

LÝ THUYẾT ĐỘ THỊ GRAPH THEORY

LÊ THỊ PHƯƠNG DUNG

NỘI DUNG

- 1. ĐẠI CƯƠNG VỀ ĐỒ THỊ**
- 2. TÍNH LIÊN THÔNG CỦA ĐỒ THỊ**
- 3. ĐƯỜNG ĐI NGẮN NHẤT TRÊN ĐỒ THỊ**
- 4. XẾP HẠNG ĐỒ THỊ**
- 5. CÂY VÀ CÂY CÓ HƯỚNG**
- 6. LUỒNG CỤC ĐẠI TRONG MẠNG**

TÀI LIỆU THAM KHẢO

1. TOÁN RỜI RẠC – *NGUYỄN TÔ THÀNH, NGUYỄN
ĐỨC NGHĨA*
 2. LÝ THUYẾT ĐỒ THỊ VÀ ỨNG DỤNG – *NGUYỄN
TUẤN ANH*
-

CHƯƠNG 3

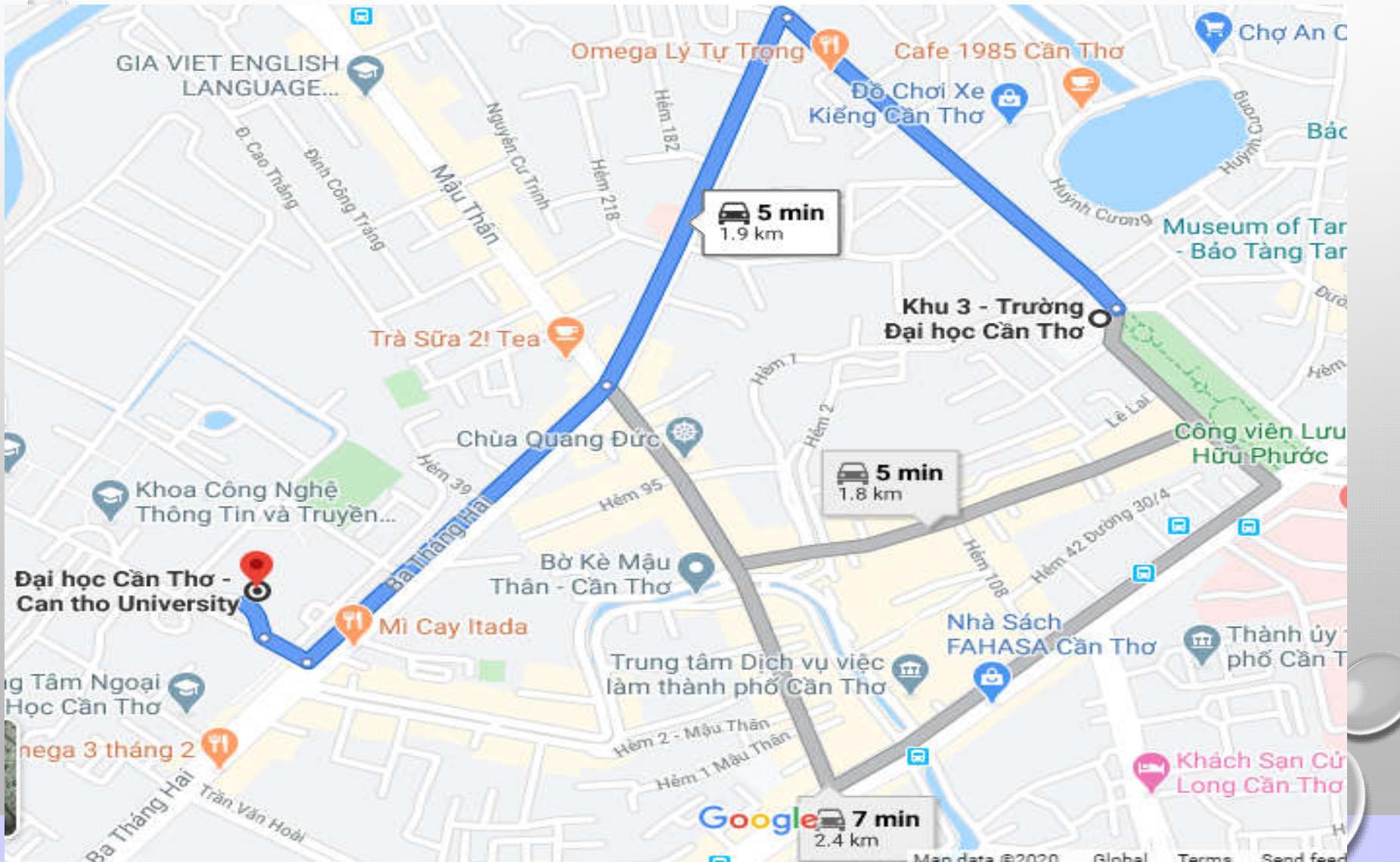
ĐƯỜNG ĐI NGẮN NHẤT TRÊN ĐỒ THỊ

NỘI DUNG:

1. BÀI TOÁN ĐƯỜNG ĐI NGẮN NHẤT
2. THUẬT TOÁN MOORE – DIJKSTRA
3. THUẬT TOÁN BELLMAN – FORDS
4. *THUẬT TOÁN FLOYD – WARSHALL*

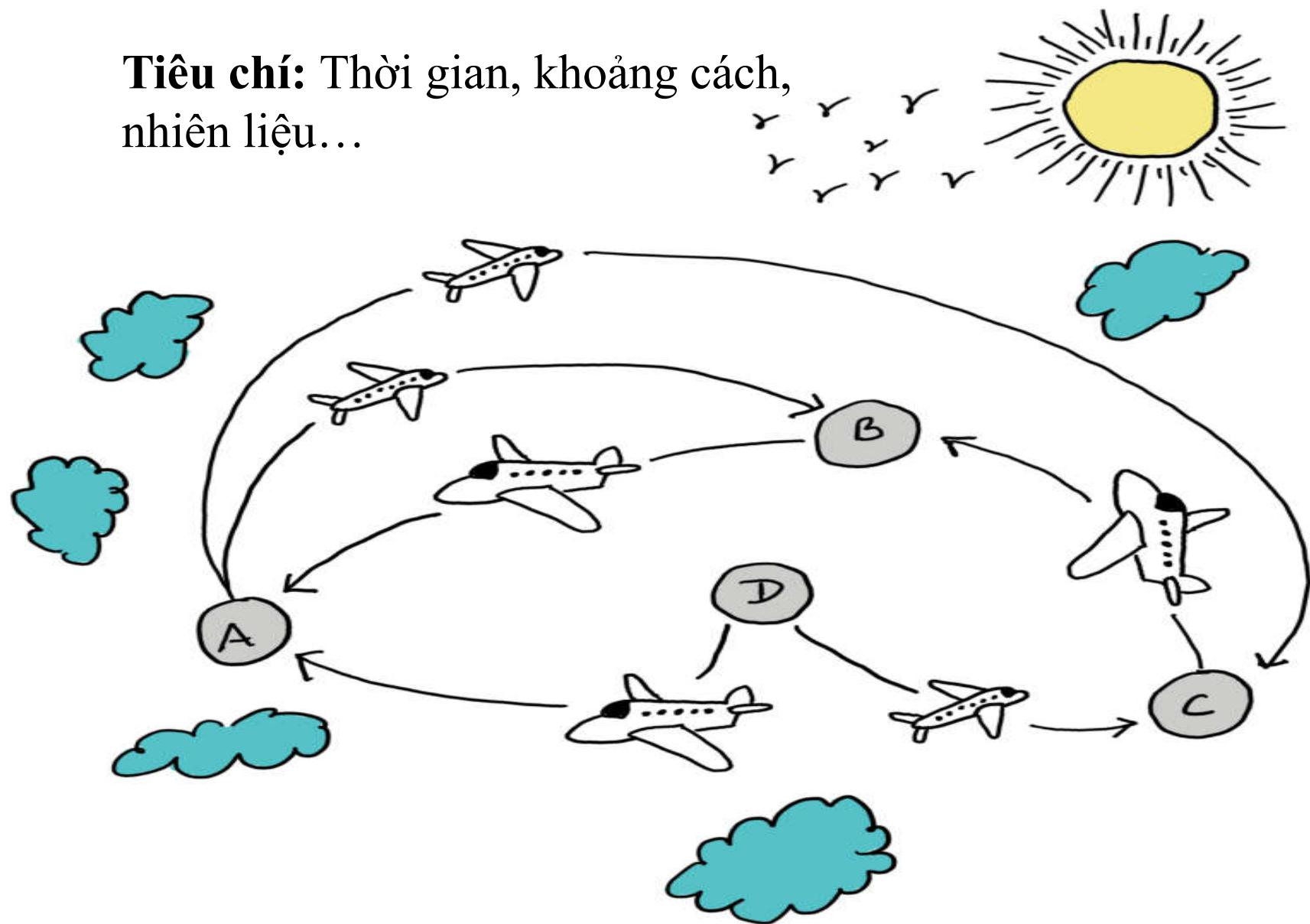
MỘT SỐ ỨNG DỤNG

Tiêu chí: Thời gian, nhiên liệu...



MỘT SỐ ỨNG DỤNG

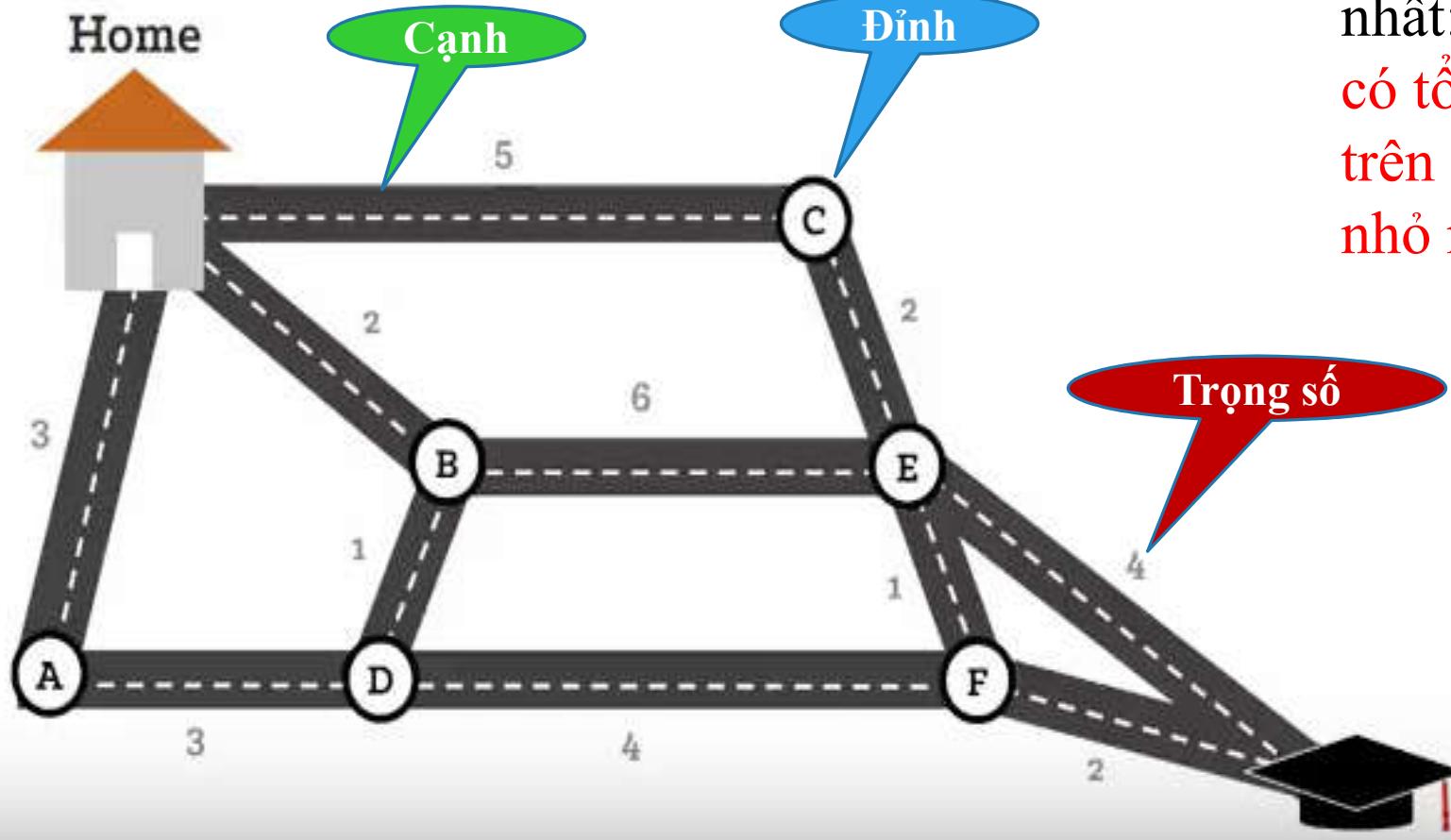
Tiêu chí: Thời gian, khoảng cách, nhiên liệu...



BÀI TOÁN ĐƯỜNG ĐI NGẮN NHẤT

Đồ thị có trọng số

Bài toán: Tìm đường đi ngắn nhất từ Home đến School

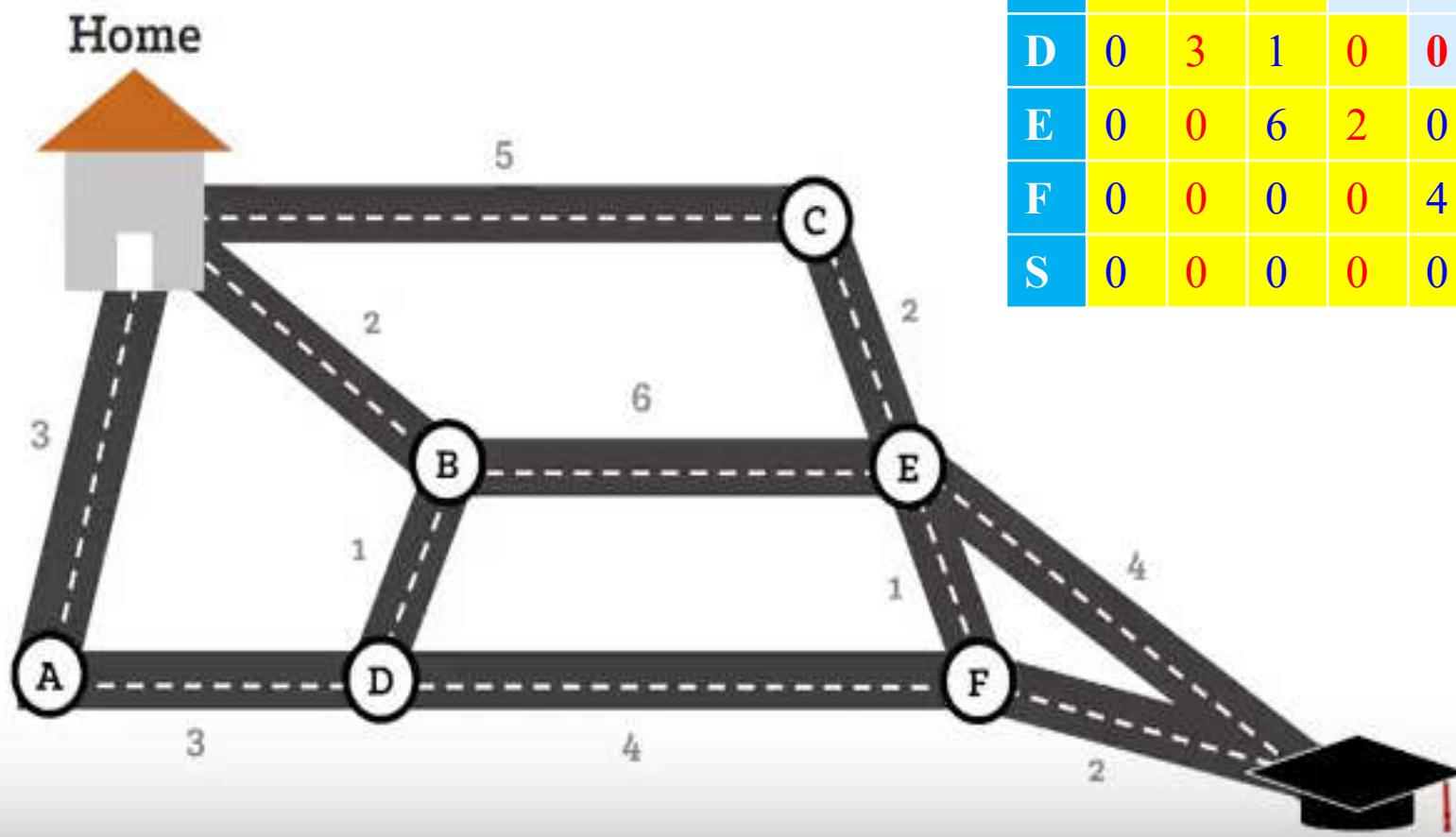


Đường đi ngắn nhất: Đường đi có **tổng trọng số** trên các cạnh là nhỏ nhất

BÀI TOÁN ĐƯỜNG ĐI NGẮN NHẤT

Ma trận trọng số

Đồ thị vô hướng có ma trận trọng số là ma trận **đối xứng**. Các giá trị bên dưới đường chéo chính có thể khuyết



	H	A	B	C	D	E	F	S
H	0	3	2	5	0	0	0	0
A	3	0	0	0	3	0	0	0
B	2	0	0	0	1	6	0	0
C	5	0	0	0	0	2	0	0
D	0	3	1	0	0	0	4	0
E	0	0	6	2	0	0	1	4
F	0	0	0	0	4	1	0	2
S	0	0	0	0	0	4	2	0

BÀI TOÁN ĐƯỜNG ĐI NGẮN NHẤT

Biểu diễn ma trận trọng số trên máy tính

```
#define MAXN 200
#define NO_EDGE 0
typedef struct {
    int n;           //số đỉnh
    int A[MAXN][MAXN];
} Graph;
```

	H	A	B	C	D	E	F	S
H	0	3	2	5	0	0	0	0
A	3	0	0	0	3	0	0	0
B	2	0	0	0	1	6	0	0
C	5	0	0	0	0	2	0	0
D	0	3	1	0	0	0	4	0
E	0	0	6	2	0	0	1	4
F	0	0	0	0	4	1	0	2
S	0	0	0	0	0	4	2	0

GIẢI THUẬT MOORE – DIJKSTRA

Professor Edsger W. Dijkstra Ph.D.

Born	11 May 1930 Rotterdam, Netherlands
Died	6 August 2002 (aged 72) Nuenen, Netherlands
Citizenship	Netherlands
Alma mater	Leiden University (B.S., M.S.) University of Amsterdam (Ph.D.)
Awards	<ul style="list-style-type: none">•SIGCSE Outstanding Contribution (1989)•Turing Award (1972)•ACM Fellow (1994)•Dijkstra Prize (2002)



Scientific career

Fields	<ul style="list-style-type: none">•Computer science•Theoretical computer science
Institutions	<ul style="list-style-type: none">•Mathematisch Centrum•Eindhoven University of Technology•Burroughs Corporation•The University of Texas at Austin

An equivalent algorithm was developed by American professor Edward Forrest Moore in 1957

GIẢI THUẬT MOORE – DIJKSTRA

- Tìm đường đi ngắn nhất từ một đỉnh đến các đỉnh khác trên **đồ thị có trọng số dương**

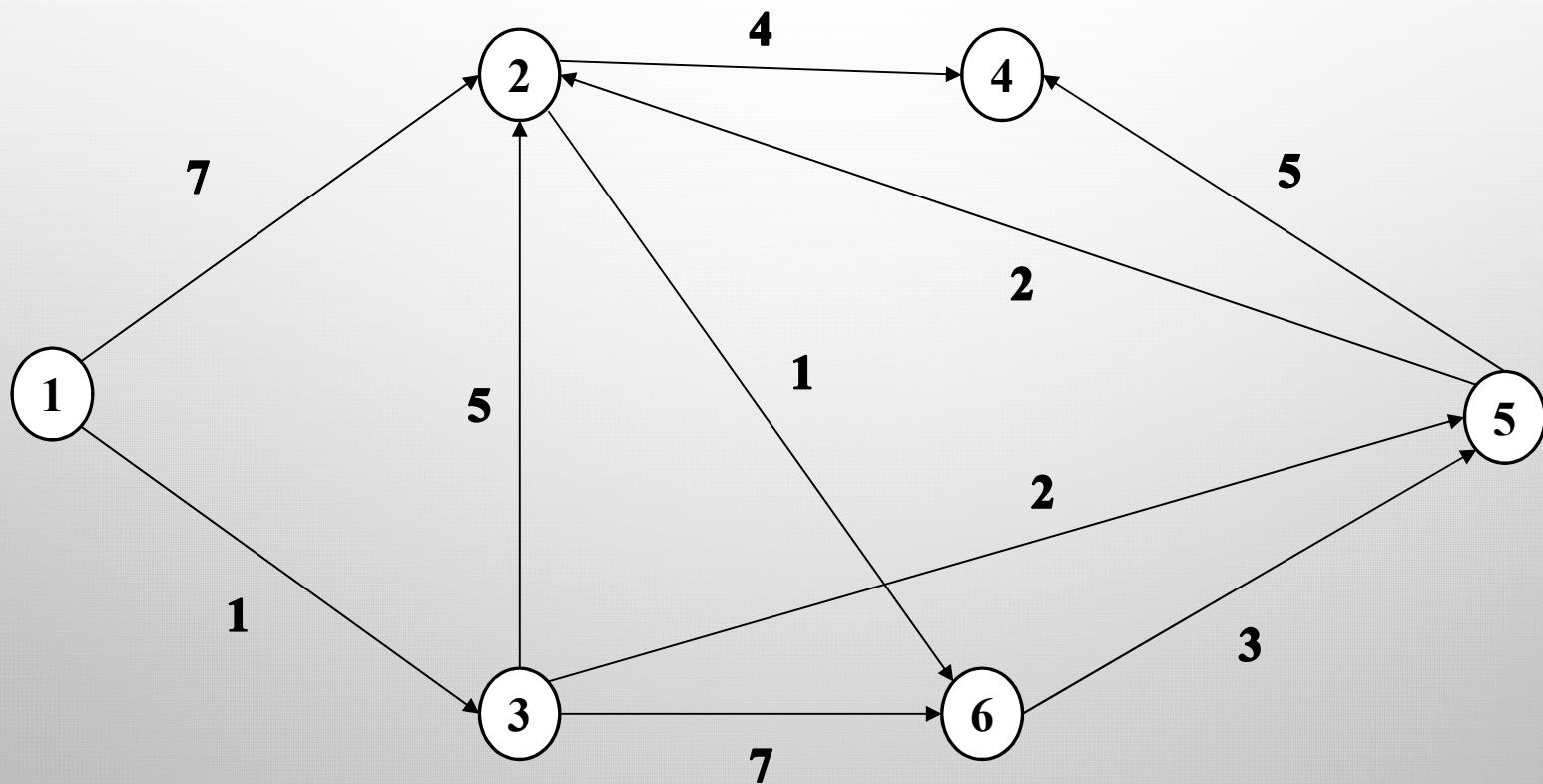
```

1 function Dijkstra(Graph, source):
2     create vertex set Q
3     for each vertex v in Graph:
4         dist[v] ← INFINITY          //Distance from source to v
5         prev[v] ← UNDEFINED        //Previous of v
6         add v to Q
7     dist[source] ← 0
8     while Q is not empty:
9         u ← vertex in Q with min dist[u]
10        remove u from Q
11        for each neighbor v of u:    // only v that are still in Q
12            alt ← dist[u] + length(u, v)
13            if alt < dist[v]:
14                dist[v] ← alt
15                prev[v] ← u
16    return dist[], prev[]

