EECS 310: Discrete Math **Lecture 7**
Graph Theory

Reading: MIT OpenCourseWare 6.042 Chapter 5.7

Review

Eulerian/Hamiltonian Tours/Paths

[[*Eulerian: "e" for edge, visit each edge exactly once; Hamiltonian: visit each vertex exactly once*]]

Claim: A connected graph has an Eulerian tour if and only if each vertex has even degree.

[[*Proof by taking arbitrary walk until get stuck (this will be a tour), then repeat process starting on edge adjacent to current tour, then splice together tours.*]]

Planar graphs

[[*Planarity: can be drawn in plane with no edge crossings; can be constructed recursively from single vertex by splitting faces and adding bridges*]]

Claim: Planar with n vertices, m edges, f faces has:

- $n - m + f = 2$

  [[*Proof by induction using recursive constructor.*]]

- $m \leq 3n - 6$

  [[*Proof by counting number of edges adjacent to faces in two ways, subbing into above formula.*]]

Trees

- Useful representation (e.g., decision trees, search trees, family trees)
- Many problems easy on trees (e.g., coloring)
- Graphs can be approximated by trees

Example: Draw a decision tree (e.g., stay home or go to school, if stay home watch t.v. or play computer games or do homework, etc.). Sometimes a tree is the natural result of the nature of the abstraction.

Formally,

Def: A *tree* is a connected graph with no cycles.

Some important properties of trees.

Claim: Every tree has the following properties:

1. Any connected subgraph of a tree is a tree
2. There is a unique simple path between any pair of vertices
3. Adding an edge between any pair of vertices creates a cycle

4. Removing any edge disconnects the graph

5. If there are at least 2 vertices, then there are at least 2 leaves

6. The number of vertices is one more than the number of edges

Proof: Of (1). By contrapositive. If H is a subgraph of G and H has a cycle. Since the edge set and node set of H is a subset of G, then that cycle also exists in G. Thus not H implies not G. So it must be that any subgraph of an acyclic graph is also acyclic. Then, if it is connected, it is a tree by definition.

□

Proof: Of (2). Suppose there are two different paths between u and v. Let x be the first place they diverge. Let y be the next place they meet. Then there are two disjoint subpaths between x and y which is a cycle. This contradicts the acyclic assumption. Thus there is only one path.

□

Proof: Of (3). There is a path between u and v already by (2). Adding in an edge to u and v will from the cycle of that path plus this new edge. □

Proof: Of (4). Suppose there is an edge between u and v. This edge constitutes a path. By (2) it is the only path connecting the two vertices, so removing it disconnects the graph.

□

Proof: Of (6).

- By induction: $P(n)$ = a tree on n vertices has $(n-1)$ edges.
- Base case $P(1)$: a tree on 1 vertex has 0 edges.
- Inductive hypothesis: $P(n)$
- Inductive step:
  - Consider tree on $P(n+1)$ vertices
  - Remove leaf. Why can we always remove a leaf? Because of definition 5.
  - By I.H., remaining tree has $(n-1)$ edges
  - Add back leaf and get $(n-1)+1$ edges

□

Things are easy on graphs. Like coloring (skip it time is short).

Proof: Via induction.

- Induction on n, number of nodes. Let P(n) be a tree of n nodes is 2-colorable.
- Base case: 2 nodes. Color each one a different color.
- Assume for P(m) for all $m \leq n$.
- Consider tree of n+1 vertices. By (5) there exists some leaf node. Remove that.
- There is now a graph on n vertices. Apply the IHOP.
- Add back in that leaf. Since it only has one edge, just color it the opposite of whatever it is connected to. Now we have a valid 2-coloring of the $n+1$ graph.
- done.

□

Def: A *spanning tree* of a graph G is a subgraph that's a tree and contains all vertices of G.

Claim: Every connected graph contains a spanning tree.

Proof: Let T be a connected subgraph of G with smallest # of edges.

- Show T acyclic by contradiction.
- Suppose T has a cycle $(v_0, v_1), \ldots, (v_k, v_0)$.
- Remove (v_k, v_0). For any x and y with path P in T using (v_k, v_0), reroute P through $(v_k, v_{k-1}), \ldots, (v_1, v_0)$.
- Hence T still connected with less edges.

□

Applications:

Consider a *weighted metric graph.*

- weights $w(v_1, v_2)$ on edges
- metric if weights satisfy triangle inequality:

  $$w(v_1, v_2) + w(v_2, v_3) \geq w(v_1, v_3)$$

  plus $w(v_1, v_2) = w(v_2, v_1)$ and $w(v_1, v_1) = 0$.

Example: cities of IL (draw sample)

- nodes are cities
- edges are roads (assume complete graph)
- weight of an edge is travel time

  NOTE: weights correspond to distance and so satisfy triangle inequality.

Question (TSP):

How to visit all cities exactly once in minimum time and return to start?

In math, find a minimum length Hamiltonian cycle

[[*Famous problem in CS, give TSP story, NP-hard in general, but can find a near-optimal solution quickly in metric graphs as above.*]]

Solution:

- Find spanning tree of minimum weight
- Double edges of tree
- Find Eulerian tour of tree ("depth-first search")

  [[*How do we know one exists?*]]
- Convert to Hamiltonian tour by "skipping" revisits

  [[*can do since graph complete*]]

Example: Draw process in IL graph

Claim: Takes at most double the time of an optimal tour.

Proof: Optimal tour is at least minimum spanning tree (why?) and skipping is actually shortcutting since metric graph. □

[[*Instance of an approximation algorithm, important CS topic.*]]