Chương 3
Giải quyết vấn đề

Lê Thanh Hương

Viện CNTT &TT – ĐHBKHN

Nội dung môn học

3.1. Tổng quan

- Khoa học trí tuệ nhân tạo quan tâm đến việc tạo ra các đối tượng có thể…
  - Hành động đúng
  - trên cơ sở hoàn cảnh cụ thể và những thứ mà nó đã biết

3.1.1. Phân loại vấn đề

- GQVĐ là quá trình xuất phát từ **hình trạng đầu**, tìm kiếm trong không gian bài toán để tìm ra **dãy toán tử** hay **dãy hành động** cho phép dẫn tới **đích**.
- **BT phát biểu chỉnh**: là BT biết rõ đầu vào, đầu ra và với mỗi lời giải giả định nào đó, có thể áp dụng thuật toán để xác định xem đó có phải là lời giải của BT ban đầu hay không.
- **BT phát biểu không chỉnh**: ngược lại

3.1.1. Phân loại vấn đề

BT phát biểu chỉnh

BT phát biểu không chỉnh

ĐPT đa thức

$O(n^{\alpha})$

ĐPT hàm mũ

$O(\alpha^{n})$

giải được

ko giải được

Giải thuật

Mẹo giải

Ví dụ 1. Bài toán đố chữ

- Hãy thay các chữ cái bằng các chữ số từ 0 đến 9 sao cho không có hai chữ cái nào được thay bởi cùng 1 số và thỏa mãn ràng buộc sau:

$$\begin{array}{rr} & \text{SEND} \\ + & \text{MORE} \\ \hline & \text{MONEY} \end{array} \qquad \begin{array}{rr} & \text{CROSS} \\ + & \text{ROADS} \\ \hline & \text{DANGER} \end{array}$$

Ví dụ 2. Bài toán rót nước

- Cho 2 bình A(m lít), B(n lít). Làm cách nào để đong được k lít ($k \le max(m,n)$) chỉ bằng 2 bình A, B và 1 bình trung gian C.
- Các thao tác rót (how):

  C → A; C → B; A → B; A → C; B → A; B → C

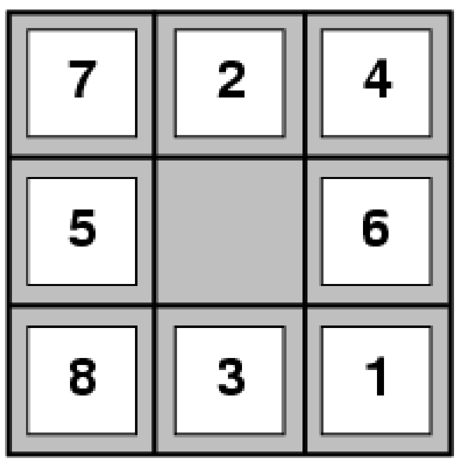
- Điều kiện: không tràn, đổ hết
- Ví dụ: m = 5, n = 6, k = 2   (what)
- Mô hình toán học:

  (x, y) → (x', y')

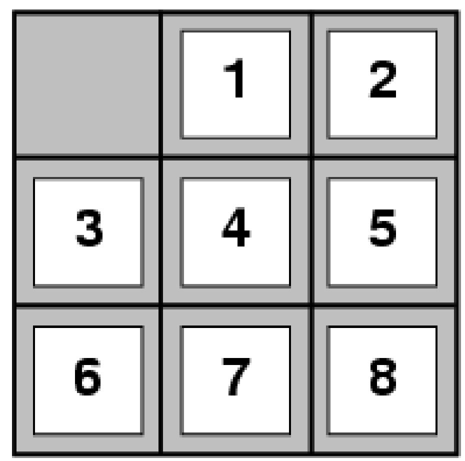
  A B   A B

Ví dụ 3. Bài toán trò chơi n^2 – 1 số

- Trong bảng ô vuông n hàng, n cột, mỗi ô chứa 1 số nằm trong phạm vi từ 1 → n^2 -1 sao cho không có 2 ô có cùng giá trị. Còn đúng 1 ô bị trống. Xuất phát từ 1 cách sắp xếp nào đó của các đó của các số trong bảng, hãy dịch chuyển các ô trống sang phải, sang trái, lên trên, xuống dưới để đưa về bảng:



Start State



Goal State

Ví dụ 4. Bài toán tháp Hà Nội

- Cho 3 cọc 1,2,3. Ở cọc 1 ban đầu có n đĩa, sắp theo thứ tự to dần từ trên xuống dưới. Hãy tìm cách chuyển n đĩa đó sang cọc 3 sao cho:
  - Mỗi lần chỉ chuyển 1 đĩa
  - Ở mỗi cọc không cho phép đĩa to nằm trên đĩa con

Bài toán tháp Hà Nội với n = 3

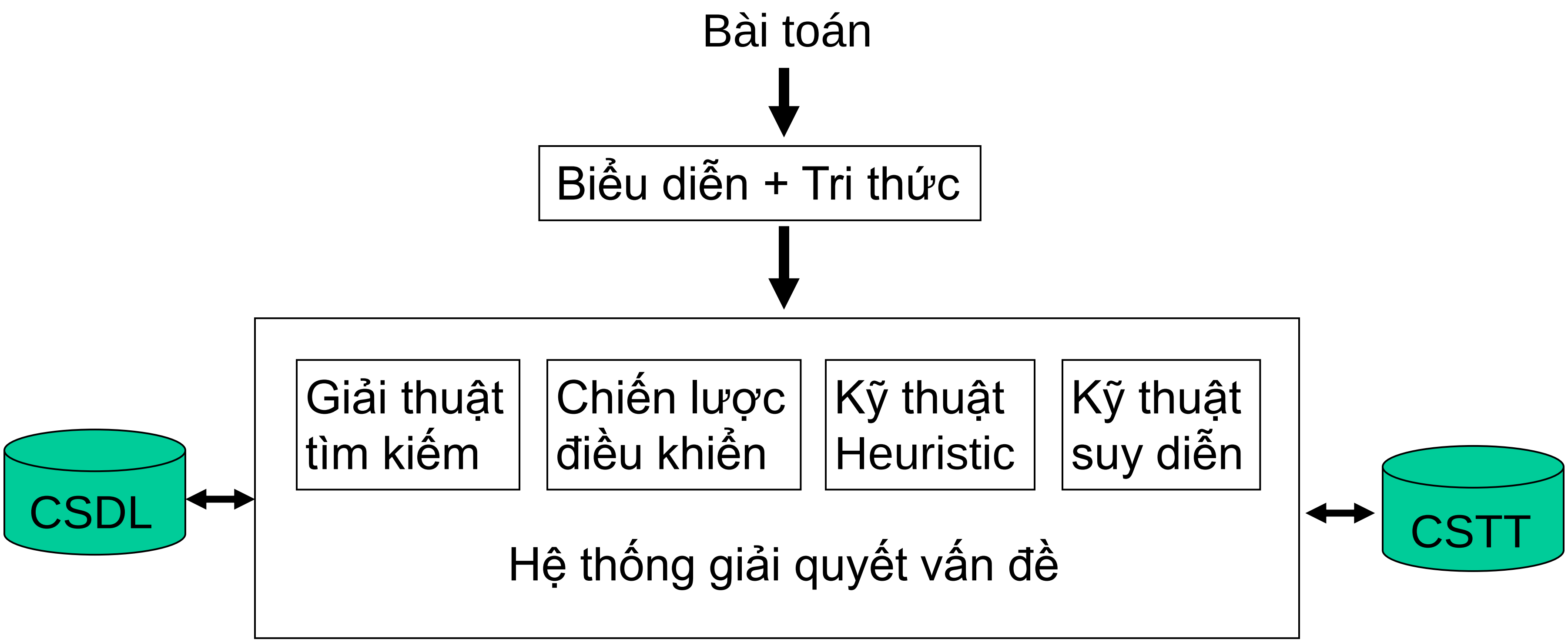
Ví dụ 5. Bài toán đố: Quan tòa - Hề - Trộm

- Có 3 người ngồi quanh 1 bàn tròn. Một người qua đường nghe thấy ba người này nói chuyện với nhau:
  - người 1 nói 2 là quan tòa
  - người 2 nói 3 là hề
  - người 3 nói 1 là trộm
- Biết rằng:
  - hề luôn nói đùa
  - quan tòa nói thật
  - trộm nói dối
- Hỏi ai là ai?

3.1.2. Các đặc trưng cơ bản của vấn đề

- Bài toán có thể phân rã?
- Không gian bài toán có thể đoán trước?
- Có tiêu chuẩn xác định lời giải tối ưu?
- Có cơ sở tri thức phi mâu thuẫn?
- Tri thức cần cho quá trình tìm kiếm hay để điều khiển?
- Có cần tương tác người – máy?

3.1.3. Những yếu tố cơ bản trong GQVĐ



Cấu trúc các hệ thống giải quyết vấn đề

3.1.4. Các phương pháp biểu diễn vấn đề

❶ Biểu diễn nhờ KGTT

- Mỗi hình trạng của bài toán tương ứng với 1 trạng thái (state)
- Mỗi phép biến đổi từ hình trạng này sang hình trạng khác tương ứng với các toán tử (operator)

❷ Qui bài toán về bài toán con

- Phân chia bài toán thành các bài toán con, các bài toán con lại được phân rã tiếp cho đến khi gặp được các bài toán sơ cấp cho phép xác định lời giải của bài toán ban đầu trên cơ sở lời giải của các bài toán con
- VD: phương pháp tinh dần từng bước trong công nghệ lập trình

3.1.4. Các phương pháp biểu diễn vấn đề

❸ Sử dụng logic hình thức

Khi giải quyết bài toán, phải tiến hành phân tích logic để thu gọn quá trình tìm kiếm, nhiều khi chứng minh được không có lời giải.

- logic mệnh đề
- logic vị từ cấp 1

cho phép:

- kiểm tra điều kiện kết thúc trong tìm kiếm đối với KGTT kiểm tra tính áp dụng được của các toán tử
- Chứng minh không tồn tại lời giải
- Mục đích: CM 1 phát biểu nào đó trên cơ sở những tiền đề và luật suy diễn đã có.

3.1.4. Các phương pháp biểu diễn vấn đề

❹ Lựa chọn phương pháp biểu diễn thích hợp nhằm:

- chia để trị
- tinh lọc thông tin
- tận dụng các phương pháp giải đã có
- phát biểu mới có thể thể hiện 1 vài tương quan nào đó giữa các yếu tố trong bài toán nhằm thu gọn quá trình giải

3.1.4. Các phương pháp biểu diễn vấn đề

❺ Biểu diễn trong máy

- dùng bảng/mảng (array): ví dụ, trò chơi n^2-1 số

Trạng thái đầu

11	14	4	7
10	6		5
1	2	13	15
9	12	8	3

Trạng thái đích

1	2	3	4
5	6	7	8
9	10	11	12
13	14	15	

$$A = (a_{ij}) = \begin{cases} 4(i-1) + j & (i, j) \neq (4,4) \\ 0 & (i, j) = (4,4) \end{cases}$$

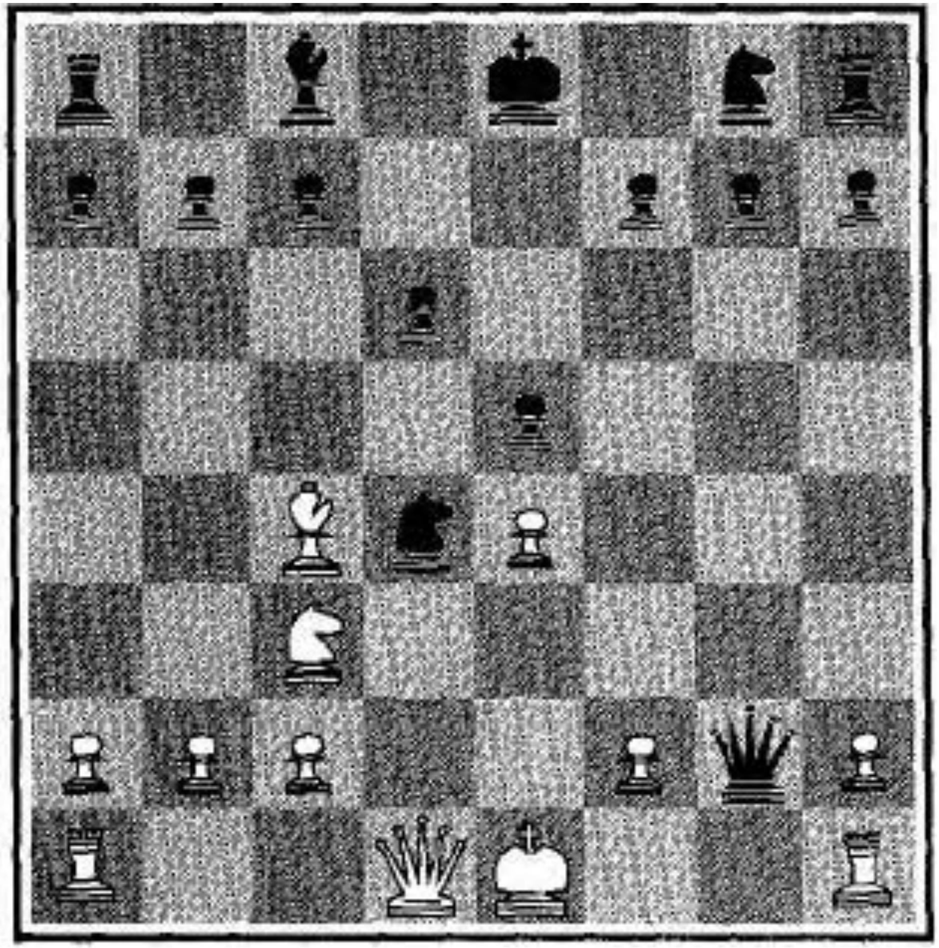
3.1.4. Các phương pháp biểu diễn vấn đề

❺ Biểu diễn trong máy

- dùng xâu ký hiệu

Ví dụ: bàn cờ Châu Âu

"T, XD, x , TgD, x , VD , x , MD, XD ,
ToD,ToD,ToD, x , x ,ToD,ToD,ToD,
x , x , x ,ToD, x , x , x , x ,
x , x , x , x ,ToD, x , x , x ,
x , x ,TgT, MD ,ToT, x , x , x ,
x , x , MT, x , x , x , x , x ,
ToT,ToT ,ToT, x , x ,ToT, HD,ToT,
XT , x , x , HT , VT, x , x , XT ."



x: ô trống, T: quân trắng đến lượt đi
XD: xe đen, TgD: tượng đen, VD: vua đen
MD: mã đen, ToD: tốt đen, HD: hậu đen
TgT: tượng trắng, ToT: tốt trắng, MT: mã trắng,
XT:xe trắng, HT: hậu trắng, VT: vua trắng

3.1.4. Các phương pháp biểu diễn vấn đề

❺ Biểu diễn trong máy

- dùng cấu trúc danh sách

Ví dụ: nghiệm của phương trình bậc 2

$$x_1 = \frac{-b + (b^2 - 4ac)^{1/2}}{2a}$$

Nội dung môn học

Chương 1. Tổng quan

Chương 2. Tác tử thông minh

Chương 3. Giải quyết vấn đề

- 3.1. Tổng quan
- 3.2. GQVĐ dựa trên tìm kiếm
- 3.3. Các kỹ thuật tìm kiếm cơ bản
- 3.4. Tìm kiếm có đối thủ
- 3.5. Tìm kiếm lời giải trên đồ thị Và/Hoặc

Chương 4. Tri thức và suy diễn

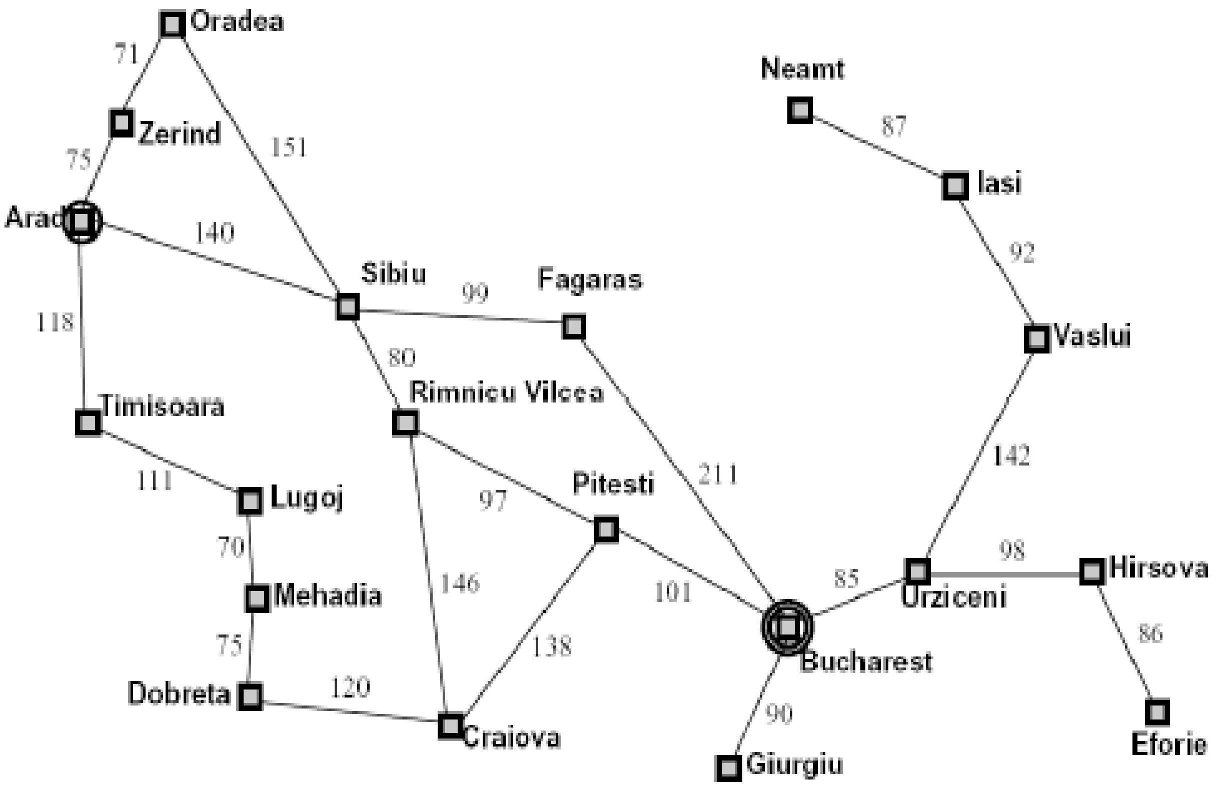
Chương 5. Học máy

3.2. GQVĐ dựa trên tìm kiếm

Để xây dựng các tác tử biết suy luận, ta cần sử dụng lý thuyết logic, xác suất, và tính hữu dụng. Các kỹ thuật tìm kiếm được nghiên cứu trước hết vì:

- Tìm kiếm là vấn đề quan trọng trong TTNT:
  - Tìm chuỗi hành động nhằm tối đa kết quả trong tương lai (lập kế hoạch)
  - Tìm kiếm trong CSTT để tìm chỗi các hành động có thể thực hiện trong tương lai (suy luận logic, xác suất)
  - Tìm các mô hình phù hợp với các quan sát (trong học máy)
- Tìm kiếm là 1 trong những thành công của các nghiên cứu về TTNT giai đoạn đầu