```

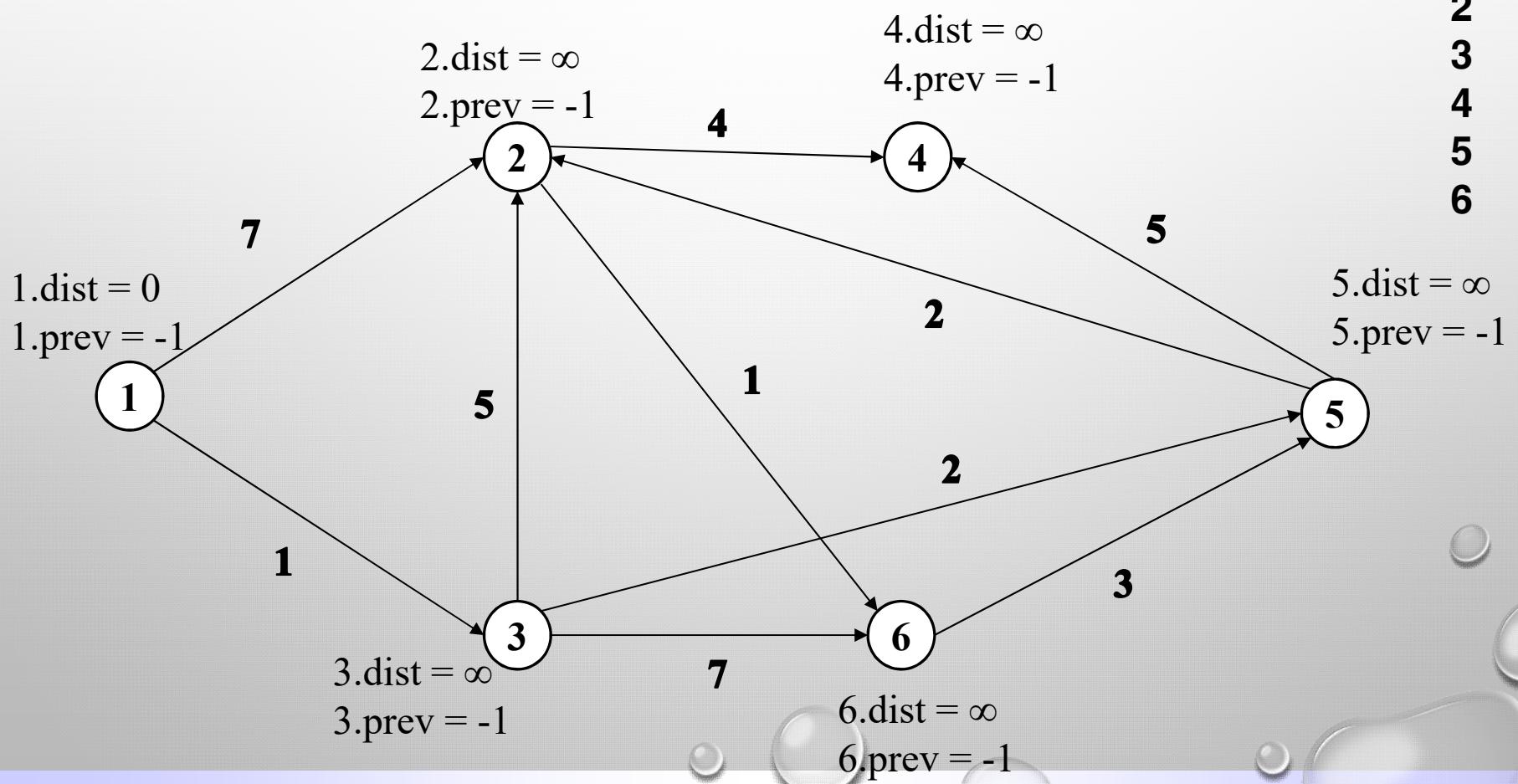
GIẢI THUẬT MOORE – DIJKSTRA

Tìm đường đi ngắn nhất từ 1 đến các đỉnh khác trên đồ thị có hướng sau đây



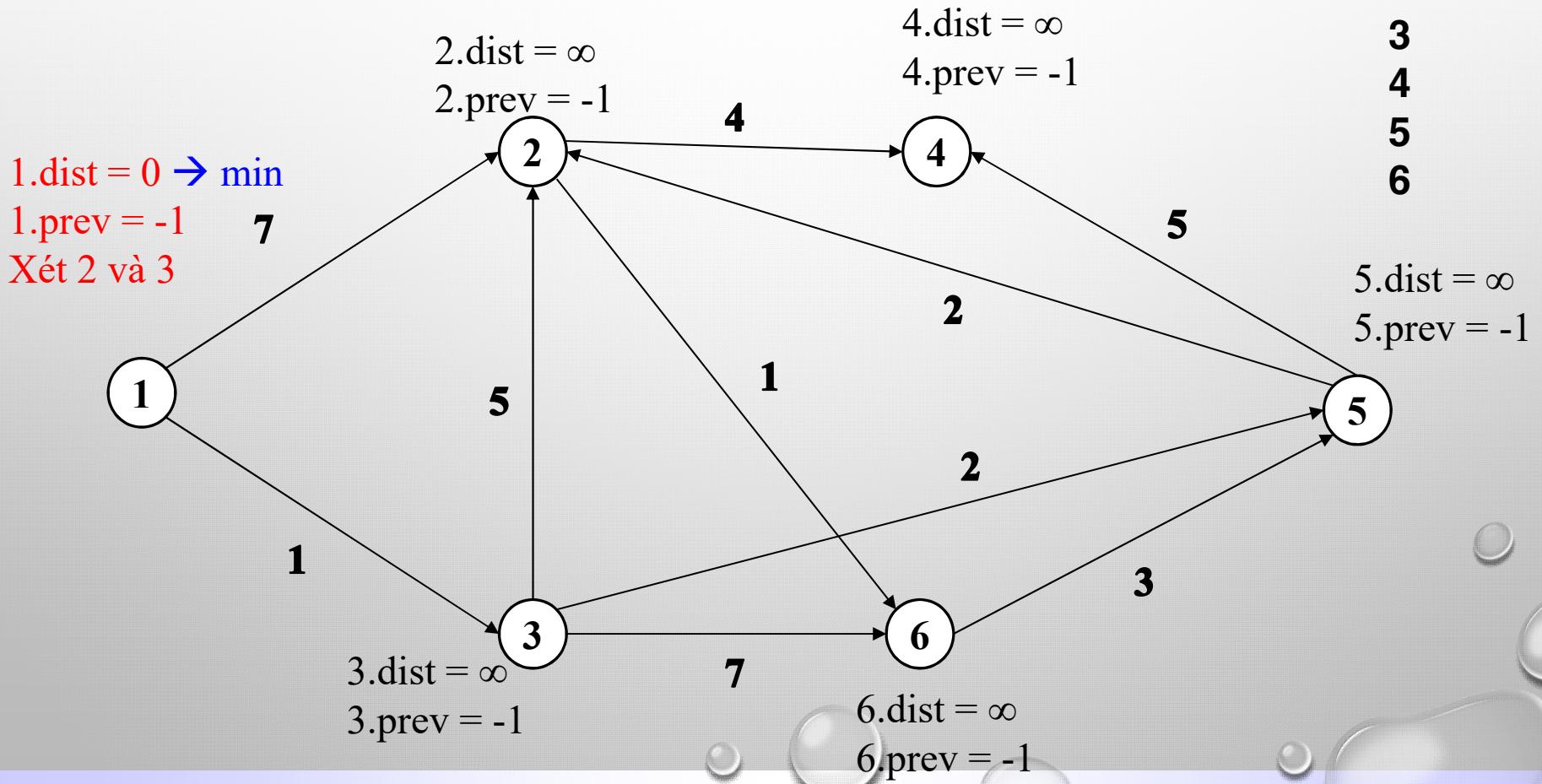
GIẢI THUẬT MOORE – DIJKSTRA

Tìm đường đi ngắn nhất từ 1 đến các đỉnh khác trên đồ thị có hướng sau đây



GIẢI THUẬT MOORE – DIJKSTRA

Tìm đường đi ngắn nhất từ 1 đến các đỉnh khác trên đồ thị có hướng sau đây



GIẢI THUẬT MOORE – DIJKSTRA

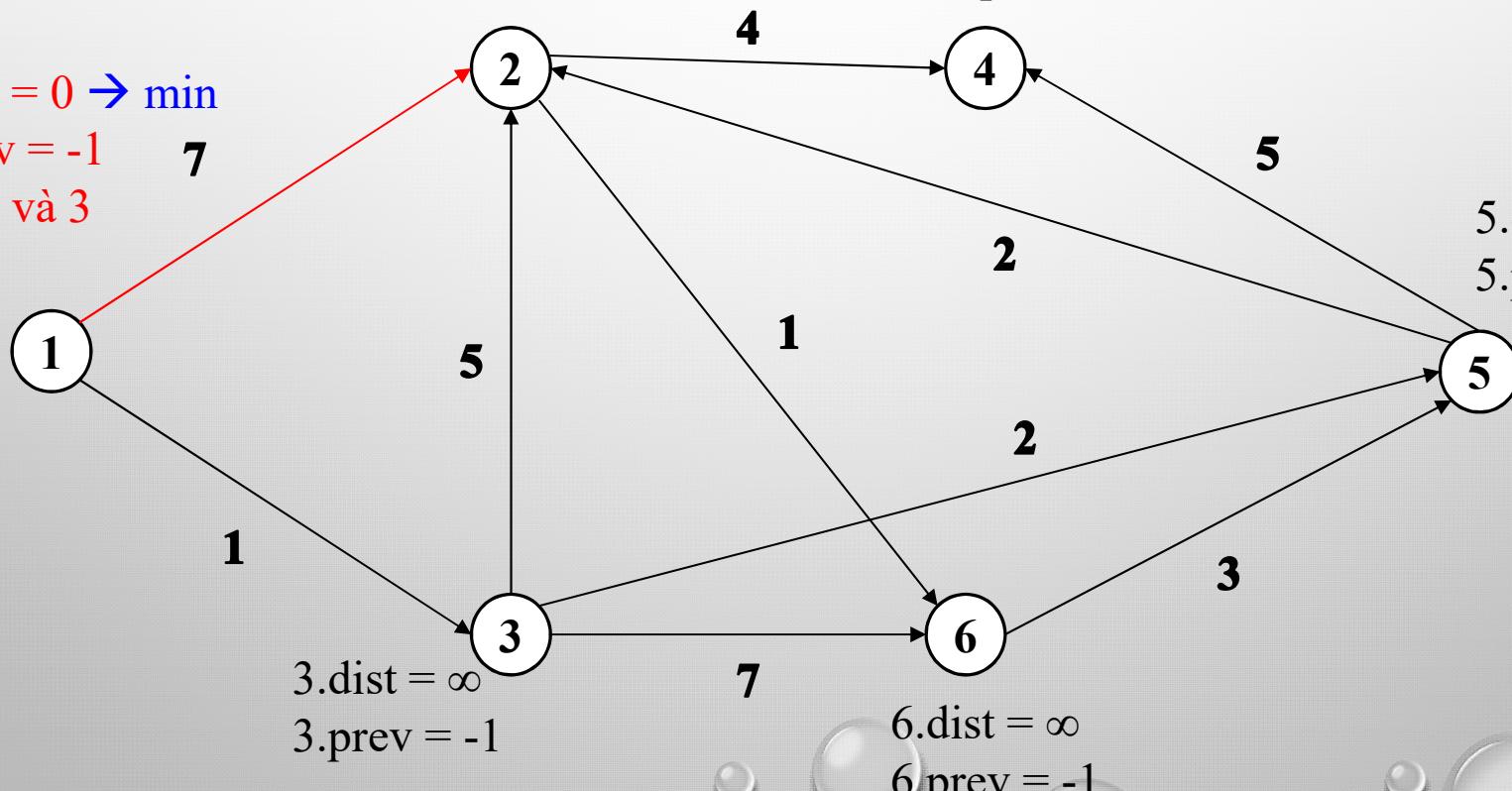
Tìm đường đi ngắn nhất từ 1 đến các đỉnh khác trên đồ thị có hướng sau đây

