LÝ THUYẾT ĐỒ 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 CỰC ĐẠI TRONG MẠNG

L.T.D.D.

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*

……………………………………………………………..

L.T.P.D.

CHƯƠNG 3

ĐƯỜNG ĐI NGẮN NHẤT TRÊN ĐỒ THỊ

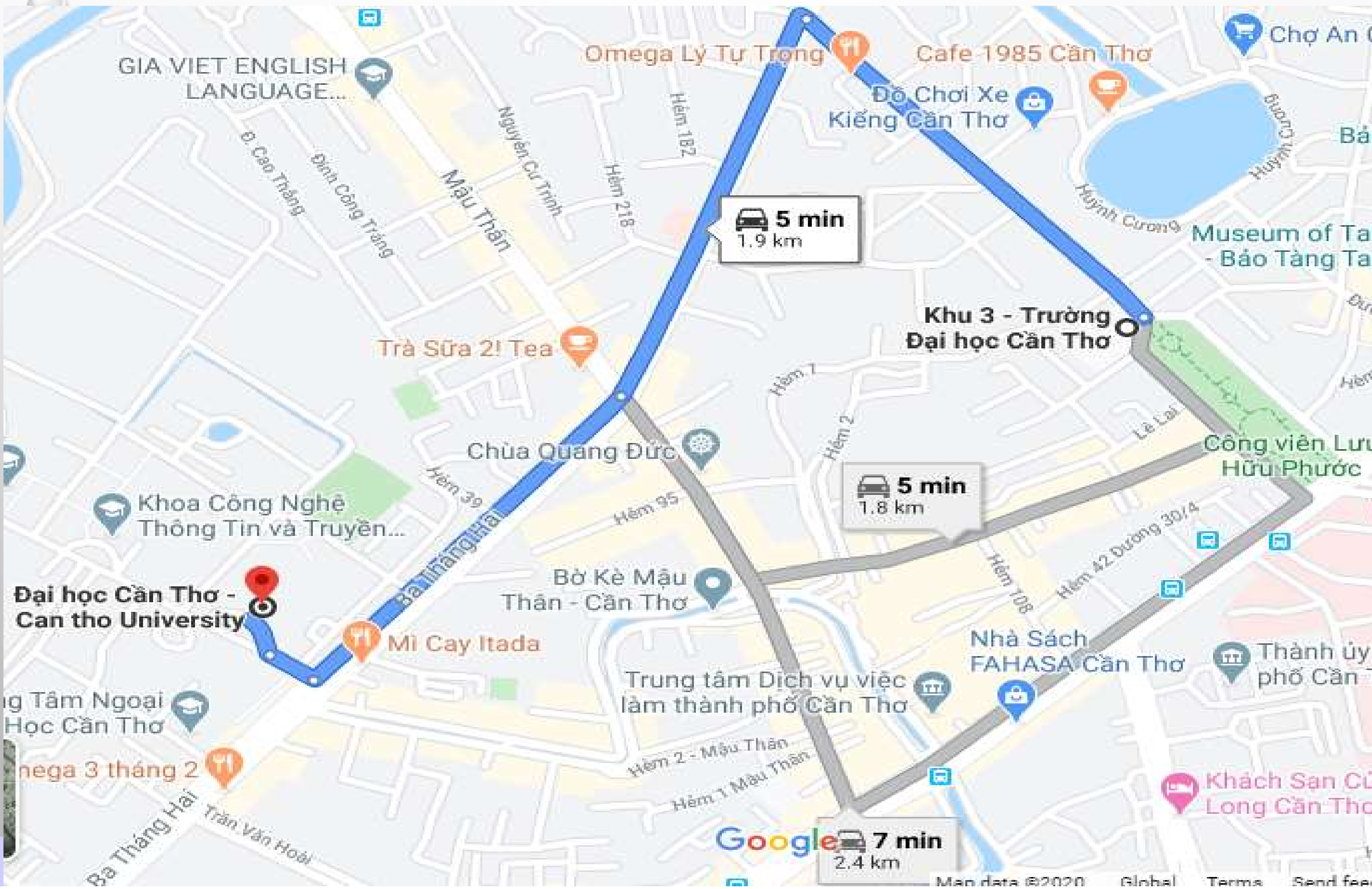
NỘI DUNG:

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

L.T.D.D.

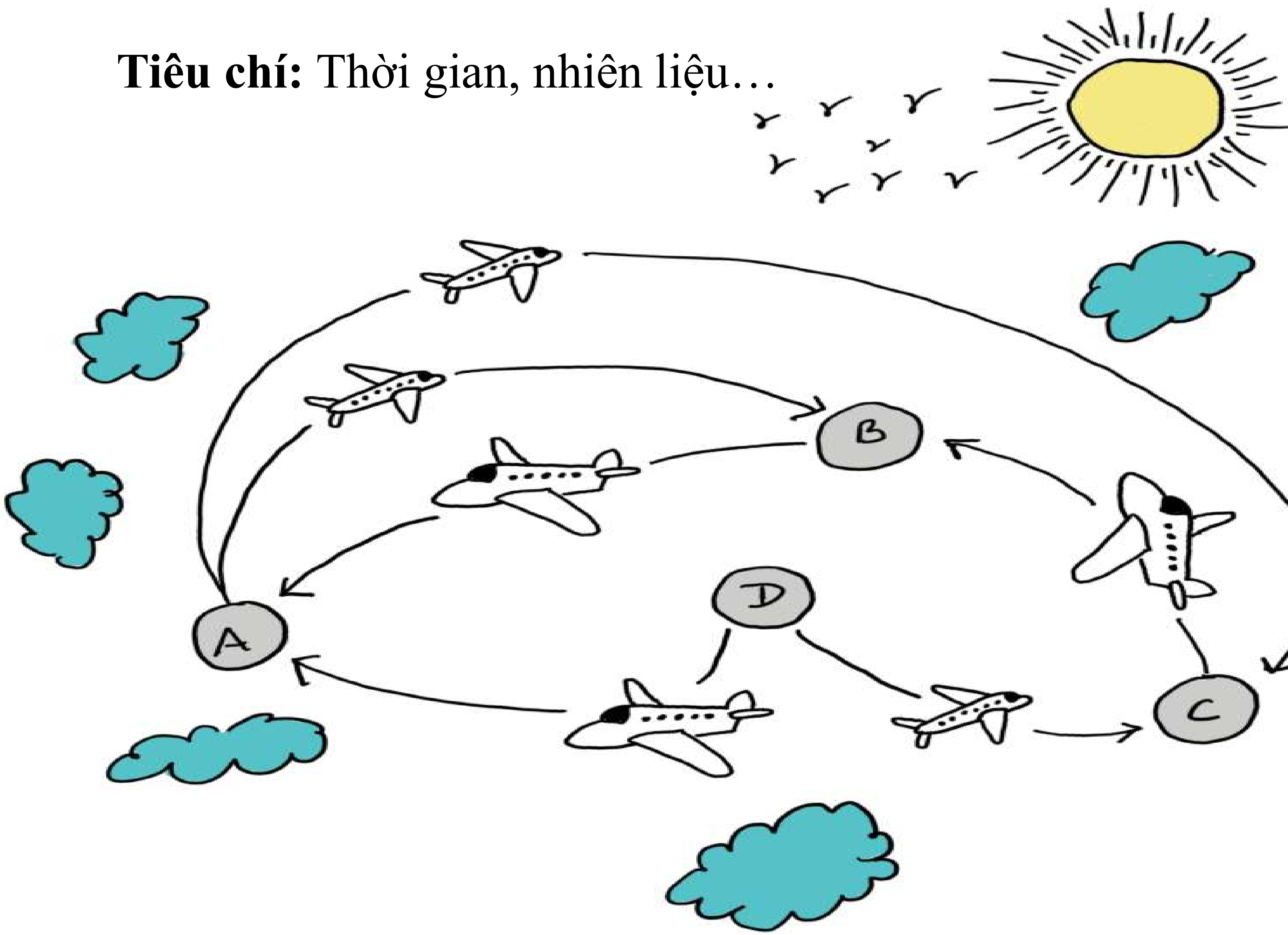
MỘT SỐ ỨNG DỤNG

Tiêu chí: Thời gian, nhiên liệu…



MỘT SỐ ỨNG DỤNG

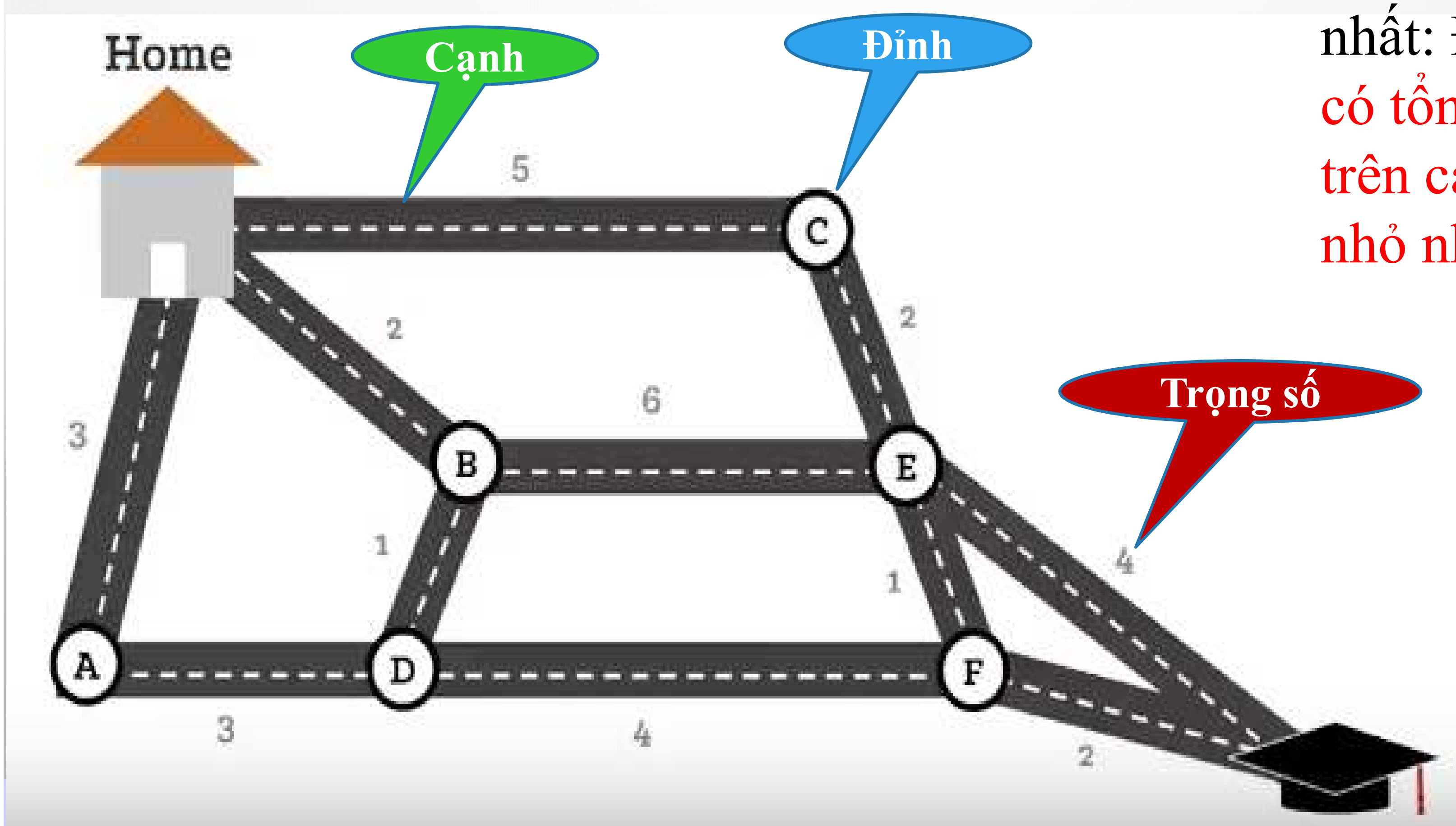
Tiêu chí: Thời gian, nhiên liệu…



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

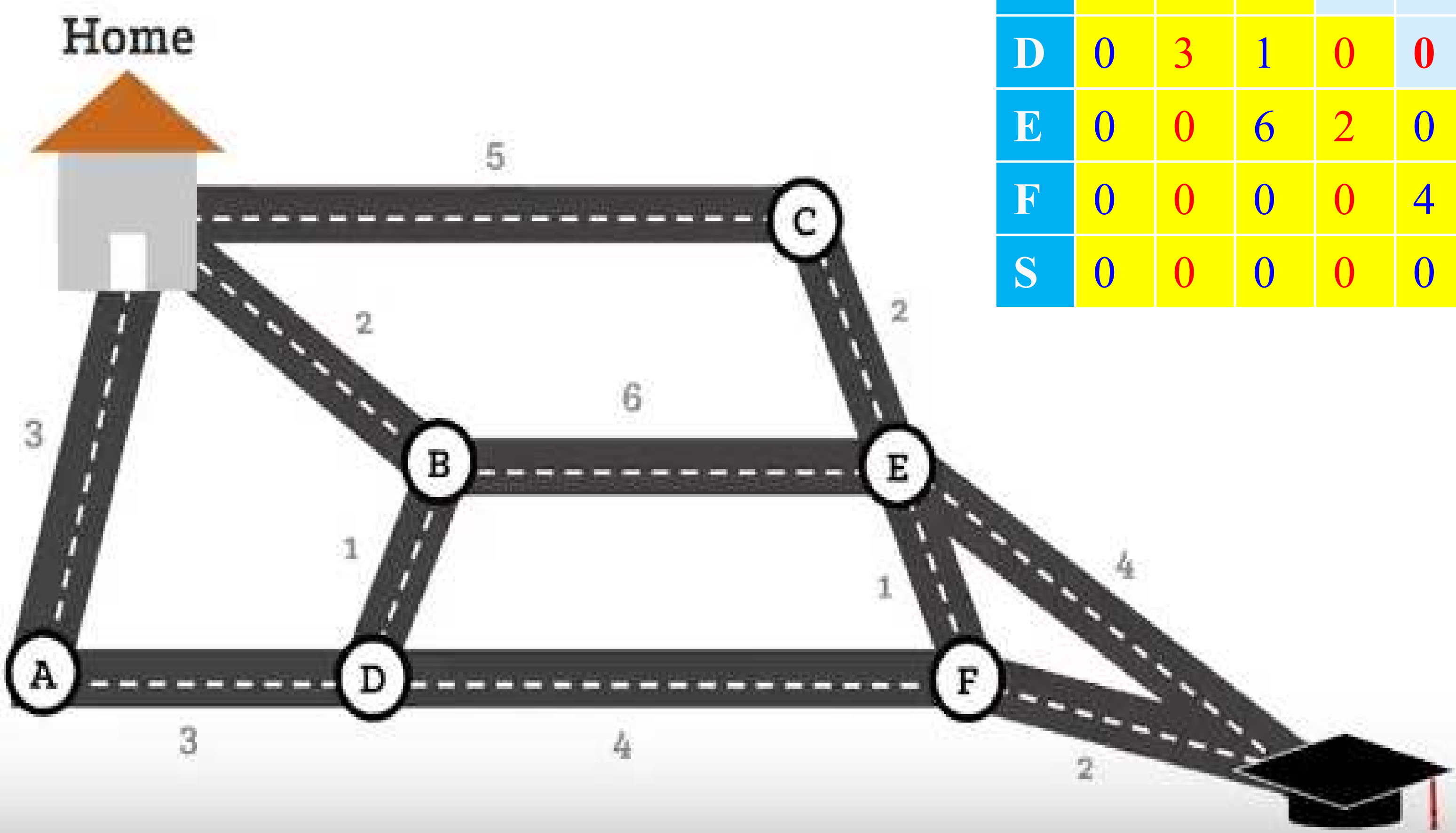
L.T.D.D.

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

L.T.D.D.

