

# ÔN THI CUỐI KỲ

## MÔN: CƠ SỞ TRÍ TUỆ NHÂN TẠO

Lớp: CQ2019/3

Học kỳ 2 2021-2022

### LỊCH SỬ PHIÊN BẢN

Ngày	Phiên bản	Tác giả	Mô tả
2022/06/15	0.1.0	Thọ	Thêm 2 nội dung mới: DLS và LCBFS Điều nội dung từ 1.1 đến 1.6
2022/06/15	0.1.1	Uyên	Thêm phần 6 và một số link bài tập
2022/06/15	0.1.4	Thọ	Thêm phần Đặt vấn đề và điều nội dung cho phần 2.1 GTS Thêm phần 1.7 So sánh chi phí Điều nội dung phần Hill Climbing
2022/06/16	0.1.6	Thọ	Điều nội dung phần A-star, GBFS, và Beam Search
2022/06/16	0.1.7	Uyên	Điều nội dung phần Logic mệnh đề (trừ Davis-Putnam)
2022/06/17	0.2.0	Thọ	Điều nội dung phần 2.6 Một số bài toán Điều nội dung 3 phương pháp học: qua quan sát, qua cây định danh, qua logic
2022/06/18	0.2.1	Minh	Thêm suy diễn tiến và suy diễn lùi
2022/06/18	0.2.2	Phú	Nó ron: phân biệt C và T
2022/06/18	0.2.3	Tùng	Beam search, CSP
2022/06/18	0.3.0	Uyên	Điều nội dung phần tìm kiếm đối kháng
2022/06/18	0.3.1	Thọ	Thêm nội dung cho Thủ tục Davis-Putnam Thêm một vài nội dung cho CSP
2022/06/18	0.3.2	Uyên	Điều nội dung phần 5.2
2022/06/18	0.3.3	Thọ	Điều nội dung phần Gom cụm
2022/06/18	0.3.4	Phú	Bổ sung định nghĩa heuristics, bổ sung việc cắt tỉa cành ở decision tree, bổ sung davis putnam (dùng ảnh ví dụ trên wiki)
2022/06/19	0.3.5	Thanh	Giải bài tập Logic vị từ - sở thích ăn uống (ở phần Nhập)

### ❖ CHÚ Ý

- Khuyến khích ghi lịch sử phiên bản mỗi khi chỉnh sửa để dễ truy vết

1.	Tìm kiếm mù (Blind/Uninformed Search)	4
1.1.	Depth First Search (DFS)	4
1.2.	Breadth First Search (BFS)	4
1.3.	Tìm kiếm theo chiều sâu giới hạn (Depth-Limited Search – DLS)	5
1.4.	Tìm kiếm lặp sâu dần (Iterative Deepening Search – IDS)	5
1.5.	Tìm kiếm theo chiều rộng chi phí nhỏ nhất (Least Cost BFS)	5
1.6.	Tìm kiếm chi phí đồng nhất (Uniform Cost Search – UCS)	6
1.7.	So sánh chi phí	9
2.	Tìm kiếm heuristic (Heuristic/Informed Search)	11
2.1.	Greedy Traveling Saleman (GTS)	11
2.1.1.	Đặt vấn đề	11
2.1.2.	GTS 1	11
2.1.3.	GTS 2	11
2.2.	Hill Climbing (thuật toán leo đồi)	12
2.3.	A-star	13
2.4.	Greedy Best First Search	14
2.5.	Beam Search	15
2.6.	Một số bài toán	16
2.6.1.	Bài toán 8-PUZZLE	16
2.6.2.	Bài toán tháp Hà Nội	18
2.6.3.	Bài toán phân công việc	18
2.6.4.	Bài toán đóng gói	19
3.	Tìm kiếm đối kháng (Adversarial Search)	20
3.1.	Tổng quan	20
3.2.	Thuật toán Mini – Max	21
3.3.	Cắt tỉa alpha – beta	22
4.	Bài toán thỏa mãn ràng buộc (Constraint Satisfaction Problem – CSP)	25
4.1.	Thuật toán Backtracking	26
4.2.	Thuật toán AC-3	28
4.3.	Thuật toán Forward Checking	29
5.	Tri thức	30
5.1.	Logic mệnh đề	30

5.1.1.	Giới thiệu khái quát	30
5.1.2.	Suy diễn tự nhiên	31
5.1.3.	Hợp giải mệnh đề	31
5.1.4.	Thuật giải Robinson	32
5.1.5.	Thủ tục Davis-Putnam	33
5.1.6.	Thuật giải Vương Hạo (Harvard)	34
5.1.7.	Suy diễn tiến, suy diễn lùi	35
5.2.	Logic bậc nhất (logic vị từ)	36
5.2.1.	Giới thiệu khái quát	36
5.2.2.	Hợp giải bậc nhất	37
6.	Các phương pháp máy học	41
6.1.	Học qua quan sát	41
6.1.1.	Thuật toán Quinlan	41
6.1.2.	Ví dụ	41
6.2.	Học bằng cách xây dựng cây định danh	43
6.2.1.	Từ CSDL đến cây định danh	43
6.2.2.	Từ cây đến luật	46
6.3.	Học qua logic	53
6.4.	Gom cụm	54
6.4.1.	Thuật toán K-means	56
7.	Neural Network	59
8.	Link bài tập	61
1.	Tìm kiếm mù (Blind/Uninformed Search)	4

## 1. Tìm kiếm mù (Blind/Uninformed Search)

### 1.1. Depth First Search (DFS)

❖ Ý tưởng

- Mở từ nút vừa mới mở nhất nếu nó còn nút con chưa mở
- Ngược lại, quay về nút trước đó trên đường đi

❖ Minh họa

START

START d

START d b

START d b a

START d c

START d c a

START d e

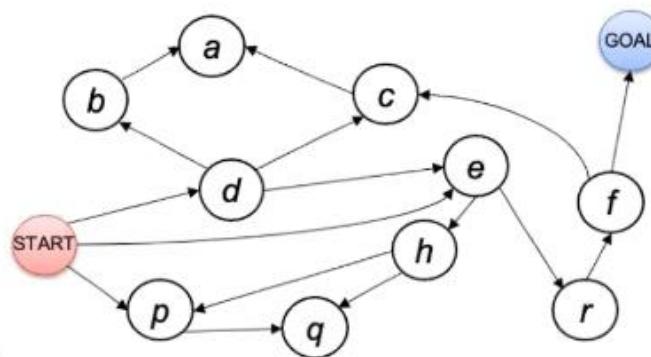
START d e r

START d e r f

START d e r f c

START d e r f c a

START d e r f GOAL

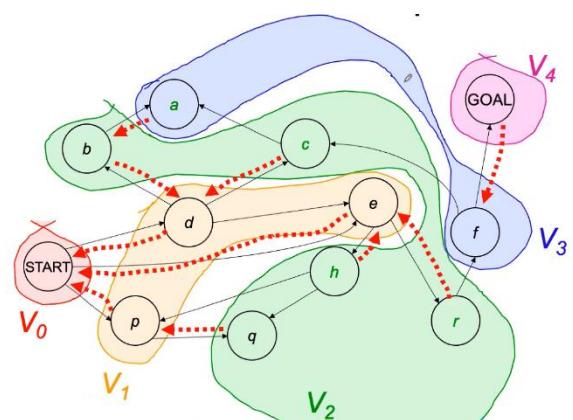
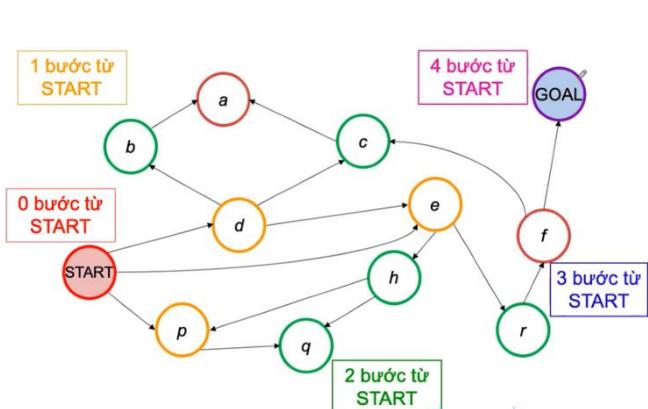


### 1.2. Breadth First Search (BFS)

❖ Ý tưởng

- Gán nhãn mọi trạng thái có thể đến được từ START trong i bước nhưng không thể ít hơn i bước.
- Lần lượt i đi từ 1 cho đến khi gặp GOAL

❖ Minh họa



### 1.3. Tìm kiếm theo chiều sâu giới hạn (Depth-Limited Search – DLS)

❖ Ý tưởng

- Tránh trường hợp cây có đường đi sâu vô hạn, ta đặt ra giới hạn độ sâu L
- Tại trạng thái có độ sâu bằng giới hạn đã đặt ra, nó sẽ không xét tiếp mà xem như trạng thái này không còn trạng thái con nào. Tức là sẽ không tăng thêm giới hạn L như IDS

### 1.4. Tìm kiếm lặp sâu dần (Iterative Deepening Search – IDS)

❖ Ý tưởng

- Tránh trường hợp cây có đường đi sâu vô hạn, ta đặt ra giới hạn độ sâu L
- Sử dụng DFS làm thủ tục con
- Thực hiện tìm các đường đi có độ dài  $\leq i$ . **Tăng dần** i từ 1 cho đến khi thành công
- Chi phí  $O(b^L)$

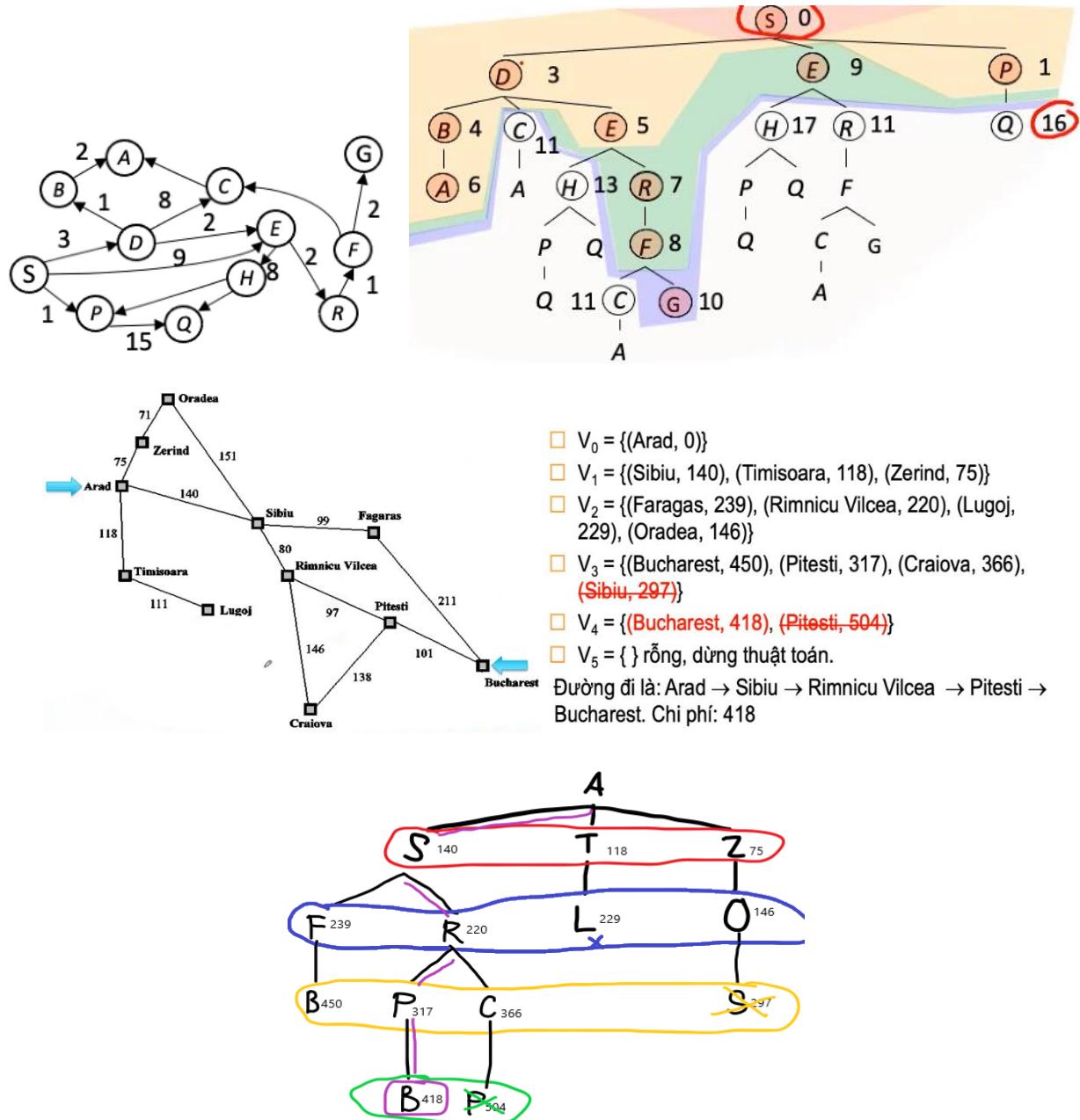
```
void DDS()
{
    MO = {T0}, ds = k;
    while MO ≠ ∅ do
    {
        n ← get (MO)
        if (n = g) then return True
        DONG = DONG ∪ {n}
        Case d(n) do
        {
            0.. ds - 1: Đặt B(n) vào đầu MO
            ds: Đặt B(n) vào cuối MO
            ds + 1:
            {
                ds = ds + k
                if K = 1 then đặt B(n) vào cuối tập MO
                else đặt B(n) vào đầu tập MO
            }
        }
    }
}
```

### 1.5. Tìm kiếm theo chiều rộng chi phí nhỏ nhất (Least Cost BFS)

❖ Ý tưởng

- Nhìn chung tương tự BFS
- Xét **tất cả** các nút đang mở, ta mở nút có chi phí hiện tại thấp nhất thay vì mở lần lượt như BFS
- Thêm các nút con vào tập trạng thái V<sub>k</sub> nếu chưa có, đồng thời **cập nhật** chi phí nếu tìm được đường đi ngắn hơn.

❖ Minh họa

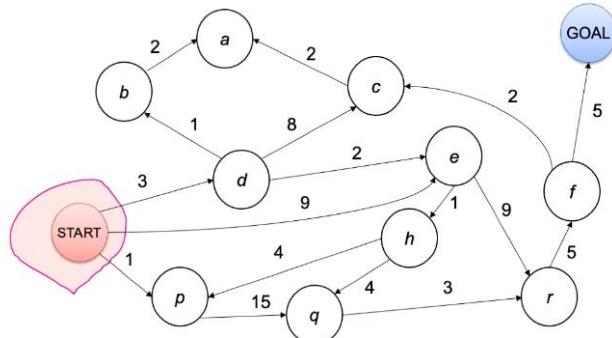
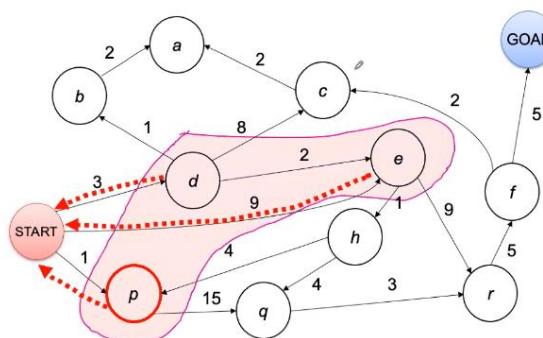
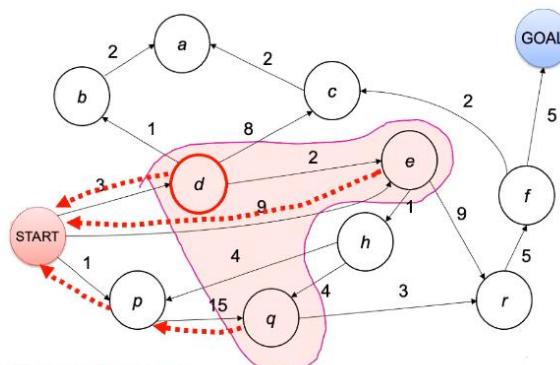
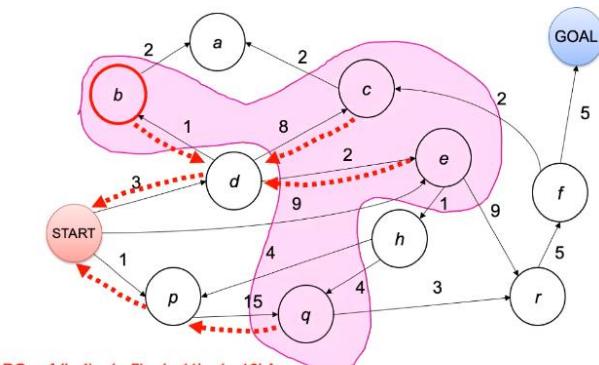
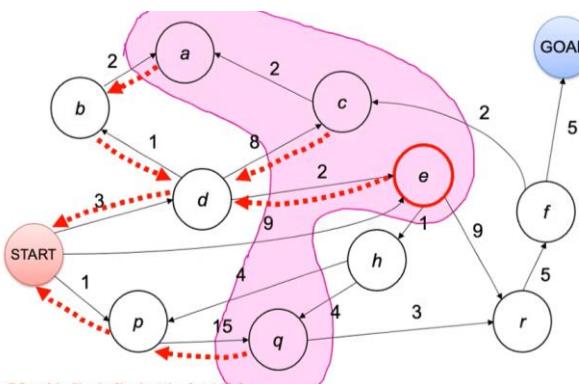
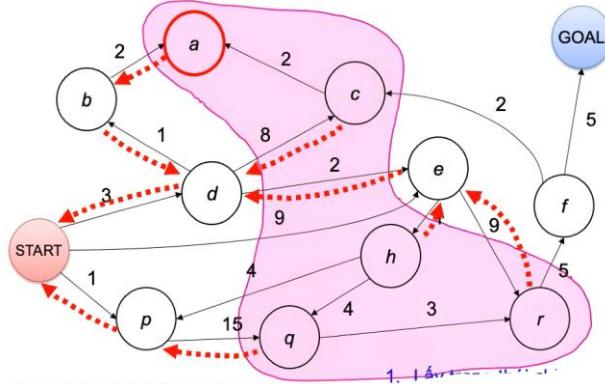
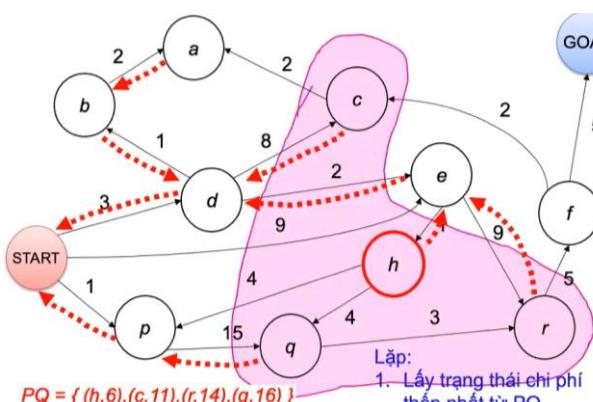
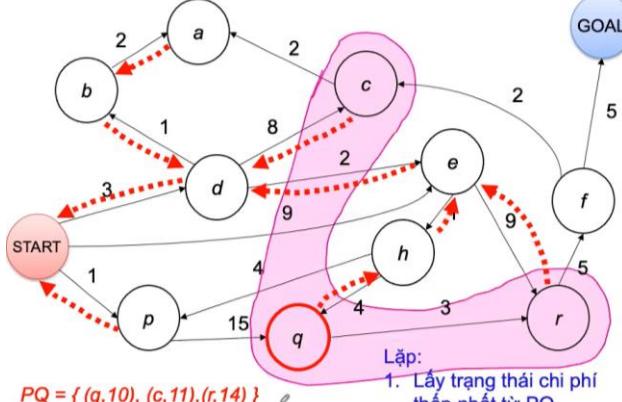


## 1.6. Tìm kiếm chi phí đồng nhất (Uniform Cost Search – UCS)

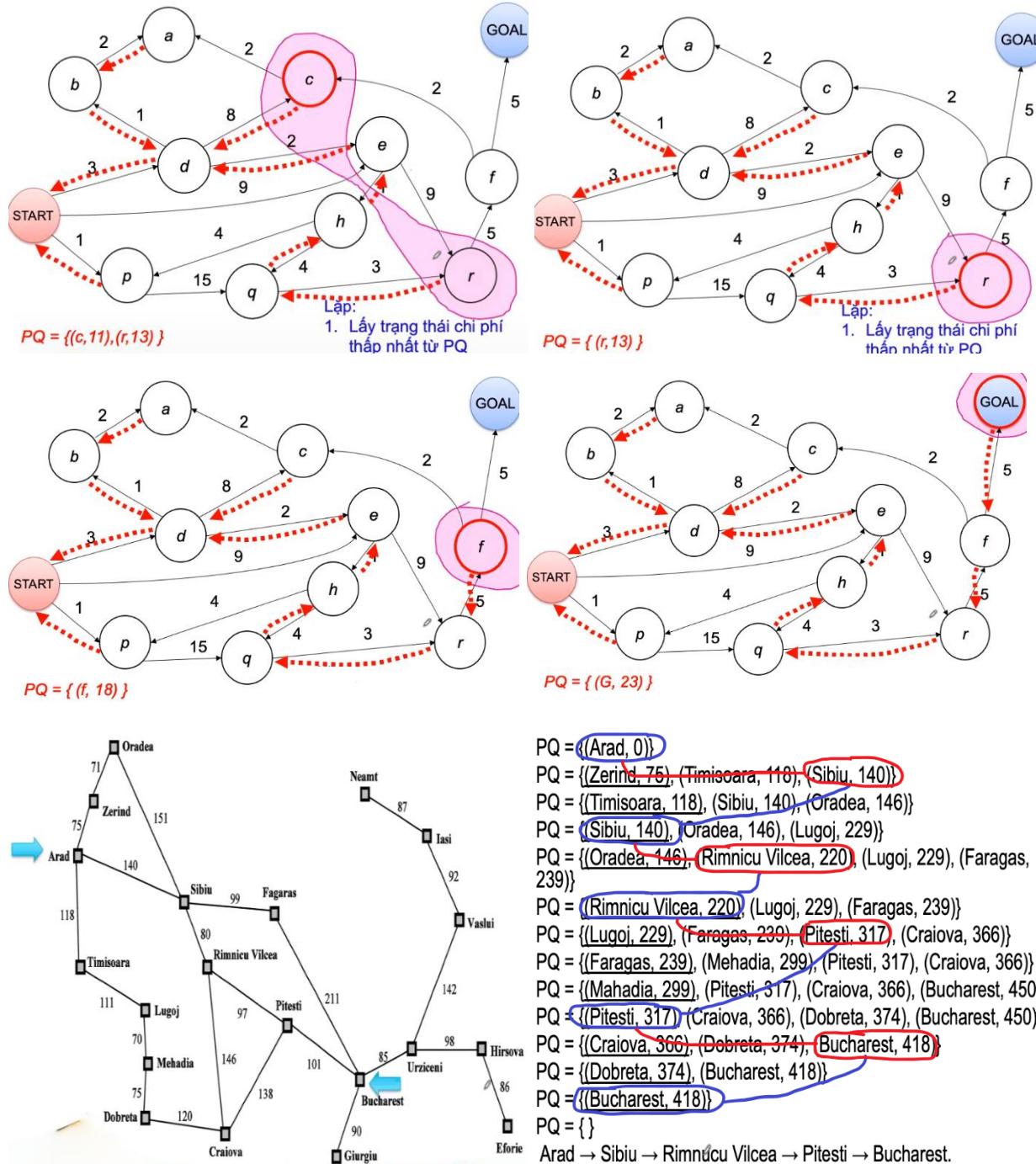
### ❖ Ý tưởng

- Sử dụng Priority Queue (PQ)
  - PQ: tập trạng thái đã mở hoặc đang đợi mở
  - Độ ưu tiên của trạng thái  $s = g(s) = \text{chi phí} \rightarrow \text{đi} \rightarrow s$  bằng đường đi quay lui
- Mở rộng trạng thái có chi phí thấp nhất trong PQ và thêm các con vào PQ (nếu chưa có). Nếu phát hiện đường đi có chi phí **thấp hơn** đường đi đã tìm thấy trước đó  $\rightarrow$  **cập nhật** chi phí  $\rightarrow$  độ ưu tiên thay đổi. Không dừng đến khi GOAL **đứng đầu** trong PQ

