Research Article

International Journal of Advanced Technology and Engineering Exploration, Vol 5(45)
ISSN (Print): 2394-5443 ISSN (Online): 2394-7454
http://dx.doi.org/10.19101/IJATEE.2018.545018

Applying greedy genetic algorithm on 0/1 multiple knapsack problem

Vinod Jain[1*] and Jay Shankar Prasad[2]
PhD Scholar, School of Computers & Information Sciences, MVN University, Palwal, Haryana, India[1]
Professor, School of Computers & Information Sciences, MVN University, Palwal, Haryana, India[2]

Abstract

Knapsack problem is a well-known optimization problem in computer science. It has many application areas in science and engineering. Knapsack problem can be solved using genetic algorithm. Multiple knapsack problem (MKP) is a special form of knapsack problem in which items are to be placed in more than one knapsack. Many researchers solve MKP problem using different techniques such as ant colony optimization (ACO), particle swarm optimization (PSO) and genetic algorithm (GA). The objective of this paper is to solve MKP problem using GA in an efficient manner. In this paper MKP is solved using greedy genetic algorithm. The proposed genetic algorithm uses greedy approach in its selection and reproduction operations of GA. The proposed greedy genetic algorithm is implemented on a standard data set and results ensure that the proposed greedy algorithm performs better than the standard genetic algorithm.

Keywords

Multiple knapsack problem, Genetic algorithm, Greedy approach.

1.Introduction

The knapsack problem is a well-known NP-hard problem. The knapsack problem is an optimization problem in which the overall profit is to be optimized and should be maximized while satisfying all the given constraints. The knapsack problem is of many types such as 0/1 knapsack problem, multidimensional knapsack problem, multiple knapsack problem etc. In this paper, a 0/1 multiple knapsack problem is solved. The MKP is a generalization of 0/1 knapsack problem. In 0/1 multiple knapsack problems a set of n items is given. A set of m knapsacks is given. Each item j has a profit Pj and weight Wj. Each knapsack i has a capacity Ci. We have to assign items in a knapsack and we can assign one item in most one knapsack. We have to assign items into knapsack such that none of the capacity constraints of the knapsack is violated. The total profit of assigning items to knapsacks should also be maximized.

MKP has a large number of applications in science and engineering. Many problems can be simulated in the form of MKP and can be solved using MKP solutions.

A typical MKP problem may have hundreds of knapsacks and thousands of items. Many techniques are used to solve MKP problem such as branch and bound, greedy approach, backtracking, bound and bound etc. These approaches guarantee to give the best possible solution but have complexity O(mn) in the worst case. As the worst case time complexity of solving this problem is very high so these problems can be solved using meta-heuristic techniques such as GA, ACO and PSO etc.

The main objective of this paper is to solve MKP problem using GA in an efficient way. A new greedy genetic algorithm is proposed which solve the MKP problem in less number of iterations as compared to standard generic algorithm.

The organization of the paper is as follows: The next section illustrates the literature survey of the work done on this problem till date. Then the proposed algorithm to solve MKP problem is given which is followed by result analysis, conclusion, and future scope.

2.Literature survey

Umbarkar et al. [1] propose a genetic algorithm with the dual population to solve optimization problems. The application of dual population improves the performance of the genetic algorithm for solving these problems. It basically removes the problem of

*Author for correspondence

premature convergence of genetic algorithm by providing diversity in the population. Pal et al. [2] provide a comparative study on solving optimizations problems using Firefly algorithm and particle swarm optimization. The author compares the convergence of the genetic algorithm for solving different linear optimization problems.

Xiao-hua et al. [3] proposed a competitive decision algorithm for multiple knapsack problems. The author proposes a competitive decision algorithm (CDA) to solve multiple knapsack problems. The basic idea used by the author is that an entity with more resources has a high chance to survive. So the algorithm works on surviving the entities with more resources and it performs well as compared to other existing algorithms. The author design CDAMKP i.e. CDA algorithm to solve MKP problem.

Kobayashi et al. [4] proposed a cryptosystem which is based upon MKP problem. Salman et al. [5] solve bicriteria sparse MKP problem. Godrich et al. [6] uses knapsack problem in multiple radar architectures for localization. Kumaraguruparan et al. [7] apply MKP problem in task scheduling. Rahim et al. [8] apply MKP problem in home energy management. Li et al. [9] apply MKP problem in graph theory in bipartite graphs. Author have been applied GA on travelling salesman problem and N-Queen problem [10−12].

