



# HUST

**ĐẠI HỌC BÁCH KHOA HÀ NỘI**  
HANOI UNIVERSITY OF SCIENCE AND TECHNOLOGY

ONE LOVE. ONE FUTURE.





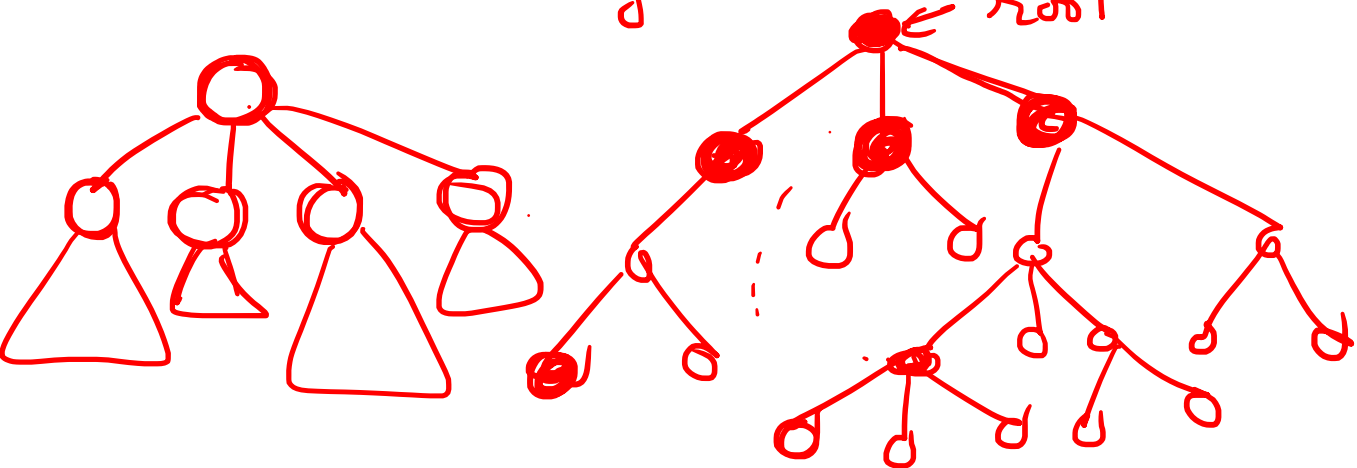
ĐẠI HỌC  
BÁCH KHOA HÀ NỘI  
HANOI UNIVERSITY  
OF SCIENCE AND TECHNOLOGY

# THUẬT TOÁN ỨNG DỤNG

THUẬT TOÁN TRÊN ĐỒ THỊ  
DFS và ứng dụng

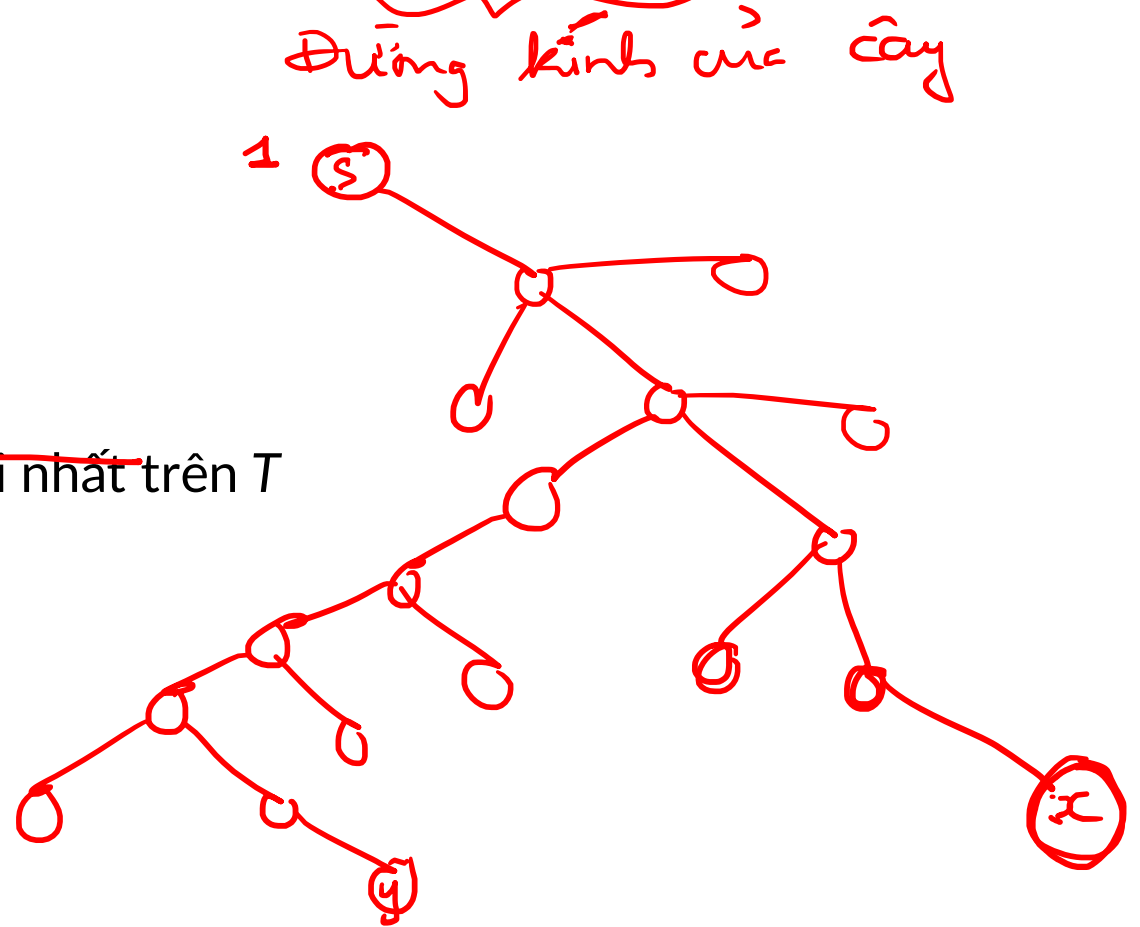
ONE LOVE. ONE FUTURE.

# NỘI DUNG

- Đường đi dài nhất trên cây → độ dài đường đi dài nhất liên thông BTC
  - Tổng đường đi trên cây
- độ dài đường đi dài nhất liên thông BTC  
Không có chu trình
- DFS, BFS, Dynamic Programming
- 

## ĐƯỜNG ĐI DÀI NHẤT TRÊN CÂY

- Cho cây  $T = (V, E)$ , mỗi cạnh  $(u, v)$  có trọng số  $w(u, v)$ . Hãy tìm đường đi có tổng trọng số lớn nhất trên  $T$