### ❖ Minh họa

 $PQ = \{ (S, 0) \}$  $PQ = \{ (p, 1), (d, 9), (e, 9) \}$  $PQ = \{ (d, 3), (e, 9), (q, 16) \}$  $PQ = \{ (b, 4), (e, 5), (c, 11), (q, 16) \}$  $PQ = \{ (e, 5), (a, 6), (c, 11), (q, 16) \}$  $PQ = \{ (a, 6), (h, 6), (c, 11), (r, 14), (q, 16) \}$  $PQ = \{ (h, 6), (c, 11), (r, 14), (q, 16) \}$  $PQ = \{ (q, 10), (c, 11), (r, 14) \}$ 

Lặp:  
1. Lấy trạng thái chi phí thấp nhất từ PQ



- ❖ Cách quay lui sau khi có được danh sách PQ
    - Giả sử nút G là phần tử đầu tiên của trạng thái cuối cùng, hay còn gọi là đích
    - **Nút trước G** là nút **đầu tiên** của trạng thái trước trạng thái đầu tiên mà G xuất hiện. Trong trường hợp G xuất hiện nhiều giá trị (do cập nhật đường đi), ta chỉ xét các trạng thái có chi phí nhỏ nhất
    - Bắt đầu từ GOAL, đệ quy cho đến khi gặp START

### 1.7. So sánh chi phí

N	số trạng thái trong bài toán
B	thừa số phân nhánh trung bình (số con trung bình) ( $B>1$ )
L	độ dài đường đi từ start đến goal với số bước (chi phí) ít nhất
LMAX	Độ dài đường đi dài nhất từ start đến bất cứ đâu
Q	kích cỡ hàng đợi ưu tiên trung bình

Thuật toán	Đủ	Tối ưu	Thời gian	Không gian
BFS	Breadth First Search	C	Nếu chi phí chuyển đổi như nhau	$O(\min(N, B^L))$
LCBFS	Least Cost BFS	C	C	$O(\min(N, B^L))$
UCS	Uniform Cost Search	C	C	$O(\log(Q) * \min(N, B^L))$
DFS	Depth First Search	K	K	N/A

Thuật toán	Đủ	Tối ưu	Thời gian	Không gian
BFS	Breadth First Search	C	Nếu chi phí chuyển đổi như nhau	$O(\min(N, B^L))$
LCBFS	Least Cost BFS	C	C	$O(\min(N, B^L))$
UCS	Uniform Cost Search	C	C	$O(\log(Q) * \min(N, B^L))$
DFS	Depth First Search	C	K	$O(B^{LMAX})$

Giả sử Không gian Tìm kiếm không chu trình

109

N	số trạng thái trong bài toán
B	thừa số phân nhánh trung bình (số con trung bình) ( $B>1$ )
L	độ dài đường đi từ start đến goal với số bước (chi phí) ít nhất
LMAX	Độ dài đường đi <b>không chu trình</b> dài nhất từ start đến bất cứ đâu
Q	kích cỡ hàng đợi ưu tiên trung bình

Thuật toán		Đủ	Tối ưu	Thời gian	Không gian
BFS	Breadth First Search	C	Nếu chi phí chuyển đổi như nhau (1)	$O(\min(N, B^L))$	$O(\min(N, B^L))$
LCBFS	Least Cost BFS	C	C	$O(\min(N, B^L))$	$O(\min(N, B^L))$
UCS	Uniform Cost Search	C	C	$O(\log(Q) * \min(N, B^L))$	$O(\min(N, B^L))$
PCDFS	Path Check DFS	C	K	$O(B^{LMAX})$	$O(LMAX)$
MEMDFS	Memoizing DFS	C	K	$O(\min(N, B^{LMAX}))$	$O(\min(N, B^{LMAX}))$
ID	Iterative Deepening	C	(1)	$O(B^L)$	$O(L)$

- ❖ PCDFS (Path Checking DFS): không đệ quy một trạng thái nếu trạng thái này đã nằm trong đường đi hiện tại
- ❖ MEMDFS (Memorizing DFS): ghi nhớ **mọi** trạng thái đã mở, không mở hai lần

## 2. Tìm kiếm heuristic (Heuristic/Informed Search)

Heuristic là phương pháp giải quyết vấn đề dựa trên phỏng đoán, ước chừng, kinh nghiệm, trực giác để tìm ra giải pháp gần như là tốt nhất (chứ không phải luôn luôn là tốt nhất) và nhanh chóng. Hàm Heuristic là hàm ứng với mỗi trạng thái hay mỗi sự lựa chọn một giá trị ý nghĩa đối với vấn đề dựa vào giá trị hàm này ta lựa chọn hành động.

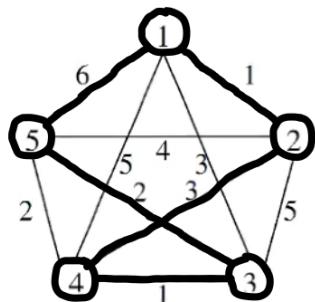
### 2.1. Greedy Traveling Saleman (GTS)

#### 2.1.1. Đặt vấn đề

- Cho một đồ thị với  $N$  đỉnh, trong đó hai đỉnh bất kỳ đều nối với nhau
- Xuất phát từ 1 đỉnh, đi qua tất cả các đỉnh còn lại 1 lần duy nhất, rồi quay về đỉnh xuất phát
- Xác định đường đi sao cho tổng chi phí là **thấp nhất**

#### 2.1.2. GTS 1

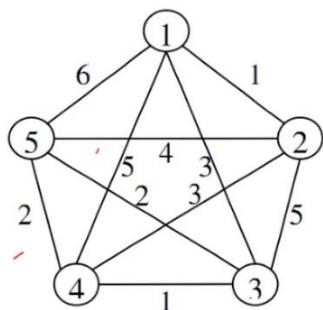
- ❖ Ý tưởng
  - Có gắng đạt được lời giải tốt nhất bằng cách tại mỗi bước đi, chọn đường đi có chi phí **thấp nhất** tại đường đi **hiện tại** và **tiếp tục đi**.
- ❖ Minh họa



Khởi đầu: Tour :=  $\emptyset$ , Cost := 0, V := 1  
 Chọn W = 2: Tour = {(1,2)}, Cost = 1  
 Chọn W = 4: Tour = {(1,2), (2,4)}, Cost = 4  
 Chọn W = 3: Tour = {(1,2), (2,4), (4,3)}, Cost = 5  
 Chọn W = 5: Tour = {(1,2), (2,4), (4,3), (3,5)}, Cost = 7  
 Trở về: Tour = {(1,2), (2,4), (4,3), (3,5), (5,1)}, Cost = 13  
 Chu trình:  $1 \rightarrow 2 \rightarrow 4 \rightarrow 3 \rightarrow 5 \rightarrow 1$ , chi phí là 13.

#### 2.1.3. GTS 2

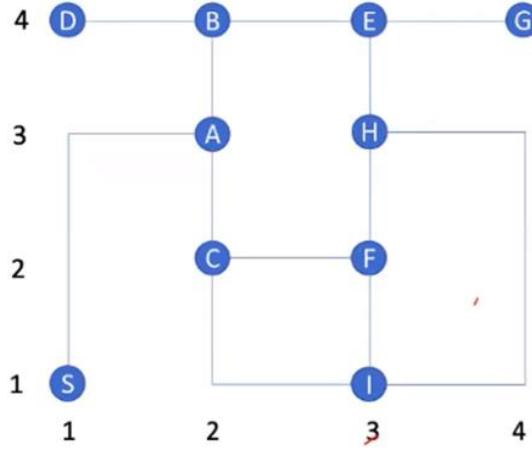
- ❖ Ý tưởng
  - GTS1 không đảm bảo tổng chi phí là thấp nhất
  - Với  $P$  cho trước ( $1 < P < N$ ), ta chọn  $P$  đỉnh và đi như GTS1, khi đó, ta có được  $P$  cách đi và chỉ giữa lại cách đi cho giá trị nhỏ nhất
- ❖ Minh họa



- Chọn  $P = 3$  đỉnh xuất phát khác nhau, kết quả thực hiện của thuật toán là:
  - ▢ Chọn xuất phát từ thành phố 1:
    - Chu trình:  $1 \rightarrow 2 \rightarrow 4 \rightarrow 3 \rightarrow 5 \rightarrow 1$ , chi phí là 13.  $\text{Cost}(1)=13$
    - $\text{Cost} = \text{Cost}(1) = 13$ .
  - ▢ Chọn xuất phát từ thành phố 2:
    - Chu trình:  $2 \rightarrow 1 \rightarrow 3 \rightarrow 4 \rightarrow 5 \rightarrow 2$ , chi phí là 11.  $\text{Cost}(2)=11$
    - $\text{Cost}(2)=11 < 13 = \text{Cost}(1) \Rightarrow \text{Cost} = \text{Cost}(2)=11$ .
  - ▢ Chọn xuất phát từ thành phố 4:
    - Chu trình:  $4 \rightarrow 3 \rightarrow 5 \rightarrow 2 \rightarrow 1 \rightarrow 4$ , chi phí là 13.  $\text{Cost}(4)=13$
    - $\text{Cost}=11 < 13 = \text{Cost}(4) \Rightarrow \text{Cost} = 11$
- Vậy lời giải tốt nhất là xuất phát từ thành phố 2 với chi phí là 11

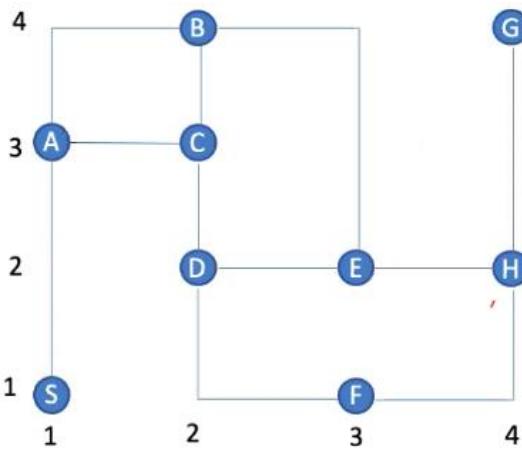
## 2.2. Hill Climbing (thuật toán leo đồi)

- ❖ Ý tưởng
  - Lựa chọn những đỉnh mà được dự đoán trước là gần tới đích nhất
  - Tính  $h$  cho những đỉnh con của đỉnh hiện tại và chọn đỉnh có  $h$  nhỏ nhất làm đỉnh kế
  - Bản chất của Hill Climbing là **greedy**. Tuy nhiên, nó **không** lưu lại nút con đã xét
- ❖ Thuật giải
  - Tại đỉnh  $N$  hiện tại
    - Nếu đỉnh hiện tại là đích  $\rightarrow$  thoát  $\rightarrow$  thành công
    - Ngược lại, tính  $h(N_i)$  cho tất cả các đỉnh con của  $N$  và chọn đỉnh cho ra giá trị nhỏ nhất làm  $next N$ . Nếu  $h(N) < h(next N)$   $\rightarrow$  thoát  $\rightarrow$  thất bại
    - Đỉnh kế tiếp là  $N = next N$
  - Lặp lại xét tiếp với đỉnh  $N$  mới
- ❖ Vấn đề của Hill Climbing
  - Mắc kẹt ở **cực trị địa phương** và các **vùng phẳng**. Có thể khắc phục bằng thuật giải **di truyền** – lai ghép và đột biến
- ❖ Minh họa



$h(n) = |\text{Tọa độ } x \text{ của đích} - \text{Tọa độ } x \text{ của } n| + |\text{Tọa độ } y \text{ của đích} - \text{Tọa độ } y \text{ của } n|$

- $h(S) = |4 - 1| + |4 - 1| = 6$   
 $h(A) = |4 - 2| + |4 - 3| = 3$   
 $h(A) = 3 < 6 = h(S) \Rightarrow A := \text{next } S$
- $h(B) = |4 - 2| + |4 - 4| = 2$   
 $h(C) = |4 - 2| + |4 - 2| = 4$   
 $\text{Min}(2, 4) = 2 = h(B) < 3 = h(A) \Rightarrow B := \text{next } A$
- $h(D) = |4 - 1| + |4 - 4| = 3$   
 $h(E) = |4 - 3| + |4 - 4| = 1$   
 $\text{Min}(1, 3) = 1 = h(E) < 2 = h(B) \Rightarrow E := \text{next } B$
- $h(G) = |4 - 4| + |4 - 4| = 0$   
 $h(H) = |4 - 3| + |4 - 3| = 2$   
 $\text{Min}(0, 2) = 0 = h(G) < 1 = h(E) \Rightarrow G := \text{next } E$



$h(n) = |\text{Tọa độ } x \text{ của đích} - \text{Tọa độ } x \text{ của } n| + |\text{Tọa độ } y \text{ của đích} - \text{Tọa độ } y \text{ của } n|$

- $h(S) = |4 - 1| + |4 - 1| = 6$   
 $h(A) = |4 - 1| + |4 - 3| = 4$   
 $h(A) = 4 < 6 = h(S) \Rightarrow A := \text{next } S$
- $h(B) = |4 - 2| + |4 - 4| = 2$   
 $h(C) = |4 - 2| + |4 - 3| = 3$   
 $\text{Min}(2, 3) = 2 = h(B) < 4 = h(A) \Rightarrow B := \text{next } A$
- $h(E) = |4 - 3| + |4 - 2| = 3$   
Do:  $h(E) = h(C) = 3 > 2 = h(B) \Rightarrow \text{Exit!}$

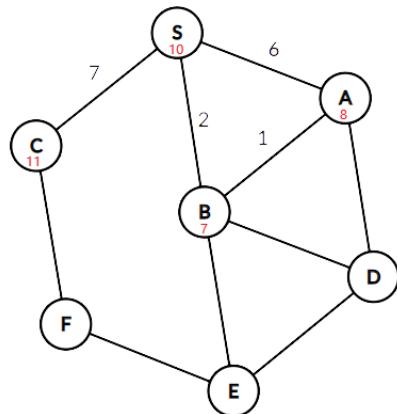
## 2.3. A-star

### ❖ Ý tưởng

- Tại đỉnh N hiện tại
  - Xét tất cả các đỉnh con của N và xác định  $f(N_i) = g(N_i) + h(N_i)$ , trong đó
    - g: chi phí đường đi từ đỉnh gốc đến đỉnh này
    - h: kết quả trả về của hàm heuristic tại đỉnh này
  - Mở rộng về đỉnh có f nhỏ nhất và lặp lại cho đến khi đến đích

### ❖ Lưu ý

- Không bỏ qua** các đỉnh trước đó đã tính toán, mà cần **cập nhật** giá trị của chúng nếu tìm được đường đi **tốt hơn**.
- Muốn xác định tốt hơn hay không, hãy so sánh **chi phí**, tức so sánh g, không quan tâm f hoặc h, vì lúc này f phụ thuộc vào g, còn h thì không đổi



Path = <S>,  $f(A)=6+8=14$ ,  $f(B)=2+7=9$ ,  $f(C)=7+11=18$

→ Chọn B

Path = <S, B>,  $f(A)=2+1+8=11 \rightarrow$  cập nhật giá trị của A

- Khi một nút đã được chọn trước đó và nó không còn nút con nào để triển khai, ta thêm nút đó vào đường đi
- Ta có thể thêm các nút đã được tính toán vào một Priority Queue với độ ưu tiên là  $f$
- ❖ Tham khảo thêm
  - [A\\* \(A Star\) Search Algorithm - Computerphile](#)

## 2.4. Greedy Best First Search

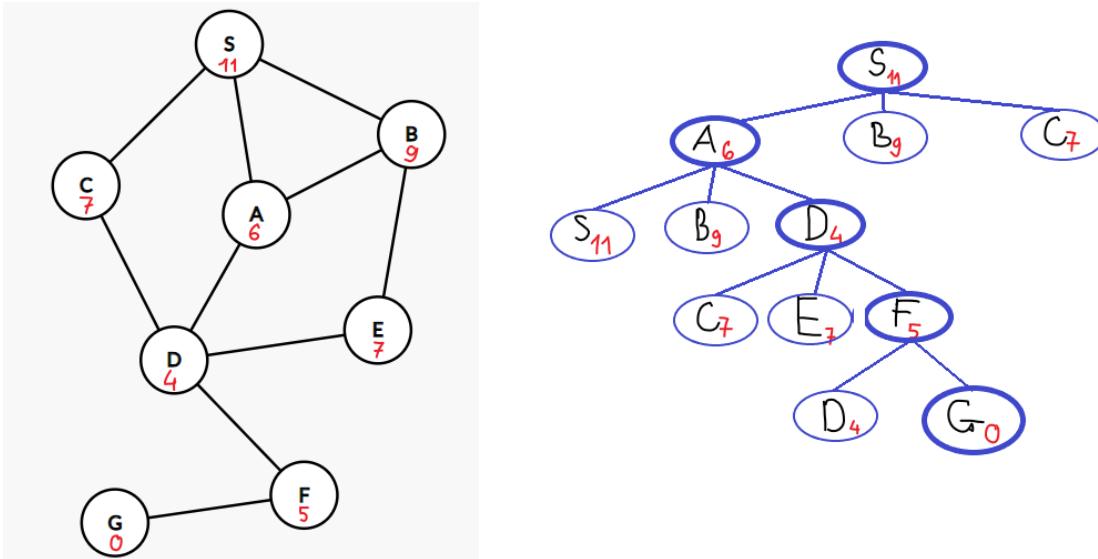
### ❖ Ý tưởng

- Với mỗi nút con của nút hiện tại đang xét, ta mở rộng nút được dự đoán trước là gần tới đích nhất, dựa vào hàm heuristic  $f(n) = h(n)$
- Tiếp tục mở rộng cho đến khi tới đích
- **Không** bỏ qua những nút đã ghé thăm

### ❖ Lưu ý

- GBFS có thể mắc kẹt trong vòng lặp → không cho ra kết quả
- Không đảm bảo đường đi tìm được là ngắn nhất

### ❖ Minh họa



## 2.5. Beam Search

### ❖ Ý tưởng

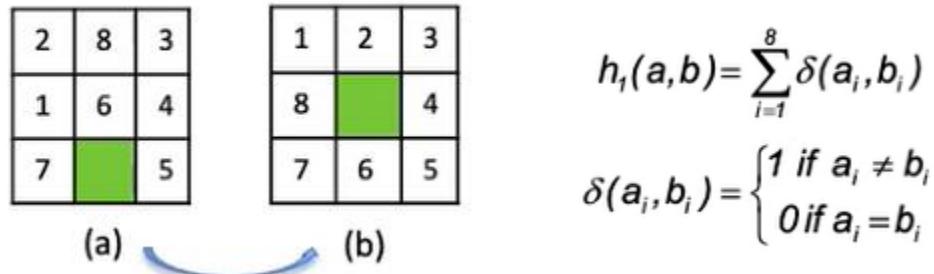
- Tương tự như GBFS, tuy nhiên thay vì chỉ mở nút tối ưu nhất, ta mở K nút có kết quả tốt nhất (K cho trước, còn gọi là beam width)
- Không lưu lại các nút con đã mở trước đó. Sau khi chọn được K nút tốt nhất, các nút khác bị bỏ đi
- Có thể xem phiên bản tối ưu của Best First Search
  - Khi  $K = 1$ , nó là Hill Climbing
  - Khi  $K = \text{infinity}$ , nó là Breadth First Search
- Đây là thuật toán local search, tức là không đảm bảo trả ra đường đi đến đích. Nguyên nhân là vì tại mỗi vòng lặp beam search chỉ lưu lại k node tối ưu nhất (của cả pool node con). Khi chỉ lấy K node tối ưu nhất, ta có thể đã bỏ qua path đến đích.
- Tại mỗi vòng lặp, beam search duyệt một lúc K node  $\Rightarrow$  mở ra được n node con của K node ấy  $\Rightarrow$  chọn ra K node tối ưu nhất trong n node con và tiếp tục vòng lặp.
- Thuật toán dừng khi:
  - OPEN rỗng (failed)
  - Goal được đưa ra khỏi OPEN (backtrace path và return kết quả)
- Là phiên bản tối ưu của Best First Search **về mặt lưu trữ**
- Khi  $K = 1$ , nó là Hill climbing
- ❖ Video ví dụ: [https://youtu.be/io13ps0\\_Fyw](https://youtu.be/io13ps0_Fyw)

