

Greedy Algorithm

Truong Ngoc Tuan

Nội dung

- ❑ Lược đồ chung
- ❑ Bài toán cái túi - 2
- ❑ Bài toán người đi du lịch
- ❑ Bài toán sắp xếp phòng họp

Nội dung

- ❑ **Lược đồ chung**
- ❑ Bài toán cái túi - 2
- ❑ Bài toán người đi du lịch
- ❑ Bài toán sắp xếp phòng họp

Lược đồ chung

□ Bài toán tối ưu

- Phương pháp Tham lam thường dùng cho các bài toán tối ưu tổ hợp
- Bài toán tối ưu tổ hợp có dạng chung
$$\min\{f(x) : x \in D\}$$
- Trong đó D là tập hữu hạn các điểm rời rạc nào đó thuộc không gian R^n

Lược đồ chung

□ 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 là bội của 10.000). Hãy tìm phương án trả tiền sao cho số tờ tiền phải trả là ít nhất.

Lược đồ chung

□ Ví dụ

- Gọi $x=(x_1, x_2, x_3, x_4)$ là một phương án trả tiền với x_1, x_2, x_3, x_4 là số tờ tiền phải trả tương ứng với các mệnh giá 100.000, 50.000, 20.000, 10.000

- Ta có:

$$\min(f= x_1+x_2+x_3+x_4)$$

- Điều kiện:

- $n=100.000 * x_1 + 50.000 * x_2 + 20.000 * x_3 + 10.000 * x_4$
- $X_i \geq 0$ ($i=1..4$)

Lược đồ chung

□ 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, ta có thể so sánh lần lượt giá trị của f tại tất cả các phương án thuộc D , cách này gọi là “duyet vét cạn”
- Khi số phần tử của D lớn (dù hữu hạn) thì việc duyệt vét cạn vẫn gặp nhiều khó khăn

Lược đồ chung

□ Phương pháp tham lam

- Phương pháp 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
- Phương pháp này rất dễ đề xuất, và thông thường không đòi hỏi thời gian tính
- Tuy nhiên, các thuật toán dạng này thường không cho kết quả tối ưu

Lược đồ chung

□ Ý tưởng

- Từ lời giải rỗng, ở mỗi bước lựa chọn 1 phần tử từ tập ứng viên và bổ sung vào lời giải hiện có
- Hàm $\text{Solution}(S)$ nhận biết tính chấp nhận được của lời giải S

Lược đồ chung

□ Ý tưởng

- Hàm $\text{Select}(C)$ chọn từ tập C ứng của viên có triển vọng nhất để bổ sung vào lời giải hiện có
- Hàm $\text{Feasible}(S+x)$ kiểm tra tính chấp nhận được của lời giải bộ phận $S+x$

Lược đồ chung

```
void greedy() {  
    // Giả sử C là tập các ứng cử viên  
    S: =  $\emptyset$  // S là lời giải xây dựng theo tt  
    while (C  $\neq \emptyset$ ) {  
        x  $\leftarrow$  Select(C)  
        C: = C \ x  
        if (Feasible(S  $\cup$  x))  
            S: = S  $\cup$  x  
    }  
    if (Solution(S))  
        return S  
}
```

Nội dung

- ❑ Lược đồ chung
- ❑ **Bài toán cái túi - 2**
- ❑ Bài toán người đi du lịch
- ❑ Bài toán sắp xếp phòng họp

Bài toán cái túi - 2

□ Bài toán

- Có n đồ vật, đồ vật thứ i có trọng lượng w_i và giá trị v_i . $i=1, 2, \dots, n$
- Tìm cách bỏ các đồ vật này vào túi có trọng lượng là m sao cho tổng trọng lượng của các đồ vật được cho vào túi không quá m . Đồng thời tổng giá trị của chúng là lớn nhất

Bài toán cái túi - 2

□ Tổng quát

- Đặt $C = \{1, 2, \dots, n\}$ là tập chỉ số của các đồ vật
- Bài toán: tìm $I \subset C$ sao cho

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

với

$$\sum_{i \in I} w_i \leq m$$

Bài toán cái túi - 2

□ Tham lam - 1


- Ý tưởng: 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 giá trị cao đến có giá trị thấp hơn) nếu dung lượng còn lại của túi đủ chứa nó


Bài toán cái túi - 2

□ Tham lam - 1

- Số lượng đồ vật: $n = 3$
- Trọng lượng của túi: $m = 19$
- Trọng lượng của các đồ vật

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

Greedy1  $I = \{1\}$
 $V = 20$

Tối ưu  $I^* = \{2, 3\}$
 $V^* = 24$

Bài toán cái túi - 2

□ Tham lam - 2


- Ý tưởng: đồ vật có giá trị 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 thấp hơn) nếu dung lượng còn lại của túi đủ chứa nó

Bài toán cái túi - 2

□ Tham lam - 2

- Số lượng đồ vật: $n = 3$
- Trọng lượng của túi: $m = 11$
- Trọng lượng của các đồ vật

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

Greedy1  $I = \{1, 2\}$
 $V = 26$

Tối ưu  $I^* = \{3\}$
 $V^* = 28$

Bài toán cái túi - 2

□ Tham lam - 3

- Ý tưởng: đồ vật có trọng số 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 (c/w)

$$\frac{C_{i1}}{W_{i1}} \geq \frac{C_{i2}}{W_{i2}} \geq \dots \geq \frac{C_{in}}{W_{in}}$$

- Chọn đồ vật từ đầu đến cuối...