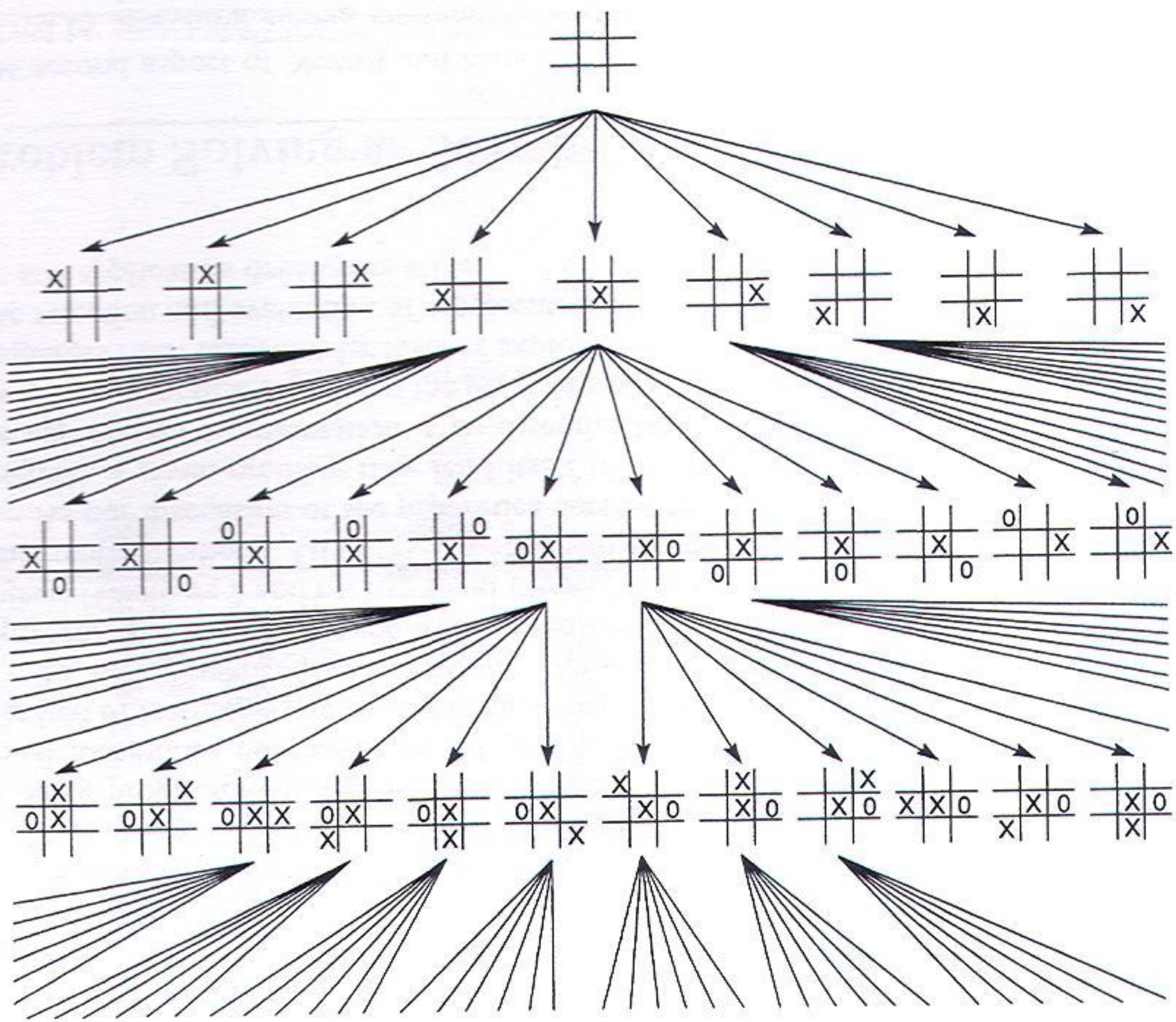
Bài toán tìm kiếm: Lập kế hoạch đường đi



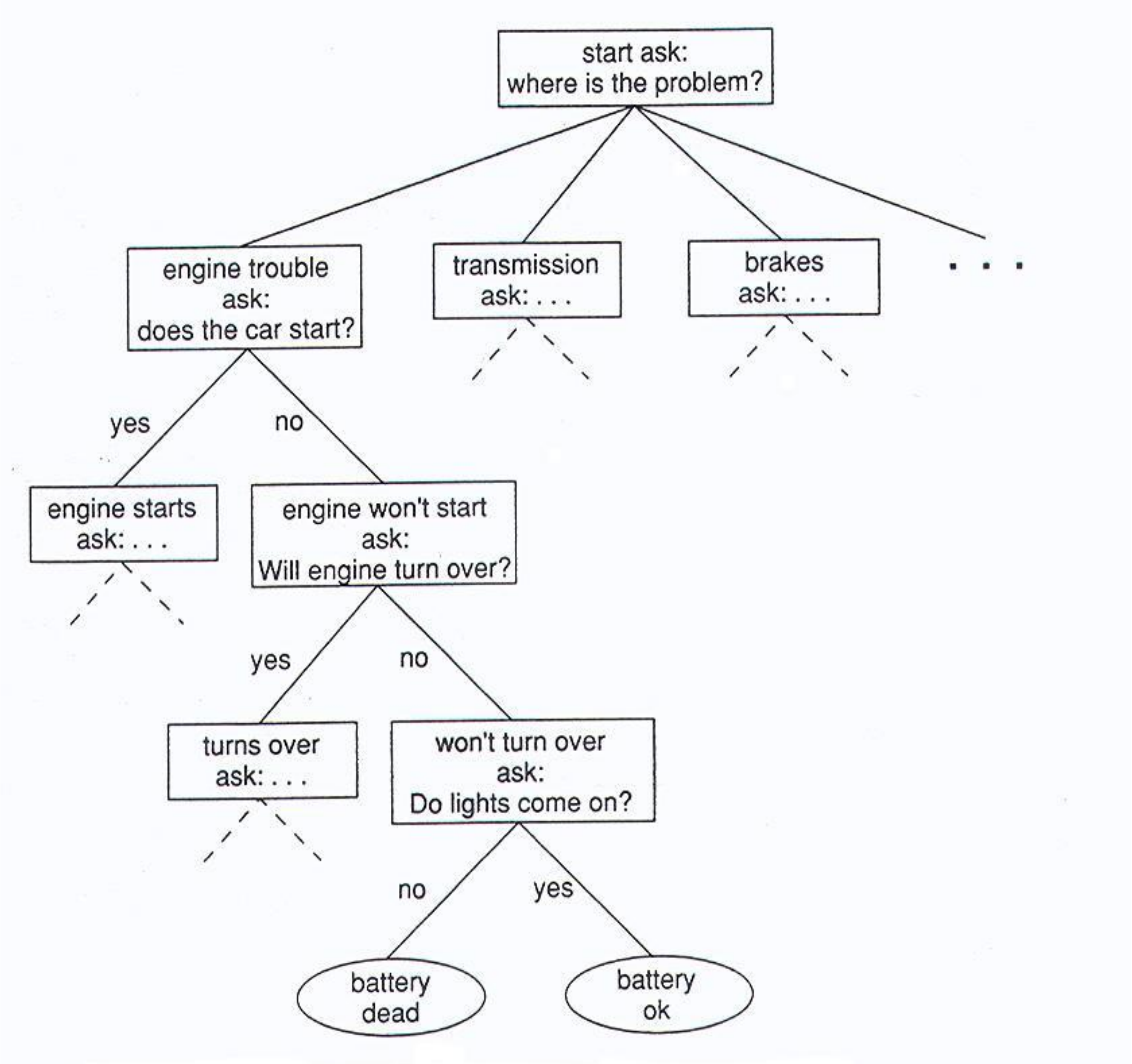
Kết quả: đi từ Arad đến Bucharest trong thời gian ngắn nhất

Môi trường: bản đồ với các thành phố, đường, và thời gian đi giữa 2 thành phố

KGTT của
Trò Chơi Tic-Tac-Toe

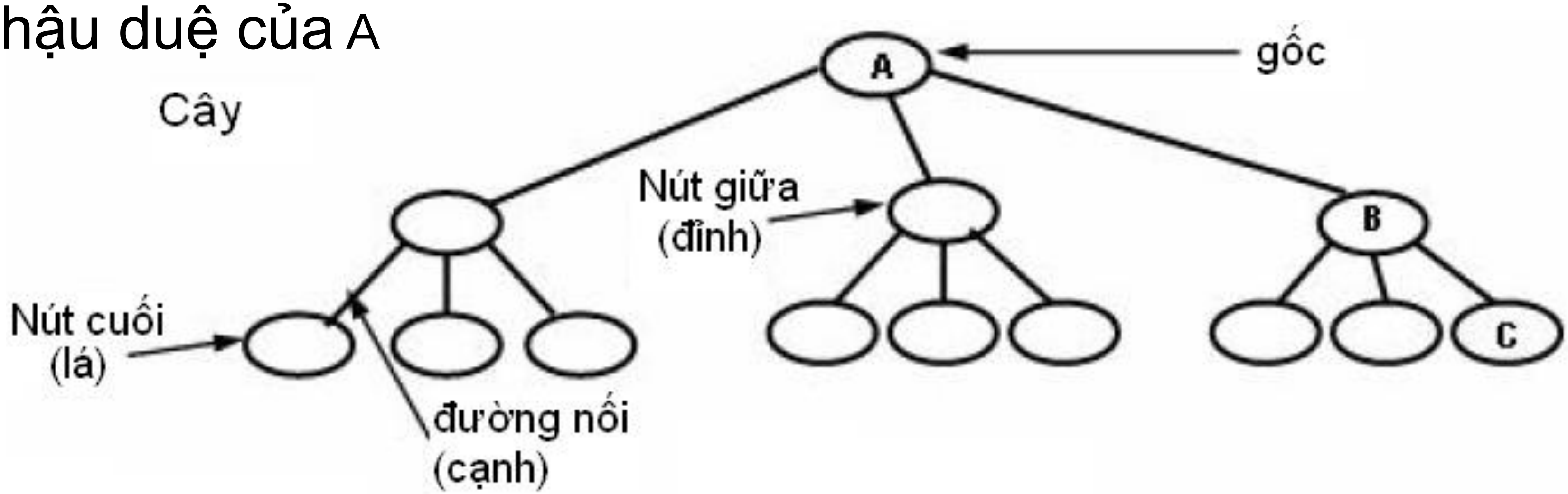


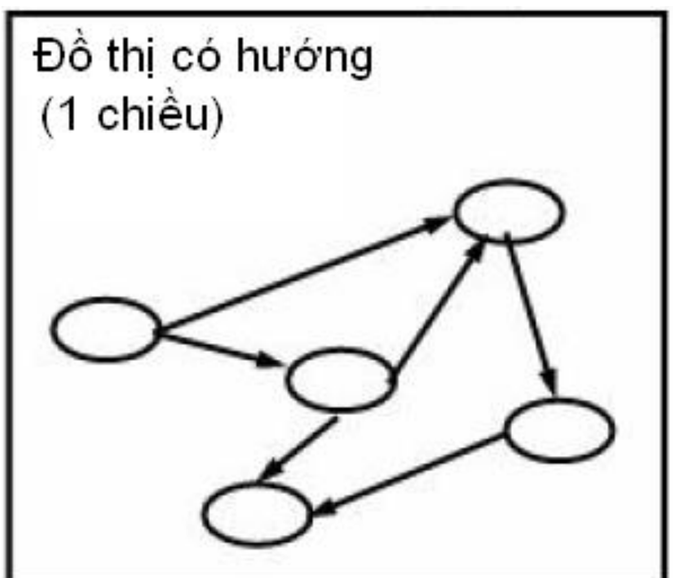
Chẩn đoán trục trặc máy móc trong ô tô

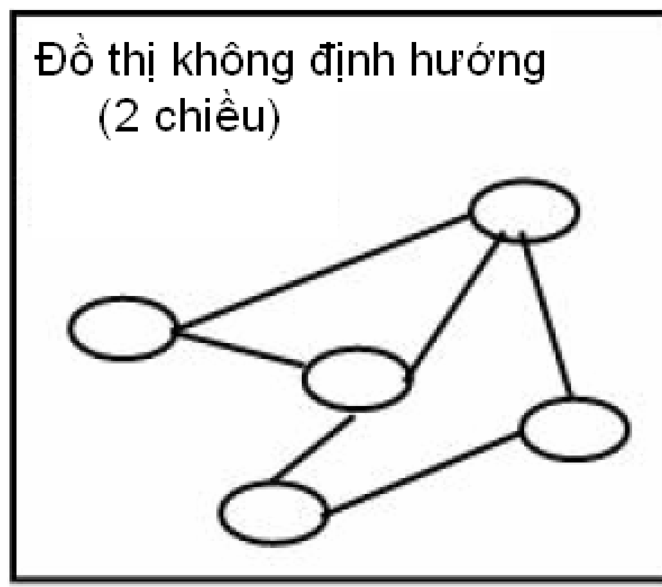


B là cha của C
C là con của B
A là tổ tiên của C
C là hậu duệ của A

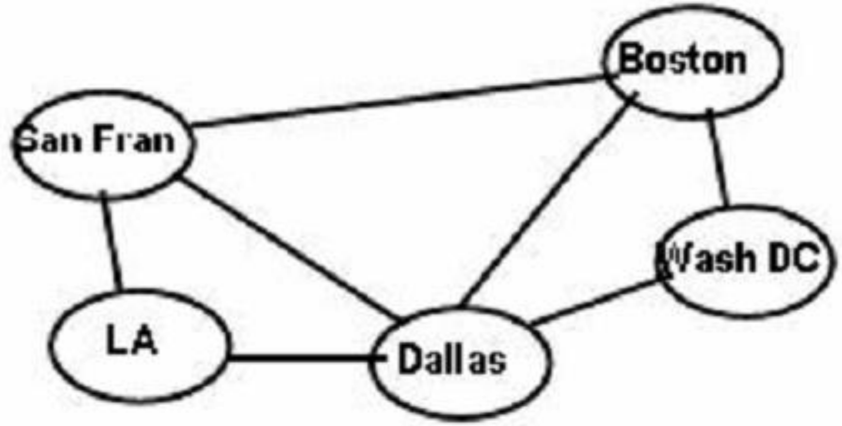
Cây và đồ thị





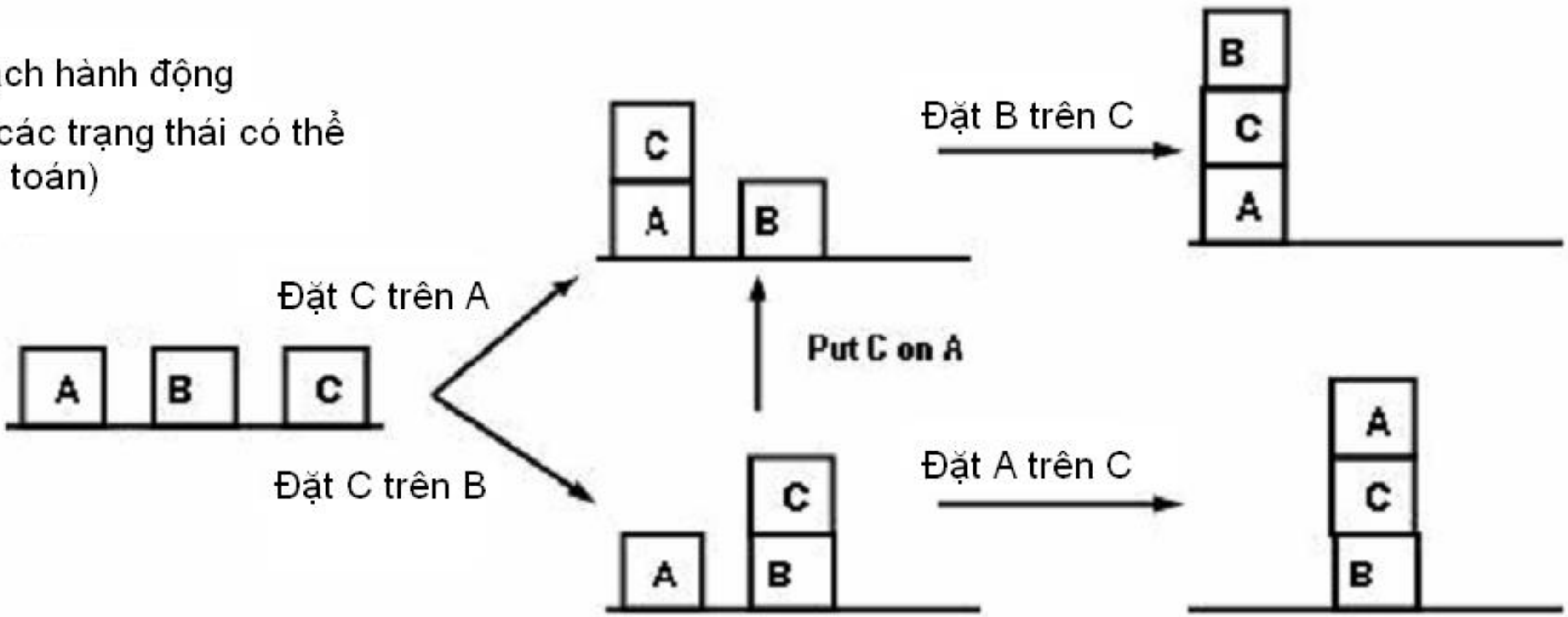


Ví dụ về đồ thị



Lộ trình đường bay

Kế hoạch hành động
(đồ thị các trạng thái có thể của bài toán)



3.2.1 Biểu diễn bài toán trong không gian tìm kiếm

Phát biểu bài toán P1:

- Cho trạng thái đầu s_0
- Cho tập trạng thái đích ĐICH
- Tìm dãy trạng thái $s_0, s_1, \ldots, s_n$ sao cho
  - $s_n \in$ĐICH và
  - $\forall i$: $s_i \rightarrow s_{i+1}$ nhờ áp dụng toán tử biến đổi

- Giá đường đi: (cộng gộp)
  - ví dụ, tổng khoảng cách, số lượng hành động đã thực hiện, …
  - c(x, a, y) là giá 1 bước, ≥ 0

- Để biểu diễn phép biến đổi trạng thái, có 2 cách viết:
  1. Cách viết dùng luật sản xuất,
     - VD: VT → VP
     - Bài toán Tháp Hà Nội

  2. Cách viết dùng ký hiệu hàm
     - VD: B = f(A)
     - Bài toán n^2-1 số

3.2.1 Biểu diễn bài toán trong không gian tìm kiếm

Phát biểu lại bài toán P1 (bài toán P2):

1. Tìm dãy trạng thái $s_0, s_1, \ldots, s_n$ sao cho
   - $s_n \in$ ĐICH và
   - $\forall i: s_i \rightarrow s_{i+1}$ (hay $\exists$ toán tử biến đổi O: $O(s_i) = s_{i+1}$)
2. Tìm dãy toán tử $O_1, \ldots, O_{n-1}, O_n$ sao cho:

   $O_n(O_{n-1}(\ldots O_1(s_0)..)) = s_n \in$ ĐICH

   hay tìm dãy sản xuất $p_1, \ldots, p_n$ sao cho

   $s_0 \Rightarrow^{p1} s_1 \Rightarrow \ldots \Rightarrow^{pn} s_n \in$ ĐICH

3.2.2. Các chiến lược tìm kiếm lời giải

VD1: Bài toán rót nước

What: A(m),B(n). Đầu: (0,0). Đích (k,*) U (*,k)

How: Thao tác rót: A → B,…

Điều kiện: không tràn, đổ hết

Biểu diễn sản xuất: (x,y) → (x', y')

m = 6, n = 5, k = 2:

6 – 5 = 1

2*6 - 2*5 = 2; 4*5 - 3*6 = 2.

USCLN(m,n)=d. Nếu k không chia hết cho d → not OK

3.2.2. Các chiến lược tìm kiếm lời giải

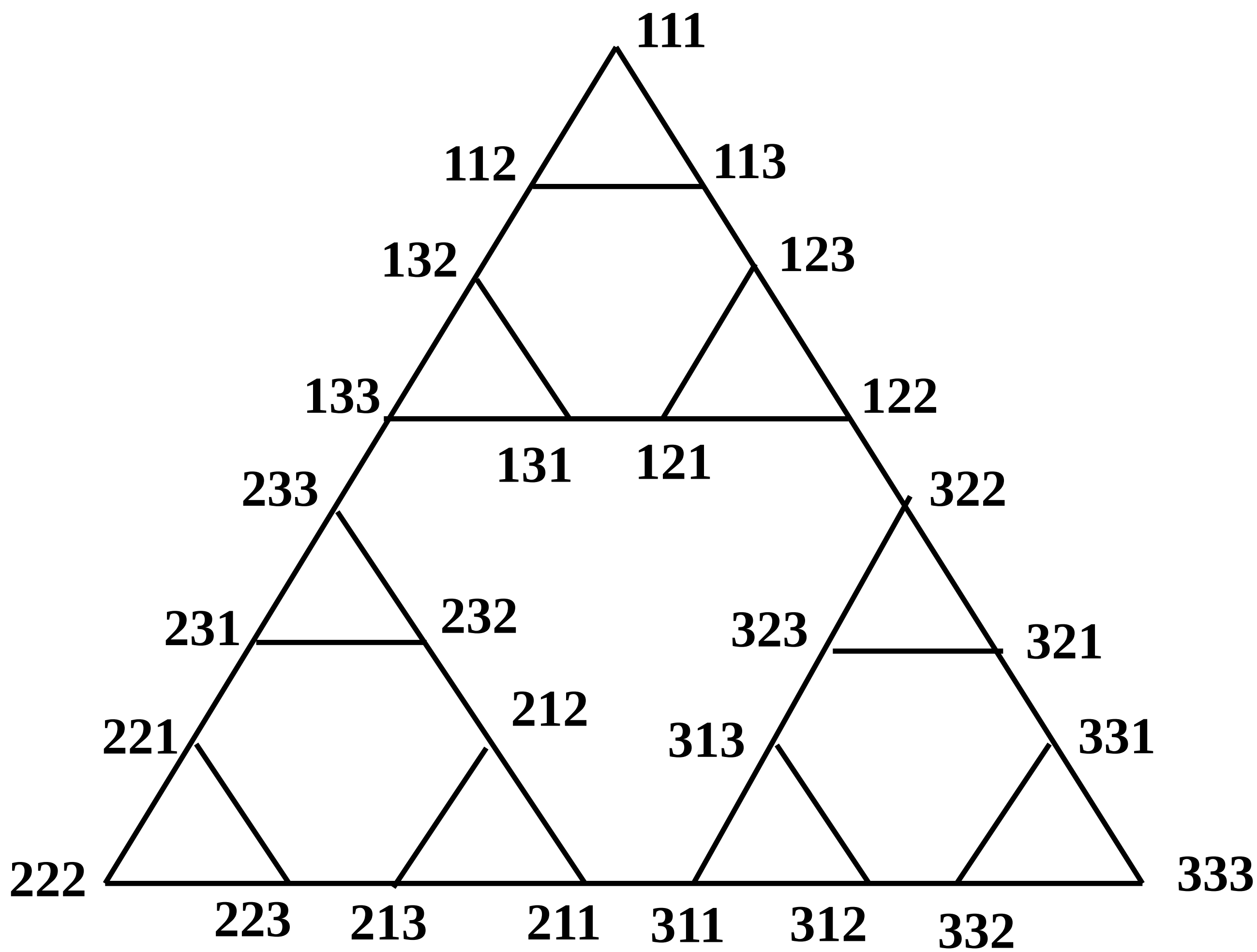
VD2: Bài toán Tháp Hà Nội, n=3

(i , j , k) Nếu i, j, k là 3 cọc riêng biệt

C B A i + j + k = 6

```
Procedure Thap(n,i,j: integer);
//nhấc n đĩa từ cọc i sang cọc j
Var k: interger;
Begin   k = 6 – i – j;
        if n=1 then Nhac(i,j)
       else begin    Thap(n-1,i,k);
                     Nhac(i,j);
                     Thap(n-1,k,j);
            end;
End;
```

