Report on two search algorithms to solve the knapsack problem

Executive summary

Simulated Annealing and Tabu Search are selected to solve the 0-1 knapsack problem because they are metaheuristics that can either avoid or escape the local optima. There are many of real-life applications of the knapsack problem. These applications are not limited to revenue management for airlines, financial modelling, as well as production and inventory control systems. For the chosen parameters, Simulated Annealing achieved an average total value of packing in 10 runs of 4413.6, while Tabu Search achieved that of 4378.0. In terms of the mentioned metric, Simulated Annealing is slightly better than Tabu Search. However, Simulated Annealing is a stochastic algorithm and Tabu Search is, on the other hand, a deterministic algorithm. The performance (total value of packing in each run) of Tabu Search is more stable than that of Simulated Annealing. In the future, further tuning and trials of parameters can be performed for the current Simulated Annealing and Tabu Search Algorithms. Parallelised Simulated Annealing algorithm and Quantum Inspired Tabu Search can be applied and implemented on 0-1 knapsack problem.

Introduction

The knapsack problem is one of the problems in combinatorial optimisation. When given a set of items, each of which has a weight and a value, determine the number of items to include such that the sum of the weights is less than the given limit and the sum of the values is as large as possible. The 0-1 knapsack problem, which is at the heart of the knapsack problem, is the problem of selecting a subset of n items for a single knapsack. The total profit of all the selected items must be maximized without the total weight exceeding the capacity of the knapsack. [1] In this report, the discussion is only on 0-1 knapsack problem. The general formulation of this problem is as follows:

Maximise $\sum_{i=1}^{n} p_i x_i$,

subject to $\sum_{i=1}^{n} w_i x_i \leq c$,

$x_i \in \{0,1\}, \qquad i = 1,2,3,\dots,n$

where p_i and w_i are the profit and weight of each item respectively, and x_i is 1 if item i is assigned to the knapsack or else 0.

Simulated Annealing and Tabu Search are selected to solve the 0-1 knapsack problem. This is because both Simulated Annealing and Tabu Search are metaheuristics that can either avoid or escape the local optima. On the other hand, local search method converges to the local optima. The search concepts of these two methods are extremely different. Whereas simulated annealing is developed from the correspondence between physical processes and combinatorial problems, Tabu search can be seen in the context of artificial intelligence. [2]

The Simulated Annealing algorithm that can be used to solve combinatorial problems works in the following way [3]:

1. Start with an arbitrary initial solution x and choose a temperature parameter T.
2. Choose a transition at random and construct the corresponding solution x'.
3. Calculate the value of the objective function and the difference d between the values of x and x'.
4. If x' is better than x, accept x' as the current solution x. Else accept the solution x' with a probability p(d, T).
5. Decrease the temperature T
6. if T > 0 GOTO 2.

The Tabu Search that be used to solve combinatorial problems works in the following way [4]:

1. Start with any initial solution as the current solution
2. Initialise a tabu memory
3. Repeat the following if the stop condition is not met
4. List N candidate actions as a subset of the current solution
5. Select the one that minimizes the extended cost function, s, from the list of candidate solutions
6. Update the Tabu memory and specify s as the current solution

Literature review

There are many of real-life applications of the knapsack problem. These applications include revenue management for airlines [5], financial modelling [6], production and inventory control systems [7], the design of queuing network models in manufacturing [8], and the control of traffic overload in telecommunications systems [9].

The stochastic Knapsack with finite horizon is a combination of the secretary problem and the integer Knapsack problem. It is useful for optimising the sale of perishable goods with low marginal cost to impatient customers. van Slyke and Young [5] extended their model to multidimensional knapsacks. This model is a generalisation of the current aircraft yield management model, as it can take into account group bookings and cargo scheduling.

A wide range of assets can be traded in the financial markets, including stocks, bonds, foreign exchange, options, commodities, real estate and futures contracts. The quality of these assets can range from very good to very poor. Often, it is difficult for investors to find good quality assets due to information asymmetry and asset price volatility. Patalia and Kulkarni [6] devised a genetic algorithm for the knapsack problem of selecting a portfolio of stocks using financial indicators.

