Dynamic Programming

- All Pairs Shortest Paths Problem
- APSP: Dynamic Programming Formulation
- Floyd Warshall Algorithm

Imdad ullah Khan

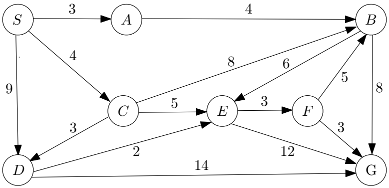
Weighted Graph

Weighted Graphs (digraphs)

- V : Set of vertices
- E : Set of edges (directed edges)
- w : cost/weight on each edge. $w : E \to \mathbb{R}$

  $\triangleright$ weights could be lengths, airfare, toll, energy
- Denoted by $G = (V, E, w)$

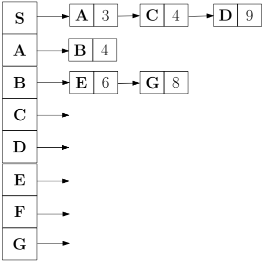
Weighted Graph Representation



Weighted Adjacency Matrix

	S	A	B	C	D	E	F	G
S	0	3	0	4	9	0	0	0
A	0	0	4	0	0	0	0	0
B	0	0	0	0	0	6	0	8
C	⋮							
D								
E								
F								
G								

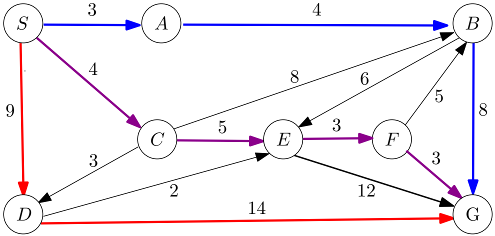
Weighted Adjacency Lists



Weight of Paths

Weight or length of a path $p = v_0, v_1, \ldots, v_k$ in weighted graphs is sum of the weights of its edges

$$C(p) \ = \ \sum_{i=1}^{k} w(v_{i-1}, v_i)$$



Three $S - G$ paths

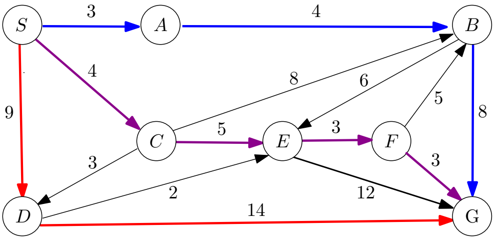
$\mathbf{C(p_1) = 3 + 4 + 8}$

$\mathbf{C(p_2) = 4 + 5 + 3 + 3}$

$\mathbf{C(p_3) = 9 + 14}$

Unweighted graphs are weighted graphs with unit edge weights

Shortest Paths



Three $S-G$ paths

$\mathbf{C(p_1) = 3 + 4 + 8}$

$\mathbf{C(p_2) = 4 + 5 + 3 + 3}$

$\mathbf{C(p_3) = 9 + 14}$

Shortest path from s to t is a path of smallest weight

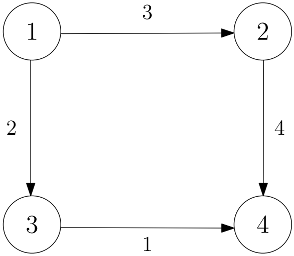
> Distance from s to t, $\mathbf{d(s,t)}$: weight of the shortest $s - t$ path

There can be multiple shortest paths

APSP Problem

Input: A weighted graph $G = (V, E, w)$
Output: Shortest paths from every vertex $u \in V$ to every other $v \in V$

The APSP problem can be represented by a $n \times n$ matrix $D = [d_{ij}]$, where $d_{ij} = d(u_i, u_j)$ for $i, j = 1, \ldots, n$, and n is the number of vertices in V.



	1	2	3	4
1	0	3	2	3
2	∞	0	∞	4
3	∞	∞	0	1
4	∞	∞	∞	0

The goal is to compute the matrix D efficiently.

APSP Applications

- Network routing: finding the optimal routes between any pair of nodes in a network
- Social network analysis: measuring the closeness or centrality of nodes in a social graph
- Bioinformatics: comparing the similarity of biological sequences or structures
- Computer vision: matching features or objects in images or videos
- Machine learning: computing the kernel matrix or the graph Laplacian for graph-based methods

APSP Applications

The APSP problem is also a building block for solving other graph problems, such as:

- Transitive closure: determining if there is a path between any pair of nodes in a graph
- Diameter: finding the longest shortest path in a graph
- Eccentricity: finding the maximum distance from a node to any other node in a graph
- Betweenness centrality: measuring the importance of a node based on the number of shortest paths passing through it