

Graph Algorithms II

Depth First Search

Depth First Search (DFS)

Depth First Search (DFS)

- BFS : First Discovered – First Explored

Depth First Search (DFS)

- BFS : First Discovered – First Explored
- DFS: Last Discovered – First Explored

Depth First Search (DFS)

- BFS : First Discovered – First Explored
- DFS: Last Discovered – First Explored

The algorithm keeps track of

Depth First Search (DFS)

- BFS : First Discovered – First Explored
- DFS: Last Discovered – First Explored

The algorithm keeps track of

$\text{color}[v]$ = white, grey, or black

Depth First Search (DFS)

- BFS : First Discovered – First Explored
- DFS: Last Discovered – First Explored

The algorithm keeps track of

$\text{color}[v]$ = white, grey, or black

$p[v]$ = u iff u discovered v

Depth First Search (DFS)

- BFS : First Discovered – First Explored
- DFS: Last Discovered – First Explored

The algorithm keeps track of

Same as
BFS

{ color[v] = white, grey, or black
p[v] = u iff u discovered v

Depth First Search (DFS)

- BFS : First Discovered – First Explored
- DFS: Last Discovered – First Explored

The algorithm keeps track of

Same as
BFS

{ color[v] = white, grey, or black

p[v] = u iff u discovered v

Different
from
BFS

Depth First Search (DFS)

- BFS : First Discovered – First Explored
- DFS: Last Discovered – First Explored

The algorithm keeps track of

Same as
BFS

$\text{color}[v]$ = white, grey, or black

$p[v]$ = u iff u discovered v

Different
from
BFS

$d[v]$ = “time” when v is discovered

Depth First Search (DFS)

- BFS : First Discovered – First Explored
- DFS: Last Discovered – First Explored

The algorithm keeps track of

Same as
BFS

$\text{color}[v]$ = white, grey, or black

$p[v]$ = u iff u discovered v

Different
from
BFS

$d[v]$ = “time” when v is discovered

$f[v]$ = “time” when v ’s exploration is completed

Depth First Search (DFS)

- BFS : First Discovered – First Explored
- DFS: Last Discovered – First Explored

The algorithm keeps track of

Same as
BFS

$\text{color}[v]$ = white, grey, or black

$p[v]$ = u iff u discovered v

Different
from
BFS

$d[v]$ = “time” when v is discovered

$f[v]$ = “time” when v ’s exploration is completed

To keep track of “time”, it uses a counter which is incremented by 1

- whenever a new node is discovered
- whenever it finishes exploring a node

DFS(G)

/ G = (V, E) */*

For each $v \in V$ do

 color[v] \leftarrow white ; $d[v] \leftarrow \infty$; $f[v] \leftarrow \infty$; $p[v] \leftarrow \text{NIL}$

End For

color	W	W	W	W	W	...	W	...	W
	1	2	3	4	5	...	v	...	n

DFS(G)

/ G = (V, E) */*

For each $v \in V$ do

$\text{color}[v] \leftarrow \text{white} ; d[v] \leftarrow \infty ; f[v] \leftarrow \infty ; p[v] \leftarrow \text{NIL}$

End For

time \leftarrow 0 /* Global variable */

color	W	W	W	W	W	...	W	...	W
	1	2	3	4	5	...	v	...	n

DFS(G)

/ G = (V, E) */*

For each v ∈ V do

 color[v] ← white ; d[v] ← ∞ ; f[v] ← ∞ ; p[v] ← NIL

End For

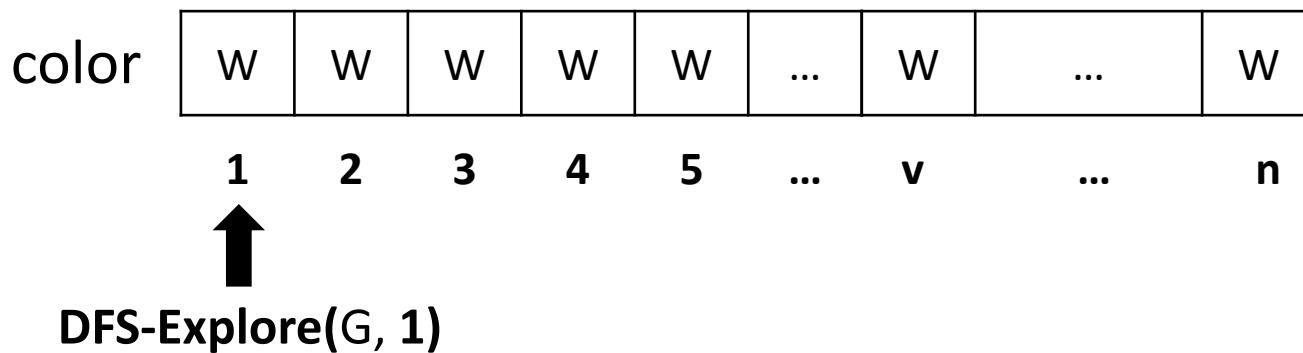
time ← 0 */* Global variable */*

For each v ∈ V do

If color[v] = white **then DFS-Explore(G, v)**

End For

End DFS



DFS(G) /* G = (V, E) */

For each $v \in V$ do

$\text{color}[v] \leftarrow \text{white} ; d[v] \leftarrow \infty ; f[v] \leftarrow \infty ; p[v] \leftarrow \text{NIL}$

End For

time \leftarrow 0 /* Global variable */

For each $v \in V$ do

If $\text{color}[v] = \text{white}$ then **DFS-Explore**(G, v)

End For

End DFS

color	W	W	B	W	W	...	W	...	W
	1	2	3	4	5	...	v	...	n



DFS-Explore(G, 1)

DFS(G) /* G = (V, E) */

For each $v \in V$ do

$\text{color}[v] \leftarrow \text{white} ; d[v] \leftarrow \infty ; f[v] \leftarrow \infty ; p[v] \leftarrow \text{NIL}$

End For

time \leftarrow 0 /* Global variable */

For each $v \in V$ do

If $\text{color}[v] = \text{white}$ then **DFS-Explore**(G, v)

End For

End DFS

color	W	W	B	W	B	...	W	...	W
	1	2	3	4	5	...	v	...	n
									

DFS-Explore(G, 1)

DFS(G) /* G = (V, E) */

For each $v \in V$ do

$\text{color}[v] \leftarrow \text{white} ; d[v] \leftarrow \infty ; f[v] \leftarrow \infty ; p[v] \leftarrow \text{NIL}$

End For

time \leftarrow 0 /* Global variable */

For each $v \in V$ do

If $\text{color}[v] = \text{white}$ then **DFS-Explore**(G, v)

End For

End DFS

color		W	B	B	W	B	...	W	...	W
		1	2	3	4	5	...	v	...	n

↑

DFS-Explore($G, 1$)

DFS(G)

/ G = (V, E) */*

For each v ∈ V do

 color[v] ← white ; d[v] ← ∞ ; f[v] ← ∞ ; p[v] ← NIL

End For

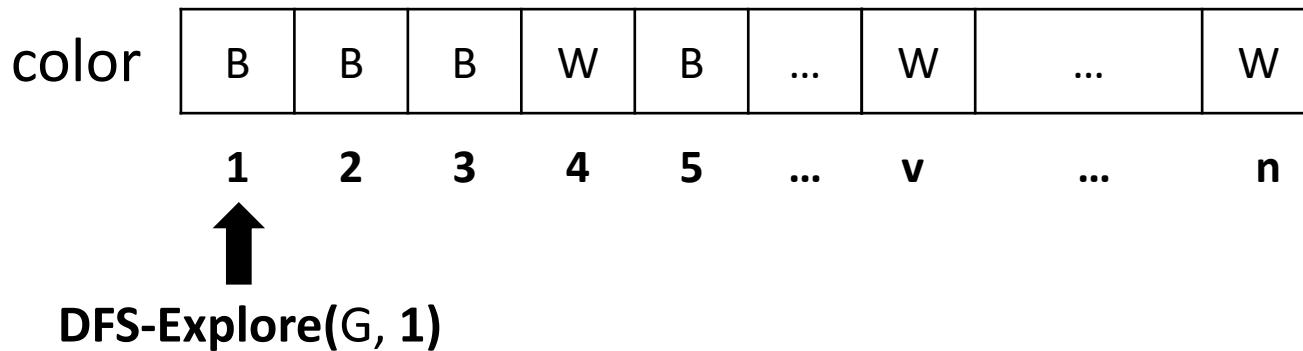
time ← 0 */* Global variable */*

For each v ∈ V do

If color[v] = white **then DFS-Explore(G, v)**

End For

End DFS



DFS(G)

/ G = (V, E) */*

For each v ∈ V do

 color[v] ← white ; d[v] ← ∞ ; f[v] ← ∞ ; p[v] ← NIL

End For

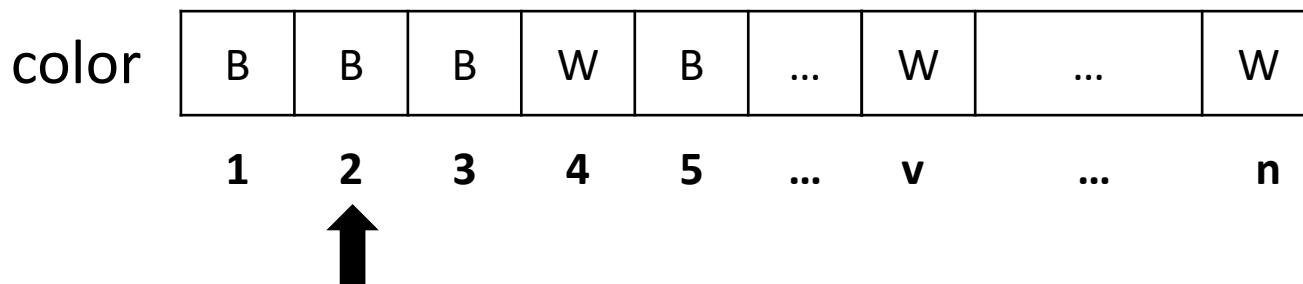
time ← 0 */* Global variable */*

For each v ∈ V do

If color[v] = white **then DFS-Explore(G, v)**

End For

End DFS



DFS(G)

/ G = (V, E) */*

For each $v \in V$ do

$\text{color}[v] \leftarrow \text{white} ; d[v] \leftarrow \infty ; f[v] \leftarrow \infty ; p[v] \leftarrow \text{NIL}$

End For

time \leftarrow 0 /* Global variable */

For each $v \in V$ do

If $\text{color}[v] = \text{white}$ then **DFS-Explore**(G, v)

End For

End DFS

color	B	B	B	W	B	...	W	...	W	
	1	2	3	4	5	...	v	...	n	
										

DFS(G)

/ G = (V, E) */*

For each $v \in V$ **do**

 color[v] \leftarrow white ; $d[v] \leftarrow \infty$; $f[v] \leftarrow \infty$; $p[v] \leftarrow \text{NIL}$

End For

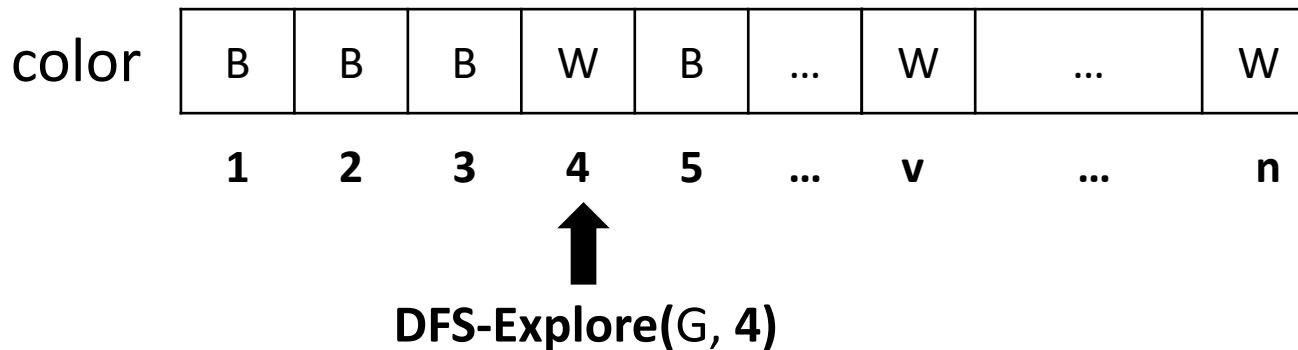
time $\leftarrow 0$ */* Global variable */*

For each $v \in V$ **do**

If color[v] = white **then** **DFS-Explore**(G, v)

End For

End DFS



DFS-Explore(G, u)

/* Explore node u */

DFS-Explore(G, u)

color[u] \leftarrow grey

/* Explore node u */

/* u was just discovered */

```
DFS-Explore(G, u)          /* Explore node u           */
    color[u] ← grey          /* u was just discovered   */
    time ← time + 1 ; d[u] ← time /* u's discovery time      */
```

```
DFS-Explore(G, u)          /* Explore node u      */
    color[u] ← grey        /* u was just discovered */
    time ← time + 1 ; d[u] ← time   /* u's discovery time */
    For each edge (u, v) ∈ E do    /* Explore edge (u,v) */
```

```
DFS-Explore(G, u)          /* Explore node u */  
    color[u] ← grey           /* u was just discovered */  
    time ← time + 1 ; d[u] ← time /* u's discovery time */  
For each edge (u, v) ∈ E do /* Explore edge (u,v) */  
    If color[v] = white then do /* u discovered v */
```

```
DFS-Explore(G, u)          /* Explore node u */  
    color[u] ← grey           /* u was just discovered */  
    time ← time + 1 ; d[u] ← time /* u's discovery time */  
For each edge (u, v) ∈ E do /* Explore edge (u,v) */  
    If color[v] = white then do /* u discovered v */  
        p[v] ← u
```

```
DFS-Explore(G, u)          /* Explore node u */  
    color[u] ← grey           /* u was just discovered */  
    time ← time + 1 ; d[u] ← time /* u's discovery time */  
    For each edge (u, v) ∈ E do /* Explore edge (u,v) */  
        If color[v] = white then do /* u discovered v */  
            p[v] ← u  
            DFS-Explore(G, v)      /* Explore v */
```

```
DFS-Explore(G, u)          /* Explore node u */  
    color[u] ← grey           /* u was just discovered */  
    time ← time + 1 ; d[u] ← time /* u's discovery time */  
    For each edge (u, v) ∈ E do /* Explore edge (u,v) */  
        If color[v] = white then do /* u discovered v */  
            p[v] ← u  
            DFS-Explore(G, v)      /* Explore v */  
        End If  
    End For
```

```
DFS-Explore(G, u)          /* Explore node u */  
    color[u] ← grey           /* u was just discovered */  
    time ← time + 1 ; d[u] ← time /* u's discovery time */  
    For each edge (u, v) ∈ E do /* Explore edge (u,v) */  
        If color[v] = white then do /* u discovered v */  
            p[v] ← u  
            DFS-Explore(G, v)      /* Explore v */  
        End If  
    End For  
    color[u] ← black          /* u's exploration is over */
```

```
DFS-Explore(G, u)                                /* Explore node u      */
    color[u] ← grey                               /* u was just discovered */
    time ← time + 1 ; d[u] ← time                /* u's discovery time */
    For each edge (u, v) ∈ E do                  /* Explore edge (u,v) */
        If color[v] = white then do              /* u discovered v      */
            p[v] ← u
            DFS-Explore(G, v)                      /* Explore v           */
        End If
    End For
    color[u] ← black                            /* u's exploration is over */
    time ← time + 1 ; f[u] ← time                /* time u's exploration is over */
```

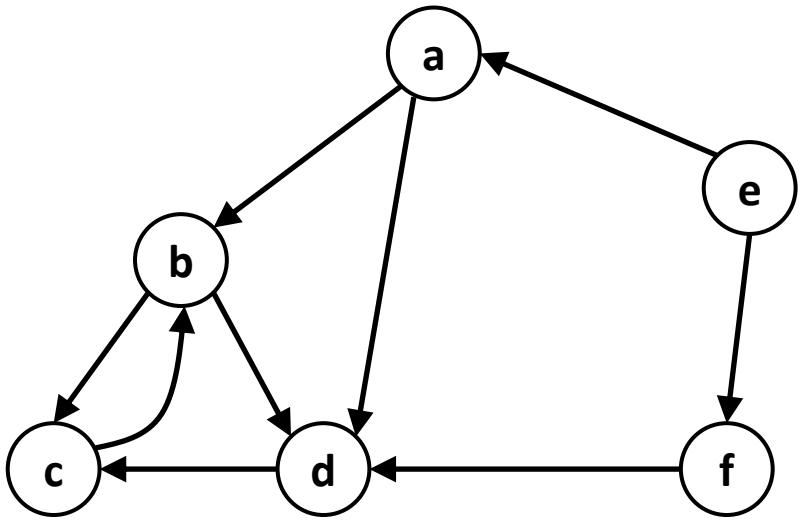
```
DFS-Explore(G, u)                                /* Explore node u      */
    color[u] ← grey                            /* u was just discovered */
    time ← time + 1 ; d[u] ← time             /* u's discovery time */
    For each edge (u, v) ∈ E do           /* Explore edge (u,v) */
        If color[v] = white then do          /* u discovered v */
            p[v] ← u
            DFS-Explore(G, v)                /* Explore v */
        End If
    End For
    color[u] ← black                          /* u's exploration is over */
    time ← time + 1 ; f[u] ← time            /* time u's exploration is over */

End DFS-Explore
```

```
DFS-Explore(G, u)                                /* Explore node u      */
    color[u] ← grey                            /* u was just discovered */
    time ← time + 1 ; d[u] ← time             /* u's discovery time */
    For each edge (u, v) ∈ E do           /* Explore edge (u,v) */
        If color[v] = white then do          /* u discovered v */
            p[v] ← u
            DFS-Explore(G, v)                /* Explore v */
        End If
    End For
    color[u] ← black                          /* u's exploration is over */
    time ← time + 1 ; f[u] ← time            /* time u's exploration is over */
```

End DFS-Explore

DFS(G)



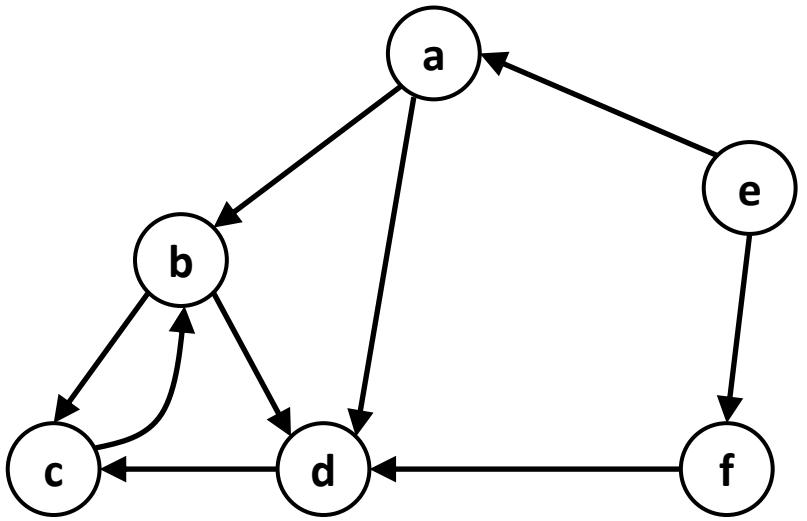
Adj List of G :

a	→	b	→	d
b	→	c	→	d
c	→	b		
d	→	c		
e	→	a	→	f
f	→	d		

Colors of nodes:

W	W	W	W	W	W
a	b	c	d	e	f

DFS(G)



Adj List of G :

a	→	b	→	d
b	→	c	→	d
c	→	b		
d	→	c		
e	→	a	→	f
f	→	d		

Colors of nodes:

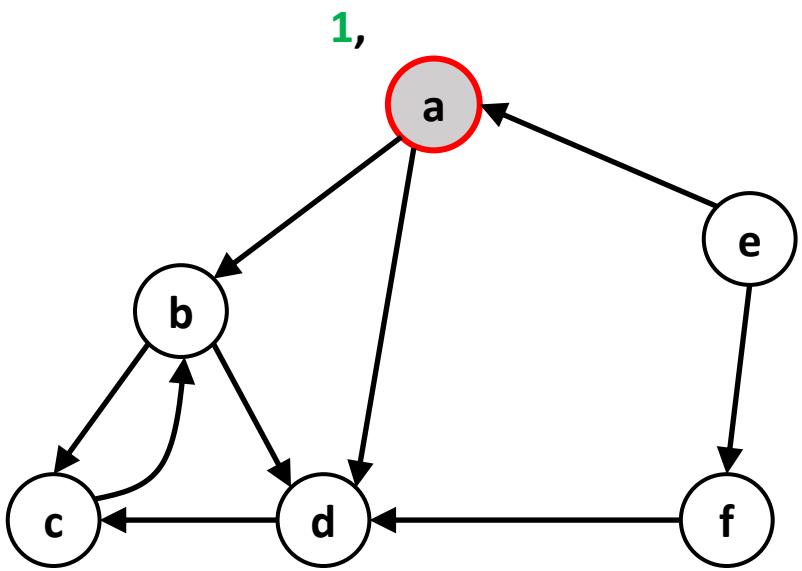
W	W	W	W	W	W
---	---	---	---	---	---

a b c d e f



DFS-Explore(G, a)

DFS(G)



Adj List of G :

a	→	b	→	d
b	→	c	→	d
c	→	b		
d	→	c		
e	→	a	→	f
f	→	d		

Colors of nodes:

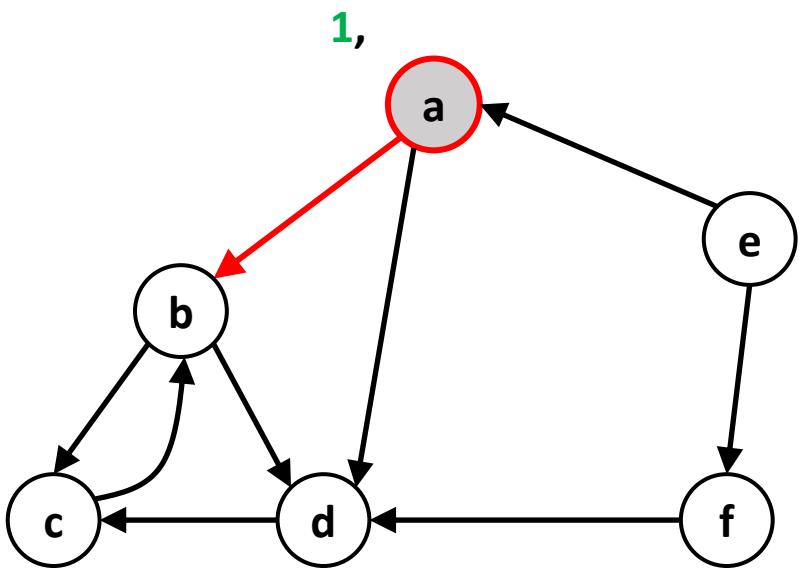
G	W	W	W	W	W
---	---	---	---	---	---

a b c d e f



DFS-Explore(G, a)

DFS(G)



Adj List of G :

a	→	b	→	d
b	→	c	→	d
c	→	b		
d	→	c		
e	→	a	→	f
f	→	d		

Colors of nodes:

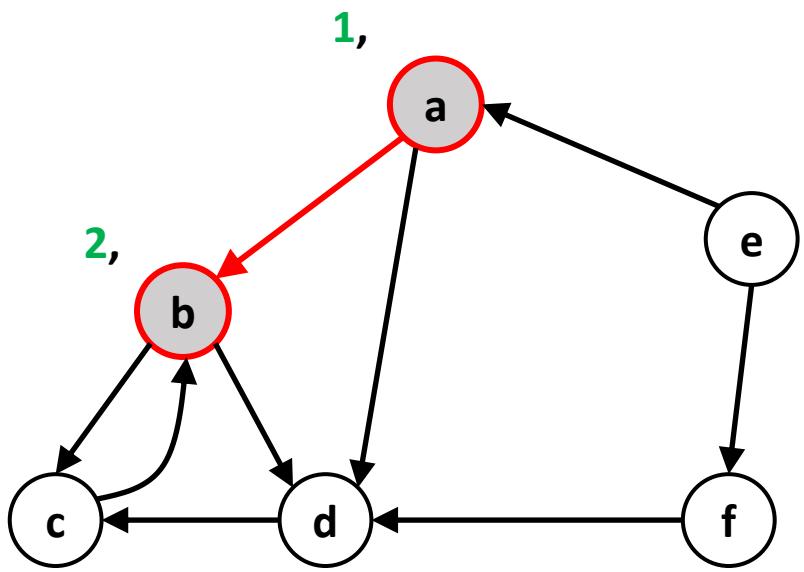
G	W	W	W	W	W
---	---	---	---	---	---

a b c d e f



DFS-Explore(G, a)

DFS(G)



Adj List of G :

a	→	b	→	d
b	→	c	→	d
c	→	b		
d	→	c		
e	→	a	→	f
f	→	d		

Colors of nodes:

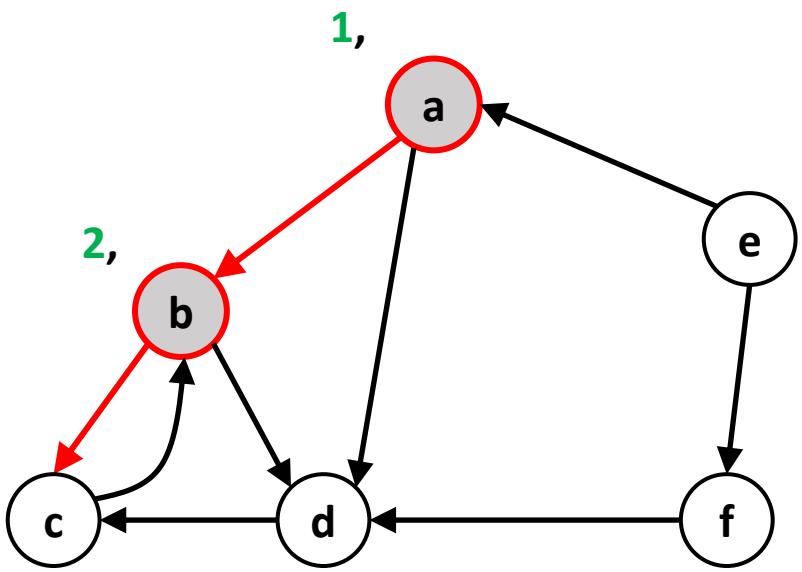
G	G	W	W	W	W
---	---	---	---	---	---

a b c d e f



DFS-Explore(G , a)

DFS(G)



Adj List of G :

a	→	b	→	d
b	→	c	→	d
c	→	b		
d	→	c		
e	→	a	→	f
f	→	d		

Colors of nodes:

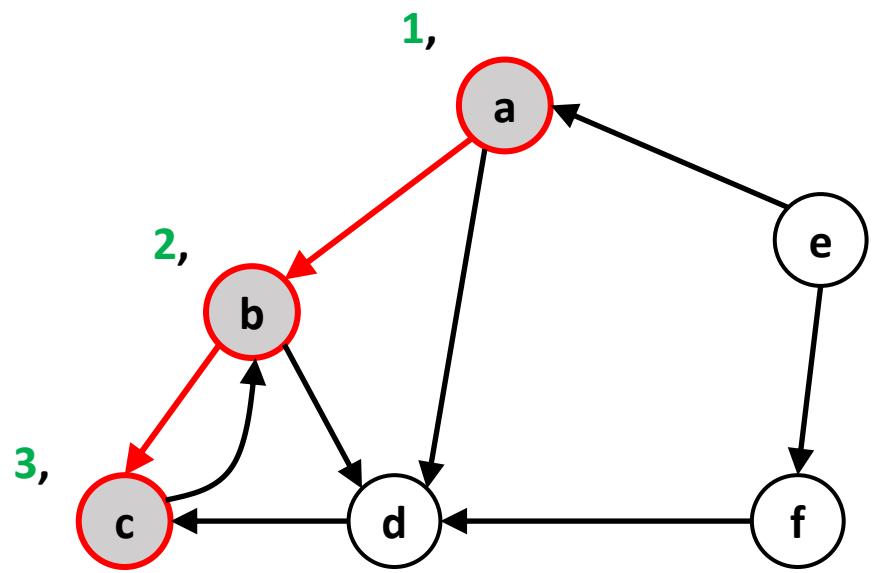
G	G	W	W	W	W
---	---	---	---	---	---

a b c d e f



DFS-Explore(G , a)

DFS(G)



Adj List of G :

a	→	b	→	d
b	→	c	→	d
c	→	b		
d	→	c		
e	→	a	→	f
f	→	d		

Colors of nodes:

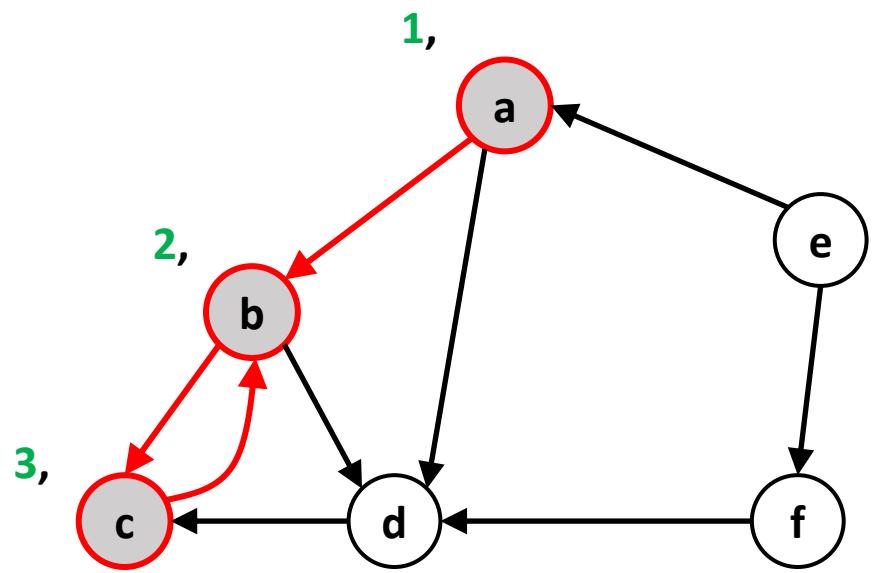
G	G	G	W	W	W
---	---	---	---	---	---

a b c d e f



DFS-Explore(G, a)

