TÌM KIẾM TRÊN KHÔNG GIAN TRẠNG THÁI

CHƯƠNG 2 (tiếp)

NỘI DUNG

- Một số khái niệm
- Cấu trúc chung của bài toán tìm kiếm
- Tìm kiếm lời giải trong không gian trạng thái
- Chiến lược tìm kiếm mù
- Tìm kiếm heuristics

CÁC CHIẾN LƯỢC TÌM KIẾM MÙ

- Tìm kiếm theo chiều rộng (Breadth First Search)
- Tìm kiếm theo chiều sâu (Depth First Search)
- Tìm kiếm sâu dần (Iterative Deepening Search)

TÌM KiẾM THEO CHIỀU RỘNG - BFS

- **Bài toán:**
  - **Vào:**
    - **Cho đồ thị G=<V,E>, V là tập các đỉnh và E là tập các cung nối các đỉnh.**
    - **Đỉnh đầu T_0.**
    - **Và tập Goal chứa các đỉnh đích.**
  - **Ra:**
    - **Đường đi p từ T_0 đến $T_G \in$ Goal**

TÌM KIẾM THEO CHIỀU RỘNG

– **Cách thực hiện:**
  - Từ đỉnh xuất phát duyệt tất cả các đỉnh kề.
  - Làm tương tự với các đỉnh vừa được duyệt.
  - Quá trình duyệt kết thúc khi tìm thấy đỉnh T_G hoặc đã hết các đỉnh để duyệt.

– **Lưu trữ:** Sử dụng hai danh sách **DONG** và **MO** hoạt động theo kiểu FIFO *(hàng đợi).*

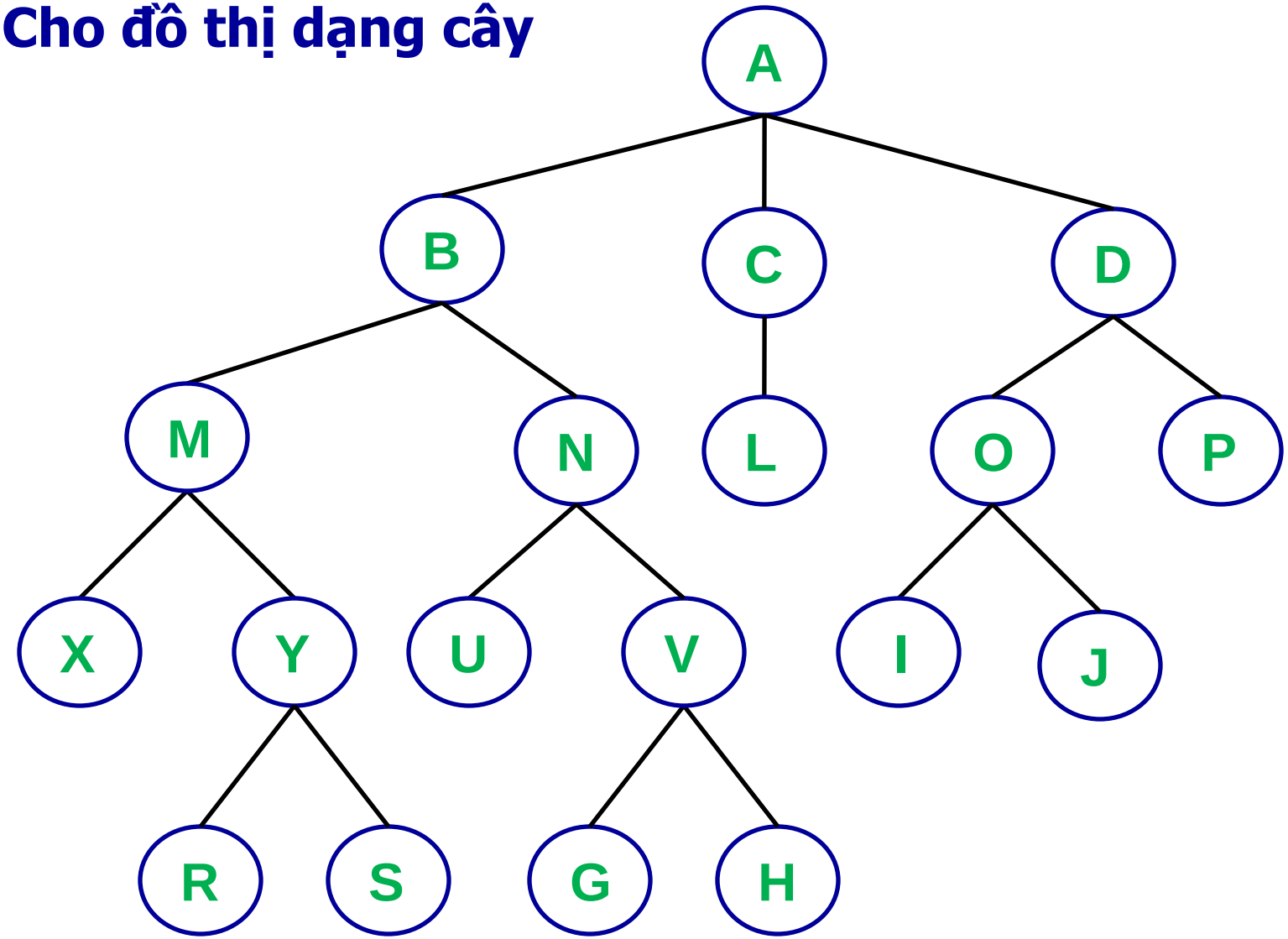
- **DONG:** Chứa các đỉnh đã xét
- **MO:** chứa các đỉnh đang xét
- $B(n)=\{m|(n,m)\in E\}$ *//Tập các đỉnh kề với n*

