Exercise 8: Don’t get Lost

Task 1: . . . everything is (probably) going to be fine

An event occurs with high probability (w.h.p.), if its probability is, for any choice of \( c \in \mathbb{R}_{\geq 1} \), at least \( 1 - n^{-c} \). Here \( n \) is the input size (in our case, \( n = |V| \)), and \( c \) is a (user-provided) parameter, very much like the \( \epsilon \) in a \((1 + \epsilon)\)-approximation algorithm.

Algorithm 1 Code for generating a random ID at node \( v \).

1: \( \text{id}_v \leftarrow \lceil c \log n \rceil \text{ random bits} \)

a) Suppose that some algorithm \( \mathcal{A} \) is called ten times, and each call succeeds w.h.p. Pick \( c \) such that for \( n \geq 10 \), all ten calls of \( \mathcal{A} \) all succeed with a probability of at least 0.999.

\( \text{Hint:} \) Union bound.

b) Let \( \mathcal{E}_1, \ldots, \mathcal{E}_k \) be polynomially many events, i.e., \( k \in n^{O(1)} \), each of them occurring w.h.p. Show that \( \mathcal{E} := \mathcal{E}_1 \cap \cdots \cap \mathcal{E}_k \), the event that all \( \mathcal{E}_i \) happen, occurs w.h.p.

c) Consider Algorithm 1, which generates random node IDs. Fix two distinct nodes \( v, w \in V \) and show that w.h.p., they have different IDs.

d) Show that w.h.p., Algorithm 1 generates pairwise distinct node IDs.

Task 2: . . . in the Steiner Forest!

In this exercise, we’re going to find a 2-approximation for the Steiner Tree problem on a weighted graph \( G = (V, E, W) \), as defined in an earlier exercise; we use the \textsc{Congest} model. Denote by \( T \) the set of nodes that need to be connected, and by \( G_T = (T, \left( \begin{smallmatrix} T \\ 2 \end{smallmatrix} \right), W_T) \) the terminal graph.

a) For each node \( v \), denote by \( t_v \) the closest node in \( T \). Show that all \( v \in V \) can determine \( t_v \) along with the weighted distance \( \text{dist}(v, t_v) \) in \( \max_{v \in V} \{ \text{hop}(v, t_v) \} + O(D) \) rounds,\(^1\) where \( \text{hop}(v, t_v) \) denotes the length of the minimum hop path from \( v \) to \( t_v \).

\( \text{Hint:} \) This essentially is a single-source Moore-Bellman-Ford with a virtual source connected to all nodes in \( T \).

b) Consider a terminal graph edge \( \{t_v, t_w\} \) “witnessed” by \( G \)-neighbors \( v \) and \( w \) with \( t_v \neq t_w \), i.e., \( v \) and \( w \) know that \( \text{dist}(t_v, t_w) \leq \text{dist}(t_v, v) + W(v, w) + \text{dist}(w, t_w) \). Show that if there are no such \( v \) and \( w \) with \( \text{dist}(t_v, t_w) = \text{dist}(v, t_v) + W(v, w) + \text{dist}(w, t_w) \), then \( \{t_v, t_w\} \) is not in the MST of \( G_T \)!

\( \text{Hint:} \) Observe that \( G \) is partitioned into Voronoi cells \( V_t = \{v \in V \mid t_v = t\} \), and that in the above case any shortest \( t_v-t_w \) path must contain a node \( u \) with \( t_u \notin \{t_v, t_w\} \), i.e., cross a third Voronoi cell. Conclude that \( \{t_v, t_w\} \) is the heaviest edge in the cycle \( \{t_v, t_u, t_w, t_v\} \).

\(^1\)These are partial shortest-path trees rooted in each \( t \in T \).
c) Show that an MST of $G_T$ can be determined and made globally known in $O(|T| + D)$ additional rounds.

**Hint:** Use the distributed variant of Kruskal’s algorithm from the lecture.

d) Show how to construct a Steiner Tree of $G$ of at most the same weight as the MST of the terminal graph in additional $\max_{v \in V}\{\text{hop}(v, t_v)\}$ rounds.

**Hint:** Modify the previous step so that the “detecting” pair $v, w$ with $\text{dist}(t_v, t_w) = \text{dist}(v, t_v) + W(v, w) + \text{dist}(w, t_w)$ is remembered. Then mark the respective edges $\{v, w\}$ and the leaf-root-paths from $v$ to $t_v$ and $w$ to $t_w$ for inclusion in the Steiner Tree.

e) Conclude that the result is a 2-approximate Steiner Tree. What is the running time of the algorithm?

**Hint:** Recall Task 2 from Exercise 6.

**Task 3**: . . . under a Heap of Presents

<table>
<thead>
<tr>
<th>weight</th>
<th>RGB</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>(255,255,0)</td>
</tr>
<tr>
<td>2</td>
<td>(34,139,34)</td>
</tr>
<tr>
<td>3</td>
<td>(165,42,42)</td>
</tr>
<tr>
<td>5</td>
<td>(255,0,0)</td>
</tr>
<tr>
<td>20</td>
<td>(193,255,244)</td>
</tr>
</tbody>
</table>

a) Determine an MST of the graph given in Figure 1!

b) Color each MST edge. The edge colors are given in the table above, i.e., an edge of weight 1 has color (255,255,0).

c) Look for other Christmas trees in the computer science literature!

**Hint:** xkcd.

d) Have a Merry Christmas and a Happy New Year!
Figure 1: Poorly disguised Christmas tree.