Bài toán cái túi - 2

□ Tham lam - 3

- Trường hợp 1: $m=19, V = 24$

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

- Trường hợp 2: $m=11, V = 28$

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

Nội dung

- ❑ Lược đồ chung
- ❑ Bài toán cái túi - 2
- ❑ **Bài toán người đi du lịch**
- ❑ Bài toán sắp xếp phòng họp

Bài toán người đi du lịch

□ Bài toán

- Một người đi du lịch muốn tham quan n thành phố T_1, \dots, 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 mãn yêu cầu bài toán sao cho chi phí là nhỏ nhất

Bài toán người đi du lịch

□ Ý tưởng

- Chọn thành phố gần nhất tính từ thành phố hiện tại
- Tổ chức dữ liệu: Đồ thị $G=(V, E)$
 V – tập đỉnh, E – tập cạnh

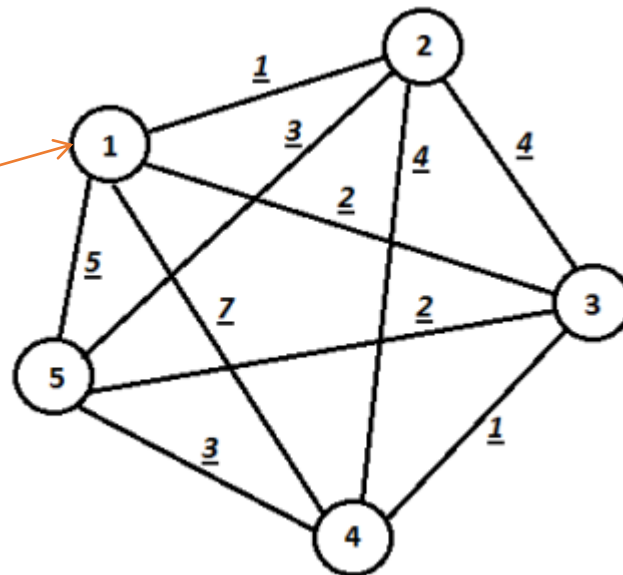
$$\begin{bmatrix} 0 & 1 & 2 & 7 & 5 \\ 1 & 0 & 4 & 4 & 3 \\ 2 & 4 & 0 & 1 & 2 \\ 7 & 4 & 1 & 0 & 3 \\ 5 & 3 & 2 & 3 & 0 \end{bmatrix}$$

Bài toán người đi du lịch

□ Minh họa

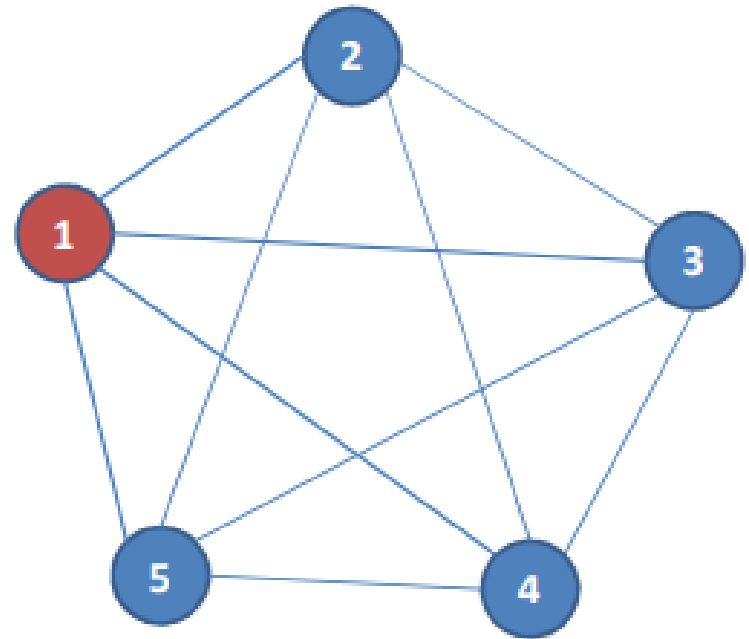
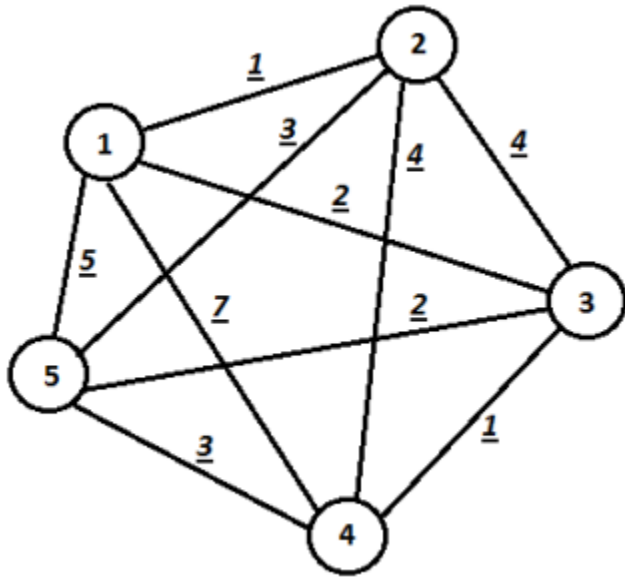
- TOUR: chứa danh sách cạnh của hành trình
- COST: chi phí theo hành trình TOUR
- u: đỉnh hiện tại
- v: đỉnh kề với u có chi phí thấp nhất

