Albert-Ludwigs-Universität
Institut für Informatik
Prof. Dr. F. Kuhn

# Exam Algorithm Theory

Monday, February 23, 2015, 09:00-10:30

Name: . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .

Matriculation Nr.: . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .

Signature: . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .

**Do not open or turn until told so by the supervisor!**

## Instructions:

- Write your name and matriculation number on the cover page of the exam and sign the document! Write your name on all sheets!
- Your signature confirms that you have answered all exam questions without any help, and that you have notified exam supervision of any interference.
- Write legibly and only use a pen (ink or ball point). Do **not** use **red**! Do **not** use a pencil!
- You are **not** allowed to use any material except for a dictionary and a hand-written summary of at most 5 A4 pages (corresponds to 5 single-sided A4 sheets!).
- There are 6 problems (with several questions per problem) and there is a total of 90 points. At most 40% are needed to pass the exam, and 80% will net you the best grade, i.e., 18 points are bonus points.
- Use a separate sheet of paper for each of the 6 problems.
- Only one solution per question is graded! Make sure to strike out any solutions that you do not want to be considered!
- **Explain your solutions! Just writing down the end result is not sufficient unless otherwise indicated.**

| Question | Achieved Points | Max Points |
|---|---|---|
| 1 | | 17 |
| 2 | | 18 |
| 3 | | 12 |
| 4 | | 14 |
| 5 | | 12 |
| 6 | | 17 |
| Total | | 90 |

# Problem 1: Short Questions (17 points)

- For the following **two** statements decide whether they are true or false. You do not need to give a proof or counter example.

  (a) (3 points) There are at most $\binom{n}{2}$ $s$-$t$ min-cuts in an $s$-$t$ flow network with $n$ nodes.

  (b) (3 points) *Brent's Theorem* says that for a given parallel computation with total work $T_1$ and span $T_\infty$, no parallel algorithm running on $p$ processors can run faster than $\frac{T_1 - T_\infty}{p} + T_\infty$.

- Solve the following **two** exercises.

  (c) (5 points) The contraction algorithm (for randomized min-cut) always succeeds in finding a min-cut when it is applied to a tree. Give an explanation why this statement is true.

  (d) (6 points) Either give an explanation if the following statement is true or provide a counter example if it is false.

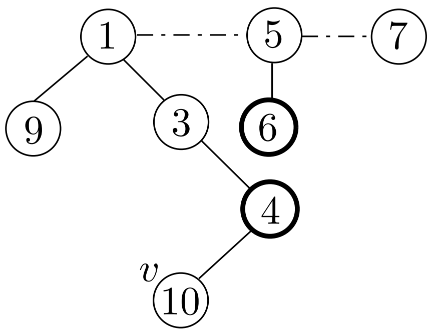
  There exists some $c \geq 1$ such that the Last In First Out (LIFO) paging algorithm is $c$-competitive.

# Problem 2: Heaps (18 points)

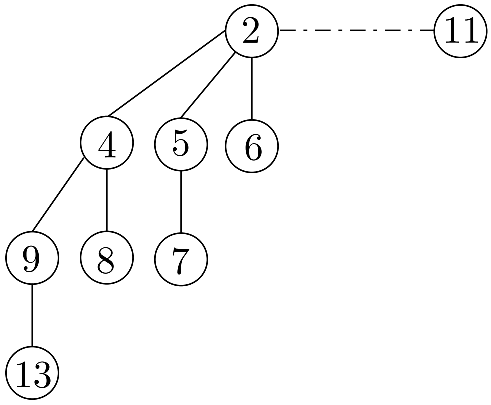
(a) (6 points) Consider the Fibonacci heap in Figure 1a (the thick nodes are marked and the thin ones are unmarked). How does the given Fibonacci heap look after inserting value $8$ and how does it look after a subsequent *decrease-key*$(v, 2)$ operation?

(b) (6 points) Consider the binomial heap in Figure 1b. How does the binomial heap look after inserting values $12$ and $14$ (in that order)? How does it look after a subsequent *delete-min* operation (multiple solutions exist; state one valid solution)?