Không gian trạng thái của bài toán Tháp Hà Nội



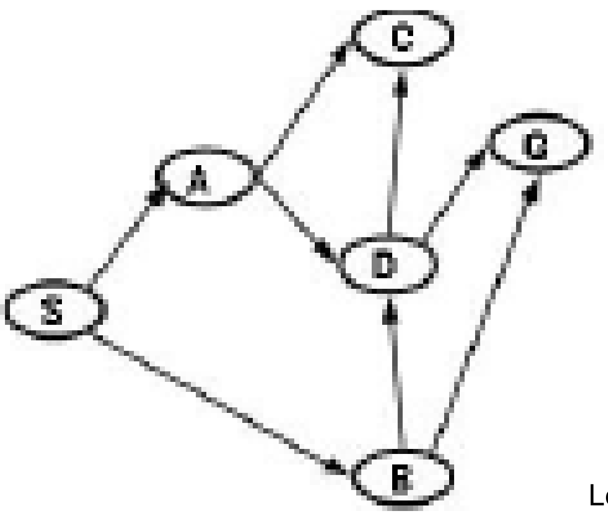
Biểu diễn bằng đồ thị

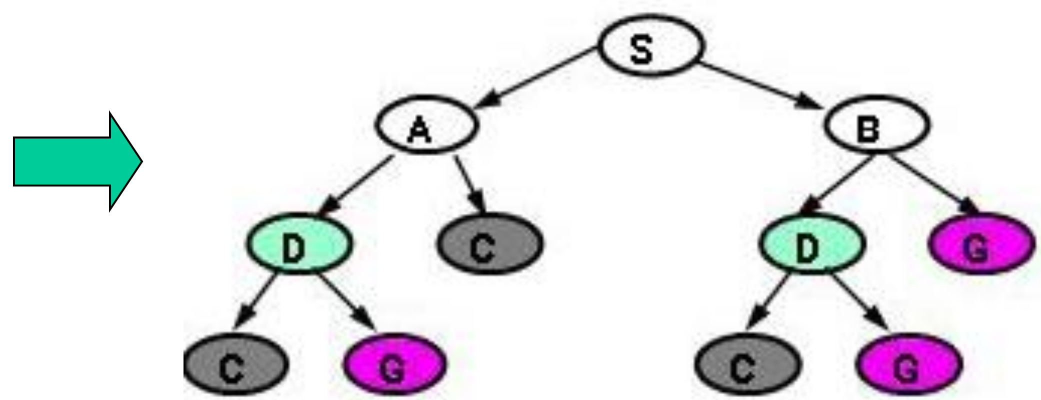
Đồ thị G là cặp G = (N,A) với N - tập các nút, A - tập các cung và với $\forall n \in N$: $\Gamma(n) = \{m \in N \mid (n,m) \in A\}$

KGTT	Đồ thị
• Trạng thái (đầu, đích) S, ababb	• nút (đầu, đích)
• Toán tử (sản xuất) $S \rightarrow Sa$	• cung
• Dãy trạng thái liên tiếp	• đường đi
• Dãy toán tử $S \rightarrow Sa \rightarrow aBa$	
• Bài toán P_1, P_2	• Tìm đường đi trên đồ thị từ đỉnh đầu n_0 (tương ứng với s_0) tới đỉnh ĐICH

Chuyển bài toán tìm kiếm trên đồ thị về tìm kiếm trên cây

- Cây là đồ thị có hướng không có chu trình và các nút có <= 1 nút cha.
- Chuyển TK trên đồ thị về TK trên cây:
  - thay các liên kết không định hướng bằng 2 liên kết có hướng
  - tránh các vòng lặp trên đường (sử dụng biến tổng thể để lưu vết các nút đã thăm)





Các đặc tính tìm kiếm

- **Tính đầy đủ**
  – Khi bài toán có lời giải thì giải thuật tìm kiếm có thể tìm thấy lời giải không?
- **Thời gian**
  – Thời gian cần thiết để tìm thấy lời giải
- **Không gian**
  – Dung lượng nhớ cần thiết để tìm thấy lời giải
- **Sự tối ưu**
  – Khi có hàm giá, giải thuật có đảm bảo tìm được lời giải tối ưu không?

Nội dung môn học

Chương 1. Tổng quan

Chương 2. Tác tử thông minh

Chương 3. Giải quyết vấn đề

- 3.1. Tổng quan
- 3.2. GQVĐ dựa trên tìm kiếm
- 3.3. Các kỹ thuật tìm kiếm cơ bản
- 3.4. Tìm kiếm có đối thủ
- 3.5. Tìm kiếm lời giải trên đồ thị Và/Hoặc

Chương 4. Tri thức và suy diễn

Chương 5. Học máy

3.3. Các kỹ thuật tìm kiếm cơ bản

Lớp	Tên	Thao tác
Bất kì 0 biết giá	TK sâu TK rộng	Khám phá có hệ thống toàn bộ cây đến khi tìm thấy đích
Tối ưu Biết giá	TK cực tiểu	Sử dụng độ đo là độ dài đường đi, tìm đường đi ngắn nhất
Tối ưu Biết giá	TK cực tiểu *	Sử dụng độ đo là độ dài đường đi và mẹo, tìm đường đi ngắn nhất

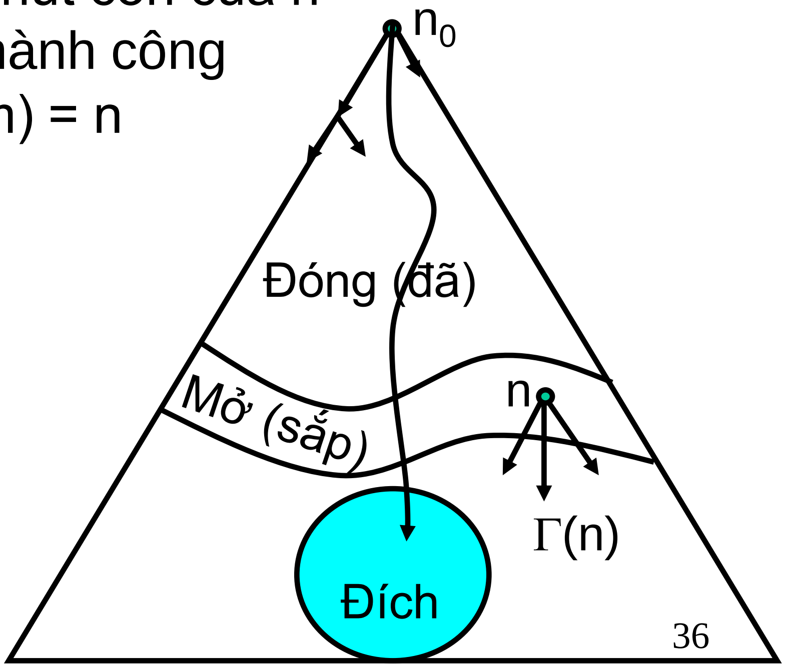
Thuật toán tìm kiếm cơ bản

Xây dựng tập Mở - tập các đỉnh sắp duyệt
Đóng - tập các đỉnh đã duyệt
n_0 - trạng thái đầu

1. Mở = $\{n_0\}$; Đóng = $\varnothing$
2. Chọn $n \in$ Mở:
   Đóng = Đóng $\cup$ $\{n\}$
   Mở = Mở $\cup$ $\Gamma(n)$ // $\Gamma(n)$: tập các nút con của n
3. Lặp (2) đến khi gặp $n^* \in$ Đích $\Rightarrow$ thành công
4. Với mỗi $m \in \Gamma(n)$, thực hiện: cha(m) = n
   p' = g, cha(g), $cha^2(g)$,…,n_0
   p = inverse(p')
   In đường đi

Các quyết định quan trọng:

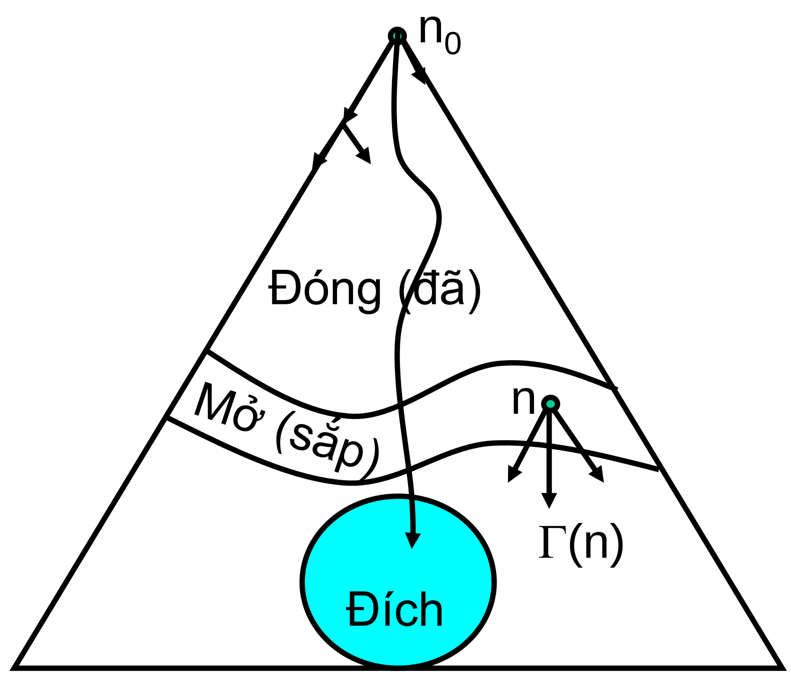
- Lấy $n \in$ Mở
- Bổ sung $\Gamma(n)$ vào Mở



Cài đặt các chiến lược tìm kiếm

Các quyết định quan trọng:

- Lấy n $\in$ Mở
- Bổ sung Γ(n) vào Mở



- Tìm kiếm sâu (Depth-first):

  Vào sau ra trước

  (LIFO – Last In First Out)

  vào → Γ(n) Mở
  ra ← n

- Tìm kiếm rộng (Breadth-first):

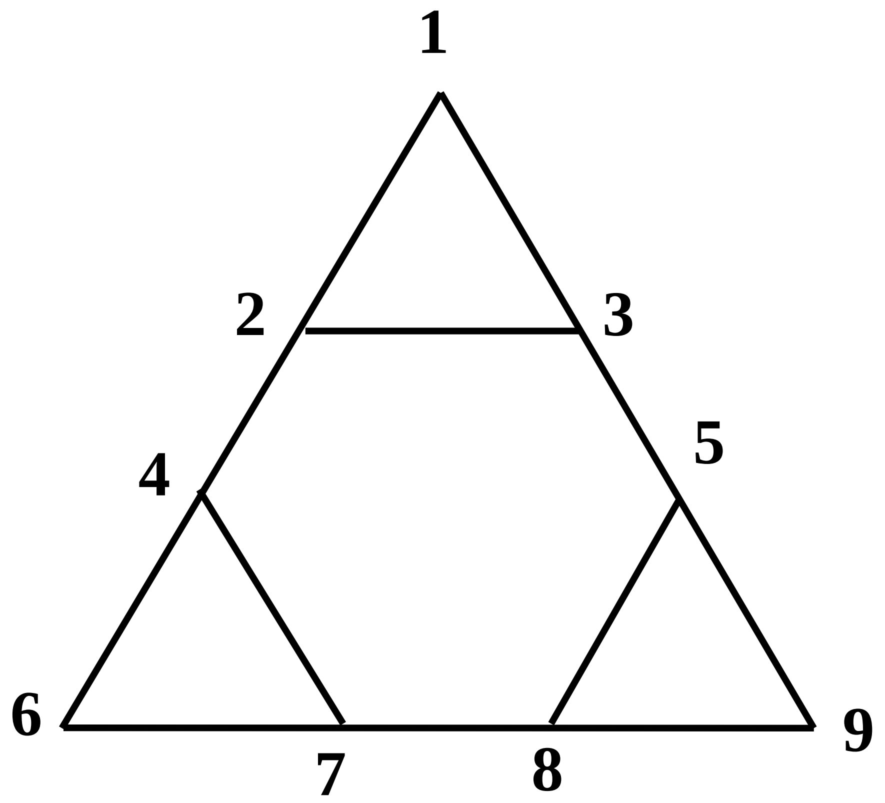
  Vào trước ra trước

  (FIFO – First In First Out)

  ra ← n Mở Γ(n) ← vào

- Tìm kiếm cực tiểu (Uniform-cost):

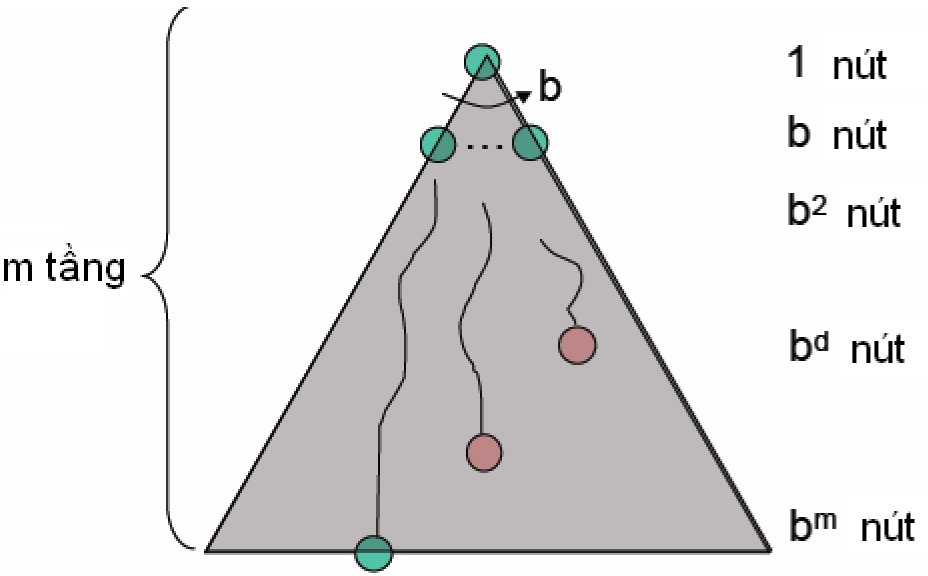
  Lấy phần tử có giá nhỏ nhất dựa trên hàm giá


1
2
3
5
4
6
9
7
8

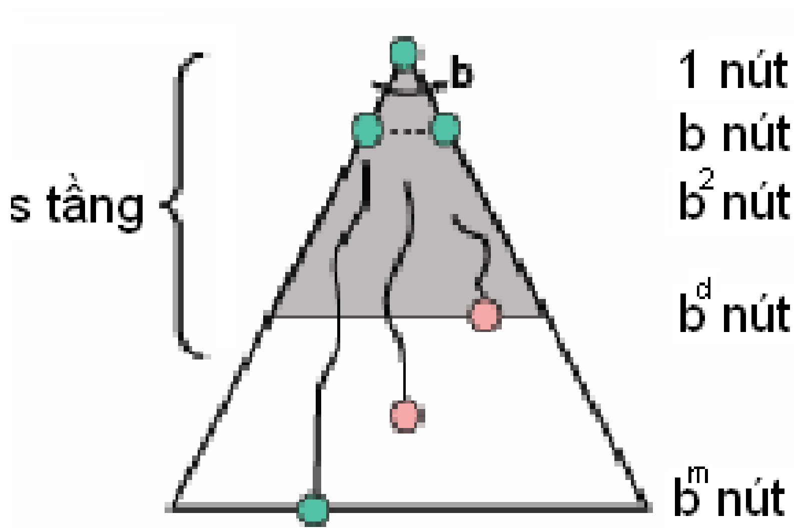
Tìm Kiếm Sâu hay Rộng?

- Có cần thiết tìm một *đường đi ngắn nhất* đến mục tiêu hay không?
- Sự *phân nhánh* của không gian trạng thái
- Tài nguyên về *không gian* và *thời gian* sẵn có
- *Khoảng cách trung bình* của đường dẫn đến trạng thái mục tiêu.
- Yêu cầu đưa ra *tất cả các lời giải* hay chỉ là lời giải tìm được đầu tiên.

Tìm kiếm sâu



Tìm kiếm rộng



Thuật toán	Hoàn thiện	Tối ưu	Thời gian	Không gian
TKS	không	không	$O(b^m)$	$O(bm)$
TKR	có	không	$O(b^{d+1})$	$O(b^{d+1})$

$$= 1 + b + b^2 + \ldots + b^d + b(b^d-1) = O(b^{d+1})$$

Tìm kiếm sâu dần

- TKS có thể cho kết quả nhưng đường đi không phải là ngắn nhất
- Tuy có $\exists$ 1 đường đi đến Đích nhưng TKS có thể không dừng.

$\Rightarrow$ chọn ngưỡng sâu D, mỗi đỉnh được gán một ngưỡng sâu d(n)

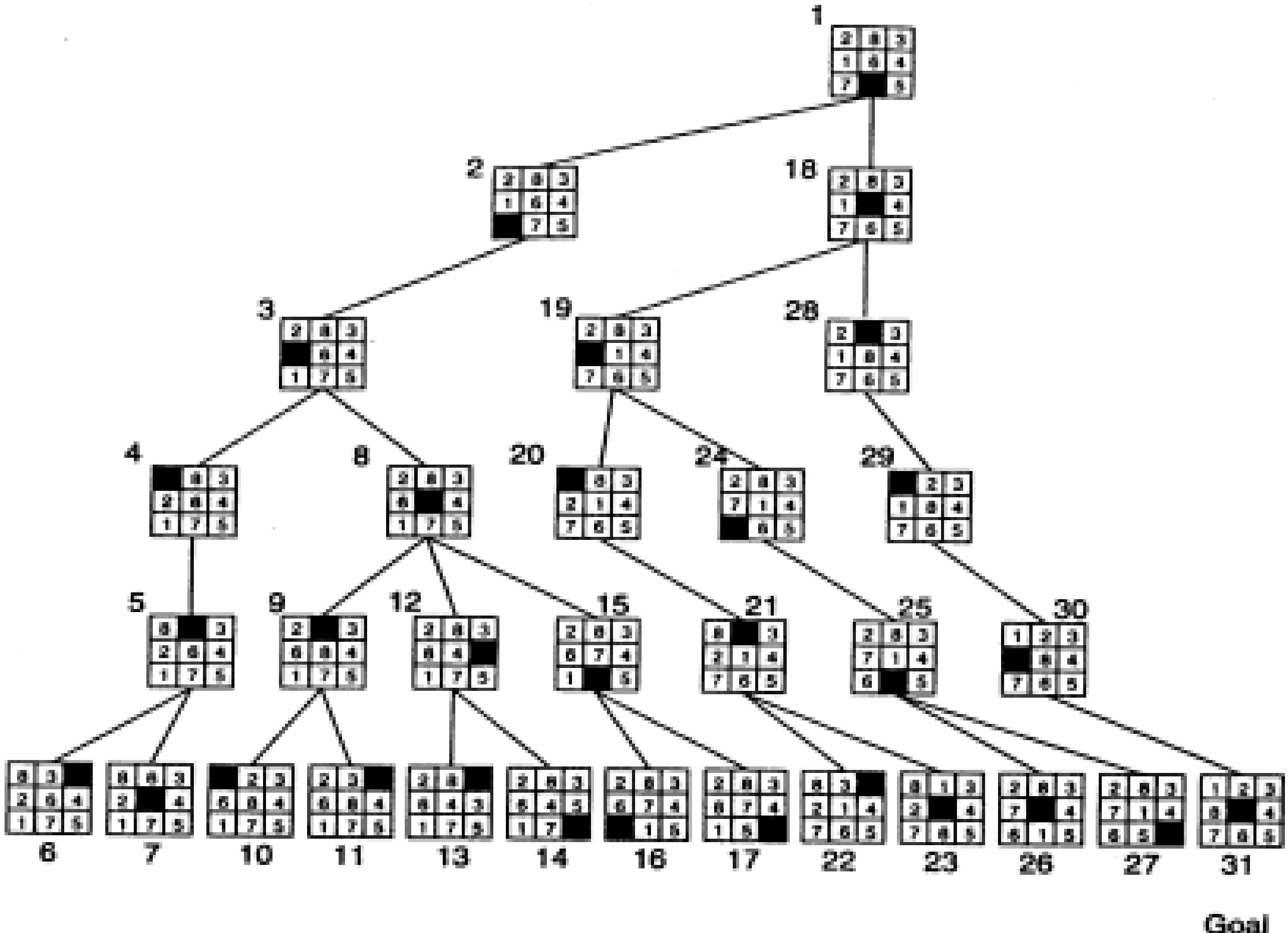
Lấy n $\in$ Mở, d(n) $\leq$ D

- Vấn đề
  - Nếu điểm đích n* có d(n*) > D?

