Recitation 11

Graph Contraction and MSTs

11.1 Announcements

- *SegmentLab* has been released, and is due **Monday afternoon**. It’s worth 135 points.
- *Midterm 2* is on **Friday, April 8**.
11.2 Contraction

In the textbook, we presented an algorithm for counting the number of connected components in a graph:

Algorithm 11.1. (Algorithm 16.20 in the textbook.)

\[
\text{countComponents}(V, E) = \\
\text{if } |E| = 0 \text{ then } |V| \text{ else} \\
\text{let } (V', P) = \text{starPartition}(V, E) \\
E' = \{(P[u], P[v]) : (u, v) \in E \mid P[u] \neq P[v]\} \\
\text{in} \\
\text{countComponents}(V', E') \\
\text{end}
\]

Now, suppose we implemented star partitioning for enumerated graphs as follows:

\[
\text{val enumStarPartition : } (\text{int} \times \text{int}) \text{ Seq.t } \times \text{int} \rightarrow \text{int Seq.t}
\]

Specifically, given a graph represented as a sequence of edges $E$ where every vertex is labeled $0 \leq v < n$, \((\text{enumStarPartition}(E, n))\) returns a mapping $P$ where $P[v]$ is the supervertex containing $v$. (If $v$ was a star center or was unable to contract, then $P[v] = v$.)

Task 11.2. Implement a function \texttt{enumCountComponents} which counts the number of components of an enumerated graph. It should take in a graph represented as $(E, n)$ and use \texttt{enumStarPartition} internally. (Hint: be careful with the value $n$... there is something subtle going on here!)

11.2.1 Cost Bounds

Task 11.3. Recall that a forest is a collection of trees. What are the work and span of \texttt{enumCountComponents} when applied to a forest? Assume that \((\text{enumStarPartition}(E, n))\) requires $O(n + |E|)$ work and $O(\log n)$ span.
11.3 Borůvka’s Algorithm

The textbook describes two versions of Borůvka’s algorithm: one which performs tree contraction at each round, and another which performs a single round of star contraction at each round. We will be using the latter, since it has better overall span ($O(\log^2 n)$ rather than $O(\log^3 n)$).

**Task 11.4.** Run Borůvka’s algorithm on the following graph. Draw the graph at each round, and identify which edges are MST edges. Use the coin flips specified.
11.4 Additional Exercises

**Exercise 11.5.** In graph theory, an *independent set* is a set of vertices for which no two vertices are neighbors of one another. The *maximal independent set* (MIS) problem is defined as follows:

For a graph \((V, E)\), find an independent set \(I \subseteq V\) such that for all \(v \in (V \setminus I)\), \(I \cup \{v\}\) is not an independent set.\(^a\)

Design an efficient parallel algorithm based on graph contraction which solves the MIS problem.

\(^a\)The condition that we cannot extend such an independent set \(I\) with another vertex is what makes it “maximal.” There is a closely related problem called *maximum independent set* where you find the largest possible \(I\). However, this problem turns out to be NP-hard!