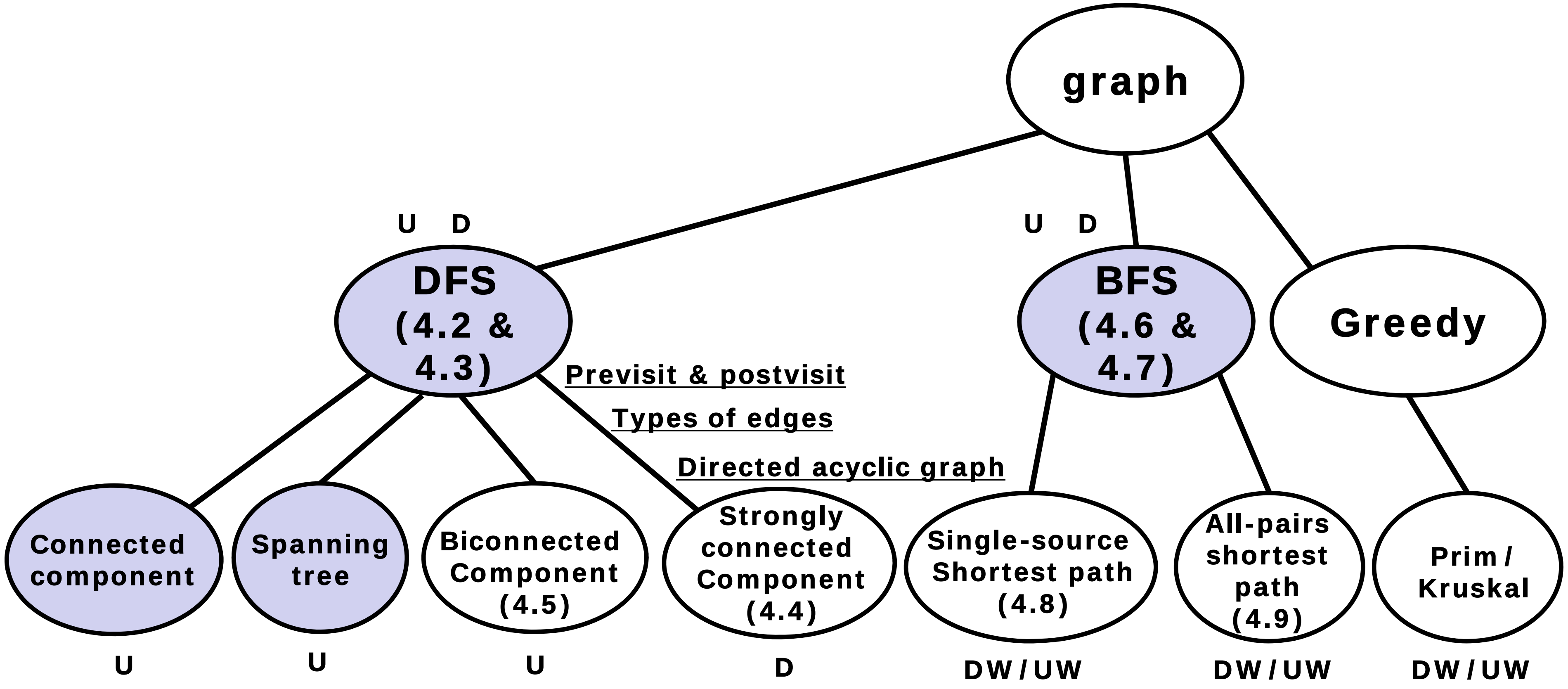
4.4 Strongly connected components

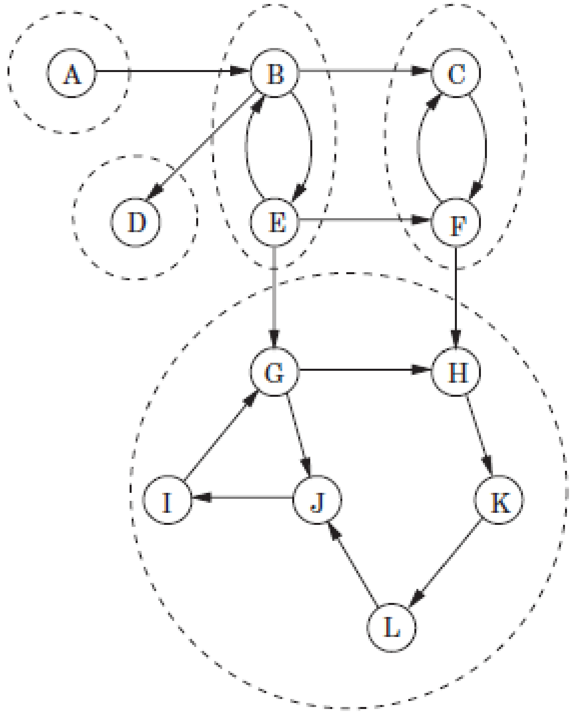
Classification of graph algorithms



4.4 Strongly connected components

(1) Connectivity for directed graph

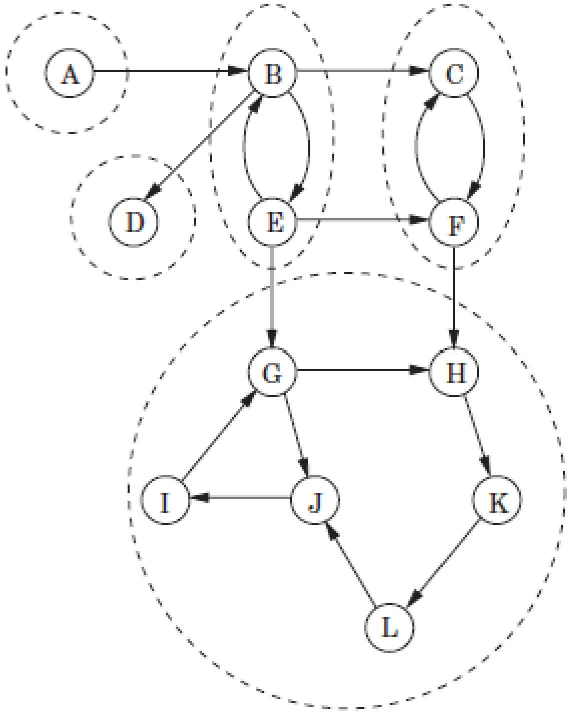
- Strongly connected
  - Two nodes u and v of a directed graph are connected if there is a path from u to v and vice versa
  - Example)
    - » B & E are connected
