Graphs, Topological Sort, and Traversals

CSE 332 Spring 2021

Instructor: Hannah C. Tang

Teaching Assistants:

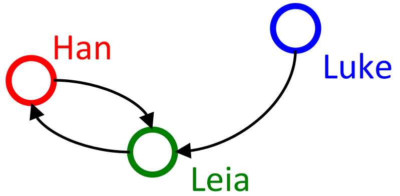
Aayushi Modi	Khushi Chaudhari	Patrick Murphy
Aashna Sheth	Kris Wong	Richard Jiang
Frederick Huyan	Logan Milandin	Winston Jodjana
Hamsa Shankar	Nachiket Karmarkar	

Graphs

- A ***graph*** is represents relationships among items
  - Very general definition because very general concept
- A ***graph*** is a pair: **G = (V, E)**
  - A set of ***vertices***, also known as ***nodes***

    $\mathbf{V = \{v_1, v_2, \ldots, v_n\}}$
  - A set of ***edges***, possibly ***directed***

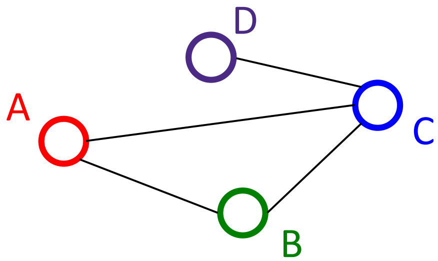
    $\mathbf{E = \{e_1, e_2, \ldots, e_m\}}$
    - Each edge $\mathbf{e_i}$ is a pair of vertices $\mathbf{(v_j, v_k)}$
    - An edge "connects" the vertices




```
V = {Han, Leia, Luke}
E = {(Luke, Leia),
     (Han, Leia),
     (Leia, Han)}
```

Undirected Graphs

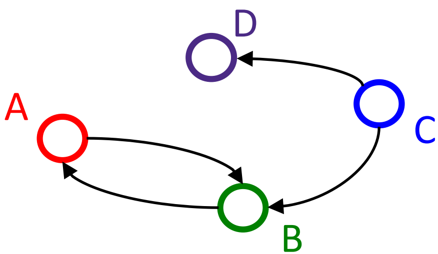
- In ***undirected graphs***, edges have no specific direction
  - Edges are always "two-way"



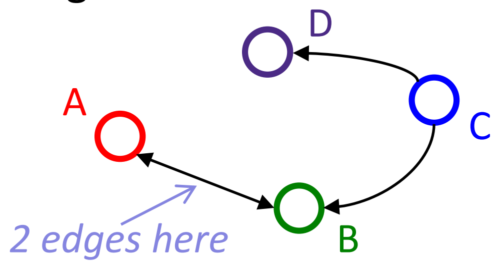
- Thus, `(u,v) ∈ E` implies `(v,u) ∈ E`
  - Only one of these edges needs to be in the set; the other is implicit
- ***Degree*** of a vertex: number of edges containing that vertex
  - i.e.: the number of adjacent vertices

Directed Graphs

- In ***directed graphs*** (aka ***digraphs***), edges have a ***direction***



or



- Thus, `(u,v) ∈ E` does *not* imply `(v,u) ∈ E`
  - `(u,v) ∈ E` means `u → v`; u is the ***source*** and `v` the ***destination***

- ***In-Degree*** of a vertex: number of in-bound edges
  - i.e.: edges where the vertex is the destination
- ***Out-Degree*** of a vertex: number of out-bound edges
  - i.e.: edges where the vertex is the source

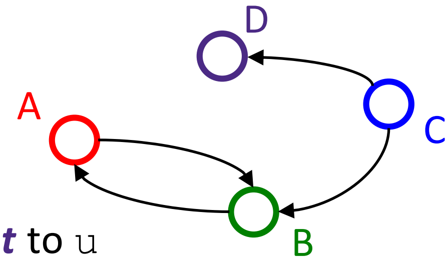
Self-edges

- A ***self-edge*** (aka a ***loop***) is an edge of the form **`(u,u)`**

- Depending on the use/algorithm, a graph may have:
  - No self edges
  - Some self edges
  - All self edges (therefore often implicit, but we will be explicit)

