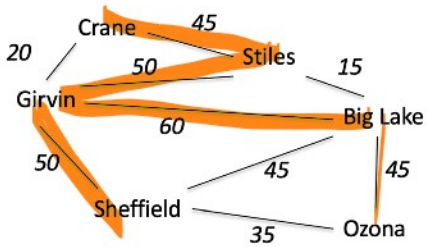
Creating electrical networks

Wednesday, October 26, 2022 1:00 PM

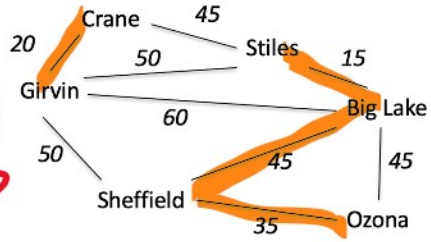


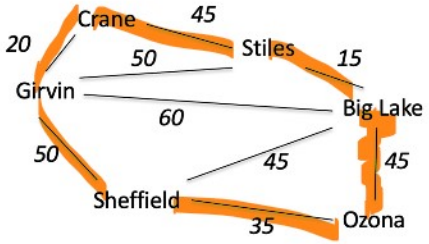
✗ not minimum

- Use the smallest length of wire possible → minimum
- Bring electricity to all of the cities (and we want a connected electrical grid) → spanning
- Don't want unnecessary connections between cities (no cycles in the electrical network)

→ tree

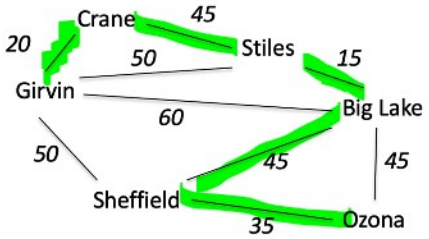
✗ not spanning

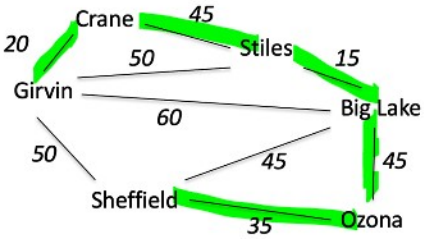




✗ not a tree

Two MSTs on this graph:

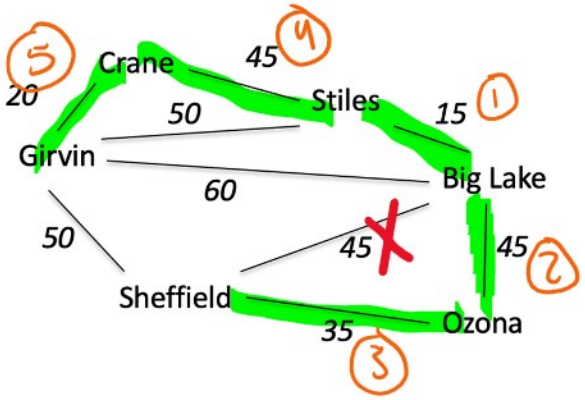




MST algorithms

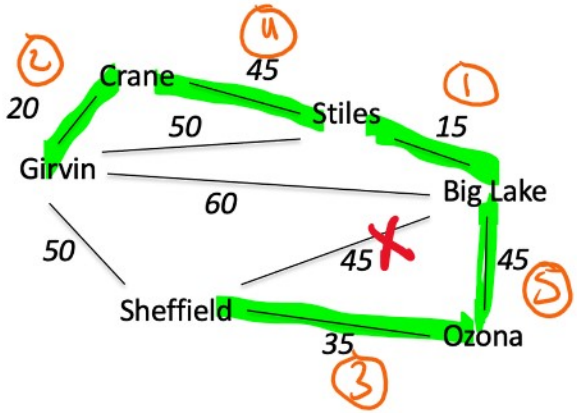
Wednesday, October 26, 2022 1:09 PM

Jarnik's/Prim's



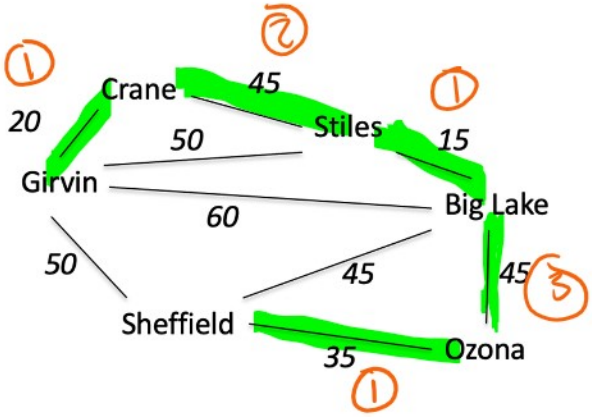
- Choose an arbitrary vertex
  Build up a tree from there:
  Of all of the possible new edges coming out of the tree, choose the smallest one that doesn't introduce a cycle and add it to the tree
  Stop when you connect all of the cities

Kruskal's



First, sort all of the edges by weight
For each step, select the smallest edge that doesn't introduce a cycle
Stop when you have all of the cities in a single tree

Sullin

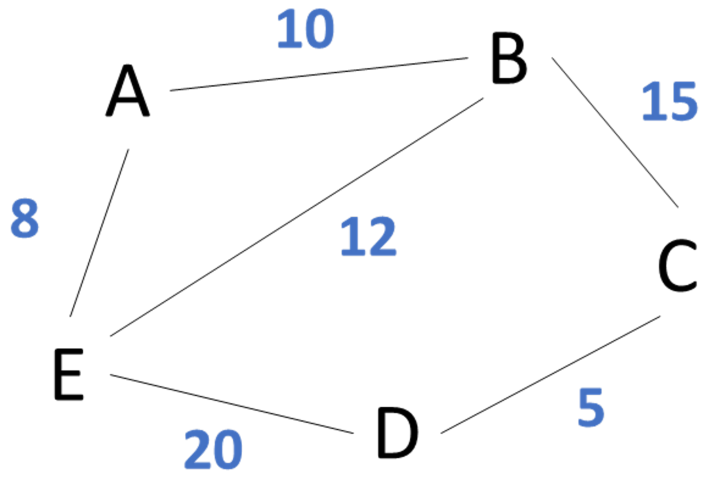


Cities pair off to form "optimal" collectives

The collectives repeatedly try to connect to each other until we have a spanning tree

Kruskal's pseudocode

Wednesday, October 26, 2022 1:11 PM




```
Notation: (u, v) to denote the edge between u and v

Inputs: E (the collection of edges),
        V (the collection of vertices)

sortedE = E as a sorted list

retTree = empty Graph

while _____________: <-- what should this end condition be?

   (u,v) = sortedE.removeFirst()

   if (u, v) doesn't introduce a cycle: <-- how do we determine this?
      add (u, v) to retTree

return retTree
```