## 2.6. Một số bài toán

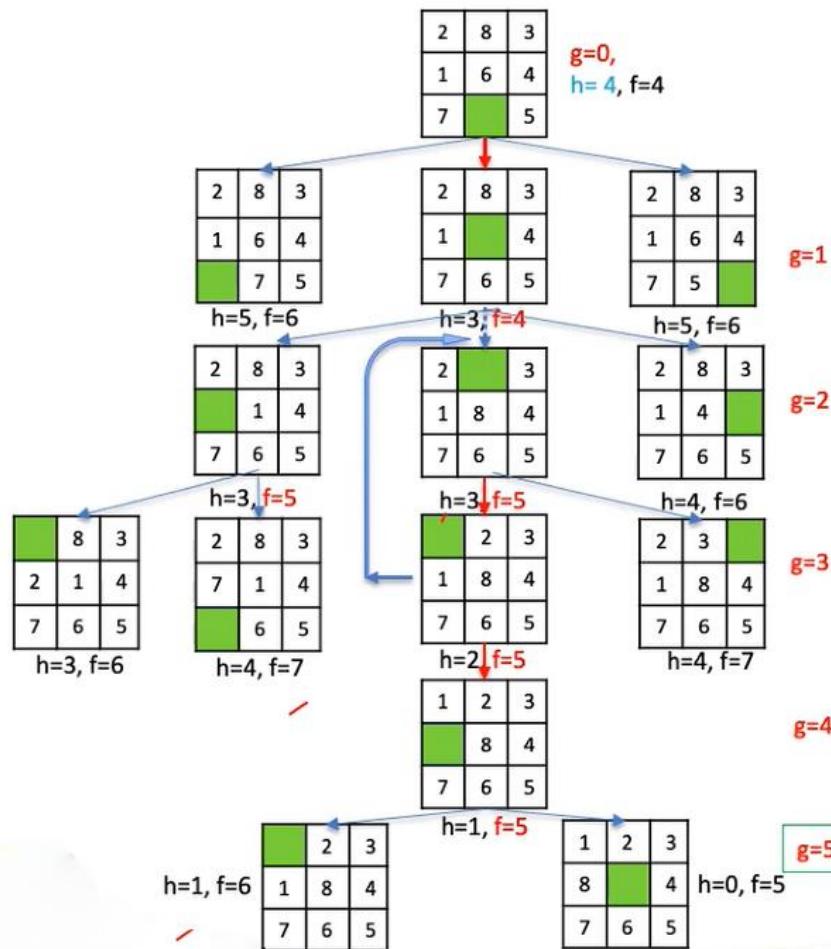
### 2.6.1. Bài toán 8-PUZZLE

❖ Cách làm 1

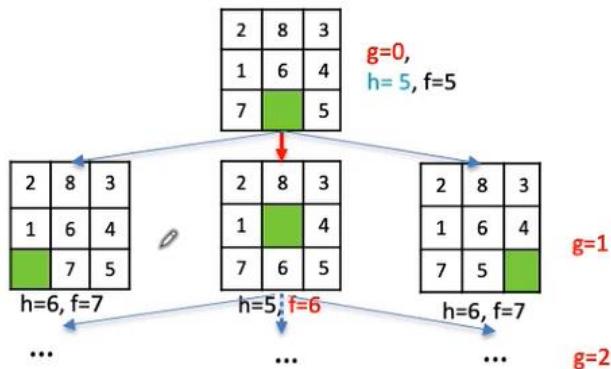
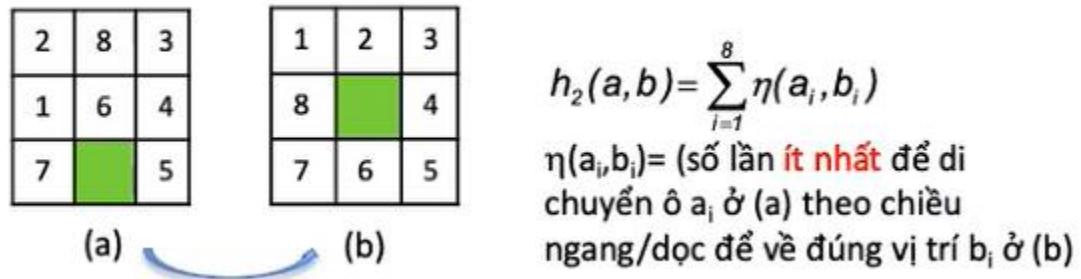
- Đưa từ trạng thái a về trạng thái b. Xét hàm  $h_1$  tại mỗi trạng thái



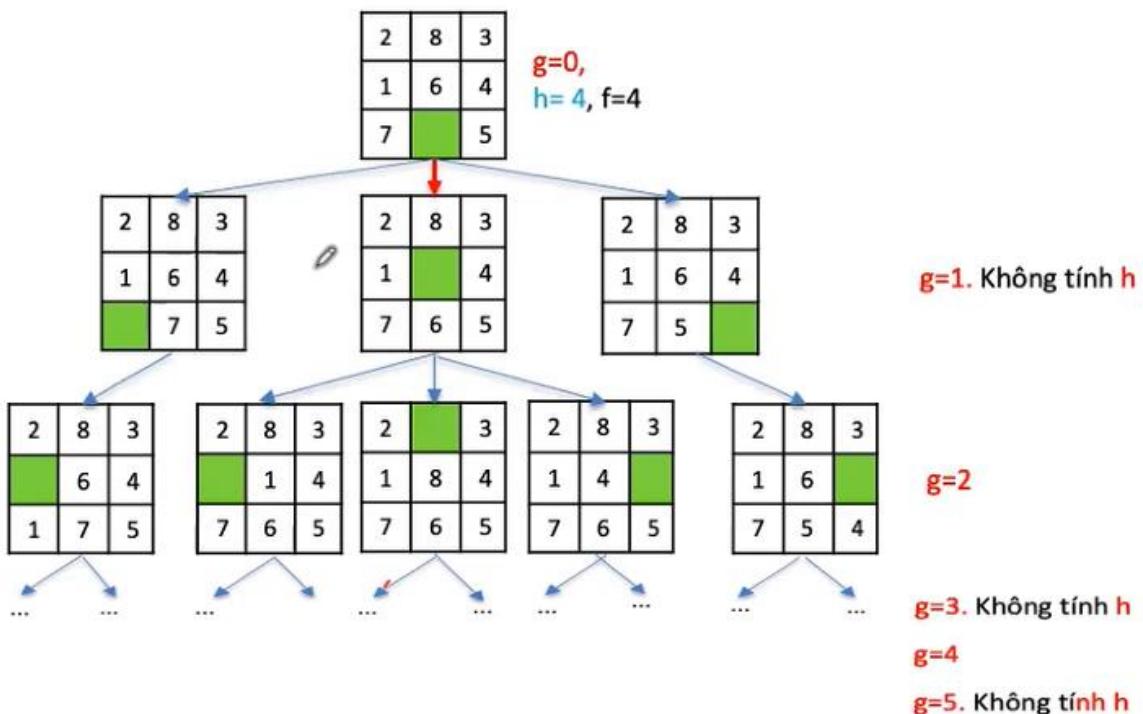
- Ta chọn các trạng thái tiếp theo có  $f = g + h$  bé nhất. Tránh chọn trạng thái đã có
- Trong trường hợp  $f$  bằng nhau, ta “chọn” cả hai và so sánh  $f$  của trạng thái tiếp theo



- Ngoài hàm  $h_1$ , ta có thể xét hàm  $h_2$  tại mỗi trạng thái



- Một cách làm khác, ta có thể không tính  $h$  ở những bước có  $g$  là số lẻ để tránh việc  $f$  trùng nhau. Tuy nhiên, việc này có thể làm ta bỏ qua lời giải nếu có một trạng thái  $h = 0$  khi  $g$  lẻ.



### 2.6.2. Bài toán tháp Hà Nội

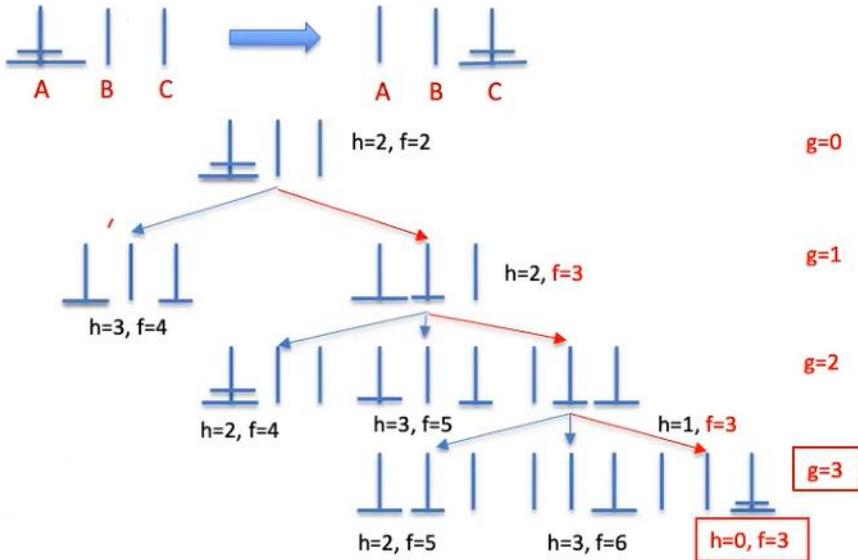
❖ Tổng quát

- Với  $n$  đĩa, ta có  $2^n - 1$  số lần dịch chuyển (đệ quy)

□ Xét  $n=2$

Xét tại c:

$$h(n) = \begin{cases} 0: & \text{ } \\ 1: & \text{ } \\ 2: & \text{ } \\ 3: & \text{ } \end{cases}$$



### 2.6.3. Bài toán phân công việc

❖ Đặt vấn đề

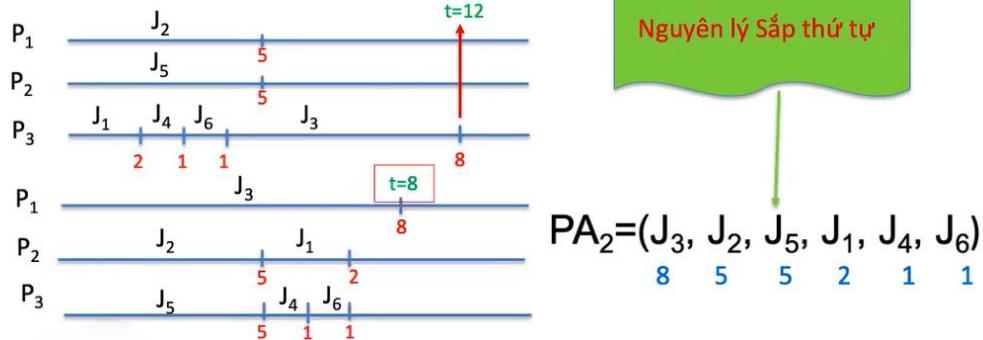
- Giả sử có  $n$  máy như nhau, có  $m$  công việc, mỗi việc cần t thời gian để hoàn thành
- Các công việc được xử lý đồng thời. Mọi máy đều có thể xử lý công việc. Thời gian đưa công việc lên các máy là 0. Các công việc không thể gián đoạn.
- Hãy lên phương án sao cho tổng thời gian thực hiện là ít nhất

❖ Mô hình 1

- PA1: Nếu có máy nào rảnh thì nạp công việc kế tiếp trong danh sách L vào (nếu có 2 hay nhiều máy cùng rảnh tại một thời điểm thì máy với chỉ số thấp sẽ được phân cho công việc). Sap xếp các công việc một cách ko có trình tự
- PA2: Là phương án mà các công việc được sắp theo thứ tự thời gian giảm dần. Tuy nhiên heuristic này không chắc đã có một phương án tối ưu.

- 03 xử lý:  $P_1, P_2, P_3$
- 06 công việc:  $t_1=2, t_2=5, t_3=8, t_4=1, t_5=5, t_6=1$

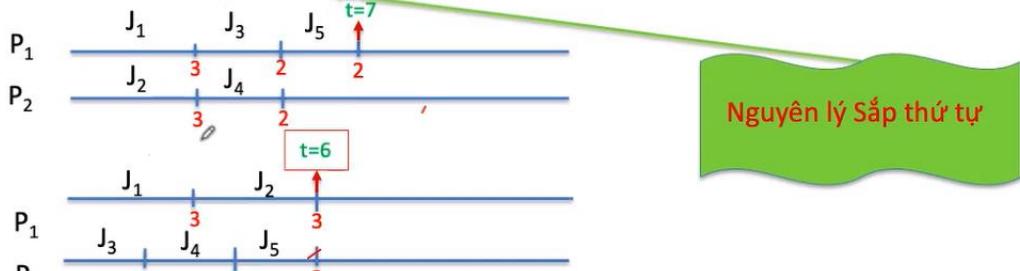
**□ PA<sub>1</sub>: (J<sub>2</sub>, J<sub>5</sub>, J<sub>1</sub>, J<sub>4</sub>, J<sub>6</sub>, J<sub>3</sub>)**



❖ Mô hình 2

- 02 xử lý:  $P_1, P_2$
- 05 công việc:  $t_1=3, t_2=3, t_3=2, t_4=2, t_5=2$

**□ PA<sub>1</sub>: (J<sub>1</sub>, J<sub>2</sub>, J<sub>3</sub>, J<sub>4</sub>, J<sub>5</sub>)**



**□ PA<sub>2</sub>: (J<sub>1</sub>, J<sub>3</sub>, J<sub>4</sub>, J<sub>2</sub>, J<sub>5</sub>)**

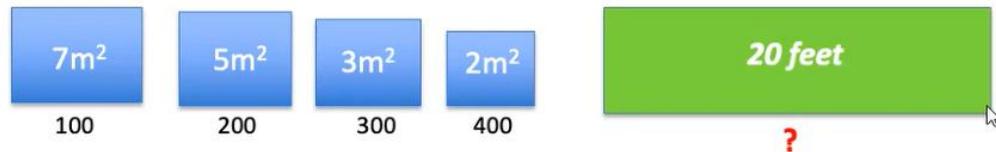
❖ Nhận xét

- Các phương án cho ra lời giải tốt chứ không tốt nhất
- Chỉ có thể cho ra lời giải tốt nhất cho một (vài) trường hợp

#### 2.6.4. Bài toán đóng gói

(Ai có link bài toán này cho mình xin với, đọc ko hiểu đê)

□ Chúng ta có các loại thùng như sau:

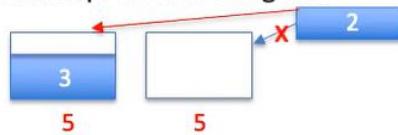


H<sub>1</sub>: Thứ tự các đối tượng: tùy ý

Cách xếp: theo thứ tự tùy ý

H<sub>3</sub>: Thứ tự các đối tượng: tùy ý

Cách xếp: ưu tiên trùng khít



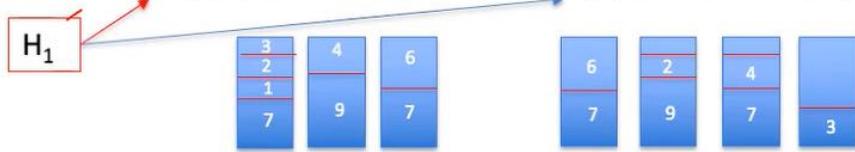
H<sub>2</sub>: Thứ tự các đối tượng: sắp theo thứ tự giảm dần

Cách xếp: theo thứ tự giảm dần

H<sub>4</sub>: Thứ tự các đối tượng: sắp theo thứ tự giảm dần

Cách xếp: ưu tiên trùng khít

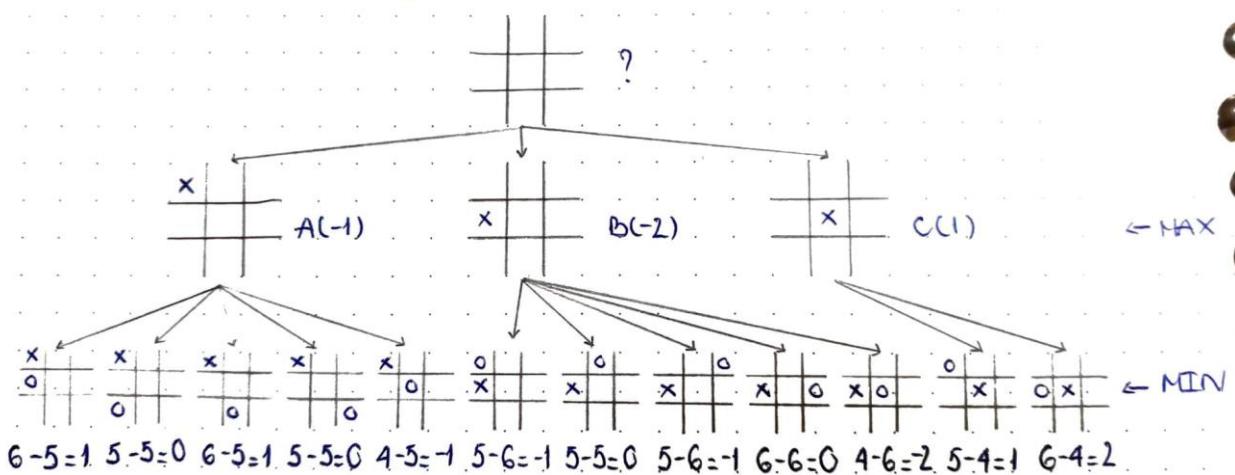
Xét bài toán: (7, 9, 7, 1, 6, 2, 4, 3)      13  
PA<sub>1</sub>: (7, 9, 7, 1, 6, 2, 4, 3)      PA<sub>2</sub>: (7, 9, 7, 6, 2, 4, 3)



### 3. Tìm kiếm đối kháng (Adversarial Search)

#### 3.1. Tổng quan

- ❖ Các tìm kiếm trong đó hai hoặc nhiều người chơi có mục tiêu xung đột đang cố gắng khám phá cùng 1 không gian tìm kiếm cho giải pháp, được gọi là **tìm kiếm đối kháng**, thường được gọi là **Trò chơi**.
- ❖ Cây trò chơi
  - Cây trò chơi = trạng thái ban đầu + luật di chuyển
  - Các nút của cây là trạng thái trò chơi  
Cạnh là các bước di chuyển của người chơi  
Liên quan đến **trạng thái ban đầu**, **hàm hành động** và **hàm kết quả**.
  - Ví dụ bài toán tic-tac-toe
 
$$f(x) = (\text{Số dòng hoặc số cột hoặc số đường chéo còn mở với MAX}) - (\text{Số dòng hoặc số cột hoặc số đường chéo còn mở với MIN})$$



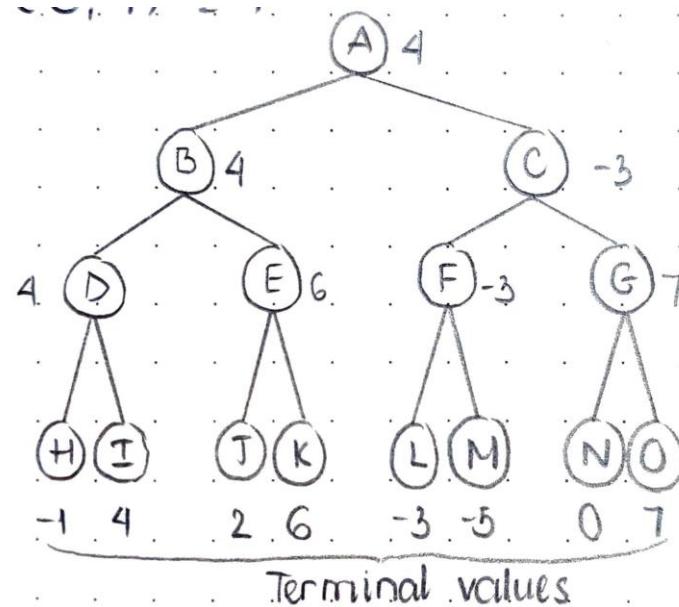
❖ Chiến lược tối ưu

- Có thể được xác định từ giá trị MINIMAX của mỗi nút, có thể được viết là  $\text{MINIMAX}(n)$ .
- **MAX** chuyển sang trạng thái giá trị tối đa và **MIN** chuyển sang trạng thái giá trị tối thiểu sau đó.

### 3.2. Thuật toán Mini – Max

- ❖ Là thuật toán **dệ quy** hoặc **quay lui** được sử dụng trong lý thuyết trò chơi và ra quyết định.  
 ❖ Sử dụng đệ quy để tìm kiếm thông qua cây trò chơi.  
 ❖ Các bước chính

- Bước 1:
  - Tạo toàn bộ cây trò chơi và áp dụng hàm tiện ích để lấy các giá trị tiện ích cho các trạng thái đầu cuối.
  - Giả sử Maximizer thực hiện lượt đầu tiên có giá trị ban đầu trong trường hợp xấu nhất bằng  $-\infty$  và Minimizer sẽ thực hiện lượt tiếp theo có giá trị ban đầu trong trường hợp xấu nhất bằng  $+\infty$ .
- Bước 2: tìm giá trị tiện ích cho Maximizer (lượt Maximizer)
  - Đối với nút D:  $\max(-1, -\infty) \rightarrow \max(-1, 4) = 4$
  - Đối với nút E:  $\max(2, -\infty) \rightarrow \max(2, 6) = 6$
  - Đối với nút F:  $\max(-3, -\infty) \rightarrow \max(-3, -5) = -3$
  - Đối với nút G:  $\max(0, -\infty) \rightarrow \max(0, 7) = 7$
- Bước 3: tìm giá trị tiện ích cho Minimizer (lượt Minimizer)
  - Đối với nút B:  $\min(4, 6) = 4$
  - Đối với nút C:  $\min(-3, 7) = -3$
- Bước 4: lượt Maximizer
  - Đối với nút A:  $\max(4, -3) = 4$



❖ Tính chất thuật toán

- **Đầy đủ:** thuật toán Mini – max chắc chắn tìm ra 1 lời giải (nếu tồn tại) trong cây tìm kiếm hữu hạn.
- **Tối ưu:** tối ưu nếu cả hai đối thủ đều chơi tối ưu.
- **Độ phức tạp thời gian:** thực hiện DFS cho cây trò chơi  $\rightarrow O(b^m)$  với  $b$  là hệ số phân nhánh của cây trò chơi và  $m$  là độ sâu tối đa cây.
- **Độ phức tạp không gian:** tương tự DFS là  $O(bm)$ .

❖ Giới hạn

- **Rất chậm** đối với các trò chơi phức tạp
- Cải thiện từ việc **cắt tỉa alpha – beta**.

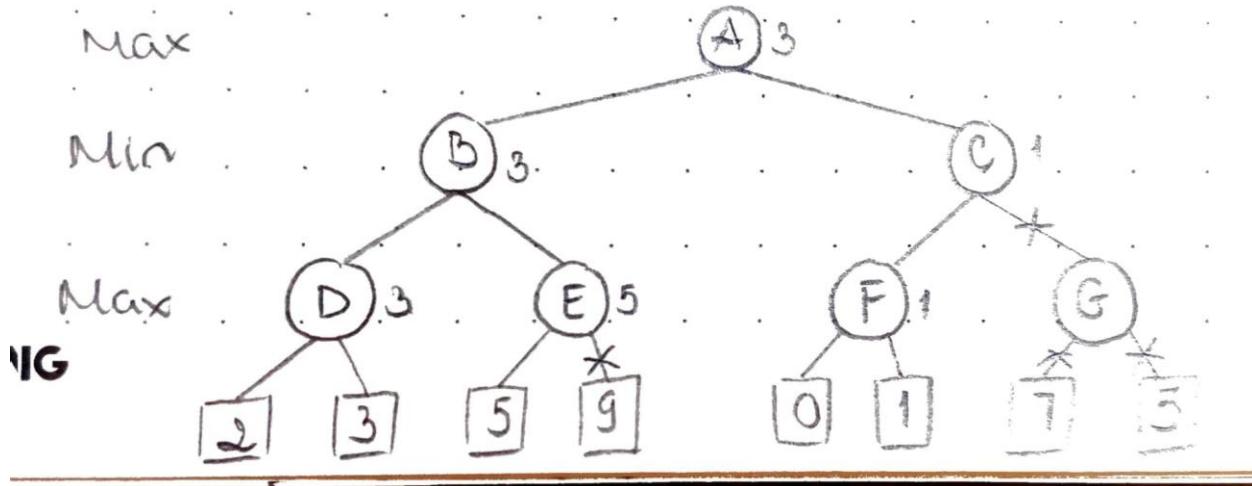
### 3.3. Cắt tỉa alpha – beta

❖ Giới thiệu tổng quan

- Kỹ thuật tối ưu hoá cho thuật toán Mini – Max.
- **Cắt tỉa:** không cần kiểm tra từng nút của cây trò chơi, chúng ta có thể tính toán quyết định Mini – max chính xác.
- Có thể áp dụng ở **bất kỳ độ sâu nào** của cây và đôi khi nó không chỉ tia lá cây mà còn toàn bộ cây phụ.
- Hai tham số được định nghĩa
  - **Alpha** sự lựa chọn tốt nhất (giá trị cao nhất) mà chúng ta đã tìm thấy cho đến nay, tại bất kỳ điểm nào trên con đường của Maximizer. Giá trị ban đầu là  $-\infty$ .
  - **Beta** sự lựa chọn tốt nhất (giá trị thấp nhất) mà chúng ta đã tìm thấy cho đến nay, tại bất kỳ điểm nào trên con đường của Minimizer. Giá trị ban đầu là  $+\infty$ .

