Lecture 15: Graph Traversals

CSE 332: Data Structures & Parallelism

Yafqa Khan

Summer 2025

Announcements

- EX06 due Friday
- EX07 released today
- Exam 2 information posted here:
  - https://courses.cs.washington.edu/courses/cse332/25su/exams/final.html
  - **Note: it will be hard to accommodate makeups; only four days to grade**
  - If you can’t make proposed makeup dates (e.g., sickness/emergency), some options:
  - Option 1: Exam 1 is worth 40% instead of 20% of overall grade
  - Option 2: Take the final exam in the next CSE 332 offering

Today

- Graph Terminologies
  - Paths vs Cycles
  - Connected vs Unconnected
  - Sparse vs dense
- Graph Data structures
  - Adjacency Matrix
  - Adjacency List
- Graph Traversals
  - DFS (Iterative + Recursive)
  - BFS
- Graph Shortest Paths
  - Dijkstra's

Today

- Graph Terminologies
  - Paths vs Cycles
  - Connected vs Unconnected
  - Sparse vs dense
- Graph Data structures
  - Adjacency Matrix
  - Adjacency List
- Graph Traversals
  - DFS (Iterative + Recursive)
  - BFS
- Graph Shortest Paths
  - Dijkstra's

Today

- Graph Terminologies
  - Paths vs Cycles
  - Connected vs Unconnected
  - Sparse vs dense
- Graph Data structures
  - Adjacency Matrix
  - Adjacency List
- Graph Traversals
  - DFS (Iterative + Recursive)
  - BFS
- Graph Shortest Paths
  - Dijkstra's

Graphs: Algorithms

Okay, we can represent graphs

Now let’s implement some useful and non-trivial algorithms

- Graph Traversals: Depth-first graph search (DFS) & Breadth-first graph search (BFS)
- Shortest paths: Find the shortest or lowest-cost path from x to y
  - Related: Determine if there even is such a path

Graphs: Traversals

Problem: In a graph `G`, find all nodes from a node `src`

- i.e., Is there a path from `src` to specific nodes?

Useful for doing something (processing) at a node (e.g., print the node)

Basic Idea:

- Keep following nodes
- "mark" nodes after visiting them such that it processes each node once

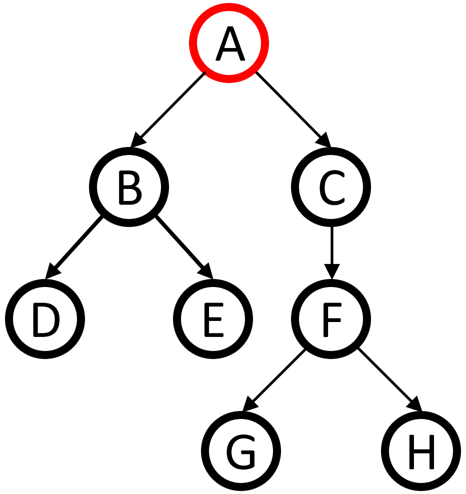
Traversal: Abstract "Pseudocode"

```
traverseGraph(Node src) {
  Set pending = new DataStructure();
  pending.add(src)
  mark src as visited
  while(pending is not empty) {
      v = pending.remove()
      for each node u adjacent to v // i.e., all of v's neighbour(s)
            if(u is not marked) {
                  mark u
                  pending.add(u)
            }
  }
}
```

Traversal: Algorithms

- Depth-First Search
  - Uses a Stack
  - (Recursively) Explore far away from `src` first
- Breadth-First Search
  - Uses a Queue
  - Explore everything near `src` first