1.dist = 0 → min
1.prev = -1
Xét 2 và 3

2.dist = ∞
2.prev = -1
Vì 1.dist+len(1,2)=7<2.dist
nên cập nhật

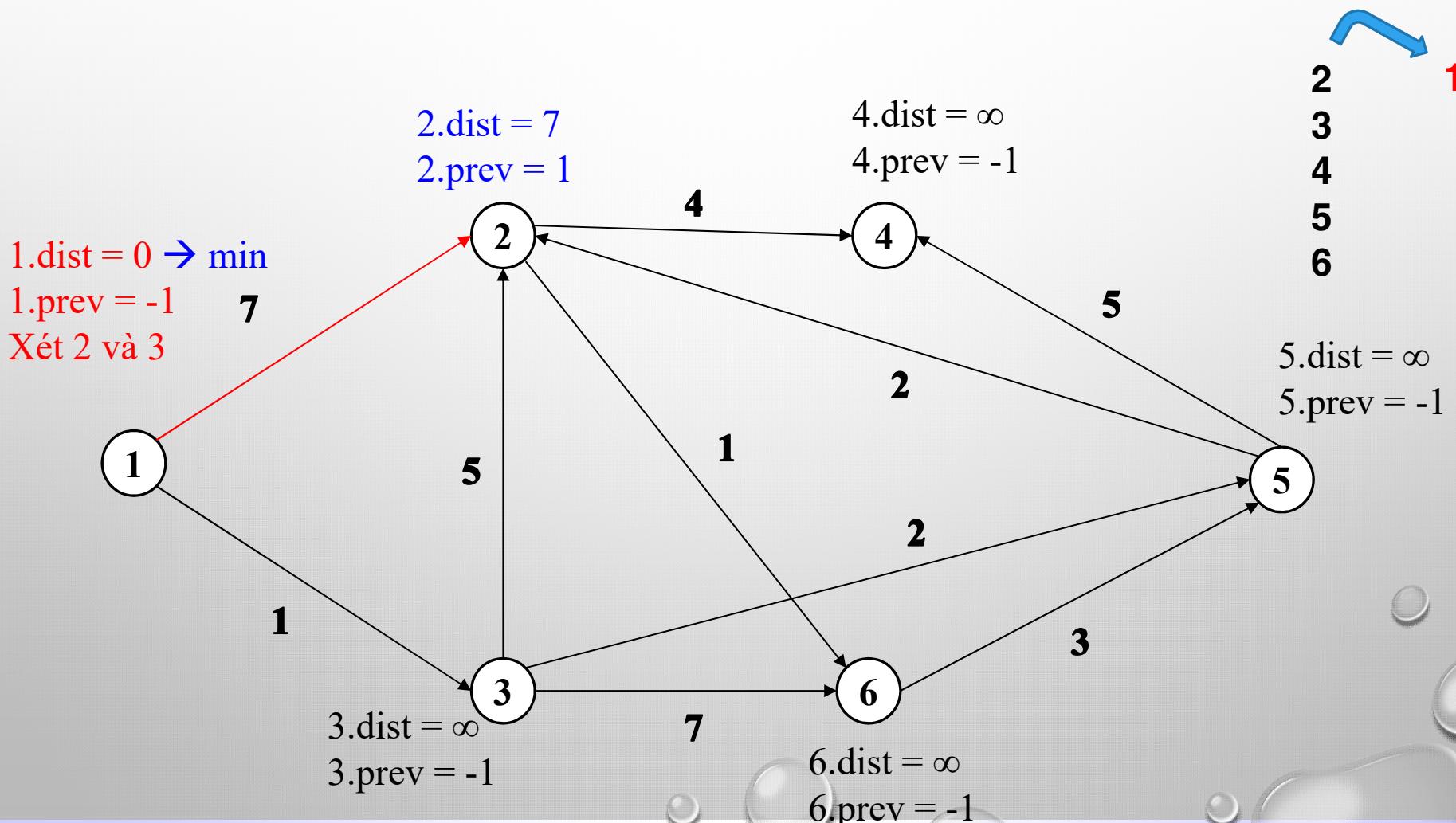
4.dist = ∞
4.prev = -1

5.dist = ∞
5.prev = -1



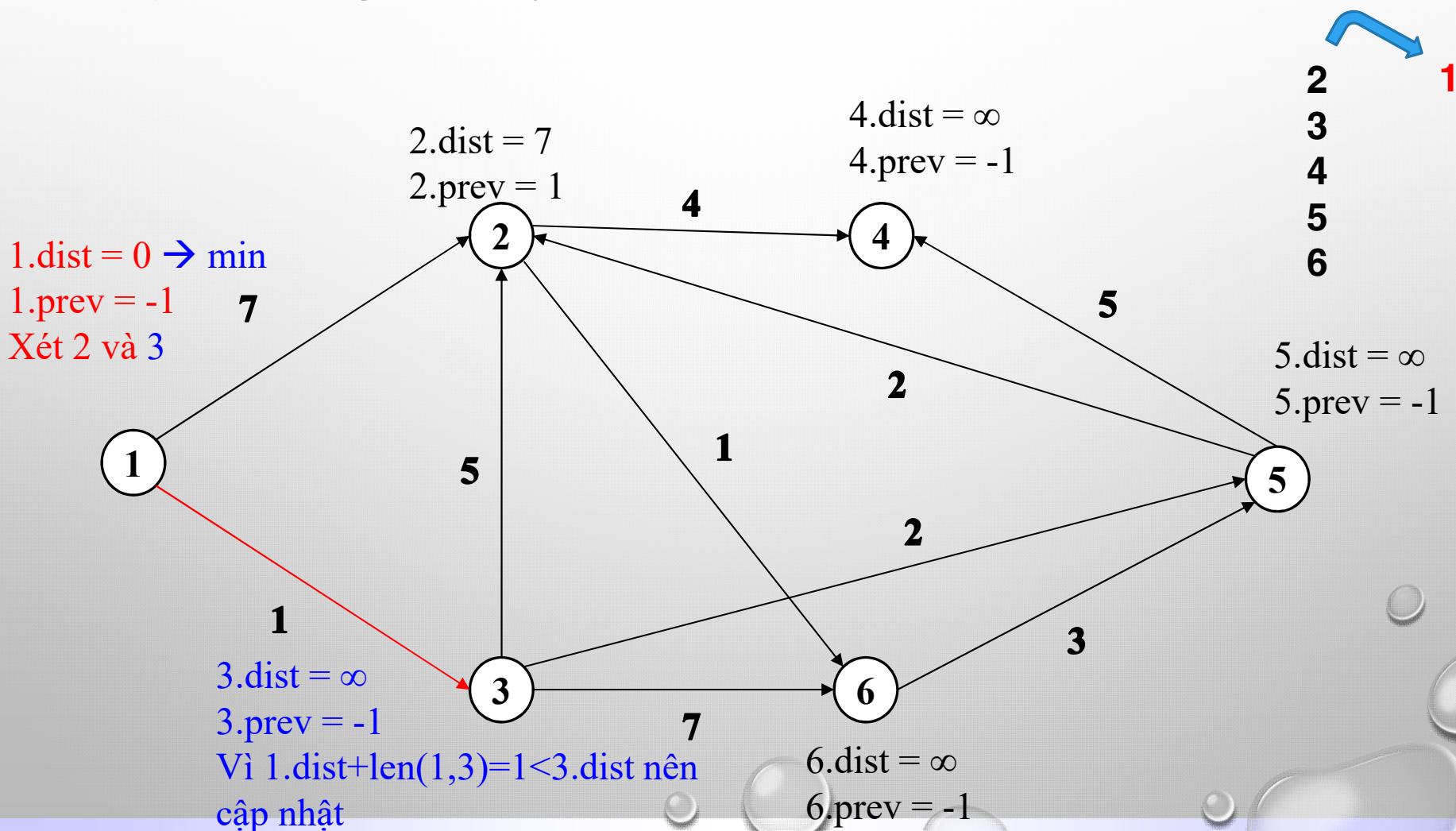
GIẢI THUẬT MOORE – DIJKSTRA

Tìm đường đi ngắn nhất từ 1 đến các đỉnh khác trên đồ thị có hướng sau đây



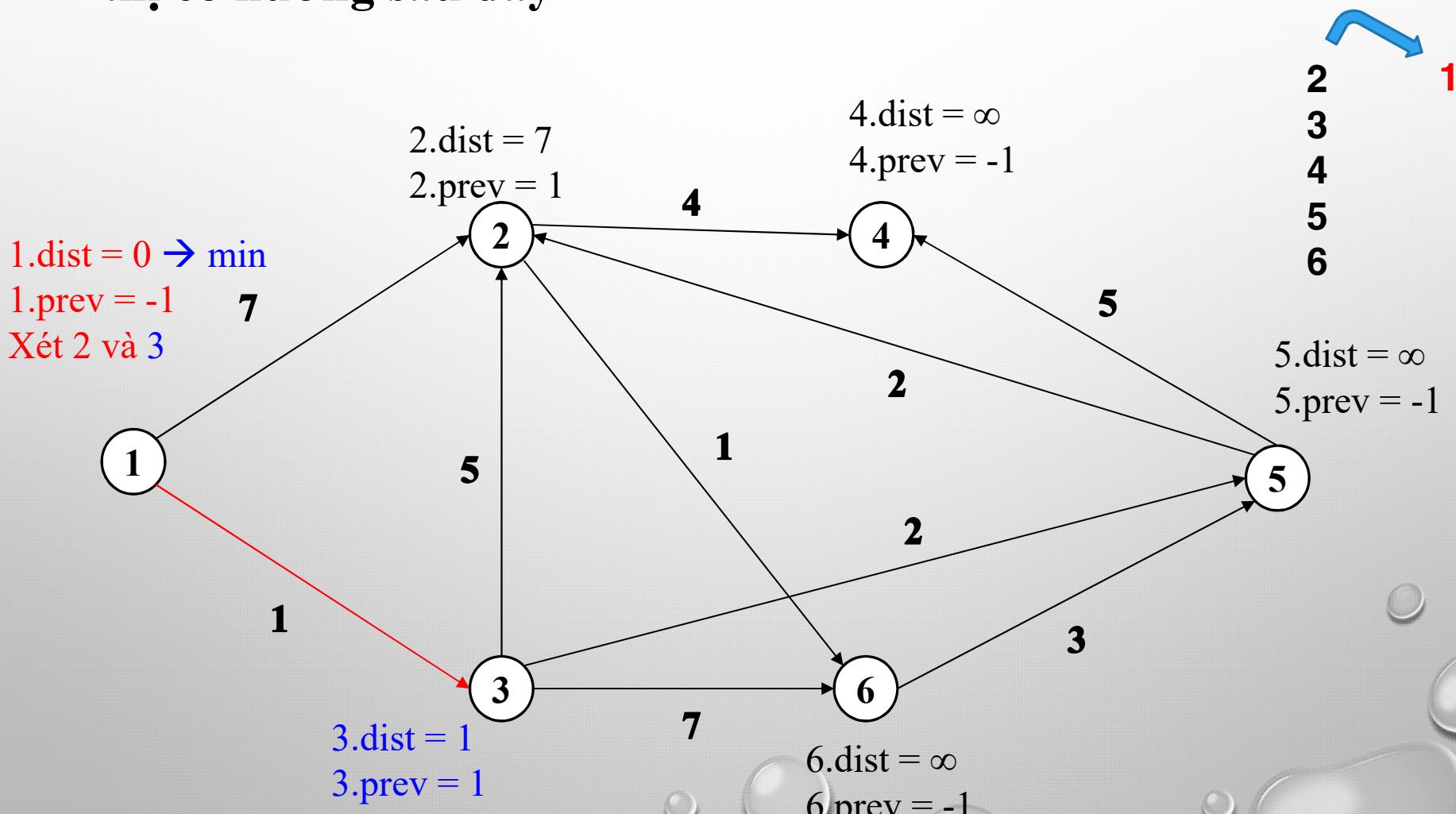
GIẢI THUẬT MOORE – DIJKSTRA

Tìm đường đi ngắn nhất từ 1 đến các đỉnh khác trên đồ thị có hướng sau đây



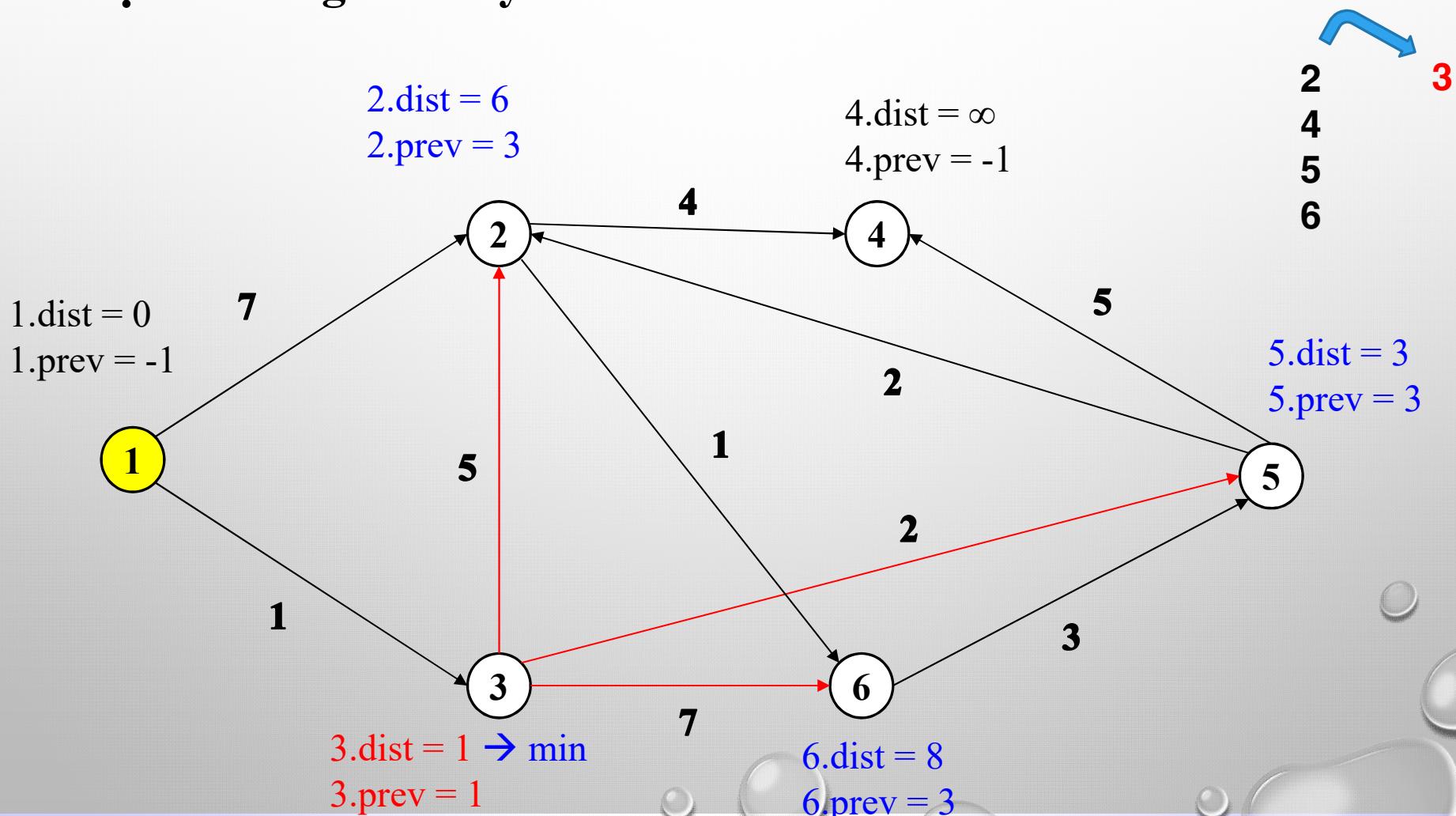
GIẢI THUẬT MOORE – DIJKSTRA

Tìm đường đi ngắn nhất từ 1 đến các đỉnh khác trên đồ thị có hướng sau đây



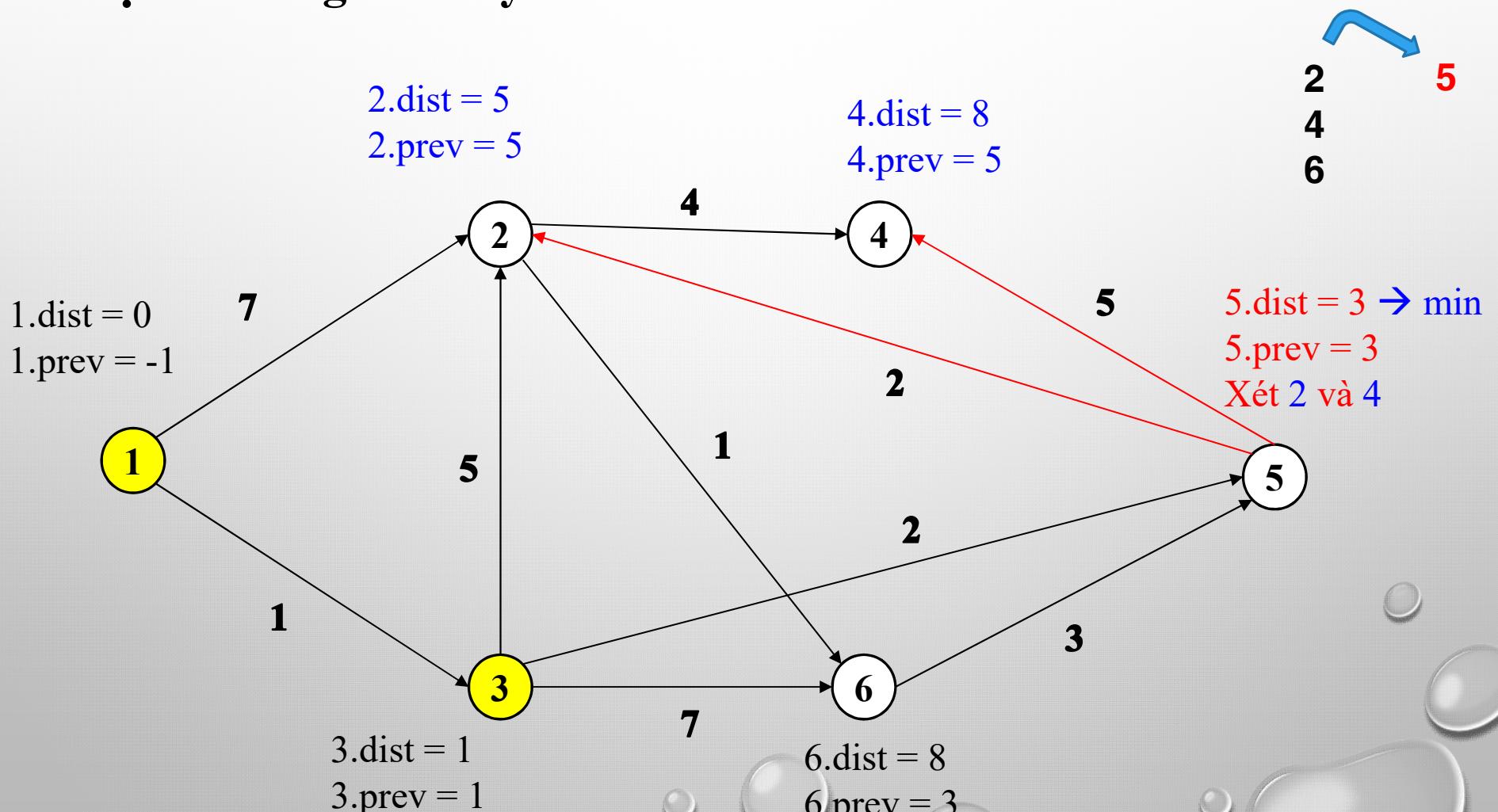
GIẢI THUẬT MOORE – DIJKSTRA

Tìm đường đi ngắn nhất từ 1 đến các đỉnh khác trên đồ thị có hướng sau đây



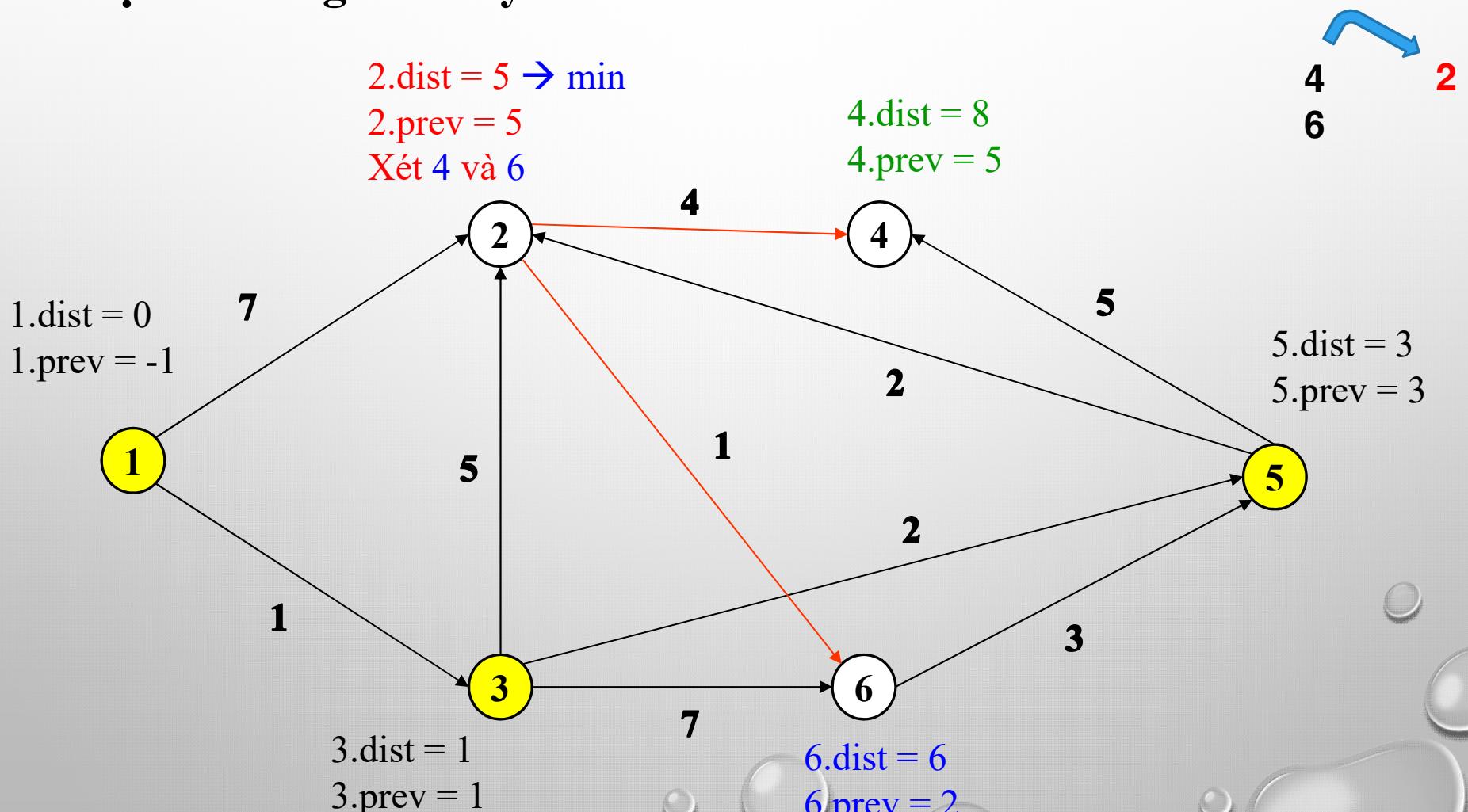
GIẢI THUẬT MOORE – DIJKSTRA

Tìm đường đi ngắn nhất từ 1 đến các đỉnh khác trên đồ thị có hướng sau đây



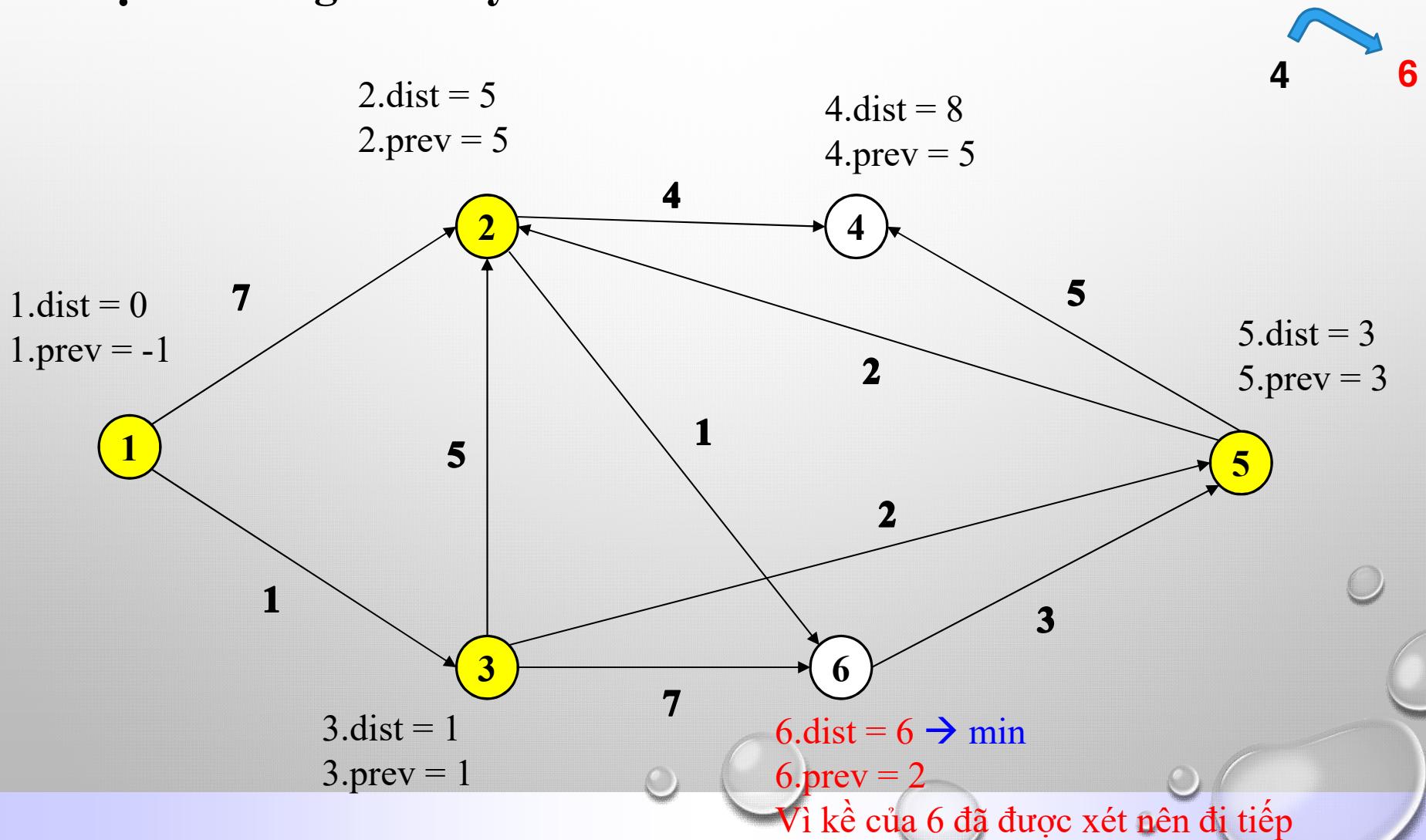
GIẢI THUẬT MOORE – DIJKSTRA

Tìm đường đi ngắn nhất từ 1 đến các đỉnh khác trên đồ thị có hướng sau đây



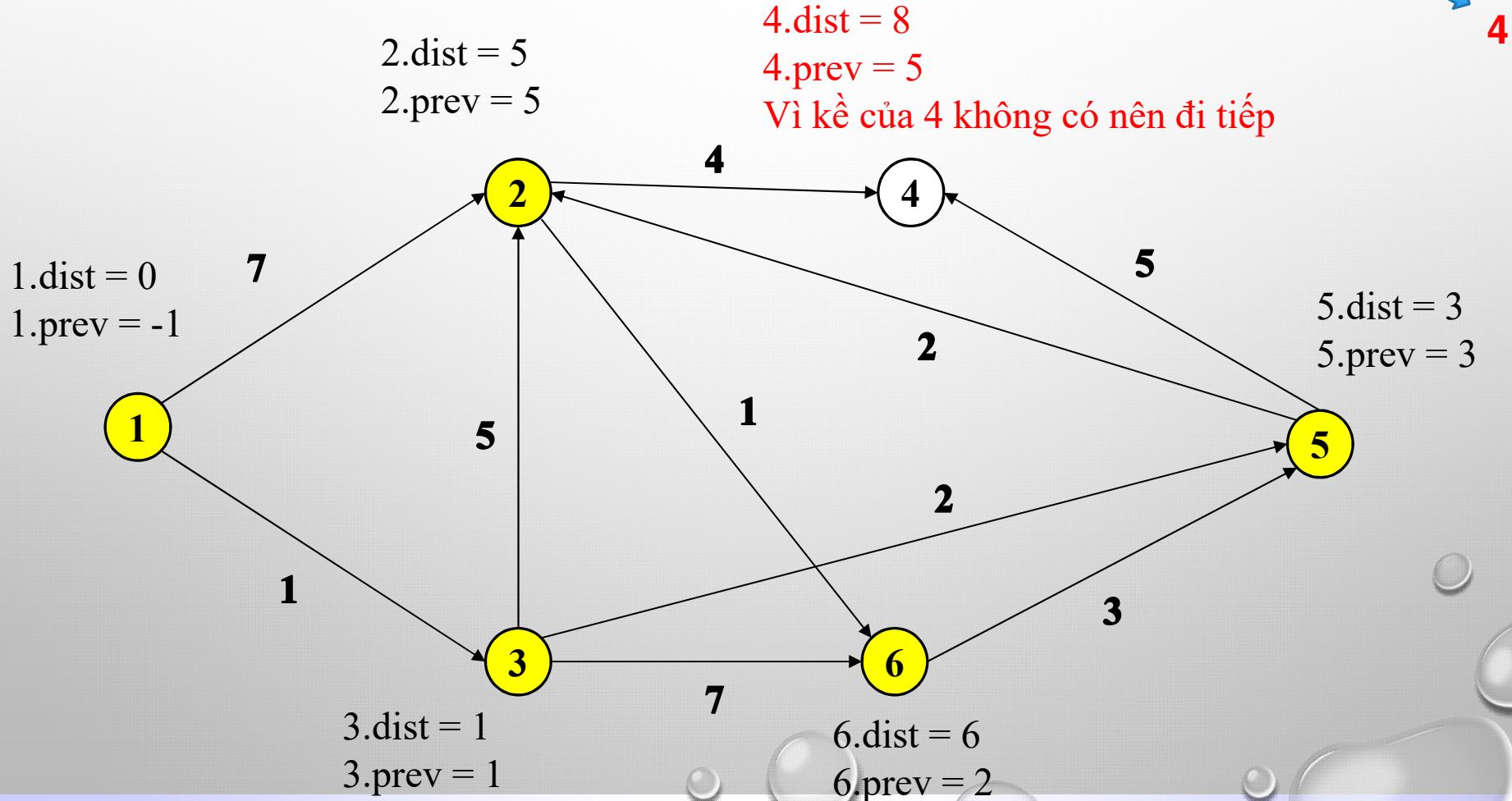
GIẢI THUẬT MOORE – DIJKSTRA

Tìm đường đi ngắn nhất từ 1 đến các đỉnh khác trên đồ thị có hướng sau đây



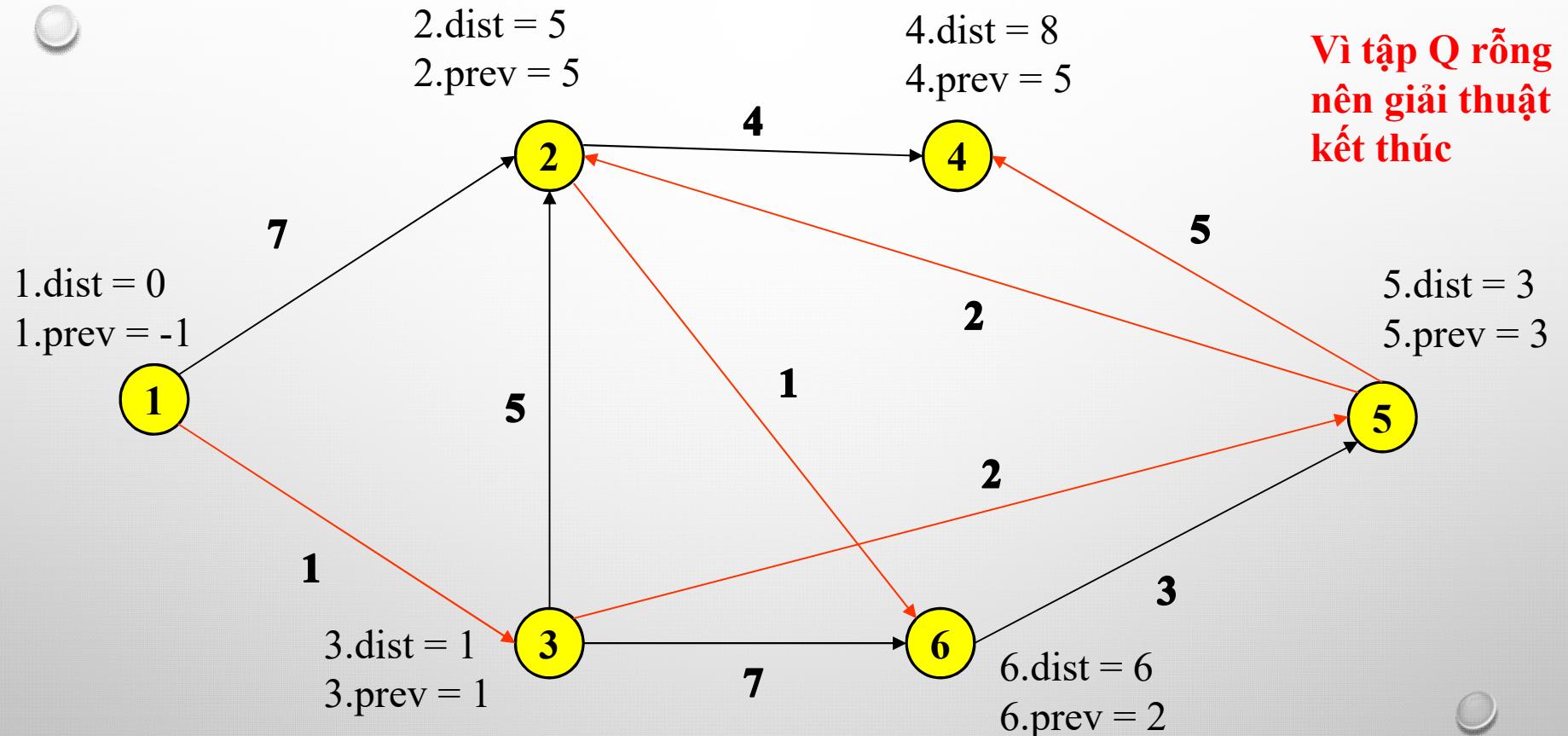
GIẢI THUẬT MOORE – DIJKSTRA

Tìm đường đi ngắn nhất từ 1 đến các đỉnh khác trên đồ thị có hướng sau đây



GIẢI THUẬT MOORE – DIJKSTRA

Kết quả đường đi



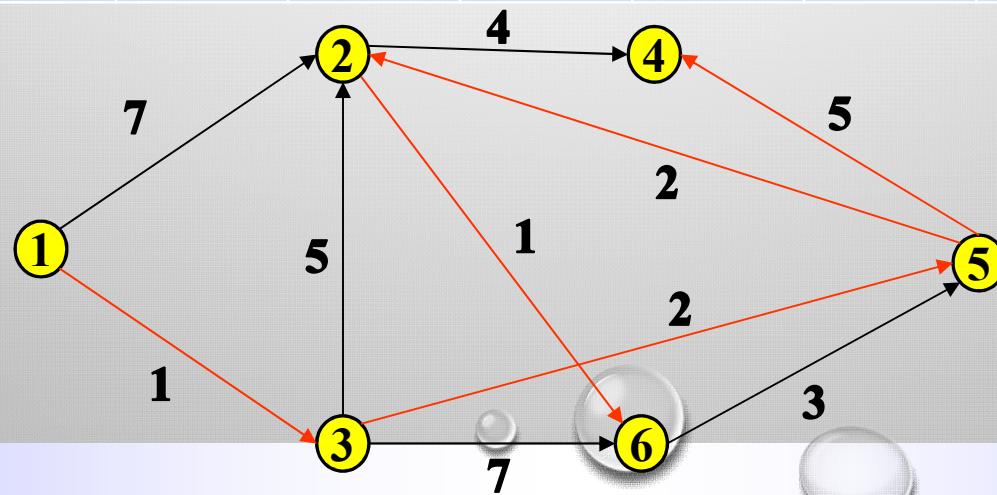
Vì tập Q rỗng
nên giải thuật
kết thúc

u	1	2	3	4	5	6
u.dist	0	5	1	8	3	6
u.prev	-1	5	1	5	3	2

GIẢI THUẬT MOORE – DIJKSTRA

Chạy thuật toán bằng cách lập bảng:

	1	2	3	4	5	6
0	(0,-1)	(∞ , -)				
1	*	(7,1)	(1,1)	(∞ , -)	(∞ , -)	(∞ , -)
2		(6,3)	*	(∞ , -)	(3,3)	(8,3)
3		(5,5)		(8,5)	*	(8,3)
4		*		(8,5)		(6,2)
5				(8,5)		*
6				*		



GIẢI THUẬT MOORE – DIJKSTRA

Bài tập: Tìm đường đi ngắn nhất từ A đến các đỉnh khác trên đồ thị vô hướng có ma trận trọng số như sau

Yêu cầu: Viết tay bài làm ra giấy và nộp cho cô

Hướng dẫn:

- Chạy thuật toán (lập bảng)
 - Vẽ đồ thị đường đi



BELLMAN – FORD (- MOORE)

Richard Ernest Bellman

Born	Richard Ernest Bellman August 26, 1920 New York City, New York, U.S.
Died	March 19, 1984 (aged 63) Los Angeles, California, U.S.
Alma mater	Princeton University Johns Hopkins University University of Wisconsin Brooklyn College

Known for	Dynamic programming Stochastic dynamic programming Curse of dimensionality Linear search problem Bellman equation Bellman–Ford algorithm Bellman's lost in a forest problem Bellman–Held–Karp algorithm Grönwall–Bellman inequality Hamilton–Jacobi–Bellman equation
