Mid-term Exam 2015

Introduction to Optimization lecture
at Université Paris Sud

Anne Auger and Dimo Brockhoff
firstname.lastname@inria.fr

September/October, 2015

Abstract

This document details the exercises for the mid-term exam of the Introduction to Optimization lecture at Université Paris Sud of 2015. The mid-term exam is published in two parts where the first one on discrete optimization is published on September 25, 2015 and needs to be handed in on Friday, October 16 (before the lecture). The second part on continuous optimization will follow soon.

In order to pass the final exam, 50% of the points of this mid-term exam need to be reached. It furthermore counts 1/3 of the final grade of the lecture.

Please hand in your solutions and in particular the source code by sending an e-mail to one of the lecturers (e-mail above). Use the keyword "mid-term exam" in the subject of the e-mail and do not forget to mention your full name.

1 Part 0: Dynamic Programming for the Knapsack Problem (0 Points)

If not yet finished, please implement the dynamic programming algorithm as well as the brute-force approach of the first exercise and hand in your (tested!) source code.

2 Part I: Branch and Bound for the Knapsack Problem (50 Points)

In the second lecture, we have learned about the branch and bound concept but did not apply it in an exercise. Hence, we will develop a branch and bound algorithm for the knapsack problem here and also implement it. Last, we will compare it with the other approaches you implemented in the first exercise (respectively in part 0 of the mid-term exam).

Questions and Tasks

a) Explain briefly (2-3 sentences) in your own words the idea behind branch and bound. (3 Points)

b) Explain how branch and bound can outperform a brute-force approach. (2 Points)

c) Formalize a first basic branch and bound algorithm for the knapsack problem without specifying the subroutine to find upper bounds (i.e. the basic algorithm structure with the definition of the branching rule and how you would prune parts of the solution tree once the upper bound technique is specified). Write down the algorithm in pseudocode. (10 Points)

d) Implement this basic branch and bound algorithm for the knapsack problem (which, for the moment, contains no pruning and thus is a brute-force enumeration algorithm). (10 Points)

e) Choose one method to compute an upper bound for a sub-problem and describe the idea behind it. (5 Points)

f) Implement the chosen idea for getting the upper bound in your algorithm of task I.d). If you wish, you can use external code (e.g. a scientific library) for this part. (10 Points)

g) Compare the two newly developed algorithms (the simple branch and bound and the version with upper bounds) with the brute-force and dynamic programming algorithms you developed in the first exercise. To this end, run all algorithms on the random instances from `researchers.`

`lille.inria.fr/~brockhof/optimizationSaclay/knapsackinstances/` and plot the runtime to solve the problem over the problem dimension. As to the runtime, please plot the median and at least the variance (better are boxplots) and send the produced plots as well. Write a few sentences about what you observe and explain why you see the differences you observe. (10 Points)