general 01MKP. Various algorithms to obtain exact solutions of such problems were designed and well documented in literature [2,6,11,12,16,30,35].

This paper is organized as follows: A brief survey of various researchers work pertaining to this problem is elucidated in section 2. The dominant principle of intercept matrix, Dominance principle based Heuristic (DPHEU) approach for solving 01MKP; surrogate constraint, pseudo-utility operator, and computational complexity of DPHEU are explained in section 3. We have furnished the results obtained by DPHEU for all the benchmark problems in section 4. This section also includes the extensive comparative study of results of our heuristic with known optimum or best solutions of 01MKP. Salient features of this algorithm are also enumerated in section 4, and concluding remarks and future direction are also given in section 5.

II. Previous Work

Exact and heuristic algorithms have been developed for the 01MKP, like many NP-Hard combinatorial optimization problems.

A. Exact algorithms

Existing exact algorithms are essentially based on branch and bound method, dynamic programming, systematic approach and 01MKP relaxation techniques such as Lagrangian, surrogate and composite relaxations.[2,6,11,12,16,30,35]. The Balas's [2] algorithm is a systematically designed one that begins with a null solution which assigns successively certain variables to 1 in such way that after testing part of all the 2^n combinations, we obtain either an optimal solution or a proof that no feasible solution exists. The algorithms by Gilmore and Gomory [12], Weingartner and Ness[35] use dynamic programming approach. The dynamic programming method solved two problems of size (n=28,m=2) and (n=105,m=2) in a forward and backward approach. Shih [30] proposed a branch and bound algorithm. In this method, an upper bound is obtained by computing the linear relaxation upper bound of each of the m single constraint knapsack problems separately and selecting the minimum objective function value among those as the upper bound. Computational results showed that this algorithm performed better than the general zero-one additive algorithm of Balas[2]. Better algorithms have been proposed by using tighter upper bounds, obtained with other 01MKP relaxation techniques such as Lagrangian, Surrogate and composite relaxations were developed by Gavish and Pirkul[11] . This algorithm was compared with the Shih's method [30] and was found to be faster by at least one order of magnitude. Osorio et al[26] used surrogate analysis and Constraint Pairing to assign some variables to zero and to separate the rest of the variables into groups(those that tend to zero and those that tend to one) They use an initial feasible integer solution to generate logical cuts based on their analysis at the root of a branch and bound tree. Due to their exponential time complexity, exact algorithms are limited to small size instances ($n = 200$ and $m = 5$).

B. Heuristic algorithms

Heuristic algorithms are designed to produce near-optimal solutions for larger problem instances. The first heuristic approach for the 01MKP concerns for a large part greedy methods. These algorithms construct a solution by adding, according to a greedy criterion, one object each time into a current solution without constraint violation.

The second heuristic is based on linear programming by solving various relaxations of the 01MKP. Balas and Martin [3] introduced a heuristic algorithm for the 01MKP which utilizes linear programming (LP) by relaxing the integrality constraints x_j = 0 or 1 to x_j= 0 to 1. Linear programming problems are not NP-hard and can be solved efficiently,e.g. with the well known Simplex algorithm. The fractional x_j are then set to 0 or 1 according to a heuristic which maintains feasibility. In last decade, several algorithms based on meta-heuristics have been developed, including simulated annealing [9], tabu search [13,17] and genetic algorithms [8,20]. More examples of heuristic algorithms for the 01MKP can be found in [22,23,28,34]. Sartaj sahini[29] presented a sequence of ϵ approximate algorithm and time estimates of such algorithm were also presented. A comprehensive review on exact and heuristic algorithms is given in [1,7,8],

This paper, we propose a heuristic algorithm based on dominance principle of intercept matrix to solve 01MKP. The main principle of the algorithm is twofold. (i) to find the optimal or near optimal solution of 01MKP by using dominance principle of intercept matrix (ii) to improve the near optimal solution by using surrogate pseudo-utility ratio of 01MKP.

III. Dominance Principle (DP)

Linear programming (LP) is one of the most important techniques used in modeling and solving practical optimization problems that arise in industry, commerce and management. Linear programming problems are mathematical models used to represent real life situations in the form of linear objective function and constraints various methods are available to solve linear programming problems. When formulating an LP model, systems analyst and researchers often include all possible constraints and variables although some of them may not be binding at the optimal solution. The presence of redundant constraints and variables does not alter the optimum solution(s), but may consume extra computational effort. Many researchers have proposed algorithms for identifying the redundant constraints and variables in LP models [5,16,18,19,21,24,25,31,32,33]. Paulraj[27] used the intercept matrix of the constraints to identify redundant constraints prior to the start of the solution process in his heuristic approach. 01MKP is a well known 0-1 integer programming problem and many variables have zero values called redundant variables. We use the intercept matrix of the constraints (2) to identify the variables of value 1 and 0.