    - » G & H are connected
    - » J & L are connected
    - » E & G are not connected



4.4 Strongly connected components

(1) Connectivity for directed graph

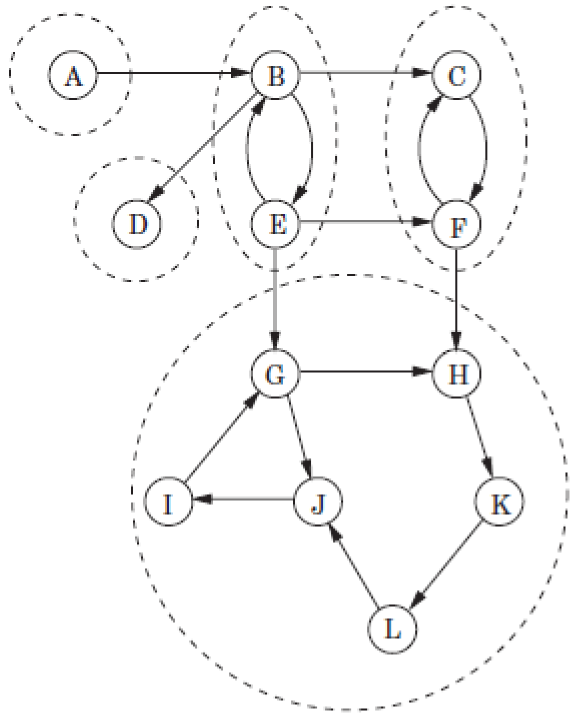
- Strongly connected component (SCC)
  - Partitioning of V into disjoint sets according to the definition of “strongly connected”
  - Example)
    - » B & E are SCC
    - » A is SCC
    - » G, H, I, J, K & L are SCC

