MIE-PAA. Homework 1:

Solving the knapsack problem by brute force and simple heuristic

September 28, 2013

1 Problem Statement.

The **knapsack** problem or **rucksack** problem is a problem in combinatorial optimization: Given a set of items, each with a mass and a value, determine the number of each item to include in a collection so that the total weight is less than or equal to a given limit and the total value is as large as possible. It derives its name from the problem faced by someone who is constrained by a fixed-size knapsack and must fill it with the most valuable items.(wikipedia.org)

The knapsack problem should be solved in two ways: by a brute force and by the cost/weight ratio greedy heuristic. We are supposed to perform the experiments on sets of 50 instances of one size.

We are given instances of data to do the experiments. They are text files, the files are named *knap n.inst.dat*, where n is the instance size. Each row describes one instance. Each instance is identified by (ID), the number of items (n) and the knapsack capacity (M) follow.

2 Analysis of possible solutions.

Brute force algorithm

Brute force algorithm is approach when we examine all possible combinations. For n-size instance we get 2^n combinations. After we construct a set:
$X = \{x_1, x_2, , x_n\}$, each x_i will be 0 or 1.

We have $\boldsymbol{n}$ - number of items, $\boldsymbol{M}$ - the knapsack capacity and two sets $\boldsymbol{w} = \{w_1, ..., w_n\}$ - that are weights of itmes and $\boldsymbol{c}$= $\{c_1, ..., c_n\}$ - costs of items. And two conditions: the first one garantees that the knapsack is not overloaded, second one give the maximum value of the knapsack.

$\sum_{i=1}^{n} w_i x_i = w_1 x_1 + w_2 x_2 + \ + w_n x_n \leq M$

$\sum_{i=1}^{n} cost_i x_i = cost_1 x_1 + cost_2 x_2 + \ + cost_n x_n$ is max.

Heuristic technique

This technique differs from previous one. In this case we use cost/weight ratio. This rate is computed for each item from the list and after computing sorting is produced by this ratio.

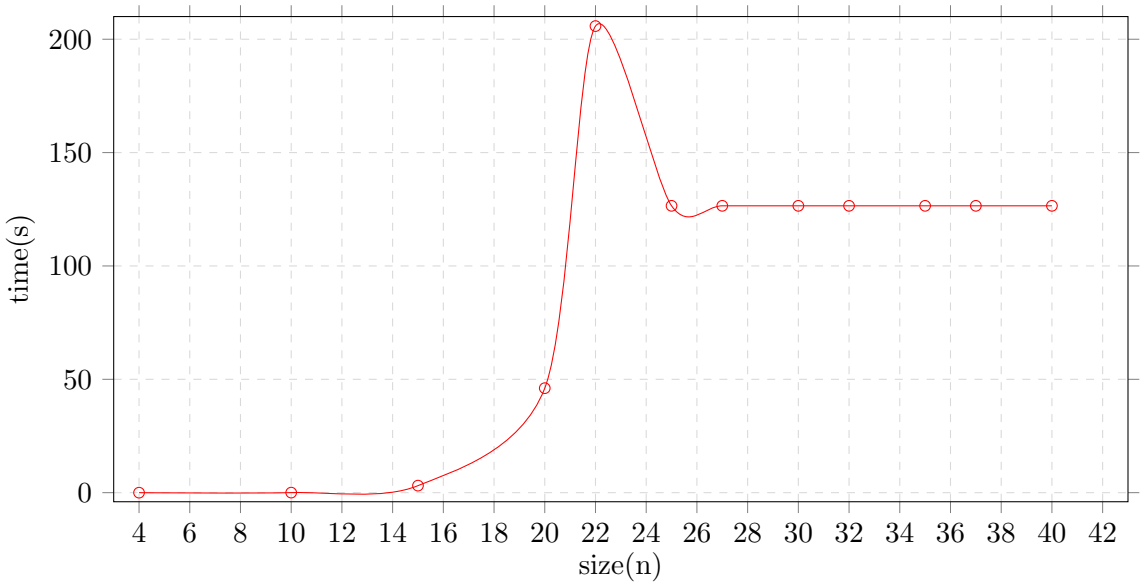
As we do in the tutorials we take items with the highest ratio, until the first item that cannot be inserted is reached due to the fact that size is fixed. After we tried to find the smaller item that can fit.

3 Brief description of solution, description of the algorithms used.

Brute force algorithm

4 The experimental results.

stuff	stuff
stuff	stuff



5 Conclusions.