TRƯỜNG ĐẠI HỌC CÔNG NGHIỆP HÀ NỘI

Đề cương bài giảng

Trí tuệ nhân tạo

Ngành : Công nghệ thông tin
Trình độ đào tạo: Đại học
Người biên soạn : Khoa CNTT

(Lưu hành nội bộ)

Hà Nội – 5 / 2021

Bài 3: Thuật toán tìm kiếm trong không gian trạng thái

Vấn đề tìm kiếm, một cách tổng quát, có thể hiểu là tìm một đối tượng thỏa mãn một số đòi hỏi nào đó, trong một tập hợp rộng lớn các đối tượng. Chúng ta có thể kể ra rất nhiều vấn đề mà việc giải quyết nó được quy về vấn đề tìm kiếm.

Các trò chơi, chẳng hạn cờ vua, cờ carô có thể xem như vấn đề tìm kiếm. Trong số rất nhiều nước đi được phép thực hiện, ta phải tìm ra các nước đi dẫn tới tình thế kết cuộc mà ta là người thắng.

Trong các lĩnh vực nghiên cứu của **Trí Tuệ Nhân Tạo**, chúng ta thường xuyên phải đối đầu với vấn đề tìm kiếm. Các kỹ thuật tìm kiếm được áp dụng để giải quyết các vấn đề và được áp dụng rộng rãi trong các lĩnh vực nghiên cứu khác của **Trí Tuệ Nhân Tạo**.

Trong phần này, chúng ta sẽ nghiên cứu các thuật toán tìm kiếm theo chiều sâu và thuật toán tìm kiếm theo chiều rộng trong bài toán tìm kiếm trạng thái mục tiêu trên không gian trạng thái.

3.1Thuật toán tìm kiếm theo chiều sâu (Depth First Search)

a. Tư tưởng của chiến lược tìm kiếm theo chiều sâu

- 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

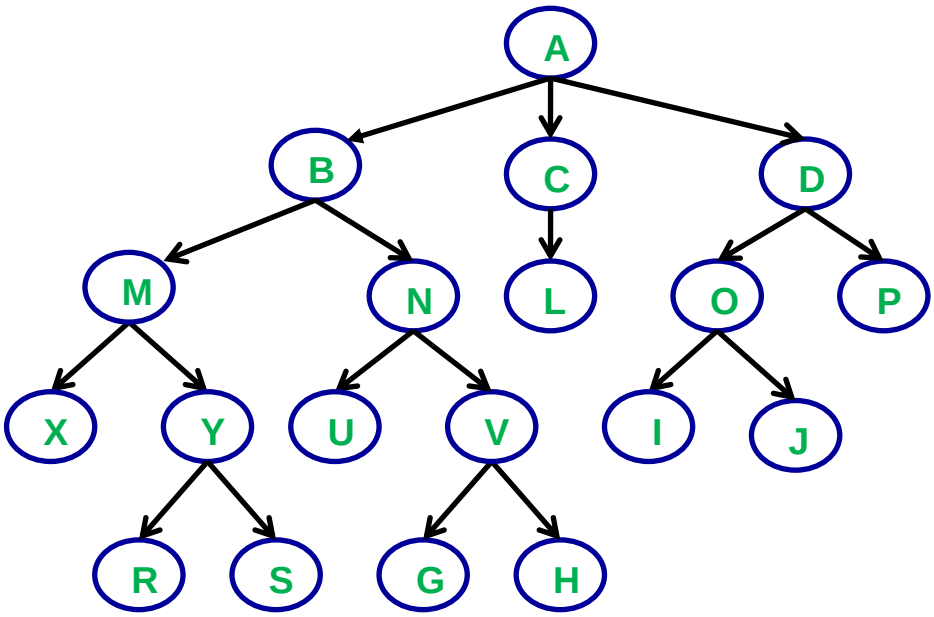
b. Thuật toán tìm kiếm theo chiều sâu

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)*.

```
1. MO = ∅;        MO = MO ∪ {T0}
2.  while  (MO != ∅)
    {       n = get(MO) // lấy đỉnh đầu trong danh sach MO
            if (n==TG) // nếu n là trạng thái kết thúc
                    return TRUE // tìm kiếm thành công, dừng
            DONG = DONG ∪ {n} //đánh dấu n đã được xét
            for các đỉnh kề v của n
                    if  (v chưa đc xét) //v chưa ở trong DONG
                     MO = MO∪ {v} //đưa v vào đầu DS MO
                            father(v)=n// lưu lại vết đường đi từ n đến v
    }
```

