CS483-09 Elementary Graph Algorithms

Instructor: Fei Li

Room 443 ST II

Office hours: Tue. & Thur. 1:30pm - 2:30pm or by appointments

`lifei@cs.gmu.edu` with subject: CS483

`http://www.cs.gmu.edu/~ lifei/teaching/cs483_fall07/`

Based on “Introduction to Algorithms” by T. Cormen, C. Leiserson, R. Rivest, and C. Stein and “Algorithms” by S. Dasgupta, C. Papadimitriou, and U. Vazirani.

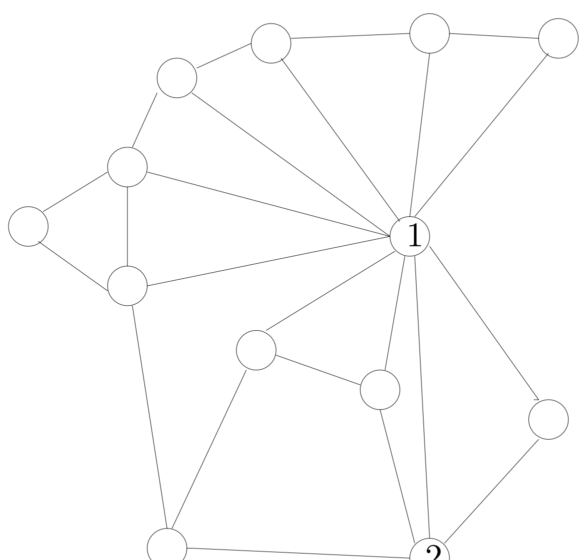
Outline

- Representation of Graphs
- Breath-first Search
- Depth-first Search
- Topological Sort

Why Graphs?

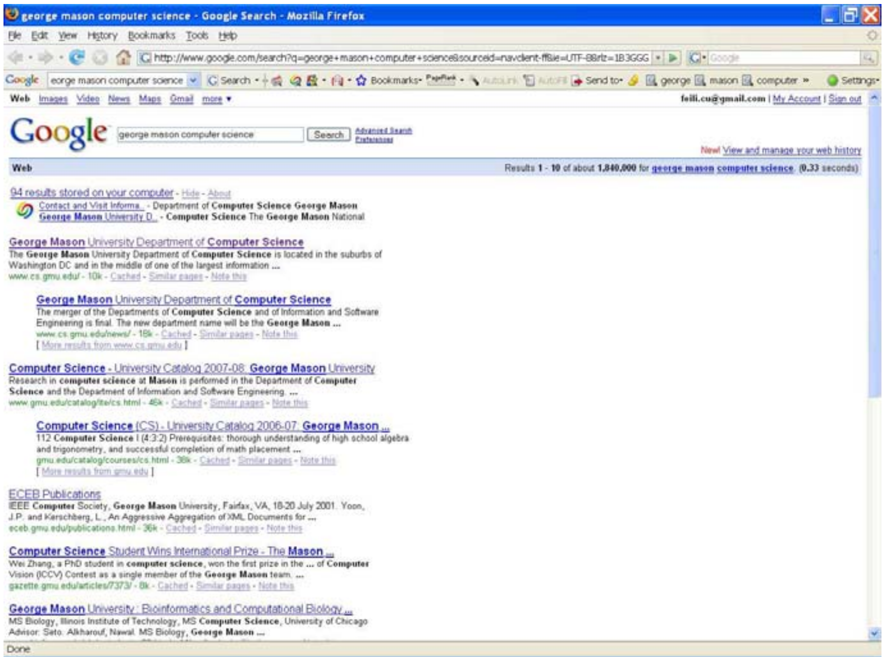
http://www.transitionsabroad.com/images/maps/south_america_map.gif

Why Graphs?



1 Brazil
2 Argentina

Why Graphs?



Why Graphs?

