ETH
Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

Institute of Theoretical Computer Science
Bernd Gärtner, Mohsen Ghaffari, Angelika Steger, David Steurer

Algorithms, Probability, and Computing — Midterm Exam — HS19

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 (14 one-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	(16)	
2	(12)	
3	(12)	
4	(12)	
5	(8)	
Σ	(60)	

Exercise 1: Six Short Questions (16 points)

No justification is required. Only give final exact answers.
(There are no negative points for wrong answers.)

(a) For the Voronoi diagram of n points in the plane, build a point-location data structure for the Voronoi diagram using the vertical trapezoidal decomposition. What are the storage space and query time for the data structure? (2 points)

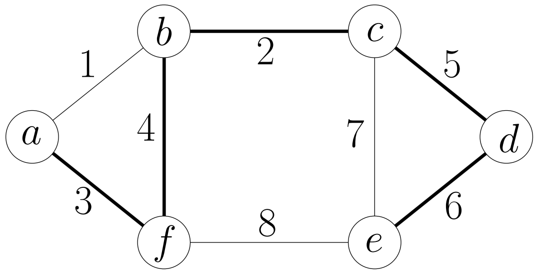
Expected Storage space: ....

Expected Query time: ....

(b) For a linear program, what is the necessary condition that there is at least one feasible solution, but none of the feasible solutions is optimal? (2 points)

Answer: ....

(c) The following figure shows a graph $G(V, E)$ and a spanning tree T (in bold) for G. Among the remaining three edges, what are the T-heavy edges? (2 points)



Answer: ....

(d) The number of vertices in the arrangement formed by m hyperplanes in $\mathbb{R}^n$ is $\mathcal{O}(m^n)$, and the number of vertices in a convex polyhedron formed by m half-spaces in $\mathbb{R}^n$ is $\mathcal{O}(m^{\lfloor \frac{n}{2} \rfloor})$. What is the number of basic feasible solutions for a linear program consisting of m constraints with respect to n variables? (2 points)

Answer: ....

(e) Solve the following recursive formula: (4 points)

$$a_n = \begin{cases} 1, & \text{if } n = 1, \text{ and} \\ \frac{2}{n}\sum_{i=1}^{n-1} a_i, & \text{otherwise.} \end{cases}$$

For $n \geq 2$, $a_n =$ ____

(f) Change the following linear program into the equational form: (4 points)

$$\begin{array}{ll} \text{Maximize} & x_1 + x_2 \\ \text{subject to} & x_1 + 6x_2 \leq 15 \\ & 4x_1 - x_2 \leq 10. \end{array}$$

Answer:

Exercise 2: Random Search Tree (12 points)

For a random binary search tree of n distinct keys, we are interested in the distance between the smallest key and the largest key, i.e., the number of edges along the path between the two corresponding nodes. Let $R(n)$ denote the random variable for such distance, and let node i denote the node holding the key of rank i.

Calculate $\mathbb{E}[R(n)]$ in the following three steps.

Part (a). (4 points) Formulate $\mathbb{E}[R(n) \mid \text{root is node } k]$ in terms of $\mathbb{E}[D_n^{(i)}]$ where $D_n^{(i)}$ is the random variable for the depth of node i as in the script. (A formal proof is NOT required.)

Part (b). (4 points) Give a formula for $\mathbb{E}[R(n)]$ based on (a).

Part (c). (4 points) Provided that $\mathbb{E}[D_n^{(1)}] = \mathcal{O}(\log n)$, prove that $\mathbb{E}[R(n)] = \mathcal{O}(\log n)$.

Exercise 3: Point Location (12 points)

Consider a set S of n line segments in the plane, and let k be the total number of crossings between line segments in S. The vertical trapezoidal decomposition for S is defined analogously as in the lecture for which vertical rays are extended from both endpoints and crossing points, and the decomposition has $\Theta(n+k)$ trapezoids. Assume that no more than two line segments cross at the same point, and no two of endpoints and crossing points share the same x-coordinate.

Prove the following two statements:

Part (a). (6 points) If we pick an r-element *random* subset R of S, the vertical trapezoidal decomposition for the r line segments in R has $\Theta(r + \frac{r^2}{n^2}k)$ trapezoids in expectation.
(*Hint*: Think about the probability that a crossing point between two fixed line segments appears in the r randomly picked line segments.)