$\Rightarrow$ Tìm kiếm sâu dần

Thuật toán	Hoàn thiện	Tối ưu	Thời gian	Không gian
TKSD	có	không	$O(b^d)$	O(bm)

Trò chơi ô đố 8-puzzle với ngưỡng sâu 5



Tìm kiếm cực tiểu

$c(n_i, n_j)$: chi phí đi từ n_i đến n_j

Xét $p = n_0, n_1, \ldots, n_k$

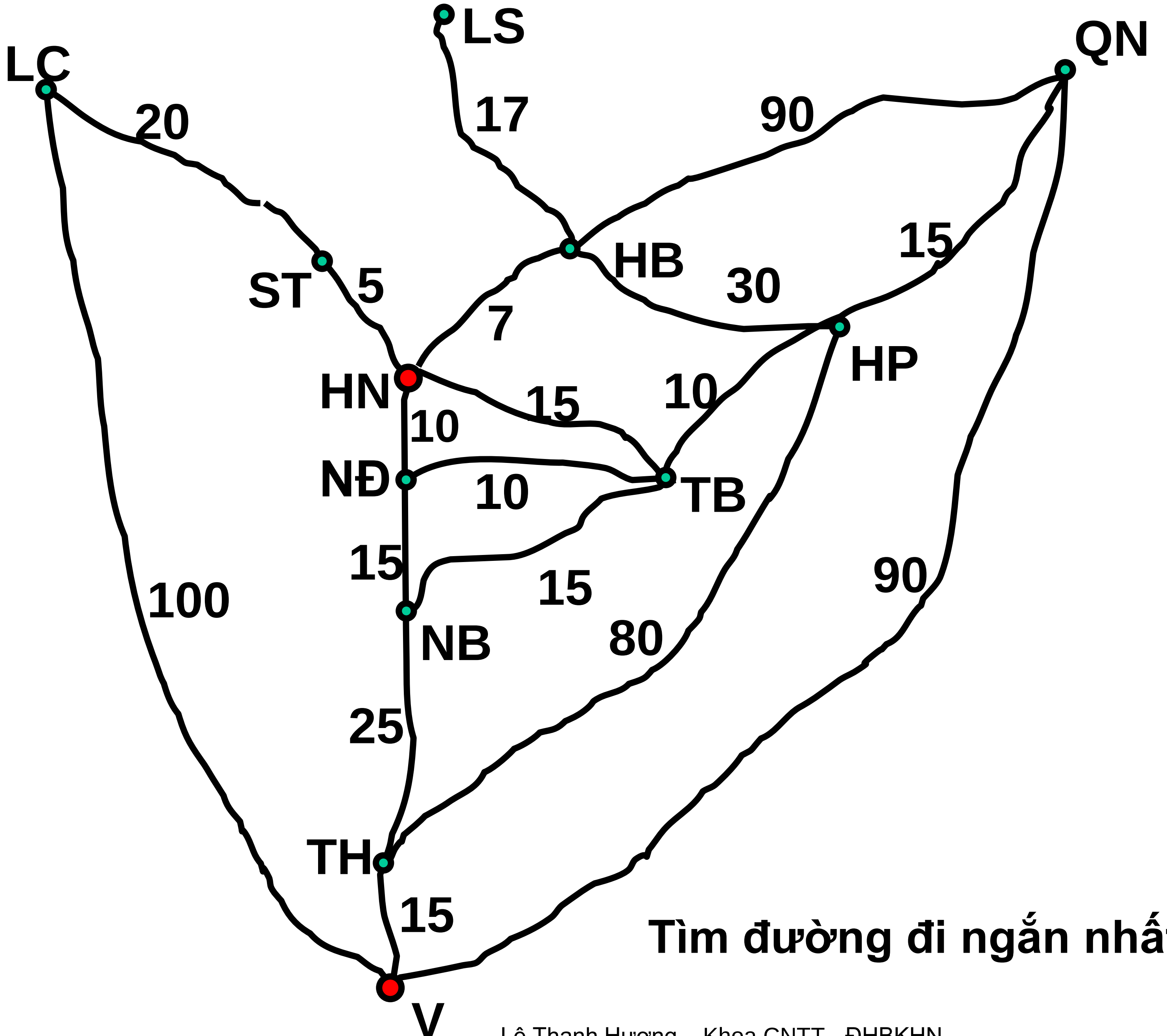
Hàm đánh giá

$c(p) = c(n_0,n_1) + c(n_1,n_2) + \ldots + c(n_{k-1},n_k)$

Lấy $n \in$ Mở: $g(n) = c(p(n_0,n))$ min

Nếu $\forall c(n_i,n_j) > \varepsilon$, C* là chi phí của lời giải tối ưu

Thuật toán	Hoàn thiện	Tối ưu	Thời gian	Không gian
TKCT	có	có	$O(b^{ceiling(C^*/\varepsilon)})$	$O(b^{ceiling(C^*/\varepsilon)})$



Tìm đường đi ngắn nhất từ HN đến V

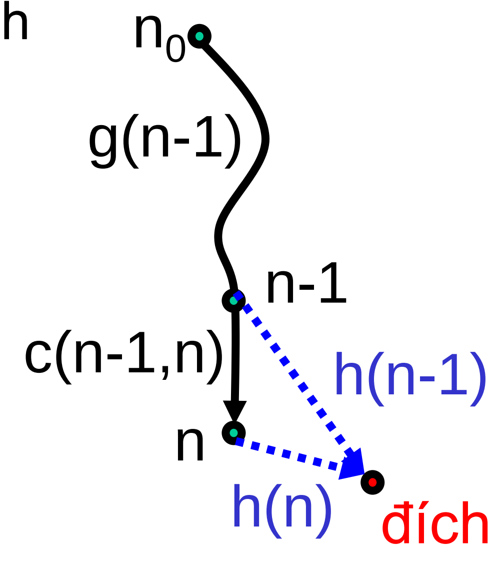
Tìm kiếm cực tiểu với tri thức bổ sung (A*)

$c(n_i, n_j)$ = chi phí đi từ n_i đến n_j

$g(n)$ = chi phí thực tế đường đi từ n_0 đến n

$h(n)$ = chi phí ước lượng đường đi từ n đến đích, do chuyên gia cung cấp

- $h(n)$ chấp nhận được nếu với $\forall n$, $0 \le h(n) \le h^*(n)$, trong đó $h^*(n)$ là chi phí thực để tới trạng thái đích từ n.
- $h(n)$ càng sát với $h^*(n)$ thì thuật toán càng mạnh



$f(n) = g(n) + h(n)$

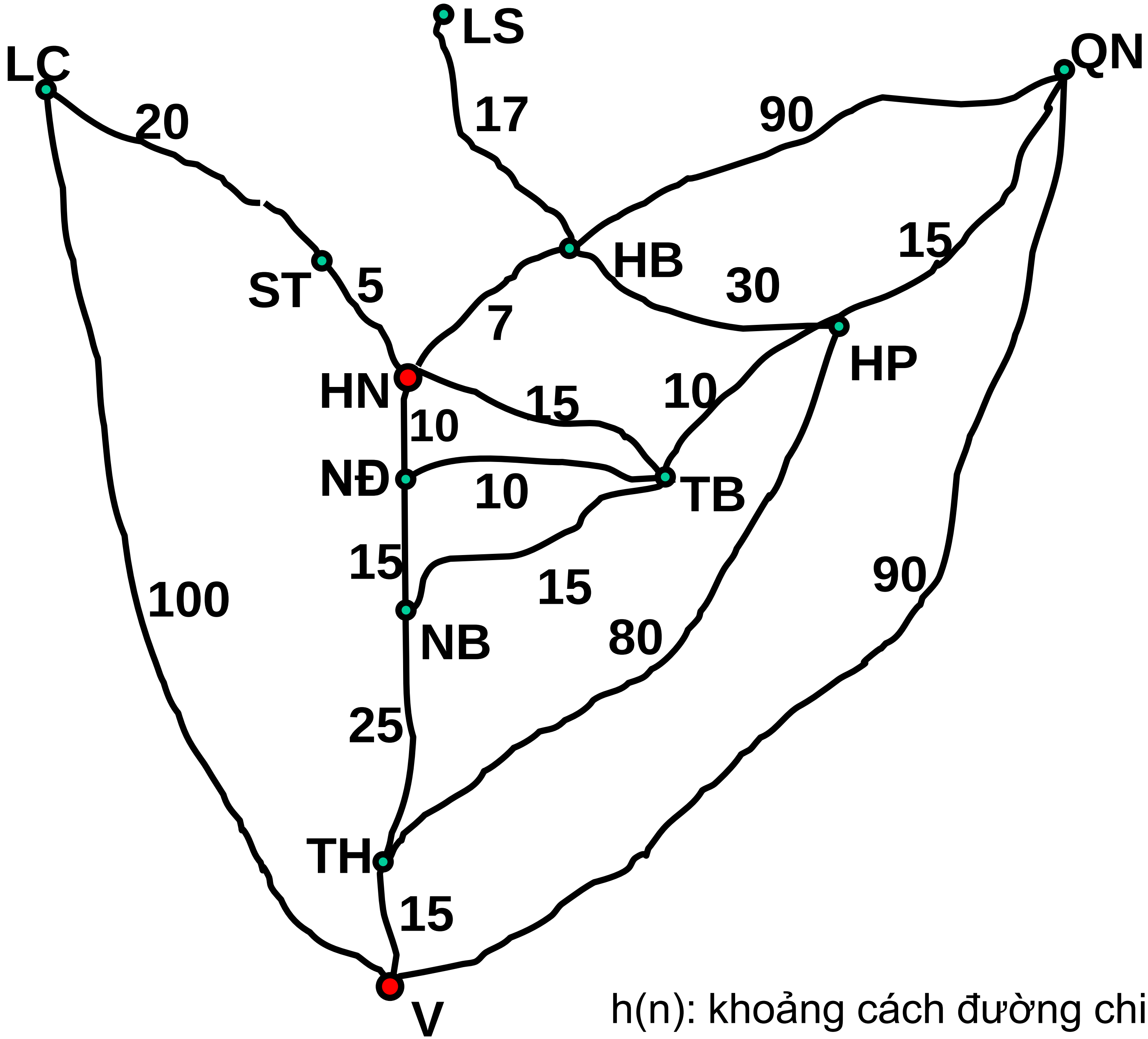
$f(n-1) = g(n-1) + h(n-1)$

$g(n) = g(n-1) + c(n-1,n)$

$f(n) = g(n-1) + c(n-1,n) + h(n)$

$\quad\;\; = f(n-1) - h(n-1) + c(n-1,n) + h(n)$

Lấy $n \in$ Mở: $f(n)$ min

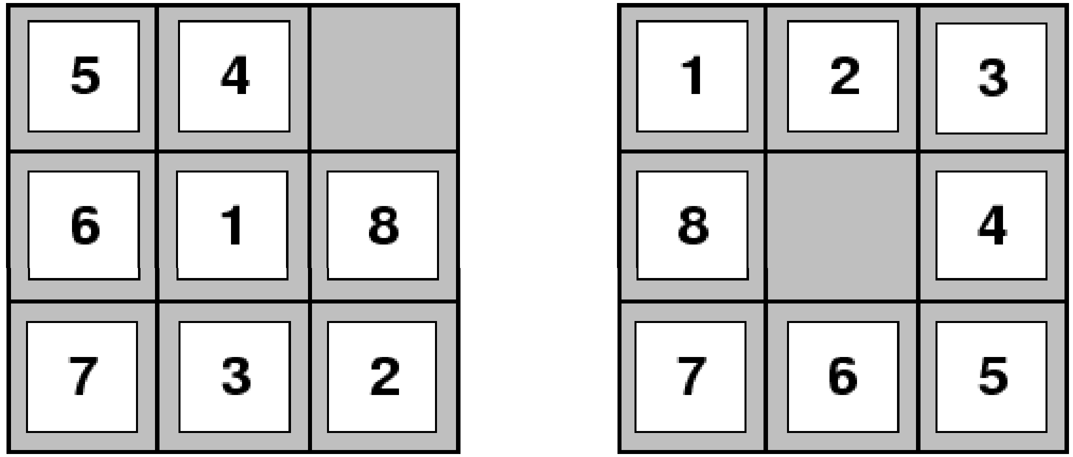


n	h(n)
HN	50
ST	60
LC	75
HB	65
LS	70
HP	80
QN	80
TB	55
NĐ	45
NB	20
TH	15
V	0

h(n): khoảng cách đường chim bay HN→ V

Một số dạng heuristic trong bài toán tìm kiếm

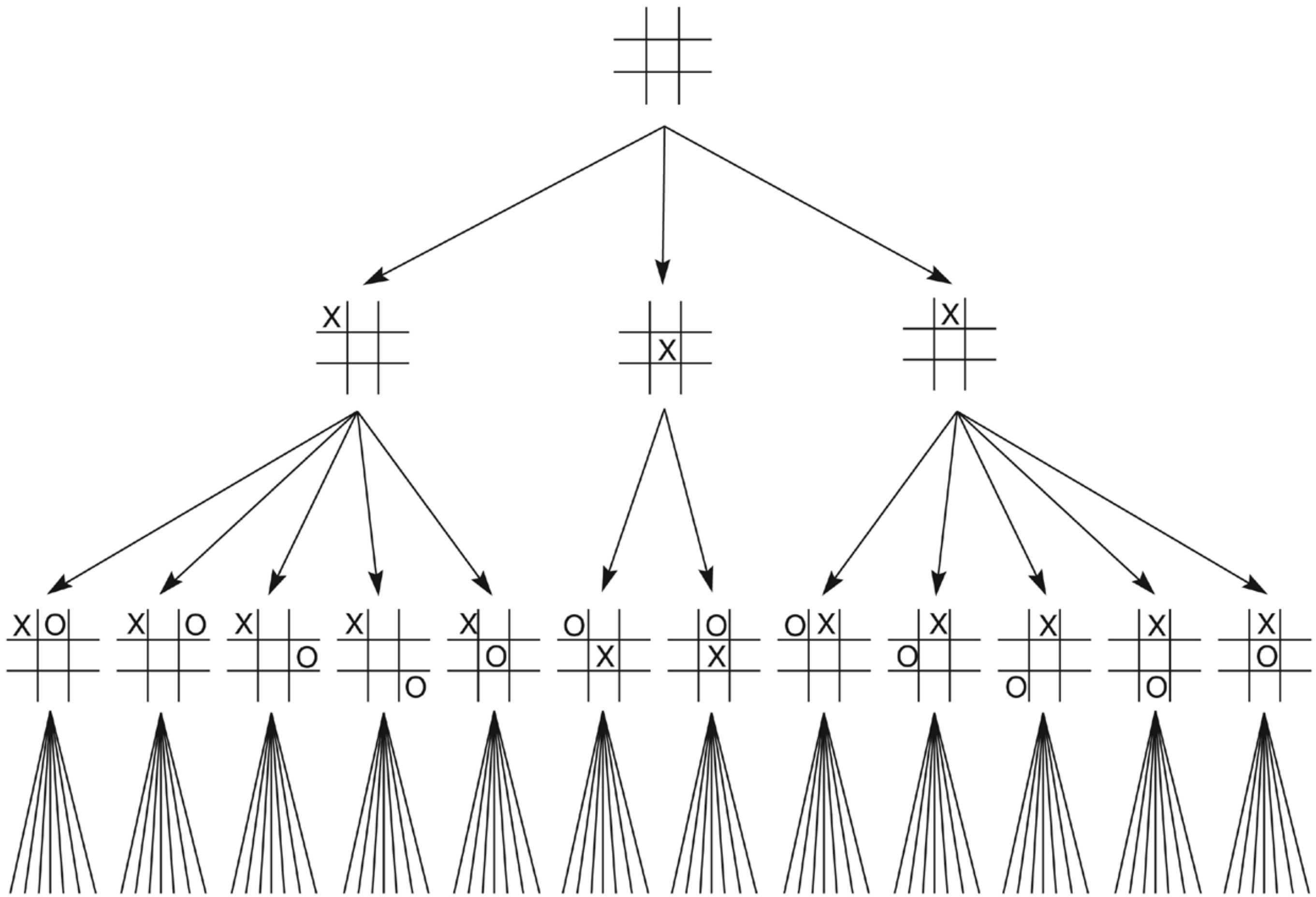
Bài toán đố 8 số



Start State Goal State

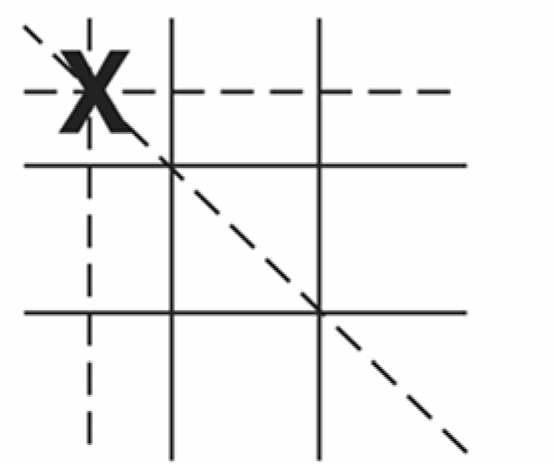
- Ví dụ về heuristic
  - Số viên sai vị trí
  - Khoảng cách Manhattan. (Khoảng cách Manhattan giữa (x_1,y_1) và (x_2,y_2) là $|x_1-x_2|+|y_1-y_2|$.
- H1(S) = 7
- H2(S) = 2+3+3+2+4+2+0+2 = 18

Trò chơi Tic-tac-toe

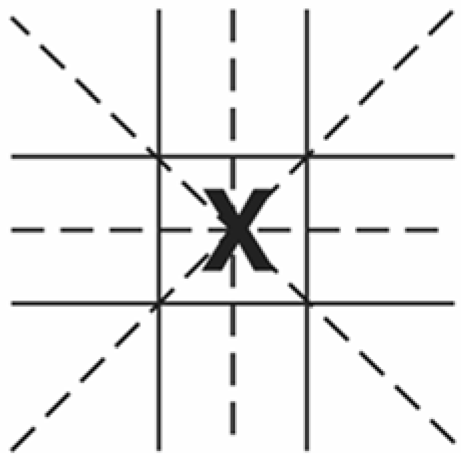


KGTT của tic-tac-toe được thu nhỏ nhờ tính đối xứng của các trạng thái

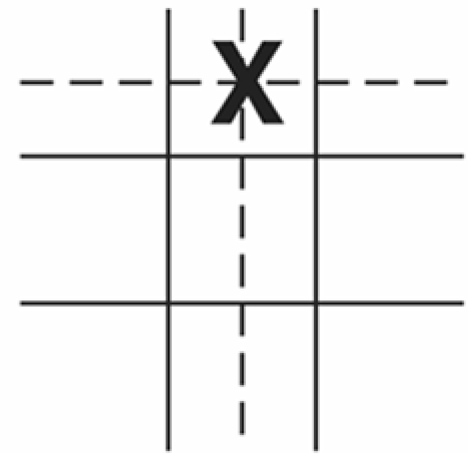
Phép đo heuristic



Chiếm 3 đường



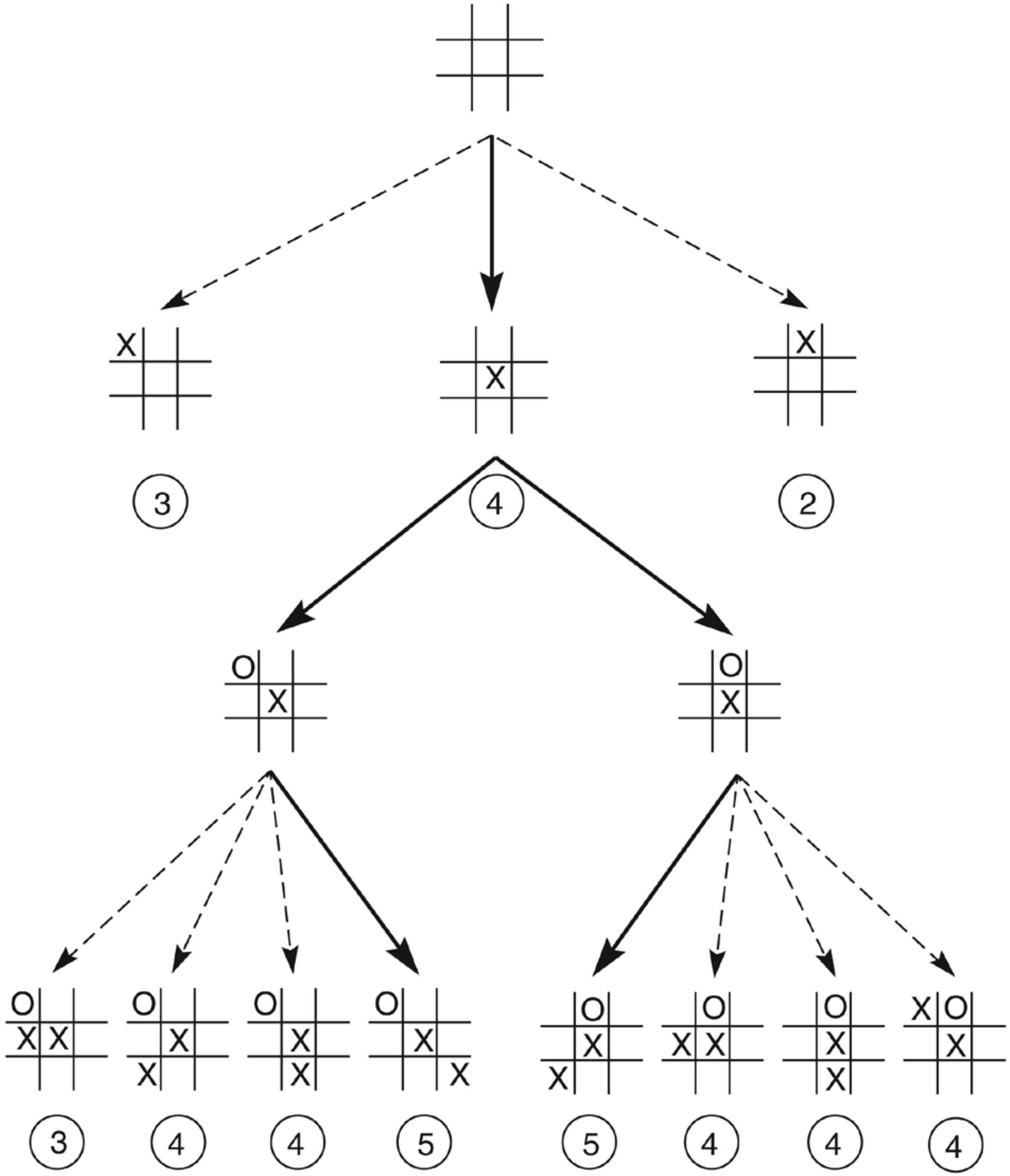
Chiếm 4 đường



Chiếm 2 đường

Heuristic "*Số đường thắng nhiều nhất*" áp dụng cho các nút con đầu tien trong tic-tac-toe.