DFS(G)



Adj List of G :

a	→	b	→	d
b	→	c	→	d
c	→	b		
d	→	c		
e	→	a	→	f
f	→	d		

Colors of nodes:

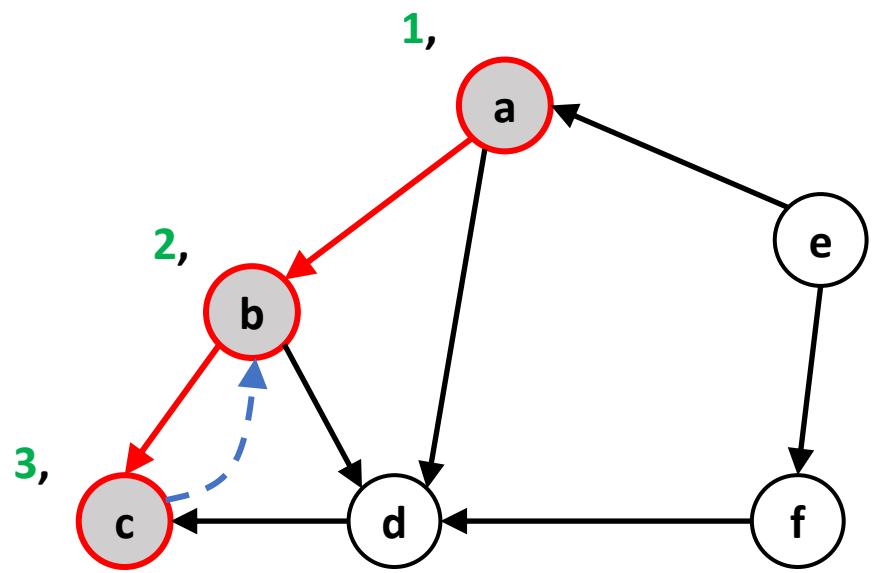
G	G	G	W	W	W
---	---	---	---	---	---

a b c d e f



DFS-Explore(G, a)

DFS(G)



Adj List of G :

a	→	b	→	d
b	→	c	→	d
c	→	b		
d	→	c		
e	→	a	→	f
f	→	d		

Colors of nodes:

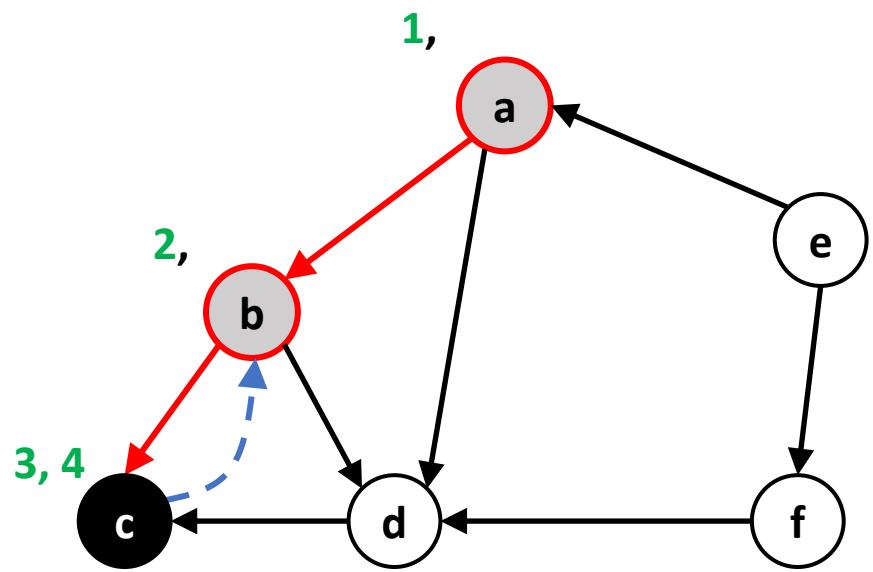
G	G	G	W	W	W
---	---	---	---	---	---

a b c d e f



DFS-Explore(G, a)

DFS(G)



Adj List of G :

a	→	b	→	d
b	→	c	→	d
c	→	b		
d	→	c		
e	→	a	→	f
f	→	d		



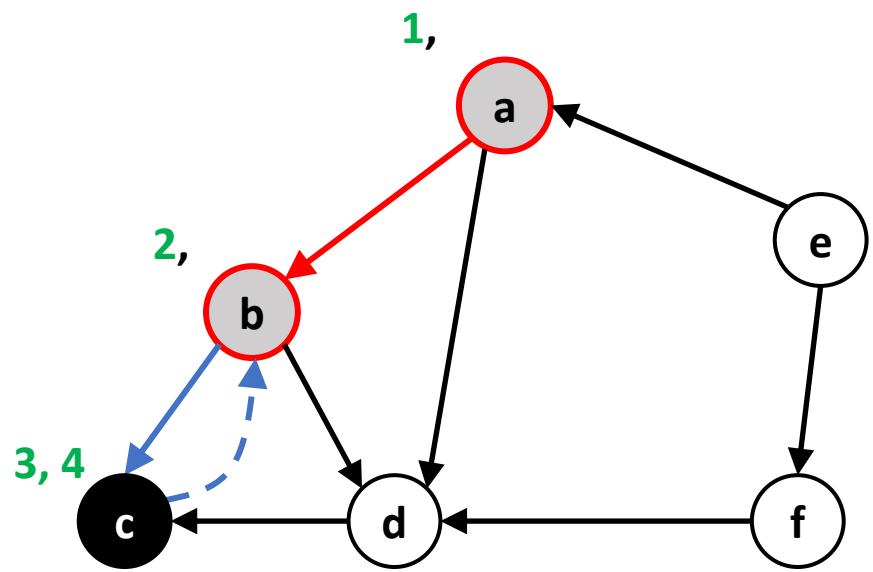
Colors of nodes:

G	G	B	W	W	W
a	b	c	d	e	f



DFS-Explore(G, a)

DFS(G)



Adj List of G :

a	→	b	→	d
b	→	c	→	d
c	→	b		
d	→	c		
e	→	a	→	f
f	→	d		

Colors of nodes:

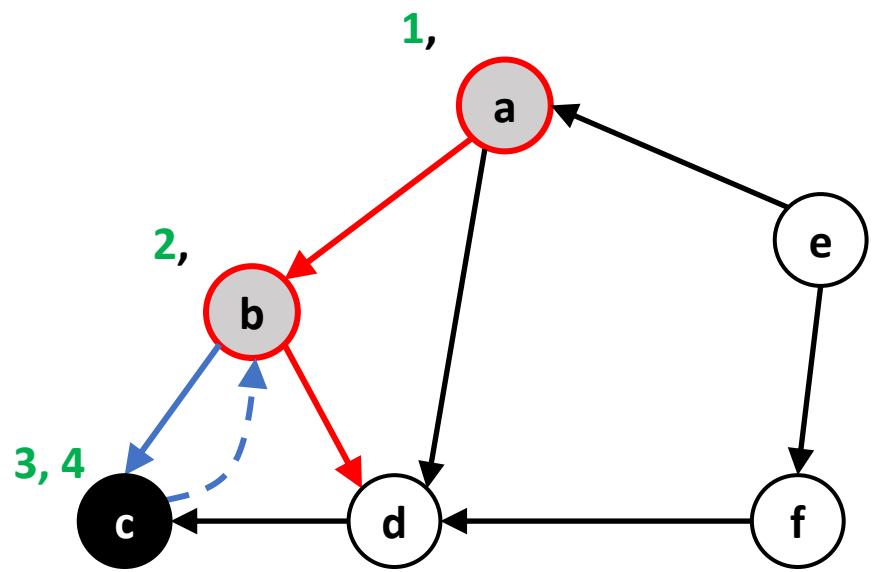
G	G	B	W	W	W
---	---	---	---	---	---

a b c d e f



DFS-Explore(G , a)

DFS(G)



Adj List of G :

a	→	b	→	d
b	→	c	→	d
c	→	b		
d	→	c		
e	→	a	→	f
f	→	d		

Colors of nodes:

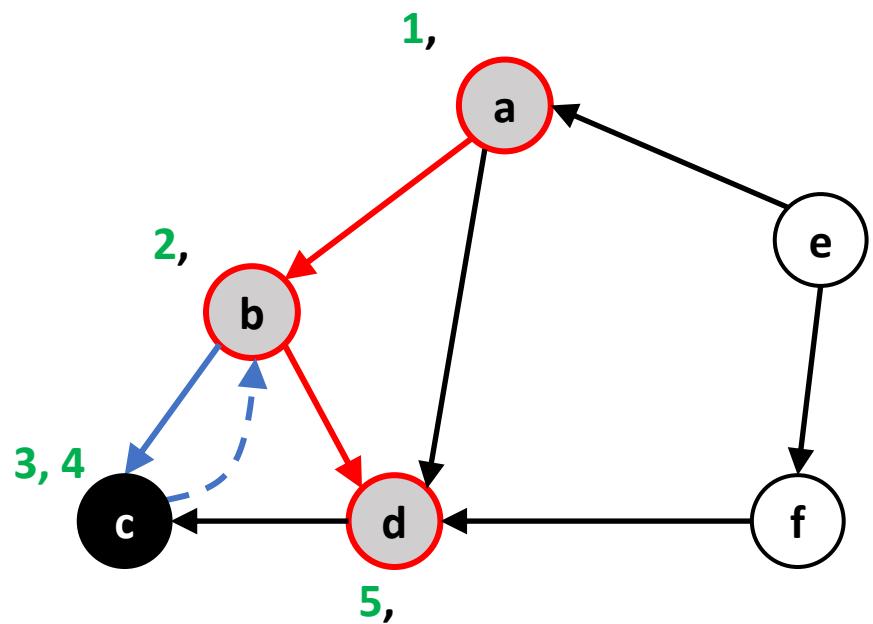
G	G	B	W	W	W
---	---	---	---	---	---

a b c d e f



DFS-Explore(G , a)

DFS(G)



Adj List of G :

a	→	b	→	d
b	→	c	→	d
c	→	b		
d	→	c		
e	→	a	→	f
f	→	d		

Colors of nodes:

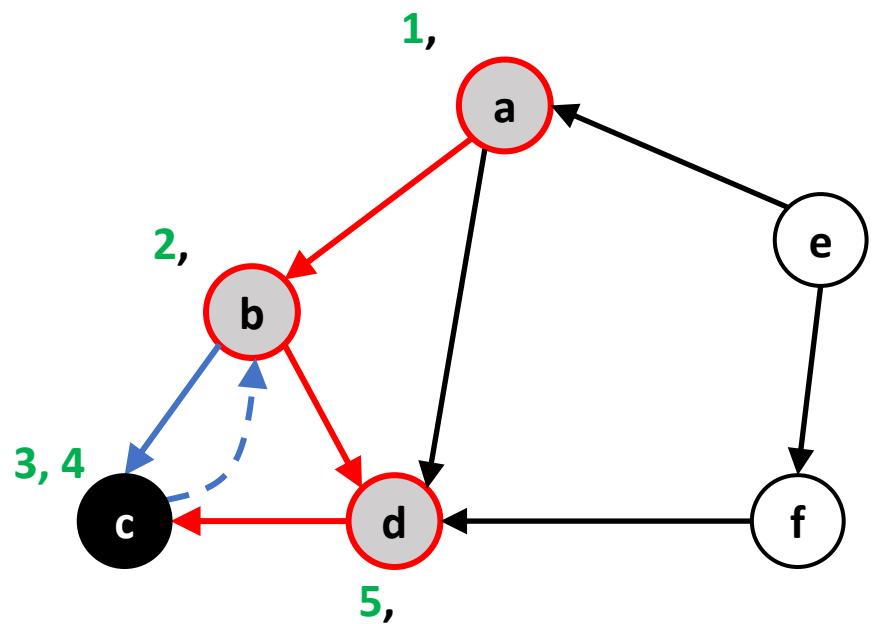
G	G	B	G	W	W
---	---	---	---	---	---

a b c d e f



DFS-Explore(G, a)

DFS(G)



Adj List of G :

a	→	b	→	d
b	→	c	→	d
c	→	b		
d	→	c		
e	→	a	→	f
f	→	d		

Colors of nodes:

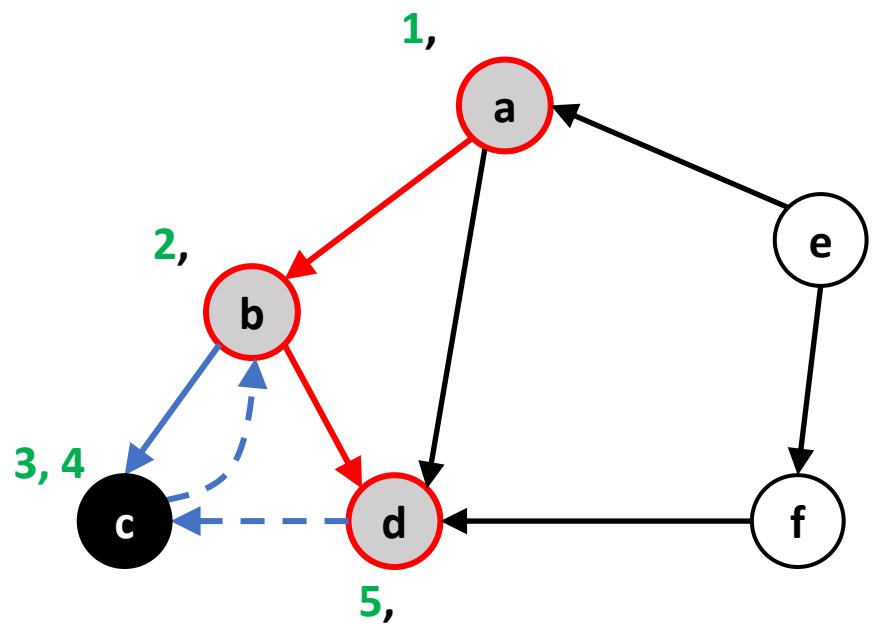
G	G	B	G	W	W
---	---	---	---	---	---

a b c d e f



DFS-Explore(G, a)

DFS(G)



Adj List of G :

a	→	b	→	d
b	→	c	→	d
c	→	b		
d	→	c		
e	→	a	→	f
f	→	d		

Colors of nodes:

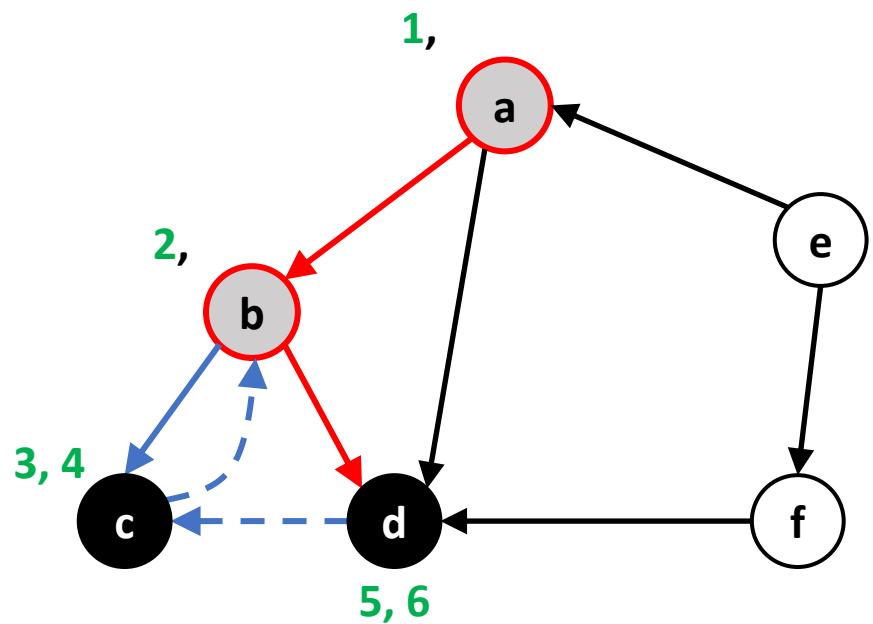
G	G	B	G	W	W
---	---	---	---	---	---

a b c d e f



DFS-Explore(G, a)

DFS(G)



Adj List of G :

a	→	b	→	d
b	→	c	→	d
c	→	b		
d	→	c		
e	→	a	→	f
f	→	d		

Colors of nodes:

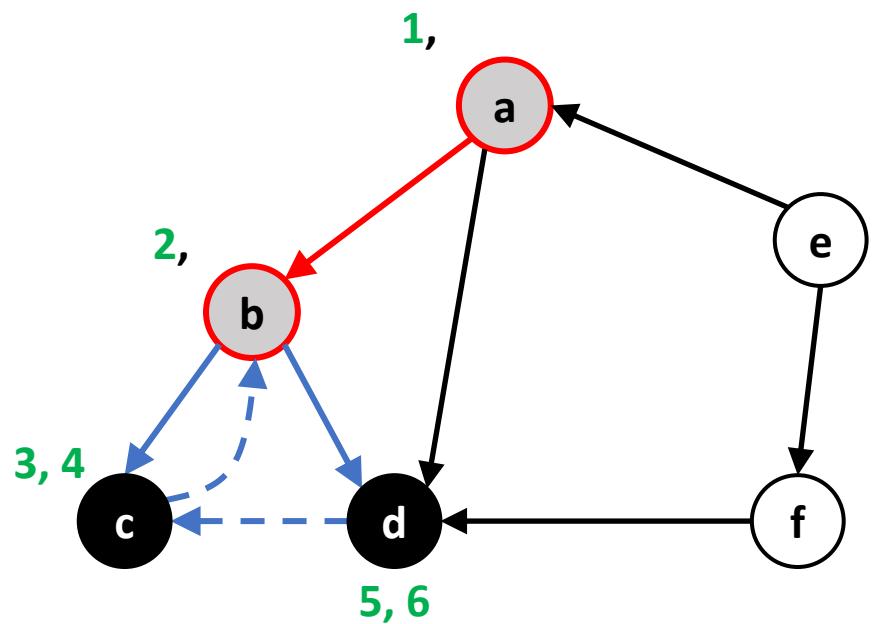
G	G	B	B	W	W
---	---	---	---	---	---

a b c d e f



DFS-Explore(G , a)

DFS(G)



Adj List of G :

a	→	b	→	d
b	→	c	→	d
✓	c	→	b	
✓	d	→	c	
e	→	a	→	f
f	→	d		

Colors of nodes:

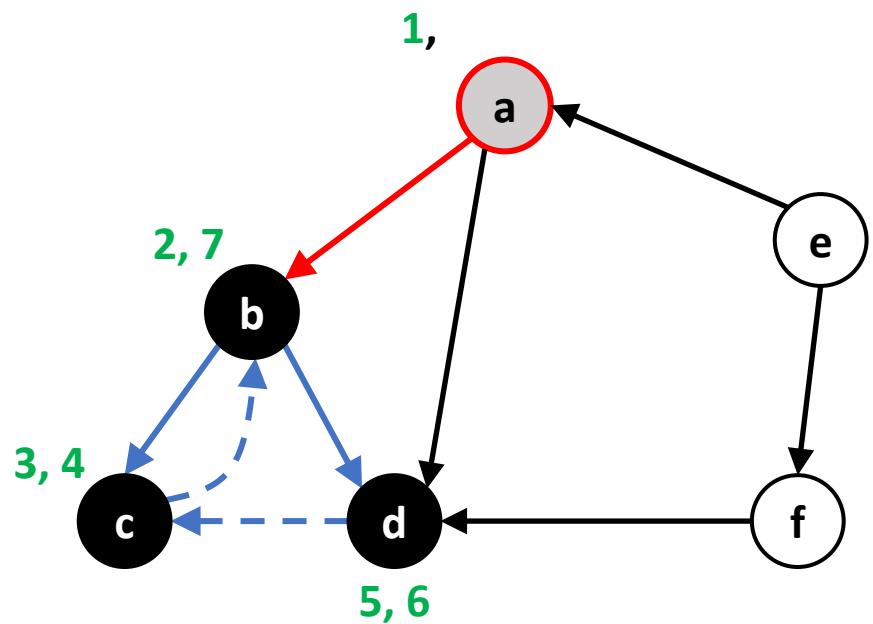
G	G	B	B	W	W
---	---	---	---	---	---

a b c d e f



DFS-Explore(G, a)

DFS(G)



Adj List of G :

a	→	b	→	d	
✓	b	→	c	→	d
✓	c	→	b		
✓	d	→	c		
e	→	a	→	f	
f	→	d			

Colors of nodes:

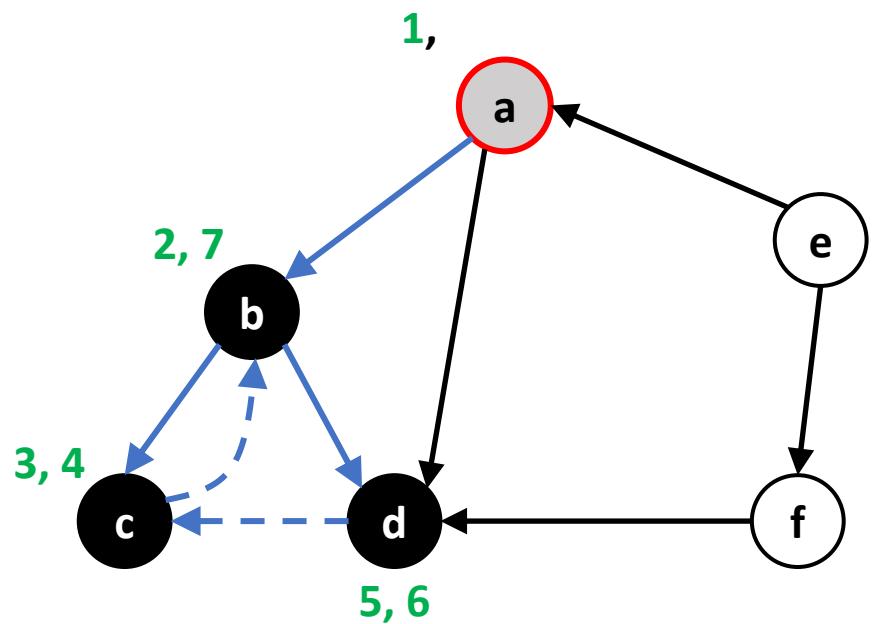
G	B	B	B	W	W
---	---	---	---	---	---

a b c d e f



DFS-Explore(G , a)

DFS(G)



Adj List of G :

a	→	b	→	d
✓ b	→	c	→	d
✓ c	→	b		
✓ d	→	c		
e	→	a	→	f
f	→	d		

Colors of nodes:

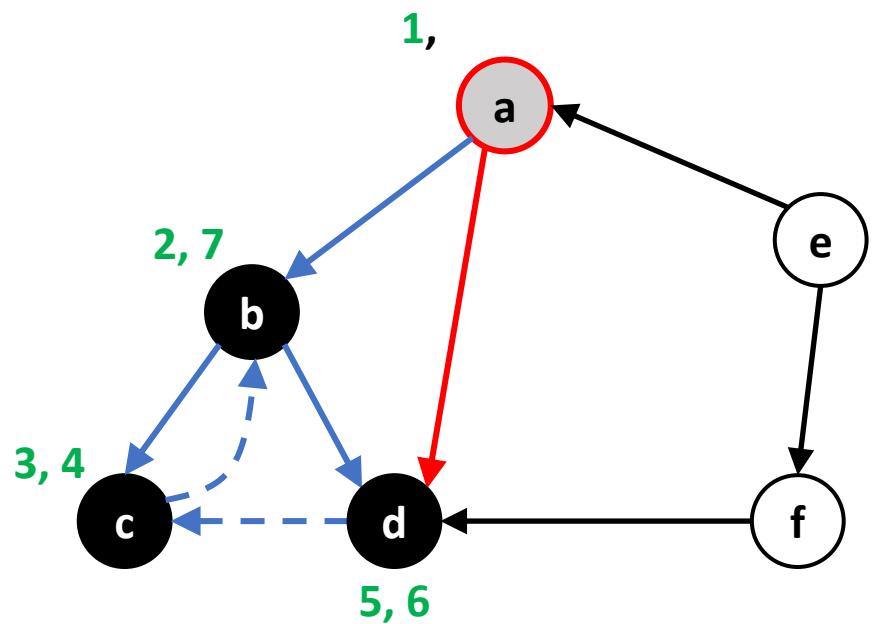
G	B	B	B	W	W
---	---	---	---	---	---

a b c d e f



DFS-Explore(G, a)

DFS(G)



Adj List of G :

a	→	b	→	d
✓ b	→	c	→	d
✓ c	→	b		
✓ d	→	c		
e	→	a	→	f
f	→	d		

Colors of nodes:

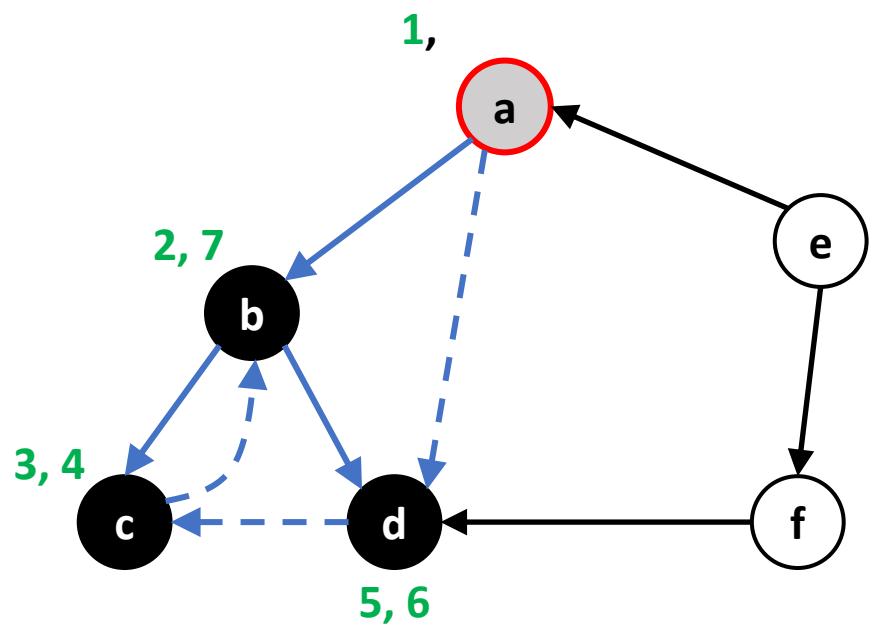
G	B	B	B	W	W
---	---	---	---	---	---

a b c d e f



DFS-Explore(G , a)

DFS(G)



Adj List of G :

a	→	b	→	d
✓ b	→	c	→	d
✓ c	→	b		
✓ d	→	c		
e	→	a	→	f
f	→	d		

Colors of nodes:

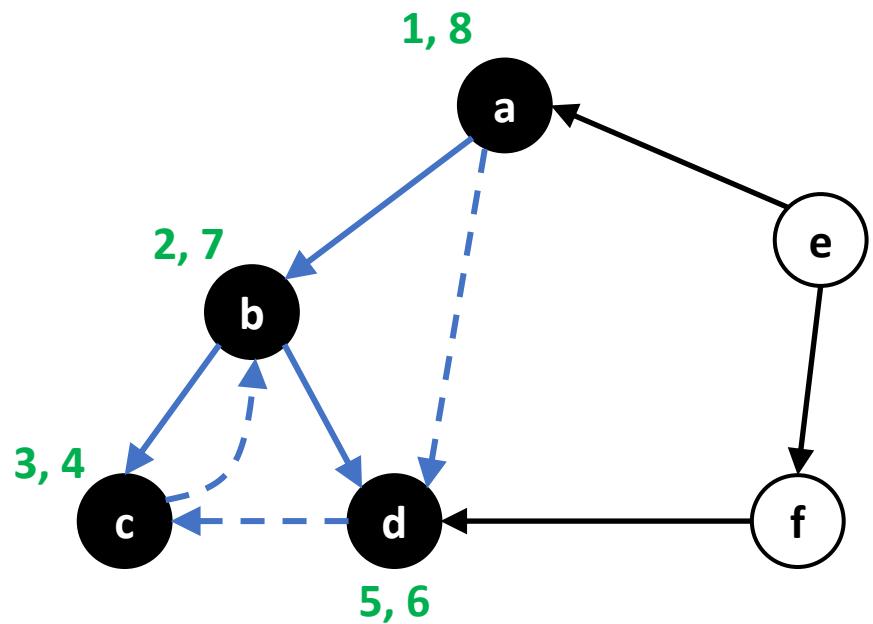
G	B	B	B	W	W
---	---	---	---	---	---

a b c d e f



DFS-Explore(G , a)

DFS(G)



Adj List of G :

✓ a	→ b	→ d
✓ b	→ c	→ d
✓ c	→ b	
✓ d	→ c	
e	→ a	→ f
f	→ d	

Colors of nodes:

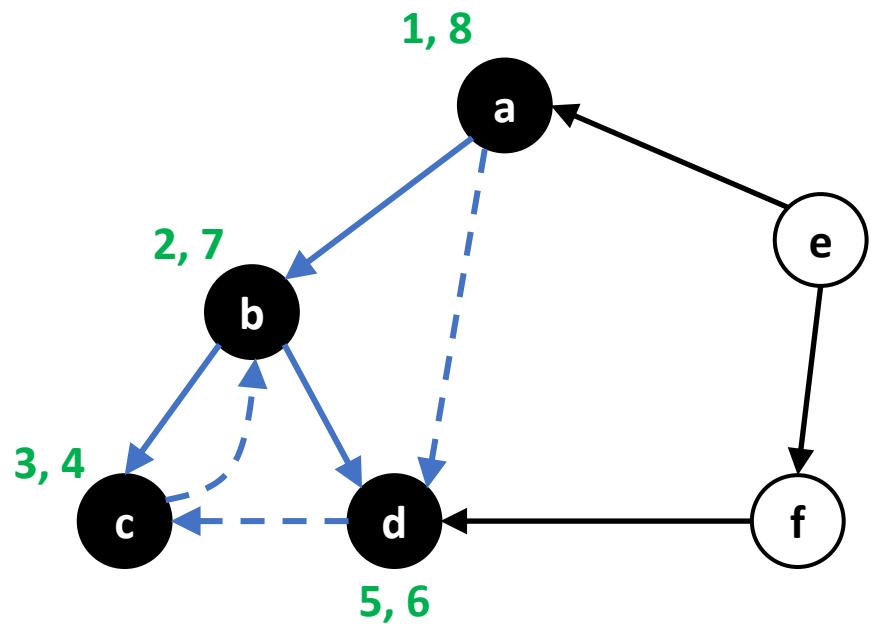
B	B	B	B	W	W
---	---	---	---	---	---

a b c d e f



DFS-Explore(G, a)