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	•SIGCSE Outstanding Contribution (1989) •Turing Award (1972) •ACM Fellow (1994) •Dijkstra Prize (2002)

Scientific career	
Fields	•Computer science •Theoretical computer science
Institutions	•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

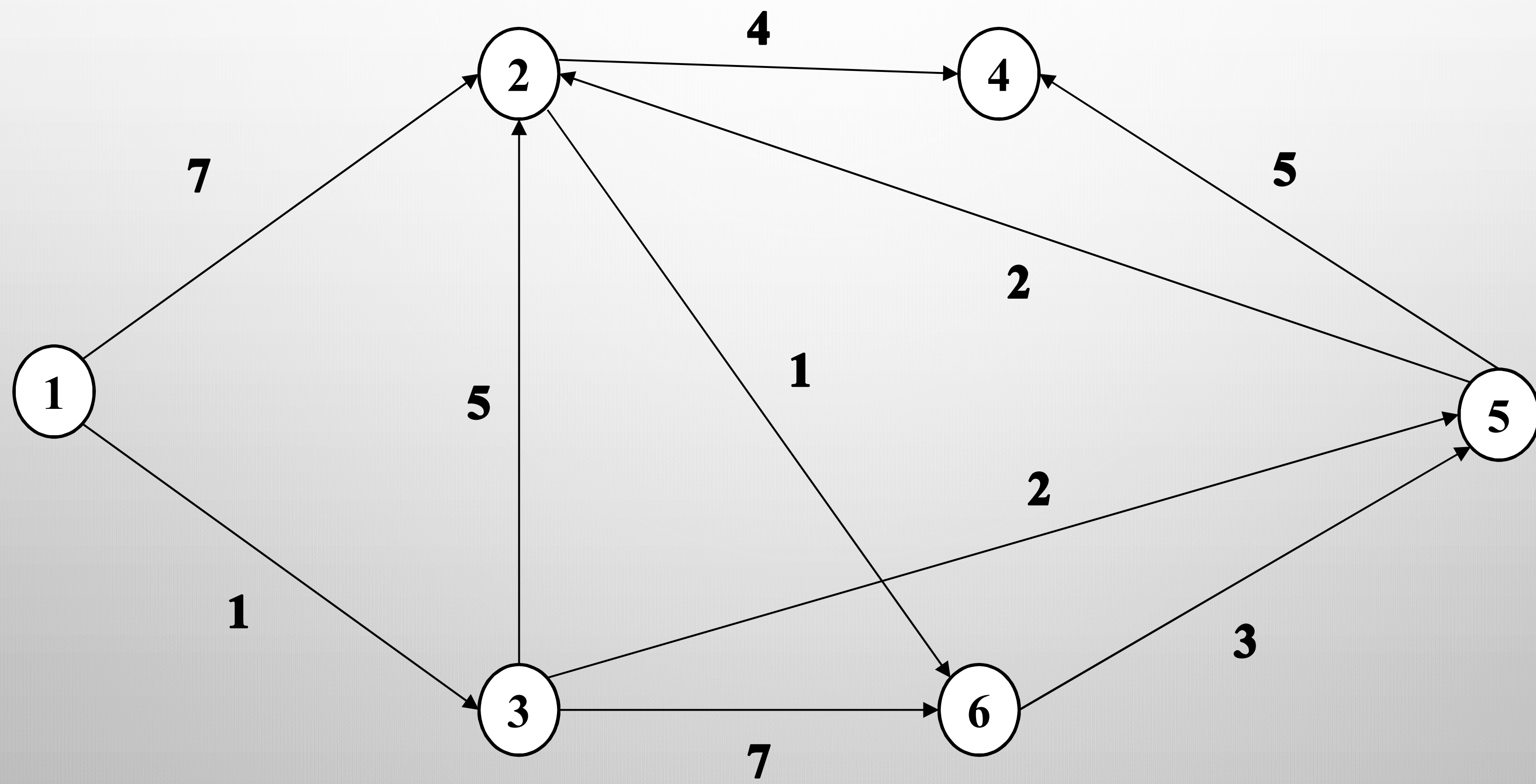
- Tìm đường đi ngắn nhất từ một đỉnh đến các đỉnh khác trên **đồ thị có trọng số dương**

```
function Dijkstra(Graph, source):
    create vertex set Q
    for each vertex v in Graph:
        dist[v] ← INFINITY              //Distance from source to v
        prev[v] ← UNDEFINED             //Previous of v
        add v to Q
    dist[source] ← 0
    while Q is not empty:
        u ← vertex in Q with min dist[u]
        remove u from Q
        for each neighbor v of u:       // only v that are still in Q
            alt ← dist[u] + length(u, v)
            if alt < dist[v]:
                dist[v] ← alt
                prev[v] ← u
    return dist[], prev[]
```

L.T.D.D.

GIẢI THUẬT MOORE – DIJKSTRA

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

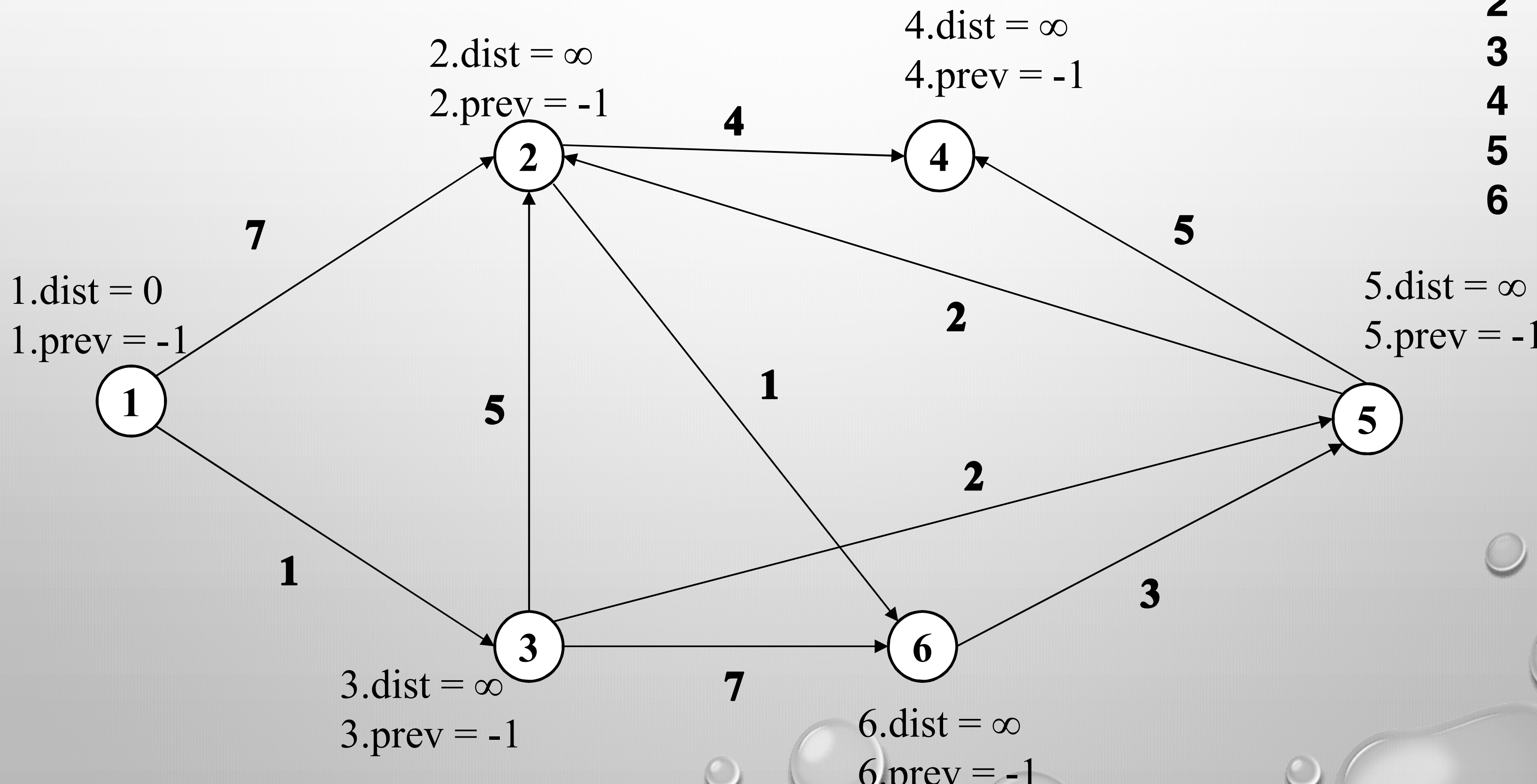


L.T.D.D.

GIẢI THUẬT MOORE – DIJKSTRA

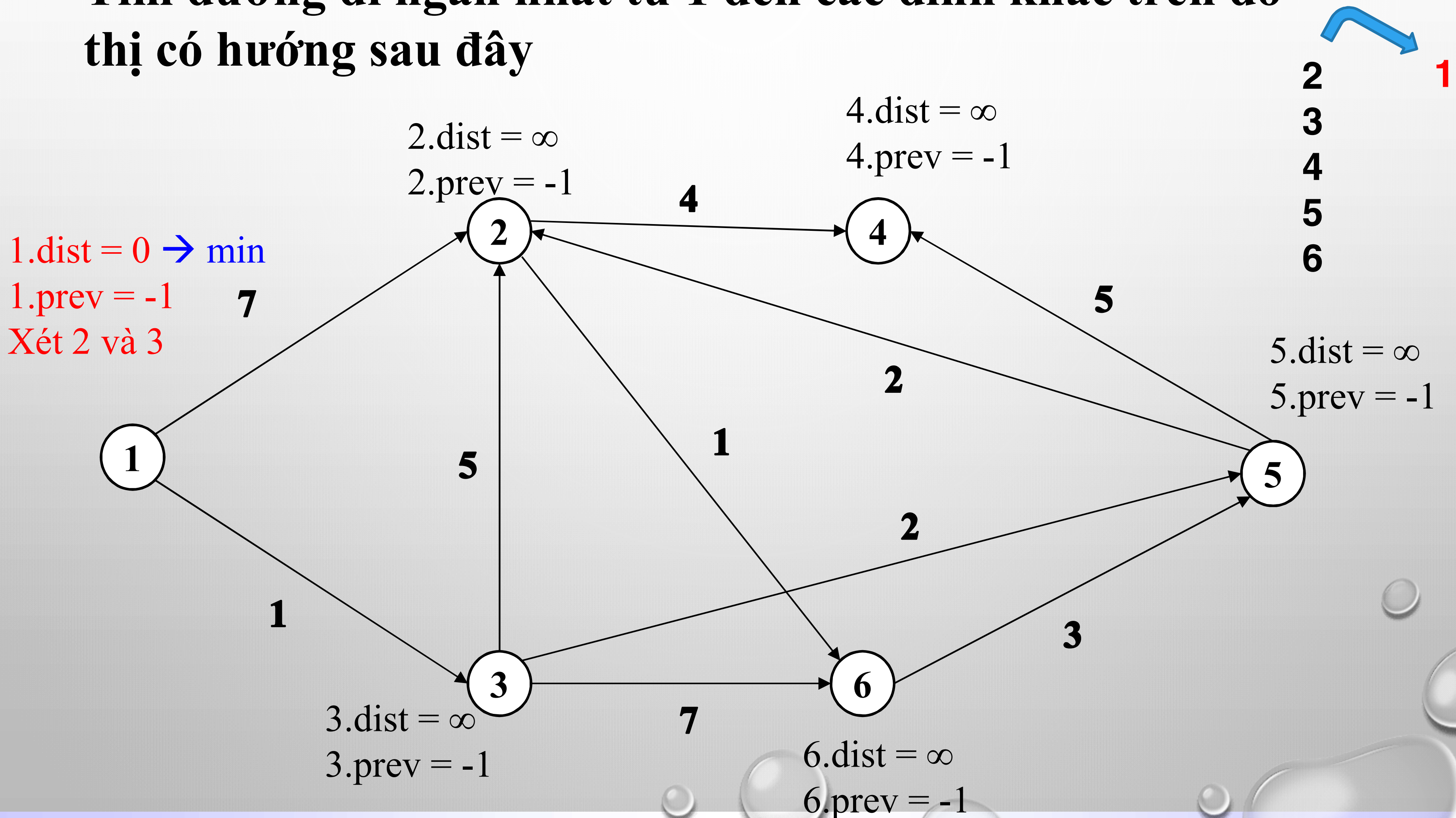
Tìm đường đi ngắn nhất từ 1 đến các đỉnh khác trên đồ thị có hướng sau đây