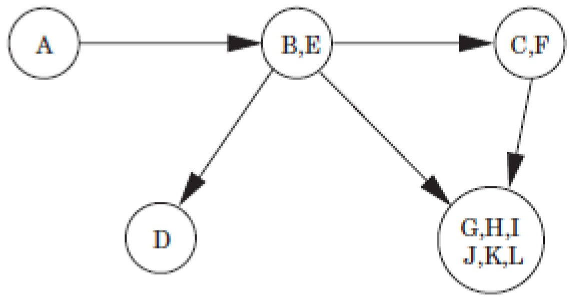


4.4 Strongly connected components

(1) Connectivity for directed graph

- Property
  - Every directed graph is a dag of its strongly connected components

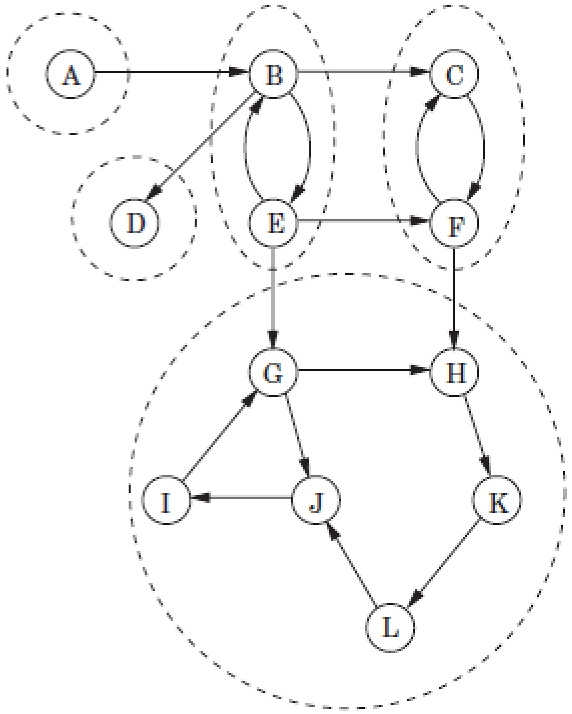


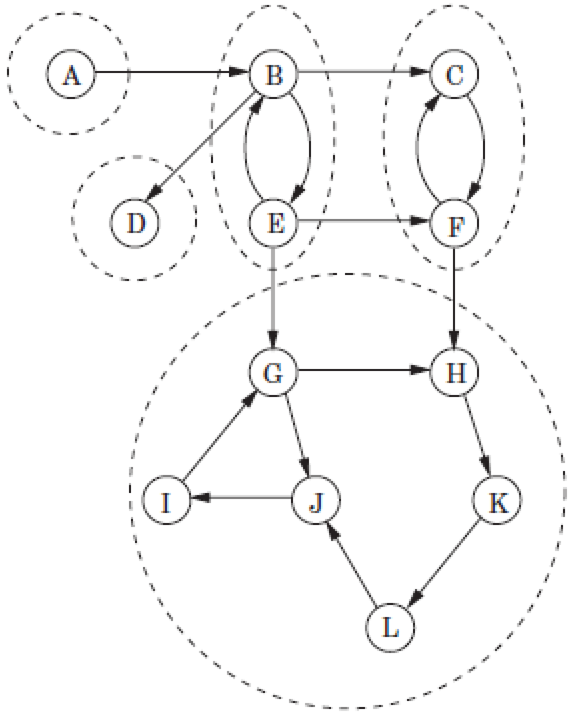


4.4 Strongly connected components

(2) Algorithm

- Property1
  - If the dfs ( ) subroutine is started at a node u, then it will terminate precisely when all nodes reachable from u have been visited