Xuất phát từ 1



Bài toán người đi du lịch

□ Minh họa

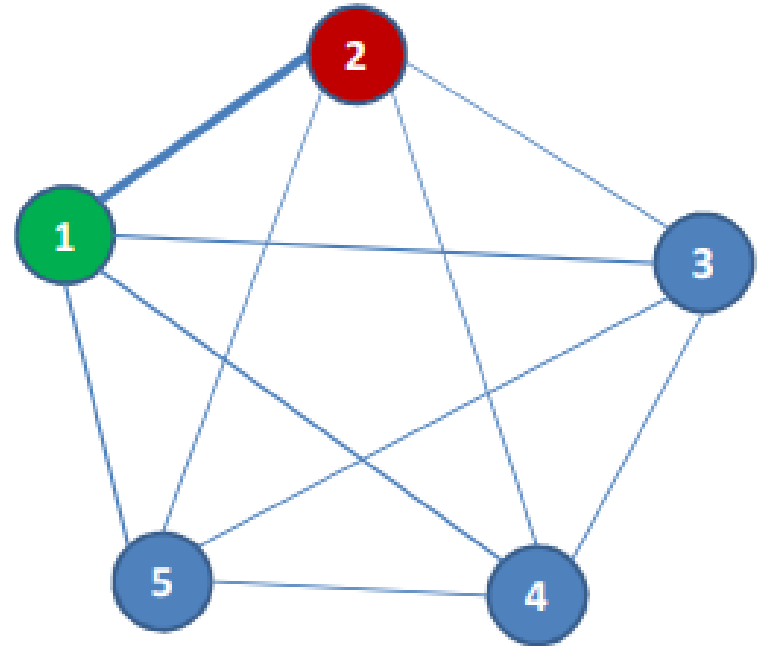
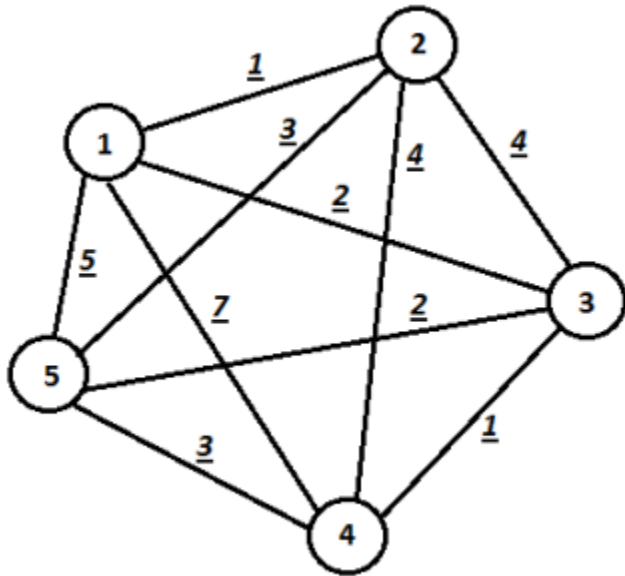