- Kết quả giống thuật toán chuẩn, nhưng loại bỏ tất cả các nút không thực sự ảnh hưởng đến quyết định cuối cùng nhưng làm cho thuật toán chậm. → nhanh hơn.
- ❖ Điều kiện dừng:  $\alpha \geq \beta$
- ❖ Điểm chính
  - Người chơi **Max** sẽ chỉ cập nhật giá trị của **alpha**
  - Người chơi **Min** sẽ chỉ cập nhật giá trị của **beta**
  - Trong khi quay lại cây, các **giá trị nút** sẽ được chuyển đến các nút trên thay vì giá trị của alpha và beta.
  - Chỉ chuyển các giá trị alpha, beta cho các **nút con**.
- ❖ Hoạt động
  - Bước 1
    - Người chơi Max bắt đầu di chuyển đầu tiên từ nút A trong đó  $\alpha = -\infty$  và  $\beta = +\infty$ .
    - Các giá trị được truyền xuống nút B và nút B truyền cùng giá trị cho con của nó – nút D.
  - Bước 2
    - Tại nút D, giá trị của  $\alpha$  sẽ được tính là lần lượt cho Max.
    - Giá trị của  $\alpha$  được so sánh với đầu tiên là 2 và sau đó là 3 và giá trị tối đa  $(2, 3) = 3$  sẽ là giá trị  $\alpha$  tại nút D và giá trị nút cũng sẽ là 3.
  - Bước 3
    - Thuật toán quay lại nút B, trong đó giá trị của  $\beta$  sẽ thay đổi vì đây là lượt của Min.
    - Bây giờ  $\beta = +\infty$ , so sánh với giá trị các nút tiếp theo có sẵn, tức là  $\min(\infty, 3) = 3$ , do đó tại nút B bây giờ là  $\alpha = -\infty$  và  $\beta = 3$ .
    - Trong bước tiếp theo, thuật toán đi qua nút tiếp theo của nút B là nút E và các giá trị của  $\alpha = -\infty$  và  $\beta = 3$  cũng sẽ được truyền qua.
  - Bước 4
    - Tại nút E, max sẽ thực hiện lần lượt và giá trị alpha thay đổi.
    - Giá trị hiện tại của alpha được so sánh với 5, do đó  $\max(-\infty, 5) = 5$ , khi đó tại nút E:  $\alpha = 5$  và  $\beta = 3$ , trong đó  $\alpha \geq \beta$   
 $\Rightarrow$  Nút tiếp theo của E sẽ được cắt tỉa, thuật toán sẽ không đi qua nó và giá trị tại nút E là 5.
  - Bước 5
    - Thuật toán quay lại cây, từ nút B đến nút A.
    - Tại nút A, giá trị alpha sẽ được thay đổi  $\alpha = 3$  và  $\beta = +\infty$ , hai giá trị được chuyển cho nút C.
    - Tại nút C,  $\alpha = 3$  và  $\beta = +\infty$  và các giá trị sẽ được truyền cho nút F.
  - Bước 6
    - Tại nút F, giá trị của  $\alpha$  được so sánh với con trái là 0 và  $\max(0, 3) = 3$  và sau đó so sánh với con phải là 1 và  $\max(3, 1) = 3$  vẫn là 3, nhưng giá trị của nút F sẽ trở thành 1.

- Bước 7
  - Nút F trả về giá trị nút 1 cho nút C, tại C:  $\alpha = 3$  và  $\beta = +\infty$ , so sánh với 1 nên  $\min(+\infty, 1) = 1$ .
  - Bây giờ tại C  $\alpha = 3$  và  $\beta = 1$  thoả  $\alpha \geq \beta$ , vì vậy nút con tiếp theo của C là G sẽ được cắt tỉa và thuật toán không tính toàn bộ cây con G.
- Bước 8
  - C bây giờ trả về giá trị 1 từ A, giá trị tốt nhất cho A là  $\max(3, 1) = 3$ .
  - Giá trị tối ưu cho Maximizer là 3.



❖ Đánh giá

- Tỉa nhánh **không ảnh hưởng** kết quả cuối cùng.
- Thứ tự các nước đi tốt có thể cải thiện hiệu quả tỉa nhánh.
- Với “thứ tự hoàn hảo”, độ phức tạp thời gian là  $O(b^{m/2}) \rightarrow$  cho phép tìm với độ sâu gấp đôi.

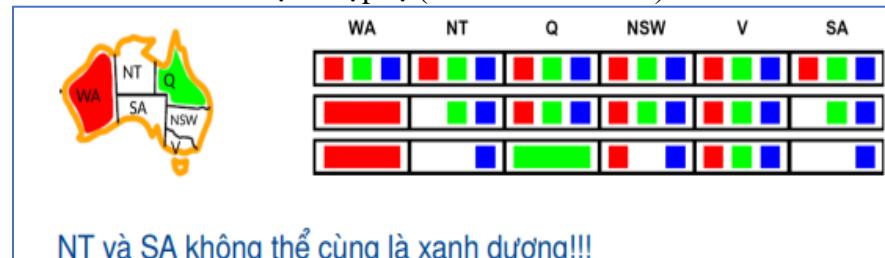
## 4. Bài toán thỏa mãn ràng buộc (Constraint Satisfaction Problem – CSP)

### \*1 số pp heuristic giải btoan CSP

- MRV (minimum remaining value): chọn **biến** có tập giá trị nhỏ nhất → tạo lỗi sớm để loại lỗi sớm (tỉa nhánh)
- DH (degree heuristic): chọn **biến** có nhiều ràng buộc nhất vs các biến còn lại để gán gtri → giảm SL nhánh con
- LCV (Least-constraining value): chọn **giá trị** có ảnh hưởng tối thiểu đến các giá trị khác. Lưu ý là chọn ‘giá trị’ chứ ko phải chọn ‘biến’ như 2 cách trên. Trong VD này, ‘biến’ là tên các lãnh thổ, ‘giá trị’ là các màu
- GH (greedy heuristic): mỗi lần chọn 1 cái tốt nhất
- MRV thấp nhất và DH cao nhất, MRV ưu tiên hơn DH.

### \*1 số thuật toán có sdung 1 trong 3 heuristic trên để giải CSP

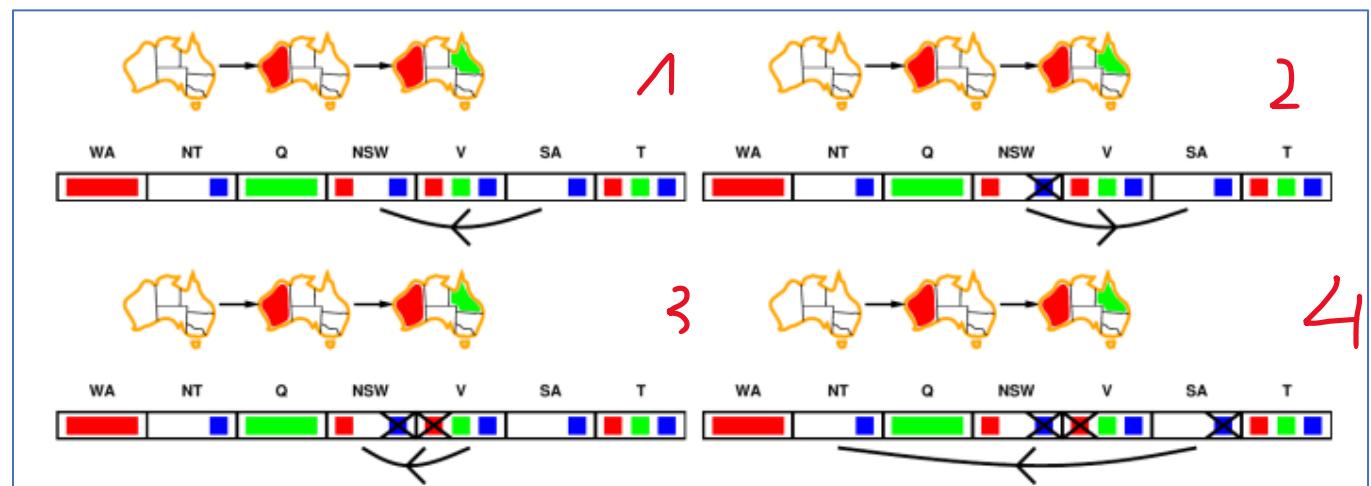
- Kiểm tra tiền + cạnh hợp lệ (chưa xài heuristic)



- i=0: trạng thái đầu, tất cả biến X (lãnh thổ) đều có khả năng nhận 3 giá trị (3 màu) R,G,B
- i=1: chọn đai WA là đỏ, khi đó các biến (lãnh thổ lân cận) WA phải loại đỏ
- i=2: chọn đai Q, làm tương tự i=1. Thấy ngay lỗi → khi đó cần lan truyền ràng buộc bằng pp **cạnh hợp lệ**

Một cạnh  $X \rightarrow Y$  là hợp lệ (arc-consistency) khi  $\forall x \in D_X, \exists y \in D_Y$  sao cho  $x$  và  $y$  không vi phạm ràng buộc. Nếu  $\exists y \in D_Y$  vi phạm ràng buộc thì cần loại gtri nào đó của  $y$ .

**Note:** chỉ check cạnh hợp lệ khi tồn tại 1 biến với đúng 1 gtri còn sót lại (VD: SA)



VD:  $x = \text{SA}$  thuộc  $\{\text{Blue}\}$  có  $y = \text{NSW}$  lân cận  $x = \text{SA}$  cũng trùng màu Blue với  $x \rightarrow$  vi phạm ràng buộc nên ta loại bỏ gtri Blue của NSW. Mà khi NSW thay đổi thì biến (lãnh thổ) lân cận nó cũng cần xét lại, tức xét V kè NSW...(làm tương tự).

#### 4.1. Thuật toán Backtracking

##### ❖ Ý tưởng

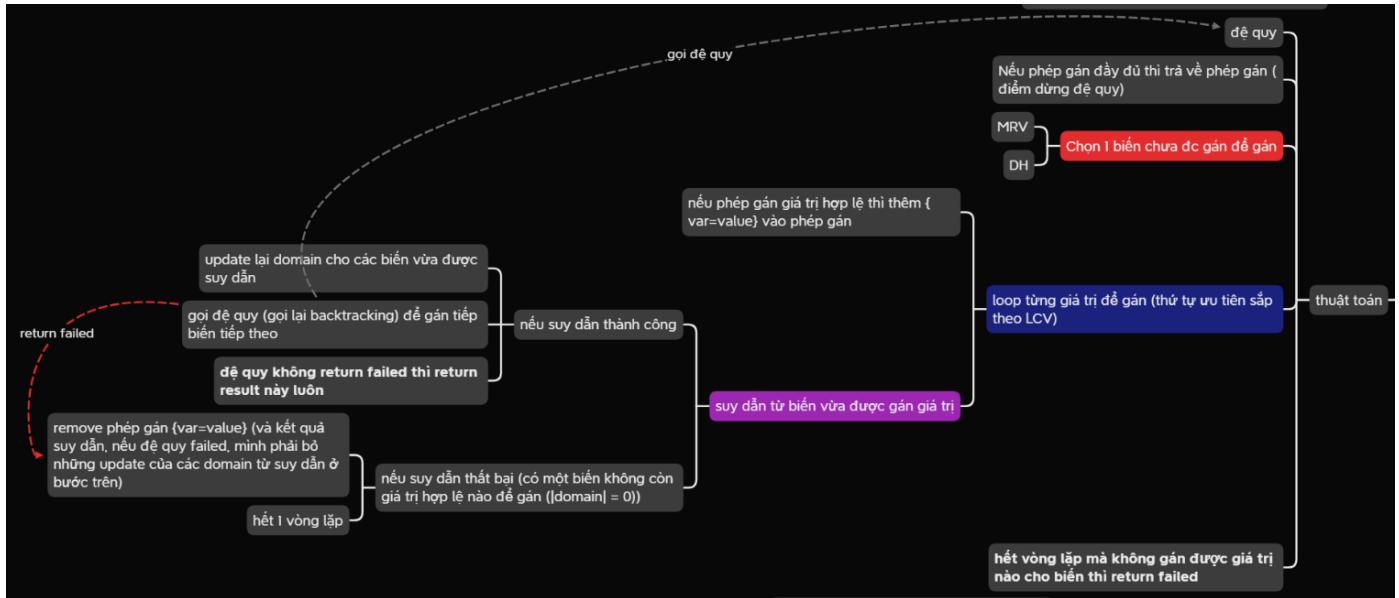
- Gán giá trị lần lượt cho các biến. Việc này chỉ được làm khi đã hoàn thành việc gán giá trị của các biến khác
- Sau mỗi phép gán giá trị cho một biến nào đó, kiểm tra các ràng buộc có được thỏa mãn bởi tất cả các biến đã được gán giá trị cho đến thời điểm hiện tại hay không
- Quay lui (backtracking) nếu có lỗi (không thỏa mãn các ràng buộc)

##### ❖ Giải thuật

```

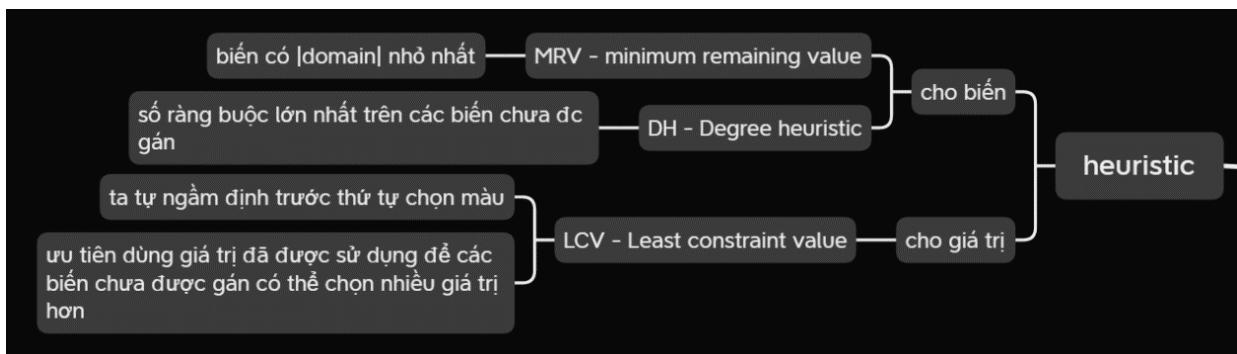
function BACKTRACKING-SEARCH(csp) returns a solution, or failure
  return BACKTRACK({ }, csp)

function BACKTRACK(assignment, csp) returns a solution, or failure
  if assignment is complete then return assignment
  var  $\leftarrow$  SELECT-UNASSIGNED-VARIABLE(csp)
  for each value in ORDER-DOMAIN-VALUES(var, assignment, csp) do
    if value is consistent with assignment then
      add {var = value} to assignment
      inferences  $\leftarrow$  INFERENCE(csp, var, value)
      if inferences  $\neq$  failure then
        add inferences to assignment
        result  $\leftarrow$  BACKTRACK(assignment, csp)
        if result  $\neq$  failure then
          return result
        remove {var = value} and inferences from assignment
      return failure
    
```



❖ Có 3 điểm cần lưu ý trong thuật toán này

- **Which variable should be assigned next (thứ tự xét các biến để gán giá trị)**
  - Dùng 2 heuristic MRV hoặc DH
  - Ưu tiên biến bị ràng buộc nhiều nhất → biến có tập giá trị hợp lệ (domain) nhỏ nhất – Minimum Remaining Values (MRV)
  - Trong trường hợp có nhiều hơn 1 biến có cùng số lượng giá trị hợp lệ, ta chọn biến ràng buộc (không ché) các biến khác (chưa được gán giá trị) nhiều nhất – Degree Heuristic (DH)
- **In what order should its values be tried (đối với một biến, thứ tự các giá trị được xét để gán)**
  - Dùng heuristic LCV – Least Constraint Value
  - Chọn giá trị ràng buộc (không ché) các biến khác (chưa được gán giá trị) ít nhất
- **What inferences should be performed**
  - AC-3 và Forward Checking: 2 thuật toán này để **lan truyền ràng buộc**, giúp **giảm domain** của từng biến chưa được gán bằng cách loại bỏ những giá trị không còn hợp lệ nữa



❖ Các vấn đề

- Lặp đi lặp lại lỗi
  - Nguyên nhân: bỏ qua, không khai thác lý do của mâu thuẫn
  - Giải pháp: phương pháp Backjumping (chuyển đến xét chồ sinh ra lỗi)
- Các thao tác kiểm tra không cần thiết
  - Lặp lại các kiểm tra ràng buộc không cần thiết
  - Giải pháp: phương pháp Backtracking (ghi nhớ các phép gán tốt/không tốt)
- Phát hiện muộn các mâu thuẫn
  - Các vi phạm chỉ được phát hiện sau khi giá trị được gán
  - Giải pháp: phương pháp Forward Checking (kiểm tra trước các ràng buộc)

#### 4.2. Thuật toán AC-3

- ❖ Dùng cho Arc Consistency (phù hợp cạnh)  
 ❖ Giải thuật

**function** AC-3(*csp*) **returns** false if an inconsistency is found  
 and true otherwise

**inputs:** *csp*, a binary CSP with components (*X*, *D*, *C*)

**local variables:** *queue*, a queue of arcs, initially all the arcs in *csp*

**while** *queue* is not empty **do**

  (*X<sub>i</sub>*, *X<sub>j</sub>*)  $\leftarrow$  REMOVE-FIRST(*queue*)

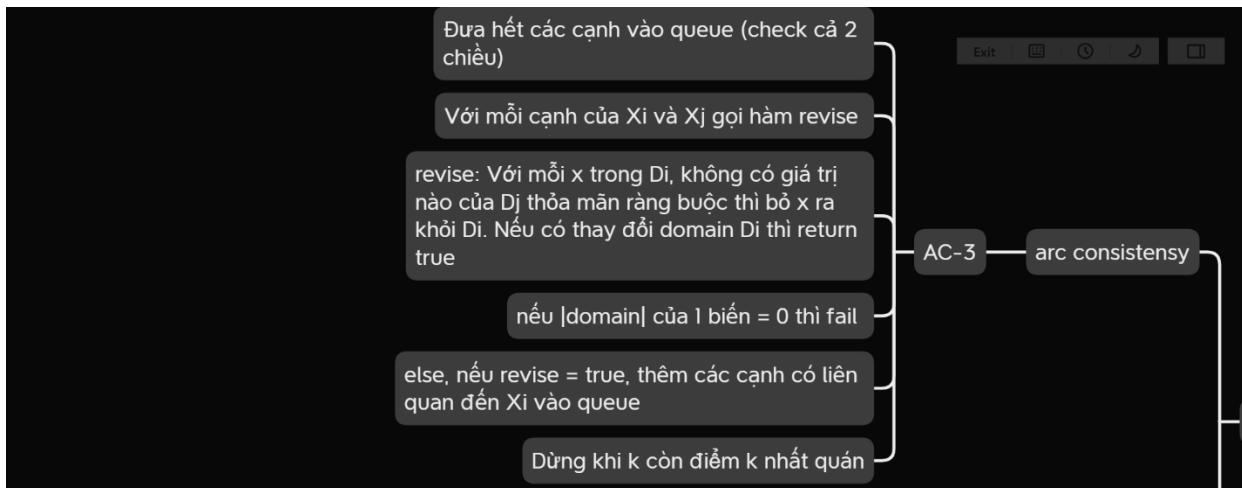
**if** REVISE(*csp*, *X<sub>i</sub>*, *X<sub>j</sub>*) **then**

**if** size of *D<sub>i</sub>* = 0 **then return** false

**for each** *X<sub>k</sub>* **in** *X<sub>i</sub>*.NEIGHBORS - {*X<sub>j</sub>*} **do**

      add (*X<sub>k</sub>*, *X<sub>i</sub>*) to *queue*

**return** true



### Thuật toán Forward Checking

- ❖ Mục đích: tránh thất bại bằng cách kiểm tra trước các ràng buộc
- ❖ Đảm bảo sự phù hợp (consistency) giữa biến đang xét với các biến có liên quan với nó
- ❖ Ý tưởng
  - Ở mỗi bước gán giá trị, theo dõi các giá trị hợp lệ (có thể được gán) đối với các biến chưa được gán giá trị
  - Loại bỏ (dừng) hướng tìm kiếm hiện tại khi có một biến (chưa được gán giá trị) nào đó **không còn giá trị hợp lệ**



## 5. Tri thức

### 5.1. Logic mệnh đề

#### 5.1.1. Giới thiệu khái quát

❖ Công thức

$$1^0 A \rightarrow B = \overline{A} \vee B \text{ (Thay kéo theo bằng phủ định và tuyễn)}$$

$$2^0 A \wedge (B \vee C) = A \wedge B \vee A \wedge C \text{ (Phân phối của tuyễn đối với hội)}$$

$$3^0 A \vee (B \wedge C) = (A \vee B) \wedge (A \vee C) \text{ (Phân phối của tuyễn đối với hội)}$$

$$4^0 \overline{A \vee B} = \overline{A} \wedge \overline{B} \text{ (Luật DeMorgan)}$$

$$5^0 \overline{A \wedge B} = \overline{A} \vee \overline{B} \text{ (Luật DeMorgan)}$$

$$6^0 A \leftrightarrow B = A \wedge B \vee \overline{A} \wedge \overline{B} \text{ (Thay phép tương đương)}$$

$$7^0 A \wedge (A \vee B) = A \text{ (Luật hấp thụ của hội đối với tuyễn)}$$

$$8^0 A \vee A \wedge B = A \text{ (Luật hấp thụ của tuyễn đối với hội)}$$

$$9^0 A \wedge B \vee \overline{B} = A \vee \overline{B} \text{ (Luật hấp thụ)}$$

$$10^0 (A \vee B) \wedge \overline{B} = A \vee \overline{B} \text{ (Luật hấp thụ)}$$

$$11^0 A \wedge B = B \wedge A \text{ (Tính giao hoán của Hội)}$$

$$12^0 A \vee B = B \vee A \text{ (Tính giao hoán của Tuyễn)}$$

$$13^0 (A \wedge B) \wedge C = A \wedge (B \wedge C) \text{ (Tính chất kết hợp của Hội)}$$

$$14^0 (A \vee B) \vee C = A \vee (B \vee C) \text{ (Tính chất kết hợp của Tuyễn)}$$

$$15^0 A \cdot A = A \text{ (Tích lũy đằng của Hội)}$$

$$16^0 A \vee A = A \text{ (Tích lũy đằng của Tuyễn)}$$

$$17^0 A \wedge \overline{A} = 0 \text{ (A và không A luôn luôn sai)}$$

$$17^0 A \vee \overline{A} = 1 \text{ (A và không A luôn luôn đúng)}$$

$$19^0 A \wedge 0 = 0 \text{ (A và hằng sai luôn luôn sai)}$$

$$20^0 A \vee 0 = 0 \text{ (A hay hằng sai luôn là A)}$$

$$21^0 A \wedge 1 = A \text{ (A và hằng đúng luôn là A)}$$

$$22^0 A \vee 1 = 1 \text{ (A hay hằng đúng luôn hằng đúng)}$$

$$23^0 \overline{\overline{A}} = A \text{ (Hai lần phủ định của mệnh đề A lại chính là A)}$$

❖ Độ ưu tiên toán tử

- $\neg, \wedge, \vee, \Rightarrow, \Leftrightarrow$

- Ví dụ:

- $A \vee B \wedge C \equiv A \vee (B \wedge C)$
- $A \wedge B \Rightarrow C \vee D \equiv (A \wedge B) \Rightarrow (C \vee D)$
- $A \Rightarrow B \vee C \Leftrightarrow D \equiv (A \Rightarrow (B \vee C)) \Leftrightarrow D$

❖ Quy tắc biểu diễn  $\Rightarrow, \Leftrightarrow$

- $A \Rightarrow B \equiv \neg A \vee B$
- $A \Leftrightarrow B \equiv (A \Rightarrow B) \wedge (B \Rightarrow A)$

- ❖ Thuật ngữ
  - Một câu là **hợp lệ** nếu và chỉ nếu chân trị của nó là **true** trong **tất cả** thể hiện.
  - Một câu là **thoả mãn được** nếu và chỉ nếu chân trị của nó là **true** trong **ít nhất một** thể hiện.
  - Một câu là **không thoả mãn được** nếu và chỉ nếu chân trị của nó là **false** trong **tất cả** thể hiện.