Phép đo heuristic



Nội dung môn học

Chương 1. Tổng quan

Chương 2. Tác tử thông minh

Chương 3. Giải quyết vấn đề

- 3.1. Tổng quan
- 3.2. GQVĐ dựa trên tìm kiếm
- 3.3. Các kỹ thuật tìm kiếm cơ bản
- 3.4. Tìm kiếm có đối thủ
- 3.5. Tìm kiếm lời giải trên đồ thị Và/Hoặc

Chương 4. Tri thức và suy diễn

Chương 5. Học máy

3.4. Tìm kiếm có đối thủ

Trò chơi đối kháng MINIMAX : Có 2 đối thủ MAX và MIN

- MAX tìm cách làm cực đại 1 hàm ước lượng nào đó: Chọn nước đi ứng với GTLN
- MIN tìm cách làm cực tiểu và chọn nước đi ứng với GTNN

Ở mỗi thời điểm:

- Nếu 1 đỉnh ứng với nước đi của MAX thì giá trị của nó là GT cực đại của các đỉnh con.
- Nếu 1 đỉnh ứng với nước đi của MIN thì giá trị của nó là GT cực tiểu của các đỉnh con.

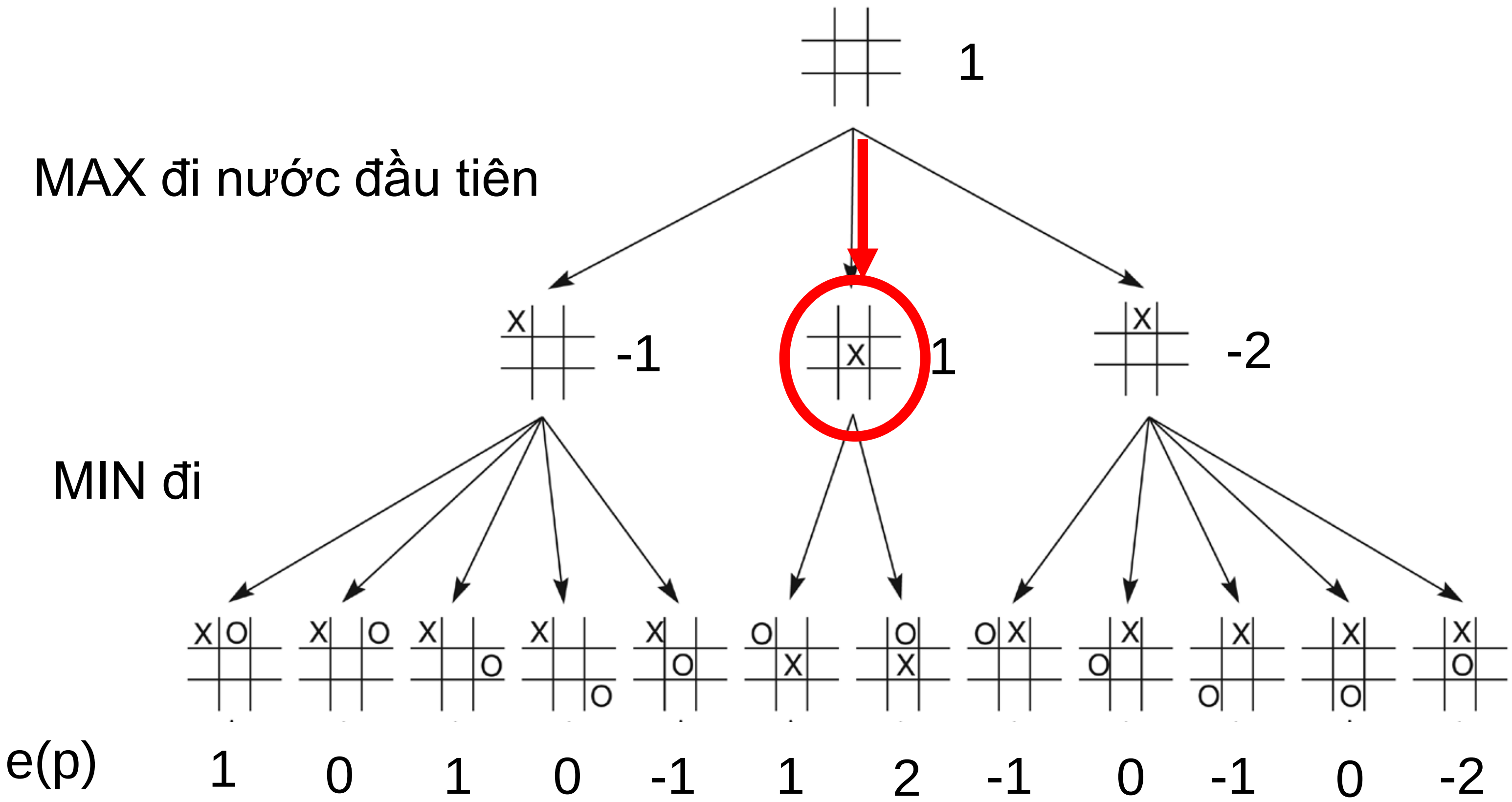
Áp dụng vào chơi cờ caro trên bảng ô vuông (Tictactoe), kích thước 3x3. MAX đặt dấu x, MIN đặt dấu o. Ở mỗi nước đi, mỗi đối thủ xem trước 2 nước.

Ước lượng e(p) đối với mỗi thế cờ p:

E(p) = (số dòng, số cột, số đường chéo còn mở đối với MAX)
- (số dòng, số cột, số đường chéo còn mở đối với MIN)

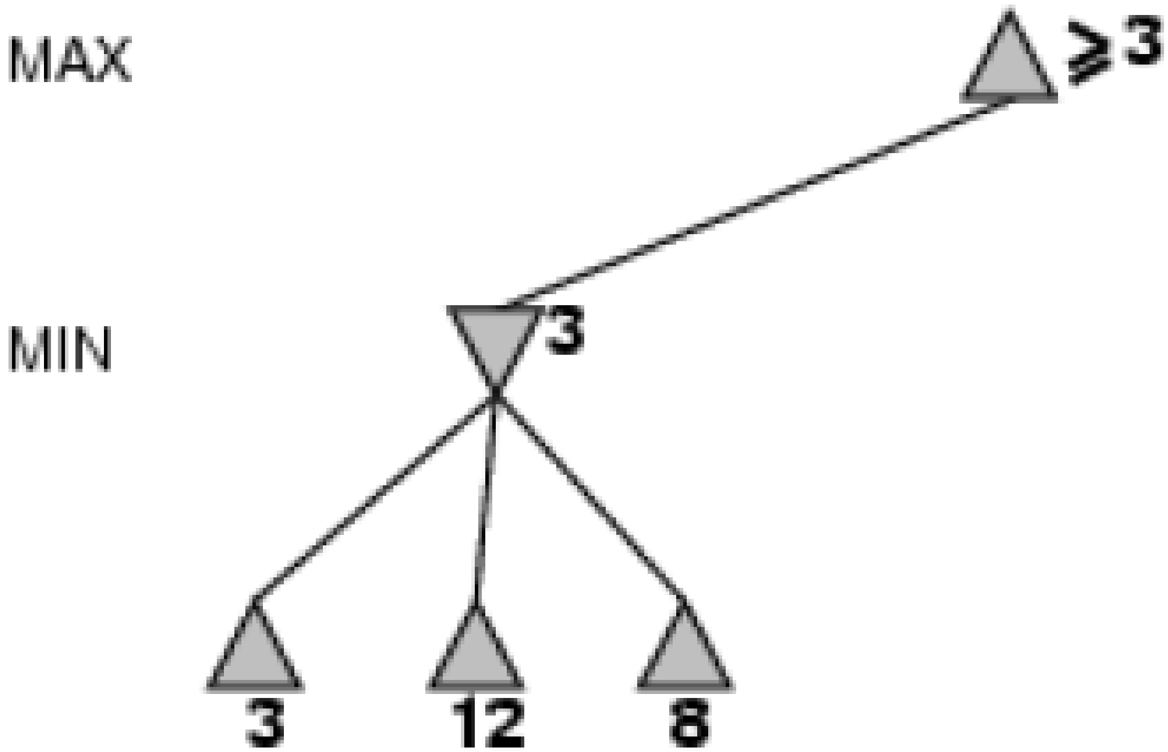
- Nếu p là thế thắng đối với MAX, e(p) = $+\infty$
- Nếu p là thế thắng đối với MIN, e(p) = $-\infty$
- MAX đi mọi đường không có o; MIN đi mọi đường không có x

KGTT của tic-tac-toe được thu nhỏ nhờ tính đối xứng của các trạng thái

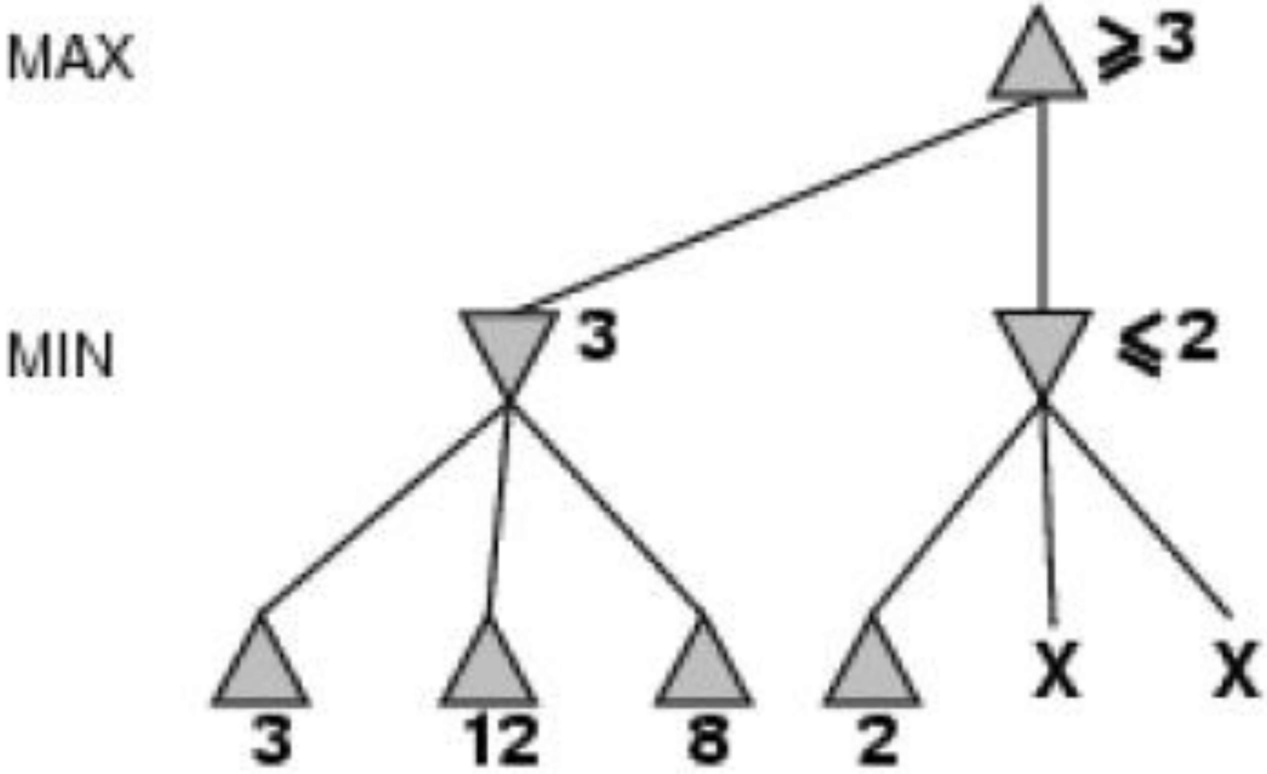


→ Tìm kiếm theo kiểu depth-first

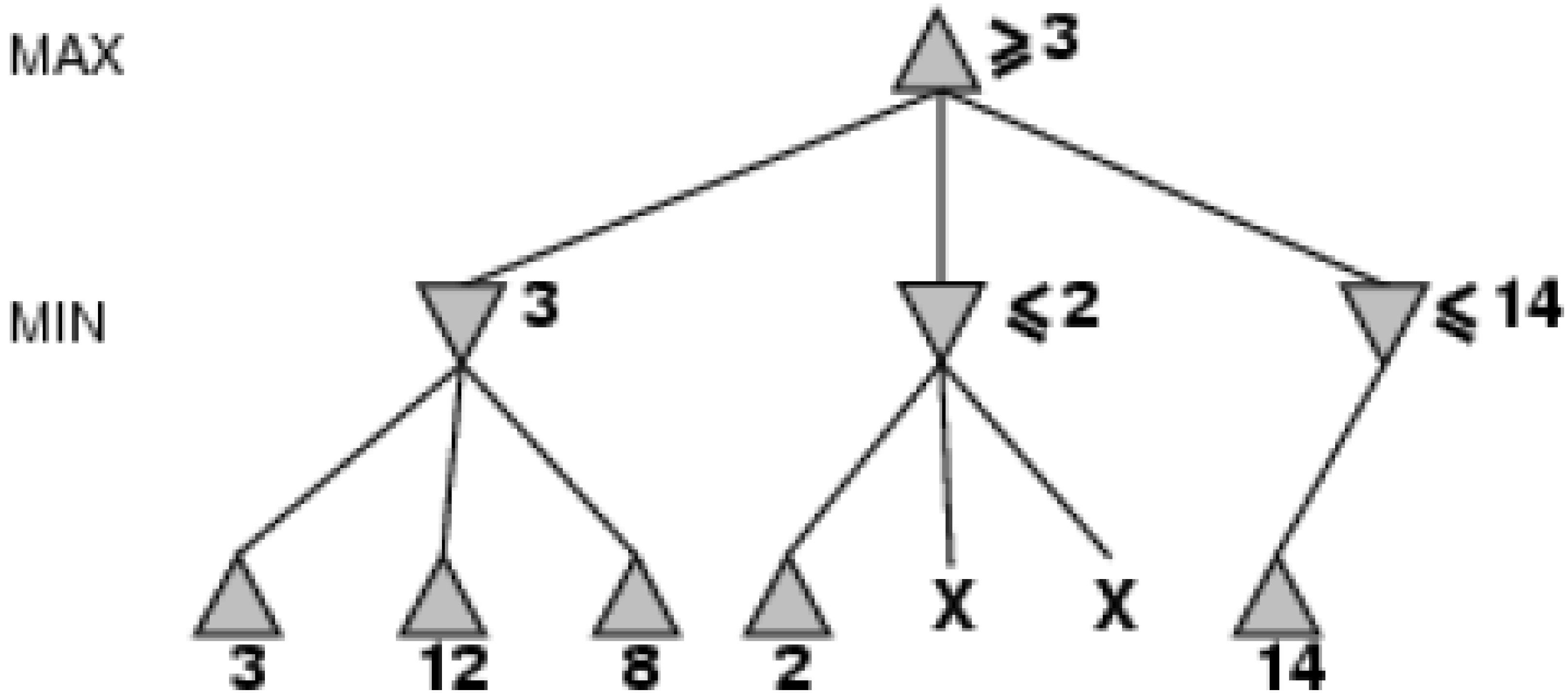
Phương pháp cắt tỉa α-β trong trò chơi minimax



