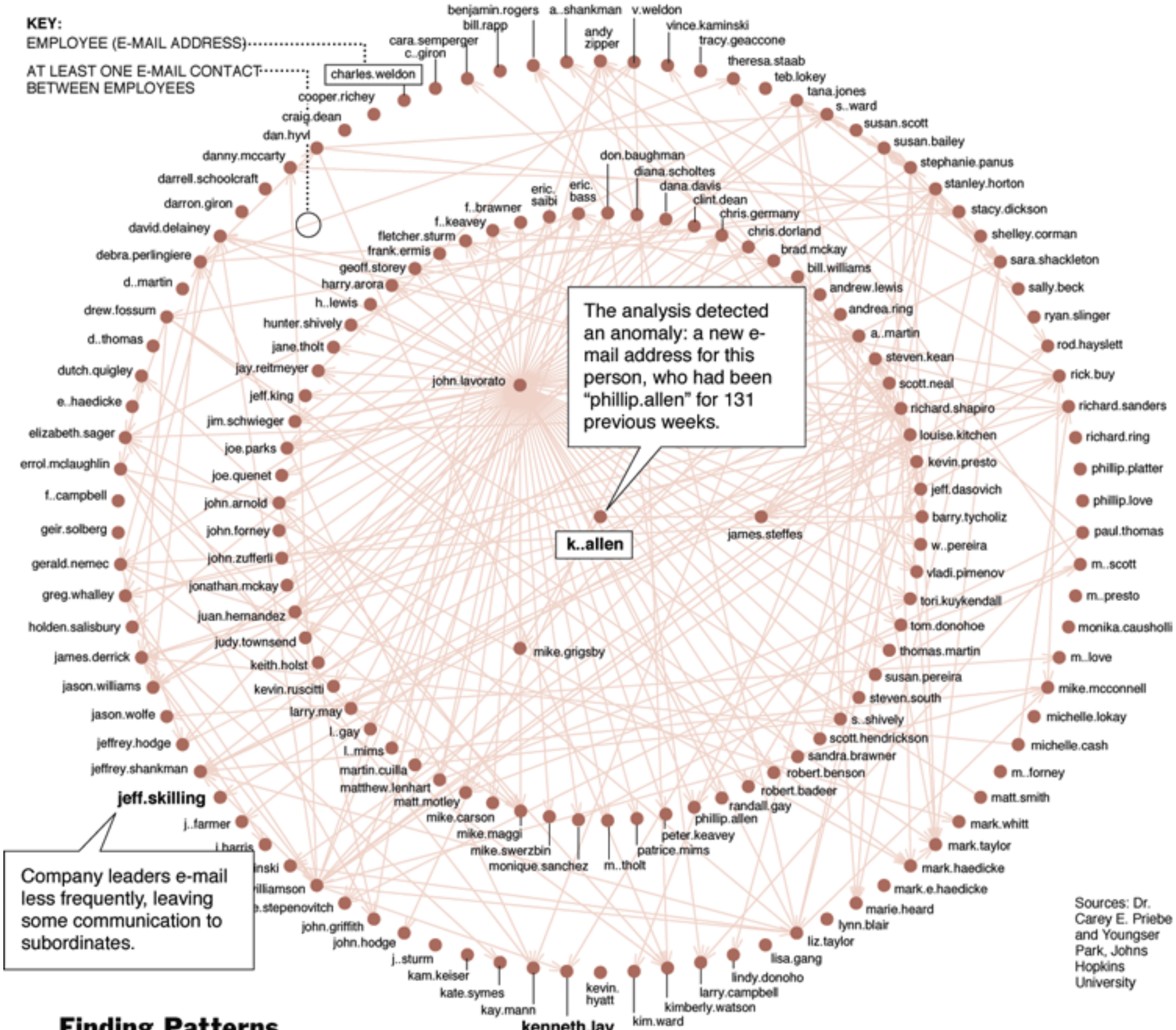
Graphs 2

Anders Kalhauge and Marjahan Begum



Finding Patterns In Corporate Chatter

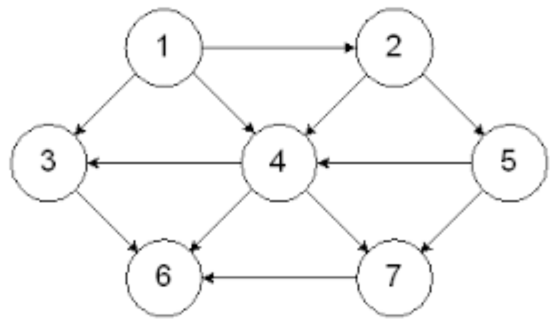
Computer scientists are analyzing about a half million Enron e-mails. Here is a map of a week's e-mail patterns in May 2001, when a new name suddenly appeared. Scientists found that this week's pattern differed greatly from others, suggesting different conversations were taking place that might interest investigators. Next step: word analysis of these messages.

Graphs - Connectivity

- x-y connectivity problem. Given two node *x* and *y*, is there a path between *x* and *y* ?
- x-y shortest path problem. Given two node *x* and *y*, what is the length of the shortest path between *x* and *y* ?
- Applications.
  - Friendster.
  - Maze traversal.
  - Fewest number of hops in a communication network.

Directed graphs

- Notation. G = (V, E).
  - Edge (u, v) leaves node u and enters node v.

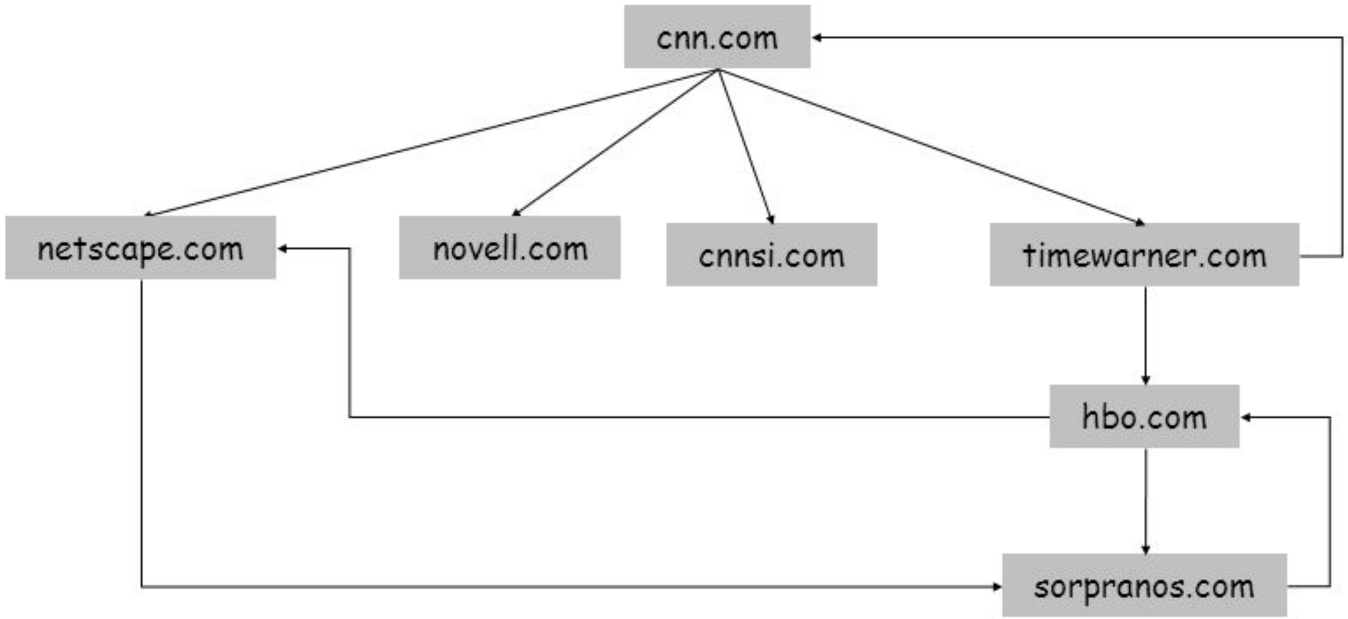


- Ex. Web graph: hyperlink points from one web page to another.
  - Orientation of edges is crucial.
  - Modern web search engines exploit hyperlink structure to rank web pages by importance.

