Exam Advanced Algorithms. 15-12-2015. Time: 18.30-21.15

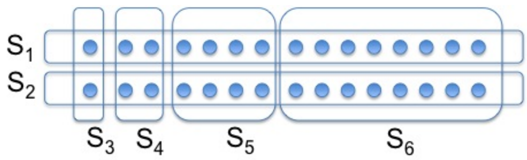
- It is not allowed to use any books, notes, or calculator. Just pen and paper.
- The table shows the maximum number of points per (sub-)question.

1a	1b	2a	2b	3	4	5	6a	6b	7a	7b	8a	8b	9a	9b	9c	9d	Σ
5	5	5	5	10	10	10	5	10	5	10	5	5	2.5	2.5	2.5	2.5	100

1. This question is about the unweighted set cover problem.

(a) Describe the greedy algorithm for unweighted set cover. Now, suppose you apply it to the example below. What are the sets chosen and in what order? What is the ratio between this solution and the optimal solution?

(b) Clearly, one can extend the lower bound by adding more sets. What does this example teach you about the approximation ratio of the greedy set cover algorithm in general? (That means, explain how you extend the example and to what approximation ratio that leads.)



2. (a) Describe a 2-approximation algorithm for the k-center problem. (No proof, just describe.)

(b) Give an instance for which the approximation ratio of the algorithm can be as large as 2. Explain.

3. In the maximum *directed* cut problem we are given as input a directed graph $G = (V, A)$. Each arc $(i, j) \in A$ has nonnegative weight $w_{ij} \geq 0$. The goal is to partition V into two sets U and $W = V \setminus U$ so as to maximize the total weight of the arcs going from U to W (that is, arcs (i, j) with $i \in U$ and $j \in W$). Give a randomized 1/4-approximation algorithm for this problem and prove this.

4. Consider the following minimization problem and the well-known Hamiltonian path problem.

DEGREE BOUNDED SPANNING TREE:

Instance:	Graph $G = (V, E)$
Solution::	A spanning tree T of G
Value:	Maximum degree of T
Goal:	Find a solution with minimum value.

HAMILTONIAN PATH:

Instance:	Graph $G = (V, E)$
Question:	Does G have a path which visits each vertex exactly once?

It is well-known that the Hamiltonian path problem is *NP*-complete. Use this to show that there is no polynomial time α-approximation algorithm for Degree Bounded Spanning Tree for any $\alpha < 3/2$, assuming $P \neq NP$.

5. For the Matlab assignment, you implemented the list scheduling algorithm which is known to be a 2-approximation algorithm for minimizing the length of the schedule on identical machines. Assume now that all jobs are relatively small. Precisely, assume that for each job k, its processing time p_k satisfies

$$p_k \leq \frac{1}{3(m-1)} \sum_{j=1}^{n} p_j,$$

where m is the number of machines and n is the number of jobs. Prove that in this case, list scheduling is a $4/3$-approximation algorithm.

6. Let $G = (V, E)$ be a graph with a non-negative weight w_j for each vertex j.

(a) Give the ILP for weighted vertex cover and give the dual of the LP-relaxation.

(b) Consider the following algorithm for weighted vertex cover.
For each vertex j define a variable y_j and for each edge (i, j) define a cost c_{ij}.

Initially, the vertex cover I is empty. Also, let
$y_j = w_j$ for each vertex $j \in V$ and let
$c_{ij} = 0$ for each edges $(i, j) \in E$.

Now repeat the following until all edges are covered:
Select an arbitrary uncovered edge (i, j) and let $m = \min\{y_i, y_j\}$.
Change the values to: $y_i = y_i - m$, $y_j = y_j - m$, and $c_{ij} = m$.
Add i to I if $y_i = 0$ and add j to I if $y_j = 0$.

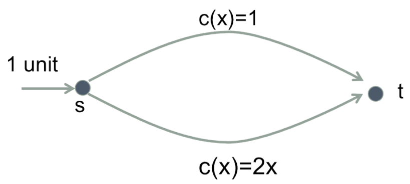
Show that this algorithm is a 2-approximation algorithm for weighted vertex cover.

7. (a) Give the strict quadratic program for the unweighted maximum cut problem and give its vector program relaxation.

(b) Give (draw) an unweighted graph for which the maximum cut has value 4 and for which the optimal value to the vector program relaxation has value strictly larger than 4. (Draw the cut and sketch a solution for the vector program relaxation with value larger than 4.)

