ETH
Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

Institute of Theoretical Computer Science
Bernd Gärtner, Rasmus Kyng, Angelika Steger, David Steurer, Emo Welzl

Algorithms, Probability, and Computing — Final Exam — HS23

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 5 exercises.
2. Check your exam documents for completeness (18 one-sided pages with 5 exercises).
3. You have **3 hours** to solve the exercises.
4. 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.
5. If you're unable to take the exam under regular conditions, immediately inform an assistant.
6. **Pencils** are not allowed. Pencil-written solutions will not be reviewed.
7. 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.
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 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.
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	(20)	
2	(20)	
3	(20)	
4	(20)	
5	(20)	
Σ	(100)	

Exercise 1: Modified Min-Cut Algorithm (20 points)

Let $c \in \mathbb{N}$ be a constant. For a graph $G = (V, E)$ with $n \geq 2c$ vertices and m edges we call a partition (S, T) of the vertices a c-*cut* if $|S| \geq c$ and $|T| \geq c$. The size of a c-*cut* is defined as the number of edges that have one endpoint in S and the other one in T. In this task we aim to develop a polynomial algorithm that finds the size of the minimum c-*cut* using the edge-contraction subroutine. Consider the following algorithm.

BasicC-Cut(G, c):
Initialize $size(v) \leftarrow 1$ for all $v \in V(G)$
While G has more than $2c$ vertices:
  Pick an edge $(u, v) \in E(G)$ uniformly at random
  $G \leftarrow G/(u, v)$
  Let w be the new vertex of G obtained by the contraction of (u, v)
  Set $size(w) \leftarrow size(u) + size(v)$
end while
Let $\mathcal{C} \leftarrow \{S \subset V(G) : \sum_{v \in S} size(v) \geq c \wedge \sum_{v \in V(G) \setminus S} size(v) \geq c\}$
Find the minimum cut $S, V(G) \setminus S$ for $S \in \mathcal{C}$ using a brute force method
return the cut $S, V(G) \setminus S$

(a) Show that the algorithm returns a cut that that has at least the size of the minimum c-*cut* of the original graph G.

(b) Assume we know that the size of the minimum c-*cut* is k. **First**, compute a lower bound on m. **Afterwards**, use this bound to show that the probability of the first contraction not changing the size of the minimum c-*cut* is at least $1 - \frac{2c}{n}$.

(c) Find a lower bound on the probability that the modified subroutine returns the minimum size of a c-*cut*.

(d) Now, we execute N independent runs of the modified edge-contraction subroutine and return the minimum value encountered in all the runs. Compute N such that the size of the minimum c-*cut* gets returned with probability at least $1 - \frac{1}{n}$. Note that N should be polynomial in n.

Exercise 2: Lines and Rectangles (20 points)

You are given a set of n lines in the plane. A query consists of the coordinates of the vertices of an axis-parallel rectangle R and you are asked to return the number of lines intersecting R. Design a data structure that can answer the query in expected $O(\log n)$ time and uses expected $O(n^2)$ space.
For simplicity, assume that no line is vertical or horizontal, and also that no vertex of R is on one of the given lines.

Exercise 3: Linear Program Duality (20 points)

Consider the following linear program.

$$\min 3a + 9b + 15c + 6d$$

such that

$$\begin{cases} 2a + b - c - d \leq -2 \\ a - 3b - 3c \leq -3 \\ a \geq 0 \\ b \geq 0 \\ c \geq 0 \\ d \geq 0. \end{cases}$$

(a) Write the dual of the linear program.

(b) Draw the feasible region of the dual in the plane and find an optimal solution for it.

(c) Find the optimal value of the solution of the linear program above. Motivate your answer.

Exercise 4: Matrix identity testing & Pfaffian orientations (20 points)

(a) Let A and B be $n \times n$ *symmetric* matrices, i.e. $A^\top = A$ and $B^\top = B$. Define the multivariate polynomials $p(x) = x^\top Ax$ and $q(x) = x^\top Bx$ with variables x in the domain $\mathbb{R}^n$.
Prove that $p(x) = q(x)$ if and only if $A = B$.