TÌM KIẾM THEO CHIỀU RỘNG

```
void BFS()
{  MO = MO∪{T0}
   while  (MO != ∅)
   {    n = get(MO) // lấy đỉnh đầu danh sach MO
        if (n==TG)
                return TRUE
        MO = MO∪B(n) //cho B(n) vào cuối DS MO
        DONG = DONG ∪ {n}
   }
}
```

TÌM KIẾM THEO CHIỀU RỘNG – Ví dụ

- **Cho đồ thị dạng cây**



TÌM KIẾM THEO CHIỀU RỘNG – Ví dụ

- Đỉnh đầu T_0=A, Goal = {R, O}
- Tìm đường đi p từ A đến một đỉnh $T_G \in$ Goal

n	B(n)	MO	DONG
		A	
A	B,C,D	B,C,D	A
B	M,N	C,D,M,N	A,B
C	L	D,M,N,L	A,B,C
D	O,P	M,N,L,O,P	A,B,C,D
M	X,Y	N,L,O,P,X,Y	A,B,C,D,M
N	U,V	L,O,P,X,Y,U,V	A,B,C,D,M,N
L	∅	O,P,X,Y,U,V	A,B,C,D,M,N,L
O →	Là đích →	Dừng	

TÌM KIẾM THEO CHIỀU RỘNG – Ví dụ

- **Đỉnh O $\in$ Goal -> Dừng quá trình tìm kiếm và xây dựng đường đi có hành trình: p = A -> D -> O**
- **Nhận xét:**
  - **Nếu trong đồ thị G tồn tại đường đi từ T_0 đến 1 đỉnh $T_G \in$ Goal thì hàm BFS sẽ dừng lại và cho đường đi p có độ dài ngắn nhất.**
  - **Với BFS các đỉnh được duyệt theo từng mức (theo chiều rộng).**
  - **Thuật toán BFS có độ phức tạp $O(b^d)$ với b là bậc của cây và d là chiều sâu của cây**

TÌM KIẾM THEO CHIỀU RỘNG – Ví dụ

- **Thời gian thực hiện BFS theo độ sâu d**

Độ sâu d	Thời gian	Không gian nhớ
4	11 giây	1 megabyte
6	18 giây	111 megabytes
8	31 giờ	11 gigabytes
10	128 ngày	1 terabyte
12	35 năm	111 terabytes
14	3500 năm	11111 terabytes

TÌM KIẾM THEO CHIỀU SÂU

– **Cách thực hiện:**

- Từ đỉnh xuất phát duyệt một đỉnh kề.
- Các đỉnh của đồ thị được duyệt theo các nhánh đến nút lá.
- Nếu chưa tìm thấy đỉnh T_G thì quay lui tới một đỉnh nào đó để sang nhánh khác.
- Việc tìm kiếm kết thúc khi tìm thấy đỉnh T_G hoặc đã hết các đỉnh