1
2
3
4
5
6



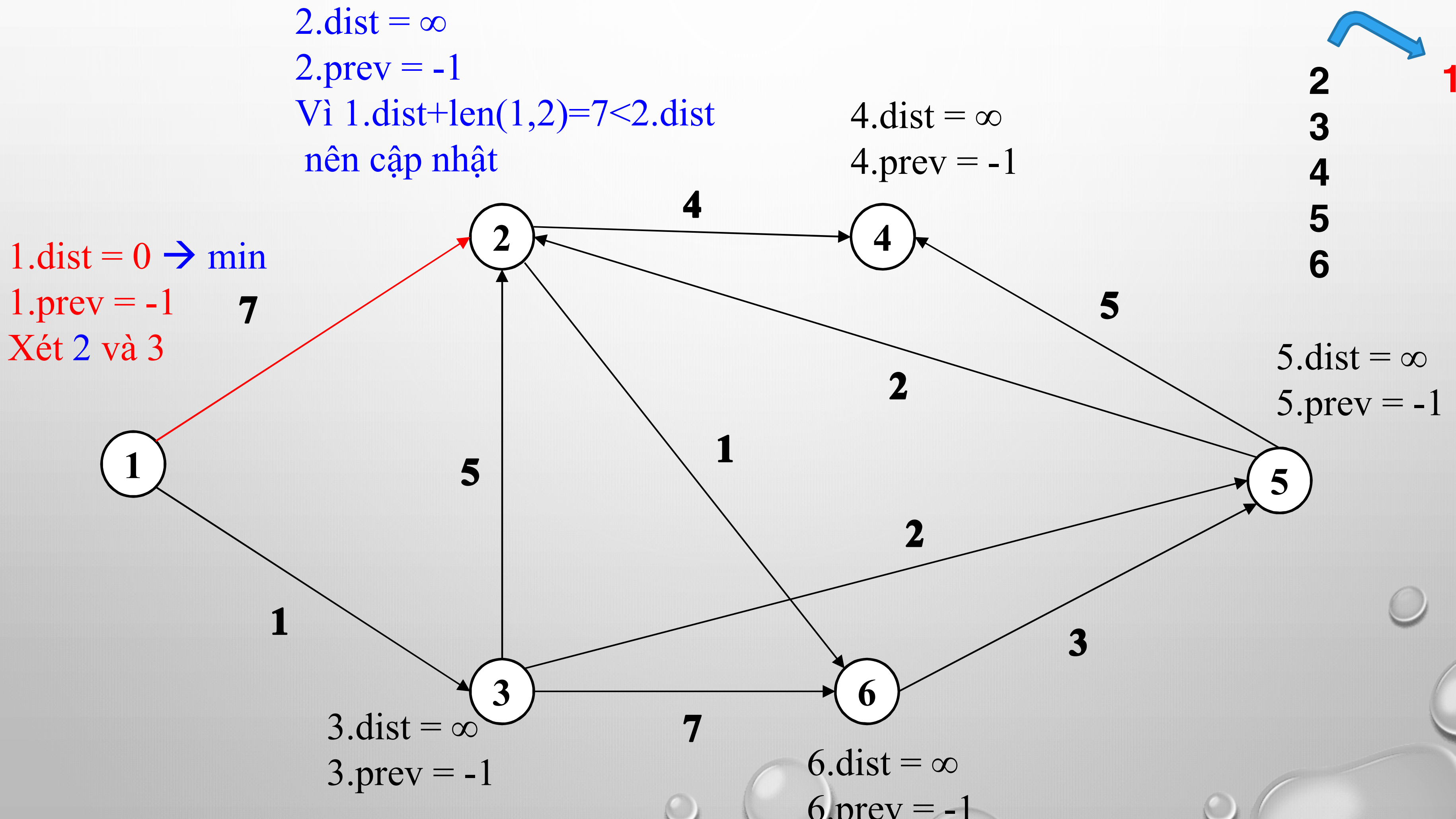
GIẢI THUẬT MOORE – DIJKSTRA

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



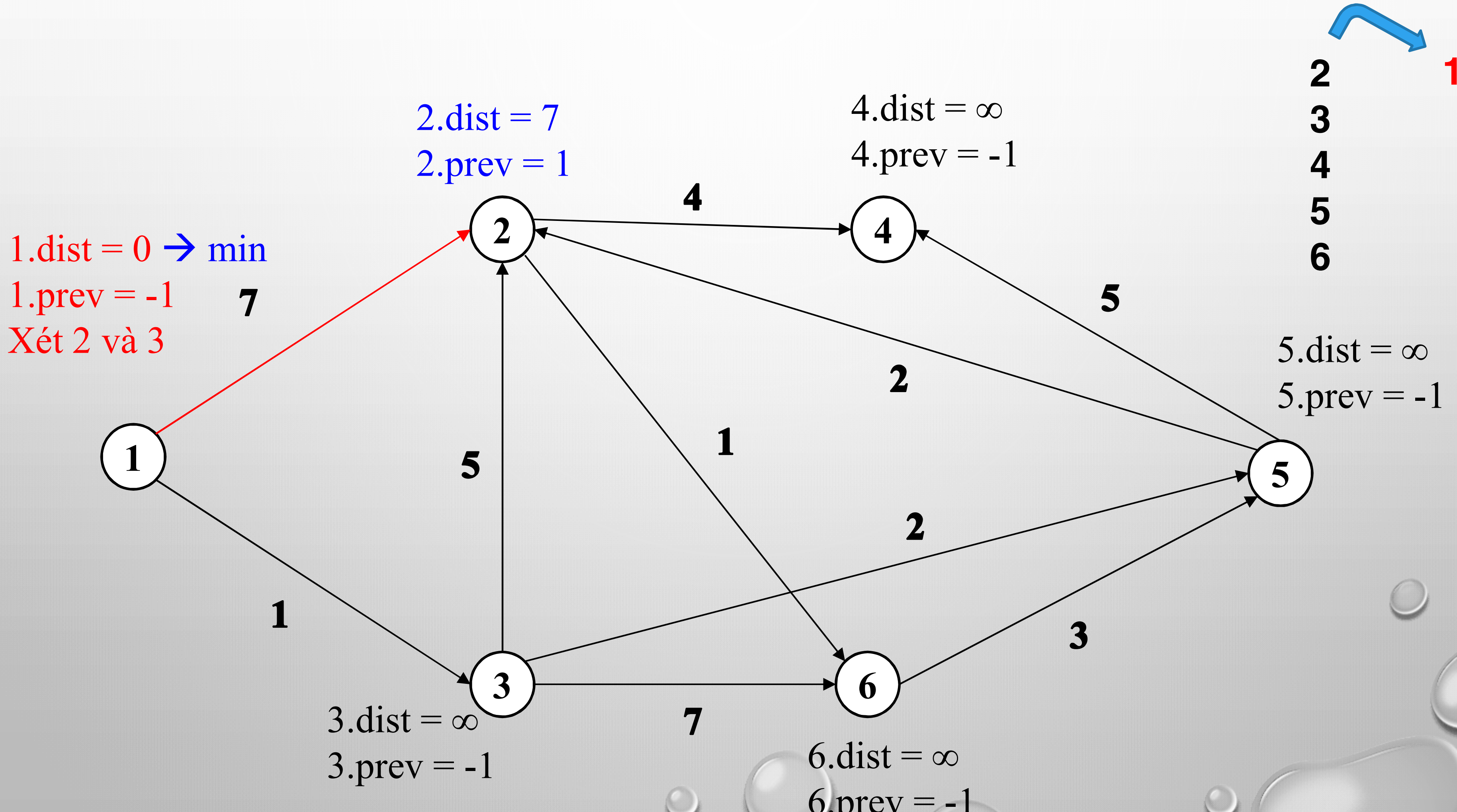
GIẢI THUẬT MOORE – DIJKSTRA

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



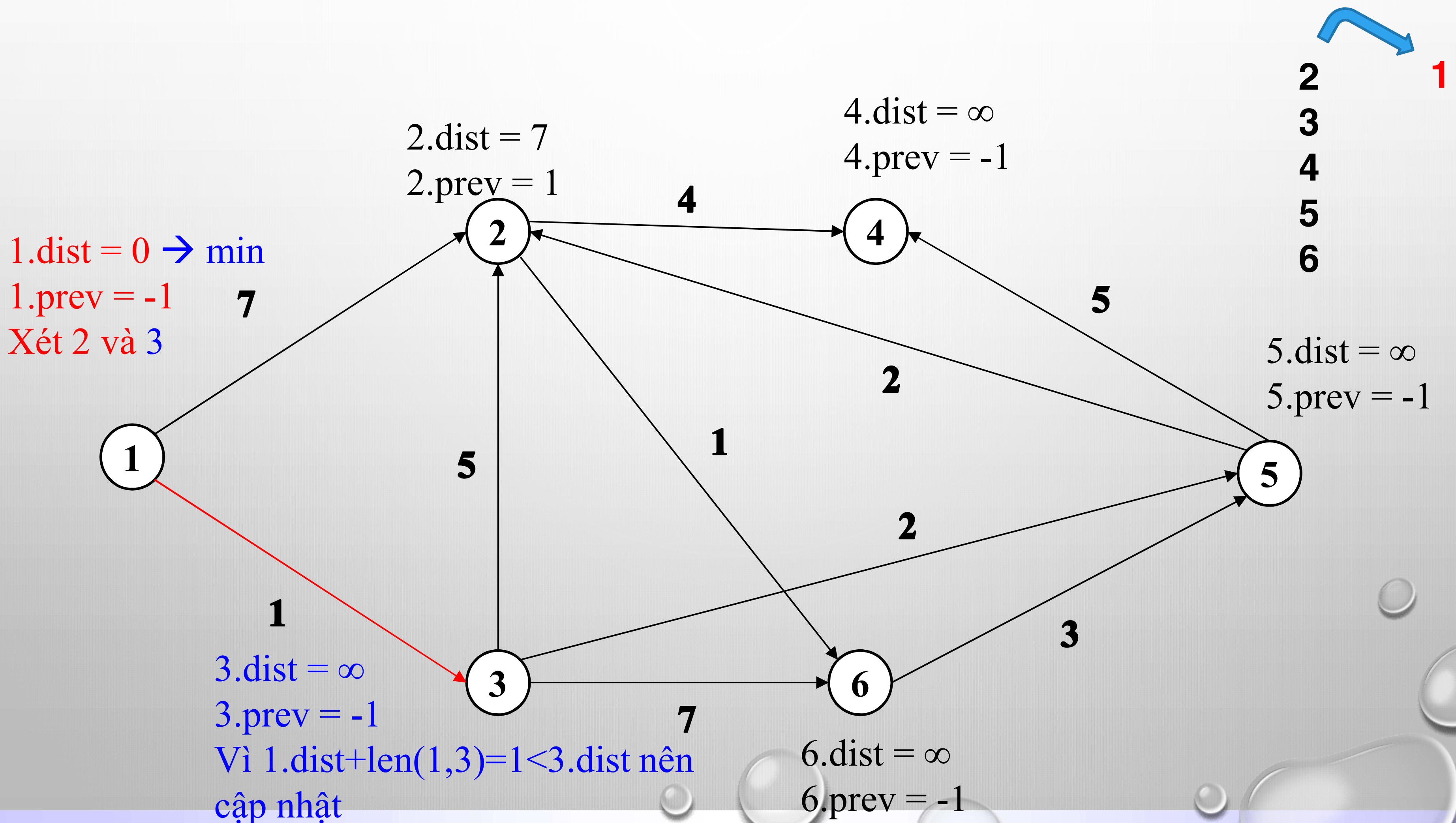
GIẢI THUẬT MOORE – DIJKSTRA

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



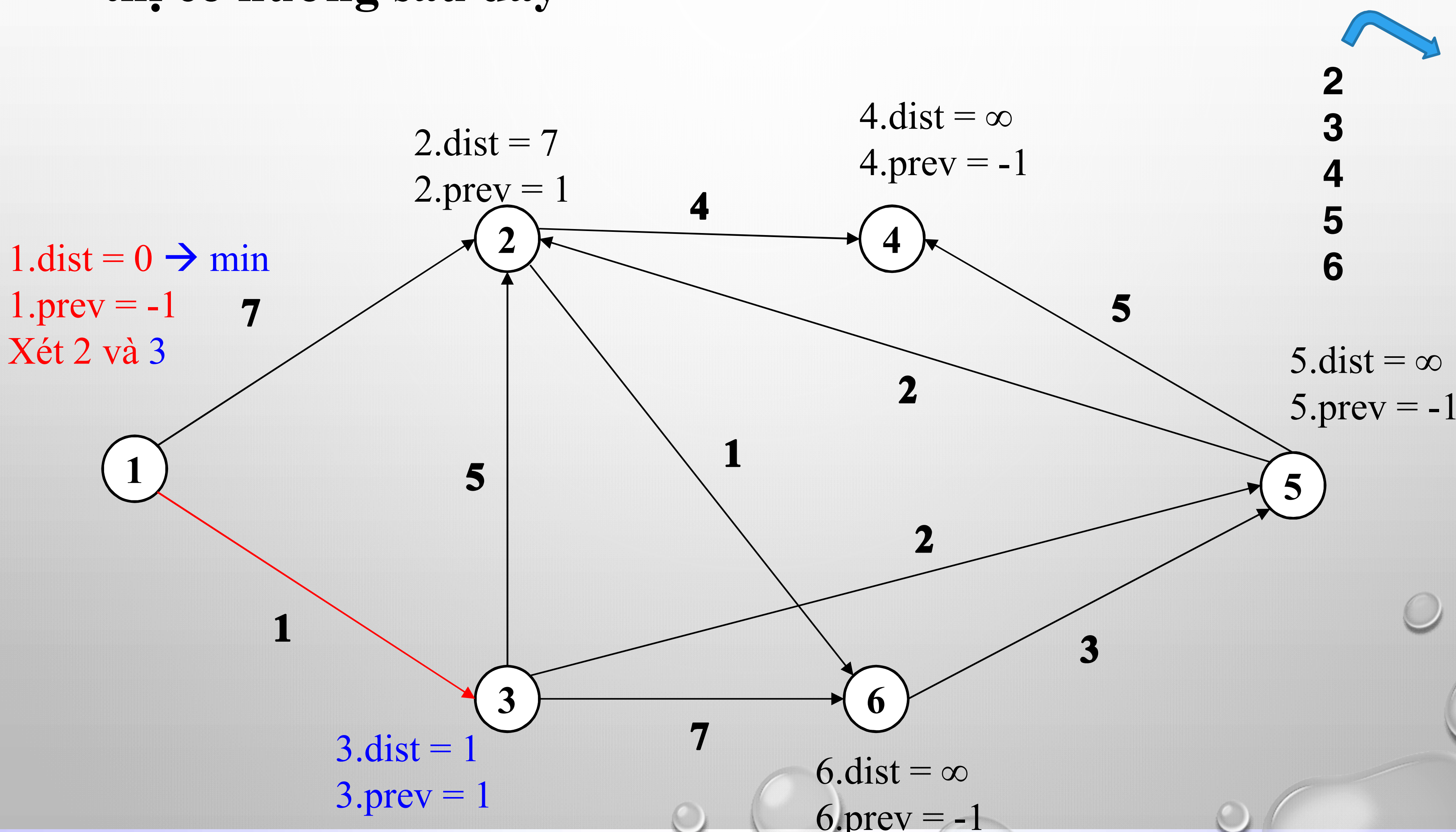
GIẢI THUẬT MOORE – DIJKSTRA

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



GIẢI THUẬT MOORE – DIJKSTRA

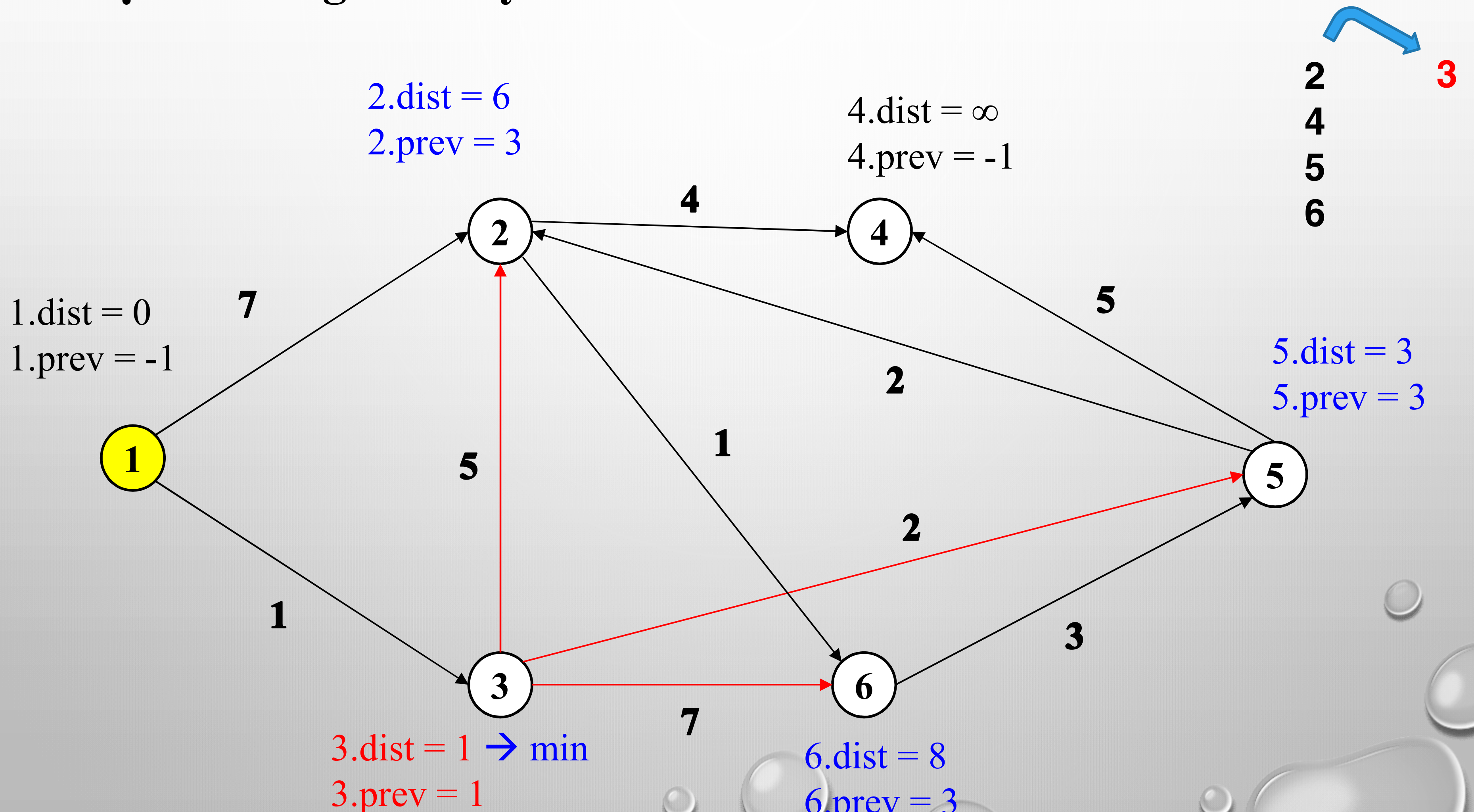
Tìm đường đi ngắn nhất từ 1 đến các đỉnh khác trên đồ thị có hướng sau đây



L.T.D.D.

GIẢI THUẬT MOORE – DIJKSTRA

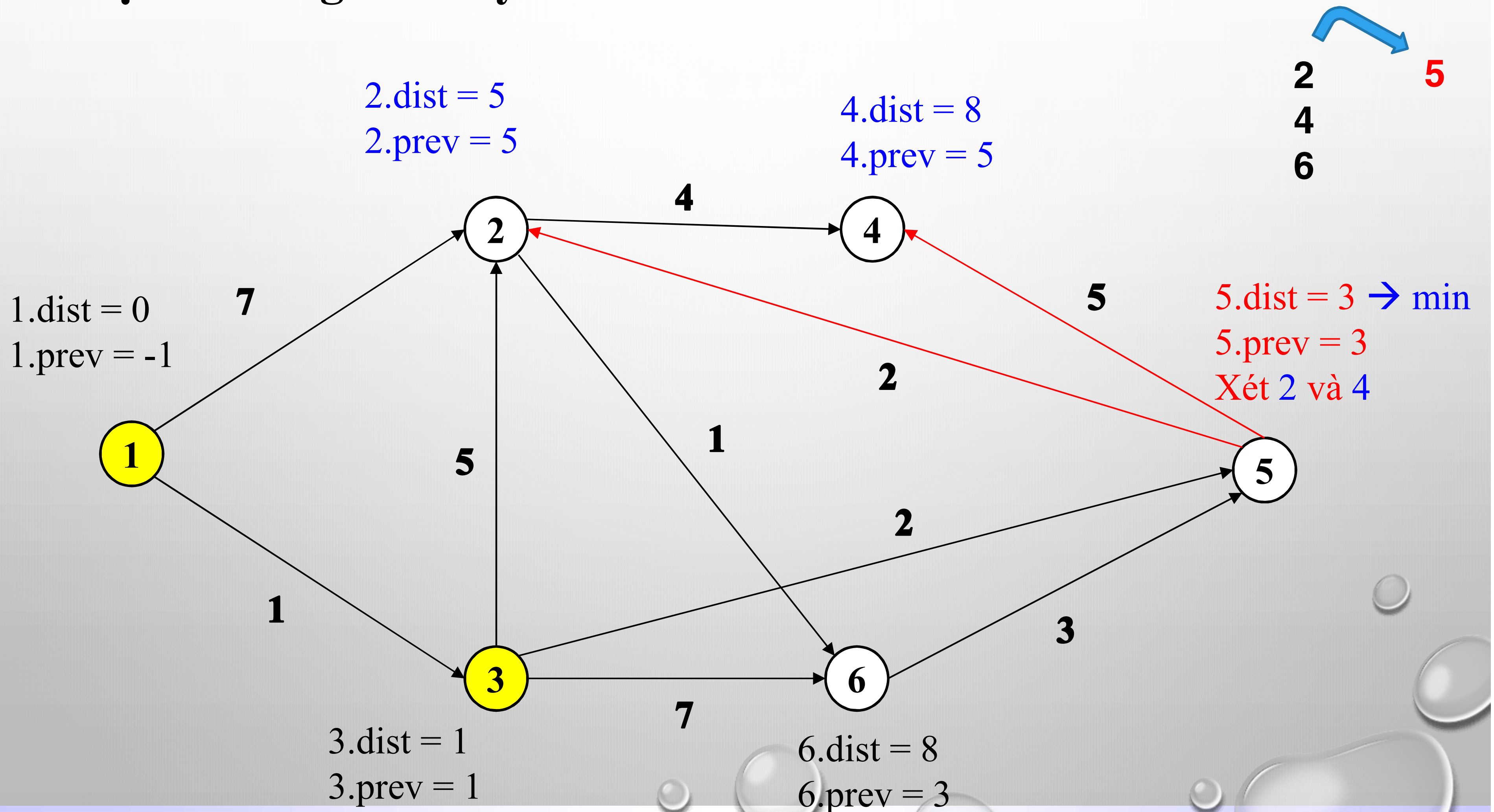
Tìm đường đi ngắn nhất từ 1 đến các đỉnh khác trên đồ thị có hướng sau đây



L.T.D.D.