3.Proposed work heading

In this work, multiple knapsack problem is solved using greedy genetic algorithm. Algorithm-1 is a standard form of the genetic algorithm which can solve any problem that can be represented in genetic form.

Algorithm-1-Standard genetic algorithm

(1) Generate initial greedy population for MKP
(2) Calculation of fitness of every chromosome
(3) Select parents- In this step parents are selected from the population to perform crossover.
(4) Perform crossover- in this step crossover is performed. The crossover operator generates new children
(5) Add Children in Population
(6) Calculation of Fitness
(7) Perform Mutation
(8) Perform more Generations
(9) End-in this step, the genetic algorithm is stopped and results are displayed.

Algorithm-2 is proposed greedy genetic algorithm that solve MKP problem. The algorithm generates a greedy initial population.

Algorithm 2: Proposed greedy genetic algorithm

(1) Represent given problem in genetic form and generate the greedy initial population of chromosomes
(2) Calculate fitness of every chromosome of the initial population
(3) Repeat steps 4 to 8 till an optimal solution is not reached
(4) Select a set of parents from initial population to perform crossover
(5) Perform cross over to generate children and calculate the fitness of every child
(6) Add newly generated children chromosomes in population and sort population by fitness
(7) Remove those chromosomes from the population which are less fit
(8) Perform mutation and create population for next iteration

Figure 1 is showing a process flowchart to solve the MKP problem using greedy genetic algorithm. The important point is that the proposed flowchart generates a greedy initial population. It is based on the concept that a better population will generate better next generation. So the seed population is very efficient in proposed algorithm.

The proposed algorithm is implemented on a data set of 0/1 Multiple knapsack problem available online at the website of Florida State University at the department of scientific computing [13]. The data set having many instances of 0/1 multiple knapsack problem. In this paper, the proposed algorithm is applied on standard MKP problems available in dataset which is p01. P01 is a set of 10 weights and profits for 2 knapsacks of capacity 70 and 127. The following *Table 1* illustrates the profit and weight of items in the p01 problem.

Table 1 is showing a candidate solution for the p01 knapsack problem. The solution put Item1 and Item4 in knapsack-1(Knp1) and Item2, Item3, and Item5 in knapsack-2 (Knp2). So the weight consumed in knapsack-1 is 67 and profit earn is 160. The weight consumed by knapsack-2 is 113 and profit earn is 166. Thus total profit earn by this solution is 326.

