2019-Winter Sogang ACM

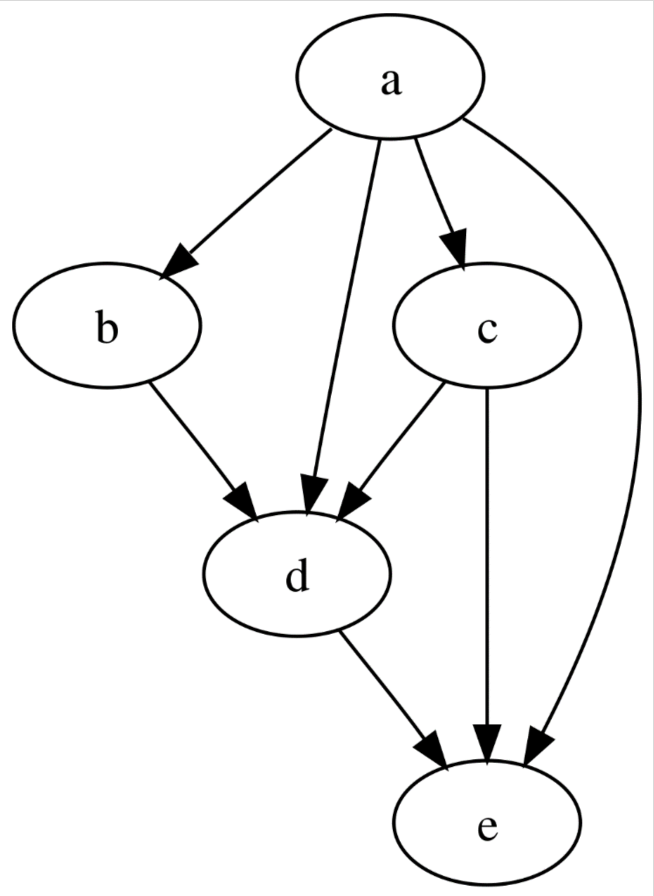
Topological Sort

서강대학교 엄태경

Terminology

DAG

- Directed Acyclic Graph
- Directed edges, no path from a vertex to itself



Topological Sort

- Ordering of vertices
- Only possible on DAG
- For every edge (*u*, *v*), *u* comes before *v*
- Example: abcde, acbde

Topological Sort

Usage

- Determine order of vertices
- Scheduling with dependency

Advanced Algorithms

- Shortest Path Algorithm in O(|V|+|E|)
- Strongly Connected Components
- 2-Satisfiability Problem

Kahn’s Algorithm (1)

Pseudo-code

```
L ← Empty list
Q ← Queue of vertices with no incoming edge
while Q is non-empty:
    remove a vertex n from Q
    add n to tail of L
    for each vertex m with an edge (n,m):
        remove edge (n,m) from the graph
        if m has no other incoming edges:
            insert m into Q
if graph has edges then
    return error (graph has cycle)
else
    return L (a topologically sorted order)
```