### 5.1.2. Suy diễn tự nhiên

- ❖ Một số luật suy diễn tự nhiên

- Modus Ponens

$$\frac{\alpha \Rightarrow \beta}{\frac{\alpha}{\beta}}$$

- Modus Tolens

$$\frac{\alpha \Rightarrow \beta}{\frac{\neg \beta}{\neg \alpha}}$$

- And – Introduction

$$\frac{\alpha}{\frac{\beta}{\alpha \wedge \beta}}$$

- And – Elimination

$$\frac{\alpha \wedge \beta}{\frac{\alpha}{\alpha}}, \quad \frac{\alpha \wedge \beta}{\beta}$$

- ❖ Hệ thống suy diễn tự nhiên

- Thường là các “chương trình kiểm tra chứng minh”, đúng nhưng không đủ.
- Dùng nhiều luật suy diễn gây hệ số phân nhánh lớn trong việc tìm một chứng minh.

### 5.1.3. Hợp giải mệnh đề

- ❖ Là phương pháp suy diễn dựa trên luật hợp giải

$$\frac{\alpha \vee \beta}{\frac{\neg \beta \vee \gamma}{\alpha \vee \gamma}}$$

- ❖ Đòi hỏi tất cả các câu được chuyển sang **dạng hội chuẩn CNF**.

- ❖ Dạng hội chuẩn CNF

- Một câu logic được biểu diễn là **nối liền các phép hội** của từ được gọi là dạng hội chuẩn (Conjunctive Normal Form) hay CNF.
- Ví dụ:

$$(A \vee B \vee \neg C) \wedge (B \vee D) \wedge (\neg A) \wedge (B \vee C)$$

- Thủ tục biến đổi gồm các bước sau:
  - Loại bỏ các dấu  $\Rightarrow$ ,  $\Leftrightarrow$ ,  $\Leftarrow$  bằng định nghĩa
  - Đưa dấu phủ định vào dùng luật De Morgan
 
$$\neg(A \vee B) \equiv \neg A \wedge \neg B$$

$$\neg(A \wedge B) \equiv \neg A \vee \neg B$$
  - Phân phối  $\vee$  vào  $\wedge$ 

$$A \vee (B \wedge C) \equiv (A \vee B) \wedge (A \vee C)$$

#### 5.1.4. Thuật giải Robinson

- Ý tưởng
  - Muốn chứng minh  $KB \Rightarrow \alpha$  đúng, ta chứng minh điều ngược lại  $KB \wedge \neg\alpha$  sai.
- Thuật giải
  - Bước 1: Biến đổi tất cả các câu thành dạng CNF.
  - Bước 2: Lấy phủ định kết luận, đưa vào KB.
  - Bước 3: Lặp
    - Nếu trong KB có chứa **hai mệnh đề phủ định nhau** ( $p$  và  $\neg p$ ) thì trả về **false**.
    - Nếu có hai mệnh đề chứa **các literal phủ định nhau** thì áp dụng **hợp giải**. Mỗi lần chỉ hợp giải **một cặp literal**
      - Lặp cho đến khi không thể áp dụng tiếp luật hợp giải.
  - Bước 4: Trả về **true**.
- Ví dụ: chứng minh R

1	$P \vee Q$
2	$P \Rightarrow R$
3	$Q \Rightarrow R$

Bước	Công thức	Suy dẫn
1	$P \vee Q$	Cho trước
2	$\neg P \vee R$	Cho trước
3	$\neg Q \vee R$	Cho trước
4	$\neg R$	Phủ định kết luận
5	$Q \vee R$	1, 2
6	$\neg P$	2, 4
7	$\neg Q$	3, 4
8	$R$	5, 7
9	■	4, 8

- Một cách phát biểu khác của thuật giải
  - Bước 1: Phát biểu lại giả thiết và kết luận của vấn đề dưới dạng CNF.

- Bước 2: Biến đổi dòng trên về thành danh sách mệnh đề:  $\{GT_i, \neg KL_i\}$
  - Bước 3: Nếu trong danh sách có 2 mệnh đề đối ngẫu nhau thì vấn đề được giải quyết xong, còn không thì chuyển sang bước 4.
  - Bước 4: Xây dựng mệnh đề mới bằng cách tuyển ( $\vee$ ) một cặp mệnh đề từ danh sách mệnh đề ở Bước 2. Nếu mệnh đề mới có **các biến mệnh đề đối ngẫu nhau** thì các biến đó được loại bỏ.
  - Bước 5: Bổ sung mệnh đề mới vào danh sách mệnh đề và loại bỏ 2 mệnh đề được tuyển thành mệnh đề mới.
  - Bước 6: Nếu không xây dựng được thêm mệnh đề mới nào và trong danh sách không có 2 mệnh đề nào đối ngẫu thì vấn đề phát biểu ở Bước 1 sai.
- ❖ Ví dụ: cho 1 vấn đề được phát biểu như sau:

$$\neg p \vee q, \neg q \vee r, \neg r \vee s, \neg u \vee \neg s \Rightarrow \neg p, \neg u$$

Viết lại thành danh sách mệnh đề

$$\{\neg p \vee q, \neg q \vee r, \neg r \vee s, \neg u \vee \neg s, p, u\}$$

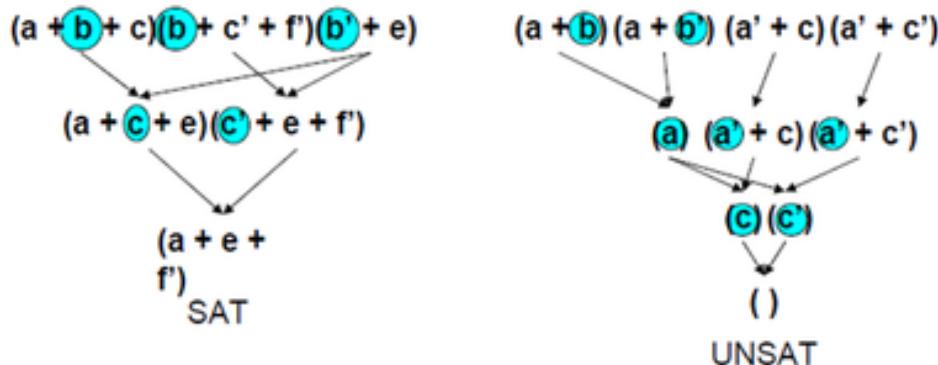
$$\{\neg p \vee r, \neg r \vee s, \neg u \vee \neg s, p, u\}$$

$$\{\neg p \vee s, \neg u \vee \neg s, p, u\}$$

$$\{\neg p \vee \neg u, p, u\}$$

$$\{\neg u, u\}$$

### 5.1.5. Thủ tục Davis-Putnam



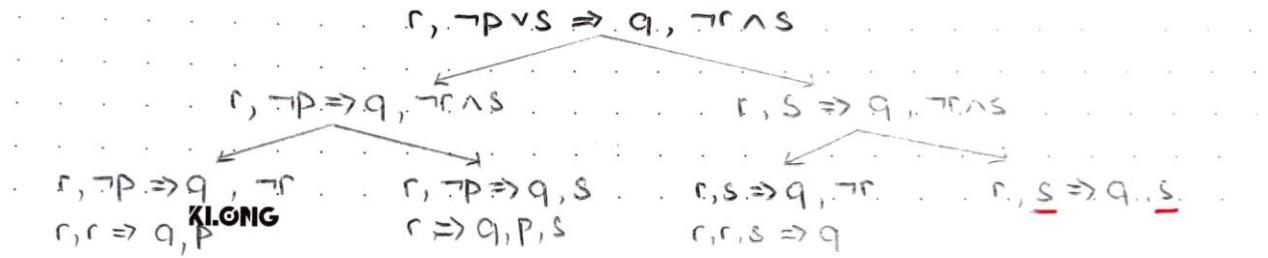
**function dp(KB)**{với mọi  $\phi$  trong các literal của KB    **do** {var KB'  $\leftarrow \{\}$ ;        **với mọi**  $\Phi_1$  trong KB **với mọi**  $\Phi_2$  trong KB            sao cho  $\phi \in \Phi_1, \neg\phi \in \Phi_2$             **do** {var  $\Phi' \leftarrow \Phi_1 - \{\phi\} \cup \Phi_2 - \{\neg\phi\}$ ;                **if** not tautology( $\Phi'$ ) **then** KB'  $\leftarrow$  KB'  $\cup \{\Phi'\}$ };            KB  $\leftarrow$  KB - { $\Phi \in$  KB |  $\phi \in \Phi$  hay  $\neg\phi \in \Phi$ }  $\cup$  KB'}};    **if** False  $\in$  KB **then** return false; **else** return true;}**function tautology( $\Phi$ )**{**if** tồn tại  $\phi: \phi \in \Phi$  và  $\neg\phi \in \Phi$  **then** return true;    **else** return false}

## 5.1.6. Thuật giải Vương Hạo (Havard)

## ❖ Thuật giải

- Bước 1: Phát biểu lại giả thiết và kết luận của vấn đề dưới dạng chuẩn CNF.
- Bước 2: Chuyển về các GT<sub>i</sub>, KL<sub>i</sub> có dạng phủ định
- Bước 3: Thay dấu  $\wedge$  trong GT<sub>i</sub> và dấu  $\vee$  trong KL<sub>j</sub> bằng dấu ", "
- Bước 4: Nếu ở GT<sub>i</sub> có dấu  $\vee$  hoặc KL<sub>j</sub> có dấu  $\wedge$  thì dòng hiện tại được tách thành 2 dòng con.
- Bước 5: Một dòng được chứng minh nếu tồn tại chung 1 mệnh đề ở cả 2 vế.
- Bước 6:
  - Nếu 1 dòng không còn dấu liên kết { $\wedge$ ,  $\vee$ } và ở cả 2 vế đều không có chung 1 biến mệnh đề nào thì dòng đó không được chứng minh.
  - Một vấn đề được giải quyết một cách trọn vẹn nếu mọi dòng dẫn xuất từ biểu diễn ở dạng chuẩn được chứng minh.

## ❖ Ví dụ: cho vấn đề được biểu diễn



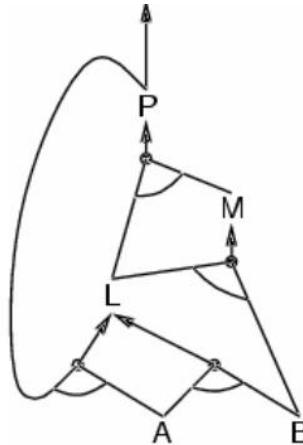
### 5.1.7. Suy diễn tiến, suy diễn lùi

#### a. Suy diễn tiến

##### ❖ Ý tưởng

- Áp dụng các luật có vé trái nằm trong cơ sở tri thức (KB)
- Bổ sung vé phải của các luật áp dụng vào KB đến khi tìm thấy kết luận

$$\begin{aligned}
 P &\Rightarrow Q \\
 L \wedge M &\Rightarrow P \\
 B \wedge L &\Rightarrow M \\
 A \wedge P &\Rightarrow L \\
 A \wedge B &\Rightarrow L \\
 A \\
 B
 \end{aligned}$$



#### b. Suy diễn lùi

##### ❖ Ý tưởng: suy diễn lùi từ kết luận KL

- Kiểm tra xem KL đã được biết chưa, nếu chưa
- Chứng minh bằng quay lui sử dụng các luật dẫn đến q
- Tránh lặp vô tận
  - Lưu trữ các đích đã được chứng minh
  - Trước khi chứng minh kiểm tra xem đích cần chứng minh đã có trong goal stack chưa?
- Tránh lặp lại công việc: kiểm tra xem KL mới
  - Đã ở trong tập đã được chứng minh chưa
  - Đã làm nhưng thất bại chưa
- Đi sâu vào thuật toán
  - Đầu:
    - Goal = tập các sự kiện cần chứng minh = KL

- Goal = {f| f cần CM cho đến thời điểm hiện tại}
- Vet = {(f,j)| để CM f thì dùng luật j: leftj → f}
- Cờ Back = true khi quay lui false không quay lui
- c. So sánh suy diễn tiến và suy diễn lùi
  - ❖ SD tiến hướng dữ liệu, tự động, không định hướng. Ví dụ, nhận dạng đối tượng, xác định hành trình
  - ❖ Có thể làm rất nhiều việc không liên quan đến KL
  - ❖ SD lùi hướng KL, thích hợp cho các bài toán giải quyết vấn đề. Ví dụ, tìm chìa khoá, lập kế hoạch thi TOEFL
  - ❖ Độ phức tạp của SD lùi thường nhỏ hơn rất nhiều so với kích thước của CSTT

## 5.2. Logic bậc nhất (logic vị từ)

### 5.2.1. Giới thiệu khái quát

- ❖ Cú pháp
  - Biểu thức (term)
    - Ký hiệu hằng: Lan, Tuấn, KHTN, ...
    - Biến: x, y, a, ...
    - Ký hiệu hàm (function) áp dụng cho **1 hay nhiều biểu thức** và trả về 1 đối tượng: f(x), tuoi(Lan), anh – cua(Tuan), ...
  - Câu (sentence)
    - Ký hiệu vị từ (predicate) áp dụng cho **không hay nhiều biểu thức** và trả về chân trị true/false: Brother(Tuan, Lan), Friend(Brother(Tuan), Lan), ...
    - Sử dụng toán tử nối câu  $\neg$ ,  $\wedge$ ,  $\vee$ ,  $\Rightarrow$ ,  $\Leftrightarrow$ ,  $\Leftarrow$  để tạo câu phức.
- ❖ Ngữ nghĩa
  - Mô hình chứa các đối tượng (các thành phần) và quan hệ giữa chúng
  - Một thê hiện xác định các tham chiếu cho
    - Các ký hiệu hằng → các đối tượng
    - Các ký hiệu vị từ → các quan hệ
    - Các ký hiệu hàm → các quan hệ hàm
  - Một câu nguyên tố predicate(term<sub>1</sub>, term<sub>2</sub>, ..., term<sub>n</sub>) là đúng nếu và chỉ nếu các đối tượng được tham chiếu bởi term<sub>1</sub>, term<sub>2</sub>, ..., term<sub>n</sub> nằm trong quan hệ được chỉ định bởi predicate.
- ❖ Lượng từ Với mọi
  - $\forall$  < biến >. < câu >
  - $\forall x$ . P đúng trong một mô hình m nếu và chỉ nếu P đúng với x trong mọi đối tượng có thê của mô hình.
  - Lỗi cần tránh:
    - Thông thường,  $\Rightarrow$  là phép nối thường đi với  $\forall$
    - Lỗi thường gặp d2ùng  $\wedge$  làm phép nối chính đi với lượng từ  $\forall$ .
- ❖ Lượng từ Tồn tại
  - $\exists$  < biến >. < câu >

- $\exists x. P$  đúng trong một mô hình m nếu và chỉ nếu  $P$  đúng với  $x$  trong một đối tượng có thể nào đó của mô hình.
- Lỗi cần tránh:
  - Thông thường,  $\wedge$  là phép nối chính với  $\exists$ .
  - Lỗi thường gặp dùng  $\Rightarrow$  làm phép nối chính với  $\exists$ .

### 5.2.2. Hợp giải bậc nhất

#### 5.2.2.1. Biến đổi dạng mệnh đề

❖ **Bước 1: Loại bỏ các dấu mũi tên**

$$\begin{aligned}\alpha \Leftrightarrow \beta &\equiv (\alpha \Rightarrow \beta) \wedge (\beta \Rightarrow \alpha) \\ \alpha \Rightarrow \beta &\equiv \neg\alpha \vee \beta\end{aligned}$$

❖ **Bước 2: Phân phối phủ định**

$$\neg\neg\alpha \equiv \alpha$$

$$\neg(\alpha \vee \beta) \equiv \neg\alpha \wedge \neg\beta$$

$$\neg(\alpha \wedge \beta) \equiv \neg\alpha \vee \neg\beta$$

$$\neg\forall x. \alpha \equiv \exists x. \neg\alpha$$

$$\neg\exists x. \alpha \equiv \forall x. \neg\alpha$$

❖ **Bước 3: Đổi tên các biến thành phần**

$$\forall x \exists y. (\neg P(x) \vee \exists x. Q(x, y)) \equiv \forall x_1. \exists y_2. (\neg P(x_1) \vee \exists x_3. Q(x_3, y_2))$$

❖ **Bước 4: Skolem hoá**

- Thay tên mới cho tất cả lượng tử tồn tại
- Thay hàm mới cho tất cả các lượng tử tồn tại ở tầm vực với mọi

- Thay tên mới cho tất cả lượng tử tồn tại

$$\exists x. P(x) \Rightarrow P(\text{Lan})$$

$$\exists x, y. R(x, y) \Rightarrow R(\text{Thing 1}, \text{Thing 2})$$

$$\exists x. P(x) \wedge Q(x) \Rightarrow P(\text{Fleep}) \wedge Q(\text{Fleep})$$

$$\exists y, \forall x. \text{Loves}(x, y) \Rightarrow \forall x. \text{Loves}(x, \text{Englebert})$$

- Thay hàm mới cho tất cả các lượng tử tồn tại ở tầm vực với mọi

$$\forall x \exists y. \text{Loves}(x, y) \Rightarrow \forall x. \text{Loves}(x, \text{beloved}(x))$$

❖ **Bước 5: Bỏ các lượng tử với mọi**

$$\forall x. \exists y. \text{Loves}(x, y) \equiv \text{Loves}(x, \text{beloved}(x))$$

❖ **Bước 6: Phân phối  $\vee$  vào  $\wedge$ , trả về các mệnh đề**

$$P(z) \vee (Q(z, w) \wedge R(w, z)) \equiv \{P(z) \vee Q(z, w), P(z) \vee R(w, z)\}$$

❖ **Bước 7: Đổi tên các biến trong từng mệnh đề**

$$\{P(z) \vee Q(z, w), P(z) \vee R(w, z)\} \equiv \{P(z_1) \vee Q(z_1, w_1), P(z_2) \vee Q(w_2, z_2)\}$$

### 5.2.2.2. Phép thế



## PHÉP THẾ

- $P(x, f(y), B)$ : một câu nguyên tố

Các thể hiện	Phép thế $\{v_1/t_1, v_2/t_2 \dots\}$	Ghi chú
$P(z, f(w), B)$	$\{x/z, y/w\}$	Đổi tên biến
$P(x, f(A), B)$	$\{y/A\}$	
$P(g(z), f(A), B)$	$\{x/g(z), y/A\}$	
$P(C, f(A), B)$	$\{x/C, y/A\}$	Phép thế cơ sở

- Áp dụng một phép thế

$$P(x, f(y), B) \{y/A\} = P(x, f(A), B)$$

$$P(x, f(y), B) \{y/A, x/y\} = P(A, f(A), B)$$

### 5.2.2.3. Phép đồng nhất

- ❖ Hai biểu thức  $w_1$  và  $w_2$  là **đồng nhất** (unifiable) khi và chỉ khi tồn tại thế s sao cho:

$$w_1s = w_2s$$

- ❖ Gọi  $w_1 = x$  và  $w_2 = y$ , đây là các **phép đồng nhất**

s	$\omega_1 s$	$\omega_2 s$
$\{y/x\}$	x	x
$\{x/y\}$	y	y
$\{x/f(f(A)), y/f(f(A))\}$	<u>f(f(A))</u>	<u>f(f(A))</u>
$\{x/A, y/A\}$	A	A

- ❖ Đồng nhất tổng quát nhất



# ĐỒNG NHẤT TỔNG QUÁT NHẤT

- $g$  là phép đồng nhất tổng quát nhất (most general unifier - MGU) của  $\omega_1$  và  $\omega_2$  khi và chỉ khi với mọi phép đồng nhất  $s$ , tồn tại  $s'$  sao cho  $\omega_1.s = (\omega_1.g)s'$

$\omega_1$	$\omega_2$	MGU
$P(x)$	$P(A)$	$\{x/A\}$
$P(f(x), y, g(x))$	$P(f(x), x, g(x))$	$\{y/x\}$ hay $\{x/y\}$
$P(f(x), y, g(y))$	$P(f(x), z, g(x))$	$\{y/x, z/x\}$
$P(x, B, B)$	$P(A, y, z)$	$\{x/A, y/B, z/B\}$
$P(g(f(v)), g(u))$	$P(x, x)$	$\{x/g(f(v)), u/f(v)\}$
$P(x, f(x))$	$P(x, x)$	Không có MGU!

