Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

Bernd Gärtner, Mohsen Ghaffari, Rasmus Kyng, David Steurer

Algorithms, Probability, and Computing — Final Exam — HS20

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: ....

Instructions

1. **The exam consists of 7 exercises. However, we grade only 5 out of the 7 exercises. In particular, you can solve either exercise 3A or exercise 3B and you can solve either exercise 5A or exercise 5B. For each elective exercise, we provide a box that you need to tick if you want us to grade this exercise. If you do not tick a box, we will grade the first exercise.**
2. **Each exercise is marked with 1, 2 or 3 stars, according to our perceived difficulty.**
3. Check your exam documents for completeness (20 one-sided pages with 7 exercises).
4. You have **3 hours** to solve the exercises.
5. 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. You may use the stars to help you start with easier problems.
6. If you're unable to take the exam under regular conditions, immediately inform an assistant.
7. **Pencils** are not allowed. Pencil-written solutions will not be reviewed.
8. No auxiliary material is allowed. Electronic devices must be turned off and should not be on your desk. We will write the current time on the blackboard every 15 minutes.
9. Provide only one solution to each exercise. Please clearly mark/scratch the solutions that should not be graded.
10. **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.**
11. You may use anything that has been introduced and proven 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.
12. Write your student-ID (**Legi-number**) on **all** sheets (and your name only on this cover sheet).

	achieved points (maximum)	reviewer's signature
1	(20)	
2	(20)	
3A or 3B	(20)	
4	(25)	
5A or 5B	(15)	
Σ	(100)	

Exercise 1: 2-approximate Minimum Cut (20 points)

Let G denote a connected, undirected and simple n-node graph with $n \geq 2$. We define $G_0 := G$ and for $i \in \{1, 2, \ldots, n-2\}$, we obtain G_i from G_{i-1} by contracting an edge chosen uniformly at random from the edge set of G_{i-1} and then removing all self-loops. The degree of a vertex is defined to be the number of edges the vertex is incident to. Let δ_i denote the minimum degree of the multigraph G_i and k denote the minimum cut size of G.

(a) **(6 points)** ($\star$) Prove that $\min_{i\in\{0,1,\ldots,n-2\}} \delta_i \geq k$.

(b) **(7 points)** ($\star$) Fix an arbitrary $i \in \{0, 1, \ldots, n-3\}$. Let k_i denote the minimum cut size of G_i and assume that $\delta_i \geq 2k_i$. Moreover, view G_i as a fixed graph, i.e., we have fixed the randomness for the first i contractions. Now, we consider the random edge that we choose to contract in the $(i+1)^{\text{th}}$ step. Prove that, over the choice of the single random edge, with probability at least $\frac{n-i-1}{n-i}$, the graphs G_i and G_{i+1} have the same minimum cut size. Formally, this means proving that $\forall H$, we have $\Pr[k_i = k_{i+1} | G_i = H] \geq \frac{n-i-1}{n-i}$.

(c) **(7 points)** ($\star\star$) Using part (b), prove that $\min_{i\in\{0,1,\ldots,n-2\}} \delta_i \leq 2k$ holds with probability $\Omega(1/n)$.

Exercise 2: Gambling (20 points)

After losing your barkeeper job due to the Corona crisis, you start looking for an alternative stream of income. Having studied hard for the final exam, nothing comes more natural in your life than probabilistic analysis. Hence, you use your new skills and start with online gambling.

Your goal is to win n francs in total. The total money you earned after day i is denoted by X_i. In the beginning, you start with $X_0 = 0$ francs in your pocket. During the i-th day, you win an additional Y_i francs where $Y_i \in_{u.a.r.} \{0, 1, \dots, n - X_{i-1}\}$. Hence, $X_i = X_{i-1} + Y_i$. Note that you never lose money, as expected from such a bright mind. Let N_n denote the number of days until you have a total of n francs, that is, N_n is equal to the smallest $i \in \mathbb{N}$ such that $X_i = n$. Moreover, we define $t_n := \mathbb{E}[N_n]$.

(a) **(10 points)** ($\star$) Prove that for $n \geq 1$, we have $t_n = \frac{n+1}{n} + \frac{1}{n}\sum_{i=0}^{n-1} t_i$.

(b) **(10 points)** ($\star$) Prove that $t_n \leq H_n + c$ for every $n \geq 0$ and some constant c independent of n. Recall that $H_n := \sum_{i=1}^{n} \frac{1}{i}$.

Exercise 3A: Randomized Line Arrangement ($\star$) (20 points)

Tick the box if you want this exercise to be graded. $\square$
You should mark exactly one of exercises $3A$ **and** $3B$.