(c) (6 points) In a sequence of operations $o_1, \ldots, o_n$, let $o_i$ be a *decrease-key* operation. Show that the *decrease-key* operation in a Fibonacci heap has constant amortized cost with the help of the potential function $\Phi = R + 2M$, where $R$ is the number of trees (length of the root list) and $M$ is the number of marked nodes that are not in the root list.



(a) Fibonacci Heap



(b) Binomial Heap

Figure 1: Initial heaps

## Problem 3: Cover all Edges (12 points)

You are given an undirected graph $G = (V, E)$, a capacity function $c : V \to \mathbb{N}$ and a subset $U \subseteq V$ of the nodes. The goal is to cover every edge with the nodes in $U$, where every node $u \in U$ can cover up to $c(u)$ of its incident edges.

Formally, we are interested in the existence of an assignment of the edges to incident nodes in $U$ such that each node $u$ gets assigned at most $c(u)$ of its incident edges.

(a) (10 points) Devise an efficient[1] algorithm to determine whether such an assignment exists *with a given subset* $U$ and a *given* cost function $c$ or not.

(b) (2 points) What is the running time of your algorithm?

## Problem 4: Randomized Max Cut (14 points)

Let $G = (V, E)$ be an undirected graph. Consider the following randomized algorithm: Every node $v \in V$ joins the set $S$ with proability $1/2$. The algorithm's output is the cut $(S, V \setminus S)$. You can assume that $(S, V \setminus S)$ actually is a cut, i.e., $\emptyset \neq S \neq V$.

(a) (10 points) Show that with probability at least $1/3$ this algorithm outputs a cut which is a $4$-approximation to a maximum cut.

**Remark:** *For a non-negative random variable* $X$*, the Markov inequality states that for all* $t > 0$ *we have* $Pr(X \geq t) \leq \frac{E[X]}{t}$.

*If you do not succeed with your choice of a random variable* $X$ *you might try a different one.*

(b) (4 points) How can you use the above algorithm to devise a $4$-approximation of a maximum cut with probability at least $1 - \left(\frac{1}{3}\right)^k$ for $k \in \mathbb{N}$. You do not need to show the success probability of your idea.

**Remark**: *If you could not solve* $a$*), you can still use the result as a black box for solving* $b$*).*

[1]Trying out all possibilities is **not** an efficient algorithm.

## Problem 5: Nearest-Neighbour TSP (12 points)

Given is a symmetric traveling salesperson problem (TSP) instance where all edge weights are either $1$ or $2$. Show that the nearest-neighbour greedy algorithm provides a factor $\frac{3}{2}$ approximation for TSP.

**Remark:** *Write a complete proof to gain full points.*

**Hint:** *You might want to see a TSP tour as a directed cycle.*

## Problem 6: Online Algorithm (17 points)

We consider the following online problem. You have an account starting with value zero. Now, you are consecutively given natural numbers $n_1, n_2, n_3, \ldots \in \mathbb{N}$ one at each time. When receiving $n_t$ you can either add or subtract it from your account under the constraint that your account does not attain a negative value, that is, you are forced to add $n_t$ to your account when the current value of your account is less than $n_t$.

Your goal is to keep the maximum value of your account, which is reached, as small as possible.

One example is:

| Input Numbers | Non Feasible | Feasible Solution | Optimal Solution |
|---|---|---|---|
| $1, 3, 3, 4, 5, 2$ | $1 + 3 - 3 - 4 + 5 - 2$ | $1 + 3 + 3 - 4 + 5 - 2$ | $1 + 3 - 3 + 4 - 5 + 2$ |
| | | (maximum=8) | (maximum=5) |

(a) (10 points) Design a deterministic online algorithm which solves the problem with a competitive ratio of 2. Prove that your algorithm is 2-competitive.

(b) (7 points) Show that there is no deterministic online algorithm with a competitive ratio smaller than $1.5$.

**Remark:** *(Two) sequences of four numbers each (with up to three different values) are sufficient to show this. Partial points are handed out if you can prove the claim for a smaller competitive ratio $\mu \in (1, 1.5)$.*