Exercise Set VI, Advanced Algorithms 2025

These exercises are for your own benefit. Feel free to collaborate and share your answers with other students. Solve as many problems as you can and ask for help if you get stuck for too long. Problems marked * are more difficult but also more fun :).

These problems are taken from various sources at EPFL and on the Internet, too numerous to cite individually.

1 In class we saw that Karger's min-cut algorithm implies that an undirected graph has at most $\binom{n}{2}$ minimum cuts. Show that this result is tight by giving a graph with n vertices and $\binom{n}{2}$ minimum cuts.

Solution: Consider the graph consisting of a single cycle with n vertices and thus n edges. Removing any two edges result in a minimum cut. There are $\binom{n}{2}$ ways of selecting the two edges to remove and hence Karger's result is tight.

2 Change Karger's algorithm so that it also works for edge-weighted graphs. Also adapt the analysis to prove that it still returns any min cut $(S^*, \overline{S^*})$ with probability at least $1/\binom{n}{2}$. (Hence, edge-weighted graphs also have at most $\binom{n}{2}$ min cuts.)

Solution: Instead of selecting the edge to contract in each iteration uniformly at random. Now select an edge proportional to its weight w_e.

To show that this idea indeed makes sense, we observe that Claim 1 from the notes of Lecture 11 still holds: the probability that we select an edge in $E(S^*, \overline{S^*})$ to contract is at most $2/n$.

Indeed, let $k = \sum_{e \in E(S^*, \overline{S^*})} w(e)$ be the weight of the min cut and $w(E) = \sum_{e \in E} w(e)$ denotes the total weight of the edges. The probability that we select an edge in the min cut $E(S^*, \overline{S^*})$ is

$$P[e \in E(S^*, \overline{S^*})] = \frac{k}{w(E)}$$

Now similar to the hand-shake lemma we have

$$\sum_{v \in V} w(\delta(v)) = 2 \cdot w(E)$$

where $\delta(v)$ denotes the edges adjacent to v and $w(\delta(v))$ is the weight of $\delta(v)$. We also have that

$$w(\delta(v)) \geq k$$

since k is the weight of the mincut. Therefore,

$$\sum_{v \in V} w(\delta(v)) = 2 \cdot w(E) \geq k \cdot n \Rightarrow w(E) \geq k \cdot n/2\,.$$

This means

$$P[e \in E(S^*, \overline{S^*})] = \frac{k}{w(E)} \leq \frac{k}{nk/2} = \frac{2}{n}$$

Hence, we have that the probability to contract an edge in $E(S^*, \overline{S^*})$ is at most $2/n$.

The analysis now continues in the exact same manner as in the unweighted case. We observe that even in a weighted graph, when we contract an edge (u, v) the size of the minimum cut does not decrease. Then, let, A_i be the event that the edge picked in step i of the loop is not in $E(S^*, \overline{S^*})$. We need to lower bound $P[A_1, A_2, \ldots, A_{n-2}]$. By Bayes rule we have,

$$P[A_1, \ldots, A_{n-2}] = P[A_1]P[A_2|A_1]P[A_3|A_1, A_2] \ldots P[A_{n-2}|A_1, A_2, \ldots, A_{n-3}].$$

From the above, we have that, for all i,

$$P[A_i|A_1, \ldots, A_{i-1}] \geq 1 - \frac{2}{n-i+1}.$$

Therefore,

$$\begin{aligned} P[A_1, \ldots, A_{n-2}] &\geq \left(1 - \frac{2}{n}\right)\left(1 - \frac{2}{n-1}\right) \ldots \left(1 - \frac{2}{3}\right) \\ &= \frac{n-2}{n} \cdot \frac{n-3}{n-1} \cdot \frac{n-4}{n-2} \cdots \frac{1}{3} \\ &= \frac{2}{n(n-1)} = 1/\binom{n}{2}. \end{aligned}$$

3 In this exercise, we are going to analyze a beautiful and simple randomized algorithm for the 2-SAT problem (due to Papadimitriou'91). Recall that in the 2-SAT problem we have n variables $x_1, x_2, \ldots, x_n$ and m clauses $C_1, \ldots, C_m$, where each clause is the disjunction of two literals. Examples of possible clauses are $x_i \vee x_j$, $x_k \vee \neg x_\ell$ and so on. The goal is to find a truth assignment to the n variables so that all clauses are satisfied (if one exists).

The algorithm we are going to analyze is ingenious and simple:

1. Select uniformly at random a truth assignment π.
2. Repeat the following for at most $O(n^2)$ steps (or until we find a satisfying assignment):

   Select an unsatisfied clause C_ℓ with variables x_i, x_j.

   Select one of the variables x_i and x_j with equal probability.

   Update π by flipping the truth assignment of the selected variable, i.e., set it to true if it was false and vice versa.

In the following, we assume for simplicity that there is a satisfying truth assignment π^* (where π^*_i denotes the Boolean value of variable x_i). (If there is no such truth assignment we cannot find one so there is nothing to prove.)

For a truth-assignment π, define $\text{dist}(\pi^*, \pi)$ as the number of variables/coordinates where $\pi_i \neq \pi^*_i$ (this is called the *Hamming distance*). Let $\pi = \pi^1, \pi^2, \ldots$ be the truth-assignments generated by the algorithm and let $d_i = \text{dist}(\pi^*, \pi^i)$. Note that as the algorithm only changes one variable in the truth assignment in each iteration, we have $d_{i+1} = d_i + \Delta_i$ where $\Delta_i \in \{-1, 1\}$. Prove that for any i

$$\Pr[\Delta_i = -1] \geq 1/2\,.$$

Once this is established, we can view the algorithm as a random walk on $\{0, 1, 2, \ldots, n\}$ that always goes to the left (to a smaller number) with probability at least $1/2$. It is known that such a random walk will reach 0 with high probability after $O(n^2)$ many steps. (Show this if you feel that you are up for a challenge.)

Solution: Let us first prove $\Pr[\Delta_i = -1] \geq 1/2$. To that end, consider i'th step of our algorithm and let C_ℓ be the clause with variables x_i, x_j that our algorithm chooses to modify. Since this clause in not satisfied, in any satisfying truth assignment π^* at least one of x_i or x_j should be different from their current value. Therefore, the variable that our algorithm chooses to flip at least with probability $1/2$ is flipped to its value in π^*. So $\Pr[\Delta_i = -1] \geq 1/2$.

In what follows, we argue that our algorithm indeed finds a satisfying assignment. To see that the algorithm reaches 0 with high probability. Notice that if there is any sequence $\Delta_k, \Delta_{k+1}, \Delta_\ell$ with n more -1's than 1's then we have reached 0. We show that the sequence $\Delta_1, \Delta_2, \ldots, \Delta_{cn^2}$ has n more -1's than 1's with constant probability. The statement then follows by doing a (constant) multiple of cn^2 many iterations and interpreting them as independent sequences. To see that $\Delta_1, \Delta_2, \ldots, \Delta_{cn^2}$ has n more -1's than 1's with constant probability notice that in the worst-case we have that $\Pr[\Delta_i = -1] = \Pr[\Delta_i = 1] = 1/2$. Assume that for the analysis as this will upper bound the probability that we are interested in. If we let $X = \sum_{i=1}^{cn^2} \Delta_i$ Then the probability of interest is

$$\Pr[X \leq -n] .$$

Now notice that n is smaller than the standard deviation of cn^2 coin flips. Indeed the variance of cn^2 unbiased ± 1 coin flips is cn^2. So this probability happens with constant probability (Here one has to be more careful and use the Binomial distribution).