Available online at www.sciencedirect.com

ScienceDirect

Procedia Computer Science 111 (2017) 71–76

Procedia Computer Science

www.elsevier.com/locate/procedia

8th International Conference on Advances in Information Technology, IAIT2016, 19-22 December 2016, Macau, China

Migrating birds optimization (MBO) algorithm to solve knapsack problem

Erkan Ulker[a]*, Vahit Tongur[b]

[a]*Department of Computer Engineering, Selcuk University, Konya 42030, Turkey*
[b]*Department of Computer Engineering, Necmettin Erbakan University, Konya 42060, Turkey*

Abstract

This study presents Migrating Birds Optimization (MBO) which is a novel meta-heuristic algorithm for the solution of knapsack problem. The knapsack problem which is classified as NP-complete problem is a combinatorial optimization problem. Its aim is to achieve maximum benefit without exceeding the capacity of the knapsack with selected item. The Migrating Birds Algorithm is designed for discrete problems. Therefore, the performance of basic the MBO algorithm is tested on the some knapsack problems and obtained results are demonstrated in detail.

Peer-review under responsibility of the organizing committee of the 8th International Conference on Advances in Information Technology

Keywords: Knapsack Problem, Meta-heuristic Optimization, Migrating Birds Optimization

1. Introduction

The knapsack problem is a combinatorial optimization problem. This problem can be described as follow: Given t objects, each object has a profit value (p) and weight (w). The aim of this problem is to place the bounded capacity (C) knapsack that maximum value from among all objects. Objective function of this problem can be expressed as follow:

* Corresponding author. Tel.: 905056429732
E-mail address: EULKER@SELCUK.EDU.TR

1877-0509
Peer-review under responsibility of the organizing committee of the 8th International Conference on Advances in Information Technology
10.1016/j.procs.2017.06.012

$$maxf(x) = \sum_{i=1}^{t} p_i x_i \quad (1)$$

$$\sum_{i=1}^{t} w_i x_i \leq C;\ x_i \in \{0,1\}, i = 1,2,\cdots,n \quad (2)$$

where, t is number of object, p_i is profit of *i th* object, w_i is weight of *i th* object and x_i demonstrate selected or not selected of *i th* object. If *i th* object is put into knapsack, $x_i = 1$, otherwise $x_i = 0$. Therefore, in this study, we are addressed 0-1 knapsack problem type. Solving of this problem can be seen as a simply. However, its computational complexity increases exponentially with problem size. Therefore, knapsack problem is an NP-complete combinatorial optimization problem and it cannot be solved with conventional methods within reasonable time.

The knapsack problem is widely used in real-life applications: loading problems [1,2], resource allocation [3,4]. Therefore, the knapsack problem has attracted the attention of researchers. There are many studies on the knapsack problem in the literature last decades. Shi proposed an improved ant colony algorithm solve to 0-1 knapsack problem [5]. Kong et al. used a new ant colony optimization algorithm for multidimensional knapsack problem [6]. Liu and Liu proposed a schema-guiding evolutionary algorithm [7]. Balev et al. presents a dynamic programming based reduction procedure for the multidimensional 0–1 knapsack problem [8]. Zou et al. proposed a novel global harmony search algorithm for 0-1 knapsack problem [9]. Zhang et al. used amoeboid organism algorithm for 0-1 knapsack problem [10]. Wang et al. proposed fruit fly optimization algorithm to solve the multidimensional knapsack problem [11]. Truong et al. used chemical reaction optimization for the 0–1 knapsack problem [12].

In this study, we are proposed Migrating Birds Optimization (MBO) algorithm that a new meta-heuristic algorithm to solve 0-1 knapsack problem. The performance of the MBO algorithm is tested on selected some 0-1 knapsack problems from literature. Obtained results showed that MBO algorithm reached to optimal solutions for 0-1 knapsack problems.

2. Migrating Birds Optimization (MBO) Algorithm

The MBO algorithm is proposed by Duman et al. [13]. It is inspired by "V" formation that during migration of migratory birds. It is designed to firstly quadratic assignment problem (QAP). The QAP is one of the discrete problems. Therefore, it is applied to combinatorial optimization problems easily.

The MBO is started with initial solutions. Afterward, these solutions are tried to improve on each step. The MBO uses a neighborhood structure to improve solutions. Also, it has a benefit mechanism that unused neighbor solutions shared with other solutions. Also, the leader bird is replaced with following bird. This process is applied to left side of the flock firstly. The leader bird is gone to end of left side and following bird is assigned as new leader. The flock is continued same formation as far as flapping parameter. Then, the leader bird is replaced with right side of the flock.