TÌM KIẾM THEO CHIỀU SÂU

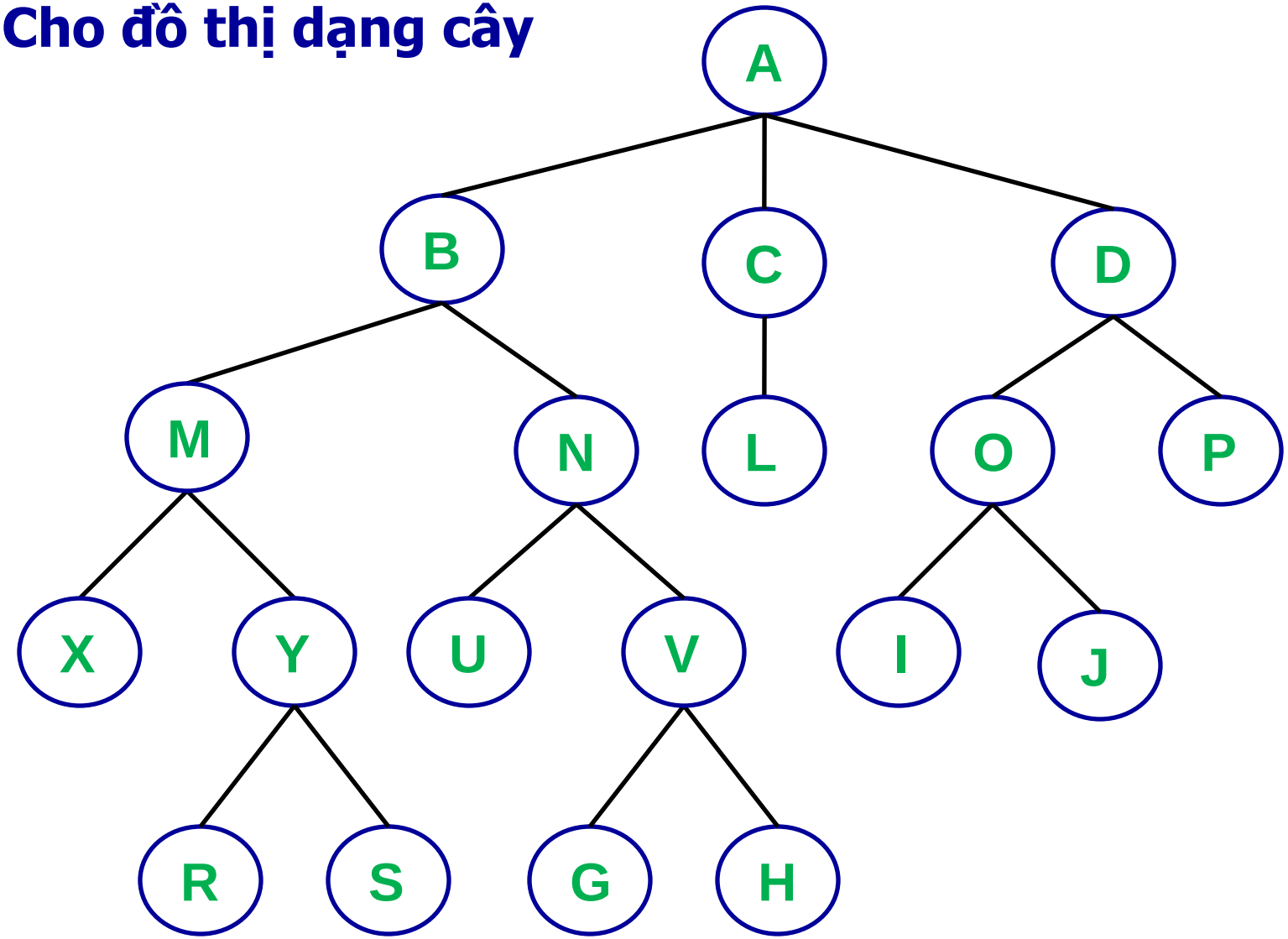
- **Lưu trữ: Sử dụng hai danh sách DONG và MO trong đó:**
  - **DONG: Chứa các đỉnh đã xét, hoạt động theo kiểu FIFO *(hàng đợi)*.**
  - **MO: chứa các đỉnh đang xét , hoạt động theo kiểu LIFO *(ngăn xếp)*.**
  - **B(n)={m|(n,m)∈E} *//Tập các đỉnh kề với n***

