INTRODUCTION TO APX – HW3

Hashing, lower bounds, parallelization

Every exercise is worth two points. Deadline: **15. 1. 2016 17:19**. Send the solution by email or hand it to me – either in person or put it in the labeled box in the corridor on the 3rd floor, opposite the room S323.

If the statements are unclear or you feel they may contain an error, do not be afraid to send me an email.

Exercise one

Design a 2-approximation algorithm for Sonet ring-loading problem.

In this problem we have a network – a cycle with n vertices. On input we get a list of requests, each request having a source vertex and a target vertex. Every request needs to be assigned one out of two possible paths from source to target (either clockwise or counterclockwise).

Our goal is to minimize the load (total number of uses) of the most loaded edge of the cycle.

Exercise two

Lower bound for strongly 2-universal hash functions using LA.

From the lecture we might recall what strongly 2-universal hash functions are: A family $\mathcal{H}$ of functions (all being hash functions, i.e. $h\colon U \to HT$) is strongly 2-universal, if for any quadruple of variables $x, x' \neq x, y, y'$ it is true that

$$P_{h \text{ randomly chosen from } \mathcal{H}}[h(x) = y \wedge h(x') = y'] = \frac{1}{|HT|^2}.$$

You might know that for every $|U| = 2^n$, $|HT| = 2^m$ we can generate a (weakly or strongly) 2-universal hash function using only $O(m+n)$ fully random bits. This is asymptotically tight, and the following sequence of observations proves this. We will use linear algebra to show this.

We now assume that $\mathcal{H}$ is any strongly 2-universal hash family.

1. Prove that if $|U| \geq 2$, then $|\mathcal{H}| \geq |HT|^2$.
2. Prove that if $|HT| = 2$, then $|\mathcal{H}| \geq |U| + 1$.

   Tip: Using functions from $\mathcal{H}$, generate for each x one vector v_x from $\mathbb{R}^{|\mathcal{H}|}$ with coordinates of type ± 1, while making sure that every pair v_x and $v_{x'}$ is orthogonal. If we find such vectors, what does it mean for the dimension of $\mathbb{R}^{|\mathcal{H}|}$? And why is the right side $|U| + 1$ and not just $|U|$?
3. Generalize the previous approach and prove that for arbitrary $|HT|$ it holds that

   $$|\mathcal{H}| \geq |U|(|HT| - 1) + 1.$$

   Tip: Using functions from $\mathcal{H}$, generate for each x a set of $|HT| - 1$ different vectors $v_{x,y} \in \mathbb{R}^{\mathcal{H}}$ so that for one x all vectors $v_{x,y}$ are linearly independent, and for two different $x \neq x'$ we have that the vectors $v_{x,y}$ and $v_{x',y'}$ are even orthogonal. What does this mean for the dimension of $\mathbb{R}^{|\mathcal{H}|}$?
4. Use the previous bullet points and conclude that if $|U| = 2^n$ and $|HT| = 2^m$, then for generating a random function from $\mathcal{H}$ we need at least $(\max(n, m) + m)$ random bits, which is asymptotically tight.

Exercise three

Consider again the problem of Graph balancing.

To recall: on input you get an undirected graph G with weights on the edges $p\colon E(G) \to \mathbb{R}^+$. Our goal is to make the graph directed so that the most loaded vertex (the vertex with the most weight directed towards it) has minimum possible load. Formally we seek an orientation of the edges which minimizes the goal function $u = \max_{v \in V} \sum_{e \in E; e \text{ directed to } v}$.

Prove that no algorithm can reach an approximation ratio lower than 3/2.

Tip: There may be many ways to prove this bound; here is a hint for a possible way: you can try to model one famous NP-complete problem using only a graph with edge weights 1 and 1/2. Then, try to argue that if an approximation algorithm gets a solution with approximation factor better than 3/2, then it must solve the NP-complete problem itself.

EXERCISE FOUR Consider graphs with the maximum degree limited by a constant, i.e. $\Delta(G) \leq \Delta$. A greedy algorithm can color this graph easily with $\Delta + 1$ colors – but it is not easy to see how to parallelize it.

Suggest a fast randomized parallel algorithm, which can properly color the graph with $\Delta + 1$ with very high probability.

Tip: You may want to build on the material from the lectures.

EXERCISE FIVE *Bonus exercise.* We consider one more problem from before, namely the k-SUPPLIER PROBLEM.

Prove that no approximation algorithm can have its ratio lower than 3.

We recall for completeness: We get a number k and $m + n$ points on input, where m of them are (in advance) marked as *suppliers* and the rest are *consumers*. Between all those points is a metric (with a triangle inequality as usual). Our task in this case is to select k suppliers so that we minimize the longest distance between a customer and its closest supplier.

Tip: At some point we may want to use the fact that k is on input as well – so we can create an instance and then choose a good k for this instance, so the instance is hard to solve (but for some other k it may be simpler).