Optimizing Inventory Management with 1D Bin Packing

Yueming Qu 20513937

May 6, 2025

Declaration: No AI tools were used in the process of programming or writing this report. All work presented here is entirely my own.

1 Introduction and Background

1.1 Problem Statement

This project has solved the one-dimensional packing problem in the context of inventory management. The goal is to develop an algorithm that effectively allocates items of different sizes to fixed-capacity boxes, such as storage units or transportation containers, while minimizing the number of boxes used.

1.2 Context and Relevance

The Bin Packing Problem (BPP) is important in both research and industry. It simulates actual tasks such as container loading, memory allocation and material cutting, where effective packaging reduces costs and waste. As an NP-hard problem, BPP can also be used as a standard for testing optimization algorithms. Solving the BPP problem has not only improved industrial efficiency but also promoted the development of general optimization techniques.

2 Methodology

2.1 Approach and Justification

To solve the Bin Packing Problem, a meta-heuristic method based on Simulated Annealing (SA) was used. Simulated Annealing is suitable for NP-hard combinatorial problems like BPP because it can escape local optima and explore a wide range of solution Spaces. This algorithm iteratively perturbed the item sequence and applies the greedy best fit heuristic algorithm to generate bin allocations. The acceptance of new solutions is controlled by the Metropolis standard which allows for the occasional acceptance of poorer solutions to avoid premature convergence. This method strikes a balance between the quality of the solution and computational efficiency, making it suitable for large-scale instances.

2.2 Implementation Details

The program begins by reading the problem instance from the JSON file. The core algorithm sorts the items in descending order and generates the initial solution by applying the best-fit heuristic. The neighbor solution is generated by perturbing the item sequence using multi-point swaps, block movement or block inversion, and the perturbation intensity depends on the current temperature. Then the program will evaluate multiple candidate solutions at each temperature step and select the best solution based on the number of boxes used. If no improvement is observed in a certain number of iterations, a rapid cooling function will be called. After running all the instances, the results will be saved in a JSON file.

3 Results and Analysis

3.1 Presentation of Results

The algorithm is tested on 10 instances. Table 1 summarizes the number of bin used and the calculation time for each instance.

Table 1: Results of Simulated Annealing on Bin Packing Instances

Instance	**Bins Used**	**Time (s)**
instance_1	52	0.0105
instance_2	59	1.6643
instance_3	24	0.2961
instance_4	27	0.0000
instance_5	47	0.0123
instance_6	49	0.0143
instance_7	37	0.5907
instance_8	52	0.0363
instance_large_9	417	9.3104
instance_large_10	374	63.9945
Total	**1138**	**75.9294**

3.2 Critical Analysis

The results show that the algorithm I used can effectively solve the large and small case packing instances within a reasonable computing time. For small instances, the algorithm finds a solution within milliseconds, while for large instances, the computing time increases but it is still practical. And the number of bins used is closed to the Best-known results.

Compared with methods such as integer programming solvers (such as CPLEX, Gurobi), meta-heuristic methods have significant advantages in scalability and speed, especially for large-scale problems where these solvers are not available. However, unlike these precise methods, Simulated Annealing cannot guarantee optimality. Because of the random nature of the algorithm, the solution may be closed to the optimal one and may vary between runs.

Compared with the sample algorithm which uses simple random search combined with the Next Fit heuristic, the results show that the sample algorithm is faster when dealing with those large instances. This is because the sample algorithm involves minimal computation and does not require iterative optimization. However, the quality of the sample solution is significantly worse because it often produces suboptimal bin allocations due to the lack of a complex search strategy and its inability to explore the solution space.

The advantages of Simulated Annealing are its ability to avoid local optima and its adaptability to different instance sizes. The adaptive neighbor generation and rapid cooling mechanism further enhance the performance. The disadvantage is parameter adjustments (such as temperature planning, perturbation intensity) may affect the results and cost more time.

4 Conclusion and Recommendations

4.1 Summary of Key Findings

The meta-heuristic algorithm based on Simulated Annealing is suitable for solving the Bin Packing Problem. This method has effectively handled both small and large-scale instances, generating high-quality solutions within acceptable computing time. However, due to its randomness, this algorithm does not always guarantee finding the optimal solution. Furthermore, when dealing with very large datasets, the computing time can become very long.

4.2 Future Work and Recommendations

In the future, we can focus on further improving the quality and computational efficiency of the algorithm. We can adopt a hybrid approach, such as combining Simulated annealing with other meta-heuristic methods (like genetic algorithms or tabu search), which can help achieve better solution quality. Furthermore, we can also implement parallel or distributed computing technologies to reduce the computing time of large-scale problems.