Bretthauer et al. [7] proposed an efficient algorithm for solving continuous and integer variable versions of resource-constrained production and inventory management models. The algorithm requires the solution of a series of two sub-problems: a non-linear knapsack

problem and a non-linear problem where only the lower and upper bounds on the variables are constrained.

Smith and N. Chikhale [8] proposed a method for finding optimal buffer allocations in constrained network design problems. In many network design applications, such as manufacturing facilities, communication networks and vehicular traffic systems, random networks of service centres with different service rates and limited waiting capacity (buffers) occur. One of the most difficult tasks for network designers is to allocate a buffer to each service centre while taking into account the total capacity of the network.

In Long Term Evolution (LTE) networks, there are many mobile bandwidth overloads that degrade the quality of service. Ferdosian et al. [9] analysed the performance of the greedy-knapsack scheduler as an alternative optimal solution for network overload conditions through a wide range of resource requirements for different classes of services.

Analysis and evaluation of the results

Initial_solution (initial_proportion_items)

- The strategy for determining the initial solution is to select an item at random from the n = 100 items (generating a random binary set). In the problem, whether item i (iϵ[1, n]) is selected or not is defined as a binary variable.

- For this reason, the initial solution may become infeasible. To prevent the initial solution from becoming infeasible, the function Initial_solution (initial_proportion_items) is defined, which allows the proportion of items in the initial solution to be changed.

- This means that 10 items are chosen at random in the initial solution (0.1) and 30 items are chosen at random in the initial solution (0.3).

Simulated Annealing

Function defined in the code: Simulated_Annealing (initial_proportion_items = 0.1, initial_temperature = 8000, iter_per_temperature = 50, final_temperature =5)

initial_proportion_items is proportion of items selected in initial solution

Neighbourhood criterion used: 1-flip neighborhood

Cooling Schedule**:** Cauchy cooling schedule is used for updating temperature:

$$T_k = \frac{Initial\ Temperature}{k+1}$$

where *k* is the temperature step.

Stopping criteria: The algorithm is set to stop when the temperature becomes less than 5.

Simulated Annealing was run for 10 times.

The best solution found is at the fifth run:

best packing [1 1 1 1 0 0 0 0 1 1 1 1 1 1 0 1 1 0 1 1 1 1 1 1 0 0 0 1 1 0 1 1 0 1 1 0 1

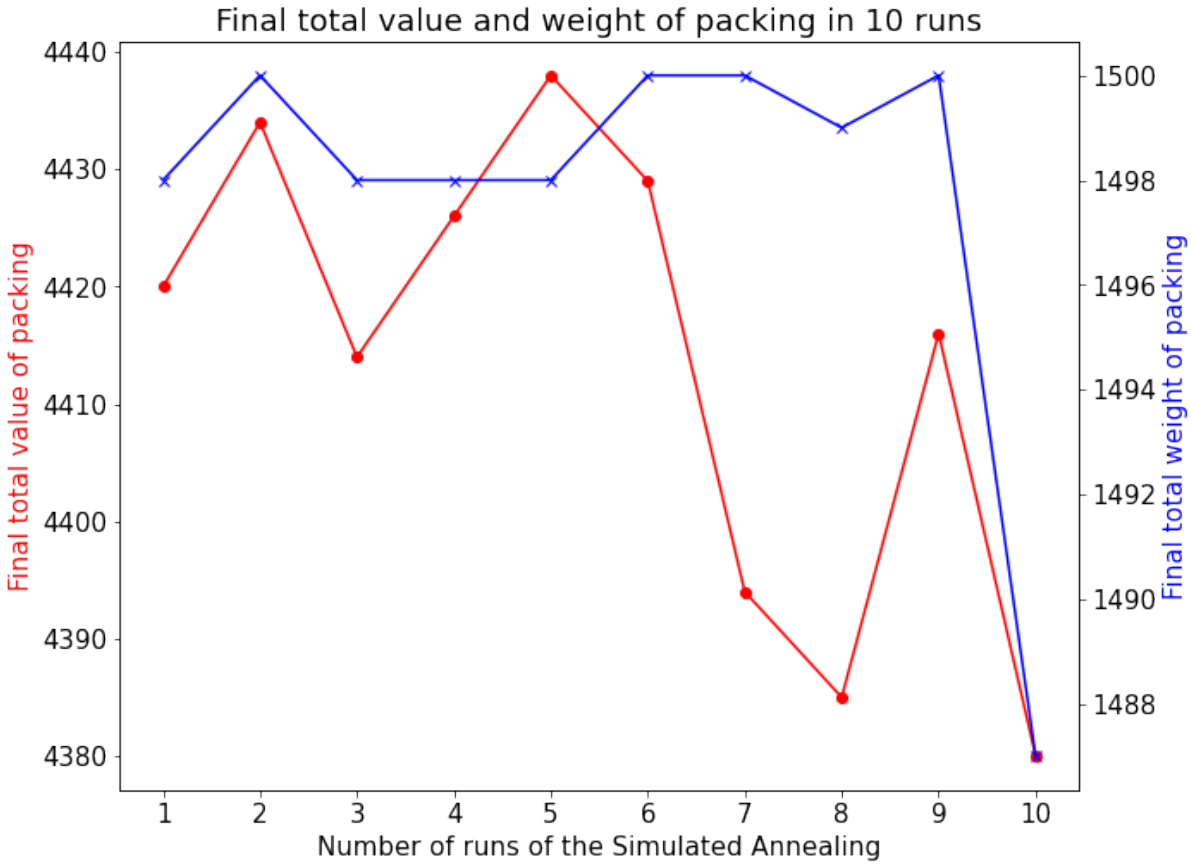
1 0 1 1 1 0 0 1 1 0 1 0 1 1 1 0 1 1 1 0 1 1 0 0 1 1 0 1 1 1 1 0 1 1 1 1 1

