Lecture : 1/15/2015

Today's Lecture

→ Tree characterizations.

→ Exhausting a graph.

1. Hamiltonian Cycles
2. Eulerian circuits.

→ Minimum Spanning Tree.

Kruskal + Proof.

Forest : A Graph that has no cycles.

Tree : A connected forest

A leaf of a tree

Proof : tree has atleast one leaf.

*Trees have atleast two leaves which is deg-one node.

Three important characteristics :

1. A connected graph
2. No cycles
3. $|E| = |V| - 1$ $(m = n-1)$

(1), (2) ⇒ 3

(1), (3) ⇒ 2. {Proof by contradiction}

2nd proof, remove one edge.

do this until no cycles are left.

(maintain connectivity)

COMPONENT: → Connected parts of a graph

(2), (3) ⇒ 1

Assume graph isn't connected. k components.

Each component:

$|E_i| = |V_i| - 1 \Rightarrow |E| = |V| - k$

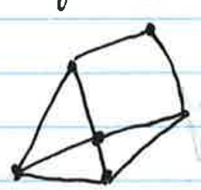
sum this up.

→ EXHAUSTING A GRAPH

visit them all in a connected

1. Hamiltonian: A simple cycle that visits ~~each~~ every node exactly once.

A cycle that



→ very hard to do

→ weighted version coming as TSP.

Hint for a5:

General Proof Technique: {Good for Exam}

If a graph has min. deg. condition, then to analyze cycles, it is helpful to look at the longest path.

$\delta(G) \geq 3$

claim: this graph has cycle of ∃ cycle of length atleast 4.

u & v must have all of their neighbors on the longest.

$\delta(G) \geq k$

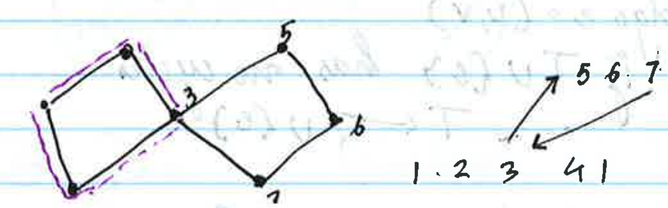
∃ a cycle of length atleast k+1

EULERIAN CIRCUIT: (might repeat nodes).

• visit each edge exactly once.

Eulerian circuits exist iff every node has an even degree and connected.

Proof:



1. Start somewhere and take unvisited edges until stuck with no unvisited edges.
2. There must exist some unvisited edge incident ~~with~~ to the current candidate circuit with same degree condn. Induct into that.

MINIMUM SPANNING TREE

→ Given weighted graph G, n nodes, m edges.
→ a spanning tree of G is a subgraph of G that is a tree with n nodes.

Spanning tree of minimum weight

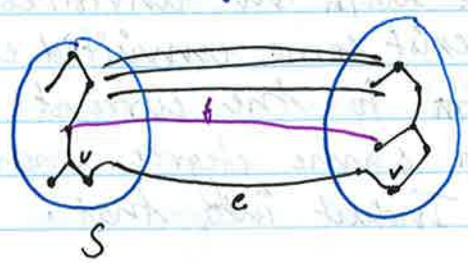
KRUSHKAL'S ALGORITHM: (Assume all weights are distinct)

1. First order the edges in increasing order.
2. Start ~~out with~~ with empty tree.
3. For each edge

E { }
for edge e = (u,v)
  if T ∪ {e} has no cycles.
    T ← T ∪ {e}.

THEOREM: The smallest edge crossing any cut must be in every MST

Proof:



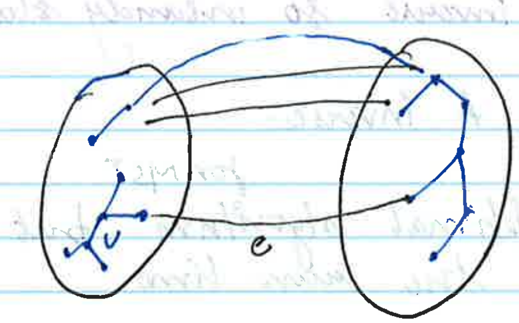
black stuff is MST

Assume f & e are different.

let e = (u,v) be smallest weight edge crossing cut.

{e} ∪ T has a cycle.

Follow path T from u to v, & replace edge that crosses cut with e.
⇒ get smaller spanning tree.
⇒ contradiction.

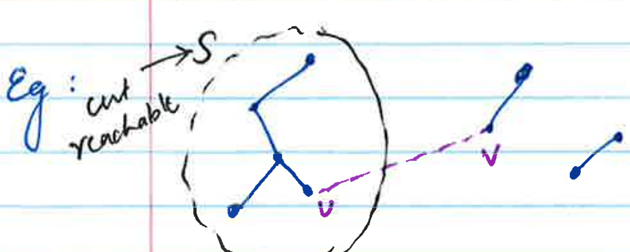


Let's look @ operation of Krushkal.

→ Only add edge e=(u,v) if does not create a cycle.
→ consider all the cut produced by all nodes reachable from u.

Claim: e smallest weight edge that leaves the cut. (s)

Eg:



∴ by Theorem it is in every MST.

RUN TIME ANALYSIS :-

→ dominated by sorting.

Naive way : Sorting $O(m \log(n))$ ✱.

→ Union find data structure $O(m\alpha(n))$ worst.
where $\alpha(n)$ inverse ackermen function.

$A(n) = 2^{2^{2^{\cdot^{\cdot^{\cdot}}}}}$ n

$\alpha(n)$ is inverse so insanely slow growing.

~~of~~ $\alpha(n)$ is A inverse.

→ We know the optimal algorithm for MST but we don't know the run time.

→ connect all these ideas to vector in R^n & spanning spaces.