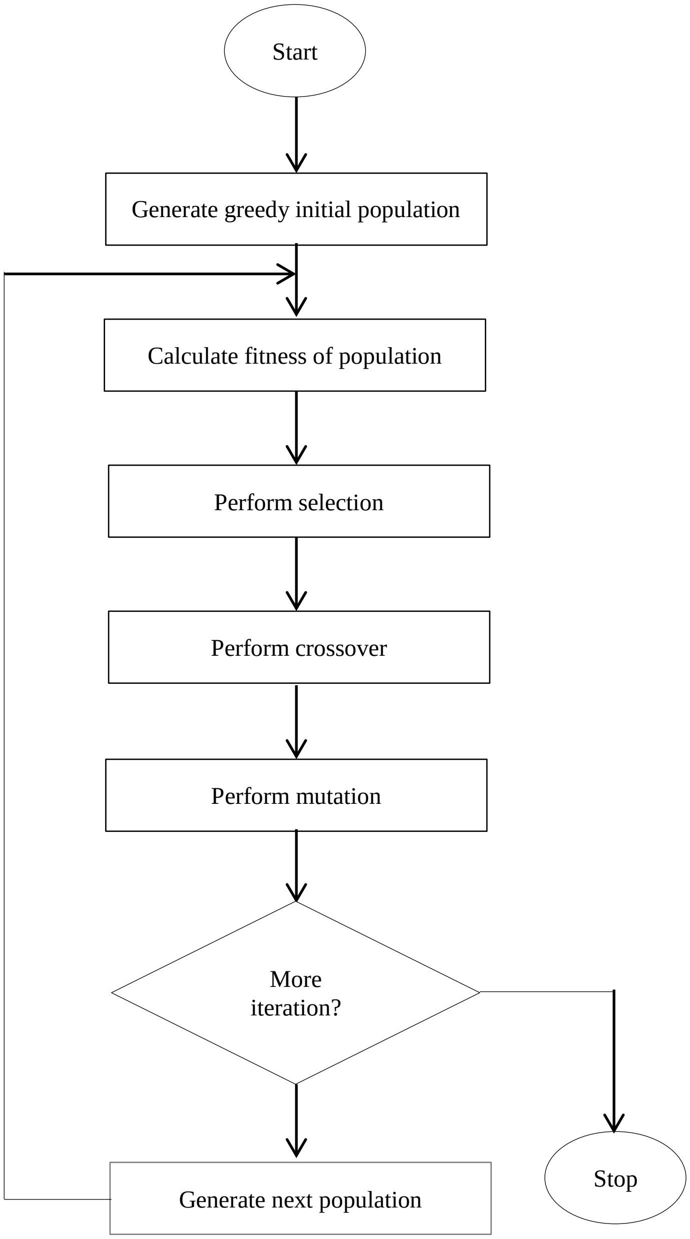

Figure 1 Proposed greedy genetic algorithm to solve MKP problem

Table 1 Representation of p01 multiple knapsack problem

Multiple knapsack instance - P01					
A sample solution of the problem					
Item	**Weight**	**Profit**	**Profit /weight**	**Knp1**	**Knp2**
Item1	23	92	4.00	1	0
Item2	31	57	1.84	0	1
Item3	29	49	1.69	0	1
Item4	44	68	1.55	1	0
Item5	53	60	1.13	0	1
Item6	38	43	1.13	0	0
Item7	63	67	1.06	0	1
Item8	85	84	0.99	0	0
Item9	89	87	0.98	0	0
Item10	82	72	0.88	0	0

4.Results and analysis

The proposed algorithm is applied to one instances (P01) of a sample data set for multiple knapsack problem. *Table 2* is showing the results of instance p01 using the standard genetic algorithm. It applies the standard genetic algorithm on P01 MKP problem and execute 50 iterations of it. The *Table 2* is showing result of the best solution in 50 iterations. *Table* shows the results after every 5 iterations. The best solution with profit 282 is achieved in 15th iteration and does not change even after 50th iteration.

The proposed greedy genetic algorithm is applied on same P01 MKP problem. *Table 3* is showing results 50 iterations of the implementation. *Table 3* is showing results of the best solution after every 5 iterations. It is clear from the *Table 3* that the best solution (with profit 326) which is achieved in 15 iterations is the best solution and does not change even after 50th iteration. *Table 4* is showing a comparison of results found using the standard genetic algorithm and greedy genetic algorithm. From the *Table 4*, it is clear that the proposed genetic algorithm performs 18.09% better than the standard genetic algorithm while solving p01 multiple knapsack problem.

Table 2 Results for solving p01 MKP problem using the standard genetic algorithm

Multiple knapsack instance - p01				
Random genetic algorithm				
Generation No.	**Best solution known**	**Best solution by GA**	**Average solution of that generation**	**Error (%)**
1	333	241	192.15	38.17
5	333	241	206.7	38.17
10	333	244	213.35	36.48
15	333	282	230.8	18.09
20	333	282	230.8	18.09
25	333	282	230.8	18.09

Multiple knapsack instance - p01				
Random genetic algorithm				
Generation No.	**Best solution known**	**Best solution by GA**	**Average solution of that generation**	**Error (%)**
30	333	282	230.8	18.09
35	333	282	230.8	18.09
40	333	282	230.8	18.09
45	333	282	230.8	18.09
50	333	282	230.8	18.09

Table 3 Results for solving p01 MKP problem using the greedy genetic algorithm

Multiple knapsack instance-p01				
Random genetic algorithm				
Generation No.	**Best solution known**	**Best solution by GA**	**Average solution of that generation**	**Error(%)**
1	333	309	231.85	7.77
5	333	309	244	7.77
10	333	309	253.7	7.77
15	333	326	263.95	2.15
20	333	326	266.1	2.15
25	333	326	268.85	2.15
30	333	326	268.85	2.15
35	333	326	268.85	2.15
40	333	326	268.85	2.15
45	333	326	268.85	2.15
50	333	326	268.85	2.15

Table 4 Results comparison using standard genetic algorithm and greedy genetic algorithm

Sr. No.	**MKP instance**	**Optimal solution known**	**Best solution found using standard GA**	**Best solution found using proposed greedy GA**
1	p01	333	282	326

5.Conclusion and future scope

From this paper, it is concluded that the performance of genetic algorithm depends upon the performance of its genetic operations. Applying greedy approach in the process of initial population generation improves the performance of the genetic algorithm in solving 0/1 multiple knapsack problem. From the results of the experiments, it is clear that the proposed greedy genetic algorithm performs 12% better as compared to the standard genetic algorithm. In future this work can be extended to apply the greedy approach in other genetic operations such as selection, crossover and mutation and its performance can be checked. The proposed algorithm can be applied to other instances of knapsack problem such as multidimensional knapsack problem. It is concluded that for solving NP-complete and NP-hard problems the application of greedy approach in operators of the genetic algorithm, improves the performance of the genetic algorithm in finding better solutions.