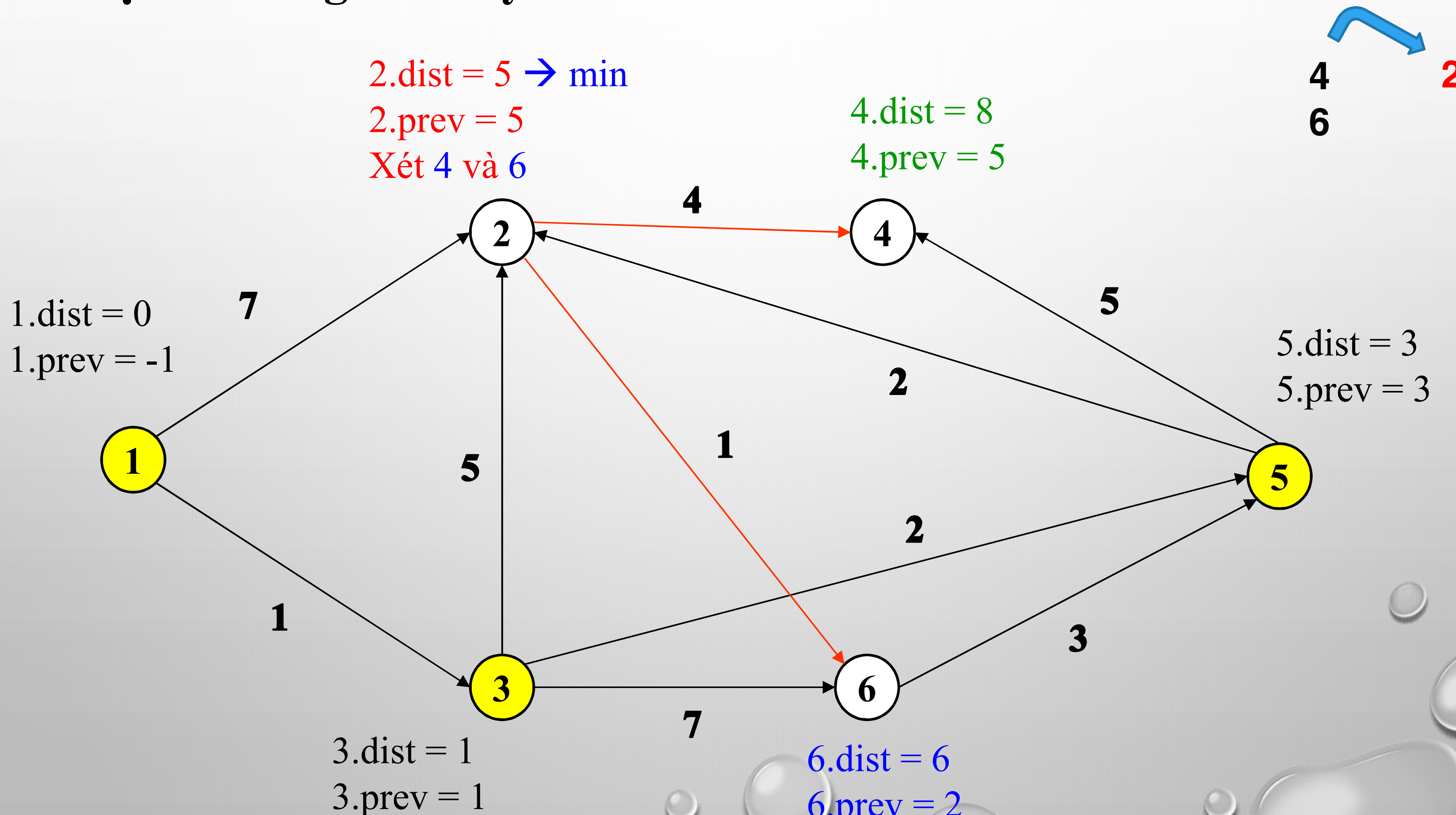
GIẢI THUẬT MOORE – DIJKSTRA

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



GIẢI THUẬT MOORE – DIJKSTRA

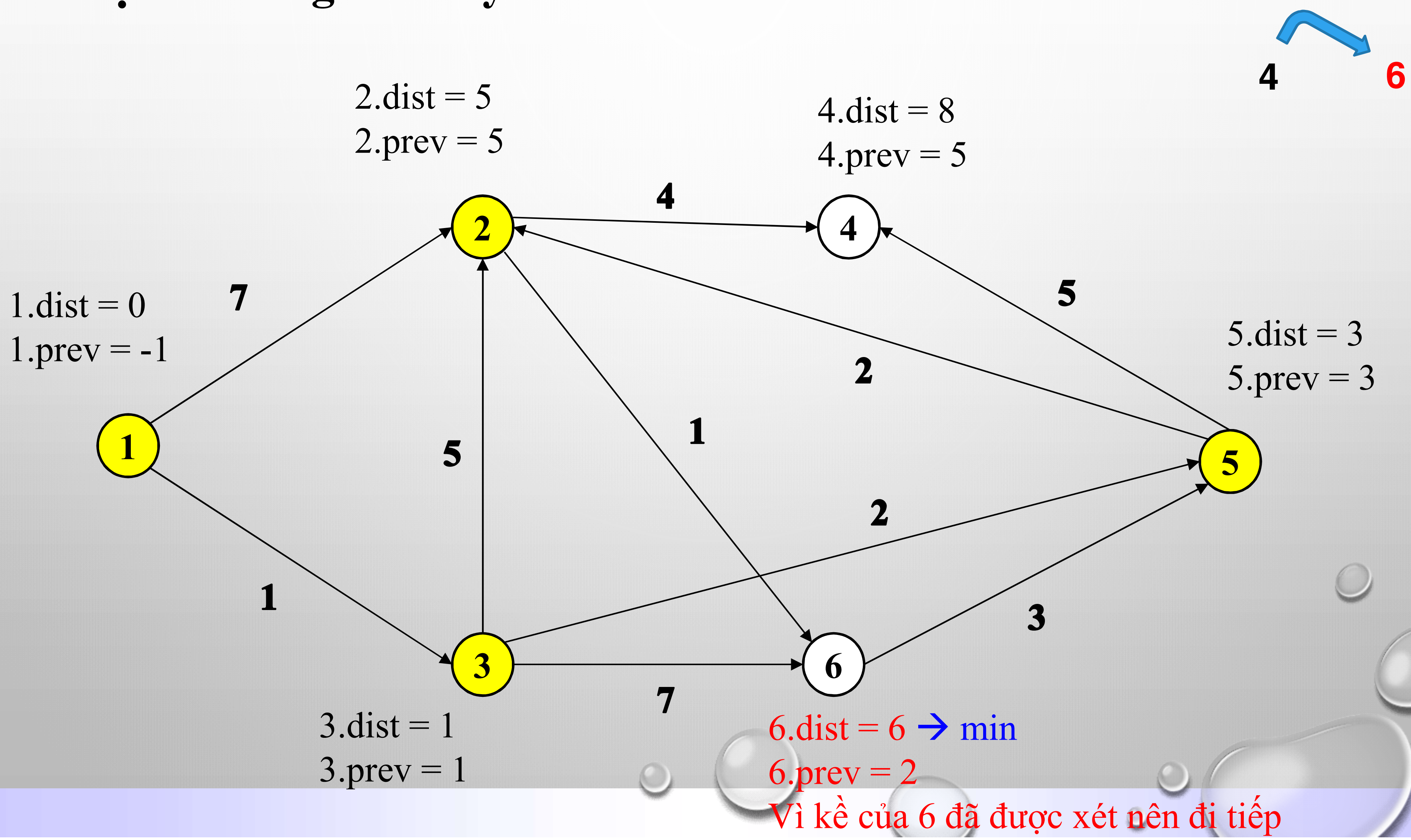
Tìm đường đi ngắn nhất từ 1 đến các đỉnh khác trên đồ thị có hướng sau đây



L.T.D.D.

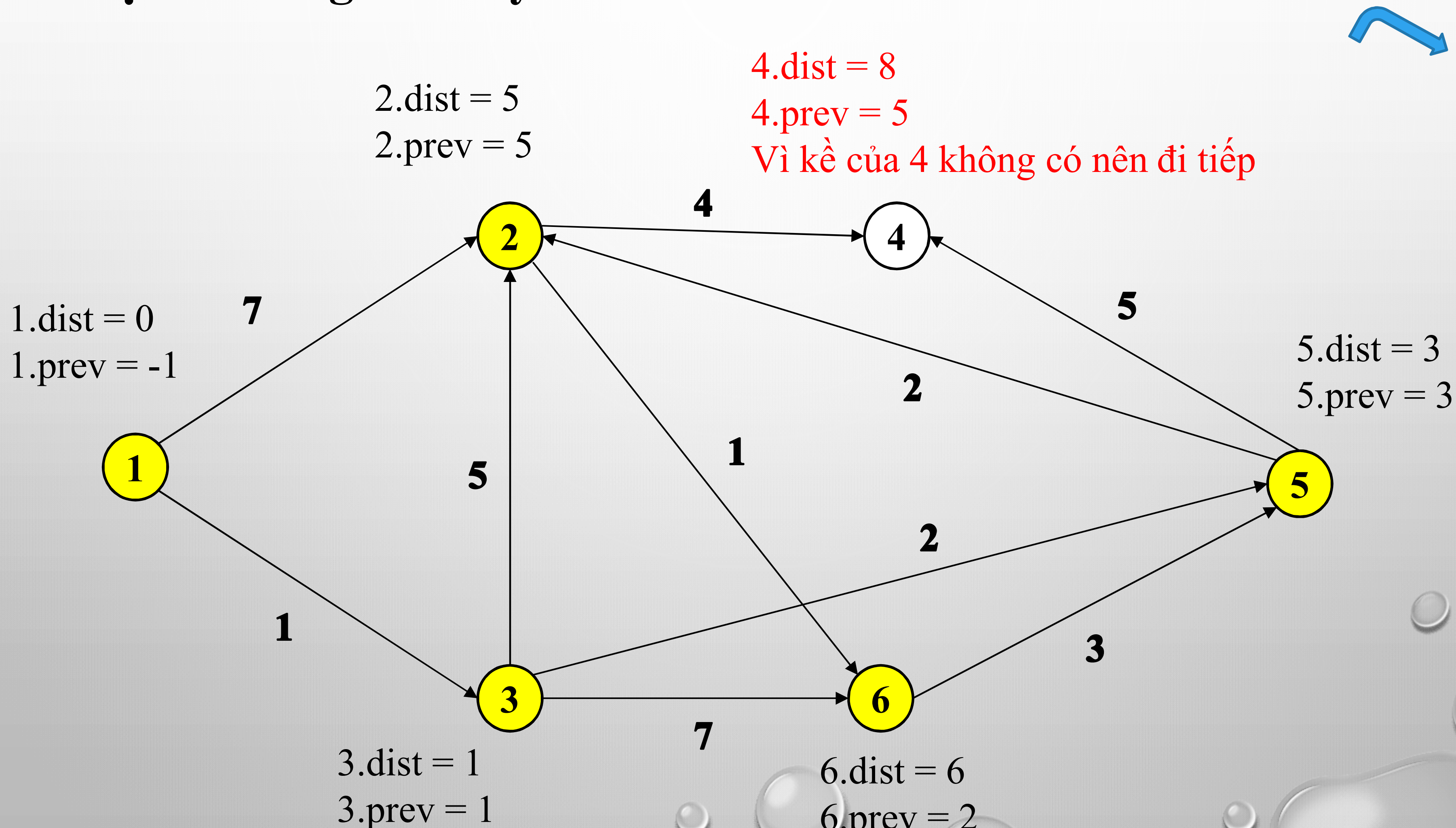
GIẢI THUẬT MOORE – DIJKSTRA

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



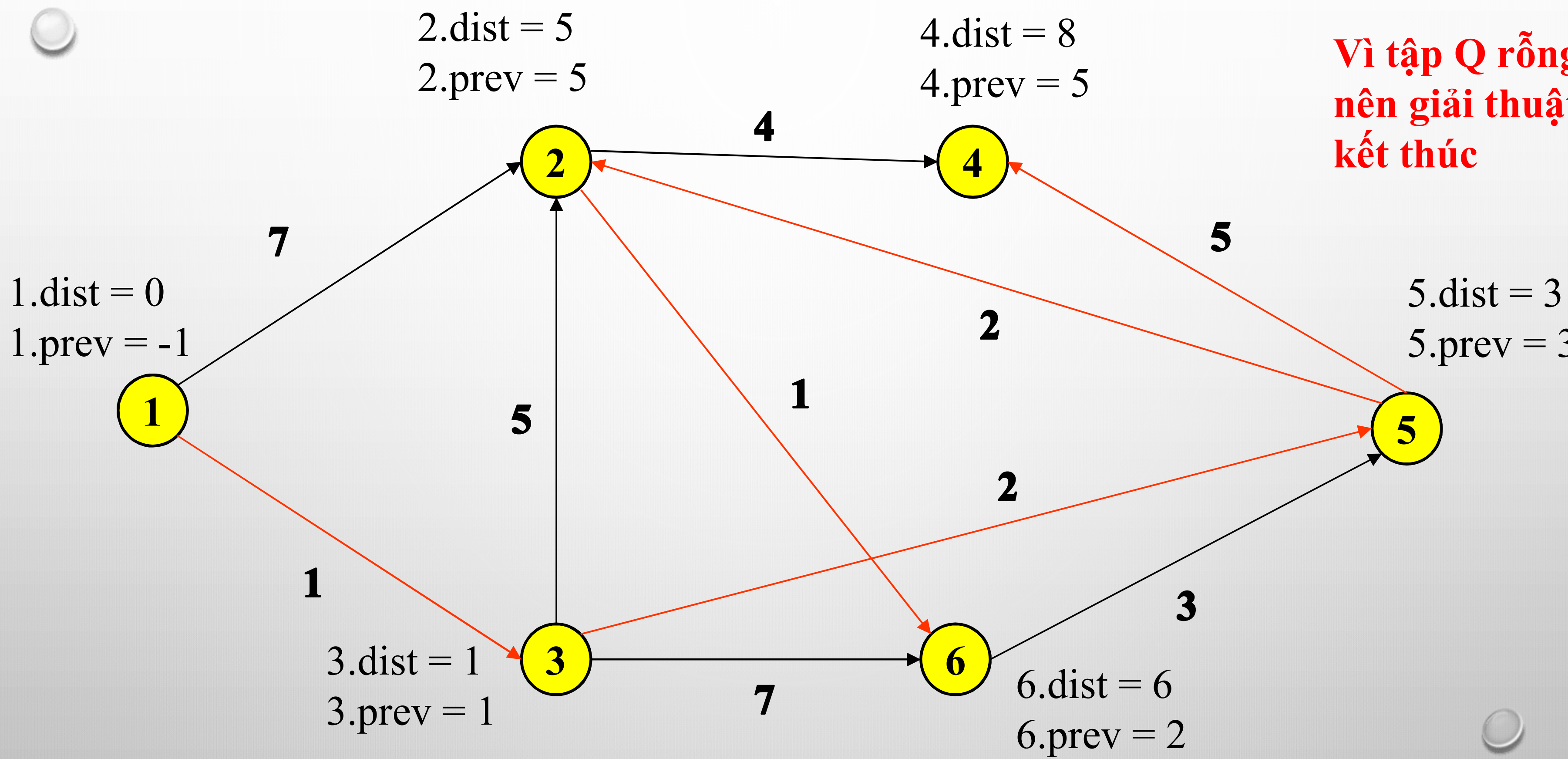
GIẢI THUẬT MOORE – DIJKSTRA

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

L.T.D.D.

