Tutor: Julien Dallot

Exercise Sheet 3

Tree Algorithms

Exercise 1 (Lightest Edges)**.** In this exercise, we use the all-to-all communication model in which each node can communicate with all other $n-1$ nodes in each synchronous communication round, i.e., the communication graph is the complete graph $\mathcal{K}_n$. Each node v has a unique identifier id_v and each edge e in the communication graph has a positive weight $w(e)$. You can assume all edge weights to be unique. The size of any message is restricted in that only a constant number of node identifiers, edge weights, and additionally a constant number of other numbers of the same magnitude can be sent in a single round.

In the lecture, we discussed algorithms to compute the minimum spanning tree (MST) in this model. Now we are interested in finding the n lightest edges overall, i.e., after the algorithm terminates, every node knows the weights of the n lightest among all $\binom{n}{2}$ edges and which nodes these edges connect. Initially, each node v knows the weights of all incident edges.

Consider the following simple algorithm:

Every node sends its i^{th} lightest edge to all other nodes in round i. After a sufficiently large number of rounds, the algorithm terminates and each node knows that the n smallest weights it has learned belong to the n lightest edges overall.

1. Show an example (that means, an assignment of edge weights to the nodes) where the above algorithm is as slow as possible!
   Hint: The problem with this simple algorithm is that nodes potentially send edge weights that have already been broadcast (by other nodes) before.

In order to overcome the problem mentioned above, we modify the algorithm in the following way: *In each round, broadcast the lightest incident edge weight that has not already been broadcast before.*

b) Prove an upper bound on the number of rounds required when the modified algorithm is used! Moreover, prove that your bound is asymptotically tight by providing a worst-case example (of the same asymptotic time complexity)!

Now, we are going to derive a *randomized* algorithm whose *expected* time complexity is only $O(1)$. Use the fact that a single node can determine the n^{th} smallest among all $\binom{n}{2}$ edge weights in $O(1)$ rounds in expectation.[1]

c) Given that node v knows the n^{th} smallest edge weight (after $O(1)$ rounds), how can all nodes (v and all other nodes) learn all the weights of the n lightest edges? Describe an algorithm that solves this problem! The total time complexity must not exceed $O(1)$ rounds!

Exercise 2 (License to Match)**.** In preparation for a highly dangerous mission, the participating agents of the gargantuan Liechtensteinian secret service (LSS) need to work in pairs of two for safety reasons. All members in the LSS are organized in a tree hierarchy. Communication is only possible via the official channel: an agent has a secure phone line to his direct superior and a secure phone line to each of his direct subordinates. Initially, each agent knows whether or not he is taking part in this mission. The goal is for each agent to find a partner.

[1] Note that this algorithm cannot be *parallelized*! This means that this subroutine cannot be used to compute all n lightest edges *in parallel* in $O(1)$ time.

1. Devise an algorithm that will match up a participating agent with another participating agent given the constrained communication scenario. A "match" consists of an agent knowing the identity of his partner and the path in the hierarchy connecting them. Assume that there is an even number of participating agents so that each one is guaranteed a partner. Furthermore, observe that[2] the phone links connecting two paired-up agents need to remain open at all times. Therefore, you cannot use the same link (i.e., an edge) twice when connecting agents with their partners.

2. What are the time and message (i.e., "phone call") complexities of your algorithm?

Exercise 3 (License to Distribute)**.** We consider another day at the office of the LSS as in Exercise 2. After the above mission was successful, the involved agents collected a large number of sensitive documents. Some agents might have a lot of them and others have none. Now they need to distribute the documents throughout the agency such that each person in the LSS has the same amount of data to process.

1. Assume that there are n agents in the LSS and that there is also a total of n documents. Devise a way for the agents to distribute their sensitive data: in the end, each agent should have exactly one document. The communication scheme is the same as above.

2. How good is your algorithm with respect to time and number of messages? You may assume that arbitrarily many documents can be sent in a single message.

Exercise 4 (Restructuring the LSS)**.** Recall the organizational structure of the LSS: each member of the LSS can communicate only with his direct superior and his direct subordinates over a secure phone line. On top of this tree hierarchy sits L, the "big boss". Members who do not have any subordinates are the field agents. All others (including L) are office workers. Let the total number of LSS members be $n > 1$.

1. In an effort to improve the efficiency of the LSS, L suspects that there are too many office workers and not enough field agents to save the world. To that end, she needs to know exactly how many people in the company are office workers and how many are field agents. Devise an efficient asynchronous, distributed algorithm, started by L, to determine those numbers.

2. What are the time and message complexities of your algorithm? Because of political turbulences, Liechtenstein is now being split into two countries, Lichtstein and Lampenstein, who each want to have their own secret services, LiSS and LaSS, respectively. The politicians agree to create two groups of people out of the original LSS. The goal is that each new group collectively has the capacity to perform the same jobs as the LSS before. The jobs were such that, in the original LSS, every member knew how to execute his own task and all the tasks of his direct contacts (i.e. the direct superior's and subordinates' tasks). And since a person is only allowed single citizenship, he can only be part of either the LiSS or the LaSS. Note that we do not care about the internal structure of the future LiSS and LaSS at this point, only about membership.

3. Devise an asynchronous algorithm that assigns each member of the LSS to either LiSS or LaSS. The algorithm is initiated by L and should terminate in time $O(depth(T))$, where T refers to the LSS structure.

4. Same task as above. Now the algorithm may be synchronous, is started by all members simultaneously and should terminate in time $O(\log^* n)$, where n is the number of members in the original LSS. You may use the algorithms from Chapter **??** as a black box. Show that your algorithm correctly solves the problem in the specified amount of time.

[2]in the case of an emergency where they lose contact

5. If Liechtenstein had been split into several countries, how many such entities could have been created maximally and why?