# Analysis and Design of Algorithms

Lecture 6,7

# The Greedy algorithms

# Nội dung

1. Lược đồ chung
2. Bài toán cái túi
3. Bài toán người du lịch
4. Đường đi ngắn nhất
5. Cây bao trùm nhỏ nhất
6. Bài toán tô màu
7. Bài toán các khoảng không giao nhau

# Nội dung

1. **Lược đồ chung**
2. Bài toán cái túi
3. Bài toán người du lịch
4. Đường đi ngắn nhất
5. Cây bao trùm nhỏ nhất
6. Bài toán tô màu
7. Bài toán các khoảng không giao nhau

# Bài toán tối ưu

- PP Tham lam thường dùng cho các bài toán tối ưu tổ hợp (tối ưu rời rạc)

- Bài toán tối ưu tổ hợp có dạng chung

$$\min\{f(x): x \in D\}$$

Trong đó D tập hữu hạn các điểm rời rạc nào đó thuộc không gian $R^n$

# Ví dụ

- Máy ATM có 4 (m) loại tiền: 100.000, 50.000, 20.000, 10.000; một người muốn rút số tiền là n (n chia hết cho 10.000). Hãy tìm phương án trả tiền sao cho số tờ tiền phải trả là ít nhất.

- Gọi x=(x1,x2,x3,x4) là một phương án trả tiền; x1, x2, x3, x4 là số tờ tiền phải trả tương ứng với các mệnh giá 100.000, 50.000, 20.000,10.000.
- Theo bài ra ta cần giải:

$$\min(f=x1+x2+x3+x4)$$

Với: điều kiện

- $100.000x1+50.000x2+20.000x3+10.000x4 = n$
- $x_i >= 0$ (i=1..4)

# Giải quyết …

- Với bài toán tối ưu tổ hợp

$$\min\{f(x):x \in D\}$$

- Để tìm phương án tối ưu của bài toán trên người ta có thể so sánh lần lượt giá trị của f tại tất cả các phương án thuộc D; cách này gọi là "duyệt vét cạn".
- Khi số phần tử của D lớn (dù là hữu hạn) thì việc duyệt vét cạn vẫn gặp nhiều khó khăn.

# PP Tham lam

- PP tham lam đưa ra quyết định dựa ngay vào thông tin đang có, và trong tương lai sẽ không xem xét lại tác động của các quyết định trong quá khứ.
- Chính vì thế các thuật toán dạng này rất dễ đề xuất, và thông thường chúng không đòi hỏi nhiều thời gian tính.
- Tuy nhiên, các thuật toán dạng này thường không cho kết quả tối ưu.

# Ý tưởng

- Xuất phát từ lời giải rỗng, thuật toán xây dựng lời giải của bài toán theo từng bước, ở mỗi bước sẽ chọn một phần tử từ tập ứng cử viên và bổ sung vào lời giải hiện có.
- Hàm Solution(S) nhận biết tính chấp nhận được của lời giải S.
- Hàm Select(C) chọn từ tập C ứng cử viên có triển vọng nhất để bổ sung vào lời giải hiện có.
- Hàm Feasible(S+x) kiểm tra tính chấp nhận được của lời giải bộ phận S+x.

# Lược đồ chung

```
procedure Greedy;
(* Giả sử C là tập các ứng cử viên *)
begin
   S: = ∅; (*  S lời giải xây dựng theo thuật toán *)
   while (C ≠ ∅) and not Solution(S) do
   begin
        x ← Select(C);
        C:= C \ x;
        if Feasible (S ∪ x) then S:= S ∪ x ;
   end;
   if Solution(S) then return S;
end;
```

# Tính đúng đắn của kết quả

- Để chỉ ra thuật toán không đúng đắn chỉ cần đưa ra một phản ví dụ (một bộ dữ liệu mà đối với nó thuật toán không cho lời giải đúng)
- Chứng minh tính đúng đắn của thuật toán khó hơn nhiều

# Nội dung

1. Lược đồ chung
2. **Bài toán cái túi**
3. Bài toán người du lịch
4. Đường đi ngắn nhất
5. Cây bao trùm nhỏ nhất
6. Bài toán tô màu
7. Bài toán các khoảng không giao nhau

# Bài toán

(**Knapsack Problem**)

- Có **n** đồ vật, đồ vật i có trọng lượng $w_i$ và giá trị $c_i$, i = 1, 2, ..., n.
- Tìm cách chất các đồ vật này vào cái túi có trọng lượng là **b** sao cho tổng trọng lượng của các đồ vật được chất vào túi là không quá b, đồng thời tổng giá trị của chúng là lớn nhất.

# Khái quát

- Ký hiệu C = {1, 2, ..., n} tập chỉ số các đồ vật.
- Bài toán đặt ra là Tìm I ⊂ C sao cho

$$V = \sum_{i \in I} c_i \rightarrow \max$$

với

$$\sum_{i \in I} w_i \leq b,$$


Biên soạn: Hà Đại Dương, duonghd@mta.edu.vn

# Tham lam 1 (Greedy1)

- Ý tưởng (tham lam): Đồ vật có giá trị lớn (nhất) còn lại được lấy trước (nếu có thể).
- Chi tiết:
  - Sắp xếp các đồ vật theo thứ tự không tăng của giá trị.
  - Chọn đồ vật từ đầu đến cuối (từ có giá trị cao đến có giá trị thấp hơn) nếu dung lượng còn lại của túi đủ chứa nó.

# Ví dụ 1

- Số lượng đồ vật **n** = 3
- Trọng lượng và giá trị các đồ vật là:

| Đồ vật | 1 | 2 | 3 |
|---|---|---|---|
| Giá trị | 20 | 16 | 8 |
| Trọng lượng | 14 | 6 | 10 |

- Trọng lượng cái túi **b** = 19

**Greedy1** → **I={1}**, **V = 20**

**Tối ưu** → **I*={2,3}**, **V* = 24**


Biên soạn: Hà Đại Dương, duonghd@mta.edu.vn

# Tham lam 2 (Greedy2)

- Ý tưởng (tham lam): Đồ vật có trọng lượng nhỏ (nhất) còn lại được lấy trước (nếu có thể).
- Chi tiết:
  - Sắp xếp các đồ vật theo thứ tự không giảm của trọng lượng.
  - Chọn đồ vật từ đầu đến cuối (từ có trọng lượng cao đến có trọng lượng thấp hơn) nếu dung lượng còn lại của túi đủ chứa nó.

# Ví dụ 2

- Số lượng đồ vật **n** = 3
- Trọng lượng và giá trị các đồ vật là:

| Đồ vật | 1 | 2 | 3 |
|---|---|---|---|
| Giá trị | 10 | 16 | 28 |
| Trọng lượng | 5 | 6 | 10 |

- Trọng lượng cái túi **b** = 11

**Greedy2** → **I={1,2}** **V = 26**

**Tối ưu** → **I*={3}** **V* = 28**


Biên soạn: Hà Đại Dương, duonghd@mta.edu.vn

# Tham lam 3 (Greedy3)

- Ý tưởng (ít tham lam): Đồ vật có đơn giá lớn (nhất) còn lại được lấy trước (nếu có thể).
- Chi tiết:
  - Sắp xếp các đồ vật theo thứ tự không tăng của giá trị một đơn vị trọng lượng (cl/wl), nghĩa là.

$$\frac{c_{i_1}}{w_{i_1}} \geq \frac{c_{i_2}}{w_{i_2}} \geq \ldots \geq \frac{c_{i_n}}{w_{i_n}}$$

  - Chọn đồ vật từ đầu đến cuối ...

# Ví dụ 3

- Trường hợp 1 (**b=19**) (V=24)

| Đồ vật | 1 | 2 | 3 |
|---|---|---|---|
| Giá trị | 20 | 16 | 8 |
| Trọng lượng | 14 | 6 | 10 |

- Trường hợp 2 (**b=11**) (V=28)

| Đồ vật | 1 | 2 | 3 |
|---|---|---|---|
| Giá trị | 10 | 16 | 28 |
| Trọng lượng | 5 | 6 | 10 |

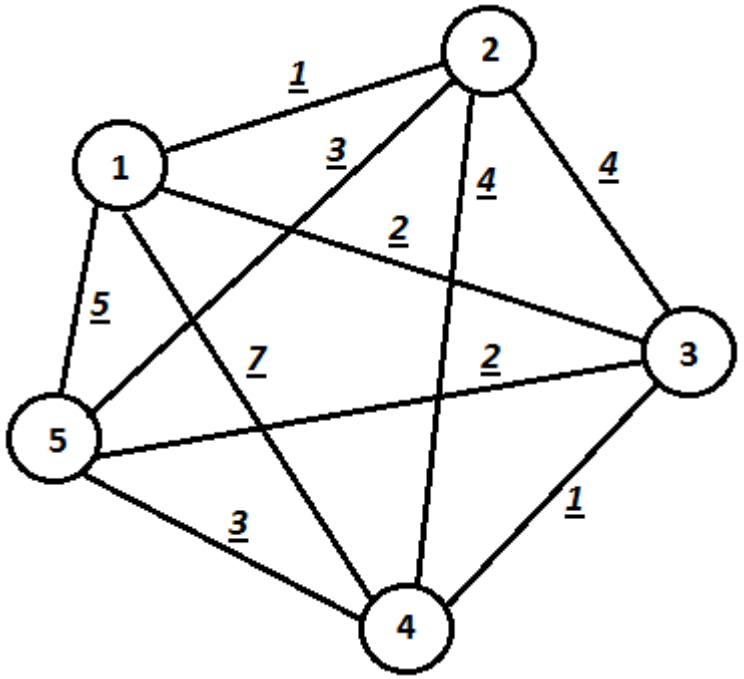
# Nội dung

1. Lược đồ chung
2. Bài toán cái túi
3. **Bài toán người du lịch**
4. Đường đi ngắn nhất
5. Cây bao trùm nhỏ nhất
6. Bài toán tô màu
7. Bài toán các khoảng không giao nhau

# Bài toán

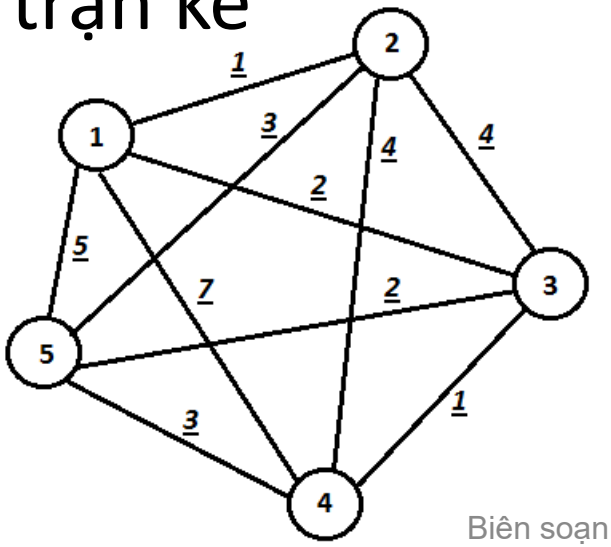
Một người du lịch muốn tham quan n thành phố $T_1$,.., $T_n$ . Xuất phát từ một thành phố nào đó, người du lịch muốn đi qua tất cả các thành phố còn lại, mỗi thành phố đi qua đúng 1 lần rồi quay trở lại thành phố xuất phát.

Gọi $C_{ij}$ là chi phí đi từ thành phố $T_i$ đến $T_j$ . Hãy tìm một hành trình thỏa yêu cầu bài toán sao cho chi phí là nhỏ nhất.



# Ý tưởng

- Ý tưởng (tham lam): Chọn thành phố gần nhất tình từ thành phố hiện thời.
- Tổ chức dữ liệu: Đồ thị G = (V,E), V – tập đỉnh (≡ T), E – Tập các cạnh (≡C). Mô tả đồ thị dạng ma trận kề




$$C = \begin{bmatrix} 0 & 1 & 2 & 7 & 5 \\ 1 & 0 & 4 & 4 & 3 \\ 2 & 4 & 0 & 1 & 2 \\ 7 & 4 & 1 & 0 & 3 \\ 5 & 3 & 2 & 3 & 0 \end{bmatrix}$$

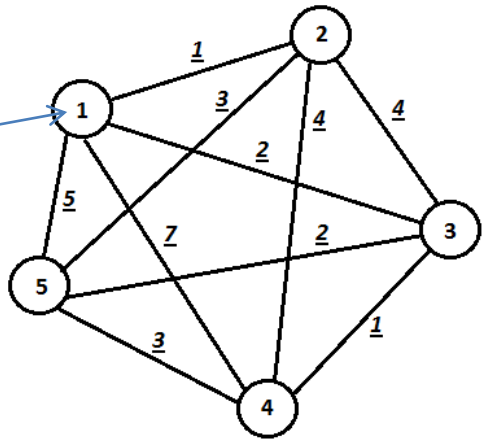

Biên soạn: Hà Đại Dương, duonghd@mta.edu.vn

# Minh họa …

- TOUR: Danh sách cạnh của hành trình
- COST: Chi phí theo hành trình TOUR
- u: Đỉnh hiện tại
- w: Kề với u có chi phí thấp nhất