DFS(G)



Adj List of G :

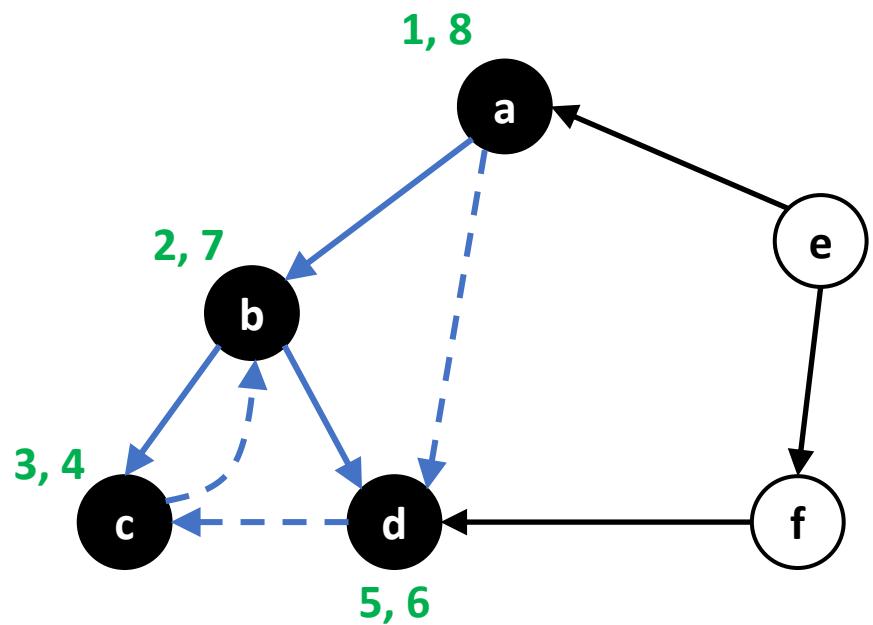
✓	a	→	b	→	d
✓	b	→	c	→	d
✓	c	→	b		
✓	d	→	c		
e		→	a	→	f
f		→	d		

Colors of nodes:

B	B	B	B	W	W
a	b	c	d	e	f

↑

DFS(G)



Adj List of G :

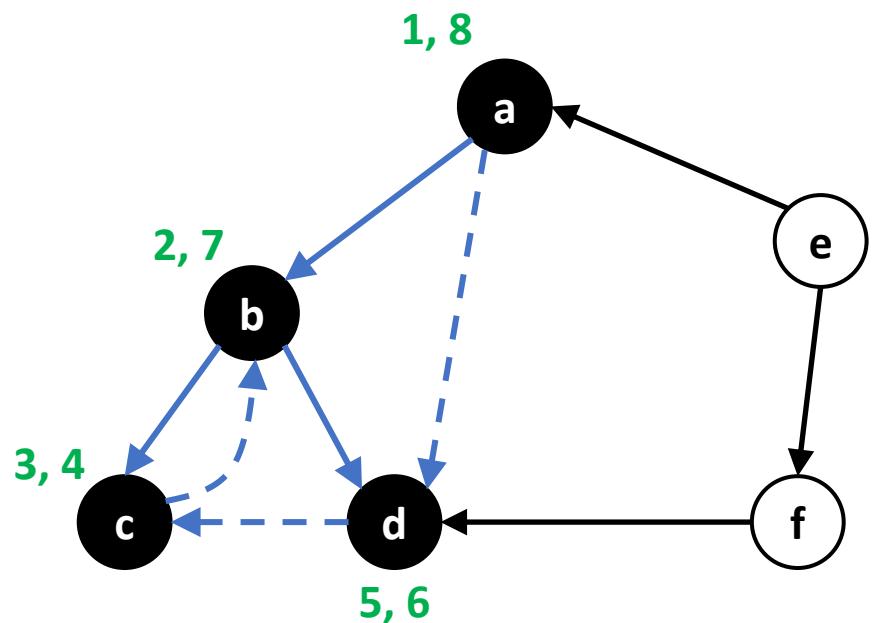
✓ a	→ b	→ d
✓ b	→ c	→ d
✓ c	→ b	
✓ d	→ c	
e	→ a	→ f
f	→ d	

Colors of nodes:

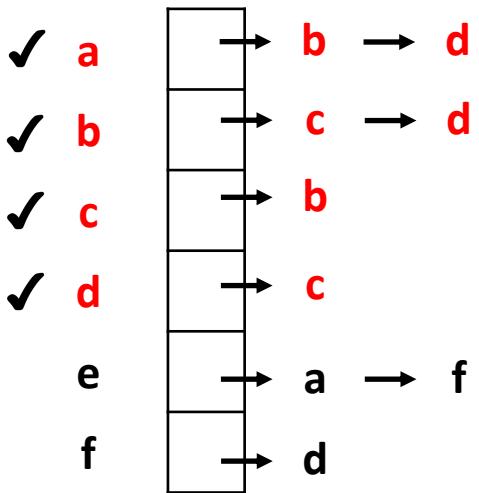
B	B	B	B	W	W
a	b	c	d	e	f

↑

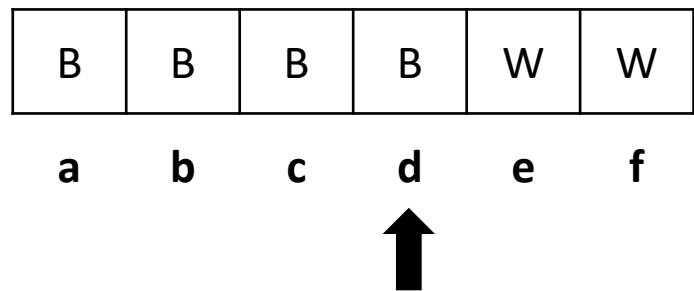
DFS(G)



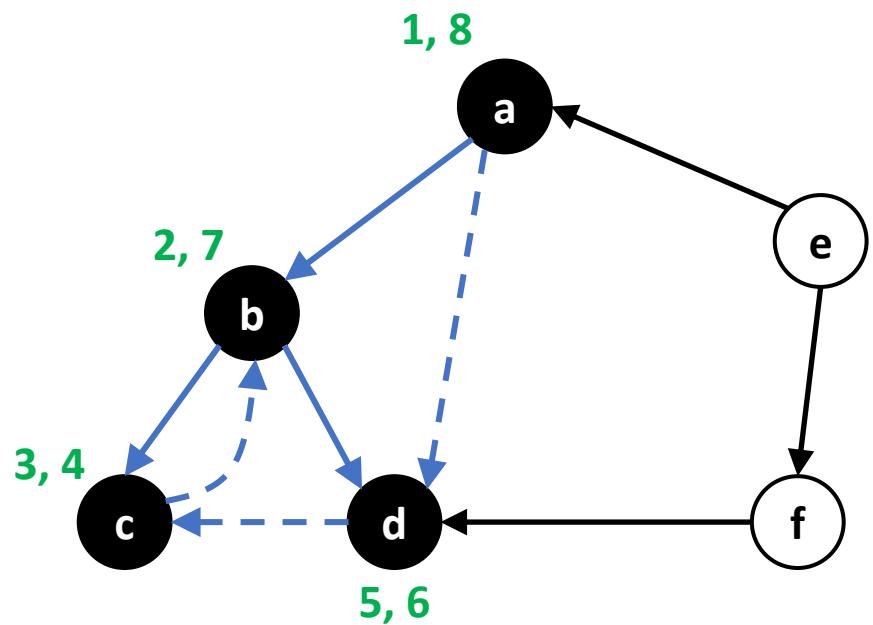
Adj List of G :



Colors of nodes:



DFS(G)



Adj List of G :

✓	a	→	b	→	d
✓	b	→	c	→	d
✓	c	→	b		
✓	d	→	c		
e		→	a	→	f
f		→	d		

Colors of nodes:

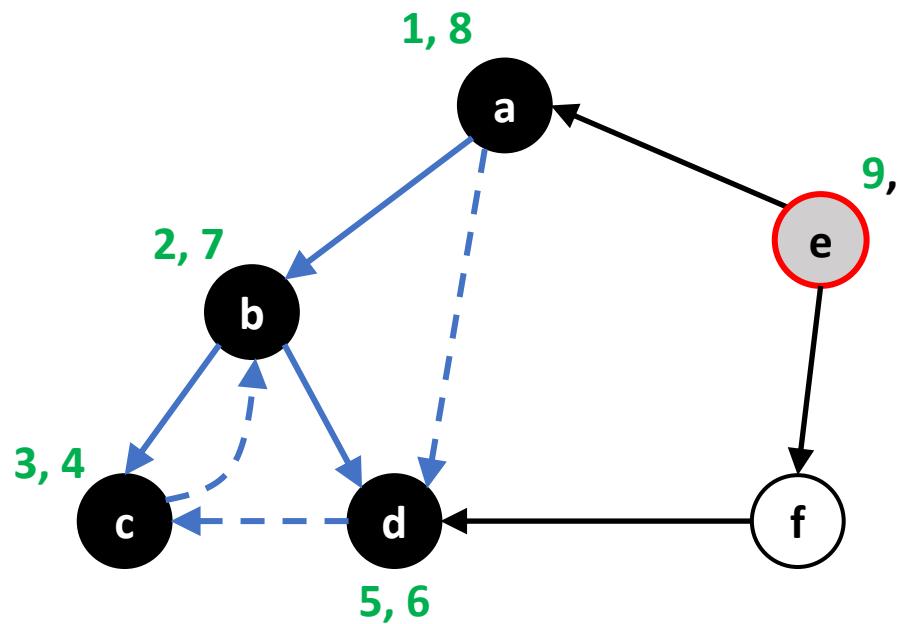
B	B	B	B	W	W
---	---	---	---	---	---

a b c d e f

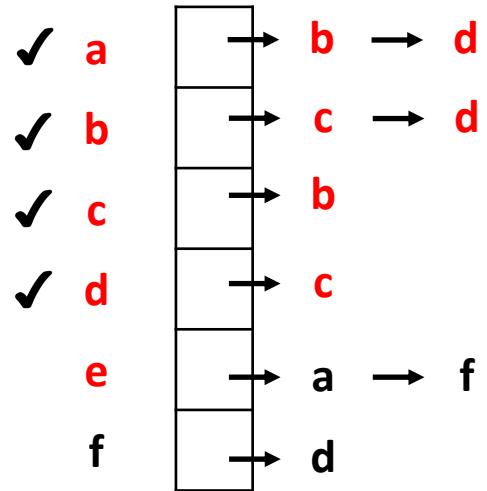


DFS-Explore(G, e)

DFS(G)



Adj List of G :



Colors of nodes:

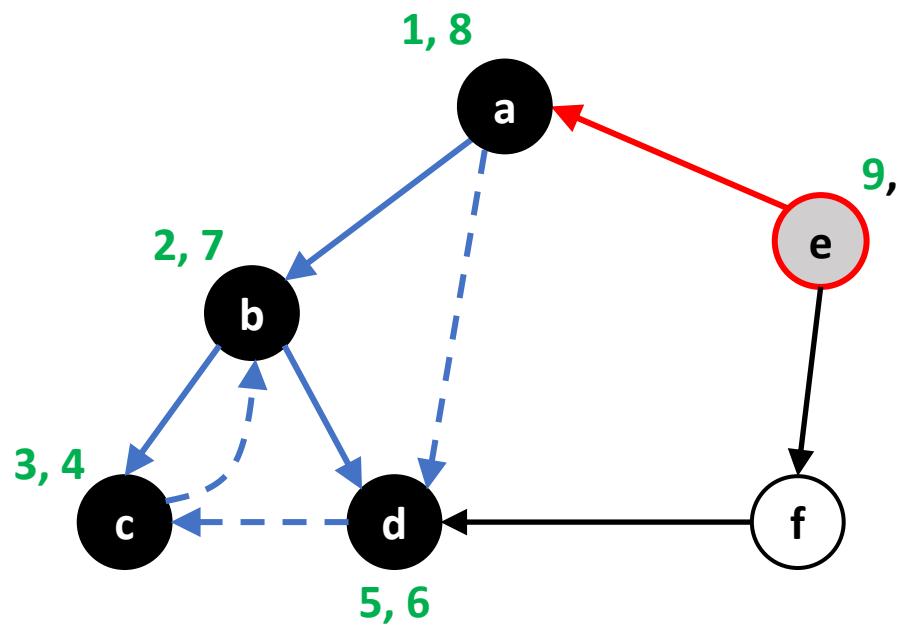
B	B	B	B	G	W
---	---	---	---	---	---

a b c d e f



DFS-Explore(G, e)

DFS(G)



Adj List of G :

✓	a	→	b	→	d
✓	b	→	c	→	d
✓	c	→	b		
✓	d	→	c		
e		→	a	→	f
f		→	d		

Colors of nodes:

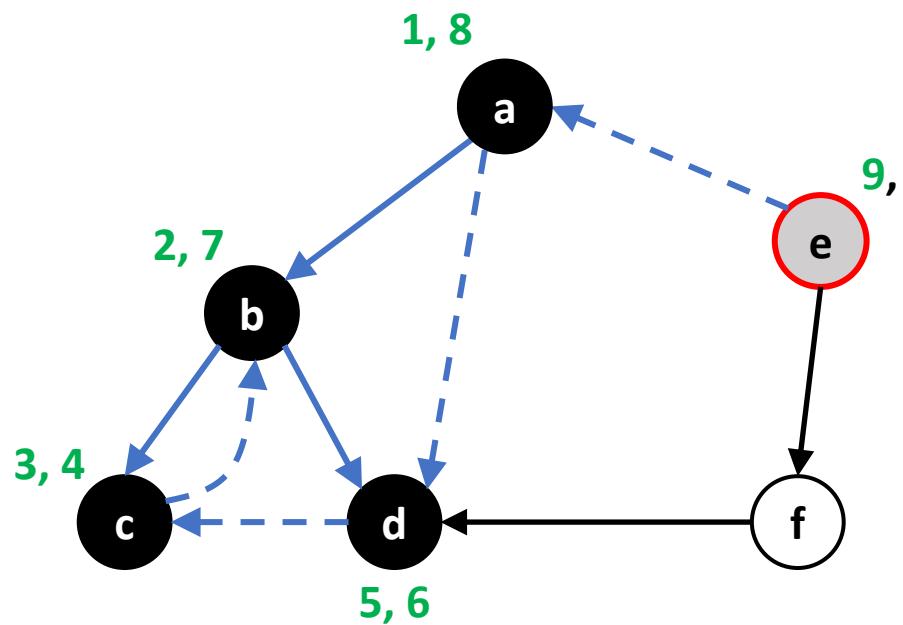
B	B	B	B	G	W
---	---	---	---	---	---

a b c d e f



DFS-Explore(G, e)

DFS(G)



Adj List of G :

✓	a	→	b	→	d
✓	b	→	c	→	d
✓	c	→	b		
✓	d	→	c		
e		→	a	→	f
f		→	d		

Colors of nodes:

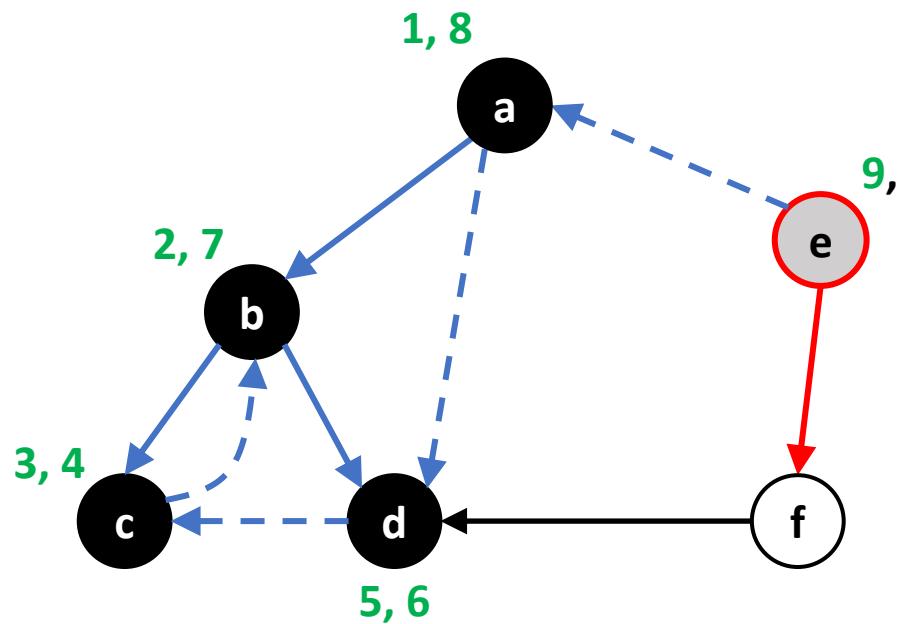
B	B	B	B	G	W
---	---	---	---	---	---

a b c d e f



DFS-Explore(G, e)

DFS(G)



Adj List of G :

✓ a	→ b	→ d
✓ b	→ c	→ d
✓ c	→ b	
✓ d	→ c	
e	→ a	→ f
f	→ d	

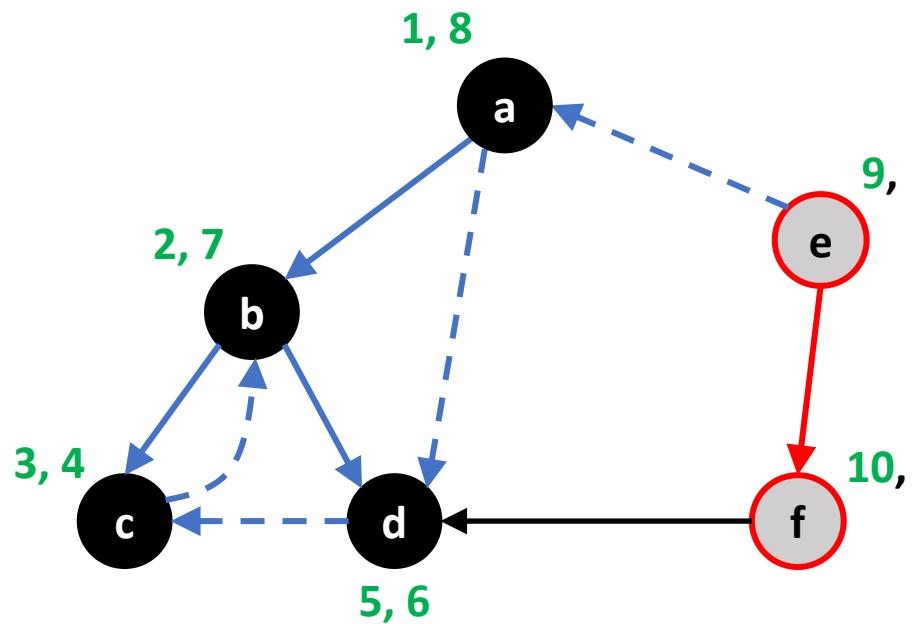
Colors of nodes:

B	B	B	B	G	W
a	b	c	d	e	f

↑

DFS-Explore(G, e)

DFS(G)



Adj List of G :

✓ a	→ b	→ d
✓ b	→ c	→ d
✓ c	→ b	
✓ d	→ c	
e	→ a	→ f
f	→ d	

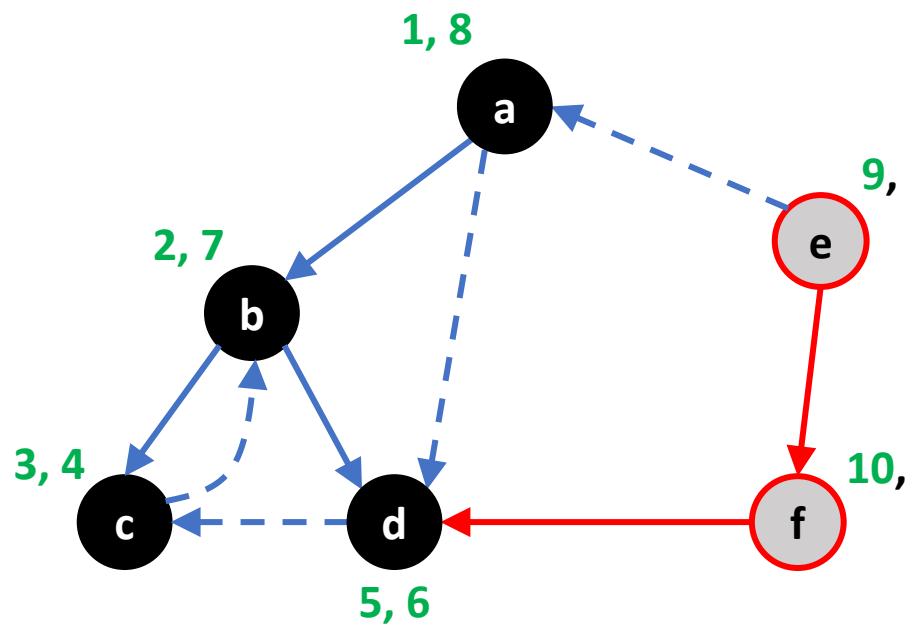
Colors of nodes:

B	B	B	B	G	G
a	b	c	d	e	f

↑

DFS-Explore(G, e)

DFS(G)



Adj List of G :

✓ a	→ b	→ d
✓ b	→ c	→ d
✓ c	→ b	
✓ d	→ c	
e	→ a	→ f
f	→ d	

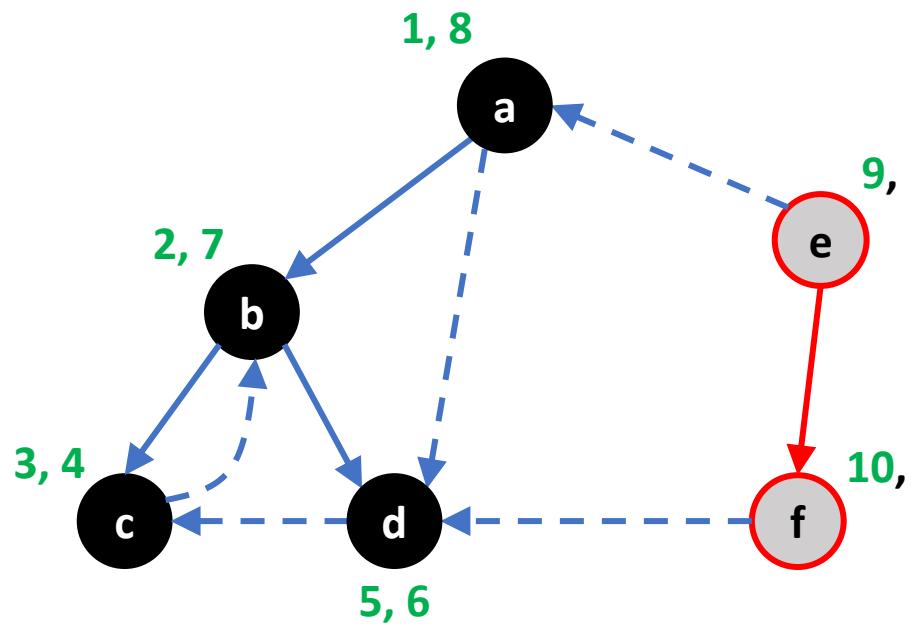
Colors of nodes:

B	B	B	B	G	G
a	b	c	d	e	f

↑

DFS-Explore(G, e)

DFS(G)



Adj List of G :

✓ a	→ b	→ d
✓ b	→ c	→ d
✓ c	→ b	
✓ d	→ c	
e	→ a	→ f
f	→ d	

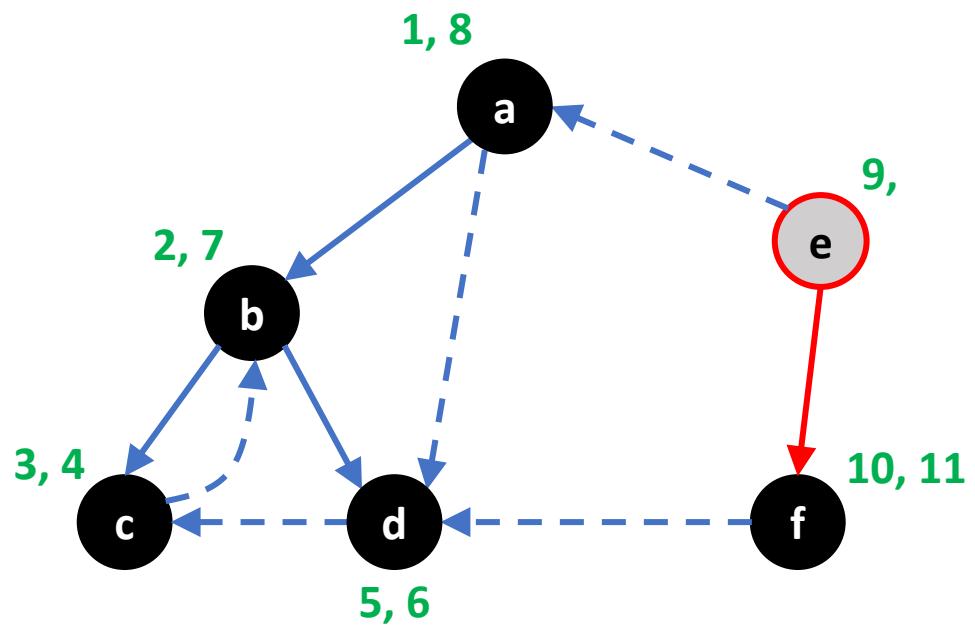
Colors of nodes:

B	B	B	B	G	G
a	b	c	d	e	f

↑

DFS-Explore(G, e)

DFS(G)



Adj List of G :

✓ a	→ b	→ d
✓ b	→ c	→ d
✓ c	→ b	
✓ d	→ c	
e	→ a	→ f
✓ f	→ d	

Colors of nodes:

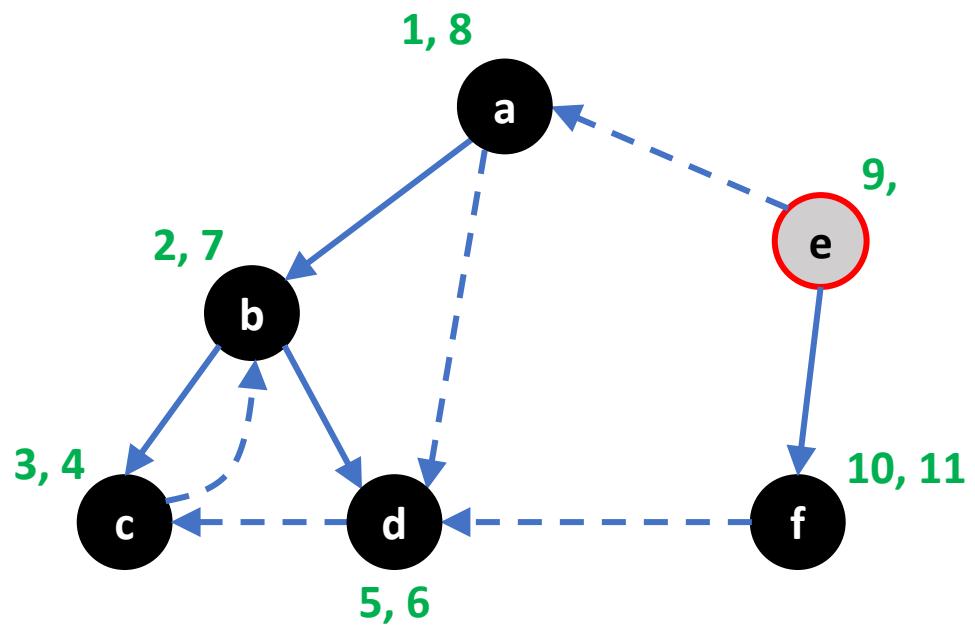
B	B	B	B	G	B
---	---	---	---	---	---

a b c d e f



DFS-Explore(G, e)

DFS(G)



Adj List of G :

✓ a	→ b	→ d
✓ b	→ c	→ d
✓ c	→ b	
✓ d	→ c	
e	→ a	→ f
✓ f	→ d	

Colors of nodes:

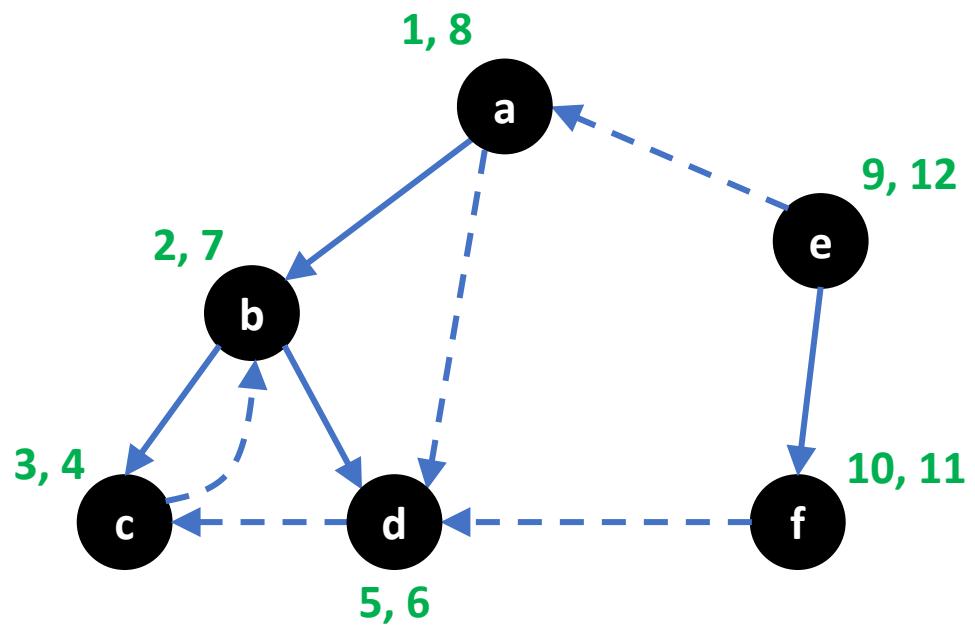
B	B	B	B	G	B
---	---	---	---	---	---

a b c d e f



DFS-Explore(G, e)