Awards	John von Neumann Theory Prize (1976) IEEE Medal of Honor (1979) Richard E. Bellman Control Heritage Award (1984)
Fields	Mathematics and Control theory

Scientific career

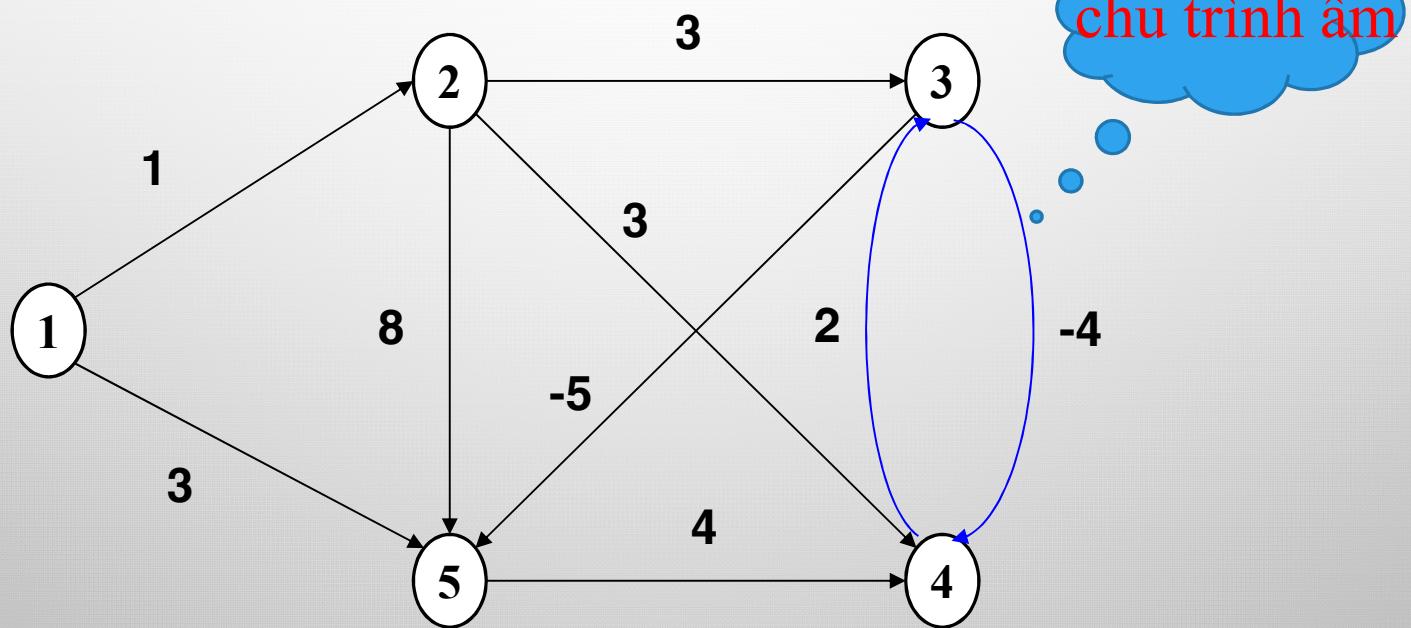
THUẬT TOÁN BELLMAN – FORD (- MOORE)



Lester Randolph Ford Jr. (September 23, 1927 – February 26, 2017) was an [American mathematician](#) specializing in [network flow](#) problems. He was the son of mathematician Lester R. Ford Sr.

THUẬT TOÁN BELLMAN – FORD (- MOORE)

Thuật toán Bellman – Ford tìm đường đi ngắn nhất **từ đỉnh xuất phát** đến tất cả các đỉnh còn lại của đồ thị có trọng số tùy ý nhưng không có chu trình âm



```
function BellmanFord(list vertices, list edges, vertex source) is
    ::distance[], predecessor[]
    //Step 1: initialize graph
    for each vertex v in vertices do
        distance[v] := inf           // Initialize the distance to all vertices to infinity
        predecessor[v] := null      // And having a null predecessor
    distance[source] := 0          // The distance from the source to itself is zero
    //Step 2: relax edges repeatedly
    for i from 1 to size(vertices)-1 do //just |V|-1 repetitions; i is never referenced
        for each edge (u, v) with weight w in edges do
            if distance[u] + w < distance[v] then
                distance[v] := distance[u] + w
                predecessor[v] := u
    //Step 3: check for negative-weight cycles
    for each edge (u, v) with weight w in edges do
        if distance[u] + w < distance[v] then
            error "Graph contains a negative-weight cycle"
    return distance[], predecessor[]
```

THUẬT TOÁN BELLMAN – FORD (- MOORE)

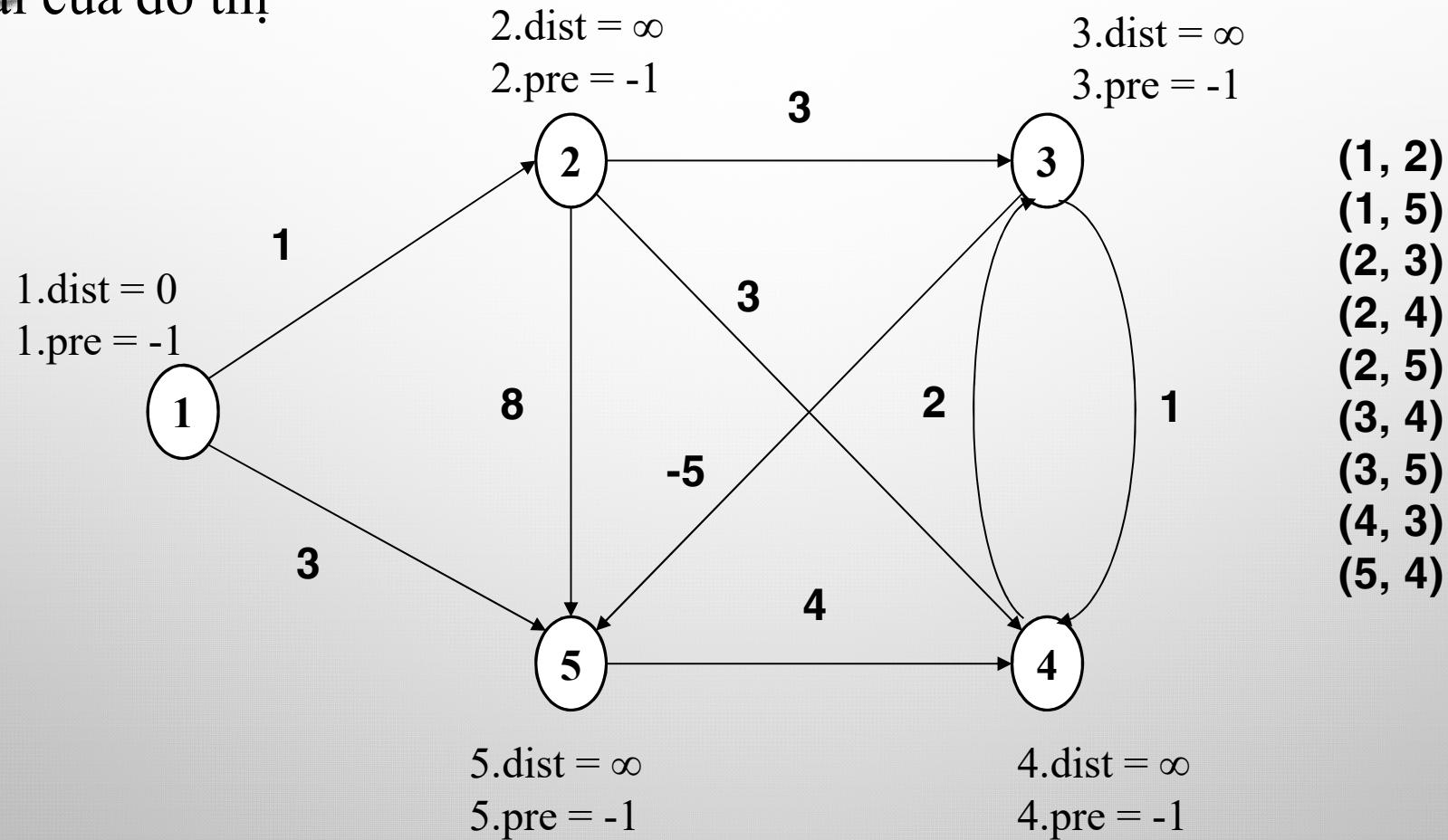
Cấu trúc đồ thị thường dùng là danh sách cung

```
typedef struct {  
    int u, v;          // cung (u,v)  
    int w;            // trọng số  
} Edge;
```

```
typedef struct {  
    int n, m;          // n đỉnh,m cung  
    Edge edges[500];   // danh sách cung  
}
```

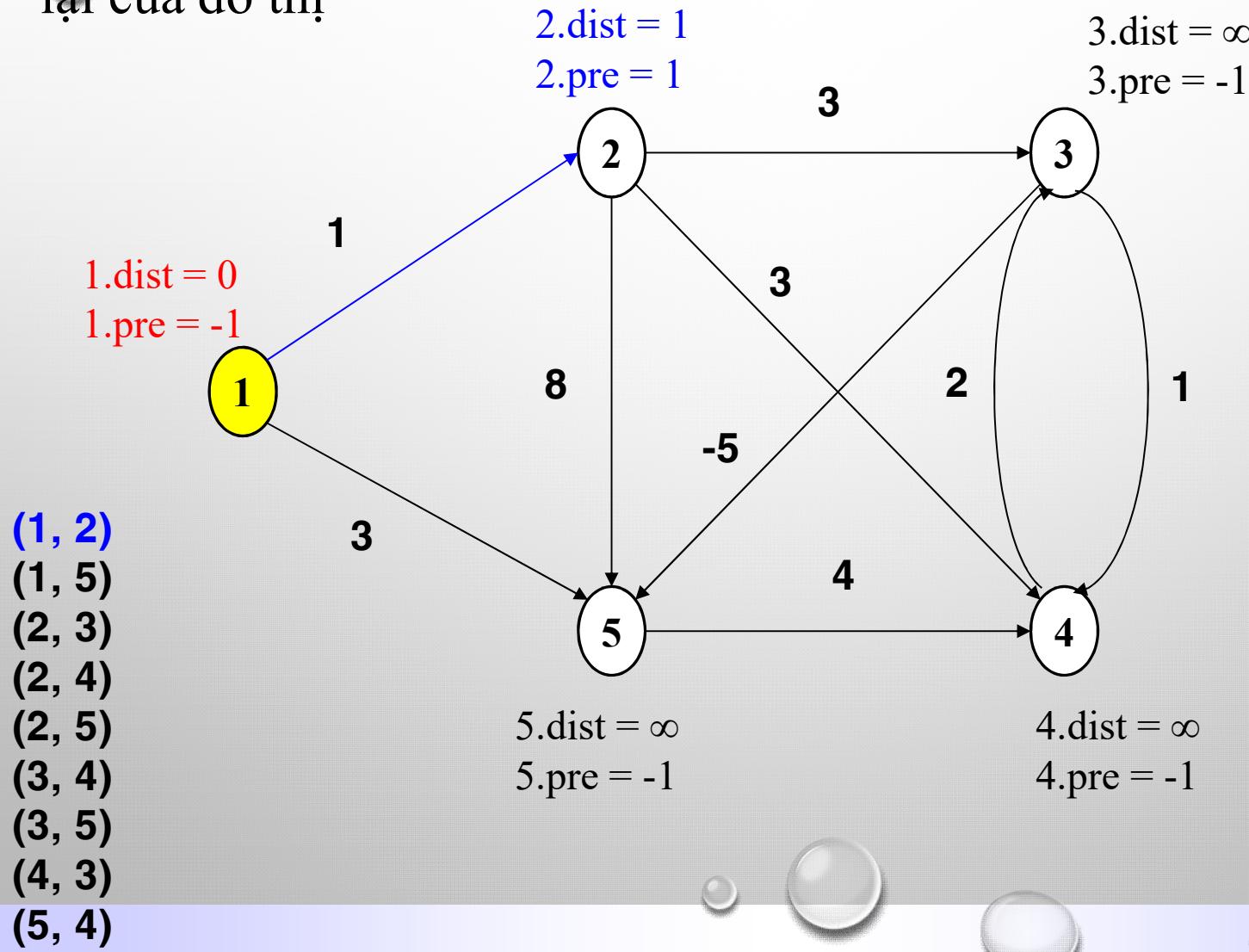
THUẬT TOÁN BELLMAN – FORD (- MOORE)

Tìm đường đi ngắn nhất từ đỉnh xuất phát 1 đến tất cả các đỉnh còn lại của đồ thị



THUẬT TOÁN BELLMAN – FORD (- MOORE)

Tìm đường đi ngắn nhất từ đỉnh xuất phát 1 đến tất cả các đỉnh còn lại của đồ thị

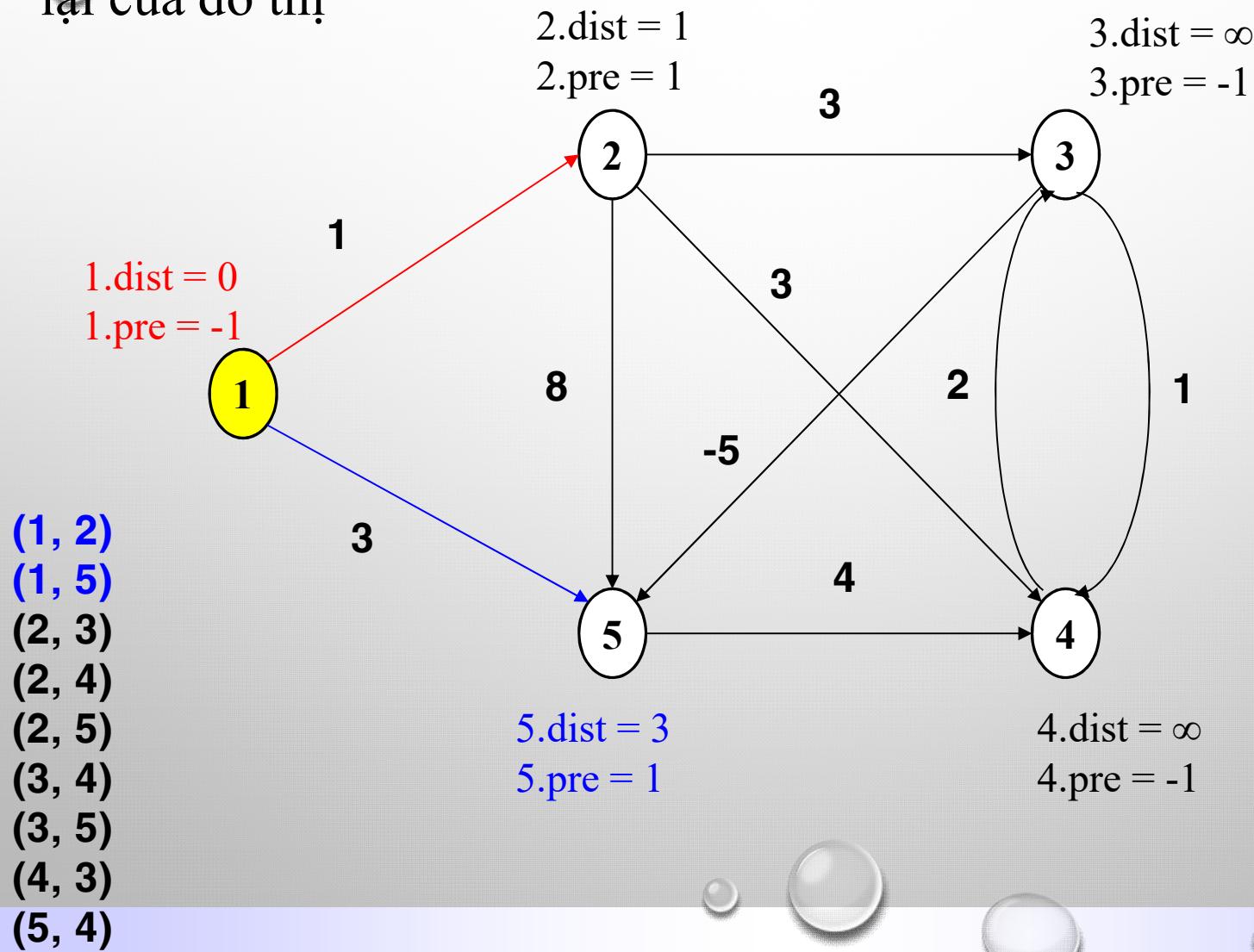


Xét đỉnh 1:

Duyệt qua tất cả các cung của đồ thị và tiến hành cập nhật nhãn tại các đỉnh nếu đường đi mới ngắn hơn đường đi cũ

