Stochastic Local Search combined with Simulated Annealing for the 0-1 Multidimensional Knapsack Problem.

Abdellah Rezoug
Department of Informatics
Faculty of Science
University M'hamed Bougara of Boumerdes
Boumerdes, Algeria
Email: abdellah.rezoug@gmail.com

Dalila Boughaci
Department of Informatics
Faculty of Electronics and Informatics
University Houari Boumedienne
Algiers, Algeria
Email: dalila_info@yahoo.fr

Amar Rezoug
Development Center
for Advanced Technologies
Algiers, Algeria
Email: amarrezoug@gmail.com

***Abstract*—The 0-1 Multidimensional Knapsack Problem (MKP) is a widely-studied problem in combinatorial optimization domaine which has been proven as NP-hard. Various approximate heuristics have been developed and applied effectively to this problem, such as local search and evolutionary methods. This paper proposes the Stochastic Local Search-Simulated Annealing (SLSA) approach that combines the stochastic local search (SLS) and the simulated annealing (SA) to solve the MKP. Three main techniques are introduced in SLSA which are: the neighborhood creation, the solution reparation and the mutation strategy. We validate the effectiveness of the proposed approach through an experimental study performed on several benchmark problems commonly used in the literature. The obtained results show that the SLS and SA, when combined appropriately can provide better results than either SLS or SA alone.**

I. Introduction

The Multidimentional Knapsack Problem (MKP) is a NP-hard combinatorial optimization problem [1] which has been widely studied in the literature [2]. The MKP has been extensively discussed because of its theoretical importance and wide range of applications. Many practical engineering design problems can be formulated as MKP, such as: the capital budgeting problem [3], the project selection [4], the cargo loading problems [5] and so on. MKP is similar to the winner determination problem in multi-unit combinatorial auctions (WDP-MUCA) [6].

Since its beginning, several approaches have been proposed to solve the MKP. The existing approaches can be classified into exact and non-exact. Among the exact approaches, we cite: Branch and Bound [7], Dynamic Programming [8], Hybrid Constraint and Integer Linear Programming [9]. These approaches have the advantage of efficiency to solve MKP problems of small size and provide exact results. However, the execution time increases in exponential manner with the size of the studied problem. In this fact, the non-exact appoaches have been preferred. Indeed, several non-exact approaches have been proposed to the MKP, all of them are based on heuristic methods. These approaches provide results close to exact results within a reasonable time. The most popular local search heuristics have been used to solve the MKP such as: Tabu Search [10], Simulated Annealing (SA) [11] and Variable Neighborhood Decomposition [12]. Also Evolutionary and hybrid heuristics have widely been used such as: Genetic Algorithm [13], [14], Neural and Neurogenetic [15], Ant Colony Optimization [16], Harmony Search [17], [18], Evolutionary Algorithm [19], Particle Swarm Optimization [20], [21], Lagrangian Relaxation [22], Cuckoo Search [23], Artificial Fish Swarm Algorithm [24] and Core base heuristics [25], etc.

It is known that the local search heuristics converge quickly but in a local optimum. The Simulated Annealing (SA) [26] is one of the famous local search methods because of its simplicity and effectiveness. Similarly the stochastic local search (SLS) is an efficient local search method. SA has the capacity to find zones not yet visited thanks to its neighborhood creation strategy. But the searching process in SA can visit the same solution several times. Comparatively Stochastic Local Search (SLS)[27] is effective in term of diversification capacity due to its neighborhood creation strategy. But this strategy based on the random and the hill-climbing require a lot of time to get good results. In this paper, we propose the Stochastic Local search-Simulated Annealing algorithm (SLSA) to solve MKP. SLSA is a hybridization between SLS and SA. Three main modifications are proposed. These three modifications are used in the proposed SLSA. Furthermore,the first modification is used to perform SA and SLS. The first modification concerns the reparation technique of SLSA, SA and SLS. The reparation operation based on iterative removing of the worst item is modified by adding the remove of an item chosen randomly according to a probability P. The second and the third modification concerns the neighborhood creation strategies of SLSA. Here, it is proposed the combination of SA and SLS strategies. Also the hill-climbing of SLS is replaced in SLSA by a mutation technique. The proposed SLSA approach is compared with SA and SLS methods for solving MKP using the available test data sets in the OR-Library [28].

