ETH zürich

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

Algorithms, Probability, and Computing **Special Assignment 1** **HS23**

- Write an exposition of your solution using a computer, where we strongly recommend to use LaTeX. **We do not grade hand-written solutions**.
- You need to submit your solution via Moodle until **November 1st** by **2 pm**. Late solutions will not be graded.
- For geometric drawings that can easily be integrated into LaTeX documents, we recommend the drawing editor IPE, retrievable at `http://ipe.otfried.org` in source code and as an executable for Windows.
- Write short, simple, and precise sentences.
- This is a theory course, which means: if an exercise does not explicitly say "you do not need to prove your answer" or "justify intuitively", then a formal proof is **always** required. You can of course refer in your solutions to the lecture notes and to the exercises, if a result you need has already been proved there.
- We would like to stress that the ETH Disciplinary Code applies to this special assignment as it constitutes part of your final grade. The only exception we make to the Code is that we encourage you to verbally discuss the tasks with your colleagues. However, you need to write down the names of all your collaborators at the beginning of the writeup. It is strictly prohibited to share any (hand)written or electronic (partial) solutions with any of your colleagues. We are obligated to inform the Rector of any violations of the Code.
- There will be two special assignments this semester. Both of them will be graded and the average grade will contribute 20% to your final grade.
- As with all exercises, the material of the special assignments is relevant for the (midterm and final) exams.

Exercise 1

40 **points**

(*Weighted Minimum Cut*)
The aim of this exercise is to prove that the MinCut algorithm that we have seen in the lecture can find the minimum cut in a weighted graph.

Given a (multi)-graph $G = (V, E)$ with n vertices and m (multi-)edges and with weights $w : E \mapsto \mathbb{R}$ on the multi-edges, define the weight of a cut $C \subseteq E$ as $w(C) = \sum_{e \in C} w(e)$. A *minimum weighted* cut is a cut with minimum weight. For simplicity we denote the weight of the minimum weighted cut in G as $\mu(G)$.

Consider the algorithm WeightedMinCut(G) reported below. The line "collapse parallel edges of G", substitutes all the set of parallel edges $e_1, e_2, \ldots, e_i$ between two vertices u and v with only one edge between (u, v) with weight $w(e_1) + w(e_2) + \cdots + w(e_i)$. Note that this operation turns the weighted (multi)graph into a weighted graph.

(a) Consider the contraction of a random multi-edge $e \in E(G)$ with probability $\frac{w(e)}{\sum_{i \in E(G)} w(i)}$. Prove that probability of $\mu(G) = \mu(G/e)$ for a randomly chosen edge e $E(G)$ is at least $1 - 2/n$. Deduce that the algorithm WeightedMinCut(G) is correct, i.e. with high probability the cut returned is the minimum cut. Note that $\mu(G/e)$ may contain multi-edges even if G does not.

(b) Suppose to have access to a black-box method SampleEdge(G) that returns a random multi-edge $e \in E(G)$ with probability $\frac{w(e)}{\sum_{i \in E(G)} w(i)}$. Argue that WeightedRandomContract(G, t) can be implemented using $O(n)$ calls to SampleEdge(G).

(c) Suppose that $W = \text{poly}(n)$ and that you can sample a number from $\{1, 2, \ldots, \text{poly}(n)\}$ in $O(1)$ time. Assume initially that $m = O(n^2)$ (there is at most a quadratic number of multi-edges). We want to design the method SampleEdge(G) that given the weight $w(e) \in \{1, 2, \ldots, W\}$ for each edge e allow you to sample from the weighted distribution.