Acknowledgment

None.

Conflicts of interest

The authors have no conflicts of interest to declare.

References

[1] Umbarkar AJ, Joshi MS, Sheth PD. Dual population genetic algorithm for solving constrained optimization problems. International Journal of Intelligent Systems and Applications. 2015; 7(2):34-40.

[2] Pal SK, Rai CS, Singh AP. Comparative study of firefly algorithm and particle swarm optimization for noisy non-linear optimization problems. International Journal of Intelligent Systems and Applications. 2012; 4(10):50-7.

[3] Xiao-hua X, Liang M, Ai-bing N. Competitive decision algorithm for multiple-choice knapsack problem based on reduction. In international conference on computer modeling and simulation 2010 (pp. 344-8). IEEE.

[4] Kobayashi K, Tadaki K, Kasahara M, Tsujii S. A knapsack cryptosystem based on multiple knapsacks. In international symposium on information theory and its applications 2010 (pp. 428-32). IEEE.

[5] Salman FS, Kalagnanam JR, Murthy S, Davenport A. Cooperative strategies for solving the bicriteria sparse multiple knapsack problem. Journal of Heuristics. 2002; 8(2):215-39.

[6] Godrich H, Petropulu AP, Poor HV. Sensor selection in distributed multiple-radar architectures for localization: a knapsack problem formulation. IEEE Transactions on Signal Processing. 2012; 60(1):247-60.

[7] Kumaraguruparan N, Sivaramakrishnan H, Sapatnekar SS. Residential task scheduling under dynamic pricing using the multiple knapsack method. Innovative Smart Grid Technologies.2012:1-6.

[8] Rahim S, Khan SA, Javaid N, Shaheen N, Iqbal Z, Rehman G. Towards multiple knapsack problem approach for home energy management in smart grid. In international conference on network-based information systems 2015 (pp. 48-52). IEEE.

[9] Li J, Li W, Wang H. The multiple knapsack problem with compatible bipartite graphs. International symposium on operations research and its applications in engineering, technology and management 2015(pp. 1-7). IEEE.

[10] Jain V, Prasad JS. Solving travelling salesman problem using greedy genetic algorithm GGA. International Journal of Engineering and Technology. 2017; 9(2):1148-54.

[11] Prasad JS, Jain V. An optimized algorithm for solving travelling salesman problem using greedy cross over operator. International conference on computing for sustainable global development 2016 (pp. 2981-4). IEEE.

[12] Farhan AS, Tareq WZ, Awad FH. Solving N queen problem using genetic algorithm. International Journal of Computer Applications. 2015; 122(12):11-4.

[13] https://people.sc.fsu.edu/~jburkardt/datasets/knapsack_multiple/knapsack_multiple.html. Accessed 26 May 2018.

Vinod Jain completed his Master of Technology (Computer Engineering) YMCA University, Faridabad, Haryana, India (2012) and Master of Computer Application (MCA), Kurukshetra University (2004). He is cureently a Research Scholar in MVN University Palwal, Haryana, India. He worked as a Assistant Professor, B.S.Anangpuria Institute of Technology and Management Faridabad, Haryana from September-2008 to January-2018. Currently working as a Assistant Professor, Rawal Institute of Engineering and Technology, Faridabad, Haryana since January 2018. He has published more than 11 papers in international journals and international conferences. His area of research include Genetic Algorithms, NP-Complete and NP-Hard Problems, Search Engine Optimization, Page Ranking, Crawling, Indexing, Web Mining etc. His current area of research is solving NP-Complete and NP-Hard problems using Genetic Algorithm.

Email: jainvinod81@gmail.com

Jay Shankar Prasad research interests are Artificial Intelligence, Pattern Recognition, Machine Learning, Computer Vision, Robotics, Humanoid Robots, Gesture Recognition, ISL Recognition, Pattern Mining, Cloud Computing etc. He has published Twelve papers in International journals and International conferences. He has 16 years of teaching and 3 years of software industry experience. He also guided many postgraduate and under graduate level projects.