International Journal of Statistics and Applied Mathematics

ISSN: 2456-1452
Maths 2025; 10(2): 26-28
© 2025 Stats & Maths
https://www.mathsjournal.com
Received: 24-11-2024
Accepted: 29-12-2024

Adel Hashem Nouri
Department of Computer Science, College of Education, University of Kufa, Najaf, Iraq

Corresponding Author:
Adel Hashem Nouri
Department of Computer Science, College of Education, University of Kufa, Najaf, Iraq

Some important algorithms used in knapsack problem

Adel Hashem Nouri

DOI: https://dx.doi.org/10.22271/maths.2025.v10.i2a.1976

Abstract
The Knapsack problem, an essential optimization issue with broad applicability, was the focus of our investigation. A number of studies and research looked at several approaches to solve it using both standard and hybrid algorithms, and they compared those approaches.

Keywords: Knapsack problem, optimization, branch and bound algorithm, genetic algorithm

Introduction

The knapsack problem, also known as the soldier's bag, is a combination optimization problem in complex mathematics. It is one of the many complex and variable problems that fall into the so-called NP-complete and NP-hard categories, according to their level of complexity. It offers a plethora of different option. It is said that complex optimization issues are those requiring...
The comprehensive research approach is very challenging, and its complexity rises as the problem's size does. This is because the optimum answer is difficult to locate and test, and the time required to do so is variable rather than constant.
The knapsack problem can be created using the best time method since there is no one algorithm for it. The fact that its calculation grows exponentially with the issue's magnitude encouraged academics to keep digging.. Through the development and implementation of various algorithms, the search techniques of these algorithms may be tweaked or combined to address various versions of the baggage issue.
For the most part, the best solutions to issues have been discovered in recent years by using straightforward algorithms.
The Minya genetic algorithm was used to resolve the backpack issue; nevertheless, the conventional genetic algorithm in In many instances, achieving an acceptable outcome is just not feasible. As a consequence of the extensive search space, the outcomes are subpar when compared to algorithms such as the greedy and ant colony algorithms.

1. Knapsack Problem

A-The goal of solving the 0-1 Knapsack Problem is to minimise the weight total without exceeding the capacity c by selecting a subset of the n items that maximise the related profit sum. First, there's the 0-1 knapsack issue, sometimes known as the linear programming problem [1].

$$\text{Maximize} \quad \sum_{j=1}^{n} p_j x_j$$

$$\text{subject to} \quad \sum_{j=1}^{n} w_j x_j \leq W,$$

$$x_j \in \{0,1\} \quad \forall j \in \{1,\ldots,n\}$$

A famous combinatorial optimization problem, the 0/1 knapsack problem (KP) usually occurs in resource allocation with budgetary limitations.

The selection of capital investments, financial portfolio management, and the most efficient method of cutting raw materials are only a few examples of the many real-world decision-making processes that include it.

B-The ability to select an item more than once is a typical variation. For every item j in the limited knapsack problem, there is a maximum allowed selection frequency uj, which can be either an integer greater than zero or infinite.

$$\text{maximize} \sum_{j=1}^{n} p_j x_j$$

$$\text{;subject to} \quad \sum_{j=1}^{n} w_j x_j \le W,$$

$$u_j \ge x_j \ge 0, x_j \quad \text{integral}$$

$$\text{for all } j$$

C-In the unbounded knapsack problem, often known as the integer knapsack problem, there is no maximum allowed selection frequency for an item:

$$\text{maximize} \sum_{j=1}^{n} p_j x_j$$

$$\text{subject to} \sum_{j=1}^{n} w_j x_j \le W,$$

$$x_j \ge 0, x_j \quad \text{integral for all } j$$

D- We obtain the multiple-choice knapsack issue when we split the items into k groups and require that one item be selected from each class:

$$\text{maximize} \sum_{i=1}^{k} \sum_{j \in N_i} p_{ij} x_{ij}$$

$$\text{subject to} \sum_{i=1}^{k} \sum_{j \in N_i} w_{ij} x_{ij} \le W,$$

$$\sum_{j \in N_i} x_{ij} = 1, \qquad \text{for all } 1 \le i \le k$$

$$x_{ij} \in \{0,1\} \qquad \text{for all } 1 \le i \le k \text{ and all } j \in N_i$$