**Với bài toán**

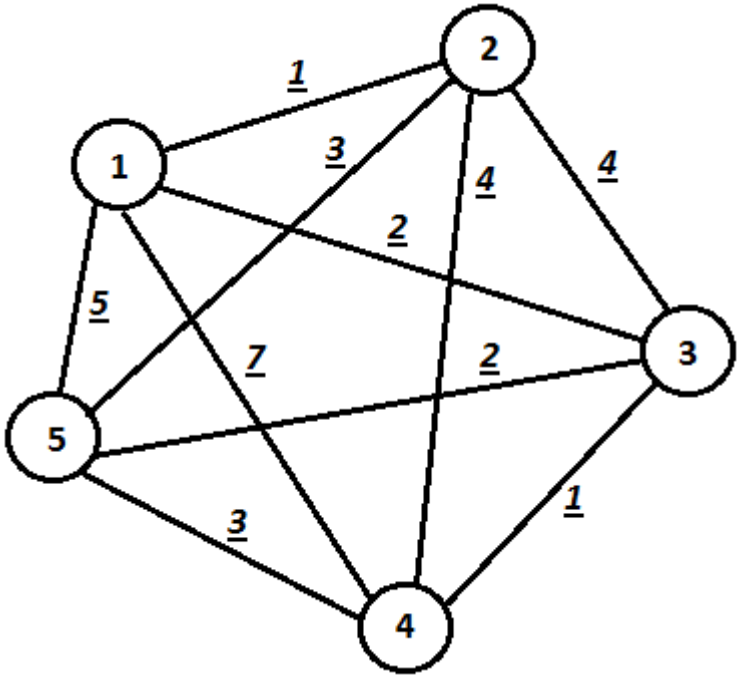
Xuất phát từ **1**

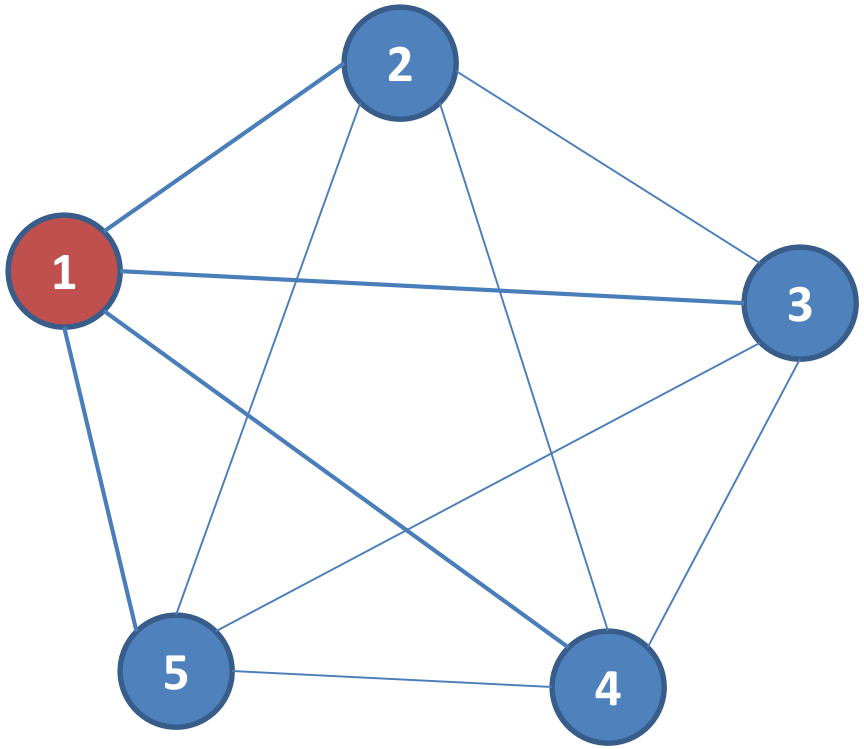




Biên soạn: Hà Đại Dương, duonghd@mta.edu.vn

# Minh họa …

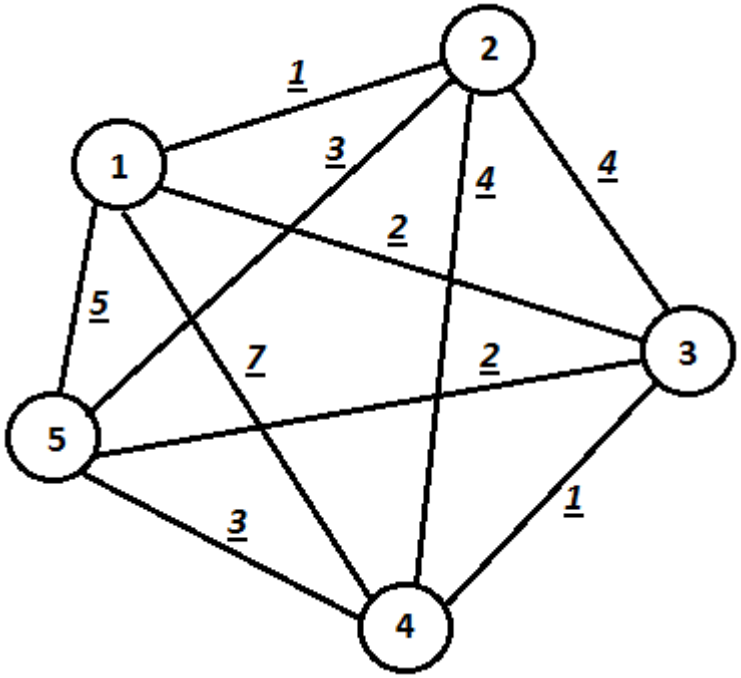


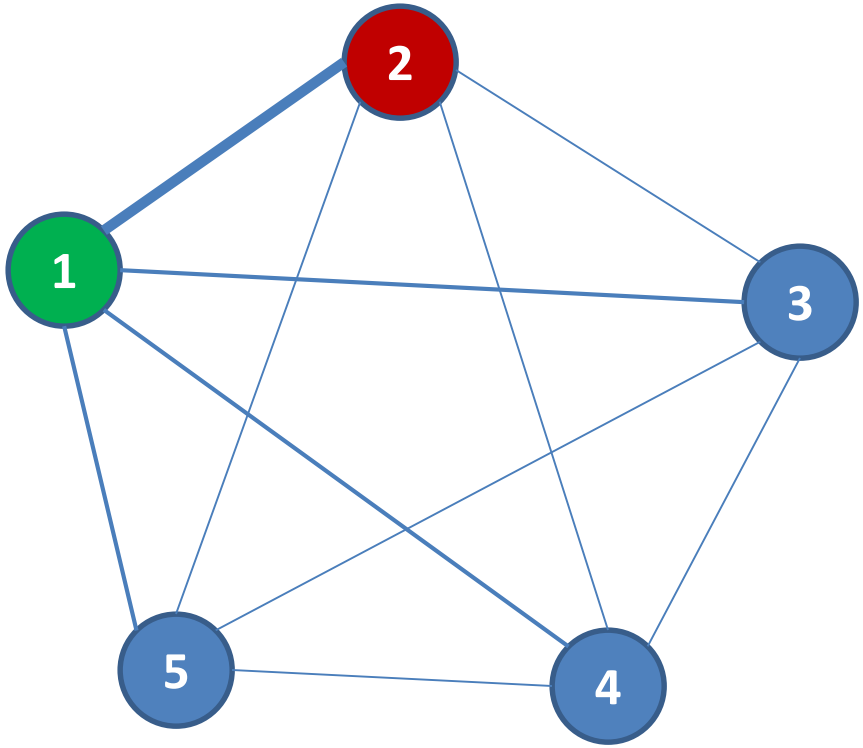




TOUR={}

COST=0

# Minh họa …

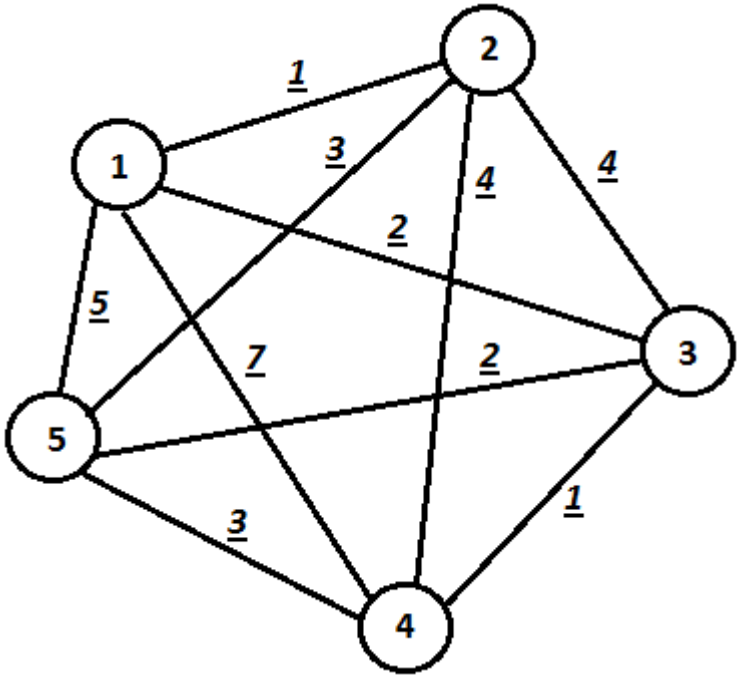


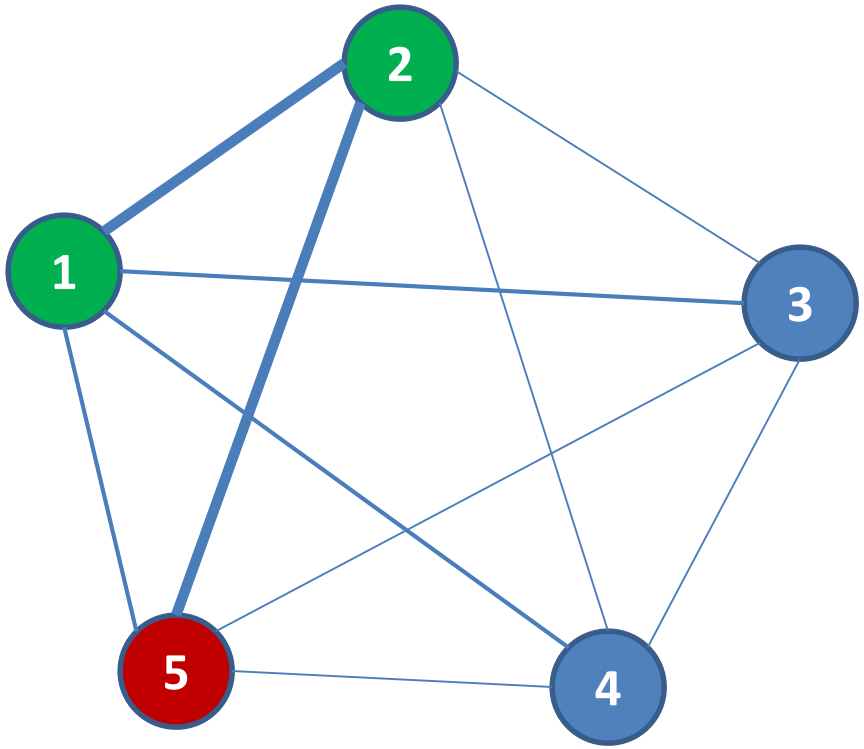




TOUR={(1,2)}

COST=1

# Minh họa …

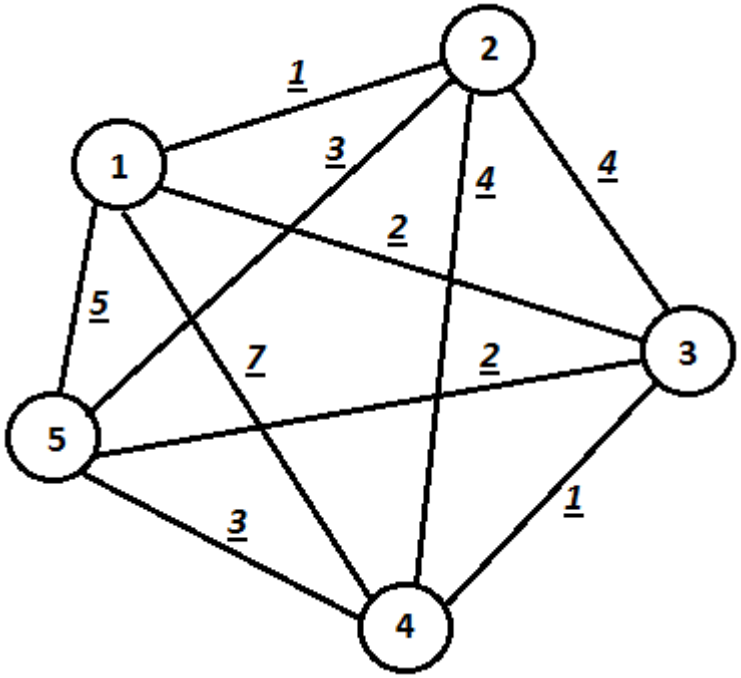


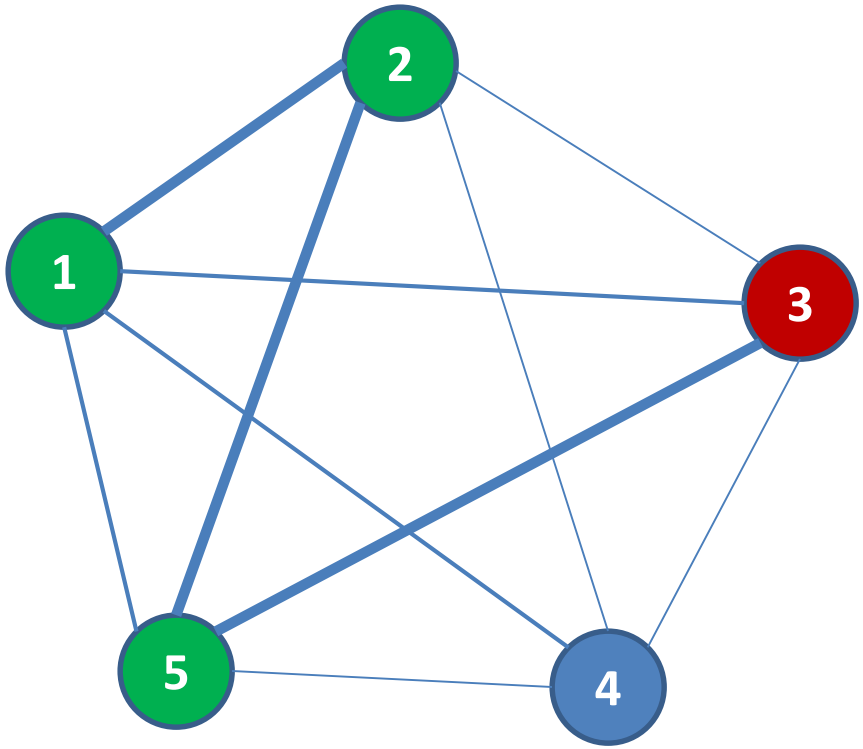




TOUR={(1,2), (2,5)}

COST=1+3

# Minh họa …

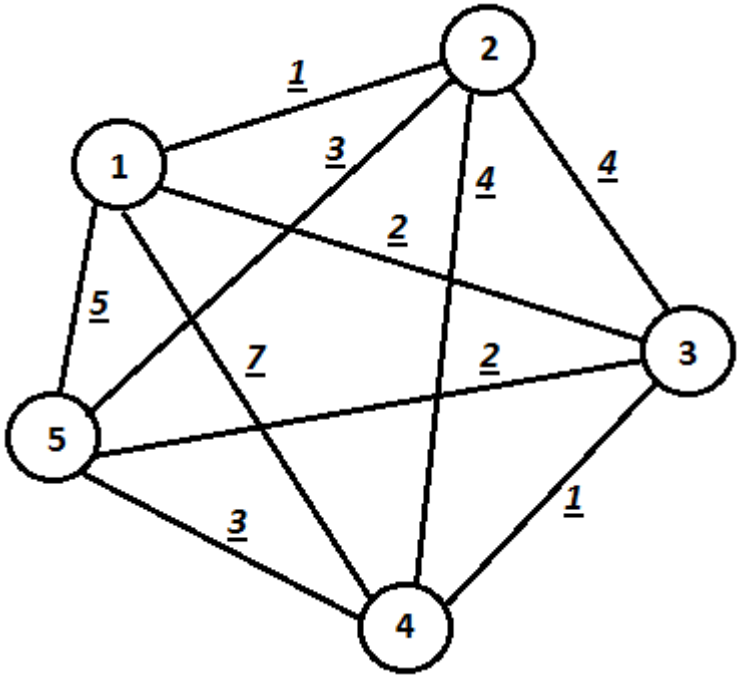


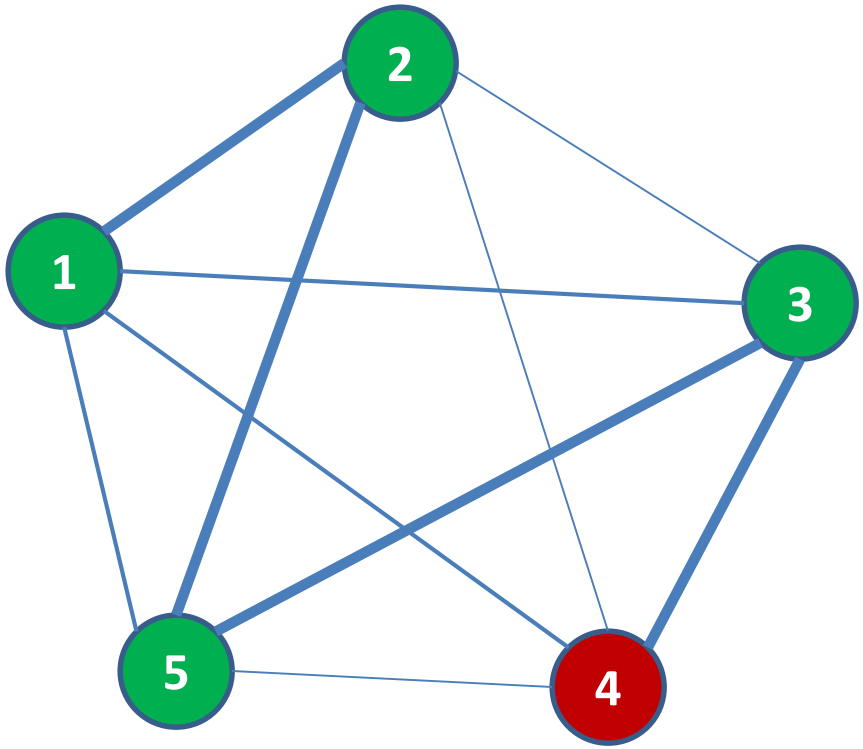




TOUR={(1,2), (2,5), (5,3)}

COST=1+3+2

# Minh họa …

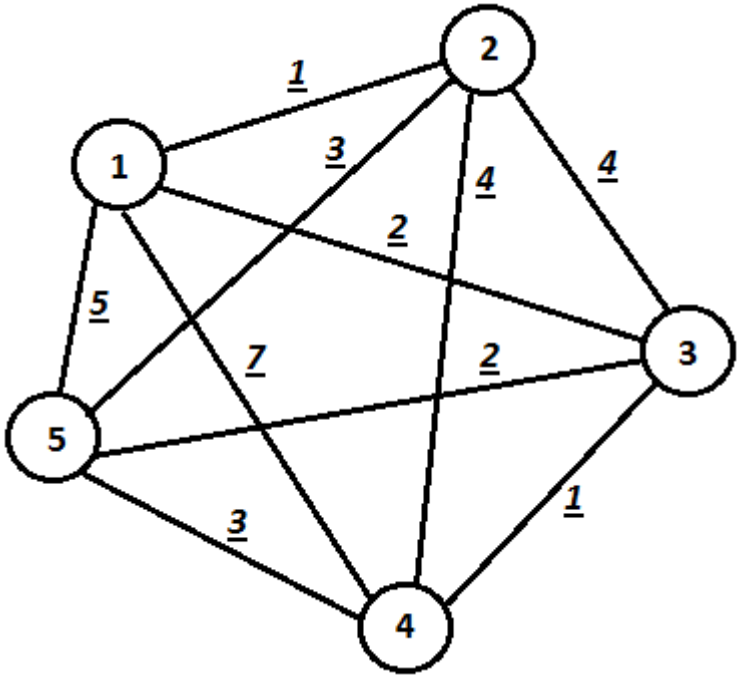


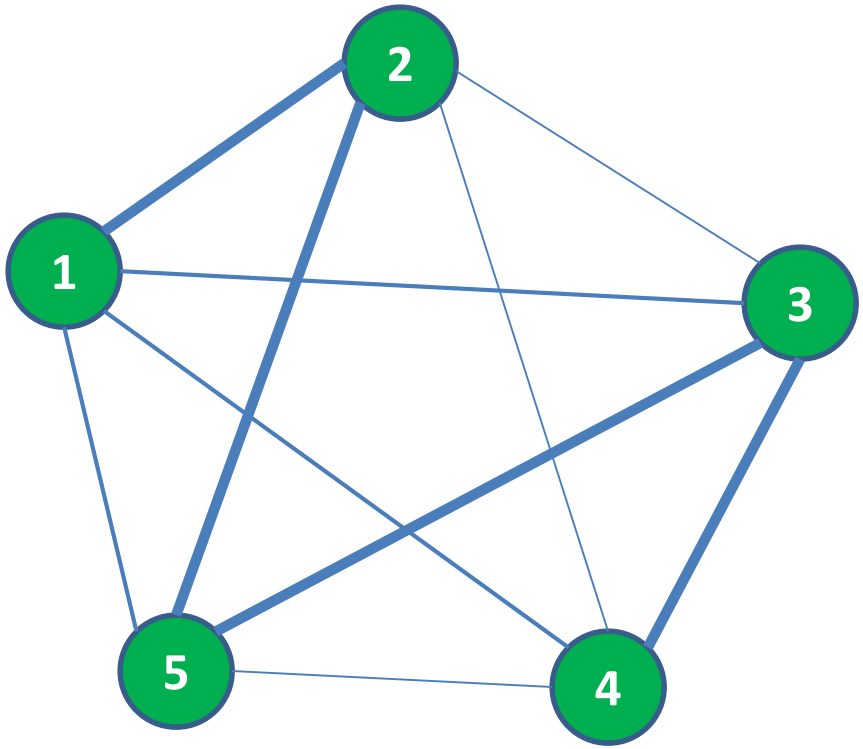




TOUR={(1,2), (2,5), (5,3), (3,4)}

COST=1+3+2+1

# Minh họa …



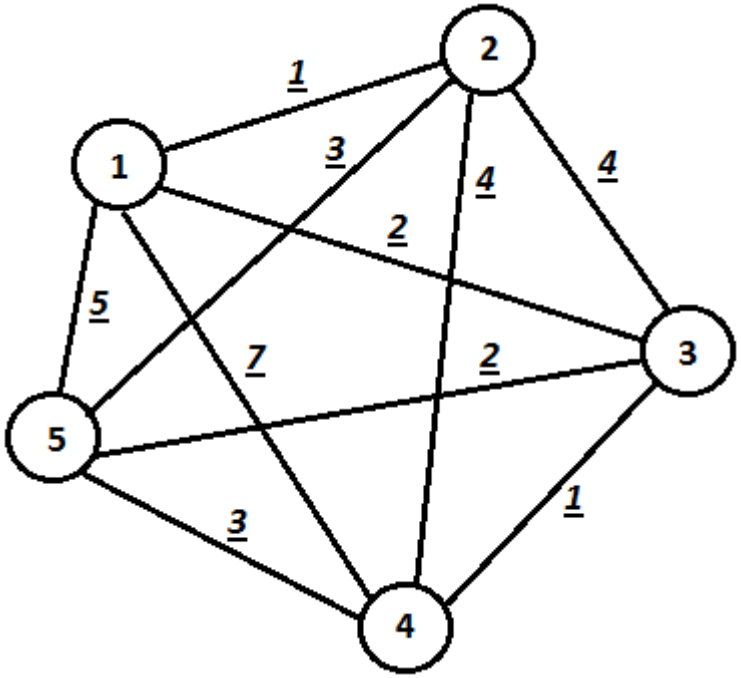



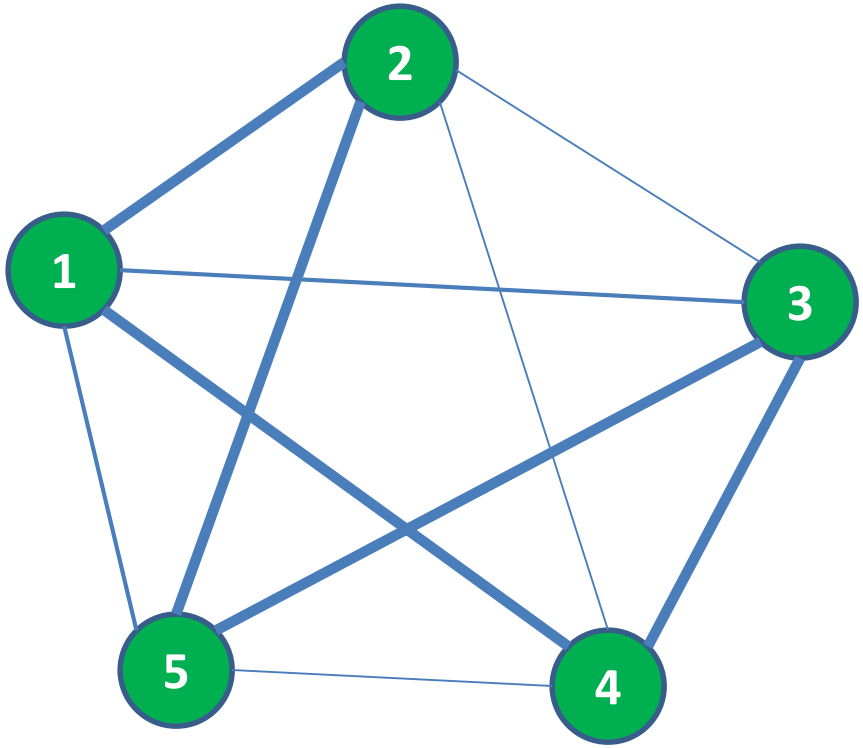

TOUR={(1,2), (2,5), (5,3), (3,4)}

COST=1+3+2+1=7

# Minh họa …

- Trở về đỉnh đầu




TOUR={(1,2), (2,5), (5,3), (3,4), (4,1)}

COST=1+3+2+1=7+7

```
int GTS (mat a, int n, int TOUR[max], int Ddau)
{
        int     v,      //Dinh dang xet
                k,      //Duyet qua n dinh de chon
                w;      //Dinh duoc chon trong moi buoc
        int     mini;   //Chon min cac canh(cung) trong moi buoc
        int COST;       //Trong so nho nhat cua chu trinh
        int daxet[max];         //Danh dau cac dinh da duoc su dung
        for(k = 1; k <= n; k++)
                daxet[k] = 0;  //Chua dinh nao duoc xet
        COST = 0;       //Luc dau, gia tri COST == 0

        int i;    // Bien dem, dem tim du n dinh thi dung
        v = Ddau;       //Chon dinh xuat phat la 1
