Baruvka's Algorithm

- **Repeat**
  - **Every node picks its minimum incoming edge and adds to the spanning tree T**
  - **Contract all edges in T**

- **How much progress is made?**
- **How implement an iteration?**
- **Total running time?**

Baruvka's Algorithm

- **Repeat**
  - **Every node picks its minimum incoming edge and adds to the spanning tree T**
  - **Contract all edges in T**

- **How much progress is made?**
- **How implement an iteration?**
- **Total running time?**

$$T(n, m) = T(n/2, m - n) + O(n + m)$$

Problem: **Edges don't decrease fast enough**

Eliminating Edges

- **The heaviest edge on any cycle is not in the MST.**
- **Let** F **be a forest**
- **Let** $w_F(u, v)$ **be the maximum weight of an edge on the path from** u **to** v **in** F **(or** ∞ **if the path does not exist.**
- **Edge** (u, v) **is F-heavy if** $w(u, v) > w_F(u, v)$ **and F-light otherwise.**

Claim: Let F **be any forest, let** (u, v) **be any edge. If** (u, v) **is F-heavy, then** (u, v) **is not in the MST.**

Ideas

- **It is good to eliminate F-heavy edges**
- **The MST T would let us eliminate all non-MST edges.**
- **What can an F that is not an MST eliminate**

Fact to accept without proof: **Given G and a forrest F, we can eliminate all F-heavy edges in** $O(n+m)$ **time (spanning tree verification).**

Algorithm $MST(G)$

1. Run 3 Bruvka phases to get G' . Let C be the contracted edges.
2. Let G'' be G' with each edge included with prob. $1/2$.
3. Recursively compute $F'' = MST(G'')$.
4. Identify the **F”-heavy** edges in G' . Delete them to obtain G''' .
5. Recursively compute $F''' = MST(G''')$
6. Return $F''' \cup C$

Note: The recursion bottons out on a graph with $O(1)$ nodes.

Key Lemma: Let H be a subgraph of G where each edge is included with probability p . Let F be a Minimum Spanning Forest of H . Then the expected number of **F-light** edges in G is at most n/p .

Recurrence $T(n,m) \leq O(n+m) + T(n/8, m/2) + T(n/8, n/4) = O(n+m)$