M.Sc. in Mathematics (Finance)

2019/2020, 1st semester

Computational Methods

Project 1 – Solving the Knapsack Problem (KP)

1. Introduction

The Knapsack Problem (KP) is a well-known problem that may be formulated as follows:

> Given a set G of n *items*, each with a value (v_i) and weight (w_i), find a subset of the goods with maximum value (i.e. the sum of the goods in the subset) whose combined weight does not exceed the **capacity** of the knapsack.

Knapsack_1.txt

```
50
it_6 43 316
it_1 12 71
it_9 15 105
it_2 20 140
it_7 18 125
```

2. Objective

Your goal is to get (approximate) solutions of instances of the **KP**, specified in files (as that shown) with the following format (**note:** the field separators are tabs: **'\t'**):

1. The first line indicates the **capacity** of the knapsack (an integer).
2. Each of the other lines specify an item, by a triple $\langle \mathbf{n_i}, \mathbf{w_i}, \mathbf{v_i} \rangle$ where $\mathbf{n_i}$ is an item id (a string), $\mathbf{w_i}$ the weight of the item (an integer) and $\mathbf{v_i}$ the value of the item (also an integer).

In the example, the optimal solution is the subset **K = {'it_1','it_2','it_7'}**, with value **v = 336** (71+140+125) and weight **w = 50** (12 + 20 + 18) that does not exceed the knapsack capacity (it is equal in this case).

3. Implementation Notes

a. Implement a function with signature

```
def knapsack(fname, mode)
```

where, for the knapsack instance stored in a file with name `fname`, returns a list of items' ids that are a (approximate) solution of the problem, together with the sum of the values and the weight of the selected items.

b. Your function should store the items read from the file with `fname`, in a list **S** of triples $\langle \mathbf{n_i}, \mathbf{w_i}, \mathbf{v_i} \rangle$, where $\mathbf{n_i}$ is the item number, $\mathbf{w_i}$ the weight of the item and $\mathbf{v_i}$ the value of the item. **Suggestion**: Sort the list by decreasing values of the ratio $\mathbf{w_i} / \mathbf{v_i}$ (the first items of the list are the best candidates for the knapsack).

c. Then you should fill **K**, the list that encodes the intended subset of **S**, by repeatedly, select an item from **S** (not yet selected), and appending it to **K**, taking into account that the weight of the items in S should not exceed the **capacity**.

d. To select the next node to append to **S** you should implement several different heuristics to select the next item, specified in parameter `mode` (a "_" means the value is not relevant):

1) `mode = (1,_,_)`: Choose the item, among those not yet chosen, that has the higher ration $\mathbf{v_i}/\mathbf{w_i}$ and does not exceed the remaining capacity of the knapsack;
2) `mode = (2, nit,_)`: Choose arbitrarily an item, among those not yet chosen. In this case, repeat `nit` times this procedure, and report the best solution obtained.
3) `mode = (3, nit, nb)`: Select arbitrarily, among the `nb` best items (i.e. those with better $\mathbf{v_i}/\mathbf{w_i}$ ratio), that do not exceed the remaining capacity (or less if this number of remaining items is less than `nb`). Repeat the procedure `nit` times, and report the best solution obtained.
4) `mode = (4, nit, nb)`: Select, among the `nb` best items that do not exceed the remaining capacity (or less if this number of remaining items is less than `nb`), with a probability that is proportional to their $\mathbf{v_i}/\mathbf{w_i}$ ratio. Repeat the procedure `nit` times, and report the best solution obtained.

4. Final Report

Write a small report explaining how you implemented the `knapsack` function, namely the auxiliary functions and data structures that you used. Moreover, report the solutions obtained in instances of the problem, obtained from file `knapsacks.zip`.

The report, as well as the files with your code, must be sent by email to the lecturer (**pb@fct.unl.pt**) with subject **project_mc_1_by_XXXXX+YYYYY** (where XXXXX and YYYYY are the numbers of the students - max 2 per group), **no later** than Friday, 20 December at 23:59.