        i = 1;
        TOUR[i] = v;  //Dua v vao chu trinh
        daxet[v] = 1;  //Dinh v da duoc xet
```

```
int GTS (mat a, int n, int TOUR[max], int Ddau)
{
    int     v,      //Dinh dang xet
            k,      //Duyet qua n dinh de chon
            w;      //Dinh duoc chon trong moi buoc
    int     mini;   //Chon min cac canh(cung) trong moi buoc
    int COST;       //Trong so nho nhat cua chu trinh
    int daxet[max];         //Danh dau cac dinh da duoc su dung
    for(k = 1; k <= n; k++)
            daxet[k] = 0;  //Chua dinh nao duoc xet
    COST = 0;       //Luc dau, gia tri COST == 0

    int i;   // Bien dem, dem tim du n dinh thi dung
    v = Ddau;       //Chon dinh xuat phat la 1
    i = 1;
    TOUR[i] = v; //Dua v vao chu trinh
    daxet[v] = 1;  //Dinh v da duoc xet
    while(i < n)
    {
            mini = VC;
            for (k = 1; k <= n; k++)
                    if(!daxet[k])
                            if(mini > a[v][k])
                            {
                                    mini = a[v][k];
                                    w = k;
                            }
            v = w;
            i++;
            TOUR[i] = v;
            daxet[v] = 1;
            COST += mini;
    }
    COST += a[v][Ddau];
    return COST;
```

# Độ phức tạp

```
while(i < n)
{
        mini = VC;
        for (k = 1; k <= n; k++)
                if(!daxet[k])
                        if(mini > a[v][k])
                        {
                                mini = a[v][k];
                                w = k;
                        }
        v = w;
        i++;
        TOUR[i] = v;
        daxet[v] = 1;
        COST += mini;
}
COST += a[v][Ddau];
return COST;
```

