Lecture: 5: Single Source Shortest Paths I

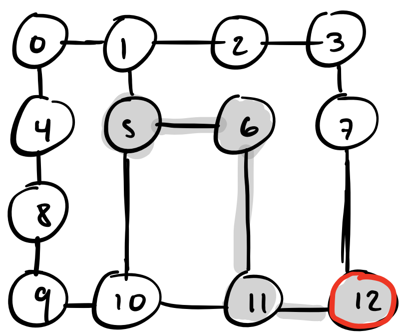
Unit: 6

Instructor: Néstor

Motivating problem: Given a weighted graph G, and a starting source vertex s, what are shortest paths from s to every other vertices of G?

I. SSSP on unweighted graph

- BFS is a natural option since it visits vertices layer by layer starting at some source vertex
  - Distance array: distance between two neighbors is 1
  - Parent array: we use this array to build the path



SSSP(5, 12): 5, 6, 11, 12

SSSP(5, 7): 5, 1, 2, 3, 7

II. SSSP on weighted graph

- BFS doesn't work because BFS only finds the path between two vertices with the minimum # of vertices → there can be a path with greater # of vertices and smaller weight

- To solve this problem: **Dijkstra's algorithm** $O(n^2 + m)$

⇓

Assumptions:
- Directed or undirected graph
- Weighted graph
- All weights non-negative

Description:

(0) Given a source node s

(1) Start with array distance D that will contain the length of the shortest path from s to any vertex v

- INITIALLY : $D[s] = 0$
  $D[v] = \infty \quad \forall v \in V \mid v \neq s$

- $D[v]$ WILL REMAIN ∞ IF v IS UNREACHABLE FROM NODE s AFTER n ITERATIONS

(2) MAINTAIN BOOLEAN ARRAY VISITED $\underline{U}$: STORES FOR EACH VERTEX WHETHER IT'S VISITED OR NOT

- INITIALLY : $U[v] = \text{FALSE} \quad \forall v \in V$

V: SET OF VERTICES
$|V| = n$

(3) THE ALGORITHM RUNS FOR n ITERATIONS, AT EACH ITERATION <u>VERTEX v</u> IS CHOSEN AS UNVISITED WHICH HAS THE LEAST VALUE $D[v]$

(IN THE FIRST ITERATION, VERTEX s IS CHOSEN)

(3.1) SELECTED VERTEX v IS MARKED AS VISITED

(3.2) ALL EDGES (v, to) ARE CONSIDERED. FOR EACH VERTEX <u>TO</u> WE TRY TO IMPROVE VALUE $\underline{D[to]}$:

- $D[to] = \min(D[to], D[v] + \text{WEIGHT}(v, to))$

(4) THE END

<u>RESTORING PATHS:</u>

- WE MAINTAIN AN ARRAY OF PREDECESSORS $\underline{P}$

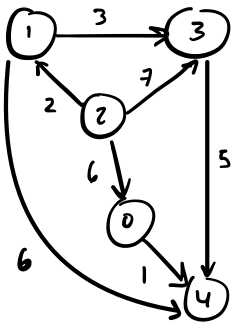
- FOR EACH VERTEX $v \neq s$, $P[v]$ IS THE PENULTIMATE VERTEX IN THE SHORTEST PATH FROM s TO v

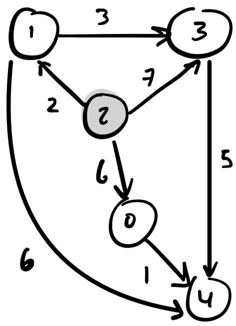
- ALGORITHM: STARTING FROM VERTEX v, WE TAKE ITS PREDECESSOR UNTIL REACHING VERTEX s

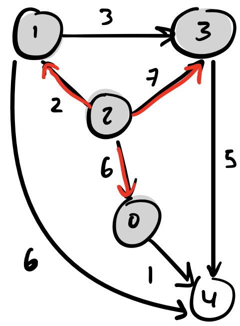
$$\text{PATH} = (s, \ldots, P[P[v]], P[v], v)$$