- A node can have a degree / in-degree / out-degree of zero

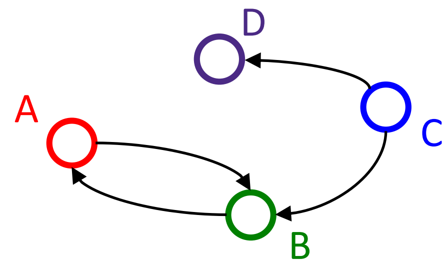
Adjacency (1 of 2)



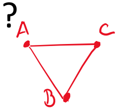
- If `(u,v) ∈ E`
  - Then `v` is a ***neighbor*** of `u`, i.e., `v` is ***adjacent*** to `u`
  - For directed edges, order matters
    - `u` is not adjacent to `v` unless `(v,u) ∈ E`

```
V = {A, B, C, D}
E = { (C, B),
      (A, B),
      (B, A)
      (C, D) }
```

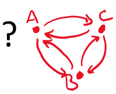
Adjacency (2 of 2)



- For a graph `G = (V,E)`:
  - `|V|` is the number of vertices
  - `|E|` is the number of edges
    - Minimum size?
      - *0*
    - Maximum size for an undirected graph with no self-edges?
      - `|V||V-1|/2 ∈ O(|V|²)`



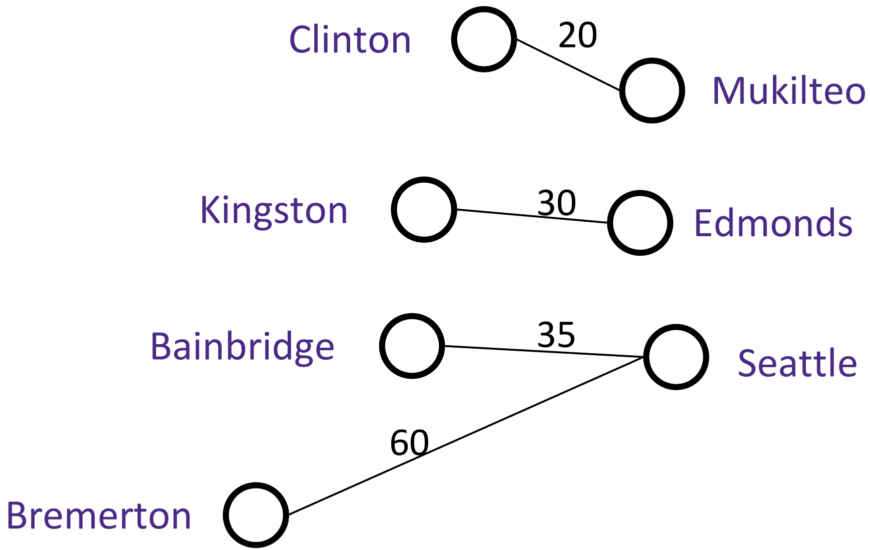
    - Maximum for a directed graph with no self-edges?
      - `|V||V-1| ∈ O(|V|²)`



    - If self-edges are allowed, add `|V|` to the answers above (applies to both undirected and directed graphs)

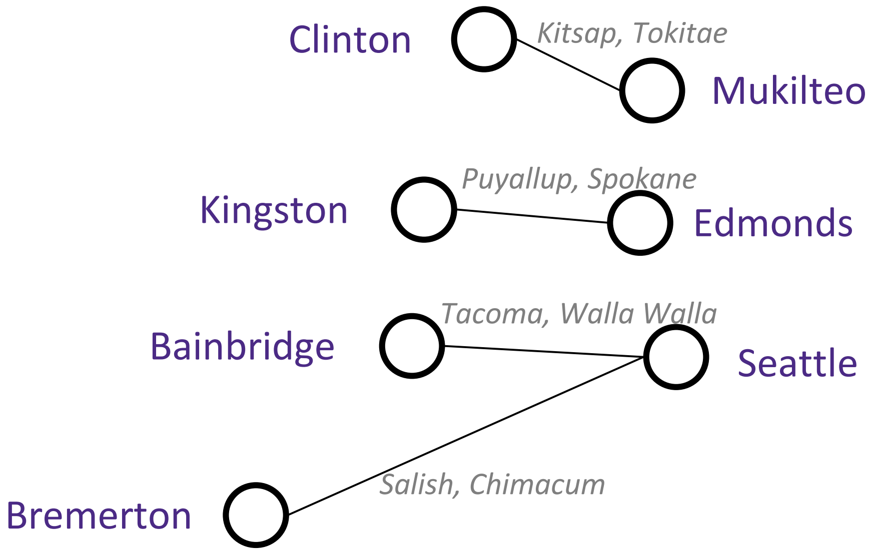
Weighted Graphs

- In a weighed graph, each edge has a ***weight*** a.k.a. ***cost***
  - Typically numeric (most examples will use ints)
  - Some graphs allow *negative weights*; many don't



