Decrease-and-Conquer

Directed Graph (digraph)

- Presented by:
  - Adjacency list
  - Adjacency Matrix

  - Matrix is not necessarily symmetric
  - Each edge has only one representation in list or matrix

Directed Graph (digraph)

- Three type of edge on DFS and BFS tree:
  - Forward: from node to descendants other than children
  - Back: from node to ancestors
  - Cross: not forward or backward

  - Presence of back edge → digraph has directed cycle
  - No back edge → digraph is directed acyclic graph (DAG)

Topological Sorting

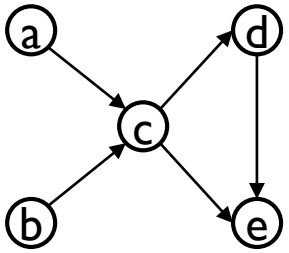
- List nodes in such order that for every edge in graph, the start node is listed before the end node

- Cannot be solved if digraph has a directed cycle
- To be possible, digraph must be a DAG

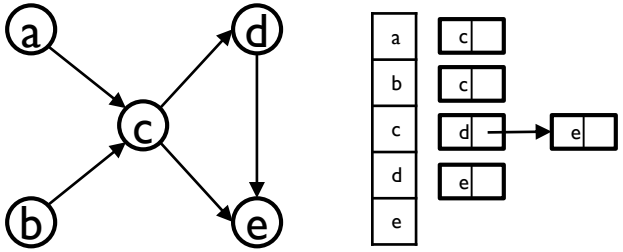
- 2 approaches:
  - DFS
  - Source removal

Topological Sorting - DFS

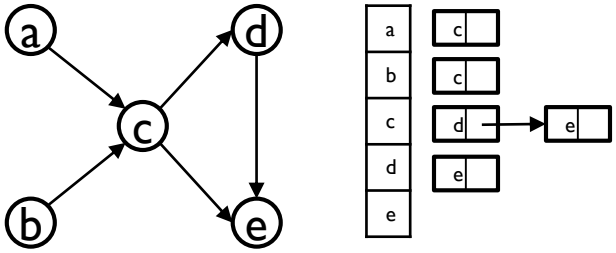
- Start from a node with no incoming edge
- No such node → exit with no solution
- Perform DFS
- Keep track of the order the nodes become dead end
- Repeat for each section of the graph
- Encounter back edge → exit with no solution
- Reverse the order



Topological Sorting - DFS



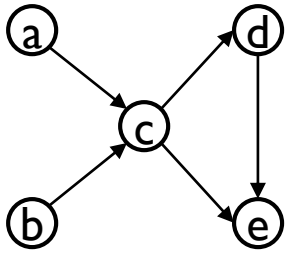
Topological Sorting - DFS

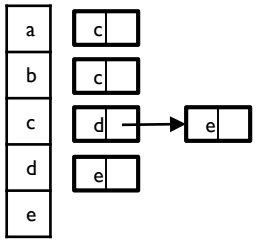


Dead end order: e → d → c → a → b
Solution order: b → a → c → d → e

Topological Sorting - DFS

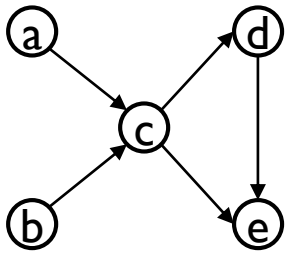
- Start from a node with no incoming edge
- No such node → exit with no solution
- Perform DFS
- Keep track of the order the nodes become dead end
- Repeat for each section of the graph
- Encounter back edge → exit with no solution
- Reverse the order

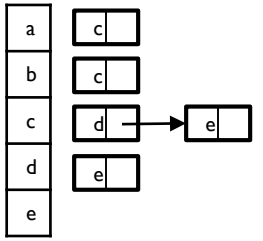




Topological Sorting - DFS

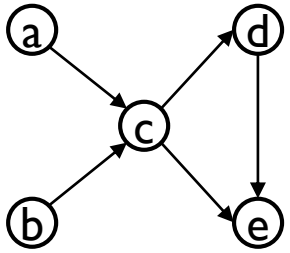
- Start from a node with no incoming edge $O(m)$
- No such node → exit with no solution
- Perform DFS
- Keep track of the order the nodes become dead end $O(m+n)$
- Repeat for each section of the graph
- Encounter back edge → exit with no solution
- Reverse the order $O(n)$
- $T(n,m) = m + m + n + n \in O(m+n)$



