

# HUST

**ĐẠI HỌC BÁCH KHOA HÀ NỘI**

HANOI UNIVERSITY OF SCIENCE AND TECHNOLOGY

ONE LOVE. ONE FUTURE.

# CẤU TRÚC DỮ LIỆU VÀ GIẢI THUẬT



ĐẠI HỌC BÁCH KHOA HÀ NỘI

TRƯỜNG CÔNG NGHỆ THÔNG TIN  
VÀ TRUYỀN THÔNG

# CẤU TRÚC DỮ LIỆU VÀ GIẢI THUẬT

TUẦN 4: NHÁNH VÀ CẬN

ONE LOVE. ONE FUTURE.

1 Sơ đồ chung nhánh và cận

2 Bài toán người du lịch

# SƠ ĐỒ CHUNG

- Nhánh và cận (Branch and Bound): một trong số các phương pháp để giải bài toán tối ưu tổ hợp
  - Dùng kỹ thuật quay lui để liệt kê tất cả các phương án, từ đó giữ lại phương án tốt nhất
  - Dùng đánh giá cận (cận trên với bài toán tìm max và cận dưới với bài toán tìm min) để cắt bớt không gian tìm kiếm trong quá trình liệt kê

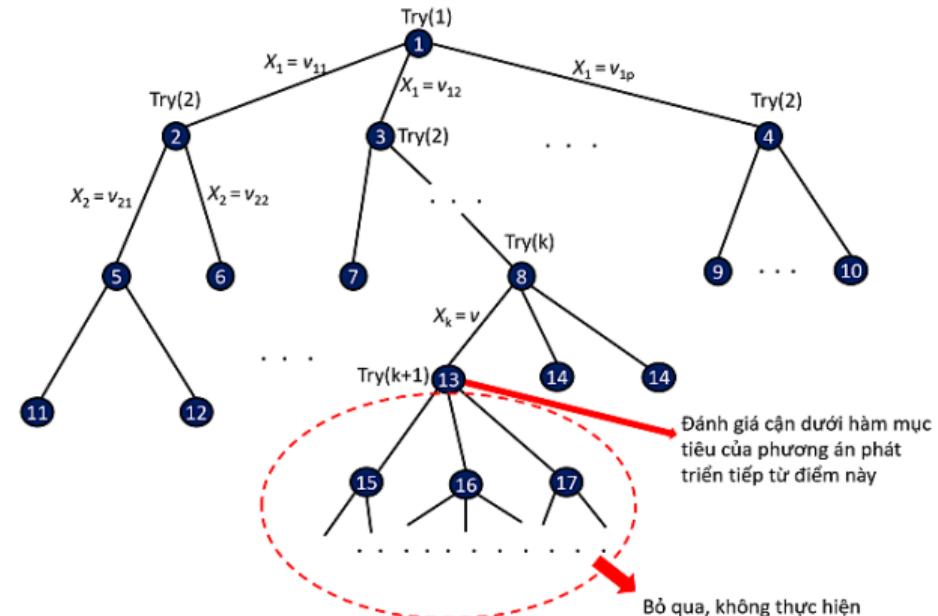


# SƠ ĐỒ CHUNG

- Xét bài toán tìm cực tiểu của hàm mục tiêu, trong đó lời giải được biểu diễn bởi một bộ các biến:

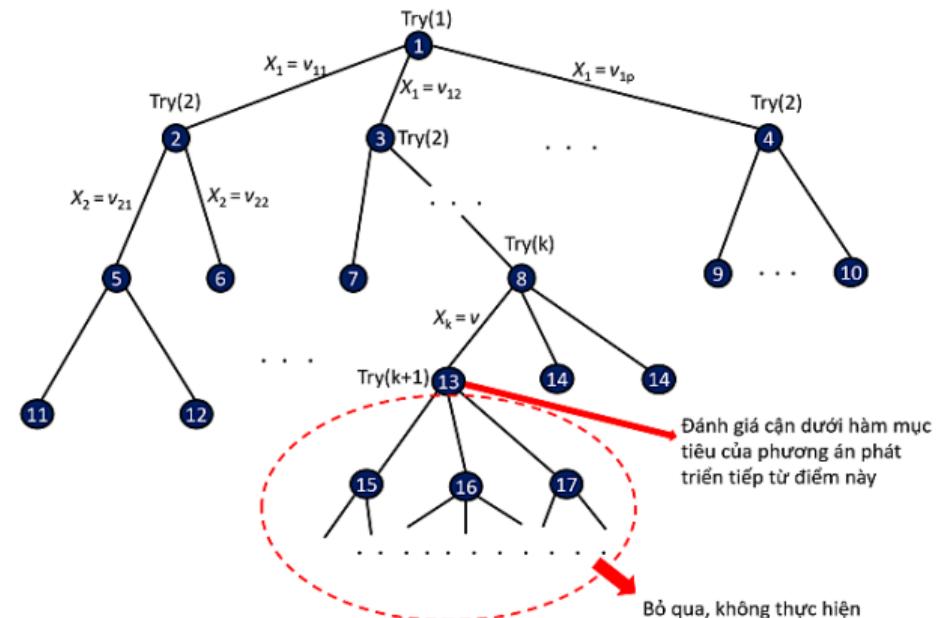
$$X = (X_1, X_2, \dots, X_n).$$

- Hàm Try( $k$ ) dùng để thử giá trị cho biến  $X_k$  trong quá trình liệt kê.
- Ký hiệu  $f^*$ : giá trị hàm mục tiêu của phương án tốt nhất đã tìm được.



## SƠ ĐỒ CHUNG

- Sau khi gán giá trị  $v$  cho  $X_k$ , ta đánh giá cận dưới  $g$  của hàm mục tiêu của các phương án phát triển tiếp từ điểm 13.
  - Nếu  $g \geq f^*$  thì không phát triển tiếp lời giải từ điểm 13.



1 Sơ đồ chung nhánh và cận

2 Bài toán người du lịch

# BÀI TOÁN NGƯỜI DU LỊCH

- Phát biểu bài toán:
  - Một người du lịch muốn đi tham quan  $n$  thành phố  $1, 2, \dots, n$ .
  - *Hành trình* là cách đi xuất phát từ thành phố 1 đi qua tất cả các thành phố còn lại, mỗi thành phố đúng một lần, rồi quay trở lại thành phố xuất phát 1.
  - Biết  $c(i, j)$  là chi phí đi từ thành phố  $i$  đến thành phố  $j$  ( $i, j = 1, 2, \dots, n$ ).
  - Tìm hành trình với tổng chi phí là nhỏ nhất.
- Một số nhận xét:
  - Số lượng hành trình của người du lịch là  $(n - 1)!$ .
  - Ta có tương ứng 1-1 giữa một hành trình của người du lịch:



Sir William Rowan Hamilton [1]  
(1805–1865)

$$1 \rightarrow x[2] \rightarrow x[3] \rightarrow \dots \rightarrow x[n] \rightarrow 1$$

với một hoán vị  $x = (x[2], x[3], \dots, x[n])$  của  $n - 1$  số tự nhiên  $2, 3, \dots, n$ .

- Chi phí hành trình:  $f(x) = c(1, x[2]) + c(x[2], x[3]) + \dots + c(x[n - 1], x[n]) + c(x[n], 1)$ .

# BÀI TOÁN NGƯỜI DU LỊCH

- Giải bằng phương pháp duyệt toàn bộ:
  - Hành trình:  $x = (1, x[2], x[3], \dots, x[n], 1)$

```
try(k) { // thử các giá trị có thể gán cho x[k]
    for v in candidates(k) do {
        if (check(v,k)) then {
            x[k] = v;
            [Update the data structure D]
            if (k == n) then solution();
            else try(k+1);
            [Recover the data structure D]
        }
    }
}
```