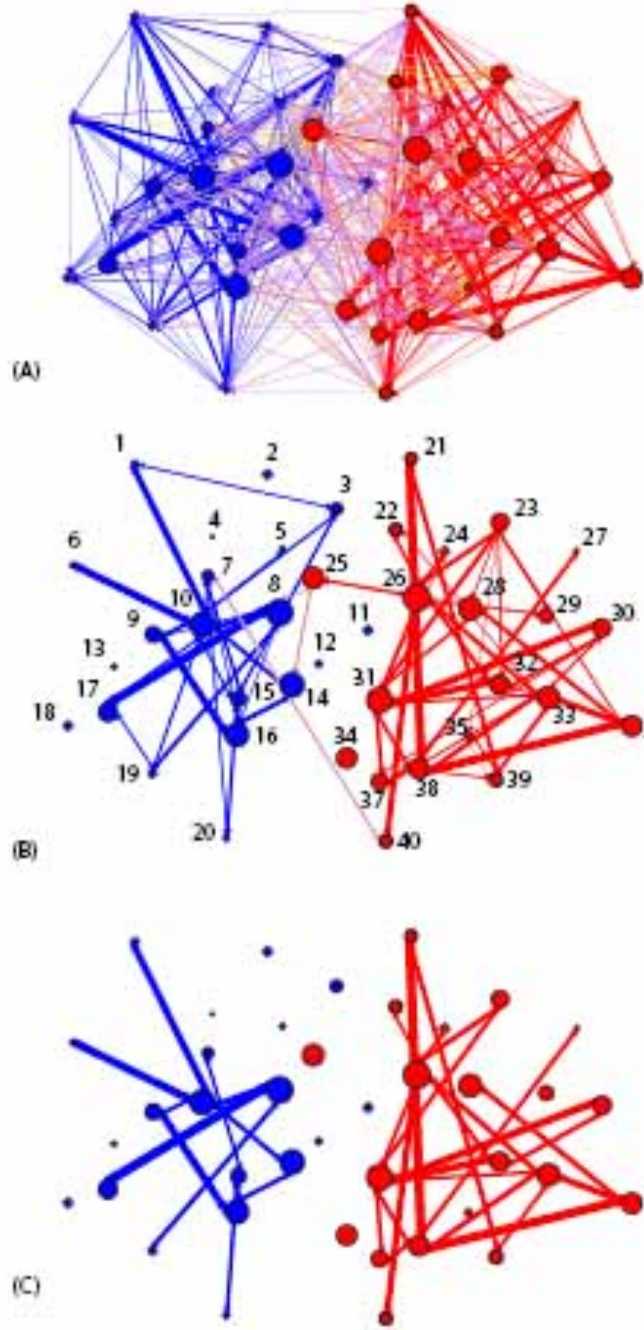
World Wide Web

Web graph.

- Node: web page.
- Edge: hyperlink from one page to another.




34 Ma
EOCENE
CRETACEOUS
JURASSIC
DEVONIAN
ORDOVICIAN
SILURIAN
CAMBRIAN
EDIACARAN
CRYOGENIAN
ARCHEAN
2500 Ma
600 Ma
543 Ma
635 Ma
525 Ma
470 Ma
415 Ma
375 Ma
200 Ma
150 Ma
540 Ma
440 Ma
55 Ma
1000 Ma
130 Ma
357 Ma
472 Ma
3500 Ma
200 Ma
RHIZARIA
CHROMISTA
ANIMALS
PLANTS
FUNGI
First life
Radiolarians
Cercozoans
Cercozoans
Choanoflagellates
Sponges
Comb Jellies
Jellyfish & relatives
Starfish & relatives
Lancelets
Sea squirts
Hagfishes
Lampreys
Chondrichthyans
Ray-finned fishes
Lobe-finned fishes
Lungfishes
Amphibians
Mammals
Turtles
Lizards & relatives
Crocodilians
Birds
Bryozoans
Flatworms
Rotifers
Ribbon worms
Lamp shells
Segmented worms
Mollusks
Arrow worms
Roundworms
Tardigrades
Onychophorans
Spiders & relatives
Millipedes & relatives
Crustaceans
Insects & relatives
Club fungi
Sac fungi
Arbuscular mycorrhizal fungi
Zygospore fungi
Chytrids
Microspridia
Slime molds & relatives
Eudicots
Monocots
Magnolids
Star anise
Water lillies
Amborella
Conifers
Gnetophytes
Ginko
Cycads
Ferns
Horsetails
Whisk ferns
Club mosses & relatives
Hornworts
Mosses
Liverworts
Charales
Coleochaetales
Bacteria
Archaebacteria
Haptophytes
Brown Algae
Diatoms
Oomycetes
Dinoflagellates
Apicomplexans
Ciliates



1 Digbys Blog
2 James Walcott
3 Pandagon
4 blog.johnkerry.com
5 Oliver Willis
6 America Blog
7 Crooked Timber
8 Daily Kos
9 American Prospect
10 Eschaton
11 Wonkette
12 Talk Left
13 Political Wire
14 Talking Points Memo
15 Matthew Yglesias
16 Washington Monthly
17 MyDD
18 Juan Cole
19 Left Coaster
20 Bradford DeLong

21 JawaReport
22 Voka Pundit
23 Roger L Simon
24 Tim Blair
25 Andrew Sullivan
26 Instapundit
27 Blogs for Bush
28 Little Green Footballs
29 Belmont Club
30 Captain's Quarters
31 Powerline
32 Hugh Hewitt
33 INDC Journal
34 Real Clear Politics
35 Winds of Change
36 Allahpundit
37 Michelle Malkin
38 WizBang
39 Dean's World
40 Volokh

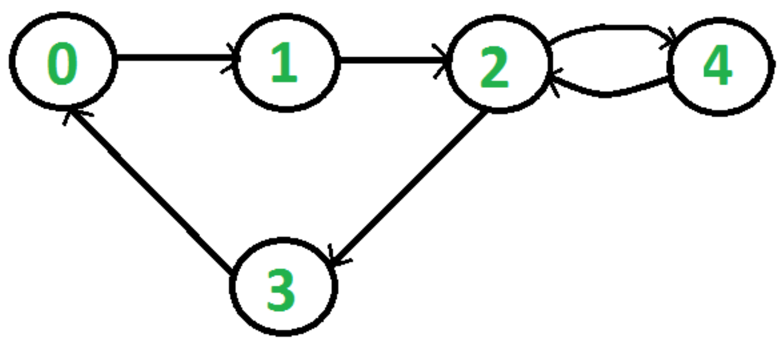
Graph Search

- Directed reachability. Given a node s, find all nodes reachable from x.
- Directed x-y shortest path problem. Given two node x and y, what is the length of the shortest path from x and y ?
- **Graph search**. BFS extends naturally to directed graphs.
- Web crawler. Start from web page s. Find all web pages linked from x, either directly or indirectly.

Strong connectivity

- Def. Nodes u and v are mutually reachable if there is a both path from u to v and also a path from v to u.

- Def. A graph is strongly connected if every pair of nodes is mutually reachable.