The rest of the paper is organized as follows: a brief background about the MKP is presented in section 2. The SA and SLS methods are explained in section 3 and 4 respectively. The proposed SLSA algorithm for solving MKP is detailed in Section 5. Section 6 describes the simulation and evaluation results of the proposed algorithm on the test data sets. Finally,

Symposium on Complex Systems and Intelligent Computing (CompSIC)
University of Souk Ahras - Université Mohamed Chérif Messaadia de Souk-Ahras
http://www.univ-soukahras.dz/en/publication/article/416

we draw the conclusions of this study in Section 7.

II. The Multidimensional Knapsack Problem (MKP)

The MKP problem is composed of N items and a knapsack with m different capacities b_i where $i \in M = \{1, \ldots, m\}$. Each item j where $j \in N = \{1, \ldots, n\}$ has a profit c_j and can occupy a_{ij} of the capacity i of the knapsack. The goal is to pack the items in the knapsack so as to maximize the profits of items without exceeding the capacities of the knapsack. The MKP is modeled as the following integer program:

$$\textbf{Maximize} \ \sum_{j=1}^{n} c_j x_j \quad x_j \in \{0,1\} \tag{1}$$

$$\textbf{Subject to}: \sum_{j=1}^{n} a_{ij} x_j \leq b_i \qquad i \in M = \{1 \ldots m\} \tag{2}$$

$$x_j \in \{0,1\} \qquad j \in \{1 \ldots n\} \tag{3}$$

The feasible solution for the MKP is represented by X, where X is a vector of size n. X contains the selected items to be packed in the knapsack. Decision variables x_j are binary where $x_j = 1$ means that the item j is packed in the knapsack, and $x_j = 0$ means that it is not packed. a_{ij} represents the space in the dimension i occupied by the item j.

III. The Simulated Annealing (SA)

The simulated annealing (SA) is an old heuristic method largely used because of its facility and efficiency. This method has been presented for the first time in [26]. In this work, we applied the SA for the MKP. SA can be decomposed into two steps. First step consists in preparing the data by generating an initial feasible solution X, which we do using the Random Key method (RK) [29]. Second step is the optimization of the initial solution. This operation is based on the temperature parameter T fixed initialy as T_0.

The optimization includes four operations given as follows:

1) Creating a neighbor solution X' of X. For that, one item I is chosen arbitrary. If I increases the objective function $f(X)$ then it will be accepted i.e. added to the solution X', otherwise it will be added if the comparison $\exp^{-(\Delta f()/T)} > R$ is true. Where R is a random value and $\Delta f() = f(X') - f(X)$.
2) **The first modification:** The first step may cause a conflict. In order to eliminate all conflicts, and make X as a feasible solution, it is repaired by removing an item. The item to be removed is chosen in two manners according to the probability $P(P = 0.7)$. Either an item is randomly chosen or the worst one in X is found and removed. This process is repeated until the elimination of all conflicts.
3) During the process, among the neighbors, every solution that increases the objective function is considered as the best solution and saved in X^*. By the end of the process X^* contains the best solution.
4) The last operation in this process consists in updating the temperature value using the Coefficient of Temperature CT. In our case the updating rule is found empirically as $T = T - CT(CT = 0.0105)$.

The optimization process is repeated for a certain Number of Iterations (NI) fixed empirically.

IV. The Stochastic Local Search (SLS)

The stochastic local search (SLS) is a local search iterative heuristic [27]. It starts with an initial solution X generated randomly according to the RK encoding. Then, it performs a certain number of local steps as follows:

1) Creating neighbor solution X' to X. Selecting an item I to be added in the current solution X. At each step, the item to be accepted is selected according to one of the two following criteria:
   - The first criterion consists on the choose of one item in a random way with a fixed probability $wp > 0$.
   - The second criterion consists on the choose of the best item to be accepted.
2) **The first modification:** To make X' feasible solution, it is repaired by removing an item repeatedly. The item to be deleted is chosen in two manners according to the probability P. either an item is randomly chosen or the worst one in X is found and removed.
3) Saving in X^* the best neighbor feasible solution found so far.