3. Implementation to Knapsack Problem of MBO Algorithm

3.1. Initial population

Each individual in the population represents a solution in search space and has a solution permutation. The permutation length is equal to number of objects and it has two different values (0 or 1). Initial population is created randomly. In the beginning, this permutation is filled with "0". Generated random value between 1 and permutation length is represented permutation array cell and this cell is filled with "1". This process is continued until not exceeding the capacity of the knapsack.

3.2. Neighborhood and sharing

In the MBO, leader bird is generated least three neighbor solutions. Generated best neighbor solution is reserved for improve current solution. Remaining solutions are shared with following birds on the left side and right side. Number of created neighbor solution and number of shared neighbor solutions are given as a parameter and these parameters are determined as follow;

$$k \in \mathbb{N}^+; k = \{3, 5, 7 \cdots\}$$

$$x \in \mathbb{N}^+; x = \{1, 2, \cdots, (k-1)/2\} \quad (3)$$

$$n = k - x$$

where, k is number of neighbor solutions of leader bird, x is number of shared neighbor solutions and n is number of neighbor solutions of remaining birds (except leader bird).

In this study, neighbourhood method is same for each bird. In this method, one of cell from permutation that filled with "0" is selected randomly and this cell is set "1". If exceeded the capacity of the knapsack, one of cell from permutation that filled with "1" is selected randomly and this cell is set "0". In this way, obtained a new neighbor solution. Generated neighbour solutions are evaluated according to objective function.

Best neighbor solution is used to improving the current solution. Remaining x neighbor solution is shared with following bird. The leader bird is shared *x/2* neighbor solutions with left side of flock and *x/2* right side. Therefore, *k* is odd number. Remaining birds that except the leader bird are shared *x* neighbor solutions with following bird.

Stop criterion is determined as fitness evolution = (problem size)^3. Problem size is number of object for the knapsack problem.

Solution of the knapsack problem with the MBO is demonstrated in the following example.

Example: Let *w* and *p* be the set of weight and profit respectively. Also, let *C* be the knapsack capacity:

w={2,5,7,3,1}, p={10,30,70,50,1}, Problem dimension = 5, C = 10

Initial population of the MBO can be shown as follow:

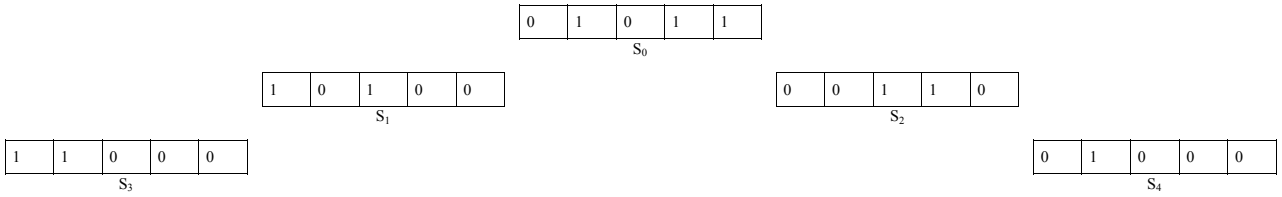

Table 1. Neighbor solutions of each bird for *k=3* and *x=1*.

Bird	Permutation	Fitness (Sum of Profit)	Neighbor	Permutation	Fitness (Sum of Profit)
			N_{01}	00110	80
S_0	01011	81	N_{02}	00101	71
			N_{03}	11001	41
			N_{11}	10101	81
S_1	10100	80	N_{12}	00110	80
			N_{13} (Shared=N_{02})	00101	71
			N_{21}	10010	60
S_2	00110	120	N_{22}	00011	51
			N_{23} (Shared=N_{03})	11001	41
			N_{31} (Shared=N_{12})	00110	80
S_3	11000	40	N_{32}	00010	50
			N_{33}	11001	41
			N_{41}	01010	80
S_4	01000	30	N_{42} (Shared=N_{22})	00011	51
			N_{43}	01001	31

S_0 is leader bird, S_1 and S_3 birds are on the left side of the flock, S_2 and S_4 birds are on the left side of the flock.