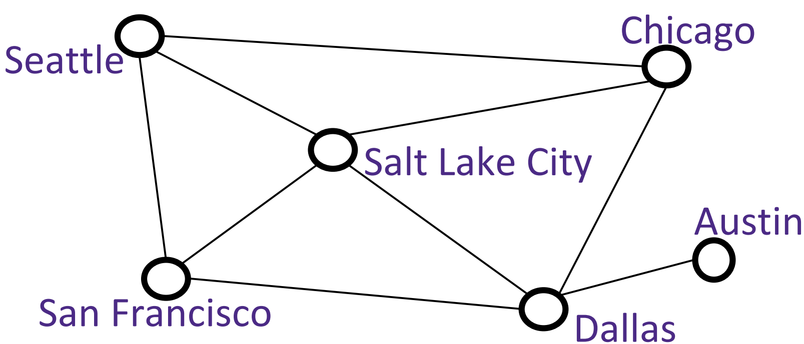
Vertex and Edge Labels

- More generally, both vertices and edges can have (possibly non-numeric) labels



Paths and Cycles (1 of 2)

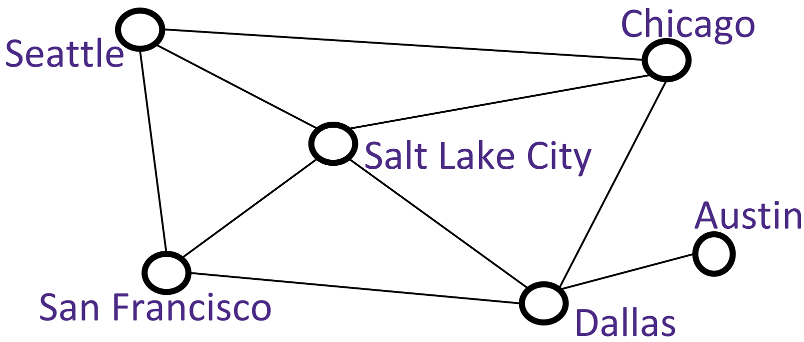
- A ***path*** is a list of vertices $[v_0, v_1, \ldots, v_n]$ such that $(v_i, v_{i+1}) \in E$ for all $0 \leq i < n$
  - You'd call it a path from v_0 to v_n
- A ***cycle*** is a path that begins and ends at the same node
  - i.e., $v_0 == v_n$

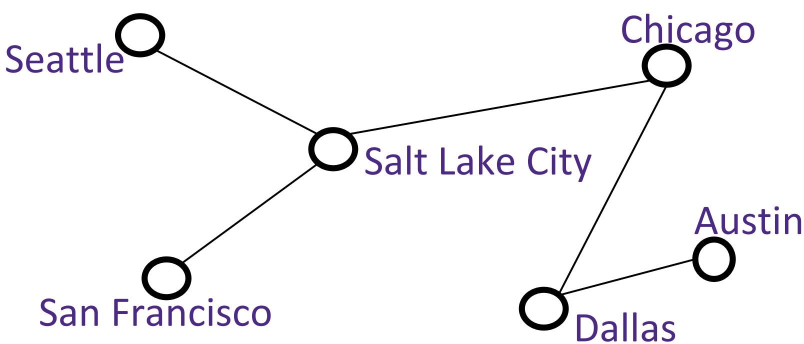


- Example path:
  - [Seattle, SLC, Chicago, Dallas, SF, Seattle]
  - Also happens to be a cycle!

