Recitation 12

Graph Contraction

12.1 Announcements

- *SegmentLab* has been released, and is due **Friday, November 17**.
- *Midterm 2* is tomorrow, **Wednesday, November 8**.

12.2 Contraction

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

Algorithm 12.1. *(Algorithm 17.22 in the textbook.)*

```
1 countComponents (V, E) =
2   if |E| = 0 then |V| else
3   let
4     (V', P) = starPartition (V, E)
5     E' = {(P[u], P[v]) : (u, v) ∈ E | P[u] ≠ P[v]}
6   in
7     countComponents (V', E')
8   end
```

with `starPartition` implemented as follows:

Algorithm 12.2. *(Algorithm 17.15 in the textbook.)*

```
1 starPartition (V, E) =
2 let
3   TH = {(u, v) ∈ E | ¬heads(u)∧heads(v)}
4   P = ⋃_{(u,v)∈TH} {u ↦ v}
5   V' = V \ domain(P)
6   P' = {u ↦ u : u ∈ V'}
7 in
8   (V', P' ∪ P)
9 end
```

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

```
val enumStarPartition : (int * int) Seq.t * int → int Seq.t
```

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

Task 12.3. *Implement a function `enumCountComponents` which counts the number of components of an enumerated graph. It should take in a graph represented as (E, n) and use `enumStarPartition` internally.*

12.2.1 Cost Bounds

Task 12.4. *Recall that a* ***forest*** *is a collection of trees. What are the work and span of* `enumCountComponents` *when applied to a forest? Assume that* (`enumStarPartition` (E, n)) *requires* $O(n + |E|)$ *work and* $O(\log n)$ *span.*

.