Technische Universität München, Zentrum Mathematik
Lehrstuhl für Angewandte Geometrie und Diskrete Mathematik

Discrete Optimization (MA 3502)

Prof. Dr. Nicole Megow | Dr. Fidaa Abed | Dr. Michael Ritter

Exercise Sheet 7

Problem 7.1 (One Tree Lagrangian Relaxation for TSP)
Let $V = [n]$, $G = (V, E)$ a complete graph and $c : E \longrightarrow \mathbb{R}_{\geq 0}$ a weight functions, which associates to each edge $e \in E$ a cost c_e. Consider the following ILP formulation of the TSP:

$$\min \sum_{e \in E} c_e x_e$$

$$\sum_{e \in E} x_e = n \tag{1}$$

$$\sum_{e \in \delta(1)} x_e = 2 \tag{2}$$

$$\sum_{e \in \delta(i)} x_e = 2 \qquad i \in [n] \setminus 1 \tag{3}$$

$$\sum_{e \in E(S)} x_e \leq |S| - 1 \qquad \emptyset \subset S \subset V \setminus v_1 \tag{4}$$

$$x \in \{0, 1\} \tag{5}$$

a) Use the inequalities (3) to construct a Lagrangian relaxation for this problem. Which well-known combinatorial problem is the relaxed problem similar to, and how can it be solved efficiently?

b) What is the subgradient of the Lagrange function?

c) Use the subgradient method to compute z_{LD} for the weighted graph depicted in Figure 1. Use the steplength $\xi_i = \frac{\sqrt{2}}{i}$ for $i \in \mathbb{N}$ and start with $\lambda^1 = 0$. What does this imply for the optimal solution for this problem?

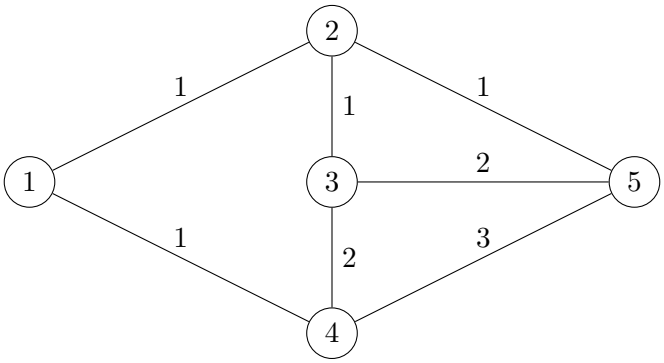

Figure 1: The graph for Problem 7.1

Please turn over.

Problem 7.2 (An Approximation Algorithm for the Knapsack Problem)
Consider an instance of the knapsack problem on $n \in \mathbb{N}$ items given by values $v \in \mathbb{N}^n$, weights $w \in \mathbb{N}^n$ and a bound $b \in \mathbb{N}$, where $w_i \leq b$ for all $i \in \{1, \ldots, n\}$. Let z^* denote the value of a maximum knapsack and z_{greedy} the value of a knapsack obtained through the following algorithm:

Input: An integer n, a vector $v \in \mathbb{N}^n$ of values, a vector $w \in \mathbb{N}^n$ of weights and a bound $b \in \mathbb{N}$ such that $w_i \leq b$ for all $i \in \{1, \ldots, n\}$.
Output: A feasible knapsack solution $I \subset \{1, \ldots, n\}$.
Sort the items such that $\frac{v_1}{w_1} \geq \cdots \geq \frac{v_n}{w_n}$.
Determine $k' := \max\left\{k \in \{1, \ldots, n\} : \sum_{i=1}^{k} w_i \leq b\right\}$.
if $\sum_{i=1}^{k'} v_i > v_{k'+1}$
then
  return $I := \{1, \ldots, k'\}$
else
  return $I := \{k' + 1\}$

a) Show that the algorithm is a 2-approximation for the knapsack problem, i. e. $z_{\text{greedy}} \geq 1/2 \cdot z^*$.

b) Prove that the constant $1/2$ is best possible for this algorithm, i. e. there is no constant $r < 2$ such that $z_{\text{greedy}} \geq 1/r \cdot z^*$. for all possible instances of the knapsack problem.

c) Show that the last step of the algorithm is essential, i. e. prove that by always returning the knapsack $I := \{1, \ldots, k'\}$ the ratio $\frac{z_{\text{greedy}}}{z^*}$ can become arbitrarily small.

Problem 7.3 (An Approximation Algorithm for Load Balancing)
Consider the Load Balancing Problem:

Input: n jobs with processing times $p_1, \ldots, p_n \in \mathbb{N}$, a number $m \in \mathbb{N}$ of processors.

Task: Find an assignment of jobs to machines that minimizes the maximum load over all machines. The load of machine $i \in [m]$ is the sum of processing times of jobs assigned to i.

a) Show that the following simple list scheduling algorithm (LS) is a $(2 - 1/m)$-approximation for Load Balancing.

LS: Consider jobs in an arbitrary, but fixed order. Assign a job to the machine that has currently minimum load.

b) Show that the analysis is tight, i. e. construct an instance of Load Balancing where LS yields a solution that is a factor $(2 - 1/m)$ away from an optimum solution.

c) Can you modify the algorithm to improve upon the approximation ratio?