- Design a data structure that can be built in $O(m)$ time and $O(m)$ space and allows you to sample from the set of all the edges $E = \{1, 2, \ldots, m\}$ with probability $\frac{w_e}{\sum_{i \in E} w(i)}$ where $w_i \in \{1, 2, \ldots, W\}$ in $O(\log m)$ time.
- Now suppose that some of the edges have been removed and others have been collapsed. Let $S \subseteq E$, such that $\sum_{i \in S} w(i) \geq \frac{\sum_{i \in E} w(i)}{2}$, be the set of edges that have not been remove and $T \subset 2^S$ be a partition of S. Devise a way to use sample a set $t \in T$ with probability $\frac{\sum_{j \in t} w(j)}{\sum_{i \in S} w(i)}$ using a constant number of calls to the data structure from the previous point in expectation. (You can assume given a $i \in S$ you can get the set $t \in T$ that contains i in constant time.)
- Explain why, building $O(\log n)$ instances of the data structure you developed in expectation, you can get an implementation SampleEdge(G) needed in point (b) with an *amortized* query time of $O(\log n)$. (The amortized query time is the sum of the time needed by the queries made during the execution of WeightedMinCut divided by the number of queries.)

(d) Let $\epsilon > 0$. Now suppose that you have access WeightedMinCut but you can only use it for graphs with weights in the range $\{1, 2, \ldots, \lceil 10n/\epsilon \rceil\}$.
Let G be a graph with weights in the range $\{1, 2, \ldots, W\}$. Design an algorithm that uses at most $\log(nW)$ call to WeightedMinCut and finds a cut C of graph G such that $w(C) \leq (1+\epsilon)\mu(G)$.

WeightedRandomContract(G, t):
While $V(G) > t$ do
  Sample $e \in E(G)$ with probability $\frac{w(e)}{\sum_{i \in E(G)} w(i)}$
  $G \leftarrow G/e$
End while
collapse parallel edges of G
Return G

WeightedMinCut(G, k):
If $n \leq 16$ then
  Compute $\mu(G)$ deterministically
  Return $\mu(G)$
else
  $t \leftarrow \lceil n/\sqrt{2} \rceil + 1$
  $H_1 \leftarrow$ WeightedRandomContract(G, t)
  $H_2 \leftarrow$ WeightedRandomContract(G, t)
  Return
$\min($WeightedMinCut$(H_1, k),$ WeightedMinCut$(H_2, k))$

Exercise 2

20 **points**

(*Nodes at distance 3 in Random search trees*)

Let $n \in \mathbb{N}$. Denote with $W(n, d)$ the expected number of nodes of depth d in a random search tree for n keys.

(a) Compute $W(n, 0)$ and $W(n, 1)$ for $n \geq 1$.

(b) Let $d \geq 2$. Write $W(n, d)$ as a function of $W(n-1, d)$ and $W(n-1, d-1)$.

(c) Prove that

$$W(n, 2) = \begin{cases} 0 & n \in \{1, 2\}, \\ 4 - 4\frac{H_{n-1}}{n} - \frac{4}{n} & n > 2. \end{cases}$$

Exercise 3

20 **points**

(*Ancestor with biggest rank*)

Recall that a node u is an ancestor of node v in a rooted tree if u lies on the unique path from v to the root on the tree. Note that v is an ancestor of itself. Furthermore, given two nodes u, v in a tree there is a unique path that connects these two nodes, we call the number of edges in this path the distance between u and v. Given a random search tree T, compute the expectation of the sum of the distances from each point to its ancestor with the biggest rank.

Exercise 4

20 **points**

(*Nearest point or segment in the square*)
We call a tiling of the unit square $[0,1] \times [0,1]$ a partition of the square into convex polygons such that the interiors of no two polygons intersect and the union of all the polygons covers the entire surface of the square.
You are given a tiling of the unit square together with a collection of points inside the square. Suppose that the tiling is given by collection of all the edges of the polygons and let n be the sum of the number of edges in the tiling and the number of points given. Design a data structure that given a query point $p \in [0,1] \times [0,1]$, returns the closest point or segment.
In order to get full score, your data structure should take $O(n)$ space and expected $O(\log n)$ query time, while the preprocessing time has to be polynomial in n. You can assume that the points and the lines supporting all edges are in general position.