TÌM KIẾM THEO CHIỀU SÂU

```
void DFS()
{  MO = MO∪{T0}
   while  (MO != ∅)
   {    n = get(MO) // lấy đỉnh đầu danh sach MO
        if (n==TG)
                return TRUE
        MO = MO∪B(n) //cho B(n) vào đầu DS MO
        DONG = DONG ∪ {n}
   }
}
```

TÌM KIẾM THEO CHIỀU SÂU – Ví dụ

- **Cho đồ thị dạng cây**



TÌM KIẾM THEO CHIỀU SÂU – Ví dụ

- Đỉnh đầu T_0=A, Goal = {R, O}
- Tìm đường đi p từ A đến một đỉnh TG $\in$ Goal

n	B(n)	MO	DONG
		A	
A	B,C,D	B,C,D	A
B	M,N	M,N,C,D	A,B
M	X,Y	X,Y,N,C,D	A,B,M
X	∅	Y,N,C,D	A,B,M,X
Y	R,S	R,S,N,C,D	A,B,M,X,Y
R →	Là đích →	Dừng	

TÌM KIẾM THEO CHIỀU SÂU – Ví dụ

- **Đỉnh R $\in$ Goal -> Dừng quá trình tìm kiếm và xây dựng đường đi có hành trình:**

  p = A -> B -> M -> Y -> R $\in$ Goal

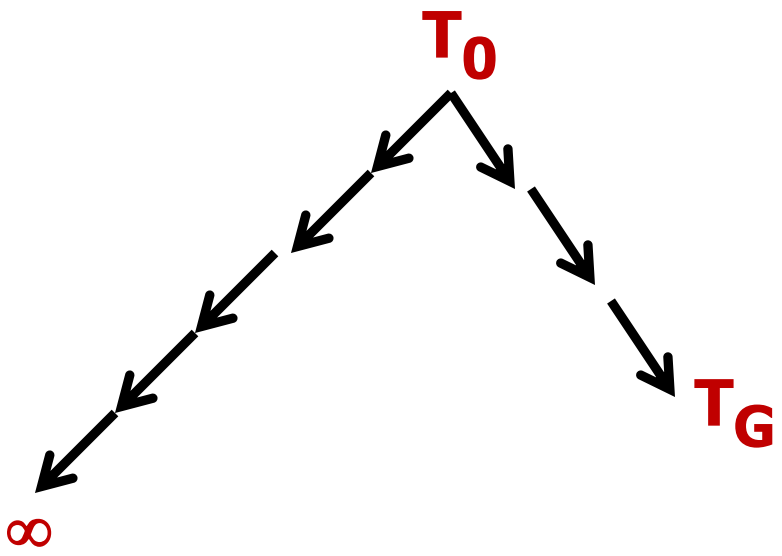
- **Nhận xét:**
  - **Nếu trong đồ thị G tồn tại đường đi từ T0 đến 1 đỉnh $T_G \in$ Goal thì hàm DFS sẽ dừng lại và cho đường đi p có độ dài có thể không ngắn nhất.**
  - **Với DFS các đỉnh được duyệt theo từng nhánh (theo chiều sâu) .**
  - **Thuật toán DFS có độ phức tạp O(bd) với b là bậc của cây và d là chiều sâu của cây. Tuy nhiên trong trường hợp xấu nhất cũng là $O(b^d)$**