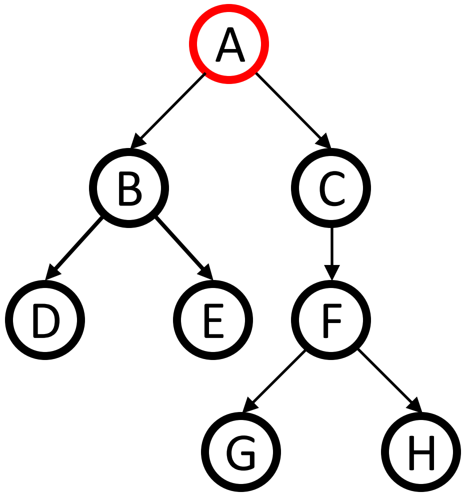
Traversal: Iterative DFS (Less common)



Order Processed:

```
IterativeDFS(Node src) {
   s = new Stack()
   s.push(src)
   mark src as visited
   while(s is not empty) {
        v = s.pop() // and "process"
        for each node u adjacent to v
        if(u is not marked)
                mark u as visited
                s.push(u)
        }
   }
}
```

Traversal: Iterative DFS (Less common) (Soln.)



Order Processed:

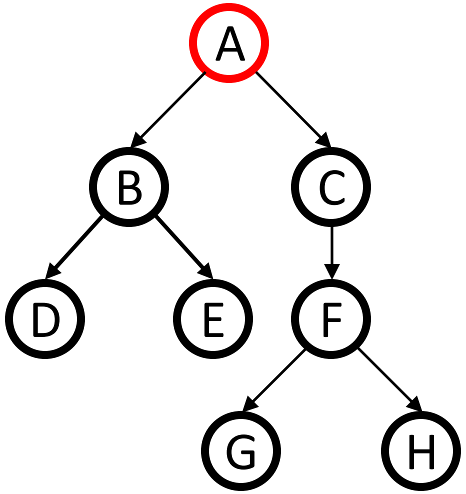
A, C, F, H, G, B, E, D

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

etc.

```
IterativeDFS(Node src) {
  s = new Stack()
  s.push(src)
  mark src as visited
  while(s is not empty) {
      v = s.pop() // and "process"
      for each node u adjacent to v
      if(u is not marked)
            mark u as visited
            s.push(u)
      }
  }
}
```

Traversal: Recursive DFS (More common)

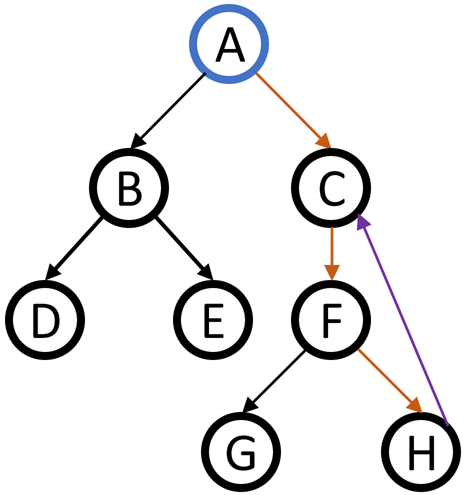


```
RecursiveDFS(Node v) {
  mark v as visited // and "process"
  for each node u adjacent to v
      if u is not marked
            RecursiveDFS(u)
}
```

Order Processed:

Same as before!

Cycle Detection



```
RecursiveDFS(Node v) {
  mark v as visited // and "process"
  for each node u adjacent to v
      if u is not marked
            RecursiveDFS(u)
}
```

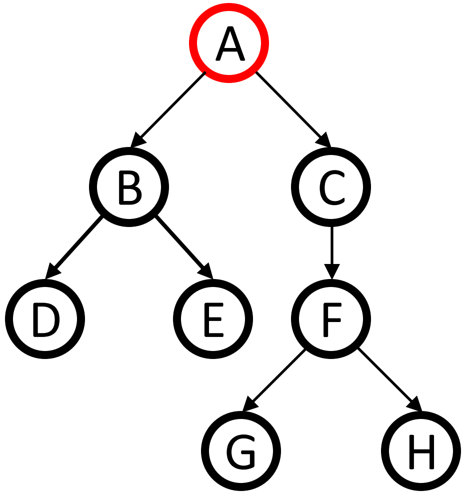
- Intuition: store the "current path" while doing DFS
- If you see a neighbor ('u' in pseudocode) that's already in the current path, then cycle

Use Iterative DFS for Exams

Recursive DFS recommended for EX7

Any Questions?