  - Ký hiệu  $A[v]$  là tập các đỉnh kề với đỉnh  $v$  trên  $T$
  - Thuật toán dựa trên duyệt theo chiều sâu
    - • Chọn 1 đỉnh  $s$  bất kỳ trên  $T$
    - • Thực hiện DFS( $s$ ) để tìm đỉnh  $x$  cách xa  $s$  nhất
      - Thực hiện DFS( $x$ ) để tìm đỉnh  $y$  cách xa  $x$  nhất
      - Đường đi từ  $x$  đến  $y$  tìm được sẽ là đường đi dài nhất trên  $T$
- Đường kính của cây
- 

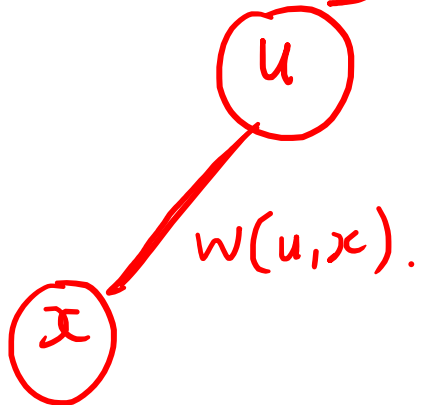


# ĐƯỜNG ĐI DÀI NHẤT TRÊN CÂY

```
Init(V, A) {  
  for v in V do d[v] = -1;  
}
```

```
DFS(u) {  
  for x in A[u] do {  
    if d[x] < 0 then {  
      d[x] = d[u] + w(u,x);  
      DFS(x);  
    }  
  }  
}
```