8. (a) Define the price of anarchy of a game.

(b) Compute the price of anarchy for the selfish routing problem in the figure. Assume the non-atomic model, that means, a single player has a negligible influence.



9. Remember that a sealed bid auction consists of the following steps.

- Bidding: Each player i makes a sealed bid b_i
- Allocation: The auctioneer chooses the amount x_i that each player i gets.
- Pricing: The auctioneer chooses the price p_i that each player i has to pay.

(a) Describe in a few sentences what it means that a mechanism has the *Dominant Strategy Incentive Compatible (DSIC) property*.

(b) What does it mean when we say that an allocation rule x is *implementable*?

(c) What does it mean when we say that an allocation rule x is *monotone*?

(d) State Meyerson's lemma.

Solutions:

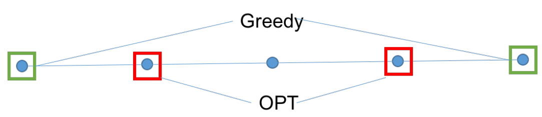
(1) (a) Greedy set cover: In each step, find a set which has the maximum number of uncovered elements and add it to the solution.
Sets chosen (in this order): $6,5,4,3$. Value is 4. Optimal value is 2. Ratio is $4/2 = 2$.

(b) In general, we can extend the example as follows:
$|S_1| = |S_2| = 2^k - 1$, and $|S_3| = 2^1, |S_4| = 2^2, \ldots, |S_{k+2}| = 2^k$.
(In the picture, $k = 4$)
The greedy algorithm chooses sets $k+2, k+1, \ldots, 3$ while the optimum remains 2. The ratio is $k/2$. Note that $n = 2^{k+1} - 2$ so $k = \log_2(n+2) - 1$.

$\Rightarrow$ ratio is $k/2 \approx \log n$.

(2) (a) Take an arbitrary point as first center. Next, in each iteration find a point with maximum distance to the already chosen centers and let that be the next center. Repeat until k centers are chosen.

(b) See figure. If Greedy starts with one of the endpoints than the value is exactly two times OPT.



(3) Assign each vertex with probability 0.5 to either side. Then, for any directed edge (i,j)

$$\Pr((i,j) \text{ in cut}) = \Pr(i \in U \text{ and } j \in W) = \frac{1}{2} \cdot \frac{1}{2} = \frac{1}{4}.$$

The expected total weight of the cut is

$$\sum_{(i,j)\in A} w_{ij} \Pr((i,j) \text{ in cut}) = \frac{1}{4} \sum_{(i,j)\in A} w_{ij} \geq \frac{1}{4}\text{OPT}.$$

(4) Assume that there does exists such an α-approximation algorithm ALG with $\alpha < 3/2$. Now make the following reduction.

Given en instance $G = (V,E)$ of the Hamiltonian Path problem (HP) we take that same graph as instance for the Degree bounded Spanning Tree problem (DBST). Now apply ALG.

$$\begin{array}{rl} \text{If } G \text{ has a HP then } \mathrm{OPT}_{DBST} = 2 & \Rightarrow \textsc{Alg} \leq \alpha \cdot 2 < 3. \\ \text{If } G \text{ has no HP then } \mathrm{OPT}_{DBST} \geq 3 & \Rightarrow \textsc{Alg} \geq 3. \end{array}$$

By looking at the value returned by ALG we can decide in polynomial time wether G has a Hamiltonian Path. This is not possible assuming $P \neq NP$. Hence, no such algorithm can exists for $\alpha < 3/2$.

(5) Let k be the job that finishes last and let s_k be its start time. Then, the length is

$$p_k + s_k \leq p_k + \sum_{j \neq k} \frac{p_j}{m} = \frac{m-1}{m} p_k + \sum_{j=1}^{n} \frac{p_j}{m} \leq \frac{1}{3m} \sum_{j=1}^{n} p_j + \sum_{j=1}^{n} \frac{p_j}{m} = \frac{4}{3m} \sum_{j=1}^{n} p_j \leq \frac{4}{3} OPT.$$