Surrogate constraints were first introduced by Glover[14,15] to provide choice rule evaluations and bonds for integer programming problems in [14] and to transform infeasible solutions into feasible solutions in the context of an evolutionary procedure in [15]. The surrogate constraint can be defined as:

$$\sum_{j=1}^{n} (\sum_{i=1}^{m} w_i a_{ij})x_j \leq \sum_{i=1}^{n} w_i b_i \tag{5}$$

where $w = w_1, w_2, ..., w_m$ is a set of surrogate multipliers(or weights) of some positive real numbers. We use this constraint as an additional constraint to the given problem. Pseudo-utility ratio can be defined for each variable, based on the surrogate constraint (5), is

$$u_j = c_j / \sum_{i=1}^{n} w_i a_{ij} \tag{6}$$

We use this ratio(6) to improve the solution obtained from the dominance principle approach. First, we add an additional constraint to the given problem by using surrogate constraint technique.Second, we construct the intercept matrix by dividing b_k values by coefficients of (2).The elements of intercept matrix are arranged in decreasing order, the leading element is the dominant variable. This process of identifying the leading element from intercept matrix is known as dominant principle. We use this dominant variable to improve the current feasible solution and this procedure provides optimum or near optimum solution of 01MKP. The dominant principle focuses at the resource matrix with lower requirement come forward to maximize the profit. The intercept matrix of the constraints (2) plays a vital role in achieving the goal, in a heuristic manner.Next we use the pseudo-utility ratio for each variable to improve the solution quality.

The algorithm is as follows. Step-(a) initialize the solution vector with zero value for all the unknowns. Next we constrcut surrogate constarint and pseudo ratio operator(step-(b) and step-(c)). In step-(d) we construct the intercept matrix and identify redundant variables through step-(e) and (f). The Values corresponding to column minimum (e_{ij}) are multiplied with corresponding cost coefficients (c_j) and the maximum among this product is chosen $ie., \underbrace{max}_{k}\ e_{ik}c_k$.

The corresponding x_k assumes the value 1 . Next we update the availability (right hand side column vector) by using the relations $b_i = b_i - a_{ik}$ for all i and the coefficients of constraints by replacing a_{ik} by 0 for all i . This process is repeated till coefficient matrix becomes null matrix. The values of updated variables x_k are substituted in the objective function to obtain the value. This process is repeated n times (step-(d) to step-(i) in DPHEU algorithm)

Pseudo-utility ratio is used to improve the objective function value. The process by examining each variable in decreasing orders of u_j and changes the variable from one to zero and zero to one as long as feasibility is not violated. We present below, the heuristic algorithm for solving 01MKP using the dominance principle approach.

(a) Initialize the solution by assigning 0 to all x_j.

(b) Surrogate constraint $\sum_{j=1}^{n}(\sum_{i=1}^{m}\ w_i a_{ij}\ )x_j \leq \sum_{i=1}^{n}\ w_i b_i$ (m+1)th constraint.

(c) Pseudo utility ratio operator $u_j = c_j / \ sum_{i=1}^{n}\ w_i a_{ij}$.

(d) Intercept matrix D $d_{jj} = b_i/a_{ij}$, if $a_{ij} > 0$,

d_{jj} = M, a large value ; otherwise.

(e) Identify 0 value variables(redundant): If any column has <1 entry in D, then the corresponding variable identified as a redundant variable.

(f) Dominant variable: Identify the smallest element(dominant variable) in each column of D.

(g) Multiply the smallest elements with the corresponding cost coefficients. If the product is Maximum in kth column, then set x_k = 1 and update the objective function value $f(x_1, x_2, ..., x_n)$.

(h) Update the constraint matrix: $b_i = b_i - a_{ij}$ for all i and set a_{ik} = 0 fo all i.

(i) If a_{ij} = 0 for all i and j, then go to step-(j) . Otherwise go to step-(d)

(j) Pseudo utility operator(to improve the current solution) Let R_i = the accumulated resources of constraint i in the solution set. For each j*(1 to n), identify j such that x_{j*} = 1 and satisfies both $c_j > c_{j*}$, $u_j > u_{j*}$, and R_i- a_{ij} + $a_{ij*} \leq b_i$, i = 1 to m. If such an j can be found, then set x_j=1 and x_{j*}=0.