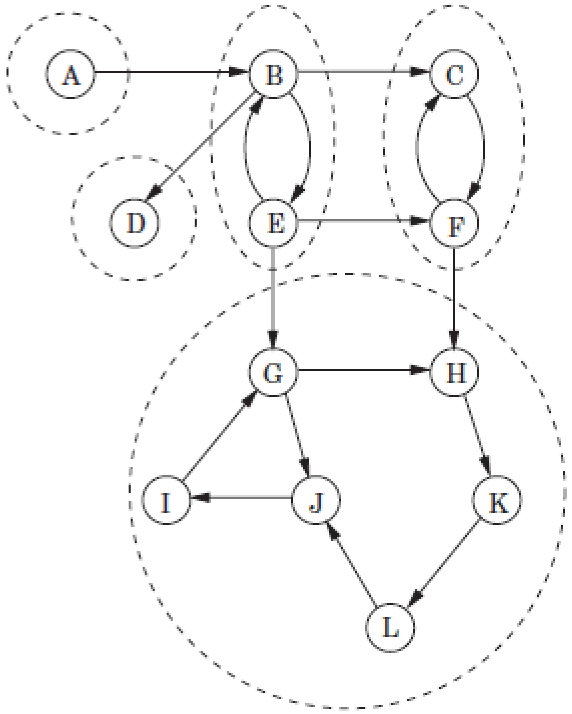


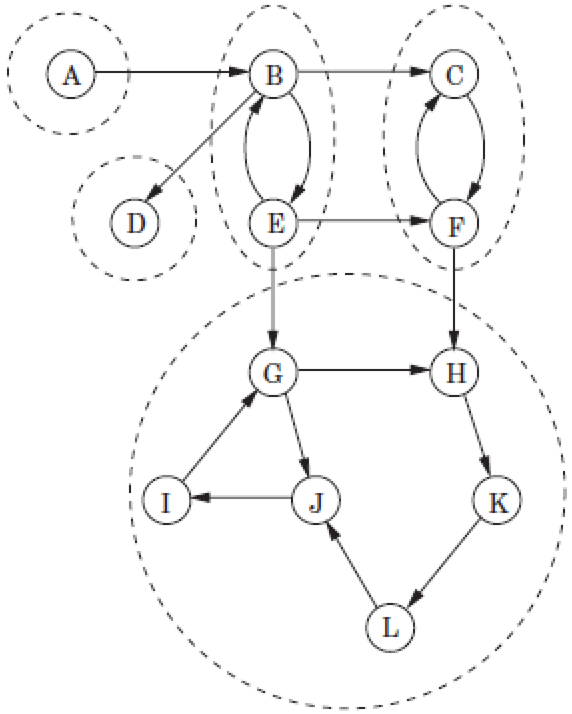


4.4 Strongly connected components

(2) Algorithm

- Property2
  - The node that receives the highest post number in a depth-first search must lie in a source strongly connected components