DFS(G)



Adj List of G :

✓ a	→ b	→ d
✓ b	→ c	→ d
✓ c	→ b	
✓ d	→ c	
✓ e	→ a	→ f
✓ f	→ d	

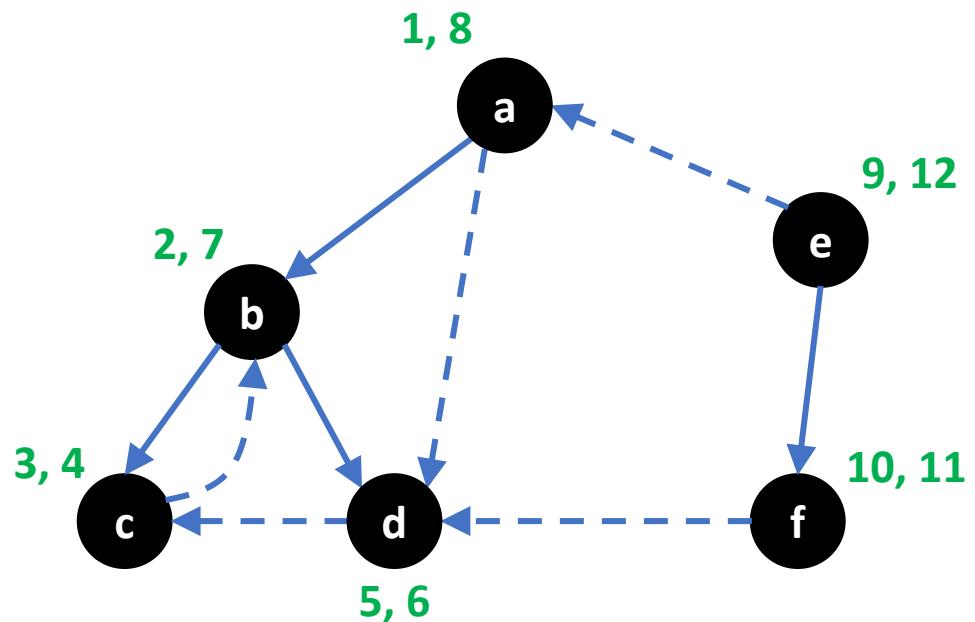
Colors of nodes:

B	B	B	B	B	B
a	b	c	d	e	f

↑

DFS-Explore(G, e)

DFS(G)



Adj List of G :

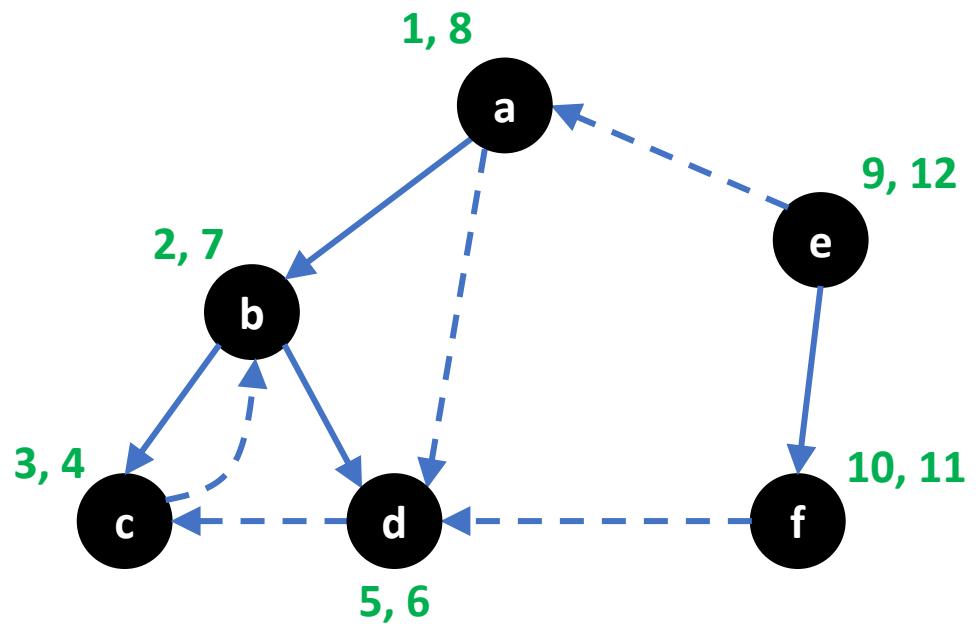
✓ a	→ b	→ d
✓ b	→ c	→ d
✓ c	→ b	
✓ d	→ c	
✓ e	→ a	→ f
✓ f	→ d	

Colors of nodes:

B	B	B	B	B	B
a	b	c	d	e	f

↑

DFS(G)



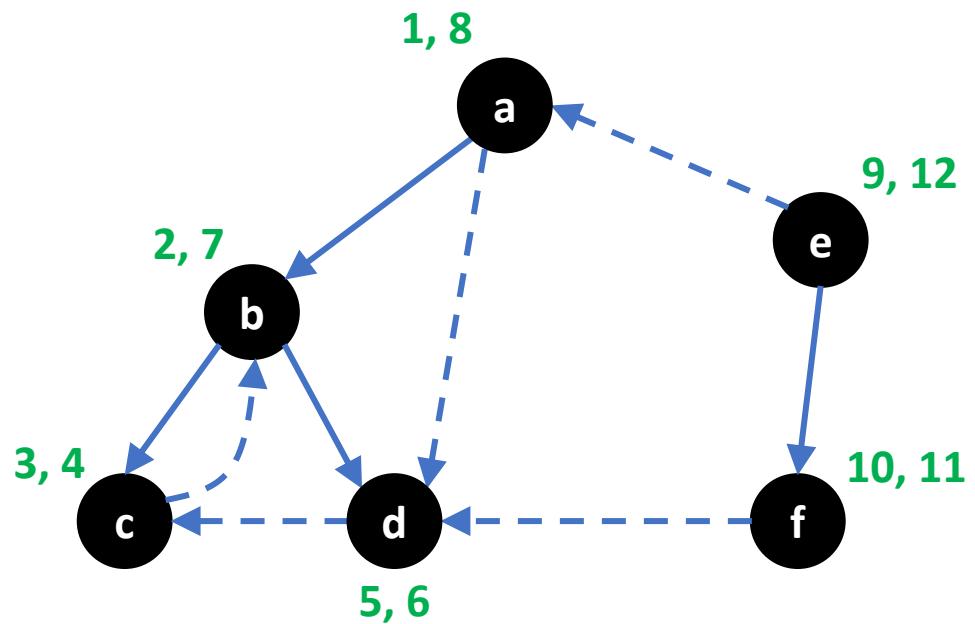
Adj List of G :

✓ a	→ b	→ d
✓ b	→ c	→ d
✓ c	→ b	
✓ d	→ c	
✓ e	→ a	→ f
✓ f	→ d	

Colors of nodes:

B	B	B	B	B	B
a	b	c	d	e	f

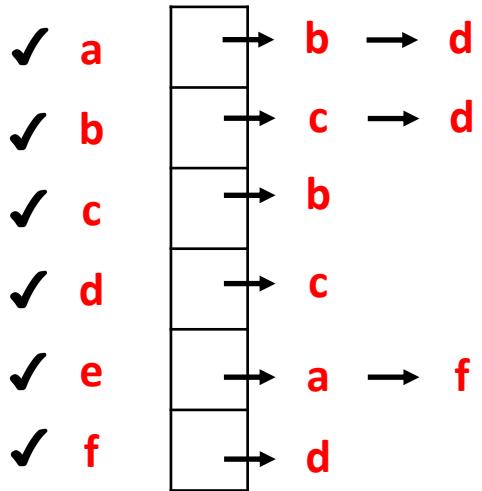
DFS(G)



Worst-case time complexity: $O(|V| + |E|)$

$O(n + m)$

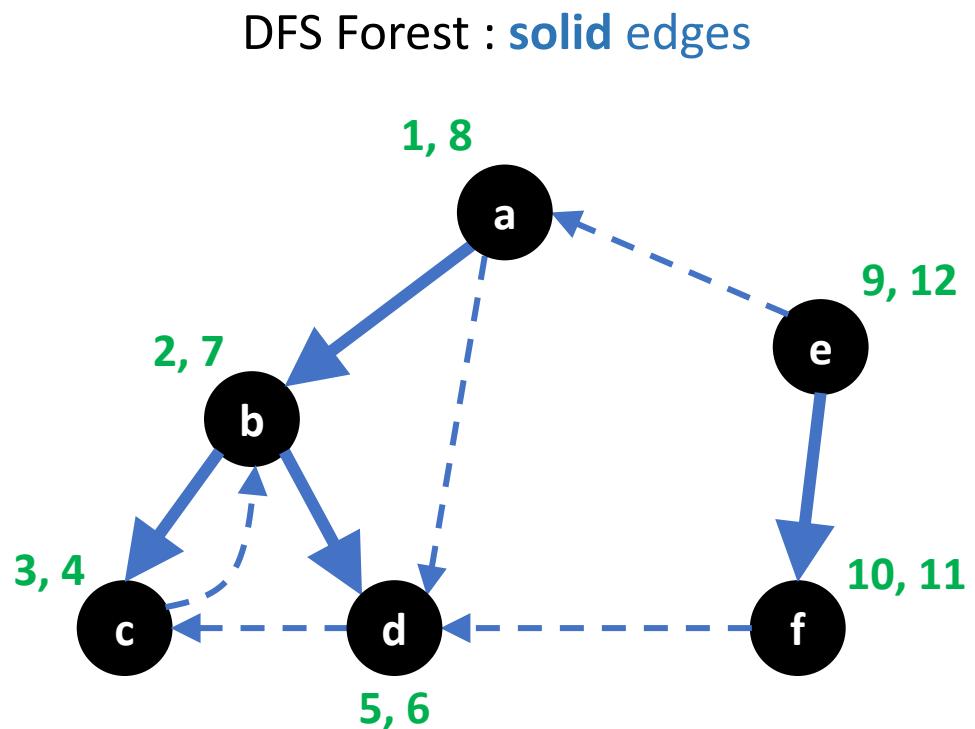
Adj List of G :



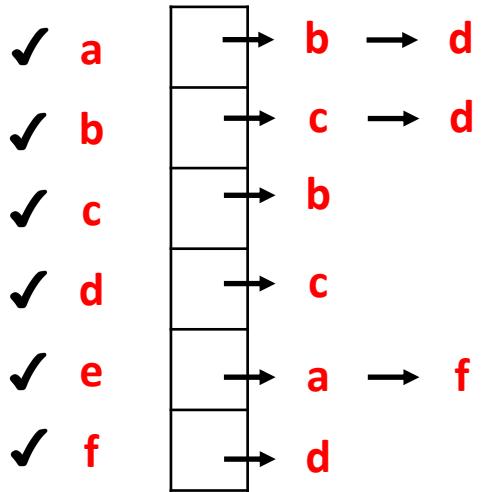
Colors of nodes:

B	B	B	B	B	B
a	b	c	d	e	f

DFS(G)



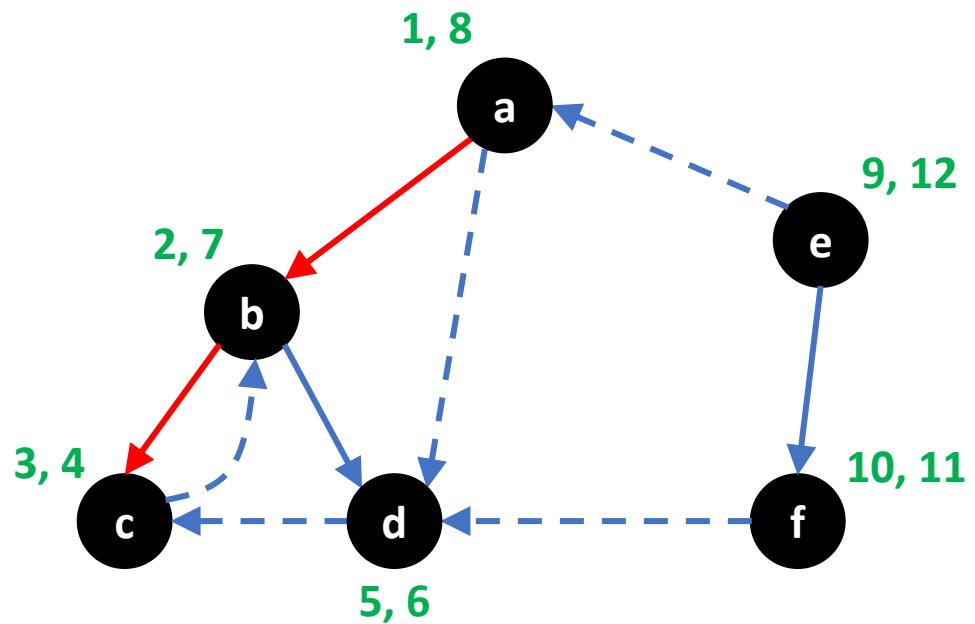
Adj List of G :



Colors of nodes:

B	B	B	B	B	B
a	b	c	d	e	f

DFS(G)



a is ancestor of c

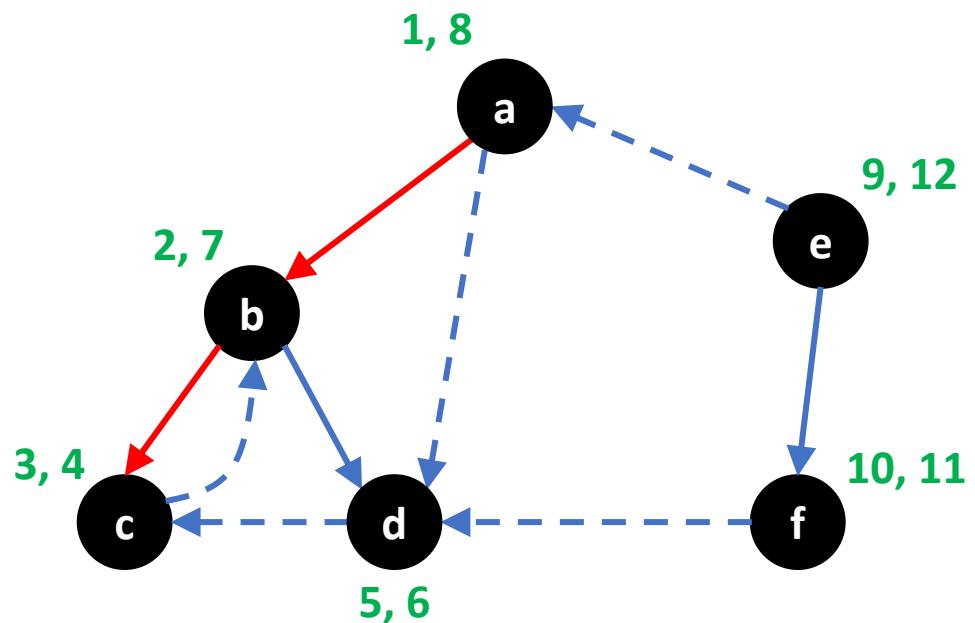
Adj List of G :

✓ a	→ b	→ d
✓ b	→ c	→ d
✓ c	→ b	
✓ d	→ c	
✓ e	→ a	→ f
✓ f	→ d	

Colors of nodes:

B	B	B	B	B	B
a	b	c	d	e	f

DFS(G)



a is **ancestor** of c
c is **descendant** of a

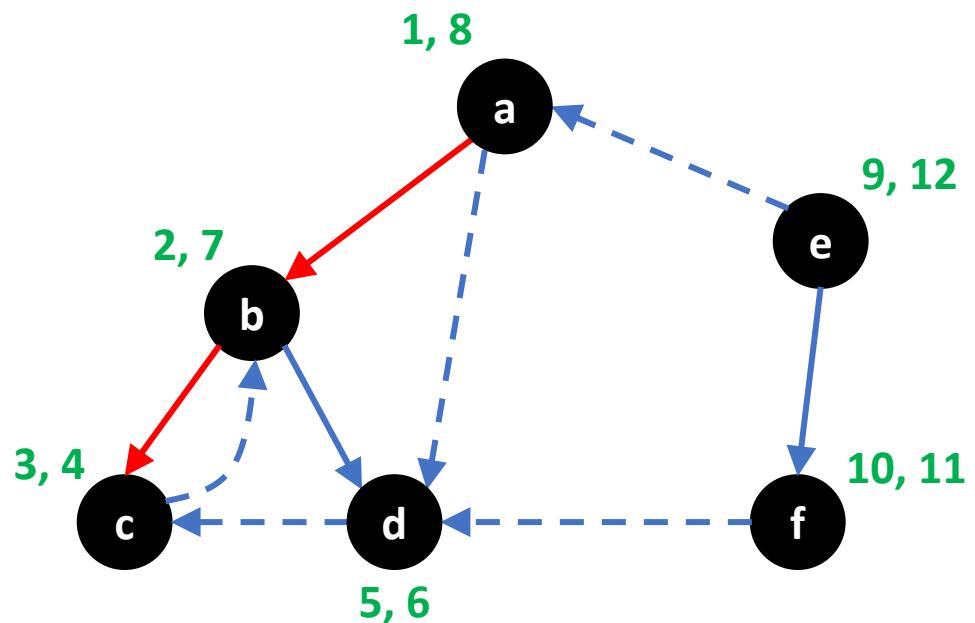
Adj List of G :

✓ a	→ b	→ d
✓ b	→ c	→ d
✓ c	→ b	
✓ d	→ c	
✓ e	→ a	→ f
✓ f	→ d	

Colors of nodes:

B	B	B	B	B	B
a	b	c	d	e	f

DFS(G)



1
d[a]

a is **ancestor** of c
c is **descendant** of a

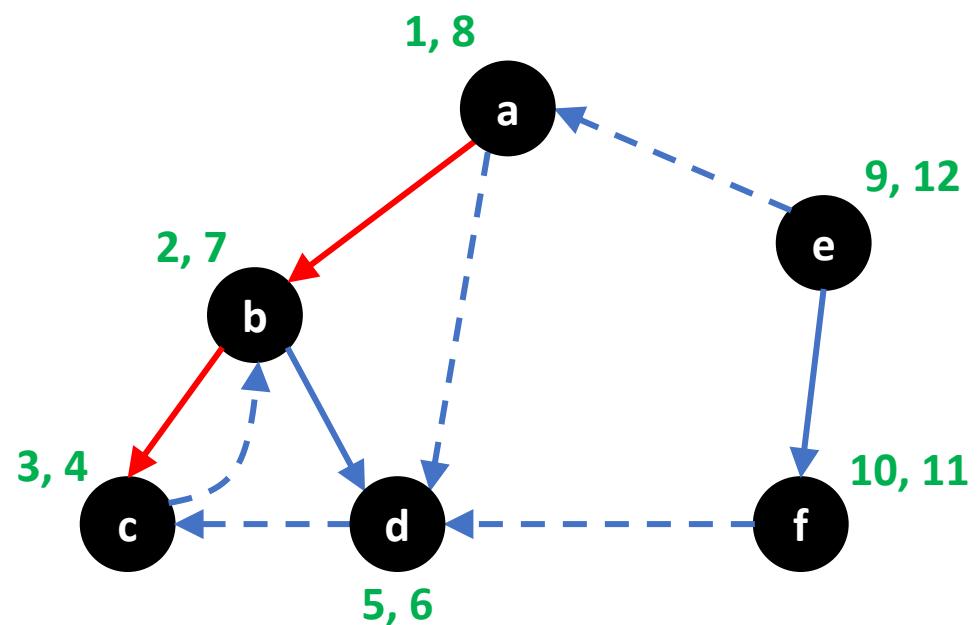
Adj List of G :

✓ a	→ b	→ d
✓ b	→ c	→ d
✓ c	→ b	
✓ d	→ c	
✓ e	→ a	→ f
✓ f	→ d	

Colors of nodes:

B	B	B	B	B	B
a	b	c	d	e	f

DFS(G)



1
d[a] < d[c]

3

a is **ancestor** of c
c is **descendant** of a

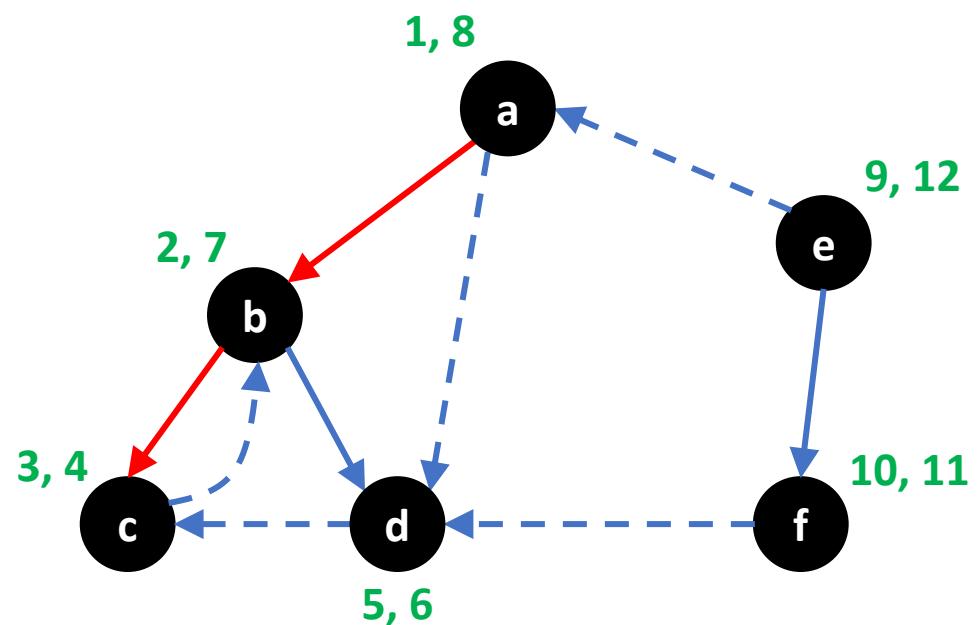
Adj List of G :

✓ a	→ b	→ d
✓ b	→ c	→ d
✓ c	→ b	
✓ d	→ c	
✓ e	→ a	→ f
✓ f	→ d	

Colors of nodes:

B	B	B	B	B	B
a	b	c	d	e	f

DFS(G)



$$\begin{matrix} 1 \\ d[a] < d[c] < f[c] \\ 3 \quad 4 \end{matrix}$$

a is **ancestor** of c
c is **descendant** of a

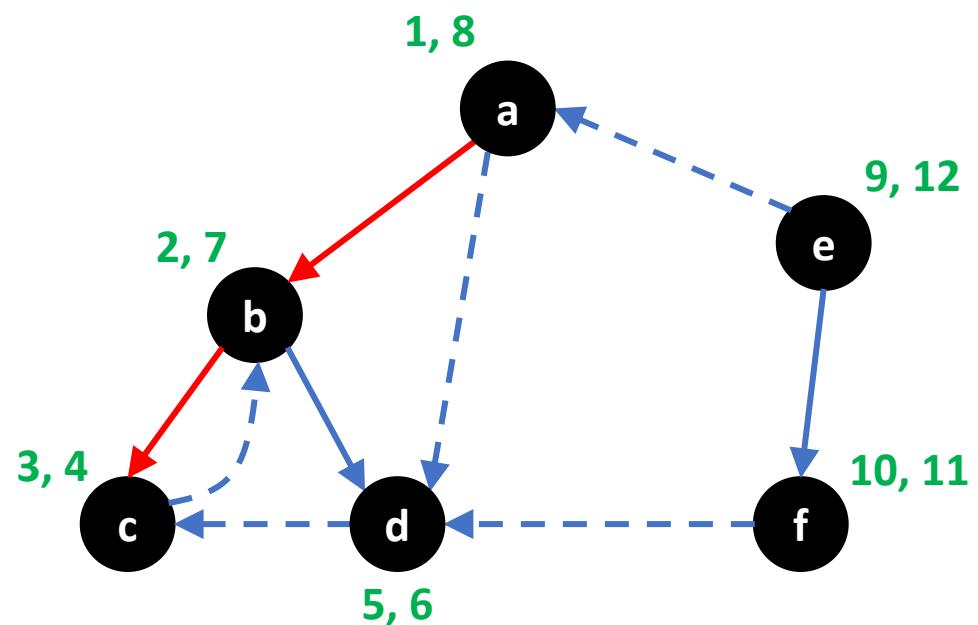
Adj List of G :

✓ a	→ b	→ d
✓ b	→ c	→ d
✓ c	→ b	
✓ d	→ c	
✓ e	→ a	→ f
✓ f	→ d	

Colors of nodes:

B	B	B	B	B	B
a	b	c	d	e	f

DFS(G)



$$d[a] < d[c] < f[c] < f[a]$$

1 8
3 4

a is **ancestor** of c
c is **descendant** of a

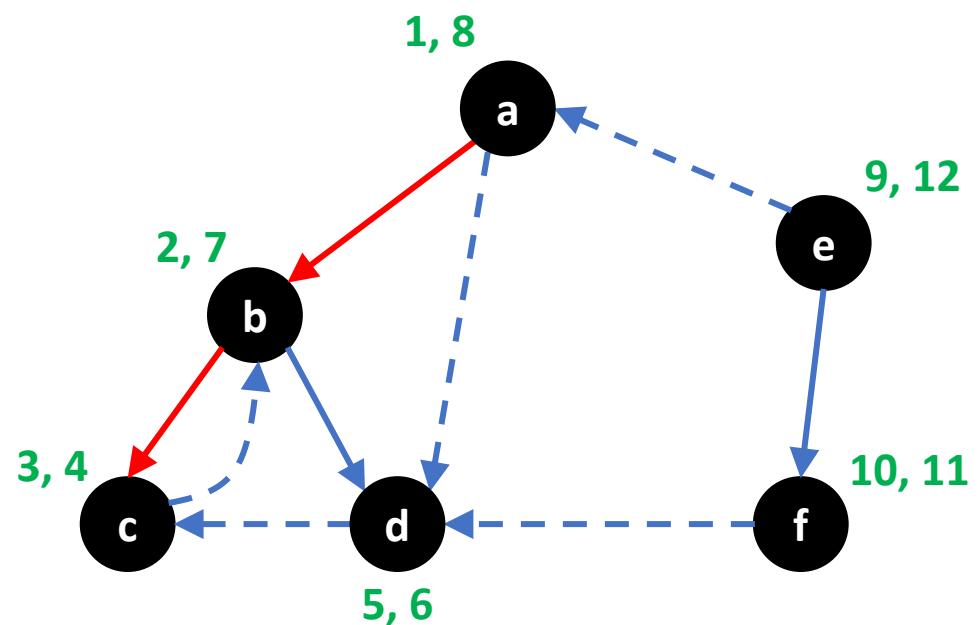
Adj List of G :

✓ a	→ b	→ d
✓ b	→ c	→ d
✓ c	→ b	
✓ d	→ c	
✓ e	→ a	→ f
✓ f	→ d	

Colors of nodes:

B	B	B	B	B	B
a	b	c	d	e	f

DFS(G)



$$d[a] < d[c] < f[c] < f[a]$$

1 8
3 4

a is **ancestor** of c
c is **descendant** of a

Adj List of G :

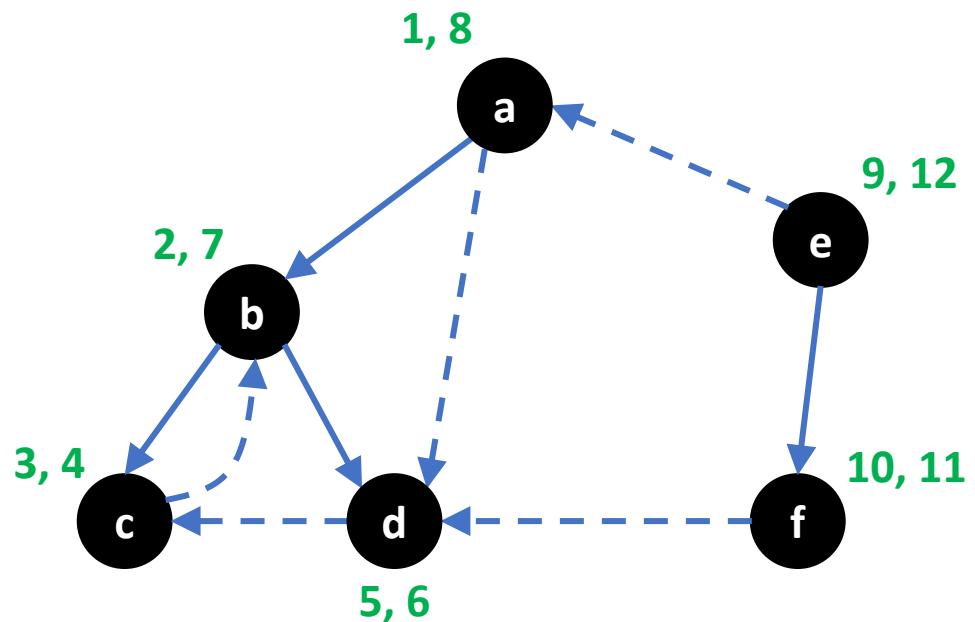
✓ a	→ b	→ d
✓ b	→ c	→ d
✓ c	→ b	
✓ d	→ c	
✓ e	→ a	→ f
✓ f	→ d	

Colors of nodes:

B	B	B	B	B	B
a	b	c	d	e	f

DFS(G)

Edge Classification by DFS



Adj List of G :