Theorem 1. DPHEU can be solved in $O(mn^2)$ time, polynomial in the number of item types and constraints.

Proof. The Worst -case complexity of finding the solutions of an 01MKP using DPHEU can be obtained as follows. Assume that there are n variables and m+1 constraints. The procedure initialization (step-(a)), requires O(n) running time. Construction of surogate constraint(step-(b)) and pseudo ration operator(step-(c)) requires O(mn) and O(n) respectively. The Formation of D matrix iterates n times, identification of less than one entry in each column , finding smallest intercept in each column ,identification rows which consists of more than one smallest intercept and updating of constraint matrix A . Since there are m constraints, step-(d), step-(e), step-(f), step-(i) respectively, O((m+1)n)time. step-(g) and step-(h) requires O(n) operations to multiply cost with corresponding smallest intercept and updating the corresponding row of the constraint matrix. The maximum number of iterations required for DPHEU is n. So the overall running time of the procedure DPHEU can be deduced as O (mn^2)

An example of 01MKP solution by DPHEU Consider Max 5000x1 +5550x2+1000x3+1500x4

Subject to

5x1 + 7x2 + 2x3 $\leq$ 9

3x1 + 2x2 $\leq$ 3

425 x1 + 300 x2 +50 x3 + 100 x4 $\leq$ 500

In the above example of 01MKP have 4 variables and 3 constraints. We add 4th constraint (surrogate constraint) by using (4) with weights 2, 3, 4.

719x1 + 1220x2 + 204x3 + 400x4 $\leq$ 2027. The algorithm begins with the initial feasible solution, thus the solution vector can be written as (0, 0, 0, 0), and objective value is 0. We update this solution by using DPHEU algorithm iteratively. This heuristic updates the solution vector and objective function value. Step - (d) to step - (i)

Iteration 1: The variable x2 dominates the other variables and improves the objective function value from 0 to 5550. Thus the new solution vector is (0, 1, 0, 0) and objective function value is 5550.

Iteration 2: The variable x4 dominates the other variables(x1 and x3) and improves the objective function value from 5500 to 7050. The solution vector is (0, 1, 0, 1).

Iteration 3: The variable x3 dominates the other variable (x1) and improves the objective function value from 7050 to 8050. The solution vector is (0, 1, 1, 1). we assign 0 to x2, the x2 variable column has $<$ 1 entry. Step -10 There is no improvement in the objective function value for this problem at this step. The final solution is x1=0, x2 = 1, x3 = 1, x4 = 1. objective function value is 8050 which is optimum.

IV. Computational results

Our DPHEU was initially tested on 55 standard problems (divided into six different sets) which are available from OR- Library (http://mscmga.ms.ic.ac.uk/jeb/orlib/mknapinfo.html)[4]. The size of these problems varies from n = 5 to 105 items and from m =2 to 50 constraints. We solved these problems-, using both the general-purpose CPLEX mixed-integer programming solver, and our DPHEU which was coded in MATLAP7. The results are shown in Table 1. The first two columns in Table 2 indicate the problem set name and the number of problems in that problem set. The next two columns report for CPLEX the average execution time, and average number of nodes required for CPLEX. The last three columns report that DPHEU algorithms average solution time, average percentage deviation from optimum solution,

and total number of optimum solution found by DPHEU for each problem set.
It is clear that from Table 1 that our DPHEU finds the optimal or near optimal in all 55 test problems. CPLEX required 1.13 seconds to solve all the problems shown in Table 1. Our DPHEU required 0.64 seconds only.
The application of HEU algorithm for Weing7 is presented in figure 1. The maximum number of iterations is 105 (n) fixed. HEU reaches the solution 1094806 at 90^{th} iterations which is best but not optimum (from step - 1 to step - 9). Next we apply pseudo-utility ratio operator to improve the solution (step 10), it takes only three iterations to improve the solution quality, 1095445, which is optimum one. The second set of tested instances is constituted of (also the largest ones with n = 100 to 2500 items m =15 to 100 constraints) 11 benchmarks (MK-GK problems) proposed very recently by Glover and Kochenberger (available at: http://hces.bus.olemiss.edu/tools.html). Table 2 compares our results and the best known results taken from the above web site.

