Technical Report: Shopping Optimizer Based on the Knapsack Problem

The knapsack problem is an optimization problem where we try to select the best set of items without passing a given limit. In this shopping case, the user enters a budget and a list of item prices. The goal is to choose the largest number of items without going over the budget. Two methods are used to solve the problem: the Greedy method and the Dynamic Programming method.

Greedy Knapsack Method

The greedy method sorts all item prices from cheapest to most expensive. Then it selects items starting from the cheapest and subtracts each price from the remaining budget. The process stops when the budget cannot fit any more items. This method works because picking cheap items usually allows the user to take more items. The time complexity is O(n log n) due to sorting, and selecting items in one pass takes O(n). It is simple and fast but does not check all combinations, so it may miss the best solution.

Dynamic Programming Knapsack Method

The dynamic programming method builds a 2D table. The rows represent how many items have been considered, and the columns represent every possible budget value. For each item, the algorithm checks whether taking it or skipping it leads to a better result. The table stores the best number of items possible at each step. After filling the table, the method backtracks to find which items were taken. This approach explores all valid combinations and always returns the optimal answer. The time complexity is $O(n \times mx)$ and the space complexity is the same because the table size depends on both the number of items and the budget.

Difference Between Greedy and Dynamic Programming

The greedy method picks items based on immediate decisions (cheapest first). It is fast but not always optimal. The dynamic programming method checks all possible solutions and guarantees the best result but uses more time and memory. Greedy has complexity $O(n \log n)$, while dynamic programming has complexity $O(n \times mx)$.

Summary: Best Method for This Case

Under these specific conditions, the Greedy Strategy is the best approach. Because the goal is strictly to maximize the number of items, the strategy of buying the cheapest items first is mathematically guaranteed to provide the correct answer. The Greedy method is faster, uses significantly less memory, and is easier to implement, making it the superior choice for this specific shopping optimization tool.

Github