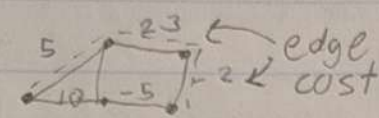
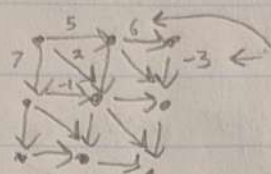


CSCI 1810

Longest Paths Problems and Shortest Paths Problems in graphs



Undirected graph



directed graph

... path has cost $5 + 2 + 3 - 2 = 26$

longest path = path of maximum cost/score
alignment $\leftrightarrow$ directed path

Our maximal alignment is the longest path in the edit graph.

The shortest path problem is solvable in polynomial time.

- Dijkstra's algorithm

$O(|V|^2)$

$G = (V, E)$

$\uparrow$
graph

DAGs = Directed Acyclic Graphs

- Dijkstra's algorithm is $O(|V| + |E|)$ on DAGs.

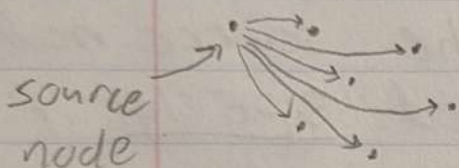
With DAGs we can solve optimally the shortest path and the longest path with the same algorithm.

$G \rightarrow -G$ Multiple by (-1) the cost of every edge to calculate longest path with shortest path algorithm.

The longest path Problem is NP-complete
-computationally Intractable

Famous problem is P vs NP ^{Believed universally} $P \neq NP$

$G = (V, E)$



Dijkstra's Algorithm (shortest path)

$G = (N, A)$ directed graph

N = set of nodes / vertices

A = set of edges

Each edge has a nonnegative cost/length.
One node is the source node/vertex.

Problem:

- Given: the graph G , the set of nodes $N = \{1, 2, 3, \dots, n\}$, the edges A , where (i, j) is an edge. $L[i, j]$ is the cost of edge (i, j)

Compute: the cost of the shortest paths from the source node to each of the other nodes in the graph G , and the shortest paths.

Dijkstra's Algorithm

Two sets C, S of vertices.

S = set of vertices already chosen

C = set of candidate vertices

S contains all the vertices whose minimal path cost from the source is already known.

source (1) -2

•7 •3 •4

•6 •5

C = the rest of the vertices (not in S).
At each step, we chose a node in C whose cost of the path to the source is smallest and add it to S .

Definition: A path from the source node to a vertex v is called special if all intermediary nodes along the path are in S .

D a vector containing the length/cost of the shortest special path to each node in G .

When we add a new node from C , v , to S , then the shortest path to v is the shortest path in the graph.

The algorithm ends with all nodes in S .

D gives us the solution: shortest paths cost

$$N = \{1, 2, \dots, n\}$$

$L[i, j] \geq 0$ if edge (i, j) exists in A
cost of edge (i, j) .

$$L[i, j] = \infty \text{ if } (i, j) \notin A$$

{Initialization}

$$S = \{1\} \text{ (source node = 1)}$$

$$C = \{2, 3, \dots, n\}$$

For $i = 2$ to n do:

$$D[i] = L[1, i]$$

{Greedy loop}

Repeat $n-2$ times:

v = node in C with minimum $D[v]$

$$S = S \cup \{v\}$$

$$C = C \setminus \{v\}$$

For $w \in C$ do:

IF $D[w] > D[v] + L[v, w]$ then:

$$D[w] = D[v] + L[v, w]$$