The process is repeated for a certain Number of Iterations (NI) that was determined empirically.

V. The proposed approach (SLSA)

In this section, we propose the combination of the stochastic local search and the simulated annealing to produce a new approach called SLSA. In the following, we explain the main component of the proposed approach SLSA for MKP. The SLSA process is based on two steps as follows:

A. *Creating the initial solution by the Random Key method*

The SLSA begins by the creation of the initial feasible solution. For that, the Random Key method[29] is used. n values in [0, 1] are generated randomly and arranged in ascending order, such that each item is one of the generated values. Secondly, the solution is built by adding items one after one, according to the order, as long as all constraints are satisfied. If the addition of an item leads that at least a constraint is broken then it is ignored. This operation continues until the last item. Thirdly, the objective function of the solution is calculated.
The creation of feasible solution by the random key is the first step in SLSA. It is followed by the optimization step.

B. Optimization by SLSA

Here the initial feasible solution X' is iteratively modified. The SLSA process performs a certain number of local steps that consists in the Creation of a neighbor solution X, The Reparation of the created neighbor solution, the Record of the best solution and the Temperature update.

Step1. Creating a neighbor solution X' of X. At each operation and with a probability $wp \in [0, 1]$, the item accepted to be packed is selected according to one of the two following criteria:

1) **The second modification.** ***SA strategy:*** Here one item I is chosen arbitrary. If I increases the objective function $f(X)$ then it will be packed to the knapsack, otherwise it will be accepted if the exponential $\exp^{(\Delta f()/T)} > r$ where r is a random value, $\Delta f() = f(X') - f(X)$ and T is a temperature value initially equal to T_0 relatively high.
2) **The third modification.** ***Mutating an item:*** replacing an item in X' by another not in X'. The replaced and the replacement items are chosen randomly.

Step2. The first modification. The first step may cause a conflict. To eliminate all conflicts, and make X' feasible solution, it is repaired by removing an item. The item to be deleted is chosen in two manners according to the probability P. either an item is randomly chosen or the worst one in X' is found and removed.

Step3. If the created neighbor solution performs the objective function value $(f(X') > f(X^*))$ then it is recorded as the best solution found so far.

Step4. After that the temperature value is updated. In our case the decreasing rule is found empirically us $T = T - 0.0105$.

The process is repeated for a certain Number of Iterations NI, which was determined empirically. The SLSA algorithm is sketched in Algorithm 1.

Algorithm 1 The SLSA pseudo-code.

Require: a feasible solution X, NI, wp, T_0
Ensure: a better feasible solution X^*
for $Cpt = 1$ to NI **do**
  if $(r < wp)$ **then**
    $I_1 = RandItem(); I_1 \in X$
    if $(f(X') + C_{I_1} > f(X))$ **then**
      $X' = X' \cup \{I_1\}$
    else
      if $(r_1 < \exp^{(\Delta f/T)})$ **then**
        $X' = X' \cup \{I_1\}$
      end if
    end if
  else
    $I_2 = RandItem(); I_2 \in X$
    $I_3 = RandItem(); I_3 \ni X$
    $X' = X' - \{I_2\}$
    $X' = X' \cup \{I_3\}$
  end if
  while (ExistConflict (X')) **do**
    if $(r_2 < P$) **then**
      $I_{min} = WorstItem(); I_{min} \in X$
      $X' = X' - \{I_{min}\}$
    else
      $I_4 = RandItem(); I_4 \in X$
      $X' = X' - \{I_4\}$
    end if
  end while
  if $(f(X') > f(X^*))$ **then**
    $X^* = X'$
  end if
  $T = T - CT$
end for
Return the best solution X^*.
Where $r, r_1, r_2 \in [0, 1]$. T is the temperature. $\Delta f = f(X^*) - f(X')$ and CT: temperature update coefficient.