Strongly Connected

Strategies for traverals of graphs

We can use either breadth-first or depth-first traversals

- Breadth-first requires a queue
- Depth-first requires a stack

Breadth-first traversal

Consider implementing a breadth-first traversal on a graph:

- Choose any vertex, mark it as visited and push it onto queue

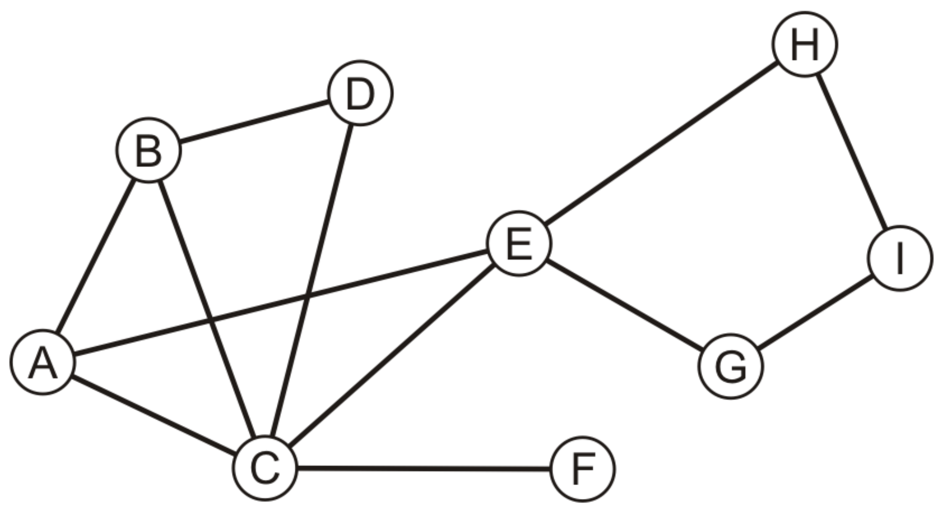
- While the queue is not empty:
  - Pop to top vertex v from the queue
  - For each vertex adjacent to v that has not been visited:
    - Mark it visited, and
    - Push it onto the queue

This continues until the queue is empty

Note: if there are no unvisited vertices, the graph is connected

Example

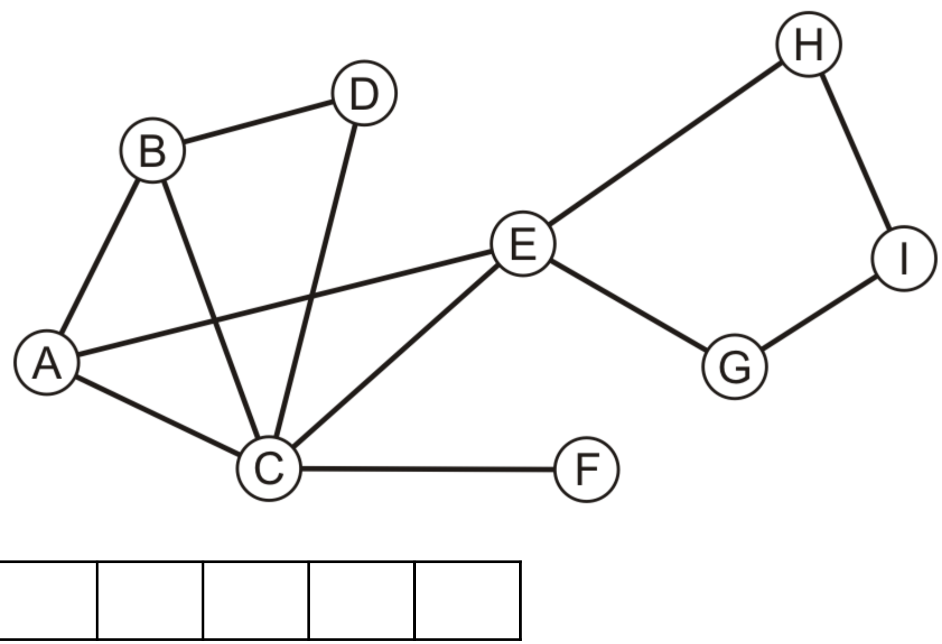
Consider this graph



Example

Performing a breadth-first traversal

– Enqueue the first vertex onto the queue

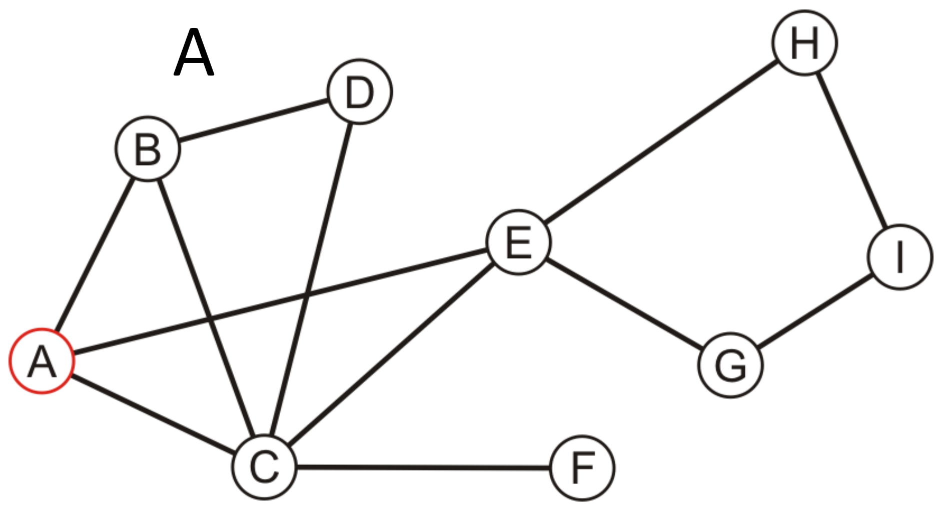


Example

Performing a breadth-first traversal

– Dequeue A and enqueue B, C and E

A

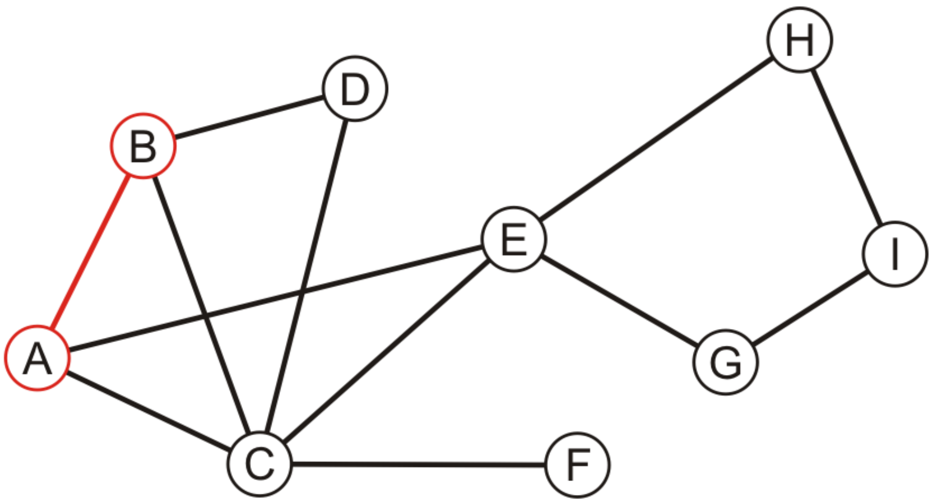


B	C	E			

Example

Performing a breadth-first traversal:

– Dequeue B and enqueue D

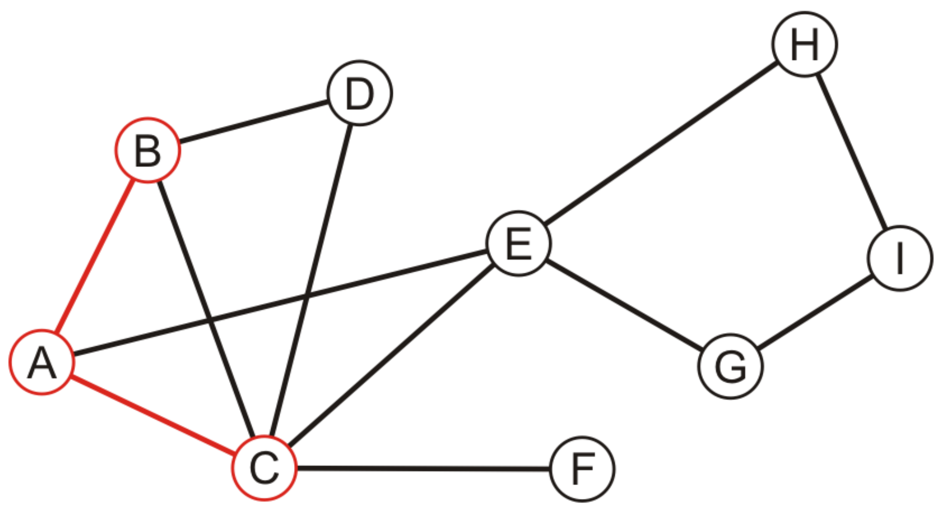


C	E	D			

Example

Performing a breadth-first traversal:

– Dequeue C and enqueue F

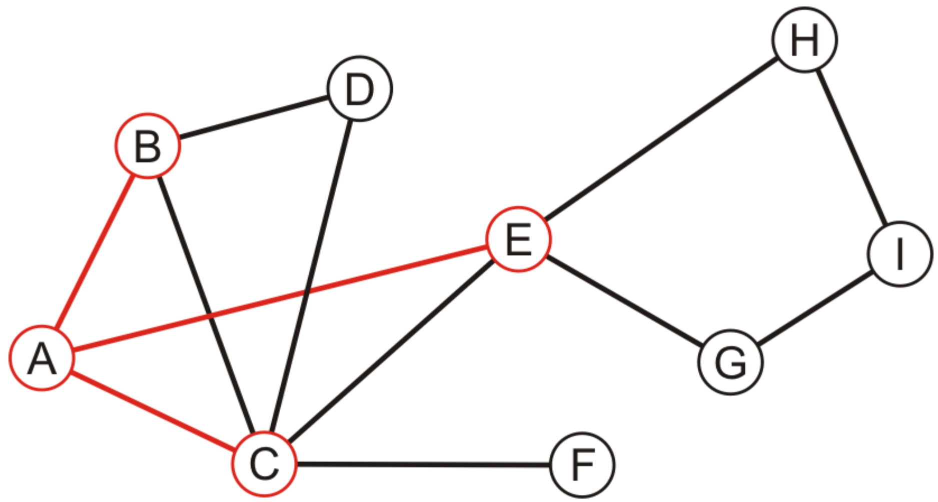


E	D	F			

Example

Performing a breadth-first traversal:

– Dequeue E and enqueue G and H

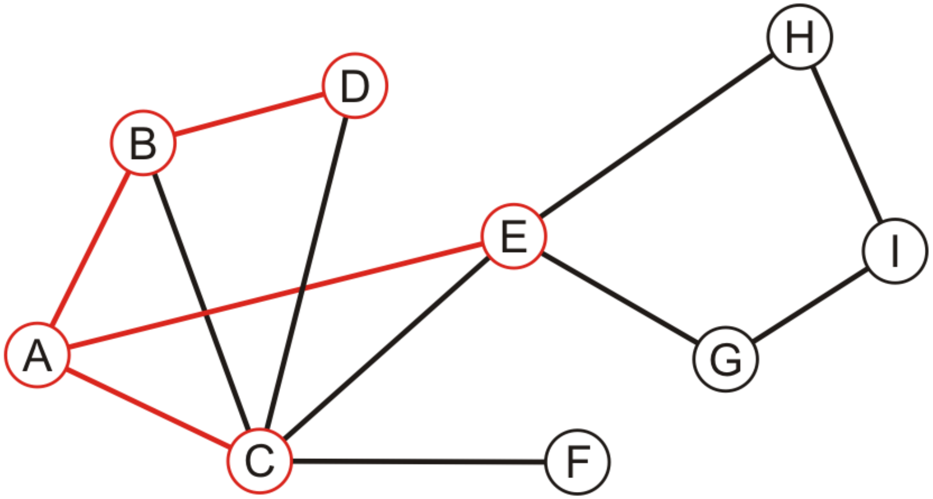


D	F	G	H		

Example

Performing a breadth-first traversal:

– Dequeue D

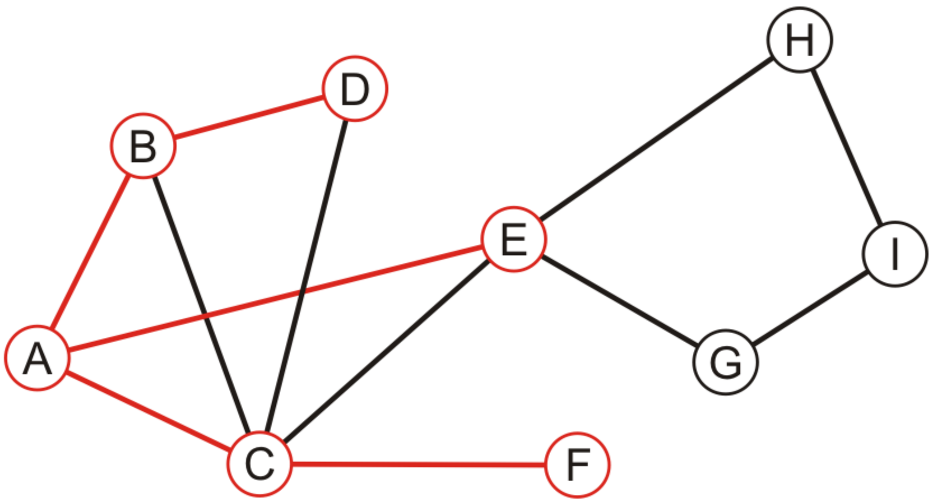


F	G	H			

Example

Performing a breadth-first traversal:

– Dequeue F

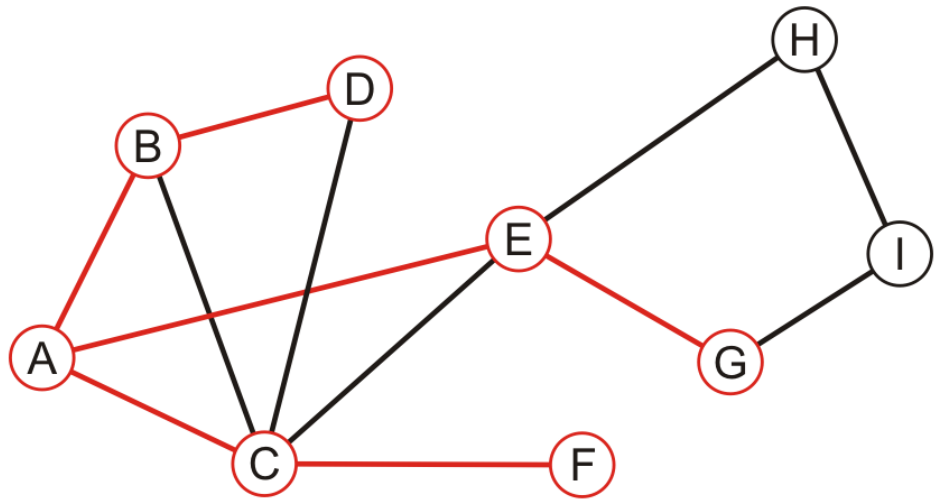


G	H				

Example

Performing a breadth-first traversal:

– Dequeue G and enqueue I

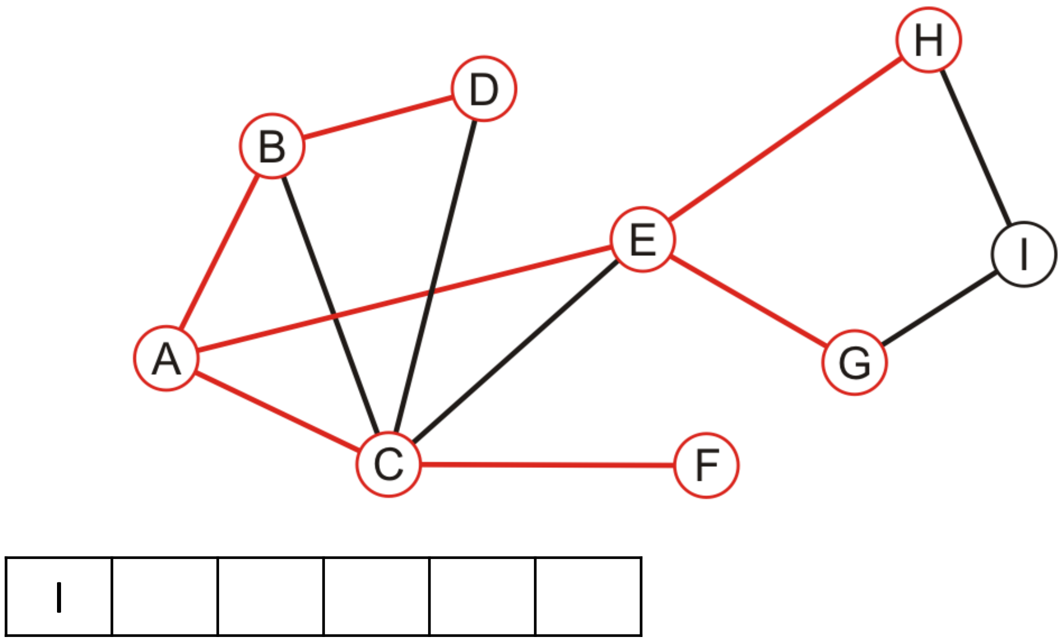


H	I				

Example

Performing a breadth-first traversal:

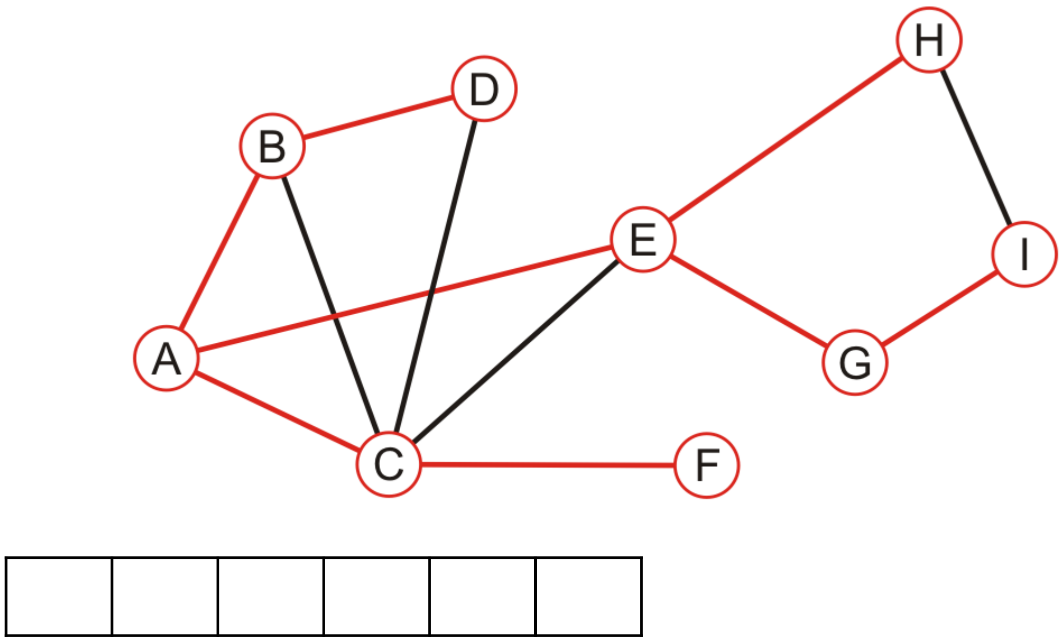
– Dequeue H



Example

Performing a breadth-first traversal:

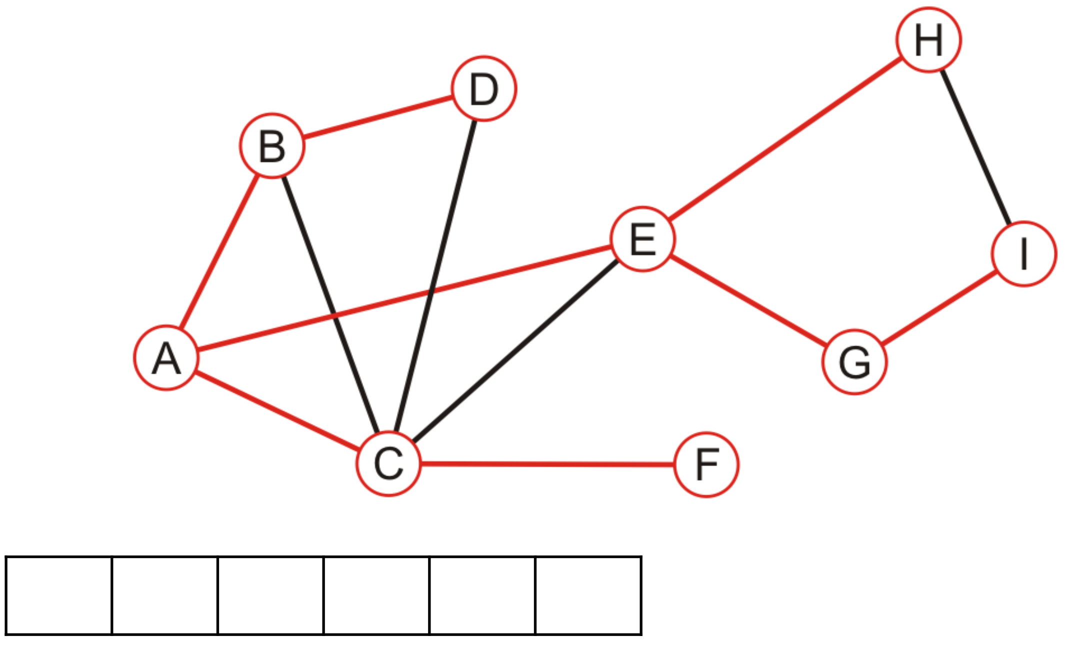
– Dequeue I



Example

Performing a breadth-first traversal:

– The queue is empty: we are finished



Explore Breadth-first traversal

- https://www.cs.usfca.edu/~galles/visualization/BFS.html

- https://en.wikipedia.org/wiki/Breadth-first_search#Example

Depth-first traversal

Consider implementing a depth-first traversal on a graph:

- Choose any vertex, mark it as visited
- From that vertex:
  - If there is another adjacent vertex not yet visited, go to it
  - Otherwise, go back to the most previous vertex that has not yet had all of its adjacent vertices visited and continue from there
- Continue until no visited vertices have unvisited adjacent vertices

Two implementations:

- Recursive
- Iterative

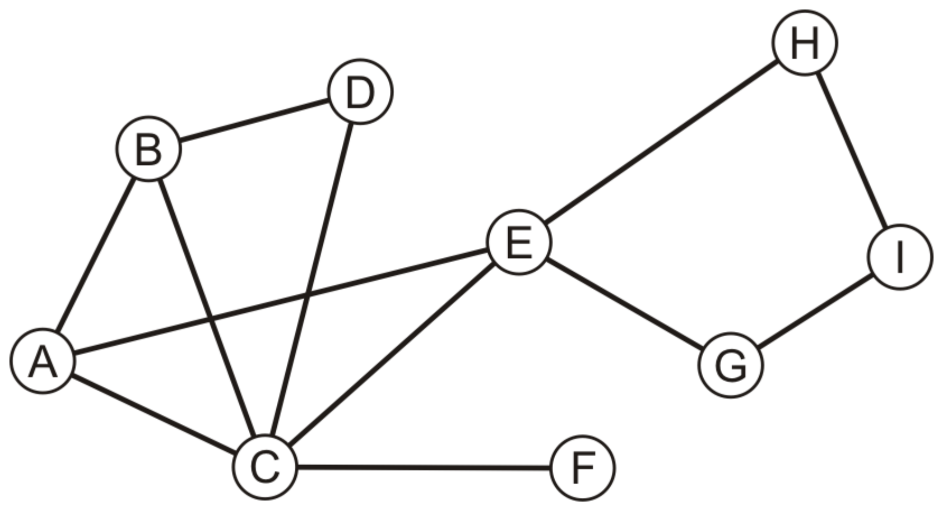
Iterative depth-first traversal

If memory is an issue, we can reduce the stack size:

- For the vertex:
  - Mark it as visited
  - Perform an operation on that vertex
  - Place it onto an empty stack
- While the stack is not empty:
  - If the vertex on the top of the stack has an unvisited adjacent vertex,
    - Mark it as visited
    - Perform an operation on that vertex
    - Place it onto the top of the stack
  - Otherwise, pop the top of the stack

Example

Perform a recursive depth-first traversal on this same graph

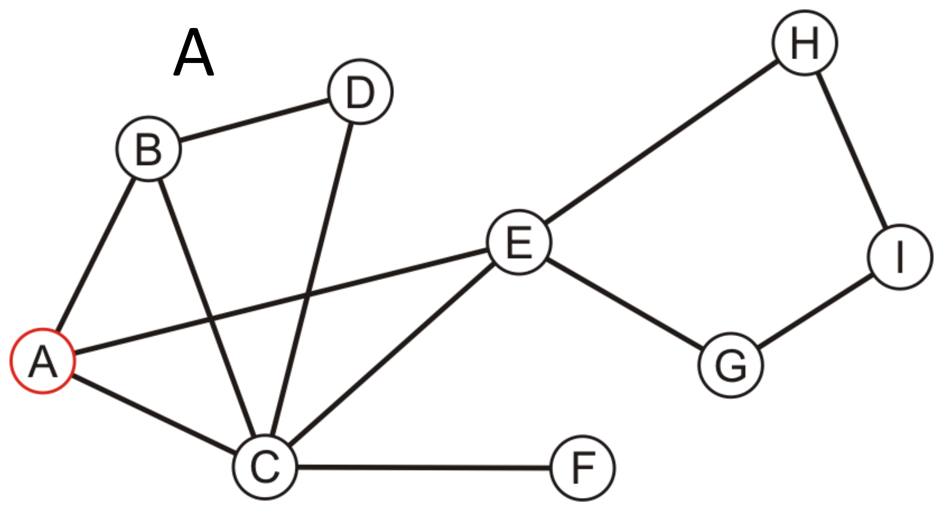


Example

Performing a recursive depth-first traversal:

– Visit the first node

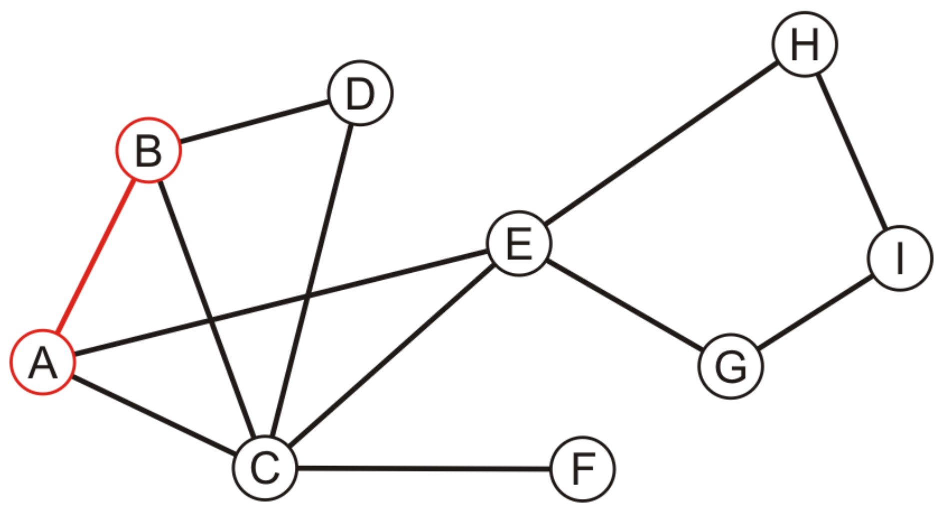
A



Example

Performing a recursive depth-first traversal:

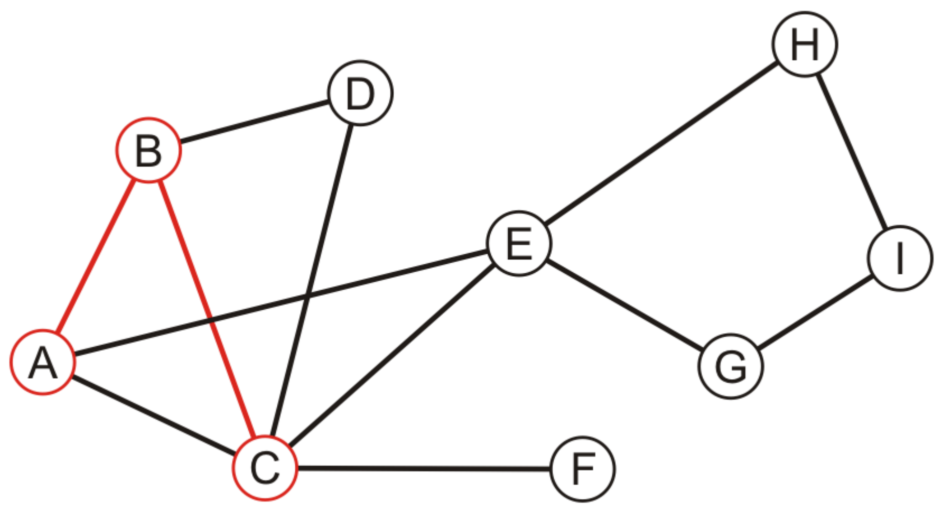
– A has an unvisited neighbor



Example

Performing a recursive depth-first traversal:

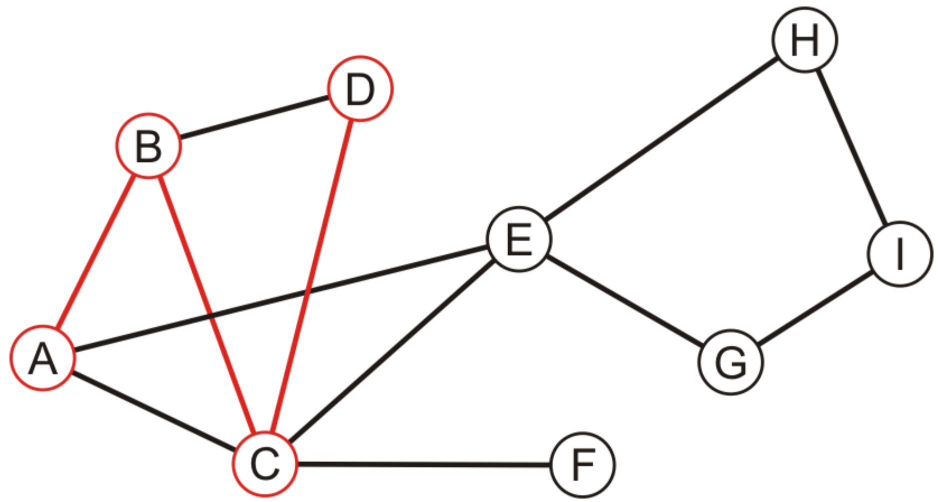
– B has an unvisited neighbor



Example

Performing a recursive depth-first traversal:

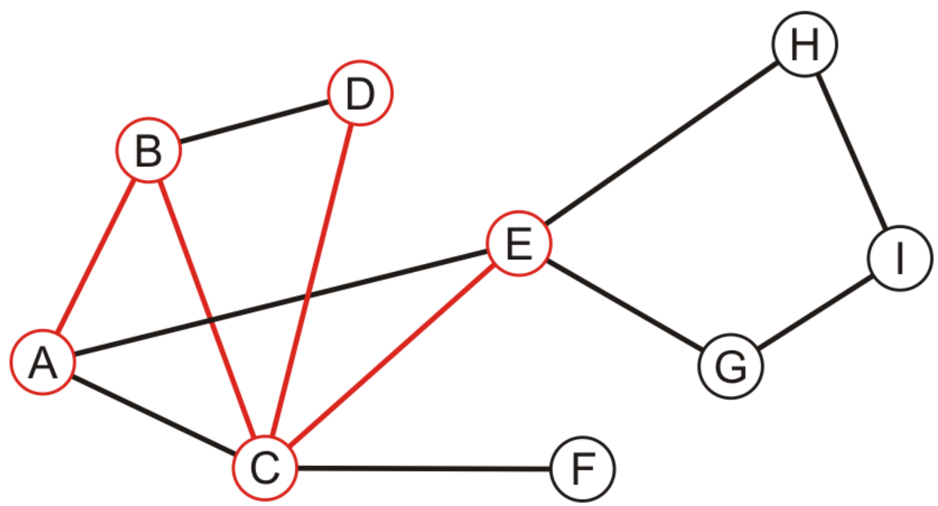
– C has an unvisited neighbor



Example

Performing a recursive depth-first traversal:

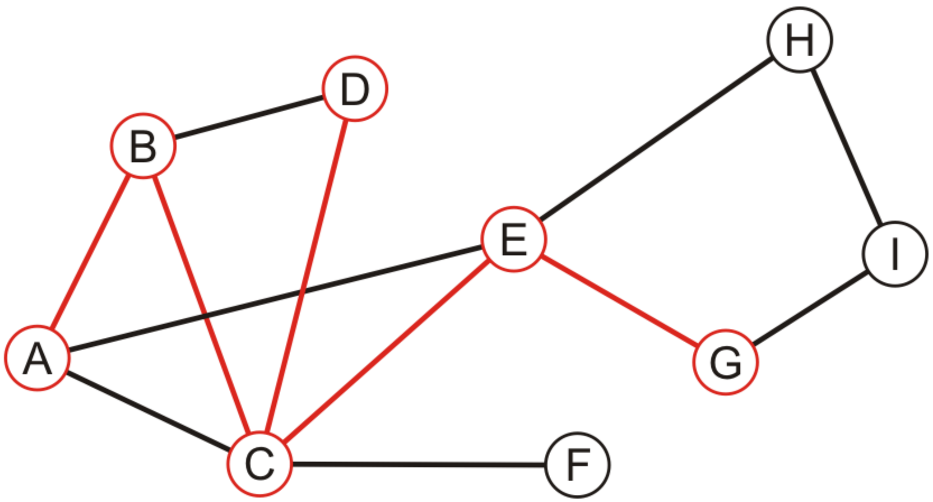
– D has no unvisited neighbors, so we return to C



Example

Performing a recursive depth-first traversal:

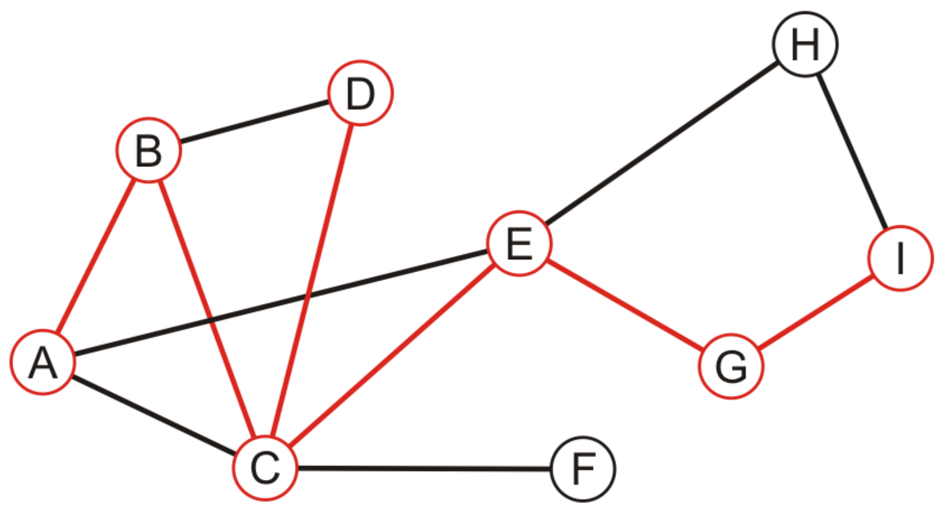
– E has an unvisited neighbor



Example

Performing a recursive depth-first traversal:

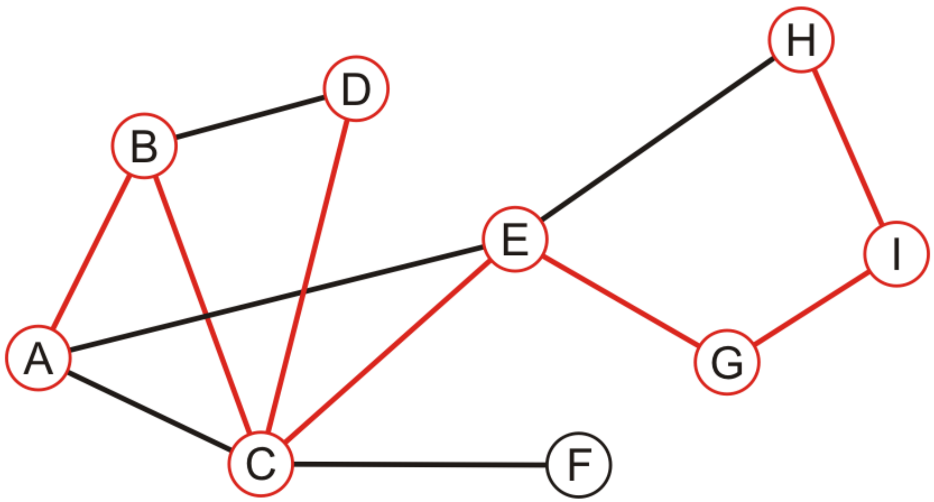
– F has an unvisited neighbor



Example

Performing a recursive depth-first traversal:

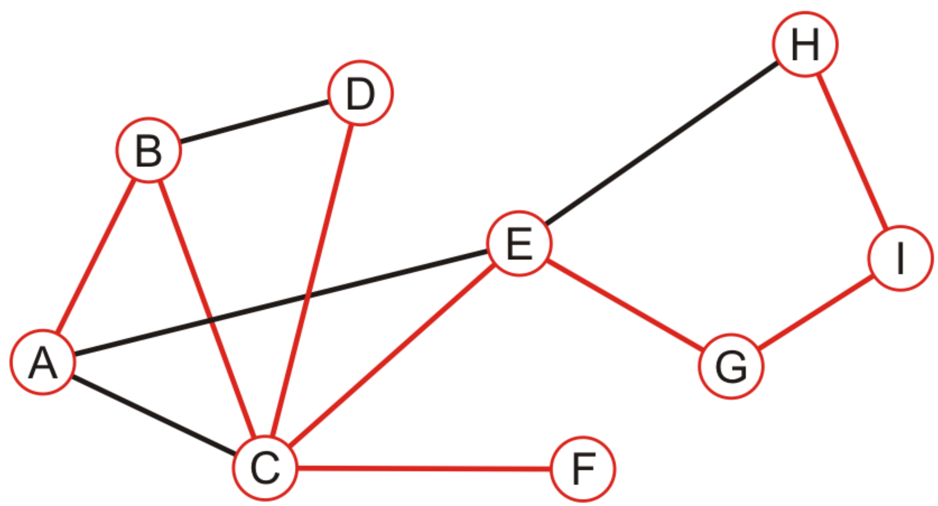
– H has an unvisited neighbor



Example

Performing a recursive depth-first traversal:

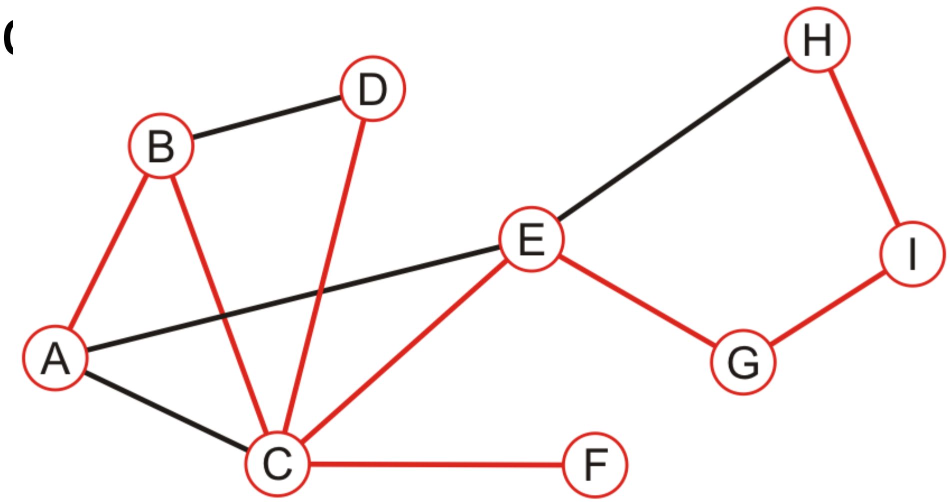
– We recurse back to C which has an unvisited neighbo



Example

Performing a recursive depth-first traversal:

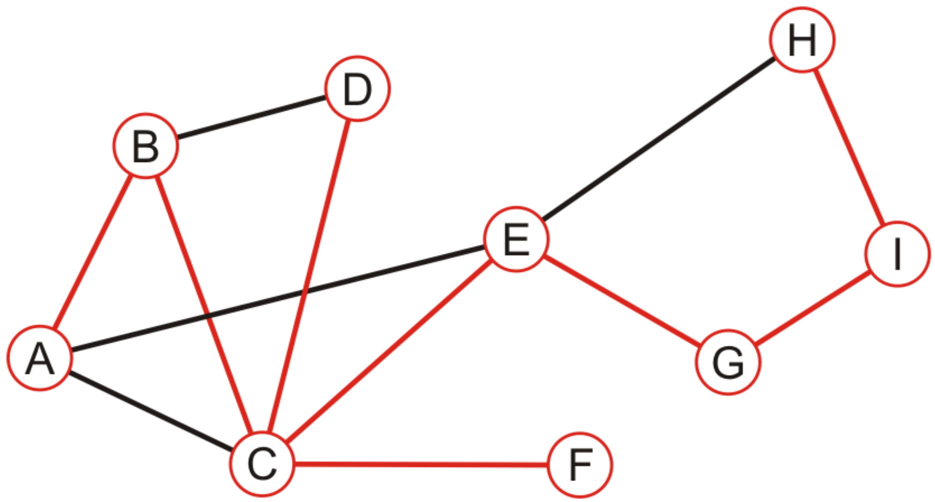
– We recurse finding that no other nodes have unvisite



Comparison

Performing a recursive depth-first traversal:

– We recurse finding that no other nodes have unvisite

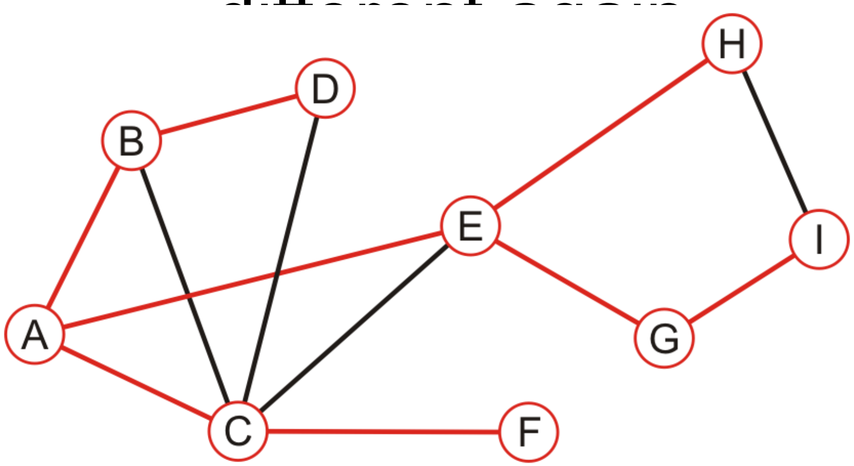


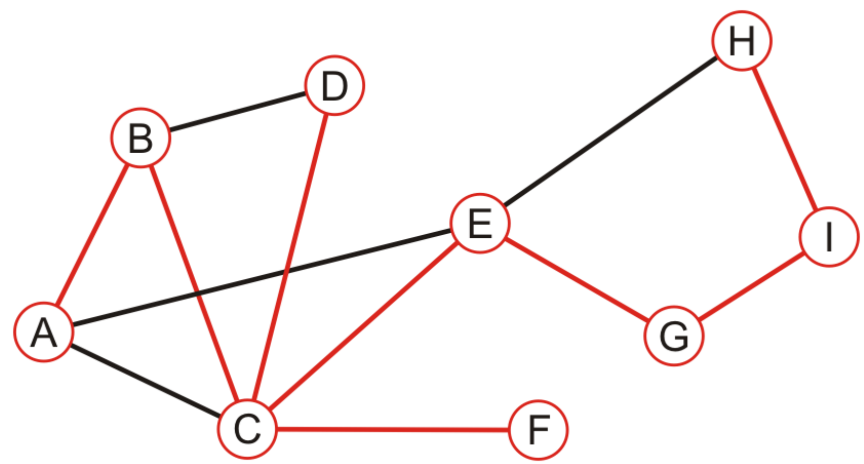
Comparison

The order in which vertices can differ greatly

A, B, C, E, D, F, G, H, I A, B, C, D, E, G, I, H, F

– An iterative depth-first traversal may also be different again





Explore depth-first traversal

- https://www.cs.usfca.edu/~galles/visualization/DFS.html

- https://en.wikipedia.org/wiki/Depth-first_search#Pseudocode

Applications

Applications of tree traversals include:

- Determining connectiveness and finding connected sub-graphs
- Determining the path length from one vertex to all others
- Testing if a graph is bipartite(next week)
- Determining maximum flow
- Cheney's algorithm for garbage collection

Summary

This topic covered graph traversals

- Considered breadth-first and depth-first traversals
- Depth-first traversals can recursive or iterative
- They are also called *searches*

Bipartite graph

- Vertices can be divided into two sets
- Show example

Simple Exercise

- Design a social network of friends in the class using a graph data structure
- Get inspirations from
  - http://introcs.cs.princeton.edu/java/45graph/Graph.java
  - https://www.cs.duke.edu/courses/cps100e/fall10/class/11_Bacon/code/Graph.html
- Implement BFS and DFS on the social network exercise