

**General rules for solving exercises**

- When handing in your solutions, please write your exercise group on the front sheet:

**Group A:** Wed 13–15 CAB G 56

**Group B:** Wed 16–18 NO C 44

**Group C:** Wed 16–18 CAB G 52

- This is a theory course, which means: if an exercise does not explicitly say “you do not need to prove your answer”, then a formal proof is **always** required.
- 

The following exercises will be discussed in the exercise classes on September 25, 2019. Please hand in your solutions in the lecture, no later than September 24.

**Exercise 1**

Let  $G = (V, E)$  be a connected graph with weights  $w : E \rightarrow \mathbb{R}$  on the edges, and define the edge boundary of set  $S$  to be

$$\partial(S) := \{\{u, v\} \in E : u \in S, v \in V \setminus S\}.$$

Assume that for every non-empty vertex set  $S \subset V$ , the edge with the minimum weight in  $\partial(S)$  is unique.

Prove that  $G$  has a unique MST. Conclude that if the weight function  $w$  is injective (i.e., no two edges have the same weight),  $G$  contains exactly one MST.

## Exercise 2

You already know that for a connected graph  $G = (V, E)$ , with  $n = |V|$  and  $m = |E|$ , the expected running time of *Randomized Minimum Spanning Tree Algorithm* ( $G$ ) (see page 5 in the lecture notes) is equal to  $\mathcal{O}(m)$ .

(i) Prove that the worst-case running time of the algorithm is equal to  $\mathcal{O}(\min\{n^2, m \log n\})$ .

(ii) Prove that the running time of the algorithm is equal to  $\mathcal{O}(m)$  with probability  $1 - o(1)$  in the following two steps.

- (a) Let  $D(n, m)$  be the worst-case running time of the recursive algorithm without considering the two recursive calls, and let  $T(n, m)$  be the worst-case running time of the recursive algorithm. It is clear that  $D(n, m) = \mathcal{O}(n + m)$ , and by (i),  $T(n, m) = \mathcal{O}(\min\{n^2, m \log n\})$ . Figure 1 represents a binary tree of running times in which every first recursive call works on a graph with at most  $\frac{n}{8}$  vertices and at most  $\frac{3}{4}m$  edges, every second recursive call works on a graph with at  $\frac{n}{8}$  vertices and at most  $\frac{3}{8}n$  edges, and the worst-case function  $T(n^{2/5}, \infty)$  will be applied when the number of vertices has shrunk down to  $n^{2/5}$ .

Prove that there exists a constant  $c_3 > 0$  such that the sum of all running times in the nodes of the tree depicted in Figure 1 is bounded from above by  $c_3 \cdot (n + m)$ .

- (b) Prove that for any connected input graph  $G$ , the running time of the algorithm is dominated by the sum over all nodes in the tree depicted in Figure 1 with probability  $1 - o(1)$  (i.e., a number that tends to 1 as  $n \rightarrow \infty$ ).

*Hint: Let  $G_1$  and  $G_2$  be the two graphs for the first and second recursive calls, respectively. Call  $G_1$  bad if  $G_1$  has more than  $\frac{n}{8}$  vertices or more than  $\frac{3}{4}m$  edges, and call  $G_2$  bad if  $G_2$  has more than  $\frac{n}{8}$  vertices or  $\frac{3}{8}n$  edges. You might apply the Chernoff bound to bound the probability that  $G_1$  or  $G_2$  is bad. For the Chernoff bound, see the help sheet on the website of the course.*

## Exercise 3

For a graph  $G = (V, E)$ , a cut is the partition of the vertex set  $V$  into two disjoint sets  $V_1$  and  $V_2$  and the size of the cut is the number of edges between  $V_1$  and  $V_2$ .

(i) Assume that  $|V|$  is even. We say a cut is balanced if  $|V_1| = |V_2| = |V|/2$ . Prove that there always exists a balanced cut of size at least  $|E|/2$ .

*Hint: Since you want to prove the existence, you might apply randomness.*

(ii) Define  $d_S(v)$  to be the number of neighbors of vertex  $v$  in a set  $S \subseteq V$ , i.e.  $d_S(v) := |\{u \in S : \{v, u\} \in E\}|$ . Now, consider the following algorithm. Partition the vertex set  $V$  into two arbitrary sets  $V_1$  and  $V_2$ . As far as there is a vertex  $v \in V_1$  (or  $v \in V_2$ ) for which  $d_{V_1}(v) > d_{V_2}(v)$  (resp.  $d_{V_2}(v) > d_{V_1}(v)$ ) move  $v$  from  $V_1$  to  $V_2$  (resp. from  $V_2$  to  $V_1$ ). Prove that this algorithm terminates and generates a cut of size at least  $|E|/2$ .

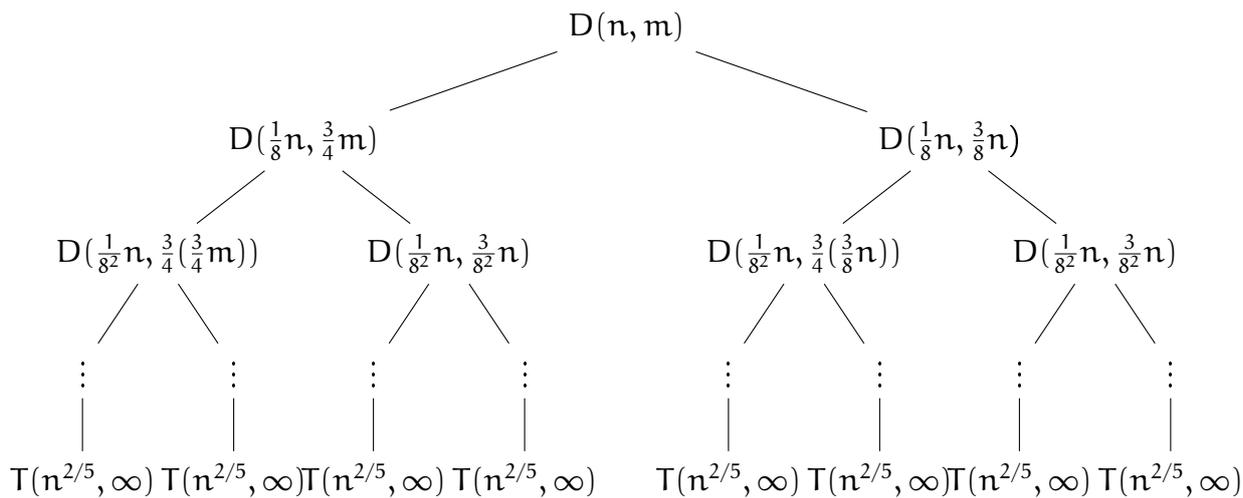


Figure 1: A binary tree of very specific running times that we use in order to prove an upper bound on the actually observed running time of the algorithm.

### Exercise 4

In a city there are  $n$  houses  $h_1, \dots, h_n$ , each of which is in need of a water supply. It costs  $c_i$  to build a well at house  $h_i$ , and it costs  $w_{ij}$  to build a pipe in between houses  $h_i$  and  $h_j$ . A house can receive water if either there is a well built there or there is some path of pipes to a house with a well. Give an algorithm to find the minimum cost to supply every house with water.

*Hint: There is a short solution.*