More Graphs
Admin stuffs

- Last homework due today
- Office hour changes starting Friday (also in Piazza)
  - Friday 6/17: 2-5 Mark in his office.
  - Sunday 6/19: 2-5 Jasmine in the UGLI.
  - Monday 6/20: 10-12 Mark in his office.
  - Monday 6/20: 5-7 Emily in the UGLI.
  - Tuesday 6/21: 10-12 Emily in the Beyster Learning Center.
  - Tuesday 6/21: 1-3 Mark in his office.
  - Wednesday 6/22: 10-12 Emily in the Beyster Learning Center.
  - Wednesday 6/22: 1:30-3 Mark in his office.
- Discussion is still on for Thursday and Friday.
- Exam is Thursday 6/23 from 4-6
  - Room information posted shortly (will be in EECS)
Last time…

- Defined some terms
  - \( G=(V,E) \)
  - Directed vs undirected graph
  - What it means to be connected
  - Etc.

- Did Dijkstra’s Algorithm
  - Finds shortest path between a pair of nodes
  - Didn’t analyze runtime though…

- Started on induction proof about connectivity.
Today

- Analyze Dijkstra’s Algorithm

- Deal with induction proof we started on last time.

- Look at a way of finding all-pairs shortest path distances
  - Floyd-Warshall Algorithm

- More terminology
  - Path, cycle, Eulerian path, Eulerian cycle, other graph applications

- (Time allowing) More terminology
  - Trees, minimum-spanning trees (MST), planar graphs
  - Brief overview of ideas associated with these things.
Let’s examine Dijkstra’s Algorithm

• What is the worst-case run time on a graph with $|V|$ nodes?
Dijkstra's algorithm.

- Maintain a set of **explored nodes** $S$ for which we have determined the shortest path distance $d(u)$ from $s$ to $u$.
- Initialize $S = \{ s \}$, $d(s) = 0$.
- Repeatedly choose unexplored node $v$ which minimizes

$$\pi(v) = \min_{e = (u,v) : u \in S} d(u) + \ell_e,$$

add $v$ to $S$, and set $d(v) = \pi(v)$.

shortest path to some $u$ in explored part, followed by a single edge $(u, v)$
Dijkstra's Shortest Path Algorithm

\[ S = \{ s, 2, 6 \} \]
\[ PQ = \{ 3, 4, 5, 7, t \} \]
Dijkstra's Shortest Path Algorithm

\[ S = \{ s, 2, 6 \} \]
\[ PQ = \{ 3, 4, 5, 7, t \} \]
Dijkstra's Shortest Path Algorithm

\[ S = \{ s, 2, 6, 7 \} \]
\[ PQ = \{ 3, 4, 5, t \} \]
Today

- Analyze Dijkstra’s Algorithm
- Deal with induction proof we started on last time.
- Look at a way of finding all-pairs shortest path distances
  - Floyd-Warshall Algorithm
- More terminology
  - Path, cycle, Eulerian path, Eulerian cycle
- More terminology
  - Trees, minimum-spanning trees (MST), planar graphs
  - Brief overview of ideas associated with these things.
Theorem 2  Every connected graph $G$ with $|V(G)| \geq 2$ has at least two vertices $x_1, x_2$ so that $G - x_i$ is connected for $i = 1, 2$.

Proof: We proceed by induction on $|V(G)|$. As a base case, observe that if $G$ is a connected graph with $|V(G)| = 2$, then both vertices of $G$ satisfy the required conclusion. For the inductive step, let $G$ be a connected graph with $|V(G)| \geq 2$ and assume that the theorem holds for every graph with $< |V(G)|$ vertices. If $G - x$ is connected for every vertex $x \in V(G)$, then we are done, so we may assume this is not so, and choose $x \in V(G)$ so that $G - x$ has components $H_1, H_2, \ldots, H_m$ where $m \geq 2$. For every $1 \leq i \leq m$ let $H'_i$ be the graph obtained from $H_i$ by adding back the vertex $x$ and all edges with one end $x$ and the other end in $V(H_i)$. So every $H'_i$ is a connected graph with at least two vertices. Furthermore, $|V(H'_i)| < |V(G)|$, so by induction, $H'_i$ must have at least one vertex $x_i \neq x$ so that $H'_i - x_i$ is connected. It then follows that $G - x_i$ is connected. Since we have such an $x_i$ for every component (and at least two components), this completes the proof.  \[\square\]
Today

- Analyze Dijkstra’s Algorithm

- Deal with induction proof we started on last time.

- Look at a way of finding all-pairs shortest path distances
  - Floyd-Warshall Algorithm

- More terminology
  - Path, Cycle, Eulerian path, Eulerian cycle

- More terminology
  - Trees, minimum-spanning trees (MST), planar graphs
  - Brief overview of ideas associated with these things.
Floyd-Warshall Algorithm

```
let dist be a |V| x |V| array of minimum distances initialized to \(\infty\) (infinity)
for each vertex \(v\)
    dist[\(v\)][\(v\)] \(\leftarrow\) 0
for each edge \((u,v)\)
    dist[\(u\)][\(v\)] \(\leftarrow\) \(w(u,v)\) // the weight of the edge \((u,v)\)
for \(k\) from 1 to |V|
    for \(i\) from 1 to |V|
        for \(j\) from 1 to |V|
            if dist[\(i\)][\(j\)] > dist[\(i\)][\(k\)] + dist[\(k\)][\(j\)]
                dist[\(i\)][\(j\)] \(\leftarrow\) dist[\(i\)][\(k\)] + dist[\(k\)][\(j\)]
end if
```

• What is it doing?

