Improved Adjacency List Representation

Rather than using nested LinkedLists, which have O(|V|) lookup, use nested HashMaps, which have expected O(1) lookup.

```
public class AdjListGraph implements Graph<ALVertex, Edge<ALVertex>> {
  /** A vertex in this graph responsible for tracking its neighbors. */
  public static class ALVertex implements Vertex<Edge<ALVertex>> {
    /** The label of this vertex */
    private final String label;

    /** Map associating neighbor's labels with Edges connecting to them. */
    private final HashMap<String, Edge<ALVertex>> outEdges;
  }

  /** Map associating labels with ALVertex objects. */
  private final HashMap<String, ALVertex> vertices;
}
```

each vertex keeps track of neighborhood

graph is collection of vertices

Space Complexity:

Check for an edge:

Iterate over neighbors:

Iterate over all edges:

Add vertex/edge:

Comparing Representations

Representation	Memory	Time to Query Edge	Time to Iterate Neighbors	Time to Iterate all Edges
Adj Matrix	$O(\|V\|^2)$	$O(1)$	$O(\|V\|)$	$O(\|V\|^2)$
Adj List (Linked)	$O(\|V\|+\|E\|)$	$O(\|V\|)$	$O(\|V\|)$	$O(\|V\|+\|E\|)$
Adj List (Map)	$O(\|V\|+\|E\|)$	$O(1)$	$O(degree)$	$O(\|V\|+\|E\|)$

Dense Graphs:

Sparse Graphs:



Lecture 22: Graph Traversals

CS 2110

November 10, 2026

Today's Learning Outcomes

81. Describe adjacency list and matrix representations of a graph and compare the space/time complexities of operations on each of these representations.

82. Perform BFS and DFS traversals of a given graph by hand and in code.

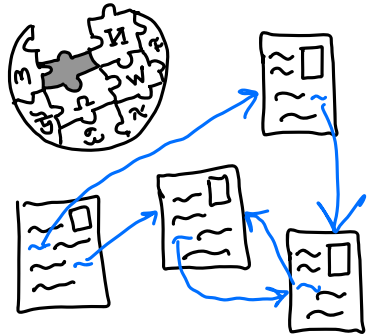
83. Visualize the state of a graph traversal at a given point of its execution, describing each vertex as either *undiscovered*, *discovered*, *visited*, and/or *settled*.

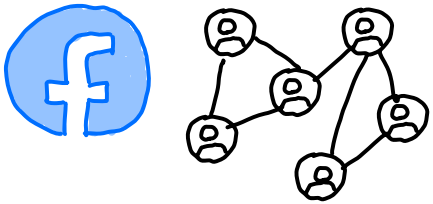
Graph Traversals

Procedures to systematically visit all vertices/edges in a graph.

Typically, we do this in an order that respects the graph's "local structure," following outgoing edges from one vertex to its neighbor.

** Local structure encodes meaningful information about the graph.

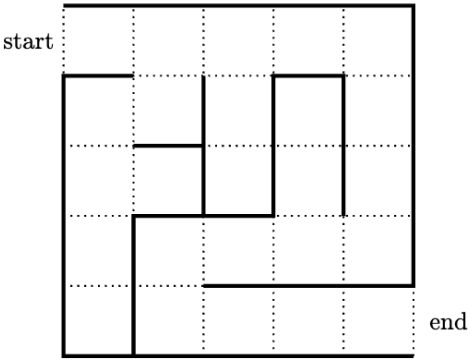


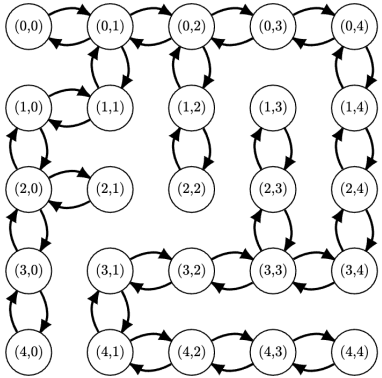




Navigating a Maze

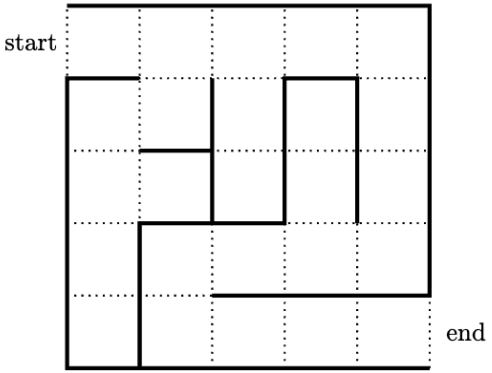
Rule: At any point, we're "standing" in one maze square and can only see whether it's possible to move to adjacent squares.