- BUILDING P : WHEN THERE IS AN IMPROVEMENT FOR SOME SELECTED VERTEX v, THERE IS AN IMPROVEMENT IN THE DISTANCE TO SOME VERTEX to, THEREFORE, WE UPDATE

$$P[to] = v$$





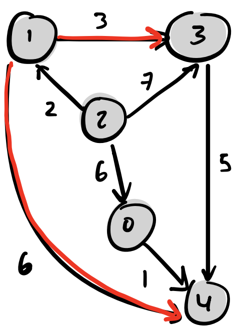


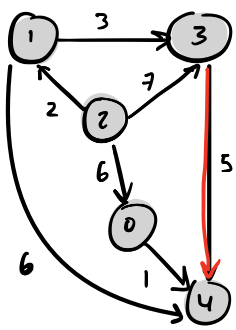
$\boxed{S = 2}$

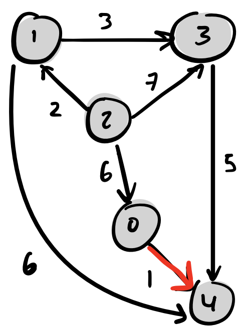
$$\begin{array}{l}
D = [\infty \;\; \infty \;\; \infty \;\; \infty \;\; \infty] \\
U = [0 \;\; 0 \;\; 0 \;\; 0 \;\; 0] \\
P = [-1 \;\; -1 \;\; -1 \;\; -1 \;\; -1] \\
\quad\;\;\; 0 \;\;\; 1 \;\;\; 2 \;\;\; 3 \;\;\; 4
\end{array}$$

$$\begin{array}{l}
D = [\infty \;\; \infty \;\; 0 \;\; \infty \;\; \infty] \\
U = [0 \;\; 0 \;\; 1 \;\; 0 \;\; 0] \\
P = [-1 \;\; -1 \;\; -1 \;\; -1 \;\; -1] \\
\quad\;\;\; 0 \;\;\; 1 \;\;\; 2 \;\;\; 3 \;\;\; 4
\end{array}$$

$$\begin{array}{l}
D = [6 \;\; 2 \;\; 0 \;\; 7 \;\; \infty] \\
U = [1 \;\; 1 \;\; 1 \;\; 1 \;\; 0] \\
P = [2 \;\; 2 \;\; -1 \;\; 2 \;\; -1] \\
\quad\;\;\; 0 \;\;\; 1 \;\;\; 2 \;\;\; 3 \;\;\; 4
\end{array}$$







$\min(D[3], D[1] + 3) = 5$

$\min(D[4], D[1] + 6) = 8$

$$\begin{array}{l}
D = [6 \;\; 2 \;\; 0 \;\; 5 \;\; 8] \\
U = [1 \;\; 1 \;\; 1 \;\; 1 \;\; 1] \\
P = [2 \;\; 2 \;\; -1 \;\; 1 \;\; 1] \\
\quad\;\;\; 0 \;\;\; 1 \;\;\; 2 \;\;\; 3 \;\;\; 4
\end{array}$$

$\min(8, D[3] + 5) = 8$

$$\begin{array}{l}
D = [6 \;\; 2 \;\; 0 \;\; 7 \;\; 8] \\
U = [1 \;\; 1 \;\; 1 \;\; 1 \;\; 1] \\
P = [2 \;\; 2 \;\; -1 \;\; 2 \;\; 1] \\
\quad\;\;\; 0 \;\;\; 1 \;\;\; 2 \;\;\; 3 \;\;\; 4
\end{array}$$

$\min(8, D[0] + 1) = 7$

$$\begin{array}{l}
D = [6 \;\; 2 \;\; 0 \;\; 7 \;\; 7] \\
U = [1 \;\; 1 \;\; 1 \;\; 1 \;\; 1] \\
P = [2 \;\; 2 \;\; -1 \;\; 2 \;\; 2] \\
\quad\;\;\; 0 \;\;\; 1 \;\;\; 2 \;\;\; 3 \;\;\; 4
\end{array}$$

III. DIJKSTRA ON SPARSE GRAPHS

- ORIGINAL DIJKSTRA'S $\rightarrow \underbrace{O(n^2 + m)}_{\text{OPTIMAL TIME COMPLEXITY ON DENSE GRAPHS WHERE } n^2 \approx m}$

- BUT ON DENSE GRAPHS WHERE $n >> m$ WE NEED TO OPTIMIZE THE FINDING OF THE NEXT UNVISITED VERTEX WITH MIN. DISTANCE

TWO OPERATIONS:

BOTH CAN BE DONE IN TIME $n \log n$ USING C++ STL SET OR PRIORITY_QUEUE ←

1. FIND UNVISITED VERTEX WITH MIN DISTANCE
2. UPDATING VALUE D[TO]

QUERYING AND UPDATING TIME OF $n \log n$

- SET IS BASED ON RED BLACK TREES
- P.Q. IS BASED ON HEAPS

DIJKSTRA'S NEW TIME COMPLEXITY : $O(n \log n + m \log n) = O(m \log n)$