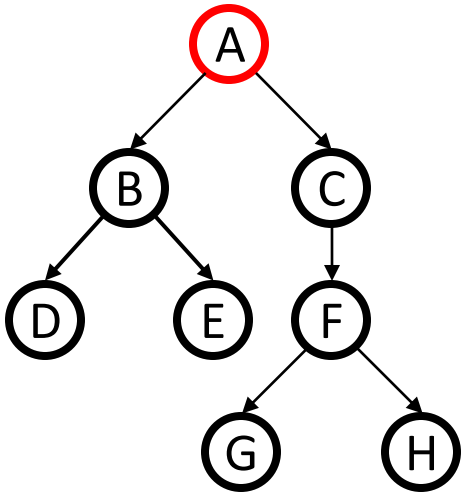
Traversal: BFS (Soln.)



Order Processed:

```
BFS(Node src) {
   s = new Queue()
   s.enqueue(src)
   mark src as visited
   while(s is not empty) {
        v = s.dequeue() // and "process"
        for each node u adjacent to v
        if(u is not marked)
               mark u as visited
               s.enqueue(u)
        }
   }
}
```

Traversal: BFS (Soln.)



Order Processed:

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

etc., any level-order traversal

```
BFS(Node src) {
   s = new Queue()
   s.enqueue(src)
   mark src as visited
   while(s is not empty) {
        v = s.dequeue() // and "process"
        for each node u adjacent to v
        if(u is not marked)
                mark u as visited
                s.enqueue(u)
        }
   }
}
```

Traversal: DFS vs BFS

- Depth-First Search (DFS):
  - Memory: Generally, DFS uses less memory compared to BFS as it only needs to store the nodes along the current branch.
  - Applications: Topological Sorting, Cycle Detection, etc.
- Breadth-First Search (BFS):
  - Memory: BFS tends to use more memory than DFS, as it needs to store all nodes at the current level before moving to the next level.
  - Applications: Shortest Paths
- 3rd Option: Iterative Deep DFS (IDDFS)
  - Use DFS with increasing depth limits
  - Good memory + finds shortest path

Traversal: Saving the Path

- Old Problem: Is there a path from `src` to specific nodes?
- New Problem: What is the path from `src` to specific nodes?

Q: How do we output the actual path?

A:

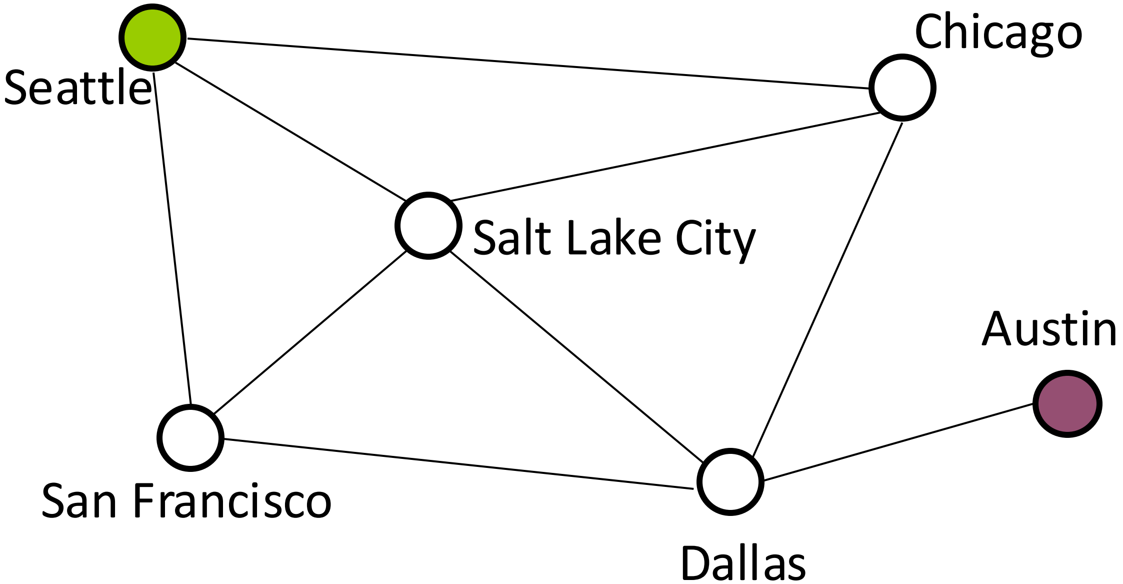
- When marking, store the predecessor (previous) node along the path
- When you're done search, follow the pred backwards to where you started (and then reverse it to get the path)