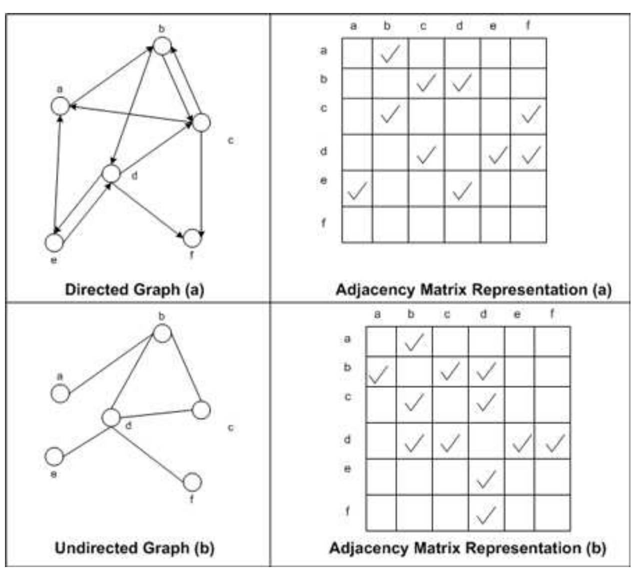
http://adcentered.typepad.com/photos

Graphs

- A graph $G = (V, E)$ is specified by a set of vertices (nodes) V and edges E between selected pairs of vertices.
- Edges are symmetric — *undirected graph*
- Directions over edges — *directed graph*
- Examples: political maps, exam conflicts, World Wide Web, etc.

Graph Representation

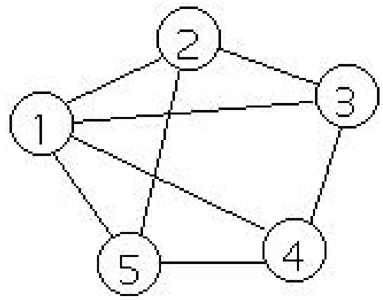
- Adjacency-matrix

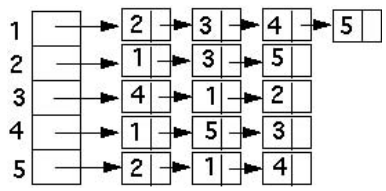


http://msdn2.microsoft.com/en-us/library/

Graph Representation

- Adjacency-list





Graph Traversal is Important

Exploring a graph is rather like navigating a maze.

Which parts of the graph are reachable from a given vetex?

`http://www.sheffordtown.co.uk/maze/index.html`

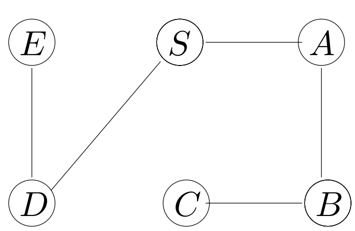
Outline

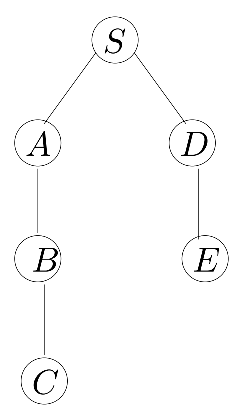
- Representation of Graphs
- Breath-first Search
- Depth-first Search
- Topological Sort

Breath-first Search (BFS)

BFS

1. Identifies all the vertices of a graph that can be reached from a designated starting point, and
2. Finds explict paths via a depth-first search tree.





Breath-first Search (BFS)

Input: Graph $G = (V, E)$, directed or undirected; vertex $s \in V$

Output: For all vertices u reachable from s, $d(u)$ is set to the distance from s to u

Intuition: Proceed layer by layer

Algorithm 0.1: BFS(G, s)

