Homework 1 Optional Problems

> *Consider a connected undirected graph G with not necessarily distinct edge costs. Consider two different minimum-cost spanning trees of $G, T \text{ and } T'$. Is there necessarily a sequence of minimum-cost spanning trees $T = T_0, T_1, T_2, \ldots, T_r = T'$ with the property that each consecutive pair T_i, T_{i+1} of MSTs differ by only a single edge swap? Prove the statement or exhibit a counterexample.*

ANSWER: Consider the following pseudocode:

```
MST-SEQ(G)
  S = {}
  for all v in V(G)
    take a cut (A, B) of T
        s.t. V(A) = {v}, V(B) = V(G) - {v}
    take a cut (A', B') of T'
        s.t. V(A') = {v}, V(B') = V(G) - {v}
    e = edge crossing cut (A, B)
    e' = edge crossing cut (A', B')

    if (e != e')
      E(T) = E(T) - {e} U {e'}
      S = S U T

  return S
```

In each iteration, we (potentially) generate a new MST that differs from the previous one by one edge, and from T' by one less edge overall. Clearly, we can transform T into T' proceeding this way.

> *Consider the following algorithm. The input is a connected undirected graph with edge costs (distinct, if you prefer). The algorithm proceeds in iterations. If the current graph is a spanning tree, then the algorithm halts. Otherwise, it picks an arbitrary cycle of the current graph and deletes the most expensive edge on the cycle. Is this algorithm guaranteed to compute a minimum-cost spanning tree? Prove it or exhibit a counterexample.*

ANSWER: This algorithm is guaranteed to compute a MST. To prove it, lets assume that T is a $MST(G)$ and that the most expensive edge e is included in T. By the *Double-Crossing Lemma* (lecture video 4.3), there exists another edge $e' \notin T, e' \neq e$ that connects the two endpoints of e.

If we delete e from T and add e', we get a spanning tree of lower weight, which is a contradiction to the assumption that T is a MST. Thus, e can't be included in the MST, and hence by deleting it, no MST edge is deleted.

> *Consider the following algorithm. The input is a connected undirected graph with edge costs (distinct, if you prefer). The algorithm proceeds in phases. Each phase adds some edges to a tree-so-far and reduces the number of vertices in the graph (when there is only 1 vertex left, the MST is just the empty set). In a phase, we identify the cheapest edge e_v incident on each vertex v of the current graph. Let $F = \{e_v\}$ be the collection of all such edges in the current phase. Obtain a new (smaller) graph by contracting all of the edges in F — so that each connected component of F becomes a single vertex in the new graph — discarding any self-loops that result. Let T denote the union of all edges that ever get contracted in a phase of this algorithm. Is T guaranteed to be a minimum-cost spanning tree? Prove it or exhibit a counterexample.*

ANSWER: This algorithm was discovered by Otakar Borůvka [1]. It is guaranteed to compute a MST, and the proof follows from the Light-Edge property. Realize that when we expand via minimum weight edges, since these edges are guaranteed to be in the MST, they cannot form a cycle, hence the result we get is a forest.

References

1. Borůvka's Algorithm (https://en.wikipedia.org/wiki/Bor%C5%AFvka%27s_algorithm) ↩

COMMENTS