$T(n) = O(n^2)$

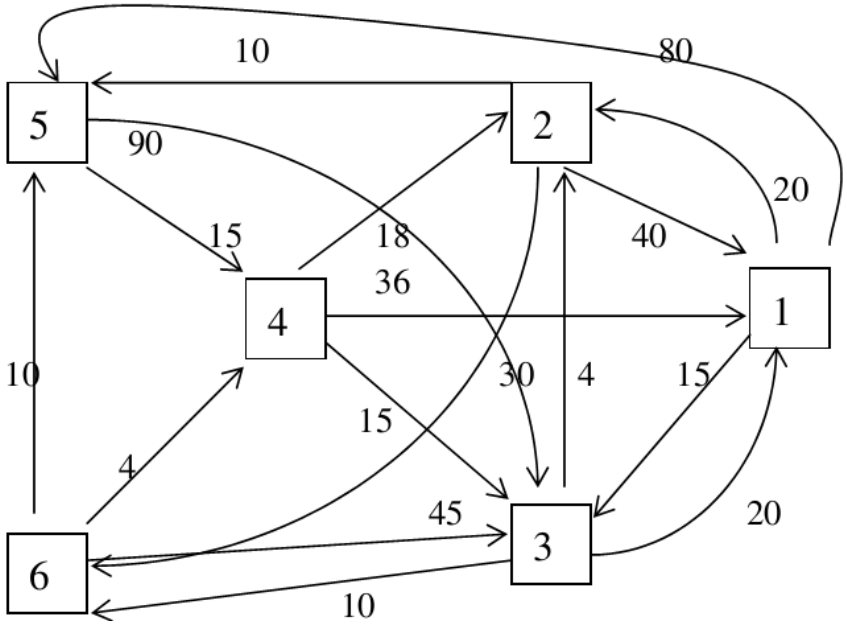

Biên soạn: Hà Đại Dương, duonghd@mta.edu.vn

# Nội dung

1. Lược đồ chung
2. Bài toán cái túi
3. Bài toán người du lịch
4. **Đường đi ngắn nhất**
5. Cây bao trùm nhỏ nhất
6. Bài toán tô màu
7. Bài toán các khoảng không giao nhau

# Bài toán

- Đồ thị G=(V,E)
  - Đơn đồ thị liên thông (vô hướng hoặc có hướng)
  - Có trọng số.
  - V: Tập đỉnh
  - E: Tập cạnh
- Tìm đường đi ngắn nhất từ $s_0 \in V$ đến tất cả các đỉnh còn lại.



# Thuật toán Dijkstra

- Ý tưởng (tham lam): Có đồ thị G=(V,E), $s_0$.
  - L(v): độ dài đường đi ngắn nhất từ $s_0$ đến đỉnh v (gọi là nhãn của v).
  - Gọi S là tập đỉnh đã xét.
  - Khởi tạo: S = {$s_0$}, L($s_0$) =0, L(v)=$\infty$ $\forall v \in V \backslash S$
  - Tại mỗi bước lặp:
    - **Cập nhập lại nhãn** các đỉnh thuộc V\S (tập V trừ tập S)
    - **Tìm đỉnh thuộc tập V\S có nhãn nhỏ nhấ**t (tham lam) kề với S để đưa vào S.

# Cập nhật nhãn L(v)

- Khởi tạo: $S = \{s_0\}$, $L(s_0) = 0$, $L(v) = \infty \ \forall v \in V \setminus S$

Với $\forall v \in V \setminus S$:

Với $\forall s \in S$: $L(v) = \min(L(v), L(s) + m(s,v))$

Trong đó **m(s,v)** là độ dài đường đi từ **s** với **v**

- Vì chỉ có L(s*) với s* là đỉnh vừa duyệt xong ở bước trước là có thay đổi về giá trị nên việc tính lại L(v) chỉ có ý nghĩa với các đỉnh kề với s*

Với $\forall v \in V \setminus S$ kề với s*:

$L(v) = \min(L(v), L(s^*) + m(s^*,v))$

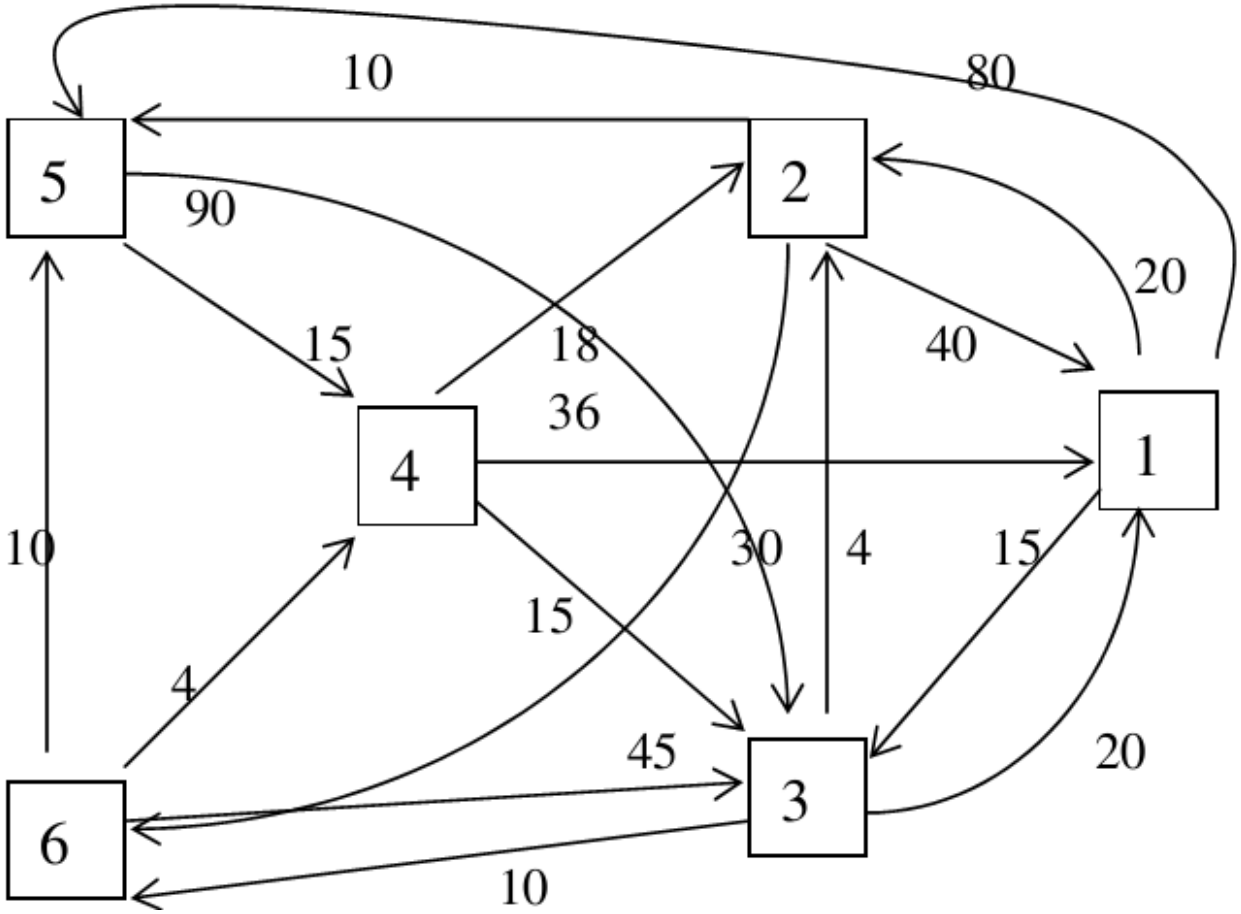
# Tìm đỉnh có nhãn nhỏ nhất s*

- Đỉnh có nhãn nhỏ nhất s*:
  - Kề với 1 trong các đỉnh $\in$ S

    Và
  - L(s*) = min(L(v): $\forall v \in V \setminus S$)