TABLE I. THE PARAMETERS OF ALGORITHMS.

Parameter	Value
Number of iteration of SLSA	100000
Probability of SLSA wp	0.98
Initial temperature	50
Coefficient T update	0.0105
Number RUN	30

VI. SIMULATION RESULTS

The algorithms SA, SLS and SLSA are coded using C++ and compiled under a PC having 2 GHz Intel Core 2 Duo processor and 2 GB RAM. In order to evaluate the efficiency and performance of the proposed SLSA, it was tested on 54 standard test problems (divided into six different sets) which are available at the OR-Library[28] maintained by Beasley. These datasets are real-world problems widely used to test and validate the algorithms effectiveness in the optimization community. These problems dimensions vary as $m = 2$ to 30 and $n = 6$ to 105. After several experiments, we set the parameters for the SLSA as in table I.

Table II contains the description of the used data. Here N and M represent the number of items and the number of constraints (the number of dimensions) respectively. Column Ins represents the number of instance in each group of data. Table III shows the obtained results by the application of SA, SLS and SLSA algorithms on the 54 instances. Column $A.V.F$ means the average fitness of all the 30 runs. Column $D.F.O$ represents the deviation from the optimal.

From these results we have identified various observations. We observed that the $A.V.F$ obtained by SLSA is better than SA and SLS in all instances of the groups: pet, sento and weish, furthermore in 43 of the 54 instances $A.V.F$ SLSA is better. We see that instances pet1, pet2, weing2 and weing3 are the less complex ones; it is the raison why the $A.V.F$ obtained by SA, SLS and SLSA is equal to the optimal. But only SLSA reaches the optimal solution in all the 30 runs in the two instances weish1 and weish4. In all groups the average $D.T.O$ of SLSA surpasses that of SA which surpasses that of SLS. Also we found that the average $D.T.O$ of SLSA is better than SA with the advance percentage of 1.2% and than SLS with the percentage of 1.31%. In the same time the average $D.T.O$