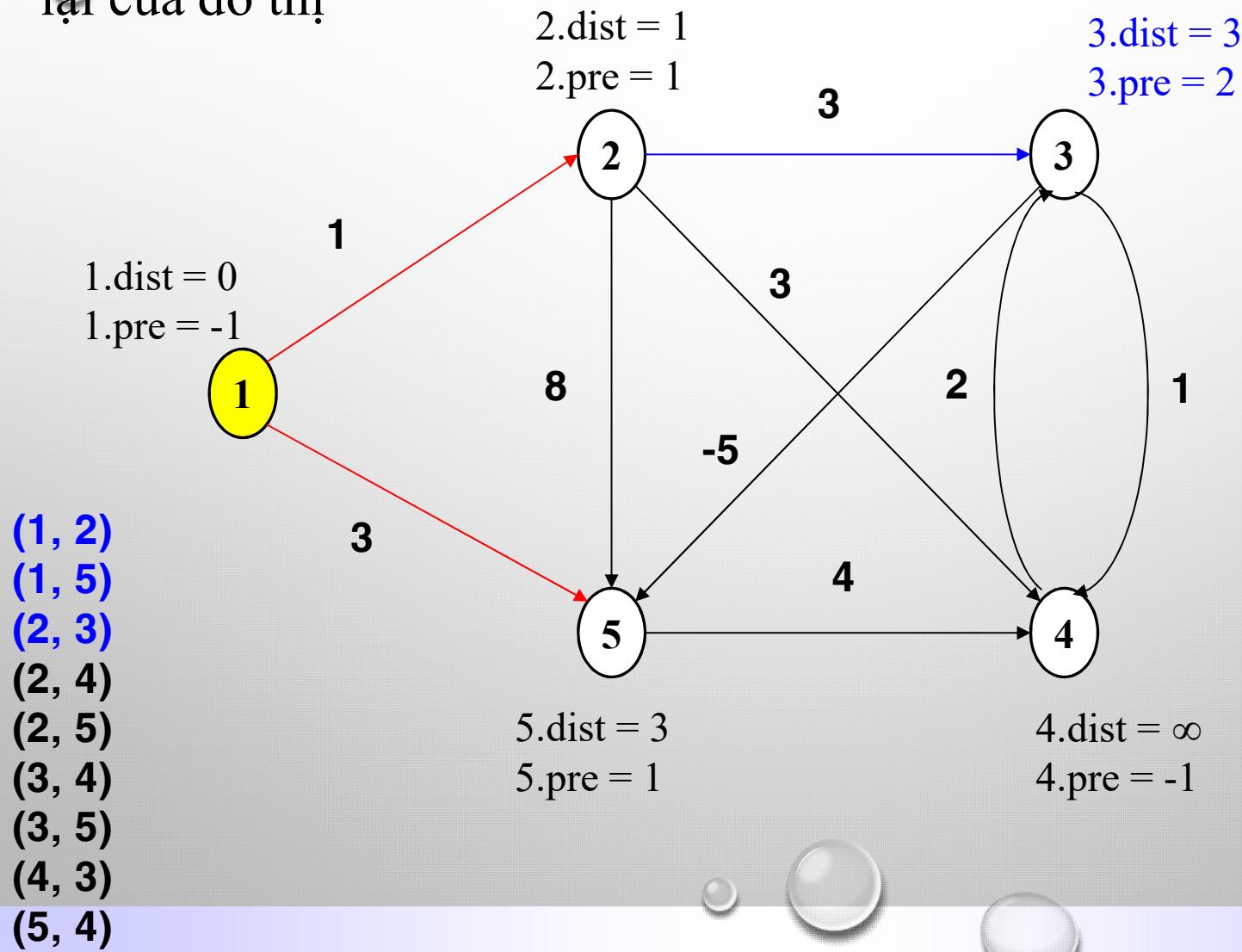
THUẬT TOÁN BELLMAN – FORD (- MOORE)

Tìm đường đi ngắn nhất từ đỉnh xuất phát 1 đến tất cả các đỉnh còn lại của đồ thị



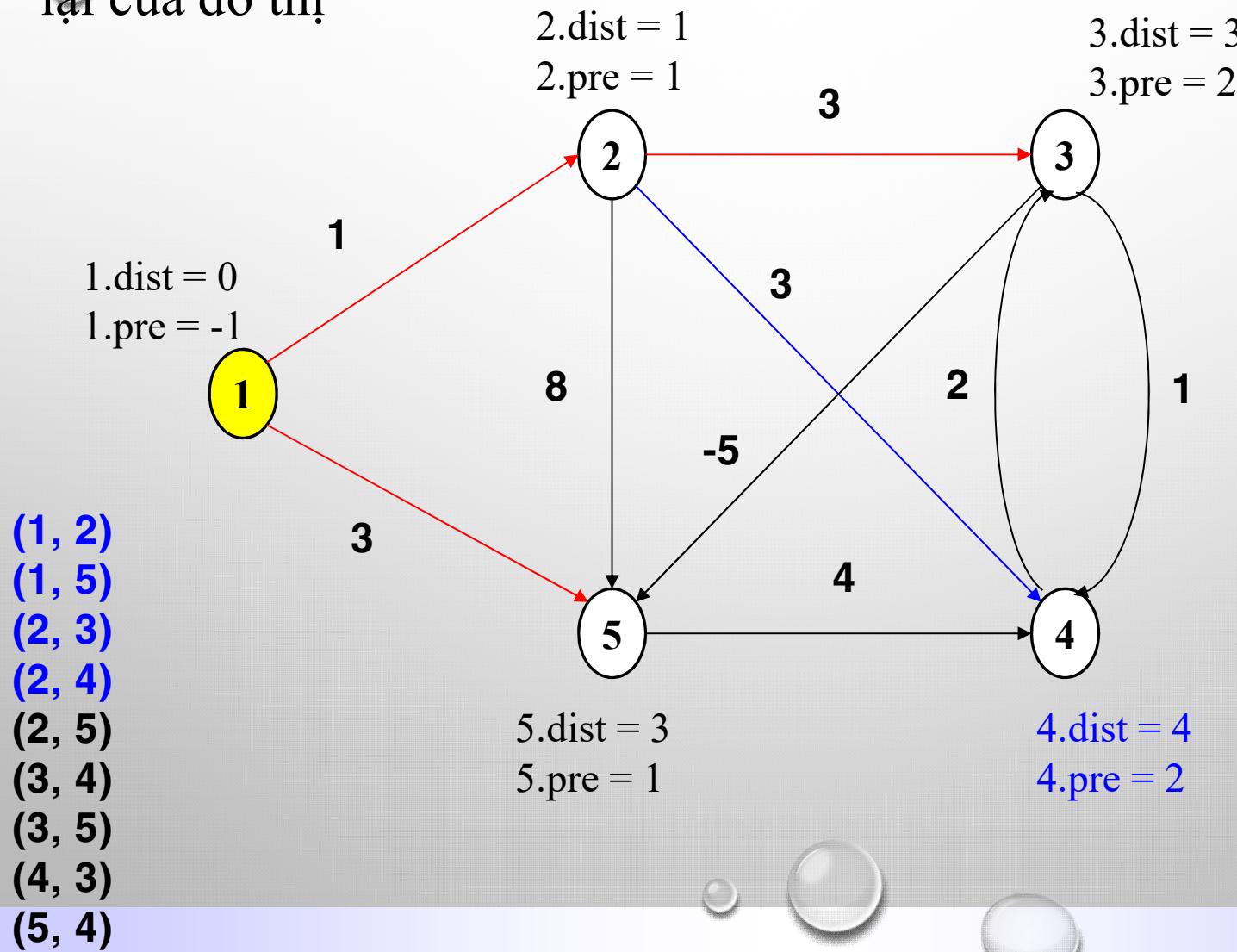
THUẬT TOÁN BELLMAN – FORD (- MOORE)

Tìm đường đi ngắn nhất từ đỉnh xuất phát 1 đến tất cả các đỉnh còn lại của đồ thị



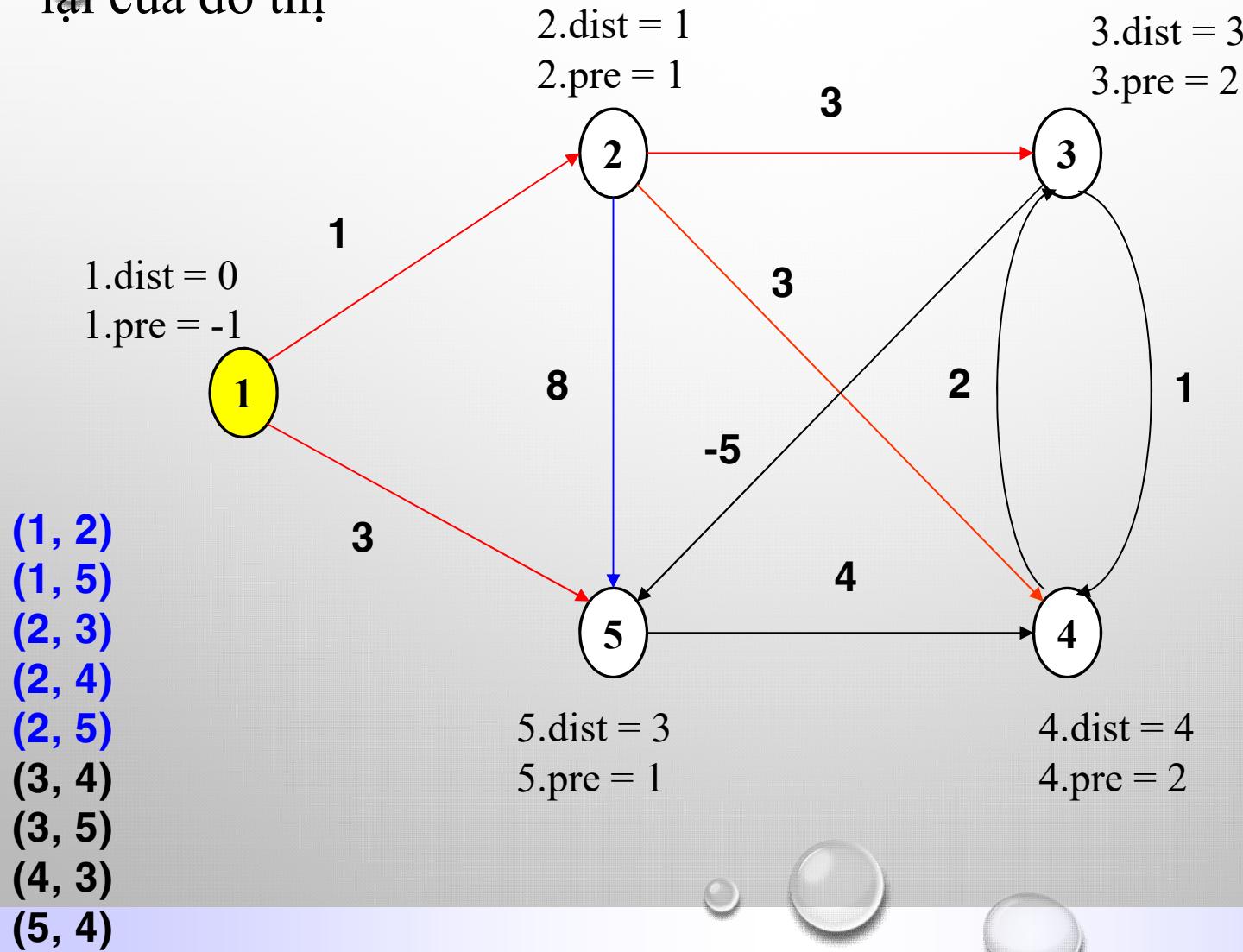
THUẬT TOÁN BELLMAN – FORD (- MOORE)

Tìm đường đi ngắn nhất từ đỉnh xuất phát 1 đến tất cả các đỉnh còn lại của đồ thị



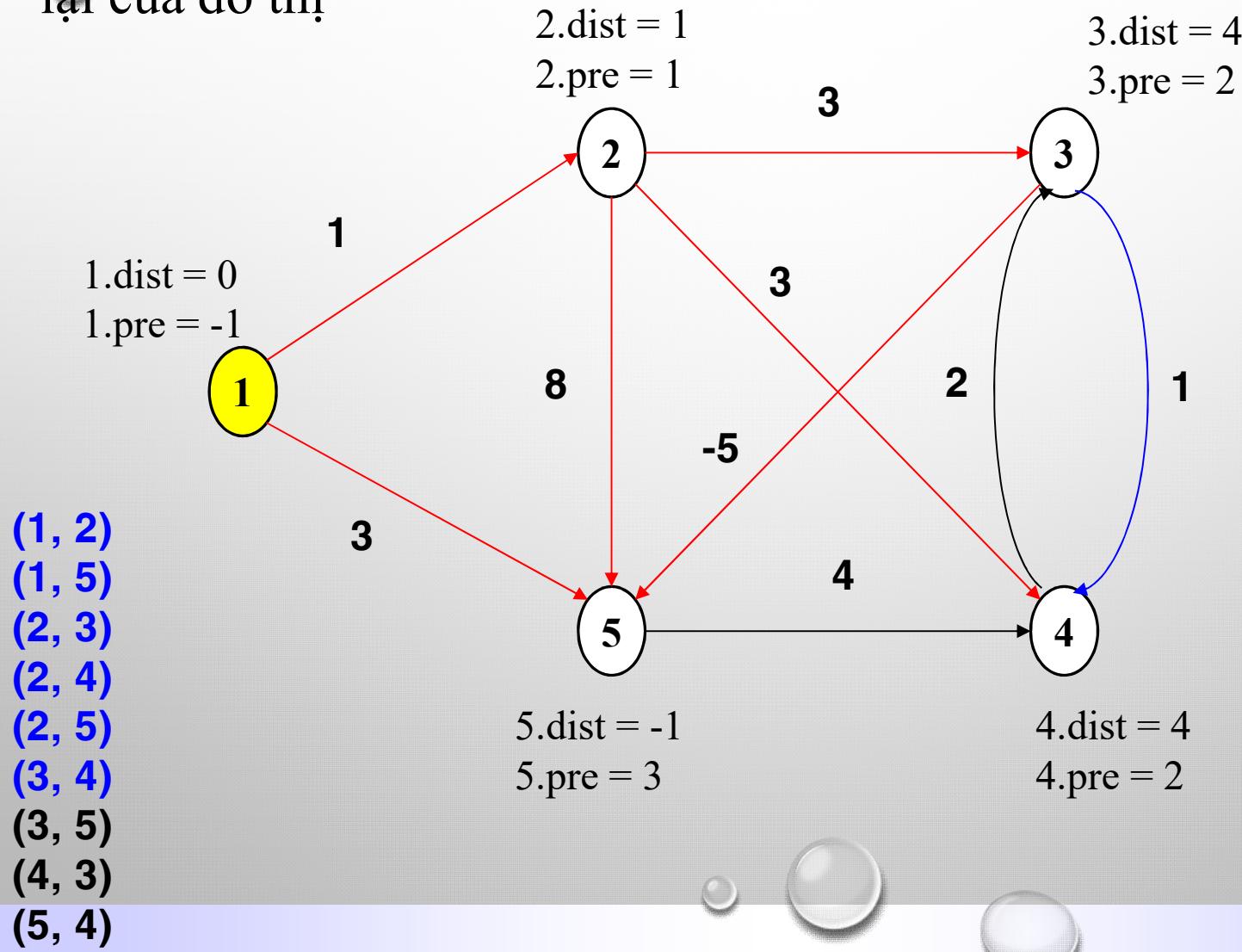
THUẬT TOÁN BELLMAN – FORD (- MOORE)

Tìm đường đi ngắn nhất từ đỉnh xuất phát 1 đến tất cả các đỉnh còn lại của đồ thị



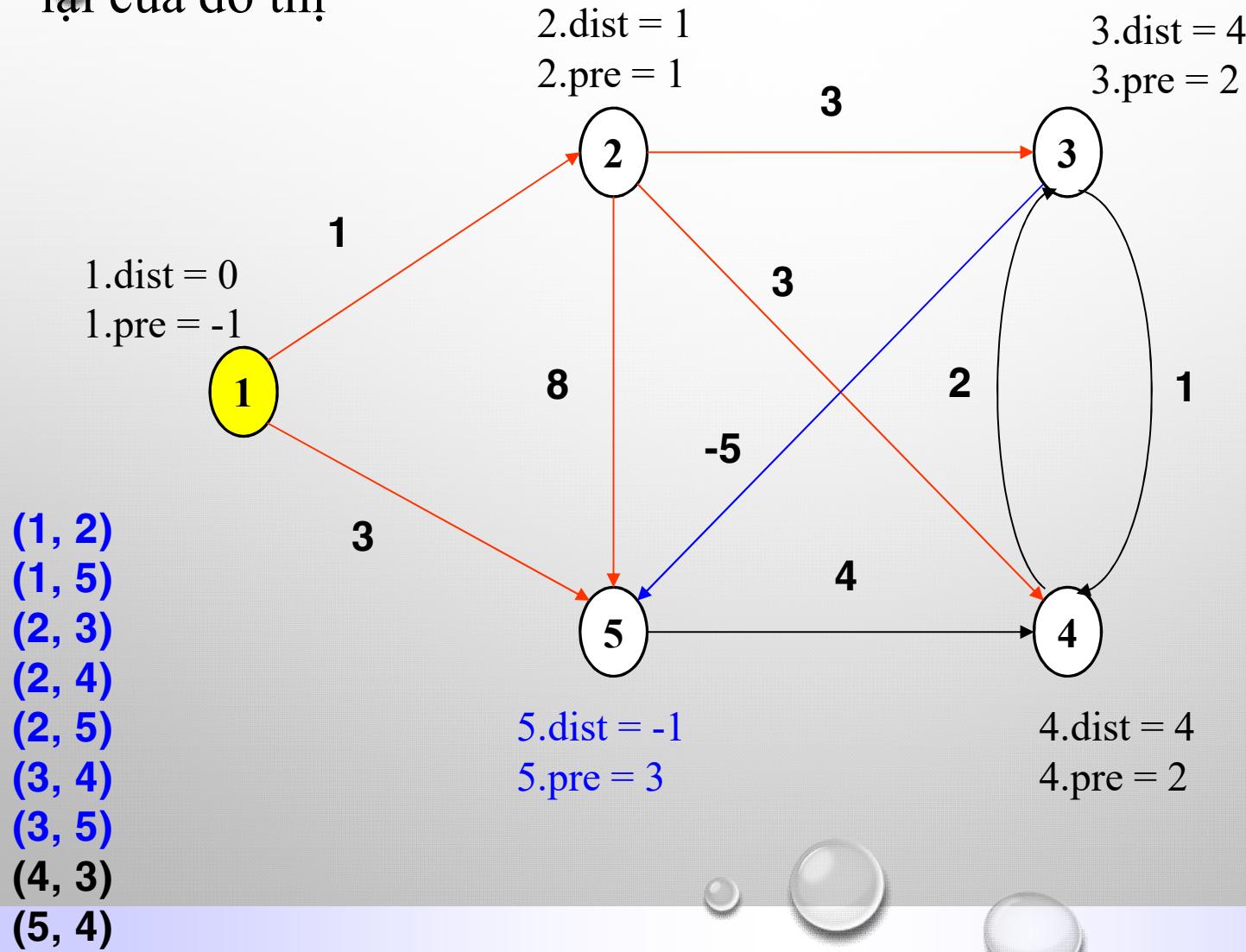
THUẬT TOÁN BELLMAN – FORD (- MOORE)

Tìm đường đi ngắn nhất từ đỉnh xuất phát 1 đến tất cả các đỉnh còn lại của đồ thị



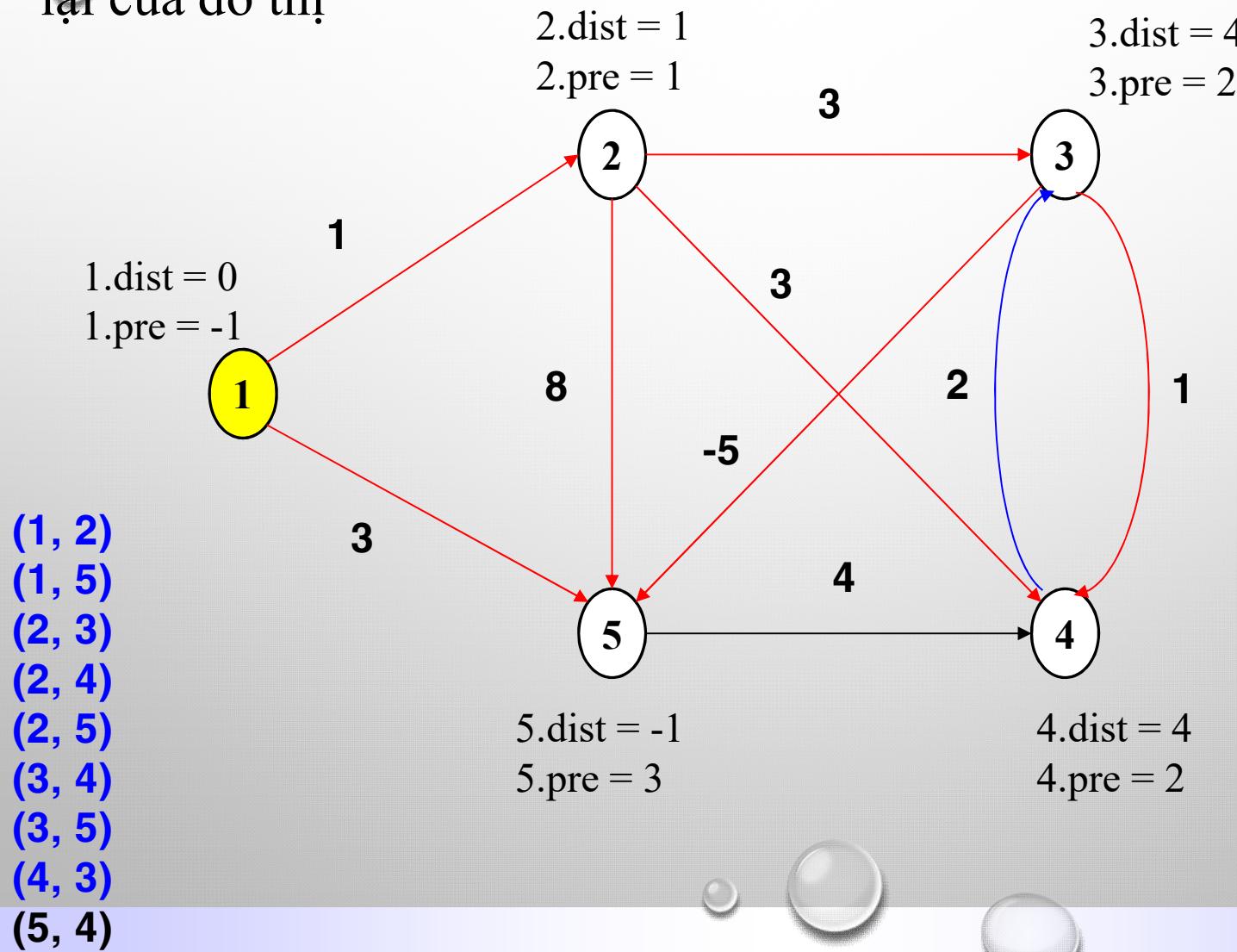
THUẬT TOÁN BELLMAN – FORD (- MOORE)

Tìm đường đi ngắn nhất từ đỉnh xuất phát 1 đến tất cả các đỉnh còn lại của đồ thị