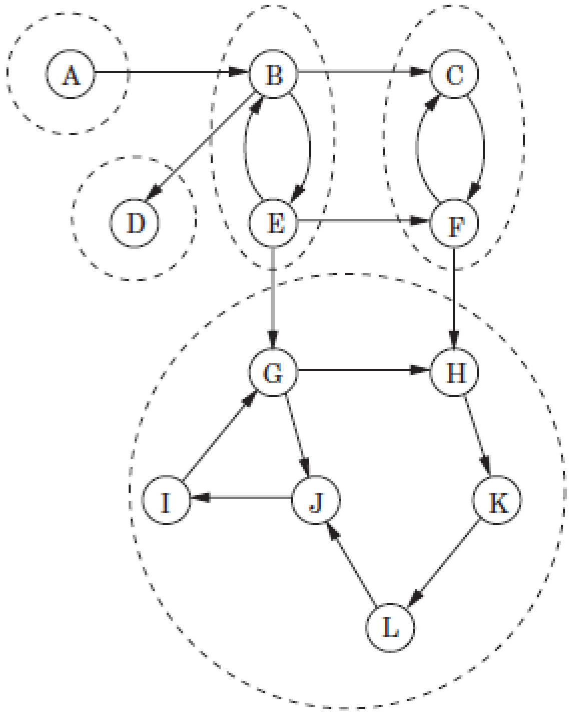


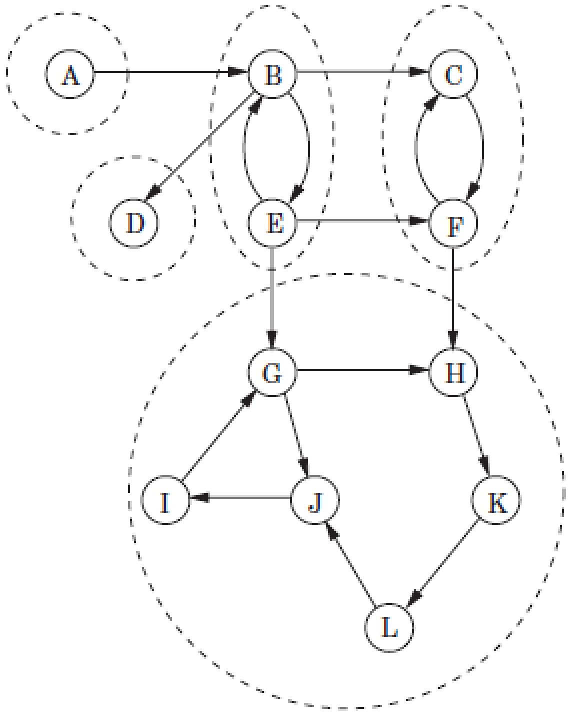


4.4 Strongly connected components

(2) Algorithm

- Property3
  - If C and C' are strongly connected components, and there is an edge from a node in C to a node in C', then the highest post number in C is bigger than the highest post number in C'





4.4 Strongly connected components

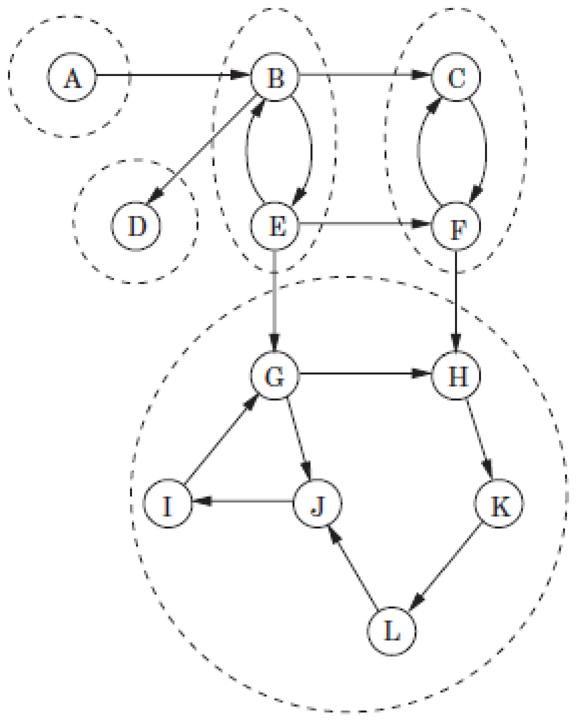
(2) Algorithm

- Strategy
  - Find a sink strongly connected component and remove it
  - Repeat this until we have only one strongly connected component
- Problem
  - How can we find a sink strongly connected component?
    - Motivation
      - » Use Property2
      - » Define G^R from G = (V, E)
      - » G^R has same V, but reverse E
      - » Sink component in G = Source component in G^R