TABLE III. SLSA VS. SA AND SLS

	SA		SLS		SLSA	
Group	A.V.F	D.F.O	A.V.F	D.F.O	A.V.F	D.F.O
	3347,97	97,95	3345,7	97,88	3345,67	97,88
hp	2998,07	94,10	2993,93	93,97	**3011,27**	94,52
	3018,67	97,69	**3027,9**	97,99	3027,07	97,96
	3034,2	95,24	3034,27	95,24	**3031,87**	95,16
	87063,5	91,48	86672,3	91,07	**90793,7**	95,40
pb	2095,8	97,98	**2101,97**	98,27	2098,6	98,11
	679,2	87,53	679,233	87,53	**727,033**	93,69
	935,5	90,39	928,333	89,69	**999,333**	96,55
	87061	100	87061	100	87061	100
	4015	100	4015	100	4015	100
	6120	100	6120	100	6120	100
pet	12206,7	98,44	12200	98,39	**12285,3**	99,08
	10384,4	97,80	10373,5	97,70	**10394,6**	97,90
	15925,3	96,30	15938,2	96,38	**15977,7**	96,62
sento	7675	98,75	7675	98,75	**7690,57**	98,95
	8580,8	98,38	8587,43	98,46	**8670,93**	99,41
	138453	98,00	138871	98,30	**141267**	99,99
	130883	100	130883	100	130883	100
	95677	100	95677	100	95677	100
weing	115709	96,96	114896	96,28	**118598**	99,38
	96936,8	98,12	96897,5	98,08	**98693,5**	99,90
	130610	99,99	130610	99,99	130610	99,99
	1087448	99,27	1086462	99,18	1086790	99,21
	583048	93,39	575396	92,16	576703	92,37
	4491,33	98,62	4505,2	98,92	**4554**	100
	4531,17	99,89	4531,5	99,90	**4535,17**	99,98
	3993,2	97,04	4002,67	97,27	**4090,77**	99,41
	4512,47	98,93	4516,17	99,01	**4561**	100
	4384,33	97,12	4391,97	97,29	**4512,2**	99,96
	5327,13	95,86	5304,6	95,45	**5465,33**	98,35
	5326,1	95,67	5312,03	95,42	**5470,27**	98,26
	5326,07	95,02	5318,03	94,88	**5483,03**	97,82
	5218,8	99,48	**5221,07**	99,52	5218,8	99,48
	6166,23	97,27	6166,33	97,27	**6224,13**	98,18
	5059,87	89,66	4978,87	88,23	**5363,67**	95,04
	6227,13	98,23	6217,57	98,08	6211,7	97,99
	5902,17	95,83	5903,8	95,85	**5977**	97,04
	6769	97,33	6765,37	97,28	6743,57	96,97
weish	7199,77	96,17	7208,23	96,28	**7336,9**	98,00
	7053,73	96,773	7051,53	96,74	**7124,87**	97,74
	8503,43	98,49	8504,53	98,51	**8507,53**	98,54
	9249,5	96,55	9245,53	96,50	**9291,2**	96,98
	6952,73	90,31	6921,7	89,91	**7207,03**	93,62
	9121,4	96,52	9155,83	96,88	**9282,5**	98,22
	8838,43	97,40	8842,53	97,44	**8860,93**	97,65
	8246,73	92,17	8178,83	91,41	**8417**	94,07
	7635,63	91,51	7611,1	91,21	**7722,83**	92,55
	9730,67	95,21	9729,57	95,20	**9777,17**	95,66
	9589,67	96,48	9595,13	96,54	**9710,37**	97,69
	8733,7	91,12	8715,93	90,94	**8944**	93,32
	8888,67	90,52	8873,4	90,36	**9145,17**	93,13
	8996,63	94,78	8958,67	94,38	8923	94,00
	8866,93	94,22	8847,47	94,02	8810,03	93,62
	10681,3	95,44	10676,9	95,40	**10830,9**	96,78
Total Average	52693,1	96,24	52512,9	96,13	**52829,1**	**97,44**

of SA is lightly better than SLS with the advance percentage of 0.11%. Fig. 1 shows clearly the difference.

In this study, three major modifications have been made to the algorithms. One of them concerned all the algorithms when two others characterized only the SLSA. These modifications gave to the algorithms more effectiveness. Firstly, repairing the solution by removing an item chosen randomly with probability P performed the obtained results by SA, SLS and SLSA. Secondly, introducing the SA neighborhood creation mechanism in SLSA allowed increasing the quality of the obtained results. Finally, the mutation represents the change that has the important impact on the SLSA conduct. This mechanism prevents the search process to reproduce solutions already visited. It provides effective means to conduct the search process in zones not yet discovered. Thanks to this

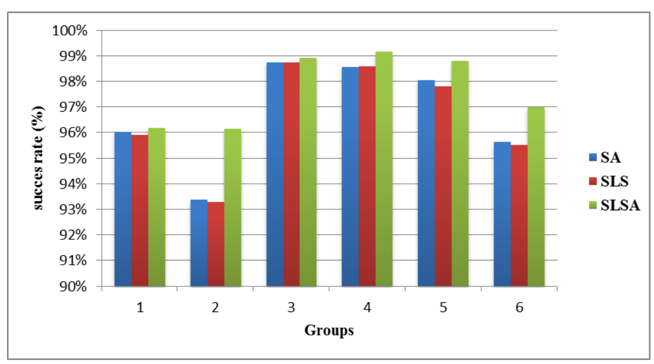

Fig. 1. SLSA vs. SA and SLS.