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

Yêu cầu: Chạy tay thuật toán Dijkstra nộp cho cô

Hướng dẫn:
- Vẽ đồ thị
- Chạy thuật toán

	A	B	C	D	E	F	G	H	K
A		1	1						4
B							9	2	3
C				7					2
D					1	5		4	3
E						2			
F							9	3	
G								5	
H									7
K									

T 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

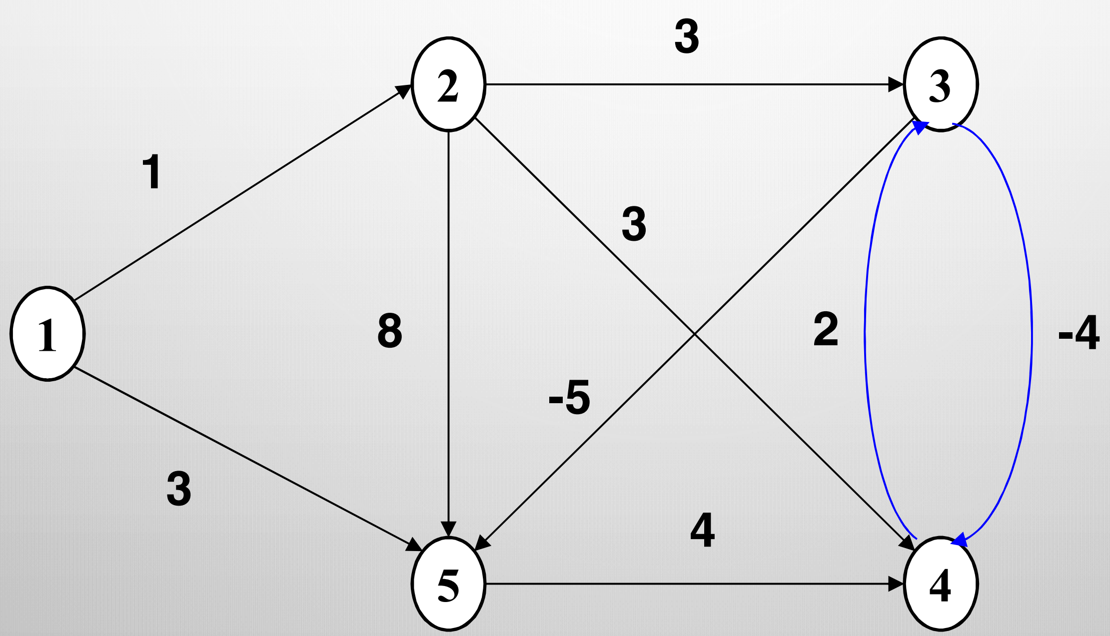
Known for	Dynamic programming Stochastic dynamic programming Curse of dimensionality Linear search problem Bellman equation Bellman–Ford algorithm Bellman's lost in a forest problem Bellman–Held–Karp algorithm Grönwall–Bellman inequality Hamilton–Jacobi–Bellman equation
Awards	John von Neumann Theory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

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



THUẬT TOÁN BELLMAN – FORD (– MOORE)

```
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)

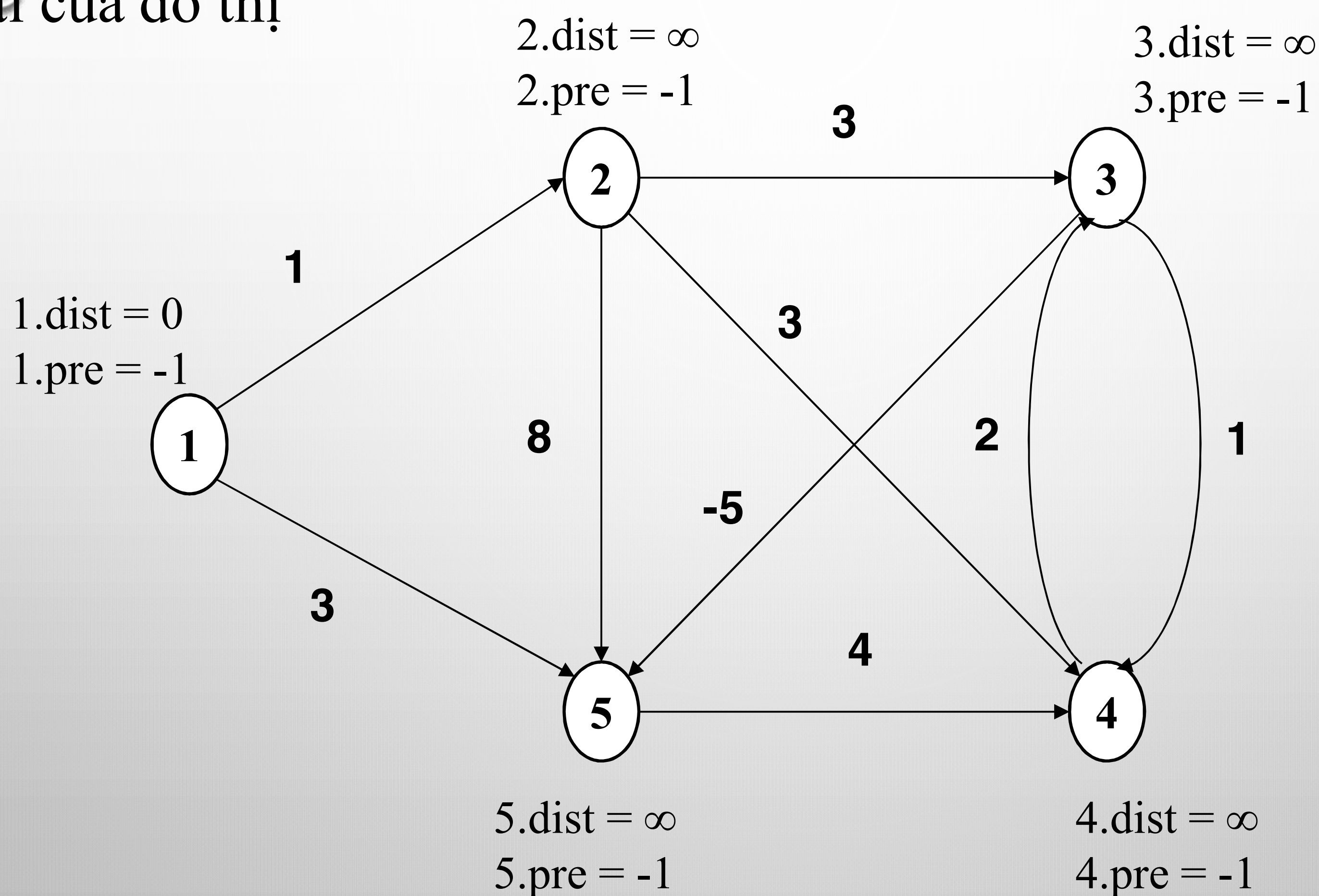
Cấu trúc đồ thị thường dùng là danh sách cung

```
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)

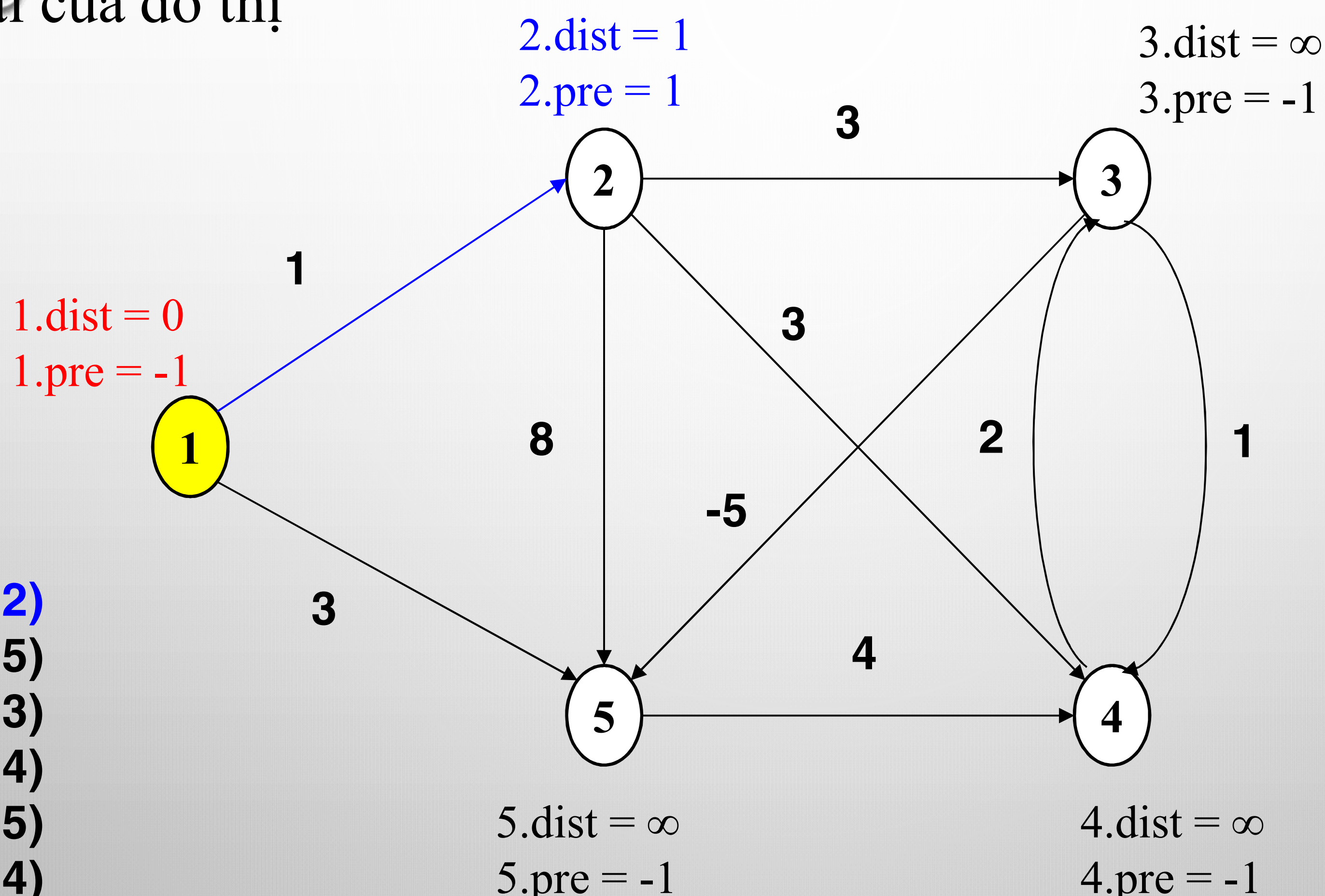
Tìm đường đi ngắn nhất từ đỉnh xuất phát 1 đến tất cả các đỉnh còn lại của đồ thị



(1, 2)
(1, 5)
(2, 3)
(2, 4)
(2, 5)
(3, 4)
(3, 5)
(4, 3)
(5, 4)

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



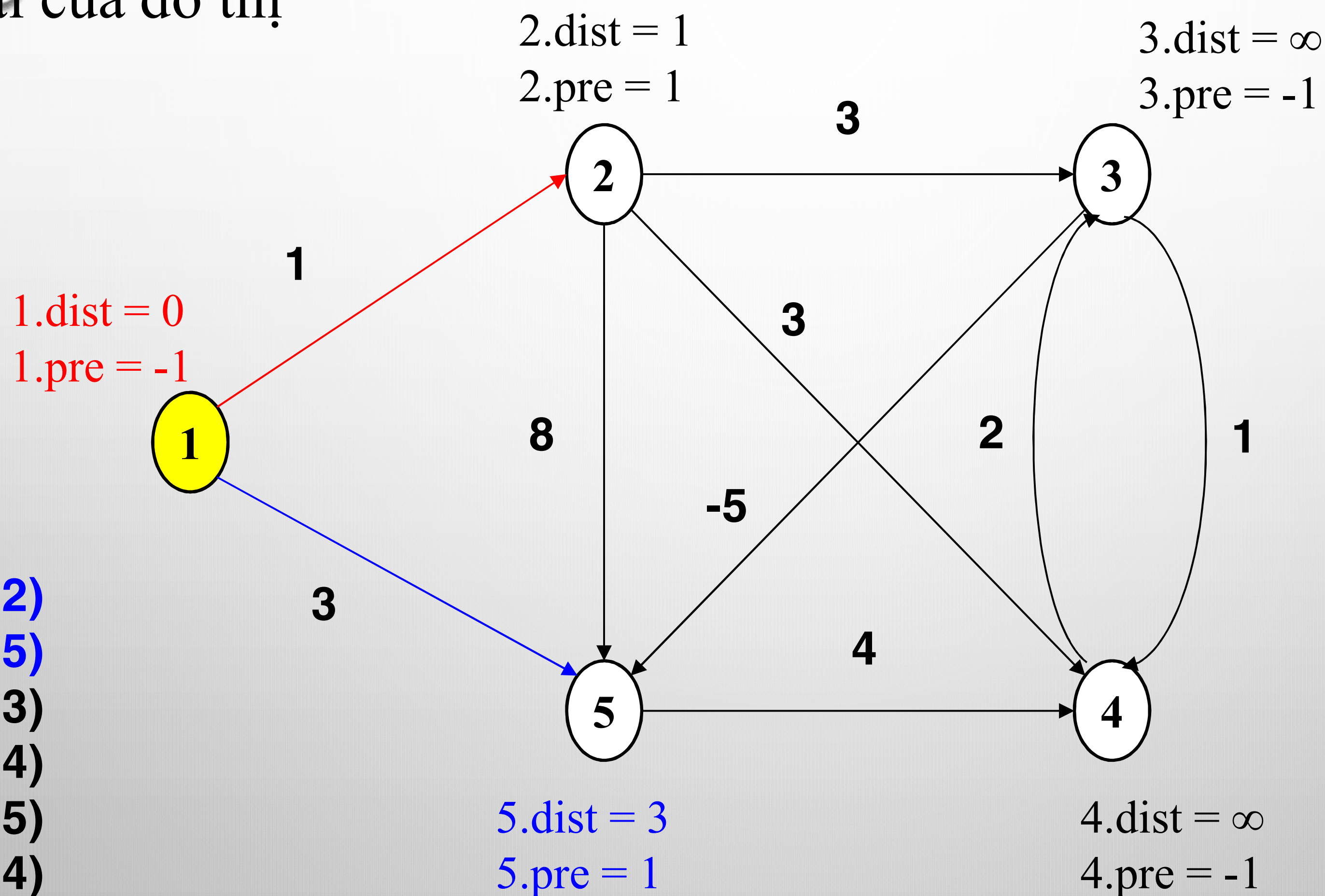
(1, 2)
(1, 5)
(2, 3)
(2, 4)
(2, 5)
(3, 4)
(3, 5)
(4, 3)
(5, 4)

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

L.T.D.D.

