CSCI 136:
Data Structures
and
Advanced Programming

Lecture 26

Graphs, part 1

Instructor: Dan Barowy

Williams

Outline

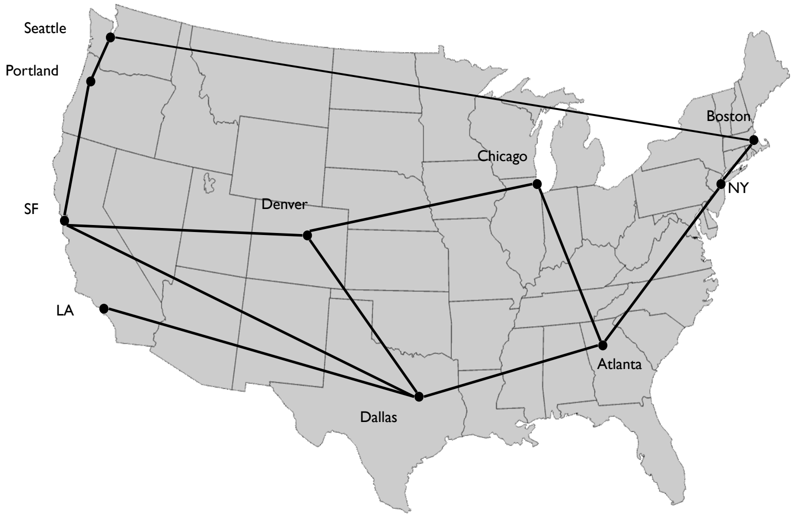
Graph definitions

Graphs

Tons of Applications

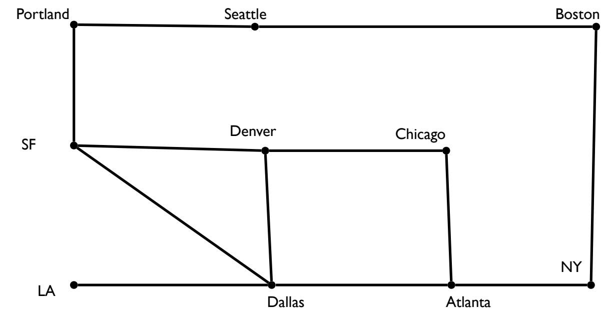
Nodes = subway stops; Edges = track between stops

Tons of Applications



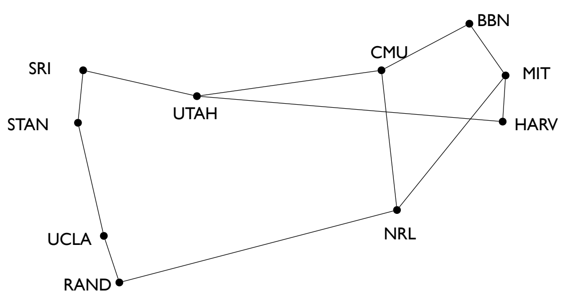
Nodes = cities; Edges = rail lines connecting cities

Tons of Applications



Note: A connection in a graph matters, but not the location of a node.

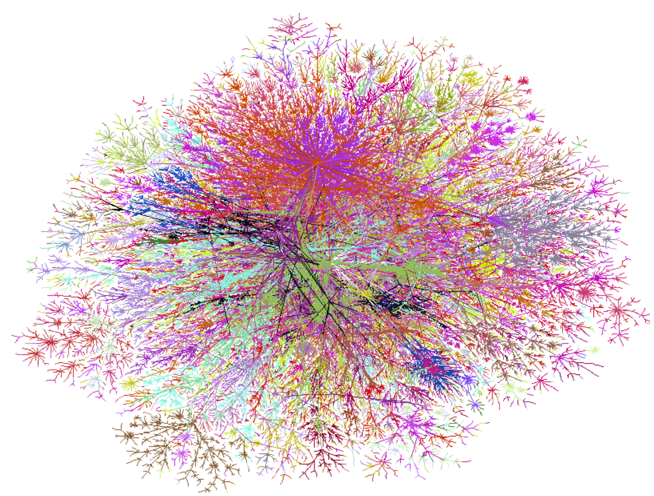
Tons of Applications



Any guesses as to what this is?