In Table 1, fitness of generated neighbor solutions are sorted from best to worst. Then, first (best) neighbor solution are used to improving own solution. Afterward, second neighbor solution is shared with following bird on the left side of flock. Third neighbor solution is shared with right side of the flock. This sharing is only for the leader bird. For others, this sharing occurs own side of the flock.

if $f(S_i) < f(N_{i1})$ then $S_i = N_{i1}$ *else* S_i is not updated.

Accordingly, new values of birds are $S_0 = 81$, $S_1 = 81$, $S_2 = 120$, $S_3 = 80$ and $S_4 = 80$.

4. Experiment Results

Performance of the MBO is tested on the ten problems that selected from literature [9]. Dimension, weights, profits and capacity of each problem are given Table 3.

MBO have been performed on a Intel(R) Core(TM) i7-4510U CPU @ 2.00GHz processor, 8 GB RAM and Linux Ubuntu 14.04 (64-bit) operating system and the MBO is coded with QT Creator 3.0.1 gcc compiler and C++ language. Parameters of MBO for this problem are given Table 2.

Table 2. Parameters of MBO for knapsack problem.

Parameter	Value
Bird (*z*)	71
Neighbor (*k*)	5
Flap (*m*)	30
Shared neighbor (*x*)	1
Stop criteria	(problem dimension)^3

Table 3. Dimension, weight, profit and capacity of ten problems.

Problem	Dimension	Weight (w), Capacity (C) and Profit (p)
P_1	10	w=(95,4,60,32,23,72,80,62,65,46),C=269,p=(55,10,47,5,4,50,8,61,85,87)
P_2	20	w = (92, 4, 43, 83, 84, 68, 92, 82, 6, 44, 32, 18, 56, 83, 25, 96, 70, 48, 14, 58), C = 878, p = (44, 46, 90, 72, 91, 40, 75, 35, 8, 54, 78, 40, 77, 15, 61, 17, 75, 29, 75, 63)
P_3	4	w=(6,5,9,7), C =20, p=(9,11,13,15)
P_4	4	w=(2,4,6,7), C =11, p=(6,10,12,13)
P_5	15	w = (56.358531, 80.874050, 47.987304, 89.596240, 74.660482, 85.894345, 51.353496, 1.498459, 36.445204, 16.589862, 44.569231, 0.466933, 37.788018, 57.118442, 60.716575), C = 375, p = (0.125126, 19.330424, 58.500931, 35.029145, 82.284005, 17.410810, 71.050142, 30.399487, 9.140294, 14.731285, 98.852504, 11.908322, 0.891140, 53.166295, 60.176397)
P_6	10	w=(30,25,20,18,17,11,5,2,1,1),C=60,p=(20,18,17,15,15,10,5,3,1,1)
P_7	7	w=(31,10,20,19,4,3,6),C=50,p=(70,20,39,37,7,5,10)
P_8	23	w = (983, 982, 981, 980, 979, 978, 488, 976, 972, 486, 486, 972, 972, 485, 485, 969, 966, 483, 964, 963, 961, 958, 959), C = 10000, p = (981, 980, 979, 978, 977, 976, 487, 974, 970, 485, 485, 970, 970, 484, 484, 976, 974, 482, 962, 961, 959, 958, 857)
P_9	5	w=(15,20,17,8,31),C=80,p=(33,24,36,37,12)
P_{10}	20	w = (84, 83, 43, 4, 44, 6, 82, 92, 25, 83, 56, 18, 58, 14, 48, 70, 96, 32, 68, 92), C = 879, p = (91, 72, 90, 46, 55, 8, 35, 75, 61, 15, 77, 40, 63, 75, 29, 75, 17, 78, 40, 44)

Each problem is run 30 times independent and obtained results are given Table 4.

As seen in Table 4, the MBO is found optimum value for all problems excluding problem P_8. Especially, it is found optimum value for problem P_3, P_4, P_6, P_7 and P_9 in all 30 runs. Also, it is found better value than optimum value in [9] for problem P6. On the other hand, running time of the MBO is too short for these problems. For each problem, the solution permutations of best results are given Table 5.

5. Conclusion

In this study, the MBO algorithm is applied to 0-1 knapsack problems that selected literature. MBO is found optimal solution at nine of the selected ten test problems. In addition, running time of MBO is quite reasonable for 0-1 knapsack problems. Consequently, MBO algorithm that is designed for quadratic assignment problems (QAP) has proved to be successful for knapsack problem. MBO algorithm can be proposed for solving the 0-1 knapsack problems.

Table 4. Obtained results from 10 test problems.

