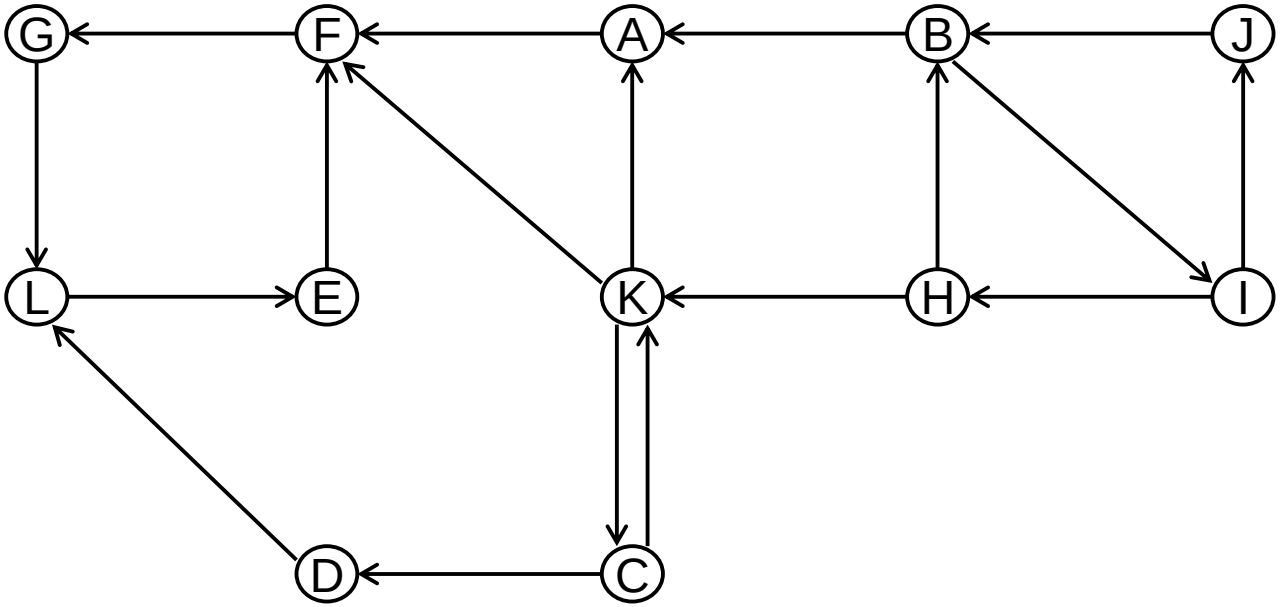
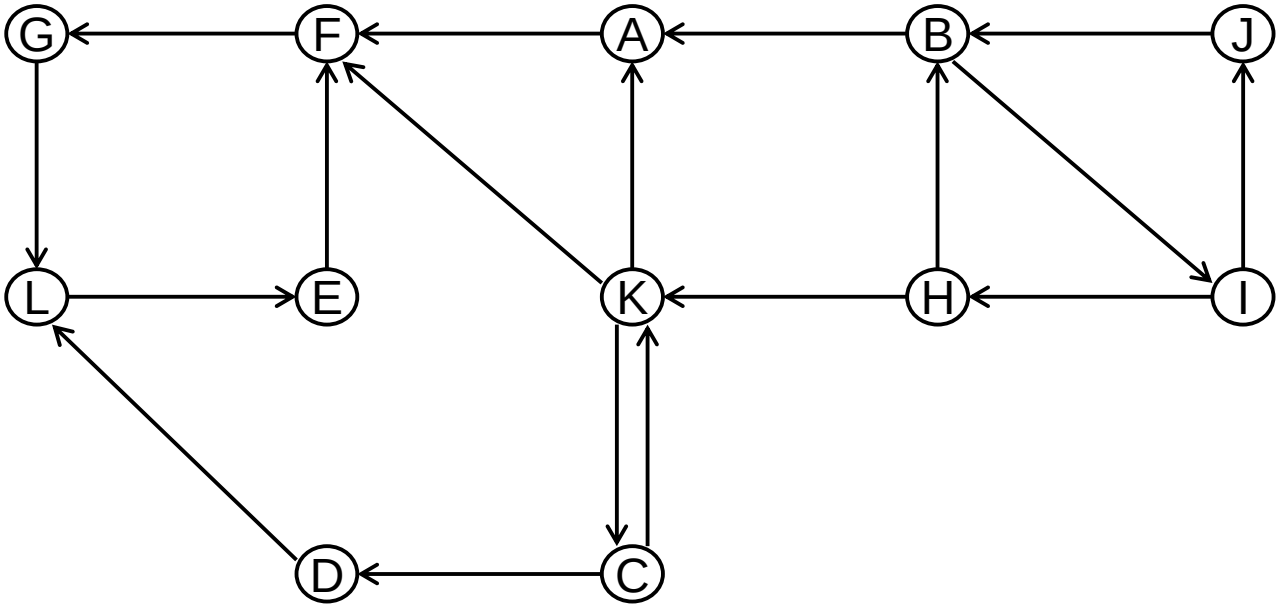
G:

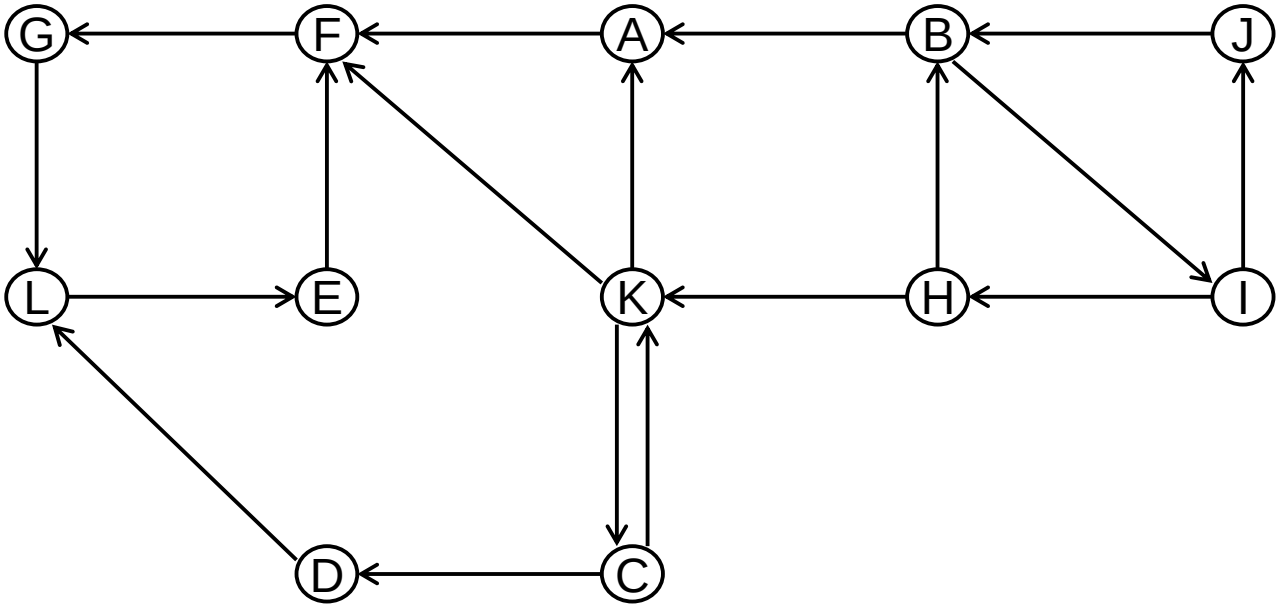


G:

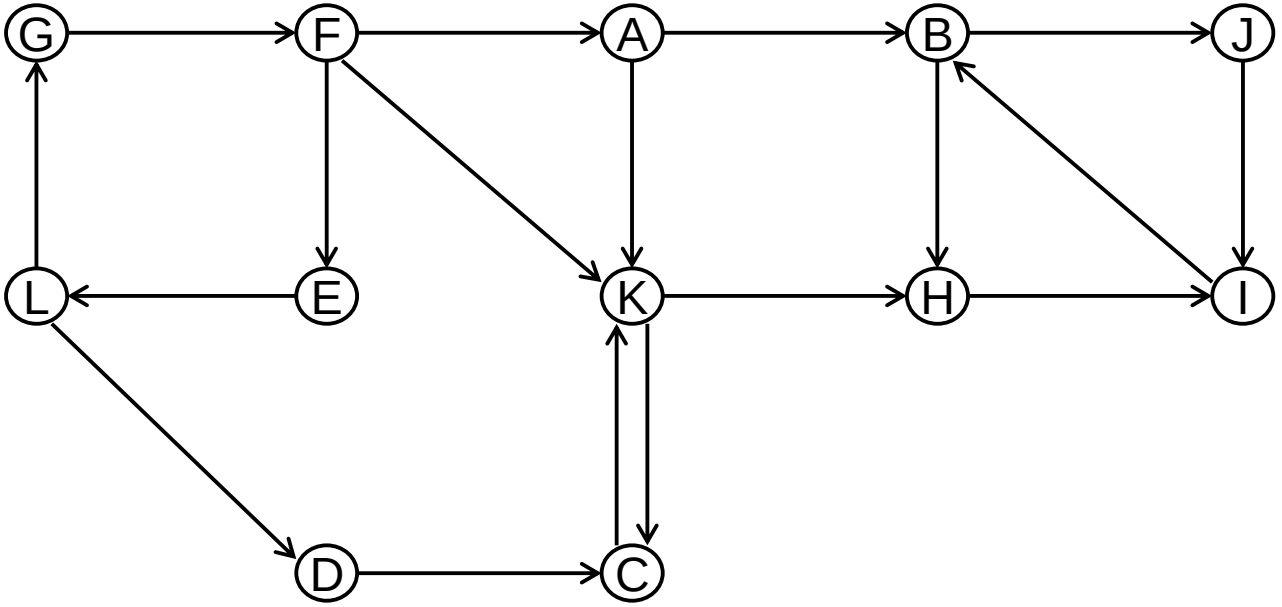


Compute the Reverse of the Graph:

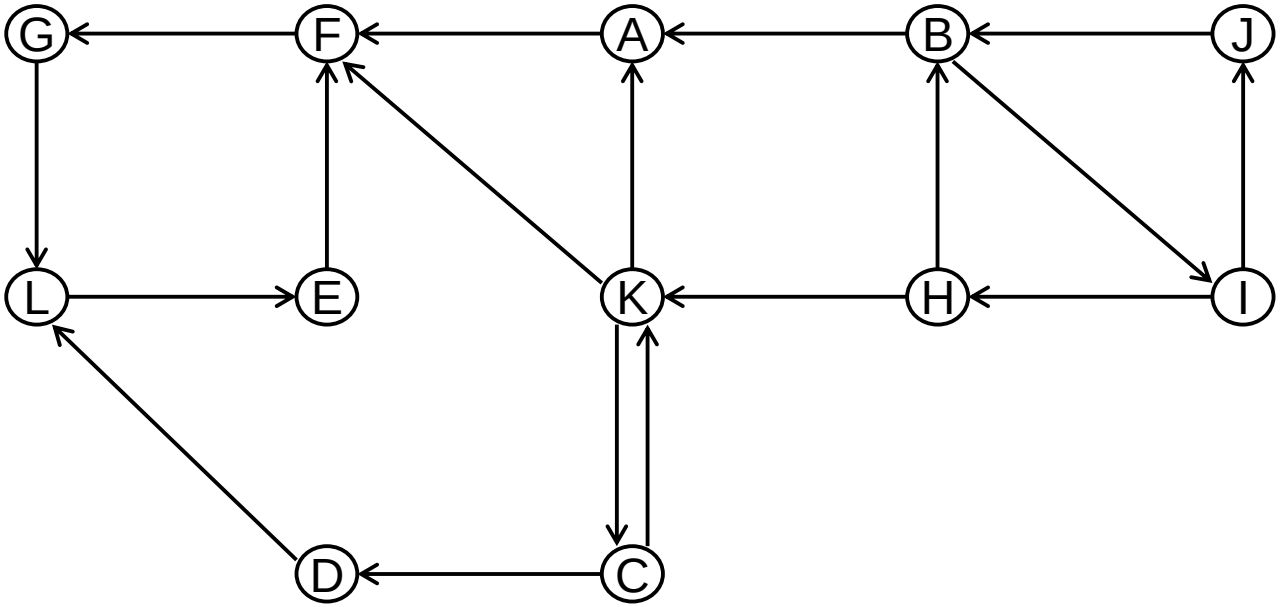
G:



G^R:

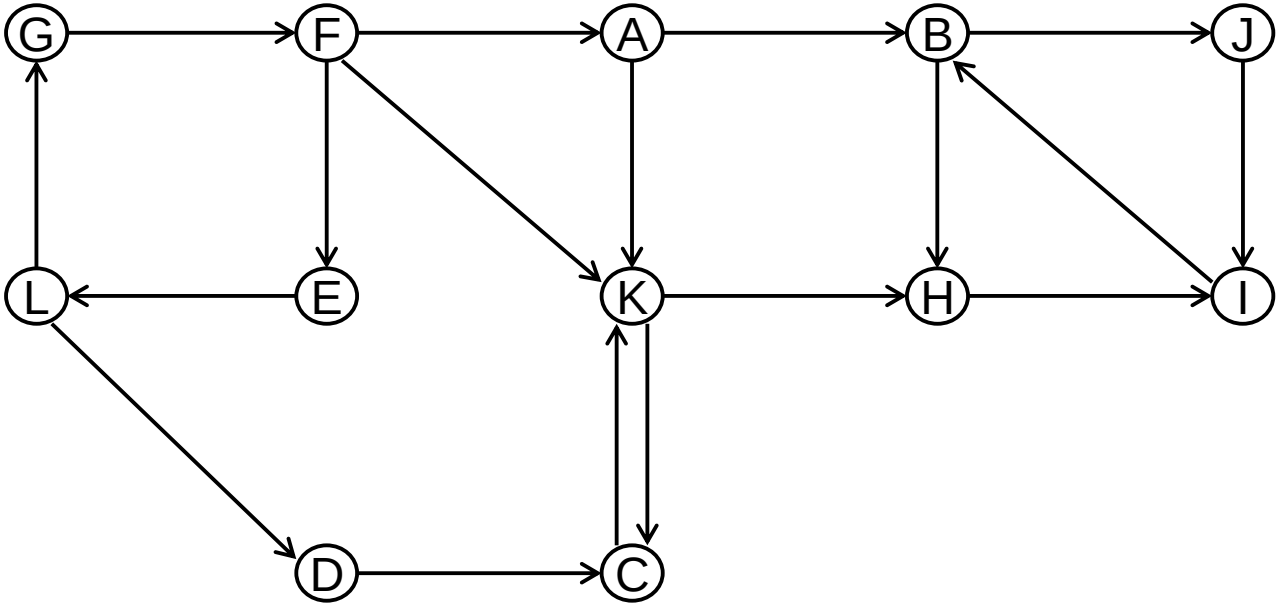


G:

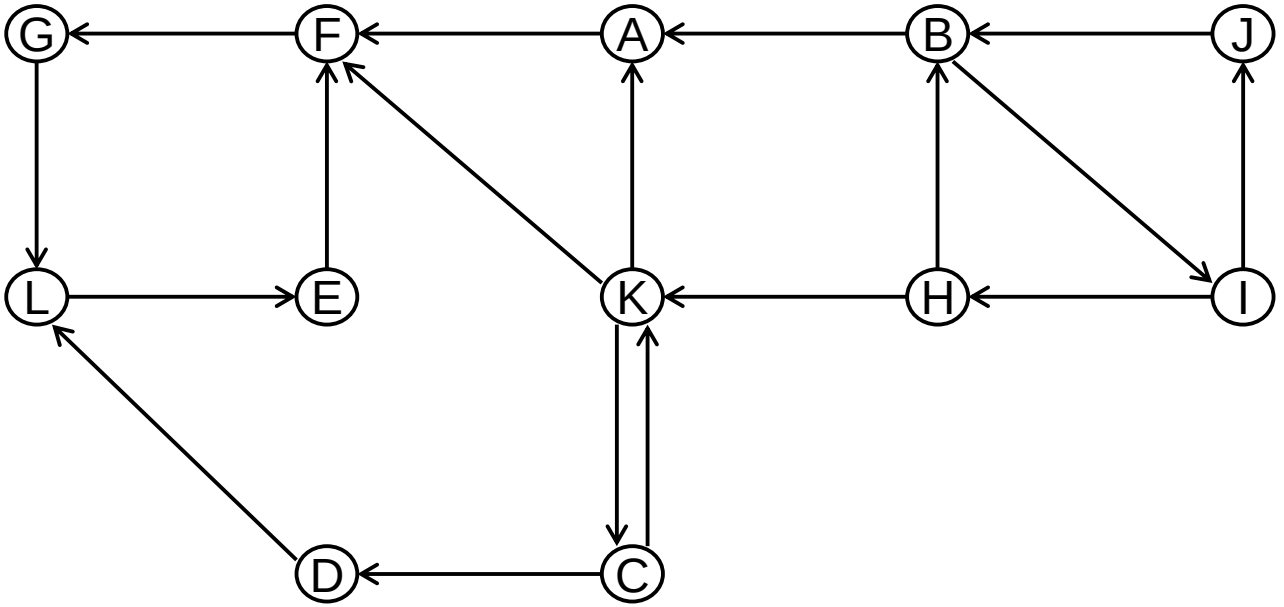


Do DFS on the reverse of G

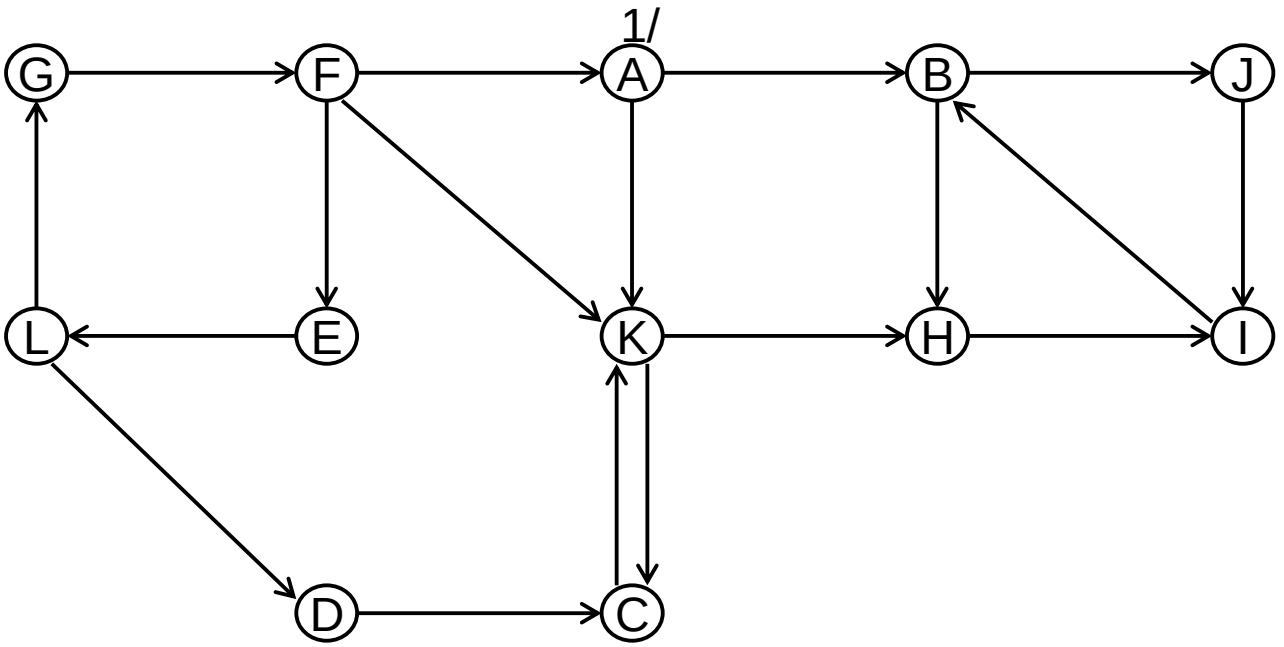
G^R:



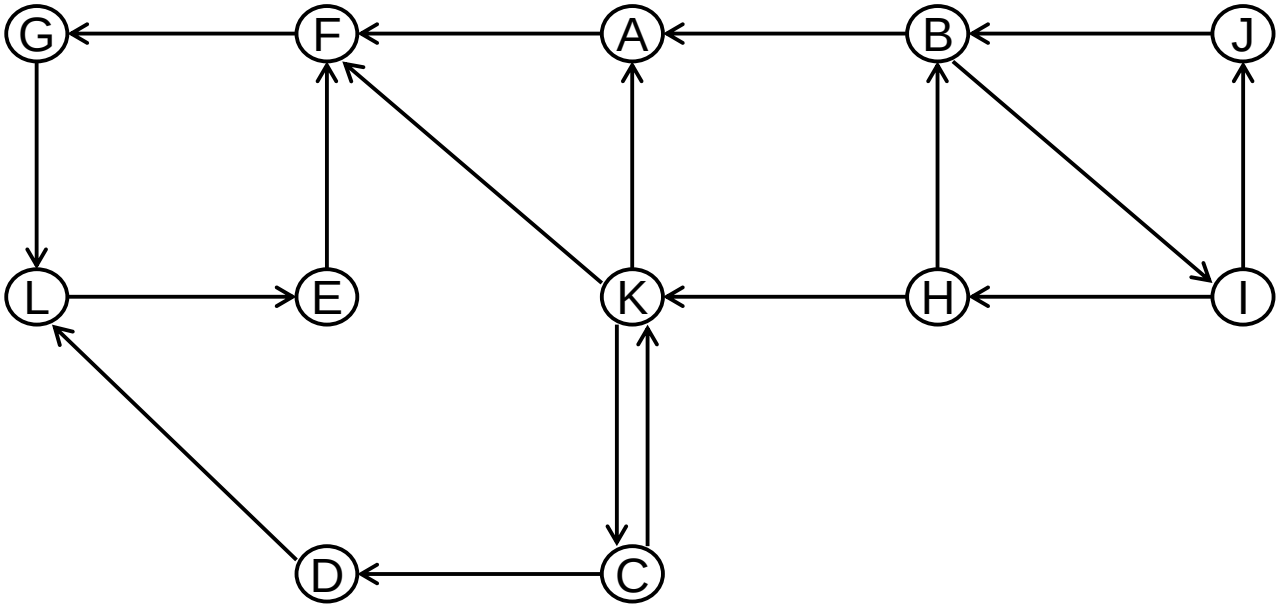
G:



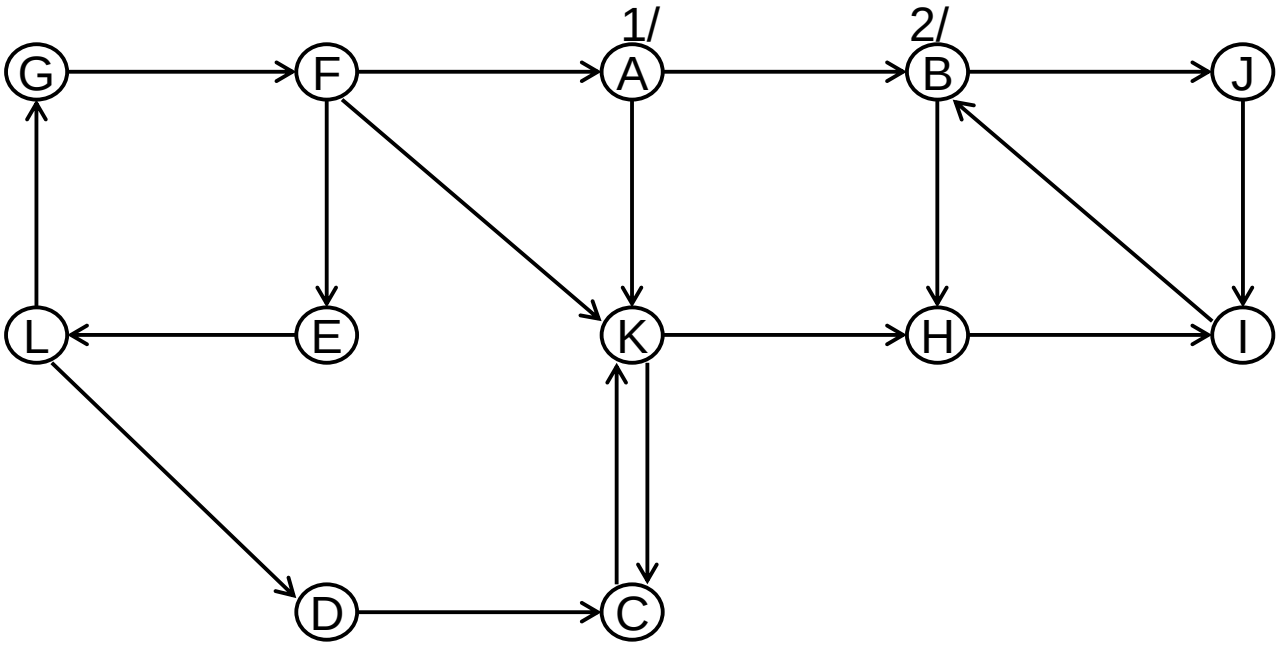
G^R:



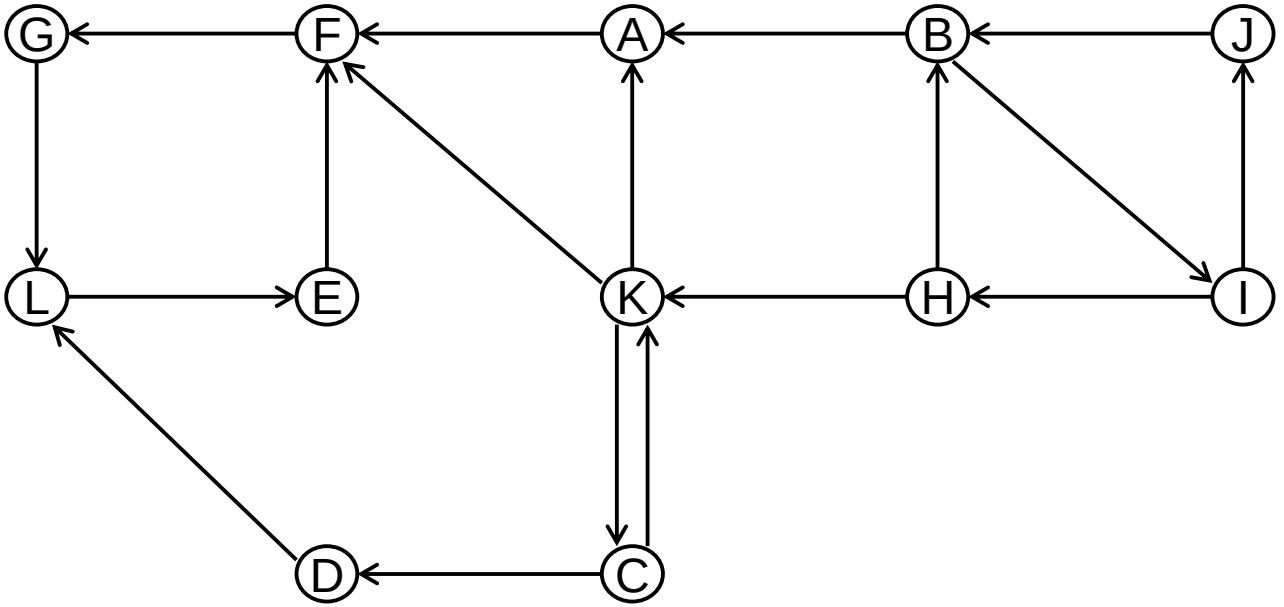
G:



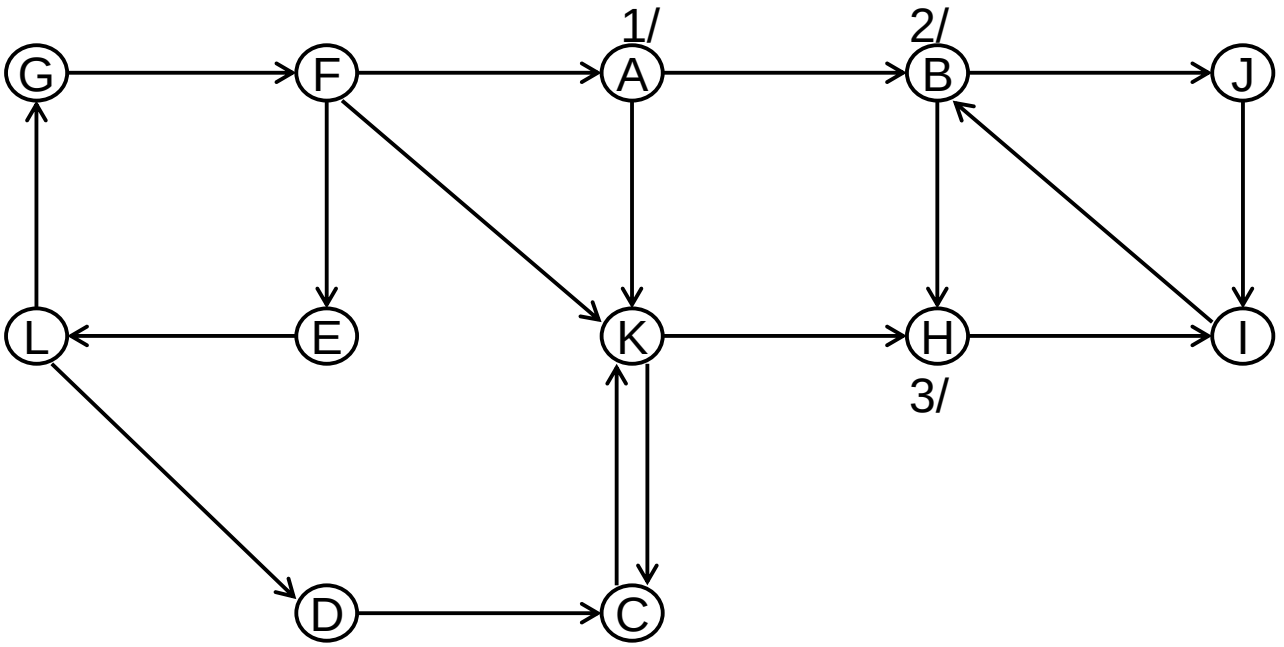
G^R:



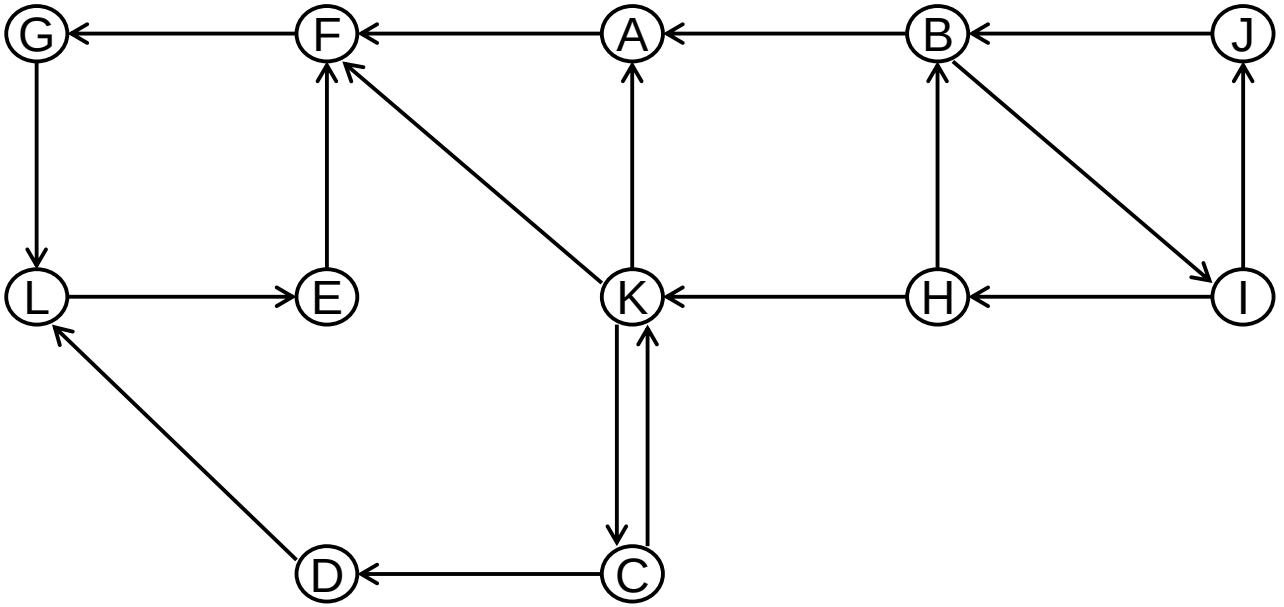
G:



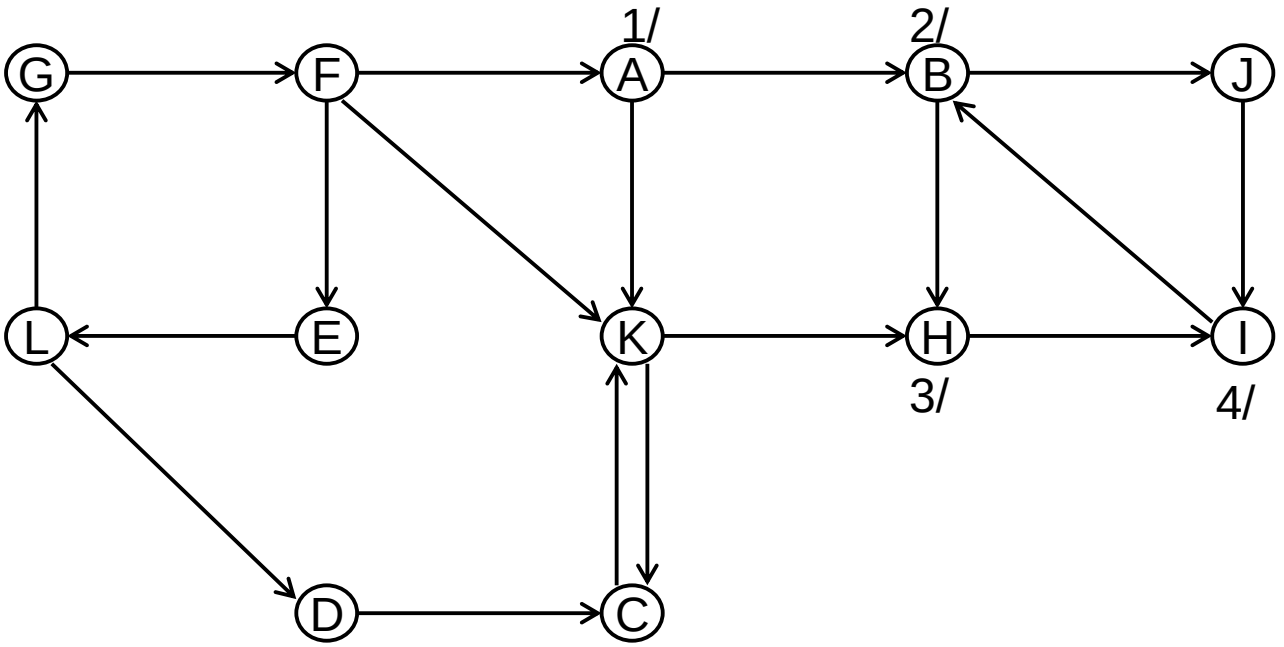
G^R:



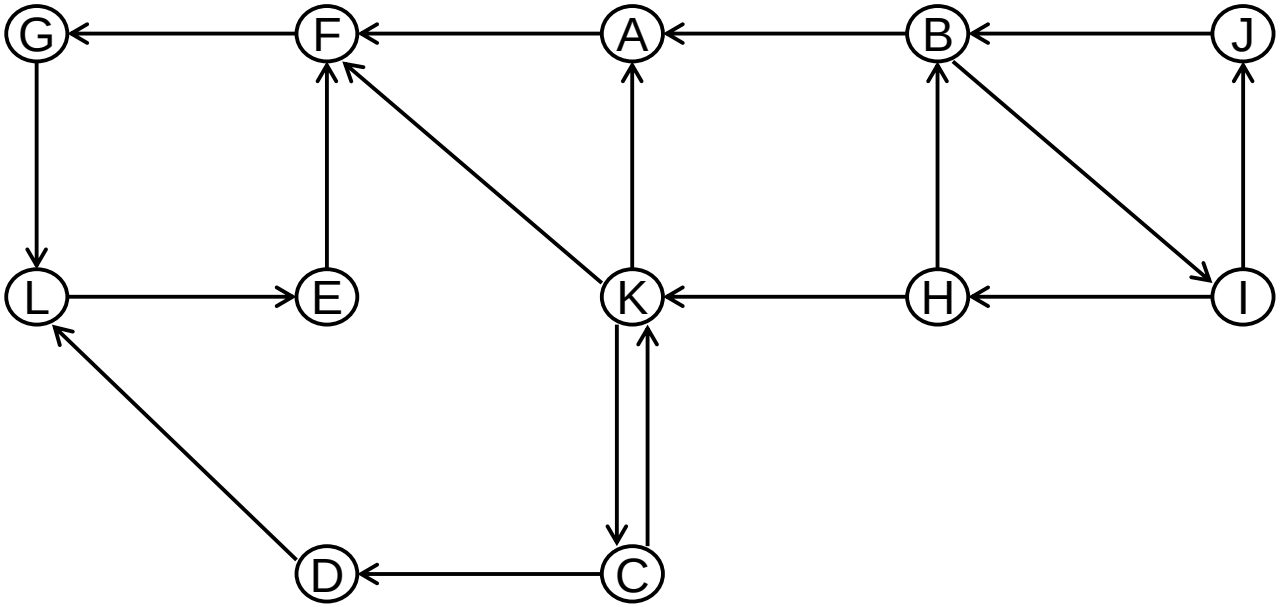
G:



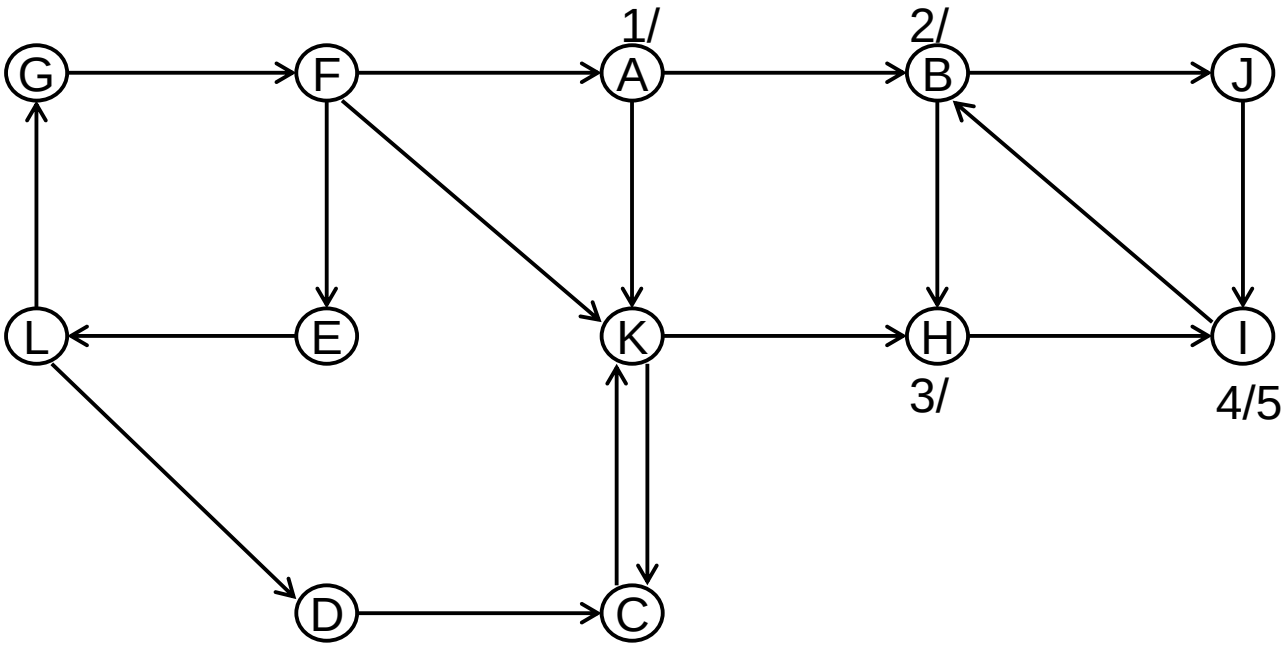
G^R:



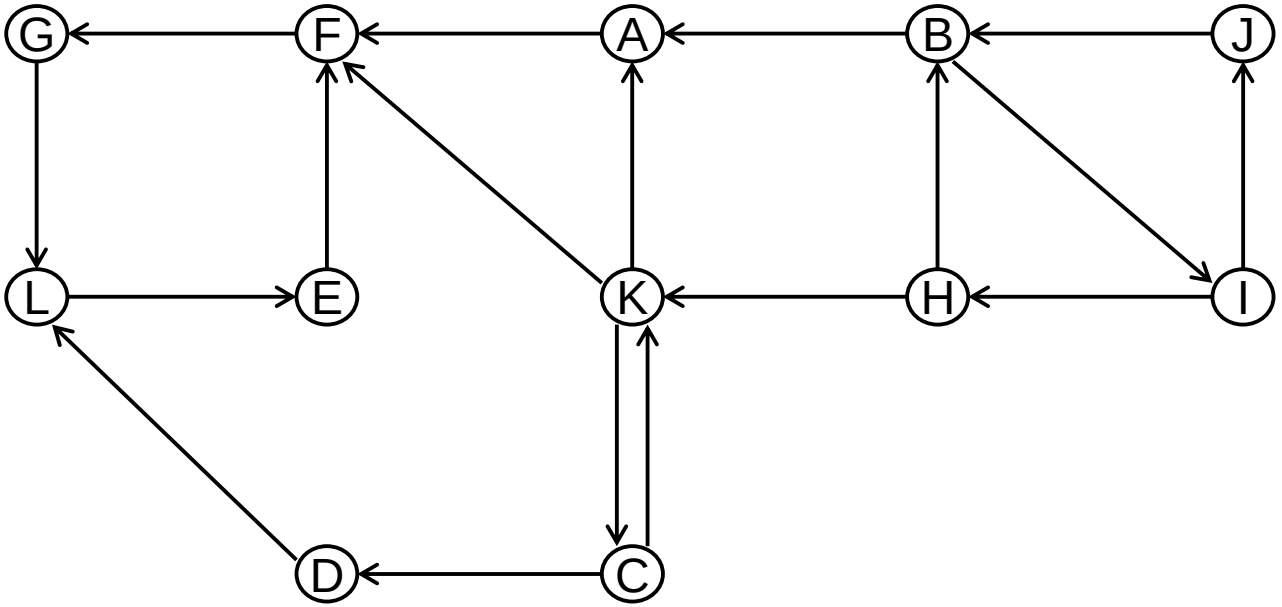
G:



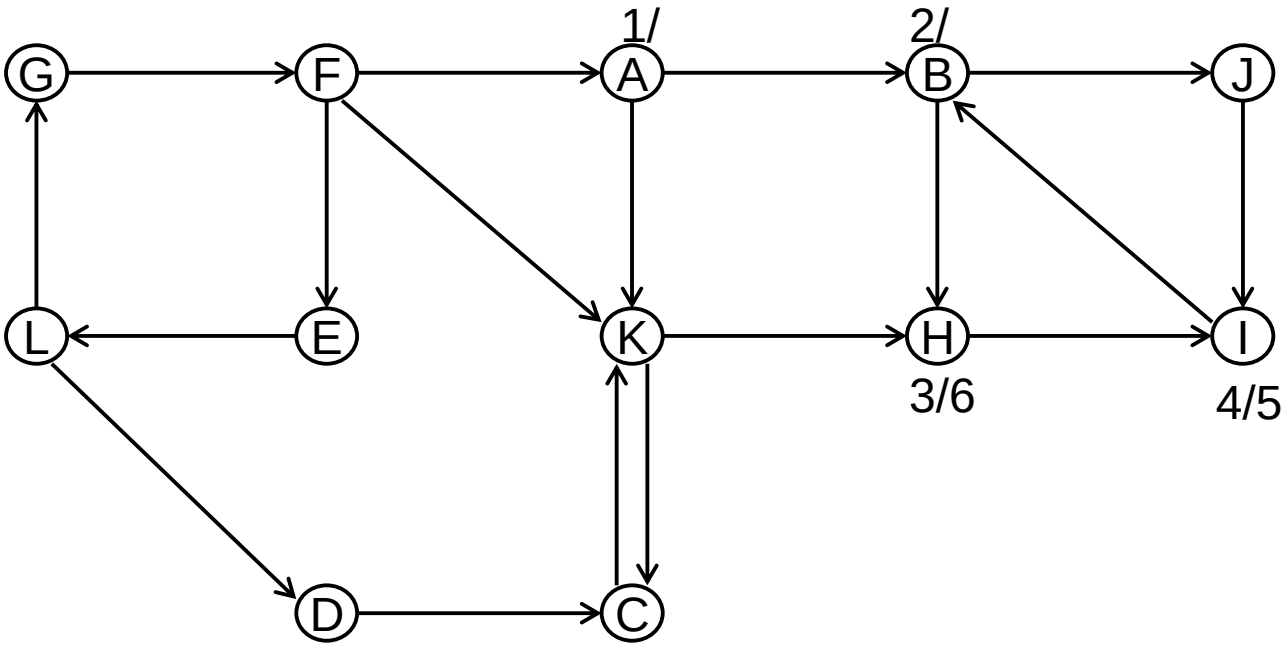
G^R:



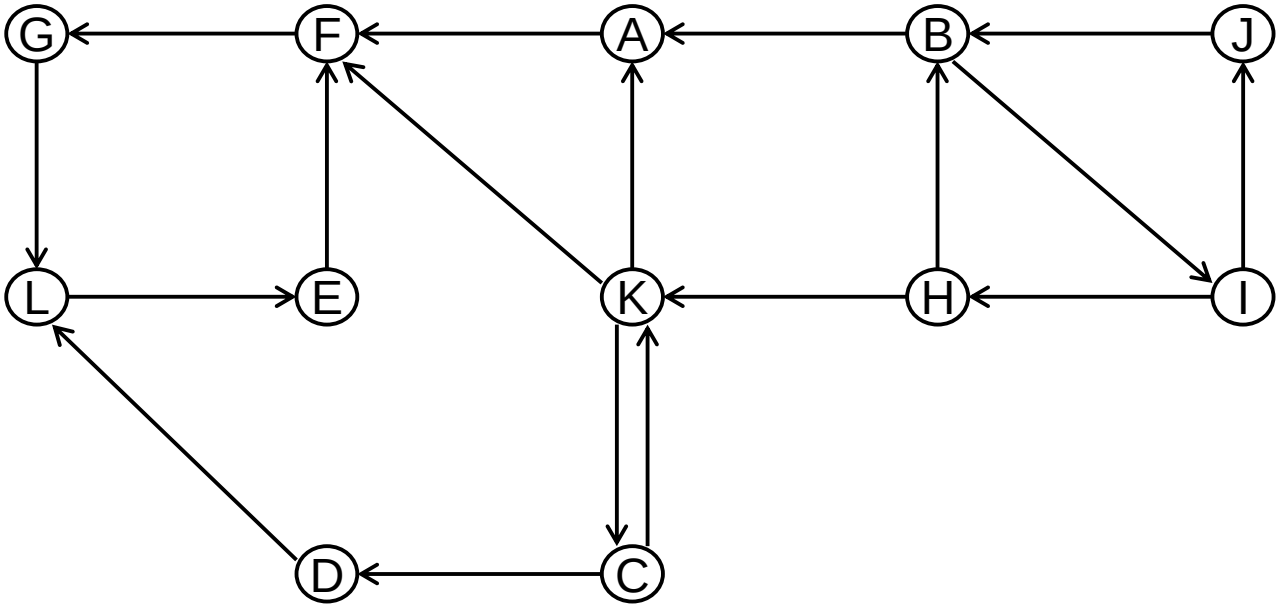
G:



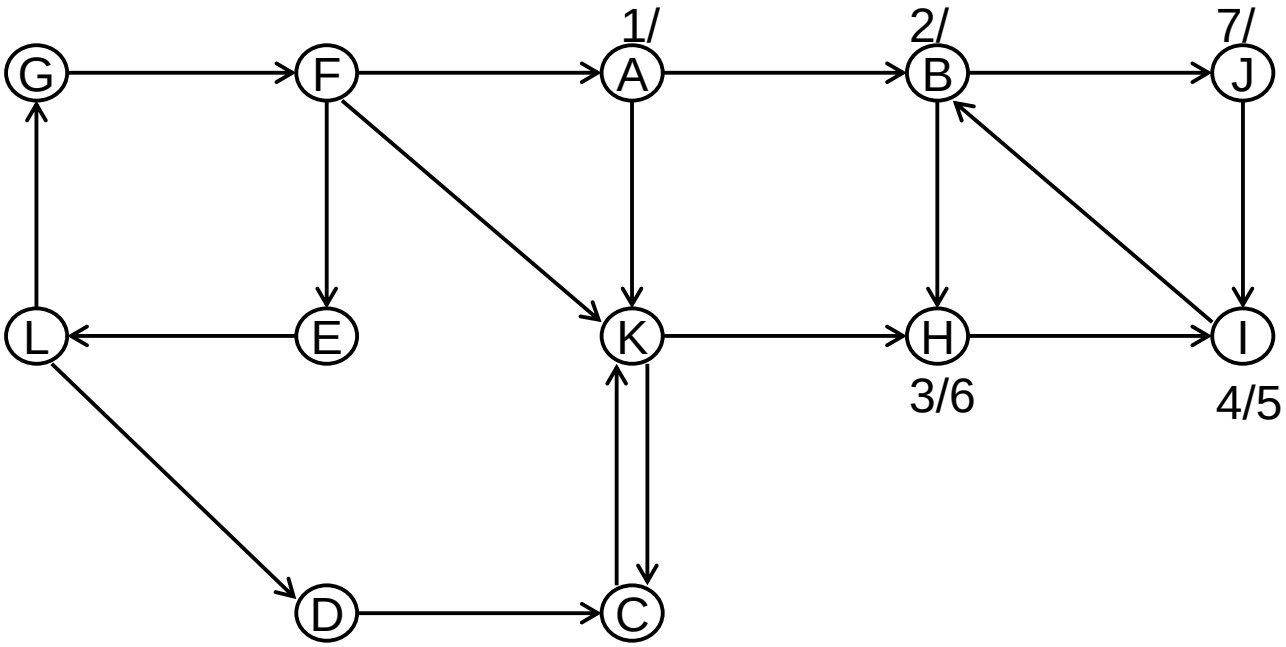
G^R:



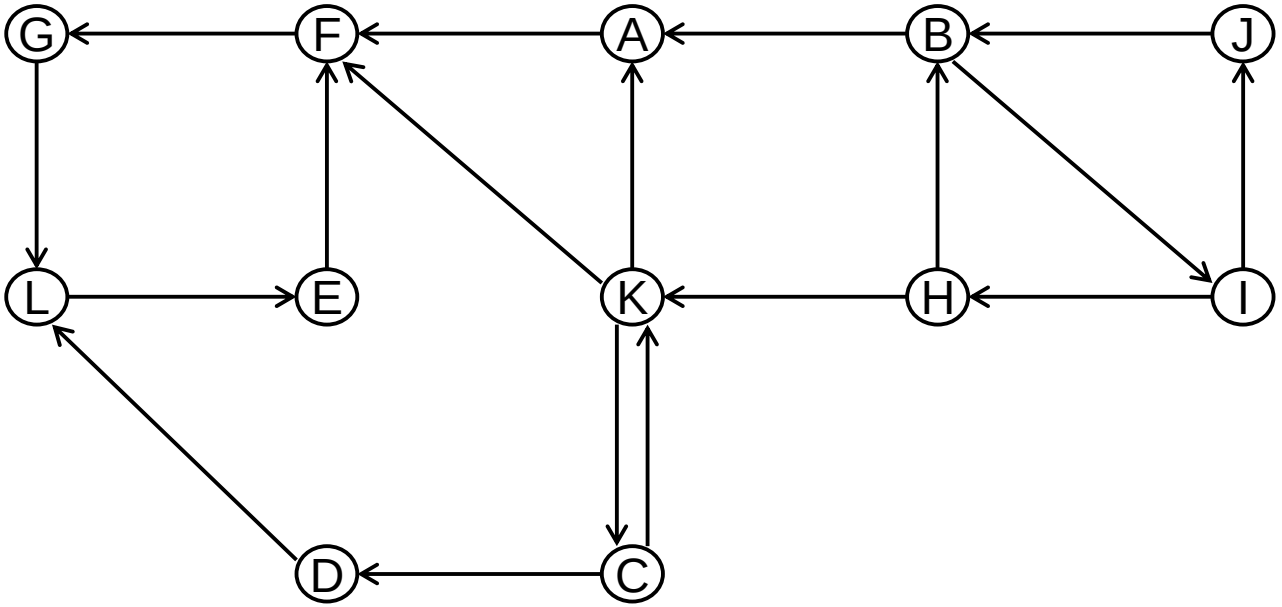
G:



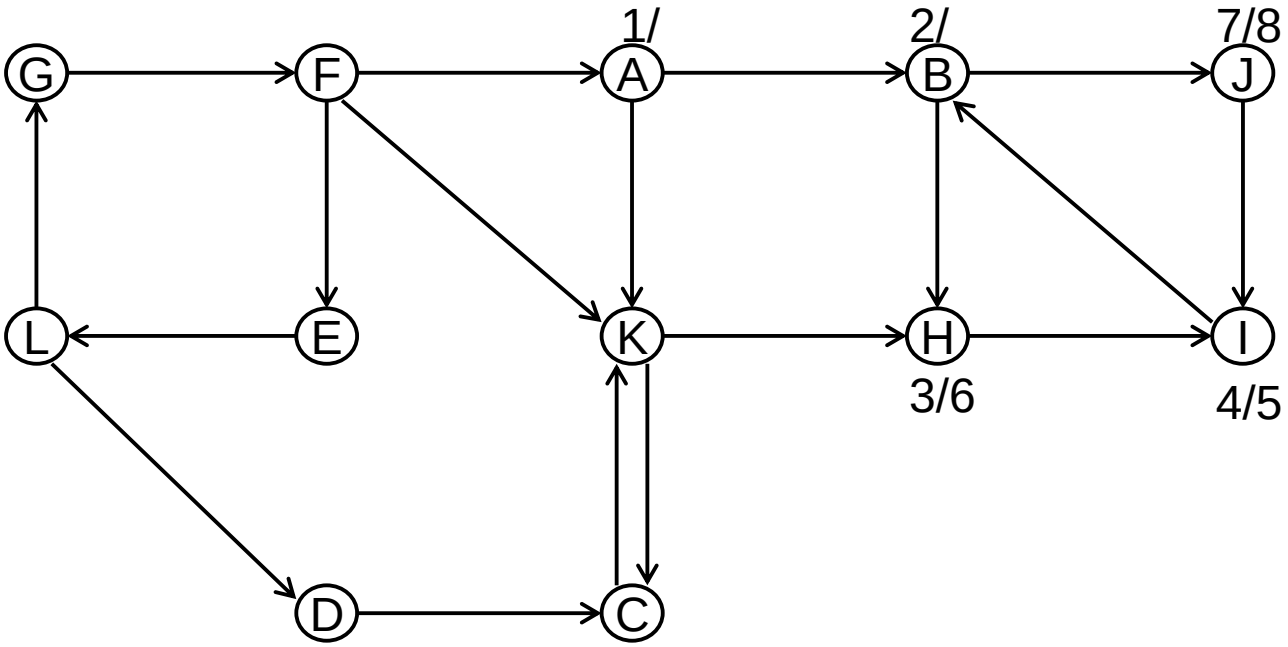
G^R:



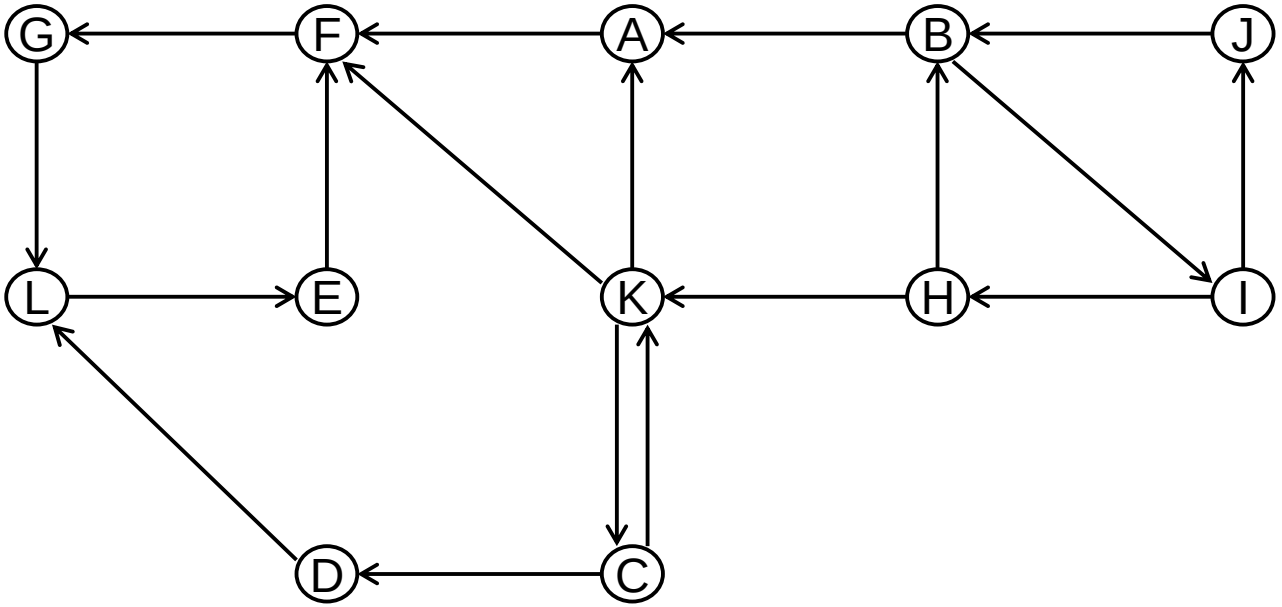
G:



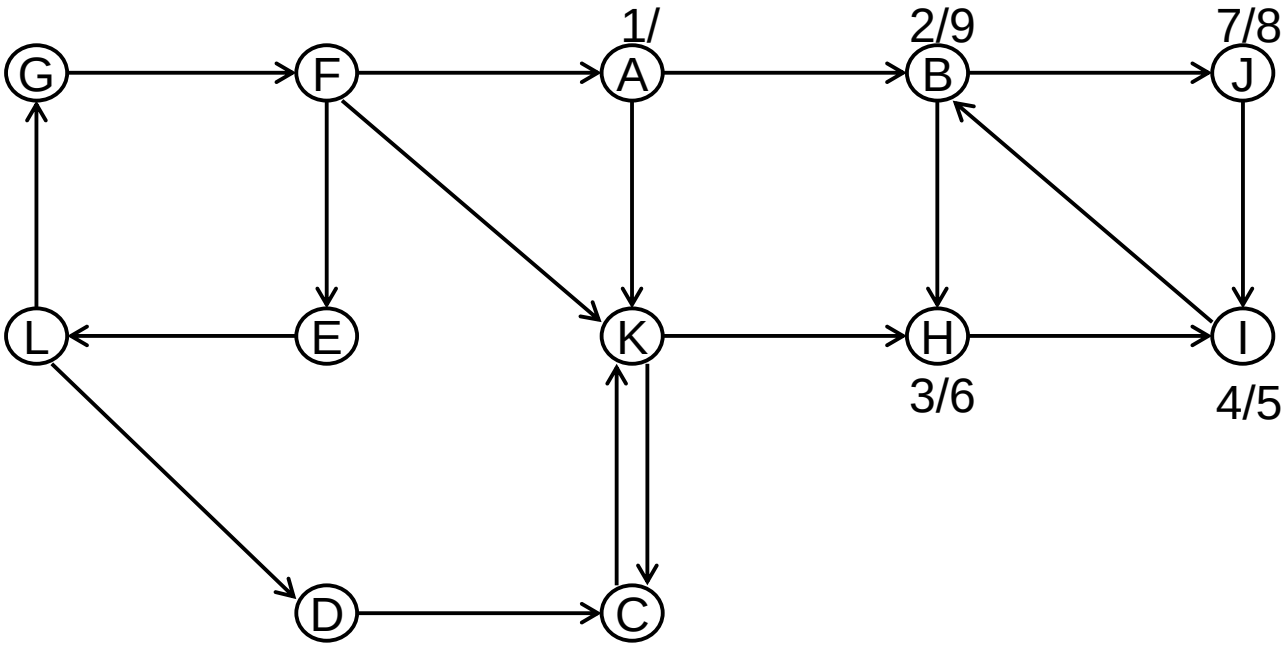
G^R:



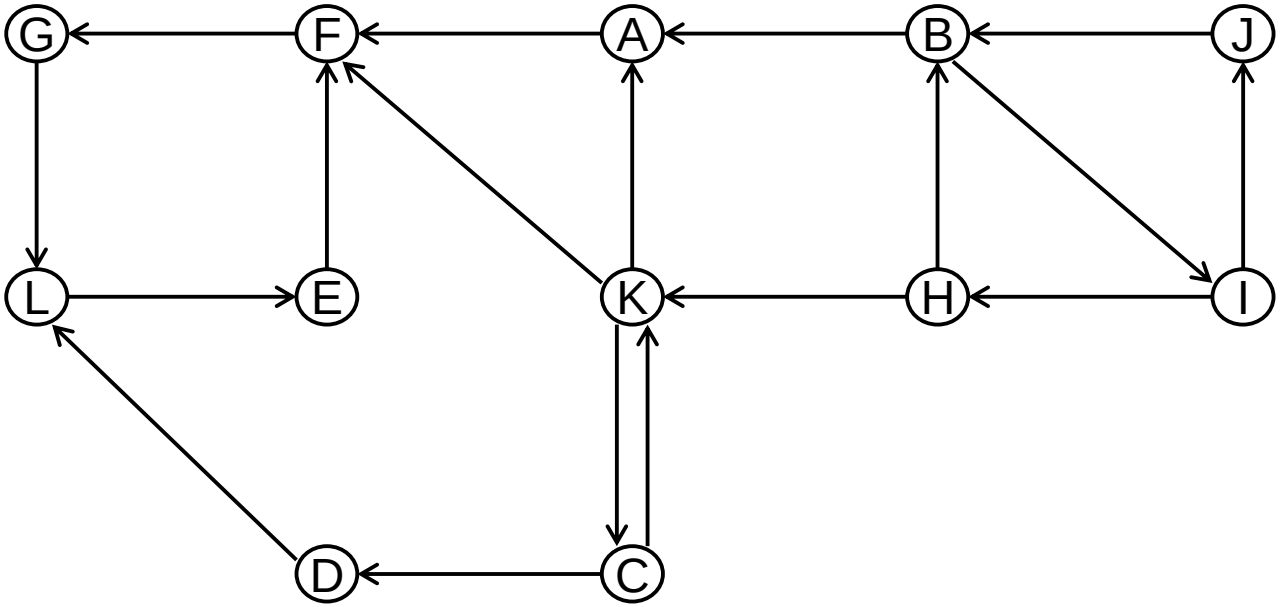
G:



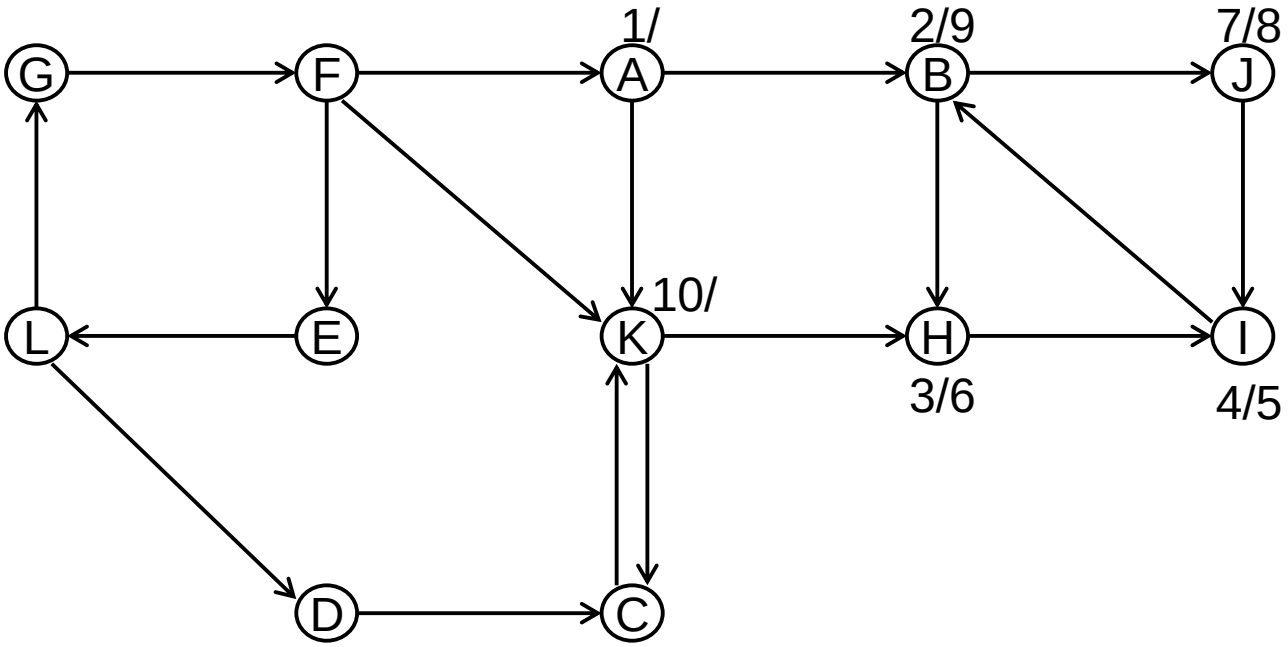
G^R:



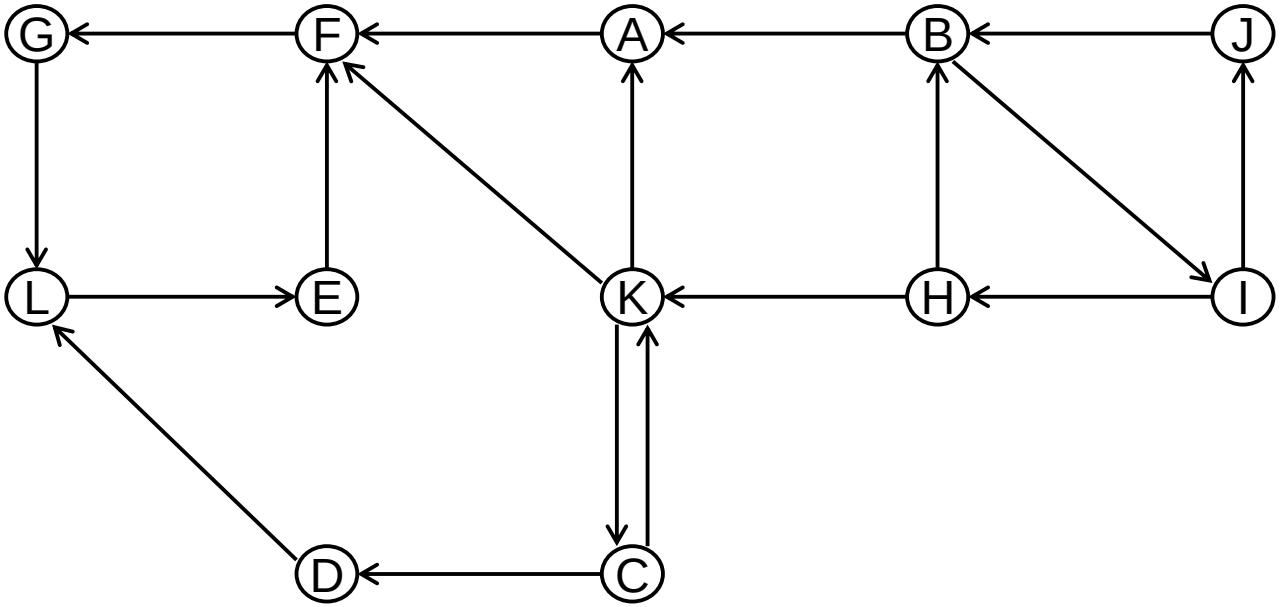
G:



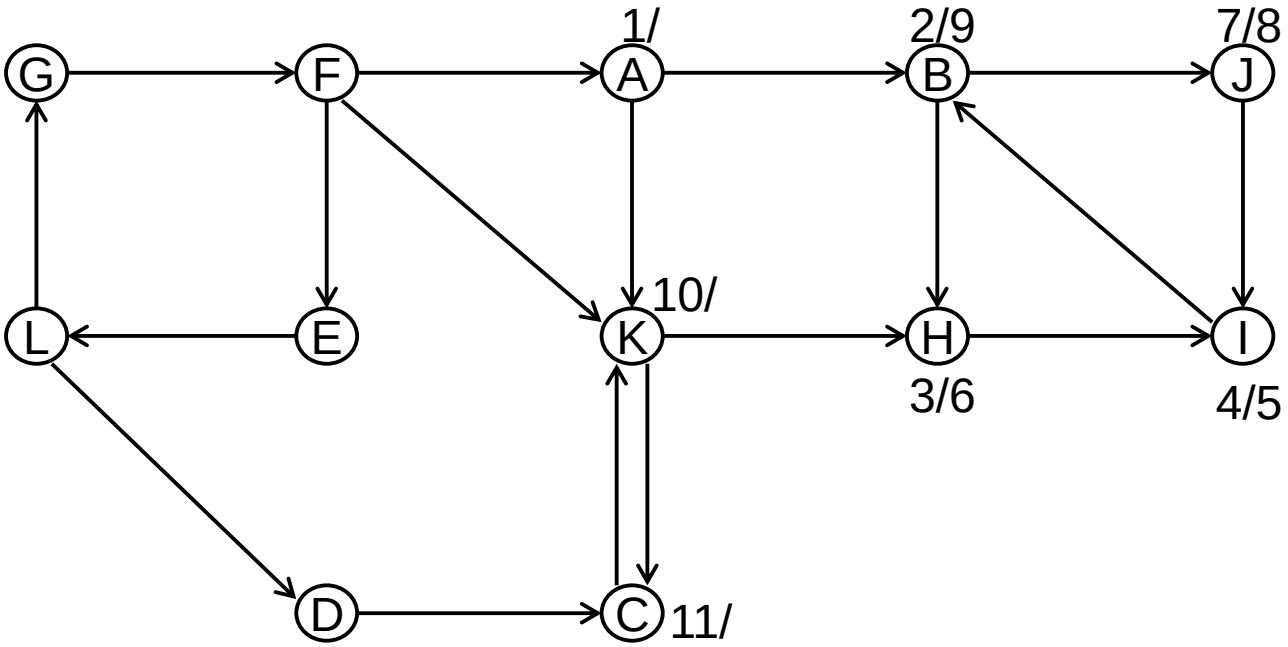
G^R:



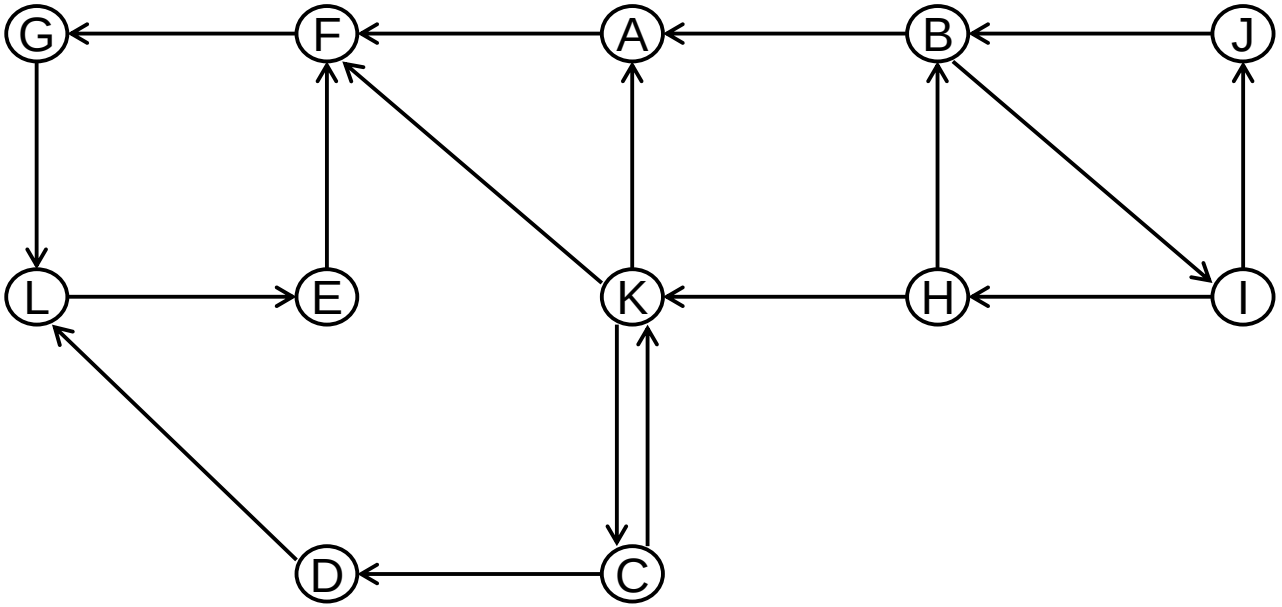
G:



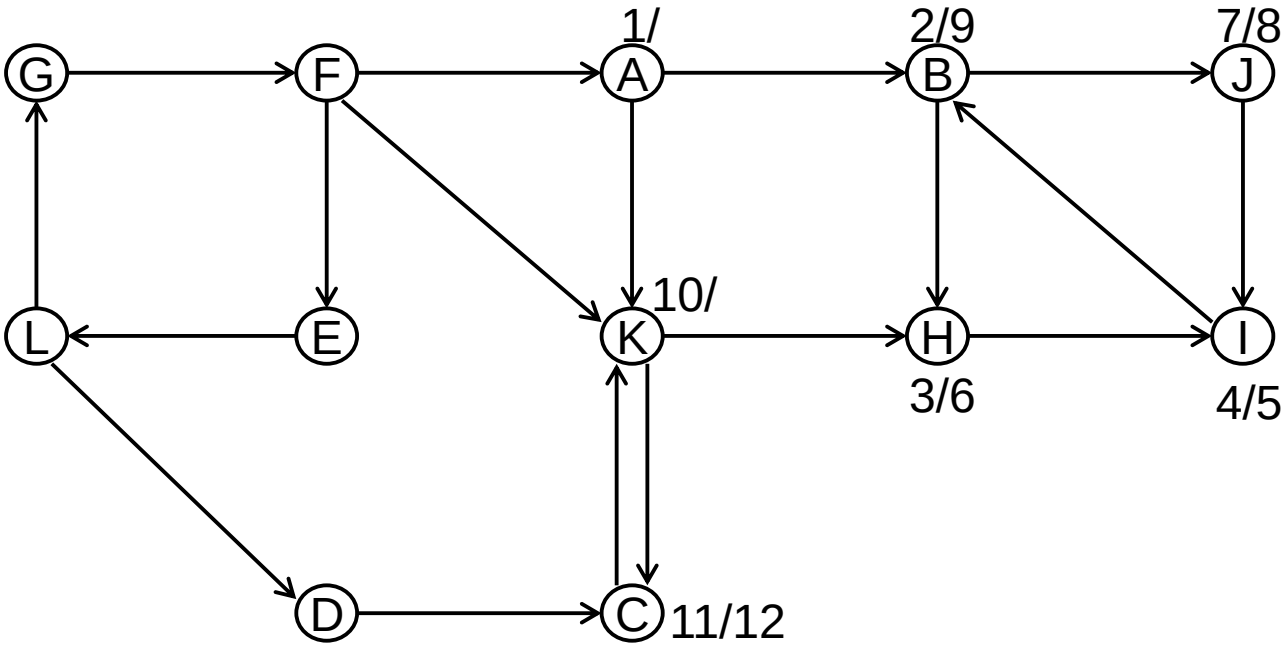
G^R:



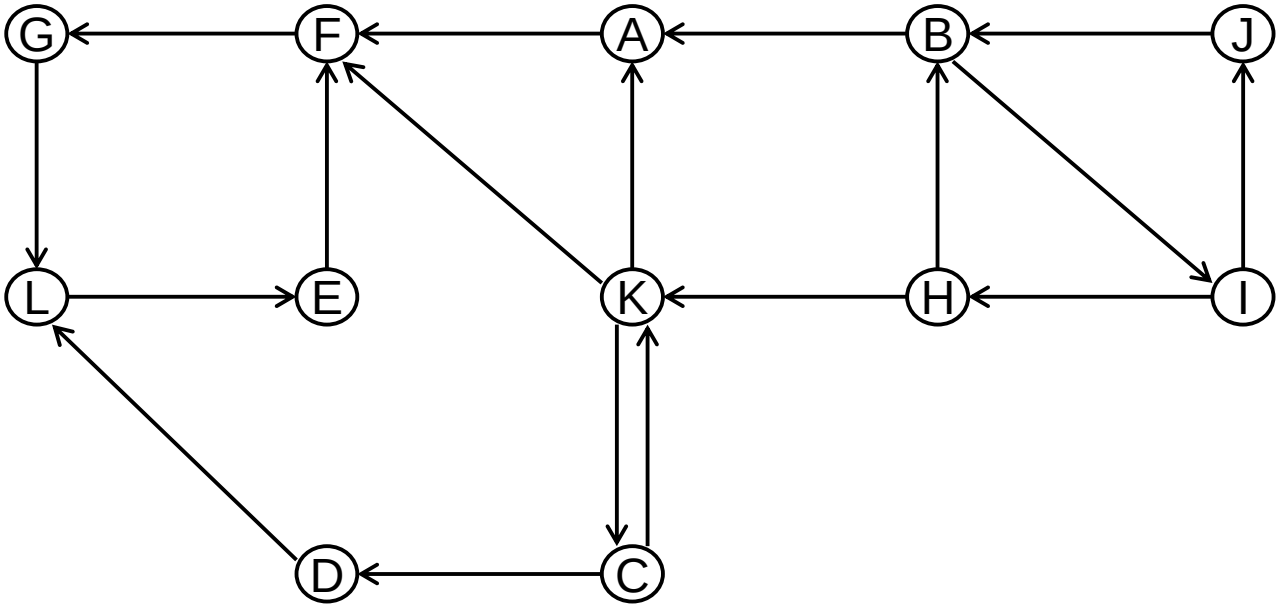
G:



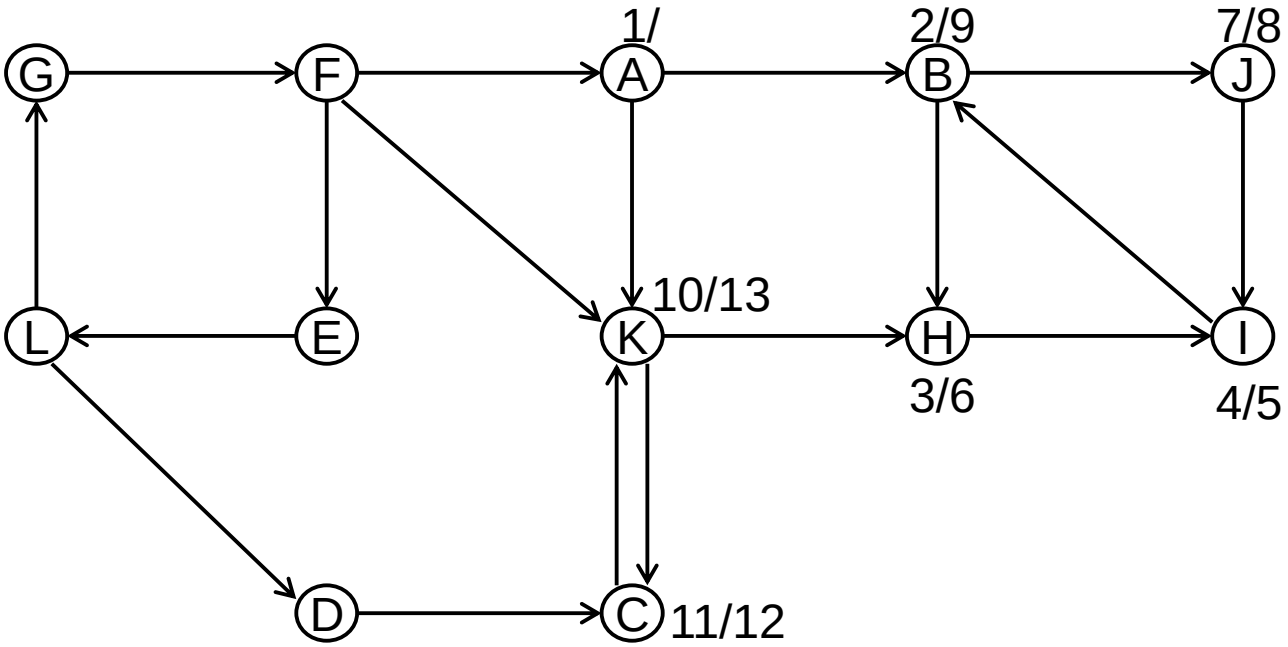
G^R:



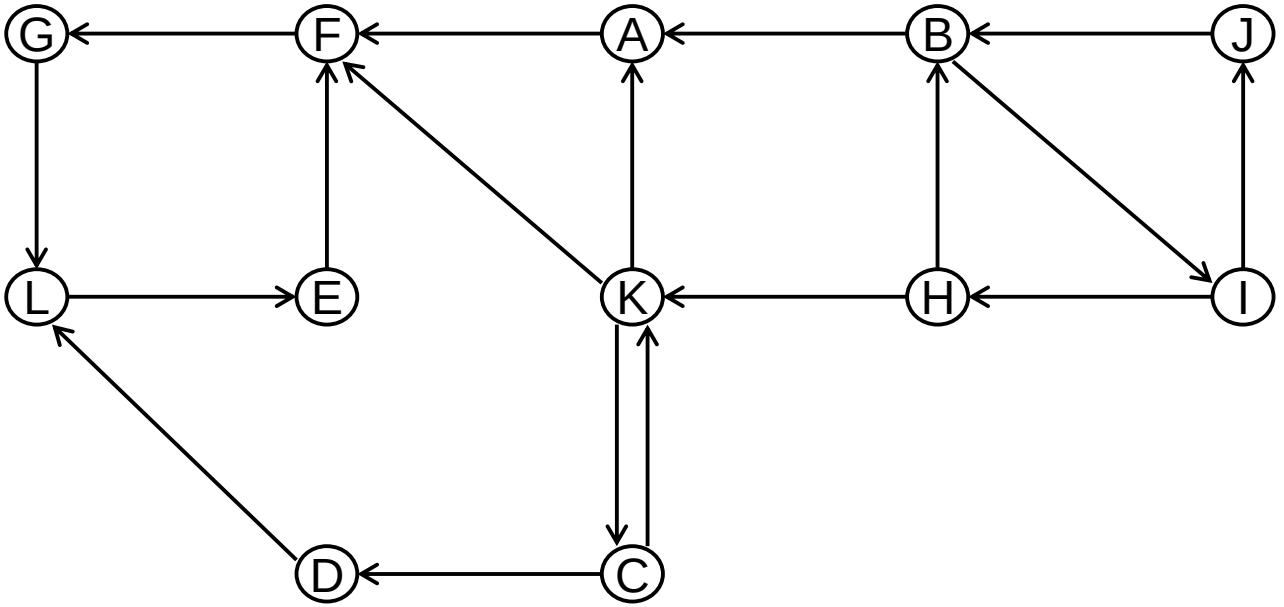
G:



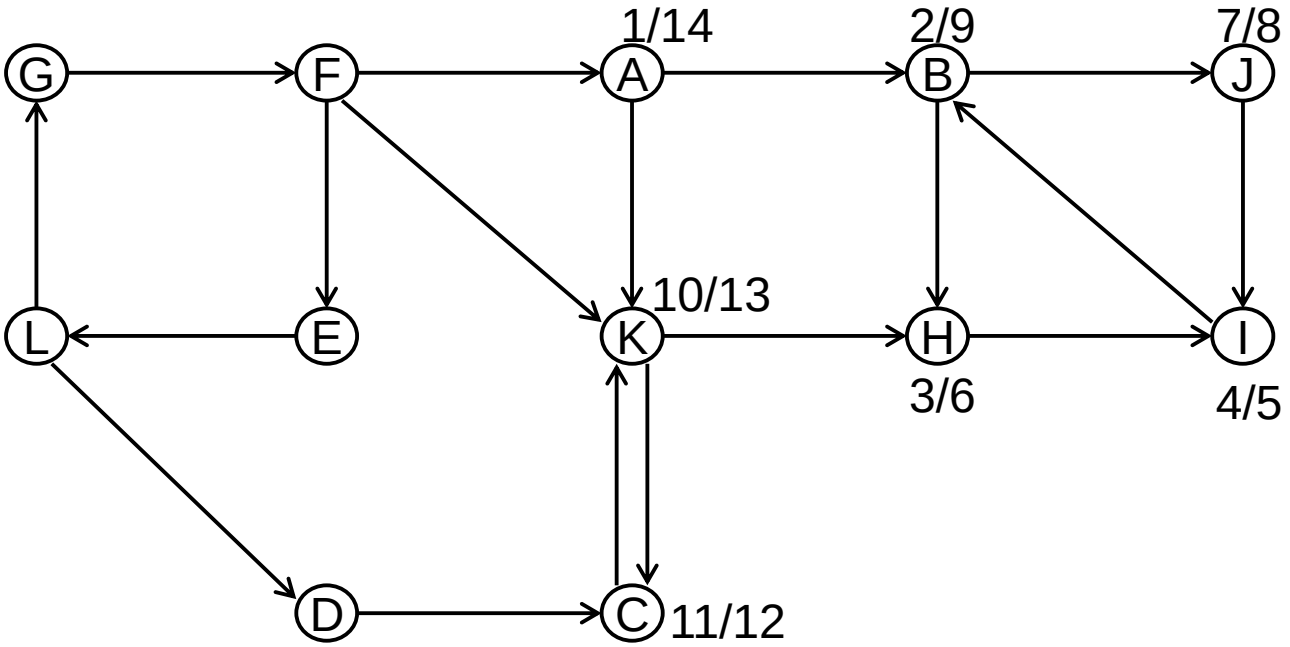
G^R:



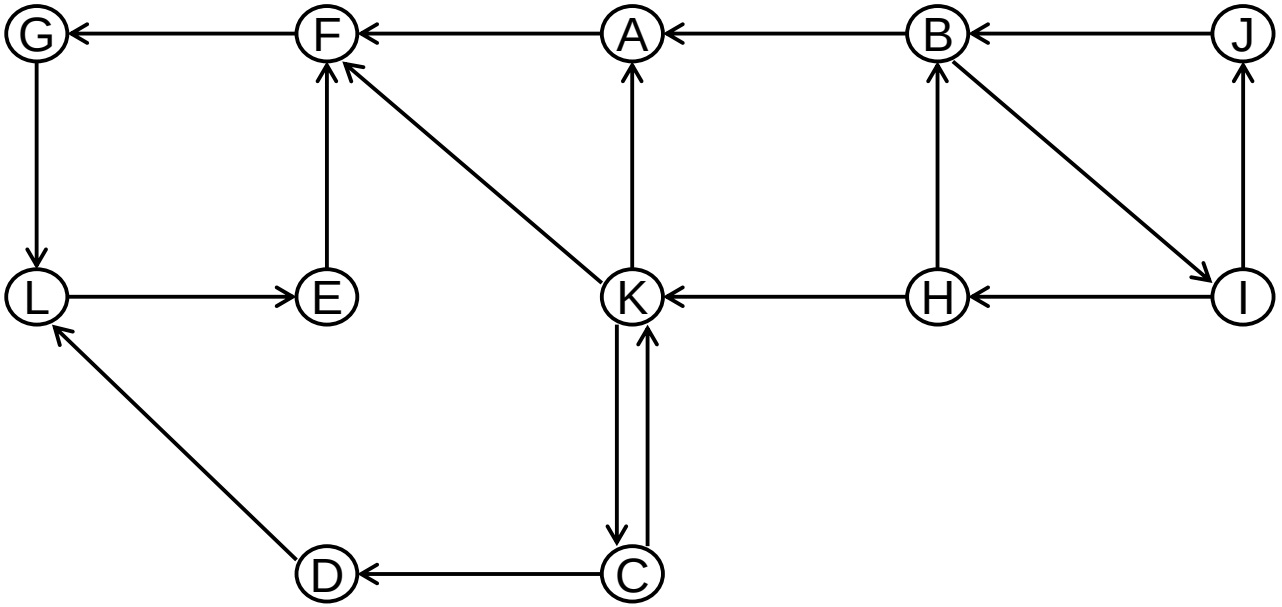
G:



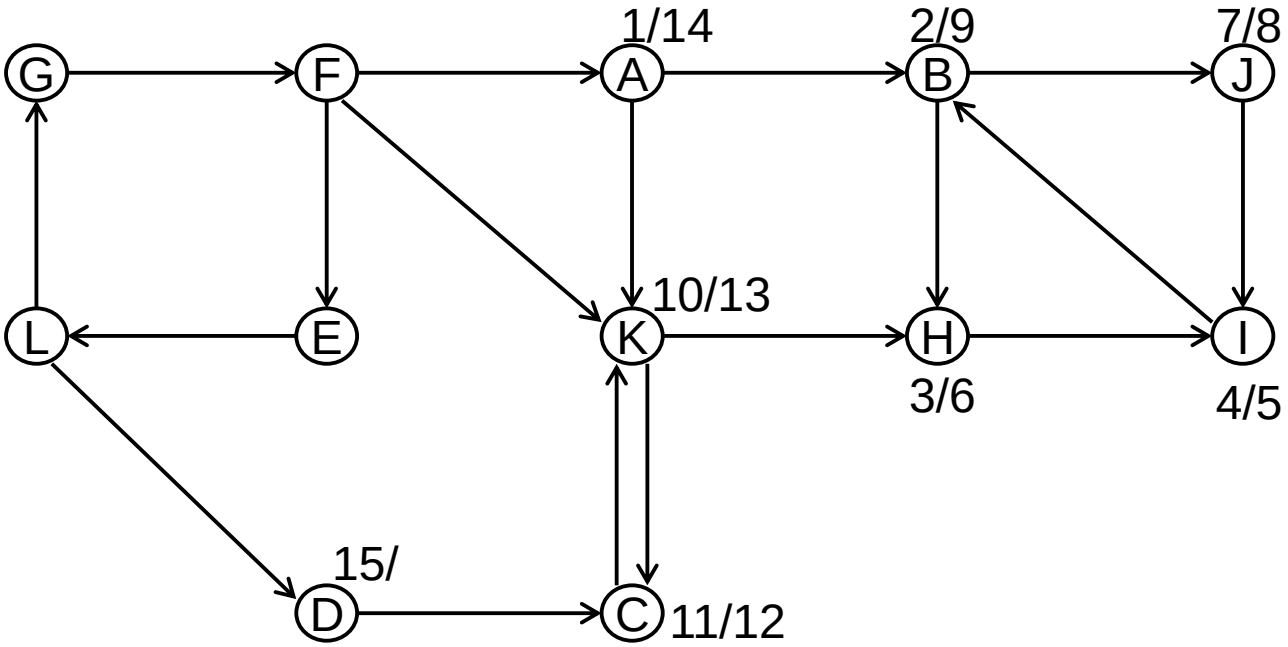
G^R:



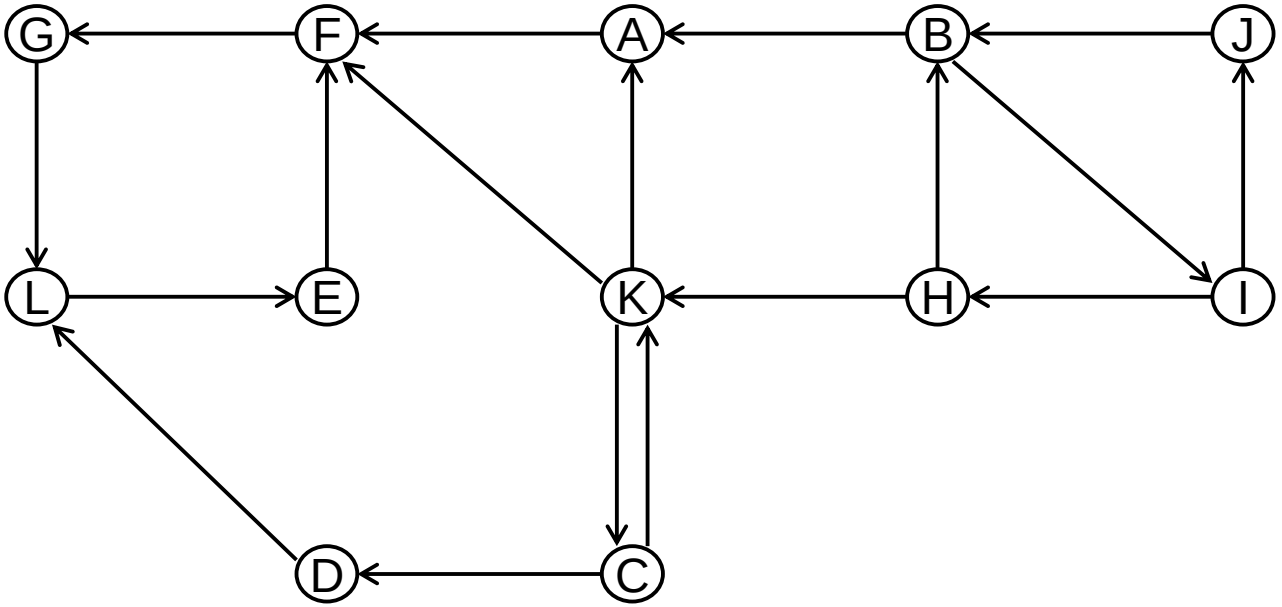
G:



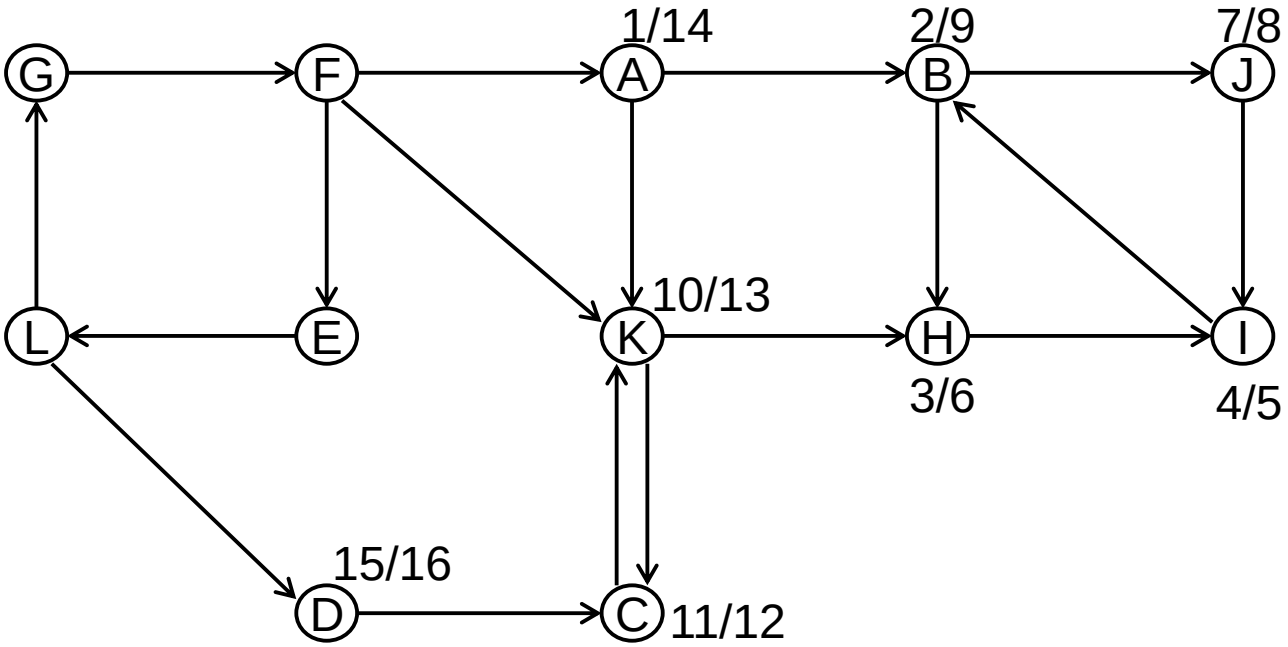
G^R:



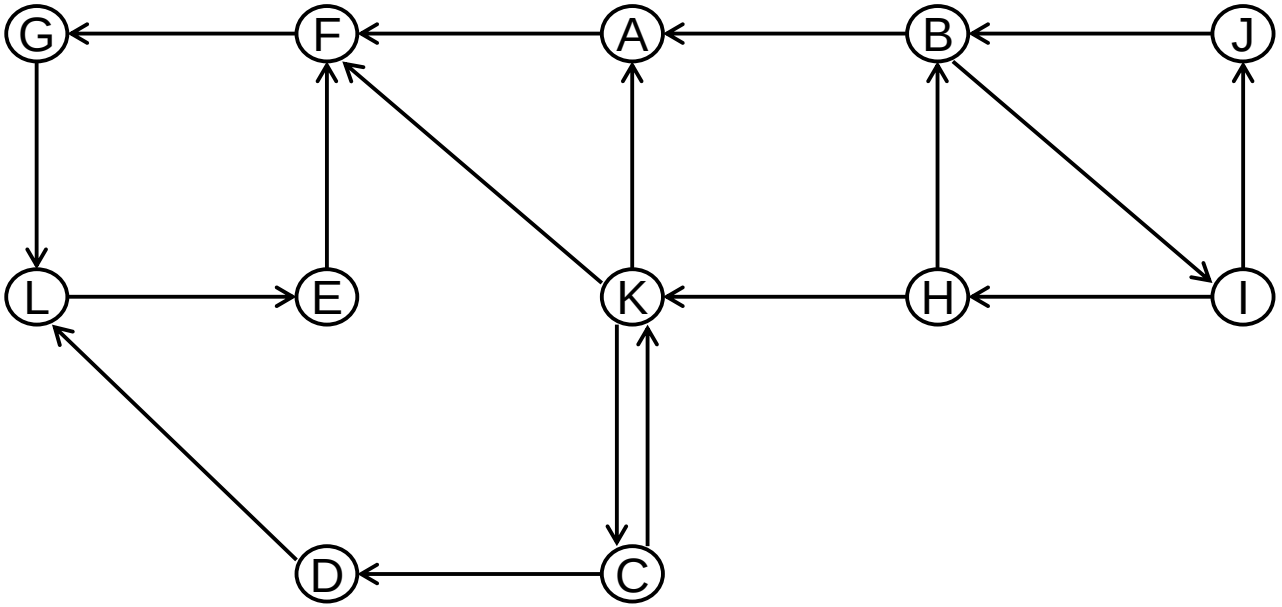
G:



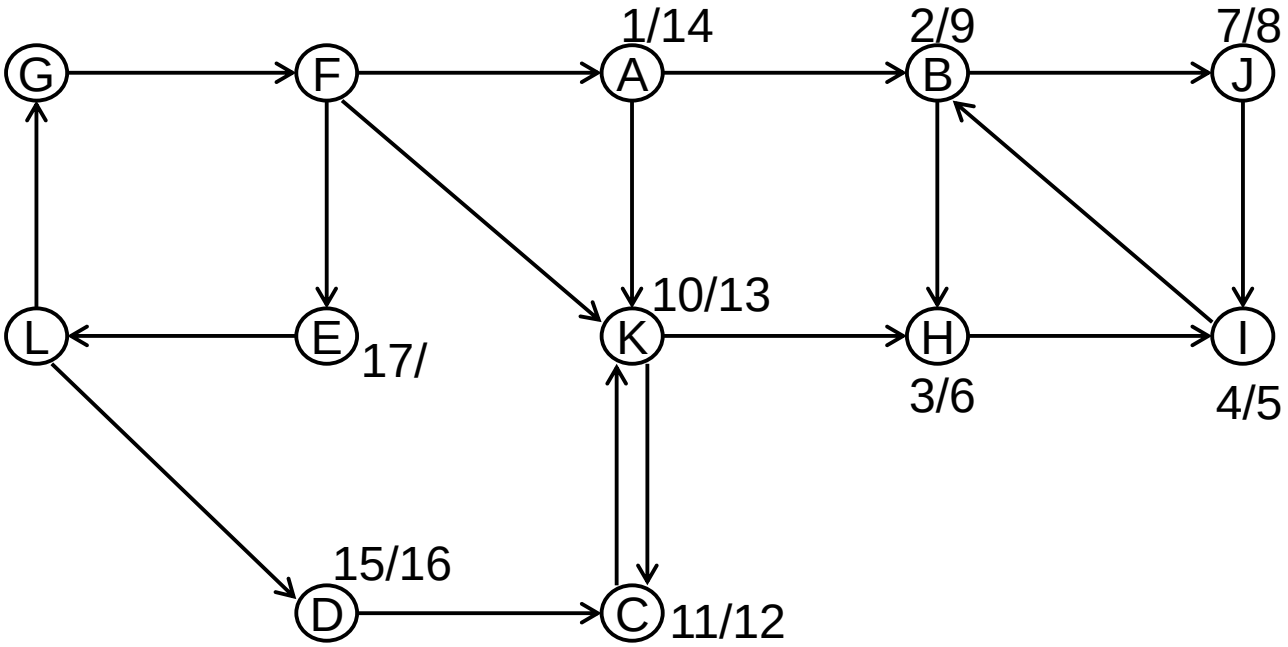
G^R:



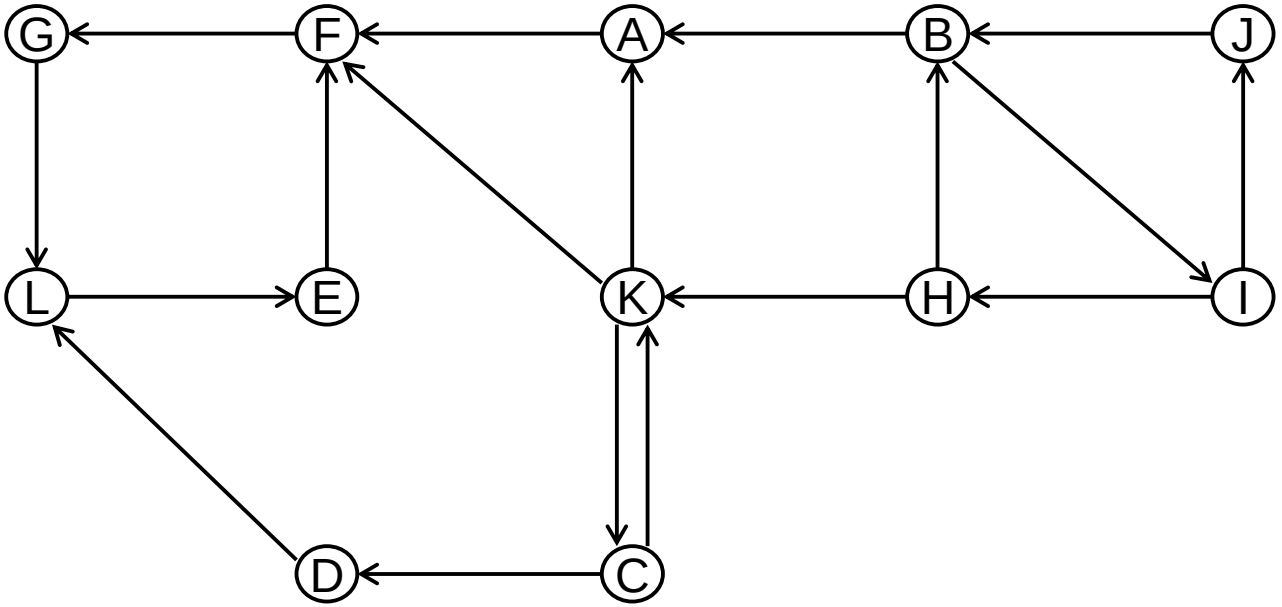
G:



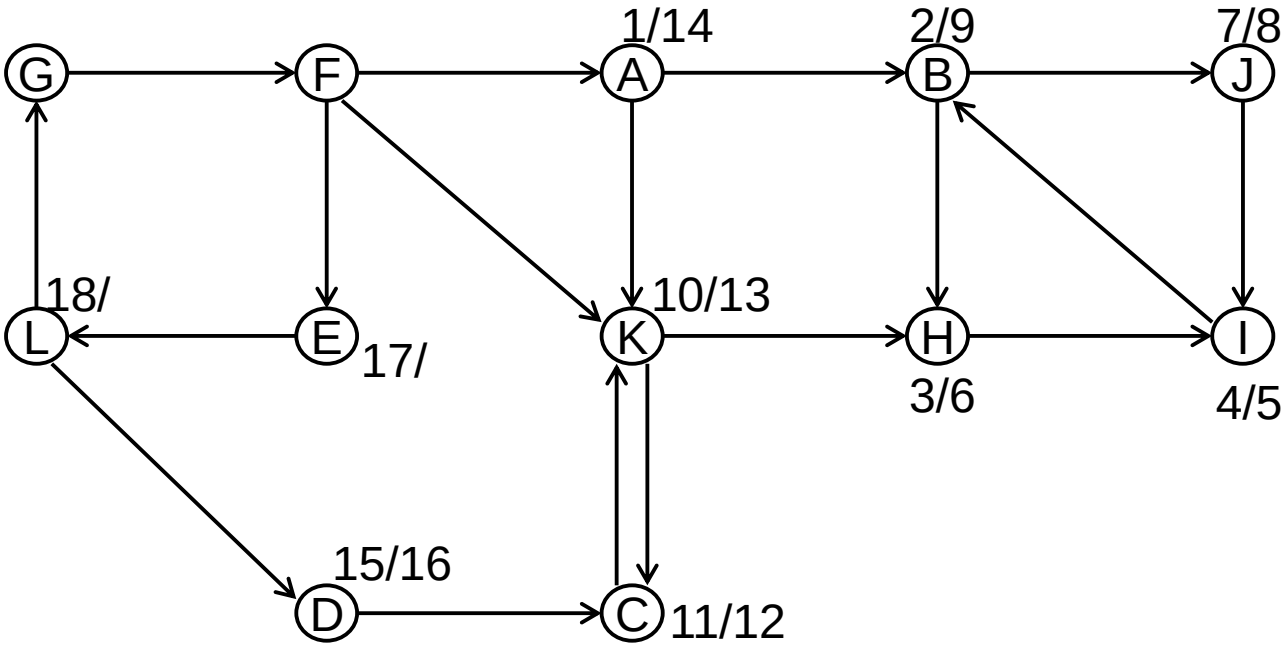
G^R:



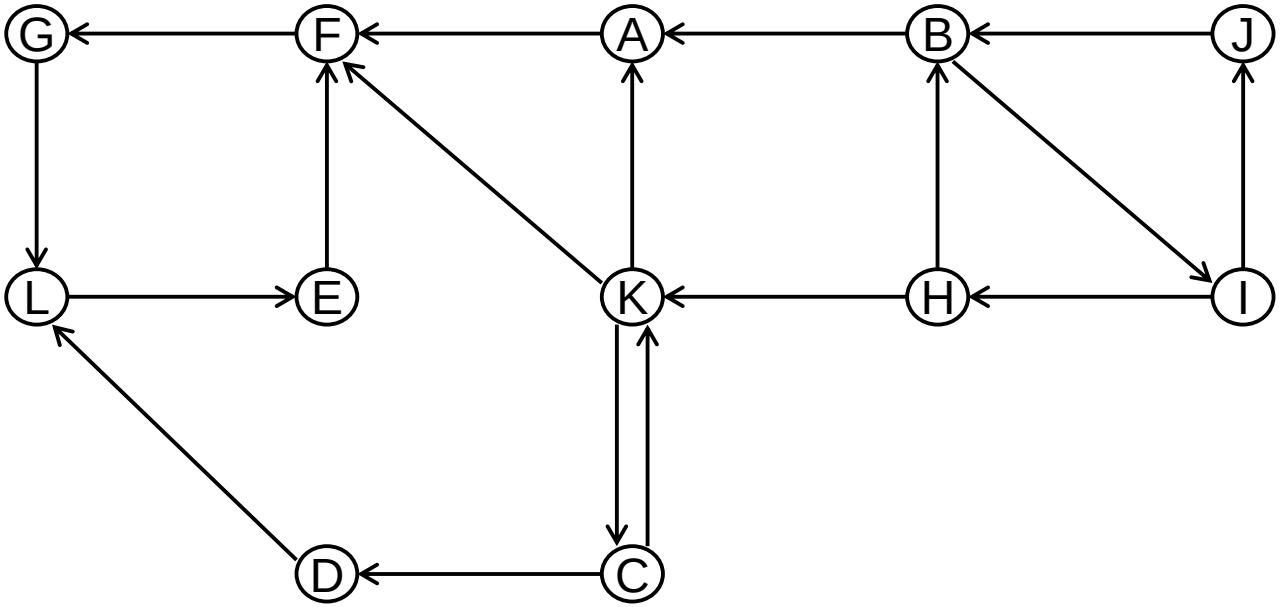
G:


G
F
A
B
J
L
E
K
H
I
D
C

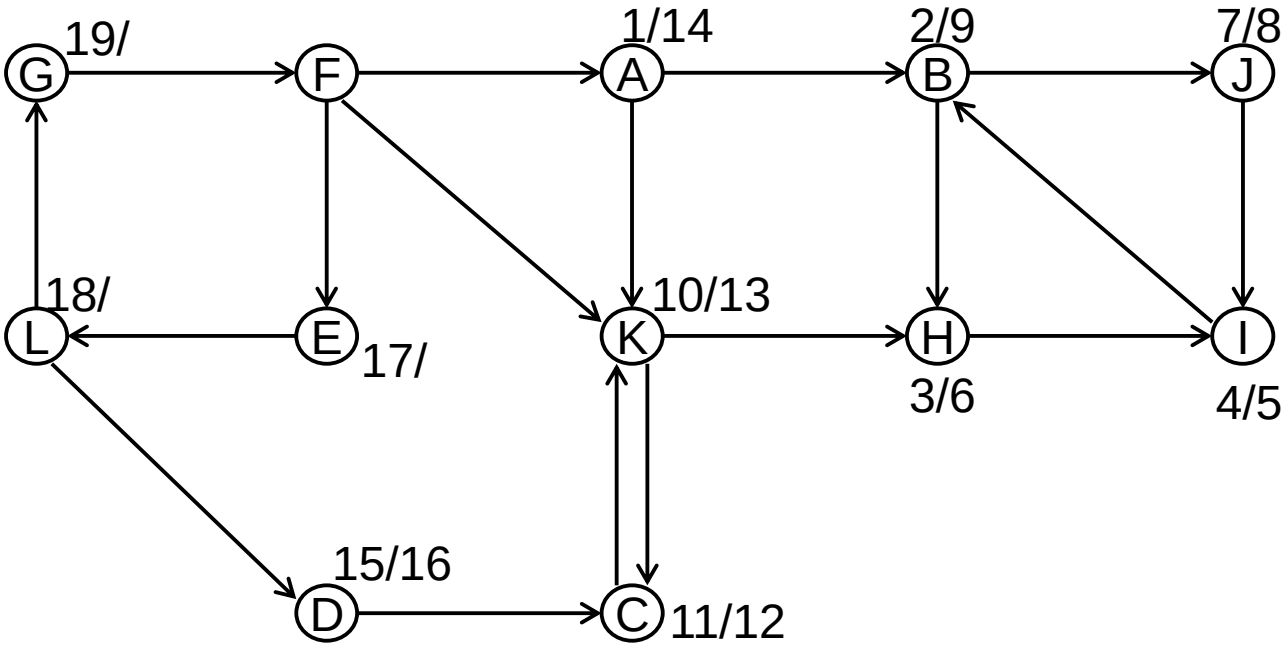
G^R:


1/14
2/9
7/8
G
F
A
B
J
18/
10/13
L
E
17/
K
H
I
3/6
4/5
15/16
D
C
11/12

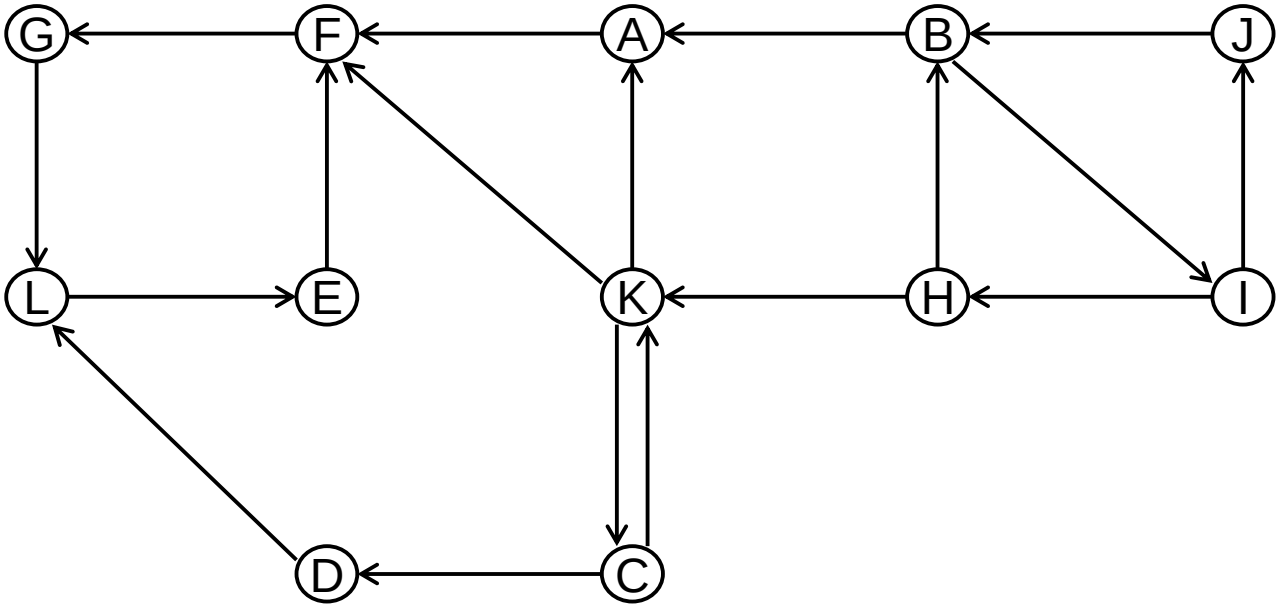
G:


G
F
A
B
J
L
E
K
H
I
D
C

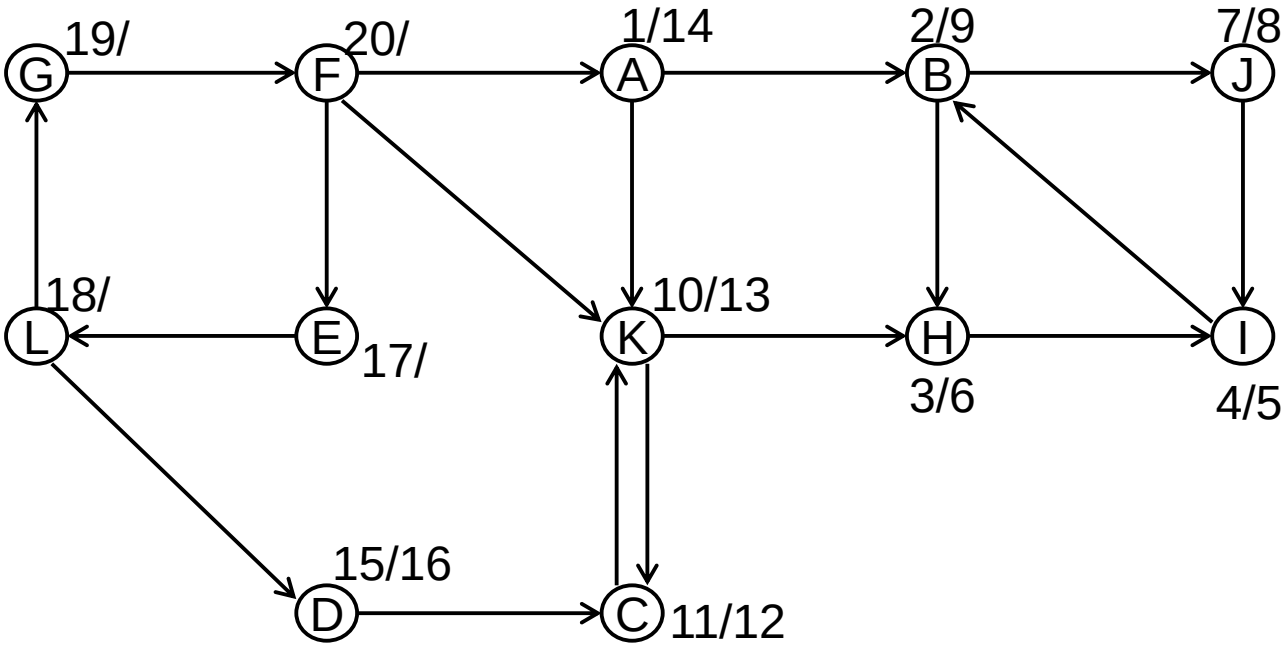
G^R:


19/
1/14
2/9
7/8
G
F
A
B
J
18/
10/13
L
E
17/
K
H
I
3/6
4/5
15/16
D
C
11/12

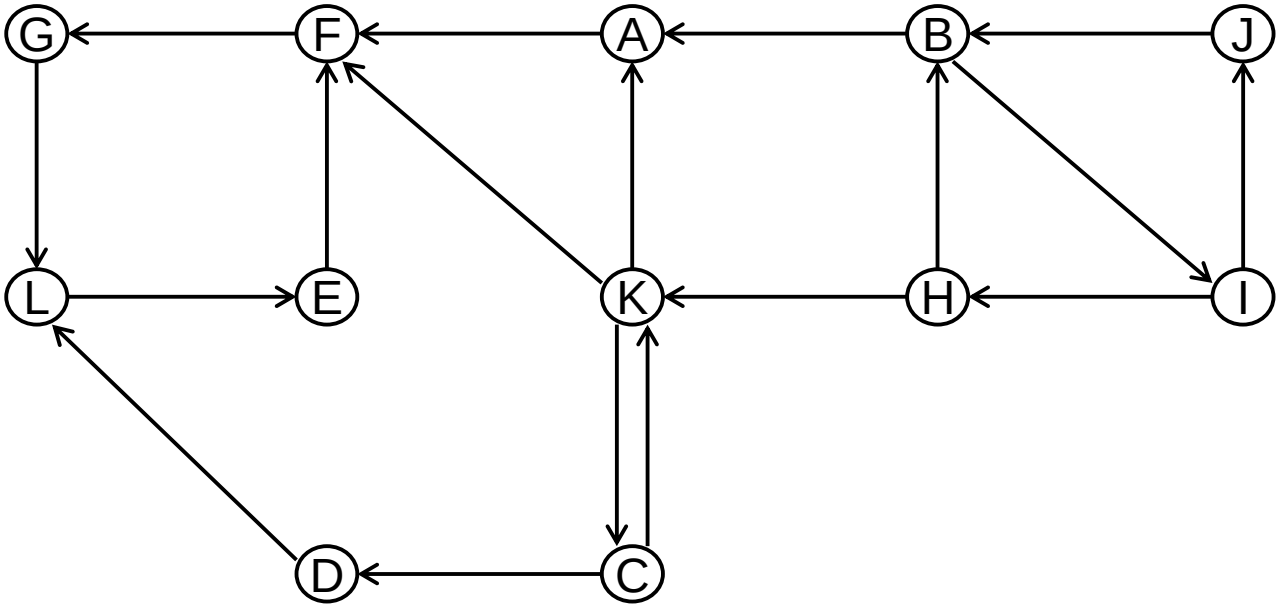
G:


G
F
A
B
J
L
E
K
H
I
D
C

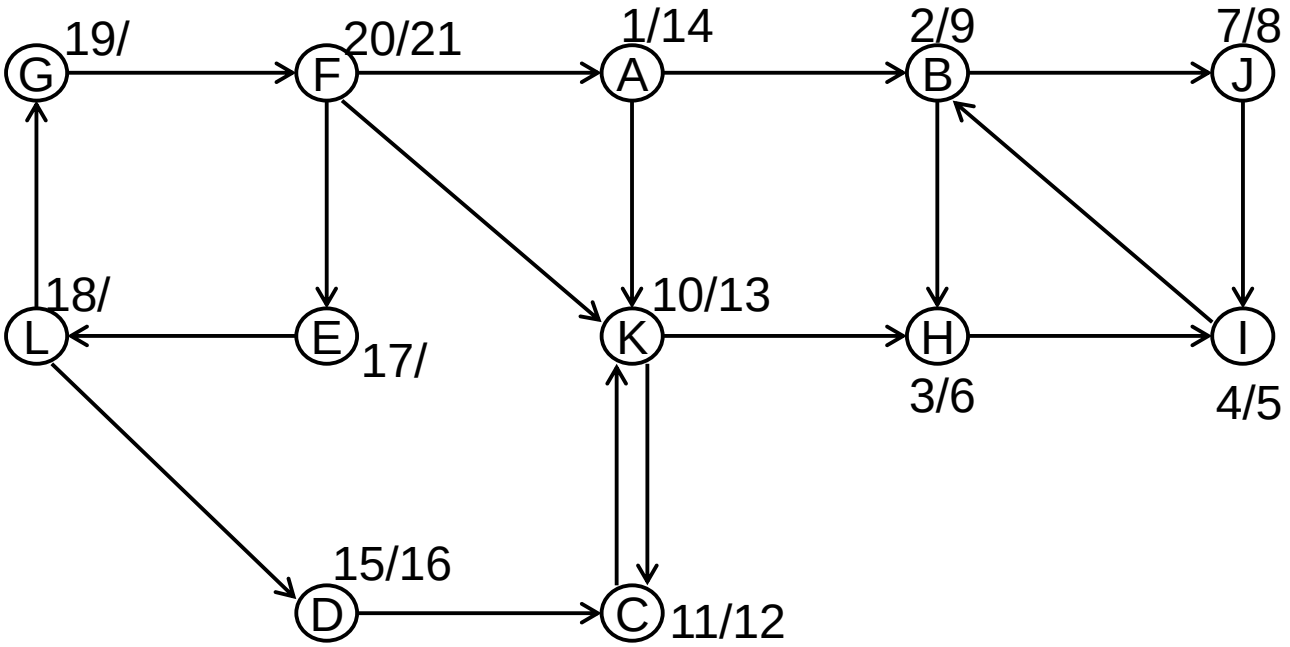
G^R:


19/
20/
1/14
2/9
7/8
G
F
A
B
J
18/
10/13
L
E
17/
K
H
I
3/6
4/5
15/16
D
C
11/12

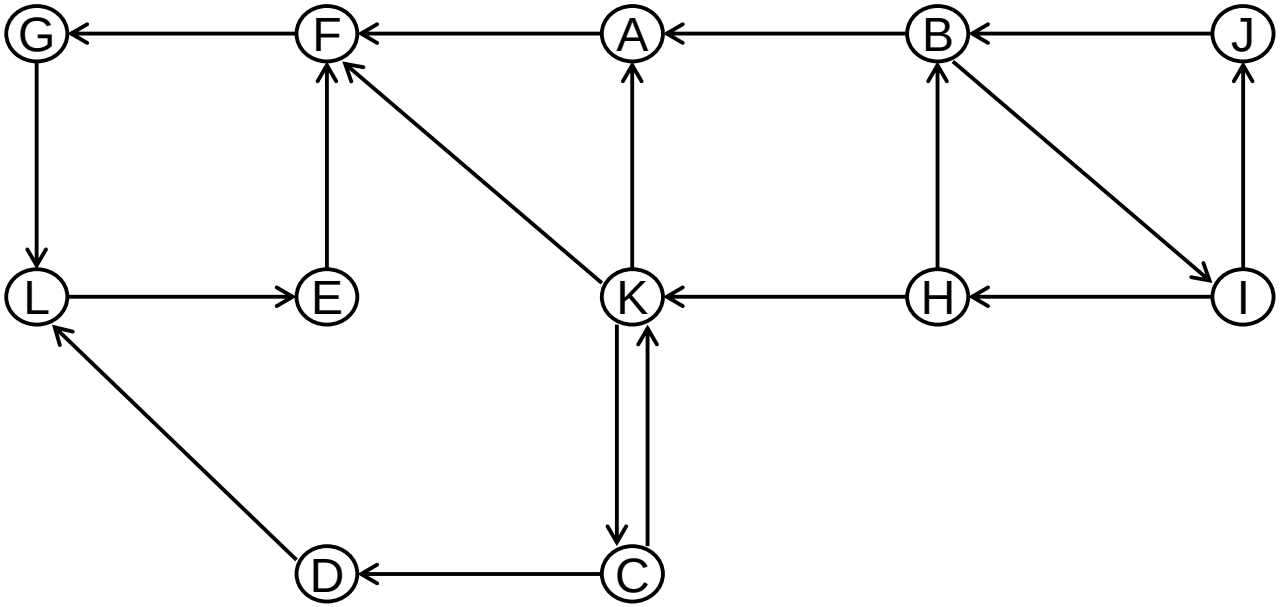
G:


G
F
A
B
J
L
E
K
H
I
D
C

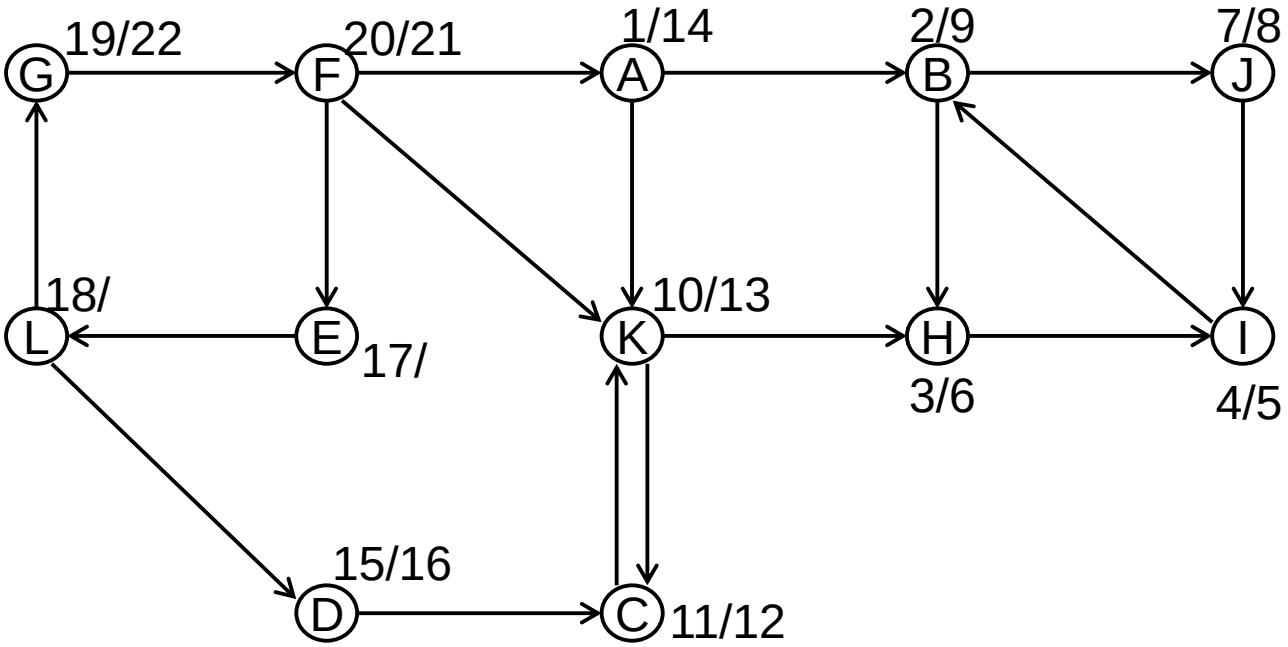
G^R:


19/
G
20/21
F
1/14
A
2/9
B
7/8
J
18/
L
E
17/
10/13
K
H
3/6
I
4/5
15/16
D
C
11/12

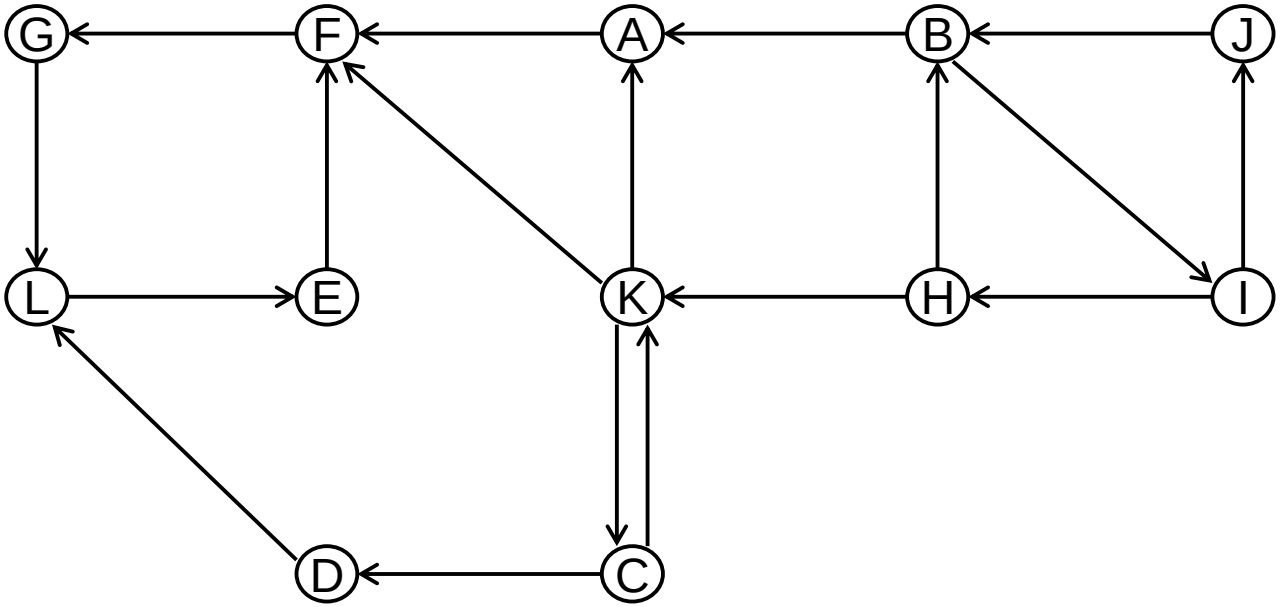
G:


G
F
A
B
J
L
E
K
H
I
D
C

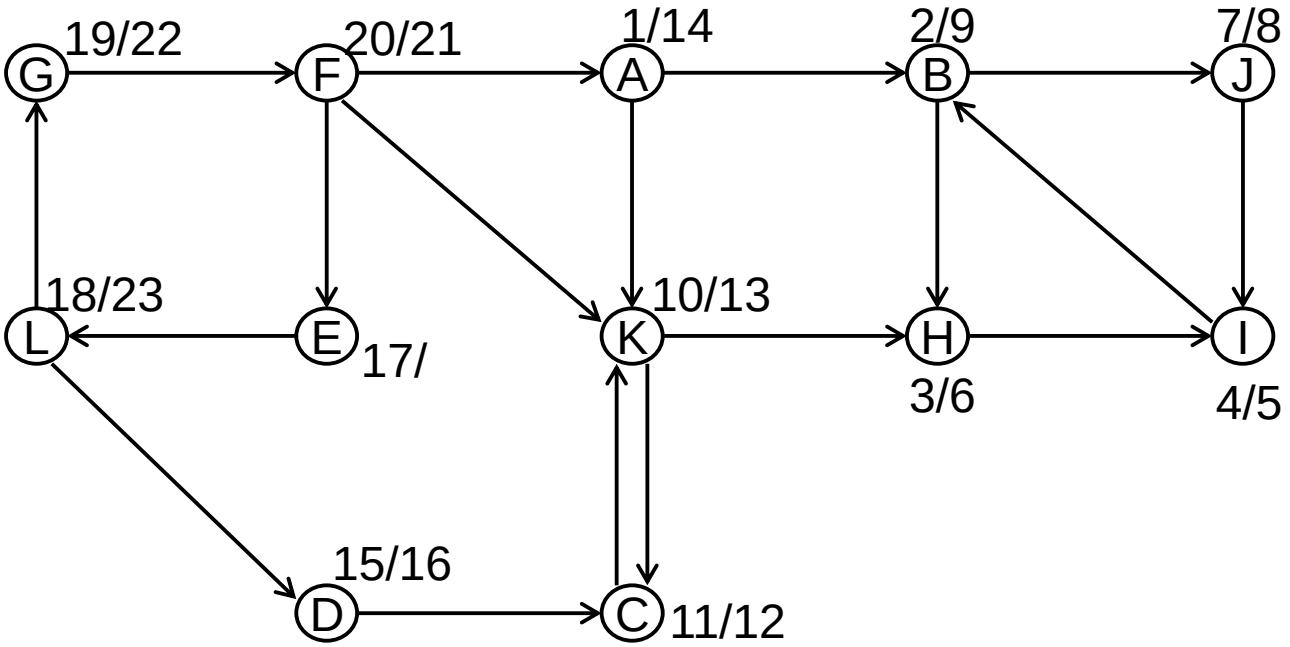
G^R:


19/22
20/21
1/14
2/9
7/8
G
F
A
B
J
18/
10/13
L
E
17/
K
H
I
3/6
4/5
15/16
D
C
11/12

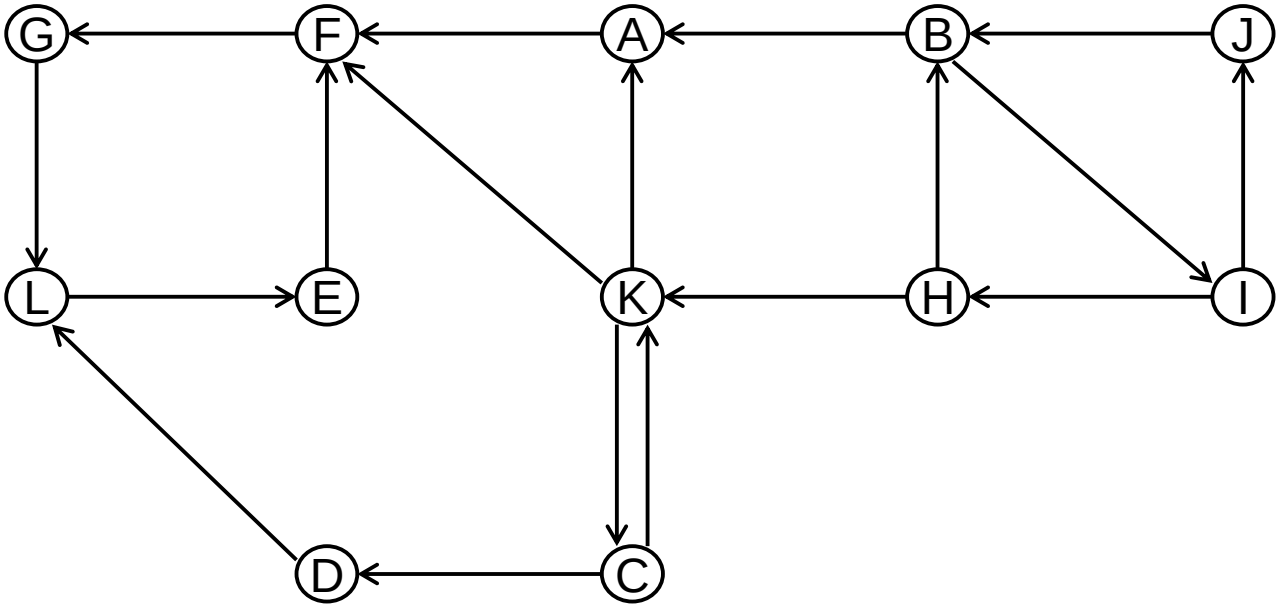
G:


G
F
A
B
J
L
E
K
H
I
D
C

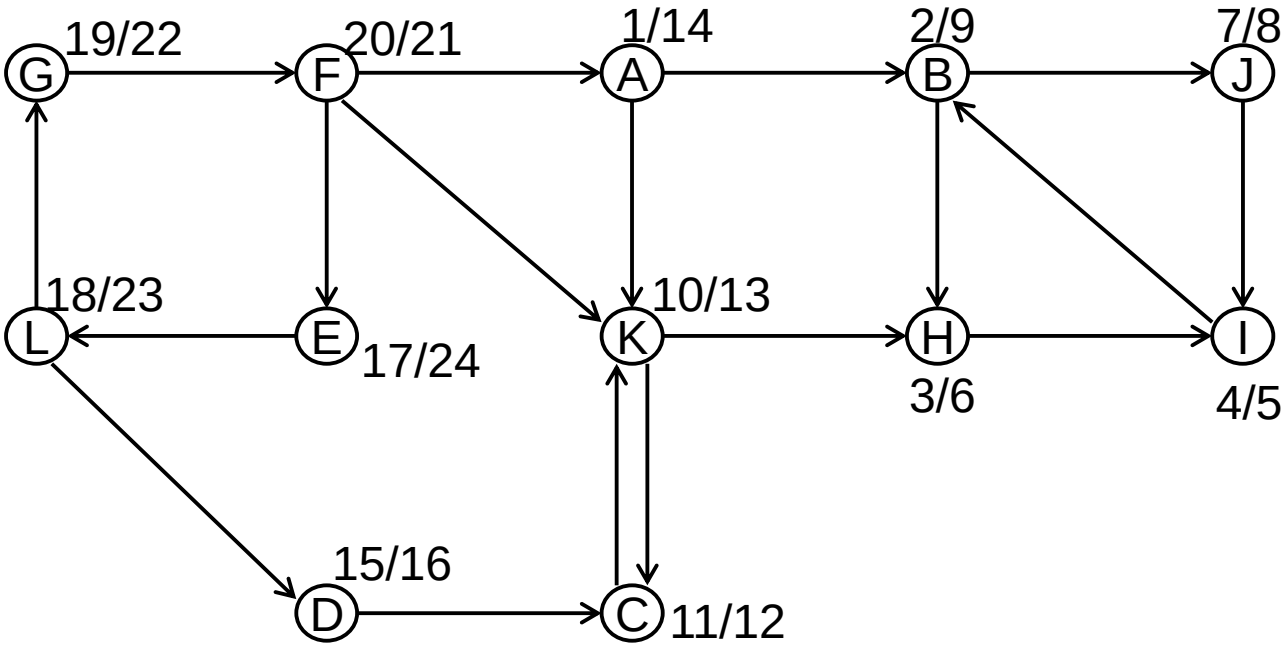
G^R:


19/22
20/21
1/14
2/9
7/8
G
F
A
B
J
18/23
10/13
L
E
17/
K
H
I
3/6
4/5
15/16
D
C
11/12

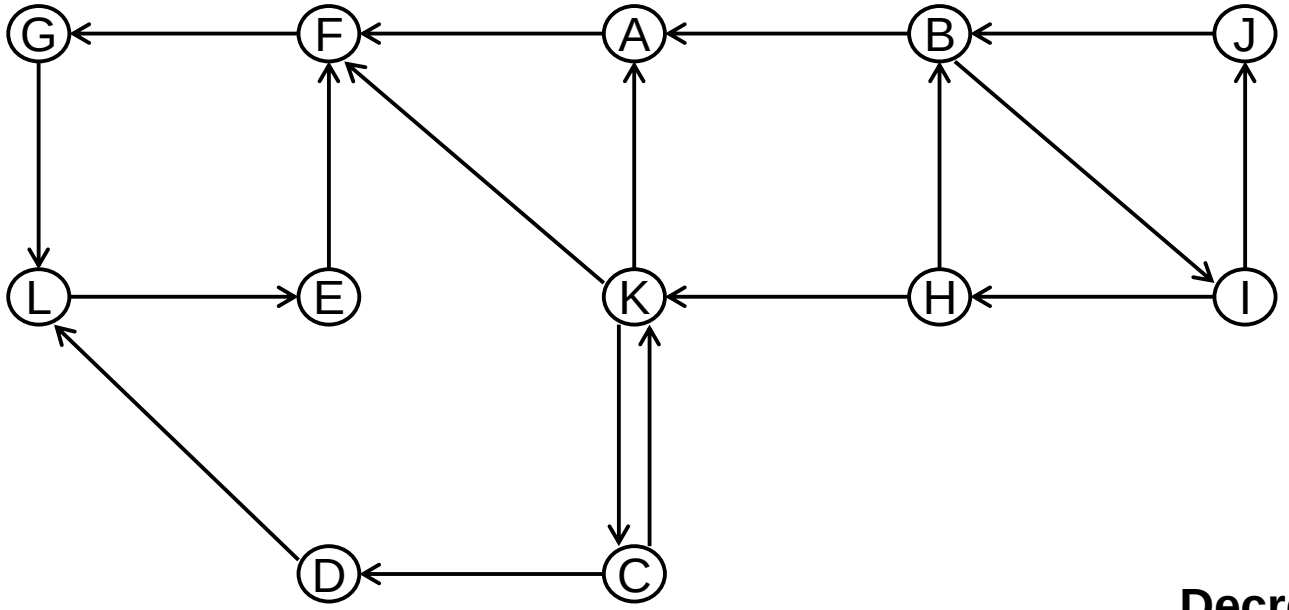
G:


G
F
A
B
J
L
E
K
H
I
D
C

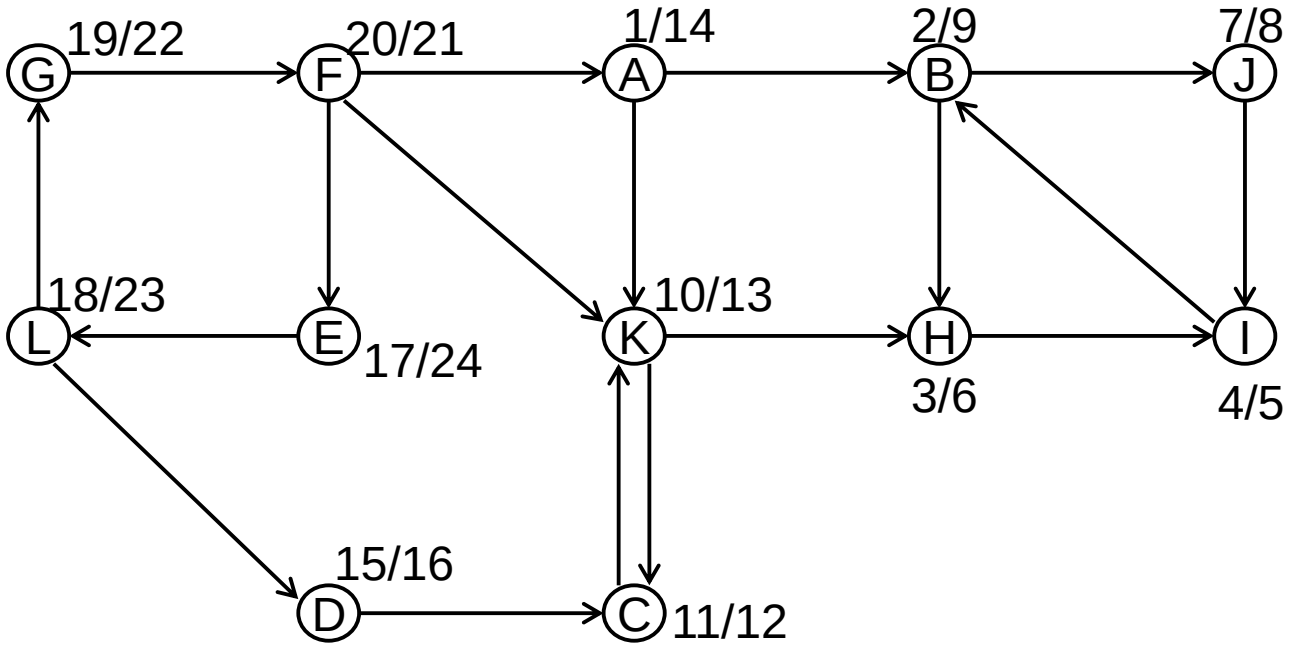
G^R:


19/22
20/21
1/14
2/9
7/8
G
F
A
B
J
18/23
10/13
L
E
17/24
K
H
I
3/6
4/5
15/16
D
C
11/12

G:



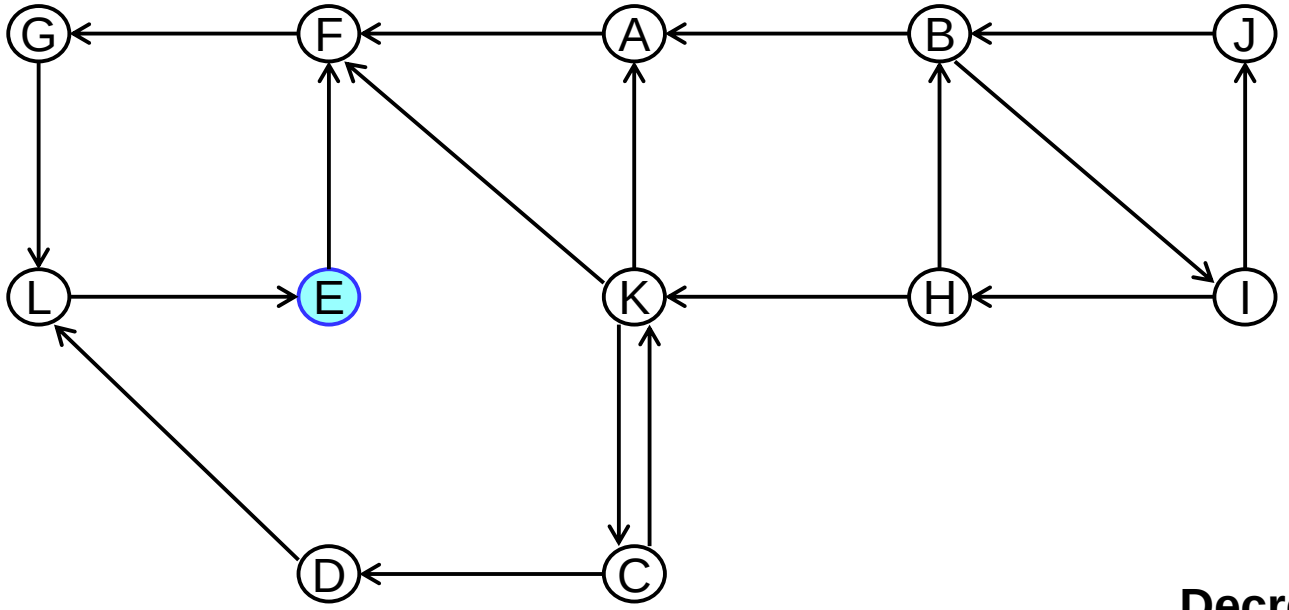
G^R:



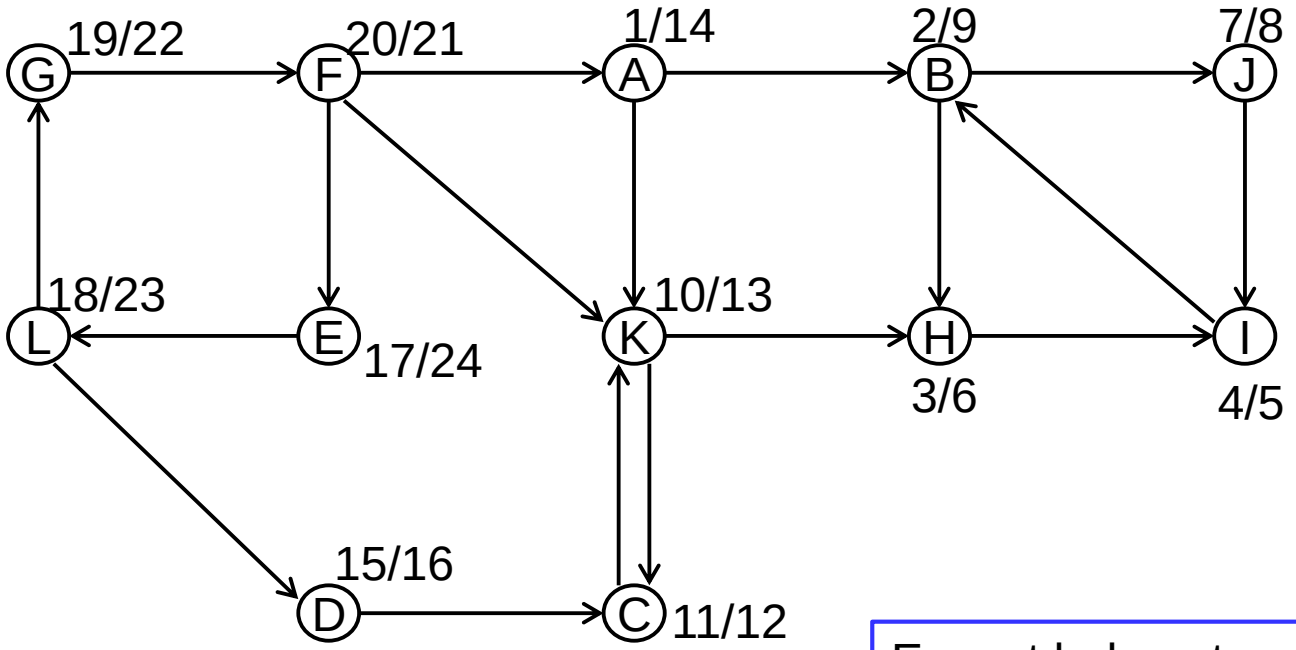
Decreasing post-visit:

E (24)
L (23)
G (22)
F (21)
D (16)
A (14)
K (13)
C (12)
B (9)
J (8)
H (6)
I (5)

G:



G^R:

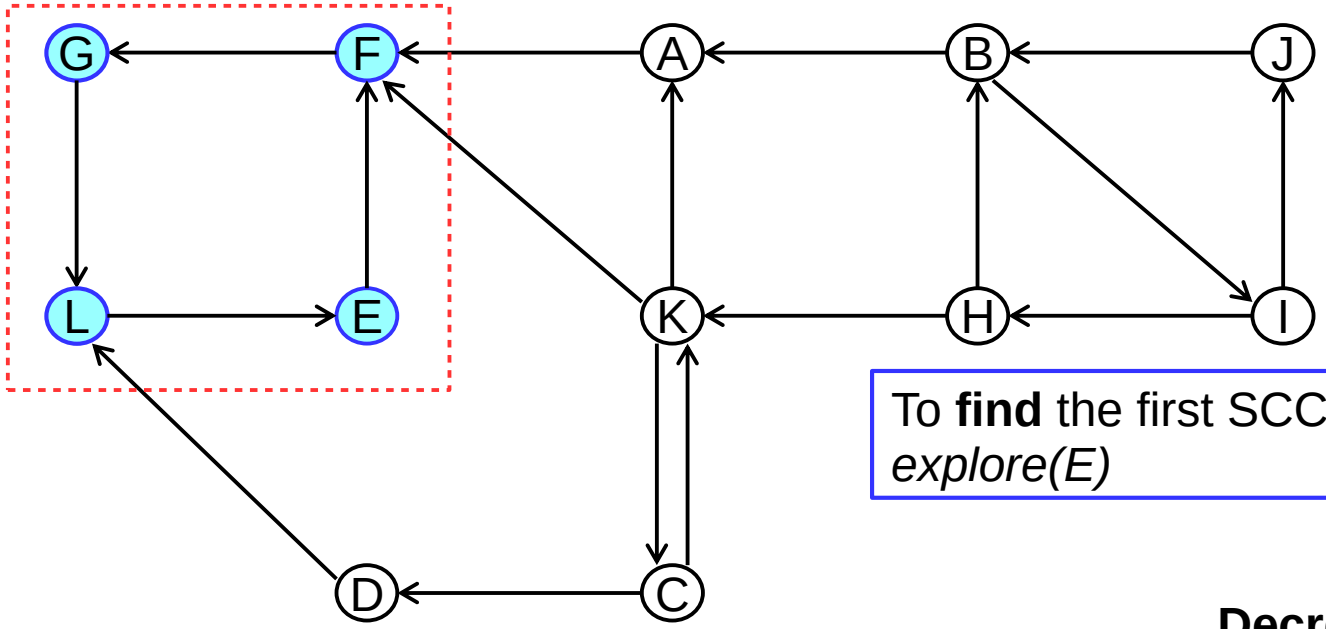


Decreasing post-visit:

E (24)
L (23)
G (22)
F (21)
D (16)
A (14)
K (13)
C (12)
B (9)
J (8)
H (6)
I (5)

E must belong to a Sink SCC of the original graph G

G:

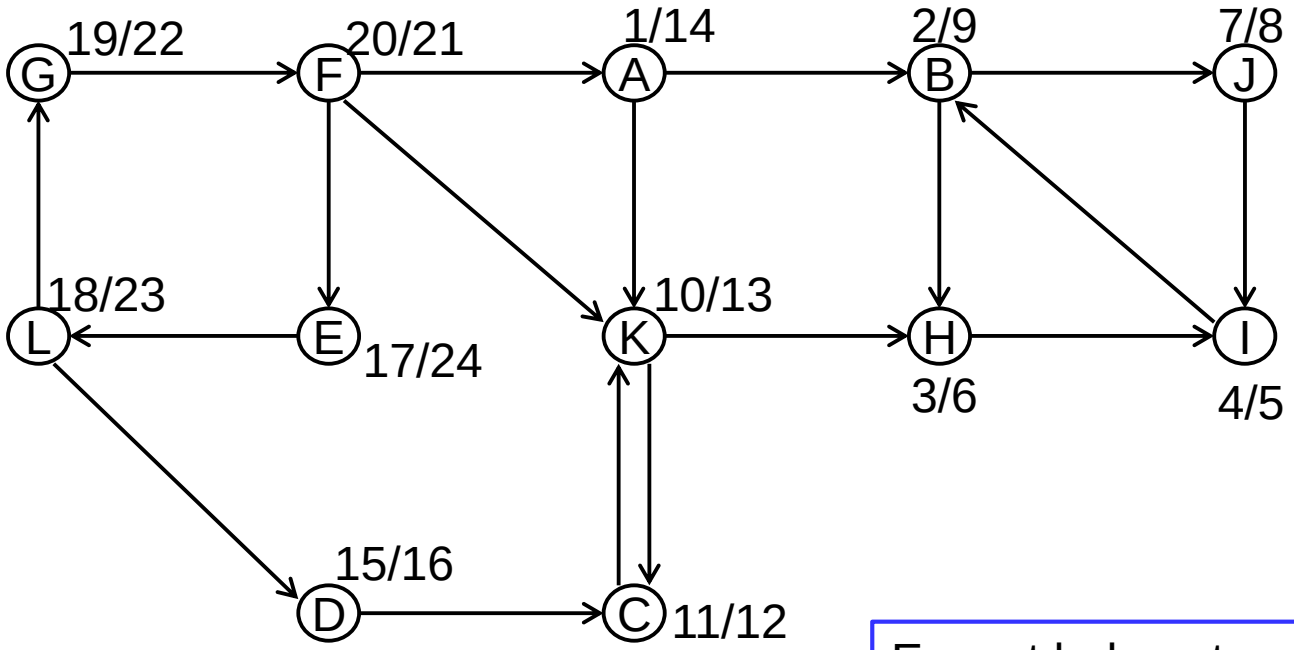


To **find** the first SCC of G we *explore(E)*

Decreasing post-visit:

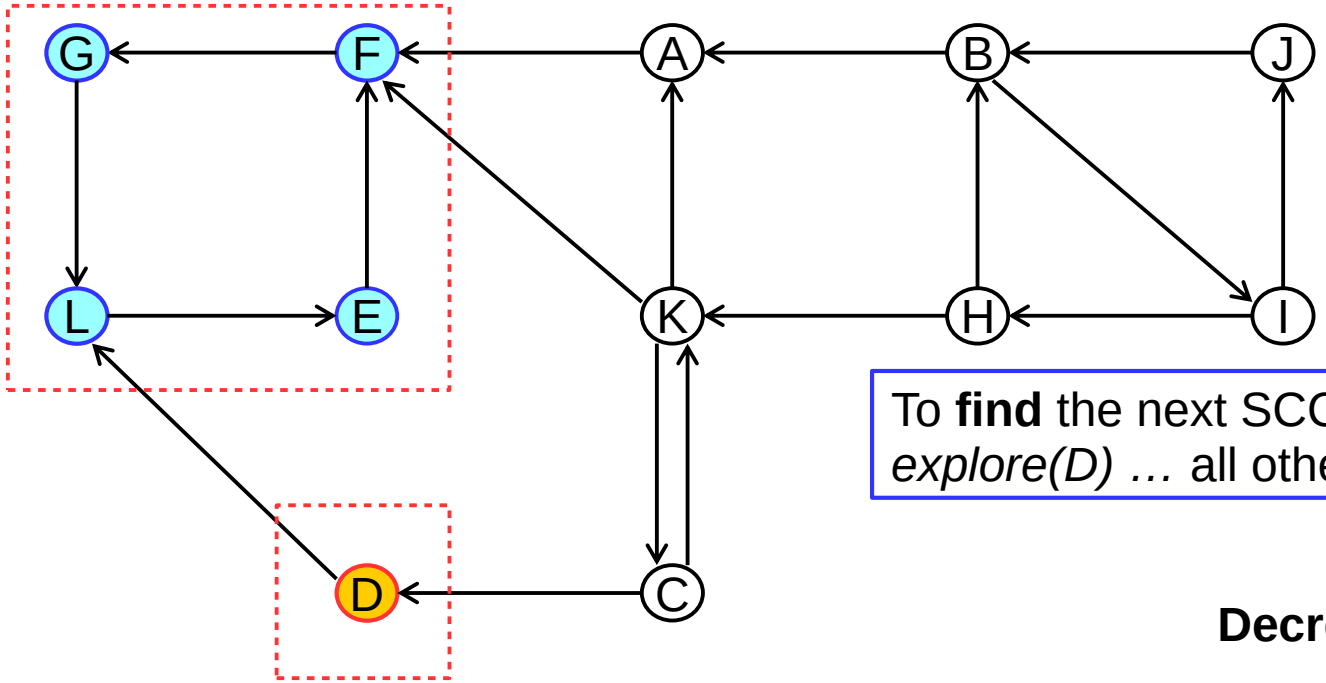
E (24)
L (23)
G (22)
F (21)
D (16)
A (14)
K (13)
C (12)
B (9)
J (8)
H (6)
I (5)

G^R:



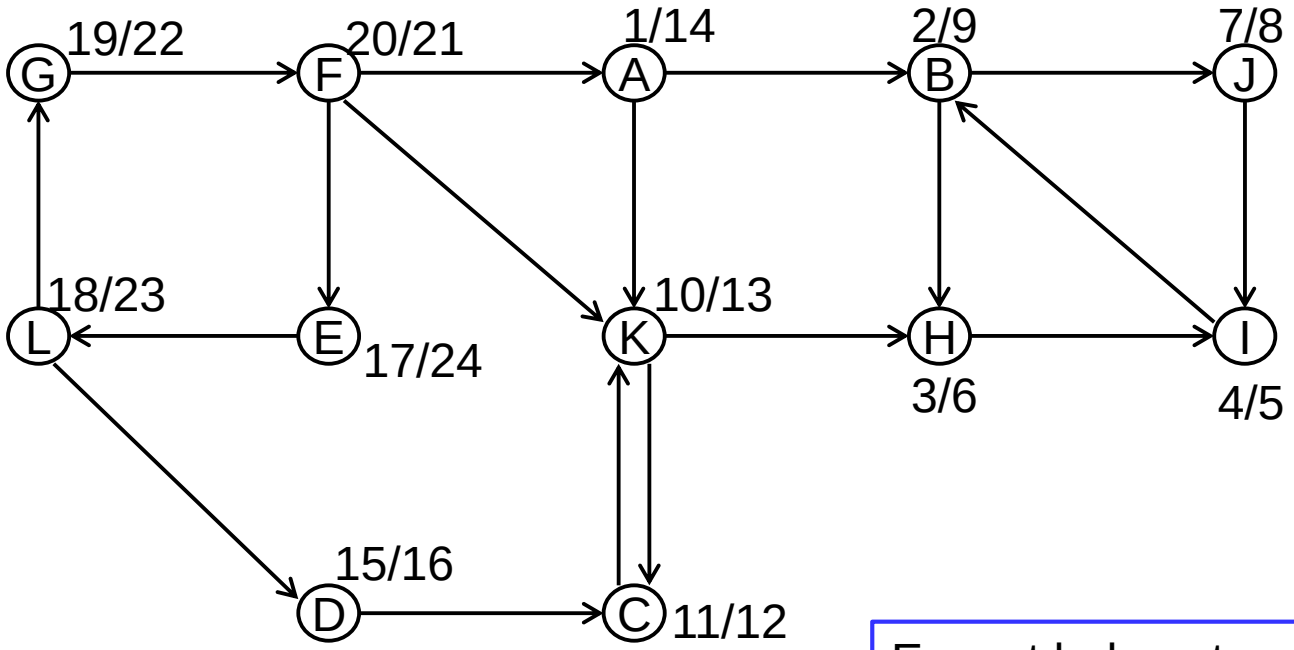
E must belong to a Sink SCC of the original graph G

G:



To **find** the next SCC of G we *explore(D)* … all other are visited

G^R:

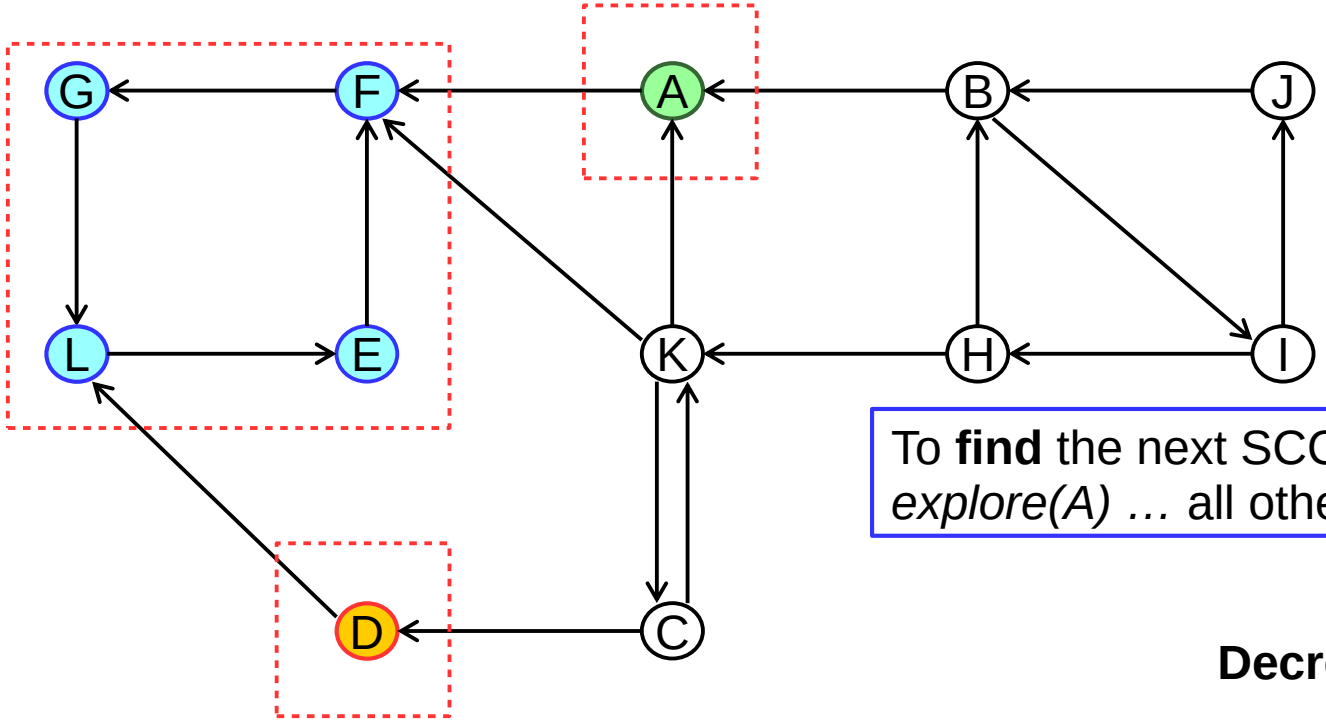


Decreasing post-visit:

E (24) - visited
L (23) - visited
G (22) - visited
F (21) - visited
D (16)
A (14)
K (13)
C (12)
B (9)
J (8)
H (6)
I (5)

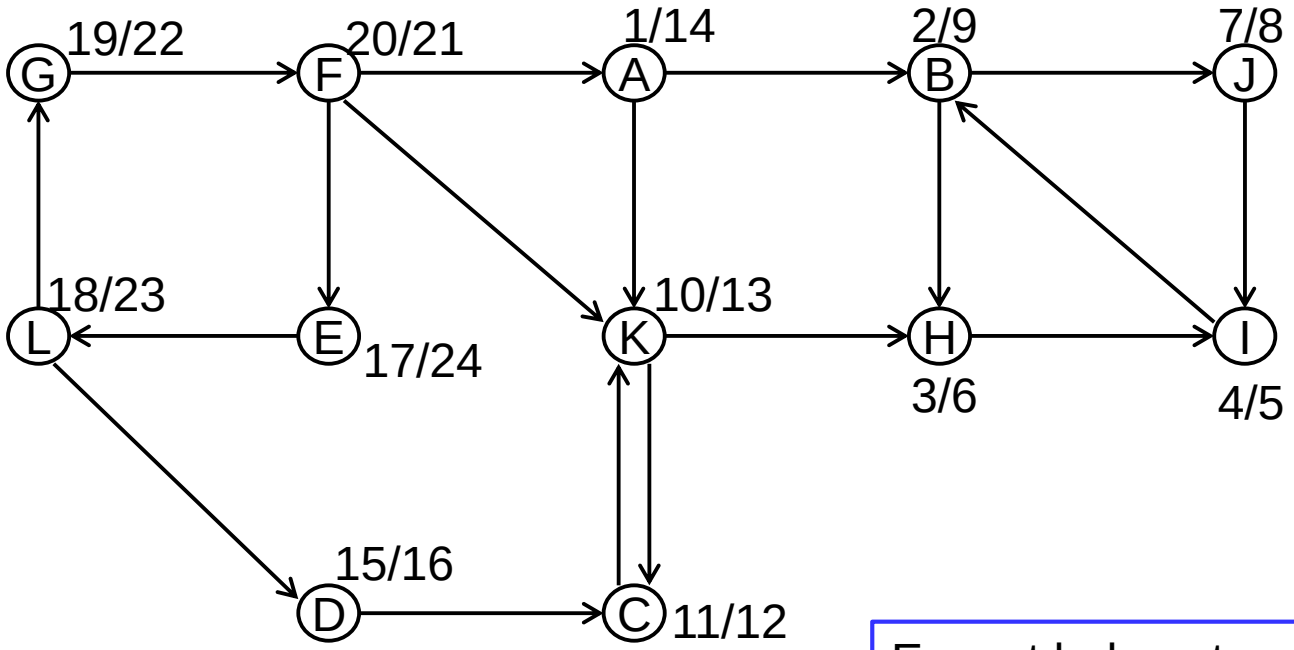
E must belong to a Sink SCC of the original graph G

G:



To **find** the next SCC of G we *explore(A)* … all other are visited

G^R:

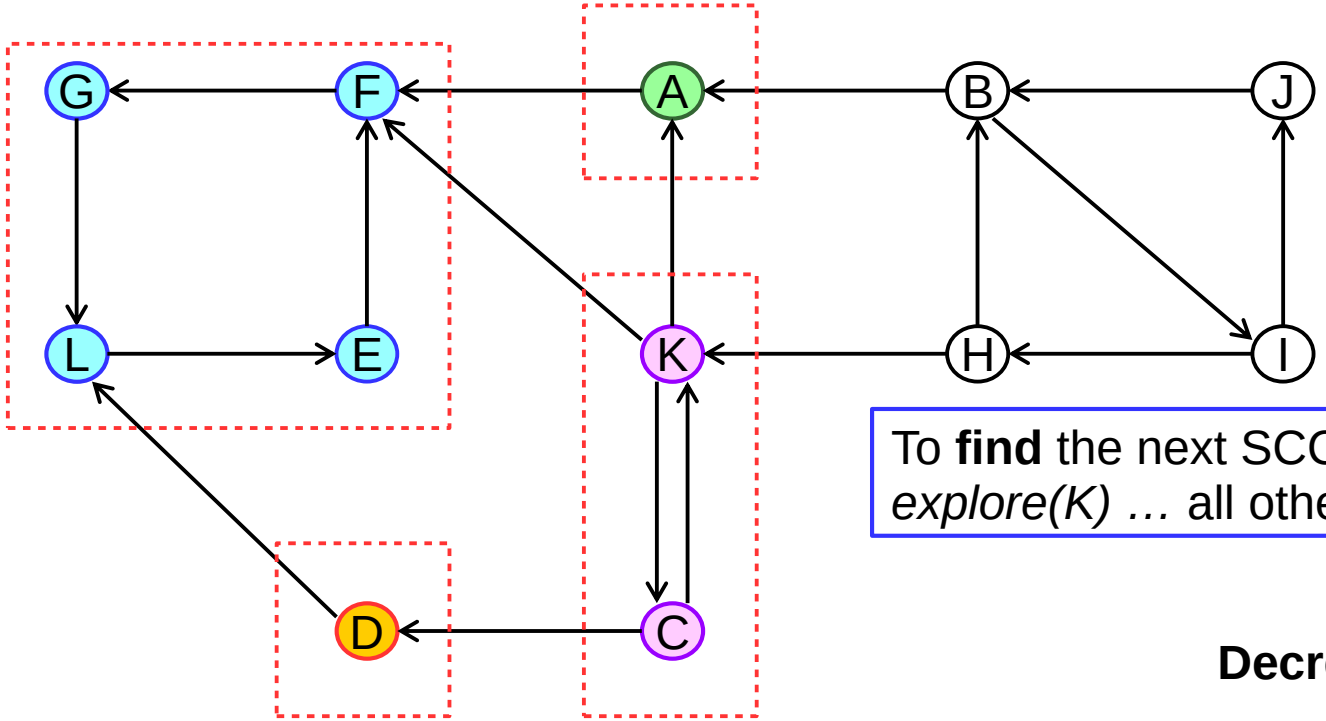


Decreasing post-visit:

E (24) - visited
L (23) - visited
G (22) - visited
F (21) - visited
D (16)- visited
A (14)
K (13)
C (12)
B (9)
J (8)
H (6)
I (5)

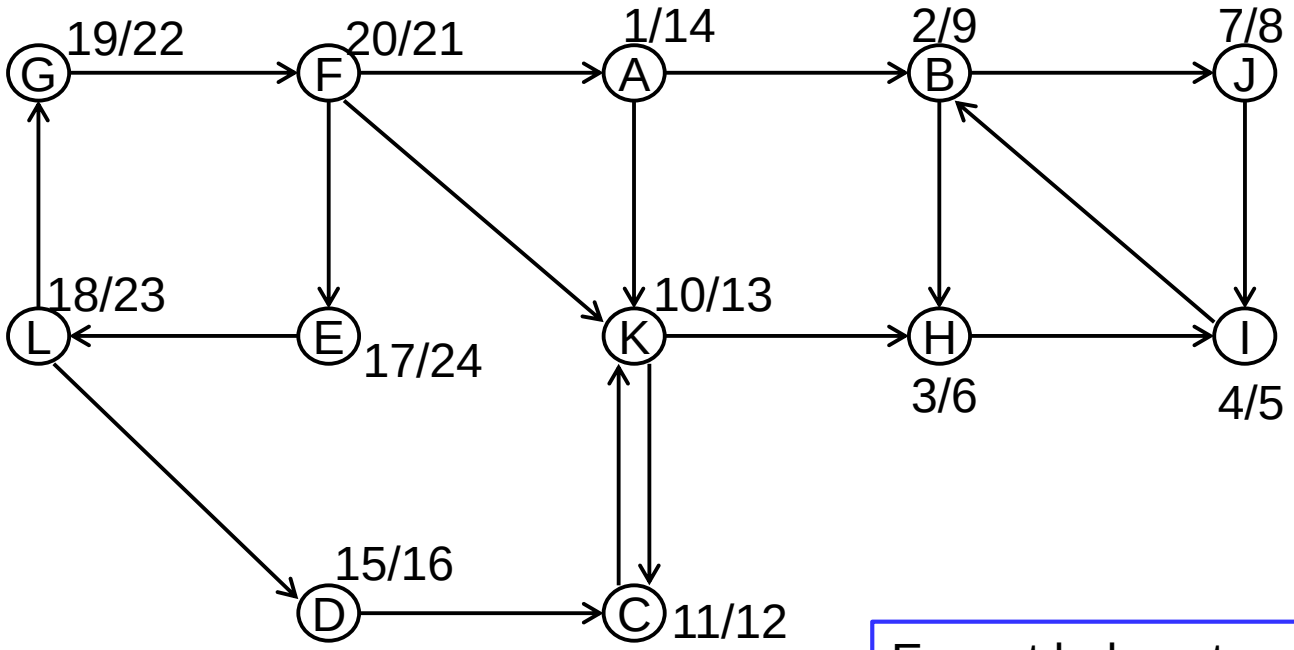
E must belong to a Sink SCC of the original graph G

G:



To **find** the next SCC of G we *explore(K)* … all other are visited

G^R:

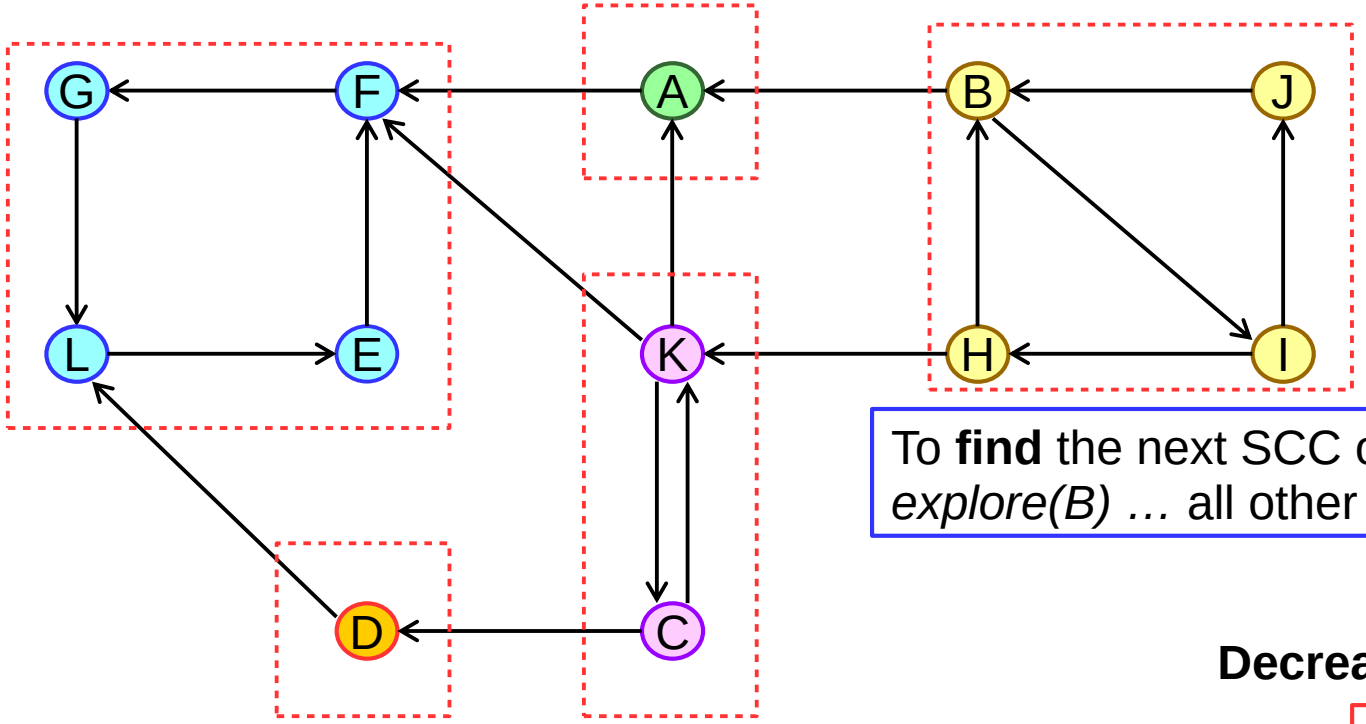


Decreasing post-visit:

E (24) - visited
L (23) - visited
G (22) - visited
F (21) - visited
D (16)- visited
A (14) - visited
K (13) - visited
C (12) - visited
B (9)
J (8)
H (6)
I (5)

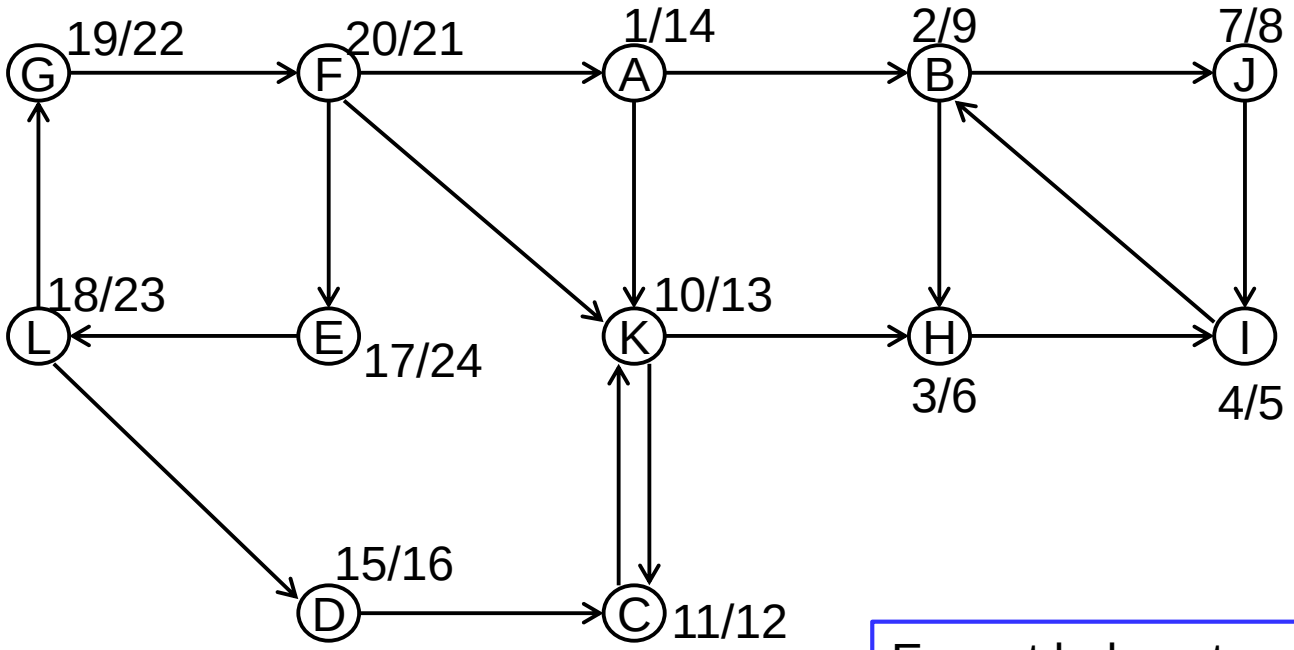
E must belong to a Sink SCC of the original graph G

G:



To **find** the next SCC of G we *explore(B)* … all other are visited

G^R:

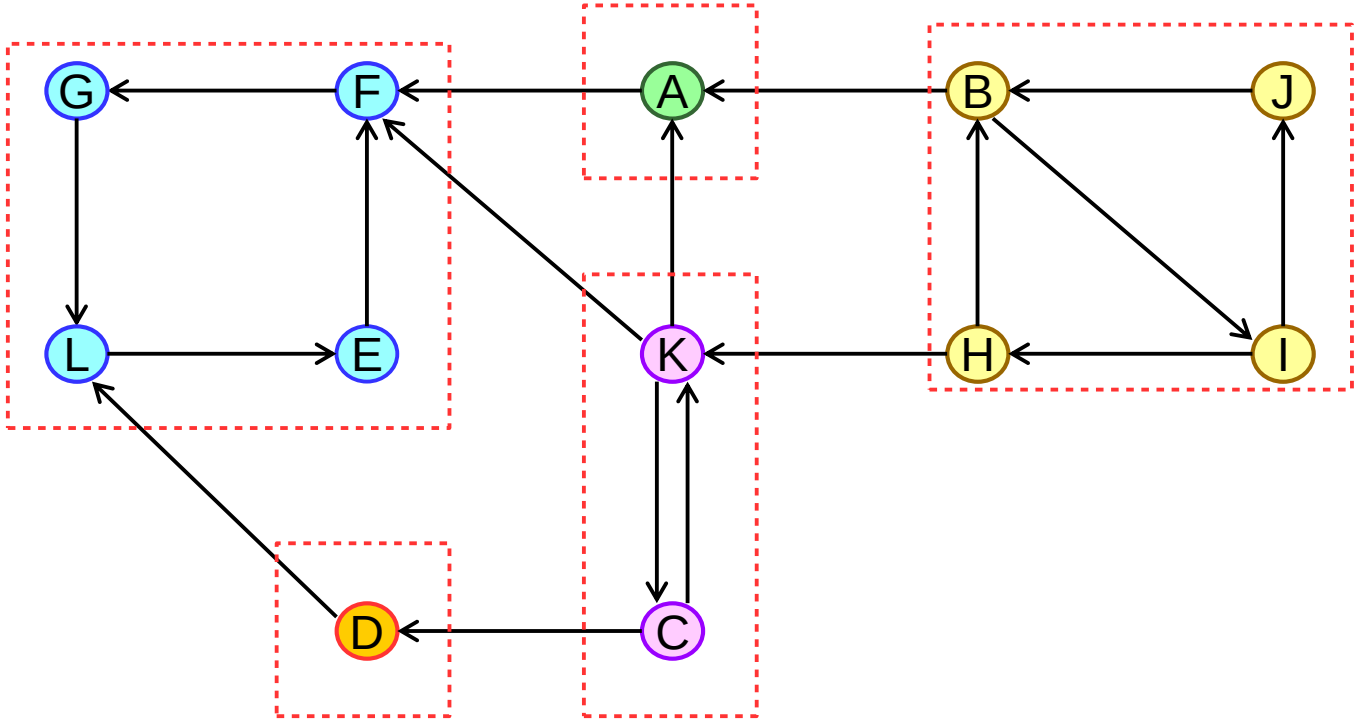


Decreasing post-visit:

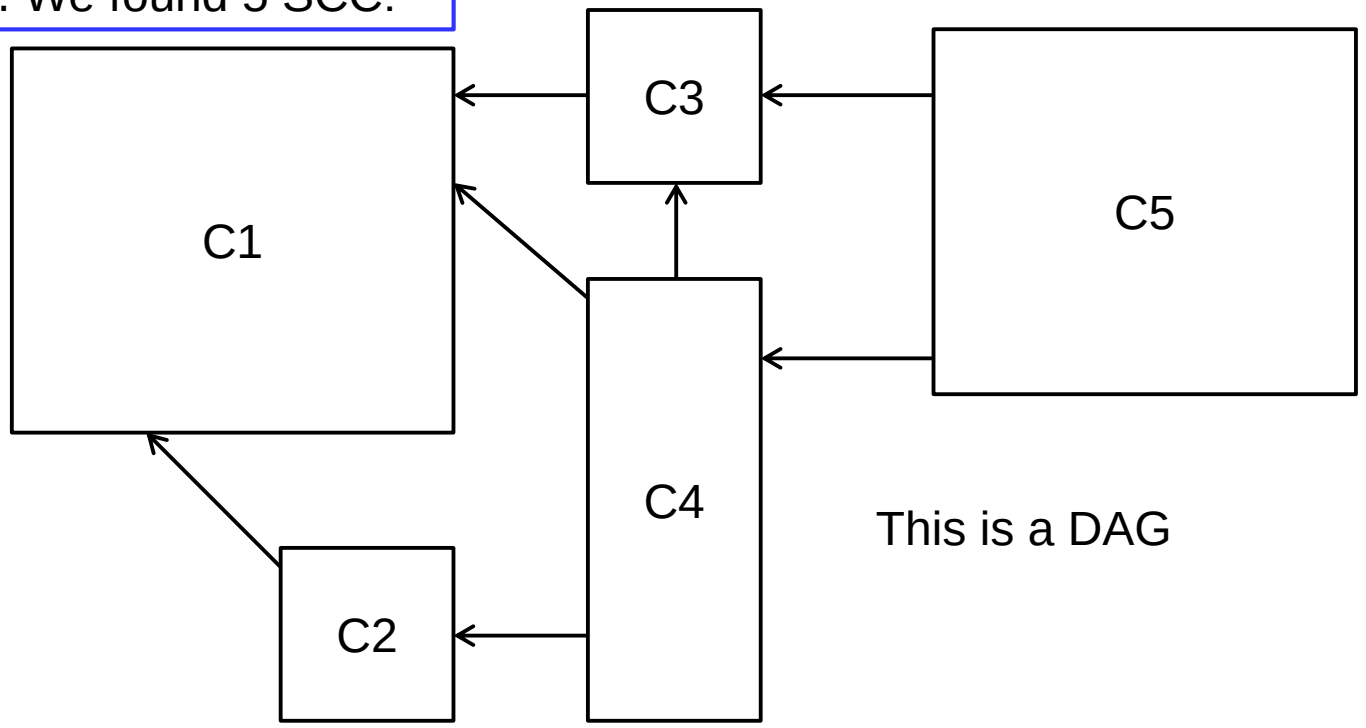
E (24) - visited
L (23) - visited
G (22) - visited
F (21) - visited
D (16)- visited
A (14) - visited
K (13) - visited
C (12) - visited
B (9)
J (8)
H (6)
I (5)

E must belong to a Sink SCC of the original graph G

G:



We are done. We found 5 SCC:



This is a DAG