$s_0=1$

# Minh họa



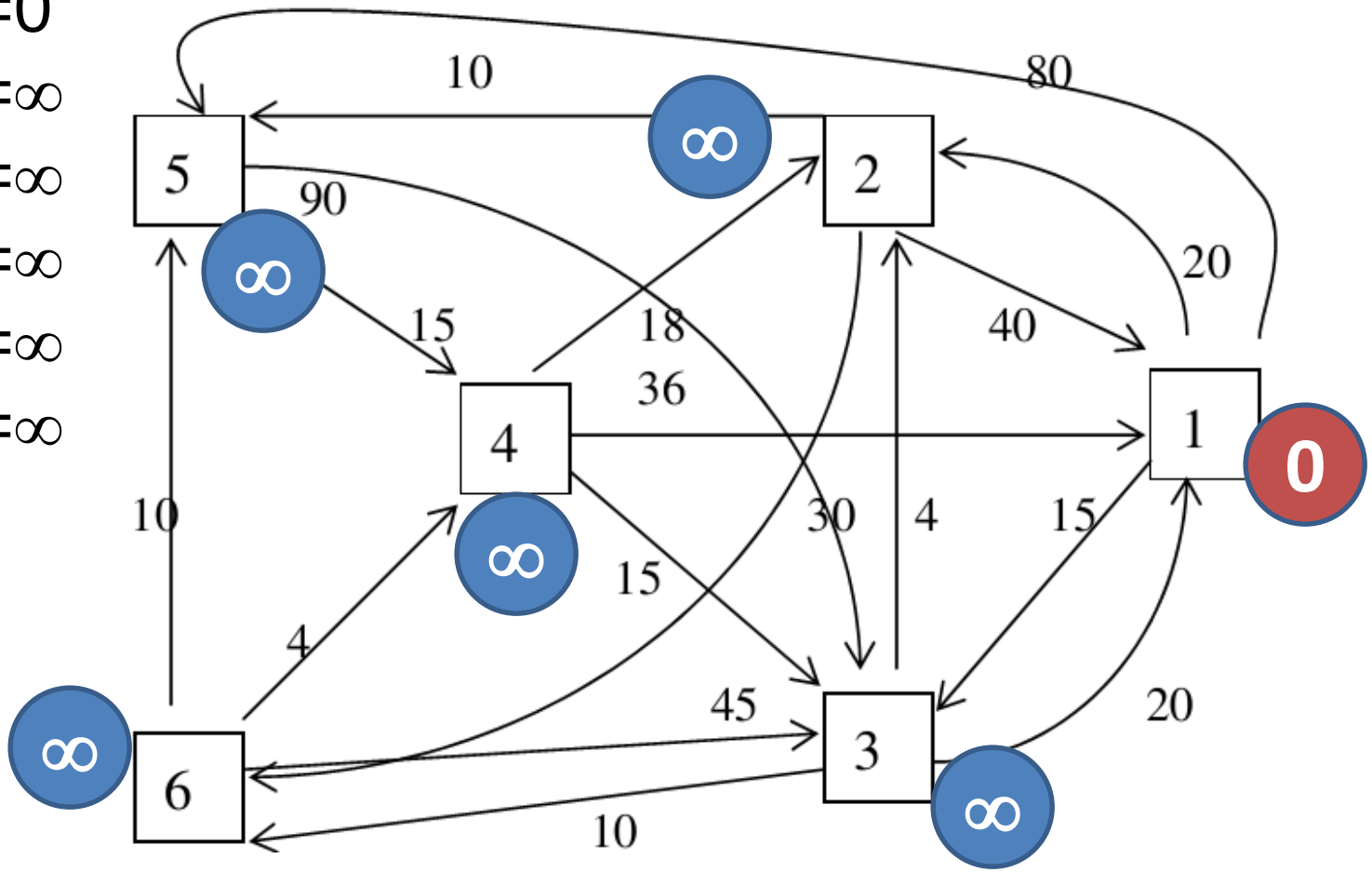

V={1,2,3,4,5,6}


Biên soạn: Hà Đại Dương, duonghd@mta.edu.vn

$s_0=1$

# Khởi tạo

S={1}
L(1)=0
L(2)=∞
L(3)=∞
L(4)=∞
L(5)=∞
L(6)=∞
s*=1




V\S={2,3,4,5,6}

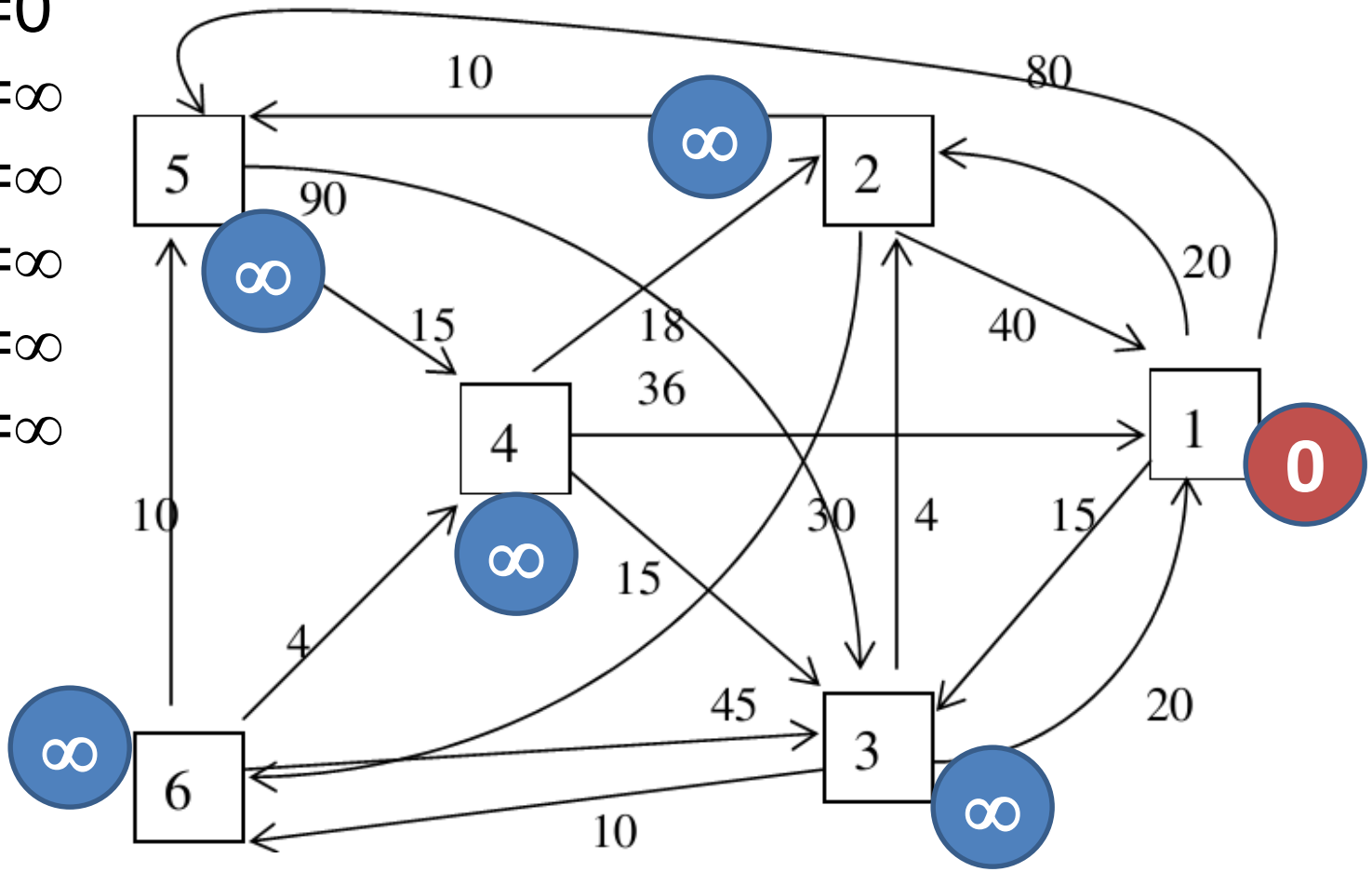

Biên soạn: Hà Đại Dương, duonghd@mta.edu.vn

$s_0$=1

# Cập nhật nhãn

$L(v) = \min(L(v), L(s^*)+m(s^*,v))$

S={1}
L(1)=0
L(2)=∞
L(3)=∞
L(4)=∞
L(5)=∞
L(6)=∞
s*=1




V\S={2,3,4,5,6}

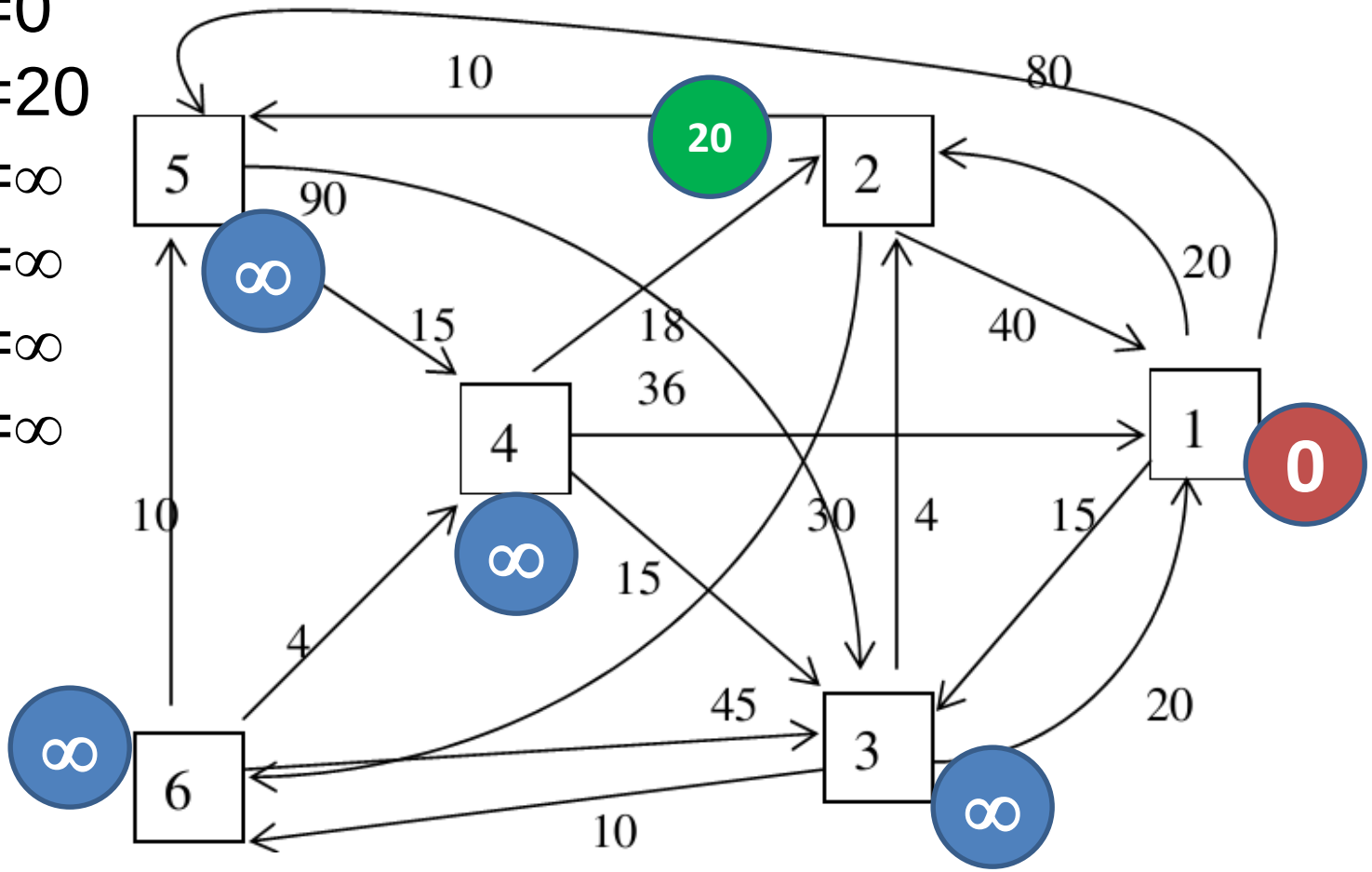

Biên soạn: Hà Đại Dương, duonghd@mta.edu.vn

$s_0$=1

# Cập nhật nhãn

$L(v) = \min(L(v), L(s^*)+m(s^*,v))$

S={1}
L(1)=0
L(2)=20
L(3)=∞
L(4)=∞
L(5)=∞
L(6)=∞
s*=1




V\S={2,3,4,5,6}

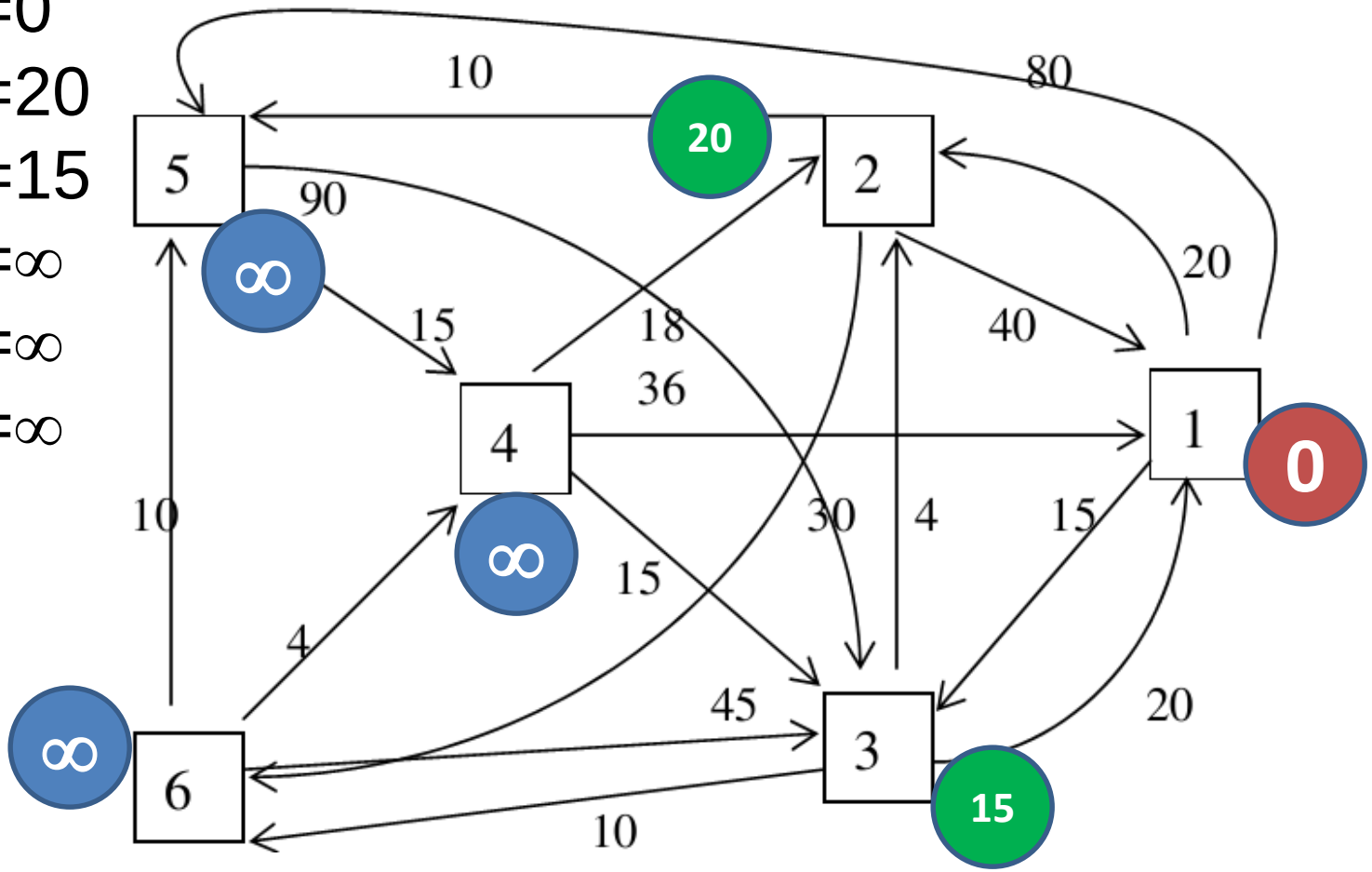

Biên soạn: Hà Đại Dương, duonghd@mta.edu.vn