c. Ví dụ thuật toán tìm kiếm theo chiều sâu.

Cho đồ thị như hình vẽ sau:



Đỉnh đầu T_0=A, T_G = {R}
Tìm đường đi p từ To đến T_G bằng phương pháp tìm kiếm theo chiều sâu?

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		

Xây dựng đường đi có hành trình: p = A -> B -> M -> Y -> R

Nhận xét:
+ Nếu trong đồ thị G tồn tại đường đi từ T_0 đến 1 đỉnh TG ∈ 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 $O(b^d)$

3.2 Thuật toán tìm kiếm theo chiều rộng (Breadth First Search)

a. Tư tưởng của chiến lược tìm kiếm theo chiều rộng

- 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 TG hoặc đã hết các đỉnh để duyệt.

b. Thuật toán tìm kiếm theo chiều rộng

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

```
1. MO = ∅; MO = MO ∪ {T0}
2.      while  (MO != ∅)
        {       n = get(MO) // lấy đỉnh đầu trong danh sach MO
                if (n==TG) // nếu n là trạng thái kết thúc
                        return TRUE // tìm kiếm thành công, dừng
                DONG = DONG ∪ {n} //đánh dấu n đã được xét
                for các đỉnh kề v của n
                        if  (v chưa đc xét) //v chưa ở trong DONG
                         MO = MO∪ {v} //đưa v vào cuối DS MO
                                father(v)=n// lưu lại vết đường đi từ n đến v
        }
```

Chúng ta có một số nhận xét sau đây về thuật toán tìm kiếm theo chiều rộng:

Trong tìm kiếm theo chiều rộng, trạng thái nào được sinh ra trước sẽ được phát triển trước, do đó danh sách MỞ được xử lý như hàng đợi. Trong bước 2, ta cần kiểm tra xem n có là trạng thái kết thúc hay không. Nói chung các trạng thái kết thúc được xác định bởi một số điều kiện nào đó, khi đó ta cần kiểm tra xem n có thỏa mãn các điều kiện đó hay không.

Nếu bài toán có nghiệm (tồn tại đường đi từ trạng thái ban đầu tới trạng thái đích), thì thuật toán tìm kiếm theo chiều rộng sẽ tìm ra nghiệm, đồng thời đường đi tìm được sẽ là ngắn nhất. Trong trường hợp bài toán vô nghiệm và không gian trạng thái hữu hạn, thuật toán sẽ dừng và cho thông báo vô nghiệm.

Đánh giá tìm kiếm theo chiều rộng:

Bây giờ ta đánh giá thời gian và bộ nhớ mà tìm kiếm theo chiều rộng đòi hỏi. Giả sử , mỗi trạng thái khi được phát triển sẽ sinh ra b trạng thái kề. Ta sẽ gọi b là nhân tố nhánh. Giả sử rằng, nghiệm của bài toán là đường đi có độ dài d. Bởi nhiều nghiệm có thể được tìm ra tại một đỉnh bất kỳ ở mức d của cây tìm kiếm, do đó số đỉnh cần xem xét để tìm ra nghiệm là:

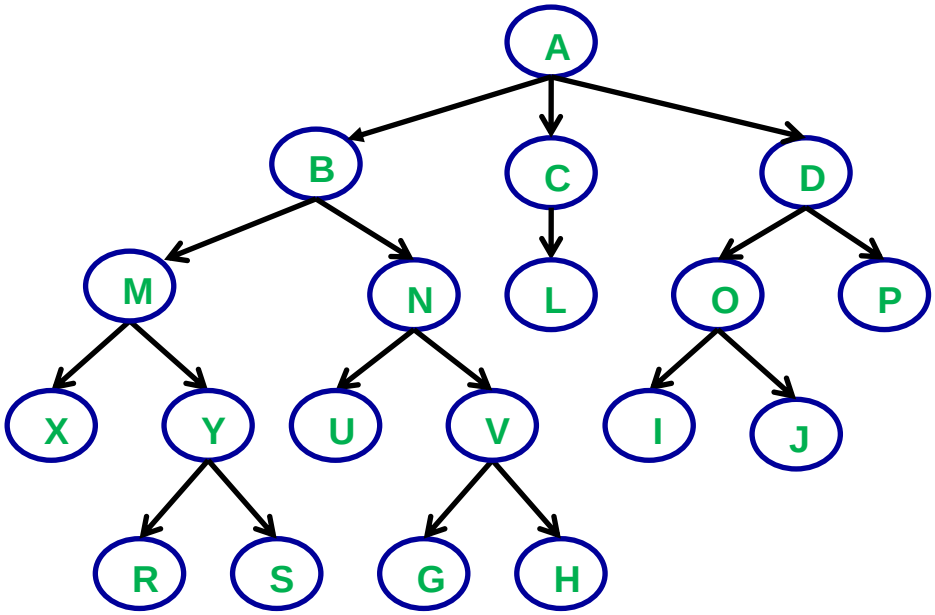
$1 + b + b^2 + \ldots + b^{d-1} + k$

Trong đó k có thể là 1, 2, …, bd. Do đó số lớn nhất các đỉnh cần xem xét là: $1 + b + b^2 + \ldots + b^{d-1}$

Như vậy, độ phức tạp thời gian của thuật toán tìm kiếm theo chiều rộng là O(bd). Độ phức tạp không gian cũng là $O(b^d)$, bởi vì ta cần lưu vào danh sách MỞ tất cả các đỉnh của cây tìm kiếm ở mức d, số các đỉnh này là b^d.

c. Ví dụ thuật toán tìm kiếm theo chiều rộng.

Cho đồ thị như hình vẽ sau:



Đỉnh đầu T_0=A, T_G= {N}.Tìm đường đi p từ To đến T_G bằng phương pháp tìm kiếm theo chiều rộng?

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	→ là đích → dừng		

Xây dựng đường đi có hành trình: p = A -> B -> N.

Nhận xét:

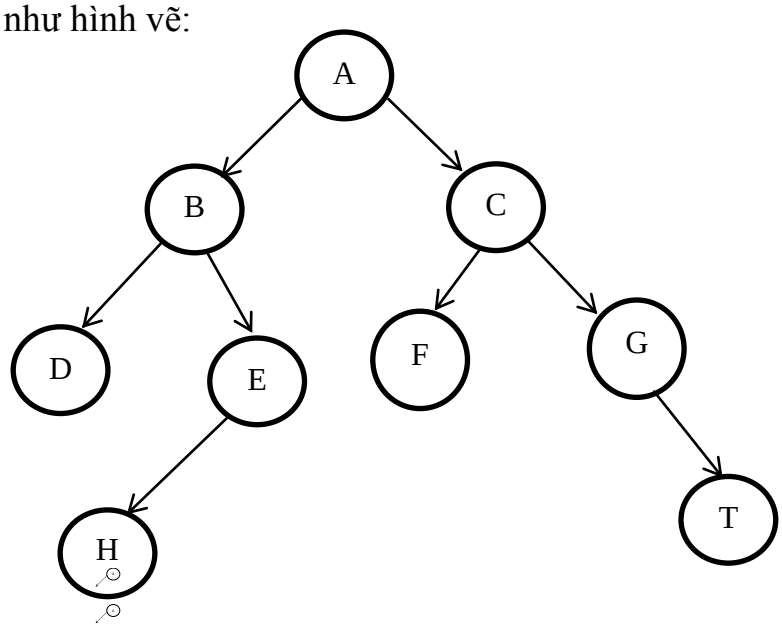
+ Nếu trong đồ thị tồn tại đường đi từ T_0 đến 1 đỉnh $T_G \in$ Goal thì hàm BFS sẽ dừng lại và cho đường đi p có độ dài ngắn nhất.
+ Với BFS các đỉnh được duyệt theo từng mức (theo chiều rộng).
+ Thuật toán BFS có độ phức tạp $O(b^d)$ với b là bậc của cây và d là chiều sâu của cây