THUẬT TOÁN BELLMAN – FORD (– MOORE)

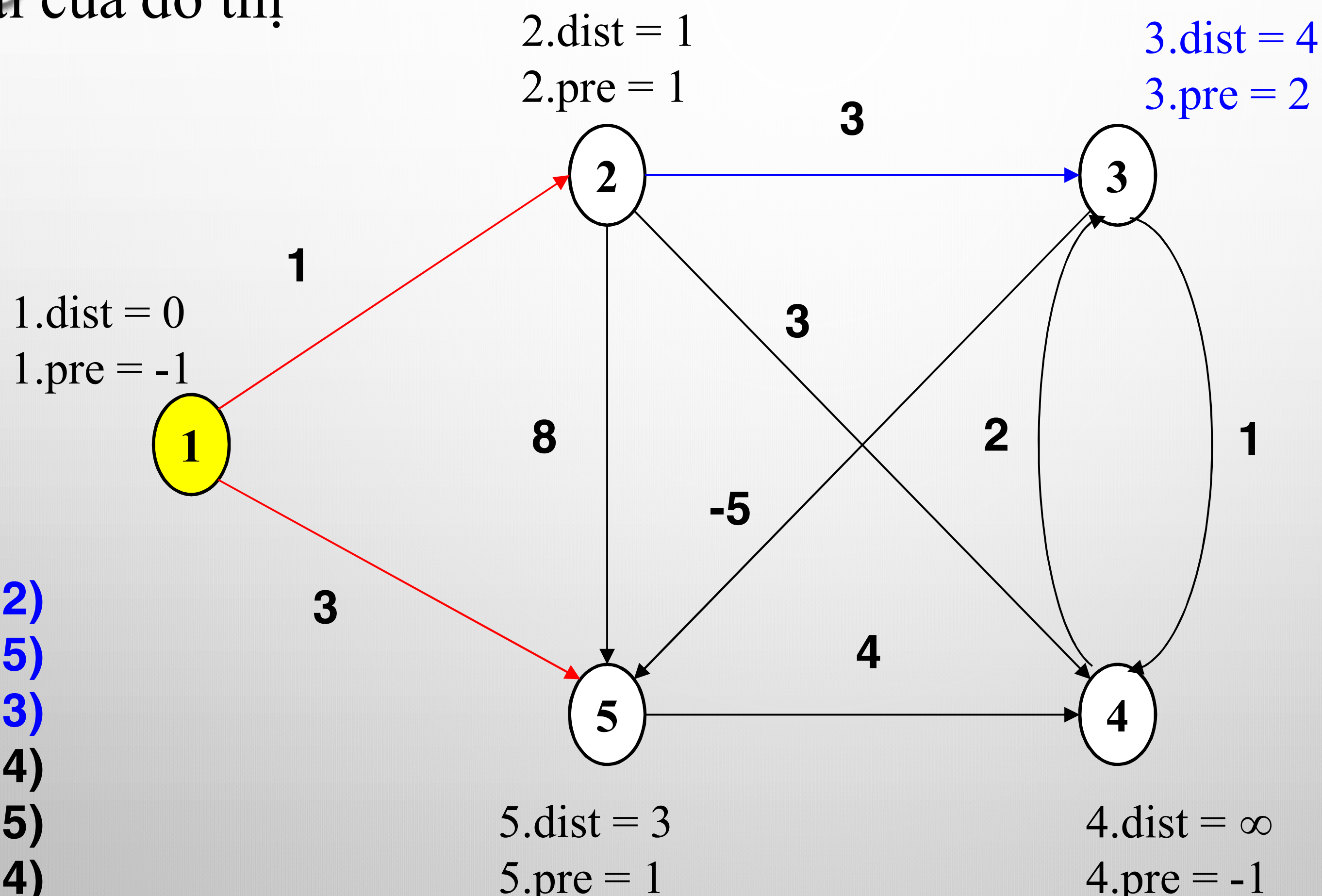
Tìm đường đi ngắn nhất từ đỉnh xuất phát 1 đến tất cả các đỉnh còn lại của đồ thị



(1, 2)
(1, 5)
(2, 3)
(2, 4)
(2, 5)
(3, 4)
(3, 5)
(4, 3)
(5, 4)

THUẬT TOÁN BELLMAN – FORD (– MOORE)

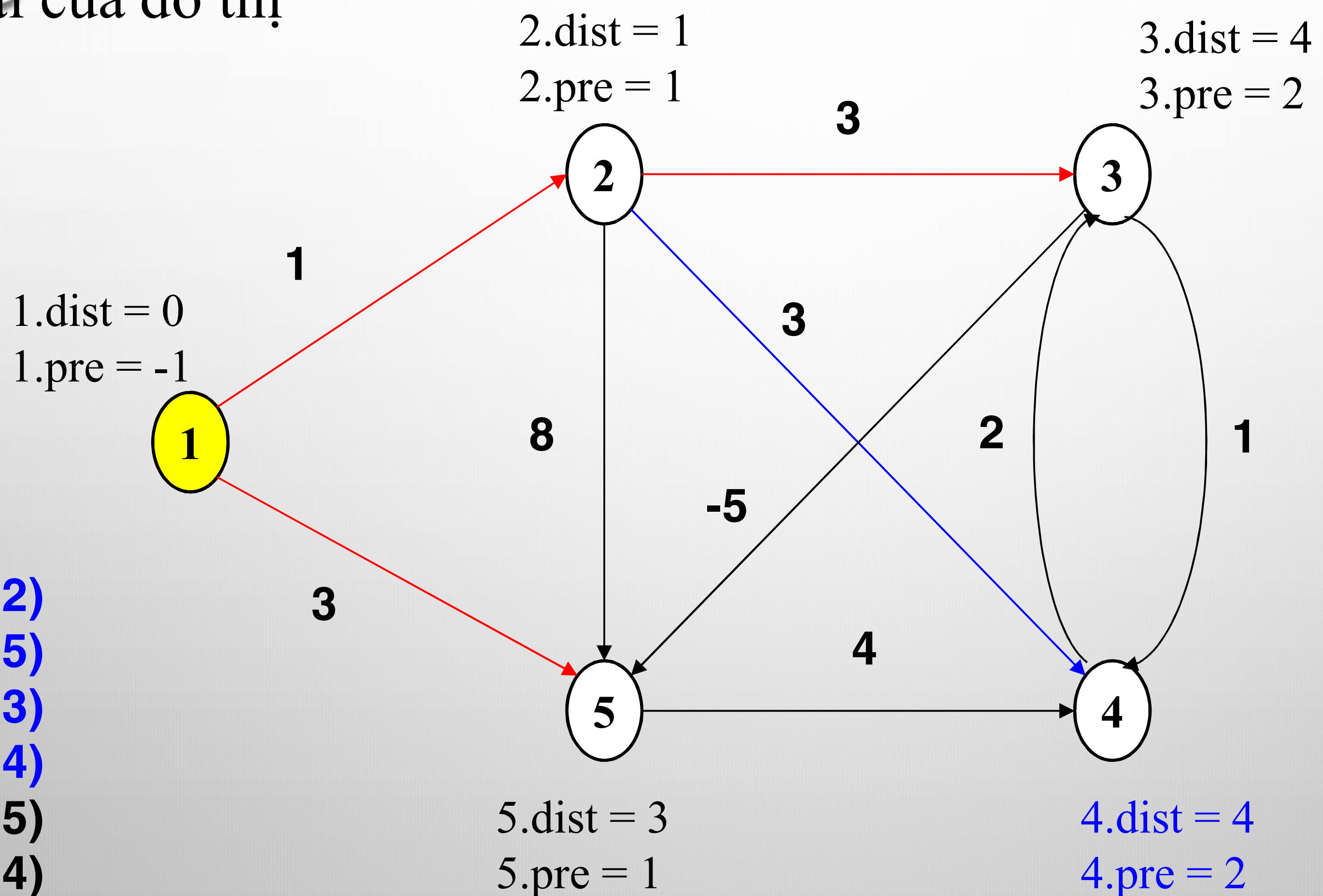
Tìm đường đi ngắn nhất từ đỉnh xuất phát 1 đến tất cả các đỉnh còn lại của đồ thị



(1, 2)
(1, 5)
(2, 3)
(2, 4)
(2, 5)
(3, 4)
(3, 5)
(4, 3)
(5, 4)

THUẬT TOÁN BELLMAN – FORD (– MOORE)

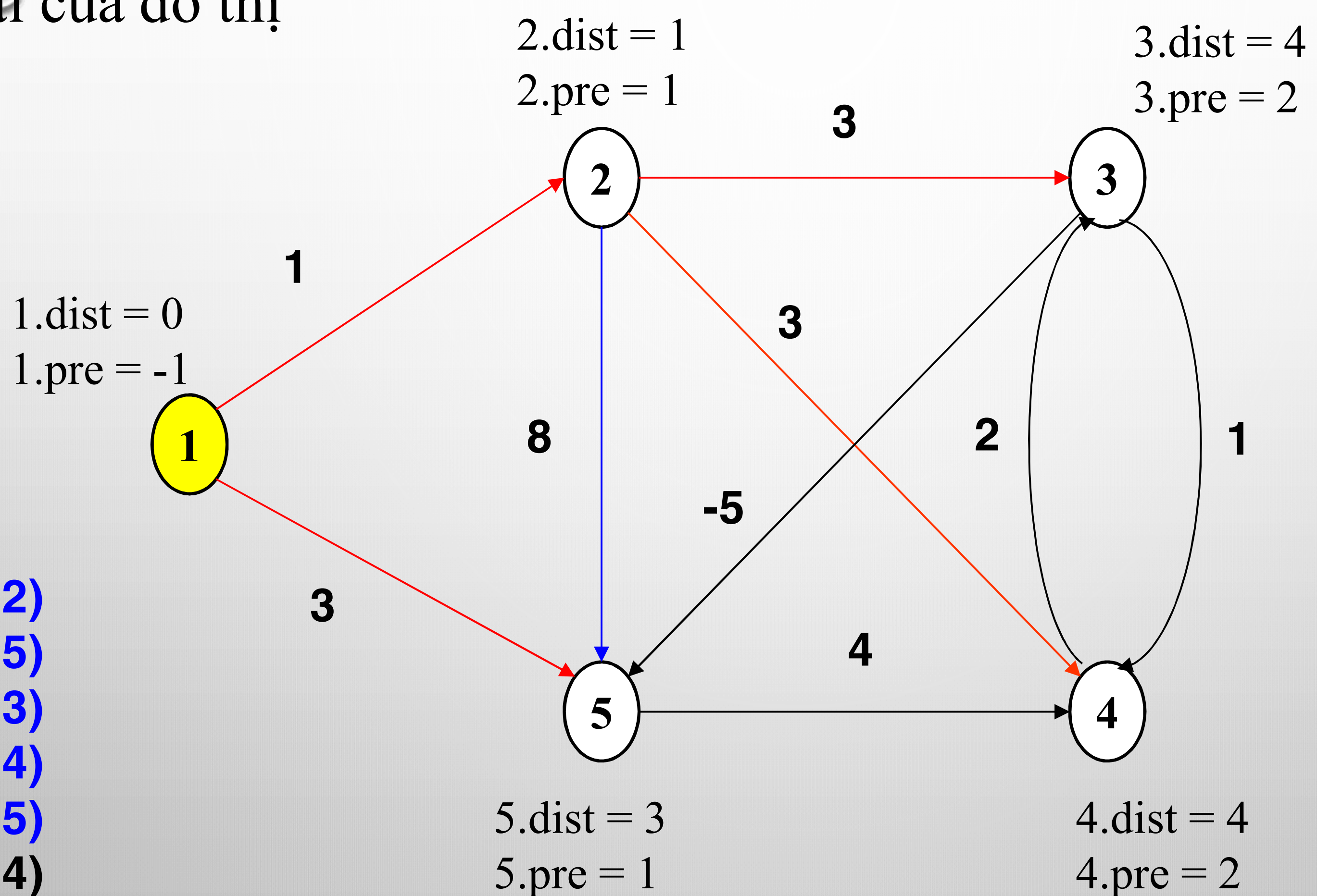
Tìm đường đi ngắn nhất từ đỉnh xuất phát 1 đến tất cả các đỉnh còn lại của đồ thị



(1, 2)
(1, 5)
(2, 3)
(2, 4)
(2, 5)
(3, 4)
(3, 5)
(4, 3)
(5, 4)

THUẬT TOÁN BELLMAN – FORD (– MOORE)

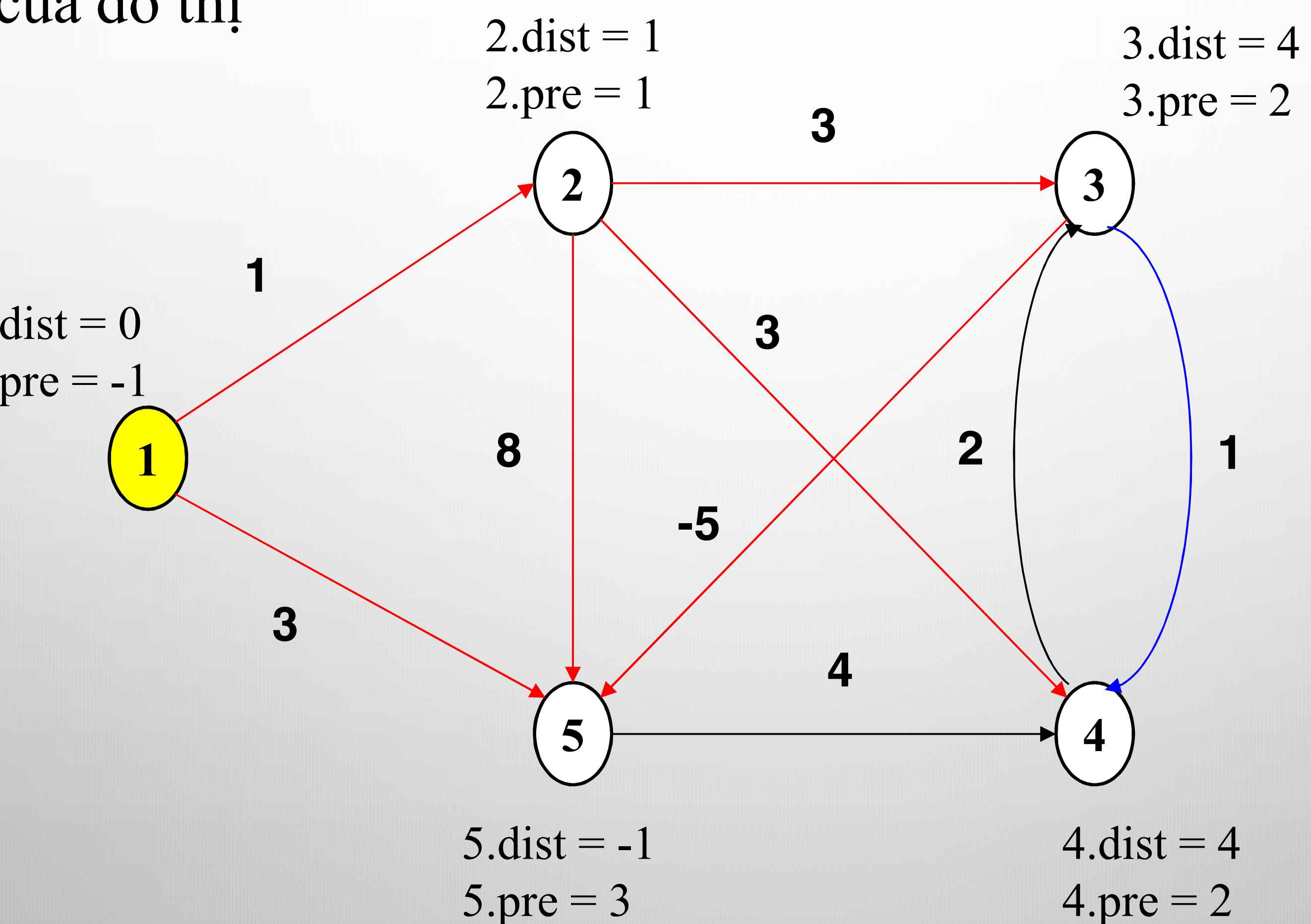
Tìm đường đi ngắn nhất từ đỉnh xuất phát 1 đến tất cả các đỉnh còn lại của đồ thị



(1, 2)
(1, 5)
(2, 3)
(2, 4)
(2, 5)
(3, 4)
(3, 5)
(4, 3)
(5, 4)

THUẬT TOÁN BELLMAN – FORD (– MOORE)

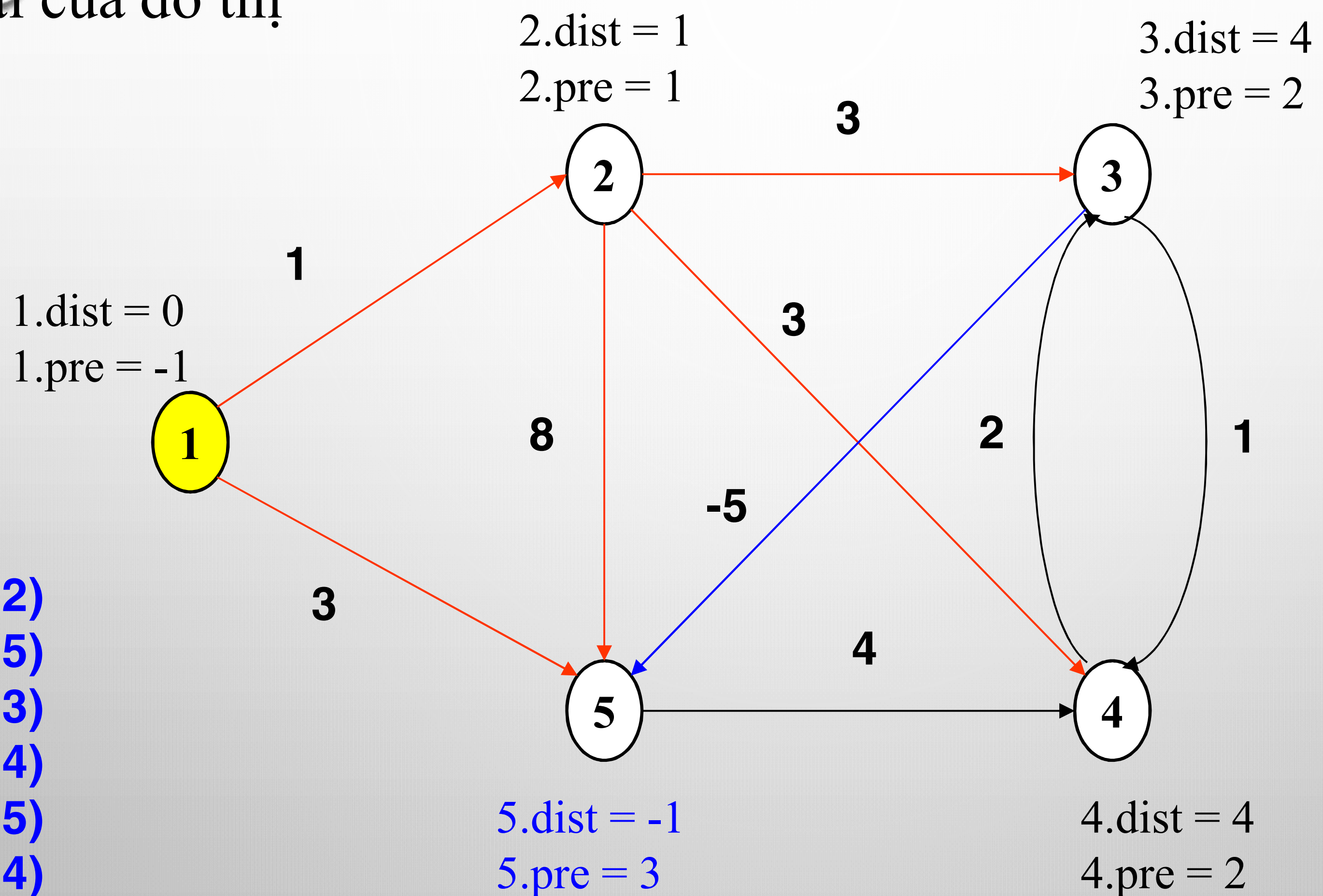
Tìm đường đi ngắn nhất từ đỉnh xuất phát 1 đến tất cả các đỉnh còn lại của đồ thị