$s_0$=1

# Cập nhật nhãn

$L(v) = \min(L(v), L(s^*)+m(s^*,v))$

S={1}
L(1)=0
L(2)=20
L(3)=15
L(4)=∞
L(5)=∞
L(6)=∞
s*=1




V\S={2,3,4,5,6}

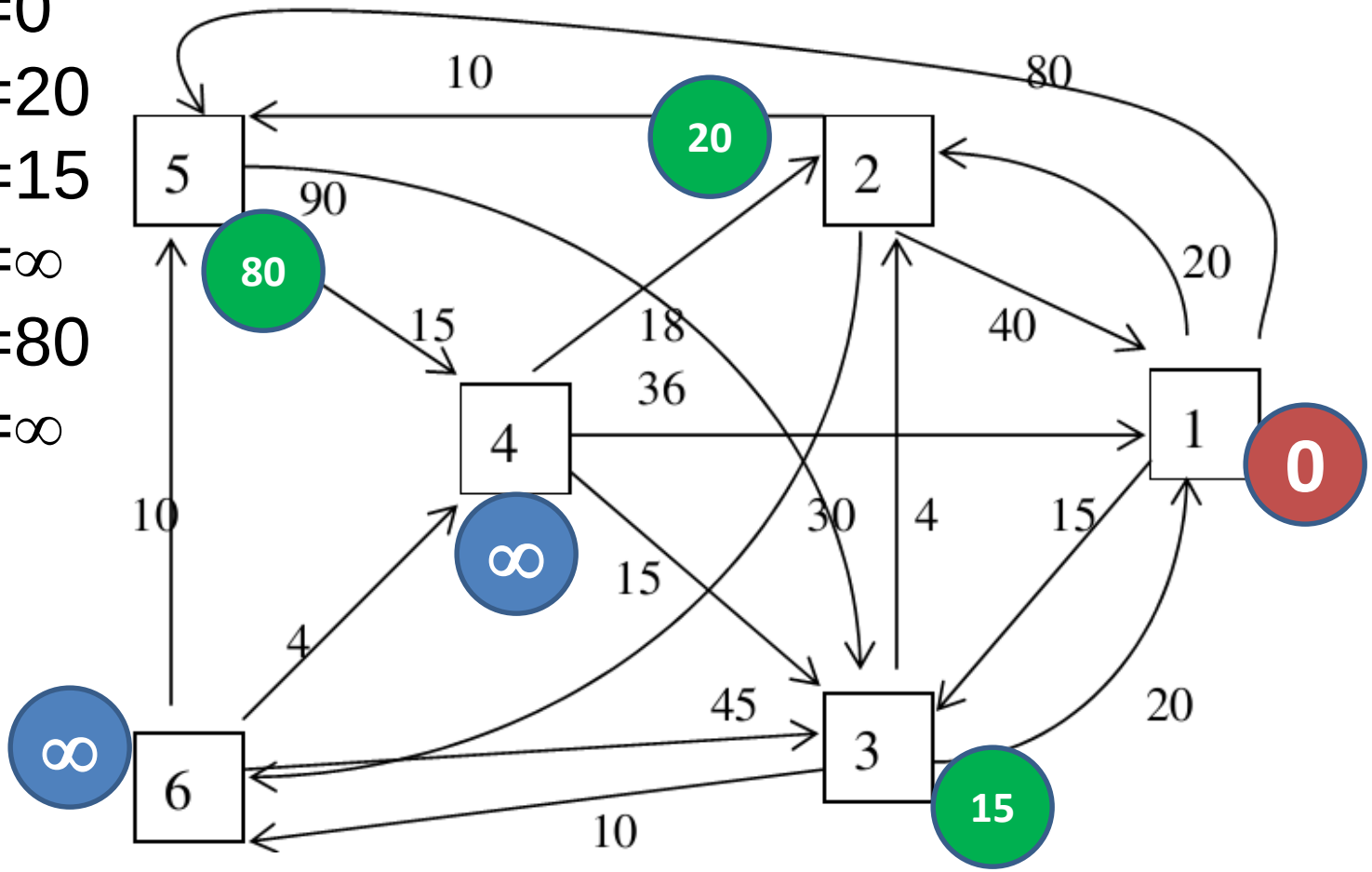

Biên soạn: Hà Đại Dương, duonghd@mta.edu.vn

$s_0$=1

# Cập nhật nhãn

$L(v) = \min(L(v), L(s^*)+m(s^*,v))$

S={1}
L(1)=0
L(2)=20
L(3)=15
L(4)=∞
L(5)=80
L(6)=∞
s*=1




V\S={2,3,4,5,6}

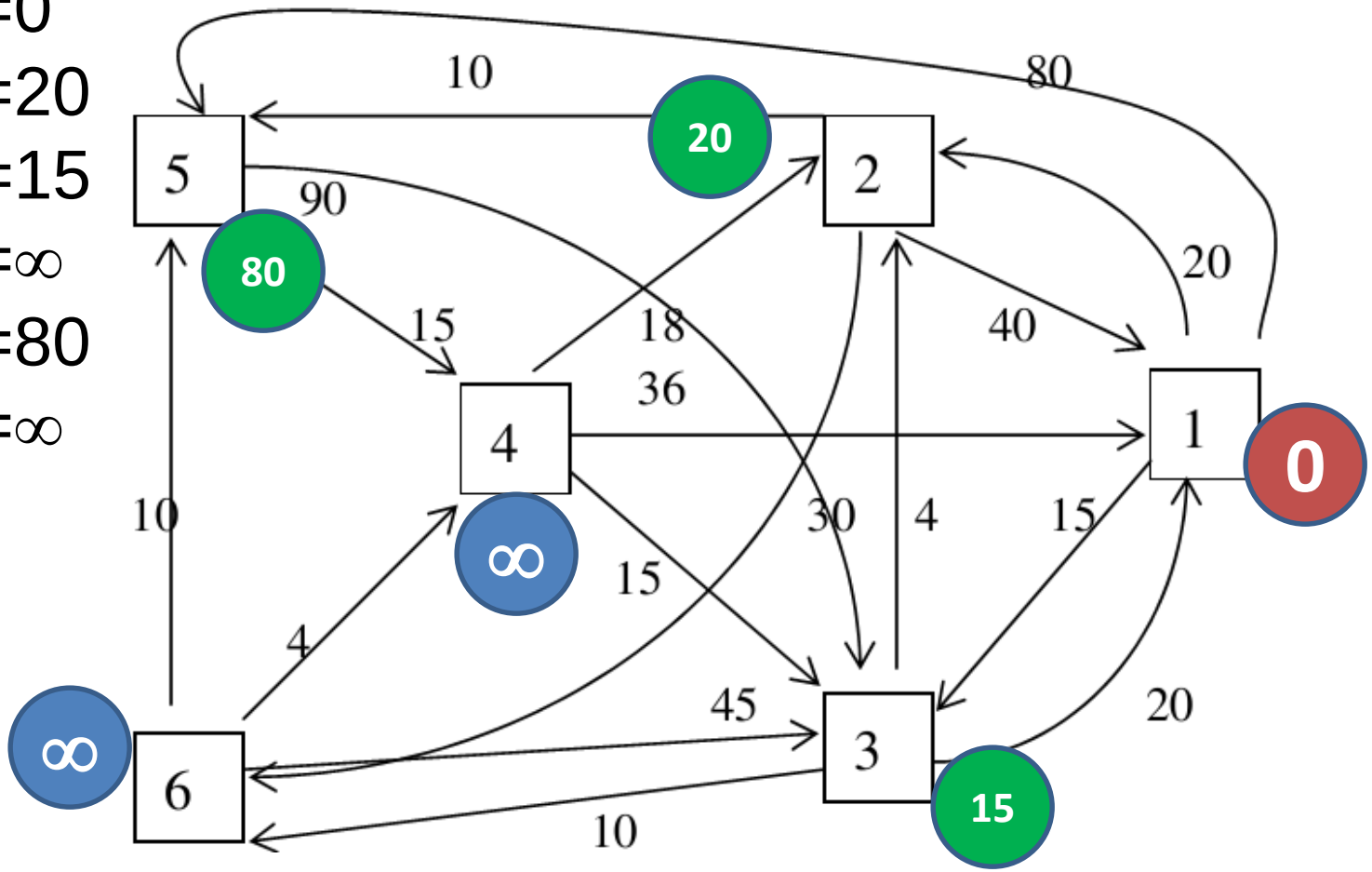

Biên soạn: Hà Đại Dương, duonghd@mta.edu.vn

$s_0$=1

# Đỉnh có nhãn nhỏ nhất s*

$L(s^*) = \min(L(v): \forall v \in V \setminus S)$

S={1}
L(1)=0
L(2)=20
L(3)=15
L(4)=∞
L(5)=80
L(6)=∞
s*=1




V\S={2,3,4,5,6}

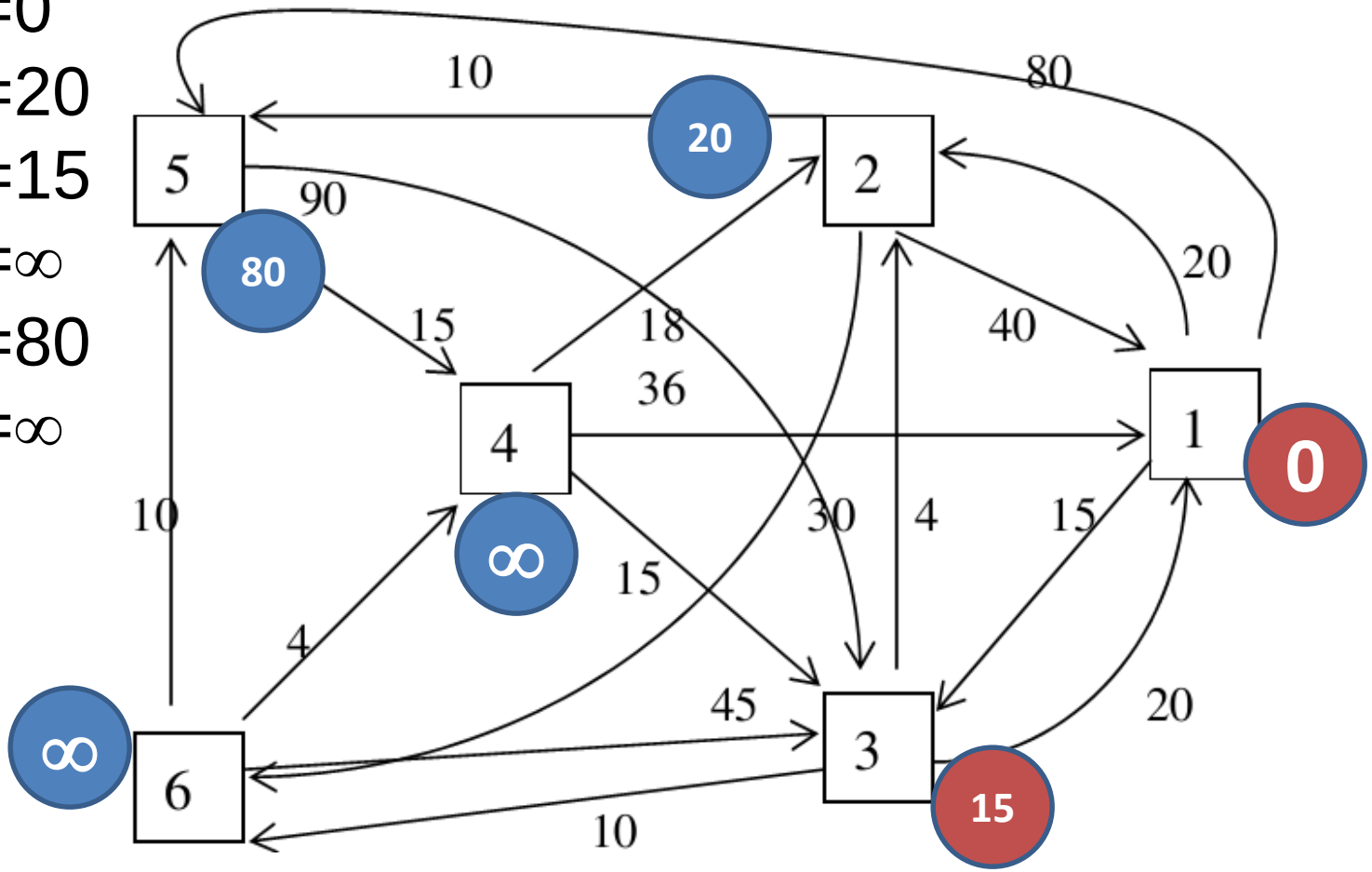

Biên soạn: Hà Đại Dương, duonghd@mta.edu.vn

$s_0=1$