Problem	Optimum Value [9]	Best	Average	Worst	Time(*s*)
P_1	295	**295**	294.6666	293	0.002
P_2	1024	**1024**	1011.1	991	0.003
P_3	35	**35**	35	35	0.0004
P_4	23	**23**	23	23	0.0004
P_5	481.0694	**481.0694**	431.4180	399.8016	0.002
P_6	50	**52**	52	52	0.001
P_7	107	**107**	107	107	0.0008
P_8	9767	9765	9765	9765	0.013
P_9	130	**130**	130	130	0.0005
P_{10}	1025	**1025**	1011.2	990	0.003

Table 5. Solution permutation of best result

Problem	Solution Permutation	Best Result
P_1	0, 1, 1, 1, 0, 0, 0, 1, 1, 1	295
P_2	1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 1, 0, 1, 0, 1, 1	1024
P_3	1, 1, 0, 1	35
P_4	0, 1, 0, 1	23
P_5	0, 0, 1, 0, 1, 0, 1, 1, 0, 1, 1, 1, 0, 1, 1	481.0694
P_6	0, 0, 1, 1, 0, 1, 1, 1, 1, 1	52
P_7	1, 0, 0, 1, 0, 0, 0	107
P_8	1, 1, 1, 1, 1, 1, 0, 1, 0, 0, 0, 0, 1, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0	9765
P_9	1, 1, 1, 1, 0	130
P_{10}	1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 1, 1, 1, 1, 0, 1, 0, 1, 1, 1	1025

Acknowledgements

This study has been supported by the Scientific Research Projects of Selcuk University (in Turkey).

References

1. Nawrocki J, Complak W, Błażewicz J, Kopczyńska S and Maćkowiaki M. The Knapsack-Lightening problem and its application to scheduling HRT tasks. *Bull Polish Acad Sci Tech Sci* 2009;**57** (1):71–77.
2. Wäscher G, Haußner H and Schumann H. An improved typology of cutting and packing problems. *European journal of operational research*. 2007;**183** (3) 1109–1130.
3. Granmo O-C, Oommen BJ, Myrer SA and Olsen MG. Learning Automata-Based Solutions to the Nonlinear Fractional Knapsack Problem With Applications to Optimal Resource Allocation. *IEEE Trans Syst Man Cybern Part B*. 2007;**37** (1) 166–175.
4. Vanderster DC, Dimopoulos NJ, Parra-Hernandez R and Sobie RJ. Resource allocation on computational grids using a utility model and the knapsack problem. Future Generation Computer Systems. 2009;**25** (1) 35–50.
5. Shi H. Solution to 0/1 Knapsack Problem Based on Improved Ant Colony Algorithm. *2006 IEEE International Conference on Information Acquisition*. 2006;1062–1066.
6. Kong M, Tian P and Kao Y. A new ant colony optimization algorithm for the multidimensional Knapsack problem. *Computers & Operations* Research. 2008;**35** (8) 2672–2683.
7. Liu Y and Liu C. A Schema-Guiding Evolutionary Algorithm for 0-1 Knapsack Problem. *2009 International Association of Computer Science and Information Technology - Spring Conference* 2009;160–164.
8. Balev S, Yanev N, Fréville A and Andonov R. A dynamic programming based reduction procedure for the multidimensional 0–1 knapsack problem. *European Journal of Operational Research*. 2008;**186** (1) 63–76.
9. Zou D, Gao L, Li S and Wu J. Solving 0–1 knapsack problem by a novel global harmony search algorithm. *Applied Soft Computing*. 2011;**11** (2) 1556–1564.
10. Zhang X, Huang S, Hu Y, Zhang Y, Mahadevan S and Deng Y. Solving 0-1 knapsack problems based on amoeboid organism algorithm. *Applied Mathematics and Computation*. 2013;**219** (19) 9959-9970.
11. Wang L, Zheng X and Wang S. A novel binary fruit fly optimization algorithm for solving the multidimensional knapsack problem. *Knowledge-Based Systems*. 2013;**48** 17–23.
12. Truong TK, Li K and Xu Y. Chemical reaction optimization with greedy strategy for the 0–1 knapsack problem. *Applied Soft Computing*. 2013;**13** (4) 1774–1780.
13. Duman E, Uysal M and Alkaya AF. Migrating Birds Optimization: A new metaheuristic approach and its performance on quadratic assignment problem. *Information Sciences*. 2012;**217** 65–77.