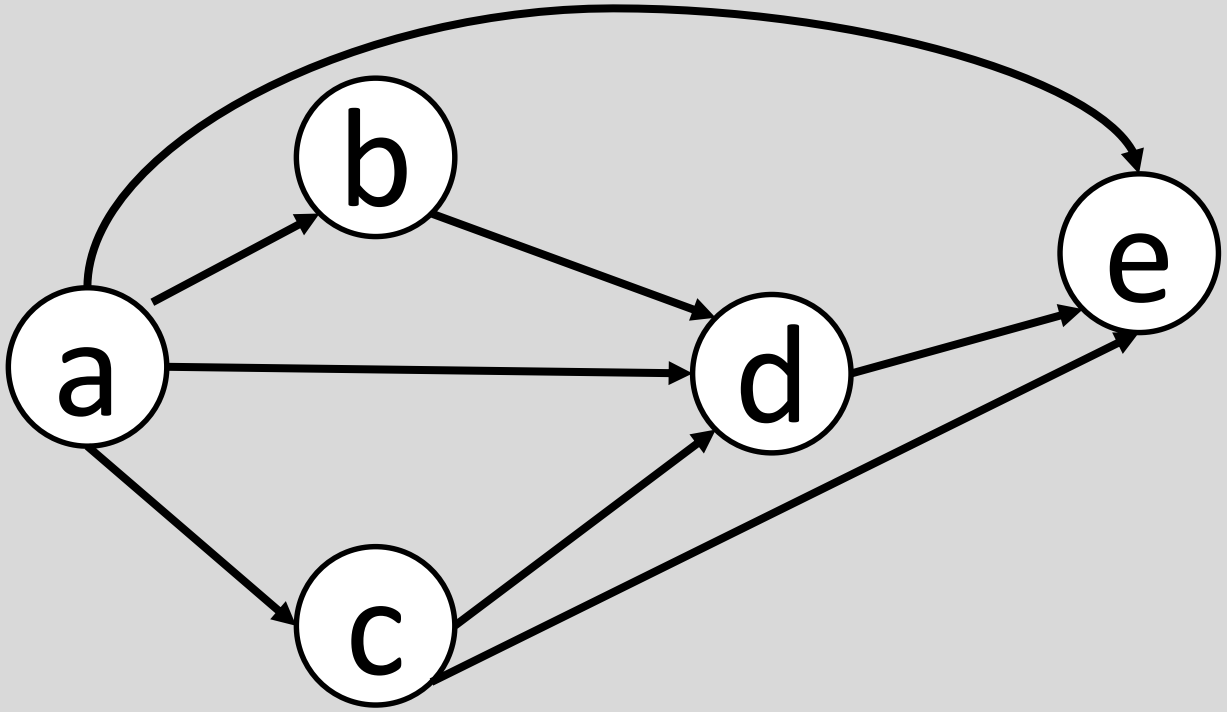
https://github.com/thebarbershop/sogang-acmicpc-2019-winter

Kahn's Algorithm (2)

Example Step 1

- L = [ ], S = {a}
- In-degree (# of incoming edges)