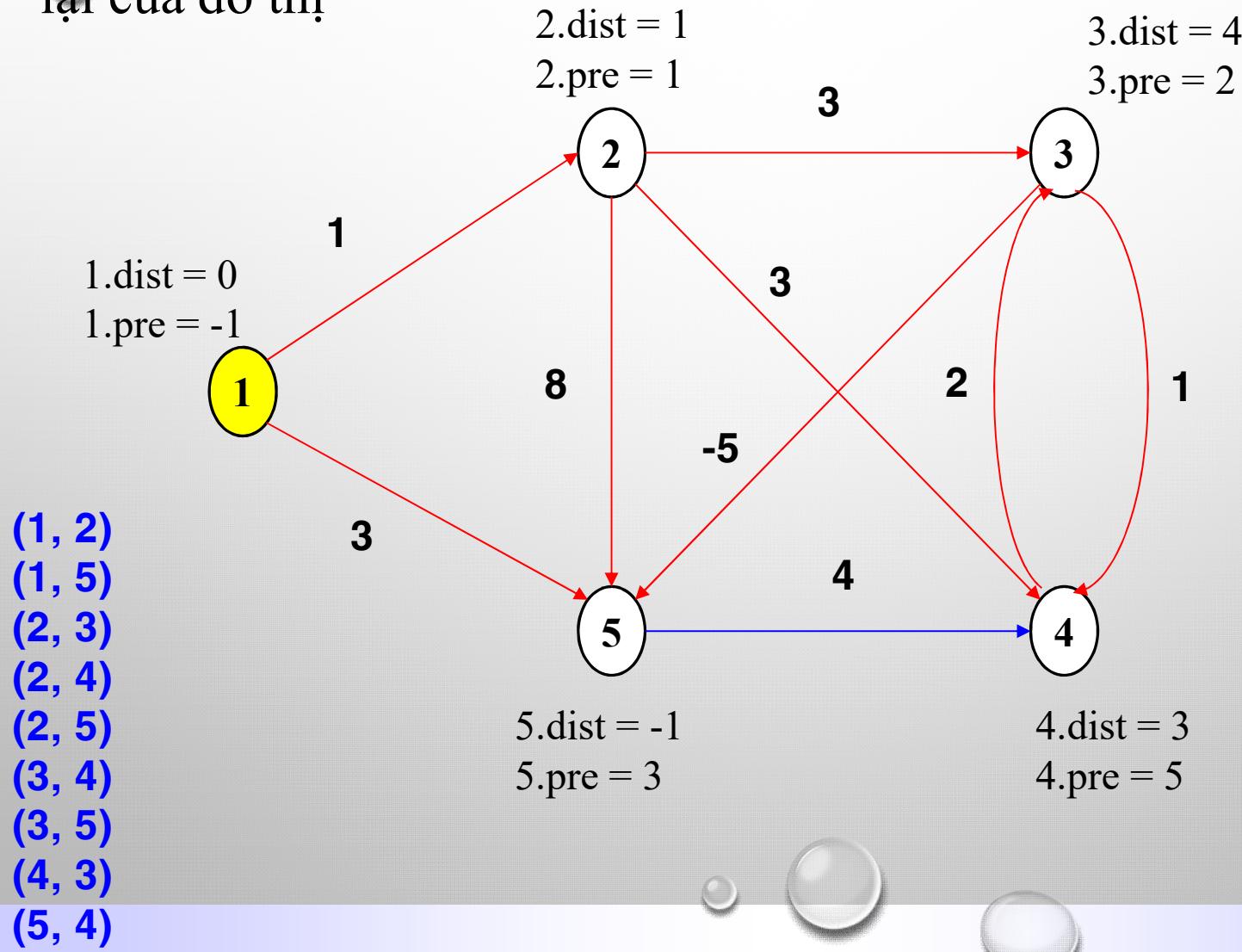
THUẬT TOÁN BELLMAN – FORD (- MOORE)

Tìm đường đi ngắn nhất từ đỉnh xuất phát 1 đến tất cả các đỉnh còn lại của đồ thị



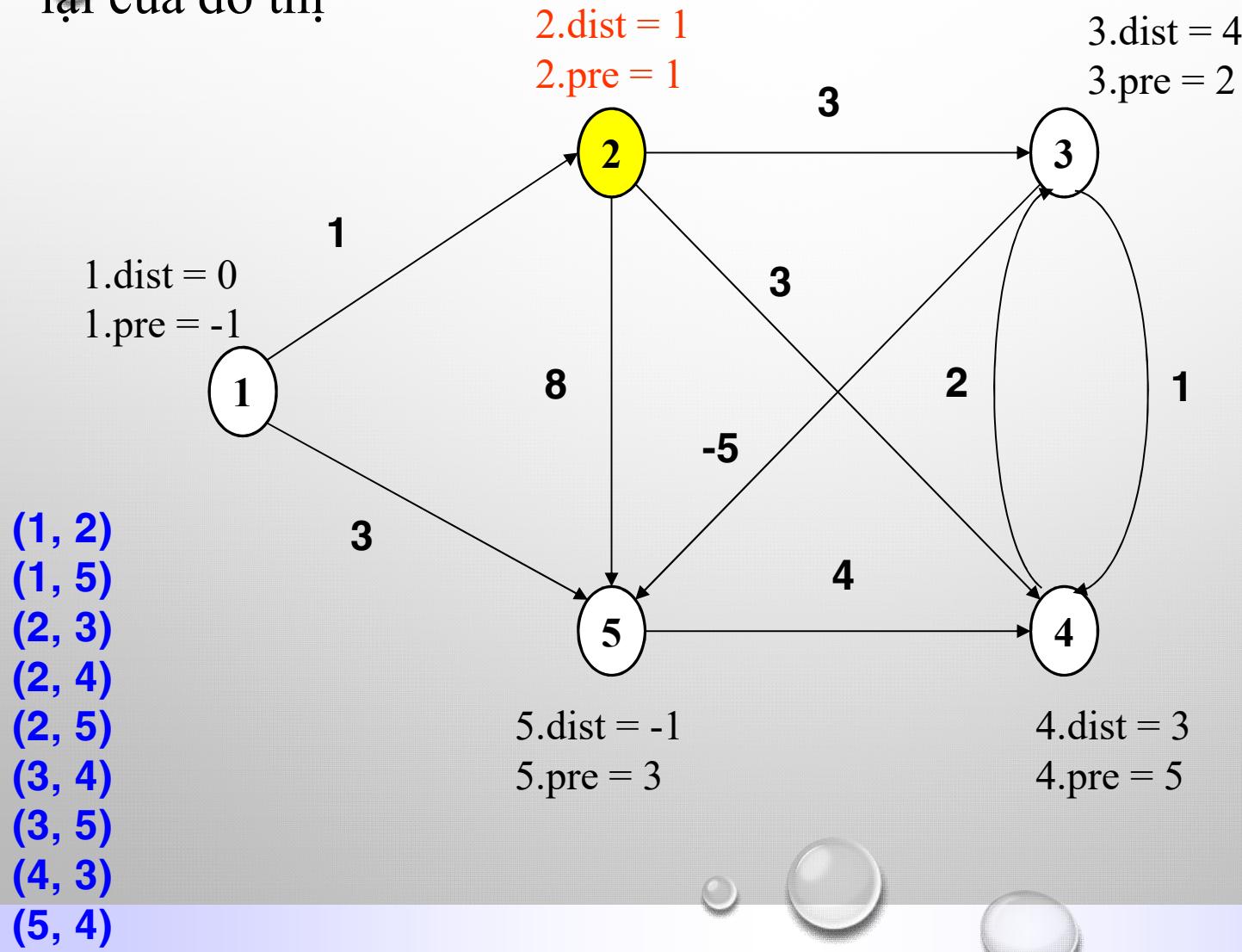
THUẬT TOÁN BELLMAN – FORD (- MOORE)

Tìm đường đi ngắn nhất từ đỉnh xuất phát 1 đến tất cả các đỉnh còn lại của đồ thị



THUẬT TOÁN BELLMAN – FORD (- MOORE)

Tìm đường đi ngắn nhất từ đỉnh xuất phát 1 đến tất cả các đỉnh còn lại của đồ thị

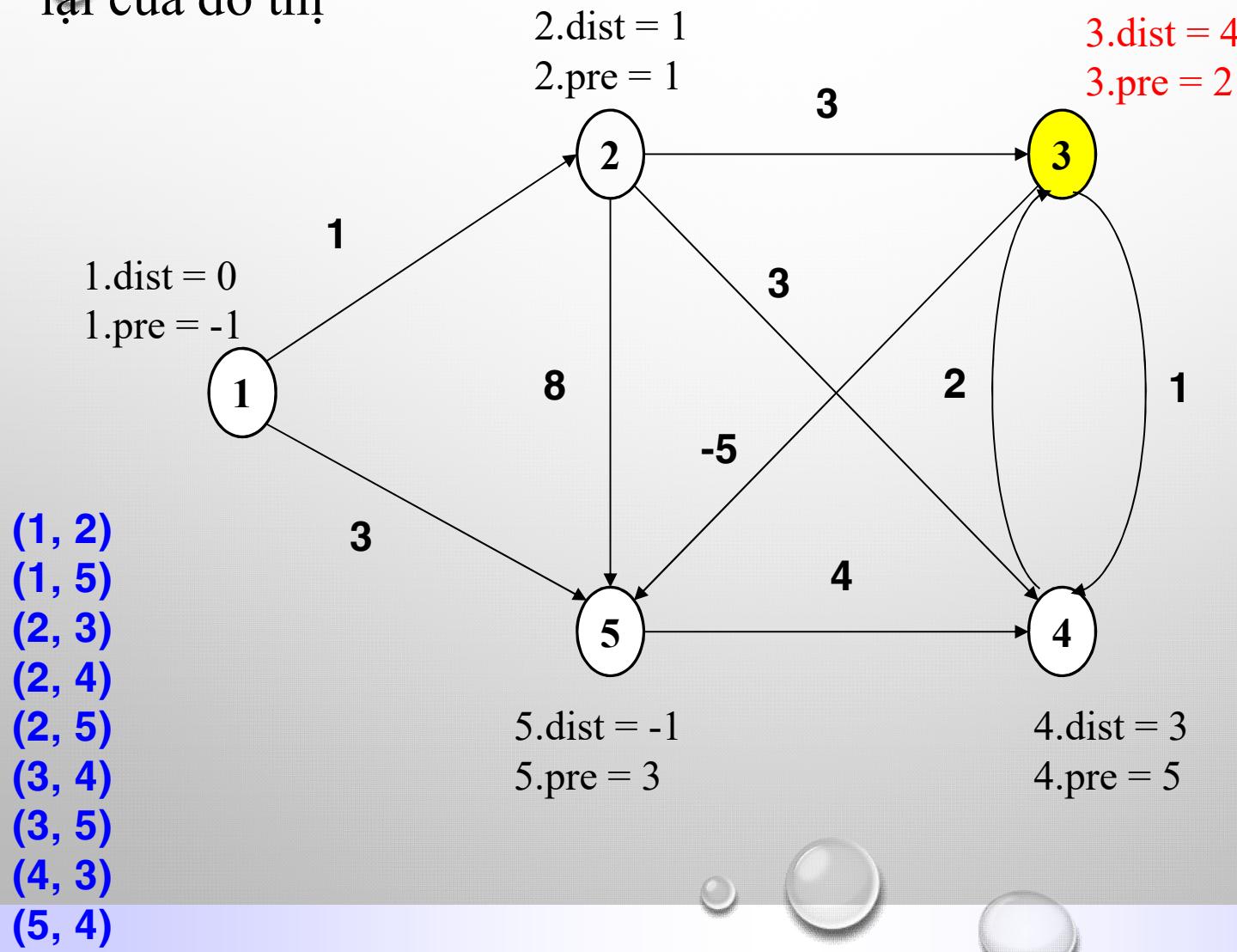


Xét đỉnh 2:

Duyệt qua tất cả các cung của đồ thị và tiến hành cập nhật nhãn tại các đỉnh nếu đường đi mới ngắn hơn đường đi cũ

THUẬT TOÁN BELLMAN – FORD (- MOORE)

Tìm đường đi ngắn nhất từ đỉnh xuất phát 1 đến tất cả các đỉnh còn lại của đồ thị

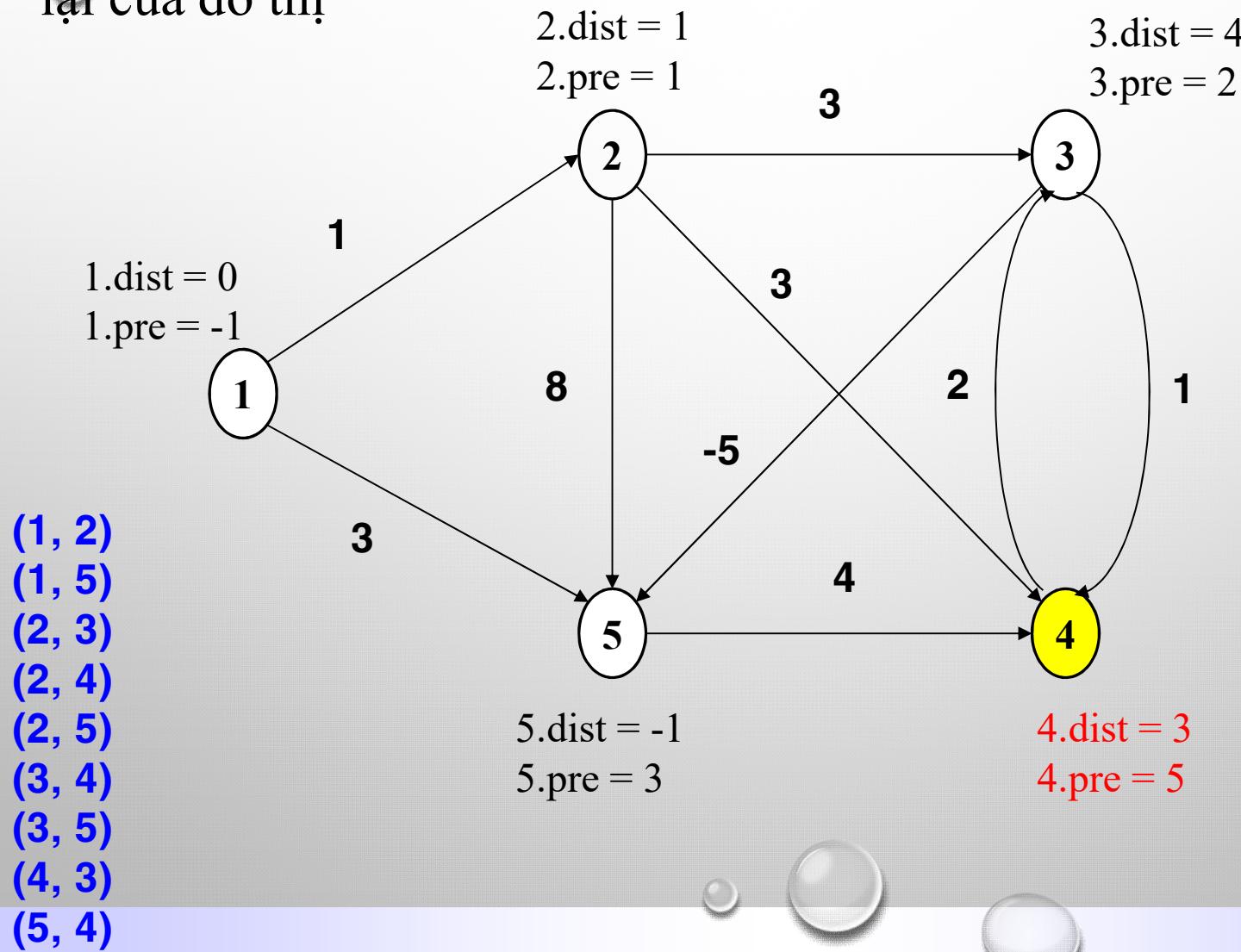


Xét đỉnh 3:

Duyệt qua tất cả các cung của đồ thị và tiến hành cập nhật nhãn tại các đỉnh nếu đường đi mới ngắn hơn đường đi cũ

THUẬT TOÁN BELLMAN – FORD (- MOORE)

Tìm đường đi ngắn nhất từ đỉnh xuất phát 1 đến tất cả các đỉnh còn lại của đồ thị

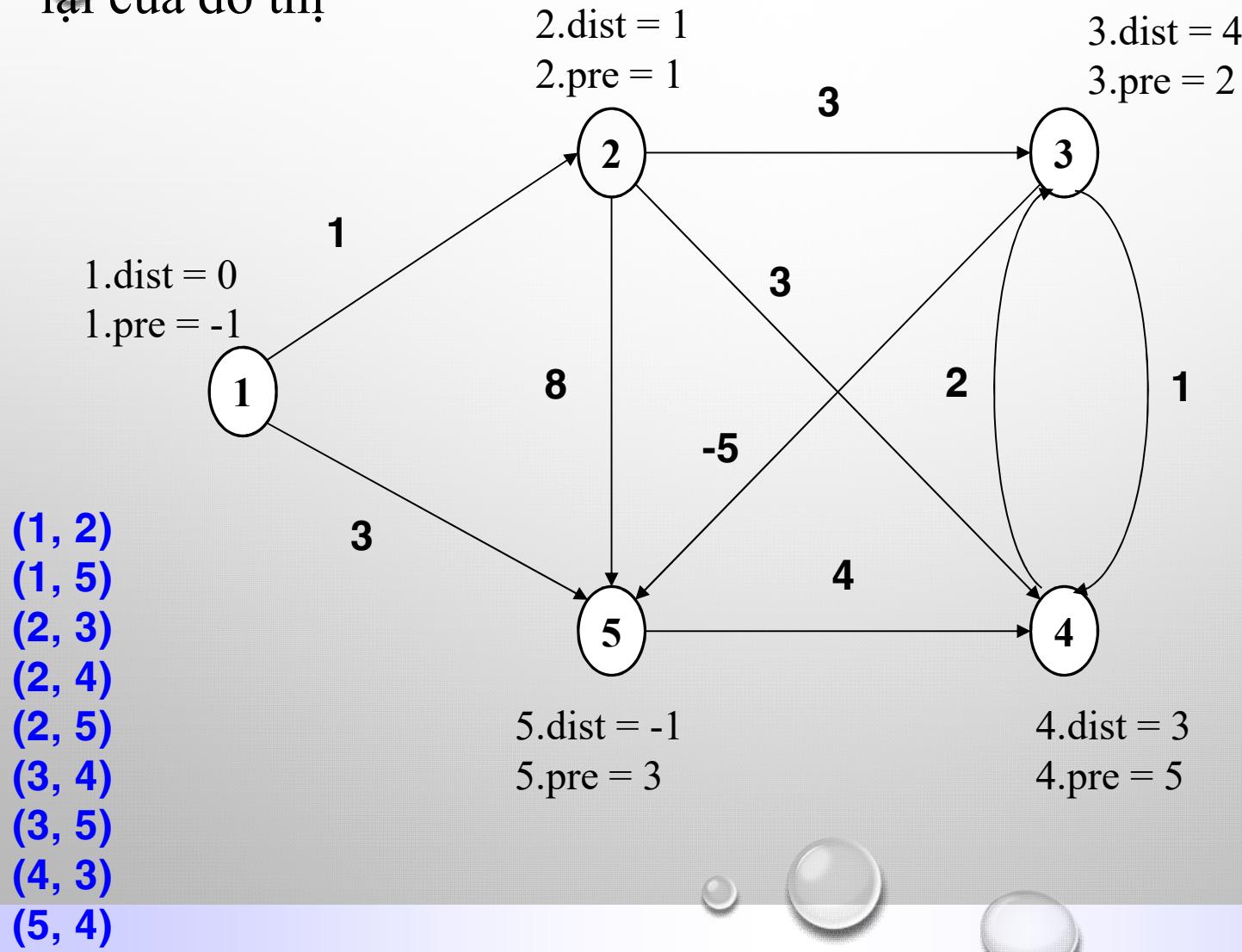


Xét đỉnh 4:

Duyệt qua tất cả các cung của đồ thị và tiến hành cập nhật nhãn tại các đỉnh nếu đường đi mới ngắn hơn đường đi cũ

THUẬT TOÁN BELLMAN – FORD (- MOORE)

Tìm đường đi ngắn nhất từ đỉnh xuất phát 1 đến tất cả các đỉnh còn lại của đồ thị



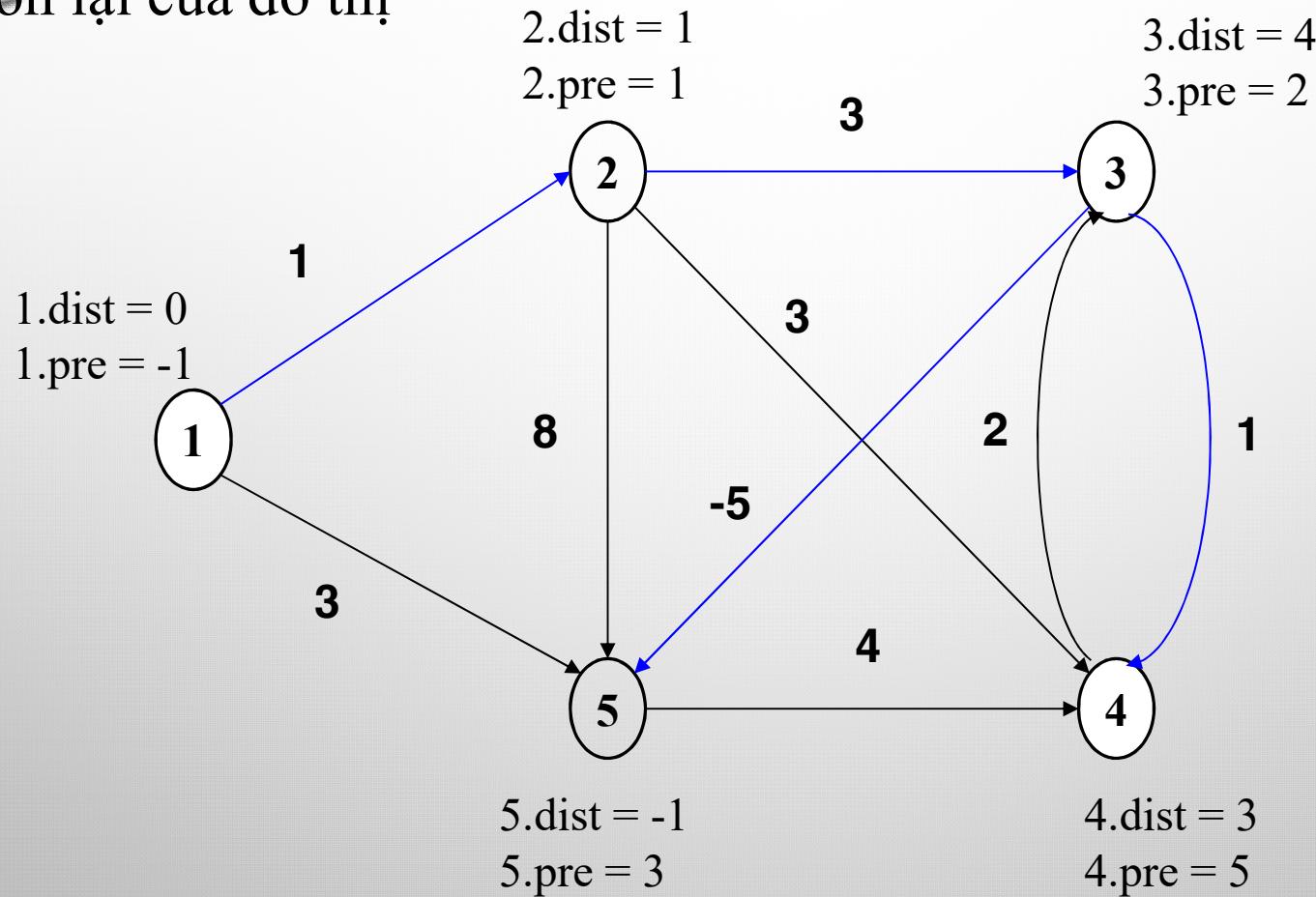
Phát hiện chu trình âm:

Duyệt qua tất cả các cung của đồ thị một lần nữa.