# Đỉnh có nhãn nhỏ nhất s*

$L(s^*) = \min(L(v): \forall v \in V \setminus S)$

S={1}
L(1)=0
L(2)=20
L(3)=15
L(4)=∞
L(5)=80
L(6)=∞
s*=1



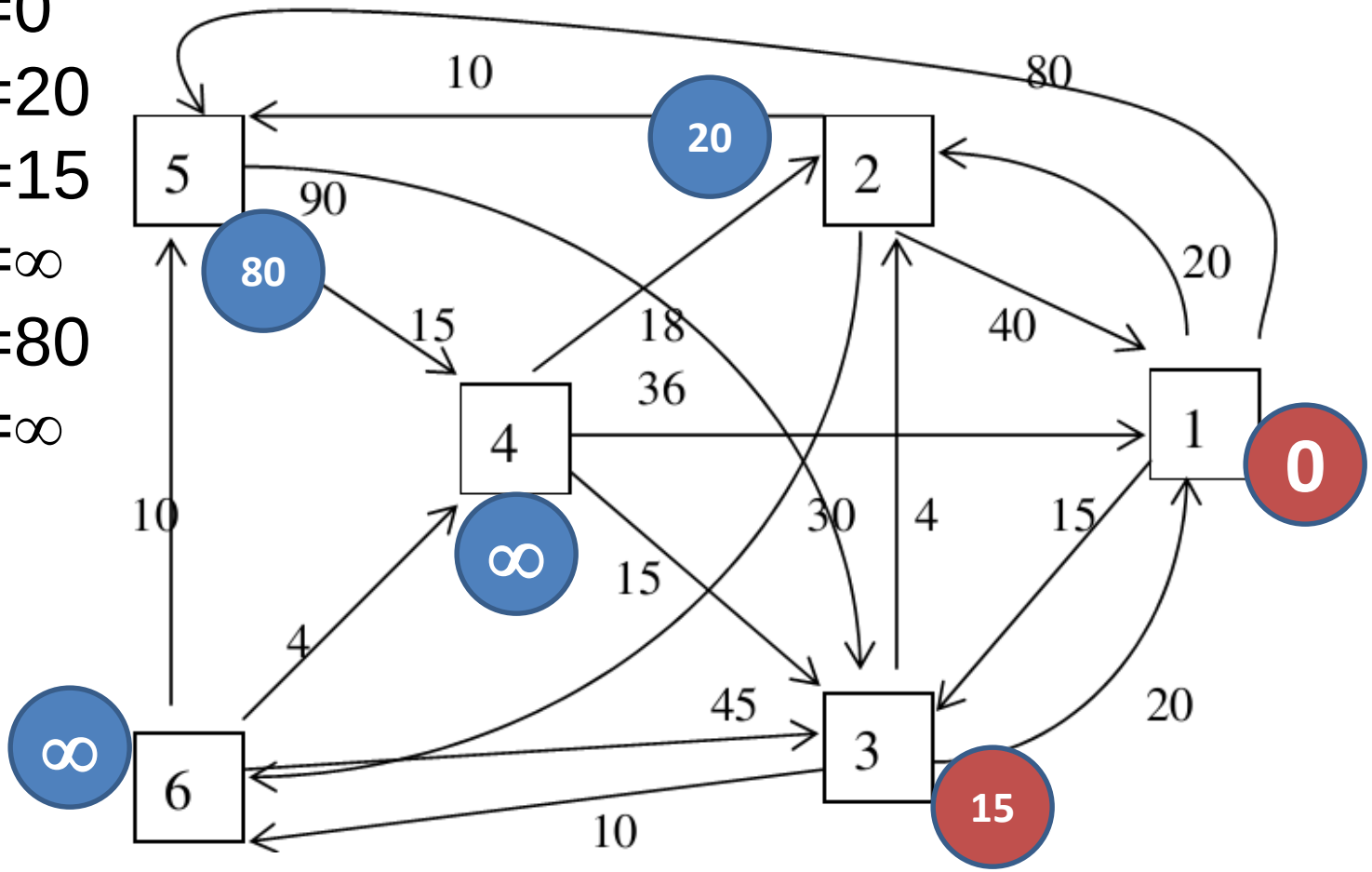

V\S={2,3,4,5,6}


Biên soạn: Hà Đại Dương, duonghd@mta.edu.vn

$s_0=1$

# Tiếp …

S={1,3}
L(1)=0
L(2)=20
L(3)=15
L(4)=∞
L(5)=80
L(6)=∞
s*=3




V\S={2,4,5,6}

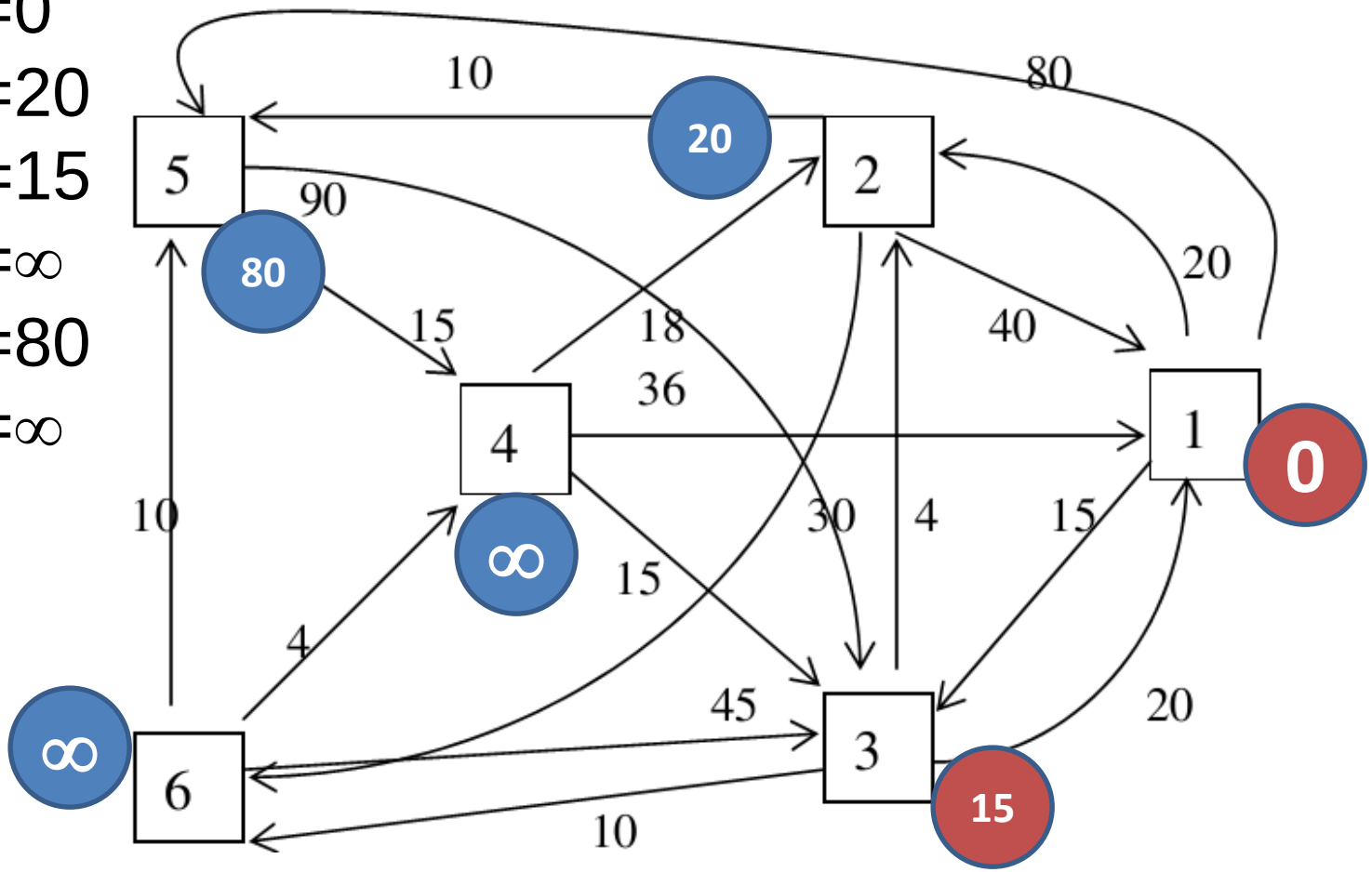

Biên soạn: Hà Đại Dương, duonghd@mta.edu.vn

$s_0$=1

# Cập nhật nhãn

S={1,3}
L(1)=0
L(2)=20
L(3)=15
L(4)=∞
L(5)=80
L(6)=∞
s*=3

$L(v) = \min(L(v), L(s^*)+m(s^*,v))$




V\S={2,4,5,6}

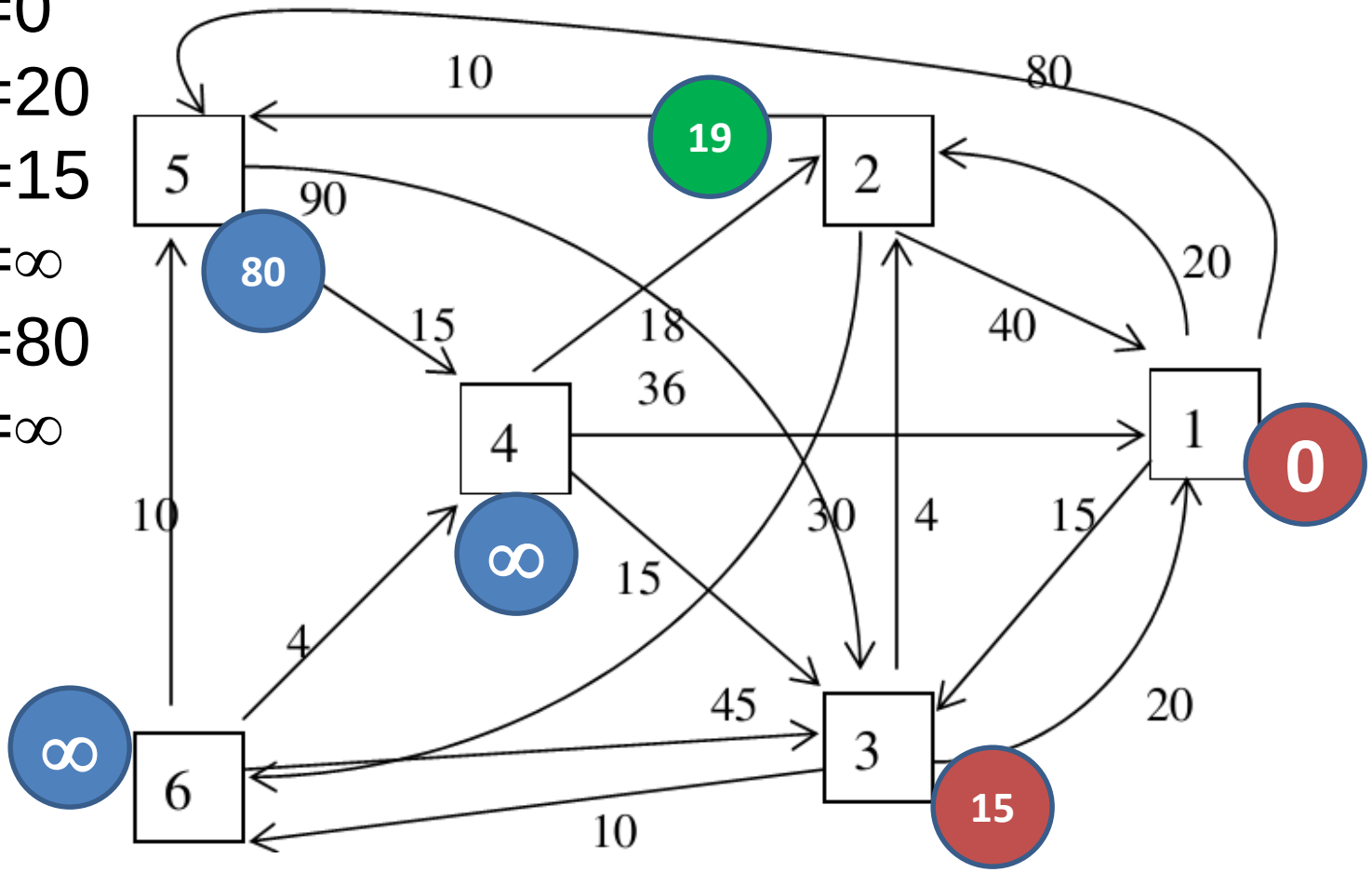

Biên soạn: Hà Đại Dương, duonghd@mta.edu.vn

$s_0$=1

# Cập nhật nhãn

S={1,3}
L(1)=0
L(2)=20
L(3)=15
L(4)=∞
L(5)=80
L(6)=∞
s*=3

L(v) = min(L(v),L(s*)+m(s*,v))