#### 5.2.2.4. Hợp giải Robinson

- Để chứng minh 1 tập KB có suy dẫn logic được 1 câu alpha hay không, viết lại KB  $\wedge \neg\alpha$  dưới dạng mệnh đề và có gắng suy dẫn ra mệnh đề sai (hợp giải 2 mệnh đề đối ngẫu)
- Chứng minh rằng:**  $P(x) \Rightarrow Q(x)$  và  $P(A)$  suy dẫn logic  $\exists z. Q(z)$

1	$\neg P(x) \vee Q(x)$	Tiền đề
2	$P(A)$	Tiền đề
3	$\neg Q(z)$	Phù định kết luận
4	$\neg P(z)$	1, 3 1, 3
5	False	2, 4 $\theta = \{x/z\}$

$$\theta = \{x/z, z/A\}$$

Bước	Công thức	Suy dẫn	Ghi chú
1	$\neg P(x) \vee Q(x)$	Tiền đề	
2	$P(A)$	Tiền đề	
3	$\neg Q(z)$	Phù định kết luận	
4	$\neg P(z)$	1, 3	$\theta = \{x/z\}$
5	False	2, 4	$\theta = \{x/z, z/A\}$

- Cho trước :  $P(x) \Rightarrow Q(x)$  và  $P(A)$  và  $P(B)$ , tìm  $z$  sao cho  $Q(z)$  là đúng

<u>Định nghĩa</u>	
1. $\neg P(x) \vee Q(x)$	Tiền đề
2. $P(A)$	Tiền đề
3. $P(B)$	Tiền đề
4. $\neg Q(z)$	Phủ định kết luận
5. $\neg P(z)$	1, 4
6. False	2, 5
7. False	3, 6
$\Theta = \{x/z\}$	
$\Theta = \{x/z, z/A\}$	
$\Theta = \{x/z, z/B\}$	



## Ví dụ:

**Chứng minh:** An là sinh viên trường ĐH KHTN.  
Biết rằng:

- An là sinh viên lớp TH-K19.
- Lớp TH-K19 là bộ phận của Khoa CNTT
- Khoa CNTT là bộ phận của trường ĐHKHTN
- Sử dụng ví dụ:  $f(x, y) - x$  là bộ phận của  $y$ .
- Ta có:

- $\forall x, y, z: f(x, y) \wedge f(y, z) \rightarrow f(x, z)$
- $\Leftrightarrow \neg(f(x, y) \wedge f(y, z)) \vee f(x, z)$
- $\Leftrightarrow \neg f(x, y) \vee \neg f(y, z) \vee f(x, z)$



+



## Ví dụ

### Biểu diễn giả thiết qua vị từ:

- 1.  $f(An, TH - K19)$
- 2.  $f(TH - K19, K, CNTT)$
- 3.  $f(K, CNTT, ĐHKHTN)$
- 4.  $\neg f(x, y) \vee \neg f(y, z) \vee f(x, z)$
- 5.  $\neg f(An, ĐHKHTN)$
- 6. hg(4,1):  $\neg f(y, z) \vee f(x, z) \rightarrow \neg f(TH - K19, z) \vee f(An, z)$ 
  - $x \leftarrow An$
  - $y \leftarrow TH - K19$
- 7. hg(6,2):  $f(An, z) \rightarrow f(An, K, CNTT)$ 
  - $z \leftarrow K, CNTT$
- 8. hg(7,4):  $\neg f(y, z) \vee f(x, z) \rightarrow \neg f(K, CNTT, z) \vee f(An, z)$ 
  - $x \leftarrow An$
  - $y \leftarrow K, CNTT$
- 9. hg(8,3):  $f(An, z) \rightarrow f(An, ĐHKHTN)$ 
  - $z \leftarrow ĐHKHTN$
- 10. hg(9,5):  $f(An, ĐHKHTN), \neg f(An, ĐHKHTN)$

## 6. Các phương pháp máy học

### 6.1. Học qua quan sát

#### 6.1.1. Thuật toán Quinlan

- ❖ Cho một bảng quan sát là một tập hợp các mẫu với các thuộc tính nhất định của một đối tượng nào đó.
- ❖ Sử dụng một độ đo để định lượng về đề ra một tiêu chuẩn nhằm chọn lựa thuộc tính mang tính chất **phân loại** để phân bảng này thành bảng con nhỏ hơn sao cho từ các bảng con này dễ phân tích tìm ra một quy luật chung (bước đệ quy)
- ❖ Từ đó, thiết lập một **cây quyết định** cho thấy thứ tự của các thuộc tính được xét

#### 6.1.2. Ví dụ

- a. Cho bảng dữ liệu và định nghĩa độ đo

STT	Hình dáng	Chiều cao	Giới tính	KQ
1	To	TB	Nam	Á
2	Nhỏ	Thấp	Nam	Á
3	Nhỏ	TB	Nam	Á
4	To	Cao	Nam	Âu
5	Nhỏ	TB	Nữ	Âu
6	Nhỏ	Cao	Nam	Âu
7	Nhỏ	Cao	Nữ	Âu
8	To	TB	Nữ	Âu

$$V(\text{hình dáng} = \text{To}) = (\hat{A}_{\text{to}}, \hat{A}_{\text{u}_\text{to}})$$

$$\hat{A}_{\text{to}} = \frac{\Sigma \text{quan sát Á có hình dáng to}}{\Sigma \text{quan sát có hình dáng to}}$$

$$\hat{A}_{\text{u}_\text{to}} = \frac{\Sigma \text{quan sát Âu có hình dáng to}}{\Sigma \text{quan sát có hình dáng to}}$$

b. Tính toán

❖ Lần 1

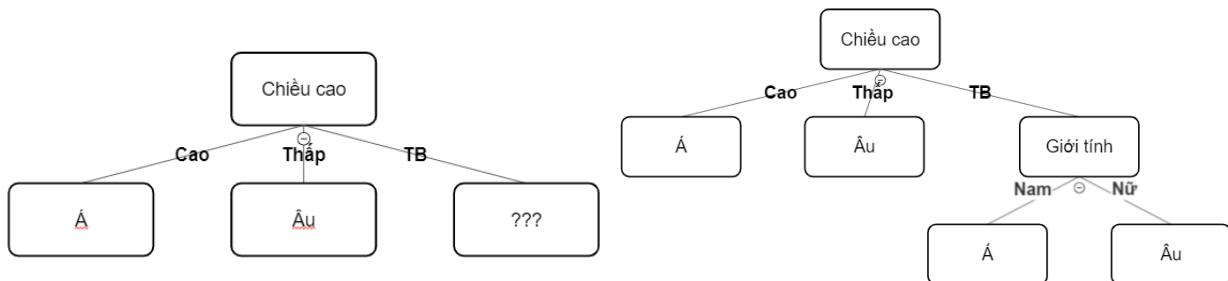
- Hình dáng
  - $V(\text{hình dáng} = \text{To}) = (1/3, 2/3)$
  - $V(\text{hình dáng} = \text{Nhỏ}) = (2/5, 3/5)$
- Chiều cao
  - $V(\text{chiều cao} = \text{cao}) = (0, 1)$
  - $V(\text{chiều cao} = \text{thấp}) = (1, 0)$
  - $V(\text{chiều cao} = \text{TB}) = (1/2, 1/2)$
- Giới tính
  - $V(\text{giới tính} = \text{Nam}) = (3/5, 2/5)$
  - $V(\text{giới tính} = \text{Nữ}) = (0, 1)$
- Nếu không có **vector đơn vị** → độ đo không phù hợp dữ liệu
- Tiêu chuẩn phân loại: chọn thuộc tính có **nhiều vector đơn vị nhất**
- Do chiều cao có nhiều vector đơn vị nhất → chọn. Ta có bảng con

STT	Hình dáng	Chiều cao	Giới tính	KQ
1	To	TB	Nam	Á
3	Nhỏ	TB	Nam	Á
5	Nhỏ	TB	Nữ	Âu
8	To	TB	Nữ	Âu

❖ Lần 2

- Hình dáng
  - $V(\text{hình dáng} = \text{To}) = (1/2, 1/2)$
  - $V(\text{hình dáng} = \text{Nhỏ}) = (1/2, 1/2)$
- Giới tính
  - $V(\text{giới tính} = \text{Nam}) = (1, 0)$
  - $V(\text{giới tính} = \text{Nữ}) = (0, 1)$
- Do giới tính có nhiều vector đơn vị nhất → chọn

c. Cây quyết định



❖ Từ cây quyết định, suy ra

- Nếu chiều cao = cao thì châu Âu
- Nếu chiều cao = tháp thì châu Á
- Nếu chiều cao = TB và giới tính = nam thì châu Á
- Nếu chiều cao = TB và giới tính = nữ thì châu Âu

## 6.2. Học bằng cách xây dựng cây định danh

- ❖ CSDL: dùng thủ tục đâm chồi → cây định danh
- ❖ Cây định danh: dùng thủ tục tia cây → bộ luật

### 6.2.1. Từ CSDL đến cây định danh

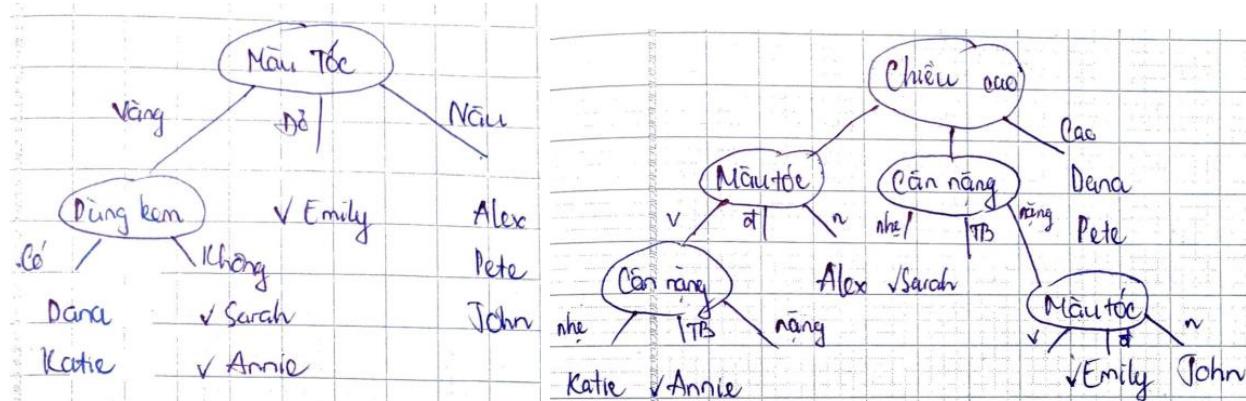
Cho CSDL

Tên	Màu tóc	Chiều cao	Cân nặng	Dùng kem	Kết quả
Sarah	Vàng	TB	Nhẹ	Không	Cháy nắng
Dana	Vàng	Cao	TB	Có	Không
Alex	Nâu	Tháp	TB	Có	Không
Annie	Vàng	Tháp	TB	Không	Cháy nắng
Emily	Đỏ	TB	Nặng	Không	Cháy nắng
Pete	Nâu	Cao	Nặng	Không	Không
John	Nâu	TB	Nặng	Không	Không
Katie	Vàng	Tháp	Nhẹ	Có	Không

a. Định nghĩa cây định danh

- ❖ Là một dạng cây quyết định, trong đó mỗi tập quyết định có thể xảy ra được thiết lập một cách ngầm định bởi danh sách các mẫu mà chúng được phân vào một lớp đã biết.

b. Ví dụ



c. Luật OCCAM

- ❖ Thế giới vốn dĩ là đơn giản. Do đó, cây định danh nhỏ nhất mà nhất quán với các mẫu chính là cây có khả năng nhất trong việc định danh chính xác các đối tượng.

d. Độ hỗn loạn trung bình

$$\text{❖ ĐHL}_{\text{TB}} = \sum_b \frac{n_b}{n_t} \times \left( \sum_b - \frac{n_{bc}}{n_b} \times \log_2 \frac{n_{bc}}{n_b} \right)$$

❖ Có giá trị **từ 0 đến 1**

❖ Trong đó

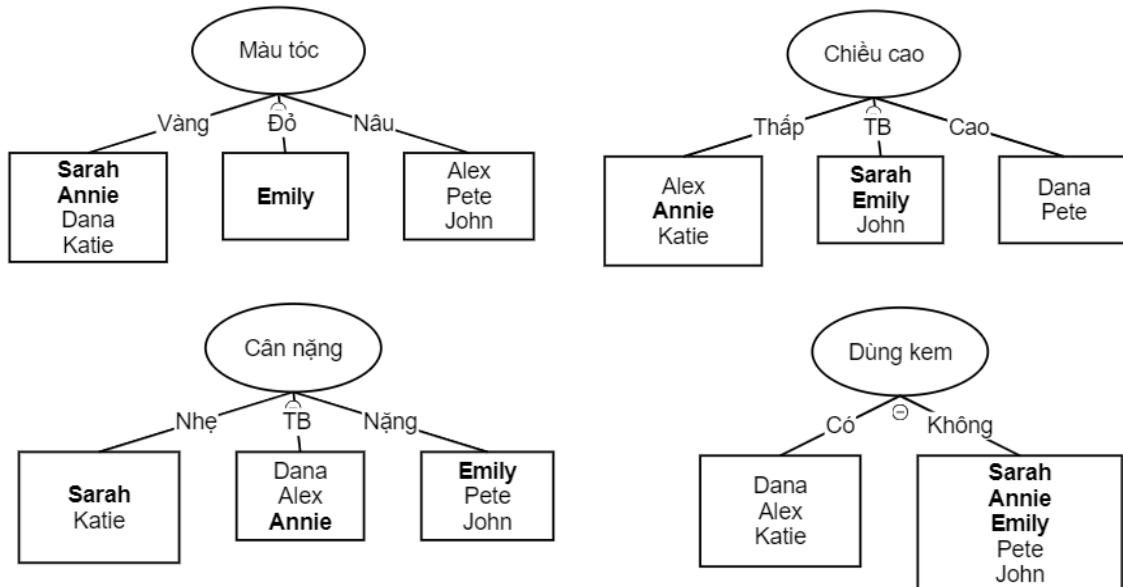
- $n_t$ : tổng số mẫu
- $n_b$ : tổng số mẫu trong nhánh b
- $n_{bc}$ : tổng số mẫu trong nhánh b thuộc lớp c
- b: số nhánh
- c: số lớp

e. Tính toán

❖ Tính lần 1

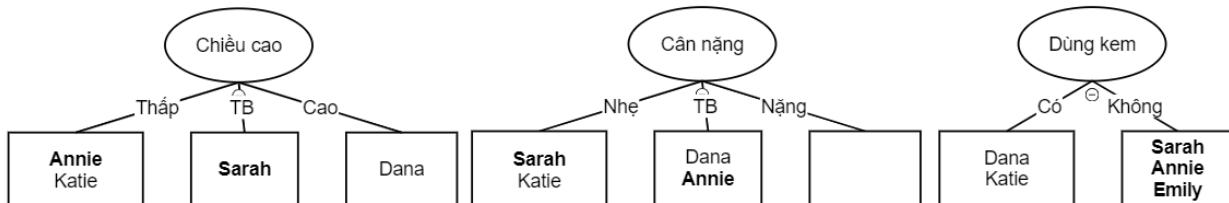
- Độ hỗn loạn: màu tóc (0.5), chiều cao (0.69), cân nặng (0.94), dùng kem (0.61)
- Màu tóc có ĐHL nhỏ nhất → chọn

$$\begin{aligned}
 DHL_{\text{màu tóc}} &= \frac{4}{8} \times \left( \frac{-2}{4} \times \log_2 \frac{2}{4} - \frac{2}{4} \times \log_2 \frac{2}{4} \right) \\
 &\quad + \frac{1}{8} \times \left( \frac{-1}{1} \times \log_2 \frac{1}{1} - \frac{0}{1} \times \log_2 \frac{0}{1} \right) \\
 &\quad + \frac{3}{8} \times \left( \frac{-0}{3} \times \log_2 \frac{0}{3} - \frac{3}{3} \times \log_2 \frac{3}{3} \right) \\
 &= \frac{4}{8} \times \left( -\log_2 \frac{2}{4} \right) = 0,5
 \end{aligned}$$



❖ Tính lần 1

- Độ hỗn loạn: chiều cao (0.5), cân nặng (1), dùng kem (0)
- Dùng kem có DHL nhỏ nhất → chọn



f. Thủ tục đâm chồi (SPROUTER)

- ❖ Lắp cho đến khi mỗi nút lá chỉ gồm các mẫu **đồng nhất**
- Chọn một nút lá có tập các mẫu **không đồng nhất**

- **Thay thế** nút lá đó bằng **nút kiểm tra** mà nó chia tập mẫu không đồng nhất đó thành tập không đồng nhất ở mức độ **tối thiểu** dựa vào độ hỗn loạn nào đó

#### 6.2.2. Từ cây đến luật

- ❖ Ta chỉ việc **theo dấu** mỗi đường dẫn từ nút gốc đến nút lá, lấy **phép thử làm tiền đề** và phân loại **nút lá làm kết luận**. Dựa theo ví dụ trên:
  - Nếu tóc vàng và có dùng kem thì không việc gì xảy ra
  - Nếu tóc vàng và không dùng kem thì bị cháy nắng
  - Nếu tóc đỏ thì bị cháy nắng
  - Nếu tóc nâu thì không việc gì xảy ra
- ❖ Sau khi đã có luật từ cây định danh
  - Loại bỏ các tiền đề không cần thiết, bằng cách xét từng luật và so sánh với CSDL
  - Loại bỏ các luật không cần thiết, dùng từ “hoặc” để nối các luật cùng kết luận
  - Thủ tục tẩy cành (PRUNER)
    - Tạo một luật cho mỗi đường đi từ nút gốc đến nút lá trong cây định danh
    - Đơn giản hóa các luật bằng cách loại bỏ các tiền đề không ảnh hưởng đến kết luận
    - Thay thế những luật có chung kết luận bằng một luật mặc định được kích hoạt khi không có luật nào được kích hoạt

- Ai có dùng kem cũng không bị cháy nắng
  - 1. Nếu tóc vàng, có dùng kem thì không sao
  - 2. Nếu tóc vàng, không dùng kem thì cháy nắng
  - 3. Nếu tóc đỏ thì cháy nắng
  - 4. Nếu tóc nâu thì không sao
- Có 2 luật cho kết luận cháy nắng, 2 luật cho kết luận không sao  
⇒ Có thể bỏ bớt 2 luật và thêm 1 luật mặc định (được áp dụng khi không có luật nào thỏa)
  - 1. Nếu tóc vàng, không dùng kem thì cháy nắng
  - 2. Nếu tóc đỏ thì cháy nắng
  - 3. Nếu không có luật nào thỏa thì không sao
- Tạo 1 luật cho mỗi đường đi từ gốc đến lá trong cây quyết định
- Đơn giản hóa mỗi luật bằng cách loại bỏ những tiền đề không ảnh hưởng KL mà cây có được
- Thay thế những luật có chung KL bằng 1 luật mặc định mà luật này sẽ được kích hoạt khi không có luật nào khác được kích hoạt

## 7. Học máy

### 7.1. Thuật toán Quinlan

Cho bảng mô tả sau:

STT	Tên	Màu tóc	Chiều cao	Cân nặng	Dùng kem	Kết quả
1	Sarah	Vàng	Trung bình	Nhẹ	Không	Cháy nắng
2	Dana	Vàng	Cao	Trung bình	Có	Không cháy nắng
3	Alex	Nâu	Thấp	Trung bình	Có	Không cháy nắng
4	Annie	Vàng	Thấp	Trung bình	Không	Cháy nắng
5	Emily	Đỏ	Trung bình	Nặng	Không	Cháy nắng
6	Peter	Nâu	Cao	Nặng	Không	Không cháy nắng
7	John	Nâu	Trung bình	Nặng	Không	Không cháy nắng
8	Katie	Vàng	Thấp	Nhẹ	Có	Không cháy nắng

Hãy sử dụng thuật toán Quinlan để xác định xem một người có bị cháy nắng hay không?

Xét thuộc tính Màu tóc:

- Vector đặc trưng thuộc tính Màu tóc:

$$V_{\text{Tóc(vàng)}} = (T(\text{vàng}, \text{cháy nắng}), T(\text{vàng}, \text{không cháy nắng}))$$

$$V_{\text{Tóc(nâu)}} = (T(\text{nâu}, \text{cháy nắng}), T(\text{nâu}, \text{không cháy nắng}))$$

$$V_{\text{Tóc(đỏ)}} = (T(\text{đỏ}, \text{cháy nắng}), T(\text{đỏ}, \text{không cháy nắng}))$$

Do đó:

- $V_{\text{Tóc(vàng)}} = \left(\frac{2}{4}, \frac{2}{4}\right) = (0.5, 0.5)$
- $V_{\text{Tóc(nâu)}} = \left(\frac{0}{3}, \frac{3}{3}\right) = (0, 1)$
- $V_{\text{Tóc(đỏ)}} = \left(\frac{1}{1}, \frac{0}{1}\right) = (1, 0)$

Tổng số vector đơn vị của thuộc tính Màu tóc là 2.

Xét thuộc tính Chiều cao:

- $V_{\text{CCao(Cao)}} = \left(\frac{0}{2}, \frac{2}{2}\right) = (0, 1)$
- $V_{\text{CCao(TB)}} = \left(\frac{2}{3}, \frac{1}{3}\right)$
- $V_{\text{CCao(Thấp)}} = \left(\frac{1}{3}, \frac{2}{3}\right)$

Tổng số vector đơn vị của thuộc tính Chiều cao là 1.

Xét thuộc tính Cân nặng:

- $V_{CN\ddot{a}ng(Nh\dot{e})} = \left(\frac{1}{2}, \frac{1}{2}\right)$
- $V_{CN\ddot{a}ng(TB)} = \left(\frac{1}{3}, \frac{2}{3}\right)$
- $V_{CN\ddot{a}ng(N\ddot{a}ng)} = \left(\frac{1}{3}, \frac{2}{3}\right)$

Tổng số vector đơn vị của thuộc tính Cân nặng là 0.

Xét thuộc tính Dùng kem:

- $V_{DKem(Có)} = \left(\frac{3}{3}, \frac{0}{3}\right) = (1, 0)$
- $V_{DKem(Kh\dot{o}ng)} = \left(\frac{3}{5}, \frac{2}{5}\right)$

Tổng số vector đơn vị của thuộc tính Dùng kem là 1.

Như vậy thuộc tính Màu tóc có số vector đơn vị nhiều nhất nên sẽ được chọn để phân hoạch. Sau khi phân hoạch ta thu được bảng rút gọn sau:

STT	Tên	Chiều cao	Cân nặng	Dùng kem	Kết quả
1	Sarah	Trung bình	Nhẹ	Không	Cháy nắng
2	Dana	Cao	Trung bình	Có	Không cháy nắng
4	Annie	Thấp	Trung bình	Không	Cháy nắng
8	Katie	Thấp	Nhẹ	Có	Không cháy nắng

Xét thuộc tính Chiều cao:

- $V_{CCao(Cao)} = \left(\frac{0}{1}, \frac{1}{1}\right) = (0, 1)$
- $V_{CCao(TB)} = \left(\frac{1}{1}, \frac{0}{1}\right) = (0, 1)$
- $V_{CCao(Th\ddot{a}p)} = \left(\frac{1}{2}, \frac{1}{2}\right)$

Tổng số vector đơn vị của thuộc tính Chiều cao là 2.

Xét thuộc tính Cân nặng:

- $V_{CN\ddot{a}ng(Nh\dot{e})} = \left(\frac{1}{2}, \frac{1}{2}\right)$
- $V_{CN\ddot{a}ng(TB)} = \left(\frac{1}{2}, \frac{1}{2}\right)$
- $V_{CN\ddot{a}ng(N\ddot{a}ng)} = (0, 0)$

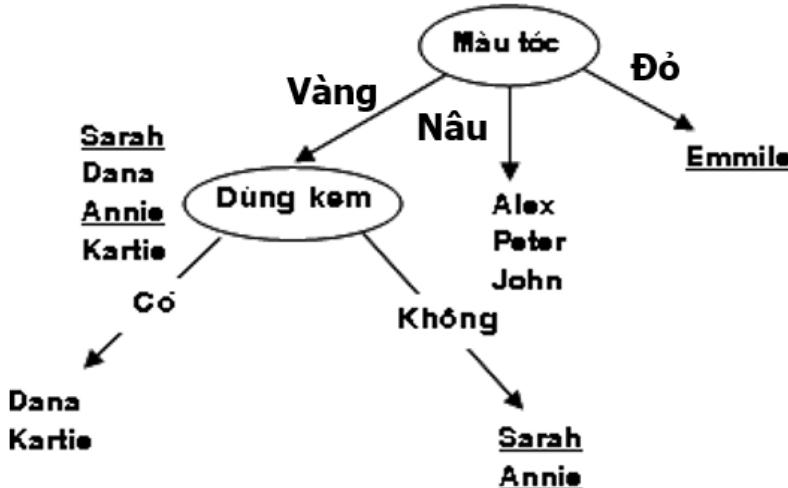
Tổng số vector đơn vị của thuộc tính Cân nặng là 0.

Xét thuộc tính Dùng kem:

- $V_{DKem(Có)} = \left(\frac{0}{2}, \frac{2}{2}\right) = (0, 1)$
- $V_{DKem(Không)} = \left(\frac{2}{2}, \frac{0}{2}\right) = (1, 0)$

Tổng số vector đơn vị của thuộc tính Dùng kem là 2.