độ dài đường đi từ  
đỉnh x phát đến u



```
LongestPathOnTree(V, A){
```

```
  Init(V, A);  
  s = select a node in V;  
  DFS(s);  
  x = select u in V such that d[u] is maximal;  
  Init(V, A);  
  DFS(x);  
  y = select u in V such that d[u] is maximal;  
  P = unique path between x and y in T;  
  return P;  
}
```

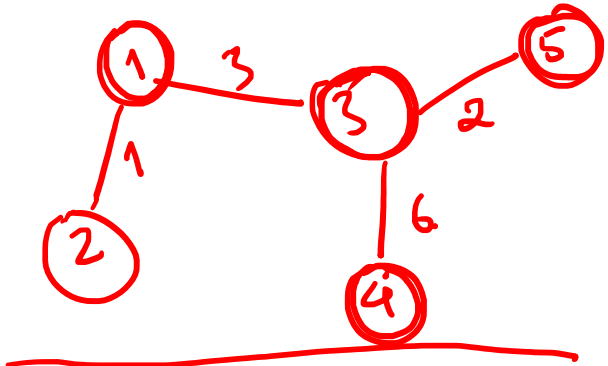
Time  $O(n+m)$

# ĐƯỜNG ĐI DÀI NHẤT TRÊN CÂY

- Độ phức tạp tính toán  $O(|V| + |E|)$

# TỔNG ĐƯỜNG ĐI TRÊN CÂY

- Cho cây  $T = (V, E)$ , mỗi cạnh  $(u, v)$  có trọng số  $w(u, v)$ . Tập đỉnh  $V$  gồm  $n$  đỉnh
- Ký hiệu:
  - $A[v]$  là tập các đỉnh kề với đỉnh  $v$  trên  $T$
  - $c(u, v)$  là độ dài đường đi duy nhất giữa 2 đỉnh  $u$  và  $v$  trên  $T$
  - $f(u)$ : tổng độ dài đường đi từ các đỉnh khác đến  $u$  trên  $T$ :  $f(u) = \sum_{v \in V} c(v, u)$
- Tìm  $f(u)$  với mọi  $u \in V$



$$\begin{aligned}f[1] &= 1 + 3 + 9 + 5 = 18 \\f[2] &= 1 + 4 + 10 + 6 = 21 \\f[3] &= 4 + 3 + 6 + 2 = 15 \\f[4] &= 9 + 10 + 6 + 8 = 33 \\f[5] &= 6 + 5 + 2 + 8 = 21\end{aligned}$$

DFS  $O(n + m)$

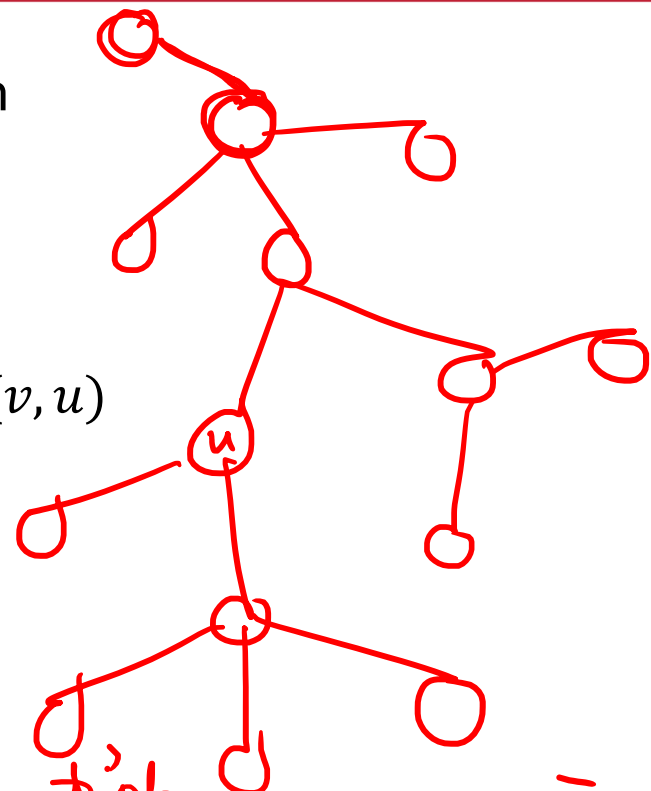
$$= O(n + n - 1)$$

$$= O(2n - 1) = O(n) \quad \text{for } u \in V \text{ do}$$

$$\text{DFS}(u) \rightarrow f[u]$$

Với mỗi đỉnh  $u \rightarrow \text{DFS}(u)$  để tìm độ dài đã từ các đỉnh khác đến  $u \rightarrow f[u]$ .

