DFS (Chap 3)
  $\rightarrow$ reachability
  $\rightarrow$ shortest paths

BFS (Chap 4)

$O(|V| + |E|)$

DFS

$\rightarrow$ stack (first in - last out)

Breadth First Search (BFS)

$\rightarrow$ queue (first in - first out)

Number of hops:

By edges

Region frontier

R: set of nodes within d hops
Inductive argument that BFS computes the shortest path from origin to each node.

Length: # of hops or # of edges on the path.

BFS: Shortest path for graphs that are unweighted.

Shortest path in weighted graphs:

$w(e)$: weight > 0

Edge
\[ \text{cost}(p) = \text{weight}(p) = \sum_{e \in p} w(e) \]

Shortest weighted path from origin to all other nodes.

**Dijkstra's Algorithm (BFS with cost)**

**Dijkstra** \((u)\):

- for all \( v \in V \)
  - \( \text{cost}(v) = \infty \)
  - \( \text{cost}(u) = 0 \)

Extract every vertex once from priority queue

For a given \( u \):
- for all neighbors \( x \) of \( u \)
  - if \( \text{cost}(u) + w(u, x) < \text{cost}(x) \):
    - \( \text{cost}(x) = \text{cost}(u) + w(u, x) \)
    - \( \text{prev}(x) = u \)

\( O(E) \) time; \( O(V \log V) \) worst case

\( \text{PQ: Priority Queue} \)
- 0) Create the queue
- 1) Extract min cost vertex
- 2) Decrease cost operation
Dijkstra's Time Complexity

1) Creation time
2) $O(|V|)$ extract min operations
3) $O(|E|)$ decrease cost operations

PQ: Array Implementation (unsorted)

$\text{dist}(A \rightarrow B) = 3$
$A \rightarrow C = 2$
$A \rightarrow D = 5$
$A \rightarrow E = 6$

$O(|V|) + O(|V|) \times O(|V|) + O(1|E|) \times O(1)$

1) Creation time $O(|V|)$
2) Extract min: scan the whole array $O(|V|)$
3) Decrease operation $O(1)$
\[ = O\left( |V|^2 + |E| \right) \]
\[ = O(|V|^2) \quad \text{array-based PQ.} \]

**PQ:** using binary tree.

- **extract min:** \( O(\log |V|) \)
- **decrease:** \( O(1) \)
- **creation:** \( O(|V|) \)

**Priority Queue:** binary tree implementation

**min-heap data structure**

**Invariant:** for each node \( u \) in the binary tree, \( \text{cost}(u) < \text{cost of any other two children} \)

\[ \Rightarrow \text{root of the tree is min} \]

```
Return 3
adj (the min heap)
```

```
tree keeps it
```
Decrease cost

Create PQ: \(\{9, 7, 10, 3, 1, 5, 4, 2, 6, 8\}\)

\(|V| = \log |V|\)

Naive:

top down
left \rightarrow right

generate
Linear time

\[
9 \ 7 \ 10 \ 3 \ 15 \ 1 \ 2 \ 6 \ 8
\]

\[
\log_2 n
\]

\[
\log_2 n \Downarrow
\]

Total case

\[
= \sum_{h=0}^{\log_2 n} h \cdot 2^h
\]

\[
= 2 \sum_{h=0}^{\log_2 n} 2^h
\]

\[
= 2 \sum_{h=0}^{\log_2 n} \frac{h}{2^h}
\]

\[
= n \sum_{h=0}^{\log_2 n} \frac{h}{2^h}
\]

\[
= \infty
\]

\[
= 1 / |1 - \gamma|
\]

\[
\text{general formula}
\]

\[
|\gamma| < 1
\]

\[
\sum_{h=0}^{\infty} h \cdot \gamma^h = \frac{\gamma}{(1 - \gamma)^2}
\]
\[ n \sum_{h=0}^{\infty} \left( \frac{1}{2} \right)^h = \frac{\frac{1}{2}}{(\frac{1}{2})^2} = 2n. \]

= \(O(n) = O(|V|)\)

**PQ:** min heap

\[ \rightarrow \text{create} = O(|V|) \]

\[ \rightarrow \text{decrease/extract} = O(\log |V|) \]

**Dijkstra:**

\[ O(|V|) + |V| \times O(\log |V|) + |E| \times O(\log |V|) \]

\[ = O\left(\frac{|V| + |E|}{\log |V|}\right) \]