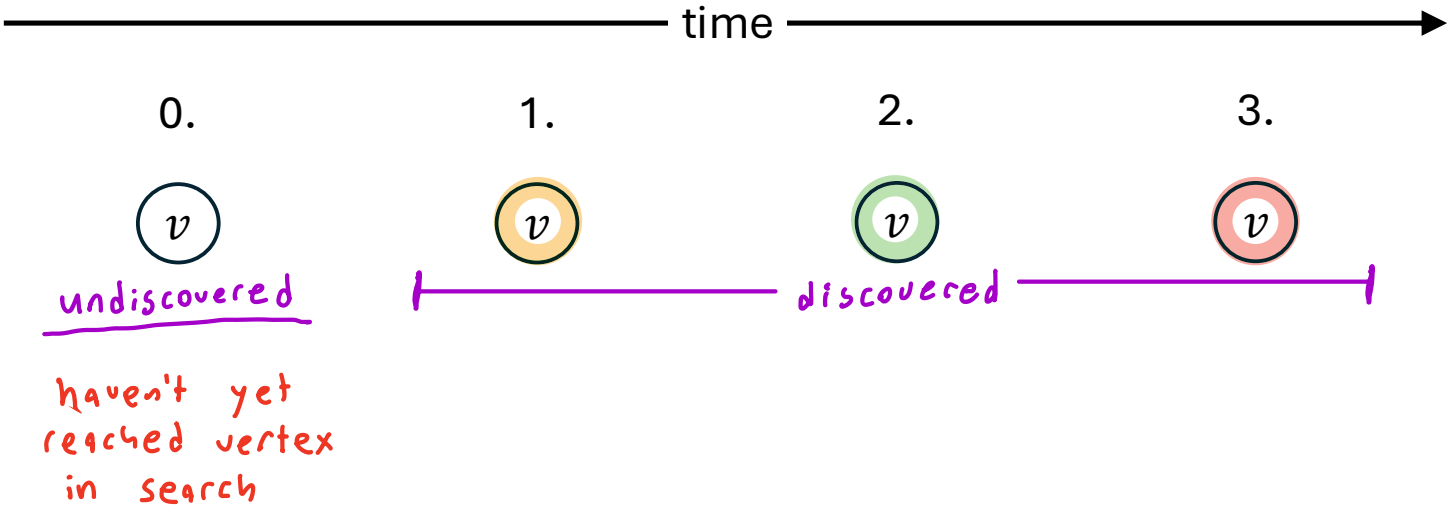


Depth-First Search (DFS)

= a search procedure that *fully explores* one path away from the source *before* backtracking to consider other paths.



Vertex Status During Search





Coding Demo: Fixing our DFS Code



DFS Complexity

```
boolean dfsRec(V current, V dst, Set<V> discovered) {
  if (current == dst) { return true; }

  for (E edge : current.outgoingEdges()) {
    V neighbor = edge.head();
    if (!discovered.contains(neighbor)) {
      discovered.add(neighbor);
      if (dfsRec(neighbor, dst, discovered)) { return true; }
    }
  }
  return false;
}
```

Search vs. Traversal

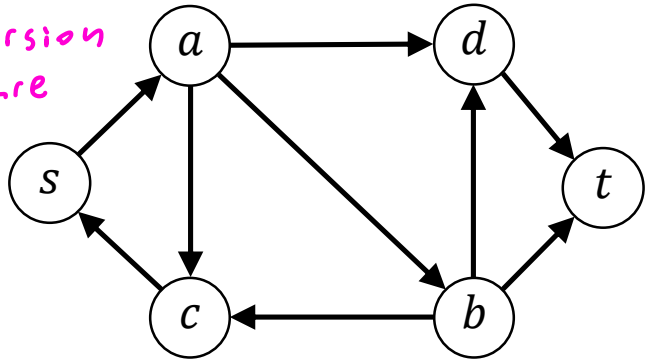
Search: Given a source s and a destination t, locate an $s \rightsquigarrow t$ path

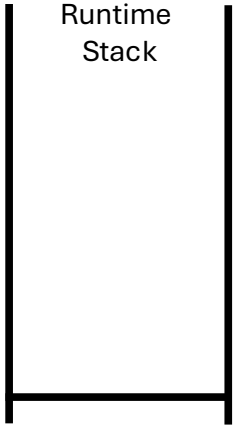
- Often, we won't need to traverse the entire graph to do this.