a	b	c	d	e
0	1	1	3	3

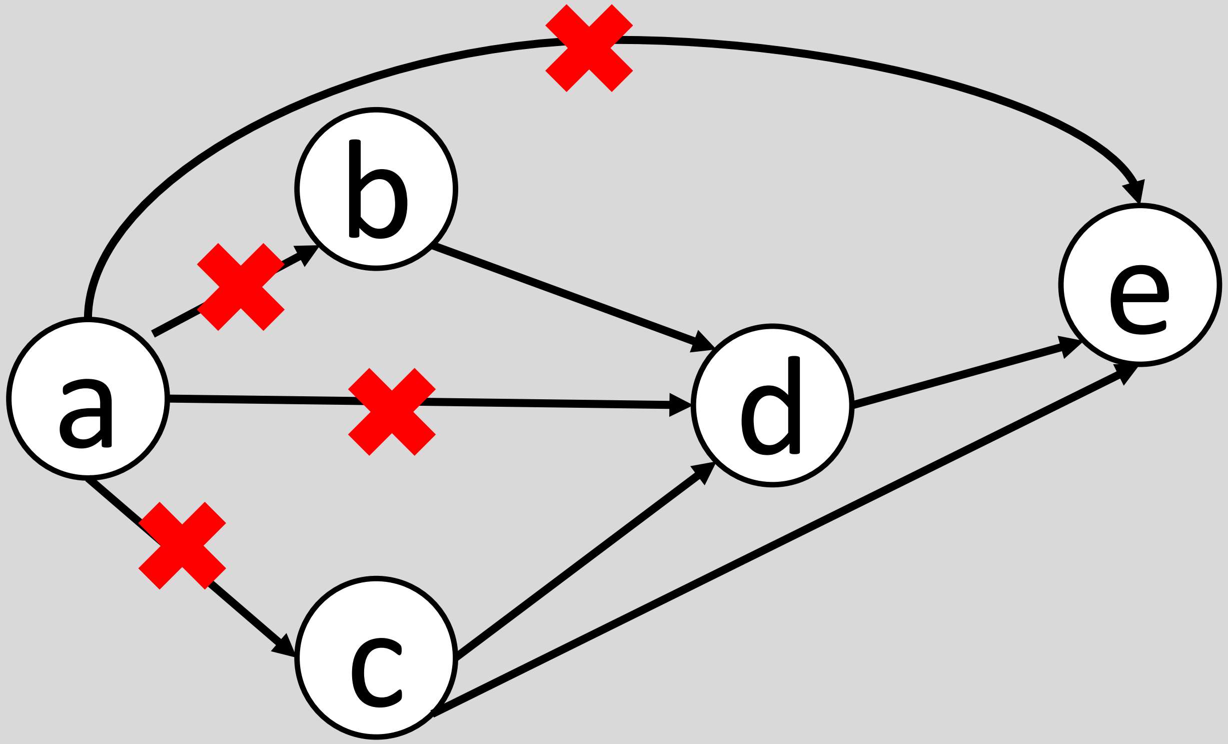


Kahn's Algorithm (3)

Example Step 2

- L = [a], S = {b, c}
- In-degree (# of incoming edges)

a	b	c	d	e
0	0	0	2	2



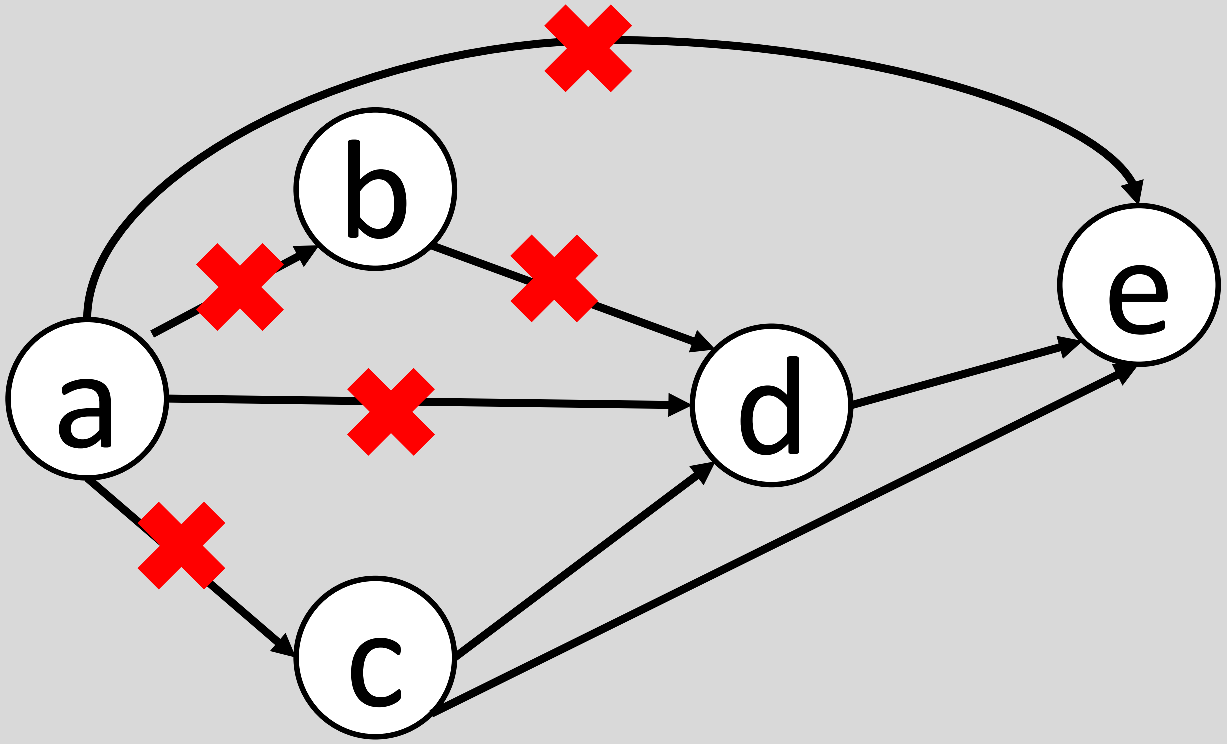
https://github.com/thebarbershop/sogang-acmicpc-2019-winter

