

Longest paths in directed graphs (maximum score path evaluations)

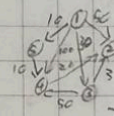
Examining using shortest paths (longest paths is NP-complete, so only really brute forceable)

Since in DAGs we can use shortest path where the sign of edge weights are flipped to find the longest (Linear time!)

In DAGs shortest path in G
longest path in -G & vice versa

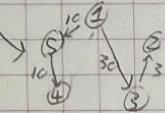
Topological order will be a key concept

Example of Dijkstra's algorithm:



step	v	C	D
0	-	$\{2, 3, 4, 5\}$	$\begin{bmatrix} 2 & 3 & 4 & 5 \\ 10 & 5 & 10 & 10 \end{bmatrix}$
1	5	$\{2, 3, 4\}$	$\begin{bmatrix} 2 & 3 & 4 & 5 \\ 10 & 5 & 10 & 10 \\ 0 & 30 & 20 & 10 \end{bmatrix}$
2	4	$\{2, 3\}$	$\begin{bmatrix} 2 & 3 & 4 & 5 \\ 10 & 5 & 10 & 10 \\ 0 & 30 & 20 & 10 \\ 0 & 30 & 20 & 10 \end{bmatrix}$
3	3	$\{2\}$	$\begin{bmatrix} 2 & 3 & 4 & 5 \\ 10 & 5 & 10 & 10 \\ 0 & 30 & 20 & 10 \\ 0 & 30 & 20 & 10 \\ 0 & 30 & 20 & 10 \end{bmatrix}$

step	P
0	$[1, 1, 1, 1]$
1	$[1, 1, 5, 1]$
2	$[4, 1, 5, 1]$
3	$[3, 1, 5, 1]$



Dijkstra's Algorithm: Greedy since at each step we choose the stepwise optimum

Let $G = \langle N, A \rangle$ be our directed graph, where $N = \{nodes\}$, $A = \{edges\}$

Each edge has a non-negative cost

Label the nodes $N = \{1, \dots, n\}$ & choose one as the source node s

Given $G \Delta L(i, j)$ the cost of the edge from i to j , we will compute

the cost of the shortest path from the source to each other node

Define two sets, $C \Delta S$. S is the set of already chosen nodes,

C is the set of candidate nodes. At any step, $C \cup S = N$, &

S contains all the nodes i whose minimal cost from s to i is known.

At each step, we choose $c \in C$ s.t. $L(s, c)$ is minimal and add c to S

Definition: A path from s to $v \in N$ is special if $\forall i \in \text{path}, i \in S$.

We will use an array D to hold the cost of the shortest special path to each node $i \in N$

When we add a new node v from C to S , the shortest special path to v is also the shortest path from s to v

$L(i, j) = \infty$ if $(i, j) \notin A$ holds the previous node on the optimal path to i

The Algorithm:

$S = \{1\}$ $P[i] = 1 \forall i \in \{2, 3, \dots, n\}$

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

For $i = 2$ to n , do:

$D[i] = L(1, i)$ The Greedy Loop

Repeat $n-2$ times

Let $v \in C$ be chosen s.t. $\forall c \in C, D[v] \leq D[c]$

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

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

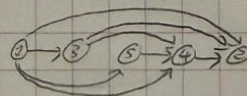
For each $w \in C$ do:

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

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

$P[w] = v$

In the above graph, we have 1, 3, 5, 4, 2



Topological Sorting Algorithm for DAG $G = (V, E)$

Let L be the list of topologically sorted vertices & v always at (last entry) DAG

$L \Delta S$ be the set of vertices with no incoming edges

While $S \neq \emptyset$ do:

$S = S \setminus \{u\}$ where $u \in S$

Add u to the tail of L

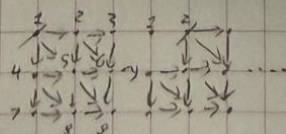
For each $v \in S$ s.t. $E = \{(u, v) \in E\} \neq \emptyset$

$E = E \setminus E_v$ edge by edge

when $\{(u, v) \in E\} = \emptyset$

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

Topological order isn't necessarily unique.



$L = \{1, 2, 3, 5, 7, 6, 8, 4\}$

$S = \{4, 2, 4, 3 \dots\}$