Cần xác định:  
1) candidates(k)  
2) check(v,k)

```
Init:
* f* = +∞; f = 0; x[1] = 1;
* for (int v = 2; v <= n; v++) visited[v] = 0;

void Try(int k) {
    for (int v = 2; v <= n; v++) {
        if (!visited[v]) {
            x[k] = v;
            visited[v] = 1;
            f = f + c(x[k-1], x[k]);
            if (k == n) { // Update record
                int ftemp = f + c(x[n], x[1]);
                if (ftemp < f*) f* = ftemp;
            }
            else Try(k + 1);
            f = f - c(x[k-1], x[k]);
            visited[v] = 0;
        }
    }
}
```

## Giải bằng phương pháp nhánh và cận

### Tính cận:

- Ký hiệu  $c_{\min} = \min\{ c(i,j) \mid i, j = 1, 2, \dots, n, i \neq j \}$  là chi phí đi lại nhỏ nhất giữa các thành phố.
- Cận ước lượng chi phí hành trình đầy đủ cho nhánh hiện tại tương ứng với hành trình bộ phận  $(1, u_2, \dots, u_k)$  đã đi qua  $k$  thành phố:

$$1 \rightarrow u_2 \rightarrow \dots \rightarrow u_{k-1} \rightarrow u_k.$$

- Nếu cận dưới  $g(1, u_2, \dots, u_k) \geq f^*$  thì không đi tiếp từ hành trình bộ phận  $(1, u_2, \dots, u_k)$ .

# BÀI TOÁN NGƯỜI DU LỊCH

- Cập ước lượng chi phí hành trình đầy đủ cho nhánh hiện tại tương ứng với hành trình bộ phận  $(1, u_2, \dots, u_k)$  đã đi qua  $k$  thành phố:  $1 \rightarrow u_2 \rightarrow \dots \rightarrow u_{k-1} \rightarrow u_k$ 
  - Chi phí phải trả theo hành trình bộ phận  $(1, u_2, \dots, u_k)$  là

$$\sigma = c(1, u_2) + c(u_2, u_3) + \dots + c(u_{k-1}, u_k).$$

- Để phát triển thành hành trình đầy đủ:

$$1 \rightarrow u_2 \rightarrow \dots \rightarrow u_{k-1} \rightarrow u_k \rightarrow u_{k+1} \rightarrow u_{k+2} \rightarrow \dots \rightarrow u_n \rightarrow 1$$

- Ta còn phải đi  $n - k + 1$  đoạn đường nữa, mỗi đoạn đường có chi phí không ít hơn  $c_{\min}$ , nên đoạn đường chưa đi có chi phí ít ra là  $(n - k + 1)c_{\min}$ .
- Vậy nếu đã đi hành trình bộ phận  $(1, u_2, \dots, u_k)$  thì đoạn đường còn lại dù đi thế nào thì tổng chi phí cũng lớn hơn hoặc bằng  $g(1, u_2, \dots, u_k) = \sigma + (n - k + 1)c_{\min}$ .

# BÀI TOÁN NGƯỜI DU LỊCH

- Hàm Try(k) tìm lời giải tối ưu cho bài toán người du lịch có sử dụng kỹ thuật **nhánh và cận**.

```
Main() {  
    //Init:  
    f* = +∞;  f = 0;  x[1] = 1;  
    for v = 2 to n do  visited[v] = false;  
    Try(2);  
    print(f*);  
}
```

```
Try(k) {  
    for v = 2 to n do {  
        if not visited[v] {  
            x[k] = v;  
            visited[v] = true;  
            f = f + c(x[k-1], x[k]);  
            if k = n then { //Update record  
                int ftemp = f + c(x[n], x[1]);  
                if (ftemp < f*) f* = ftemp;  
            }  
            else {  
                g = f + (n-k+1)*cmin;  
                if g < f* then Try(k+1);  
            }  
            f = f - c(x[k-1], x[k]);  
            visited[v] = false;  
        }  
    }  
}
```

A large, stylized graphic of the letters "HUST" is composed of numerous small red dots arranged in a circular, spiral-like pattern against a dark blue background.

**HUST**

**THANK YOU!**