Kahn’s Algorithm (4)

Example Step 3

- L = [a, b], S = {c}
- In-degree (# of incoming edges)

a	b	c	d	e
0	0	0	1	2

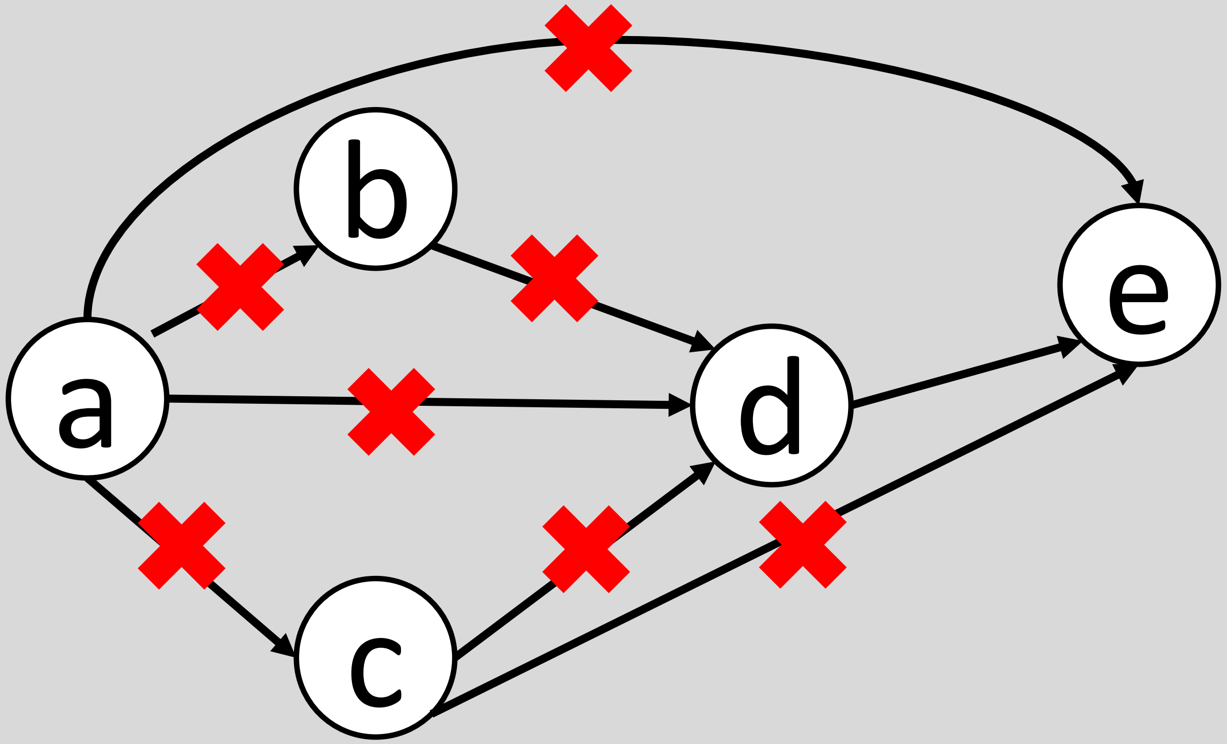


Kahn's Algorithm (5)