TOUR = {}

COST = 0

Bài toán người đi du lịch

□ Minh họa

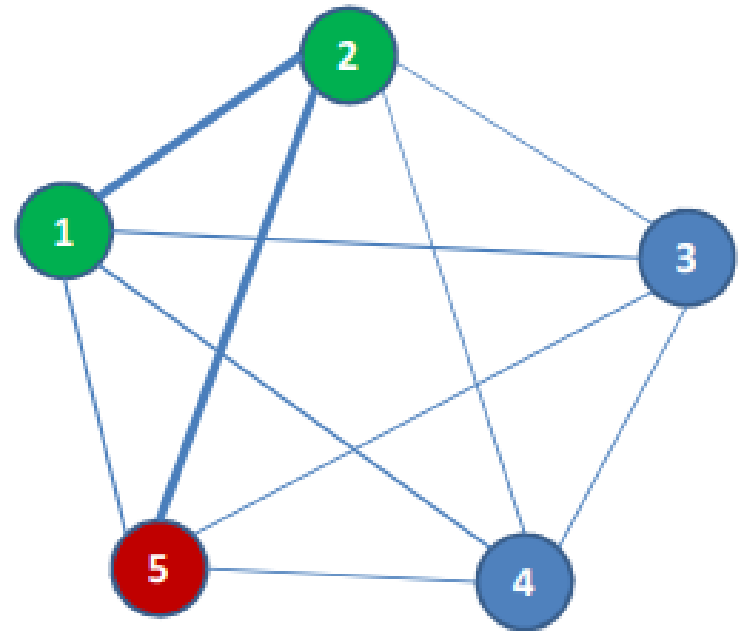
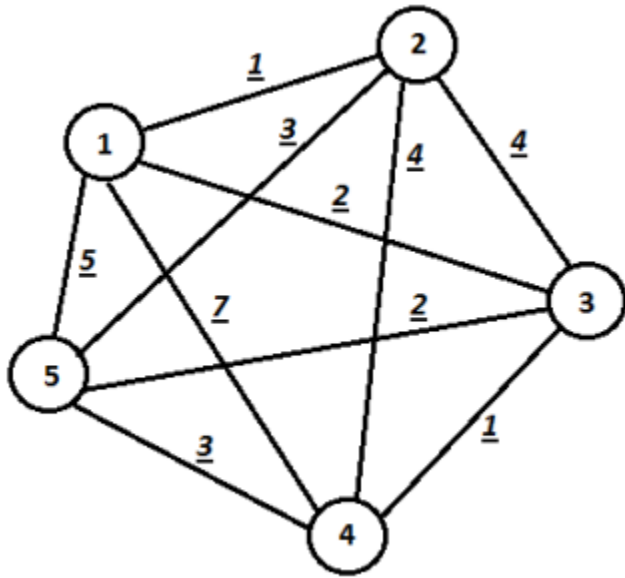


TOUR = {(1, 2)}

COST = 1

Bài toán người đi du lịch

□ Minh họa

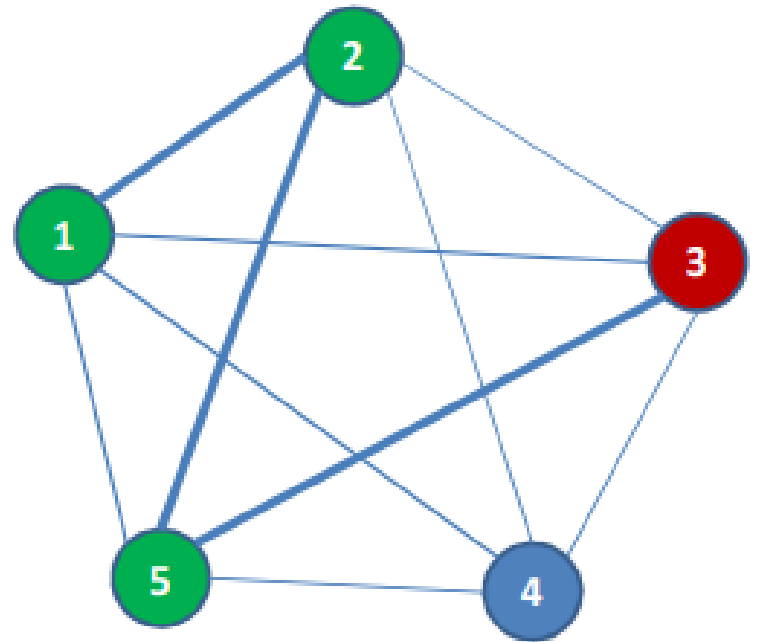
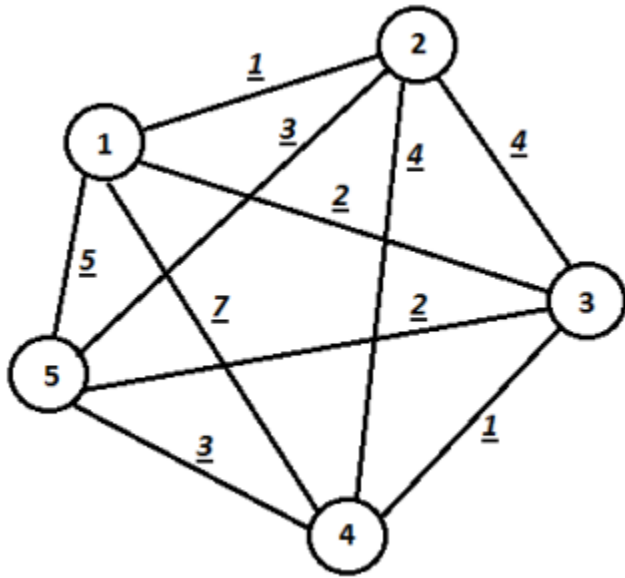