Paths and Cycles (2 of 2)

- A graph that does not contain any cycles is ***acyclic***

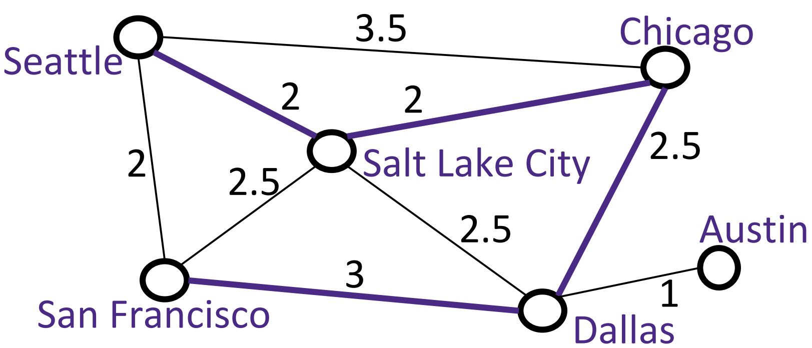




Path Length and Cost

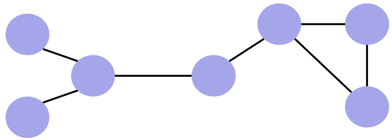
- ***Path length***: *Number* of edges in a path
  - Also called "unweighted cost"
- ***Path cost***: *Sum of the weights* of each edge in a path

- Example: P = [Seattle, SLC, Chicago, Dallas, SF]
  - length(P) = 4
  - cost(P) = 9.5

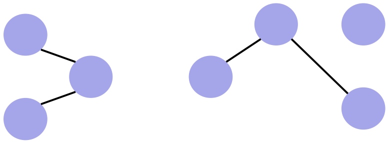


Undirected Graph Connectivity

- An undirected graph is ***connected*** if for all pairs of vertices **`u,v`**, there exists a *path* from **`u`** to **`v`**

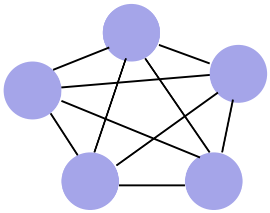


Connected graph



Disconnected graph

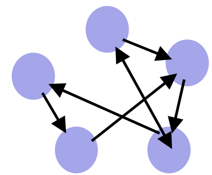
- An undirected graph is ***complete*** (aka ***fully connected***) if for all pairs of vertices **`u,v`**, there exists an *edge* from **`u`** to **`v`**



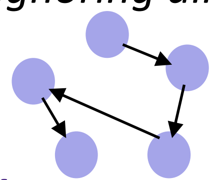
(not pictured: self edges)

<u>Directed</u> Graph Connectivity

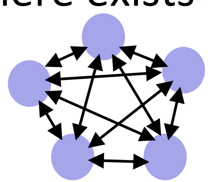
- A directed graph is ***strongly connected*** if for all pairs of vertices **`u,v`**, there exists a *path* from **`u`** to **`v`**



- A directed graph is ***weakly connected*** if for all pairs of vertices **`u,v`**, there exists a path from **`u`** to **`v`** *ignoring direction of edges*



- A directed graph is ***complete*** (aka ***fully connected***) if for all pairs of vertices **`u,v`**, there exists an *edge* from **`u`** to **`v`**

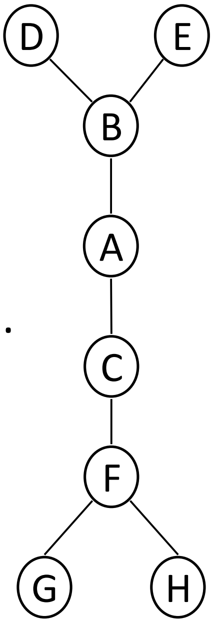


(not pictured: self edges)

Trees as Graphs

- A ***tree*** is a graph that is:
  - acyclic
  - connected

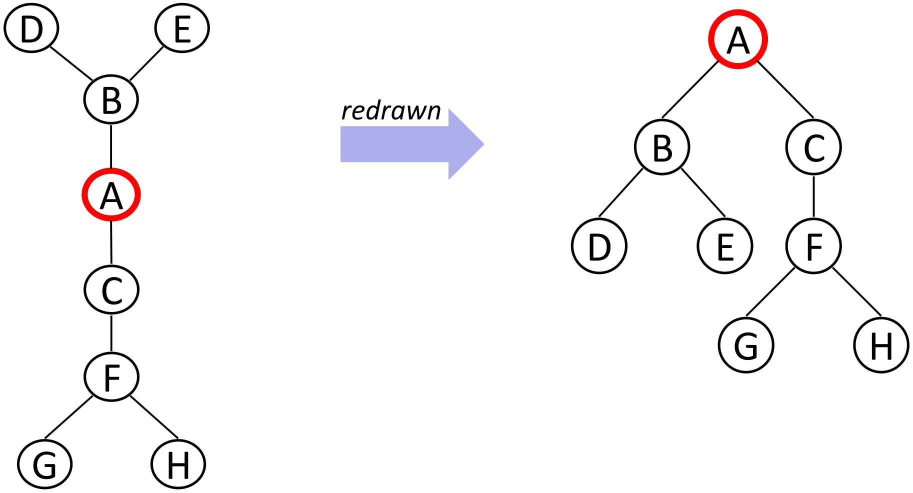
Example:



- So all trees are graphs, but not all graphs are trees
- How does this relate to the trees we know and love?...