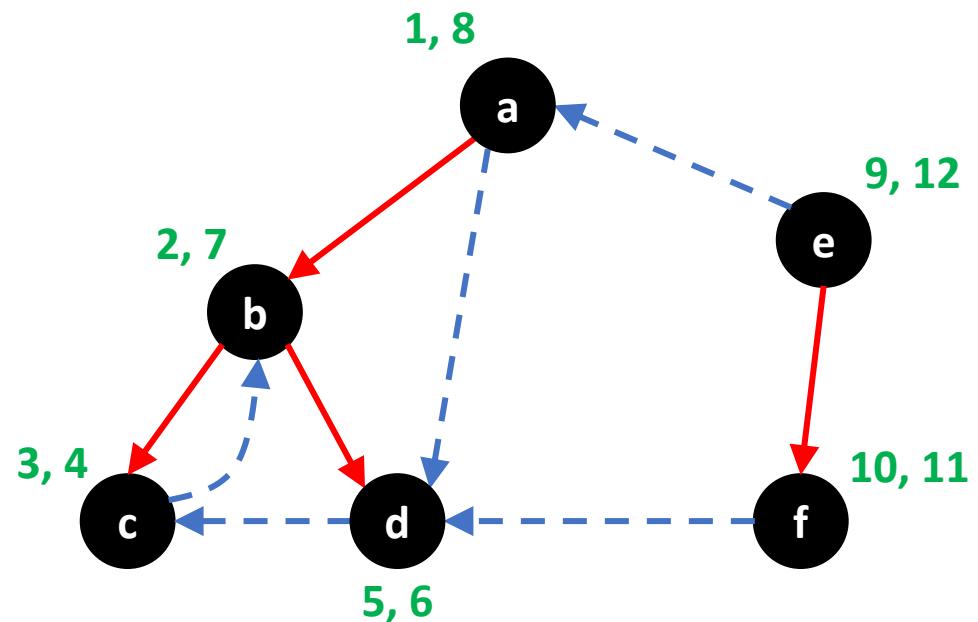
✓ a	→ b	→ d
✓ b	→ c	→ d
✓ c	→ b	
✓ d	→ c	
✓ e	→ a	→ f
✓ f	→ d	

Colors of nodes:

B	B	B	B	B	B
a	b	c	d	e	f

DFS(G)

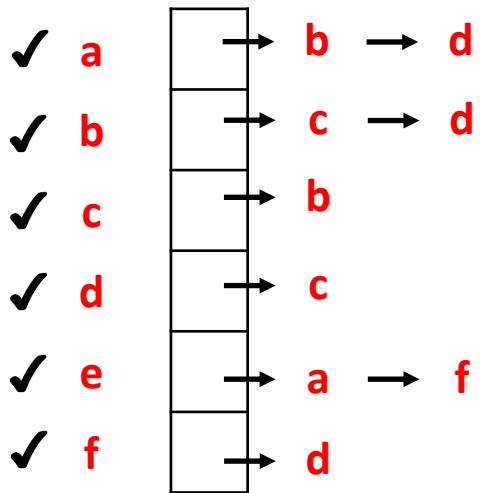
Edge Classification by DFS



An edge $(u, v) \in E$ is a

Tree edge \Leftrightarrow u is the **parent** of v in the DFS forest

Adj List of G :

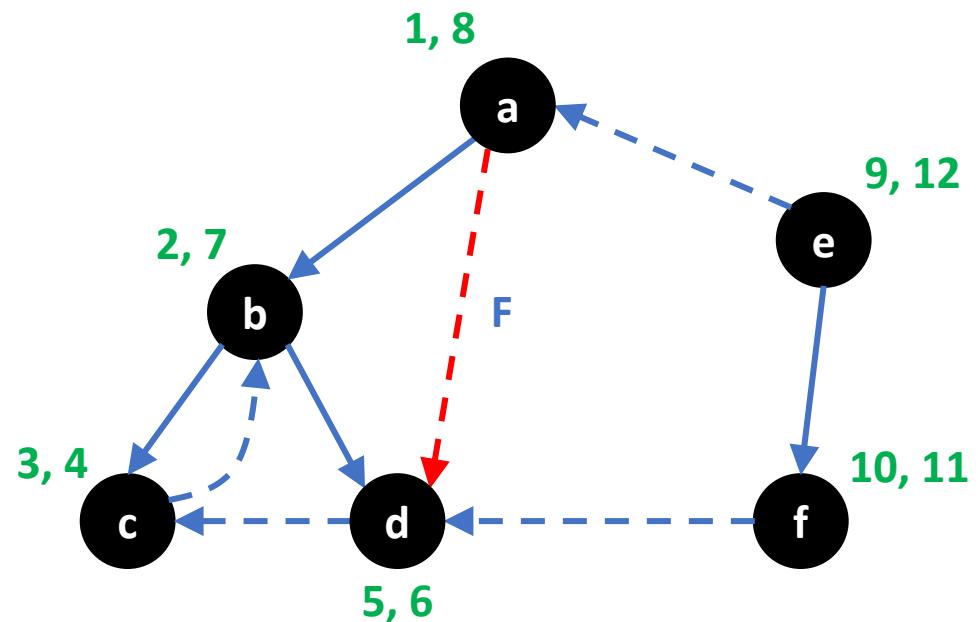


Colors of nodes:

B	B	B	B	B	B
a	b	c	d	e	f

DFS(G)

Edge Classification by DFS



A **non-tree** edge $(u, v) \in E$ is a

Forward edge \Leftrightarrow u is an **ancestor** of v in the DFS forest

Adj List of G :

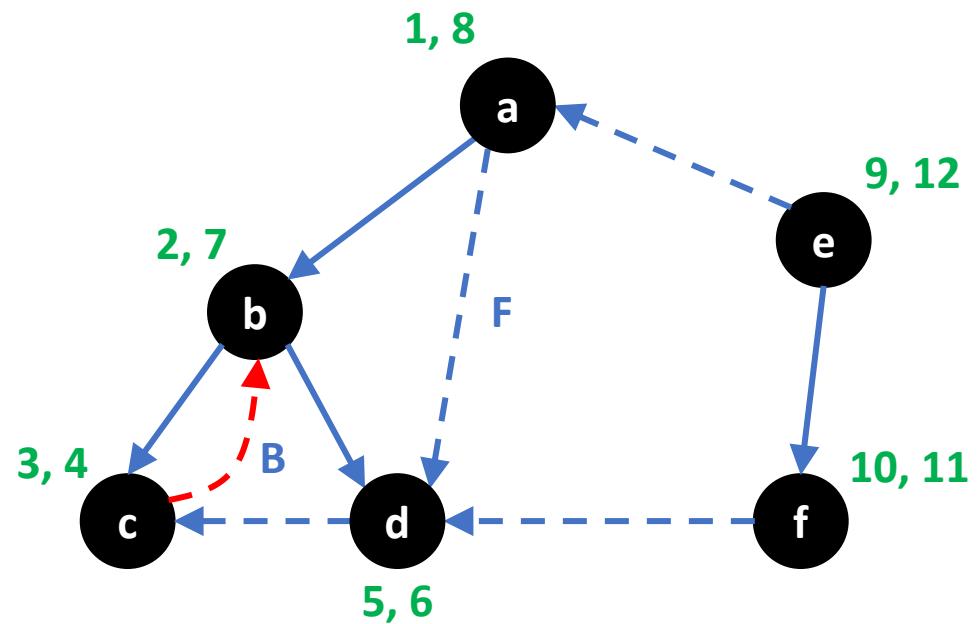
✓ a	→ b	→ d
✓ b	→ c	→ d
✓ c	→ b	
✓ d	→ c	
✓ e	→ a	→ f
✓ f	→ d	

Colors of nodes:

B	B	B	B	B	B
a	b	c	d	e	f

DFS(G)

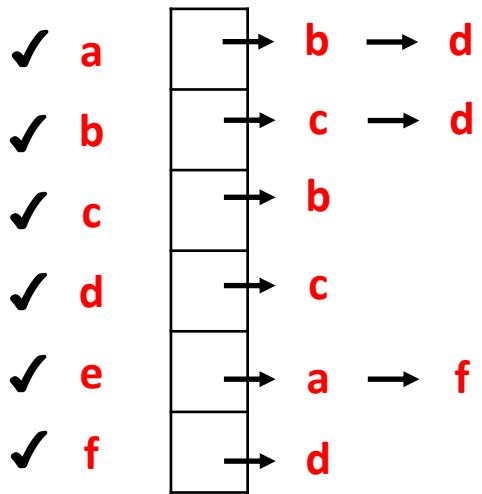
Edge Classification by DFS



A **non-tree** edge $(u, v) \in E$ is a

Back edge $\Leftrightarrow u$ is a **descendent** of v in the DFS forest

Adj List of G :

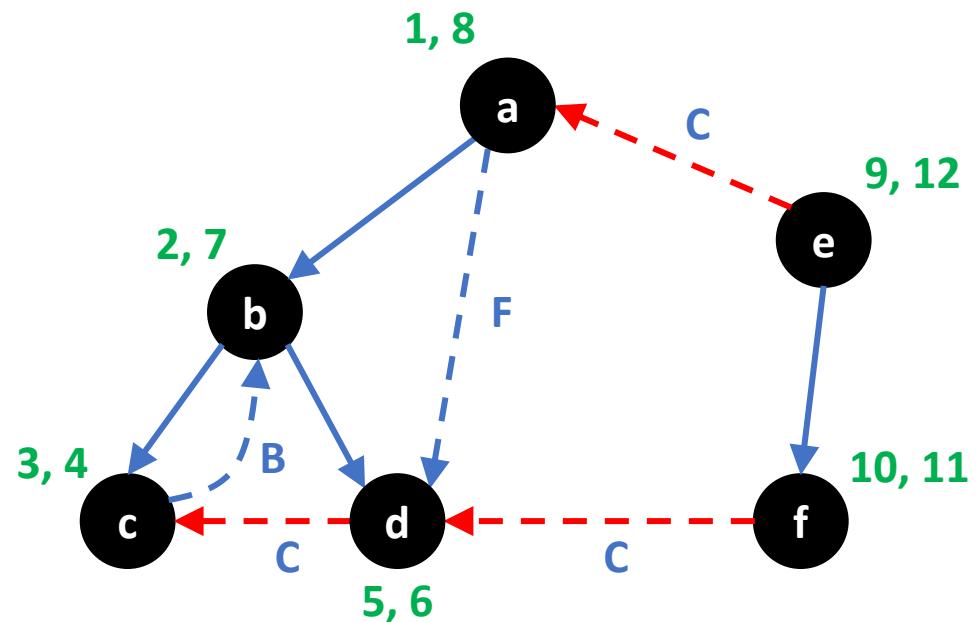


Colors of nodes:

B	B	B	B	B	B
a	b	c	d	e	f

DFS(G)

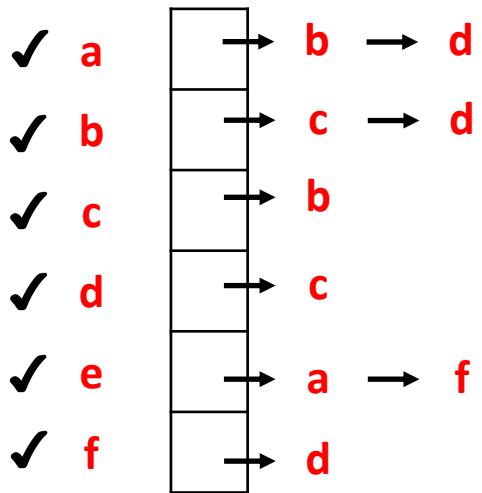
Edge Classification by DFS



A **non-tree** edge $(u, v) \in E$ is a

Cross edge $\Leftrightarrow u$ is **neither ancestor nor descendent** of v
in the DFS forest
 \Leftrightarrow not a forward or back edge

Adj List of G :



Colors of nodes:

B	B	B	B	B	B
a	b	c	d	e	f

Edge Classification by DFS

A DFS of a directed graph $G = (V, E)$ classifies its edges as follows:

$(u,v) \in E$ is a :

- 1. Tree edge**

Edge Classification by DFS

A DFS of a directed graph $G = (V, E)$ classifies its edges as follows:

$(u,v) \in E$ is a :

1. **Tree edge** \Leftrightarrow u is the **parent** of v in the DFS forest

Edge Classification by DFS

A DFS of a directed graph $G = (V, E)$ classifies its edges as follows:

$(u,v) \in E$ is a :

1. **Tree edge** \Leftrightarrow u is the **parent** of v in the DFS forest

Non-tree
edges

Edge Classification by DFS

A DFS of a directed graph $G = (V, E)$ classifies its edges as follows:

$(u,v) \in E$ is a :

1. **Tree edge** \Leftrightarrow u is the **parent** of v in the DFS forest
2. **Forward edge** \Leftrightarrow u is an **ancestor** of v in the DFS forest

Non-tree
edges

Edge Classification by DFS

A DFS of a directed graph $G = (V, E)$ classifies its edges as follows:

$(u,v) \in E$ is a :

1. **Tree edge** \Leftrightarrow u is the **parent** of v in the DFS forest
2. **Forward edge** \Leftrightarrow u is an **ancestor** of v in the DFS forest
3. **Back edge** \Leftrightarrow u is a **descendant** of v in the DFS forest

Non-tree
edges

Edge Classification by DFS

A DFS of a directed graph $G = (V, E)$ classifies its edges as follows:

$(u,v) \in E$ is a :

1. **Tree edge** \Leftrightarrow u is the **parent** of v in the DFS forest
2. **Forward edge** \Leftrightarrow u is an **ancestor** of v in the DFS forest
3. **Back edge** \Leftrightarrow u is a **descendant** of v in the DFS forest
4. **Cross edge** \Leftrightarrow u is **neither ancestor nor descendent** of v in the forest

Non-tree
edges

Claim 1 :

u is an ancestor of v in a DFS of G

Claim 1 :

u is an ancestor of v in a DFS of $G \Leftrightarrow$

Claim 1 :

u is an ancestor of v in a DFS of $G \Leftrightarrow d[u]$

Claim 1 :

u is an ancestor of v in a DFS of $G \Leftrightarrow d[u] < d[v]$

Claim 1 :

u is an ancestor of v in a DFS of $G \Leftrightarrow d[u] < d[v] < f[v]$

Claim 1 :

u is an ancestor of v in a DFS of $G \Leftrightarrow d[u] < d[v] < f[v] < f[u]$

Claim 1 :

u is an ancestor of v in a DFS of $G \Leftrightarrow d[u] < d[v] < f[v] < f[u]$

Exploration of u

Exploration
of v

Claim 1 :

u is an ancestor of v in a DFS of $G \Leftrightarrow d[u] < d[v] < f[v] < f[u]$

Exploration of u

Exploration
of v

Claim 2 :

For any 2 nodes u and v ,

Claim 1 :

u is an ancestor of v in a DFS of $G \Leftrightarrow d[u] < d[v] < f[v] < f[u]$

Exploration of u

Exploration
of v

Claim 2 :

For any 2 nodes u and v , we CANNOT have $d[u]$

Claim 1 :

u is an ancestor of v in a DFS of $G \Leftrightarrow d[u] < d[v] < f[v] < f[u]$

Exploration of u

Exploration
of v

Claim 2 :

For any 2 nodes u and v , we CANNOT have $d[u] < d[v]$

Claim 1 :

u is an ancestor of v in a DFS of $G \Leftrightarrow d[u] < d[v] < f[v] < f[u]$

Exploration of u

Exploration
of v

Claim 2 :

For any 2 nodes u and v , we CANNOT have $d[u] < d[v] < f[u]$

Claim 1 :

u is an ancestor of v in a DFS of $G \Leftrightarrow d[u] < d[v] < f[v] < f[u]$

Exploration of u

Exploration
of v

Claim 2 :

For any 2 nodes u and v , we CANNOT have $d[u] < d[v] < f[u] < f[v]$

Claim 1 :

u is an ancestor of v in a DFS of $G \Leftrightarrow d[u] < d[v] < f[v] < f[u]$

Exploration of u

Exploration
of v

Claim 2 :

For any 2 nodes u and v , we CANNOT have $d[u] < d[v] < f[u] < f[v]$

Exploration of v

Exploration of u

Claim 1 :

u is an ancestor of v in a DFS of $G \Leftrightarrow d[u] < d[v] < f[v] < f[u]$

Exploration of u

Exploration
of v

Claim 2 :

For any 2 nodes u and v , we CANNOT have $d[u] < d[v] < f[u] < f[v]$

Exploration of v

Exploration of u

Claim 3 :

If $(u,v) \in E$ then $d[v] < f[u]$

Claim 1 :

u is an ancestor of v in a DFS of $G \Leftrightarrow d[u] < d[v] < f[v] < f[u]$

Exploration of u

Exploration
of v

Claim 2 :

For any 2 nodes u and v , we CANNOT have $d[u] < d[v] < f[u] < f[v]$

Exploration of v

Exploration of u

Claim 3 :

If $(u,v) \in E$ then $d[v] < f[u]$

Because v is surely discovered before we finish exploring u .

Claim: After a DFS of a directed graph, $\textcircled{u} \rightarrow \textcircled{v} \in E$ is of type:

Claim: After a DFS of a directed graph, $\textcircled{u} \rightarrow \textcircled{v} \in E$ is of type:

Tree or Forward edge $\Leftrightarrow d[u] < d[v] < f[v] < f[u]$



Claim: After a DFS of a directed graph, $\textcircled{u} \rightarrow \textcircled{v} \in E$ is of type:

Tree or Forward edge $\Leftrightarrow d[u] < d[v] < f[v] < f[u]$

Back edge

Claim: After a DFS of a directed graph, $\textcircled{u} \rightarrow \textcircled{v} \in E$ is of type:

$$\text{Tree or Forward edge} \Leftrightarrow d[u] < d[v] < f[v] < f[u]$$

$$\text{Back edge} \Leftrightarrow d[v] < d[u] < f[u] < f[v]$$

Claim: After a DFS of a directed graph, $\textcircled{u} \rightarrow \textcircled{v} \in E$ is of type:

Tree or Forward edge $\Leftrightarrow d[u] < d[v] < f[v] < f[u]$

Back edge $\Leftrightarrow d[v] < d[u] < f[u] < f[v]$

Cross edge

Claim: After a DFS of a directed graph, $\textcircled{u} \rightarrow \textcircled{v} \in E$ is of type:

Tree or Forward edge $\Leftrightarrow d[u] < d[v] < f[v] < f[u]$

Back edge $\Leftrightarrow d[v] < d[u] < f[u] < f[v]$

Cross edge $\Leftrightarrow d[v] < f[v] < d[u] < f[u]$

OR

$d[u] < f[u] < d[v] < f[v]$?

Claim: After a DFS of a directed graph, $\textcircled{u} \rightarrow \textcircled{v} \in E$ is of type:

Tree or Forward edge $\Leftrightarrow d[u] < d[v] < f[v] < f[u]$

Back edge $\Leftrightarrow d[v] < d[u] < f[u] < f[v]$

Cross edge $\Leftrightarrow d[v] < f[v] < d[u] < f[u]$

OR

$d[u] < f[u] < d[v] < f[v]$

Impossible because
of Claim 3 !

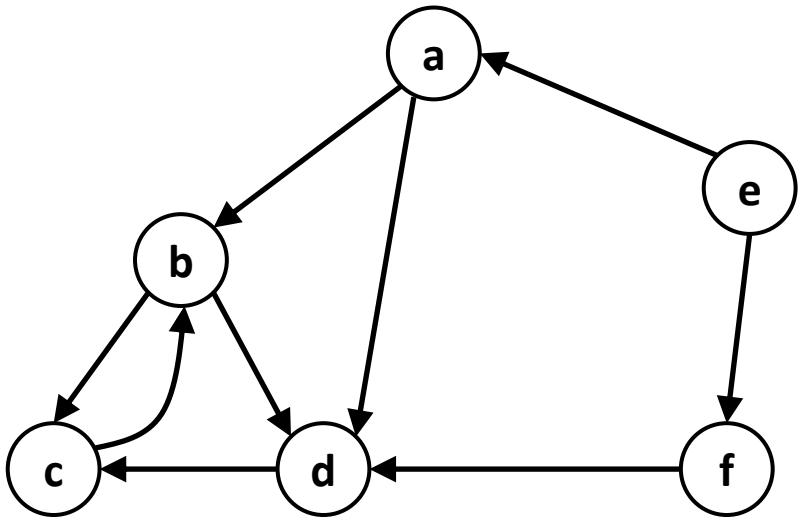
Claim: After a DFS of a directed graph, $\textcircled{u} \rightarrow \textcircled{v} \in E$ is of type:

Tree or Forward edge $\Leftrightarrow d[u] < d[v] < f[v] < f[u]$

Back edge $\Leftrightarrow d[v] < d[u] < f[u] < f[v]$

Cross edge $\Leftrightarrow d[v] < f[v] < d[u] < f[u]$

DFS(G)



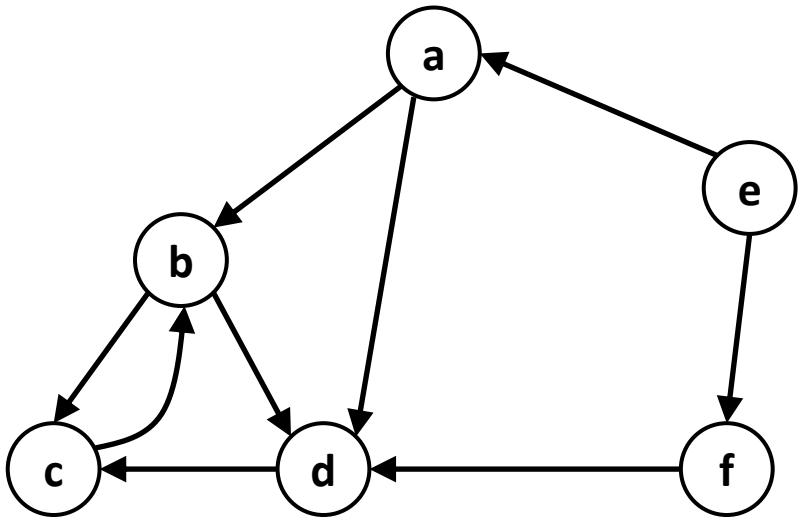
Adj List of G :

a	→	b	→	d
b	→	d	→	c
c	→	b		
d	→	c		
e	→	a	→	f
f	→	d		

Colors of nodes:

W	W	W	W	W	W
e	a	b	c	d	f

DFS(G)



Adj List of G :

a	→	b	→	d
b	→	d	→	c
c	→	b		
d	→	c		
e	→	a	→	f
f	→	d		

Colors of nodes:

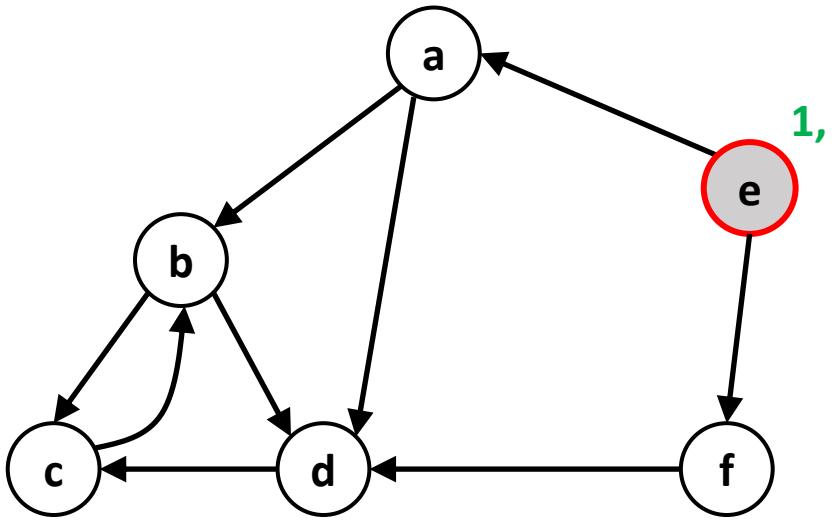
W	W	W	W	W	W
---	---	---	---	---	---

e a b c d f



DFS-Explore(G, e)

DFS(G)



Adj List of G :

a	→	b	→	d
b	→	d	→	c
c	→	b		
d	→	c		
e	→	a	→	f
f	→	d		

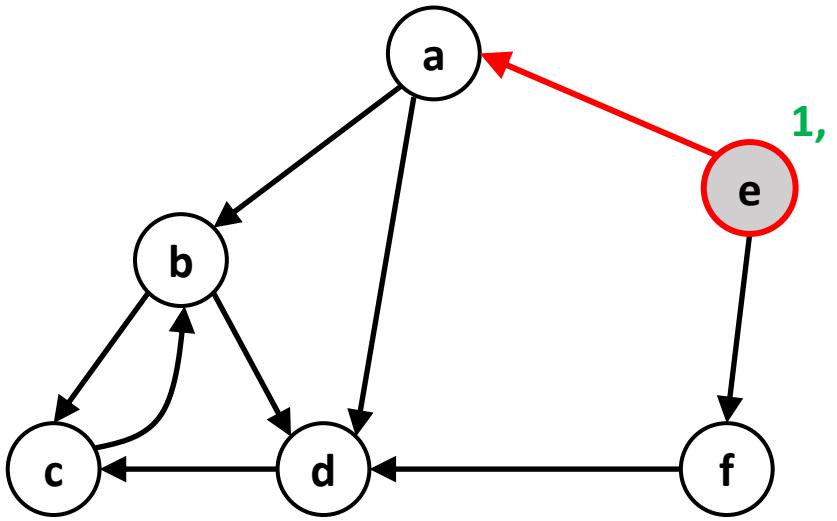
Colors of nodes:

G	W	W	W	W	W
e	a	b	c	d	f



DFS-Explore(G , e)

DFS(G)



Adj List of G :

a	→	b	→	d
b	→	d	→	c
c	→	b		
d	→	c		
e	→	a	→	f
f	→	d		

Colors of nodes:

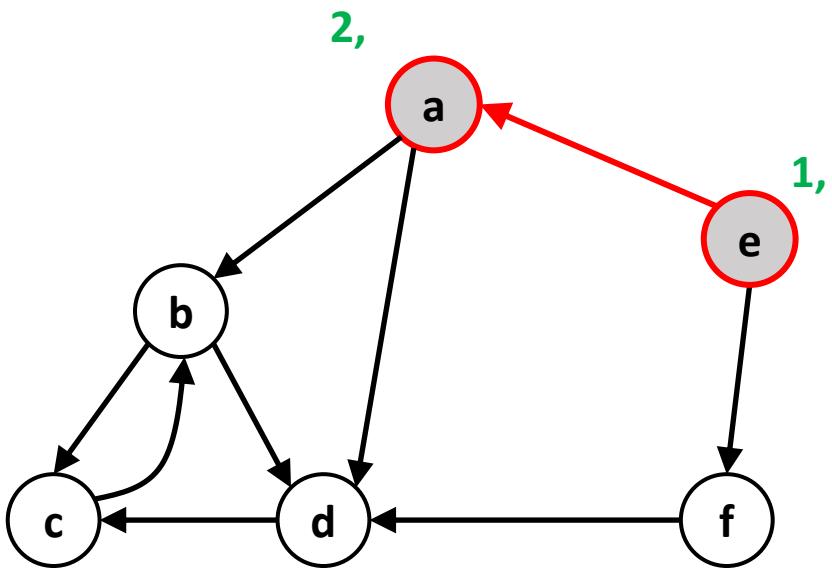
G	W	W	W	W	W
---	---	---	---	---	---

e a b c d f



DFS-Explore(G, e)

DFS(G)



Adj List of G :

a	→	b	→	d
b	→	d	→	c
c	→	b		
d	→	c		
e	→	a	→	f
f	→	d		

Colors of nodes:

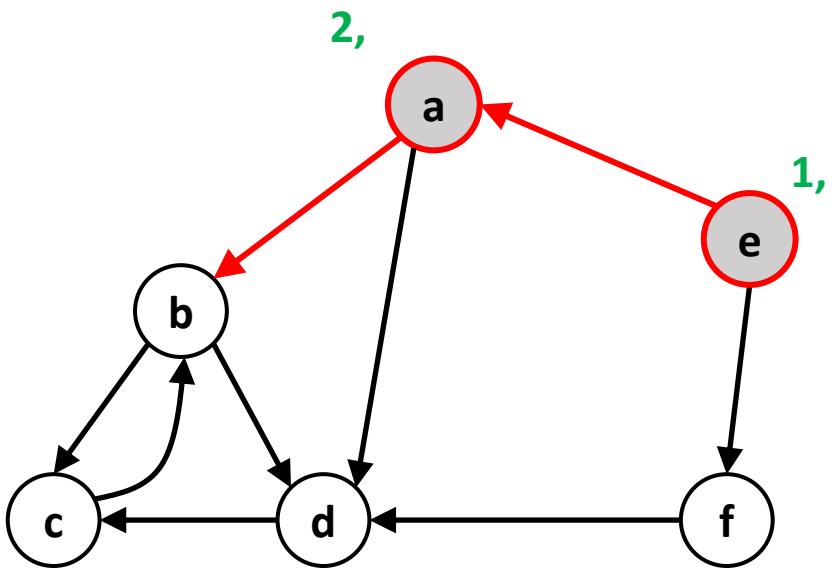
G	G	W	W	W	W
---	---	---	---	---	---

e a b c d f



DFS-Explore(G, e)

DFS(G)



Adj List of G :

a	→	b	→	d
b	→	d	→	c
c	→	b		
d	→	c		
e	→	a	→	f
f	→	d		

Colors of nodes:

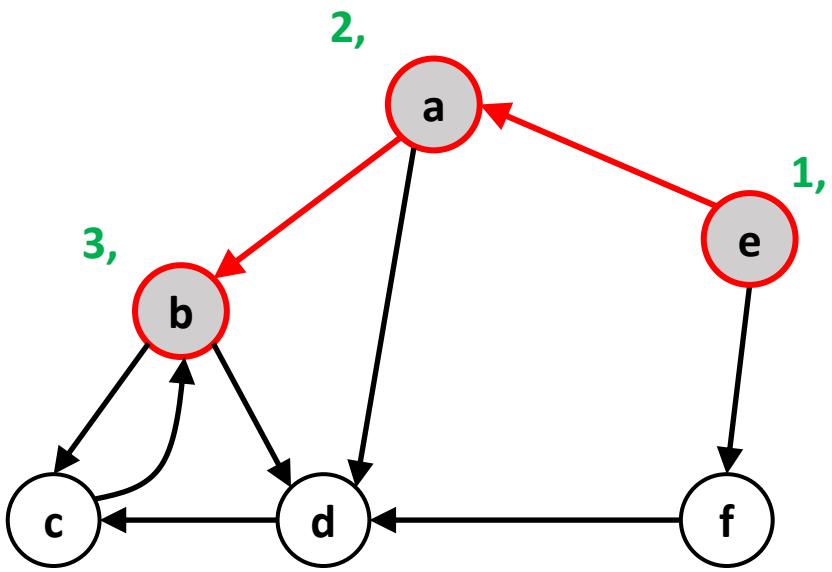
G	G	W	W	W	W
---	---	---	---	---	---

e a b c d f



DFS-Explore(G, e)