Ta lại thấy lần này có đến hai thuộc tính có cùng max vector đơn vị. Tuy nhiên ta sẽ chọn thuộc tính Dùng kem vì thuộc tính này không còn phân hoạch những người cháy nắng và không cháy nắng.



Vậy ta rút ra tập lập luận là:

```

If (MauToc == "Nâu")
    KQ = "Không cháy nắng";
If (MauToc == "ĐỎ")
    KQ = "Cháy nắng";
If (MauToc == "Vàng" AND DungKem == "Có")
    KQ = "Không cháy nắng";
If (MauToc == "Vàng" AND DungKem == "Không")
    KQ = "Cháy nắng";
  
```

## 7.2. Giải thuật Độ hỗn loạn trung bình

Dựa vào bảng mô tả sau, hãy rút ra tập luật số 9 mang kết quả nhóm gì.

**Nhận xét.** Entropy bằng 0 nếu tất cả các ca trong S đều thuộc về cùng một lớp. Chẳng hạn như, nếu tất cả các ca đều dương thì  $P_{\oplus} = 1$  và  $P_{\ominus} = 1$ , do vậy:

$$\text{Entropy}(S) = -1\log_2 1 - 0\log_2 0$$

Entropy bằng 1 nếu tập S chứa số ca dương và âm bằng nhau. Nếu số các case này khác nhau thì Entropy nằm giữa 0 và 1.

Trường hợp tổng quát, nếu S bao gồm c lớp, thì Entropy của S được tính bằng 6 công thức sau:

$$\text{Entropy}(S) = \sum_{i=1}^c -p_i \log_2 p_i$$

trong đó  $p_i$  là tỉ lệ của các ca thuộc lớp  $i$  trong tập S.

STT	Vóc dáng	Quốc tịch	Gia cảnh	Nhóm
1	Nhỏ	Đức	Độc thân	A
2	Lớn	Pháp	Độc thân	A
3	Lớn	Đức	Độc thân	A
4	Nhỏ	Ý	Độc thân	B
5	Lớn	Đức	Có gia đình	B
6	Lớn	Ý	Độc thân	B
7	Lớn	Ý	Có gia đình	B
8	Nhỏ	Đức	Có gia đình	B
9	Nhỏ	Pháp	Có gia đình	?

Test lần 1:

Độ hỗn loạn trung bình của Vóc dáng:

- Lớn:  $2_A, 3_B$
- Nhỏ:  $1_A, 2_B$
- Độ hỗn loạn<sub>Vóc dáng</sub> =  $\frac{5}{8} \times \left( -\frac{2}{5} \log_2 \frac{2}{5} - \frac{3}{5} \log_2 \frac{3}{5} \right) + \frac{3}{8} \times \left( -\frac{1}{3} \log_2 \frac{1}{3} - \frac{2}{3} \log_2 \frac{2}{3} \right) = 0.95$

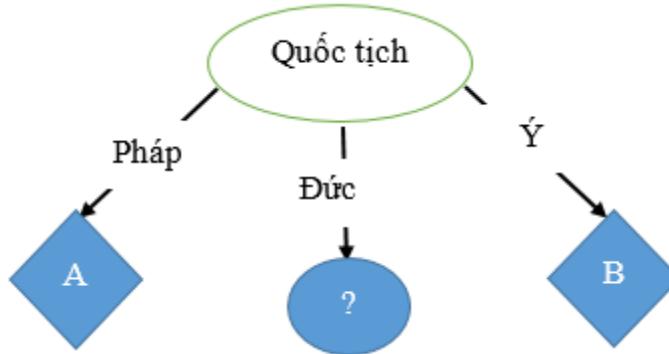
Độ hỗn loạn trung bình của Quốc tịch:

- Đức:  $2_A, 2_B$
- Ý:  $0_A, 3_B$
- Pháp:  $1_A, 0_B$
- Độ hỗn loạn<sub>Quốc tịch</sub> =  $\frac{4}{8} \left( -\frac{2}{4} \log_2 \frac{2}{4} - \frac{2}{4} \log_2 \frac{2}{4} \right) + 0 + 0 = 0.5$

Độ hỗn loạn trung bình của Gia cảnh:

- Độc thân:  $3_A, 2_B$
- Có gia đình:  $0_A, 3_B$
- Độ hỗn loạn<sub>Gia cảnh</sub> =  $\frac{5}{8} \left( -\frac{3}{5} \log_2 \frac{3}{5} - \frac{2}{5} \log_2 \frac{2}{5} \right) + 0 = 0.61$

So sánh ta thấy thuộc tính Quốc tịch có độ hỗn loạn trung bình thấp nhất nên ta chọn thuộc tính này làm nút gốc.



*Test lần 2:*

Sau khi test lần 1, ta có bảng dữ liệu nhỏ hơn như sau:

STT	Vóc dáng	Gia cảnh	Gia cảnh
1	Nhỏ	Độc thân	A
3	Lớn	Độc thân	A
5	Lớn	Có gia đình	B
8	Nhỏ	Có gia đình	B

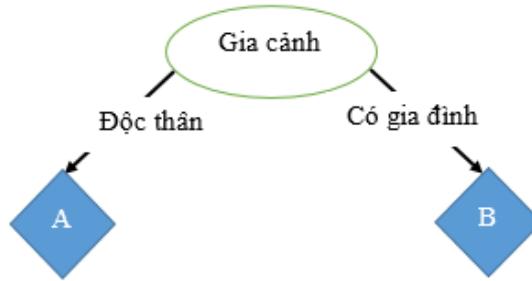
Độ hỗn loạn trung bình của Vóc dáng:

- Lớn:  $1_A, 1_B$
- Nhỏ:  $1_A, 1_B$
- Độ hỗn loạn<sub>Vóc dáng</sub> =  $\frac{2}{8} \left( -\frac{1}{2} \log_2 \frac{1}{2} - \frac{1}{2} \log_2 \frac{1}{2} \right) + \frac{2}{8} \left( -\frac{1}{2} \log_2 \frac{1}{2} - \frac{1}{2} \log_2 \frac{1}{2} \right) = 0.5$

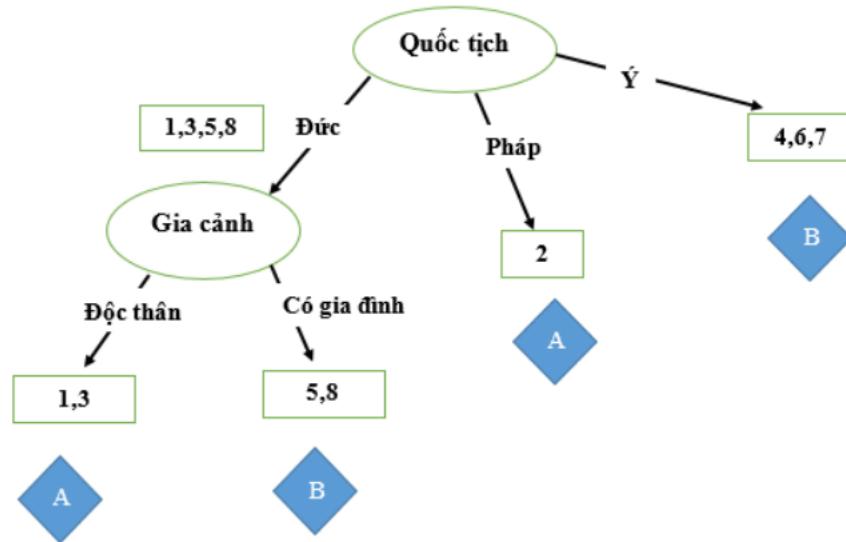
Độ hỗn loạn trung bình của Gia cảnh:

- Độc thân:  $2_A, 0_B$
- Có gia đình:  $0_A, 2_B$
- Độ hỗn loạn<sub>Gia cảnh</sub> =  $0 + 0 = 0$

Dễ dàng nhận thấy rằng độ hỗn loạn của Gia cảnh là thấp nhất nên ta chọn thuộc tính này làm nút nhánh.



Cuối cùng ta được cây quyết định:



Vậy ta có tập luật:

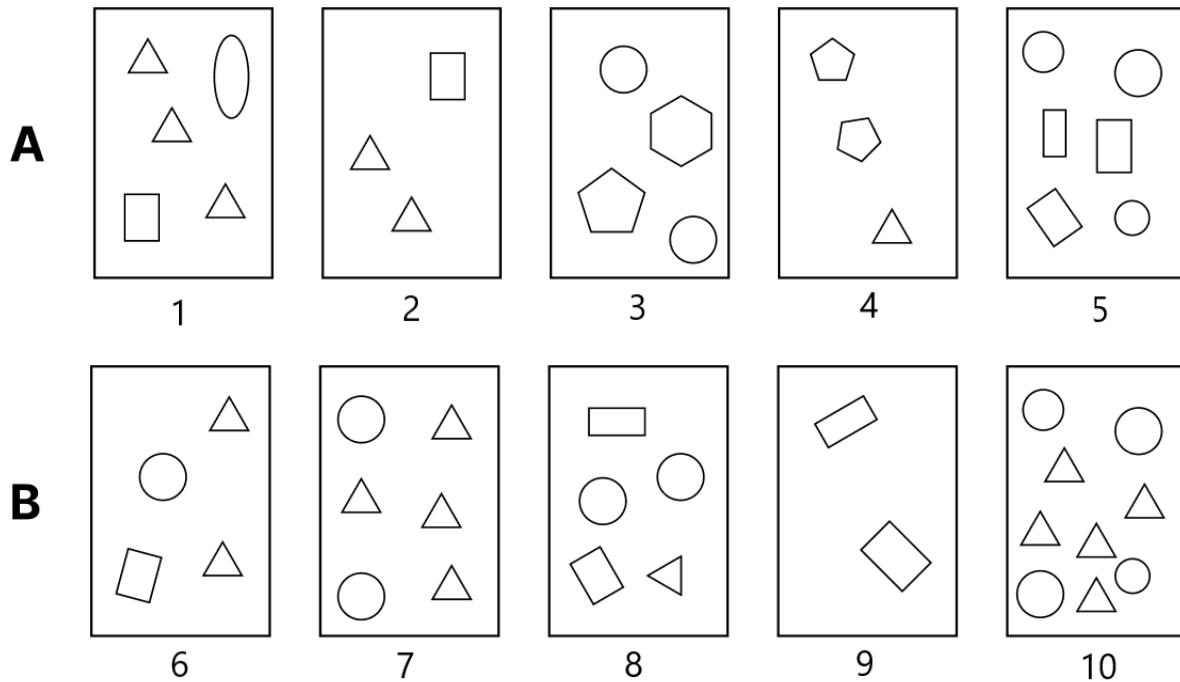
```

If (QuocTich == "Pháp")
    Nhóm = A;
If (QuocTich == "Ý")
    Nhóm = B;
If (QuocTich == "Đức" AND GiaCanh == "Độc thân")
    Nhóm = A;
If (QuocTich == "Đức" AND GiaCanh == "Có gia đình")
    Nhóm = B;
  
```

Từ tập luật trên ta có thể suy ra: Người có vóc dáng nhỏ, mang quốc tịch Pháp, có gia đình thì thuộc nhóm A.

### 7.3. Học qua logic

Cho 2 CSDL ảnh



❖ Xét CSDL A

- Gọi  $P_i$  ( $i = 1 \rightarrow 5$ ) lần lượt là các mệnh đề “có hình tam giác”, “có hình tròn”, “có hình elip”, “có hình tứ giác”, “có hình đa giác”

	P1	P2	P3	P4	P5
(1)	1	1	1	1	0
(2)	1	0	0	1	0
(3)	0	1	0	0	1
(4)	0	1	0	0	1
(5)	0	1	0	1	0

$x \in A :$

\* thoá mãn:  $f(A) = P_1 \cdot P_2 \cdot P_3 \cdot P_4 \cdot P_5 + P_1 \cdot \bar{P}_2 \cdot \bar{P}_3 \cdot P_4 \cdot \bar{P}_5$   
 $+ \bar{P}_1 \cdot P_2 \cdot \bar{P}_3 \cdot \bar{P}_4 \cdot P_5 + \bar{P}_1 \cdot \bar{P}_2 \cdot \bar{P}_3 \cdot P_4 \cdot P_5 + P_1 \cdot \bar{P}_2 \cdot \bar{P}_3 \cdot \bar{P}_4 \cdot \bar{P}_5$

$\Rightarrow f(A)$  là 1 hàm boolean, rút gọn  $f(A)$  ta có:

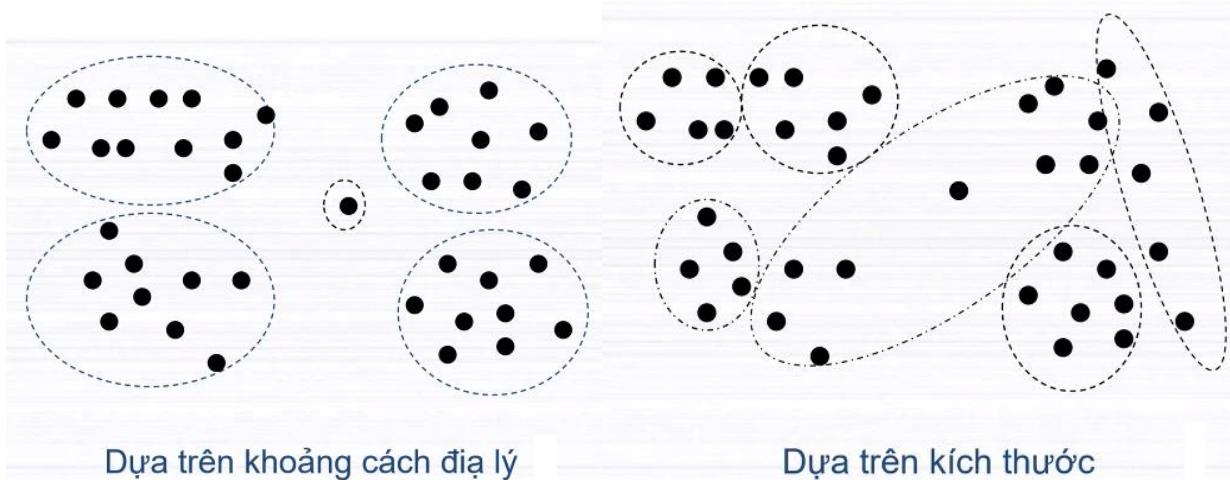
$$f(A) = \bar{P}_1 \cdot P_2 + P_1 \cdot (P_4 \cdot P_5 + \bar{P}_2 \cdot \bar{P}_3)$$

$x \in A : x \in \begin{cases} \bar{P}_1 \cdot P_2 \\ P_1 \cdot P_2 \cdot P_3 \\ P_1 \cdot \bar{P}_2 \cdot \bar{P}_3 \end{cases}$

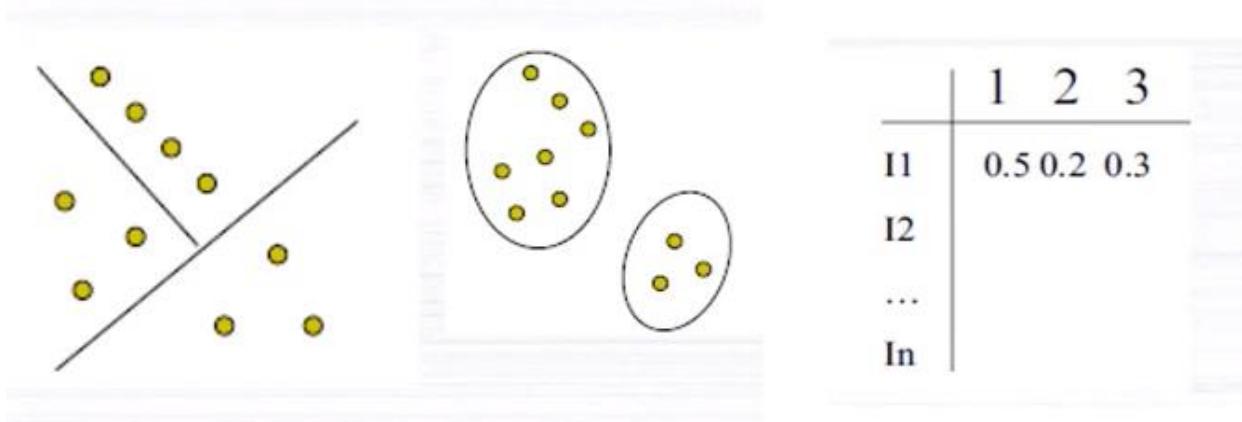
#### 7.4. Gom cụm

##### a. Giới thiệu

- ❖ Là quá trình gom nhóm các đối tượng thành cụm. Các đối tượng trong cùng một cụm có nhiều tính chất chung, và có những tính chất khác với những đối tượng của cụm khác.
  - ❖ Là dạng học **không giám sát**.
  - ❖ Cách phân loại
    - Dựa trên khoảng cách địa lý
    - Dựa trên kích thước



- ❖ Cách biểu diễn
    - Phân chia bằng ranh giới
    - Các khối cầu
    - Theo xác suất



- b. Tiêu chuẩn gom cụm
    - ❖ Tạo ra các cụm chất lượng
      - Sự giống nhau giữa các đối tượng chung cụm cao
      - Sự giống nhau giữa các đối tượng khác cụm thấp
  - c. Độ đo khoảng cách
    - ❖ Dùng để xác định sự khác nhau hay giống nhau giữa hai đối tượng

## Khoảng cách Minkowski:

$$d(i, j) = \sqrt{(|x_{i1} - x_{j1}|^q + |x_{i2} - x_{j2}|^q + \dots + |x_{ip} - x_{jp}|^q)}$$

Với  $\mathbf{i} = (x_{i1}, x_{i2}, \dots, x_{ip})$  và  $\mathbf{j} = (x_{j1}, x_{j2}, \dots, x_{jp})$  là các đối tượng dữ liệu  $p$ -chiều và  $q$  là số nguyên dương.

## Nếu $q=1$ , d là khoảng cách Manhattan :

$$d(i, j) = |x_{i1} - x_{j1}| + |x_{i2} - x_{j2}| + \dots + |x_{ip} - x_{jp}|$$

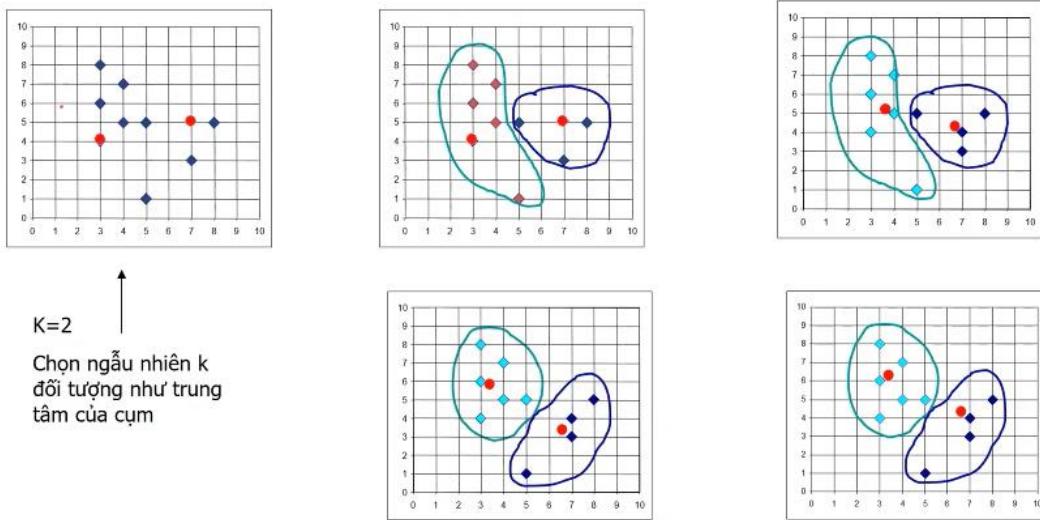
## Nếu $q=2$ , d là khoảng cách Euclide :

$$d(i, j) = \sqrt{(|x_{i1} - x_{j1}|^2 + |x_{i2} - x_{j2}|^2 + \dots + |x_{ip} - x_{jp}|^2)}$$

### 7.4.1. Thuật toán K-means

Cho số k mỗi cụm được biểu diễn bằng giá trị trung bình của dữ liệu trong cụm đó.

- **Bước 1 :** Chọn ngẫu nhiên k đối tượng như là những trung tâm cụm.
- **Bước 2 :** Gán từng đối tượng còn lại vào cụm có trung tâm cụm gần nó nhất (dựa trên độ đo khoảng cách Euclidean).
- **Bước 3 :** Tính lại giá trị trung tâm của từng cụm bằng giá trị trung bình mới của các đối tượng trong cụm. Di chuyển về trung tâm cụm mới.
- **Bước 4:** Nếu các trung tâm cụm không có gì thay đổi thì dừng. Ngược lại quay lại bước 2.



## Ví dụ

**Cho tập dữ liệu 1 chiều sau và k =2:  
 $\{2, 4, 10, 12, 3, 20, 30, 11, 25\}$**

➤ **Bước 1:** Giả sử chọn trung tâm cụm là  $m_1 = 3$  và  $m_2 = 4$

➤ **Bước 2:** Gán các đối tượng vào trung tâm. Thu được:

- $K_1 = \{2, 3\}$
- $K_2 = \{4, 10, 12, 20, 30, 11, 25\}$

➤ **Bước 3 :** Tính lại trung tâm cụm:  $m_1 = 2.5, m_2 = 16$

➤ **Bước 4:** Quay về bước 2

➤ **Bước 2:** Gán các đối tượng vào trung tâm mới. Thu được:

- $K_1 = \{2, 3, 4\}$
- $K_2 = \{10, 12, 20, 30, 11, 25\}$

➤ **Bước 3:** Tính lại trung tâm cụm:  $m_1 = 3, m_2 = 18$

➤ **Bước 4:** Quay về bước 2

➤ **Bước 2:** Gán các đối tượng vào trung tâm mới. Thu được:

- $K_1 = \{2, 3, 4, 10, 11, 12\}$
- $K_2 = \{20, 25, 30\}$

➤ **Bước 3:** Tính lại trung tâm cụm :  $m_1 = 7, m_2 = 25$

➤ **Bước 4:** Thuật toán dừng vì các trung tâm cụm không thay đổi.