Bảng so sánh 2 thuật toán DFS và BFS

	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 - Thuận lợi khi 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

II. Bài tập

Bài 1: Cho đồ thị như hình vẽ:



Yêu cầu bài toán: Tìm đường đi p từ A đến một đỉnh $T_G \in$ Goal bằng thuật toán tìm kiếm theo chiều sâu biết dỉnh đầu T_0=A, Goal = {H, T}

Bài làm:

- 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

- 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

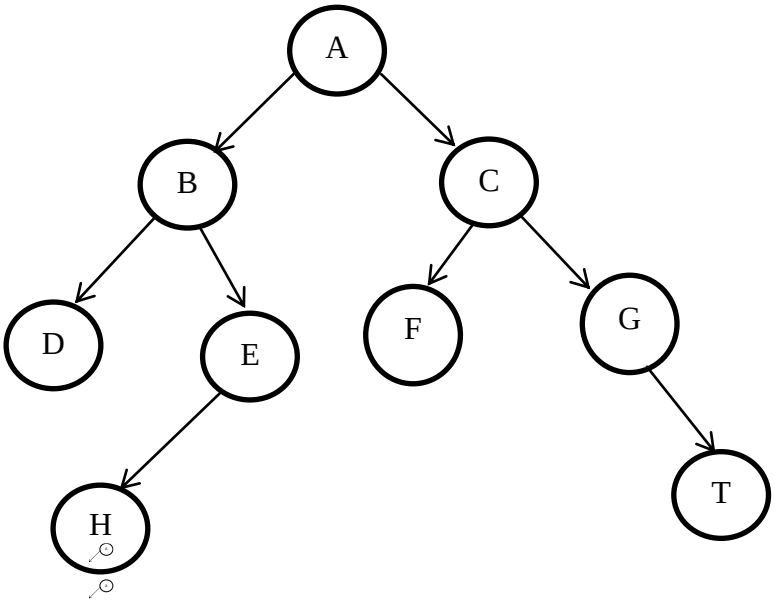
n	B(n)	MO	DONG
		A	
A	B,C	B,C	A
B	D,E	D,E,C	A,B
D	∅	E,C	A,B,D
E	H	H,C	A,B,D,E
H	là đích → dừng		

Cha	Con
A	B
A	C
B	D
B	E
E	H

Đường đi từ A đến Goal là : A -> B -> E -> H

Bài 2: Đỉnh đầu T_0=A, Goal = {H, F}

Tìm đường đi p từ A đến một đỉnh $T_G \in$ Goal



- 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

  – 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.
- 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)∈E} //Tập các đỉnh kề với n

N	B(n)	MO	DONG
		A	
A	B,C	B,C	A
B	D,E	C,D,F	A,B
C	F,G	D,E,F,G	A,B,C
D	∅	E,F,G	A,B,C,D
E	H	F,G,H	A,B,C,D,E
F → là đích → dừng			

Cha	con
A	B
A	C
B	D
B	E
C	F
C	G
E	H

⇨ Đường đi từ A đến Goal là : A -> C -> F

Tài liệu tham khảo

[1] *Trí tuệ nhân tạo*
Trường Đại học Công Nghiệp Hà Nội - NXB Giáo dục Việt Nam, 2012

[2] *Trí tuệ nhân tạo*
Đinh Mạnh Tường - NXB Khoa học kỹ thuật, 2002.