Graph Representation

Graph Representations

Definition

Graph: An ordered pair defined as $G = (V, E)$, where V is a set of vertices and E is a set of vertex pairs signifying edges

1. **Adjacency list:** Store a list of vertices, and maintain a list of connections from each vertex
2. **Edge list:** Store a list of (v, u) edges
3. **Adjacency matrix:** Store a $|V| \times |V|$ matrix with a 1 in a position if that edge exists

Graph Traversal

Breadth-first search

BFS$(G, start)$

- foreach vertex $v \in G$, label v undiscovered, $v.d \leftarrow \infty$
- $start$'s label $\leftarrow$ discovered, $d \leftarrow 0, \pi \leftarrow$ nil
- Q.enqueue($start$)
- **while** Q not empty **do**
  - $u \leftarrow Q$.dequeue
  - **for all** neighbor v of u **do**
    - **if** v is undiscovered **then**
      - label v discovered, $v.d \leftarrow u.d + 1, v.\pi \leftarrow u$
      - Q.enqueue(v)
  - label u finished

Which vertices does Q hold?
What is the time complexity?

Factoids

Each vertex is only added to Q once

$\delta(u,v)$ is the minimum number of edges to get from u to v

$v.d \geq \delta(start, v)$
Proof: Show that $v.d \geq \delta(start, v)\ \forall v$ via invariant:

1. Holds at start
2. $v.d$ is updated to $u.d + 1 \geq \delta(start, u) + 1 \geq \delta(start, v)$
3. At termination, $v.d = \delta(start, v) =$ shortest path length

Depth-first Search

DFS(G, $start$)

```
foreach vertex v ∈ G, label v undiscovered
DFS-VISIT(G, start)
```

DFS-VISIT(G, u)

```
label u discovered
for all neighbor v of u do
    if v is undiscovered then
        v.π ← u
        DFS-VISIT(G, v)
label u finished
```

Discovery and finished labels?

What is the time complexity?