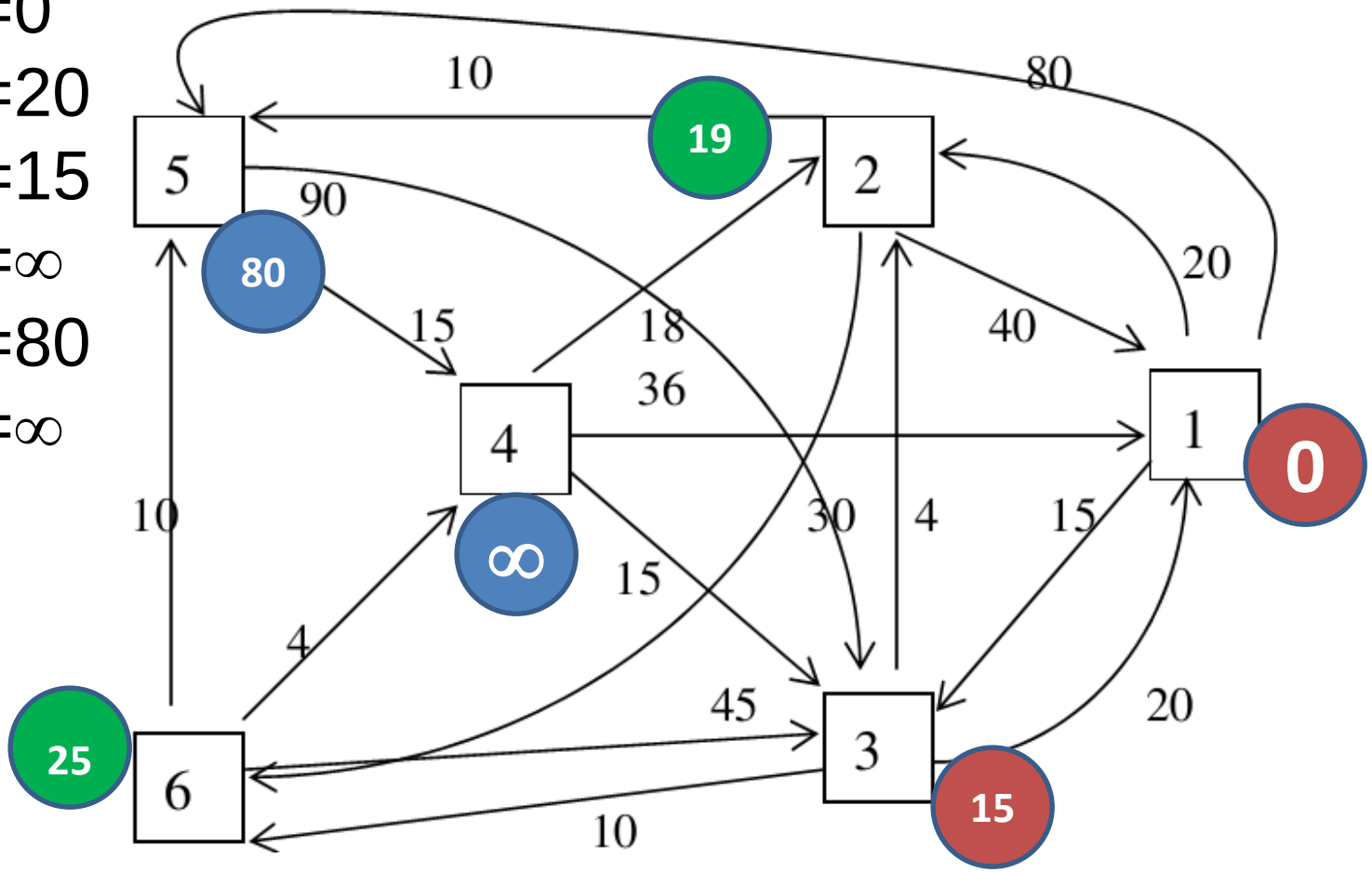

V\S={2,4,5,6}

$s_0$=1

# Cập nhật nhãn

S={1,3}
L(1)=0
L(2)=20
L(3)=15
L(4)=∞
L(5)=80
L(6)=∞
s*=3

$L(v) = \min(L(v), L(s^*)+m(s^*,v))$



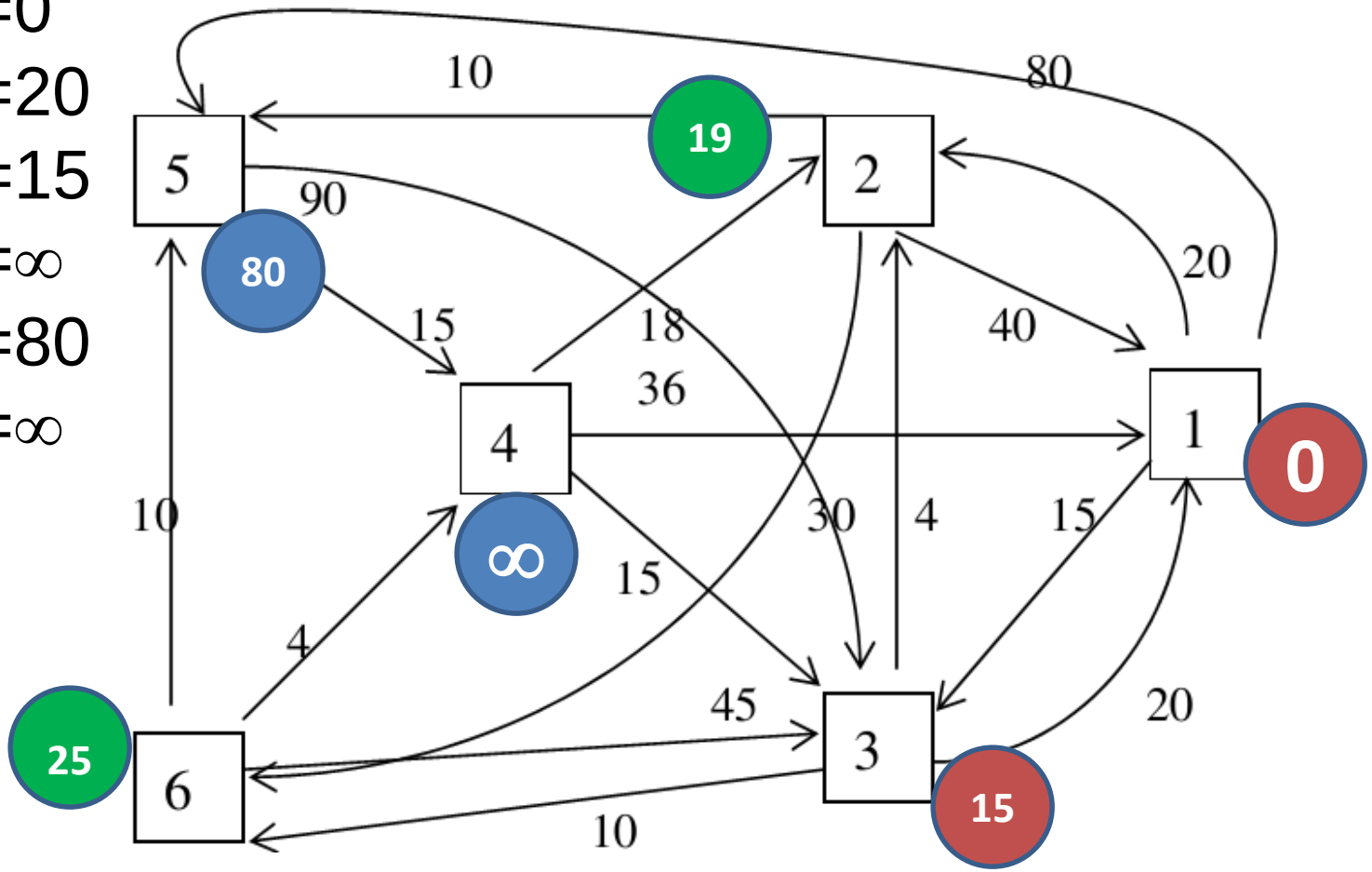

V\S={2,4,5,6}


Biên soạn: Hà Đại Dương, duonghd@mta.edu.vn

$s_0=1$

# Tiếp tục …

S={1,3}
L(1)=0
L(2)=20
L(3)=15
L(4)=∞
L(5)=80
L(6)=∞
s*=3




V\S={2,4,5,6}


Biên soạn: Hà Đại Dương, duonghd@mta.edu.vn

$s_0$=1

# Kết thúc

S={1,3,2,6,4,5}

L(1)=0

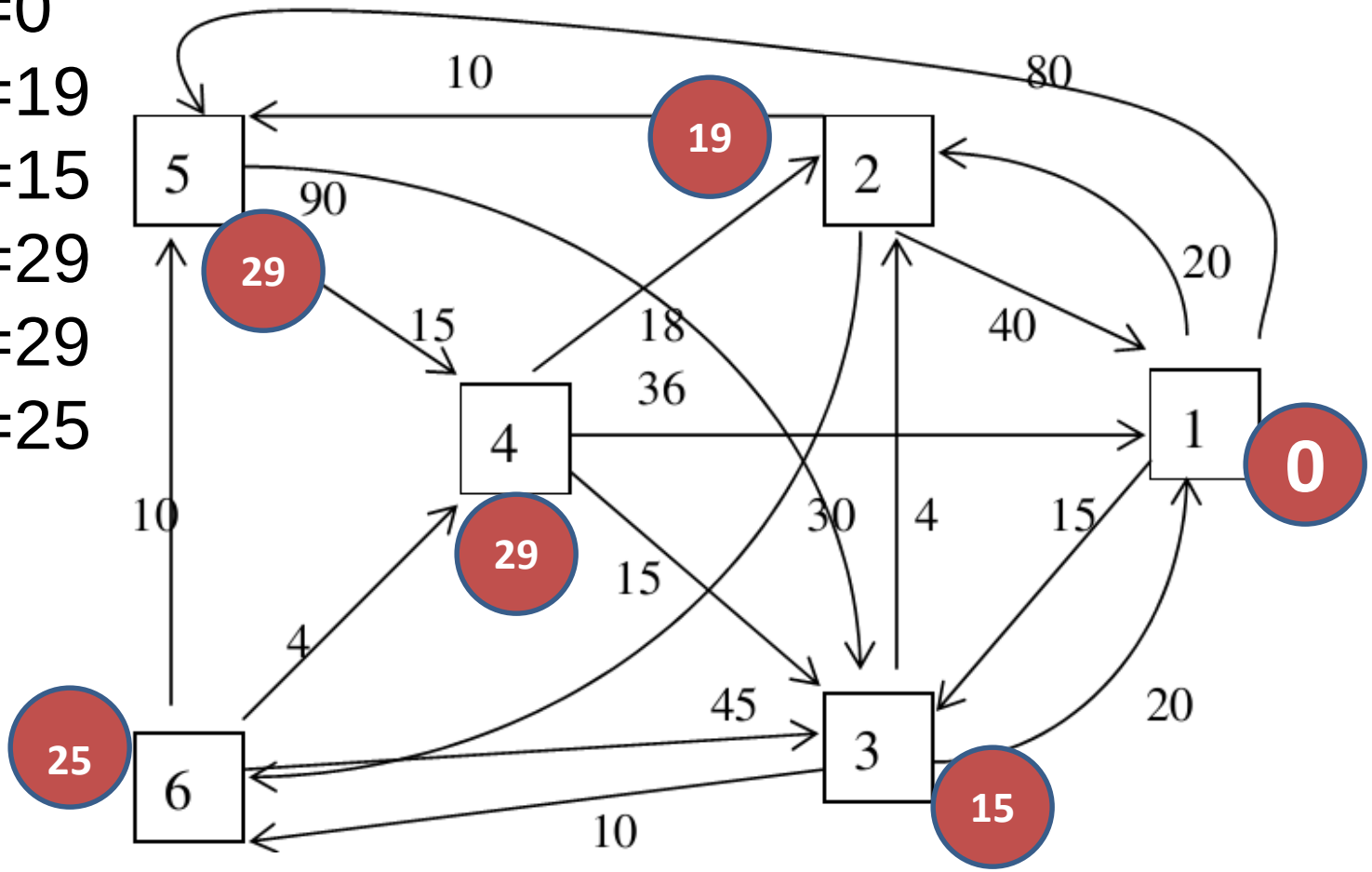
L(2)=19

L(3)=15

L(4)=29

L(5)=29

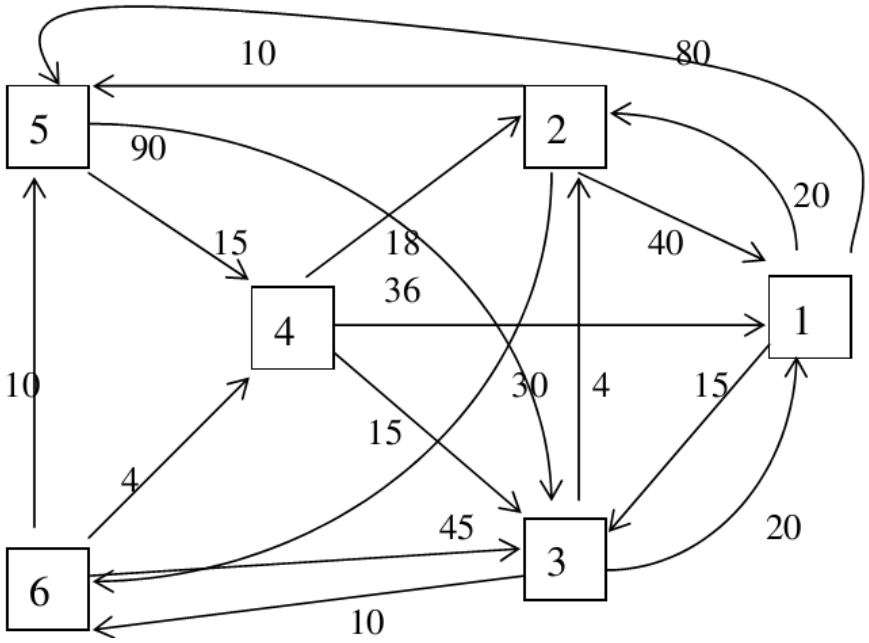
L(6)=25




V\S={}



Biên soạn: Hà Đại Dương,
duonghd@mta.edu.vn




| Bước lặp | Đường đi ngắn nhất là đường đi từ đỉnh 1 | đến đỉnh | Chiều dài của đường đi ngắn nhất từ đỉnh s (=1) đến các đỉnh khác : tsnn[] | | | | | |
|---|---|---|---|---|---|---|---|---|
| | | | 1 | 2 | 3 | 4 | 5 | 6 |
| Bước1 | 1→ 3 | 3 | - | 20 | 15 | ∞ | 80 | ∞ |
| Bước2 | 1→3→ 2 | 2 | - | 19 | - | | | 25 |
| Bước3 | 1→3→ 6 | 6 | - | - | - | | 29 | 25 |
| Bước4 | 1→3→ 6→4 | 4 | - | - | - | 29 | | - |
| Bước5 | 1→3→ 2→5 | 5 | - | - | - | - | 29 | - |

# Cài đặt

- Biểu diễn G qua ma trận trọng số cạnh

$$a = (a_{uv})_{nxn};$$

$$a_{uv} = \begin{cases} trọng\ số\ của\ (u,v); (u,v) \in E; \\ \infty; (u,v) \notin E; \end{cases}$$

- Mảng L[i]  nhãn đỉnh I
- Mảng Daxet[i]: 0 i chưa xét, 1 i đã xét
- Mảng Ddnn[i]: Giá trị của nó là đỉnh trước trong đường đi ngắn nhất đến i

# Cài đặt …

```
void dijkstra( int s)
{
        int Ddnn[max]; // Chứa đường đi ngắn nhất từ s đến đỉnh t tại mỗi bước
        int i,k,Dht,Min;
        int Daxet[max]; //Đánh dấu các đỉnh đã đưa vào S
        int L[max];
        for ( i = 1; i <= n; i++)
        {
                Daxet[i] = 0;
                L[i] = VC;
        }
        //Dua dinh s vao tap dinh S da xet
        Daxet[s] = 1;
        L[s] = 0;
        Dht = s;
        int h = 1; //đếm mỗi bước : cho đủ n-1 bước
```

```
while (h<= n-1)
{
        Min = VC;
        for ( i = 1; i <= n; i++)
                if(!Daxet[i])
                {
                        if ( L[dht] + a[dht][i] < L[i] ) //Tính lại nhãn
                        {
                                L[i] = L[dht] + a[dht][i] ;
                                Ddnn[i] = dht;
                                //gan dinh hien tai la dinh truoc dinh i tren lo trinh
                        }
                        if(L[i] < Min) // Chon đỉnh k
                        {
                                Min = L[i];
                                k = i;
                        }
                }
                // Tại bước h : tìm được đường đi ngắn nhất từ s đến k : Ddnn[]
                xuatdd(s,k,Ddnn);
                cout<<"\nTrong so : "<<L[k];
                dht = k;// Khoi dong lai Dht
                Daxet[dht] = 1;         //Dua nut k vao tap nut da xet
                h++;
}
```

# Cài đặt …

```
}
//*****************************
void xuatdd(int s, int k, int Ddnn[max])
{
        int i;
        cout<<"\nDuong di ngan nhat tu "<<s<<" den "<<k<<" la : ";
        i = k;
        while(i != s)
        {
                cout<<i<<"<--";
                i = Ddnn[i];
        }
        cout<<s;
}
```

# Kết quả thuật toán

- Thuật toán Dijkstra cho kết quả tối ưu
- $T(n) = O(n^2)$

# Bài tập

1. Thực hiện từng bước bài toán người du lịch theo giải thuật tham lam với các dữ liệu sau: Bắt đầu từ đỉnh 1, ma trân chi phí được mô tả như sau:

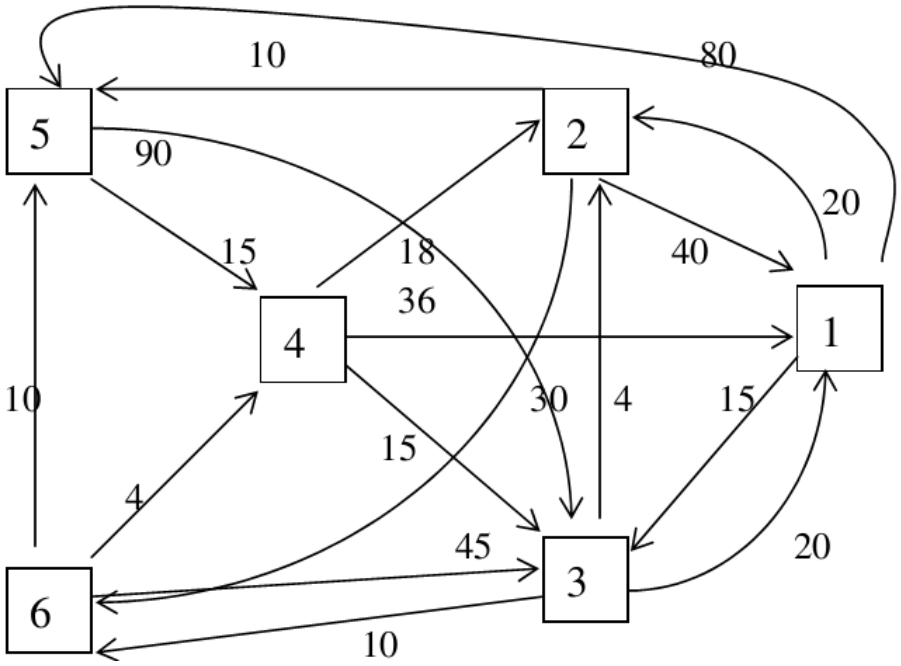
| | | | | |
|---|---|---|---|---|
| 0 | 3 | 5 | 2 | 6 |
| 3 | 0 | 6 | 7 | 3 |
| 5 | 6 | 0 | 5 | 4 |
| 2 | 7 | 5 | 0 | 1 |
| 6 | 3 | 4 | 1 | 0 |

| | | | | |
|---|---|---|---|---|
| 0 | 1 | 6 | 3 | 2 |
| 1 | 0 | 5 | 7 | 1 |
| 6 | 5 | 0 | 6 | 3 |
| 3 | 7 | 6 | 0 | 2 |
| 2 | 1 | 3 | 2 | 0 |

| | | | | |
|---|---|---|---|---|
| 0 | 7 | 2 | 6 | 1 |
| 7 | 0 | 5 | 4 | 7 |
| 2 | 5 | 0 | 1 | 3 |
| 6 | 4 | 1 | 0 | 5 |
| 1 | 7 | 3 | 5 | 0 |

# Bài tập

2. Thực hiện từng bước thuật toán Dijstra bắt đầu từ đỉnh 2, 3, 4 trên đồ thị sau



# Bài tập

3. Đề xuất giải thuật tham lam giải bài toán trả tiền máy ATM?
4. Cài đặt thuật toán người du lịch. Đánh giá độ phức tạp bằng thực nghiệm và so sánh với lý thuyết
5. Cài đặt thuật toán Dijkstra. Đánh giá độ phức tạp bằng thực nghiệm và so sánh với lý thuyết