Homework Assignments #5

15146312 박강민

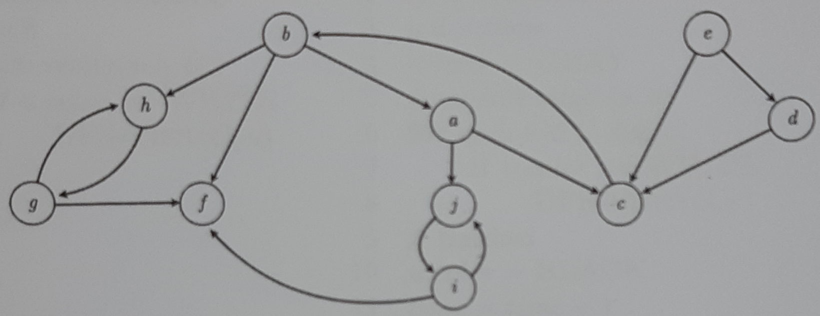
Due: 2020/06/12 24:00

Assessment policy:

- Give full points when correct, 1/n for solving each n subproblems. 0 for totally wrong or none, -1 for each errors.
- There may be partial points for proofs if the direction is correct.

1. Strongly Connected Components

We want to compute strongly connected components of the graph G below.



Having the pseudocode for computing STRONGLY-CONNECTED-COMPONENTS, we will

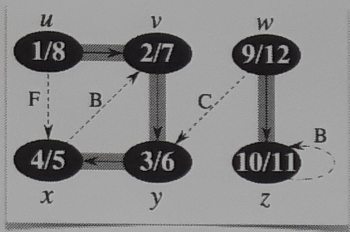
```
STRONGLY-CONNECTED-COMPONENTS(G)
1  call DFS(G) to compute finishing times u.f for each vertex u
2  compute G^T
3  call DFS(G^T), but in the main loop of DFS, consider the vertices
       in order of decreasing u.f (as computed in line 1)
4  output the vertices of each tree in the depth-first forest formed in line 3 as a
       separate strongly connected component
```

(a) Represent graph G in an adjacency-list representation. Assume that each adjacency list is ordered alphabetically. *(2 pts)*

a → c → j
b → a → f → h
c → b
d → c
e → c → d
f
g → f → h
h → g
i → f → j
j → i

(b) Apply DFS procedure for the graph, printing out the discovered / finished time for each vertex. Also classify each edges, into one of tree / back / forward / cross edge. Assume that vertices are iterated in alphabetical order. *(5 pts)*

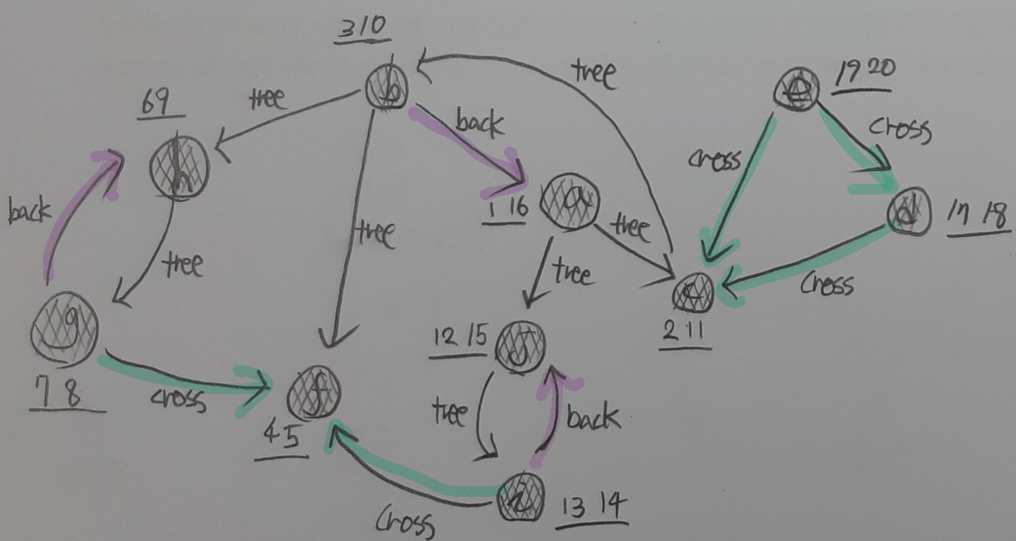
The figure used in the slide:



The DFS procedure used in the slide is as follows:

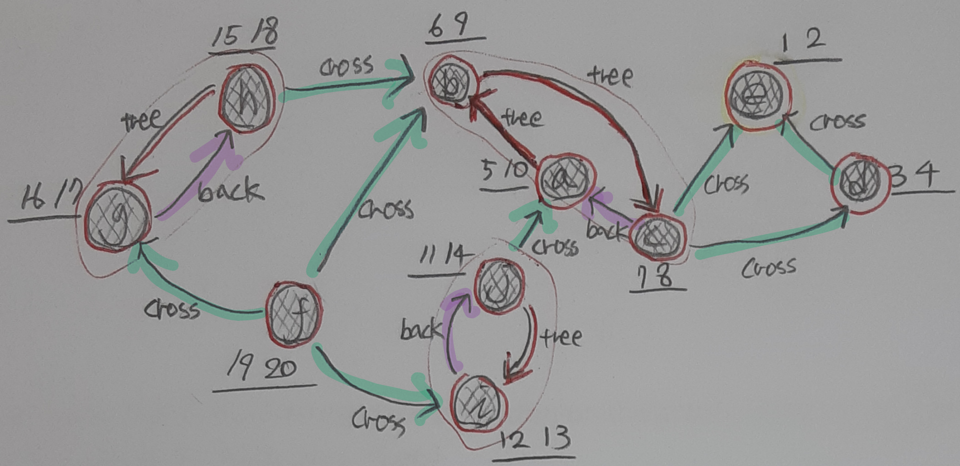
```
DFS(G)
for each vertex u ∈ G.V
        u.color = WHITE
time = 0
for each vertex u ∈ G.V
        if u.color == WHITE
                DFS-VISIT(G, u)
```

```
DFS-VISIT(G, u)
// white vertex u has just been discovered
time = time + 1
u.d = time
u.color = GRAY
// explore edge (u, v)
for each v ∈ G.Adj[u]
        if v.color == WHITE
                DFS-VISIT(G, v)
// finished
u.color = BLACK
time = time + 1
u.f = time
```



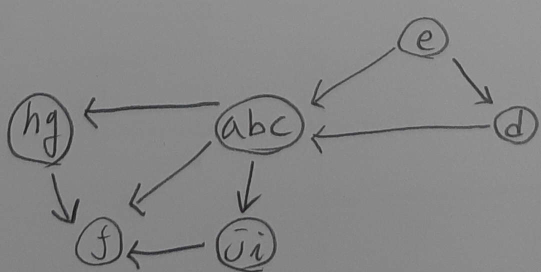
(c) The transpose of a directed graph $G = (V, E)$ is the graph $G^T = (V, E^T)$, where $E^\mathrm{T} = \{(v, u) \in V \times V : (u, v) \in E\}$.

Compute $\mathrm{DFS}(G^T)$, and represent the answer in the figure as in (b). Consider the in order of decreasing $u.f$. *(3 pts)*



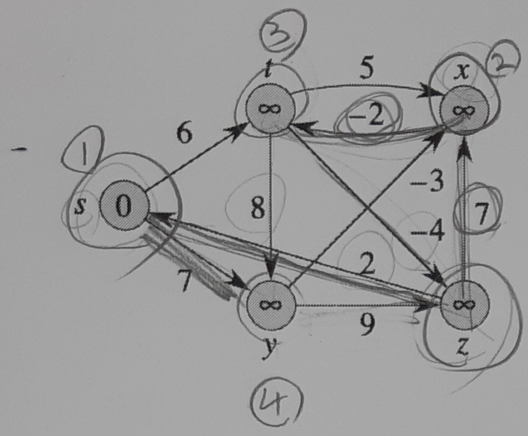
(d) Using the results above, show the strongly connected components of G, by printing out the vertices of each component in each line. Also draw a component graph G^{SCC}, which contracts all edges whose incident vertices are within the same strongly connected component of G. *(3 pts)*

(e)
(d)
(a b c)
(j i)
(h g)
(f)



2. Shortest paths

Consider the following graph:



(a) Apply Dijkstra's shortest-path algorithm on the graph, using vertex z as the source. Show the d and π values for each pass. *(5 pts)*

-1

```
DIJKSTRA(G, w, s)
INITIALIZE-SINGLE-SOURCE(G, s)
S = ∅
Q = G.V
while Q ≠ ∅
    u = EXTRACT-MIN(Q)
    S = S ∪ {u}
    for each vertex v ∈ G.Adj[u]
        RELAX(u, v, w)
```

d:

		s	t	x	y	z
	0	∞	∞	∞	∞	∞
z	1	2	∞	7	∞	0
s	2	2	8	7	9	
x	3		5	7	9	
t	4		5			
y	5			6	9	

π:

	s	t	x	y	z
0	NIL	NIL	NIL	NIL	NIL
1	z	NIL	z	NIL	
2	z	s	z	s	
⋮	z	x	z	s	
	z	x	z	s	
	z	x	~~z~~ y	s	

(b) According to (a), what is the shortest path from vertex z to x? Prove or disprove that the result is correct. *(5 pts)*

z → x

-1

this result is not correct.

because of the negative weight, although the shortes path from z to x is already set as z → x with distance 7, the path z → s → y → x has the distance 6, Dijkstra's algorithm is not appropriate the graph has negative weight.