IE 495 Lecture 14

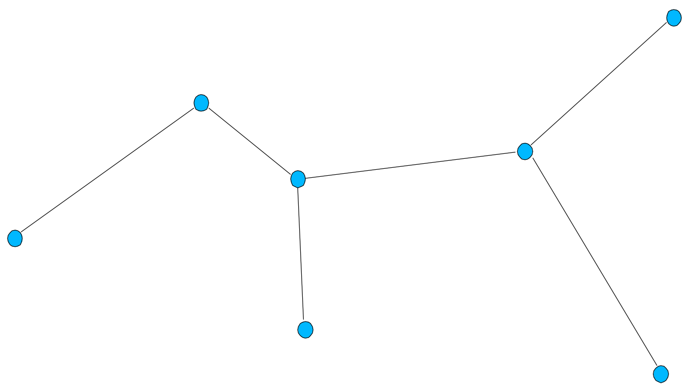
October 17, 2000

Reading for This Lecture

- Primary
  - Horowitz and Sahni, Chapter 4
  - Kozen, Lecture 3
- Secondary
  - Miller and Boxer, Chapter 12 (up to page 286)

Spanning Trees

- We are given a graph $G = (V, E)$.
- A spanning tree of E is a *maximal acyclic subgraph* (V, T) of G.
- A spanning tree always has $|V|-1$ edges (why?).



Minimum Spanning Tree

- We associate a weight w_e with each edge e.
- Objective: Find a spanning tree of minimum weight.
- Applications

Prim's Algorithm

```
S is the set of nodes in the tree

S = {0}
for (i = 1; i < n; i++){

  SELECT v ∉ S nearest to S;
  S = UNION(S, v);
}
```

Analysis of Prim's Algorithm

- Correctness
- Optimality
- Implementation
- Complexity

Kruskal's Algorithm

T is the set of edges in the tree

```
T = Ø
for (i = 0; i < m; i++){
  SELECT the cheapest edge e
  if (feasible(UNION(T, e)){
     UNION(T, e);
}
```

Analysis of Kruskal's Algorithm

- Correctness
- Optimality
- Implementation
- Complexity

Parallel MST

- Prim's Algorithm
- Each processor is responsible for a subset of the nodes.
- Implementation
- Analysis

Baruvka's Algorithm

- At each step, select all edges that connect some component of the graph to it's nearest neighbor.
- Add all these edges to the tree simultaneously.
- Why does this work?

- Sequential Implementation