Notes 14

Those who joined the class late and didn't do psets: come tell me.

Graphs in this lecture are undirected unless explicitly stated otherwise.

Note: Although we have defined a path to be a sequence of edges, it is equivalent to consider the sequence of vertices (in order). We shift back and forth between these notations as convenient.

1 Cheap Connectivity

Suppose we want to wire n dorms together. What is the smallest number of wires that will do the job?

Ie, what graph is connected but has the fewest edges?

When is an edge unnecessary? When it is in a cycle.

Lemma 1.1 A set of n vertices can be put into a connected graph of $n-1$ edges ($n-1$ edges are sufficient).

Proof. Induction. Clear for 1. Assume the lemma for n. Consider $n+1$ vertices. Remove 1 vertex u, and hook up the rest using $n-1$ edges. Add an edge from v to u for n edges. Claim: this gives a connected graph.

Proof of connectivity: need to show all vertex pairs have a path. True by inductive hypothesis for all vertex pairs not including u. So consider a pair (v, w). By the IH, there is a path $v, x_1, \ldots, x_k, w$ in the smaller graph. Then $u, v, x_1, \ldots, x_k, w$ is a v-w path in the larger graph. ∎

Lemma 1.2 Any n vertex connected graph has at least $n-1$ edges ($n-1$ edges are necessary).

Proof. Well ordering. Consider a smallest counterexample, with n vertices and $m < n-1$ edges and $n+m$ minimum. Can't have a degree 0 vertex (disconnected). Suppose all vertices have degree at least 2. Then by the handshaking lemma, the number of edges is at least n.

So there is a degree 1 vertex u with (unique) neighbor v. Claim: deleting vertex u and edge (u, v) gives a smaller counterexample. The resulting graph has $n-1$ vertices,

$m-1 < (n-1)-1$ edges. It remains to prove that it is connected. Consider vertex pair x,y with $x, y \neq u$. These were originally connected by a path, thus by a simple path. Is vertex u on this simple path? No, because passing through u requires using (u, v) twice in succession, which would mean the path wasn't simple. The fact that u isn't on the path implies that the edge (u, v) also isn't on the path. So that path is still present after deleting u. So the smaller graph is connected. (A contradiction.) ■

(We actually implicitly proved a fact we'll use later: deleting a degree 1 vertex preserves connectivity.)

So, we can do $n-1$ edges and $n-1$ edges is the best we can do.

What kinds of graphs have this property (ie, connected with $n-1$ edges)?

Trees.

2 Trees

Important special type of graph for CS applications.

2.1 Definitions

Tree: Minimal Connected graph.

Trees can actually be defined 3 different ways:

1. A connected graph with no cycles (equivalently, a connected forest)
2. A connected graph that is disconnected by the removal of any edge, ie, a "minimal" connected graph.
3. A graph with a unique simple path between every pair of vertices

Forest: No cycles, but not necessarily connected.

In nontrivial trees (with more than one node), use terminology:
Leaf – node with only one neighbor.
Internal node – non-leaf

We will prove these 3 definition are equivalent.

To do so, prove $1 \rightarrow 2 \rightarrow 3 \rightarrow 1$. Equivalence follows by transitivity of implication.

Lemma 2.1 ($1 \to 2$) A connected graph with no cycles is disconnected by the removal of any edge.

Proof. By contradiction. Suppose we can delete edge (u, v) without disconnecting the graph. This means there is a path from u to v that doesn't use (u, v), and thus a simple path. Adding edge (u, v) to a simple path that doesn't use (u, v) creates a cycle. Contradicts claim of no cycles. ■

Lemma 2.2 ($3 \to 1$) A graph with a unique simple path between any pair of nodes is connected and has no cycles.

Proof. Connected obvious. No cycles by contradiction. Suppose we have a cycle $v_1, \ldots, v_k, v_1$. Then there are 2 simple paths from v_1 to v_k, namely $v_1, \ldots, v_k$ and the edge (v_1, v_k). ■

Lemma 2.3 ($2 \to 3$) A connected graph that is disconnected by the removal of any edge has a unique simple path between any pair of vertices.

Proof. Suppose there are two different simple u-v paths. Call one P_1: $u = x_0, x_1, \ldots, x_k = v$. Call the other P_2: $u = y_0, \ldots, y_l = v$. Let i be the smallest such that $x_i \neq y_i$ (why does this have to exist?). Claim: we can remove (y_{i-1}, y_i) without disconnecting G. To see this, consider a pair of vertices z and w. There's a simple path between them at the start. It remains to show there's still a path (doesn't need to be simple) after removing (y_{i-1}, y_i). We use proof by cases.

Case 1: (y_{i-1}, y_i) wasn't on the path. Then the path remains after removing (y_{i-1}, y_i).