1 1 1 0 1 0 0 1 1 0 1 0 1 0 1 1 0 1 1 1 0 1 0 1 0 1]

Value: 4438 Weight: 1498

Total number of random steps: 61609, Total number of improvements: 52072

Number of packings checked: 79950



Average total value of packing in 10 runs = 4413.6

Average total weight of packing in 10 runs = 1497.8

Tabu Search

Function defined in the code: Tabu_Search(initial_proportion_items = 0.1, tabu_tenure = 30, max_super_best_steps = 100)

Tabu criteria: When any item is selected or deselected in the current solution, it is forbidden (impossible) to change its state before the step of tabu_tenure.

Aspirational criteria: If there is a solution in the neighbourhood of the current solution with a higher target value than the previously observed target value, then the tabu criterion is ignored and the current solution is moved to that neighbourhood.

Stop criterion: If e_super_best (the highest target value) does not change under max_super_best_steps, then the tabu search is stopped.

Tabu Search was run for 10 times.

The best solution found is at every run

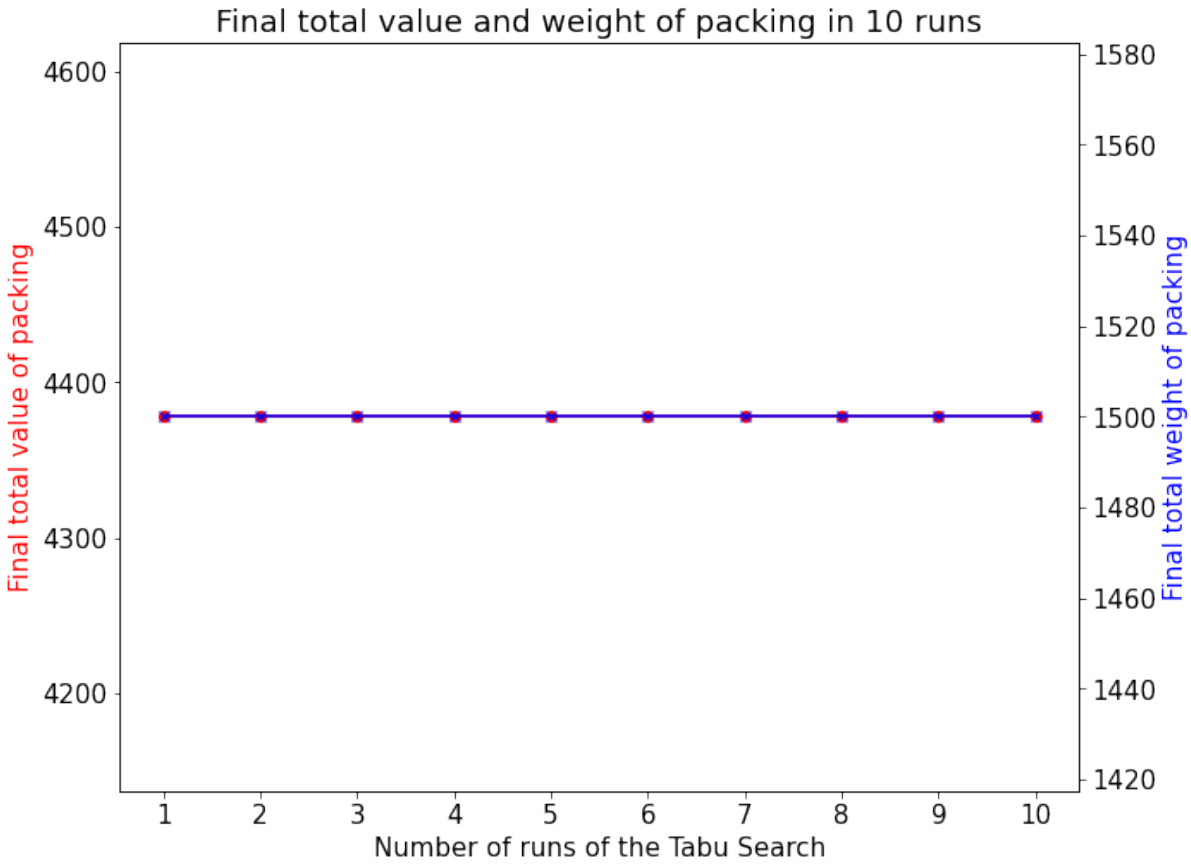
best packing [1 1 1 1 0 0 0 0 1 1 1 1 1 1 0 1 0 0 1 1 0 1 1 1 0 0 0 1 1 0 1 1 0 1 1 0 1

1 0 1 1 1 0 0 1 1 0 1 0 1 1 1 0 1 0 1 0 1 1 0 1 1 1 0 0 0 0 1 0 1 1 0 1 1

1 1 1 0 1 0 0 1 1 1 1 0 1 1 1 1 0 1 1 1 0 1 0 1 0 0]

Value: 4378 Weight: 1500

Number of packings checked: 16000



Average total value of packing in 10 runs = 4378.0

Average total weight of packing in 10 runs = 1500.0

Comparison of results

For the chosen parameters, Simulated Annealing achieved an average total value of packing in 10 runs of 4413.6, while Tabu Search achieved an average total value of packing in 10 runs of 4378.0. In terms of the average total value of packing in 10 runs, Simulated Annealing