SO SÁNH BFS VÀ DFS

	BFS	DFS
Thứ tự các đỉnh khi duyệt đồ thị	Các đỉnh được duyệt theo từng mức	Các đỉnh được duyệt theo từng nhánh
Độ dài đường đi p từ T_0 đến T_G	Ngắn nhất	Có thể không ngắn nhất
Tính hiệu quả	- Chiến lược có hiệu quả khi lời giải nằm ở mức thấp (gần gốc cây) - Thuận lợi khi tìm kiếm nhiều lời giải	- Chiến lược có hiệu quả khi lời giải nằm gần hướng đi được chọn theo phương án - Tìm kiếm 1 lời giải
Sử dụng bộ nhớ	Lưu trữ toàn bộ KGTT của bài toán	Lưu trữ các TT đang xét
Trường hợp tốt nhất	Vét cạn toàn bộ	Phương án chọn đường đi chính xác có lời giải trực tiếp
Trường hợp xấu nhất	Vét cạn	Vét cạn

TÌM KIẾM SÂU DẦN

- Trong trường hợp đồ thị tồn tại đường đi p từ T_0 đến $T_G \in$ Goal có thể DFS không dừng vì đi theo nhánh vô tận



- Để khắc phục người ta đưa vào thủ tục tìm kiếm theo chiều sâu một đại lượng giới hạn độ sâu

TÌM KIẾM SÂU DẦN

- **Ký hiệu:** d(n) là độ sâu hiện tại của đỉnh n

  d(0) = 0

  d(m) = d(n) + 1 nếu m $\in$ B(n)
- Trong trường hợp này thuật toán có thể dừng nhưng có thể cho kết quả không mong muốn.
- Vì đỉnh đích T_G nằm ở dưới độ sâu nên để khắc phục nhược điểm này người ta tăng dần độ sâu.
- Phương pháp sử dụng kỹ thuật này gọi là tìm kiếm sâu dần (Iterative Deepening Search)

THUẬT TOÁN TÌM KIẾM SÂU DẦN

- **Vào:** Đồ thị G =<V,E>

  Đỉnh đầu T_0 và Goal chứa các đỉnh đích

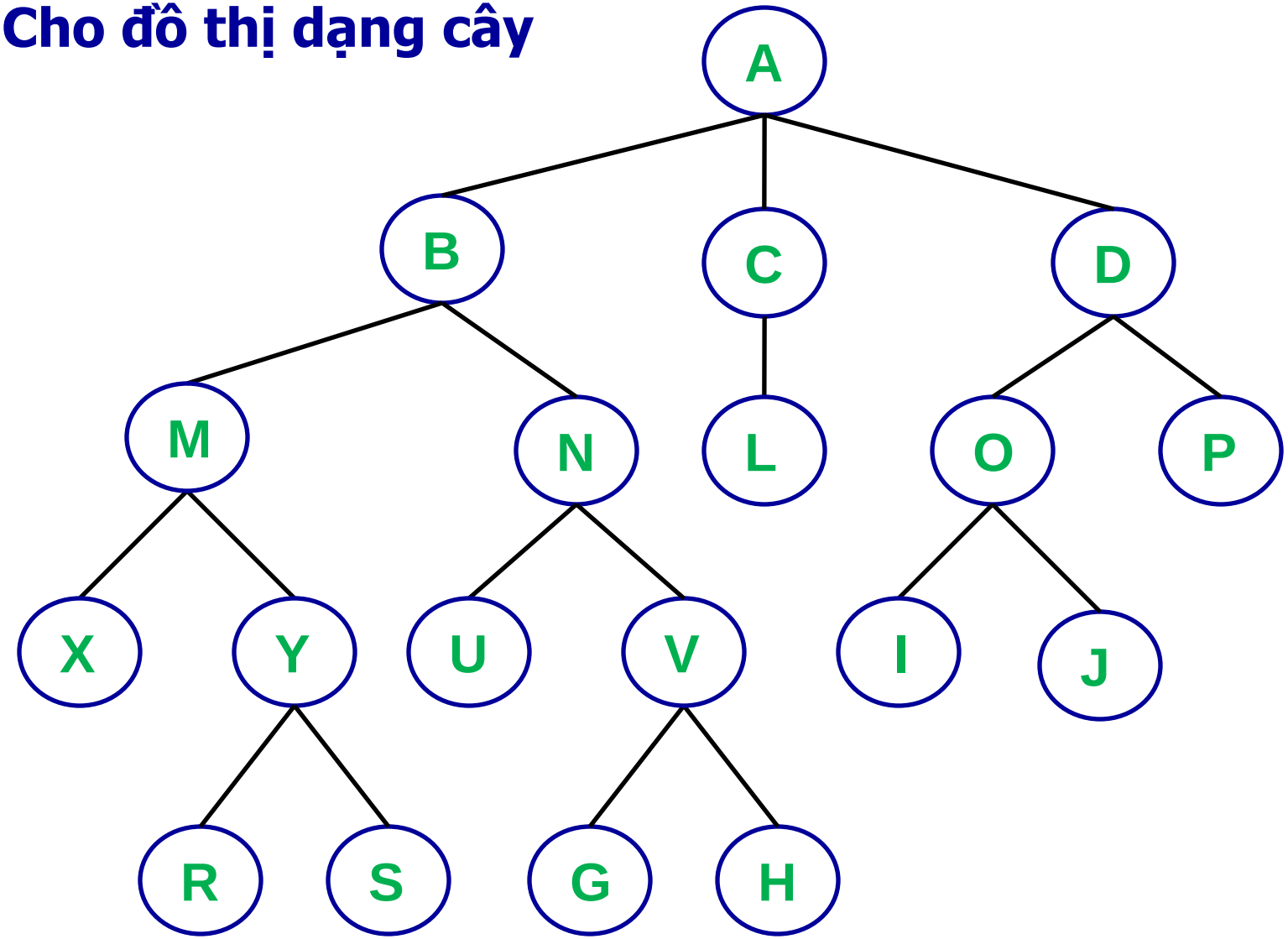
  Độ sâu ds = k giới hạn độ sâu
- **Ra:** Đường đi p: T0 -> TG $\in$ Goal.
- **Phương pháp:**
  - Sử dụng 2 danh sách **DONG** và **MO**.
  - **DONG** hoạt động theo nguyên tắc **FIFO**.
  - **MO** vừa hoạt động theo nguyên tắc **FIFO** vừa hoạt động theo nguyên tắc **LIFO**.