Part (b). (6 points) If we construct the vertical trapezoidal decomposition by inserting line segments in a random order, the expected total number of trapezoids that appear during this randomized incremental construction is $\Theta(n + k)$.
(*Hint*: You can use the fact that a trapezoid is defined by $\Theta(1)$ line segments without proving this fact.)

Exercise 4: Linear Programming (12 points)

Consider the following linear program (formulated as a geometric problem):

> Given a set H of n half-planes in the xy-plane, compute the lowest vertex of the convex polyhedron that is the intersection among all half-planes in H.

To ensure the existence of the lowest vertex, we add one more half-plane $h_0 : y \geq 0$. There exist at most two lowest vertices, but we only need to compute one.

Investigate the following randomized incremental algorithm:

1. Generate a *random* permutation $(h_1, h_2, \ldots, h_n)$ of half-planes in H, and let l_i denote the boundary line of h_i for $0 \leq i \leq n$.
2. Assign v to be the intersection point between l_0 and l_1.
3. For $2 \leq i \leq n$,
   if $v \notin h_i$, then update v to be a lowest vertex of the convex polyhedron $\bigcap_{j=0}^{i} h_j$.
4. return v.

Assume that all the half-planes are **upper** (i.e., contains the point $(0, \infty)$), and assume that no boundary line is vertical, each pair of boundary lines intersect exactly at a point, and no more than two boundary lines have a common intersection point.

Complete the following two tasks:

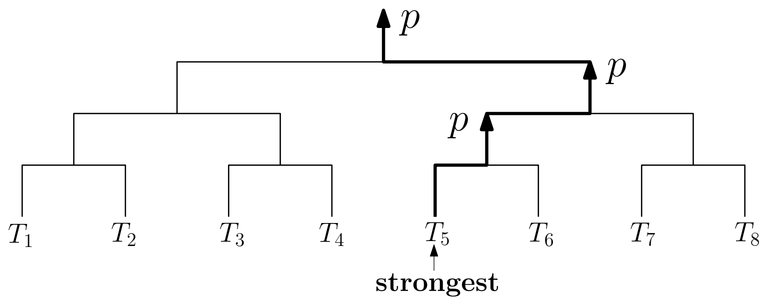
Part (a). (6 points) Prove that if $v \notin h_i$, then one lowest vertex of the convex polyhedron $\bigcap_{j=0}^{i} h_j$ is an intersection point between l_i and a boundary line in $l_0, l_1, \ldots, l_{i-1}$.

Part (b). (6 points) Analyze the probability that $v \notin h_i$ and prove that the expected running time is $\mathcal{O}(n)$. You can assume that (a) is correct and the update in Step 3 takes $\mathcal{O}(i)$ time.
(*Note:* If you are only able to prove that the expected running time is $\mathcal{O}(n \log n)$, you can still get 4 points.)

Exercise 5: Bootstrapping (8 points)

The rough concept of the bootstrapping technique is to repeatedly perform the same algorithm to reduce the running time or to increase the success probability. For the latter, a simple, practical example is sport games. Many competitions play a series of games between two teams and select the one that wins more games as the winner.

Consider a knockout tournament for n teams with $\log_2 n$ rounds. In the first round, each team is paired with another team to play one game, only the winner of each game advances to the second round, and the second round repeats the same procedure. In other words, for $1 \leq i \leq \log_2 n$, the i-th round pairs $\frac{n}{2^{i-1}}$ teams, each pair of teams play one game, and only the winners advances to the next round. The figure below shows a tournament for 8 teams. Assume that there is a unique strongest team, the strongest team wins each game independently with a fixed probability $p \geq 2/3$, and n is a power of two.



Complete the following three tasks:

Part (a). (3 points) Analyze the probability in terms of p that the strongest team is the champion.

Part (b). (3 points) We define a *match* to be a series of $2k+1$ games, and define that a team *wins* a match if it wins at least $k+1$ games in the match. Now, we replace each game in the tournament with a match. Prove that for some $k \in \Theta(\log n)$, the probability that the strongest team wins a match is at least $1 - \frac{1}{n^2}$.

(*Hint*: According to the Chernoff bound, the probability that the strongest team loses a match is at most $\exp(-\frac{1}{2}\frac{((2k+1)p-k)^2}{(2k+1)p})$.)

Part (c). (2 points) Apply part (b) to prove that the strongest team is the champion with probability at least $1 - \frac{\log_2 n}{n^2} \geq 1 - \frac{1}{n}$.