is slightly better than Tabu Search. However, Simulated Annealing is a stochastic algorithm and Tabu Search is, on the other hand, a deterministic algorithm. The performance (total value of packing in each run) of Tabu Search is more stable than that of Simulated Annealing. For Simulated Annealing, using the chosen parameters, the maximum total value of packing is 4438, and the minimum total value of packing is 4380. On the other hand, the total value of packing of Tabu Search is the same for every run of the algorithm. Also, the total value of packing of Simulated Annealing fluctuates for every run of the algorithm. Nevertheless, using the chosen parameters, the total value of packing in each run of Simulated Annealing, is higher than that of Tabu Search.

Conclusion

There are some disadvantages of Simulated Annealing. There are a lot of tuneable parameters in Simulated Annealing algorithm. It can take a long time to run if the annealing schedule is very long. [10]

A number of methods have been introduced to improve the computational efficiency of Simulated Annealing algorithms [11]. However, most of these methods use heuristics that simplify the search process by limiting the cost function or reducing the likelihood of generating future actions that would otherwise be rejected.

A more efficient way to increase the speed of computation is to use parallel processor platforms to run Simulated Annealing algorithms. In order for the parallel Simulated Annealing algorithm to have the same desirable convergence properties as the sequential algorithm, the sequential decision sequence must be retained or a different decision sequence must be used so that the resulting Markov chain has the same probability distribution as the sequential algorithm [12].