(6)(a)

$$\begin{array}{lll} \text{(LP) min} & Z = \sum_{j=1}^{m} w_j x_j & \\ s.t. & x_i + x_j \geqslant 1 & \text{for all } (i,j) \in E. \\ & x_j \geqslant 0 & \text{for all } j \in V. \end{array}$$

$$\begin{array}{lll} \text{(D) max} & Z = \sum_{i=1}^{n} y_{ij} & \\ s.t. & \sum_{i:(i,j) \in E} y_{ij} \leqslant w_j & \text{for all } j \in V. \\ & y_{ij} \geqslant 0 & \text{for all } (i,j) \in E. \end{array}$$

(b)
By construction of the algorithm, for any vertex j

$$w_j \geq \sum_{i:(i,j) \in E} c_{ij}. \tag{1}$$

Let I^* be an optimal vertex cover. Then (1) implies

$$\mathrm{OPT} = \sum_{j \in I^*} w_j \geq \sum_{j \in I^*} \sum_{i:(i,j) \in E} c_{ij} \geq \sum_{(i,j) \in E} c_{ij}. \tag{2}$$

The inequality follows since each edge is covered by at least one $j \in I^*$ and therefore appears at least once in the summation.

The algorithm selects exactly those vertices j for which the inequality (1) is tight, i.e., for which

$$w_j = \sum_{i:(i,j) \in E} c_{ij}, \text{ for all } j \in I.$$

Therefore, the cost ALG of the algorithm is

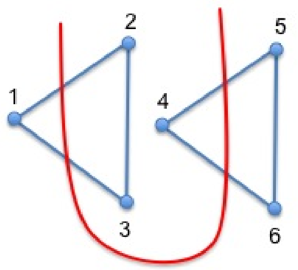
$$\textsc{Alg} = \sum_{j \in I} w_j = \sum_{j \in I} \{ \sum_{i:(i,j) \in E} c_{ij} \} \leq 2 \sum_{(i,j) \in E} c_{ij} \leq 2\textsc{Opt}.$$

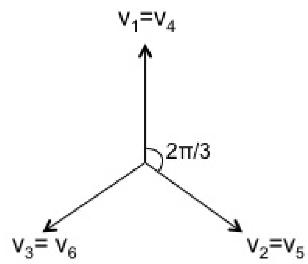
The first inequality follows since each edge appears at most twice in the summation. The second inequality follows from (2).

(7) (a)

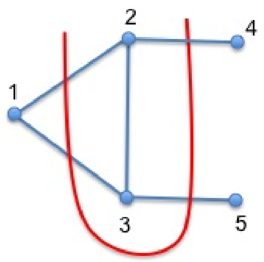
$$\begin{aligned}
\text{(QP)}\quad \max\quad & \tfrac{1}{2}\sum_{(i,j)}(1-y_iy_j)w_{ij} & \\
s.t.\quad & y_i^2 = 1 & i=1,\dots,n. \\
\text{(VP)}\quad \max\quad & \tfrac{1}{2}\sum_{(i,j)}(1-v_i\cdot v_j)w_{ij} & \\
s.t.\quad & v_i\cdot v_i = 1,\ v_i\in\mathbb{R}^n & i=1,\dots,n.
\end{aligned}$$

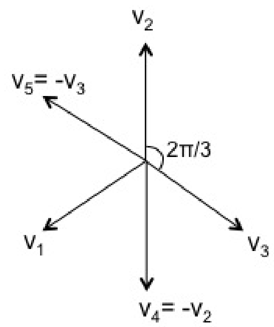
(b) An easy example is to take two triangles. The value of the VP solution in the picture is $\frac{1}{2}6(1--0.5)=4.5$





N.B. The graph above is not connected but that is OK. Here is a possible example with a connected graph: The VP solution in the picture has value $\frac{1}{2}(3(1--0.5)+2(1--1))=4.25$.





(8) (a) POA= Ratio of (worst) equilibrium value and the optimal value.
(b) In the equilibrium $x=0.5$ and the travel time is 1 for each player. For arbitrary x the average travel time is $(1-x)\cdot 1+x\cdot 2x=2x^2-x+1$. The minimum value is $7/8$ and is attained for $x=1/4$. The POA= $8/7$.

(9) (a) Each player has a dominant strategy and that is bidding its true value.
(b) x is implementable if there is a pricing p such that (x,p) has the DSIC

property.
(c) Bidding more does not lead to a smaller allocation.
(d) It is enough here to mention just the first of the 3 parts of the lemma: x is implementable iff it is monotone.