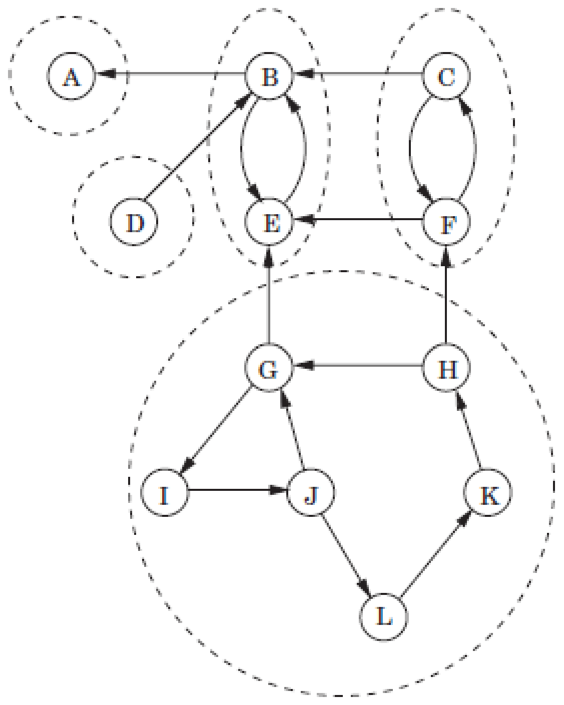
4.4 Strongly connected components

– Example

G:



G^R:



4.4 Strongly connected components

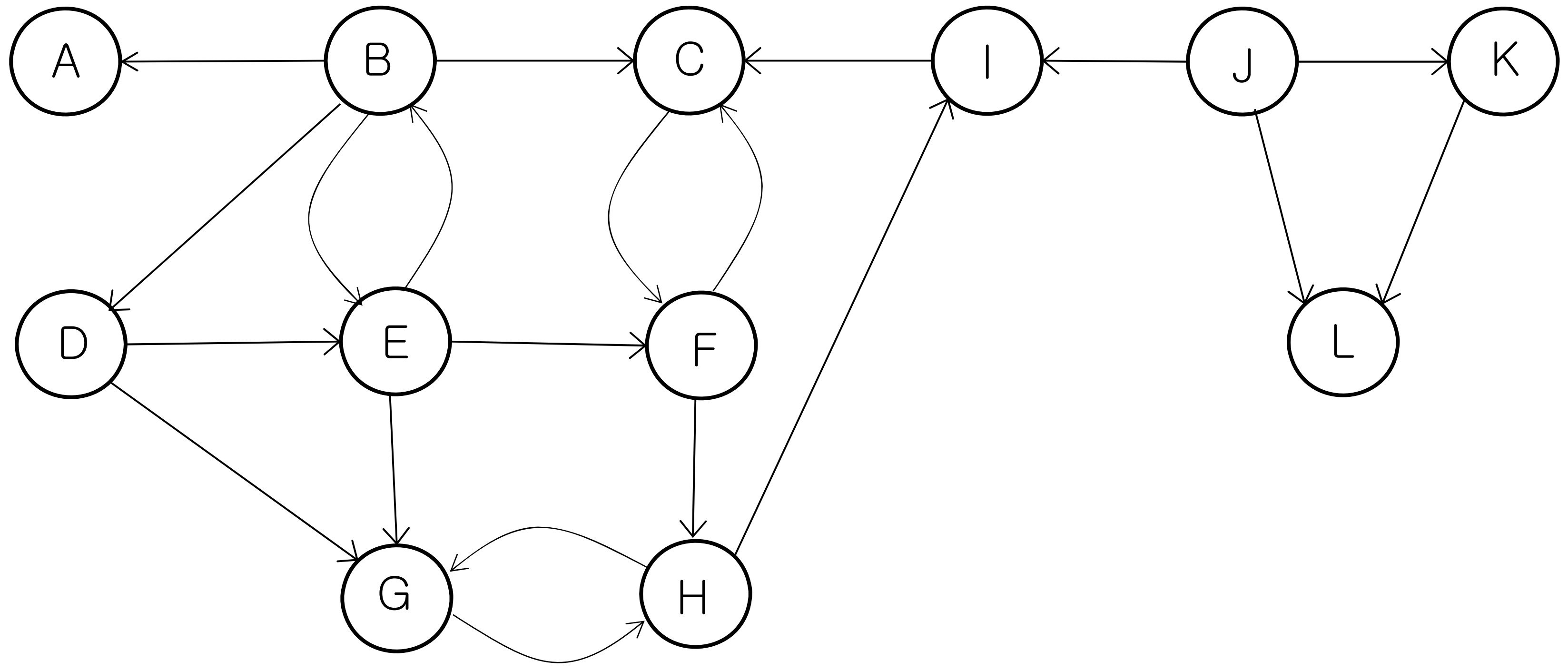
(2) Algorithm

- Steps
  - Compute G^R from G
  - Run depth-first search on G^R
  - Run the undirected connected components algorithm
    - Process the vertices in decreasing order of their post numbers from the previous step

