Matrikelnr.:

Name:

Discrete Optimization

Prof. Dr. Stefan Funke WS 2012/13

Institut für Formale
Methoden der Informatik
Universität Stuttgart

Test Exam WS 2012/13

Note:

- Please **immediately** complete the first page with your name and matriculation number
- Keep the exam sheets closed until told to begin with the exam by an instructor
- Check for completeness of the exam sheets (10 pages)
- Carefully read each question before starting to answer
- Any attempt of cheating leads to immediate failing of the exam
- Please have your ID/passport and student card visible on the table
- Write your name and matriculation number on *each* sheet of paper
- All questions assume notation and methodology as taught in the lecture in the 2012/13 term; you can deviate from that notation/methodology but then you have to explain your solution in a self-contained manner.
- If possible, write the solutions on the exam sheets; if necessary additional sheets of paper are available, please ask for them (and write your name / matriculation number on each of the additional sheets, too)
- This is **not** an openbook exam! You are only allowed to use pen, pencil, ruler, and eraser. No calculator of any kind is allowed.
- **No mobile phones! Making use of a mobile phone is interpreted as cheating!**
- Answers can be given in German or English

Block 1

This part is about (integer) linear programming.

Problem 1-1 A mining company produces 10 tons of red ore and 8 tons of black ore each day. The ores are used to produce three different kinds of alloy, *soft*, *hard*, and *strong* alloy. 1 ton of soft alloy requires 5 tons of red and 3 tons of black ore. The requirements for hard alloy are 3/5, for strong alloy 5/5. The different alloys can be sold for 250/300/400 EUR/ton, respectively.

Set up a linear program which determines the optimal amount of alloys to produce of each kind to maximize the revenue.

Describe in detail the meaning of each variable as well as each constraint.

Problem 1-2 Write down the dual LP for the following primal linear program:

$$\begin{array}{lrrcl}
\min & -35x & -10y & & \\
\text{s.t.} & x & +y & \geq & 1 \\
 & 0.5x & +y & \leq & 7 \\
 & -0.5x & +y & \geq & 0 \\
 & -0.5x & +y & \leq & 5
\end{array}$$

Problem 1-3 Consider the following two-dimensional linear program:

$$\begin{array}{lrrcr}
\max & x & +y & & \\
\text{s.t.} & -x & -y & \leq & -2 \\
 & 0.5x & +y & \leq & 8 \\
 & 0.5x & -y & \leq & 0 \\
 & -0.5x & +y & \leq & 4
\end{array}$$

(a) Draw its feasible region here:

(b) Describe a sequence of steps that the (primal) simplex algorithm might follow when starting at the feasible corner with smallest x-coordinate (you can add this sequence to the drawing).

What is the objective function value of the optimal solution?

What if we require the variables to have *integral* values only; what are the feasible solutions (add them to the drawing)? What is the optimum solution?

(c) Give some (naive) upper bound on the number of steps primal simplex algorithm has to perform before reaching the optimum solution (in terms of the number of constraints n and the number of variables d).

Problem 1-4

Consider a (primal) linear program P in standard form and its dual P_D. Assume P_D has feasible solutions of arbitrarily large objective function value. What does that mean for P?

Block 2

This block of problems is about LP-based approximation algorithms.

Problem 2-1

(a) Define the set cover (SC) problem.

(b) Define the cardinality vertex cover (CVC) problem.

(c) Which of the two problems is harder to solve? Explain.

(d) Describe a greedy algorithm for CVC analogous to the greedy SC algorithm covered in class. What upper bound on the approximation quality of this algorithm can you give (no proof, just a few words)?

Problem 2-2

(a) Give an ILP formulation for the SC problem.

(b) Consider the following approximation strategy for the SC problem:

1. solve the LP relaxation of the above ILP
2. select all vertices which have non-zero value in the optimum LP-relaxation

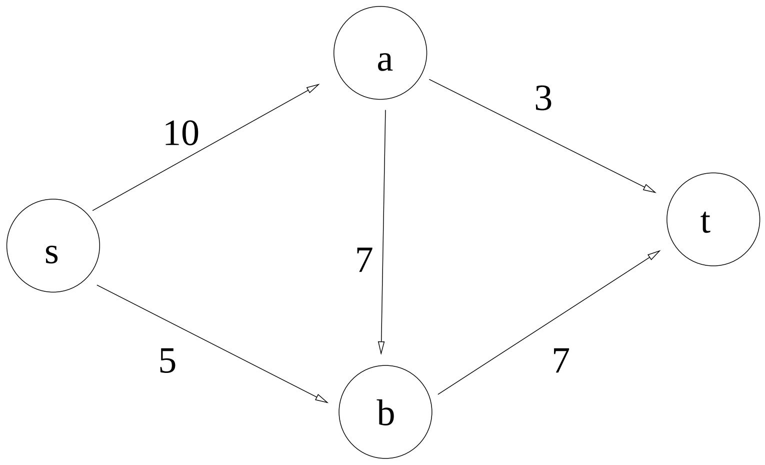
Argue that this strategy leads to a feasible solution and derive an upper bound on the approximation factor achieved.

Block 3

This block is about network flow algorithms as covered in the lecture.

Problem 3-1

(a) Compute the maximum s-t flow in the following network using the Ford-Fulkerson algorithm:



Assume that in each round the s-t path with maximum bottleneck value is used for augmentation; write explicitly the current flow f as well as the residual network after each augmentation.

(b) Argue briefly, why the Ford-Fulkerson algorithm terminates for integral edge capacities.

Block 4

Problem 4-1

a) Determine the value of the most valuable knapsack for a capacity of 60kg and the following items:

Item	1	2	3	4	5	6
Value (EUR)	20	10	30	10	40	0
Weight (kg)	17	12	23	14	25	10

b) Explain how you can actually determine the *content* of the most valuable feasible knapsack once the dynamic programming table has been completed.

c) What is the running time of the dynamic programming solution for knapsack? Why does this running time not contradict the fact that knapsack is an NP-hard problem?

Problem 4-2

(a) Describe an algorithm for computing a 2-approximation for the cardinality vertex cover problem which does not rely on linear programming (neither in the algorithm itself nor in the analysis).

(b) Prove that the algorithm indeed computes a solution at most a factor 2 worse than the optimum solution. Also show that the algorithm does not perform better than that.

Problem 4-3

(a) Define the Precedence Constraint Scheduling (PCS) problem.

(b) Prove that if we knew a polynomial-time algorithm which guarantees an approximation ratio of 5/4 for PCS, P=NP.