(b) Let $p > 2$ be a fixed prime number. Alice holds a symmetric $n \times n$ matrix A with entries in $\{0, 1, \ldots p-1\}$ and Bob holds a symmetric $n \times n$ matrix B with entries in $\{0, 1, \ldots p-1\}$. Suppose that Alice and Bob both have access to a shared random vector v uniformly sampled from $\{0, 1, \ldots, p-1\}^n$. Alice and Bob want to test whether A is equal to B but they can exchange only one message that consists of a number in $\{0, 1, \ldots, p-1\}$. Devise a protocol that allows Bob to check whether $A = B$ after getting only one message from Alice. The protocol should return always *yes* if $A = B$ and *no* with probability at least $\frac{p-2}{p}$ if $A \neq B$.
Hint: Use (a).

(c) The following graph G contains some directed and some undirected edges. Give an orientation of the undirected edges in G to obtain a Pfaffian orientation of G.

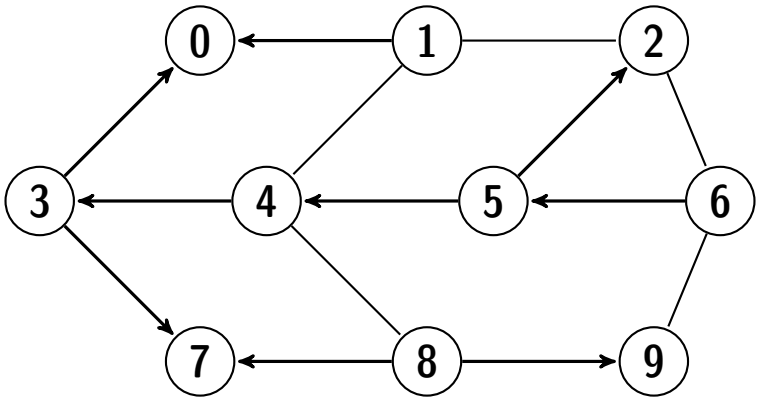

Figure 1: The graph G.

Exercise 5: A Parallel Algorithm for Global Min-Cut (20 points)

Recall the BasicMinCut(G) algorithm for global min-cut. Given a simple connected graph $G = (V, E)$, it repeatedly contracts a random remaining edge until only two vertices are left. Alternatively, one *contraction pass* can be described in the following way, which is equivalent to the BasicMinCut(G) algorithm.

1. Randomly permute the edge set E yielding a permutation $\pi : [m] \mapsto E$.
2. Let $i \leftarrow 1$. While there are more than two vertices left, contract edge $\pi(i)$ if its endpoints have not been merged and increment $i \leftarrow i + 1$.
3. Count and return the number of edges between the two left over vertices.

In chapter 1 of the lecture notes it is shown that BasicMinCut(G) succeeds in finding the global min-cut with probability at least $2/n^2$, and therefore also $O(n^2 \log n)$ contraction passes suffice to find the global min-cut with high probability. In this exercise, we turn the above contraction pass version into a parallel algorithm in the ARBITRARY CRCW model.

a) Argue that a contraction pass is equivalent to BasicMinCut(G).

b) Assume that you are given a graph $G = (V, E)$ and a set $S \subseteq E$. Devise a parallel algorithm that decides if there are exactly 2 vertices left after contracting all the edges in S. If so, it should return the number of edges between the two leftover vertices. Otherwise, it should return ONE if there is just one vertex, and MANY if there are more than 2 vertices left.

   The algorithm should have depth $O(\log n)$ and work $O(m \log n)$.

You may assume that you can sample a random permutation of the edge set with $O(\log n)$ *depth and* $O(m)$ *work for both exercise c) and d).*

c) Given the algorithm developed in b), show that one contraction pass can be implemented with depth $O(\log n)$ and work $O(m^2 \log n)$.

   Hint: Guess the number of edges that need to be contracted until there are only 2 vertices left and execute all guesses in parallel.

d) Devise an alternative implementation of a contraction pass that merely needs $O(m \log^2 n)$ work, but pays $O(\log^2 n)$ depth instead. Conclude that a min-cut can be found with high probability in depth $O(\log^2 n)$ and work $O(n^2 m \log^3 n)$.