4.4 Strongly connected components

(2) Algorithm

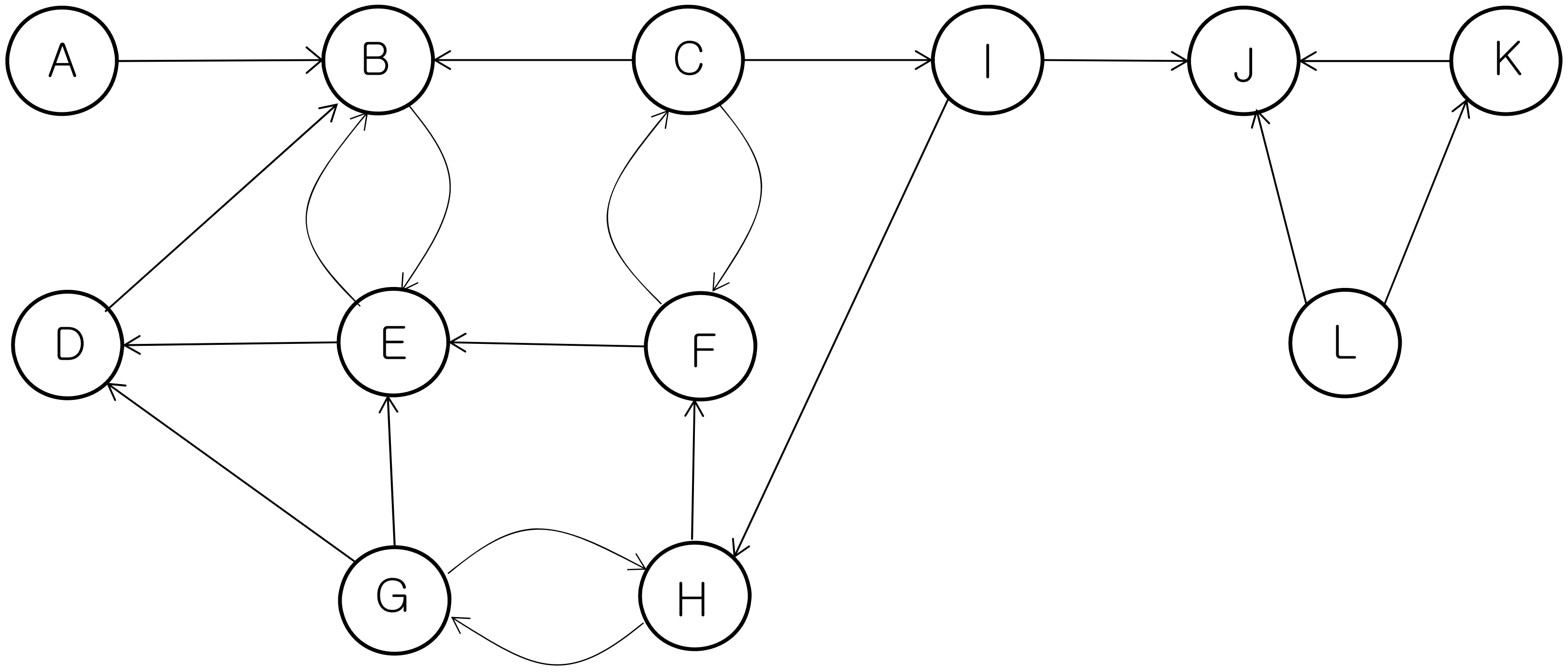
– Test (알파벳 순으로)