DFS(G)



Adj List of G :

a	→	b	→	d
b	→	d	→	c
c	→	b		
d	→	c		
e	→	a	→	f
f	→	d		

Colors of nodes:

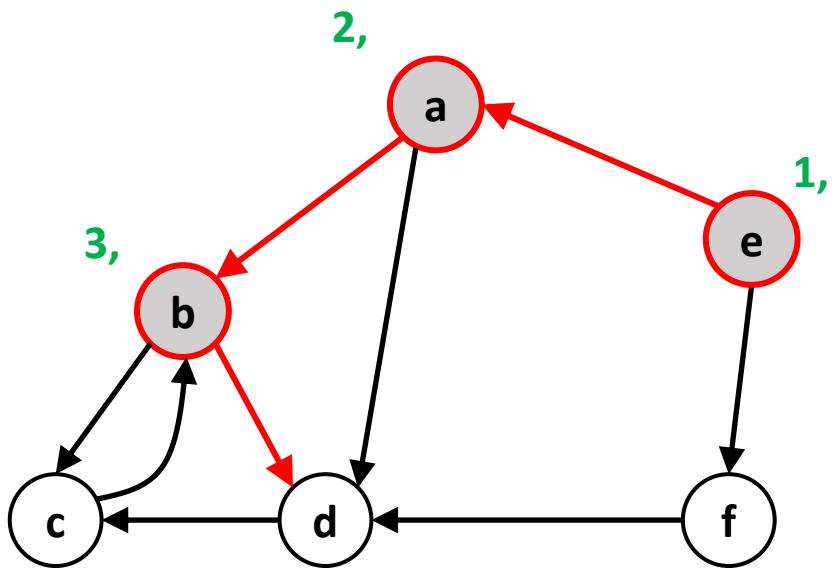
G	G	G	W	W	W
---	---	---	---	---	---

e a b c d f



DFS-Explore(G , e)

DFS(G)



Adj List of G :

a	→	b	→	d
b	→	d	→	c
c	→	b		
d	→	c		
e	→	a	→	f
f	→	d		

Colors of nodes:

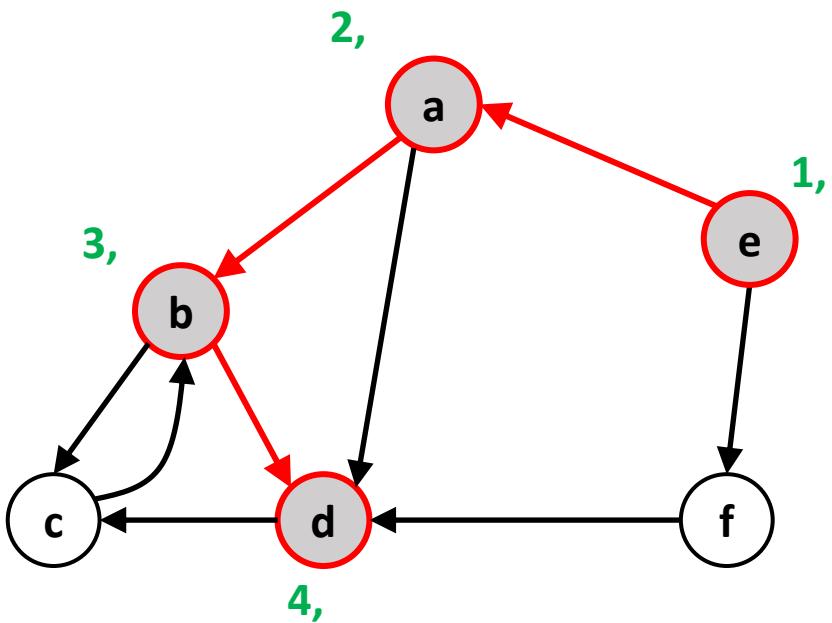
G	G	G	W	W	W
---	---	---	---	---	---

e a b c d f



DFS-Explore(G, e)

DFS(G)



Adj List of G :

a	→	b	→	d
b	→	d	→	c
c	→	b		
d	→	c		
e	→	a	→	f
f	→	d		

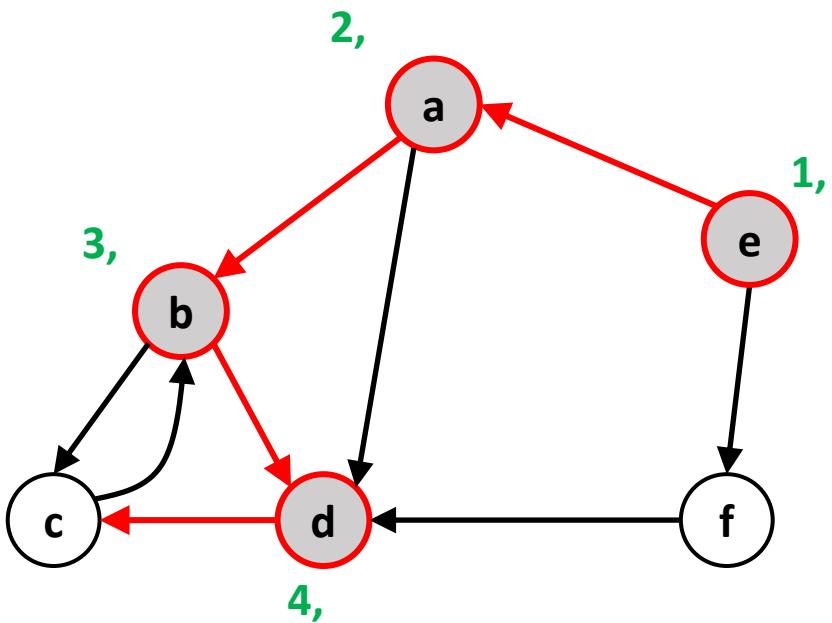
Colors of nodes:

G	G	G	W	G	W
---	---	---	---	---	---

↑

DFS-Explore(G, e)

DFS(G)



Adj List of G :

a	→	b	→	d
b	→	d	→	c
c	→	b		
d	→	c		
e	→	a	→	f
f	→	d		

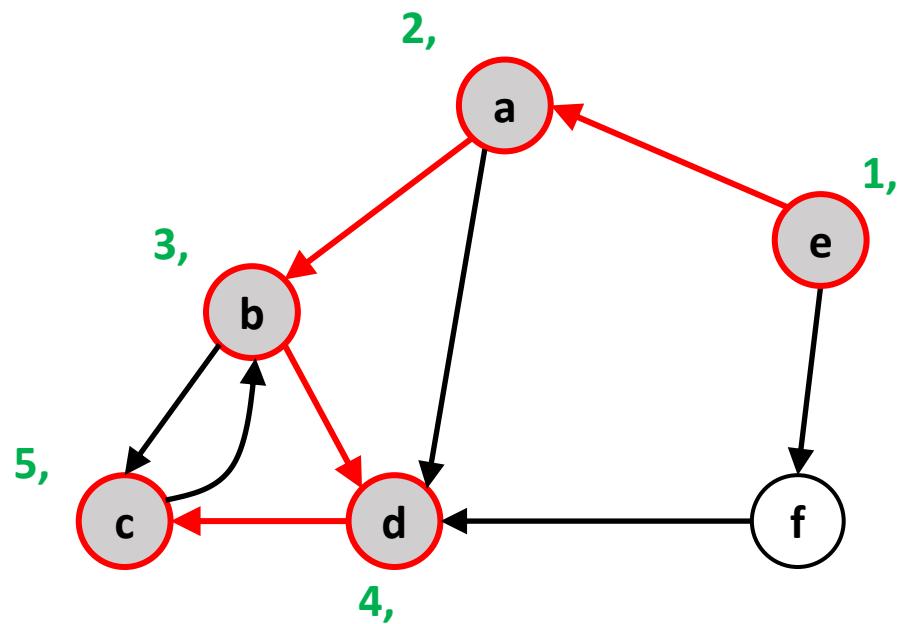
Colors of nodes:

G	G	G	W	G	W
---	---	---	---	---	---

↑

DFS-Explore(G, e)

DFS(G)



Adj List of G :

a	→	b	→	d
b	→	d	→	c
c	→	b		
d	→	c		
e	→	a	→	f
f	→	d		

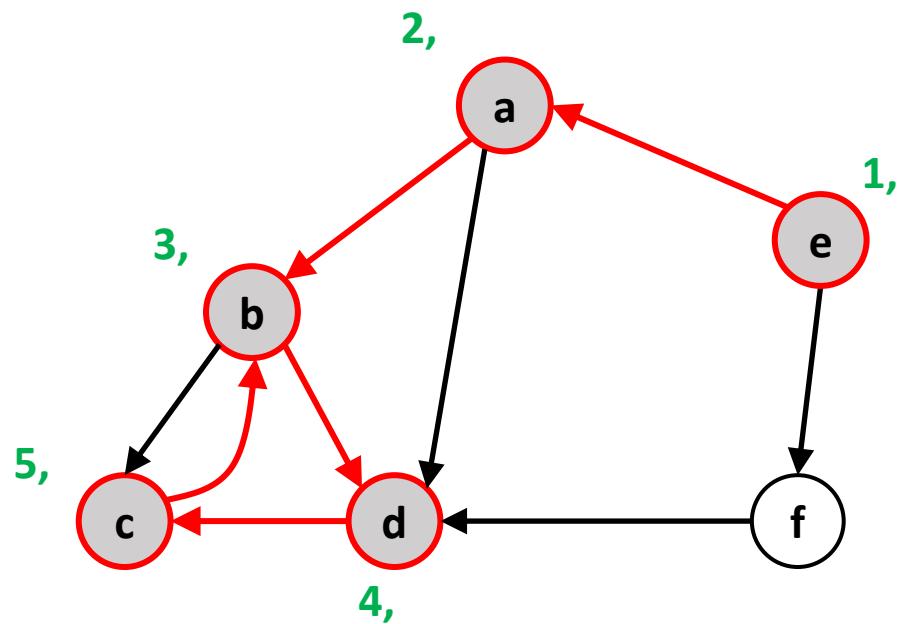
Colors of nodes:

G	G	G	G	G	W
---	---	---	---	---	---



DFS-Explore(G , e)

DFS(G)



Adj List of G :

a	→	b	→	d
b	→	d	→	c
c	→	b		
d	→	c		
e	→	a	→	f
f	→	d		

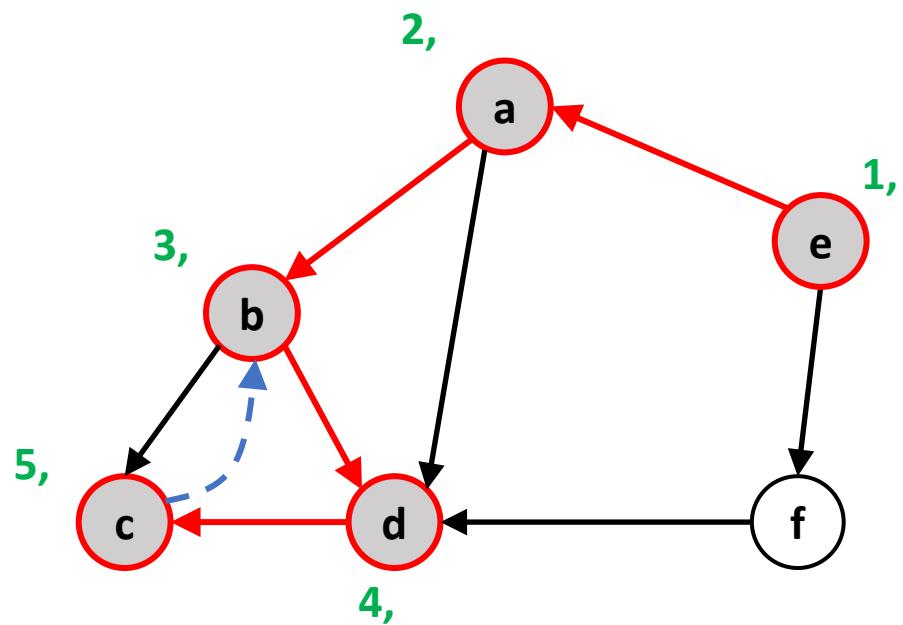
Colors of nodes:

G	G	G	G	G	W
---	---	---	---	---	---



DFS-Explore(G, e)

DFS(G)



Adj List of G :

a	→	b	→	d
b	→	d	→	c
c	→	b		
d	→	c		
e	→	a	→	f
f	→	d		

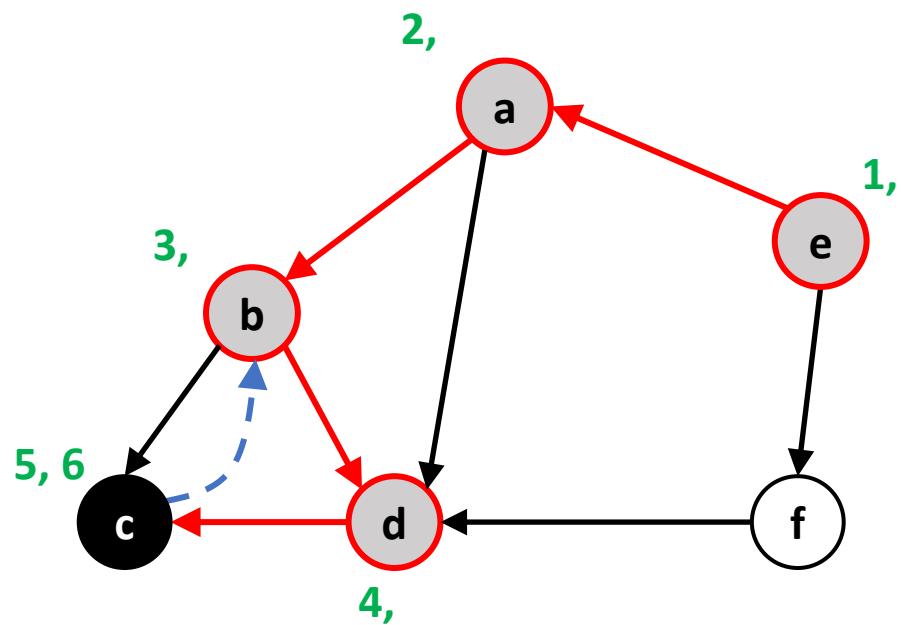
Colors of nodes:

G	G	G	G	G	W
---	---	---	---	---	---



DFS-Explore(G, e)

DFS(G)



Adj List of G :

a	→	b	→	d
b	→	d	→	c
c	→	b		
d	→	c		
e	→	a	→	f
f	→	d		

Colors of nodes:

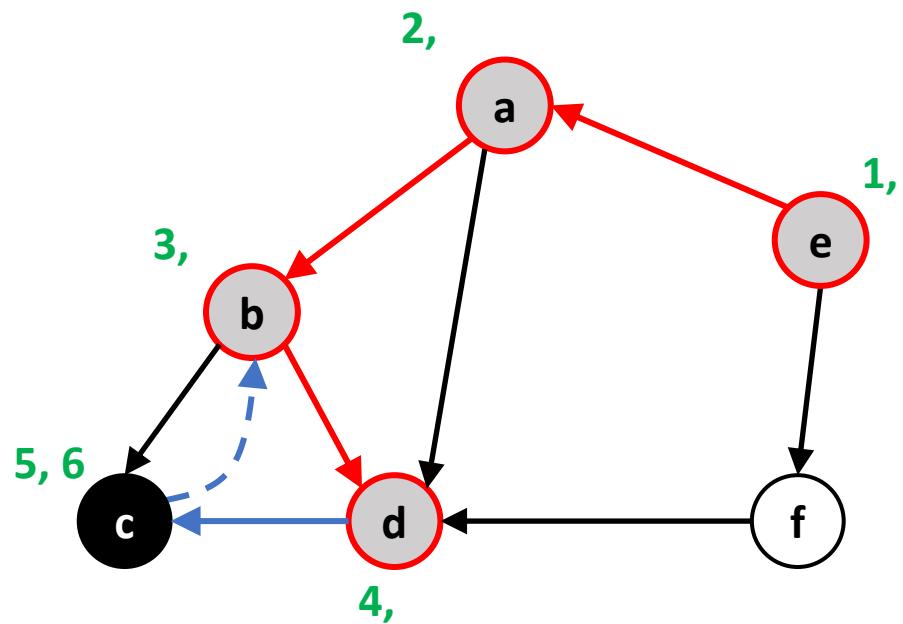
G	G	G	B	G	W
---	---	---	---	---	---

e a b c d f



DFS-Explore(G, e)

DFS(G)



Adj List of G :

a	→	b	→	d
b	→	d	→	c
c	→	b		
d	→	c		
e	→	a	→	f
f	→	d		

Colors of nodes:

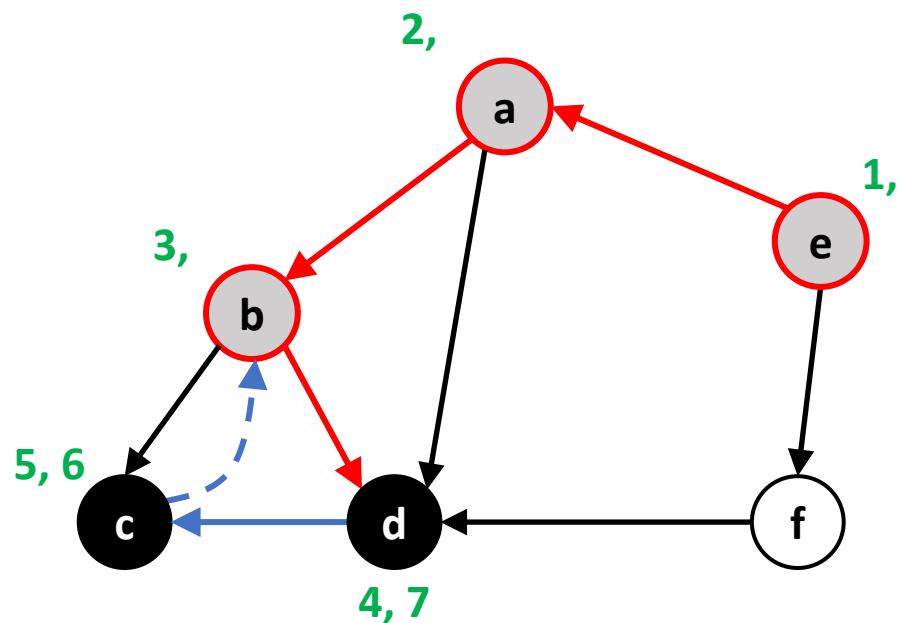
G	G	G	B	G	W
---	---	---	---	---	---

e a b c d f



DFS-Explore(G, e)

DFS(G)



Adj List of G :

a	→	b	→	d
b	→	d	→	c
c	→	b		
✓ d	→	c		
e	→	a	→	f
f	→	d		

Colors of nodes:

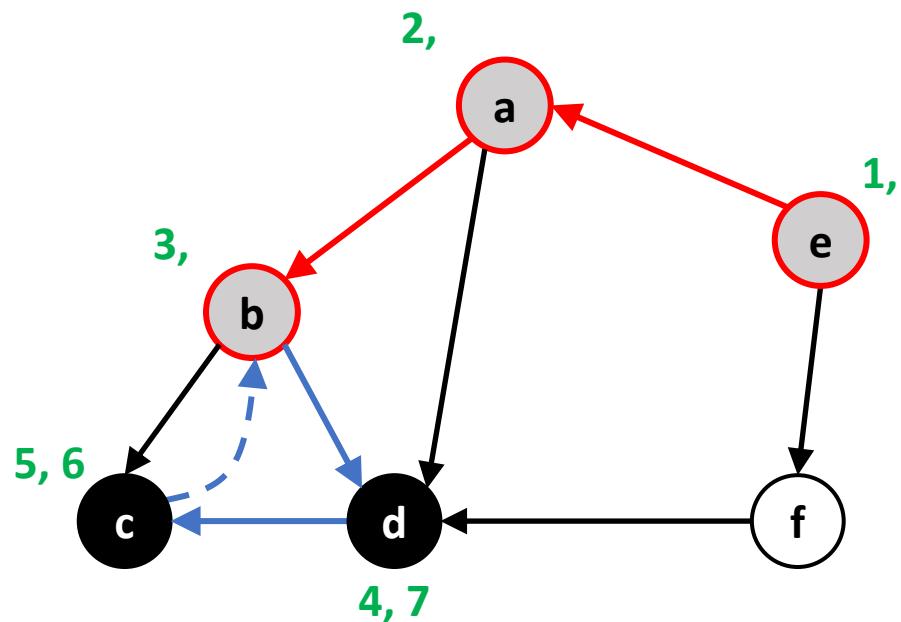
G	G	G	B	B	W
---	---	---	---	---	---

e a b c d f



DFS-Explore(G, e)

DFS(G)



Adj List of G :

a	→	b	→	d
b	→	d	→	c
✓	c	→	b	
✓	d	→	c	
e	→	a	→	f
f	→	d		

Colors of nodes:

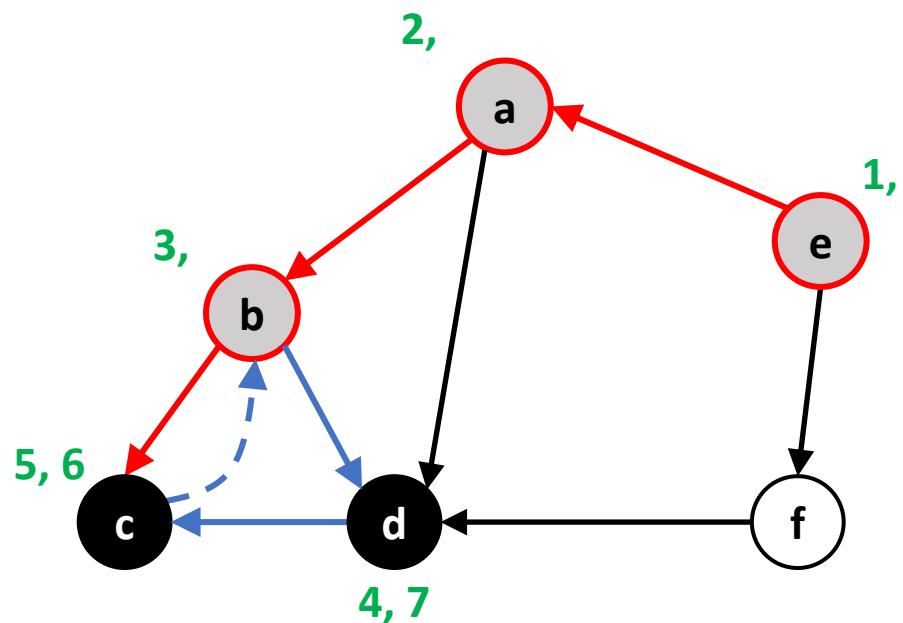
G	G	G	B	B	W
---	---	---	---	---	---

e a b c d f



DFS-Explore(G, e)

DFS(G)



Adj List of G :

a	→	b	→	d
b	→	d	→	c
✓	c	→	b	
✓	d	→	c	
e	→	a	→	f
f	→	d		

Colors of nodes:

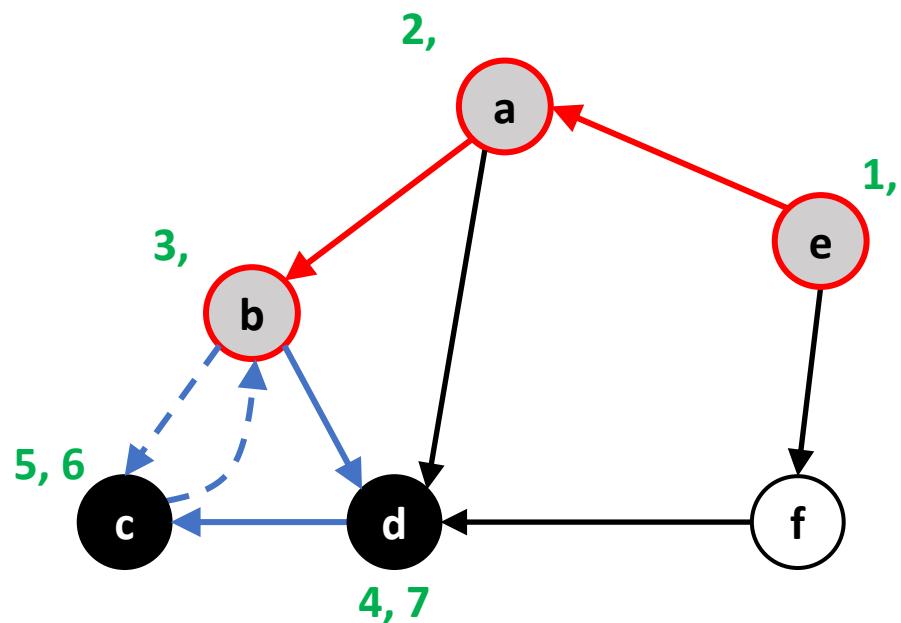
G	G	G	B	B	W
---	---	---	---	---	---

e a b c d f



DFS-Explore(G, e)

DFS(G)



Adj List of G :

a	→	b	→	d
b	→	d	→	c
✓	c	→	b	
✓	d	→	c	
e	→	a	→	f
f	→	d		

Colors of nodes:

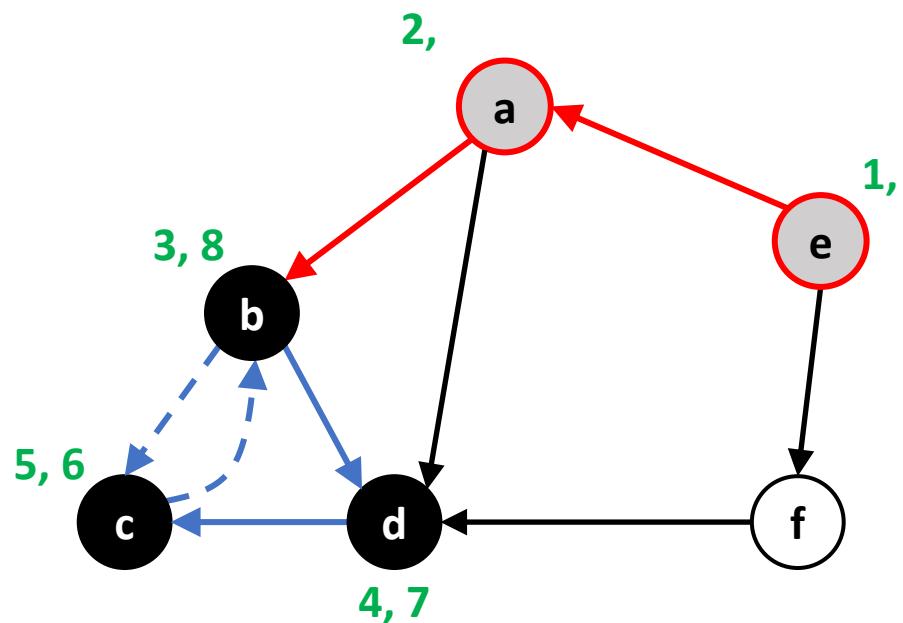
G	G	G	B	B	W
---	---	---	---	---	---

e a b c d f



DFS-Explore(G, e)

DFS(G)



Adj List of G :

a	→	b	→	d
✓ b	→	d	→	c
✓ c	→	b		
✓ d	→	c		
e	→	a	→	f
f	→	d		

Colors of nodes:

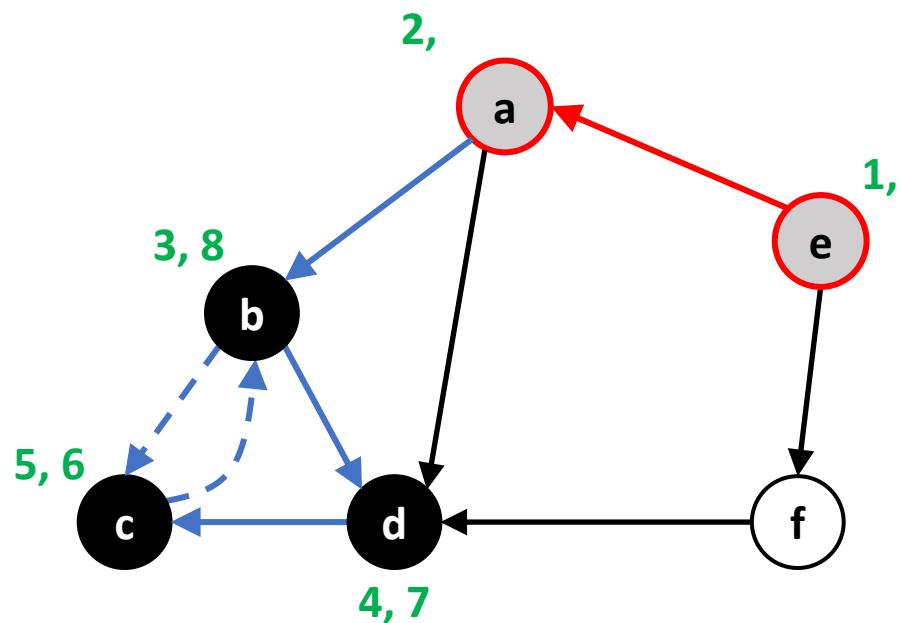
G	G	B	B	B	W
---	---	---	---	---	---

e a b c d f



DFS-Explore(G, e)

DFS(G)



Adj List of G :

a	→	b	→	d
✓ b	→	d	→	c
✓ c	→	b		
✓ d	→	c		
e	→	a	→	f
f	→	d		

Colors of nodes:

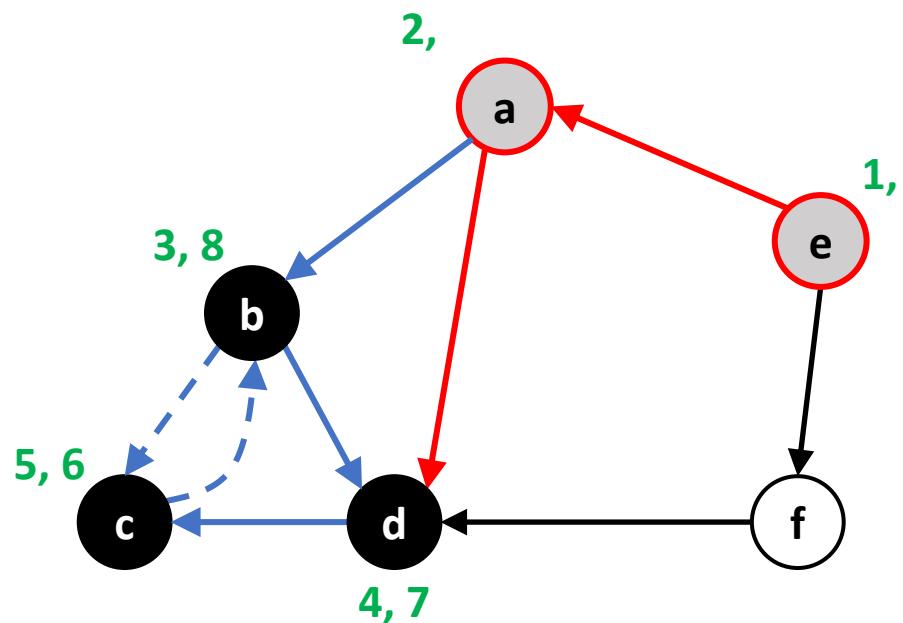
G	G	B	B	B	W
---	---	---	---	---	---

e a b c d f



DFS-Explore(G, e)

DFS(G)



Adj List of G :

a	→	b	→	d
✓ b	→	d	→	c
✓ c	→	b		
✓ d	→	c		
e	→	a	→	f
f	→	d		

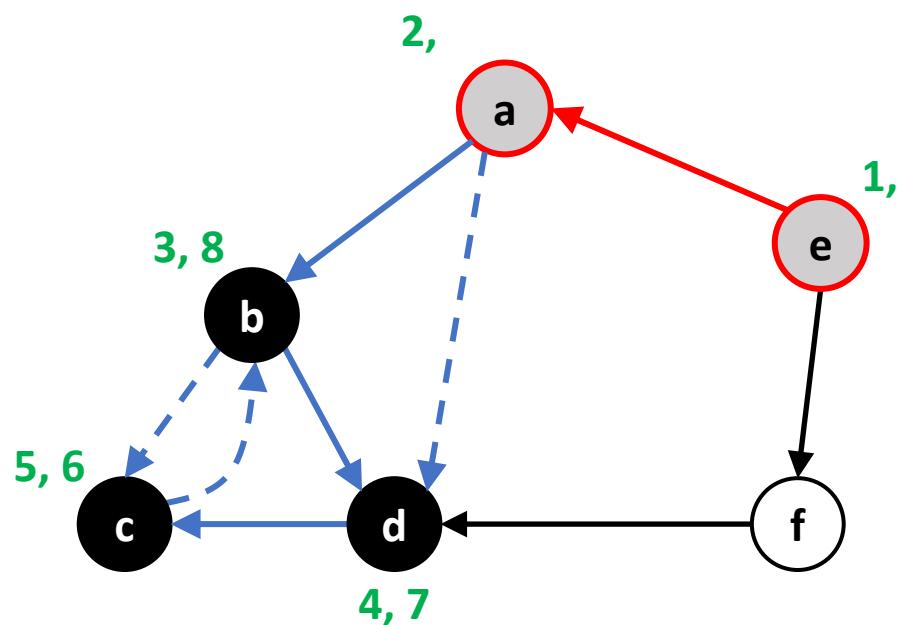
Colors of nodes:

G	G	B	B	B	W
---	---	---	---	---	---



DFS-Explore(G , e)

DFS(G)



Adj List of G :

a	→	b	→	d
✓ b	→	d	→	c
✓ c	→	b		
✓ d	→	c		
e	→	a	→	f
f	→	d		

Colors of nodes:

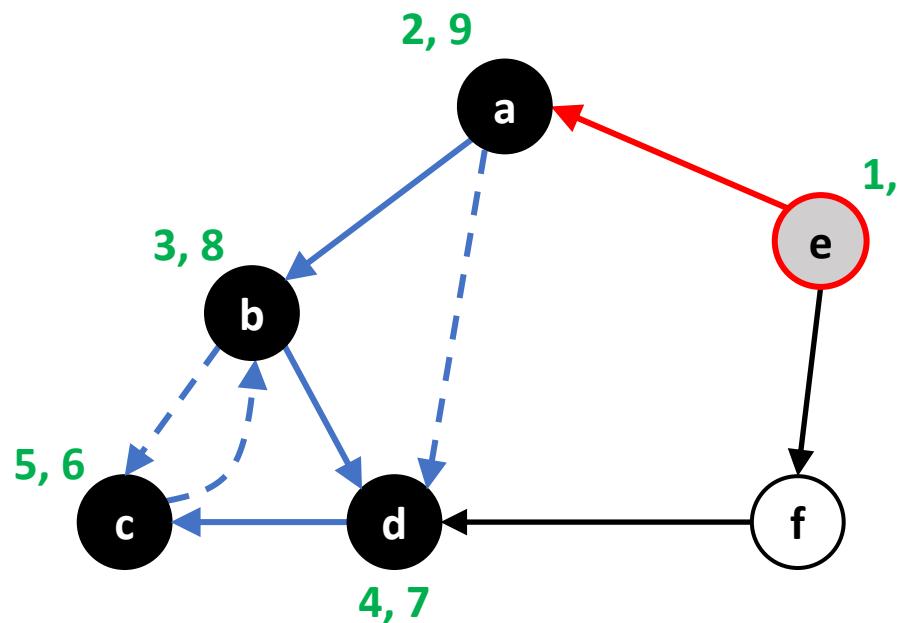
G	G	B	B	B	W
---	---	---	---	---	---

e a b c d f



DFS-Explore(G, e)

DFS(G)



Adj List of G :

✓ a	→ b	→ d
✓ b	→ d	→ c
✓ c	→ b	
✓ d	→ c	
e	→ a	→ f
f	→ d	

Colors of nodes:

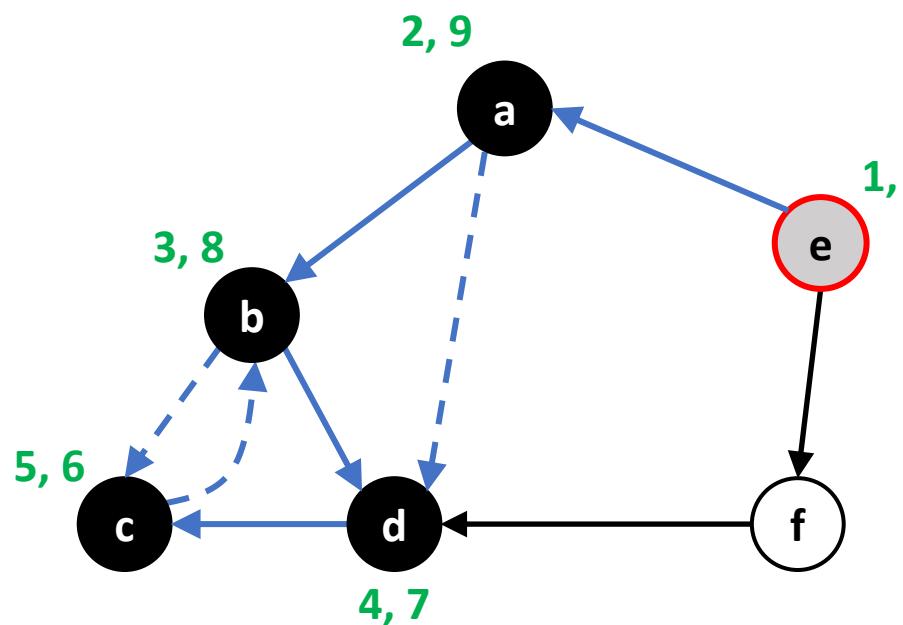
G	B	B	B	B	W
---	---	---	---	---	---

e a b c d f



DFS-Explore(G, e)

DFS(G)



Adj List of G :

✓ a	→ b	→ d
✓ b	→ d	→ c
✓ c	→ b	
✓ d	→ c	
e	→ a	→ f
f	→ d	

Colors of nodes:

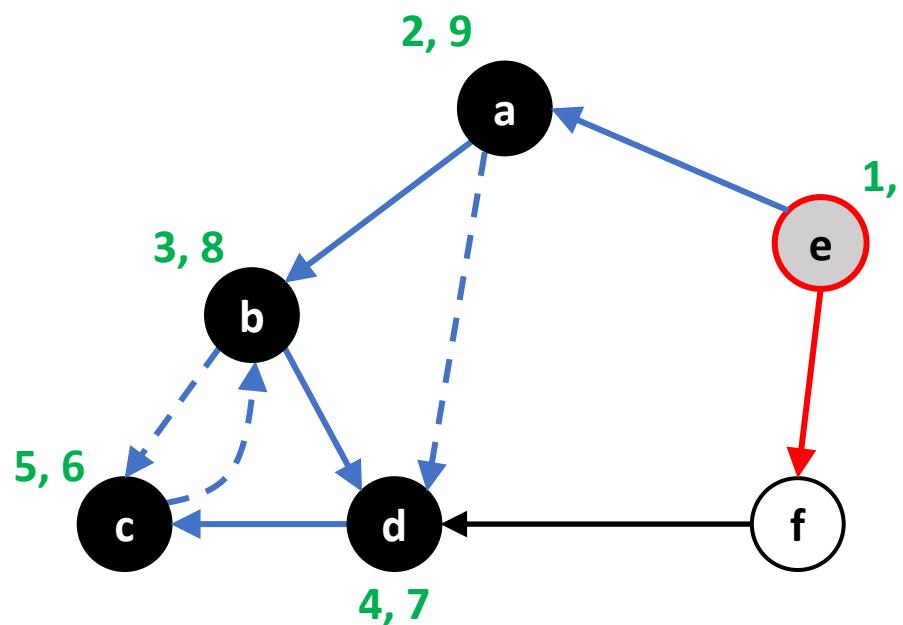
G	B	B	B	B	W
---	---	---	---	---	---

e a b c d f



DFS-Explore(G , e)

DFS(G)



Adj List of G :

✓ a	→ b	→ d
✓ b	→ d	→ c
✓ c	→ b	
✓ d	→ c	
e	→ a	→ f
f		→ d

Colors of nodes:

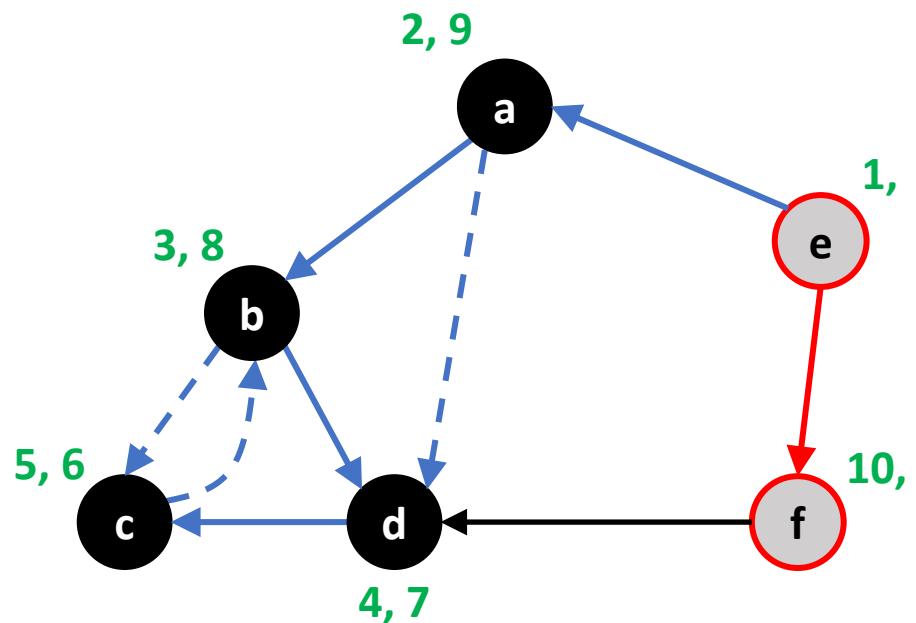
G	B	B	B	B	W
---	---	---	---	---	---

e a b c d f



DFS-Explore(G, e)

DFS(G)



Adj List of G :

✓ a	→ b	→ d
✓ b	→ d	→ c
✓ c	→ b	
✓ d	→ c	
e	→ a	→ f
f	→ d	

Colors of nodes:

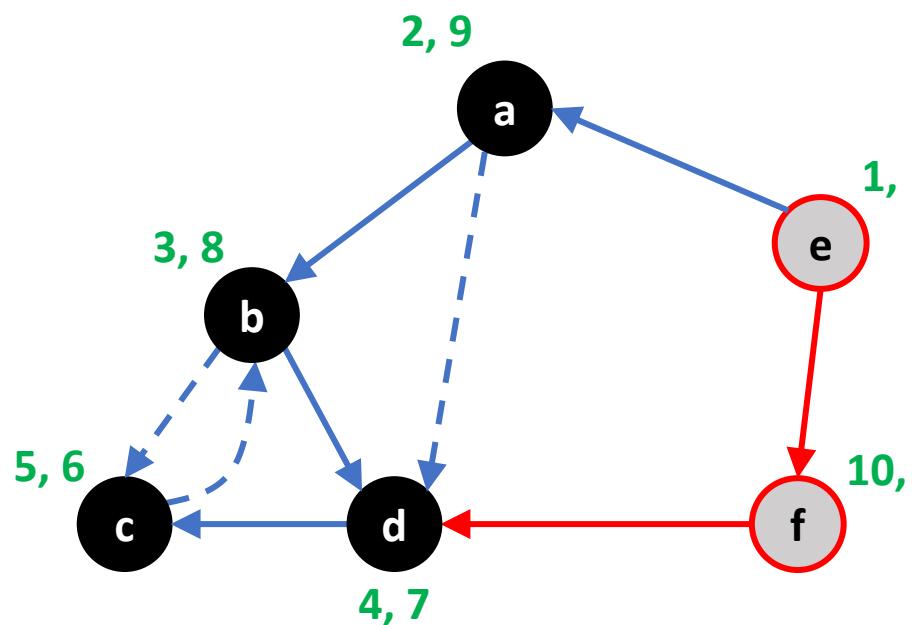
G	B	B	B	B	G
---	---	---	---	---	---

e a b c d f



DFS-Explore(G , e)

DFS(G)



Adj List of G :

✓ a	→ b	→ d
✓ b	→ d	→ c
✓ c	→ b	
✓ d	→ c	
e	→ a	→ f
f	→ d	

Colors of nodes:

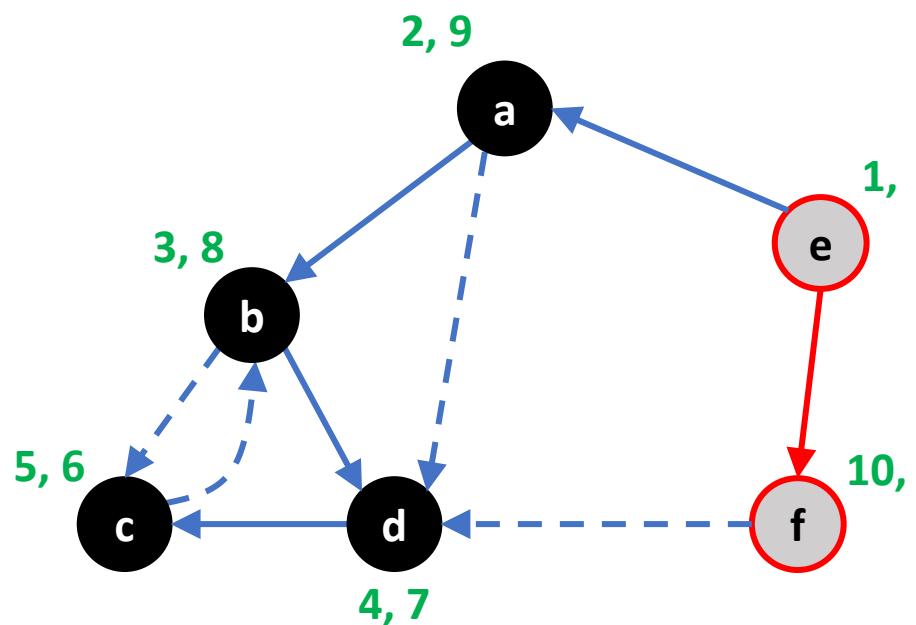
G	B	B	B	B	G
---	---	---	---	---	---

e a b c d f



DFS-Explore(G , e)

DFS(G)



Adj List of G :

✓ a	→ b	→ d
✓ b	→ d	→ c
✓ c	→ b	
✓ d	→ c	
e	→ a	→ f
f	→ d	

Colors of nodes:

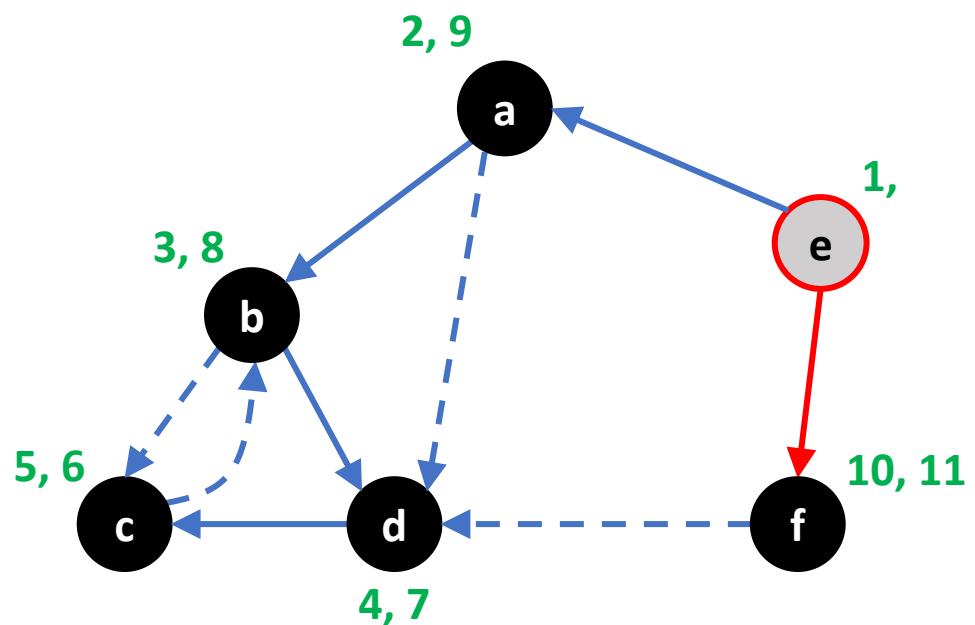
G	B	B	B	B	G
---	---	---	---	---	---

e a b c d f



DFS-Explore(G, e)

DFS(G)



Adj List of G :

✓ a	→ b	→ d
✓ b	→ d	→ c
✓ c	→ b	
✓ d	→ c	
e	→ a	→ f
✓ f	→ d	

Colors of nodes:

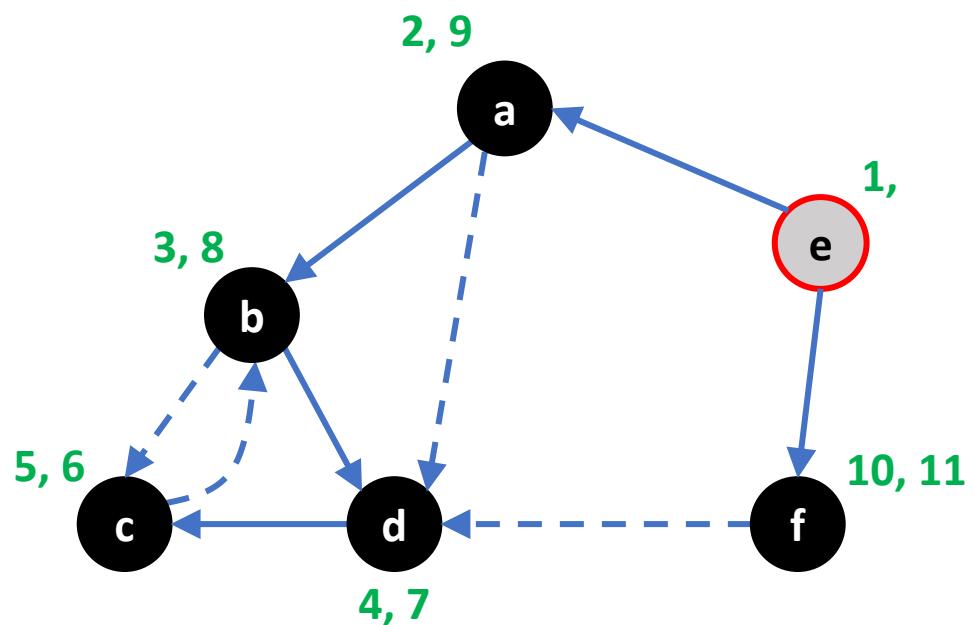
G	B	B	B	B	B
---	---	---	---	---	---

e a b c d f



DFS-Explore(G, e)

DFS(G)



Adj List of G :

✓ a	→ b	→ d
✓ b	→ d	→ c
✓ c	→ b	
✓ d	→ c	
e	→ a	→ f
✓ f	→ d	

Colors of nodes:

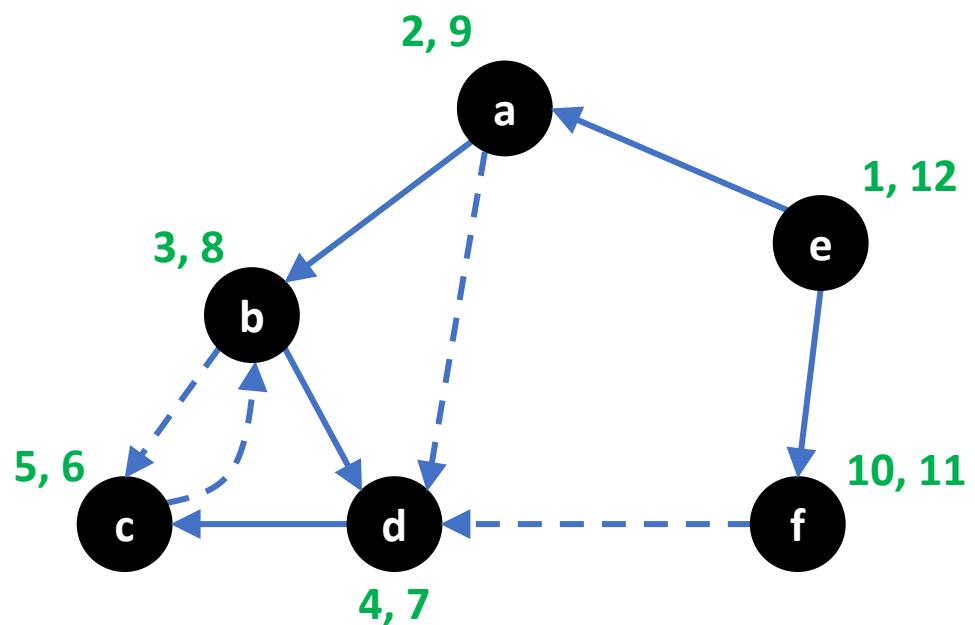
G	B	B	B	B	B
---	---	---	---	---	---

e a b c d f

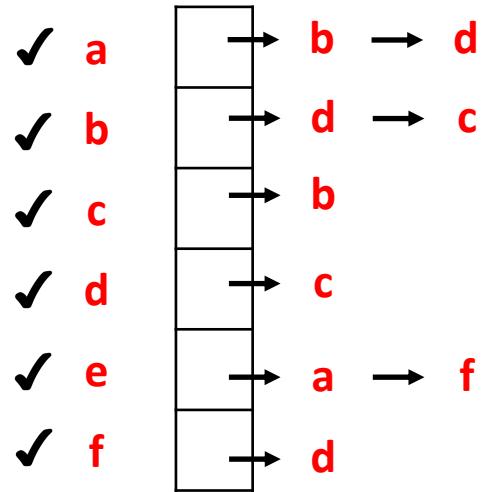


DFS-Explore(G , e)

DFS(G)



Adj List of G :



Colors of nodes:

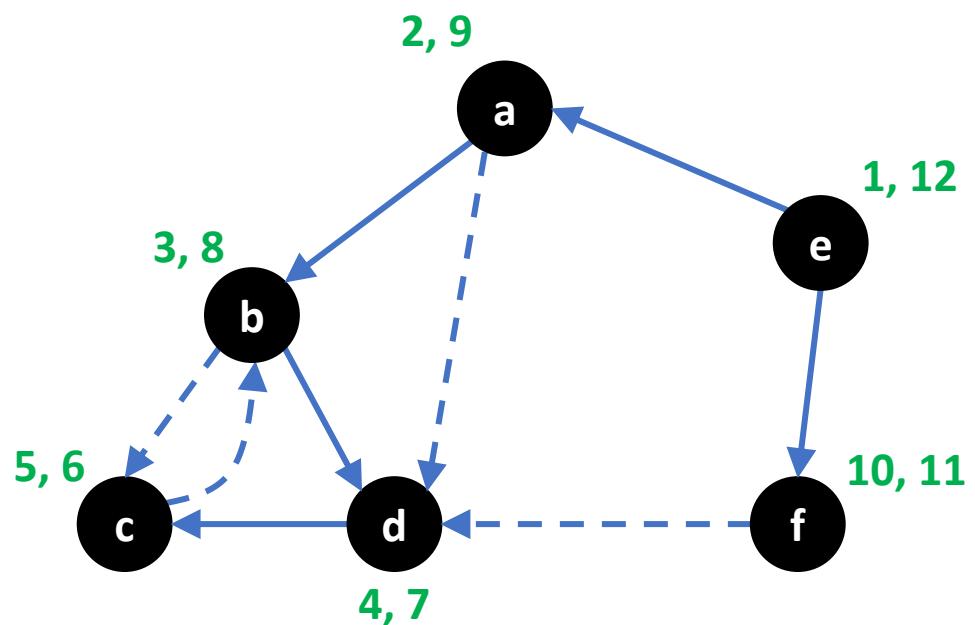
B	B	B	B	B	B
---	---	---	---	---	---

e a b c d f



DFS-Explore(G, e)

DFS(G)



Adj List of G :

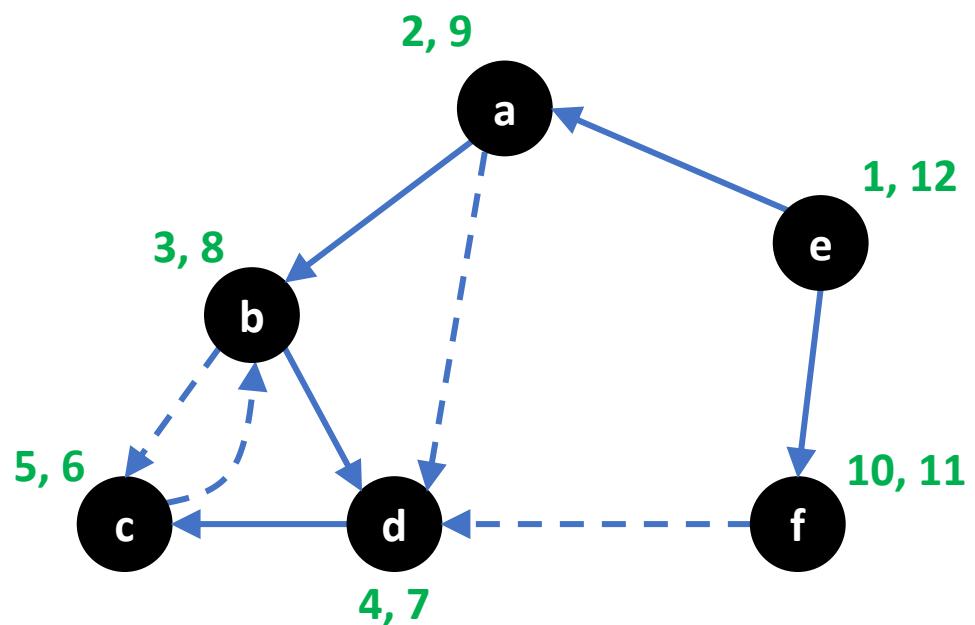
✓ a	→ b	→ d
✓ b	→ d	→ c
✓ c	→ b	
✓ d	→ c	
✓ e	→ a	→ f
✓ f	→ d	

Colors of nodes:

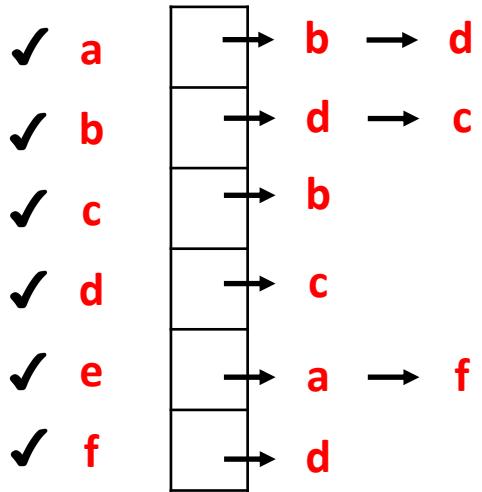
B	B	B	B	B	B
e	a	b	c	d	f

↑

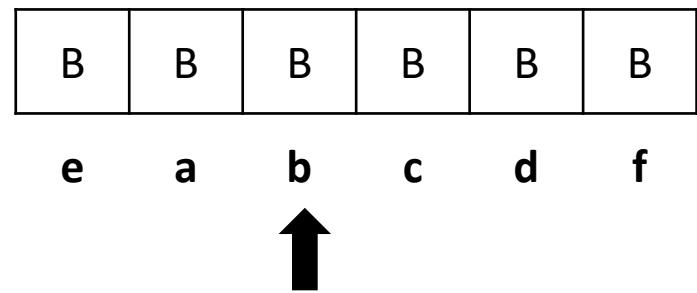
DFS(G)