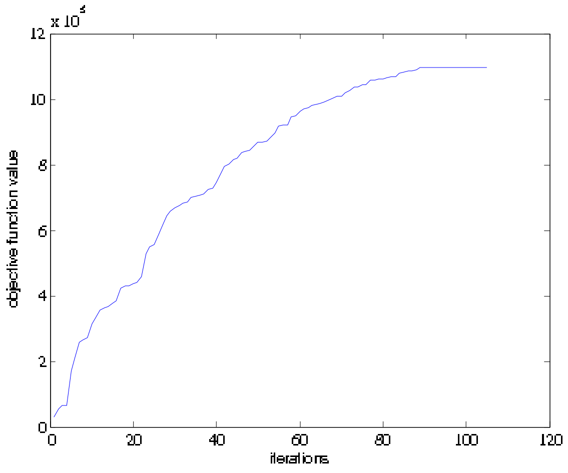

Fig. 1. Iteration wise Objective function value for Weing7

TABLE I
COMPUTATIONAL RESULTS FOR CPLEX AND THE DPHEU FOR SMALL SIZE PROBLEMS

Problem set name	Number of Problems	CPLEX	DPHEU		
		Average Solution time	Average Solution time	A.P.O.D	N.O.P.T
HP	2	0.3	0.2	0.0	2
PB	6	2.8	1.8	0.04	5
PETERSON	7	0.2	0.2	0.0	7
SENTO	2	12.0	3.8	0.0	2
WEING	8	0.6	0.4	0.0	8
WEISH	30	0.5	0.4	0.03	28
average		1.13	0.64		

A.P.O.D = Average percentage of deviation
N.O.P.T = Number of problems for which the DPHEU finds the optimal solution

The first two columns in Table 2 indicate the sizes (m and n), third column CPLEX solution of problems; the last two columns report the DPHEU solution and percentage of deviation from the best solutions of the problems. It can be seen from Table 2 that DPHEU is found to be better

TABLE II
COMPUTATIONAL RESULTS FOR CPLEX AND THE DPHEU FOR LARGE SIZE PROBLEMS

n	m	CPLEX Best Feasible Solution	DPHEU Solution	Percentage of deviation
100	15	3766	3766	0
100	25	3958	3958	0
150	25	5650	5656	0.26(improved)
150	50	5764	5767	0.05(improved)
200	25	7557	7557	0.0
200	50	7672	7672	0
500	25	19215	19215	0
500	50	18801	18806	0.03(improved)
1500	25	58085	58085	0.0
1500	50	57292	57294	0.003(improved)
2500	100	95231	95231	0.0

in 4 out of 11 problems than CPLEX. As both tables and figure clearly demonstrate, the DPHEU is able to localize the global optimum or near optimal point for all the test problems in quick time. Our approach is used to reduce the search space to find the near-optimal solutions of the 01MKP. The computational complexity is cubic and the space complexity is O(mn) . DPHEU reaches the optimum or near optimum point in less number of iterations where the maximum number of iterations is the size of projects (variables). Our heuristic algorithm identifies the zero value variables quickly.

V. CONCLUSION

In this paper, we have presented the dominance principle based approach for tackling the NP-Hard 0-1 Multi constrained knapsack problem (01MKP). This approach has been tested on 66 state-of-art benchmark instances and has led to given near optimal solutions for most of the tested instances. Our approach is heuristic with $O(mn^2)$ complexity and it requires maximum of n iterations to solve the 01MKP. The experimental data show that the optimality achieved by this heuristic lies between 98 and 100 percentage. The basic idea behind the proposed approach may be explored to tackle other NP-Hard Problem.

REFERENCES

[1] Arnaud Freville., The multidimensional 0-1 knapsack problem: An overview, European Journal of Operational Research, 155, 1-21,2004.
[2] Balas E. An additive algorithm for solving linear programs with zero-one Variables, Operations Research, 13 : 517-546,1965.
[3] E.Balas,C.H.martin: pivot and Complement - a Heuristic for 0-1 programming, Management Science 26(1),1980.
[4] J.E.Beasley: OR-Library; Distributing Test Problems by Electronic Mail, Journal of Operational Research Society 41, pp.1069-1072, 1990.
[5] Brearley,A.L., G.Mitra and H.P Williams, Analysis of mathematical programming problem prior to applying the simplex algorithm , Math. Prog., 8: 54-83. 1975.
[6] Cabot A.V.An enumeration algorithm for knapsack problems, Operations Reserch ,18:306-311,1970.
[7] P.C.Chu: A Genetic Algorithm Approach for Combinatorial Optimization Problems,Ph.D.thesis at Management School,Imperial College of Science , London, 1997.
[8] P.C.Chu and J.E.Beasley.A genetic algorithm for the multidimensional knapsack problem, Journal of Heurisic,4:63-86,1998.
[9] A.Drexl. A simulated annealing approach to the multiconstraint zero-one knapsack problem,Computing,40:1-8,1988.
[10] M.R.Garey and D.S.Johnson,Computers and Intractability: A Guide to the theory of NP-completeness,W.H.Freeman and company, San Francisco.1979.