4.4 Strongly connected components

(2) Algorithm

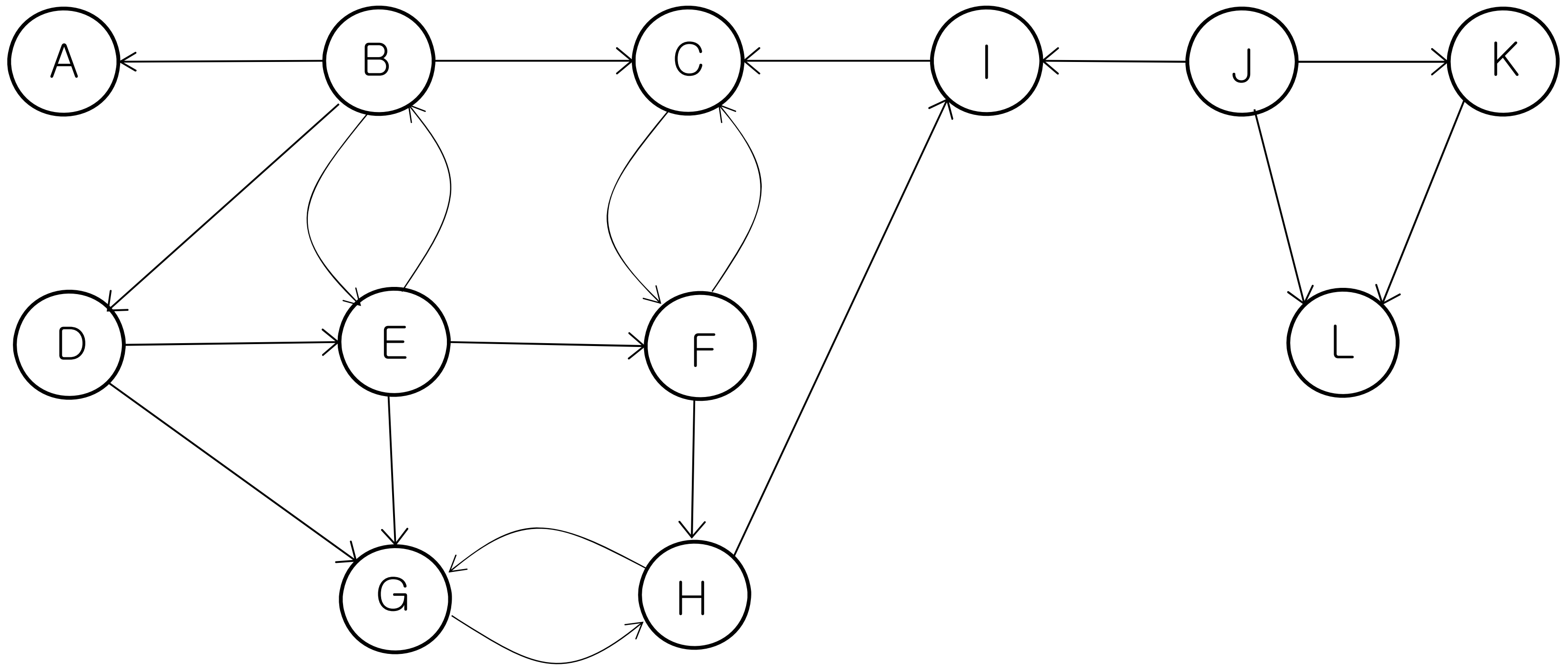
– Test (알파벳 순으로)



4.4 Strongly connected components

(2) Algorithm

– Test (알파벳 역순으로)



All about graph

Type	Purpose	Operations	Performance
DFS	Traverse all vertices	Visiting all vertices & visiting all edges	O(n) + O(m)
SCC	Finding SCC	DFS on G^R and G	O(DFS)
BCC			
BFS			
Dijkstra			
Floyd			
Kruskal (Greedy)			
Prim (Greedy)			
MultiStage (Dynamic)			

4.4 Strongly connected components

다음은 strongly connected component를 찾는 알고리즘에 대한 설명이다. 올바른 것을 모두 고르시오.

(a) strongly connected component들은 dag를 형성한다.

(b) strongly connected component에서 가장 높은 post num을 갖는 vertex를 가진 component는 항상 sink이다.

(c) strongly connected component를 계산하는 연산 시간은 O(n + m)이다 (n은 vertex의 수, m은 edge의 수)

(d) strongly connected component의 결과는 시작하는 vertex에 따라서 다르다.