Time  $O(n^2)$





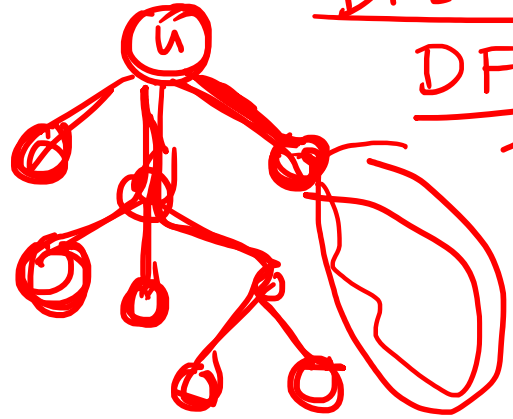
# TỔNG ĐƯỜNG ĐI TRÊN CÂY

- Chọn một đỉnh  $s$  bất kỳ trên  $T$  làm gốc, thực hiện duyệt theo chiều sâu trên  $T$  xuất phát từ  $s$ :

- $p(u)$ : đỉnh cha của  $u$  (là đỉnh mà từ đó thuật toán thăm  $u$ )
- $d(u)$ : tổng độ dài đường đi từ các đỉnh con cháu của  $u$  đến  $u$
- $N(u)$ : số lượng đỉnh con cháu của  $u$  (kể cả đỉnh  $u$ )

DFS 2 lần:

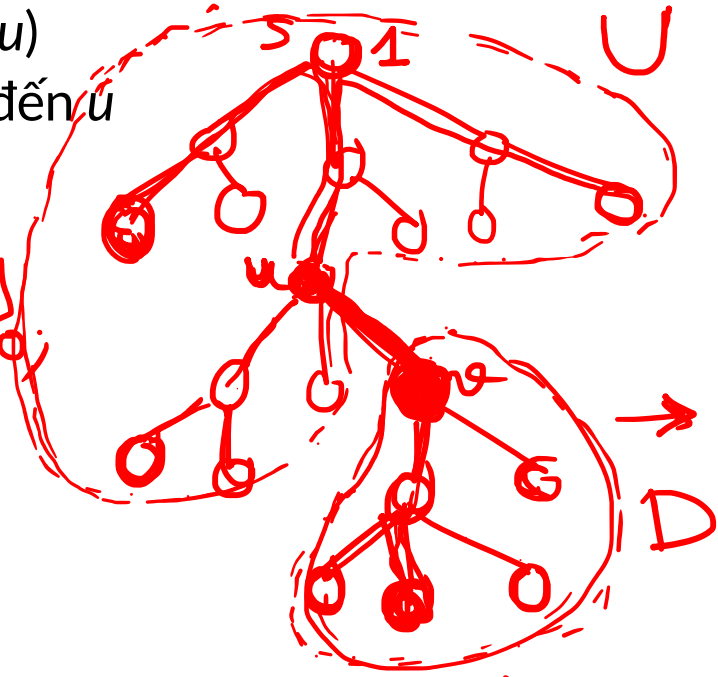
DFS 1(u): tính được  $d[u]$  và  $N[u]$   
 $\text{for } v \in A[u] \text{ do } \{$   
 $\text{DFS 1}(v);$   
 $d[u] = d[u] + d[v] + w(u,v)$   
 $N[u] = N[u] + N[v];$   
 $\}$



$N[u] = 8$

DFS 2(u): đã tính được  $p[u]$   
 $\rightarrow$  tính  $f[u]$  với mỗi đỉnh con  $v$  của  $u$   
 $F = f[u] - (d[v] + w(u,v) * N[v])$   
 $f[v] = F + w(u,v) * (n - N[v]) + d[v]$   
DFS 2(v)

time =  $O(n)$   
 $+ O(n) =$   
 $O(n)$



DFS 1(1)

$f[1] =$   
 $d[1]$

DFS 2(1)