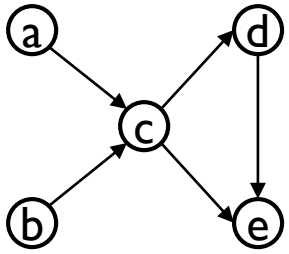


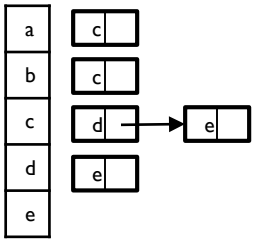
Topological Sorting – Source removal

- Start from a node with no incoming edge
- No such node → exit with no solution
- While there is node remaining,
  - Identify a node with no incoming edge
  - No such node → exit with no solution
  - Else delete it and all of its outgoing edges
  - Add to the topological order



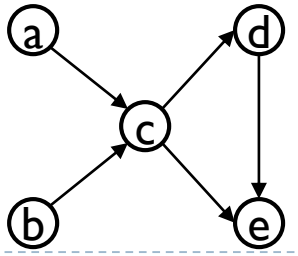
Topological Sorting – Source removal

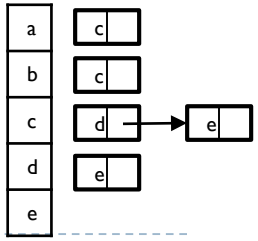




Topological Sorting – Source removal

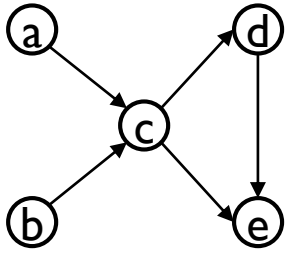
- Start from a node with no incoming edge
- No such node → exit with no solution
- While there is node remaining,
  - Identify a node with no incoming edge
  - No such node → exit with no solution
  - Else delete it and all of its outgoing edges
  - Add to the topological order

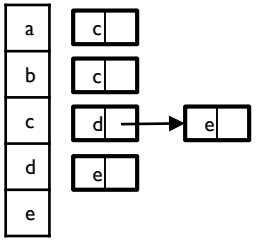




Topological Sorting – Source removal

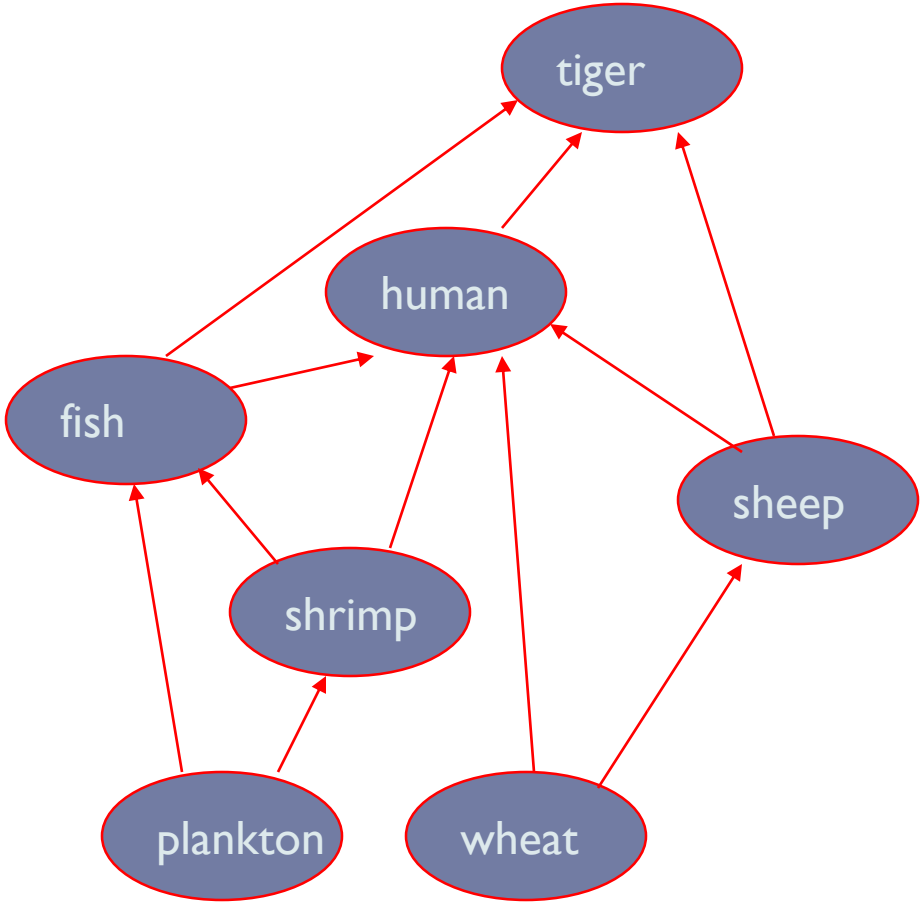
- Start from a node with no incoming edge $O(m)$
- No such node → exit with no solution
- While there is node remaining, $O(n)$
  - Identify a node with no incoming edge $O(m)$
  - No such node → exit with no solution
  - Else delete it and all of its outgoing edges $O(1)$
  - Add to the topological order
- $T(n) = m + n \times m \in O(nm)$