BFS with Path Saving

```
IterativeDFS(Node src) {
  s = new Queue()
  s.enqueue(src)
  src.pred = null // same as marking src as visited
  while(s is not empty) {
      v = s.dequeue() // and "process"
      for each node u adjacent to v
      if(u is not marked)
            u.pred = v // previous node of u in the path is v
            s.enqueue(u)
      }
  }
}
```

Traversal: BFS Shortest Path Example

What is the shortest path from Seattle to Austin?



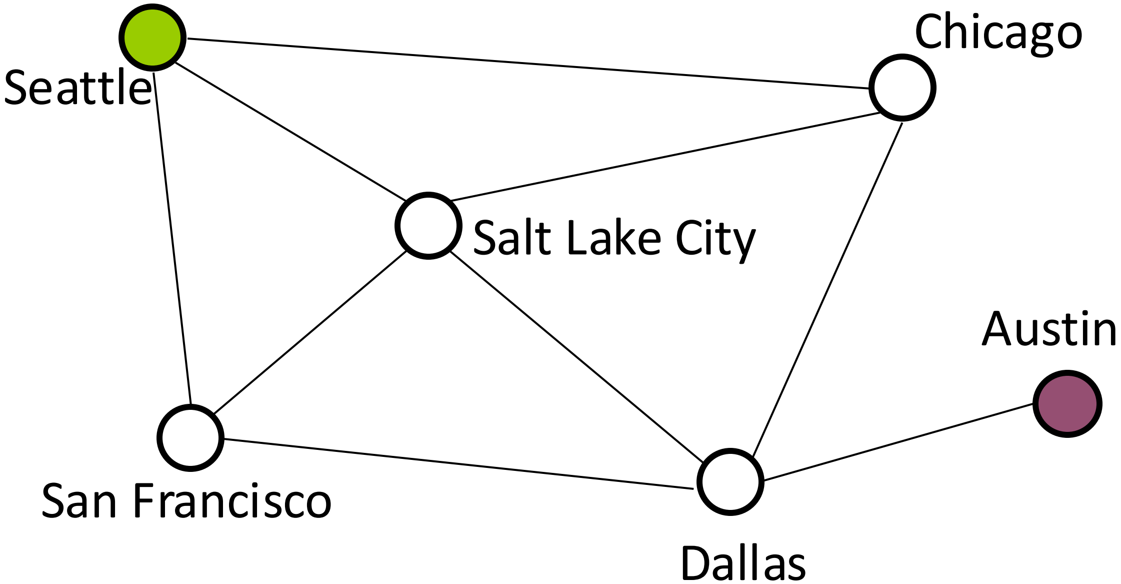
Traversal: BFS Shortest Path Example (Soln.)

What is the shortest path from Seattle to Austin?

Seattle -> Chicago -> Dallas -> Austin

Seattle -> Salt Lake City -> Dallas -> Austin

Seattle -> San Francisco -> Dallas -> Austin



Any Questions?