Project Optimization and regression:

Exact and approximate methods

to solve 0-1 Knapsack problem

Description

The 0/1 Knapsack Problem and Logistics

Transportation companies such as TNT and Royal Mail face daily problems in *logistics*. Consider the following simple logistics problem, which you will solve:

An airline cargo company has 1 aeroplane which it flies from the UK to the US on a daily basis to transport some cargo. In advance of a flight, it receives bids for deliveries from (many) customers. Customers state the weight of the cargo item they would like delivered, and the amount they are prepared to pay. The airline is *constrained* by the total amount of weight the plane is allowed to carry. The company must choose a subset of the packages (bids) to carry in order to make the maximum possible profit, given the weight limit that they must respect.

In mathematical form the problem is: Given a set of N items each with weight w_i and value v_i, for i=1 to N, choose a subset of items (e.g. to carry in a knapsack) so that the total value carried is maximized, and the total weight carried is less than or equal to a given carrying capacity, C.

This kind of problem is known as a 0/1 Knapsack Problem. A Knapsack Problem is any problem that involves packing things into limited space or a limited weight capacity. The problem above is "0/1" because we either do carry an item: "1"; or we don't: "0". Other problems allow that we can take more than 1 or less than 1 (a fraction) of an item. Below is a description of a fractional problem.

1. **Approximate method for the KP**

   Propose your own approximate algorithm to solve a large instance of KP.

   For example:

   The greedy algorithm is very simple. It sorts the items in decreasing value-to-weight ratio. Then it adds them in one by one in that order, skipping over any items that cannot fit in the knapsack, but continuing to add items

that do fit until the last item is considered. There is no backtracking to be done.

Improvements: To do even better, we might have to take out some of the items we have put in and replace them by some other ones.

2. **Exact method for the KP**

   Propose your own algorithm to solve the KP.

   For example: You can implement a simple algorithm to generate all the combinations. Be careful, because you can generate feasible and non-feasible solutions.

3. **Comparison of the obtained results**

   For all instances:
   Compare the quality (given in %) of the solutions obtained by the approximate algorithm and those by the exact algorithm.

4. **Instances (8 instances) and details :**

   https://people.sc.fsu.edu/~jburkardt/datasets/knapsack_01/knapsack_01.html

   Deadline: 28/05/2023