Albert-Ludwigs-Universität, Institut für Informatik
Prof. Dr. Fabian Kuhn
M. Ahmadi, P. Schneider, J. Uitto

January 18, 2018

Algorithm Theory - Winter Term 2017/2018
Exercise Sheet 6

Hand in by Thursday 10:15, January 25, 2017

Exercise 1: Trinary Tree (2+5+3 Points)

Consider a complete, full trinary tree with n leaves, i.e. all leaves have the same distance (= height) ℓ from the root and all nodes except for the leaves have three children. Each leave is given a boolean value. According to the values of the leaves we can determine values of the inner nodes. The value of an inner node is defined as the majority value of its three direct children. The objective is to compute the value of the root. The performance of an algorithm to solve this problem is measured by the number of leaves that need to be read by the algorithm.

(a) Is there a deterministic algorithm that can determine the value of the root, such that for any given input, the algorithm does not need to read the values of all leaves? Explain your answer.

(b) Design a recursive, randomized algorithm to determine the value of the root with certainty but reading only a proportion of q^ℓ leaves in expectation (for $q < 1$).

(c) Based on the result of (b) give a tight upper bound (in the number of leaves n) for the expected number of leaves that are read by the algorithm.

Exercise 2: Compare Polynomials (2+3+5 Points)

You are given two polynomials p and q of degree n. However, you are not given the polynomials in an explicit form. You can only evaluate them at some value $x \in \{1, \dots, 2n\}$ (i.e., you can compute $p(x)$ and $q(x)$ for values $x \in \{1, \dots, 2n\}$). You want to find out whether the two polynomials are identical.

(a) Pick $x \in \{1, \dots, 2n\}$ uniformly at random. What is the minimum probability that $p(x) \neq q(x)$ in case $p \neq q$?

(b) Give an efficient[1] randomized *Monte Carlo* algorithm that tests whether the two polynomials are identical in $\mathcal{O}(\log \frac{1}{\varepsilon})$. In specific, if $p = q$, your algorithm should always return "yes", if $p \neq q$, your algorithm is allowed to err with probability at most $\varepsilon \in (0, 1)$.

(c) Prove the properties of your algorithm.

Exercise 3: Max Cut (1+2+4+3 Points)

Let $G = (V, E)$ with $n = |V|, m = |E|$ be an undirected, unweighted graph. Consider the following randomized algorithm: Every node $v \in V$ joins the set S with probability $\frac{1}{2}$. The output is $(S, V \setminus S)$.

(a) What is the probability to actually obtain a cut?

[1] Measured by the number of polynomial evaluations it needs to perform.

(b) For $e \in E$ let random variable $X_e = 1$ if e crosses the cut, and $X_e = 0$, else. Let $X = \sum_{e \in E} X_e$. Compute the expectation $\mathbb{E}[X]$ of X.

(c) Show that with probability at least 1/3 this algorithm outputs a cut which is a $\frac{1}{4}$-approximation to a maximum cut (i.e. a cut of maximum possible size is at most 4 times as large).

Remark: For a non-negative random variable X, the Markov inequality states that for all $t > 0$ we have $\mathbb{P}(X \geq t) \leq \frac{\mathbb{E}[X]}{t}$.

Hint: Apply the Markov inequality to the number of edges **not** *crossing the cut.*

(d) Show how to use the above algorithm to obtain a $\frac{1}{4}$-approximation of a maximum cut with probability at least $1-\left(\frac{2}{3}\right)^k$ for $k \in \mathbb{N}$.

Remark: If you did not succeed in (c) an exercise you can use the result as a black box for (d).

Exercise 4: Graph Connectivity (2+4+4 Points)

Let $G=(V,E)$ be a graph with n nodes and edge connectivity[2] $\lambda \geq \frac{16 \ln n}{\varepsilon^2}$ (where $0<\varepsilon<1$). Now every edge of G is removed with probability $\frac{1}{2}$. We want to show that the resulting graph $G'=(V,E')$ has connectivity $\lambda' \geq \frac{\lambda}{2}(1-\varepsilon)$ with probability at least $1-\frac{1}{n}$. This exercise will guide you to this result.

Remark: If you don't succeed in a step you can use the result as a black box for the next step.

(a) Assume you have a cut of G with size $k \geq \lambda$. Show that the probability that the same cut in G' has size *strictly smaller* than $\frac{k}{2}(1-\varepsilon)$ is at most $e^{-\frac{\varepsilon^2 k}{4}}$.

(b) Let $k \geq \lambda$ be fixed. Show that the probability that at least one cut of G with size k becomes a cut of size *strictly smaller* than $\frac{k}{2}(1-\varepsilon)$ in G' is at most $e^{-\frac{\varepsilon^2 k}{8}}$.

Hint: You can use that for every $\alpha \geq 1$, the number of cuts of size at most $\alpha\lambda$ is at most $n^{2\alpha}$.

(c) Show that the probability that at least one cut of G with *any* size $k \geq \lambda$ becomes a cut of size *strictly smaller* than $\frac{k}{2}(1-\varepsilon)$ in G', is at most $\frac{1}{n}$.

Hint: Use another union bound.

[2] The connectivity of a graph is the size of the smallest cut $(S, V \setminus S)$ in G.