TOUR = {(1, 2), (2, 5)}

COST = 1 + 3

Bài toán người đi du lịch

□ Minh họa

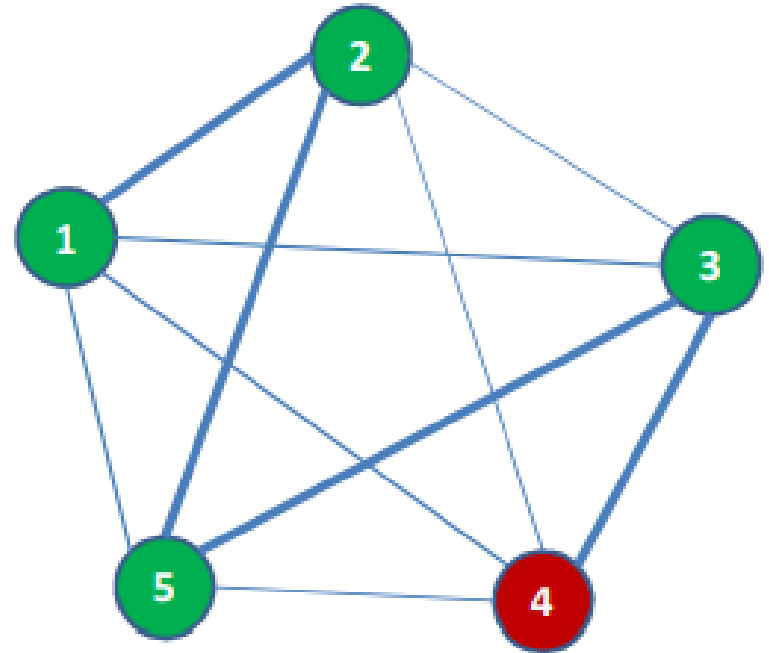
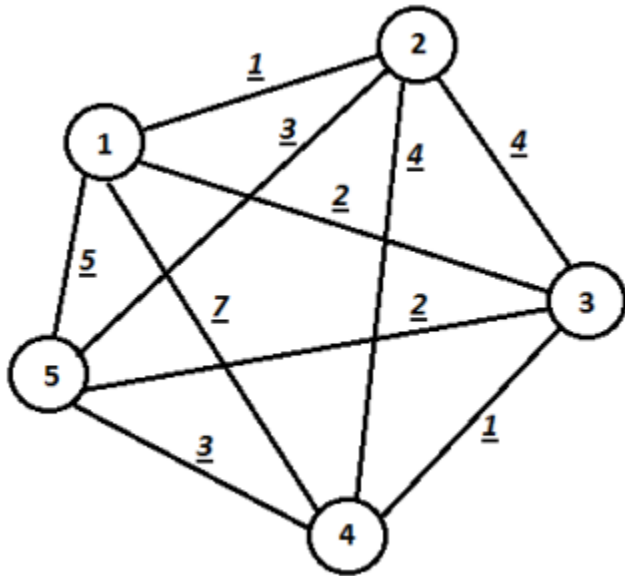


TOUR = $\{(1, 2), (2, 5), (5, 3)\}$

COST = $1 + 3 + 2$

Bài toán người đi du lịch

□ Minh họa

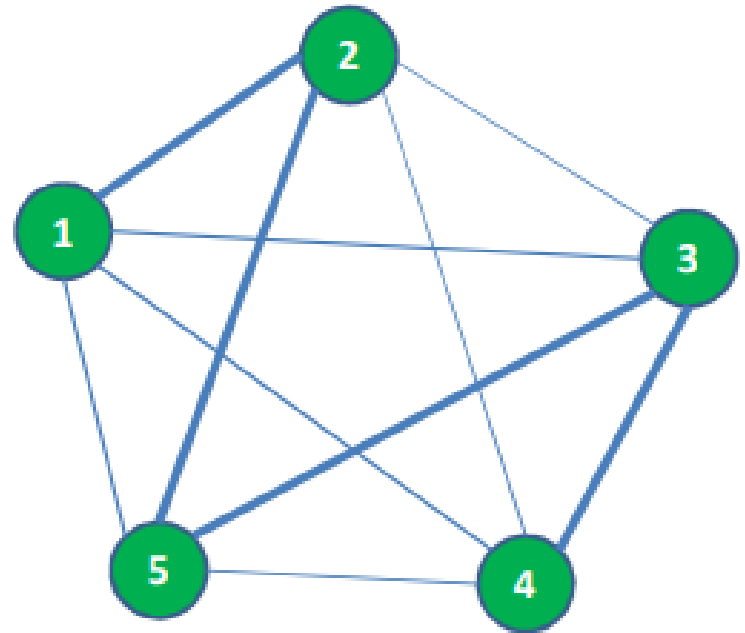
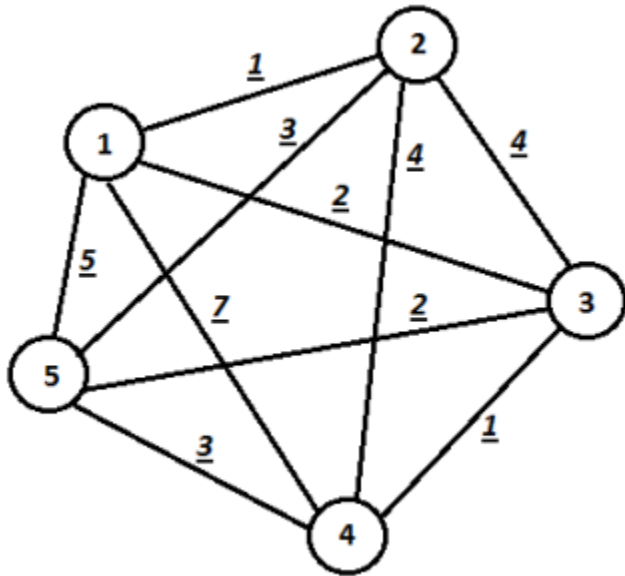


