1/24 Minimum Spanning Tree

Wednesday, January 24, 2018 5:57 PM

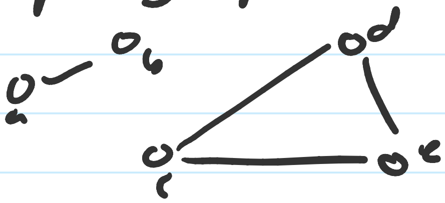
MST or
Minimum Spanning
Tree

Graph 2sets
Vertex Set V or V(G)
Edge set E or E(G)

Vertex(Points or
Nodes)
Edge (connection)

e=(a,b) where
$a,b \in V$

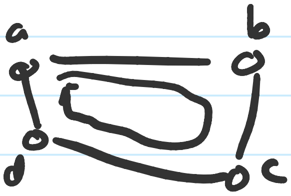
Example graph



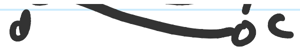
V={a,b,c,d,e}

Tree a connect graph
with no self loops or
cycles

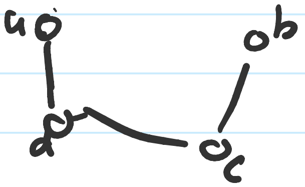
example cycle



-(a,b)



$-(a,b)$



$w(a,d)$
$w(d,c)$
$w(b,c)$

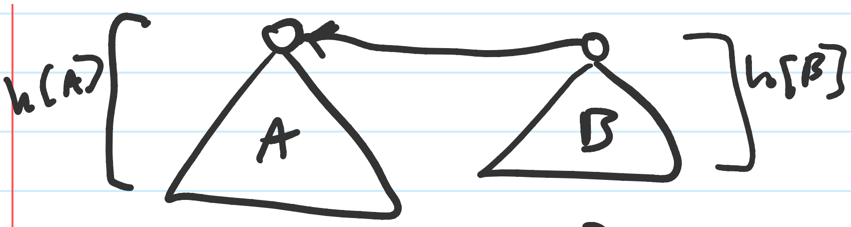
Disjoint set revisited

$V = \{a\}, \{c\}, \dots,$
$\{b\}, \{d\}, \dots,$

for our purposes
$= \{1\}, \{2\}, \{3\}, \dots, \{n\}$

```
find(int n)
    returns a representative for
    the set containing n;

union(int a, int b)
    int rep a
    int rep b
    adjust the disjoint set
    so rep a will represent
    set containing b or vice
    versa
```



if (h[A] > h(B))
then
adding B to A
won't chage the height
of A
otherwise
The height of A going
into B will be at most
h[B]+1

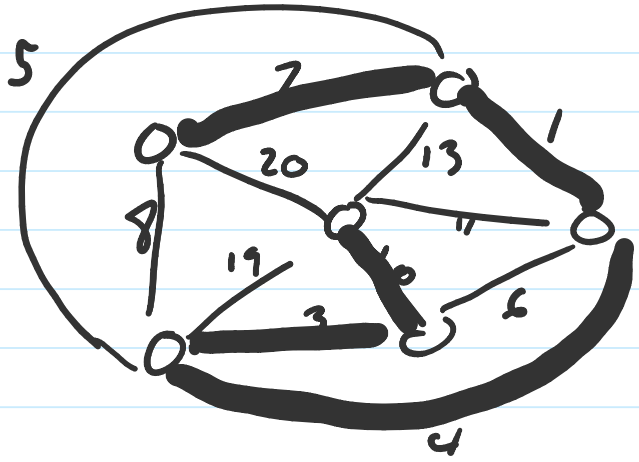
Suppose we have
some connected graph G

if we add an edge to
G we induce a cycle

add edge (u,v)

Sort all the edges by weight

By greedily selecting the least weight edge and adding it to our graph if the two vertexes are disconnected we can build a tree.



A tree with n Nodes contains n-1 Edges

A forest is a graph (potentially disconnected) with no cycles.

Kruskal's works by greedily adding edges until all edges have been checked

Kruskal's step by step

1. Initialize chosen Edges to ∅ (Empty Set)

Kruskal's step by step

1. Intialize chosenEdges to Ø (Empty Set)
2. Inialize a Disjoint Set to track connected nodes
3. Loop through edges least to greatest
   - 3a. Check if current edge connects two separate components.
   - 3b. Add edge if components are disconnected
   - 3c. Update Disjoint Set if neccessary
4. Return chosen Edge Set.