"Algorithm Design and Analysis"

Project Documentation

Dr. Azimifar

TA Team: *Ali Maher, Amir Hosein Ezati*

Deadline: *will be announced later*

"0/1 Knapsack"

Introduction

The 0/1 Knapsack Problem is a classic optimization challenge involving selecting items with distinct values and weights to maximize the total value within a given capacity constraint. It revolves around choosing the best combination of items to include in a knapsack, considering their individual values and weights while ensuring the overall weight doesn't exceed the knapsack's capacity.

This problem finds widespread applications in various domains, including finance, resource allocation, inventory management, and production scheduling, offering solutions for scenarios where items have different values and sizes, but space is limited.

Project Outlines

Your objective in this project is to tackle the 0/1 Knapsack Problem (KP). In order to attempt that you should suggest **at least three** algorithms based on your course topics that you think they can lead you to the optimal solution.

A number of standard instances will be provided to you so that you can test your methods on them.

Instance Format

The input file format for the 0/1 Knapsack Problem should adhere to the following structure:

```
N
id_1 profit_1 weight_1
id_2 profit_2 weight_2
id_3 profit_3 weight_3
...
id_N profit_N weight_N
W
```

Where:

- `N` is the number of items available to choose from.
- `id_i`, `profit_i`, and `weight_i` denote the identifier, profit, and weight of the i-th item, respectively.
- `W` represents the maximum capacity of the knapsack.

Example:

```
3
1 3 8
2 2 8
3 9 1
10
```

In this instance:

- There are three items available (with IDs 1, 2, and 3).
- Item 1 has a profit of 3 and a weight of 8.
- Item 2 has a profit of 2 and a weight of 8.
- Item 3 has a profit of 9 and a weight of 1.
- The maximum capacity of the knapsack is 10 units.

Remark

- The maximum number of items is 1200.
- The maximum capacity is 10^10 g.

Solution

The solution should be represented as an array containing the IDs of the items used in your answer, separated by dashes. For instance, if items 1, 2, 5, and 8 are used, your solution should be represented as follows: 1 - 2 - 5 - 8.

- **No time limitations:** There's no specified time limit for running the problem.
- **Use learned algorithms:** Stick to the algorithms taught in the course :)
- **Single use of items:** Each item in the knapsack problem can be used only once. Ensure your algorithms consider this constraint.
- **Score/time Evaluation:** Your score will be based on the accuracy of your solution and the time taken to compute it. Make sure your programs accurately calculate the time to optimize both factors.
- **No external libraries:** Avoid using external libraries or pre-built functions (like genetic algorithm libraries) not covered in the course.
- **File Work:** Your program has to read a txt input as input file.

Grading

Hence there are many approaches to attack this problem and many instances with different shapes and structures, your results will be different.

Regarding this, the top 5 results will get the maximum grade and the grades of the rest are given based on them.

- Feel free to ask any questions or seek guidance.
- **No cheating is permitted**, and any discovered cheating will result in a score of zero for the respective assessment.
- Make sure to attend for the presentation.

Best Wishes

Ali Maher / Amir Hosein Ezati