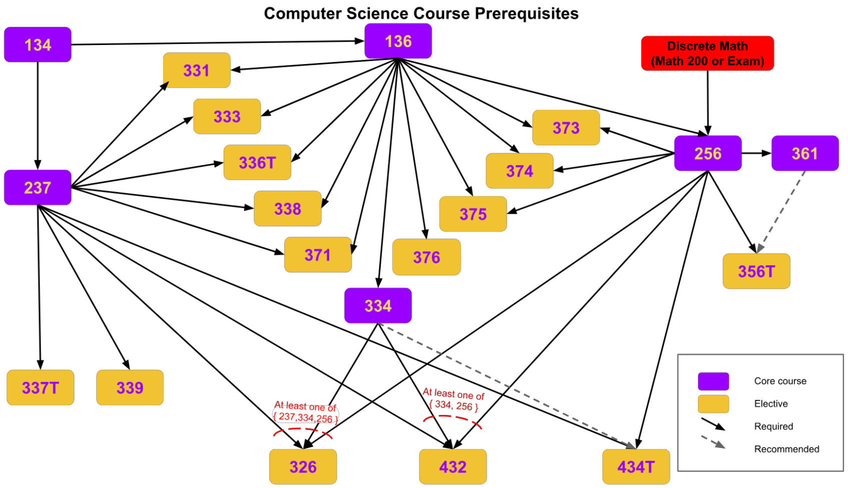
(The Internet, circa 1972.)

Tons of Applications

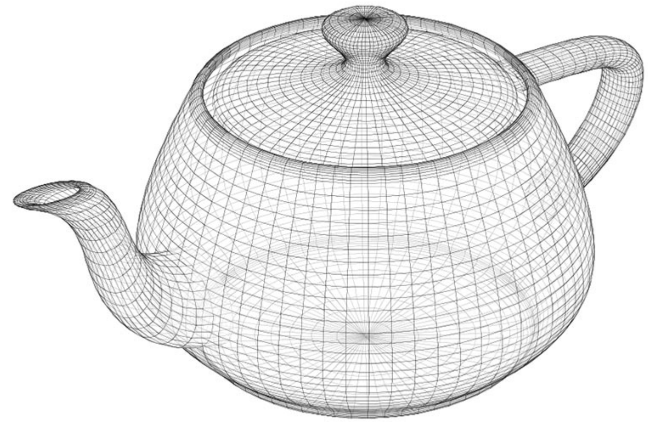


(The Internet, circa 1998.)

Tons of Applications



Tons of Applications



A "wireframe" model

Undirected graph

An **undirected graph G** is an abstract data type that consists of two sets:

- a set **V** of **vertices** (or **nodes**), and
- a set **E** of **undirected edges**.

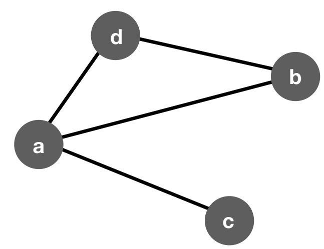
A graph can be used to represent any structure in which pairs of elements are, in some sense, "related."

In an undirected graph, data can be associated either with a vertex, an edge, or both.

Example: vertex data = city; edge data = distance.

Undirected edges make sense here because the distance from Williamstown to Boston is the same as the distance from Boston to Williamstown.

Undirected graph



G = (V, E)

Directed graph

A **directed graph G** is an abstract data type that consists of two sets:

- a set **V** of **vertices** (or **nodes**), and
- a set **E** of **directed edges**.

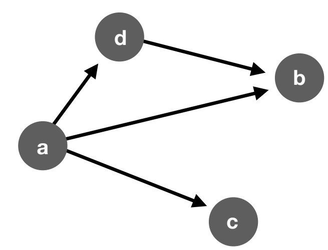
A directed graph can be used to represent any structure in which pairs of elements are "one-way related."

In a directed graph, data can be associated either with a vertex, an edge, or both.

Example: vertex data = people; edge data = "loves".

Directed edges make sense here because… unrequited love. See (countless) examples from popular culture.

Directed graph



G = (V, E)

Walking a graph

A **walk from u to v** in a graph **G = (V, E)** is an alternating sequence of vertices and edges

$$u = v_0, e_1, v_1, e_2, v_2, \ldots, v_{k-1}, e_k, v_k = v$$

such that $e_i = \{v_i, v_{i+1}\}$ for $i = 1, \ldots, k$

- A walk **starts** and **ends** with a **vertex**.
- A walk can travel over any edge and any vertex any number of times.
- If **no edge** appears more than once, the walk is a **path**.
- If **no vertex** appears more than once, the walk is a **simple path**.