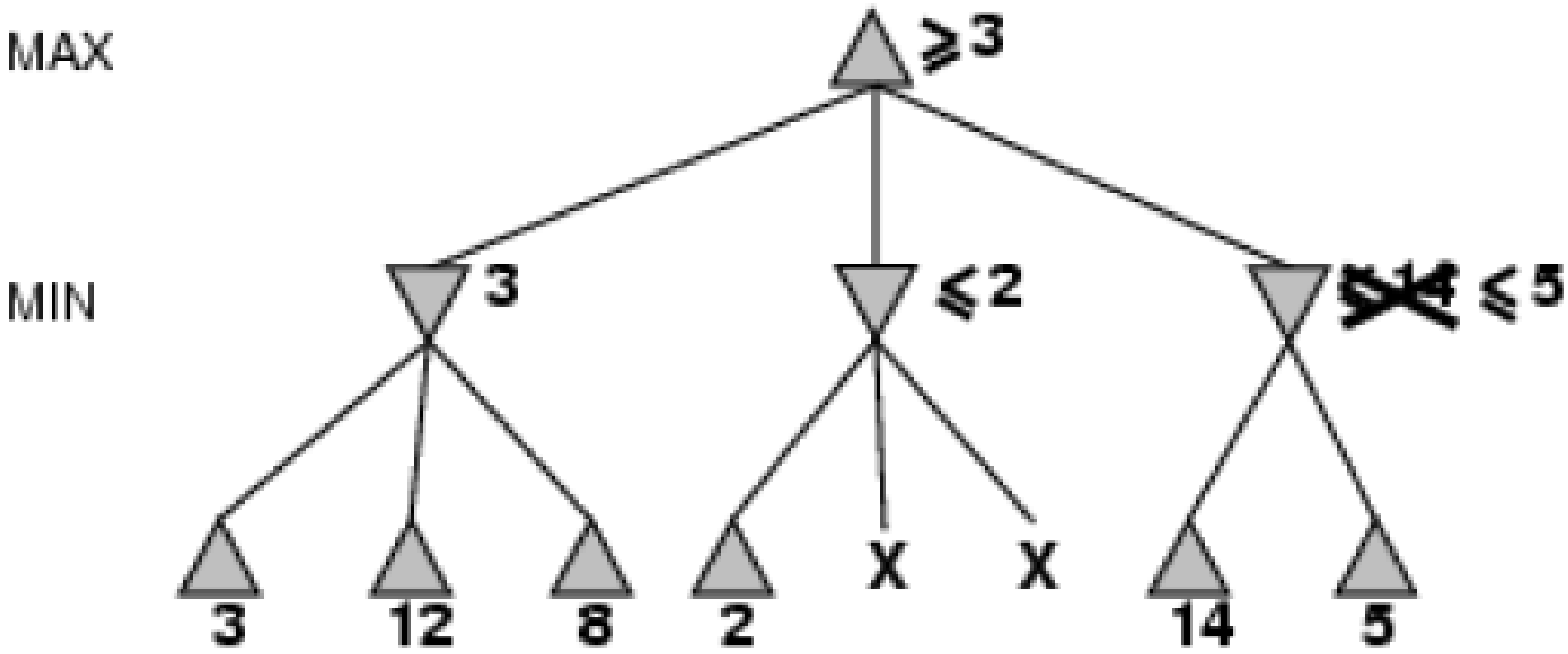
Phương pháp cắt tỉa α-β



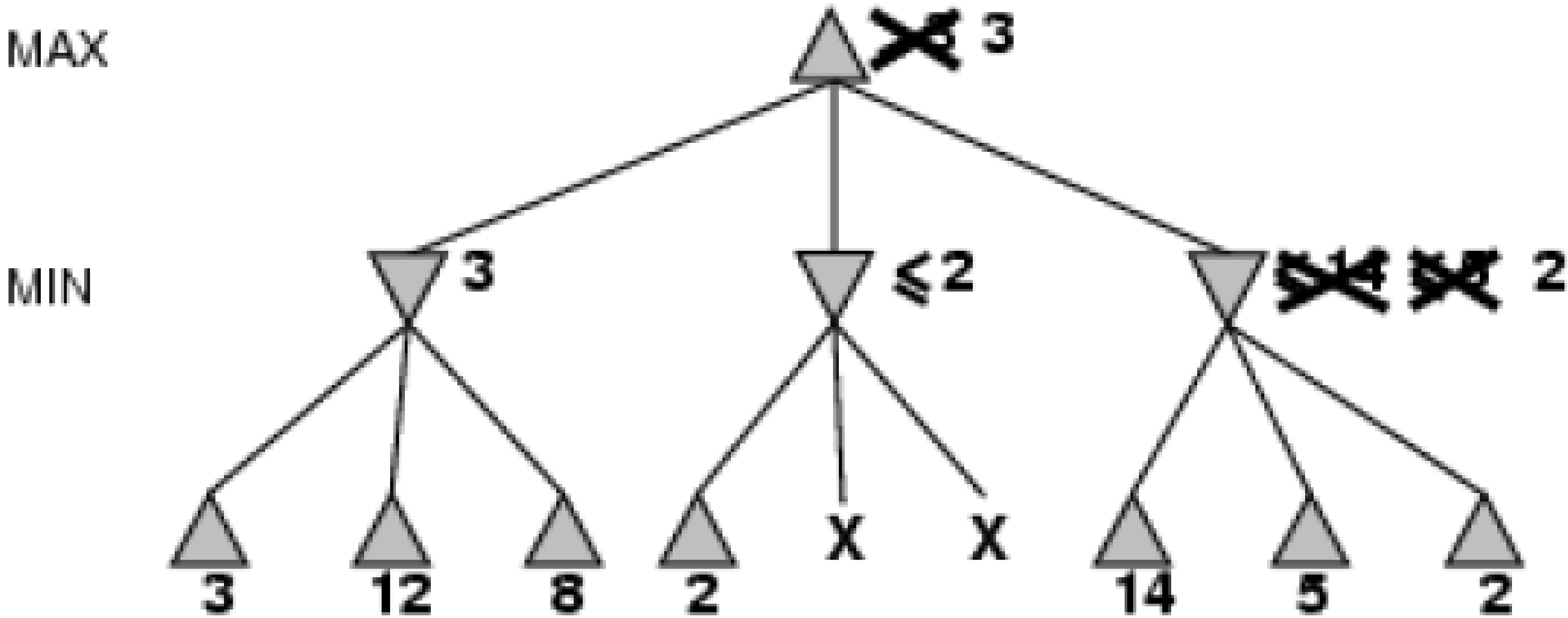
Phương pháp cắt tỉa α-β



Phương pháp cắt tỉa α-β



Phương pháp cắt tỉa α-β

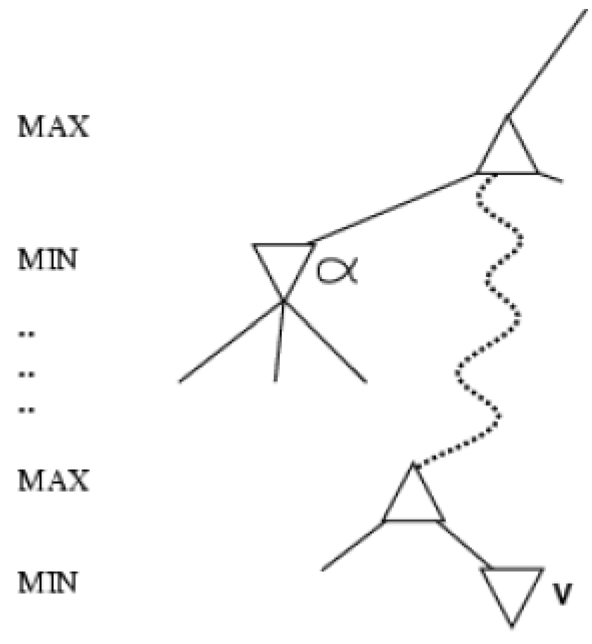


Phương pháp cắt tỉa α-β

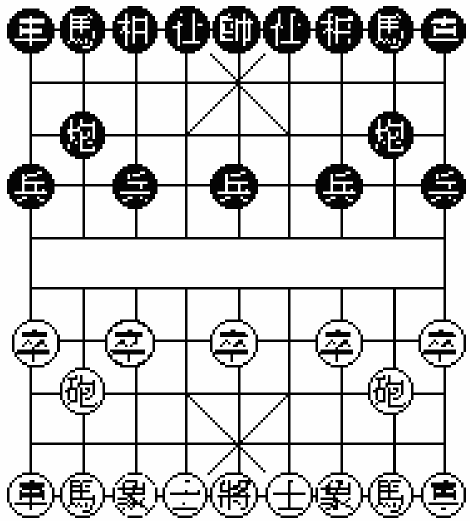
- Cắt cụt không làm ảnh hưởng tới kết quả cuối cùng
- Sắp xếp thứ tự duyệt tối ưu sẽ nâng cao hiệu quả của quá trình cắt cụt
- Trong trường hợp tốt nhất, độ phức tạp thời gian = $O(b^m/2)$

Tại sao gọi là α-β?

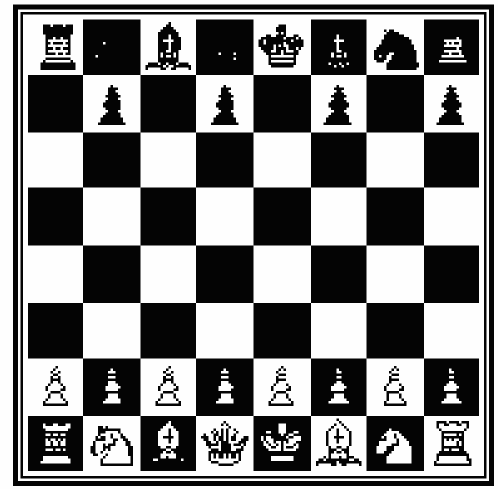
- α là giá trị của lựa chọn tốt nhất được tìm thấy ở thời điểm hiện tại trên đường đi của max
- Nếu v tồi hơn α, max sẽ không duyệt nó → cắt cụt nhánh đó
- Định nghĩa β tương tự đối với min



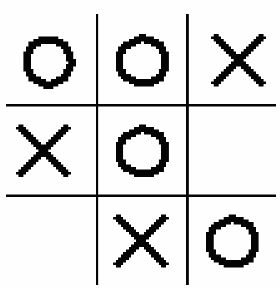
Một số trò chơi đối kháng (minimax)



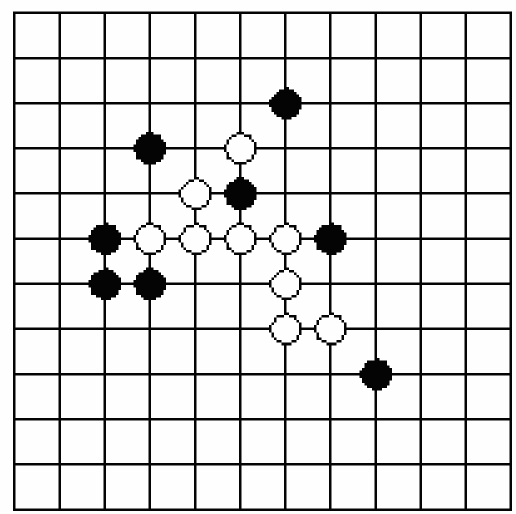
a) Cờ Tướng



b) Cờ Vua (cờ Quốc Tế)



c) Tictactoe



d) Go-moku (cờ caro)

Nội dung môn học

Chương 1. Tổng quan

Chương 2. Tác tử thông minh

Chương 3. Giải quyết vấn đề

- 3.1. Tổng quan
- 3.2. GQVĐ dựa trên tìm kiếm
- 3.3. Các kỹ thuật tìm kiếm cơ bản
- 3.4. Tìm kiếm có đối thủ
- 3.5. Tìm kiếm lời giải trên đồ thị Và/Hoặc

Chương 4. Tri thức và suy diễn

Chương 5. Học máy

3.5. Tìm kiếm lời giải trên đồ thị Và/Hoặc

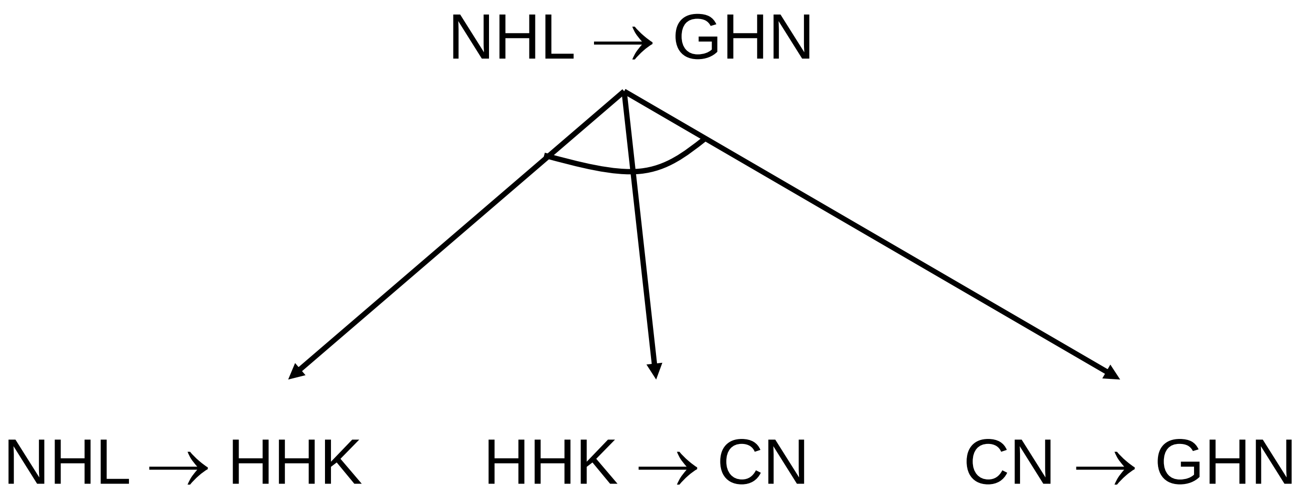
Phân rã bài toán thành bài toán con

VD1: Tìm đường đi từ Nhà hát lớn đến Ga Hà Nội.

BT1. Đi từ Nhà hát lớn đến Hồ hoàn kiếm

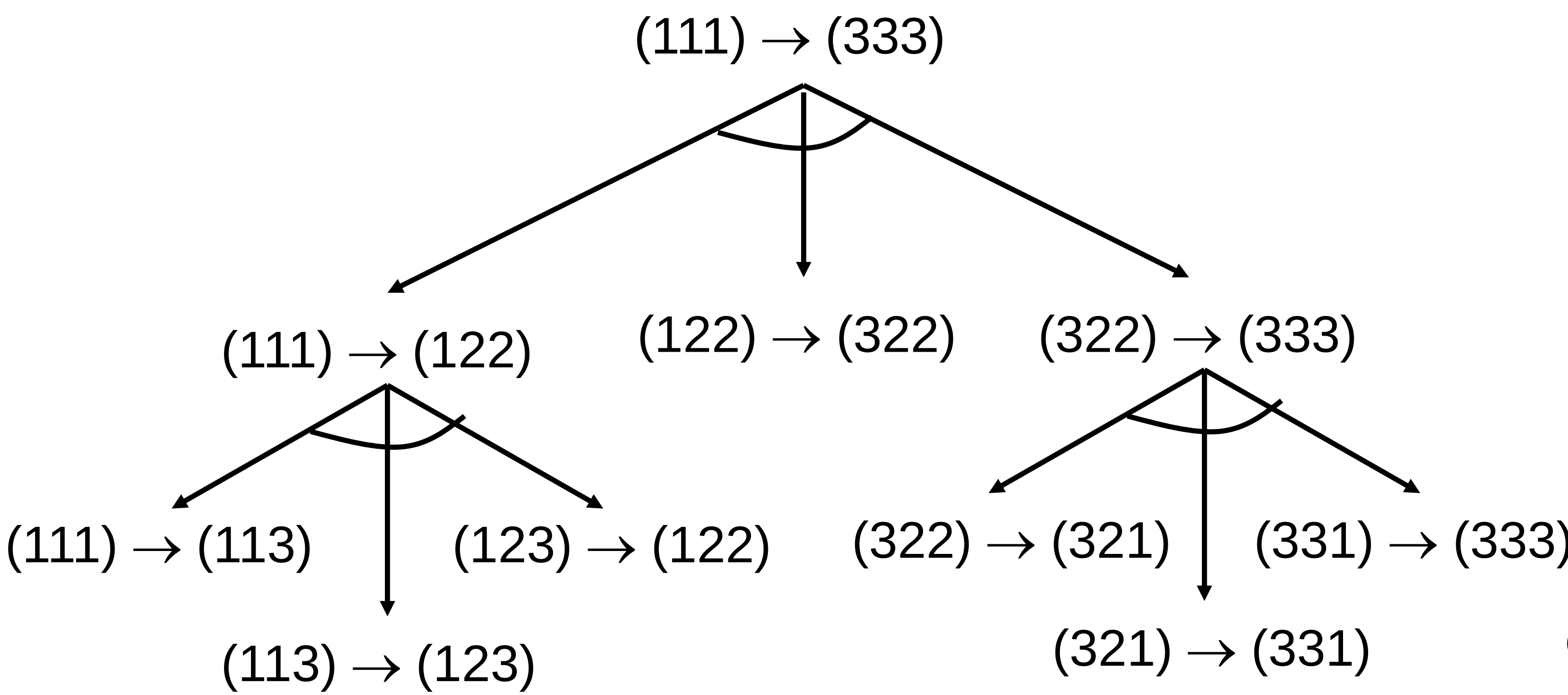
BT2. Đi từ Hồ hoàn kiếm đến Cửa Nam

BT3. Đi từ Cửa Nam đến Ga Hà Nội



Tìm kiếm lời giải trên đồ thị Và/Hoặc

VD2: Tháp Hà Nội n = 3
Bài toán đầu (111) → (333) được qui về 3 bài toán con:
BT1. (111) → (122): chuyển 2 đĩa AB từ cọc 1 sang cọc 2
BT2. (122) → (322): chuyển đĩa C từ cọc 1 sang cọc 3
BT3. (322) → (333): chuyển 2 đĩa AB từ cọc 2 sang cọc 3
BT2 giải được ngay, BT1 và BT3 tiếp tục phân rã

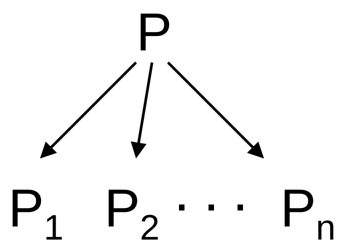


Qui bài toán về BT con

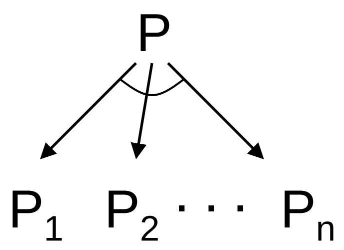
- Bài toán
- Qui về BT con
  - Phát biểu lại BT
    $P \rightarrow P_1,\ldots,P_n$
  - Phân rã P thành
    $P_1,\ldots,P_n$
- BT xuất phát
- BT sơ cấp (nguyên tử) ($\exists$ thuật giải để giải quyết)
- Giải BT P

Đồ thị V/H

- Đỉnh
- Cung
  - Cung Hoặc $P \rightarrow P_1$
    …
    $P \rightarrow P_n$



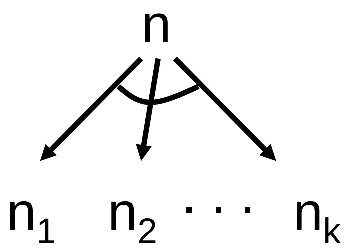
  - Cung Và $P \rightarrow P_{1,}$, …,
    $P \rightarrow P_n$

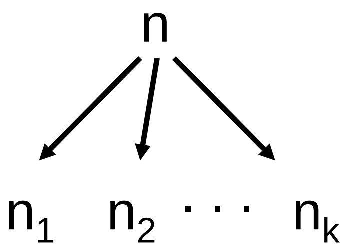


- Đỉnh gốc n_0
- Đỉnh kết thúc
- XD đồ thị lời giải

Đỉnh giải được

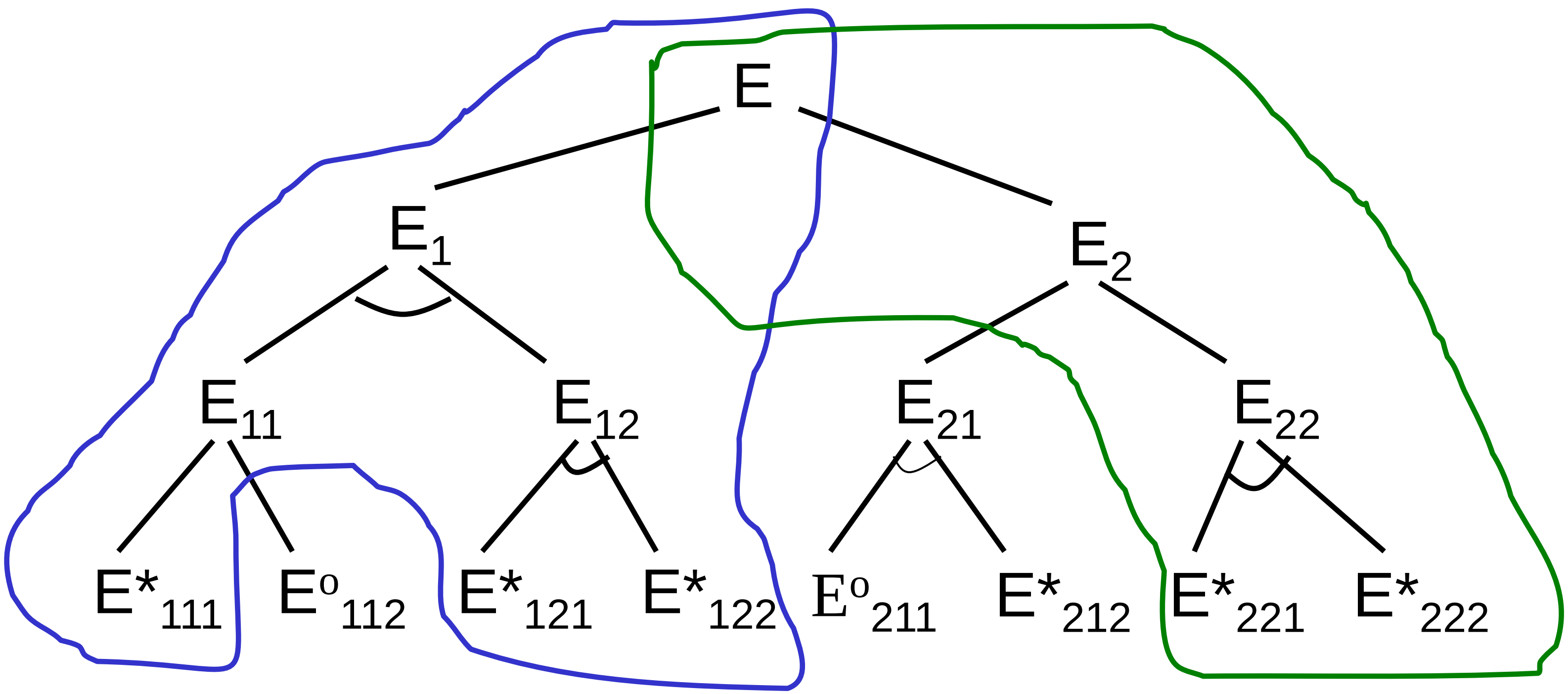
1. Đỉnh kết thúc ($\Leftrightarrow$ bài toán sơ cấp) giải được
2. Giả sử n có con $n_1,\ldots,n_k$
   - $n_1, \ldots, n_k \in N_V$
     n giải được $\Leftrightarrow \forall n_i$ giải được
   - $n_1, \ldots, n_k \in N_H$
     n giải được $\Leftrightarrow \exists n_i$ giải được





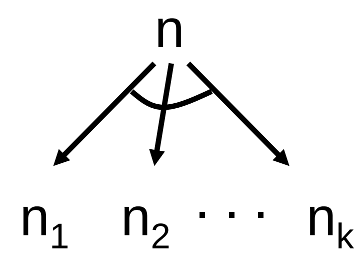
Cây lời giải T

- là cây con của đồ thị G
  - – $n_0 \in T$
  - – $\forall$ đỉnh $n \in T$, n giải được



Đỉnh không giải được

1. n là lá ($\Gamma(n) = \varnothing$), n không kết thúc
2. n có con $n_1,\ldots,n_k$
   – $n_i \in N_V$, $n_{kgd} \Leftrightarrow \exists\ n_i$ không giải được
   – $n_i \in N_H$, $n_{kgd} \Leftrightarrow \forall\ n_i$ không giải được

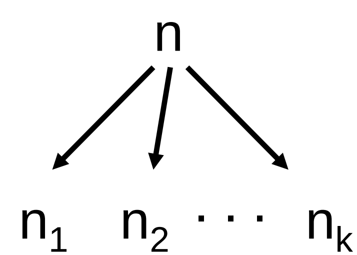


Qui ước: $n \in N$ là 1 bài toán nào đó

nhan(n) = true nếu đỉnh n giải được

false nếu đỉnh n không giải được

kxd nếu đỉnh n không xác định



Với bài toán P, cần xác định nhan(n_0), kéo theo đồ thị lời giải

Thuật toán gán nhãn đỉnh giải được

procedure GD(n)

