CS-E3190 Principles of Algorithmic Techniques

06. Randomized Algorithms – Graded Exercise

Please read the following **rules** very carefully.

- Do not consciously search for the solution on the internet.
- You are allowed to discuss the problems with your classmates but you should **write the solutions yourself**.
- Be aware that **if plagiarism is suspected**, you could be asked to have an interview with teaching staff.
- Each week the second exercise is an **individual exercise**, and the teaching staff will not give hints or help with them. You are allowed to ask for hints for the first exercise.
- In order to ease grading, we want the solution of each problem and subproblem to start on a **new page**. If this requirement is not met, **points will be reduced**.

1. **Randomized leader election.** (7p.) In computer networks, a frequent problem is how to choose one of the computers to be the *leader*, ie. a computer that will have a particular role in the network (eg. the root of a spanning tree can be considered to be the leader of the graph).

   We assume the network is a clique of size n where the nodes are the computers and the edges are communication links. The computers are identical to each other, but they are allowed to use randomness.

   Consider the following randomized algorithm that runs in parallel on each computer at the same time:

   - Each computer picks a value $v \in \{1, \dots, n\}$ independently and uniformly at random and sends v to all other computers.
   - After receiving the values of all computers, if exactly one computer has chosen the value 1, it is chosen as the leader and we stop.
   - Otherwise, repeat.

   (a) (1p.) Show that the probability that the algorithm terminates in the first iteration is $p = \left(1 - \frac{1}{n}\right)^{n-1}$. Notice that $p \geq \frac{1}{e^2}$.

   (b) (1p.) What is the probability that no leader is picked by kth iteration?

   (c) (2p.) What is the expected runtime of the algorithm when $n = 10$?
   Hint: look up geometric random variables to get to the end of the computation.

   (d) (3p.) Show that the algorithm runs in $O(\log n)$ time with high probability.
   Hint: use question (b).

2. **Individual exercise: Stronger partitioning.** (3p.) In the Tutorial Exercise 2 we showed that we can partition the vertex set of a tree $T = (V, E)$ into two sets V_1 and V_2, such that each connected component of $T[V_1]$ and $T[V_2]$ has diameter $O(\log n)$ with high probability.

   Suppose we wanted a similar partition, only with diameter $O(\log \log n)$. More formally, we want to partition the vertex set of a tree $T = (V, E)$ into two sets V_1 and V_2, such that each connected component of $T[V_1]$ and $T[V_2]$ has diameter $O(\log \log n)$ with high probability. If we were to use the same algorithm and the same analysis as in the Tutorial Exercise 2, where would we fail?