



Design and Analysis
of Algorithms I

Graph Primitives

Dijkstra's Algorithm: The Basics

Single-Source Shortest Paths

Input: directed graph $G=(V, E)$. ($m=|E|$, $n=|V|$)

- each edge has non negative length l_e
- source vertex s

Output: for each $v \in V$, compute
 $L(v) :=$ length of a shortest s - v path in G

Assumption:

1. [for convenience] $\forall v \in V, \exists s \Rightarrow v$ path
2. [important] $l_e \geq 0 \quad \forall e \in E$

use bellman ford (DP) for negative edge lengths

Length of path
= sum of edge lengths



Path length = 6

you can always detect which edges aren't reachable from S as a preprocessing step by applying BFS/DFS. And then delete that part from graph. Alternatively dijk will naturally figure out which vertices are reachable from S .

So this is a minor point

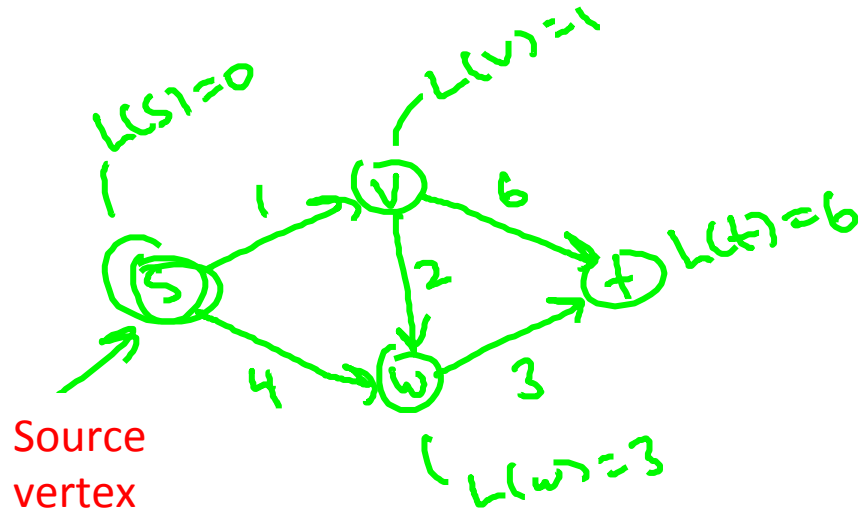
One of the following is the list of shortest-path distances for the nodes s, v, w, t , respectively. Which is it?

☐ 0,1,2,3

☐ 0,1,4,7

☐ 0,1,4,6

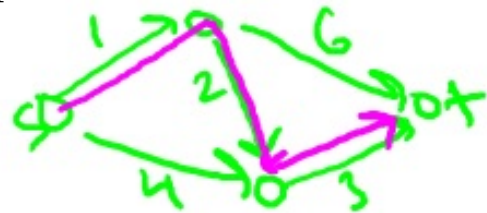
☒ 0,1,3,6



Why Another Shortest-Path Algorithm?

Question: doesn't BFS already compute shortest paths in linear time?

Answer: yes, IF $l_e = 1$ for every edge e .



Question: why not just replace each edge e by directed path of l_e unit length edges:



Answer: blows up graph too much

Solution: Dijkstra's shortest path algorithm.

Dijkstra's Algorithm

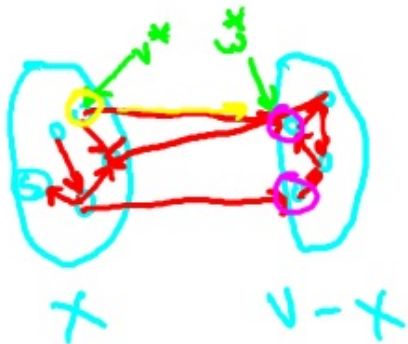
This array
only to help
explanation!