Example Step 4

- L = [a, b, c], S = {d}
- In-degree (# of incoming edges)

a	b	c	d	e
0	0	0	0	1

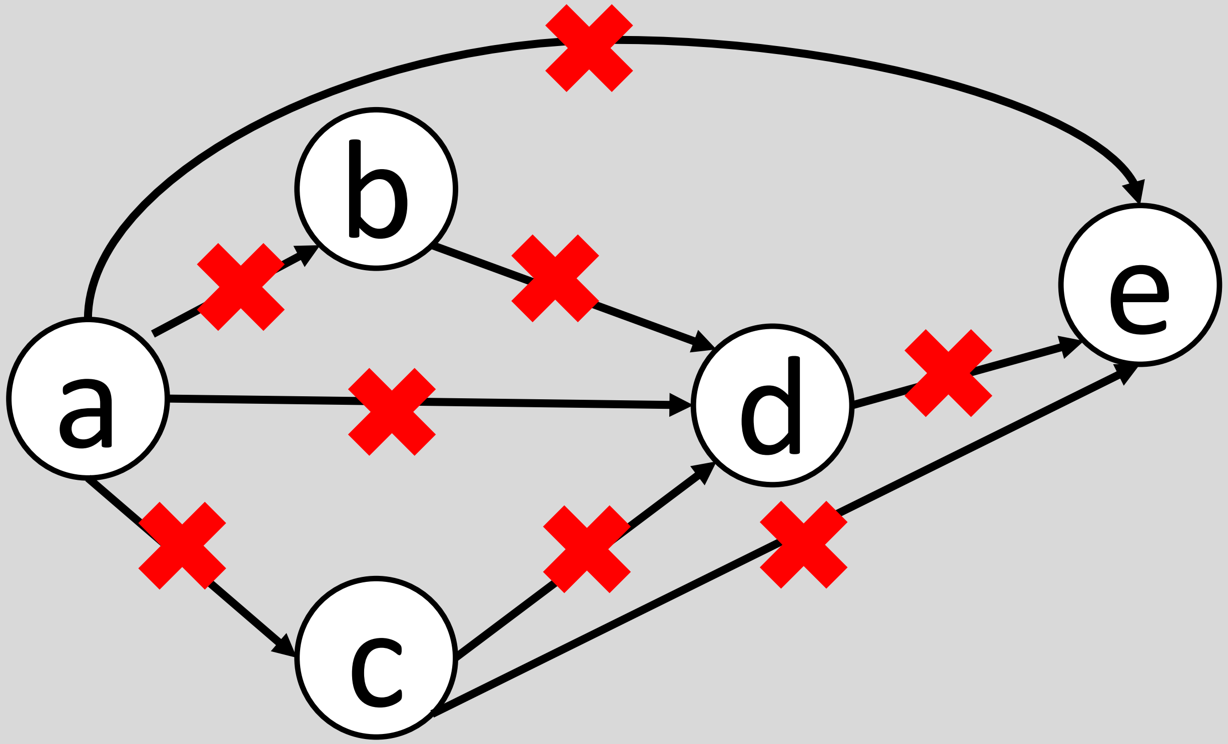


Kahn’s Algorithm (6)

Example Step 5

- L = [a, b, c, d], S = {e}
- In-degree (# of incoming edges)