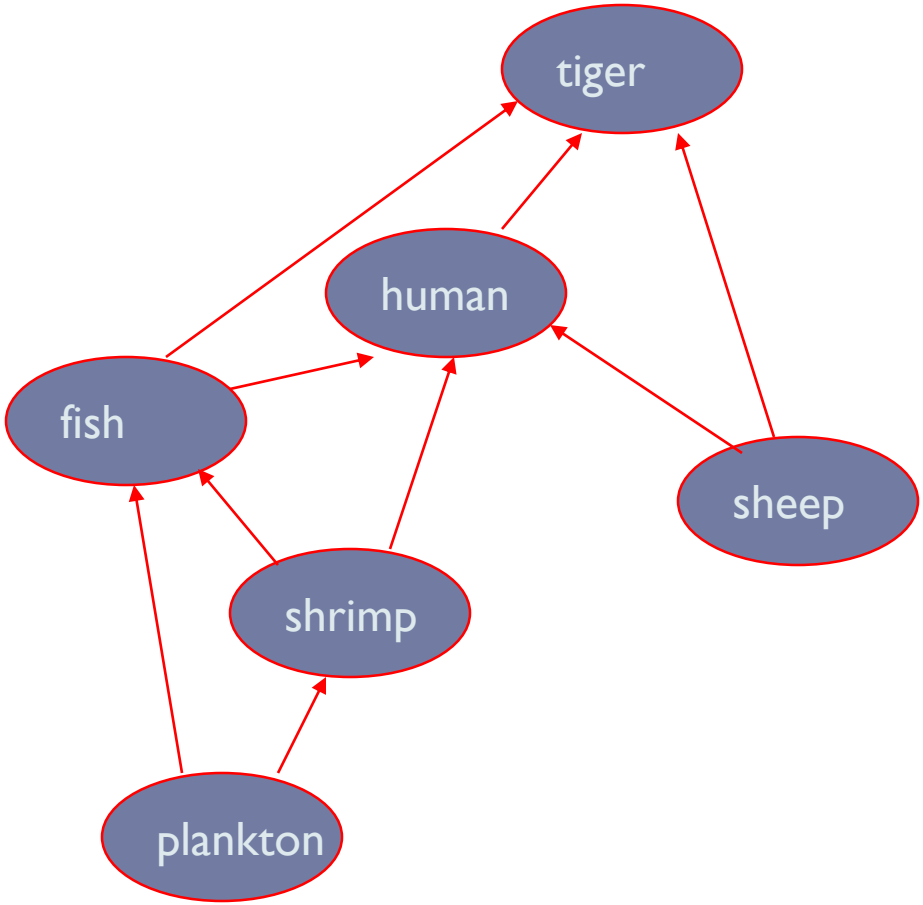


Topological Sorting Example

Order the following items in a food chain

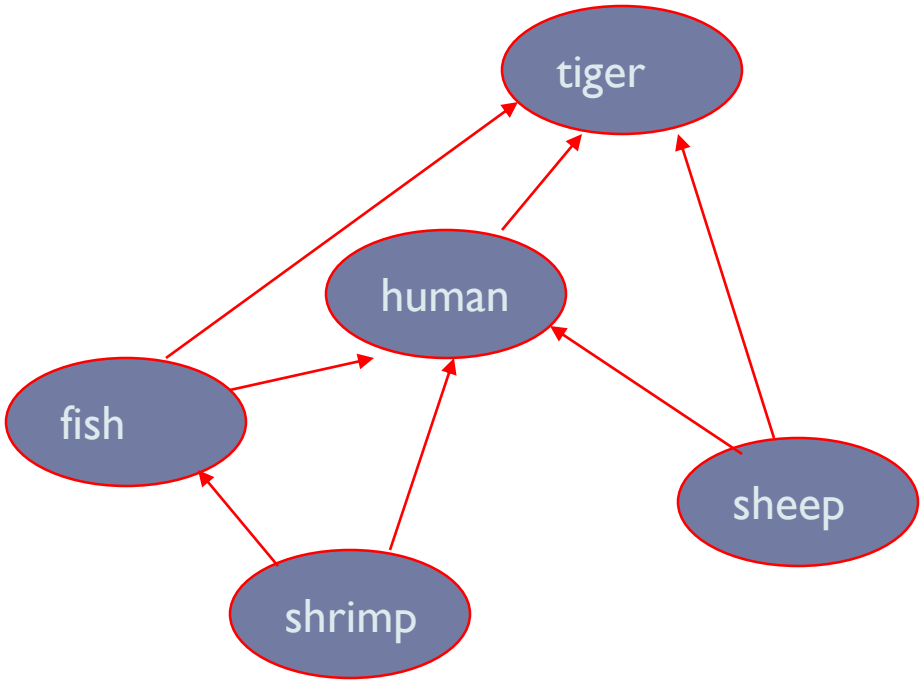


Topological Sorting Example