E- The subset sum issue (or its equivalent choice problem) arises if the profits and weights of each item are the same:

$$\text{maximize} \sum_{j=1}^{n} p_j x_j$$

$$\text{subject to} \quad \sum_{j=1}^{n} p_j x_j \le W,$$

$$x_j \in \{0,1\}$$

3. The Algorithms used in Knapsack Problem

3.1. Branch-and-Bound Algorithm

This algorithm design model for discrete and combinational optimisation problems, as well as mathematical optimization, is known as Branch and Bound (BB, B&B, or BnB). It involves breaking optimization problems into smaller subproblems and then using a select function to remove subproblems that cannot contain the optimal solution. The branch-and-reduce algorithm lists candidate solutions by searching the state space in a systematic way.

The potential solutions are organized in a rooted tree structure, with the complete set at the base. The algorithm iteratively checks each branch against the estimated upper and lower bounds of the optimal solution before enumerating candidate solutions for that branch. If a branch cannot produce a better solution than the best solution the algorithm has found thus far, it is discarded.

3.1.1. A solution to the Multiple-choice Multidimensional Knapsack problem (MMKP) was presented by researchers Mhand HIFIň Slim SADFI and Abdelkader SBIHI in 2004.

The algorithm's guiding idea is to construct a search tree that can enumerate the sorted solutions in the appropriate orde

ˆXp, ˆXp−1, . . . , ˆXmax

A fixed item in the solution corresponds to each developed node in the search tree. A node in the search tree represents an answer. We keep track of the developed node for item j in class Ji by using the notation nij.

Using the best-first search technique, the algorithm implemented a branch-and-bound method. Prior to this, we started with a lower bound (a plausible solution) calculated using Hifi's heuristic.

In addition, many branches of the search tree might be understood by combining the original lower constraint with the intervening higher bounds. Results from the trial proved that the suggested strategy could handle medium and small cases with up to 1000 items in each of 50 classes, each with 20 items, and up to seven restrictions [2].

3.1.2. The multiple knapsack problem (MKP) was solved in 2009 by researcher Alex S. Fukunaga using a branch-and-bound algorithm.

Think of m bins with capacity c1,..., cm and n items with weights w1,..., wn and profits p1,..., pn. The goal of solving the 0-1 Multiple Knapsack Problem (MKP) is to maximise the total profit of the things by packing them into the containers in such a way that their combined weight does not exceed the capacity of the containers, and each item is assigned to no more than one container.

At each node, he integrated Pisinger's bound-and-bound technique into our bin-completion solver. This involves dividing the SMKP solution into the remaining bins, with the goal of validating the SMKP upper limit. It should be remembered that in bin-completion, at depth b, m− b bins are empty. The SMKP cases are solved using Pisinger's Minknap 0-1 Knapsack solver algorithm. If there are many optimum solutions to the SMKP instance, the algorithm is adjusted to return the one with the least weight total. This makes it more probable that the bound-and-bound subset sum solver, just like Pisinger's Mulknap solver, can divide the solution [3].

3.2 Genetic Algorithm

The genetic algorithm is an approach to optimization that uses the principles of natural selection—the driving force behind biological evolution—to solve problems with or without constraints. An ensemble of solutions is iteratively refined using the genetic algorithm. Using the present population as a starting point, the genetic algorithm chooses parents at each stage to generate offspring. The population "evolves" as time goes on to find the best possible answer.

When the objective function is discontinuous, no differentiable, stochastic, or very nonlinear, the genetic algorithm may be used to address optimization issues that aren't good fits for traditional optimization techniques. When some variables must be integers, the evolutionary algorithm may solve mixed integer programming problems.

3.2.1. In 2020, researchers Modestus OKWU, Omonigho B. Otanocha, Henry O. Omoregbee, and Bright A. Edward conducted a study using a genetic algorithm.