TOUR = {(1, 2), (2, 5), (5, 3), (3, 4)}

COST = 1 + 3 + 2 + 1

Bài toán người đi du lịch

□ Minh họa



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

COST = 1 + 3 + 2 + 1 + 7

Bài toán người đi du lịch

□ Minh họa

```
while (i < n){  
    min =  $\infty$   
    for(k = 1; k  $\leq$  n; k++){  
        if(min > a[u][k]){  
            min = a[u][k];  
            v = k;  
        }  
        u = v; i++;  
        TOUR[i] = u; daxet[u] = 1; COST += min;  
    }  
    COST += a[u][START];  
    return COST;
```

Nội dung

- ❑ Lược đồ chung
- ❑ Bài toán cái túi - 2
- ❑ Bài toán người đi du lịch
- ❑ Bài toán sắp xếp phòng họp

Bài toán sắp xếp phòng họp

□ Bài toán

- Có n cuộc họp cần thực hiện. Cuộc họp thứ i ($i=1..n$) bắt đầu tại thời điểm s_i và kết thúc ở thời điểm f_i
- Giả sử có duy nhất 1 phòng họp. Hãy chọn ra số lượng cuộc họp tối đa mà phòng họp có thể tổ chức được.

Bài toán sắp xếp phòng họp

□ Tham lam - 1

- Gọi C là tập các cuộc họp ban đầu
- Gọi S là tập các cuộc họp được chọn
- Sắp xếp các cuộc họp theo thứ tự tăng dần của thời gian bắt đầu (s_i)

Bài toán sắp xếp phòng họp

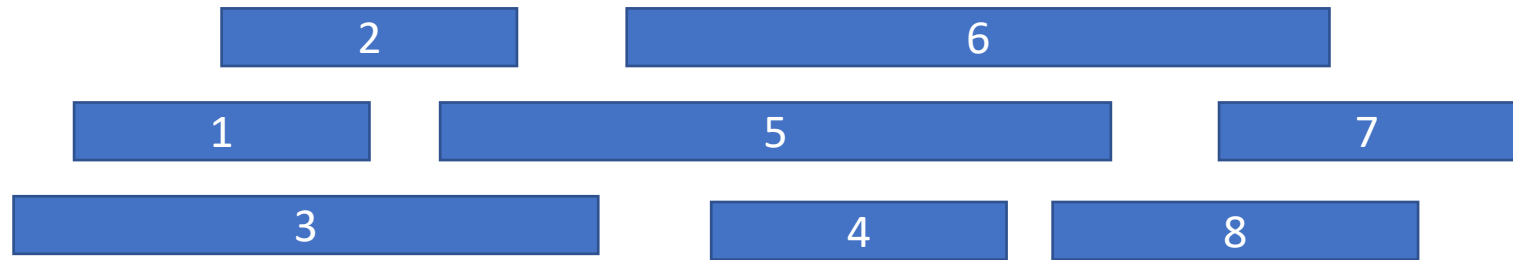
□ Tham lam - 1

- Lần lượt xét các đoạn trong danh sách theo thứ tự đã sắp xếp và bổ sung đoạn thẳng đang xét vào S nếu nó không có điểm chung với bất kỳ đoạn nào trong S

Bài toán sắp xếp phòng họp

□ Minh họa

- Cho 8 cuộc họp

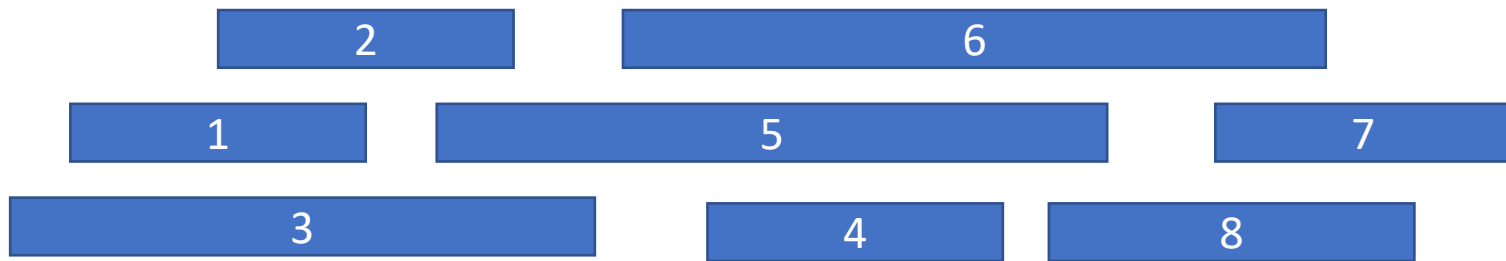


- Sắp xếp các cuộc họp theo thứ tự tang dần của thời điểm bắt đầu theo thứ tự các cuộc họp