• Why does it work?
Floyd-Warshall Algorithm

```
let dist be a |V| × |V| array of minimum distances initialized to ∞ (infinity)
for each vertex v
    dist[v][v] ← 0
for each edge (u,v)
    dist[u][v] ← w(u,v) // the weight of the edge (u,v)
for k from 1 to |V|
    for i from 1 to |V|
        for j from 1 to |V|
            if dist[i][j] > dist[i][k] + dist[k][j]
                dist[i][j] ← dist[i][k] + dist[k][j]
            end if
```
Today

- Analyze Dijkstra’s Algorithm
- Deal with induction proof we started on last time.
- Look at a way of finding all-pairs shortest path distances
  - Floyd-Warshall Algorithm
- More terminology
  - Path, Cycle, Eulerian path, Eulerian cycle
- More terminology
  - Trees, minimum-spanning trees (MST), planar graphs
  - Brief overview of ideas associated with these things.
A graph \( G = (V,E) \) consists of a non-empty set \( V \) of vertices (or nodes), and a set \( E \) of edges. Each edge is associated with two vertices (possibly equal), which are its endpoints.

A path from \( u \) to \( v \) of length \( n \) is:

a sequence of edges \( e_1, e_2, \ldots, e_n \) in \( E \), such that there is a sequence of vertices \( u = v_0, v_1, \ldots, v_n = v \) such that each \( e_i \) has endpoints \( v_{i-1} \) and \( v_i \).

A path is a circuit when \( u = v \).

A graph is connected when there exists a path from every vertex to every other vertex.
Paths and Circuits

Given a graph $G = (V,E)$:

Is there a path/circuit that **crosses each edge** in $E$ **exactly once**?
If so, $G$ is an **Eulerian** graph,
and you have an Eulerian path, or an Eulerian circuit.

Is there a path/circuit that **visits each vertex** in $V$ **exactly once**?
If so, $G$ is a **Hamiltonian** graph,
and you have a Hamiltonian path or circuit.
Leonhard Euler lived in Königsberg

He liked to take walks. A famous local puzzle was whether you could take a walk that would cross each bridge exactly once.

Euler solved this problem (in 1736) and thus founded graph theory.
The Graph Abstraction

In the physical world:
   Each bridge connects exactly two land-masses.
   Each land-mass may have any number of bridges.

Suggests a graph abstraction:
   Represent a bridge by an *edge* in the graph.
   Represent a land-mass by a *vertex*.
Is there an Euler circuit/path?

Circuit

Path

Not Traversable
Does G have an Euler Path/Circuit?

Theorem: A connected multigraph has an Euler path iff it has exactly zero or two vertices of odd degree.

Why?

What if it has only one?

When does it have an Euler circuit?
Do these graphs have Euler paths/circuits?

(A) Yes, it has a circuit.
(B) Yes, it has a path (but no circuit).
(C) No, it has neither path nor circuit.
Does this have an Euler path?

In his article on the Königsberg bridges, Euler considered another bridge problem, which is illustrated below.

Two islands, A and B, are surrounded by water that leads to four rivers. Fifteen bridges cross the rivers and the water surrounding the islands. Is it possible to make a trip that crosses each bridge exactly once?

(A) Yes      (B) No
Applications of Euler Paths

Planning routes through graphs that provide efficient coverage of the edges in the graph, without multiple traversals.

Postal delivery routes
Snowplowing routes
Testing network connections
  Utility transmission network
  Communication network
Eulerian Path vs. Hamiltonian Path

- **Eulerian:**
  - Traverse each edge exactly once

- **Hamiltonian**
  - Traverse each node exactly once
Hamiltonian Paths and Circuits

Given a graph, is there a path that passes through each vertex in the graph exactly once? (aka a Hamiltonian path)

dodecahedron  Isomorphism graph  solution
Hamilton circuit

Do these graphs have Hamilton circuits?

(A) Yes, both.  
(B) G does, H doesn’t.  
(C) G doesn’t, H does.  
(D) No, neither.
Deciding whether a graph is **Eulerian** is a simple examination of the degrees of the vertices.

Simple linear-time algorithms exist for finding the path or circuit.

Deciding whether a graph is **Hamiltonian** is, in general, much more difficult (“NP complete”). Finding a Hamiltonian path or circuit is equally hard.
A weighted graph has numbers (weights) assigned to each edge.

The length of a path is the sum of the weights in the path.
Traveling Salesperson’s Problem

What is the shortest route in a given map for a salesperson to visit every city exactly once and return home at the end?

Mathematically:
Given an graph with weighted edges, what is the Hamiltonian circuit of least weight?
Applies to both directed and undirected graphs.
The Traveling Salesman Problem

Problem: Given a graph G, find the shortest possible length for a Hamilton circuit.

What is the solution to TSP for the following graph?

(A) 16  (B) 17  (C) 18  (D) 19  (E) 20
Traveling Salesperson’s Problem

The TSP is NP Complete: intractable in general.

Very large instances have been solved with heuristics.

An instance with 13,509 cities, solved in 2003.
Applications

The Traveling Salesperson’s Problem and the Hamilton Path/Circuit Problem have many applications.

- Planning and logistics (of course!)
- Manufacture of microchips
- DNA sequencing