Department of Industrial Engineering & Operations Research

IEOR 162 Linear Programming and Network Flows (Spring 2022)

Dynamic Programming for DAG (Directed Acyclic Graph)

Requirements

The graph has no cycles. Negative costs are allowed.

Concept: Topological ordering

Each node $i \in V$ is assigned a unique number $\ell(i)$ between $1, \ldots, n$, such that $\ell(i) < \ell(j)$ for all $(i, j) \in A$. In other words, "all arcs point to a node with a higher label".

Input

- Graph $G = (V, A)$ with arc cost c_{ij}.
- Source node s.

Output

- $d(i)$: Distance from s to i for all $i \in V$.
- $\text{pred}(i)$: Predecessor of node i for all $\in V$.

Steps

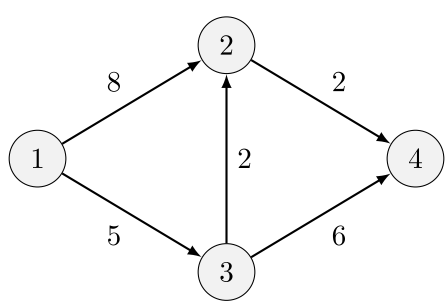
1. Find topological ordering $\ell(\cdot)$ by assigning each node a unique number between $1, \ldots, n$ where n is the number of nodes. This is the order in which we process the nodes.
2. Set $d(s) = 0$.
3. In order of increasing $\ell(j)$, update the distance and predecessor for node j:

$$d(j) = \min_{i\,:\,(i,j)\in A} \{d(i) + c_{ij}\} \qquad \textbf{(Update rule)}$$

$$\text{pred}(i) = \arg\min_{i\,:\,(i,j)\in A} \{d(i) + c_{ij}\}$$

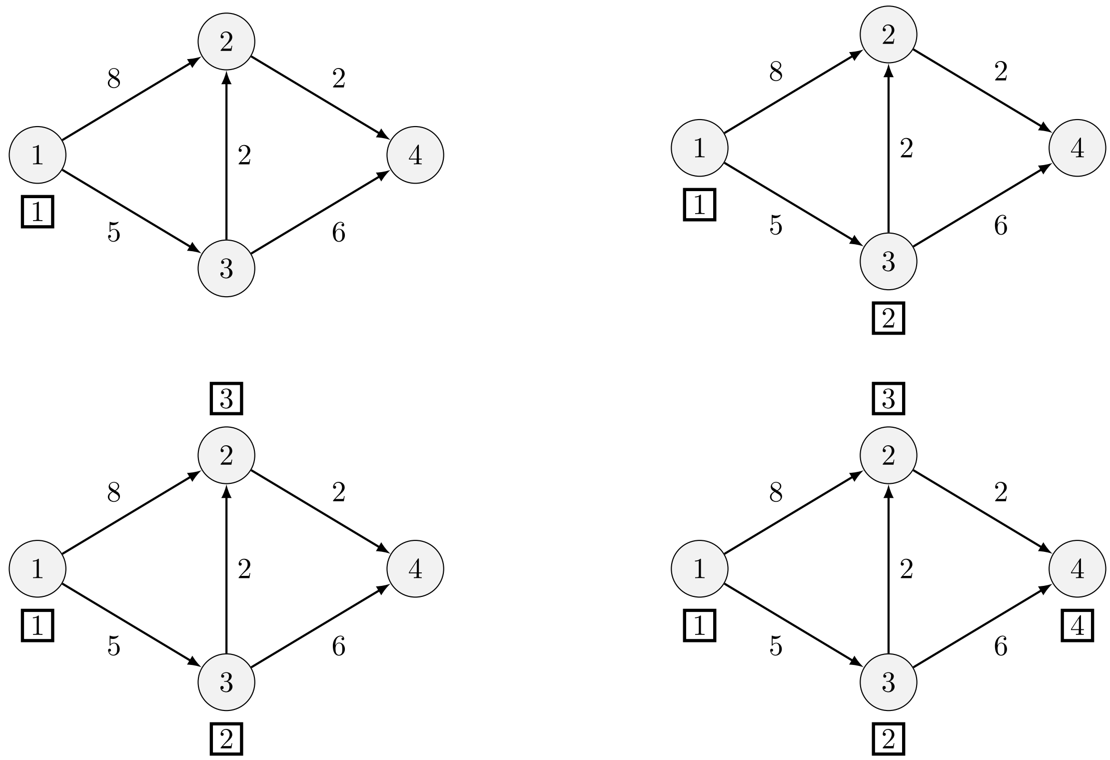
Example

We want to find the shortest paths from 1 to all other nodes.



Step 1: Topological sort

Iteratively label the node with no incoming arcs from unlabeled nodes with the lowest unused label.



Step 2 & 3: Computing the distances

Record in table:

Node	1	2	3	4
Dist (pred)				

Node 1 : $d(1) = 0$.

Node 3 : $d(3) = \min\{0 + 5\} = 5$, $\text{pred}(3) = 1$.

Node 2 : $d(2) = \min\{0 + 8, 5 + 2\} = 7$, $\text{pred}(2) = 3$.

Node 4 : $d(4) = \min\{7 + 2, 5 + 6\} = 9$, $\text{pred}(4) = 2$.