/* n gd hay nhan(n) = true tuỳ thuộc vào thông tin gd của các đỉnh con của n */

$\{_1$ if kt(n) then nhan(n)=true

else $\{_2$

if n có các đỉnh con là đỉnh VÀ $n_1,\ldots,n_k$ then

$\{_3$ gd(n_1);…; gd(n_k);

if(nhan(n_1) and … and nhan(n_k) then nhan(n) = true $\}_3$

if n có các đỉnh con là đỉnh HOẶC $n_1,\ldots,n_k$ then

$\{_4$ gd(n_1);…;gd(n_k);

if(nhan(n_1) or … or nhan(n_k) then nhan(n) = true $\}_4$

$\}_2$

$\}_1$

Thuật toán gán nhãn đỉnh K giải được

procedure KGD(n)

$\{_1$ if not kt(n) then nhan(n)=false

else $\{_2$

if n có các đỉnh con là đỉnh VÀ $n_1,\ldots,n_k$ then

$\{_3$ gd(n_1);…; gd(n_k);

if(not nhan(n_1) or… or not nhan(n_k) then nhan(n) = false$\}_3$

if n có các đỉnh con là đỉnh HOẶC $n_1,\ldots,n_k$ then

$\{_4$ gd(n_1);…;gd(n_k);

if(not nhan(n_1) and … and not nhan(n_k) then nhan(n) = false$\}_4$

$\}_2$

$\}_1$

Thuật toán gán nhãn đỉnh giải được

procedure GD(n)

/* n gd hay nhan(n) = true tuỳ thuộc vào thông tin gd của các đỉnh con của n */

$\{_1$ if kt(n) then nhan(n)=true

else $\{_2$

if n có các đỉnh con là đỉnh VÀ $n_1,\ldots,n_k$ then

$\{_3$ gd(n_1);…; gd(n_k);

if(nhan(n_1) and … and nhan(n_k) then nhan(n) = true $\}_3$

if n có các đỉnh con là đỉnh HOẶC $n_1,\ldots,n_k$ then

$\{_4$ gd(n_1);…;gd(n_k);

if(nhan(n_1) or … or nhan(n_k) then nhan(n) = true $\}_4$

$\}_2$

$\}_1$

TK rộng mới

TK sâu mới:
MO là stack

Vào: Cây V/H G=(N,A) với đỉnh đầu n_0

Ra: cây lời giải

PP:/* sử dụng 2 d/s queue MO, DONG*/

$\{_1$ MO = $\{n_0\}$; DONG = $\varnothing$;

while MO $\neq \varnothing$ do $\{_2$

n $\leftarrow$ get(MO); DONG $\leftarrow$ DONG $\cup$ {n};

if Γ(n) $\neq \varnothing$ then $\{_3$ MO $\leftarrow$ MO $\cup$ Γ(n);

if trong Γ(n) có đỉnh m kết thúc then $\{_4$

nhan(m) = true; gd(n_0);

if nhan(n_0) then exit('thanh cong')

else Loại khỏi MO các đỉnh có tổ tiên là đỉnh giải được

$\}_4$ $\}_3$

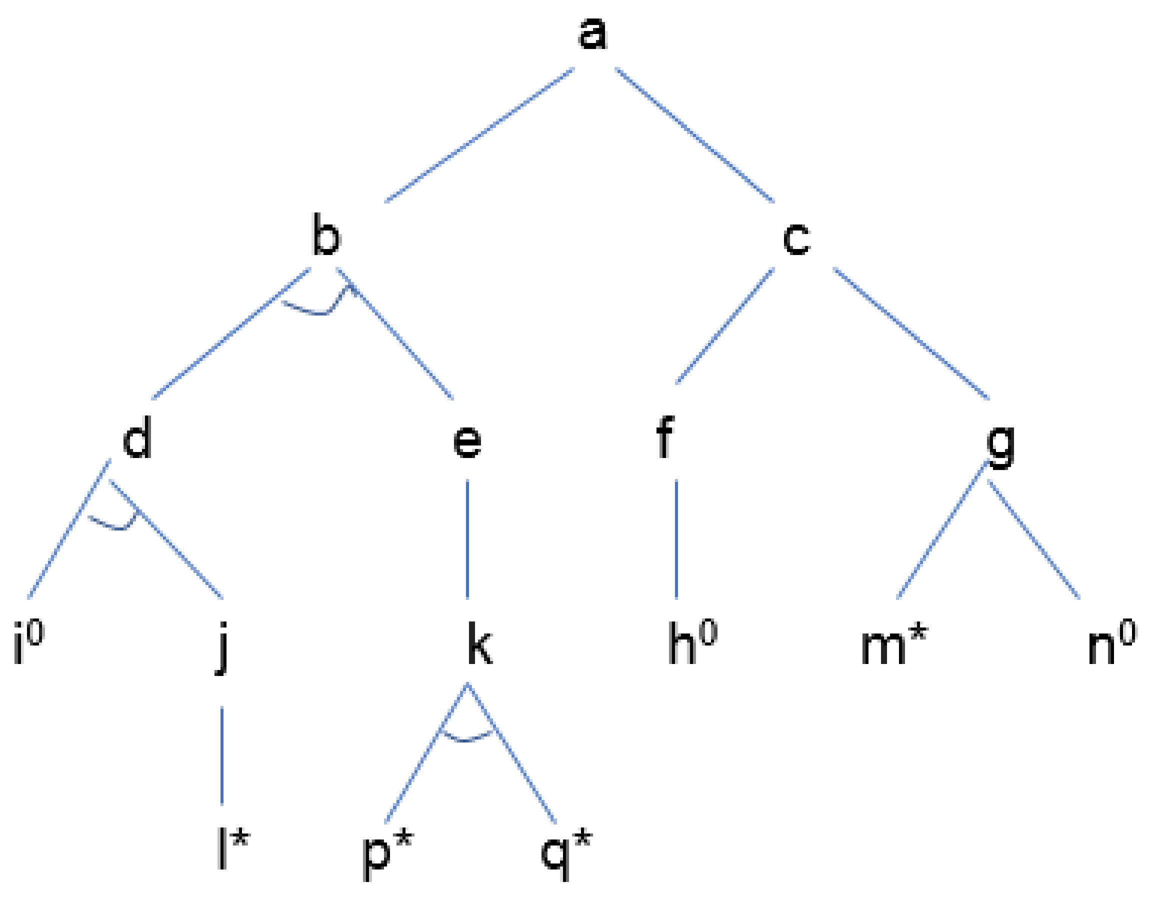
else$\{_5$ nhan(n) = false; kgd(n_0);

if not nhan(n_0) then exit('khong thanh cong')

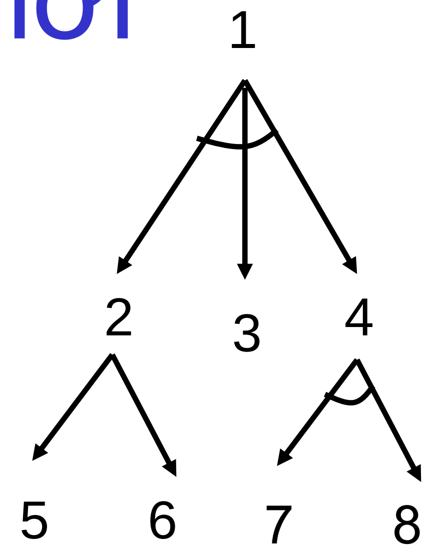
else Loại khỏi MO các đỉnh có tổ tiên là đỉnh không giải được $\}_5$ $\}_2$ $\}_1$

Bài tập

- Áp dụng thuật toán tìm kiếm rộng mới, xác định cây lời giải cho cây và/hoặc dưới đây:



Tìm kiếm cực tiểu mới



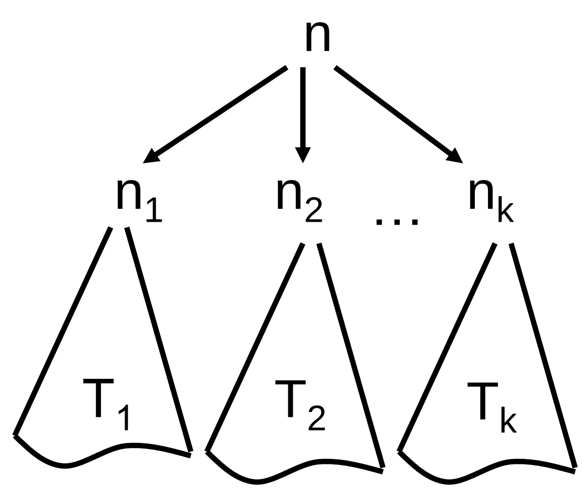
Đặt vấn đề: Tìm cây T* để chi phí C(T*) min

Thực tiễn đưa ra 2 mô hình về giá cây T

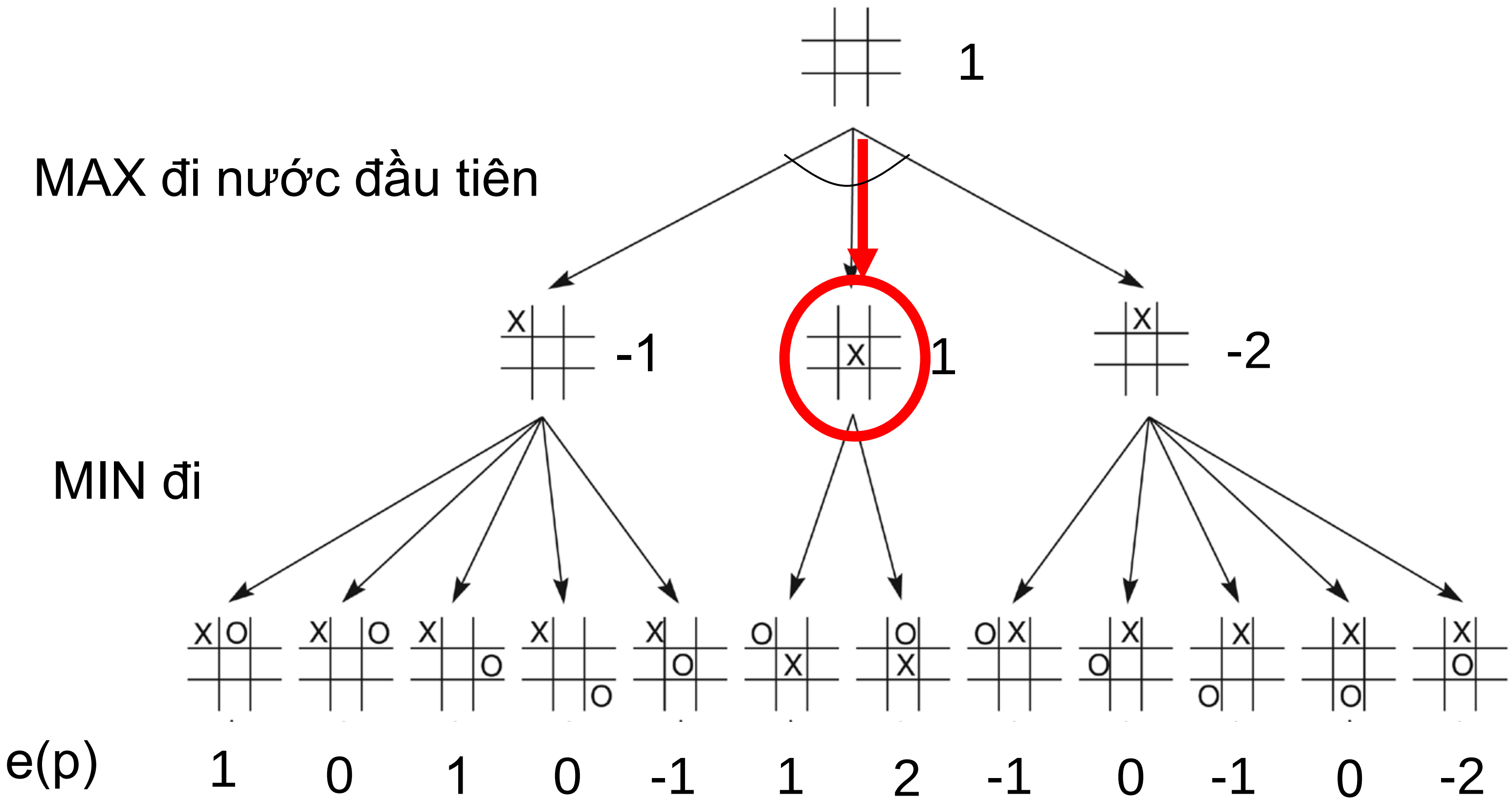
- Giá tổng cộng: $C_{\sum}(T) = \sum_{a \in T} c(a)$
- Giá max: $C_{\max}(T) = \max_{p:n_0 \to leaves} (c(p))$

Giá tối ưu để giải bài toán là h(n), h(n) có các tính chất sau:

- Nếu n là đỉnh kết thúc, h(n) = 0
- Nếu n có con là $n_1, \ldots, n_k$
- $\forall n_i \in N_H$: $h(n) = \min[h(n_i) + c(n,n_i)]$
- $\forall n_i \in N_V$:
  - Giá Σ: $h(n) = [\,\Sigma h(n_i) + \Sigma c(n,n_i)]$
  - Giá max: $h(n) = \max[h(n_i) + c(n,n_i)]$
- Nếu n là đỉnh không giải được, $h(n) = \infty$



KGTT của tic-tac-toe



Tìm kiếm cực tiểu mới

- Trong quá trình tìm kiếm, tại mỗi bước có **1 tập các cây con gốc n_0** sao cho chúng có thể thành phần trên của cây lời giải cuối - cây lời giải tiềm tàng T_0.
- Xây dựng cây T_0 dựa theo h'

Tìm kiếm cực tiểu mới

Cách xác định cây lời giải tiềm tàng T_0

1. Đỉnh đầu $n_0 \in T_0$
2. Nếu $n \in T_0$ có các đỉnh con $n_1 \rightarrow n_k$ là:
   1. đỉnh HOẶC:
      - chọn n_i: $c(n,n_i) + h'(n_i)$ min, $nhan(n_i) \neq kgd$
   2. đỉnh VÀ:
      - chọn $n_1, \ldots, n_k$ vào T_0 nếu $\forall n_i$: $nhan(n_i) \neq kgd$

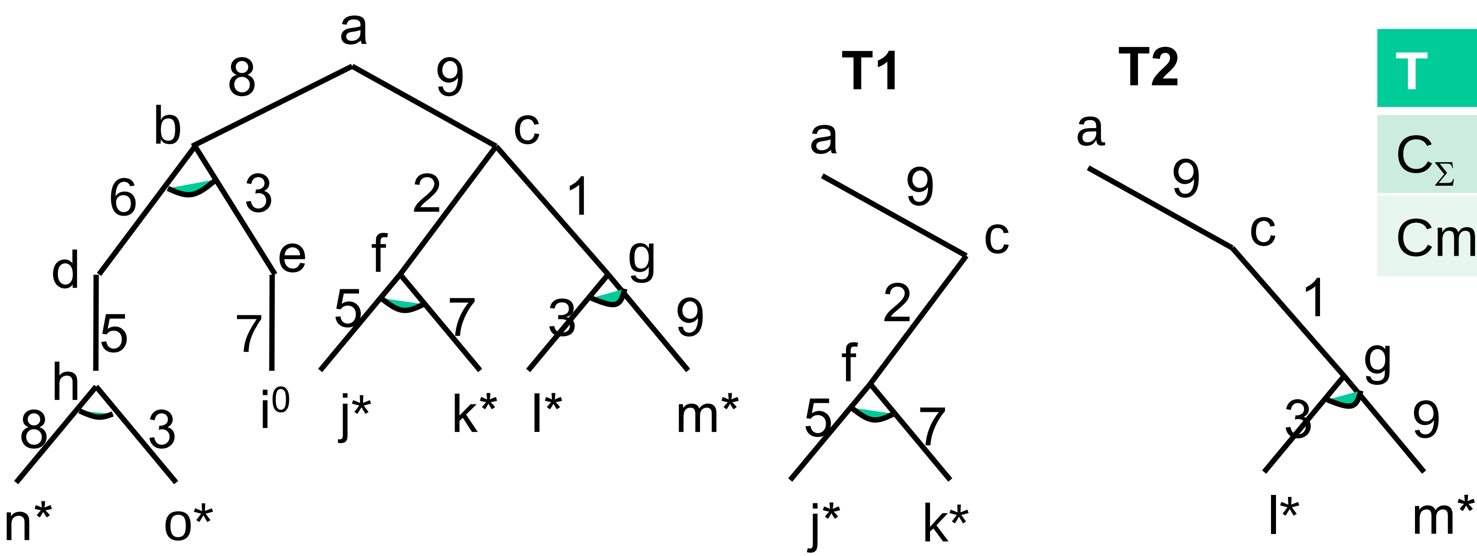
Nhận xét:

- Nếu cây V/H chỉ chứa đỉnh hoặc → TKCT
- Nếu c = 1 và h'=0 với ∀nút và sử dụng giá max → TKRM
- Nếu $h'(n) \leq h(n)$ với $\forall n$ và $\exists \delta$: $\forall a \in A$, $c(a) \geq \delta$ thì TKCTM dừng và cho kết quả là cây lời giải tối ưu

TKCT mới

1. MO = $\{n_0\}$; DONG = $\varnothing$; $T_0 = n_0$
2. Chọn n $\in$ MO $\cap$ lá (T_0):
    DONG $\leftarrow$ DONG $\cup$ {n};
    if kt(n) then { nhan(n) = true; gd(n_0);
        if nhan(n_0) then exit('thanh cong')
        else Loại khỏi MO các đỉnh có tổ tiên là đỉnh giải được
    } else { // n không kết thúc
        if Γ(n) $\neq \varnothing$ then { MO $\leftarrow$ MO $\cup$ Γ(n);
            Với mỗi m $\in\Gamma$(n), tính h'(m)
            Với mỗi m $\in$MO $\cup$ DONG, tính h'(m)
        else{ // n không kết thúc và không có con
            nhan(n) = false; kgd(n_0);
            if not nhan(n_0) then exit('khong thanh cong')
            else Loại khỏi MO các đỉnh có tổ tiên là đỉnh không giải được
        }
    }
3. Lặp lại 2 đến khi MO = $\varnothing$

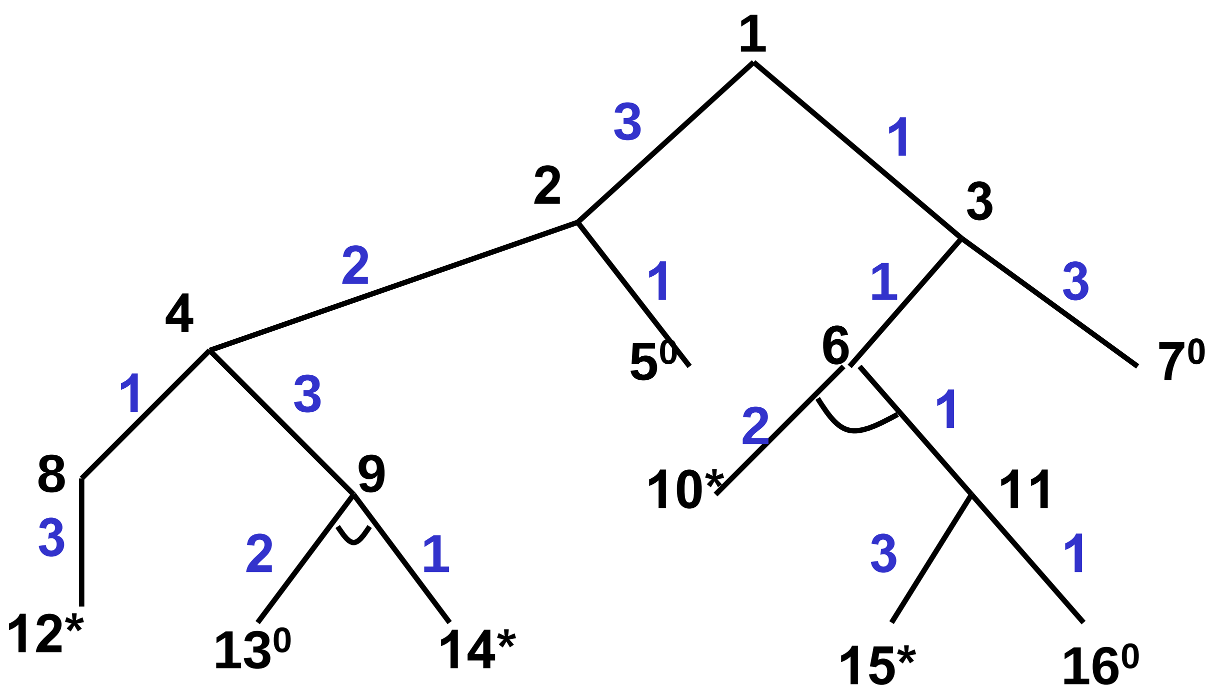
TKCT mới



T	T1	T2
C_Σ	23	22
Cmax	18	19

n	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o
h_Σ	22	∞	13	16	∞	12	12	11	∞	0	0	0	0	0	0
h_{max}	18	∞	9	13	∞	7	9	8	∞	0	0	0	0	0	0
h’	4	3	3	2	2	2	2	1	∞	0	0	0	0	0	0

n	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
h’	4	3	3	2	∞	2	∞	1	1	0	1	0	∞	0	0	∞