(1, 2)
(1, 5)
(2, 3)
(2, 4)
(2, 5)
(3, 4)
(3, 5)
(4, 3)
(5, 4)

THUẬT TOÁN BELLMAN – FORD (– MOORE)

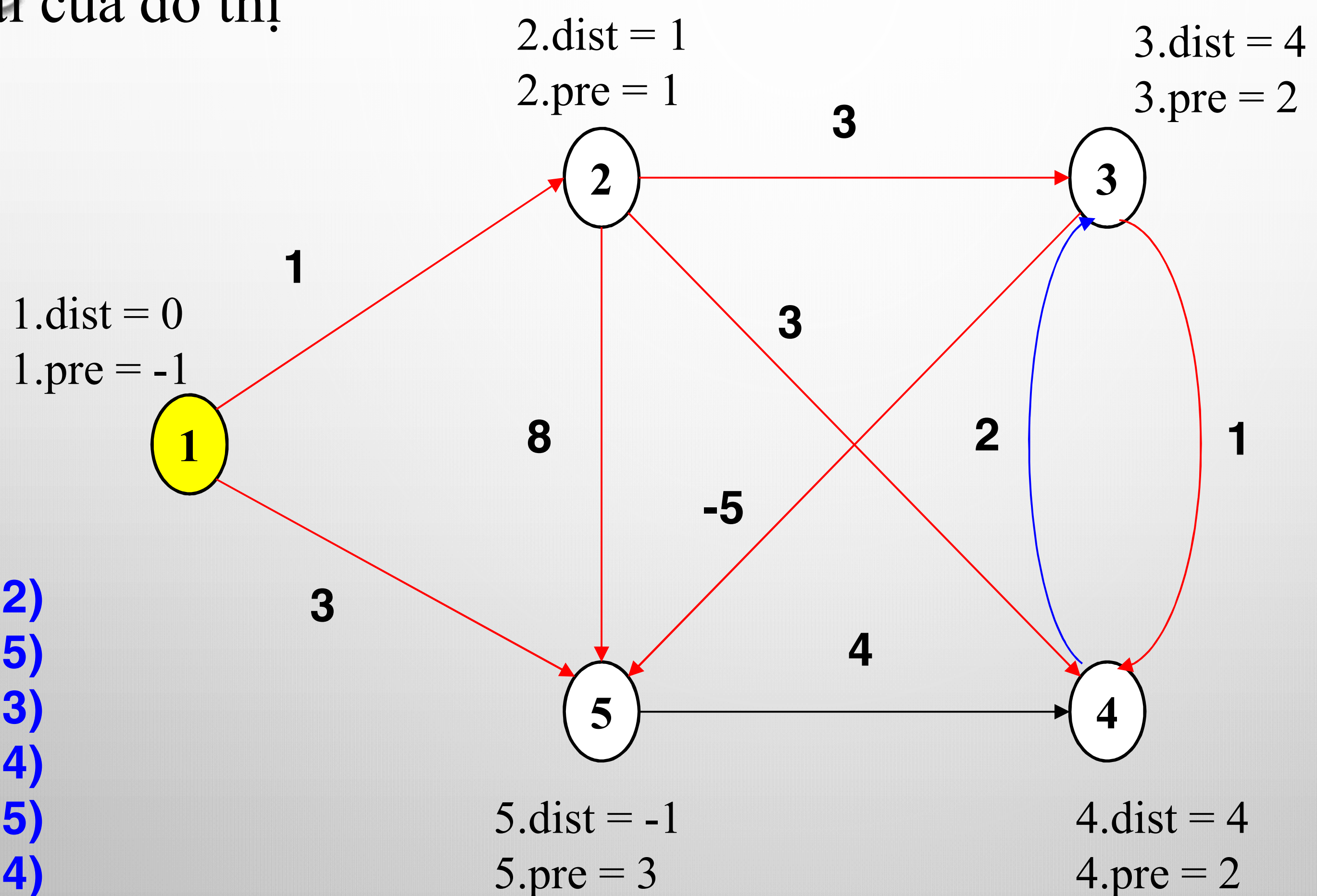
Tìm đường đi ngắn nhất từ đỉnh xuất phát 1 đến tất cả các đỉnh còn lại của đồ thị



(1, 2)
(1, 5)
(2, 3)
(2, 4)
(2, 5)
(3, 4)
(3, 5)
(4, 3)
(5, 4)

THUẬT TOÁN BELLMAN – FORD (– MOORE)

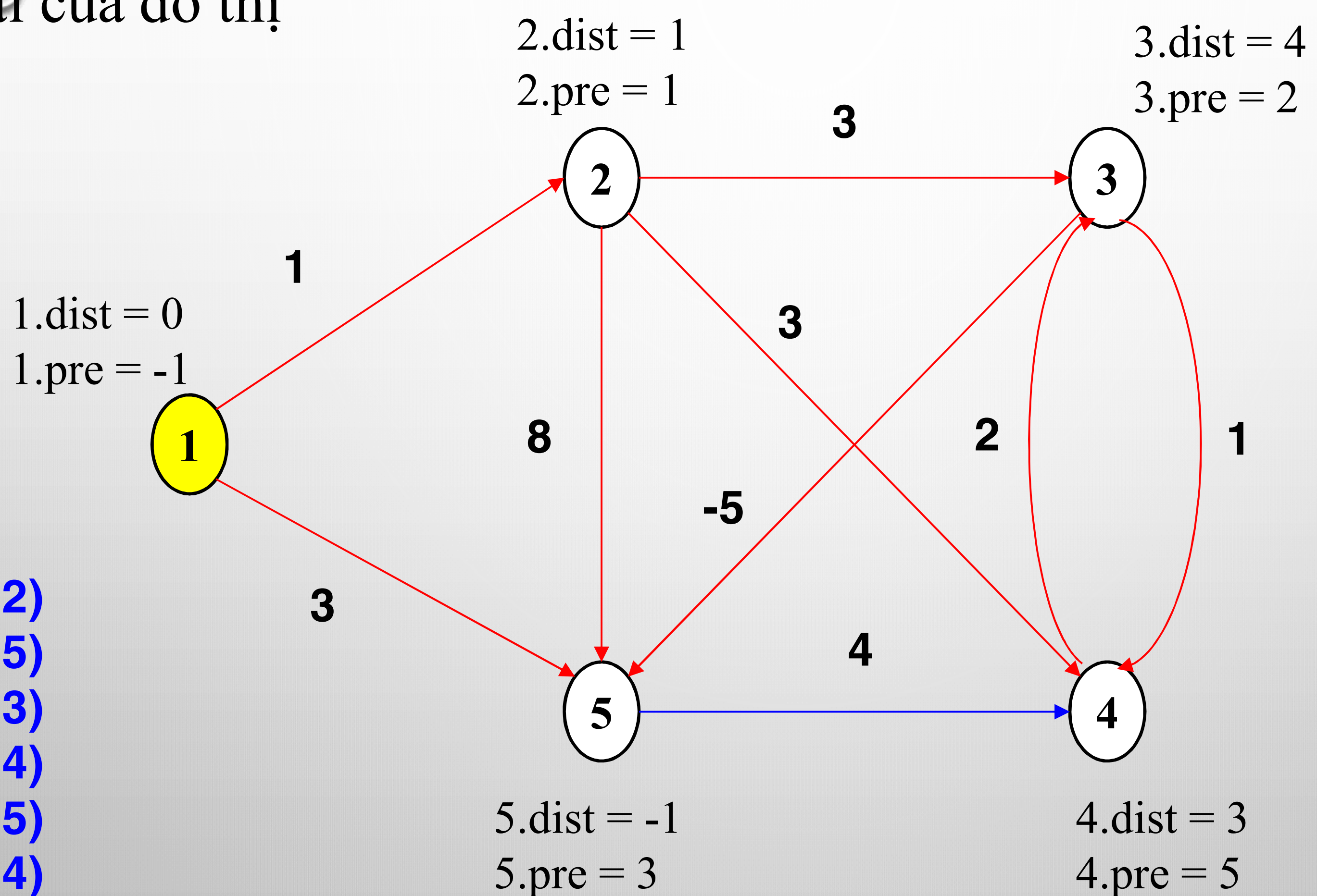
Tìm đường đi ngắn nhất từ đỉnh xuất phát 1 đến tất cả các đỉnh còn lại của đồ thị



(1, 2)
(1, 5)
(2, 3)
(2, 4)
(2, 5)
(3, 4)
(3, 5)
(4, 3)
(5, 4)

THUẬT TOÁN BELLMAN – FORD (– MOORE)

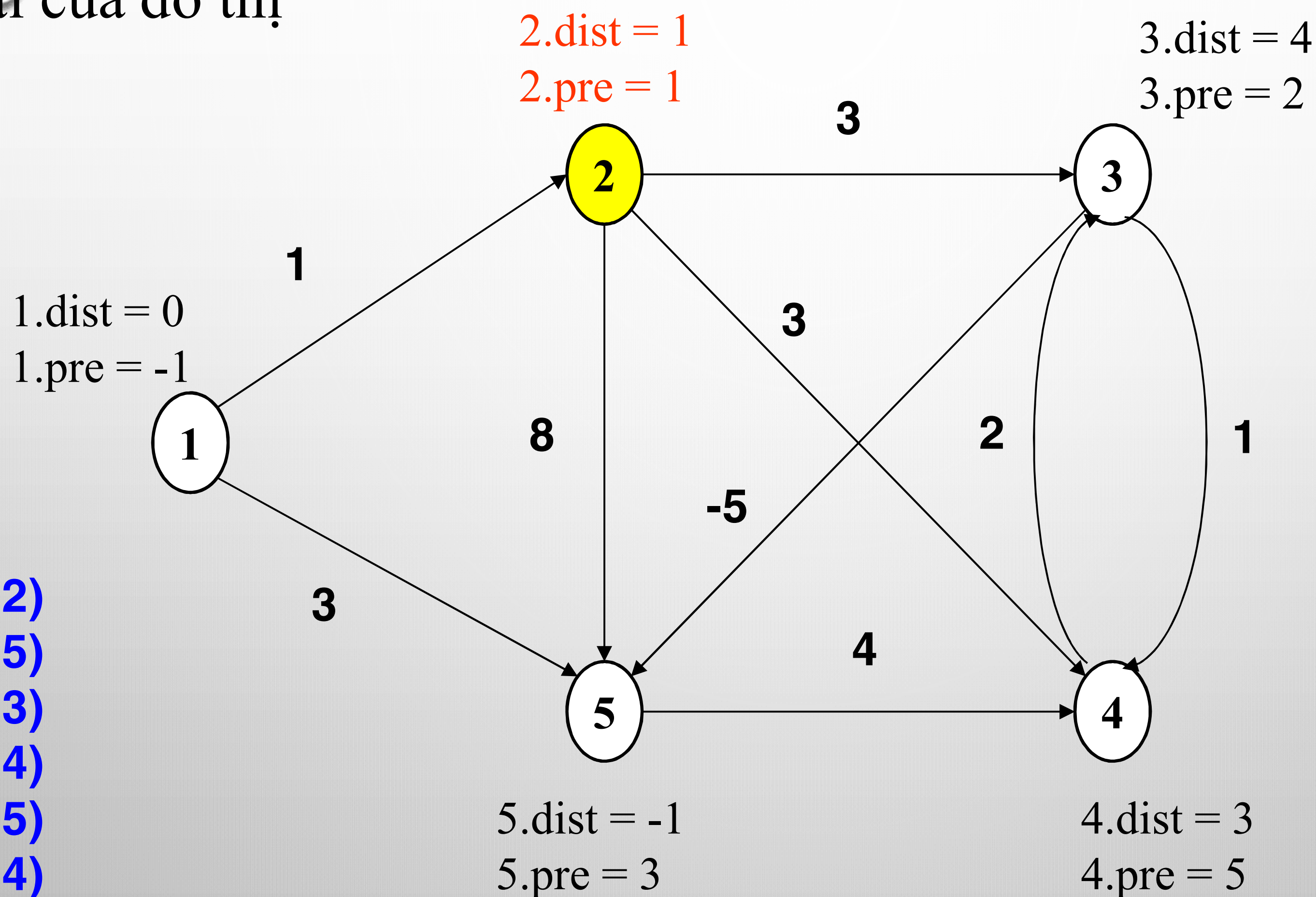
Tìm đường đi ngắn nhất từ đỉnh xuất phát 1 đến tất cả các đỉnh còn lại của đồ thị



(1, 2)
(1, 5)
(2, 3)
(2, 4)
(2, 5)
(3, 4)
(3, 5)
(4, 3)
(5, 4)

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

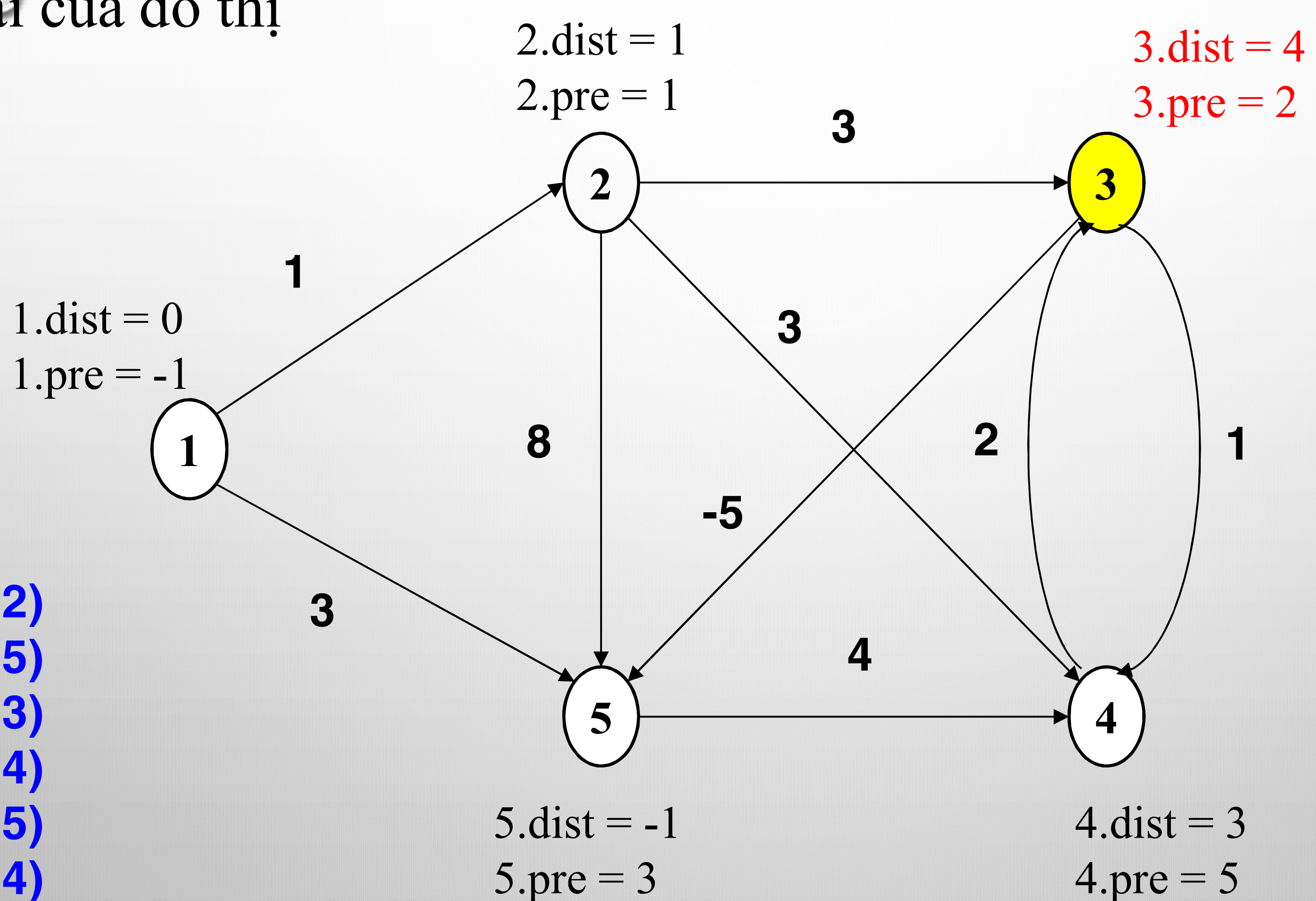


Xét đỉnh 2:
Duyệt qua tất cả các cung của đồ thị và tiến hành cập nhật nhãn tại các đỉnh nếu đường đi mới ngắn hơn đường đi cũ