Genetic algorithm (GA) is an effective stochastic strategy for solving complicated combinatorial problems, according to a research. The research team used GA to solve the knapsack 0-1 problem (KB). This method can identify near-optimal or near-linear solutions to the NP problem by reducing the complexity of KP from exponential to linear.

There is no specific combination that would provide the precise weight or capacity that the 35 kilogramme bag can hold, according to the GA model's answer. However, the model suggests a range of 34 kg to 36 kg as feasible. With a bag weight of 35 kg, the most practical or almost ideal approach would be to reduce the weight of the goods it can carry to 34 kg, which would result in a profit of 16. If the weight exceeds 34 kg, the bag may rupture. For process optimization in the engineering domain, such as gas pipeline optimization and quality control, the authors suggest using genetic algorithms in future studies [4].

3.2.2. Scientists Hakimi, D., Oyewola, Do., Yahaya, Y., And Bolarin, G. proposed a genetic algorithm (GA) solution to the backpack issue in 2016.

Six distinct ones were compared

We have many types of crossovers, including: single-point, two-point, dispersed, computational, and intermediate. Triple aspect

To check whether the pouch issue has converged, one uses pouch problems. In light of the outcomes of our experiments: The optimal solution for the bag issue was two-point (TP) intersections.

There are two proposed solutions to the Knapsack problem: Point Junction (TP), Single Point Junction (SP), Sparse intercept (SC), arithmetic intercept (AM), heuristic intersection (HE), and median crossover (I, T). According to our research and experimental findings, two things.

It was found that the intersection (TP) performed better than SP, SC, AM, HE, and IT. The findings pointed to the possibility of using this Two-point (TP) intersection to resolve the Bag issue.

In addition, knapsack problem convergence is tested in three dimensions, and the results demonstrate that two-point crossover (TP) is an excellent method for solving both small and big knapsack issues. As a potentially lucrative solution strategy for Knapsack issues, two-point crossover (TP) might be suggested [5].

3.2.3 Scientist Ali Nadi Ünal offered a number of strategies for transforming the rise in GA diversity into a living, breathing ecosystem in 2013. Memory-Based GA (MBGA) and Random Migrants Based on GA (RIGA) were two of these approaches that were compared. According to the findings, MBGA works better.

In the file "Changing Environment," RIGA addresses the issue of several backpacks (0/1).

It is devised and executed to solve the MKP issue

We examine two methods for dealing with dynamic environments: A memory-based genetic algorithm (MBGA) and a random gene-based migratory algorithm (RIGA). One metric utilised is offline performance. Weigh two methods from among three distinct alternatives. Biological context (10, 100, 500 ms) [6].

Conclusion

The Knapsack problem is one of the important complex optimization problems and has been studied over many years by several researchers. In this research, we have addressed some solutions to various types of these problems by studying some algorithms that have been proposed by researchers in terms of efficiency, speed and accuracy. We have discussed the Branch-and-Bound algorithm and the genetic algorithm and compared them in terms of results.

References

1. Kolte J, Jhaa D, Datt R. Strategies for implementing the Knapsack Problem. Int J Sci Technol Manag. 2017, 6(6).
2. Hifi M, Sadfi S, Sbihi A. An Exact Algorithm for the Multiple-choice Multidimensional Knapsack Problem. Maison des Sciences Économiques; c2004. p. 106-112.
3. Fukunaga AS. A Branch-and-Bound Algorithm for Hard Multiple Knapsack Problems. Draft, Oct 2009.
4. Okwu M, Otanocha OB, Omoregbee HO, Edward BA. Appraisal of genetic algorithm and its application in 0-1 knapsack problem. J Mech Energy Eng. 2020;4(44):39-46.
5. Hakimi D, Oyewola DO, Yahaya Y, Bolarin G. Comparative analysis of genetic crossover operators in knapsack problem. Full-text Available Online. 2016;20(3):593-596.
6. Ünal AN. A genetic algorithm for the multiple knapsack problem in dynamic environment. In: Proceedings of the World Congress on Engineering and Computer Science, Vol II; c2013.