$$C = \{3, 1, 2, 5, 6, 4, 8, 7\}$$

Bài toán sắp xếp phòng họp

□ Khởi tạo

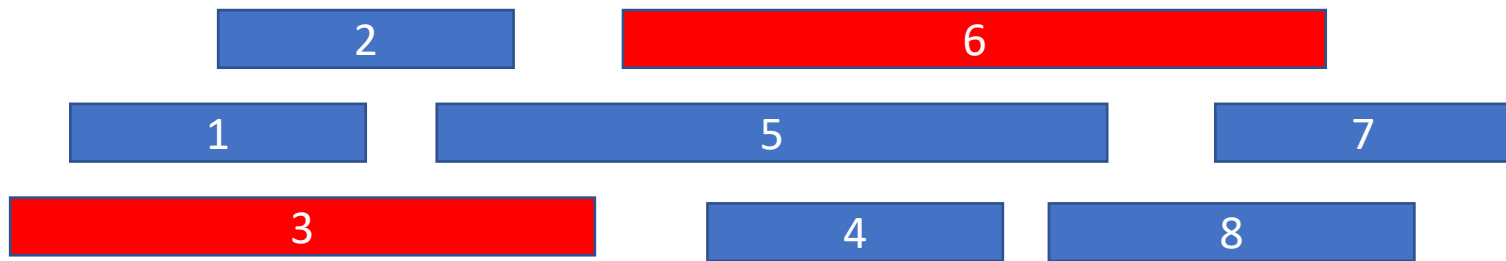


$C = \{3, 1, 2, 5, 6, 4, 8, 7\}$

$S = \{\}$

Bài toán sắp xếp phòng họp

□ Kết quả

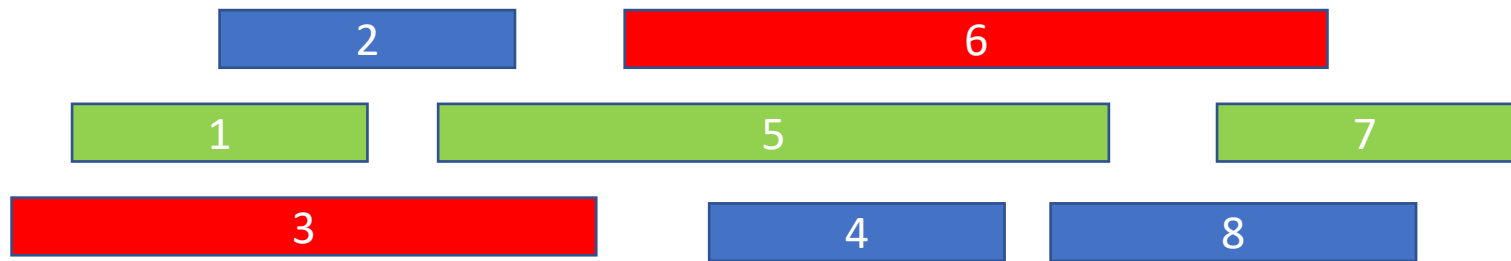


$$C = \{3, 1, 2, 5, 6, 4, 8, 7\}$$

$$S = \{3, 6\}$$

Bài toán sắp xếp phòng họp

□ Kết quả tối ưu



$$C = \{3, 1, 2, 5, 6, 4, 8, 7\}$$

$$S = \{3, 6\}$$

Phương án tốt hơn $S = \{1, 5, 7\}$

Bài toán sắp xếp phòng họp

□ Tham lam - 2

- Gọi C là tập các cuộc họp ban đầu
- Gọi S là tập các cuộc họp được chọn
- Sắp xếp các cuộc họp theo thứ tự tăng dần của thời gian thực hiện cuộc họp ($f_i - s_i$)

Bài toán sắp xếp phòng họp

□ Tham lam - 2

- Lần lượt xét các đoạn trong danh sách theo thứ tự đã sắp xếp và bổ sung đoạn thẳng đang xét vào S nếu nó không có điểm chung với bất kỳ đoạn nào trong S

Bài toán sắp xếp phòng họp

□ Tham lam - 3

- Gọi C là tập các cuộc họp ban đầu
- Gọi S là tập các cuộc họp được chọn
- Sắp xếp các cuộc họp theo thứ tự không giảm của thời gian kết thúc cuộc họp (f_i)