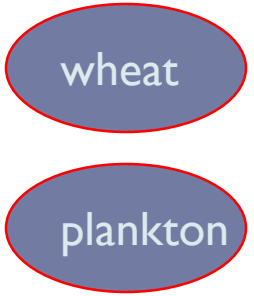


call stack: wheat

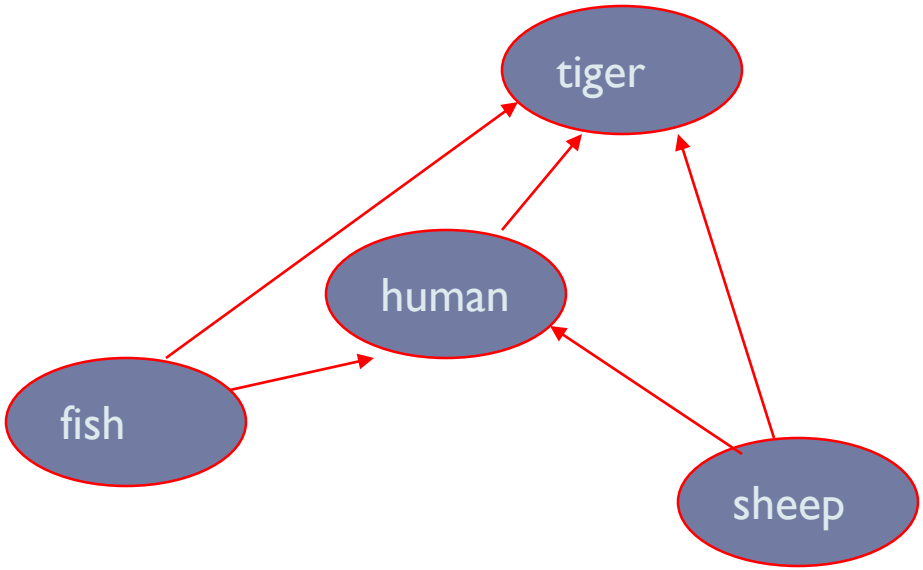
Topological Sorting Example



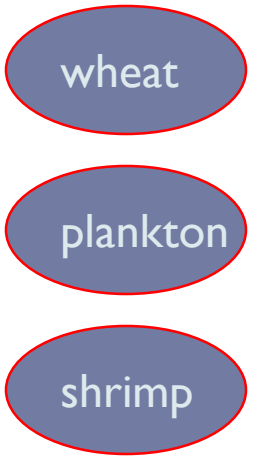
call stack:



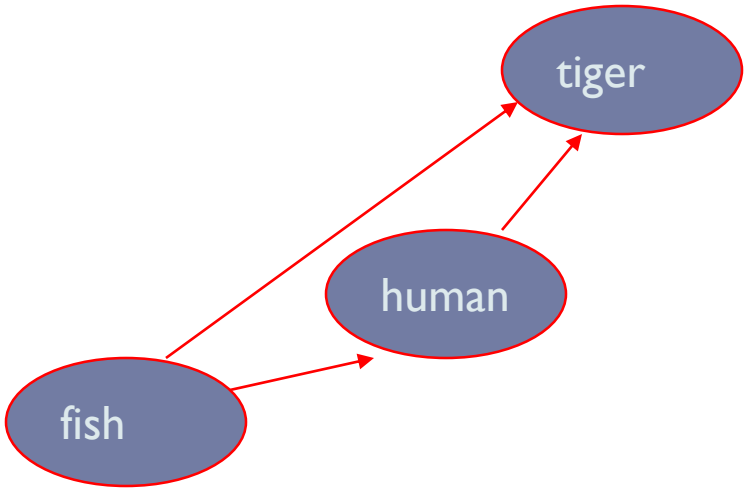
Topological Sorting Example



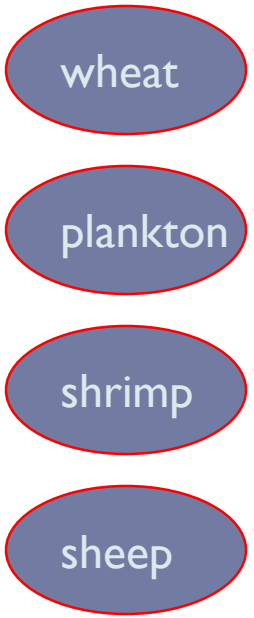
call stack:



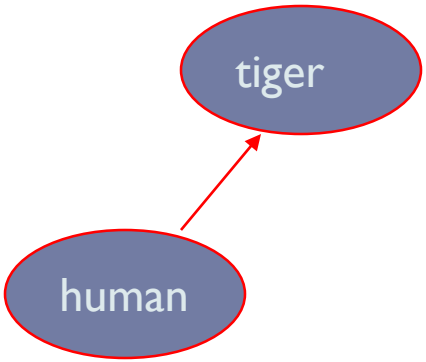
Topological Sorting Example



call stack:

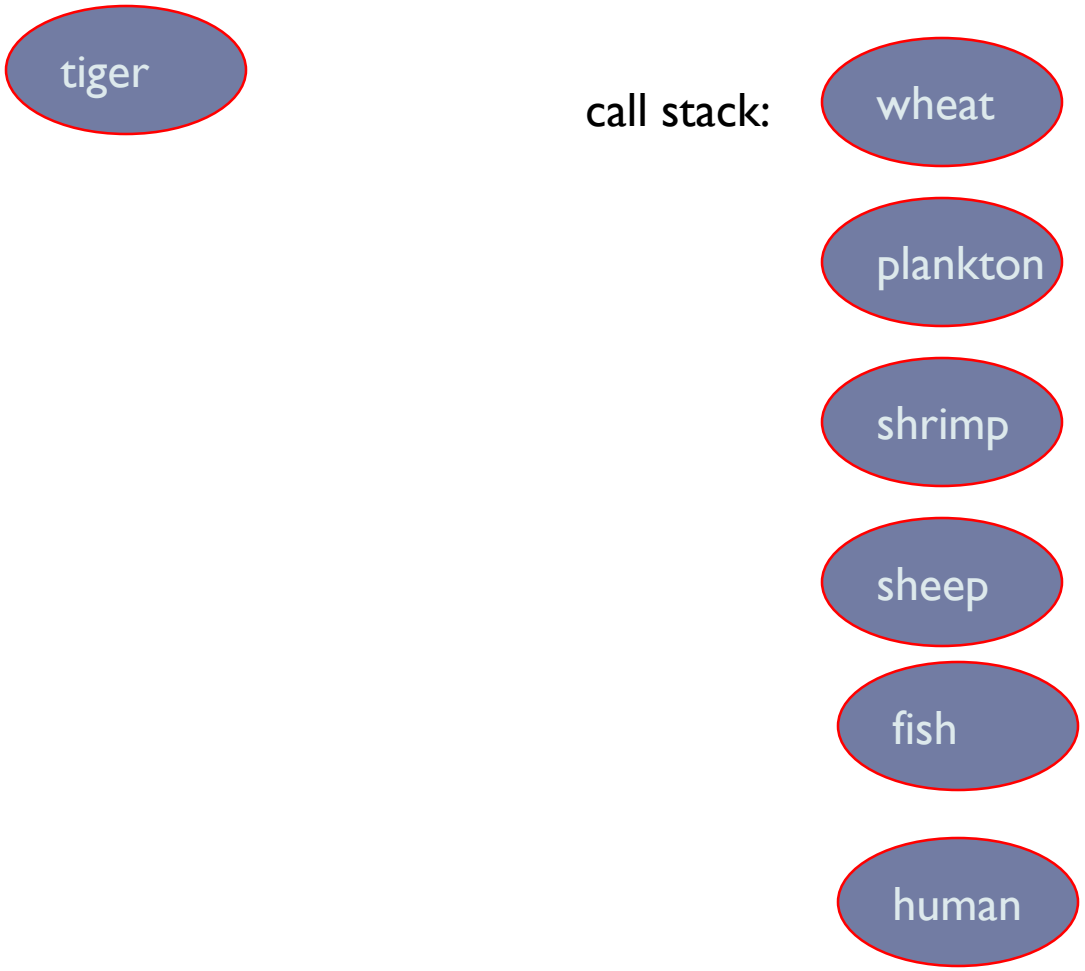


Topological Sorting Example



call stack:

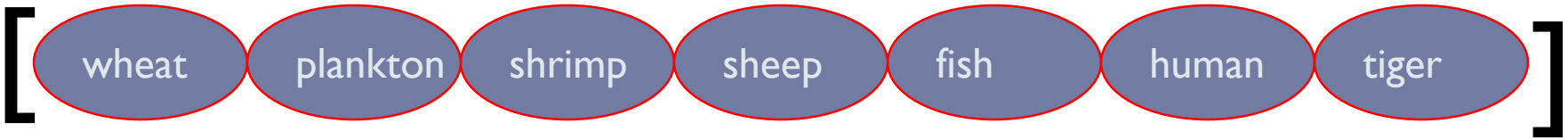
Topological Sorting Example



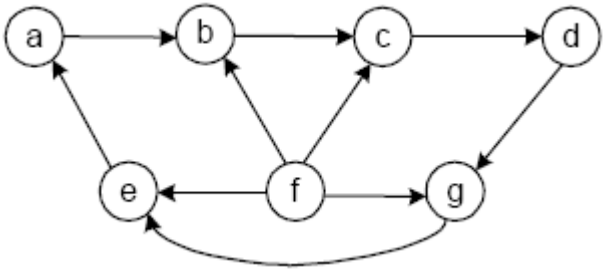
Topological Sorting Example

call stack:

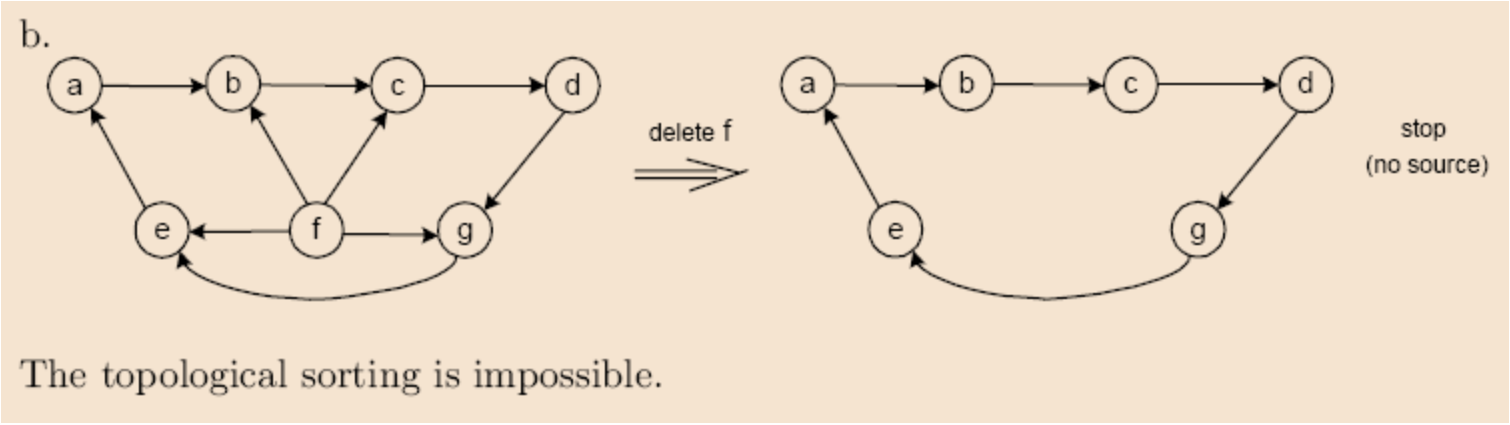
Topological Sorting Example



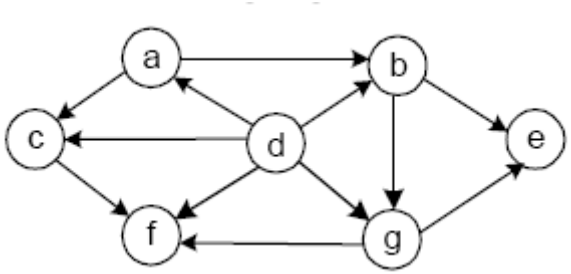
Discussion



Discussion

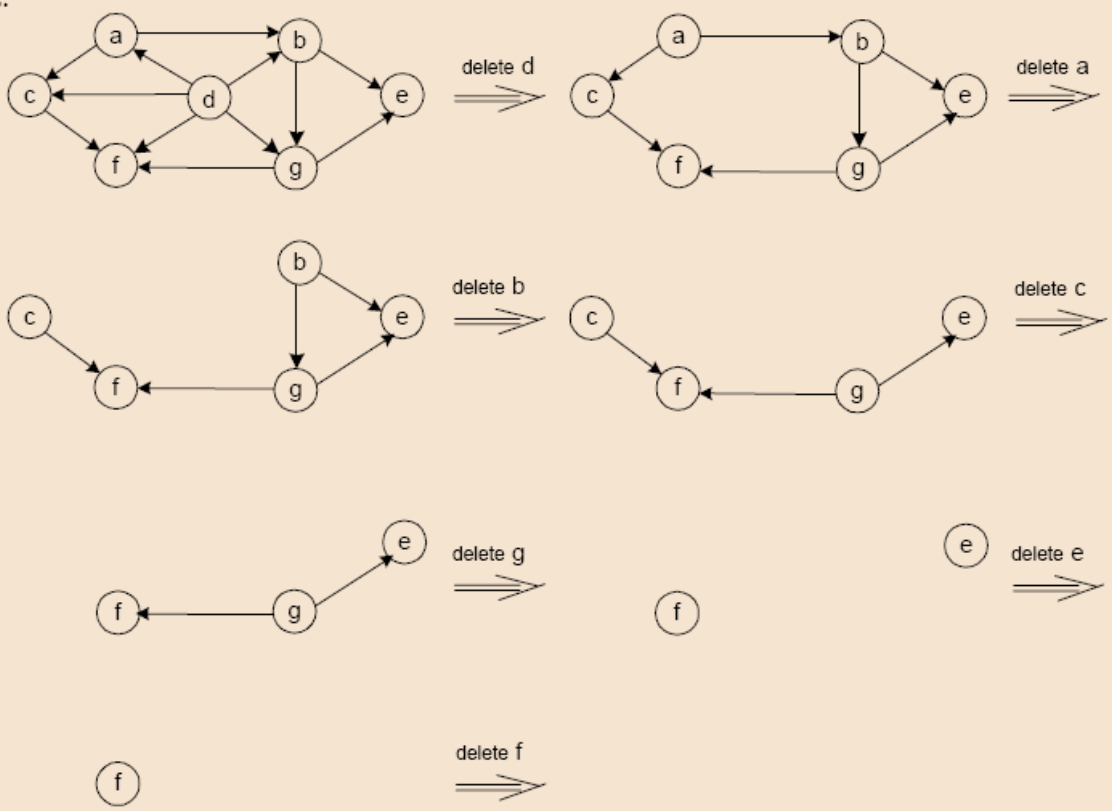

b.
a
b
c
d
e
f
g
delete f
a
b
c
d
e
g
stop
(no source)
The topological sorting is impossible.

Discussion



Discussion

5. a.



The topological ordering obtained is $d \quad a \quad b \quad c \quad g \quad e \quad f$.