# TỔNG ĐƯỜNG ĐI TRÊN CÂY

- DFS1( $u$ ): duyệt theo chiều sâu ở pha thứ nhất
  - Mục đích: tính  $d(x)$  và  $N(x)$  với mọi đỉnh  $x$  là con cháu của  $u$
  - Khi DFS1( $u$ ) thực hiện xong thì  $d(u)$  được tính xong và nó sẽ được dùng để tính  $d(p(u))$
  - Thực hiện: với mỗi đỉnh  $v \in A[u]$ :
    - Gọi DFS1( $v$ )
    - Cập nhật:  $d(u) = d(u) + N(v) * d(v)$
    - $N(u) = N(u) + N(v)$
- DFS2( $u$ ): duyệt theo chiều sâu ở pha thứ hai
  - Mục đích: Khi DFS2( $u$ ) được gọi thì  $f(u)$  đã được tính toán xong và ta sẽ tính toán  $f(v)$  với mỗi đỉnh  $v$  là con của  $u$
  - Thực hiện: với mỗi đỉnh  $v \in A[u]$  mà chưa được thăm
    - $F = f(u) - (d(v) + w(u,v) * N(v))$
    - $f(v) = F + d(v) + w(u,v) * (n - N(v))$
    - Gọi DFS2( $v$ )

- DFS1( $u$ ): duyệt theo chiều sâu ở pha thứ nhất
  - Mục đích: tính  $d(x)$  và  $N(x)$  với mọi đỉnh  $x$  là con cháu của  $u$
  - Khi DFS1( $u$ ) thực hiện xong thì  $d(u)$  được tính xong và nó sẽ được dùng để tính  $d(p(u))$
  - Thực hiện: với mỗi đỉnh  $v \in A[u]$ :
    - Gọi DFS1( $v$ )
    - Cập nhật:  $d(u) = d(u) + N(v) * d(v)$
    - $N(u) = N(u) + N(v)$

# TỔNG ĐƯỜNG ĐI TRÊN CÂY

DFS1(u){

```
for v in A[u] do {  
    if p(v) = 0 then {  
        p(v) = u;  
        DFS1(v);  
        d(u) = d(u) + d(v) + N(v)*w(u,v);  
        N(u) = N(u) + N(v);  
    }  
}
```

Phase1(){

```
for v in V do {  
    p(v) = 0; d(v) = 0; N(v) = 1; f(v) = 0;  
}  
p(1) = 1; DFS1(1);  
}
```

DFS2(u){

```
for v in A[u] do {  
    if p(v) = 0 then { // xét v là con của u  
        F = f(u) - (d(v) + N(v)*w(u,v));  
        f(v) = F + d(v) + w(u,v)*(n - N(v));  
        p(v) = u; DFS2(v);  
    }  
}
```

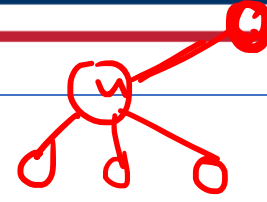
Phase2(){

```
for v in V do { p(v) = 0; }  
f(1) = d(1); p(1) = 1; DFS2(1);  
}
```

Main(){

Phase1(); Phase2();

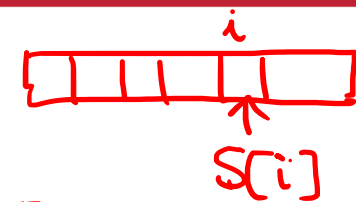
}



# TỔNG ĐƯỜNG ĐI TRÊN CÂY

- Độ phức tạp tính toán  $O(|V| + |E|)$

Tập độc lập lớn nhất trên cây,  $w[u]$  trọng số của đỉnh  $u$



Tìm tập con các đỉnh sao cho:

- 2 đỉnh kề nhau không cùng được chọn
- Tổng trọng số các đỉnh được chọn  $\rightarrow$  MAX

$\Rightarrow$  Dynamic Programming.

