Administrative Details

- Lab 10 – due today at noon
- Any questions?
- This week
  - Wrap up graphs on Wed
  - Hashmaps on Wed and Fri
  - Course evals on Fri
- Optional lab this week
- MAKEUP CLASS WED 1:10-2:00 in Wege
- Self-scheduled final
- I will be out of town Friday - Sunday

Last Time

- Looked at GraphMatrix and GraphList implementations
  - Adjacency Matrix
    - Define graph using an adjacency matrix
    - Matrix keeps track of edge weights
  - Adjacency List
    - Each vertex keeps track of out edges

Today’s Outline

- Continue talking about graphs
- Darwin!!

Reachability

- There are two ways to measure reachability in our graph
  - Depth-first search and breadth-first search
- How did we do DFS and BFS in trees?
- DFS uses a stack
  - Stack records path from src to current node
  - Like pre-order tree traversal with visited flags to only visit nodes once
  - Runtime: O(|E|)
- BFS uses a queue
  - Queue records nodes whose out edges have not been explored
  - Like level-order tree traversal
  - Runtime: O(|E|)

Suppose we applied a direction to the edges in our campus map…

- Paraske
- West
- Brookman
- Jesup
- Arts
- Pool
- TCL
- TBL
- Griffin
- School
void reachableFrom(Graph<V,E> g, V src) {
    if (!g.visited(src)) {
        g.visit(src);
        Iterator<V> neighbors = g.neighbors(src);
        while (neighbors.hasNext()) {
            V next = neighbors.next();
            if (!g.visited(next))
                reachableFrom(g, next);
        }
    }
}

Depth-First Search

Find first unvisited neighbor of TCL...

TBL

Find first neighbor of TBL...

TBL

Find first neighbor of Bronfman...

Bronf

Find next neighbor of Bronf...none exist! Pop stack...

Bronf

Find first neighbor of West...none exist! Pop stack...

West
Depth-First Search

Find next neighbor of TBL...none exist! Pop stack...

Find first neighbor of Art...

Find first unvisited neighbor of Pool...none exist!
Depth-First Search

Find next neighbor of Jesup...none exist!

Find next neighbor of TCL...none exist!

We're done!

Breadth-First Search

```java
void reachableFrom(Graph<V,E> g, V src) {
    Queue todo = new QueueList();
    g.visit(src);
    todo.enqueue(src);
    while (!todo.isEmpty()) {
        V node = todo.dequeue();
        Iterator<V> neighbors = g.neighbors(node);
        while (neighbors.hasNext()) {
            V next = neighbors.next();
            if (!g.visited(next)) {
                g.visit(next);
                todo.enqueue(next);
            }
        }
    }
}
```
Breadth-First Search

Find all unqueued/unvisited neighbors of TCL...

todo queue 25

Find all unqueued/unvisited neighbors of TBL...

todo queue 26

Find all unqueued/unvisited neighbors of Bronf...

todo queue 27

Find all unqueued/unvisited neighbors of Jesup...

todo queue 28

Find all unqueued/unvisited neighbors of West...

todo queue 29

Find all unqueued/unvisited neighbors of Art...

todo queue 30
Breadth-First Search

Find all unqueued/unvisited neighbors of Pool…

dfs

DFS vs. BFS

DFS: BFS:

DFS vs. BFS

DFS: BFS:

DFS vs. BFS

DFS: BFS:
DARWIN!!!!!