Advanced Algorithms 2017
Exam Questions (Jan. 2018)

Writing:
Motivate all your answers. Answers, even correct ones, that lack motivation do not count.
On the other hand, answer concisely and to the point, without digressions. Do not write wordy essays. The exam questions are designed in such a way that each of them can be answered completely in a few lines.

Submission:
Write your name and ID number on the submission.
Mail your answers to `ptr@chalmers.se` as PDF attachment strictly before the given deadline.
(Alternatively you may hand in on paper, in room EDIT 6478.)
Do not wait until the last few minutes, but submit when you are done.
You may revise your submission arbitrarily often until the deadline, and only the last version will be considered.
If you have any problems to stick to the deadline for an important reason, inform us in good time.
In the case of unforeseen computer system problems, the deadline may be extended if necessary.

Help:
You must do the exam completely on your own. Neither group work nor external help is permitted.
Used literature beyond the course material must be cited.
Questions may be directed to the teachers only.
Mail questions (if you have any) to `ptr@chalmers.se`.
However, no solution hints will be given. Only questions about the interpretation and correct understanding of the exam problems will be answered.
Utmost academic honesty is expected. Cheating can lead to failure on the entire course and further consequences.

Remark: The course web page might contain additional information and updates during exam time.

1. For the greedy algorithm for load balancing (makespan minimization) with n jobs of lengths t_j and m machines we had shown that $m(T - t_j) \leq \sum_{k=1}^{m} T_k$. Here, T was the makespan achieved by the algorithm, T_k was the final load of the kth machine, and j was the index of some job that ends at time T. From this inequality we had derived an approximation ratio of at most 2.

1.1. Explain why the following slightly stronger inequality holds true:
$m(T - t_j) + t_j \leq \sum_{k=1}^{m} T_k$.

1.2. Then, use this inequality to derive the slightly better approximation ratio $2 - 1/m$.

2. A circular disk with radius $2r$ can be covered by 6 circular disks with radius r; in the following you can use this statement without proof.

Suppose that we are given a set S of n points in the plane, and we want to cover all points in S by a minimum number of circular disks of radius 1. Let k be the optimal number of disks needed. Propose a polynomial-time algorithm to produce a covering with at most $6k$ circular disks of radius 1.

3. Consider the project selection problem, however with some modifications: As before, every project has some profit (revenue) p_i, which can be positive or negative. A directed edge (i, j) means that project i has project j as a prerequisite. We assume that all transitive edges exist; that is: Whenever the directed edges (i, j) and (j, k) exist, the directed edge (i, k) must exist as well. However, we make the precedence constraints "soft": For every directed edge (i, j) it is now permitted to select project i and not to select project j, but for every such pair we must pay an amount q_{ij} which is deducted from the total profit. All p_i and q_{ij} are given. The goal is to maximize $\sum_{i \in A} p_i - \sum_{(i,j) \in X} q_{ij}$, where A denotes the set of all selected projects, and X denotes the set of all violated constraints.

Modify the known network for project selection, such that a minimum cut corresponds to an optimal solution to the new problem. Here you don't have to give a complete description and correctness proof. It suffices to say what you change, and give a brief informal explanation.

4. Here is an alternative approximation algorithm for finding a maximum independent set in a graph of maximum node degree Δ. Its approximation ratio is inferior to the already known ratio $1/\Delta$, but it has the advantage of being parallelizable.

Algorithm: Paint every node black with probability $1/\Delta$, and white otherwise, and do this independently for all nodes. Return the set I of nodes v with the property that v is black and all neighbors of v are white.

4.1. Is I always an independent set, or can the algorithm produce an invalid solution? More precisely: How large is the failure probability (of I not being an independent set)?

4.2. What is the probability for any fixed node to be put in I? Motivate your calculation.

4.3. Show that the expected number of nodes in I is at least c/Δ, where c is some constant. (That is, c does not depend on Δ. You don't have to specify c numerically.) Again, give a clear motivation.

4.4. As the last step, prove that the expected approximation ratio of the algorithm is at least c/Δ.

5. From some statistical data we get a set R of n non-negative real numbers, and we want to compute their sum s. But n is huge, and adding all these numbers is expensive and time-consuming. To get a quick estimate we simply pick a number from R at random (that is, each with probability $1/n$). For some small fixed k, we repeat this procedure k times independently and compute the sum t of the k selected numbers. Then we estimate s as tn/k. It would be nice to know whether the estimate from this very simple "algorithm" is good or somehow biased or flawed. Precisely asked:

5.1. Due to the random choices, t is a random variable. Is it true that $E[t \cdot n/k] = s$? Give either a proof or a counterexample. Of course, only one of these options can be correct.

5.2. Consider the special case that the numbers in R are just 0s and 1s. Can we use Chernoff bounds to limit the deviation of our estimate of s from the true value? (You don't have to do any calculation here. Only answer yes or no, and briefly explain your answer.)