Walking in circles

A **closed walk** in a graph **G = (V, E)** is a walk

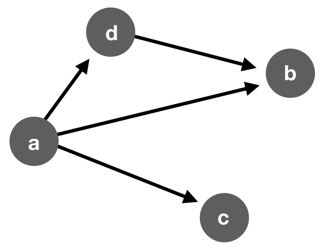
$$v_0, e_1, v_1, e_2, v_2, \ldots, v_{k-1}, e_k, v_k$$

such that each $v_0 = v_k$

- A **circuit** is a **path** where $v_0 = v_k$ (no repeated edges)
- A **cycle** is a **simple path** where $v_0 = v_k$ (no repeated vertices except v_0)
- The **length** of a walk is the number of edges in the sequence.

Walking on graphs vs digraphs

In a **directed graph**, a walk can only follow the **direction of the arrows**.



There is **no directed walk** from **b** to **a**.

Useful theorems

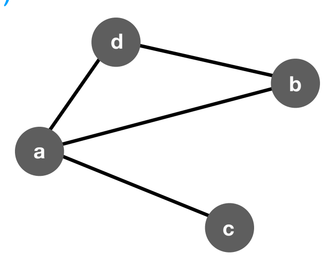
(about undirected graphs)

- If there is a **walk** from **u** to **v**, then there is a **walk** from **v** to **u**.
- If there is a **walk** from **u** to **v**, then there is a **path** from **u** to **v** (and from **v** to **u**).
- If there is a **path** from **u** to **v**, then there is a **simple path** from **u** to **v** (and **v** to **u**).
- Every **circuit** through **v** contains a **cycle** through **v**.
- Not every **closed walk** through **v** contains a **cycle** through **v**.

Degree

The **degree** of a vertex **v** is the **number of edges** incident to **v**.

Denoted: `deg(v)`

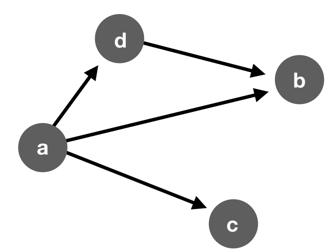


What is the degree of **c**? of **a**?

Degree on Digraphs

The **in-degree** of a vertex **v** is the **number of incoming edges** incident to **v**.

Denoted: `in-deg(v)`

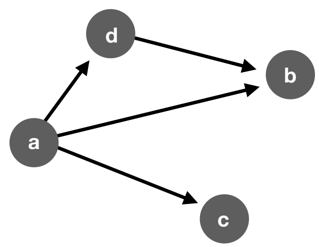


What is the in-degree of **c**? of **a**?

Degree on Digraphs

The **out-degree** of a vertex **v** is the **number of outgoing edges** incident to **v**.

Denoted: `out-deg(v)`



What is the out-degree of **c**? of **a**?

Degree theorem

For any graph **G = (V, E)**

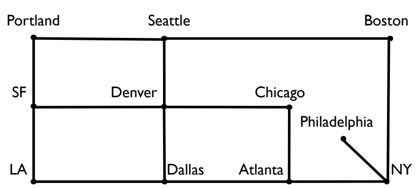
$$\sum_{v \in V} \deg(v) = 2\,|E|$$

where **|E|** is the number of edges in **G**.

Proof: by induction on **|E|**.

Hint: How does **removing an edge** change the equation?