Let L be a set of n lines in $\mathbb{R}^2$. We assume that no two lines are parallel, that no three lines intersect in a common point, and that the distances between the pairwise intersection points of these lines are distinct. Moreover, let $\ell_1, \ell_2, \ldots, \ell_n$ be a permutation of L chosen uniformly at random, and suppose that $\ell_1, \ldots, \ell_n$ are inserted one by one into an initially empty line arrangement. Compute (exactly) the expected number of insertions which change the length of the shortest edge in the current line arrangement (where we count the insertions of the first two lines ℓ_1 and ℓ_2 as changes).

Exercise 3B: Checking Equality ($\star$) (20 points)

Tick the box if you want this exercise to be graded. $\square$
You should mark exactly one of exercises $3A$ **and** $3B$.

Assume that Alice and Bob each have an n-bit binary number. Alice would like to know whether they both have the same number or not. Thus, she asks Bob to send her a message consisting of at most $O(\log n)$ bits. Afterwards, Alice says "Yes" if she believes that they both have the same number and "No" otherwise. Derive a randomized strategy that allows Bob to send a message consisting of $O(\log n)$ bits with the following property. If Alice and Bob have the same number, then Alice always says "Yes". If Alice and Bob have a different number, then Alice says "No" with probability $1 - O(1/n^5)$.

Hint 1: Let N denote some integer. You can use without a proof that the number of distinct prime divisors of N is $O(\log |N|)$.
Hint 2: For each $k \in \mathbb{N}$, let $h(k)$ denote the number of primes less than k. You can use without a proof that $h(k) = \Omega(k/\log(k))$.

Exercise 4: Linear Programing (25 points)

Let P denote an $n \times n$ matrix. We assume that all entries in P are nonnegative, i.e., $P_{ij} \geq 0$ for all $i, j \in \{1, 2, \ldots, n\}$, and each column sums up to 1, i.e., $\sum_{i=1}^{n} P_{ij} = 1$ for every $j \in \{1, 2, \ldots, n\}$. The goal of this exercise is to prove the existence of an n-dimensional vector π with all entries of π being nonnegative and all entries in π summing up to 1 such that $P\pi = \pi$. Intuitively, P_{ij} can be interpreted as the probability that a token sitting at the i-th vertex in an n-node graph moves in one step to vertex j.

(a) **(8 points)** ($\star$) Devise a Linear Program with the property that $\pi \in \mathbb{R}^n$ is a feasible solution if and only if π fulfills the conditions stated above. As we only care about feasibility, you can take $\min 0$ as the objective function of your LP.

(b) **(8 points)** ($\star\star$) Write down the dual of the Linear Program devised in the previous subtask. You can get the full number of points without formally proving that your LP is indeed the dual of the LP devised in the previous subtask, though it might help you to get partial points.

(c) **(5 points)** ($\star\star\star$) Prove that the dual is feasible and bounded.

(d) **(4 points)** ($\star$) Explain why the dual of the LP being feasible and bounded implies that the primal LP is feasible.

Exercise 5A: Isolation Lemma ($\star\star$) (15 points)

Tick the box if you want this exercise to be graded. $\square$
You should mark exactly one of exercises $5A$ **and** $5B$.

Let $G = (V, E)$ be a connected, undirected and simple n-node graph with m edges. Under any weight function $w\colon E \to \{1, \dots, W\}$, the length of a path in G is the sum of the weights of the edges in that path. A weight function is said to be good if the following two conditions hold for each vertex $x \in V$.

1. For all vertices $y \in V \setminus \{x\}$, the shortest path from x to y is unique.

2. For any two distinct vertices $y, z \in V \setminus \{x\}$, the shortest path length from x to y is different from the shortest path length from x to z.

Show that there exists a good weight function with $W = O(n^c)$ for some fixed positive constant c that you are free to chose.
Hint: You might need to use a variation of the Isolation Lemma. In that case, you only need to discuss the necessary changes in the statement and the argument.

Exercise 5B: Parallel Minimum with Advice ($\star\star$) (15 points)

Tick the box if you want this exercise to be graded. $\square$
You should mark exactly one of exercises $5A$ **and** $5B$.

You are given as input an array $A[1..n]$ of n integers. In addition, you receive an arbitrary value x with the guarantee that x is at least as large as the smallest element in A and no larger than the $n^{1/10}$-th smallest element in the array. Devise a parallel algorithm (in CRCW PRAM, in case of concurrent writes, an arbitrary write takes effect) that computes the value of the smallest element in the array with probability $1 - O(1/n^5)$. Your algorithm should have $O(1)$ depth and use $O(n)$ total work. For simplicity, you can assume that all memory locations are set to 0 in the beginning and all the elements in A are at least 1. You get partial credit in case your algorithm succeeds with probability $1 - O(1/n^c)$ for some constant $c \in (0, 5)$.