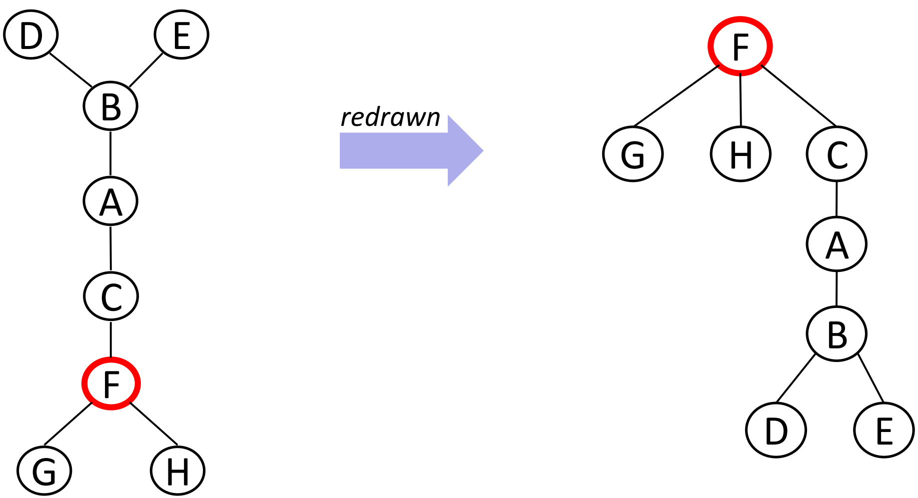
Rooted Trees (1 of 2)

- We've previously worked with ***rooted trees***, where:
  - We identify a unique ("special") vertex: the root
  - We think of edges as **directed**: parent to children
- The same tree can be redrawn as multiple rooted trees depending on which node you pick as the root



Rooted Trees (2 of 2)

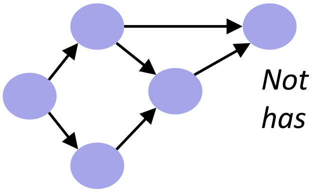
- We've previously worked with ***rooted trees***, where:
  - We identify a unique ("special") vertex: the root
  - We think of edges as **directed**: parent to children
- The same tree can be redrawn as multiple rooted trees depending on which *node you pick as the root*



Directed Acyclic Graphs (aka DAGs)

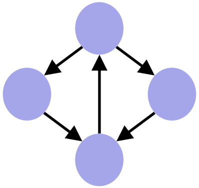
- A ***DAG*** is a directed graph with no directed cycles

- Every rooted directed tree is a DAG
  - But not every DAG is a rooted directed tree:



Not a rooted directed tree; has an undirected cycle

- Every DAG is a directed graph (by definition!)
  - But not every directed graph is a DAG:



Not a DAG; has a directed cycle

Density / Sparsity (1 of 2)

- *Recall*:
  - In an undirected graph, $0 \le |E| < |V|^2$
  - In a directed graph: $0 \le |E| \le |V|^2$

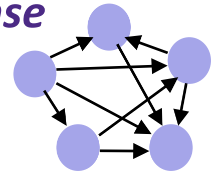
So for any graph, $|E| \in O(|V|^2)$

- One more fact:
  - In a *connected* undirected graph, $|E| \ge |V|-1$
  - In a *weakly connected* directed graph, $|E| \ge |V|-1$
  - In a *strongly connected* directed graph, $|E| \ge |V|$

So for any *connected* graph, $|E| \in \Omega(|V|)$

Density / Sparsity (2 of 2)

- We do not always approximate as $|E|$ as $O(|V|^2)$
  - This is a *correct* bound, it's just oftentimes not *tight*
- If it is tight, i.e. $|E| \in \Theta(|V|^2)$, we say the graph is ***dense***
  - Intuitively: "lots of edges"



- If $|E| \in O(|V|)$ we say the graph is ***sparse***
  - Sparse: "most (of the possible) edges missing"