Initialize:

- $X = [s]$ [vertices processed so far]
- $A[s] = 0$ [computed shortest path distances]
- $B[s] = \text{empty path}$ [computed shortest paths]

Main Loop

- while $X \neq V$:



-need to grow
x by one node

Main Loop cont'd:

- among all edges $(v, w) \in E$ with $v \in X, w \notin X$, pick the one that minimizes

$$A[v] + l_{vw}$$

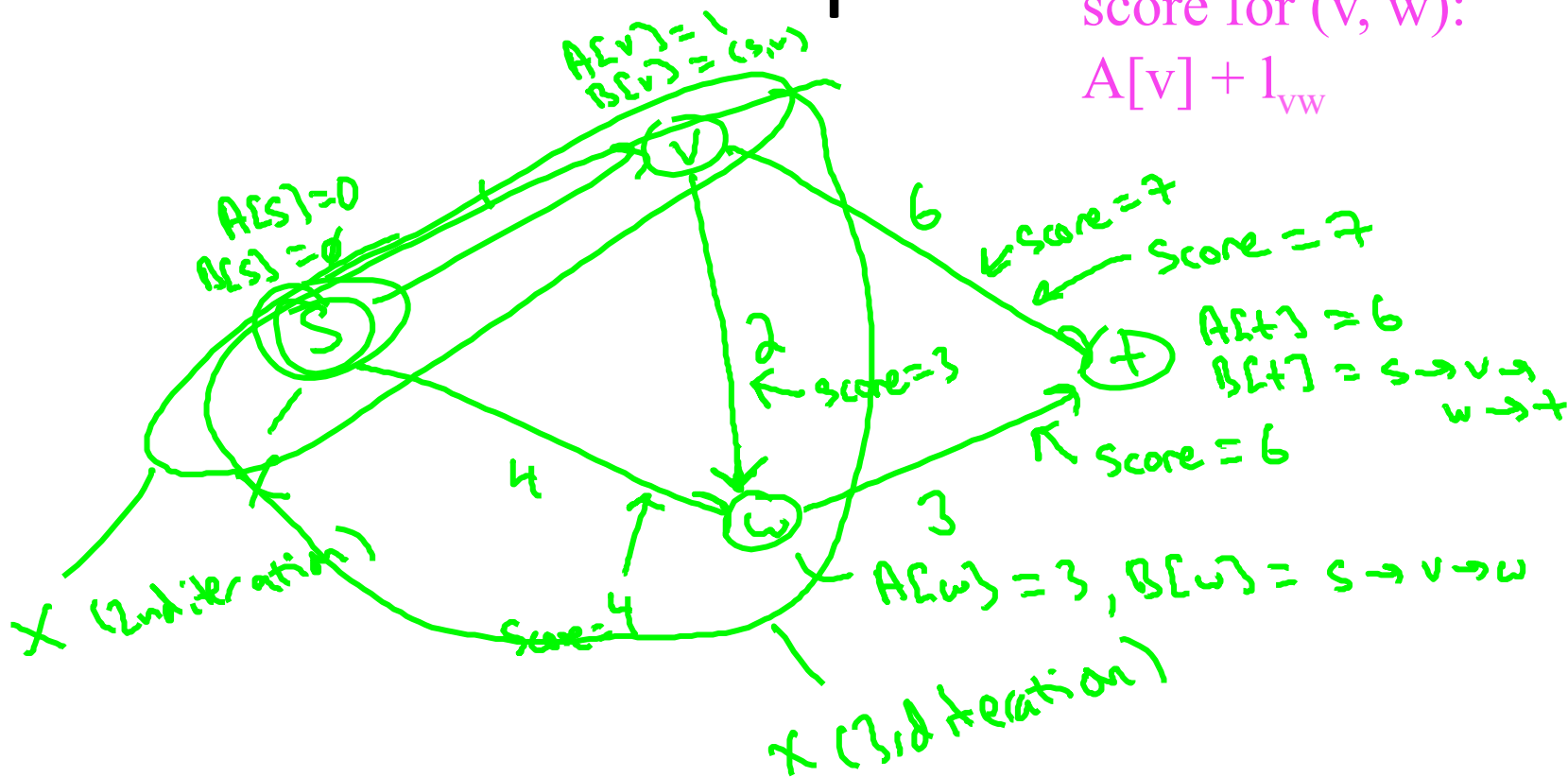
[call it (v^*, w^*)]

→ Already
computed in
earlier iteration

- add w^* to X
- set $A[w^*] := A[v^*] + l_{v^*w^*}$
- set $B[w^*] := B[v^*] \cup (v^*, w^*)$

Example

Dijkstra's greedy
score for (v, w) :
 $A[v] + l_{vw}$



Non-Example

Question: why not reduce computing shortest paths with negative edge lengths to the same problem with non negative lengths? (by adding large constant to edge lengths)

Problem: doesn't preserve shortest paths !

Also: Dijkstra's algorithm incorrect on this graph !
(computes shortest s-t distance to be -2 rather than -4)