Nội dung môn học

Chương 1. Tổng quan
Chương 2. Tác tử thông minh
Chương 3. Giải quyết vấn đề
- 3.1. Tổng quan
- 3.2. GQVĐ dựa trên tìm kiếm
- 3.3. Các kỹ thuật tìm kiếm cơ bản
- 3.4. Tìm kiếm có đối thủ
- 3.5. Tìm kiếm lời giải trên đồ thị Và/Hoặc
- 3.6. Tìm kiếm dựa trên thỏa mãn ràng buộc

Chương 4. Tri thức và suy diễn
Chương 5. Học máy

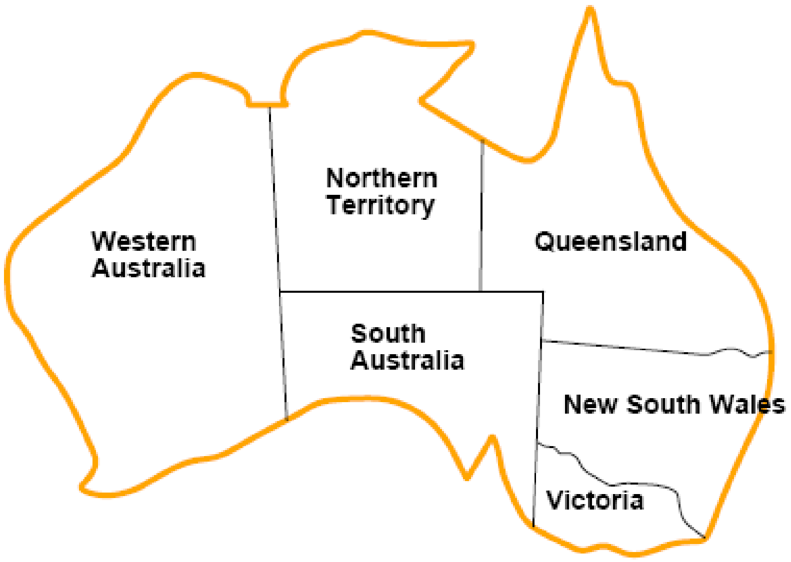
3.6. Bài toán thỏa mãn ràng buộc

Constraints Satisfaction Problems (CSPs)

- Bài toán tìm kiếm tiêu chuẩn
  - Mỗi trạng thái là 1 hộp đen
- Cấu trúc dữ liệu cho phép biểu diễn các trạng thái và hàm chuyển trạng thái
- CSPs
  - Mỗi trạng thái gồm các biến Xi có giá trị trong miền Di
  - Việc kiểm tra trạng thái đích là việc kiểm tra các ràng buộc đối với các biến

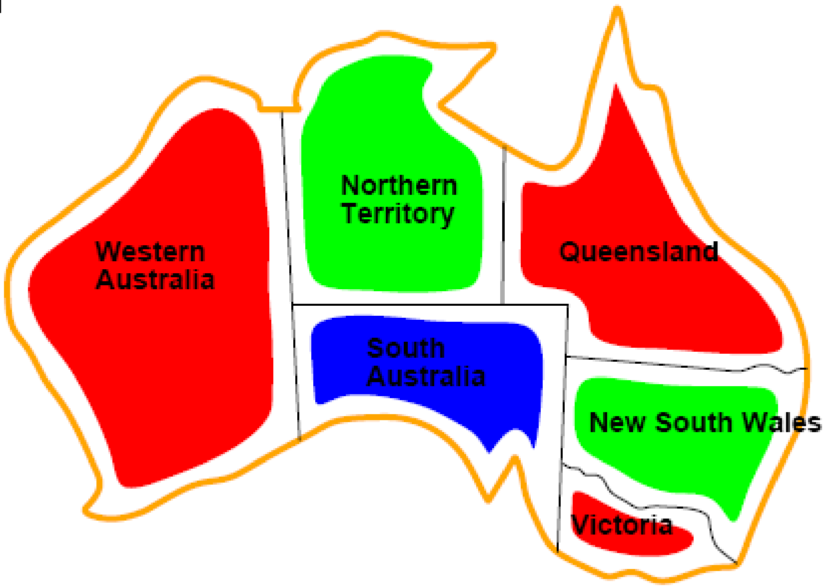
Ví dụ: Tô màu bản đồ

- Các biến:
  - WA, NT, Q, NSW, V , SA
- Miền giá trị
  - D_i = {red, green, blue}
- Ràng buộc:
  - Các hàng xóm phải khác màu
    - WA /= NT
    - WA /= SA
    - NT /= SA
- …



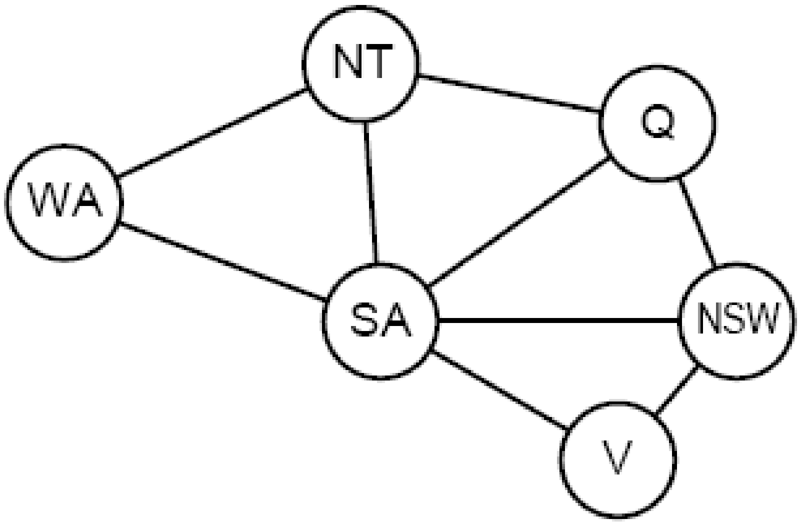
Ví dụ: Tô màu bản đồ

- Đáp án là phép gán các biến thỏa mãn mọi ràng buộc
  - WA = đỏ
  - NT = xanh
  - Q = đỏ
  - NSW = xanh
  - V = đỏ
  - SA = xanh



Đồ thị ràng buộc

- Ràng buộc nhị phân
  - Mỗi ràng buộc liên quan hai biến
- Đồ thị ràng buộc
  - Nút là biến
  - Cạnh là ràng buộc



Các biến thể của CSPs

- Biến rời rạc
  - Miền hữu hạn, ví dụ: giải bài toán SAT
  - Miền vô hạn, ví dụ, lập lịch làm việc
    - Biến là bắt đầu/kết thúc ngày làm việc
    - Biểu diễn ràng buộc, ví dụ: StartJob1 + 5 <= StartJob3
    - Các ràng buộc tuyến tính là có thể giải được, các ràng buộc phi tuyến tính là không giải được
- Biến liên tục
  - Ví dụ: thời gian bắt đầu/kết thúc quan sát vũ trụ bằng kính viễn vọng Hubble
  - Các ràng buộc tuyến tính có thể giải được bằng cách
- sử dụng Lập trình tuyến tính

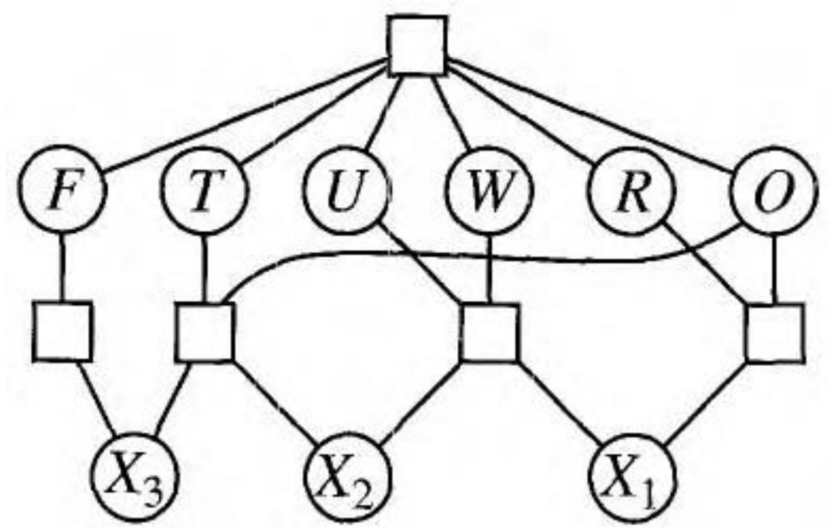
Các loại ràng buộc

- Các ràng buộc một biến
  - ví dụ: SA / = xanh
- Ràng buộc nhị phân
  - ví dụ: SA / = WA
- Ràng buộc đa biến
  - Liên quan ít nhất 3 biến
- Hạn chế mềm
  - Ưu tiên, ví dụ, màu đỏ tốt hơn màu xanh lá cây
  - Hàm chi phí trên các biến

Ví dụ: Bài toán đố chữ

- Biến:
  - $F, T, O, U, R, W, X_1, X_2, X_3$
- Miền:
  - {0,1,2,3,4,5,6,7, 8,9}
- Ràng buộc:
  - Alldiff(F,T,O,U,R,W)
  - $O+O = R+10*X_1$
  - $X_1+W+W= U+10*X_2$
  - $X_2+T+T= O+10*X_3$
  - $X_3=F$

TWO
+
TWO
FOUR



Các bài toán CSP thực tế

- Phân công công việc
  - Vd: Ai dạy lớp nào
- Lập kế hoạch
  - Ví dụ, khi nào và nơi lớp học diễn ra
- Thiết kế phần cứng
- Bảng tính
- Lập kế hoạch vận chuyển
- Lập kế hoạch sản xuất

CSPs theo tìm kiếm tiêu chuẩn

- Trạng thái
  - Được xác định bởi các biến được gán giá trị cho đến nay
- Trạng thái ban đầu
  - Chưa có biến nào được gán
- Hàm chuyển
  - Gán một giá trị cho biến kế tiếp mà không xung đột với bộ giá trị hiện tại
- Thất bại nếu không có phép gán nào được chấp nhận
- Kiểm tra trạng thái đích
  - Tất cả các biến được gán giá trị và không có xung đột

CSPs theo tìm kiếm tiêu chuẩn

- Có giải pháp xuất hiện ở độ sâu d với n biến
  - Sử dụng tìm kiếm theo chiều sâu
- Đường dẫn không liên quan
- Số lá
  - $n!\ d^n$
    - Quá nhiều

Tìm kiếm quay lui

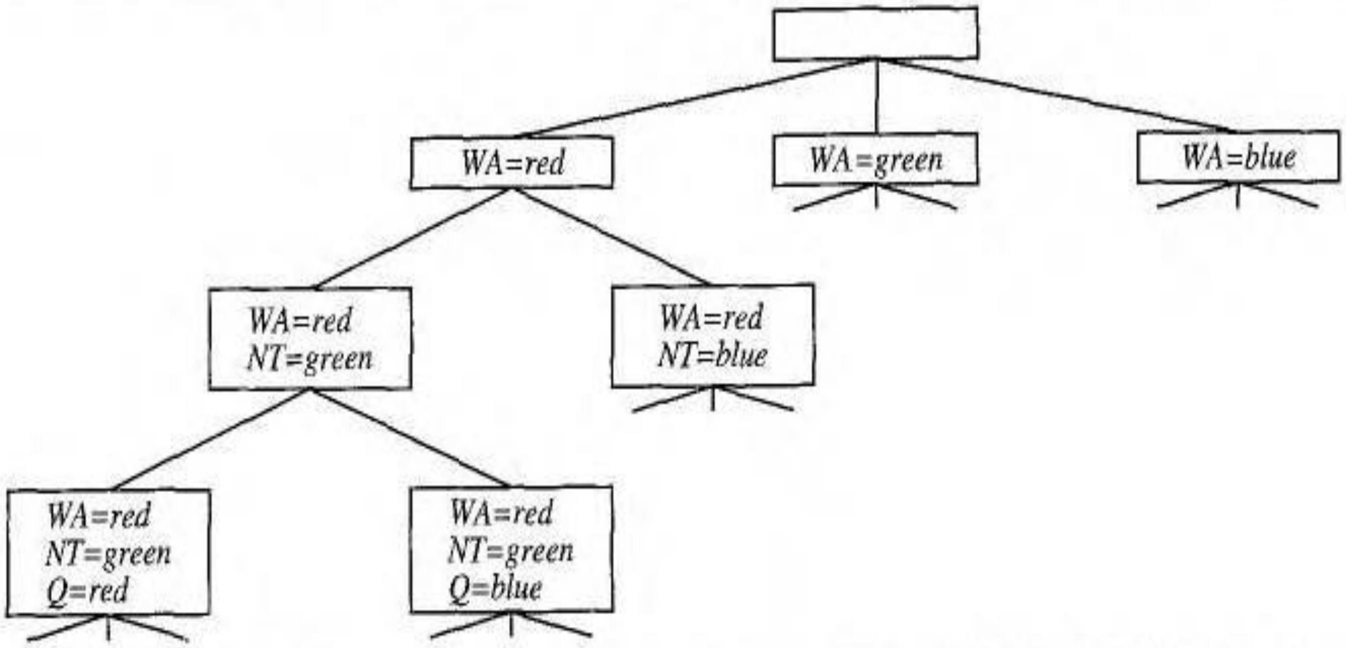
- Việc gán giá trị là phép giao hoán, vd
  - {WA=red, NT =green}
  - {NT =green, WA=red}
- Gán từng biến đơn
  - Chỉ xét từng biến một
  - d^n lá
- Tìm kiếm quay lui
  - Tìm kiếm sâu + Gán từng biến đơn
- Tìm kiếm quay lui là thuật toán không có chi phí cơ bản (uninformed algorithm) cho CSP
  - Có thể giải n-Queen với n = 25

Tìm kiếm quay lui

```
function BACKTRACKING-SEARCH(csp) returns solution/failure
   return RECURSIVE-BACKTRACKING({ }, csp)

function RECURSIVE-BACKTRACKING(assignment, csp) returns soln/failure
   if assignment is complete then return assignment
   var ← SELECT-UNASSIGNED-VARIABLE(VARIABLES[csp], assignment, csp)
   for each value in ORDER-DOMAIN-VALUES(var, assignment, csp) do
       if value is consistent with assignment given CONSTRAINTS[csp] then
           add {var = value} to assignment
           result ← RECURSIVE-BACKTRACKING(assignment, csp)
           if result ≠ failure then return result
           remove {var = value} from assignment
   return failure
```

Tìm kiếm quay lui



Cải tiến tìm kiếm quay lui

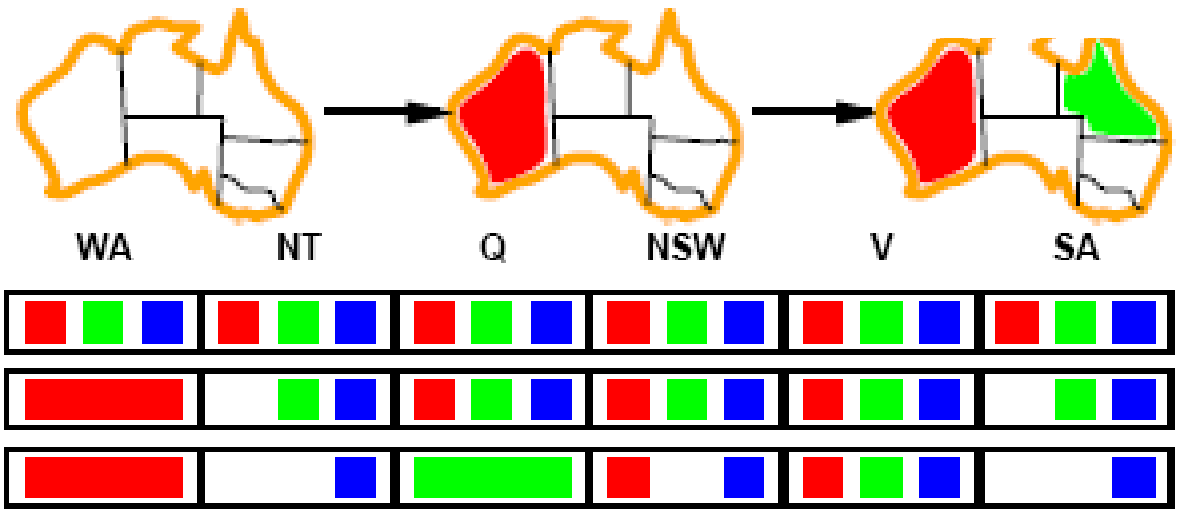
- Biến nào nên được gán tiếp theo?
- Nên thử các giá trị của biến theo trình tự nào?
- Chúng ta có thể phát hiện sớm đường đi dẫn đến thất bại không?
- Chúng ta có thể tận dụng cấu trúc vấn đề không?

Chọn biến

- Còn lại ít giá trị nhất
  - Chọn biến có ít giá trị hợp lệ nhất
- Mức độ heuristic
  - Chọn biến có nhiều ràng buộc heuristic nhất với các biến còn lại
- Cập nhật các giá trị hợp lệ cho các biến chưa được gán giá trị
  - Kết thúc khi có bất cứ biến nào không còn giá trị có thể gán

Tìm kiếm kiểu tiến

- Lan truyền ràng buộc



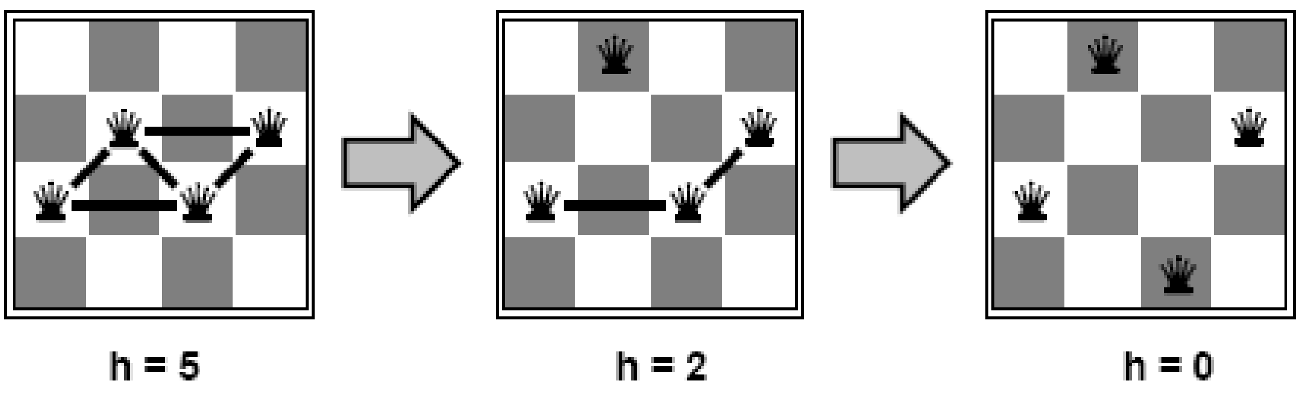
  - NT và SA không thể cùng màu blue
  - Cách lan truyền đơn giản nhất để các phép gán trị bền vững:
  - X -> Y bền vững nếu với mỗi giá trị x của X có vài giá trị y của Y

Thuật toán lặp cho CSP

- Thuật toán leo đồi, mô phỏng tôi luyện
  - Mỗi nút tương ứng 1 bộ giá trị hoàn chỉnh
- Có cho phép các trạng thái với các ràng buộc không?
- Có các toán tử gán lại các biến
- Lựa chọn biến
- Ngẫu nhiên
- Lựa chọn giá trị nhằm tối thiểu các mâu thuẫn về ràng buộc
  - Chọn giá trị vi phạm ít ràng buộc nhất
- tức là leo đồi với h (n) = tổng số ràng buộc vi phạm

Ví dụ: 4-Queens

- Trạng thái: 4 hậu trên 4 cột (4*4 = 256 states)
- Toán tử: di chuyển hậu trên cột
- Kiểm tra đích: không còn tấn công
- Hàm giá: h(n) = số tấn công



Tóm tắt

- CSP là một bài toán đặc biệt:
  - trạng thái được xác định bởi các giá trị của một tập biến xác định
  - kiểm tra mục tiêu trên cơ sở các ràng buộc với các biến
- Quay lui = tìm kiếm theo chiều sâu với mỗi nút ứng với một biến được gán trị
- Việc sắp trật tự các biến và các mẹo lựa chọn giá trị giúp giảm đáng kể không gian tìm kiếm
- Kiểm tra theo chiều tiến ngăn chặn phép gán dẫn đến thất bại về sau.
- Lan truyền ràng buộc cho phép thu hẹp các giá trị cho các biến và phát hiện xung đột
- Việc biểu diễn ràng buộc cho phép phân tích cấu trúc vấn đề
- CSP cấu trúc cây có thể giải quyết trong thời gian tuyến tính
- Lặp lại việc kiểm tra nhằm tối thiểu hóa xung đột khá hiệu quả trong thực tế

Bài tập - Bài toán đố chữ

- Hãy thay các chữ cái bằng các chữ số từ 0 đến 9 sao cho không có hai chữ cái nào được thay bởi cùng 1 số và thỏa mãn ràng buộc sau:

TWO	CROSS
+	+
TWO	ROADS
FOUR	DANGER