Traversal: Systematically visit every vertex and edge in the graph, taking some action during the visit

Visitation vs Settlement Orders

* see the animated version in the lecture notes





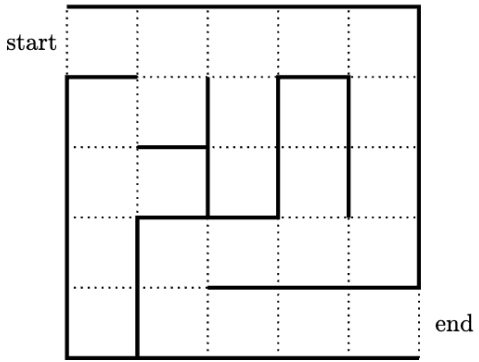
Visitation:

Settlement:

Breadth-First Search

DFS "focuses in" on one path at a time.

Alternative (BFS): "Fan out" a search party of "scouts" who follow all paths simultaneously





Coding Demo: BFS Traversals



BFS Complexity

```
void bfs(V src, Consumer<V> action) {
  Set<V> discovered = new HashSet<>();
  Queue<V> frontier = new LinkedList<>();
  discovered.add(src); frontier.add(src);

  while(!frontier.isEmpty()) {
    V v = frontier.remove();
    action.accept(v);
    for (E edge : v.outgoingEdges()) {
      V neighbor = edge.head();
      if (!discovered.contains(neighbor)) {
        discovered.add(neighbor);
        frontier.add(neighbor);
}}}}
```

O(1)

discovery adds to frontier
one vertex removed per iteration

O(|V|) iterations, one per discovered vertex

O(1) · O(|V|) = O(|V|)

O(|V|) iterations per outer loop iteration, but O(|E|) total iterations

O(1) · O(|E|) = O(|E|)

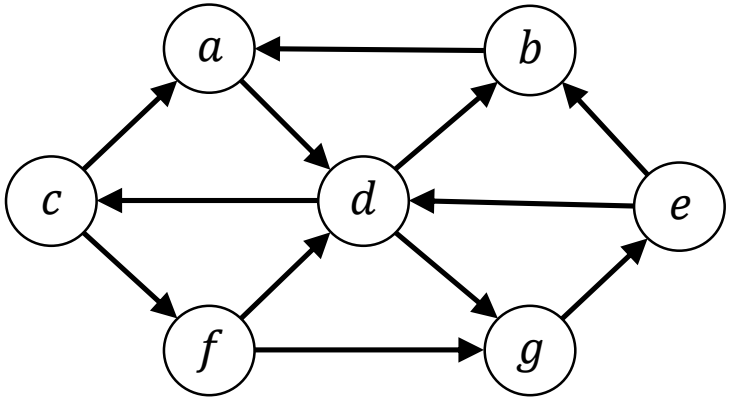
O(1) · O(|V|) = O(|V|)

Overall: O(|V|+|E|)
does same work as DFS, just in different order

Space Complexity O(|V|) for discovered, frontier

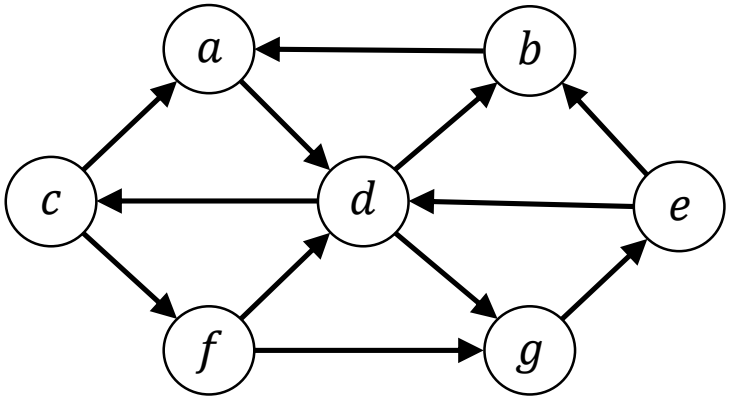
Traversal Levels

BFS sweeps across the graph in layers (or *levels*), expanding outward from the source.



Setting up Next Lecture

Highlight the edges that discovered new vertices during BFS



What structure do these edges form?
How do these edges help us find paths from the source vertex?