ETH zürich

Institute of Theoretical Computer Science
Rasmus Kyng, Bernd Gärtner, David Steurer, Vera Traub

Algorithms, Probability, and Computing   Midterm Exam   HS25

Candidate

First name: ....

Last name: ....

Student ID (Legi) Nr.: ....

I attest with my signature that I was able to take the exam under regular conditions and that I have read and understood the general remarks below.

Signature: ....

General remarks and instructions

1. Check your exam documents for completeness (8 two-sided pages with 5 exercises).
2. You have 2 hours to solve the exercises.
3. You can solve the exercises in any order. You should not be worried if you cannot solve all exercises! Not all points are required to get the best grade.
4. Immediately inform an assistant in case you are not able to take the exam under regular conditions. Later complaints will not be accepted.
5. Pencils are not allowed. Pencil-written solutions will not be reviewed.
6. No auxiliary material allowed. All electronic devices must be turned off and are not allowed to be on your desk. We will write the current time on the blackboard every 15 minutes.
7. Attempts to cheat lead to immediate exclusion from the exam and can have judicial consequences.
8. Provide only one solution to each exercise. Please clearly mark/scratch the solutions that should not be graded.
9. All solutions must be understandable and well-founded. Write down the important thoughts in clear sentences and keywords. Unless stated otherwise, no points will be awarded for unfounded or incomprehensible solutions.
10. You may use anything that has been introduced and proved in the lecture or in the exercise sessions. You do not need to re-prove it, but you need to state it clearly and concretely. However, if you need something *different* than what we have shown, you must write a new proof or at least list all necessary changes.
11. Write your student-ID (Legi-number) on all sheets (and your name only on this cover sheet).

	achieved points (maximum)	reviewer's signature
1	(10)	
2	(10)	
3	(10)	
4	(10)	
5	(10)	
Σ	(50)	

Exercise 1: Short Questions (10 points)

No justification is required. There are no negative points for wrong answers.

(a) (2 points)

- Draw the binary search tree with the smallest possible height containing the keys 1, 2, 3, 4, 5, 6, 7.
  Answer:

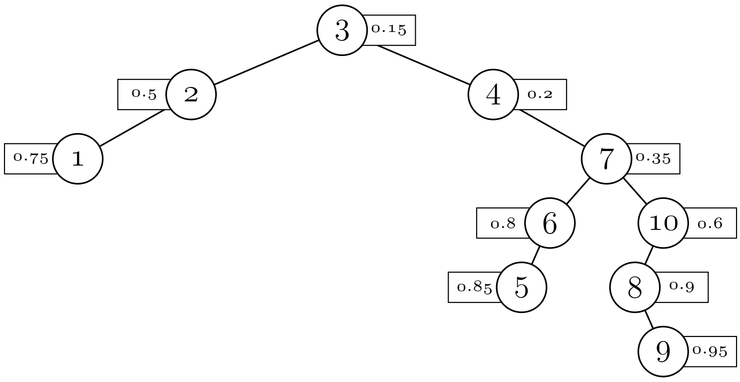
- Give an insertions sequence of the keys 1, 2, 3, 4, 5, 6, 7 that leads to the tree with smallest possible height as above.
  Answer:

(b) (2 points) Let $l_n = \frac{n+1}{3}$ be the expected number of leaves of a random binary search tree with $n \geq 3$ elements as discussed in the lecture.

For general trees, one often defines a leaf as a degree one vertex. What is the expected number of degree one vertices of a random binary search tree with $n \geq 3$ elements?

(c) (1 points)

Consider the treap shown below. We remove node 10. How many rotations are performed during the removal of node 10?



Answer: ....

(d) (2 points) There are n skiers competing at the Lauberhorn ski race. Skier number i will take t_i seconds to complete the course. The order in which the skiers start is chosen uniformly at random and the next skier starts once the previous skier finished. Suppose that throughout the competition we maintain a scoreboard that displays the 3 fastest skiers up to that point. What is the expected number of times the scoreboard will change throughout the competition. You can express your answer using $O(\cdot)$ notation.

Answer: ....

(e) (3 points) Suppose that a linear program P has a feasible solution x. Let D be the dual program of P. Then

- The optimal value of D may be infinity. [ ] False [ ] True
- D always has a feasible solution. [ ] False [ ] True
- D always has the same number of constraints as P. [ ] False [ ] True

Exercise 2: Biased Random Search Trees Continued (10 points)

In special assignment 1, we considered binary search trees for which the probability of inserting the i-th element was proportional to 2^{-i} and derived an expression for l_n, the expected number of leaves in such a tree. We derived

$$l_n = \begin{cases} 0 & \text{if } n = 0 \\ 1 & \text{if } n = 1 \\ \frac{1}{(1-2^{-n})} \sum_{i=1}^{n} \frac{1}{2^i} \left(l_{i-1} + l_{n-i}\right) & \text{otherwise.} \end{cases}$$