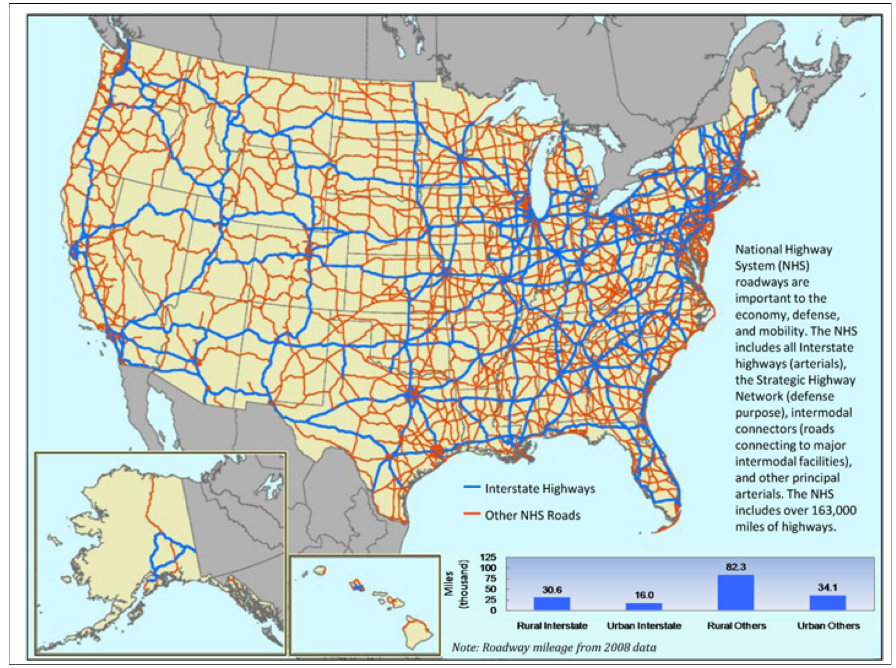
Activity

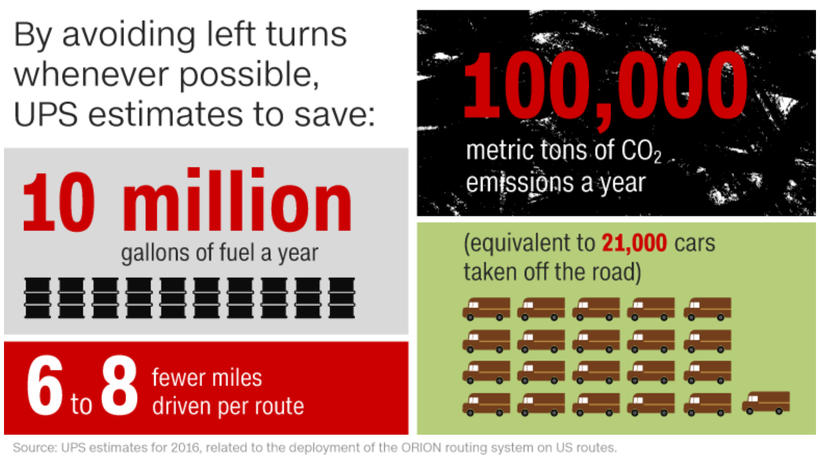


Walk:
ex:
Path:
ex:
Simple path:
ex:
Closed Walk:
ex:
Circuit:
ex:
Cycle:
ex:

Degree:
Max Degree Vertex:
Min Degree Vertex:

Recall the example from our first class





A study on crash factors in intersection-related accidents from the US National Highway Traffic Safety Association shows that turning left is one of the leading "critical pre-crash events" … About 61 percent of crashes that occur while turning or crossing an intersection involve left turns, as opposed to just 3.1 percent involving right turns.

source: cnn.com

Finding Shortest Paths

Data: road segments

road segment: (source, destination, length)

Input: source, destination

Output: shortest path

path: (segment$_1$, …, segment$_n$)

The Algorithm: Dijkstra's Algorithm

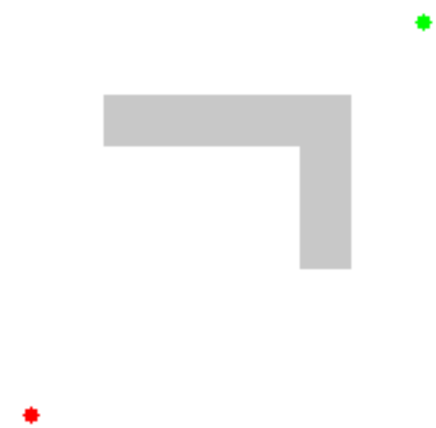
Data structures:

graph: essential representation of a "road network"

priority queue: ordered set of next roads to try

also uses: lists, arrays, stacks, …

Dijkstra's Algorithm



Recap & Next Class

Today we learned:

Graph definitions

Next class:

Graph ADT operations

Graph representations

Interesting graph problems