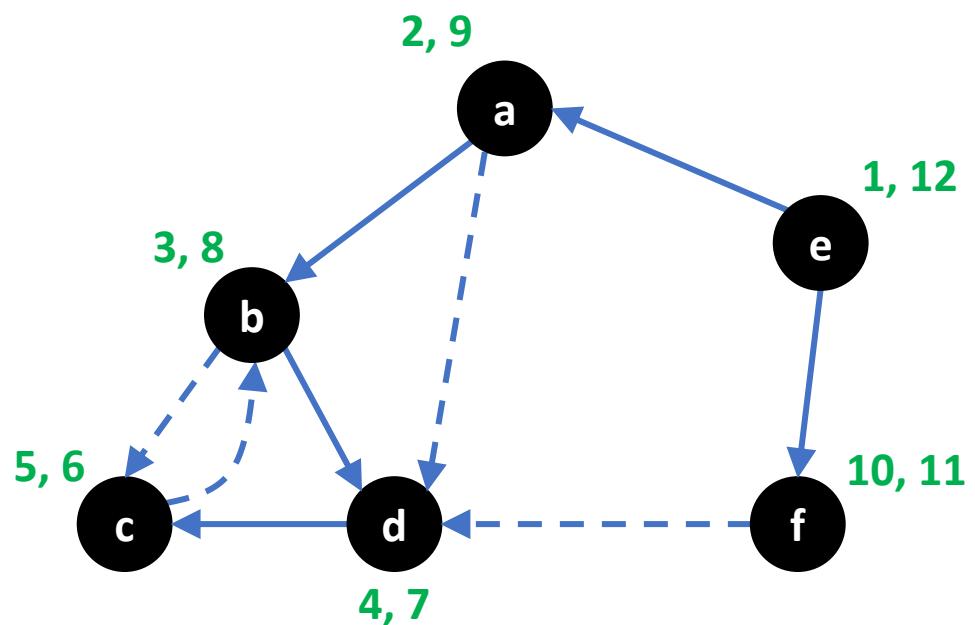
Adj List of G :



Colors of nodes:



DFS(G)



Adj List of G :

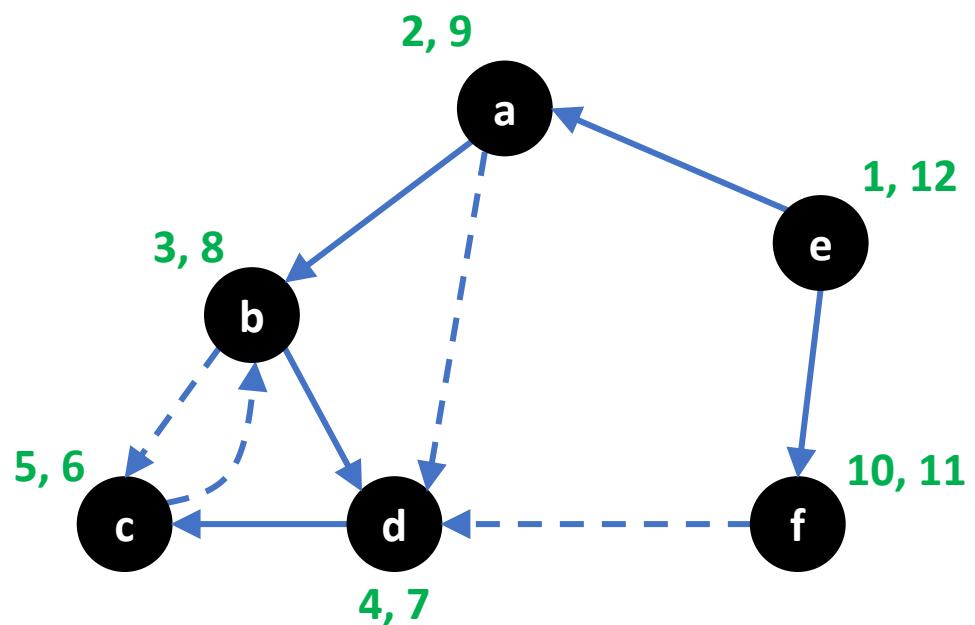
✓ a	→ b	→ d
✓ b	→ d	→ c
✓ c	→ b	
✓ d	→ c	
✓ e	→ a	→ f
✓ f	→ d	

Colors of nodes:

B	B	B	B	B	B
e	a	b	c	d	f

↑

DFS(G)



Adj List of G :

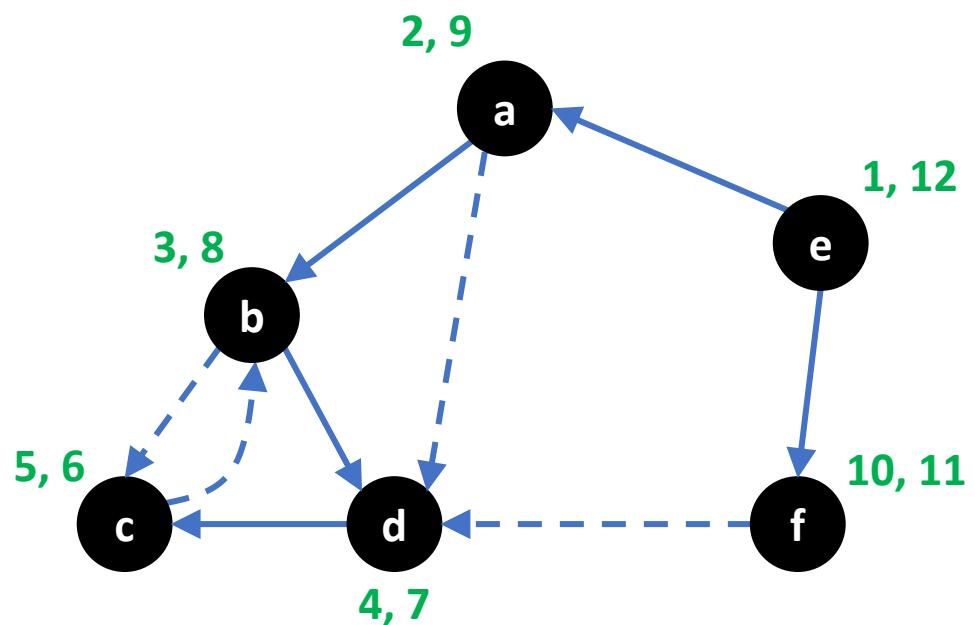
✓ a	→ b	→ d
✓ b	→ d	→ c
✓ c	→ b	
✓ d	→ c	
✓ e	→ a	→ f
✓ f	→ d	

Colors of nodes:

B	B	B	B	B	B
e	a	b	c	d	f

↑

DFS(G)



Adj List of G :

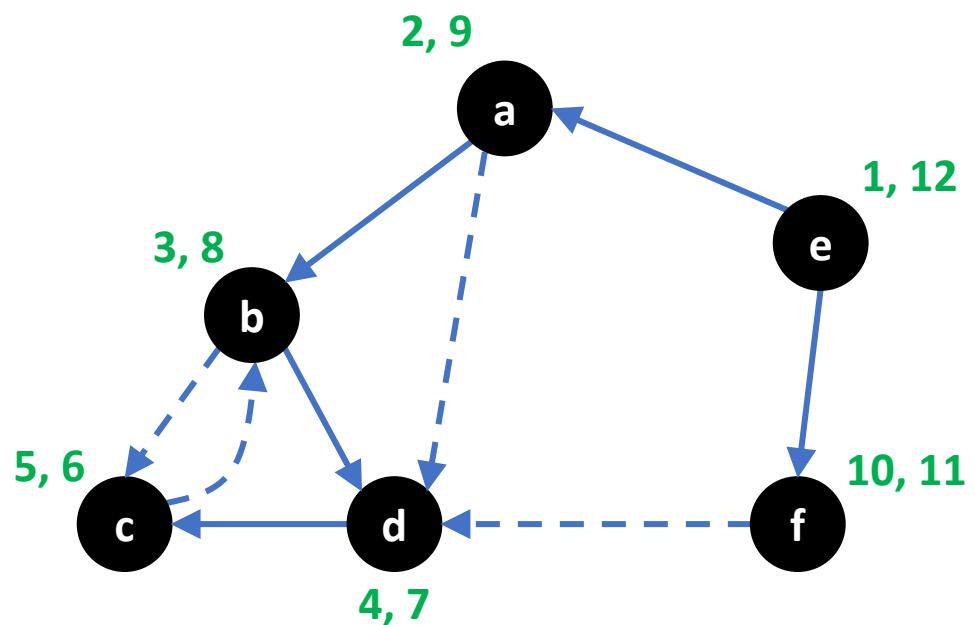
✓ a	→ b	→ d
✓ b	→ d	→ c
✓ c	→ b	
✓ d	→ c	
✓ e	→ a	→ f
✓ f	→ d	

Colors of nodes:

B	B	B	B	B	B
e	a	b	c	d	f

↑

DFS(G)



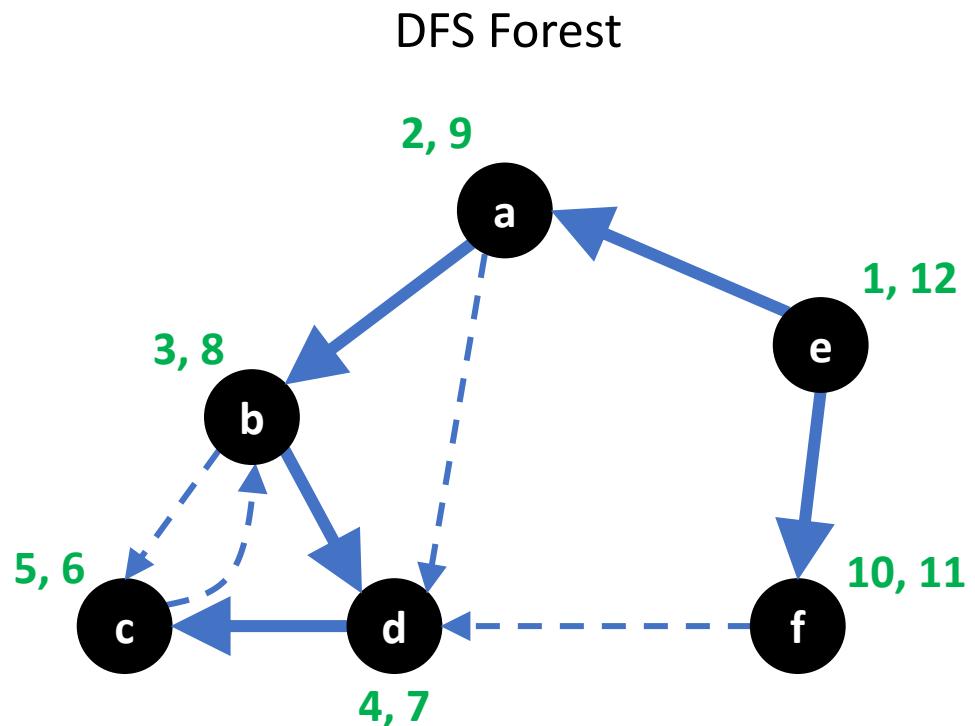
Adj List of G :

✓ a	→ b	→ d
✓ b	→ d	→ c
✓ c	→ b	
✓ d	→ c	
✓ e	→ a	→ f
✓ f	→ d	

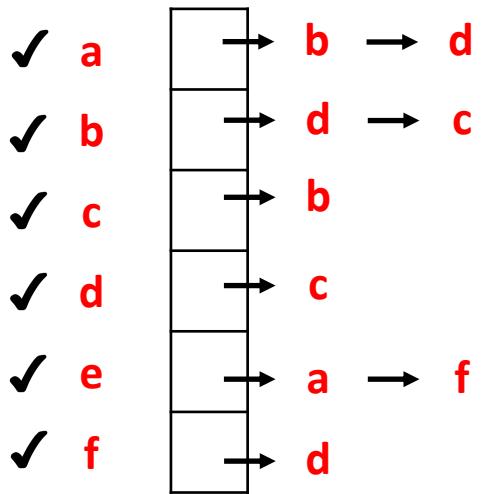
Colors of nodes:

B	B	B	B	B	B
e	a	b	c	d	f

DFS(G)



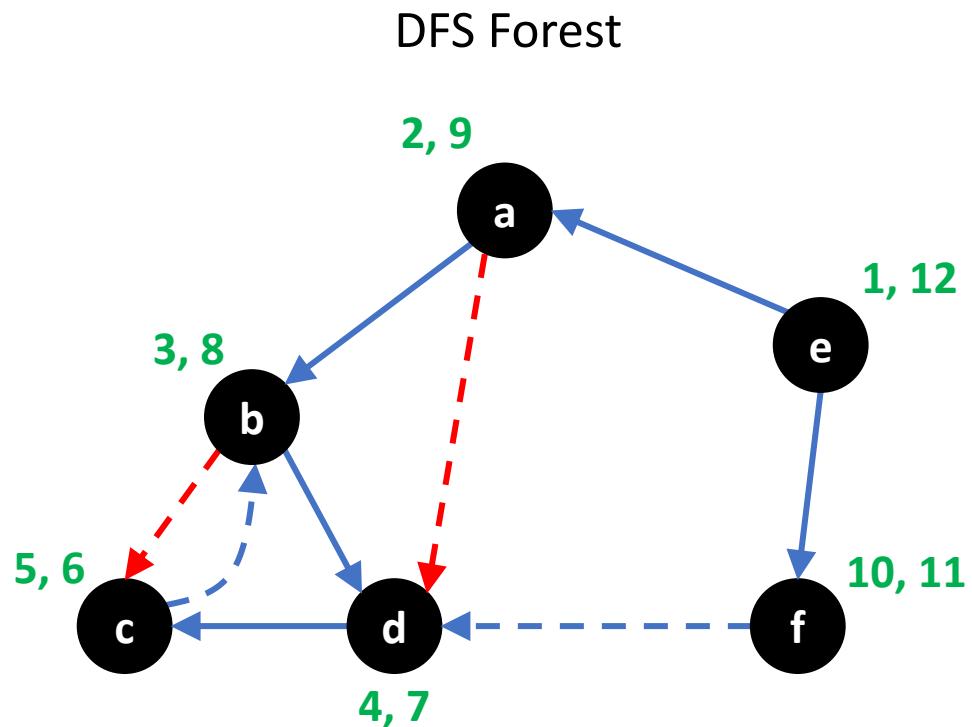
Adj List of G :



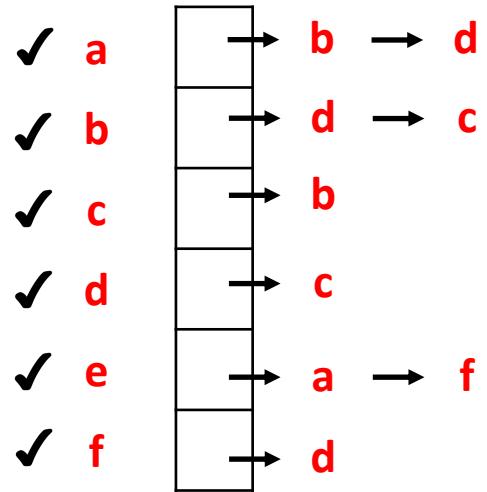
Colors of nodes:

B	B	B	B	B	B
e	a	b	c	d	f

DFS(G)



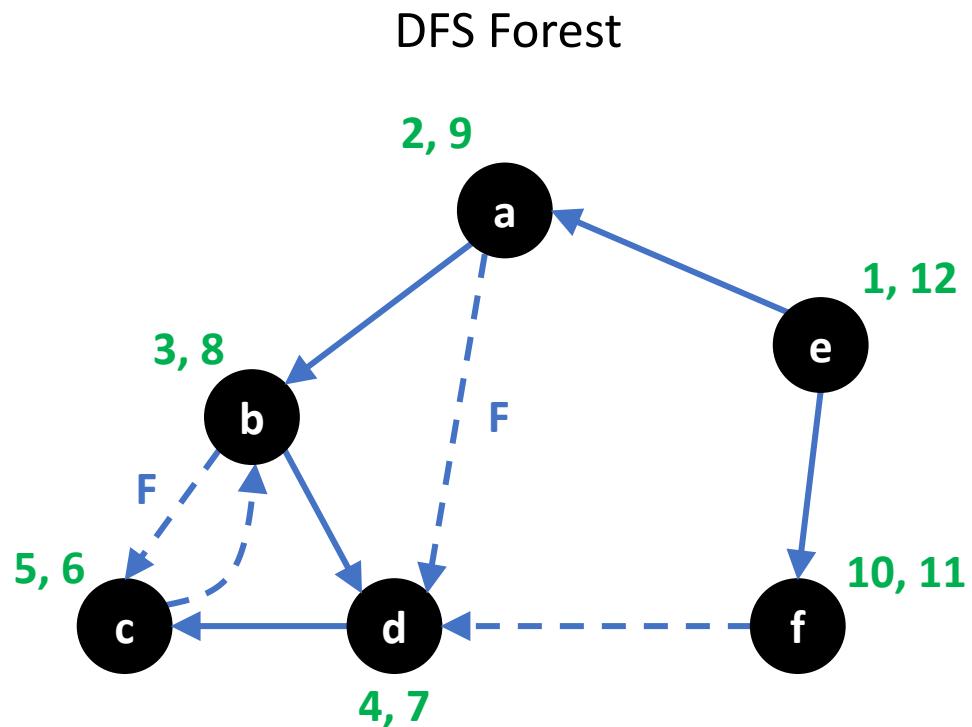
Adj List of G :



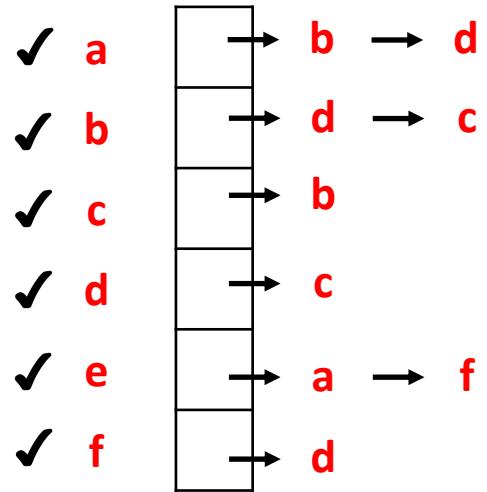
Colors of nodes:

B	B	B	B	B	B
e	a	b	c	d	f

DFS(G)



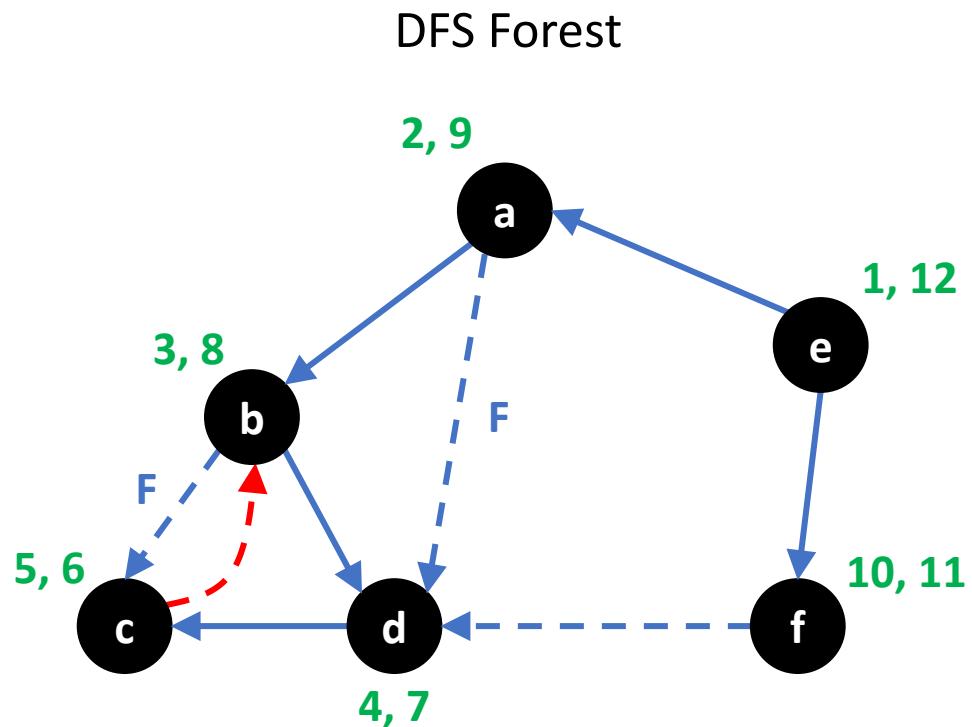
Adj List of G :



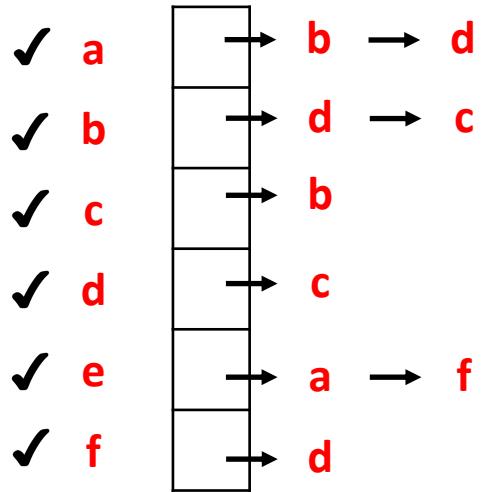
Colors of nodes:

B	B	B	B	B	B
e	a	b	c	d	f

DFS(G)



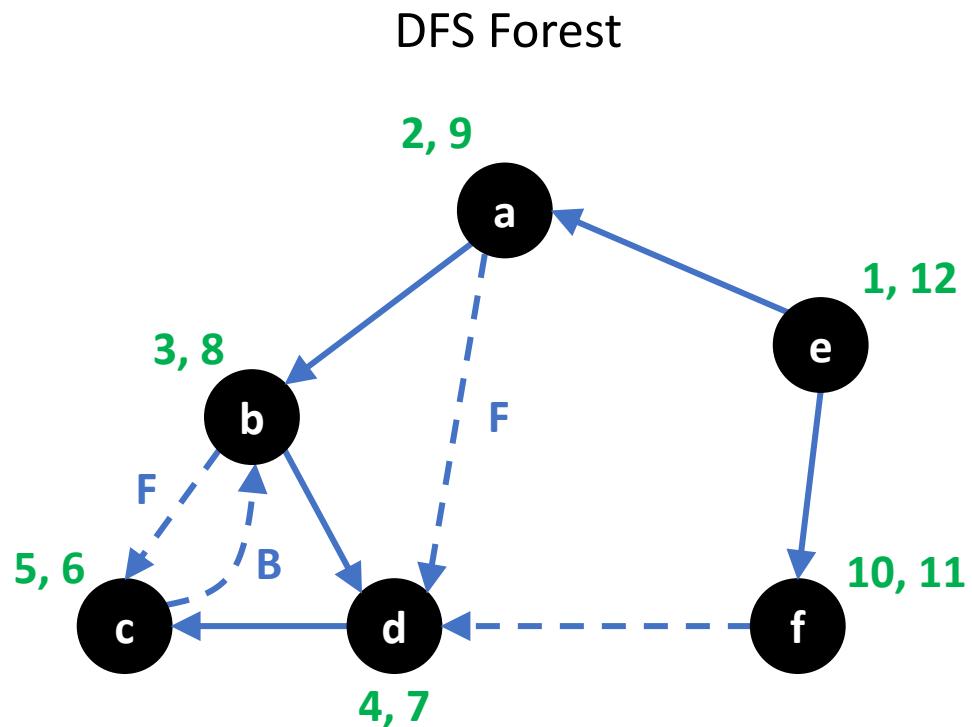
Adj List of G :



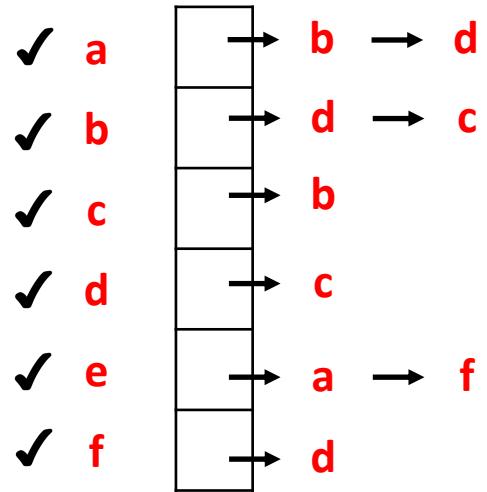
Colors of nodes:

B	B	B	B	B	B
e	a	b	c	d	f

DFS(G)



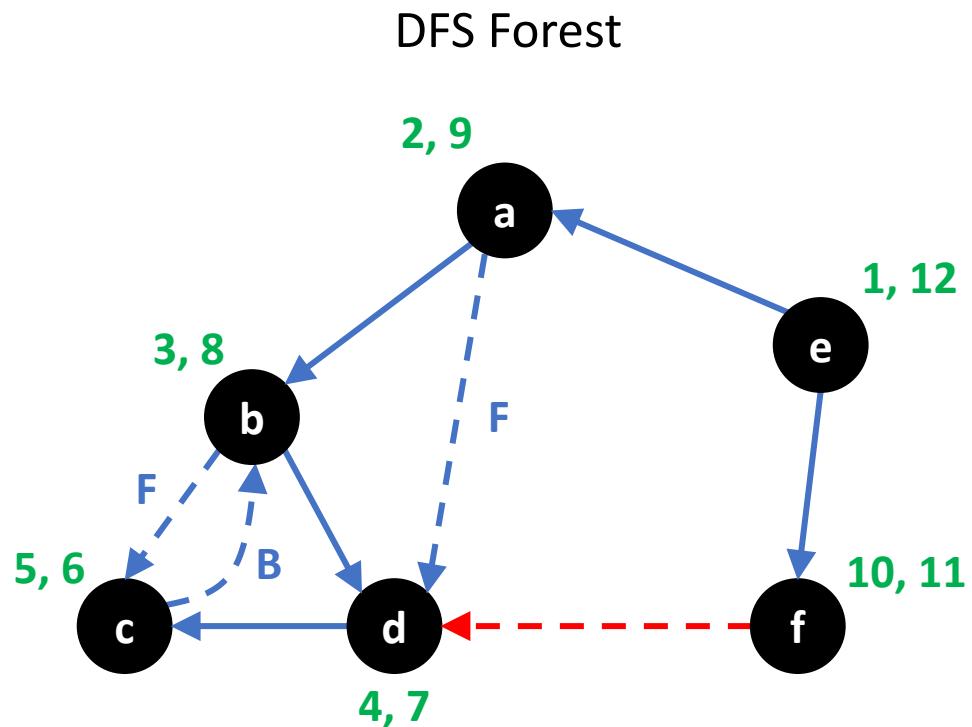
Adj List of G :



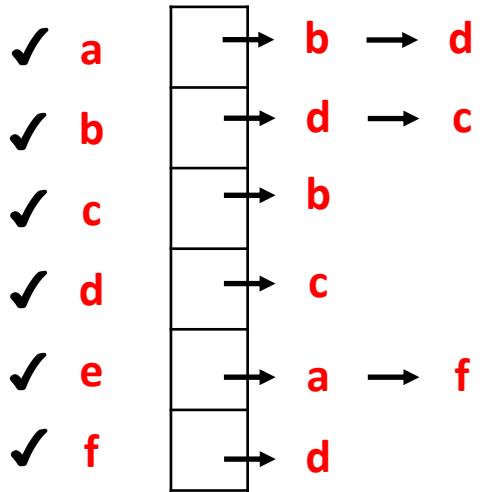
Colors of nodes:

B	B	B	B	B	B
e	a	b	c	d	f

DFS(G)



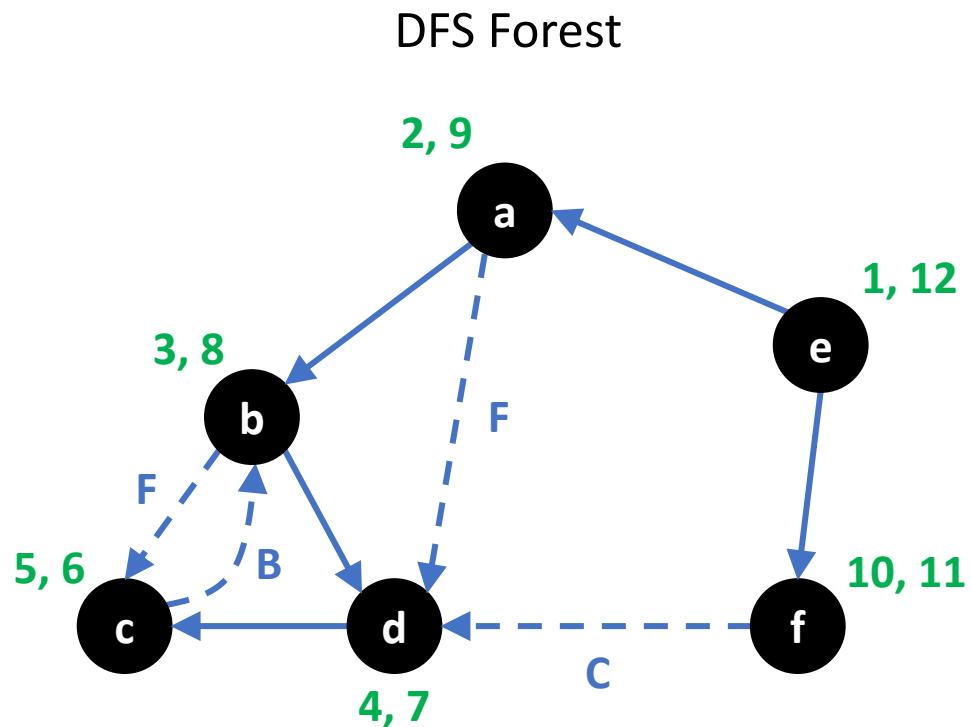
Adj List of G :



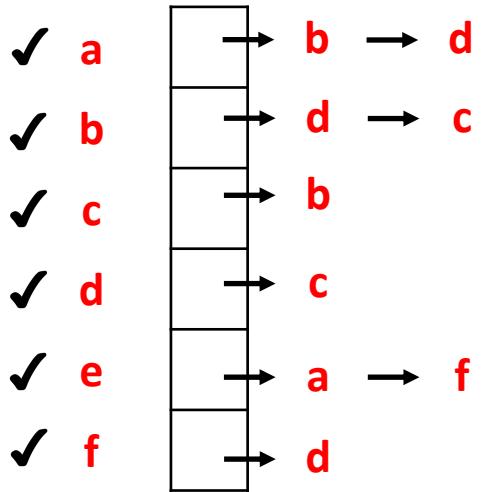
Colors of nodes:

B	B	B	B	B	B
e	a	b	c	d	f

DFS(G)



Adj List of G :



Colors of nodes:

B	B	B	B	B	B
e	a	b	c	d	f