(1, 2)
(1, 5)
(2, 3)
(2, 4)
(2, 5)
(3, 4)
(3, 5)
(4, 3)
(5, 4)

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



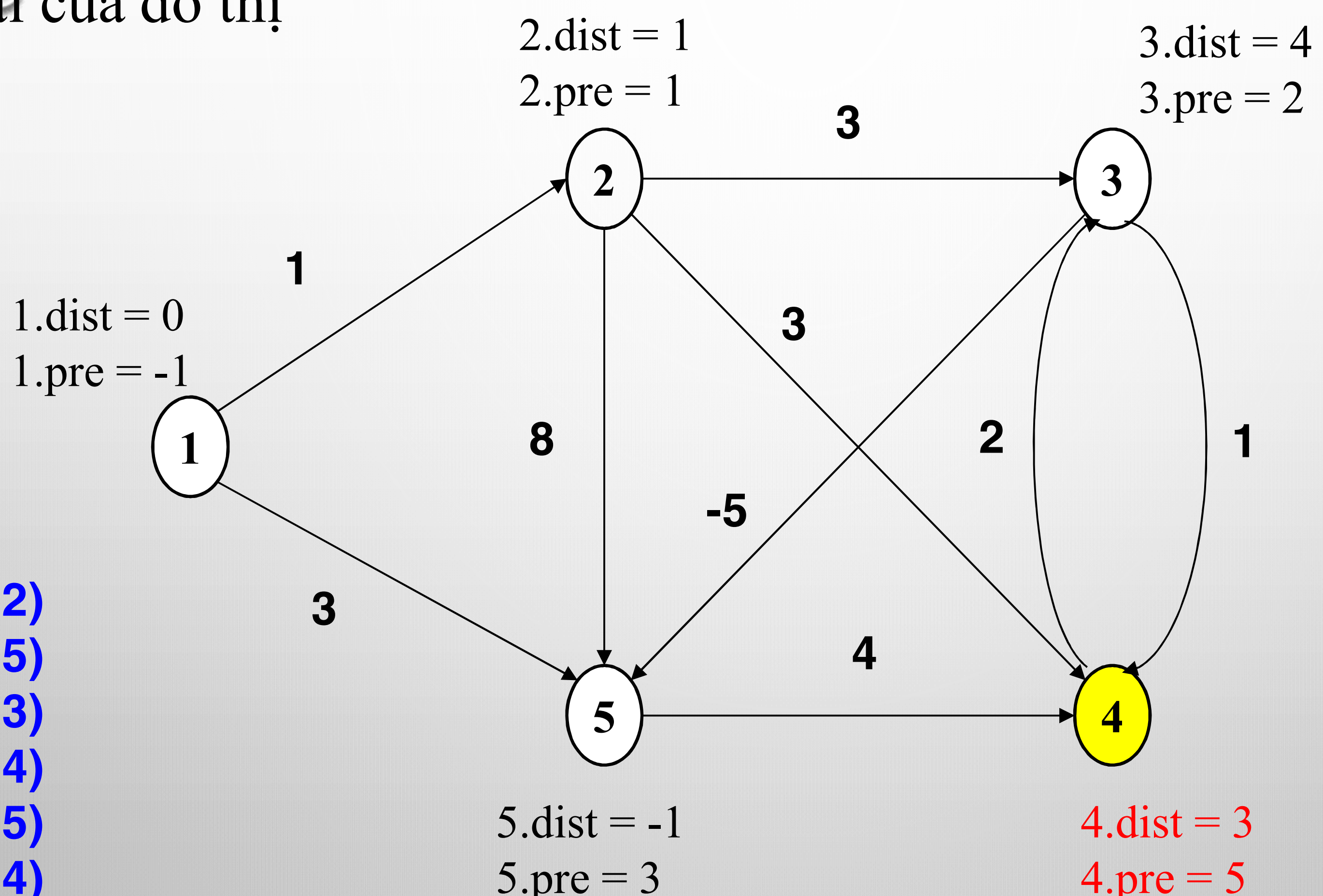
(1, 2)
(1, 5)
(2, 3)
(2, 4)
(2, 5)
(3, 4)
(3, 5)
(4, 3)
(5, 4)

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

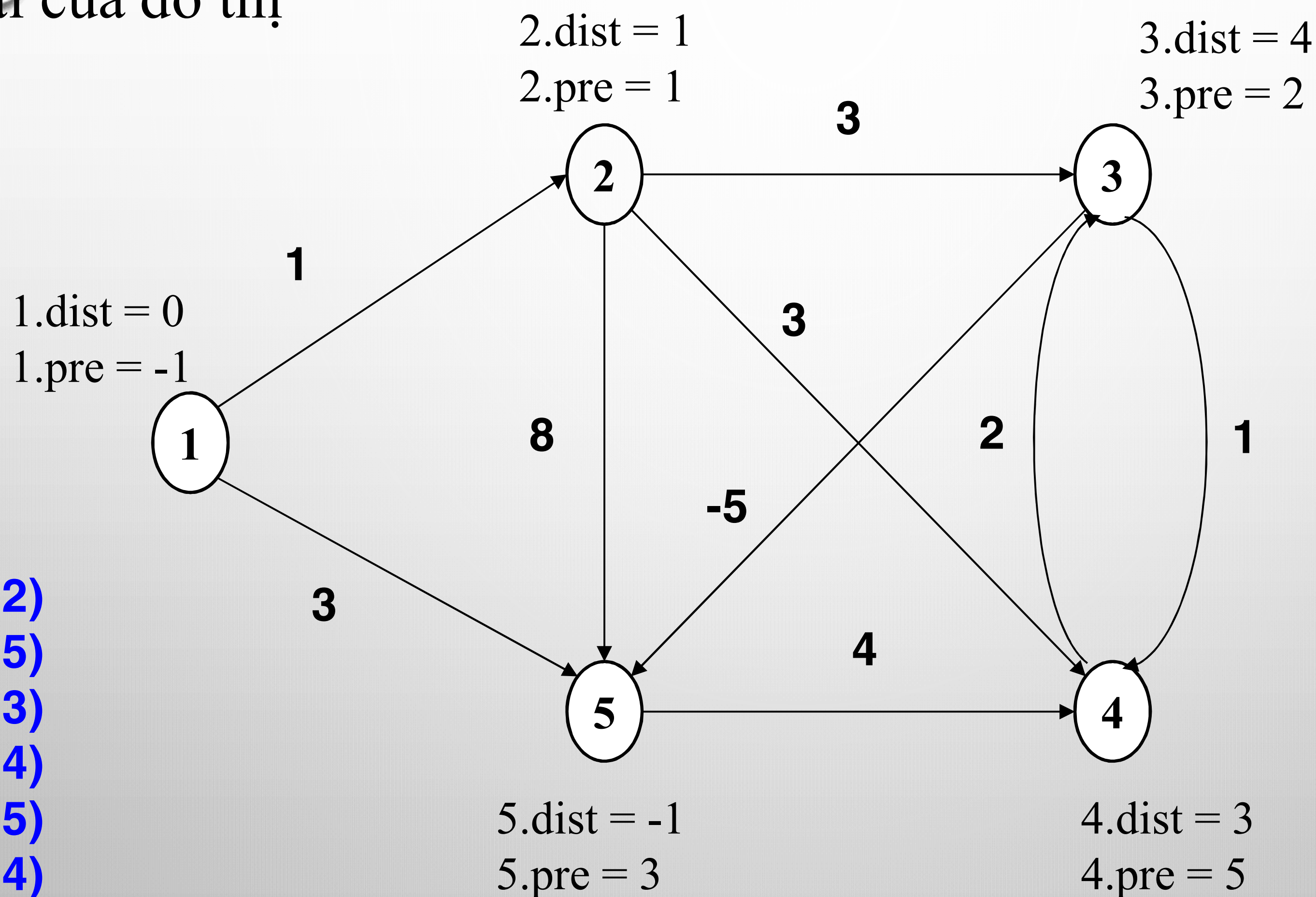


(1, 2)
(1, 5)
(2, 3)
(2, 4)
(2, 5)
(3, 4)
(3, 5)
(4, 3)
(5, 4)

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



(1, 2)
(1, 5)
(2, 3)
(2, 4)
(2, 5)
(3, 4)
(3, 5)
(4, 3)
(5, 4)

Phát hiện chu trình âm:

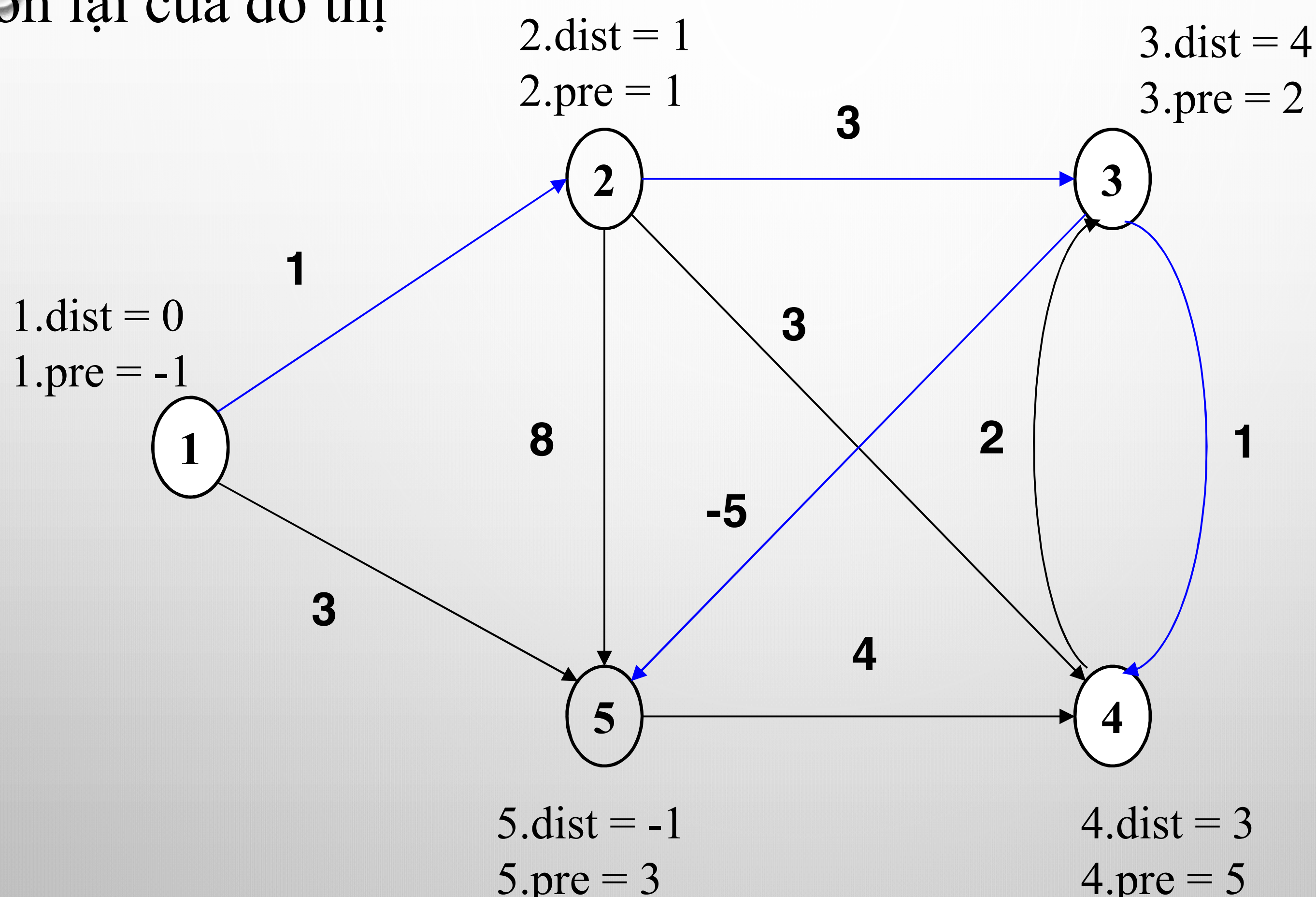
Duyệt qua tất cả các cung của đồ thị một lần nữa.

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.

L.T.D.D.

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



Không cần duyệt đỉnh 5.

Mời các em sinh viên giải thích!

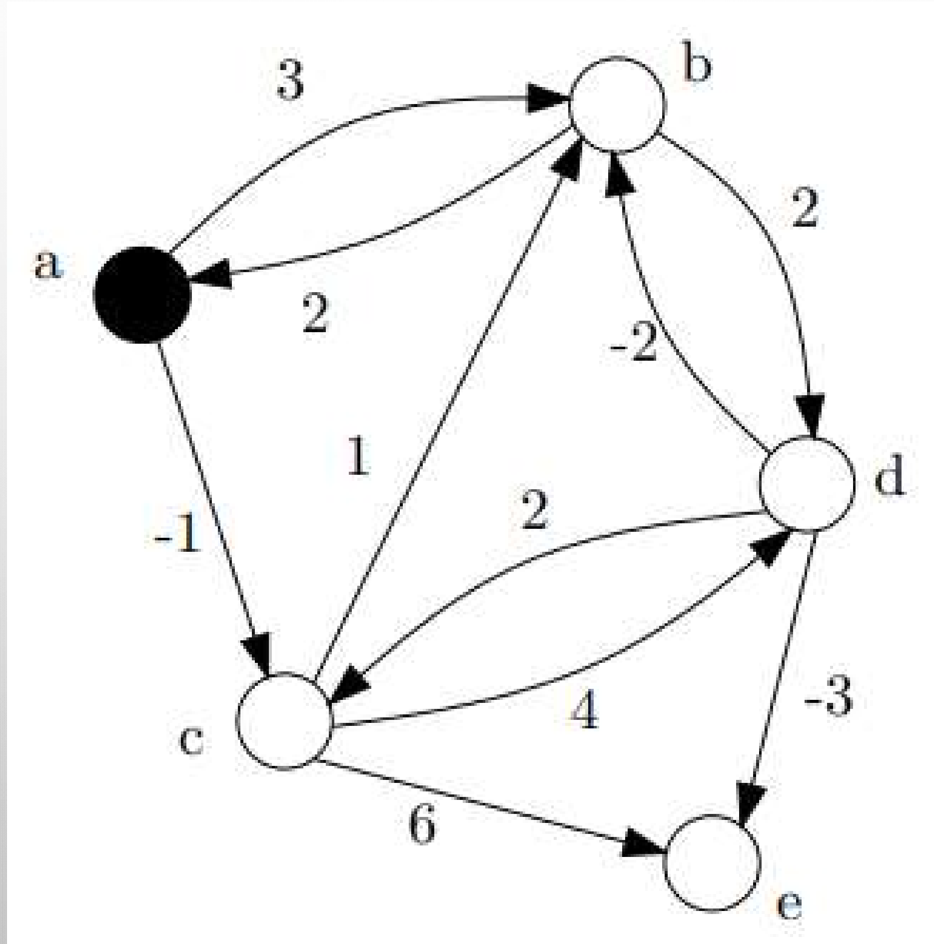
THUẬT TOÁN BELLMAN – FORD (– MOORE)

Mời các em sinh viên giải thích:

1. Trong đồ thị trên, tại sao không cần duyệt đỉnh 5?
2. Kết quả của mỗi bước lặp (sau khi duyệt một đỉnh) có thể khác nhau không? Tại sao?

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 BELLMAN – FORD (– MOORE)

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