TABLE II. BENCHMARKS DESCRIPTION

Dataset	Number of instances	Dimensions (N, M)
hp	2	(28,4), (35,4)
pb	6	(27,4), (34,4), (29,2), (20,10), (40,30), (37,30)
pet	6	(10,10), (15,10), (20,10), (28,10),(39,5), (50,5)
wento	2	(60,30), (60,30)
weing	8	(28,2), (28,2), (28,2), (28,2), (28,2), (28,2), (105,2), (105,2)
weish	30	(30,5), (30,5), (30,5), (30,5), (30,5), (40,5), (40,5), (40,5), (40,5), (50,5), (50,5), (50,5), (50,5), (60,5), (60,5), (60,5), (60,5), (70,5), (70, 5),(70,5), (70,5), (80,5), (80,5), (80,5), (80,5), (90,5), (90,5), (90,5), (90,5), (90,5)

mechanism and the SA neighborhood strategy SLSA avoids stagnation in the local optima and succeed to find solution very close to the optimal.

VII. CONCLUSION

This paper aims to propose a local search solution to the 0-1 multidimensional knapsack problem (MKP). The suggested solution is the combination of the Stochastic Local Search method (SLS) with the simulated annealing method (SA). The proposed approach is called the Stochastic Local Search-Simulated Annealing (SLSA). In SLSA, three main techniques were proposed: the neighborhood creation, the solution reparation and the mutation strategy. In order to show the effectiveness of SLSA, several tests were carried out on a large range of benchmarks known by their complexity. Also the algorithm was compared with SA and SLS algorithms. In conclusion, the use of the three techniques allowed SLSA to obtain good results and surpass significantly SA and SLS with the percentage of 1.2% and 1.31% respectively. Similarly the SLSA succeed to reach or at least be close to the optimal, indeed the overall success rate is of 97.44% .

REFERENCES

[1] Garey M.R., and Johnson D.S., (1979) 'Computers and intractability: A guide to the theory of NP-completeness'. New York: W. H. Freeman & Co.

[2] Puchinger J., Raidl G., Pferschy U., (2007) 'The multidimensional knapsack problem, Structure and algorithms', Technical Report 006149, National ICT Australia, Melbourne, Australia.

[3] Meier H., Christofides N., Salkin G., (2001)'Capital budgeting under uncertainty - an integrated approach using contingent claims analysis and integer programming', O*perations Research* 49, pp.196-206.

[4] Beaujon G.J., Martin S.P., McDonald C.C., (2001) 'Balancing and optimizing a portfolio of R&D projects', *Naval Research Logistics 48* pp.18-40.

[5] Shih W., (1979) 'A branch and bound method for the multi constraint zero-one knapsack problems', *Journal of the Operations Research Society* 30, pp.369-378.

[6] Kelly Terence. (2005) 'Generalized Knapsack Solvers for Multi-unit Combinatorial Auctions: Analysis and Application to Computational Resource Allocation', *P. Faratin and J.A. Rodrguez-Aguilar (Eds.): AMEC* 2004, LNAI 3435 pp.73-86,

[7] Fukunaga, A.S. (2011)'A branch-and-bound algorithm for hard multiple knapsack problems', *Annals of Operations Research* , Vol.184, pp.97-119.

[8] Volgenant, A. and Zoon, J. (1990) 'An Improved Heuristic for Multidimensional 0-1 Knapsack Problems', *Journal of Operational Research Society*, Vol.41, pp.963-970.

[9] Oliva, C. Michelon, P. Artigues C. (2001) 'Constraint and linear programming: Using reduced costs for solving the zero/one multiple knapsack problem'. *Paper presented at the it 1st International Conference on Constraint Programming, Proceedings of the Workshop on Cooperative Solvers in Constraint Programming, CoSolv 01*. 2001. pp.87-98.

[10] Vasquez M., Vimont Y., (2005) 'Improved results on the 0-1 multidimensional knapsack problem', *Eur. J. Oper. Res*. 165 pp.70-81.

[11] Cho, J.H. and Kim, Y.D. (1997) 'A simulated annealing algorithm for resource-constrained project scheduling problems', *Journal of the Operational Research Society*, Vol.48, pp.736-744.

