Albert-Ludwigs-Universität, Inst. für Informatik
Prof. Dr. Fabian Kuhn
H. Ghodselahi, Y. Maus

January 30, 2017

Algorithm Theory, Winter Term 2016/17
Problem Set 8

hand in (hard copy or electronically) by 13:55, February 06, 2017, tutorial session will be on February 09, 2017.

Exercise 1: Greedy Approximation for Knapsack (17 points)

In the lecture, we have considered the (0-1)-Knapsack problem: There are n items with positive weights $w_1, \ldots, w_n$ and values $v_1, \ldots, v_n$ and a knapsack (a bag) of capacity W such that $w_i \leqslant W$ for all $1 \leqslant i \leqslant n$. A feasible solution to the problem is a subset of the items such that their total weight does not exceed W. The objective is to find a feasible solution of maximum possible total value.

Consider the following greedy algorithm:

1. Sort the n items such that $\frac{v_1}{w_1} \geqslant \frac{v_2}{w_2} \geqslant \ldots \geqslant \frac{v_n}{w_n}$.
2. Fill the knapsack sequentially with items in the above sorted order starting with the item with largest value per weight. The algorithm stops either if there are no more items left or it reaches an item $k \leqslant n$ which does not fit, i.e., $w_k > W - \sum_{i=1}^{k-1} w_i$.

a) (5 points) Show that the solution of the greedy algorithm can be arbitrarily bad compared to an optimal solution.

b) (12 points) Using a modification to the greedy algorithm, it is possible to get a 2-approximation for the problem. Present such a modified greedy algorithm and show that it provides approximation factor of 2.

   Hint: *For the sake of analyzing your algorithm, you might use the result on fractional knapsack problem (cf. problem set 3, exercise 1, part a).*

Exercise 2: LIFO Paging (8 points)

Either give an explanation if the following statement is true or provide a counter example if it is false.

There exists some constant $c \geqslant 1$ such that the Last In First Out (LIFO) paging algorithm is c-competitive.

Exercise 3: Online Vertex Cover (15+5* points)

Given a graph $G = (V, E)$. A set $S \subseteq V$ is called a *vertex cover* if and only if for every edge $\{u, v\} \in E$ at least one of its endpoints is in S. The minimum vertex cover problem is to find such a set S of minimum size.

We are considering the following online version of the minimum vertex cover problem. Initially, we are given the set of nodes V and an empty vertex cover $S = \emptyset$. Then, the edges appear one-by-one in an online fashion. When a new edge $\{u, v\}$ appears, the algorithm needs to guarantee that the edge is covered (i.e., if this is not already the case, at least one of the two nodes u and v needs to be added to S). Once a node is in S it cannot be removed from S.

a) (15 points) Provide an online algorithm with competitive ratio at most 2. That is, your online algorithm needs to guarantee at all times that the vertex cover S is at most by a factor 2 larger than a current optimal vertex cover. Discuss the correctness of your algorithm.

b) (5 bonus points) Show that there does not exist any online algorithm that can provide a better competitive ratio than 2.