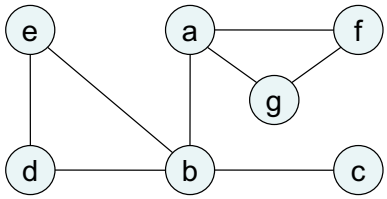
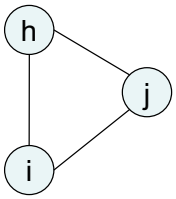
Lecture 4: Reachability in undirected graphs

What parts of a graph are reachable from a given vertex?





With an adjacency list representation, this is like navigating a maze...

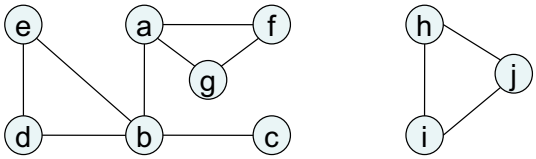
Potential difficulty	Don't go round in circles	Don't miss anything
Classical solution	Piece of chalk to mark visited junctions	Ball of string – leads back to starting point
Cyber-analog	Boolean variable for each vertex: visited or not	STACK

An exploration procedure

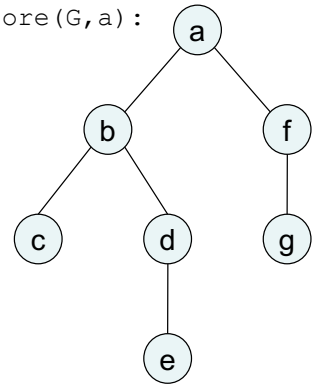
```
procedure explore(G,v)

input: graph G = (V,E); node v in V
output: visited[u] is set to true
   for all u reachable from v

visited[v] = true
for each edge (v,u) in E:
   if not visited[u]:
        explore(G,u)
```



explore(G,a):



Does “explore” work?

```
procedure explore(G,v)
visited[v] = true
for each edge (v,u) in E:
   if not visited[u]:
        explore(G,u)
```

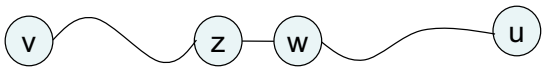
Does it actually halt?

For any node u, explore(G,u) is called at most once;
thereafter visited[u] is set.

Does it visit everything reachable from v?

Suppose it misses node u reachable from v; we’ll derive a contradiction.

Pick any path from v to u, and let z be the last node on the path that was visited.



But w would not have been overlooked during explore(G,z); this is a contradiction.

Alternative proof

```
procedure explore(G,v)
visited[v] = true
for each edge (v,u) in E:
   if not visited[u]:
        explore(G,u)
```

Does explore(G,v) visit everything reachable from v?

Do a proof by induction.