



CS 225

Data Structures

April 15 – Floyd-Warshall's Algorithm

G Carl Evans

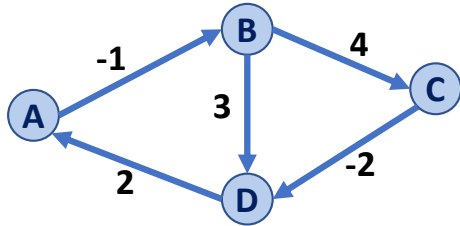
Floyd-Warshall Algorithm

Floyd-Warshall's Algorithm is an alternative to Dijkstra in the presence of **negative-weight edges** (not **negative weight cycles**).

```
FloydWarshall(G):
6   Let d be a adj. matrix initialized to +inf
7   foreach (Vertex v : G):
8       d[v][v] = 0
9   foreach (Edge (u, v) : G):
10      d[u][v] = cost(u, v)
11
12  foreach (Vertex w : G):
13      foreach (Vertex u : G):
14          foreach (Vertex v : G):
15              if (d[u, v] > d[u, w] + d[w, v])
16                  d[u, v] = d[u, w] + d[w, v]
```

Floyd-Warshall Algorithm

```
FloydWarshall(G):  
6   Let d be a adj. matrix initialized to +inf  
7   foreach (Vertex v : G):  
8     d[v][v] = 0  
9   foreach (Edge (u, v) : G):  
10    d[u][v] = cost(u, v)  
11  
12  foreach (Vertex w : G):  
13    foreach (Vertex u : G):  
14      foreach (Vertex v : G):  
15        if d[u, v] > d[u, w] + d[w, v]:  
16          d[u, v] = d[u, w] + d[w, v]
```

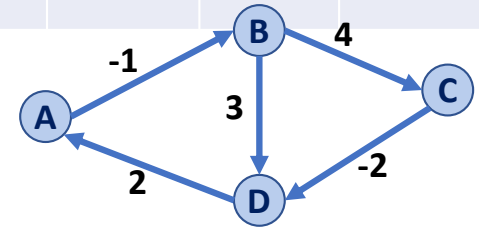


	A	B	C	D
A				
B				
C				
D				

Floyd-Warshall Algorithm

```
12  foreach (Vertex k : G):
13    foreach (Vertex u : G):
14      foreach (Vertex v : G):
15        if  $d[u, v] > d[u, \mathbf{k}] + d[\mathbf{k}, v]$ :
16           $d[u, v] = d[u, \mathbf{k}] + d[\mathbf{k}, v]$ 
```

	A	B	C	D
A	0	-1	∞	∞
B	∞	0	4	3
C	∞	∞	0	-2
D	2	∞	∞	0

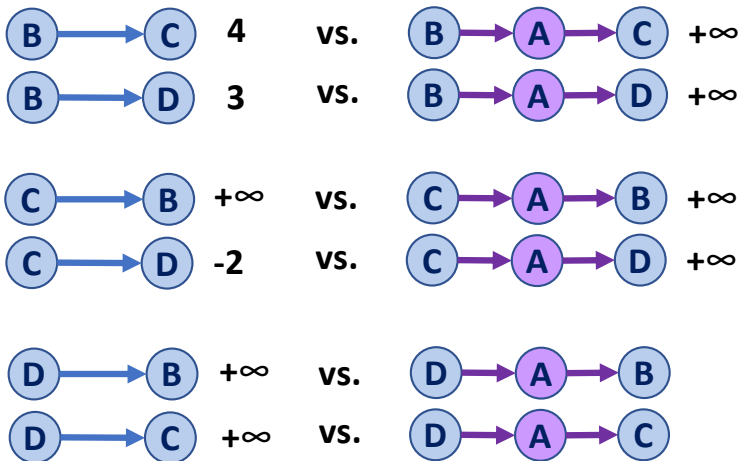


Floyd-Warshall Algorithm

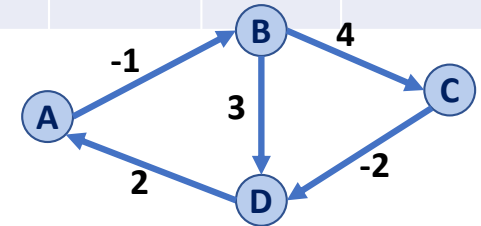
```

12  foreach (Vertex k : G):
13      foreach (Vertex u : G):
14          foreach (Vertex v : G):
15              if d[u, v] > d[u, k] + d[k, v]:
16                  d[u, v] = d[u, k] + d[k, v]
    
```

Let us consider k=A:



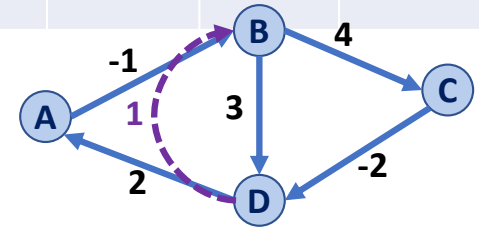
	A	B	C	D
A	0	-1	∞	∞
B	∞	0	4	3
C	∞	∞	0	-2
D	2	∞	∞	0



Floyd-Warshall Algorithm

```
12  foreach (Vertex k : G):
13    foreach (Vertex u : G):
14      foreach (Vertex v : G):
15        if  $d[u, v] > d[u, \mathbf{k}] + d[\mathbf{k}, v]$ :
16           $d[u, v] = d[u, \mathbf{k}] + d[\mathbf{k}, v]$ 
```

	A	B	C	D
A	0	-1	∞	∞
B	∞	0	4	3
C	∞	∞	0	-2
D	2	1	∞	0

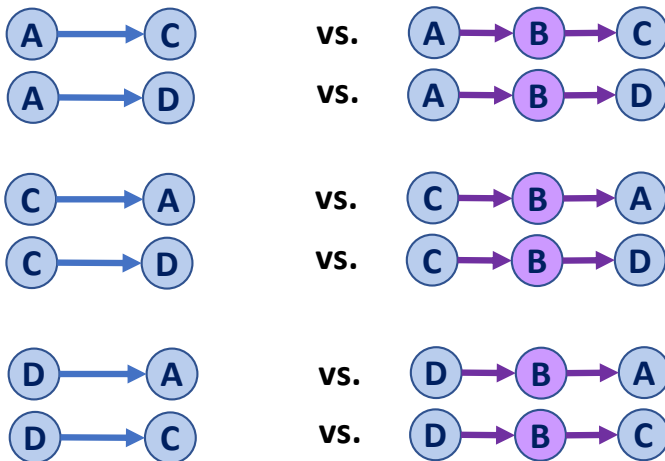


Floyd-Warshall Algorithm

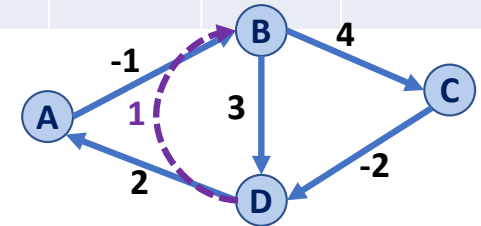
```

12  foreach (Vertex u : G):
13      foreach (Vertex v : G):
14          foreach (Vertex k : G):
15              if d[u, v] > d[u, k] + d[k, v]:
16                  d[u, v] = d[u, k] + d[k, v]
    
```

Let us consider k=B:

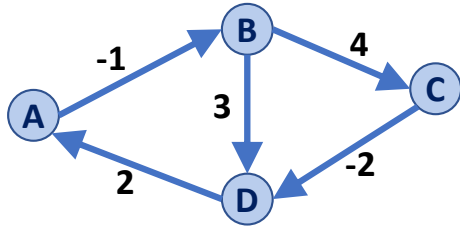


	A	B	C	D
A	0	-1	∞	∞
B	∞	0	4	3
C	∞	∞	0	-2
D	2	1	∞	0



Floyd-Warshall Algorithm

```
FloydWarshall(G):  
6   Let d be a adj. matrix initialized to +inf  
7   foreach (Vertex v : G):  
8     d[v][v] = 0  
9   foreach (Edge (u, v) : G):  
10    d[u][v] = cost(u, v)  
11  
12  foreach (Vertex w : G):  
13    foreach (Vertex u : G):  
14      foreach (Vertex v : G):  
15        if d[u, v] > d[u, w] + d[w, v]:  
16          d[u, v] = d[u, w] + d[w, v]
```



	A	B	C	D
A				
B				
C				
D				

	A	B	C	D
A				
B				
C				
D				

Floyd-Warshall Algorithm

Running Time?

```
FloydWarshall(G):
6   Let d be a adj. matrix initialized to +inf
7   foreach (Vertex v : G):
8       d[v][v] = 0
9   foreach (Edge (u, v) : G):
10      d[u][v] = cost(u, v)
11
12  foreach (Vertex u : G):
13      foreach (Vertex v : G):
14          foreach (Vertex w : G):
15              if d[u, v] > d[u, w] + d[w, v]:
16                  d[u, v] = d[u, w] + d[w, v]
```