Nếu còn có thể cập nhật giá trị đường đi thì ta kết luận đồ thị có chu trình âm. Ngược lại thì không có chu trình âm.

THUẬT TOÁN BELLMAN – FORD (- MOORE)

Kết quả đường đi ngắn nhất từ đỉnh xuất phát 1 đến tất cả các đỉnh còn lại của đồ thị

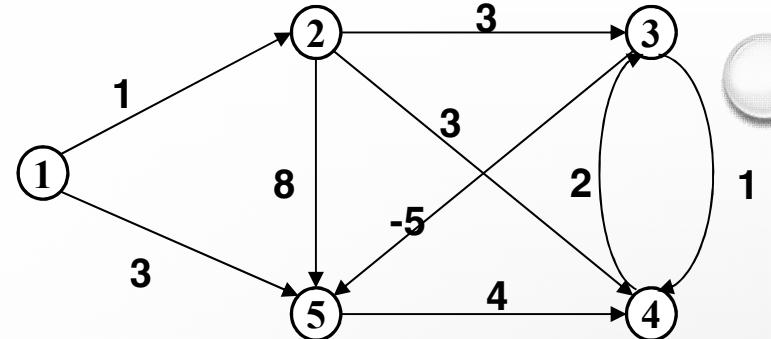


Ở bước lặp, giải thuật BF chỉ duyệt n-1 đỉnh
Lần lặp thứ n dùng để phát hiện chu trình âm

THUẬT TOÁN BELLMAN – FORD (- MOORE)

Chạy thuật toán bằng cách lập bảng:

	1	2	3	4	5
KT	(0,-)	(∞,-)	(∞,-)	(∞,-)	(∞,-)

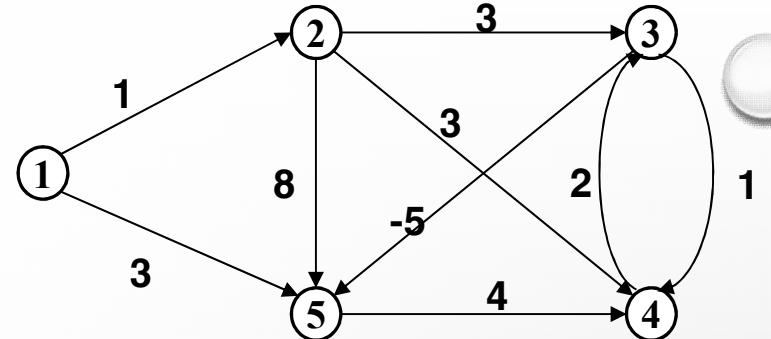


		Định 1:				
(x,y)	w	1	2	3	4	5
(1,2)	1		(1,1)			
(1,5)	3					(3,1)
(2,3)	3			(4,2)		
(2,4)	3				(4,2)	
(2,5)	8					(3,1)
(3,4)	1			(4,2)		
(3,5)	-5					(-1,3)
(4,3)	2			(4,2)		
(5,4)	4				(3,5)	
KQ		(0,-)	(1,1)	(4,2)	(3,5)	(-1,3)

		Định 2:				
1	2	3	4	5		
		(1,1)				
						(-1,3)
			(4,2)			
					(3,5)	
						(-1,3)
					(3,5)	
						(-1,3)
			(4,2)			
					(3,5)	
KQ		(0,-)	(1,1)	(4,2)	(3,5)	(-1,3)

THUẬT TOÁN BELLMAN – FORD (- MOORE)

	1	2	3	4	5
KQ 2	(0,-)	(1,1)	(4,2)	(3,5)	(-1,3)



		Định 3:				
(x,y)	w	1	2	3	4	5
(1,2)	1		(1,1)			
(1,5)	3					(-1,3)
(2,3)	3			(4,2)		
(2,4)	3				(3,5)	
(2,5)	8					(-1,3)
(3,4)	1			(3,5)		
(3,5)	-5					(-1,3)
(4,3)	2		(4,2)			
(5,4)	4			(3,5)		
KQ		(0,-)	(1,1)	(4,2)	(3,5)	(-1,3)

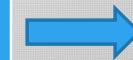
		Định 4:				
1	2	3	4	5		
		(1,1)				
					(-1,3)	
			(4,2)			
				(3,5)		
					(-1,3)	
				(3,5)		
					(-1,3)	
KQ		(0,-)	(1,1)	(4,2)	(3,5)	(-1,3)

THUẬT TOÁN BELLMAN – FORD (- MOORE)

BF cải tiến

*Nếu kết quả lần lặp
thứ i giống với kết quả
lần lặp thứ i-1 thì giải
thuật dừng*

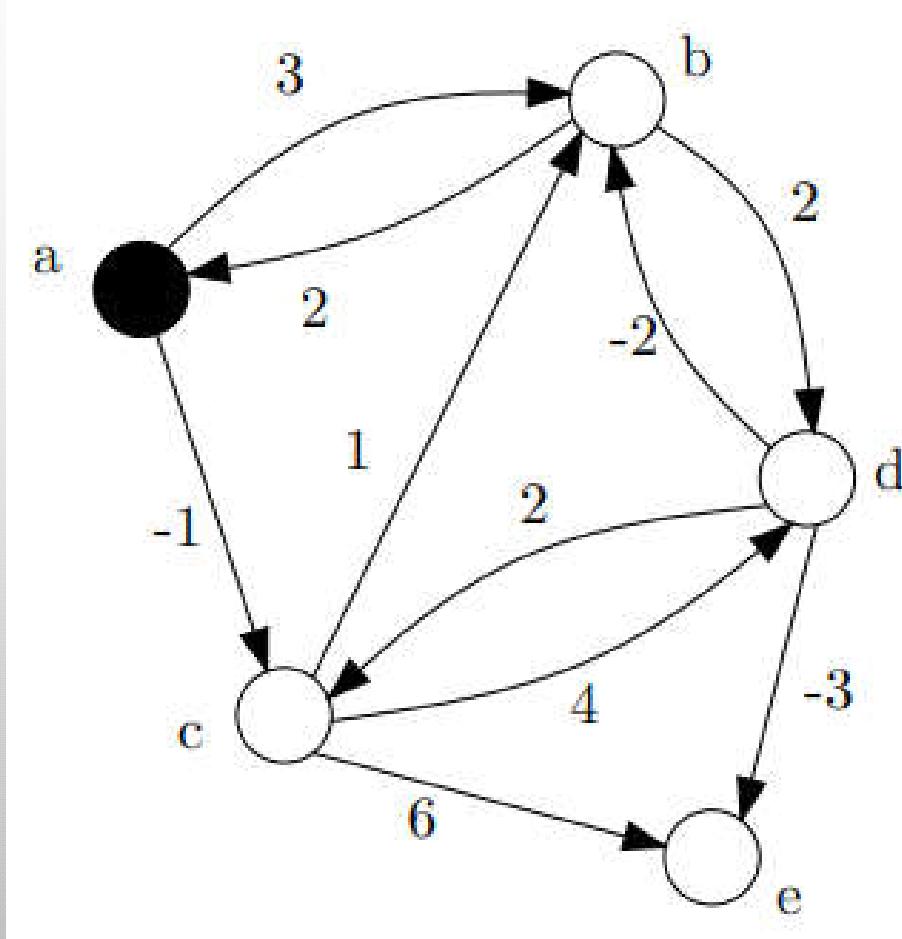
	1	2	3	4	5
KT	(0,-)	(∞ , -)			
KQ 1	(0,-)	(1,1)	(4,2)	(3,5)	(-1,3)
KQ 2	(0,-)	(1,1)	(4,2)	(3,5)	(-1,3)



*Vì kết quả
lần lặp 2
giống với
kết quả lần
lặp 1 nên
thuật toán
dừng*

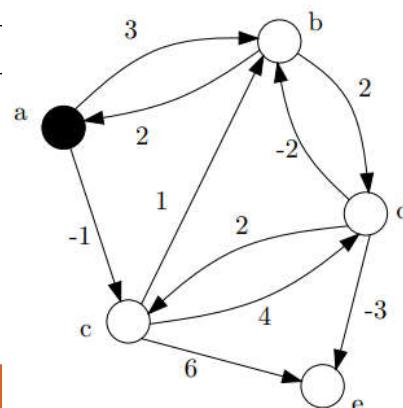
THUẬT TOÁN BELLMAN – FORD (- MOORE)

Bài tập: Tìm đường đi ngắn nhất từ đỉnh a đến tất cả các đỉnh còn lại của đồ thị



THUẬT TOÁN BEL

	a	b	c	d	e
KT	0,-1	$\infty, -1$	$\infty, -1$	$\infty, -1$	$\infty, -1$



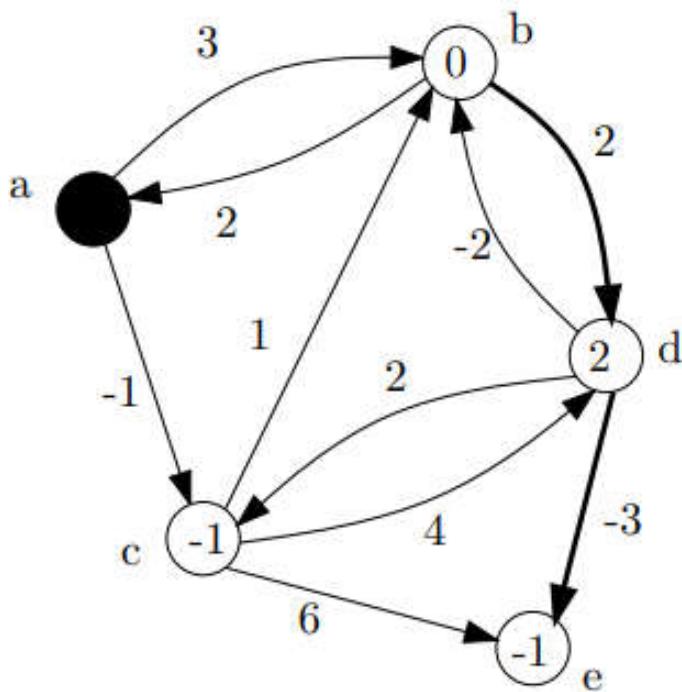
RD (- MOORE)

Định a:						Định b:					Định c:					
x,y	w	a	b	c	d	e	a	b	c	d	e	a	b	c	d	e
a,b	3		3,a					0,c					0,c			
a,c	-1			-1,a					-1,a					-1,a		
b,a	2	0,-1					0,-1					0,-1				
b,d	2				5,b					2,b				2,b		
c,b	1		0,c					0,c					0,c			
c,d	4			3,c					2,b					2,b		
c,e	6				5,c					0,d					-1,d	
d,b	-2		0,c				0,c					0,c				
d,c	2			-1,a				-1,a					-1,a			
d,e	-3				0,d					-1,d				-1,d		
KQ		0,-1	0,c	-1,a	3,c	0,d	0,-1	0,c	-1,a	2,b	-1,d	0,-1	0,c	-1,a	2,b	-1,d

THUẬT TOÁN BELLMAN – FORD (- MOORE)

Các giá trị tại lần lặp đỉnh c không đổi so với kết quả lần lặp đỉnh b trước đó nên giải thuật dừng

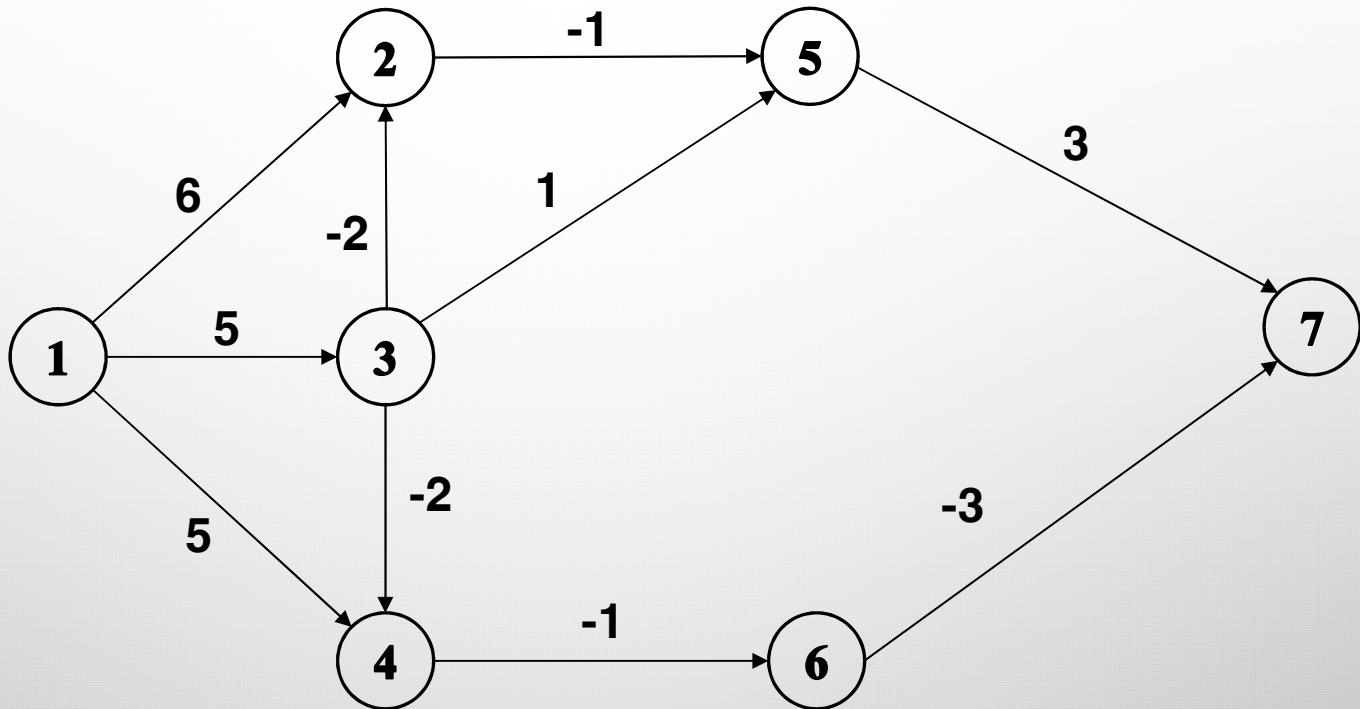
a	b	c	d	e
0,-1	0,c	-1,a	2,b	-1,d



THUẬT TOÁN BELLMAN – FORD (- MOORE)

Bài tập: Tìm đường đi ngắn nhất từ đỉnh 1 đến tất cả các đỉnh còn lại của đồ thị

x,y	w
1,2	6
1,3	5
1,4	5
2,5	-1
3,2	-2
3,4	-2
3,5	1
4,6	-1
5,7	3
6,7	-3



Yêu cầu: Viết tay bài làm ra giấy và nộp cho cô

Hướng dẫn:

Chạy thuật toán (lập bảng)

Vẽ đồ thị đường đi

THUẬT TOÁN FLOYD – WARSHALL

Thuật toán Floyd – Warshall còn được gọi là thuật toán Floyd, thuật toán Roy–Warshall, thuật toán Roy–Floyd, hay thuật toán WFI.



Awards	Turing Award (1978) Computer Pioneer Award (1991)
--------	--

Scientific career	
Fields	Computer science
Institutions	Illinois Institute of Technology Carnegie Mellon University Stanford University

Professor	
Robert W Floyd	
Born	June 8, 1936 New York City, New York , United States
Died	September 25, 2001 (aged 65) Stanford, California, United States
Citizenship	United States
Education	University of Chicago (B.A., 1953, 1958)
Known for	Floyd–Warshall algorithm Floyd–Steinberg dithering Floyd's cycle-finding algorithm Floyd's triangle ALGOL

THUẬT TOÁN FLOYD – WARSHALL

Thuật toán Floyd – Warshall còn được gọi là thuật toán Floyd, thuật toán Roy–Warshall, thuật toán Roy–Floyd, hay thuật toán WFI.



Stephen Warshall	
Born	November 15, 1935 New York City
Died	December 11, 2006 (aged 71) Gloucester, MA
Known for	Floyd–Warshall algorithm

THUẬT TOÁN FLOYD – WARSHALL

Thuật toán Floyd – Warshall còn được gọi là thuật toán Floyd, thuật toán Roy–Warshall, thuật toán Roy–Floyd, hay thuật toán WFI.



Bernard Roy

France

Born	15 March 1934
Died	28 October 2017

<https://www.euro-online.org/web/pages/1622/bernard-roy-1934-2017>

THUẬT TOÁN FLOYD – WARSHALL

Thuật toán Floyd – Warshall tìm đường đi ngắn nhất **giữa tất cả các cặp đỉnh** của **đồ thị có trọng số tùy ý** nhưng **không có chu trình âm**

let dist be a $|V| \times |V|$ array of minimum distances initialized to infinity

let next be a $|V| \times |V|$ array of vertex indices initialized to **null**

procedure *FloydWarshallWithPathReconstruction()* **is**

for each edge (u, v) **do**

$\text{dist}[u][v] \leftarrow w(u, v)$ // *The weight of the edge (u, v)*

$\text{next}[u][v] \leftarrow v$

for each vertex v **do**

$\text{dist}[v][v] \leftarrow 0$

$\text{next}[v][v] \leftarrow v$

for k **from** 1 **to** $|V|$ **do** // *standard Floyd-Warshall implementation*

for i **from** 1 **to** $|V|$

for j **from** 1 **to** $|V|$

if $\text{dist}[i][j] > \text{dist}[i][k] + \text{dist}[k][j]$ **then**

$\text{dist}[i][j] \leftarrow \text{dist}[i][k] + \text{dist}[k][j]$

$\text{next}[i][j] \leftarrow \text{next}[i][k]$

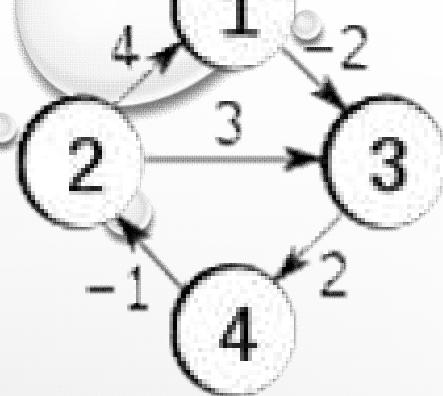
THUẬT TOÁN FLOYD – WARSHALL

Thuật toán Floyd – Warshall tìm đường đi ngắn nhất giữa tất cả các cặp đỉnh của **đồ thị có trọng số tùy** nhưng **không có chu trình âm**

```
procedure Path(u, v)
    if next[u][v] = null then
        return []
    path = [u]
    while u ≠ v
        u ← next[u][v]
        path.append(u)
    return path
```

L.T.P.D.

THUẬT TOÁN FLOYD – WARSHALL



for k from 1 to |V| do

for i from 1 to |V|

for j from 1 to |V|

if dist[i][j] > dist[i][k] + dist[k][j] then

dist[i][j] ← dist[i][k] + dist[k][j]

next[i][j] ← next[i][k]

k=1:

$d(1,1)+d(1,1)=0 \rightarrow d(1,1)=0; d(1,2)=\infty;$

$d(1,3)=\infty; d(1,4)=\infty$

```

for each edge (u, v) do
    dist[u][v] ← w(u, v)
    next[u][v] ← v
for each vertex v do
    dist[v][v] ← 0
    next[v][v] ← v
for k from 1 to |V| do
    for i from 1 to |V|
        for j from 1 to |V|
            if dist[i][j] > dist[i][k] + dist[k][j] then
                dist[i][j] ← dist[i][k] + dist[k][j]
                next[i][j] ← next[i][k]

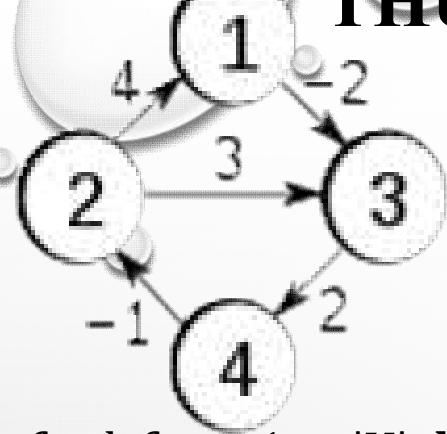
```

		j			
		1	2	3	4
i	k = 0	0,1	$\infty, 2$	-2,3	$\infty, 4$
	1	4,1	0,2	3,3	$\infty, 4$
	2	$\infty, 1$	$\infty, 2$	0,3	2,4
	3	$\infty, 1$	-1,2	$\infty, 3$	0,4

		j			
		1	2	3	4
i	k = 1				
	1				
	2				
	3				
	4				

L.T.P.D.

THUẬT TOÁN FLOYD – WARSHALL



for k from 1 to |V| do

for i from 1 to |V|

for j from 1 to |V|

if dist[i][j] > dist[i][k] + dist[k][j] then

dist[i][j] ← dist[i][k] + dist[k][j]

next[i][j] ← next[i][k]

k=1:

$d(1,1)+d(1,1)=0 \rightarrow d(1,1)=0; d(1,2)=\infty;$

$d(1,3)=\infty; d(1,4)=\infty$

$d(2,1)+d(1,1)=4 \rightarrow d(2,1)=4; d(2,2)=\infty;$

$d(2,1)+d(1,3)=3-1=2 \rightarrow \text{d}(2,3)=2;$

$d(2,4)=\infty$ (vì $d(1,4)=\infty$)

```

for each edge (u, v) do
    dist[u][v] ← w(u, v)
    next[u][v] ← v
for each vertex v do
    dist[v][v] ← 0
    next[v][v] ← v
for k from 1 to |V| do
    for i from 1 to |V|
        for j from 1 to |V|
            if dist[i][j] > dist[i][k] + dist[k][j] then
                dist[i][j] ← dist[i][k] + dist[k][j]
                next[i][j] ← next[i][k]

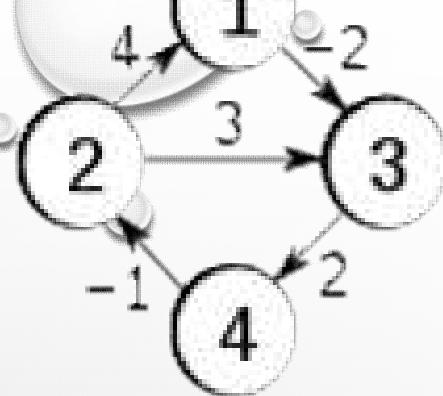
```

		j			
		1	2	3	4
i	k = 0	0,1	$\infty, 2$	-2,3	$\infty, 4$
	1	4,1	0,2	3,3	$\infty, 4$
	2	$\infty, 1$	$\infty, 2$	0,3	2,4
	3	$\infty, 1$	-1,2	$\infty, 3$	0,4

		j			
		1	2	3	4
i	k = 1	0,1	$\infty, 2$	-2,3	$\infty, 4$
	1				
	2				
	3				
	4				

L.T.P.D.