Case 2: (y_{i-1}, y_i) was on the path. WOLOG, y_{i-1} is encountered first. So the path looked like $z, \ldots, y_{i-1}, y_i, \ldots, w$. So there's a different path: go from z to y_{i-1} on the first part of z-w path, then from y_{i-1} to u on the first part of P_2, then from u to v on P_1, then from v to y_i on the second part of P_2, then from y_i to w on the second part of the z-w path. ■

2.2 Spanning Trees and Forests

We've just argued that the "minimal" (poset, under edge containment) connected graphs are trees. It turns out that every connected graph actually contains one:

Definition 2.4 A *spanning tree* of a graph is a subset of the graph edges that forms a tree on all the graph nodes.

Notice that the notion "tree" is an intrinsic concept (cares only about the nodes adjacent to the edges of the tree), while "spanning tree" is extrinsic (cares about whether you have all the graph's nodes).

Lemma 2.5 Every connected graph contains a spanning tree.

Proof. Well ordering. Consider smallest counterexample. If cannot delete any edge and stay connected, it is a tree (3rd definition of tree). So delete an edge, get smaller counterexample. Contradiction. ■

How to find a spanning tree? Graph search.

Some graphs aren't connected; don't have spanning trees. Instead, have spanning *forests*:

Definition 2.6 A *forest* is any acyclic graph.

Lemma 2.7 Each connected component of a forest is a tree.

Proof. Each connected component of a forest is connected and acyclic. ■

Definition 2.8 A *spanning forest* for a graph G is a forest that connects every pair of vertices that were connected in G,

People take some liberties with "tree:" call a set of edges a tree if it connects its vertices and is acyclic. So we talk about a forest as being "a collection of trees" even though each tree is only on vertices of a connected component, not on all graph vertices.

In particular, a spanning forest G is just a collection of trees, one spanning each connected component of G.

2.3 Counting Tree Edges. Induction off Leaves

Definition 2.9 A *leaf* is a node of degree 1.

Lemma 2.10 Any non-trivial tree has a leaf.

Proof. Consider a longest simple path $v_1, \ldots, v_k$ (why does such exist?). Claim v_1 is a leaf. Proof: suppose not. Then has a second neighbor x. Is $x = v_i$? No, that would give a cycle. Thus $x, v_1, \ldots, v_k$ is a longer simple path, a contradiction. ■

Lemma 2.11 If we remove a leaf and its edge from a tree, we get a tree.

Note: many of you tried this on the exam. But didn't prove these lemmas. (That approach would have required proving other lemmas as well).

Proof. As shown earlier (in the proof of Lemma 1.2), deleting a degree one vertex (and its incident edge) preserves connectivity. It also doesn't create cycles. So what we get still has the properties of definition 1 of a tree. ■

Lemma 2.12 Any tree on n vertices has $n-1$ edges.

Proof. (Trivial base case). Inductive step: Start with an arbitrary tree with $n+1$ nodes. Remove one node and its incident edge. By the previous lemma, the result is a tree (with n nodes). By the IH, this tree has $n-1$ edges. So the original tree has n edges. ∎

Lemma 2.13 Any tree is 2-colorable.

Proof. (Trivial base case). Inductive step: Start with an arbitrary non-trivial tree. Remove one node and its incident edge. By Lemma 2.11, the result is a tree. 2-color it (possible by the IH). Replace the leaf and its edge. Color the replaced node the color different from its neighbor. All nodes are then colored. The coloring is valid because all adjacent pairs except across the replaced edge are still validly colored from when the smaller tree was 2-colored, and that pair is OK because of the way we chose the color for the node after we replaced it. ∎

3 Euler Tours

Konigsberg bridge problem: can we cross every bridge exactly once, without repeats, returning to start?

This problem has a graph representation.

Euler, 1736, proved NO. So we call such tours "Euler Tours."

Theorem 3.1 If undirected G has Euler tour, then G is connected and every vertex has even degree.

Proof. Direct every edge according to tour. We enter a vertex as many times as we leave it. So every vertex must have indegree equal to outdegree. So total even. ∎

100 years later, Hierholzer proved consequence (trouble with coming second—no one remembers your name!)

Theorem 3.2 If G is connected and has even degree, then it has an Euler tour.

Lemma 3.3 If G has even degree and at least one edge, then it has a circuit.

Proof.

If connected, handshaking proves n edges, so not a tree, so cycle. But this also applies to unconnected: consider a component that contains at least one edge and apply the above. (There must be such a component because every edge of the graph must be in one such component). ■

Proof. (Of main result) By contradiction.

Idea: Let C be the longest circuit without any repeated edges. Delete it. Delete all isolated vertices. Since C is not an Euler tour, some vertices remain. Some vertex u that remains is on C. Why? Because G is connected. Take any remaining vertex v, it has a path to a vertex in C. The first time we reach an edge of C, we stop. This gives u. Now, by the lemma, find a circuit containing u. Attach to C, giving a bigger circuit. (A contradiction.)
■

4 Hamiltonian Paths

A slightly different question: is there a tour that traverses every *vertex* exactly once?

Such a cycle is called a *Hamiltonian Cycle.*

In a small variant, a simple path that traverses every vertex exactly once is a *Hamiltonian Path.*

Small change, but suddenly, instead of answer being trivial, becomes very complicated. It is believed that no computer program can answer this question efficiently (it is NP-complete).