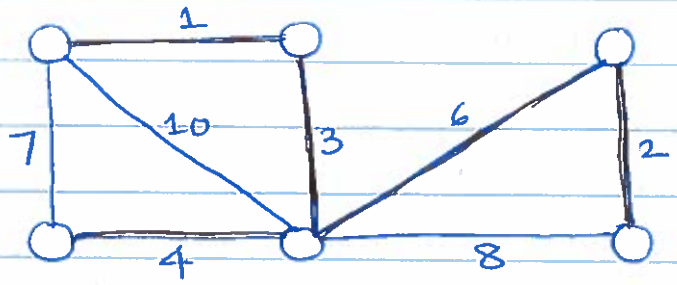
10/16/18

Greedy Algorithms

Minimum Spanning Tree (MST)

Def: Min Spanning Tree
Min Total Weight

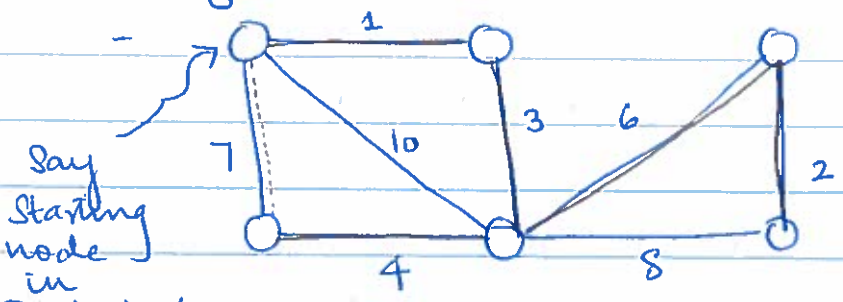
Spanning tree
↓
a cyclic subset of edges that connects all vertices



MST T is black edges

? Is there a notion of a starting node?

- No. Unlike shortest path, here we are trying to reduce the overall path length.
- Think of a routing problem to minimize overall length of wire.



Say Starting node in Dijkstra's,
we would have edge with wt 7 but MST has 1, 3, 4 wt. paths.

→ Cycle Property: Let C be any cycle (in the original graph G) and let edge (v, w) be the highest weight edge in C.
Then MST doesn't contain this edge.

Assume all edge weights distinct, so no ties

★ Proof: Left to the reader! ☺

→ Cut Property: Let S be any subset of nodes and suppose edge (v, w) is the minimum weight edge with one end in S and the other in V \ {S}
Then MST contains this edge.

all vertices except those in S

→ PRIM'S ALGORITHM:

Always a tree that grows.

Choose arbitary node S.
Grow T from S, at each step adding shortest edge connecting T with a node not in T.

→ KRUSKAL'S ALGORITHM:

Data structure for implementation is ~~union~~ find

Intermediate step need not be a tree.

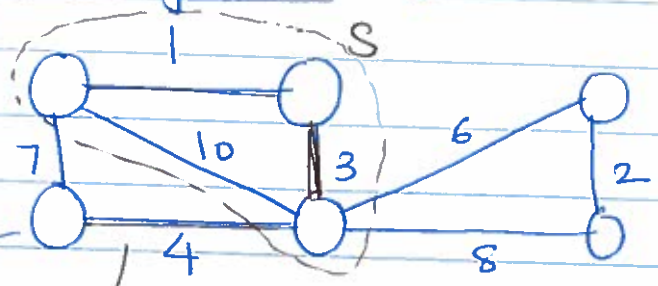
Choose edges in order from shortest to longest, ~~skipped~~ skipping ones that would make a cycle.

Not discussing implementation in class but do read it in your free time!

- Both algorithms have the same runtime.

→ Proof of Correctness & Complexity

1) Prim's Algorithm - (Use Cut Property)



v.d = 4

How do we know we need to choose this edge?
- Use cut property

Q → set of nodes not in tree T yet

Exactly like Dijkstra's except:

(i) v.d = upper bound on distance from current tree T to v

(ii) RELAX (v, u):
If u ∈ Q & u.d > w(v,u)
u.d = w(v,u)
u.π = v (parent set here)

↓ For entire algorithm pseudocode refer the slides.

⇒ Huffman Coding: Data Compression (Greedy)
- ZIP, JPEG, PNG
- Encode symbols using bits
- Simplest: Fixed length per symbol
  eg: ASCII → 8 bits per character
- Save Space: variable length per character.
  Eg: Morse-Code. ↳ Shorter length for common characters.

Need a separator for variable length - Morse code uses a pause between letters → But wasted capacity

No encoding of a symbol is a prefix of another

Better Solution: Use a prefix code.

eg:

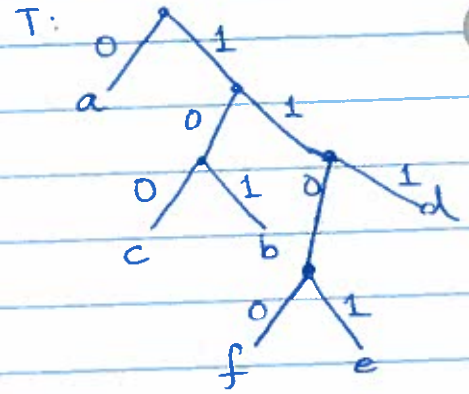
a	0
b	101
c	100
d	111
e	1101
f	1100

0 0 | 0 | 1 1 0 1
↓ ↓ ↓ ↓
a a b e

(No need of an explicit separator)

→ Can represent prefix code as a binary tree

Symbols = leaves



→ To use as little space as possible:

Want to find a prefix tree T to minimize:

$$ABL(T) = \sum_{x \in C} f(x) \cdot d_T(x)$$

$d_T(x)$ → depth of x in T = size of encoding x

ABL(T) → Average Bits per letter

$f(x)$ → frequency of x

$x \in C$ → set of symbols

→ Huff Algo: Given symbols & their frequencies, generate optimal prefix code.

optimal → minimiz ABL(T)

prefix code → tree T