a	b	c	d	e
0	0	0	0	0



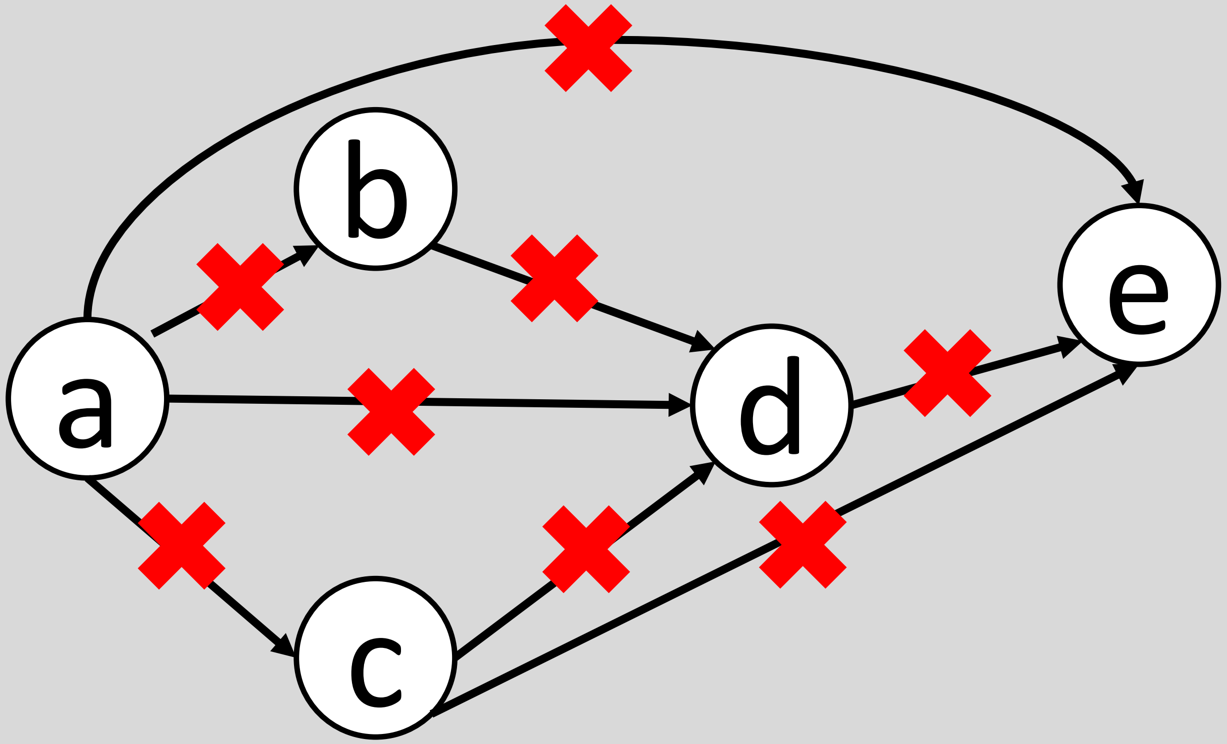
https://github.com/thebarbershop/sogang-acmicpc-2019-winter

Kahn’s Algorithm (7)

Example Step 6

- L = [a, b, c, d, e], S = {}
- In-degree (# of incoming edges)

a	b	c	d	e
0	0	0	0	0



https://github.com/thebarbershop/sogang-acmicpc-2019-winter

DPS Approach

Pseudo-code

```
L ← Empty list
while there are unmarked nodes:
    select an unmarked node n
    visit(n)

 function visit(node n)
    if n has a permanent mark: return
    if n has a temporary mark: stop (not a DAG)
    set temporary mark on n
    for each node m with edge (n,m):
        visit(m)
    set permanent mark on m
    add n to head of L
```

https://github.com/thebarbershop/sogang-acmicpc-2019-winter

Shortest Path

Pseudo-code

```
Shortest-path(node s, graph G=(V,E)):
  D ← array of length ¦V¦
  Initialize D[s] = 0, all other D[i] = ∞
  P ← array of length ¦V¦
  Initialize all P[i] = null

  for each vertex u in topological-order(G):
      if u is topologically before s: continue
      for each vertex v with edge (u,v):
          w ← weight of edge (u,v)
          if d[v] > d[u] + w:
              d[v] ← d[u] + w,
              p[v] ← u
```

https://github.com/thebarbershop/sogang-acmicpc-2019-winter

Practice

BOJ

- 3665 최종 순위
- 2252 줄세우기
- 3665 최종 순위
- 1948 임계경로
- 1005 ACM Craft

https://github.com/thebarbershop/sogang-acmicpc-2019-winter