Dynamic Programming

- Single Source Shortest Paths Problem
- Dynamic Programming Formulation
- Bellman-Ford Algorithm

Imdad ullah Khan

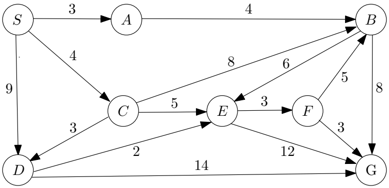
Weighted Graph

Weighted Graphs (digraphs)

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

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

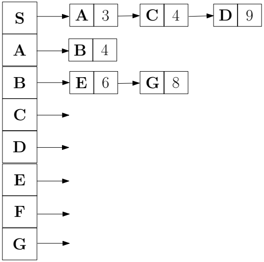
Weighted Graph Representation



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								

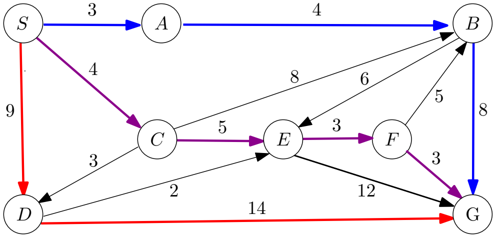
Weighted Adjacency Lists



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

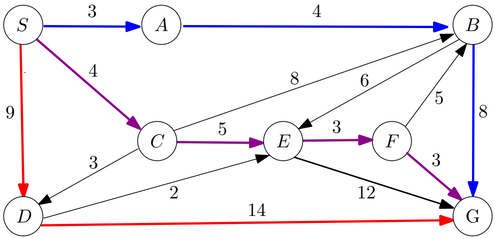
$\mathbf{C(p_1) = 3 + 4 + 8}$

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

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

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

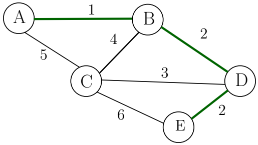
Distance from s to t, $\mathbf{d(s, t)}$: weight of the shortest $s - t$ path

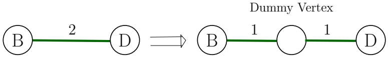
There can be multiple shortest paths

SSSP Problem

Input: A weighted graph G and a source vertex $s \in V$
Output: Shortest paths from s to all vertices $v \in V$

- For unweighted graphs (unit weights) BFS from s will work
- BFS running time: $O(n + m)$
- For weighted graph replace each edge e by directed path of $w(e)$ unit weight edges





- What if weights are not integers or are negative
- Blows up size of the graph a lot
- Dijkstra's algorithm for shortest paths in weighted graph

Dijkstra Algorithm

Input: A weighted graph G and a source vertex $s \in V$
Output: Shortest paths from s to all vertices $v \in V$

Dijkstra's algorithm solves SSSP for both directed and undirected graphs

Assumptions:

1. All vertices are reachable from s
   - Otherwise there is no shortest path (distance $= \infty$)
   - Easy to get $R(s)$ in preprocessing (e.g., BFS or DFS)
2. All edge weights are non-negative
   - Bellman-Ford algorithm deals with negative weights

Dijkstra Algorithm with negative weights

Input: A weighted graph G and a source vertex $s \in V$
Output: Shortest paths from s to all vertices $v \in V$

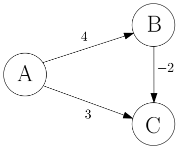
Shortcomings of Dijkstra's Algorithm

1. No guarantee to work with negative weights
   - Graph with negative weights on edges are not uncommon
   - e.g. vertices could represent companies
   - Edges could be financial transactions
   - weights of edges could be return to source company from transactions
2. Assumes global knowledge of the graph (very centralized algorithm)
   - Cannot be used for Internet routing
   - Used in BGP

Dealing with negative weights

Input: A weighted graph G and a source vertex $s \in V$
Output: Shortest paths from s to all vertices $v \in V$

Multiply each edge weight with -1



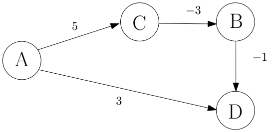
- Already positive edge weights become negative

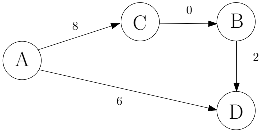
Dealing with negative weights

Input: A weighted graph G and a source vertex $s \in V$
Output: Shortest paths from s to all vertices $v \in V$

Shift all weights to positive side, i.e. add a constant C to each weight

$$C \geq \max_{e \in E} |w(e)|$$



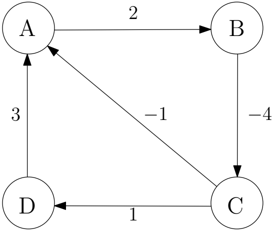


- $d(A, D) = 1$ via the path $ACBD$
- Now AD is the shortest path from A to D with length 6
- Longer paths (w.r.t hop count) get more weight

Negative Cycle

Some applications of SSSP may involve graphs with negative edge weights, e.g. modeling cash flows, heat dissipation, or arbitrage opportunities

Negative edge weights can create **negative cycles**, cycles whose sum of edge weights is negative



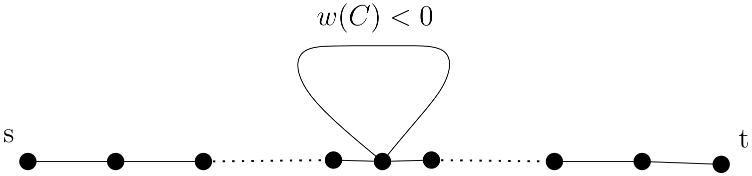
The cycle $\langle A, B, C, A \rangle$ in the graph has a negative weight of -3

Negative Cycle

Negative cycles make the notion of shortest path ill-defined

We can go around the cycle an infinite number of times and reduce the path length indefinitely

> If there is a negative cycle on a $s \to t$ path, then there is no shortest path from s to t

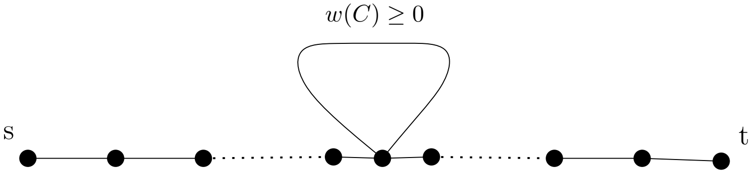


Negative Cycle

Lemma: If G does not have any negative cycles and t is reachable from s, then there exists a cheapest path from s to t that is simple, and has at most $|V| - 1$ edges

Proof: If the cheapest $s \to t$ path, P has $\geq |V|$ edges, then P contains a cycle

Removing the cycle gives a path cheaper than P



SSSP Problem

Input: A weighted graph G and a source vertex $s \in V$
Output: Shortest paths from s to all vertices $v \in V$

This assumes no negative weight cycle in G

$\triangleright$ Actually, no negative cycle reachable from the source vertex s is sufficient

Input: A weighted graph G and a source vertex $s \in V$
Output: Shortest paths from s to all $v \in V$ or output a negative cycle