for $\forall u \in V$

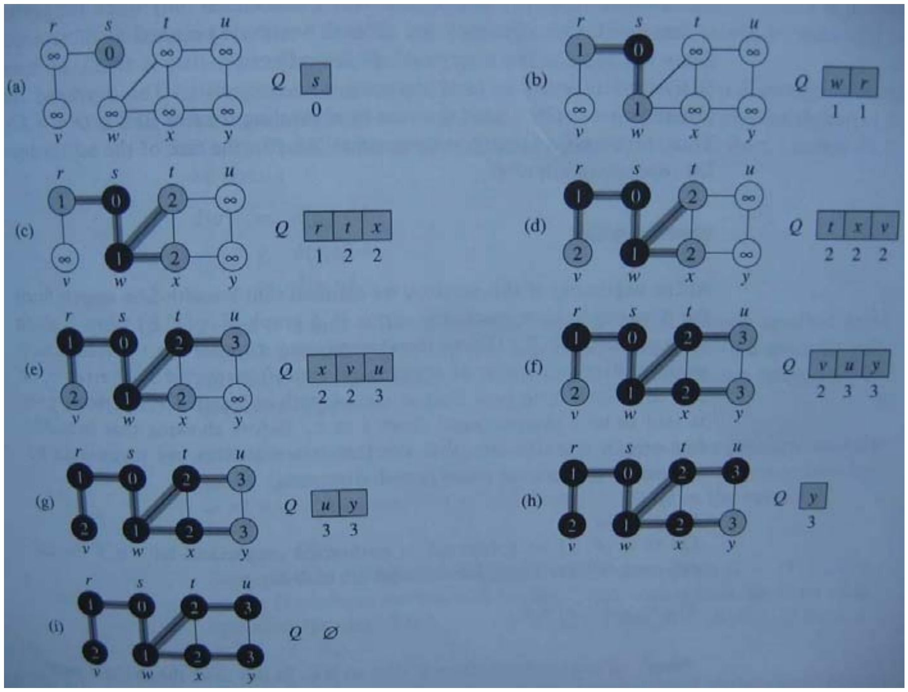
$\quad d(u) = \infty$

$d(s) = 0$

$Q = [s]$

while $Q \neq \emptyset$

$$\begin{cases} u = \text{Pop } (Q) \\ \textbf{for } (u, v) \in E \\ \begin{cases} \textbf{if } d(v) = \infty \\ \textbf{then } \begin{cases} \text{Push } (Q, v) \\ d(v) = d(u) + 1 \end{cases} \end{cases} \end{cases}$$



Breath-first Search (BFS)

- The correctness proof: Use an induction method
- The overall running time of BFS is $O(|V| + |E|)$.
  - Each vertex is put on the queue exactly once, when it is first encountered, so there are $2 \cdot |V|$ queue operations.
  - Over the course of execution, this loop looks at each edge once (in directed graphs) or twice (in undirected graphs), and therefore takes $O(|E|)$ time.

Outline

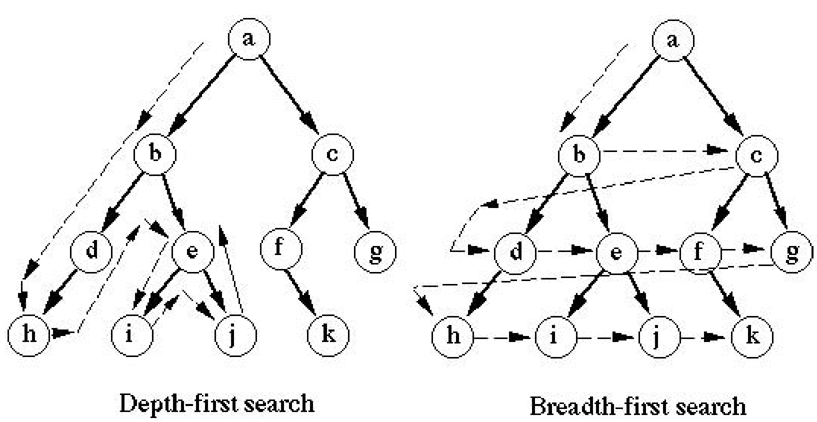
- Representation of Graphs
- Breath-first Search
- Depth-first Search
- Topological Sort

Depth-first Search

Input: Graph $G = (V, E)$, directed or undirected; vertex $s \in V$

Output: All vertices u reachable from s in timestamps of visiting

Intuition: Explore each vertex as much as you can



http://www.cse.unsw.edu.au/ billw/Justsearch1.gif

Depth-first Search (DFS)

$\pi[u]$: the parent of a node u.

$time[u]$: timestamp when u is first discovered.

Algorithm 0.2: DFS($G(V, E)$)

for each vertex $u \in V(G)$

do color$[u] \leftarrow$ WHITE

$\pi[u] \leftarrow$ NIL

time $\leftarrow 0$

for each vertex $u \in V(G)$

do if color$[u] =$ WHITE

then DFS-VISIT(u)

Algorithm 0.3: DFS-VISIT(u)

color$[u]$ ← GRAY

//White vertex u has just been discovered.

$d[u]$ ← time ← time $+ 1$

for each $v \in$ Adj$[u]$

//Explore edge (u, v).

do if color$[v]$ = WHITE

then $\pi(u) \leftarrow u$

DFS-VISIT(v)

color$[u]$ ← BLACK

//Blacken u; it is finished.

$d[u]$ ← time ← time $+ 1$