THUẬT TOÁN TÌM KIẾM SÂU DẦN

```
void IDS() {
    MO = {T0}; ds = k;
    while (MO != ∅) {
        n = get(MO);
        if (n==TG) return TRUE;
        DONG = DONG ∪ {n};
        switch (d(n)) {
            0…ds-1: Đặt B(n) vào đầu MO
            ds: Đặt B(n) vào cuối MO
            ds+1:  ds = ds+k;
                if (k==1) Đặt B(n) vào cuối MO
                else Đặt B(n) vào đầu MO
        }
    }
}
```

TÌM KIẾM SÂU DẦN – Ví dụ

- **Cho đồ thị dạng cây**



TÌM KIẾM SÂU DẦN – Ví dụ

- Đỉnh đầu T_0=A, Goal = {R, H}, độ sâu k=2

n	d(n)	B(n)	MO	DONG
			A	
A	0	B,C,D	B,C,D	A
B	1	M,N	M,N,C,D	A,B
M	2	X,Y	N,C,D,X,Y	A,B,M
N	2	U,V	C,D,X,Y,U,V	A,B,M,N
C	1	L	L,D,X,Y,U,V	A,B,M,N,C
L	2	∅	D,X,Y,U,V	A,B,M,N,C,L
D	1	O,P	O,P,X,Y,U,V	A,B,M,N,C,L,D
O →	Là đích →	Dừng		

TÌM KIẾM THEO CHIỀU SÂU – Ví dụ

- **Đỉnh O ∈ Goal -> Dừng quá trình tìm kiếm và xây dựng đường đi có hành trình: p = A -> D-> O ∈ Goal**
- **Kết luận:**
  - **Nếu trong đồ thị G tồn tại đường đi từ T_0 đến 1 đỉnh T_G ∈ Goal thì hàm IDS sẽ dừng lại và cho đường đi p có độ dài khác độ dài đường đi ngắn nhất không quá k-1.**
- **Nhận xét**
  - **TK sâu dần là sự kết hợp giữa DFS và BFS .**
  - **Nếu k=1 thì thuật toán trở thành BFS.**
  - **Nếu k là chiều cao của cây thì thuật toán là DFS.**
  - **Thuật toán này có độ phức tạp $O(b^d)$ với b là bậc của cây và d là chiều sâu của cây.**