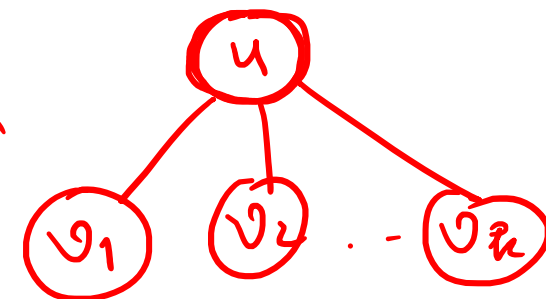
BT con:

(+)  $S0[u]$ : là tổng trọng số tập con max của các đỉnh trên cây gốc  $u$

, [Chọn] chọn  $u$

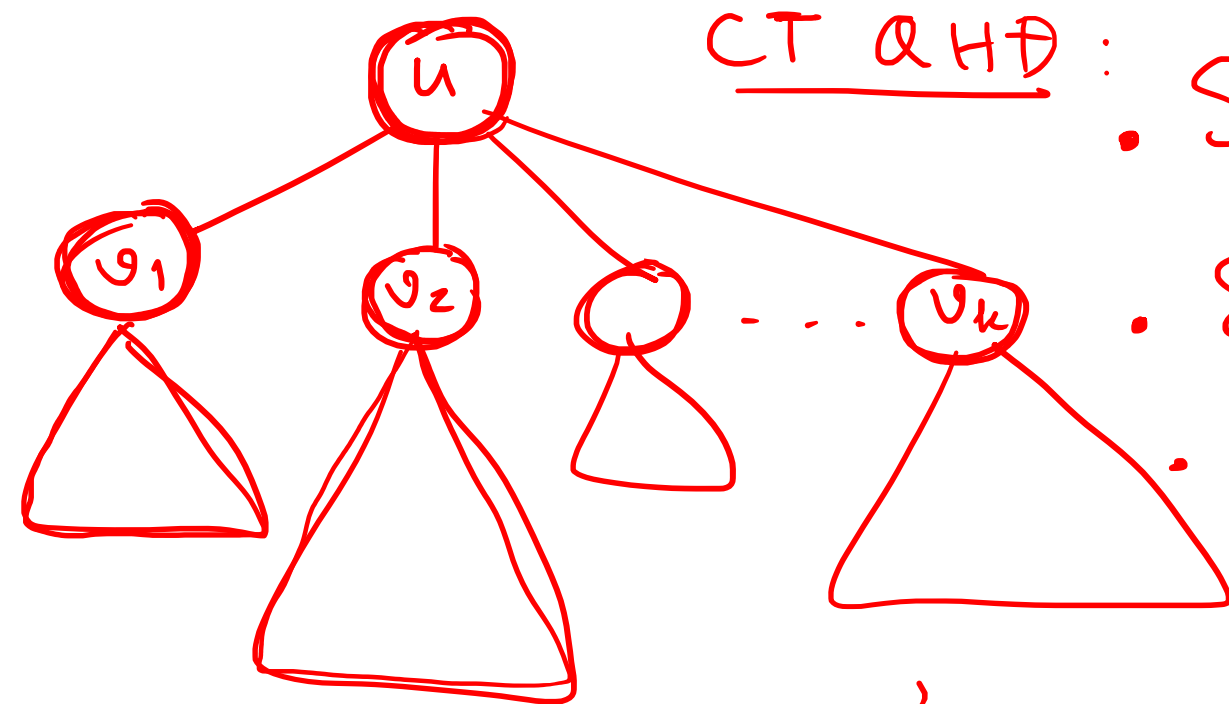
(+)  $S1[u]$ : tổng trọng số tập con max của các đỉnh cây gốc  $u$ , [Có] chọn  $u$

(+)  $S[u]$ : tổng trọng số tập con max trên cây gốc  $u$



$S[u] = f(\dots, S[v_1], S[v_2], \dots, S[v_k])$

$$S[u] = \max \{S0[u], S1[u]\}$$



CT QHTĐ :

$$SO[u] = \sum_{v \in \text{children}(u)} S[v]$$

$$S1[u] = w[u] + \sum_{v \in \text{children}(u)} SO[v]$$

$$S[u] = \max\{SO[u], S1[u]\}$$

→ Bottom-Up: giải BT con nhỏ nhất (nút lá: 07 có nút con)

→ Top-down: Đệ quy có nhớ.

A large graphic on the left side of the slide. It features a dark blue background with a circular pattern of red dots of varying sizes, creating a sense of depth and movement. The word "HUST" is centered within this graphic in a bold, white, sans-serif font.

# HUST

# THANK YOU !