[12] Hanafi S., Lazic J., Mladenovic N., Wilbaut C. (2009). 'Variable Neighborhood Decomposition Search with Bounding for Multidimensional Knapsack Problem'. *Information Control Problems in Manufacturing*, 13 - Partie 1, V.A. Trapeznikov Institute of Control Sciences (Russie), pp. 2018 - 2022.

[13] Chu, P. and Beasley, J. (1998) 'A Genetic Algorithm for the Multidimensional Knapsack Problem', *Journal of Heuristics*, Vol.4, pp.63-86.

[14] Farhad Djannaty and Saber Doostdar, (2008) 'A Hybrid Genetic Algorithm for the Multidimensional Knapsack Problem'. *Int. J. Contemp. Math. Sciences*, Vol. 3, no. 9, pp.443-456

[15] Jason Deane and Anurag Agarwal,(2013) 'Neural, Genetic, And Neurogenetic Approaches For Solving The 0-1 Multidimensional Knapsack Problem', *International Journal of Management & Information Systems* - First Quarter 2013 Volume 17, Number 1.

[16] Feng L. Ke, Z., Ren Z., Wei X., (2010) 'An ant colony optimization approach for the multidimensional knapsack problem', *Journal of Heuristics* 16, pp.65-83.

[17] Shouheng Tuo, Longquan Yong, and Fang'an Deng,(2014) 'A Novel Harmony Search Algorithm Based on Teaching-Learning Strategies for 0-1 Knapsack Problems', *The Scientific World Journal* Article ID 637412, 19 pages.

[18] Zou,D. Gao, L. Li, S. and Wu, J. (2011) 'Solving 01 knapsack problem by a novel global harmony search algorithm', *Applied Soft Computing*, Vol.11, pp.1556-1564.

[19] Liu, Y. Liu, C. (2009) 'A schema-guiding evolutionary algorithm for 01 knapsack problem'. *Paper presented at the International Association of Computer Science and Information Technology Spring Conference*. 2009. pp.160-164.

[20] Ktari Raida and Chabchoub Habib,(2013) 'Essential Particle Swarm Optimization queen with Tabu Search for MKP resolution', *Computing* 95, pp.897-921.

[21] Mingchang Chih, Chin-Jung Lin, Maw-Sheng Chern, Tsung-Yin Ou, (2014) 'Particle swarm optimization with time-varying acceleration coefficients for the multidimensional knapsack problem', *Applied Mathematical Modelling* 38, pp.1338-1350.

[22] Yourim Yoon and Yong-Hyuk Kim, (2013) 'A Memetic Lagrangian Heuristic for the 0-1 Multidimensional Knapsack Problem', *Discrete Dynamics in Nature and Society*, Article ID 474852, 10 pages.

[23] Yanhong Feng, Ke Jia, and Yichao He, (2014) 'An Improved Hybrid Encoding Cuckoo Search Algorithm for 0-1 Knapsack Problems', *Computational Intelligence and Neuroscience*, Article ID 970456, 9 pages.

[24] Abul Kalam Azad, Ana Maria A.C. Rocha, Edite M.G.P. Fernandes, (2014) 'Improved Binary Artificial Fish Swarm Algorithm for the 0-1 Multidimensional Knapsack Problems', *Swarm and Evolutionary Computation 14* pp.66-75.

[25] Della Croce F., Grosso A., (2012) 'Improved core problem based heuristics for the 0-1 multidimensional knapsack problem', *Computers & Operations Research* 39 pp.27-31.

[26] Kirkpatrick, S. Gelatt, C.D. Vecchi, P.M. (1983) 'Optimization By Simulated Annealing'. *Science* Vol.220, pp.671-680.

[27] Boughaci, D. Benhamou, B. Drias, H. (2010) 'Local Search Methods

for the Optimal Winner Determination Problem in Combinatorial Auctions', *Journal of Math Model Algor*, Vol.9, No.1, pp.165-180.

[28] http://www.brunel.ac.uk/ mastjjb/jeb/info.html

[29] Bean, J.C. (1994) 'Genetics and random keys for sequencing and optimization', *in ORSA Journal of Computing*, Vol.6, No.2, pp.154-160.