[11] Gavish B. and pirkul H.Efficient algorithms for solving multiconstraint zero- one problems to optimality, Mathematical programming,31:78-105,1985.
[12] Gilmore P.C and Gomory R.E ,The theory and computations of Knapsack Functions, Operations Research,14:1045-1075,1966.
[13] F.Glover and G.A.Kochenberger.Critical event tabu search for multidimensional knapsack problems.Metaheuristics: The Theory and Applications, pages 407-427.KluwerAcademic Publishers,1996.
[14] Glover. F, A Multiphase-Dual Algorithm for the Zero-One Integer Programming Problem, Operations Research 13,879-919. 1965.
[15] Glover. F, Heuristics for Integer Programming Using Surrogate Constraints, Decision Sciences 8, 156-166,1977
[16] Gowdzio, J.,Presolve analysis of linear program prior to applying an interior point method,Inform. J.Comput., 9: 73-91. 1997
[17] S.Hanafi and A.Freville.An efficient tabu search approach for the 0-1 multidimensional knapsack problem, European Journal of Operational Research,106:659-675,1998.
[18] Ioslovich, I., Robust reduction of a class of large scale linear program, Siam J.Optimization, 12: 262-282,2002.
[19] Karwan,M.H., V. Loffi, J. Telgan and S. Zionts, Redundancy in mathematical Programming, A State of the Art Servey (Berlin: Springer-Verlag). 1983,
[20] S. Khuri, T. Baeck, J. Heitkoetter, The Zero/One Multiple Knapsack Problem and Genetic Algorithms, Proceedings of the 1994 ACM Symposium on Applied Computing, SAC'94, Phoenix, Arizona. pp188-193, ACM press,1994
[21] Kuhn, H.W. and R.E . Quant, An Experimental Study of the Simplex Method, In: Metropolis, N. et al.(Eds.). Preceedings of Symposia in Applied Mathematics. Providence, RI: Am. Math. Soc., 15: 107-124. 1962.
[22] M.J.Magazine,O.Oguz , A Heuristic Algorithm for the Multidimensional zero-one knapsack problem,European Journal of Operational Research 16.pp.319-326,1984.
[23] S.Martello and P.Toth. Knapsack problems: Algorithms and Computer Implementations, John Wiley and Sons, chichester, West Sussex, England,1990.
[24] Matthesiss,T.H., , An Algorithm for determining irrelevant constraints and all vertices in systems of linear inequalities, Operat. Res., 21: 247-260. 1973
[25] Meszaros. C. and U.H. Suhl, Advanced preprocessing techniques for linear and quadratic programming, Spectrum, 25: 575-595. 2003.
[26] Osrio M.A.Glover F.and HammerP. Cutting and surrogate constraint analysis for improved multidimensional knapsack solutions, Technical Report HCES-08-00,Hearing Center for Enterprise Science,2000.
[27] Paulraj, S., C. Chellappan and T.R. Natesan, A heuristic approach for identification of redundant constraints in linear programming models, Int. J. Com. Math., 83(8): 675-683. 2006,
[28] H.Pirkul: A Heuristic Solution procedure for the Multiconstrained zero-one Knapsack problem,Naval Research Logistics 34,pp.161-172,1987.
[29] Sartaj Sahni, Approximate Algorithms for the 01 knapsack problem, ACM Vol 29,no 1 , pp 113-124,1975
[30] Shih W. A. branch and bound method for the multiconstraint zero-one knapsack problem, Journal of Operational Research Society,30: 369-378,1979.
[31] Srojkovic, N.V. and P.S. Stanimirovic, Two direct methods in linear Programming, European J. Oper. Res., 131: 417-439. 2001.
[32] Telgan, J.,Identifying redundant constraints and implicit equalities in system oflinear constraints, Manage. Sci., 29: 1209-1222,1983.
[33] Tomlin, J.A. and J.S Wetch, Finding duplicate rows in a linear programming Model, Oper. Res. Let., 5: 7-11. 1986.
[34] A.Volgenant,J.A.Zoon : An Improved Heuristic for Multidimensional 0-1 Knapsack Problems, Journal of the Operational Research Society 41,pp.963-970,1990.
[35] Weingartner H.M and Ness D.N,Methods for the solution of the multidimensional 0/1 knapsack problem, Operations Research,15:83-103,1967.