Similar to Simulated Annealing, there are also some limitations for Tabu Search. There can be a large number of iterations. Fine tuning of parameters are needed for improving results. [2]

Chiang et al. [13] proposed a new quantum-inspired evolutionary algorithm (QEA) called Quantum Inspired Tabu Search (QTS). QTS is based on classical tabu search and superposition, which is a feature of quantum computing. The measurement of quantum bits is a random operation, which adds diversity. The strength is also increased by using quantum revolving gates to search for attractive regions. experimental results for the 0-1

knapsack problem are compared with other heuristics such as classical genetic algorithms, tabu search algorithms and the original QEA. The results show that QTS performs more efficiently than the other heuristics and has no early convergence.

For further development, further tuning and trials of parameters can be performed for the current Simulated Annealing and Tabu Search Algorithms. Moreover, parallelised Simulated Annealing algorithm and Quantum Inspired Tabu Search can be applied and implemented on 0-1 knapsack problem.

Reference

[1] H. Kellerer, U. Pferschy, and D. Pisinger, Knapsack Problems. New York: Springer-Verlag, 2004.

[2] A. Heinrici, “A comparison between simulated annealing and tabu search with an example from the production planning,” in Operations Research Proceedings 1993, H. Dyckhoff, U. Derigs, and M. Salomon, Eds., Berlin, Germany, 1994, pp. 498–503.

[3] P. J. M. van Laarhoven and E. H. L. Aarts, Simulated Annealing: Theory and Applications. New York, NY, USA: Springer, 1987.

[4] F. Glover and M. Laguna, “Tabu search,” in Handbook of Combinatorial Optimization. New York, NY, USA: Springer, 1998, pp. 2093–2229.

[5] R. Van Slyke and Y. Young. Finite stochastic knapsacks with applications to yield management. Operations Research, vol. 48, pp. 151-172, January 2000.

[6] T. P. Patalia and G. R. Kulkarni, "Design of Genetic Algorithm for Knapsack Problem to Perform Stock Portfolio Selection Using Financial Indicators," 2011 International Conference on Computational Intelligence and Communication Networks, 2011, pp. 289-292, doi: 10.1109/CICN.2011.60.

[7] K. M. Bretthauer, B. Shetty, S. Syam, and R. J. Vokurka, “Production and inventory management under Multiple Resource Constraints,” Mathematical and Computer Modelling, vol. 44, no. 1-2, pp. 85–95, 2006.

[8] J. M. G. Smith and N. Chikhale, “Buffer allocation for a class of nonlinear stochastic Knapsack Problems,” Annals of Operations Research, vol. 58, no. 5, pp. 323–360, 1995.

[9] N. Ferdosian, M. Othman, K. Y. Lun and B. M. Ali, "Optimal solution to the fractional knapsack problem for LTE overload-state scheduling," 2016 IEEE 3rd International Symposium on Telecommunication Technologies (ISTT), 2016, pp. 97-102, doi: 10.1109/ISTT.2016.7918092.

[10] A.Y. Zomaya and R. Kazman, “Simulated Annealing Techniques,” Handbook of Algorithms and Theory of Computation, M.J. Atallah, ed., CRC Press, Boca Raton, Fla., 1999, pp. 37.1–37.19

[11] I. O. Bohachevsky, M. E. Johnson, and M. L. Stein, “Generalized simulated annealing for function optimization,” Technometrics, vol. 28, no. 3, pp. 209–217, 1986.

[12] T. M. Nabhan and A. Y. Zomaya, "A parallel simulated annealing algorithm with low communication overhead," in IEEE Transactions on Parallel and Distributed Systems, vol. 6, no. 12, pp. 1226-1233, Dec. 1995, doi: 10.1109/71.476165.

[13] H.-P. Chiang, Y.-H. Chou, C.-H. Chiu, S.-Y. Kuo, and Y.-M. Huang, ‘‘A quantum-inspired Tabu search algorithm for solving combinatorial optimization problems,’’ Soft Comput., vol. 18, no. 9, pp. 1771–1781, 2014