THUẬT TOÁN FLOYD – WARSHALL



for k from 1 to |V| do

 for i from 1 to |V|

 for j from 1 to |V|

 if $\text{dist}[i][j] > \text{dist}[i][k] + \text{dist}[k][j]$ then

$\text{dist}[i][j] \leftarrow \text{dist}[i][k] + \text{dist}[k][j]$

$\text{next}[i][j] \leftarrow \text{next}[i][k]$

k=1:

$d(2,1)+d(1,1)=4 \rightarrow d(2,1)=4; d(2,2)=\infty;$

$d(2,1)+d(1,3)=3-1=2 \rightarrow \text{d}(2,3)=2;$

$d(2,4)=\infty$ (vì $d(1,4)=\infty$)

$d(3,1)=\infty; d(3,2)=\infty$ (vì $d(3,1)=\infty$);

$d(3,3)=0; d(3,4)=2$

```

for each edge (u, v) do
    dist[u][v] ← w(u, v)
    next[u][v] ← v
for each vertex v do
    dist[v][v] ← 0
    next[v][v] ← v
for k from 1 to |V| do
    for i from 1 to |V|
        for j from 1 to |V|
            if dist[i][j] > dist[i][k] + dist[k][j] then
                dist[i][j] ← dist[i][k] + dist[k][j]
                next[i][j] ← next[i][k]

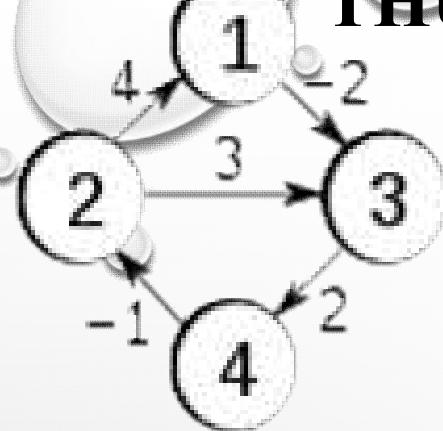
```

		j			
		1	2	3	4
i	k = 0	0,1	$\infty, 2$	-2,3	$\infty, 4$
	1	4,1	0,2	3,3	$\infty, 4$
	2	$\infty, 1$	$\infty, 2$	0,3	2,4
	3	$\infty, 1$	-1,2	$\infty, 3$	0,4

		j			
		1	2	3	4
i	k = 1	0,1	$\infty, 2$	-2,3	$\infty, 4$
	1	4,1	0,2	2,1	$\infty, 4$
	2				
	3				
	4				

L.T.P.D.

THUẬT TOÁN FLOYD – WARSHALL



for k from 1 to |V| do

for i from 1 to |V|

for j from 1 to |V|

if dist[i][j] > dist[i][k] + dist[k][j] then

dist[i][j] ← dist[i][k] + dist[k][j]

next[i][j] ← next[i][k]

k=1:

$d(3,1)=\infty$; $d(3,2)=\infty$ (vì $d(3,1)=\infty$);

$d(3,3)=0$; $d(3,4)=2$

$d(4,1)=\infty$; $d(4,2)=-1$; $d(4,3)=\infty$; $d(4,4)=0$

```

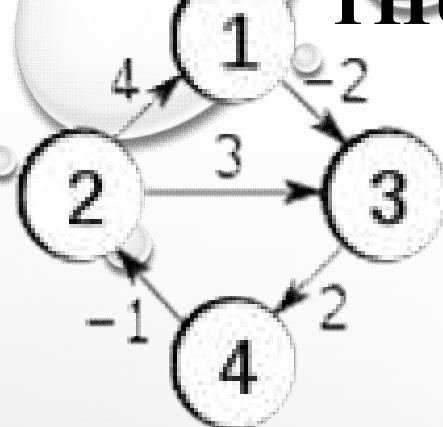
for each edge (u, v) do
    dist[u][v] ← w(u, v)
    next[u][v] ← v
for each vertex v do
    dist[v][v] ← 0
    next[v][v] ← v
for k from 1 to |V| do
    for i from 1 to |V|
        for j from 1 to |V|
            if dist[i][j] > dist[i][k] + dist[k][j] then
                dist[i][j] ← dist[i][k] + dist[k][j]
                next[i][j] ← next[i][k]
  
```

		j			
		1	2	3	4
i	$k = 0$	0,1	$\infty, 2$	-2,3	$\infty, 4$
	1	4,1	0,2	3,3	$\infty, 4$
	2	$\infty, 1$	$\infty, 2$	0,3	2,4
	3	$\infty, 1$	-1,2	$\infty, 3$	0,4

		j			
		1	2	3	4
i	$k = 1$	0,1	$\infty, 2$	-2,3	$\infty, 4$
	1	4,1	0,2	2,1	$\infty, 4$
	2	$\infty, 1$	$\infty, 2$	0,3	2,4
	3				

L.T.P.D.

THUẬT TOÁN FLOYD – WARSHALL



for k from 1 to |V| do

for i from 1 to |V|

for j from 1 to |V|

if dist[i][j] > dist[i][k] + dist[k][j] then

dist[i][j] ← dist[i][k] + dist[k][j]

next[i][j] ← next[i][k]

k=1:

d(4,1)=∞; d(4,2)=-1; d(4,3)=∞; d(4,4)=0

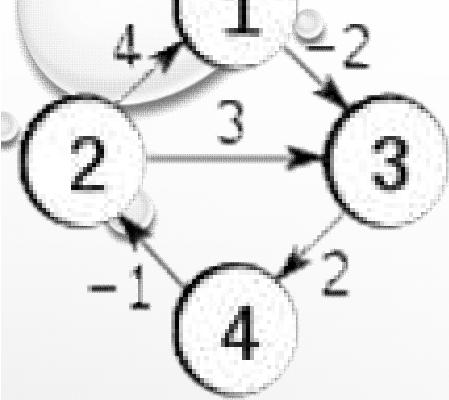
```

for each edge (u, v) do
    dist[u][v] ← w(u, v)
    next[u][v] ← v
for each vertex v do
    dist[v][v] ← 0
    next[v][v] ← v
for k from 1 to |V| do
    for i from 1 to |V|
        for j from 1 to |V|
            if dist[i][j] > dist[i][k] + dist[k][j] then
                dist[i][j] ← dist[i][k] + dist[k][j]
                next[i][j] ← next[i][k]
  
```

		j			
		1	2	3	4
i	k = 0	0,1	∞,2	-2,3	∞,4
	1	4,1	0,2	3,3	∞,4
	2	∞,1	∞,2	0,3	2,4
	3	∞,1	-1,2	∞,3	0,4

		j			
		1	2	3	4
i	k = 1	0,1	∞,2	-2,3	∞,4
	1	4,1	0,2	2,1	∞,4
	2	∞,1	∞,2	0,3	2,4
	3	∞,1	-1,2	∞,3	0,4

THUẬT TOÁN FLOYD – WARSHALL



```

for each edge  $(u, v)$  do
     $\text{dist}[u][v] \leftarrow w(u, v)$ 
     $\text{next}[u][v] \leftarrow v$ 
for each vertex  $v$  do
     $\text{dist}[v][v] \leftarrow 0$ 
     $\text{next}[v][v] \leftarrow v$ 

for  $k$  from 1 to  $|V|$  do
    for  $i$  from 1 to  $|V|$ 
        for  $j$  from 1 to  $|V|$ 
            if  $\text{dist}[i][j] > \text{dist}[i][k] + \text{dist}[k][j]$  then
                 $\text{dist}[i][j] \leftarrow \text{dist}[i][k] + \text{dist}[k][j]$ 
                 $\text{next}[i][j] \leftarrow \text{next}[i][k]$ 

```

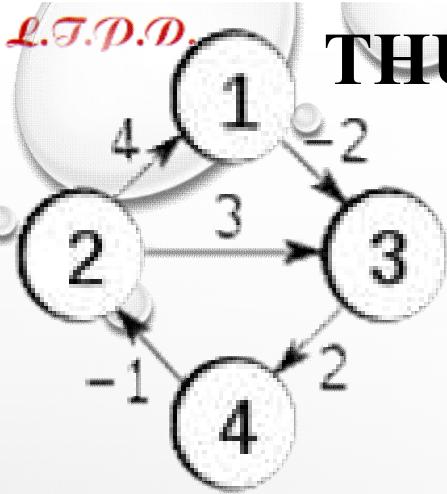
Nhận xét:

1. Tại bước lặp thứ k :
 - Giá trị tại dòng k không đổi
 - Giá trị tại cột k không đổi
2. Nếu đồ thị không có chu trình âm thì giá trị distance tại đường chéo luôn là 0

		j			
		1	2	3	4
i	$k = 0$	0,1	$\infty, 2$	-2,3	$\infty, 4$
	1	4,1	0,2	3,3	$\infty, 4$
	2	$\infty, 1$	0,2	0,3	2,4
	3	$\infty, 1$	-1,2	$\infty, 3$	0,4

		j			
		1	2	3	4
i	$k = 1$	0,1	$\infty, 2$	-2,3	$\infty, 4$
	1	4,1	0,2	2,1	$\infty, 4$
	2	$\infty, 1$	0,2	0,3	2,4
	3	$\infty, 1$	-1,2	$\infty, 3$	0,4

L.T.P.D.



THUẬT TOÁN FLOYD – WARSHALL

```

for each edge  $(u, v)$  do
     $\text{dist}[u][v] \leftarrow w(u, v)$ 
     $\text{next}[u][v] \leftarrow v$ 
for each vertex  $v$  do
     $\text{dist}[v][v] \leftarrow 0$ 
     $\text{next}[v][v] \leftarrow v$ 

for  $k$  from 1 to  $|V|$  do
    for  $i$  from 1 to  $|V|$ 
        for  $j$  from 1 to  $|V|$ 
            if  $\text{dist}[i][j] > \text{dist}[i][k] + \text{dist}[k][j]$  then
                 $\text{dist}[i][j] \leftarrow \text{dist}[i][k] + \text{dist}[k][j]$ 
                 $\text{next}[i][j] \leftarrow \text{next}[i][k]$ 

```

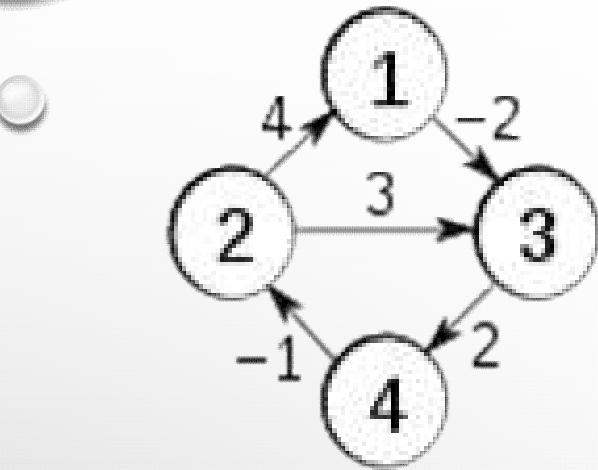
Nhận xét:

1. Tại bước lặp thứ k :
 - Giá trị tại dòng k không đổi
 - Giá trị tại cột k không đổi
2. Nếu đồ thị không có chu trình âm thì giá trị distance tại đường chéo luôn là 0
3. Giá trị tại ô (i,j) cập nhật (nếu có) bằng giá trị tại ô (i,k) cộng giá trị tại ô (k,j)

		j			
		1	2	3	4
i	$k = 0$	0,1	$\infty, 2$	-2,3	$\infty, 4$
	1	4,1	0,2	3,3	$\infty, 4$
	2	$\infty, 1$	0,2	0,3	2,4
	3	$\infty, 1$	-1,2	$\infty, 3$	0,4

		j			
		1	2	3	4
i	$k = 1$	0,1	$\infty, 2$	-2,3	$\infty, 4$
	1	4,1	0,2	2,1	$\infty, 4$
	2	$\infty, 1$	0,2	0,3	2,4
	3	$\infty, 1$	-1,2	$\infty, 3$	0,4

THUẬT TOÁN FLOYD – WARSHALL

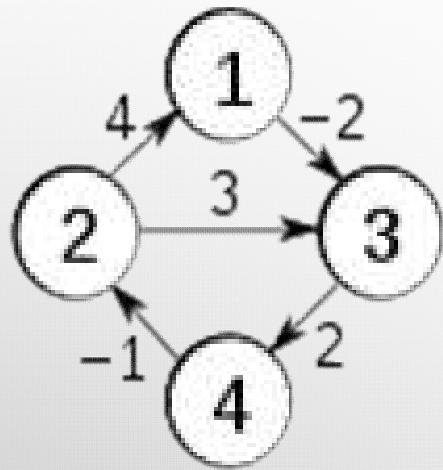


		j			
		1	2	3	4
i	$k = 2$	0,1	$\infty, 2$	-2,3	$\infty, 4$
	2	4,1	0,2	2,1	$\infty, 4$
	3	$\infty, 1$	$\infty, 2$	0,3	2,4
	4	3,2	-1,2	1,2	0,4

		j			
		1	2	3	4
i	$k = 1$	0,1	$\infty, 2$	-2,3	$\infty, 4$
	2	4,1	0,2	2,1	$\infty, 4$
	3	$\infty, 1$	$\infty, 2$	0,3	2,4
	4	$\infty, 1$	-1,2	$\infty, 3$	0,4

		j			
		1	2	3	4
i	$k = 3$	0,1	$\infty, 2$	-2,3	0,3
	2	4,1	0,2	2,1	4,1
	3	$\infty, 1$	$\infty, 2$	0,3	2,4
	4	3,2	-1,2	1,2	0,4

THUẬT TOÁN FLOYD – WARSHALL



		j			
		1	2	3	4
i	1	0,1	$\infty, 2$	-2,3	0,3
	2	4,1	0,2	2,1	4,1
	3	$\infty, 1$	$\infty, 2$	0,3	2,4
	4	3,2	-1,2	1,2	0,4

		j			
		1	2	3	4
i	1	0,1	-1,3	-2,3	0,3
	2	4,1	0,2	2,1	4,1
	3	5,4	1,4	0,3	2,4
	4	3,2	-1,2	1,2	0,4

THUẬT TOÁN FLOYD – WARSHALL

Cách truy vết đường đi:

```

procedure Path(u, v)
    if next[u][v] = null then
        return []
    path = [u]
    while u ≠ v
        u ← next[u][v]
        path.append(u)
    return path

```

		j			
		1	2	3	4
i	1	0,1	-1,3	-2,3	0,3
	2	4,1	0,2	2,1	4,1
	3	5,4	1,4	0,3	2,4
	4	3,2	-1,2	1,2	0,4

Path(4,1): $u=4$

$\text{next}(4,1)=2 \rightarrow u=2$

$\text{next}(2,1)=1 \rightarrow u=1=v \rightarrow \text{stop}$

path=4→2→1

Path(1,4): $u=1$

$\text{next}(1,4)=3 \rightarrow u=3$

$\text{next}(3,4)=4 \rightarrow u=4=v \rightarrow \text{stop}$

path=1→3→4

Path(3,1): $u=3$

$\text{next}(3,1)=4 \rightarrow u=4$

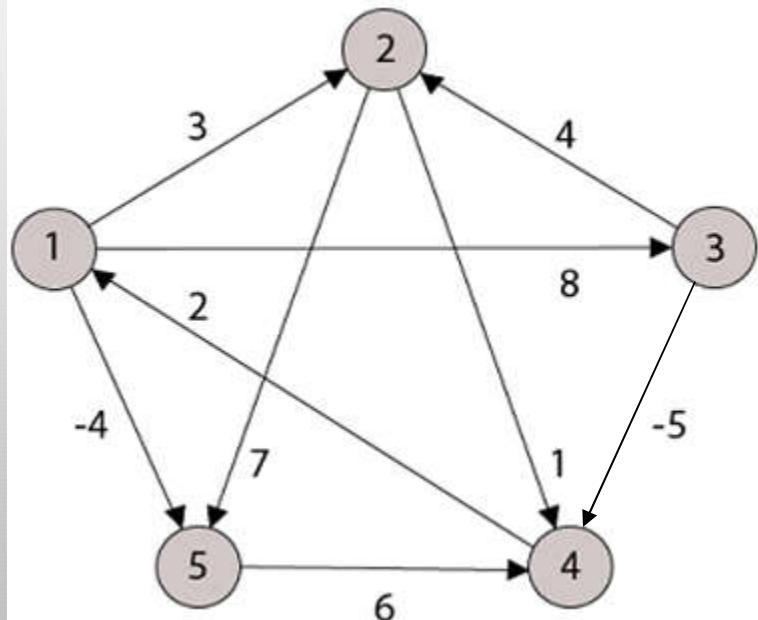
$\text{next}(4,1)=2 \rightarrow u=2$

$\text{next}(2,1)=1 \rightarrow u=1=v \rightarrow \text{stop}$

path=3→4→2→1

THUẬT TOÁN FLOYD – WARSHALL

Bài tập 1: Tìm đường đi ngắn nhất giữa tất cả các cặp đỉnh



		j				
		1	2	3	4	5
i	1	0,1	3,2	8,3	$\infty,4$	-4,5
	2	$\infty,1$	0,2	$\infty,3$	1,4	7,5
	3	$\infty,1$	4,2	0,3	-5,4	$\infty,5$
	4	2,1	$\infty,2$	$\infty,3$	0,4	$\infty,5$
	5	$\infty,1$	$\infty,2$	$\infty,3$	6,4	0,5

		j				
		1	2	3	4	5
i	1	0,1	3,2	8,3	$\infty,4$	-4,5
	2	$\infty,1$	0,2	$\infty,3$	1,4	7,5
	3	$\infty,1$	4,2	0,3	-5,4	$\infty,5$
	4	2,1	5,1	10,1	0,4	-2,1
	5	$\infty,1$	$\infty,2$	$\infty,3$	6,4	0,5

THUẬT TOÁN FLOYD – WARSHALL

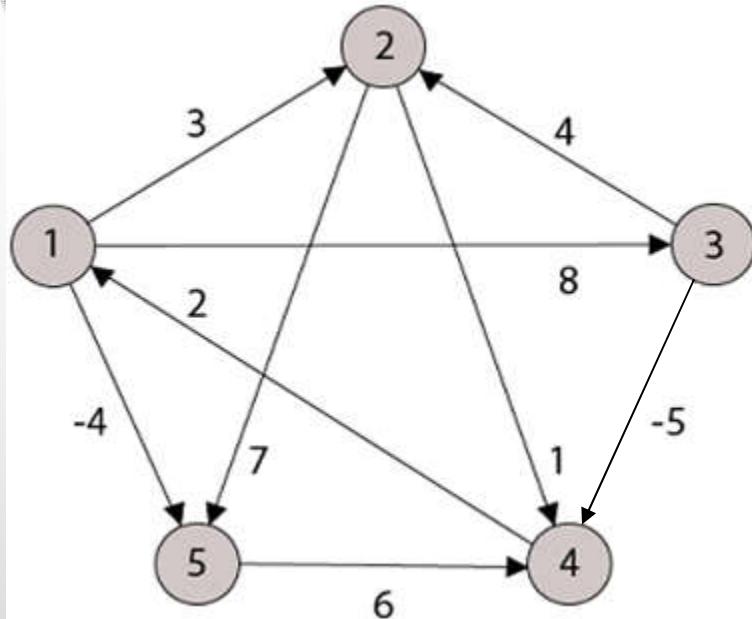
$k = 2$		j				
		1	2	3	4	5
i	1	0,1	3,2	8,3	4,2	-4,5
	2	∞ ,1	0,2	∞ ,3	1,4	7,5
	3	∞ ,1	4,2	0,3	-5,4	11,2
	4	2,1	5,1	10,1	0,4	-2,1
	5	∞ ,1	∞ ,2	∞ ,3	6,4	0,5

$k = 4$		j				
		1	2	3	4	5
i	1	0,1	3,2	8,3	3,3	-4,5
	2	3,4	0,2	11,4	1,4	-1,4
	3	-3,4	0,4	0,3	-5,4	-7,4
	4	2,1	5,1	10,1	0,4	-2,1
	5	8,4	11,4	16,4	6,4	0,5

$k = 3$		j				
		1	2	3	4	5
i	1	0,1	3,2	8,3	3,3	-4,5
	2	∞ ,1	0,2	∞ ,3	1,4	7,5
	3	∞ ,1	4,2	0,3	-5,4	11,2
	4	2,1	5,1	10,1	0,4	-2,1
	5	∞ ,1	∞ ,2	∞ ,3	6,4	0,5

$k = 5$		j				
		1	2	3	4	5
i	1	0,1	3,2	8,3	2,5	-4,5
	2	3,4	0,2	11,4	1,4	-1,4
	3	-3,4	0,4	0,3	-5,4	-7,4
	4	2,1	5,1	10,1	0,4	-2,1
	5	8,4	11,4	16,4	6,4	0,5

THUẬT TOÁN FLOYD – WARSHALL



		j				
		1	2	3	4	5
i	$k = 5$	0,1	3,2	8,3	2,5	-4,5
	2	3,4	0,2	11,4	1,4	-1,4
	3	-3,4	0,4	0,3	-5,4	-7,4
	4	2,1	5,1	10,1	0,4	-2,1
	5	8,4	11,4	16,4	6,4	0,5

Path(1,4): $u=1$

$\text{next}(1,4)=5 \rightarrow u=5$

$\text{next}(5,4)=4 \rightarrow u=4=v \rightarrow \text{stop}$
 $\text{path}=1 \rightarrow 5 \rightarrow 4$

Path(3,1): $u=3$

$\text{next}(3,1)=4 \rightarrow u=4$

$\text{next}(4,1)=1=v \rightarrow \text{stop}$
 $\text{path}=3 \rightarrow 4 \rightarrow 1$

Truy vết đường đi:

TÓM TẮT CHƯƠNG 3 ĐƯỜNG ĐI NGẮN NHẤT

Thuật toán Moore-Dijkstra

Tìm đường đi ngắn nhất từ đỉnh xuất phát đến các đỉnh khác trên **đồ thị có trọng số dương**

Thuật toán bellman – fords

Thuật toán Bellman – Ford tìm đường đi ngắn nhất từ đỉnh xuất phát đến tất cả các đỉnh còn lại của đồ thị có trọng số tùy ý nhưng không có chu trình âm

Thuật toán floyd – warshall

Thuật toán Floyd – Warshall tìm đường đi ngắn nhất **giữa tất cả các cặp đỉnh** của đồ thị có trọng số tùy ý nhưng không có chu trình âm