Bài toán sắp xếp phòng họp

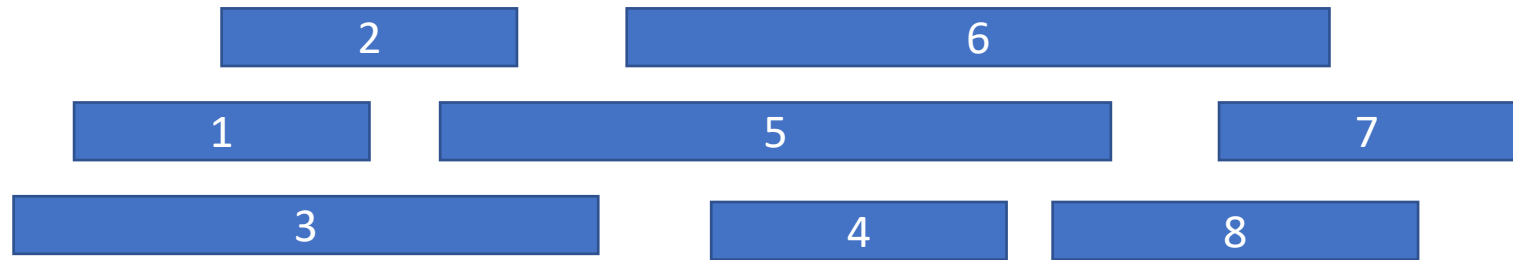
□ Tham lam - 3

- Lần lượt xét các đoạn trong danh sách theo thứ tự đã sắp xếp và bổ sung đoạn thẳng đang xét vào S nếu nó không có điểm chung với bất kỳ đoạn nào trong S

Bài toán sắp xếp phòng họp

□ Minh họa

- Cho 8 cuộc họp

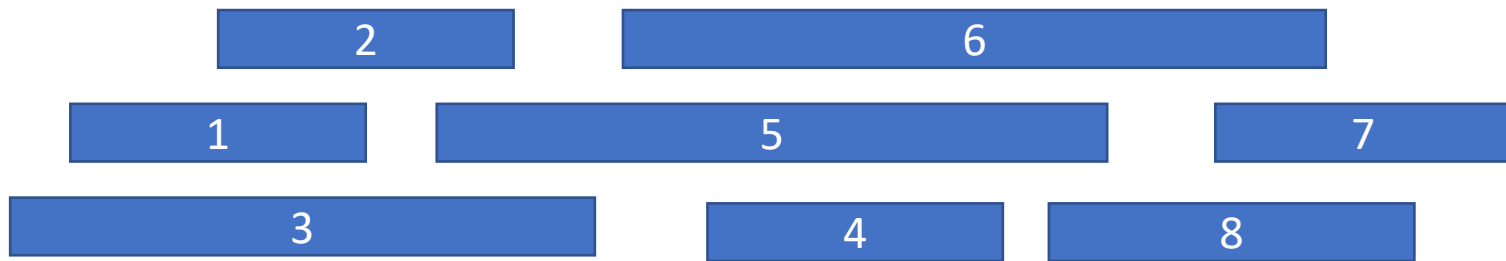


- Sắp xếp các cuộc họp theo thứ tự không giảm của thời điểm kết thúc theo thứ tự các cuộc họp

$$C = \{1, 2, 3, 4, 5, 6, 8, 7\}$$

Bài toán sắp xếp phòng họp

□ Khởi tạo

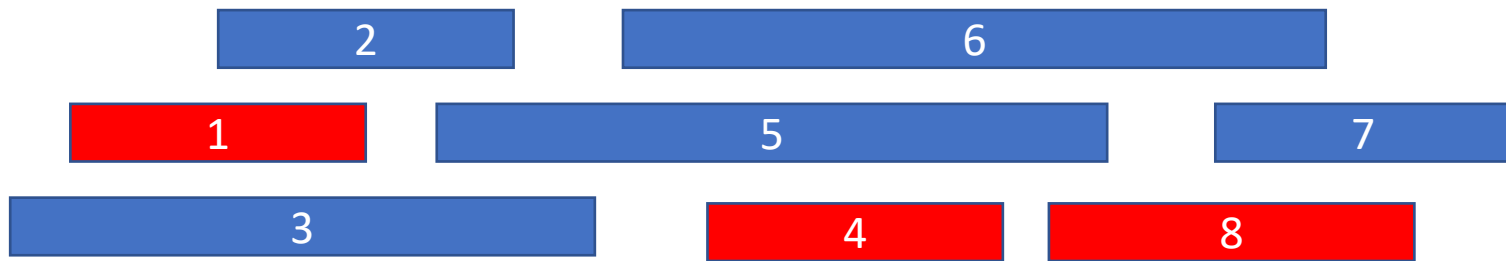


$C = \{1, 2, 3, 4, 5, 6, 8, 7\}$

$S = \{\}$

Bài toán sắp xếp phòng họp

□ Kết quả



$$C = \{3, 1, 2, 5, 6, 4, 8, 7\}$$

$$S = \{1, 4, 8\}$$

Bài toán sắp xếp phòng họp

□ Cài đặt

```
void SX_PhongHop(s, f){  
    sort(s, f) //theo thu tu tang cua f[i]  
    S = {1}  
    t = 1  
    for i= 2 to n:  
        if (f[t] < s[i]){  
            t = i;  
            S = S  $\cup$  {i}  
        }  
    }
```