In the special assignment, we did not ask you to solve the recurrence. In this exercise, we will do so step by step. For $n \geq 3$, we have

$$\begin{aligned} (1-2^{-n})l_n - (1-2^{-(n-1)})l_{n-1} &= \sum_{i=1}^{n} \frac{1}{2^i} \left(l_{i-1} + l_{n-i}\right) - \sum_{i=1}^{n-1} \frac{1}{2^i} \left(l_{i-1} + l_{n-1-i}\right) \\ &= \left(1 + 2^{-n}\right) l_{n-1} - \underbrace{\sum_{i=1}^{n-1} \frac{1}{2^i} l_{n-i}}_{=:S_{n-i}} . \end{aligned}$$

a) Obtain an expression for $S_n := \sum_{i=1}^{n} \frac{1}{2^i} l_{n+1-i}$ that only involves l_n and l_{n+1} for $n \geq 2$.

b) Prove that $S_n = \frac{1}{2} l_n + \frac{1}{2} S_{n-1}$ for $n \geq 2$

c) Obtain an expression of l_n in terms of l_{n-1} and l_{n-2} for $n \geq 3$.

   Hint: First apply b), and then a).

d) Use the expression you derived in c) to prove that $l_n = a \cdot n + b$ for some $a, b \in \mathbb{R}$ for $n \geq 3$.

Exercise 3: Geometry (10 points)

In the lecture, we often assume that a set of points is in general position, meaning that no 3 points lie on a common line.

a) Describe a simple algorithm that verifies that a set of n points in two dimensions is in general position in time $O(n^3)$.

b) Improve the algorithm to run in time $O(n^2 \log n)$.

Exercise 4: Linear Programming (10 points)

In this exercise, we are interested in certain Linear Programs. Let $A \in \mathbb{R}^{n\times m}$, $b \in \mathbb{R}^n$ and consider the Linear Program

$$\begin{aligned} &\max \alpha \\ \text{subject to} \quad & Ax = \alpha b, \\ & -1 \le x_i \le 1 \quad \forall i = 1, 2, \ldots, m \end{aligned} \tag{P1}$$

Assume that (P1) is feasible with optimal value $1 \le \mathrm{OPT}_1 \le m$. For this exercise, we define $\|x\|_\infty := \max_{i=1,2,\ldots,m} |x_i|$.

a) Assume that you are given an oracle that, given any value $\alpha \in \mathbb{R}$ as input, tells you whether or not a vector $x \in \mathbb{R}^m$ that satisfies $Ax = \alpha b$ and $-1 \le x_i \le 1$ for all $i = 1, 2, \ldots, m$ exists. Show that for any $1 > \epsilon > 0$, using $O(\log(m/\epsilon))$ calls to the oracle suffice to find a vector $x \in \mathbb{R}^m$ satisfying $Ax = (\mathrm{OPT}_1 - \epsilon)b$ and $\|x\|_\infty \le 1$.

b) Consider now the following optimization problem P2 with optimal value OPT_2:

$$\begin{aligned} &\min_{x \in \mathbb{R}^m} \|x\|_\infty \\ \text{subject to} \quad & Ax = b \end{aligned} \tag{P2}$$

Suppose we are given an optimal solution $x_1^\star$ to linear program P1. Give a linear time algorithm that constructs an optimal solution $x_2^\star$ to problem P2. Also show the reverse: Given an optimal solution $x_2^\star$ to P2, construct an optimal solution $x_1^\star$ to problem P1 in linear time.

c) Show that P2 is actually a Linear Program. More precisely, show that there exists a vector $c \in \mathbb{R}^{m+1}$, a matrix $M \in \mathbb{R}^{(2m+2n)\times(m+1)}$ and a vector $d \in \mathbb{R}^{2m+2n}$ s.t. for the linear program

$$\begin{aligned} &\min_{y \in \mathbb{R}^{m+1}} c^T y \\ \text{subject to} \quad & My \le d \end{aligned} \tag{P3}$$

we have that its optimal value OPT_3 satisfies $\mathrm{OPT}_3 = \mathrm{OPT}_2$.

You cannot use the value OPT_2 in your solution.

Exercise 5: MST (10 points)

Let $G_1 = (V, E_1, w_1)$ and $G_2 = (V, E_2, w_2)$ be two connected, weighted graphs on the same vertex set V. Assume that $E_1 \cap E_2 = \emptyset$, i.e., that G_1 and G_2 share no edges. Assume further that both w_1 and w_2 are injective, which ensures that unique MSTs T_1 and T_2 of G_1 and G_2 exist.

Consider now the graph $G = G_1 \cup G_2$, i.e., G is a graph on vertex set V and edge set $E_1 \cup E_2$, s.t. $w(e) = w_1(e)$ if $e \in E_1$ and $w(e) = w_2(e)$ if $e \in E_2$. Assume that this new weight function w is also injective. Prove that the unique MST T of G satisfies that $E_T \subset E_{T_1} \cup E_{T_2}$.

Hint: Use that a spanning tree T is an MST if and only if every non-tree edge is T-heavy. Recall that an edge $e \in E \setminus E_T$ is T-heavy if and only if the unique cycle C of the graph $T \cup e$ satisfies that $\max_{e' \in C} w(e') = w(e)$.