**Ví dụ**  
**Cho tập dữ liệu**

Điểm	Hoành độ	Tung độ
1	1	3
2	2.5	3
3	4.5	3
4	1	1.5
5	1.5	4
6	3	4.5
7	2	2
8	3	2
9	4	4
10	5	5
11	3	1

### Tính các khoảng cách

$$d(i, j) = \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2}$$

Centroid	X	Y
1	1.0	3.0
2	2.5	3.0
3	4.5	3.0

Điểm	Hoành độ	Tung độ	d(1,j)	d(2,j)	d(3,j)
1	1	3	0	1.5	3.5
2	2.5	3	1.5	0	2
3	4.5	3	3.5	2	0
4	1	1.5	1.5	2.1	3.8
5	1.5	4	1.1	1.4	3.2
6	3	4.5	2.5	1.6	2.1
7	2	2	1.4	1.1	2.7
8	3	2	2.2	1.1	1.8
9	4	4	3.2	1.8	1.1
10	5	5	4.5	3.2	2.1
11	3	1	2.8	2.1	2.5

### Gán đối tượng vào cụm có khoảng cách nhỏ nhất

Điểm	Hoành độ	Tung độ	$d(1,j)$	$d(2,j)$	$d(3,j)$
1	1	3	0	1.5	3.5
2	2.5	3	1.5	0	2
3	4.5	3	3.5	2	0
4	1	1.5	1.5	2.1	3.8
5	1.5	4	1.1	1.4	3.2
6	3	4.5	2.5	1.6	2.1
7	2	2	1.4	1.1	2.7
8	3	2	2.2	1.1	1.8
9	4	4	3.2	1.8	1.1
10	5	5	4.5	3.2	2.1
11	3	1	2.8	2.1	2.5

### Tính lại trung tâm cụm

Cụm 1	X	Y
1	1	3
4	1	1.5
5	1.5	4

Cụm 2	X	Y
2	2.5	3
6	3	4.5
7	2	2
8	3	2
11	3	1

Cụm 3	X	Y
3	4.5	3
9	4	4
10	5	5

### Tính các khoảng cách

$$d(i,j) = \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2}$$

Centroid	X	Y
1'	1.5	2.9
2'	2.7	1.7
3'	4.1	4.1

Điểm	Hoành độ	Tung độ	$d(1',j)$	$d(2',j)$	$d(3',j)$
1	1	3	0.5	2.1	3.3
2	2.5	3	1.0	1.3	1.9
3	4.5	3	3	2.2	1.2
4	1	1.5	1.5	1.7	4
5	1.5	4	1.1	2.6	2.6
6	3	4.5	2.2	2.8	1.2
7	2	2	1	0.8	3
8	3	2	1.7	0.4	2.4
9	4	4	2.7	2.6	0.5
10	5	5	4.1	4	1.3
11	3	1	2.4	0.8	3.3

### Tính lại trung tâm cụm

Cụm 1	X	Y
1	1	3
2	2.5	3
4	1	1.5
5	1.5	4

Cụm 2	X	Y
7	2	2
8	3	2
11	3	1

Cụm 3	X	Y
3	4.5	3
6	3	4.5
9	4	4
10	5	5

## 8. Neural Network

$$X = \sum_{i=1}^n x_i * w_i$$

$$Y = \begin{cases} +1, & X \geq 0 \\ -1, & X < 0 \end{cases}$$

1	2	3
4	5	6
7	8	9

Vs

1	2	3
4	5	6
7	8	9

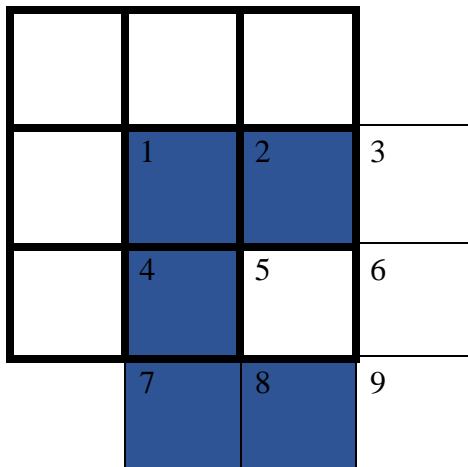
Với mỗi hình, ta:

- Ở tầng 1: Đặt một máy cảm biến 9 pixel ở mỗi vị trí ô vuông. Tổng cộng có 9 máy cảm biến ở tầng 1.
- Ở tầng 1, mỗi máy cảm biến các pixel sẽ cho ra bộ chín số (1 tại chỗ có tô đen hoặc 0 tại chỗ ko có), đây là input cho mạng nơ ron của ta.
- Input này qua bộ trọng số:  

-1	
----	--

-1	
-1	
-1	
2	0.5
-1	
-1	
-1	
-1	

để hiện ra output là 1 hay -1. **Ví dụ** khi đặt máy cảm biến ở vị trí số 1 ở chữ C:



thì ra bộ số 000 011 010, ta có

$$-1*0 + -1*0 + -1*0 + -1*0 + 2*1 + -1*1 + -1*0 + -1*1 + -1*0 = 0 < 0.5, \text{ vậy nên output là } -1$$

- 9 máy cảm biến cho ra 9 output, 9 output này là bộ số input cho tầng 2 với bộ trọng số như sau:

1	
1	
1	
1	
1	
1	
1	
1	
1	0.5

- Nhận xét: từ bộ trọng số tầng 2, ta thấy, để có kết quả  $Y=1$  thì  $X$  phải  $\geq 0.5$ . Nghĩa là trong 9 input từ tầng 1 thì chỉ cần ít nhất 1 input có giá trị là 1.
- Đáng chú ý, nếu ta đặt thử ở tất cả 9 vị trí của 2 hình thì chỉ duy nhất tại vị trí số 8 của chữ T là có bộ số 010 010 000, và duy nhất bộ số này cho ra output là 1. Chính nhờ nó mà input tầng 2 thỏa điều kiện trên.
- Nghĩa là, kết quả cuối cùng thì chữ C sẽ ra kết quả =-1 và T sẽ ra kết quả =1

Bài tập: tìm bộ trọng số (ở cả 2 tầng) để có thể phân biệt chữ U và chữ H

## 9. Link bài tập

- ❖ [Chương trình dịch \(txnam.net\)](#)
- ❖ [\(PDF\) AI Trí tuệ nhân tạo Bài tập Logic Câu 1 | HungTruong VFX - Academia.edu](#)
- ❖ [Bài tập lý thuyết cơ sở trí tuệ nhân tạo - Tài liệu, Luận văn \(thuvientailieu.vn\)](#)

## NHÁP

### Cơ sở trí tuệ nhân tạo

STT	Hình dáng	Chiều cao	Giới tính	KQ
1	To	TB	Nam	Á
2	Nhỏ	Thấp	Nam	Á
3	Nhỏ	TB	Nam	Á
4	To	Cao	Nam	Âu
5	Nhỏ	TB	Nữ	Âu
6	Nhỏ	Cao	Nam	Âu
7	Nhỏ	Cao	Nữ	Âu
8	To	TB	Nữ	Âu

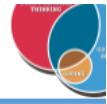
STT	Hình dáng	Chiều cao	Giới tính	KQ
1	To	TB	Nam	Á
3	Nhỏ	TB	Nam	Á
5	Nhỏ	TB	Nữ	Âu
8	To	TB	Nữ	Âu

$$V(\text{hình dáng} = \text{To}) = (\hat{A}_{\text{to}}, \hat{A}_{\text{u}_\text{to}})$$

$$\hat{A}_{\text{to}} = \frac{\Sigma \text{quan sát Á có hình dáng to}}{\Sigma \text{quan sát có hình dáng to}}$$

$$\hat{A}_{\text{u}_\text{to}} = \frac{\Sigma \text{quan sát Âu có hình dáng to}}{\Sigma \text{quan sát có hình dáng to}}$$

## Ví dụ 3: sở thích ăn uống



- Cho cơ sở tri thức

1. Hùng thích tất cả các loại thực phẩm
2. Táo là thực phẩm
3. Gà là thực phẩm
4. Bất cứ thứ gì mọi người ăn và không bị hại đó là thực phẩm
5. Phong ăn đậu phộng và vẫn còn sống
6. Lan ăn bất cứ thứ gì Phong ăn

- Sử dụng phương pháp hợp giải để:

- Chứng minh Hùng thích đậu phộng
- Trả lời câu hỏi “Lan ăn thực phẩm nào”

Ví dụ 3: Sở thích ăn uống.

\*  $\text{Thich}(x) \rightarrow \text{Thich}(A, x) : A \text{ thích } x.$   
 $\text{Thucpham}(x) : x \text{ là thực phẩm.}$   
 $\text{An}(A, x) : A \text{ ăn } x$   
 $\text{Hai}(A, x) : A \text{ bị hại bởi } x$

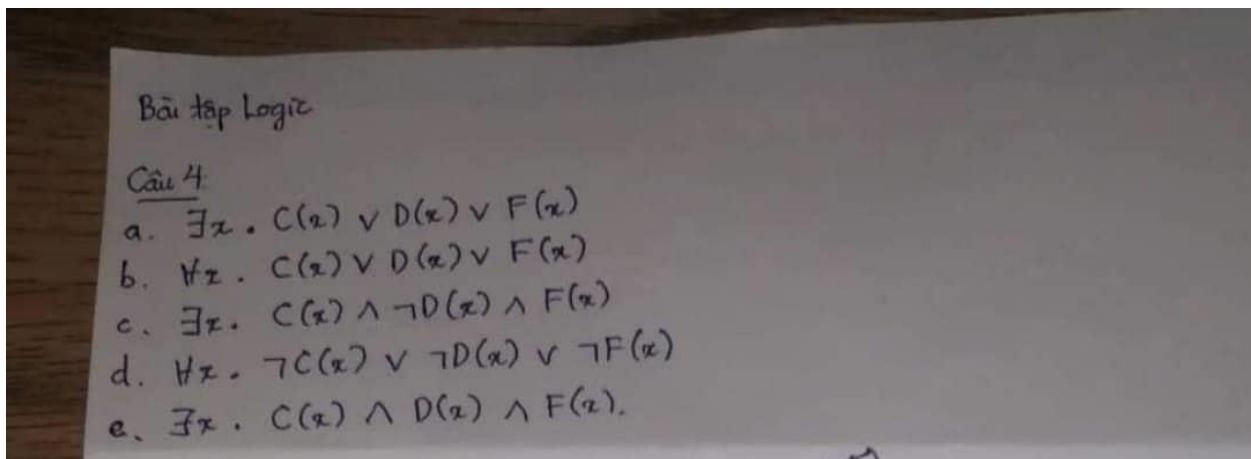
- |   |   |                            |
|---|---|----------------------------|
| 1. $\text{Thucpham}(x) \rightarrow \text{Thich}(\text{Hung}, x)$<br>2. $\text{Thucpham}(\text{Tao})$<br>3. $\text{Thucpham}(\text{Ga})$<br>4. $\text{An}(A, x) \wedge \neg \text{Hai}(A, x) \rightarrow \text{Thucpham}(x)$<br>5. $\text{An}(\text{Phong}, \text{DauPhong}) \wedge \neg \text{Hai}(\text{Phong}, \text{DauPhong})$<br>6. PR: $\text{An}(\text{Phong}, x) \rightarrow \text{ba An}(\text{Lan}, x)$ | } | Khai báo vị trí.           |
| (1) $\neg \text{Thucpham}(x) \vee \text{Thich}(\text{Hung}, x)$<br>(2) $\text{Thucpham}(\text{Tao})$<br>(3) $\text{Thucpham}(\text{Ga})$<br>(4) $\neg \text{An}(A, x) \vee \text{Hai}(A, x) \vee \text{Thucpham}(x)$<br>(5) $\text{An}(\text{Phong}, \text{DauPhong})$<br>(6) $\neg \text{Hai}(\text{Phong}, \text{DauPhong})$<br>(7) $\neg \text{An}(\text{Phong}, x) \vee \text{An}(\text{Lan}, x)$             |   | Biểu diễn tri thức.        |
| (8) $\text{An}(\text{Lan}, \text{DauPhong})$<br>(9) $\text{Thucpham}(\text{DauPhong})$<br>(10) $\text{Thich}(\text{Hung}, \text{DauPhong})$   | } | Hợp giải và tách.          |
| $((6), (7), \{x / \text{DauPhong}\})$<br>$((4), (5), (6), \{x / \text{DauPhong}\})$<br>$((1), (9), \{x / \text{DauPhong}\})$  |   | ((6), (7), {x / DauPhong}) |
|   |   |                            |

Vậy: Hung thích ăn đậu phộng và Lan ăn đậu phộng

## Câu 4

Đặt  $C(x)$ : “ $x$  có một con mèo”,  $D(x)$ : “ $x$  có một con chó”,  $F(x)$ : “ $x$  có một con chồn”. Biểu diễn các phát biểu sau theo  $C(x)$ ,  $D(x)$ ,  $F(x)$ , các lượng từ và các phép nối logic. Xét không gian biến là các sinh viên trong lớp

- a) Một sinh viên trong lớp có một con mèo, một con chó hay một con chồn.
- b) Tất cả sinh viên trong lớp có một con mèo, một con chó hay một con chồn.
- c) Một sinh viên nào đó có một con mèo và một con chồn nhưng không có chó.
- d) Không có sinh viên nào trong lớp có một con mèo, một con chó và một con chồn.
- e) Với mỗi loại con vật trên, có một sinh viên trong lớp có một con.



## Câu 5

Đặt: L(x): “x là một nhà logic”; C(x): “x uống café”; W(x): “x làm việc chăm chỉ”, T(x): “x phát biểu định lý”; f(x): hàm trả ra giá trị là bạn của x (giả sử mỗi người có đúng 1 bạn).

1. Phát biểu các câu sau dưới dạng logic bậc nhất (có sử dụng dấu =):

- a. Không nhà logic nào uống café
- b. Bất kỳ ai là một nhà logic cũng đều là bạn của ai đó
- c. Không người nào phát biểu được định lý lại có một người bạn uống café.

10/15/2008

---

Môn: Trí tuệ nhân tạo

Giảng viên: Tô Hoài Việt

d. Ai có một người bạn làm việc chăm chỉ thì hoặc là một nhà logic hoặc cũng là một người làm việc chăm chỉ  
e. Mọi người bạn là một nhà logic.

2. Từ các tiền đề sau:

- a. Tất cả nhà logic đều uống café.
- b. Bất kỳ ai không phát biểu được định lý đều không uống café.
- c. Có một số người mà bạn của họ là nhà logic.

Chứng minh rằng: Có một nhà logic uống café và phát biểu được định lý.

Câu 5:

~~1. a.  $\forall x. L(x) \Rightarrow \neg C(x)$  ( $\neg L(x) \vee \neg C(x)$ )~~

~~b.  $\forall x, \exists y. L(x) \Rightarrow$~~

~~b.  $\forall x. \exists y. L(x) \vee$~~

1. a.  $\forall x. \neg L(x) \vee \neg C(x)$

b.  $\forall x. \neg L(x) \vee \exists y. y = f(x)$

c.  $\forall x. \neg T(x) \vee \neg C(f(x))$

d.  $\forall x. \neg W(f(x)) \vee L(x) \vee W(x)$

e.  $\forall x. L(f(x))$

2. a.  $\forall x. \neg L(x) \vee C(x)$

b.  $\forall x. T(x) \vee \neg C(x)$

c.  $\exists x. L(f(x))$

KL:  $\exists x. C(x) \wedge T(x)$

Viết lại mệnh đề và kết luận; ta có:

(1)  $\neg L(x) \vee C(x)$

(2)  $T(x) \vee \neg C(x)$

(3)  $L(f(x))$

(4)  $C(x)$

(5)  $T(x)$

Chứng minh:

(1)  $\neg L(x) \vee C(x)$

(2)  $T(x) \vee \neg C(x)$

(3)  $L(f(x))$

(4)  $C(f(x))$

(5)  $T(f(x))$

(6)  $C(f(x))$   $((1), (3), \{x/f(x)\})$

(7)  $T(f(x))$   $((2), (6), \{x/f(x)\})$

$\Rightarrow \exists x. C(x) \wedge T(x)$ .  $((6), (7), \{f(x)/x\})$ .

Câu 3 Đề thi năm 2021-2022 (A7)

$X = \{1, 2, 3, 4, 5, 6, 7, 8\}$

$D = (G, R, C, B)$

$C = \{(1+2), (1+3), (1+4), (2+4), (2+6), (3+4), (3+7), (4+5), (5+6), (5+7), (6+7), (6+8), (7+8)\}$

a.  $1=R, 4=C, 5=R, 8=C, 6=B$   
Ta có bảng ràng buộc:

	R	C	B
1	✓		
2			✓
3		✓	
4			✓
5	✓		
6			✓
7		✓	
8		✓	

- Gán  $R \Rightarrow 1=R$ . Khi đó vì  $2, 3, 4$  gần 1 nên loại bỏ  $R$  ra khỏi giá trị của  $2, 3, 4$ .
- Gán  $4=C$ . Khi đó vì  $2, 3, 5$  gần 4 nên loại  $C$  ra khỏi giá trị  $2, 3, 5$ .
- Gán  $5=R$ . Khi đó vì  $6, 7$  gần 5 nên loại  $R$  ra khỏi giá trị  $6$  và  $7$ .
- Gán  $8=C$ . Khi đó vì  $6$  và  $7$  gần 5 nên loại  $C$  ra khỏi  $6$  và  $7$ .
- Gán  $6=B$ . Khi đó vì  $6$  và  $7$  gần  $6$  nên loại  $B$  ra khỏi  $6$  và  $7$ .
- Lú này:  $2$  và  $7$  đều không còn giá trị khả dụng nên việc gán giá trị là không thể, đồng nghĩa phải thực hiện quay lui và dừng tìm kiếm.

b. Thuật toán AC-3 sẽ phát hiện ràng buộc sớm hơn trong trường hợp này.

Cách thức hoạt động: với mỗi giá trị được gán, thuật toán AC-3 sẽ kiểm tra ràng buộc của đối tượng được gán và các đối tượng liên kề nó.

c. Triển khai thuật toán

D	C
$1=R$	$C_1: 1 \neq 2$
$2=R, C, B$	$C_2: 2 \neq 4$
$3=R, C, B$	$C_3: 2 \neq 6$
$4=C$	
$5=R$	
$6=B$	
$7=R, C, B$	
$8=C$	

~~C1:  $1 \neq 2 \Rightarrow D_2 = (C, B) \rightarrow$  Xét  $C_2: 2 \neq 4$  và  $C_3: 2 \neq 6$~~

~~C2:  $2 \neq 4 \Rightarrow D_2 = (B) \rightarrow$  Xét  $C_4: 4 \neq 3$  và  $C_5: 4 \neq 5$~~

~~C3:  $2 \neq 6 \Rightarrow D_2 = \emptyset \rightarrow$  Dừng thuật toán~~

- Gán  $1=R$ . Xét  $C_1: 1 \neq 2 \Rightarrow D_2 = (C, B)$ . Thêm  $R$   $C_2: 2 \neq 4$  và  $C_3: 2 \neq 6$

- Gán  $4=C$ . Xét  $C_2: 2 \neq 4 \Rightarrow D_2 = (B)$

- Gán  $6=B$ . Xét  $C_3: 2 \neq 6 \Rightarrow D_2 = \emptyset$ . Dừng

Vậy AC-3 phát hiện sai cố/soi hơn

**Question 1.** Convert the following English sentences into FOL sentences, using only the predicates given inside the square brackets.

1. All green apples are sour. [Green<sup>1</sup>, Apple<sup>1</sup>, Sour<sup>1</sup>]

$\forall x \text{.Green}(x) \wedge \text{Apple}(x) \rightarrow \text{Sour}(x)$

2. All babies love some green apples. [Baby<sup>1</sup>, Loves<sup>2</sup>, Apple<sup>1</sup>, Green<sup>1</sup>]

$\forall x \text{Baby}(x) \rightarrow [\exists y \text{.Green}(y) \wedge \text{Apple}(y) \wedge \text{Loves}(x,y)]$

3. Some babies do not love any sour apple. [Baby<sup>1</sup>, Loves<sup>2</sup>, Apple<sup>1</sup>, Sour<sup>1</sup>]

$\exists x \text{Baby}(x) \wedge [\forall y \text{.Sour}(y) \wedge \text{Apple}(y) \rightarrow \neg \text{Loves}(x,y)]$

4. Mary eats only one apple. [Apple<sup>1</sup>, Eat<sup>2</sup>]

$\exists x \text{Apple}(x) \wedge \text{Eat}(\text{Mary},x) \wedge [\forall y \text{.Apple}(y) \wedge \neg(x=y) \rightarrow \neg \text{Eat}(\text{Mary},y)]$

**Question 2.** Find (if it were possible) the Most General Unifier (MGU) of

a)  $P(g(h(x)), f(g(h(b))), f(x))$  and  $P(y, f(y), z)$

b)  $P(g(h(x)), f(h(y)), y)$  and  $P(g(z), f(z), h(a))$

c)  $P(x, h(b), h(x))$  and  $P(f(g(y)), y, h(f(g(h(a)))))$

d)  $P(x, g(x), z)$  and  $P(f(y), g(f(b)), h(y))$

e)  $P(f(g(x)), g(b), h(x))$  and  $P(f(y), y, h(c))$

f)  $P(x, h(x), h(y))$  and  $P(f(g(z)), h(f(g(b))), h(z))$

a)  $\theta = \{y / g(h(b)), x / b, z / f(b)\}$

b)  $\theta = \{x / h(a), y / h(a), z / h(h(a))\}$

c) No MGU

d)  $\theta = \{x / f(b), y / b, z / h(b)\}$

e) No MGU

f)  $\theta = \{x / f(g(b)), z / b, y / b\}$

**Question 3.** Consider the following text. "Anyone passing his history exam and winning the lottery is happy. But anyone who studies or is lucky can pass all his exams. John did not study but John is lucky. Anyone who is lucky wins the lottery."

- a) For each of the axiom above, write the FOL sentence that best expresses its intended meaning, using only the following predicates

PASS(x, y): "x passes the y exam"      HAPPY(x): "x is happy"      LUCKY(x): "x is lucky"

STUDY(x): "x studies"

WINLOT(x): "x wins the lottery"

- 1)  $\forall x. \text{PASS}(x, \text{history exams}) \wedge \text{WINLOT}(x) \Rightarrow \text{HAPPY}(x)$
- 2)  $\forall x. \text{STUDY}(x) \vee \text{LUCKY}(x) \Rightarrow \forall y. \text{PASS}(x, y)$
- 3)  $\neg \text{STUDY}(\text{John}) \wedge \text{LUCKY}(\text{John})$
- 4)  $\forall x. \text{LUCKY}(x) \Rightarrow \text{WINLOT}(x)$
- b) Convert the above FOL clauses to clausal form
- 1)  $\neg \text{PASS}(x, \text{history exams}) \vee \neg \text{WINLOT}(x) \vee \text{HAPPY}(x)$
  - 2)  $\neg \text{STUDY}(x) \vee \text{PASS}(x, y)$
  - 3)  $\neg \text{LUCKY}(x) \vee \text{PASS}(x, y)$
  - 4)  $\neg \text{STUDY}(\text{John})$
  - 5)  $\text{LUCKY}(\text{John})$
  - 6)  $\neg \text{LUCKY}(x) \vee \text{WINLOT}(x)$
- c) Use resolution to answer the question "Is John happy?"
- 7)  $\neg \text{HAPPY}(\text{John})$  Negation of conclusion
  - 8)  $\neg \text{PASS}(\text{John}, \text{history exams}) \vee \neg \text{WINLOT}(\text{John})$  from 1 and 7. 0 = {x/John}
  - 9)  $\text{WINLOT}(\text{John})$  from 5 and 6. 0 = {x/John}
  - 10)  $\neg \text{PASS}(\text{John}, \text{history exams})$  from 8 and 9. 0 = {x/John}
  - 11)  $\text{PASS}(\text{John}, y)$  from 3 and 5. 0 = {x/John}
  - 12)  $\bullet$  from 10 and 11. 0 = {x/John, y/history}

Conclusion: Therefore, John is happy.

**Question 4.** Consider the following KB.

1. Buffalo(x) $\wedge$ Pig(y) $\rightarrow$ Faster(x,y)	4. Buffalo(Bob)
2. Pig(y) $\wedge$ Slug(z) $\rightarrow$ Faster(y,z)	5. Pig(Pat)
3. Faster(x,y) $\wedge$ Faster(y,z) $\rightarrow$ Faster(x,z)	6. Slug(Steve)

Use forward chaining in first-order logic to prove **Faster(Bob, Steve)**. If several rules apply, use the one with the smallest number. Do not forget to indicate the unification at every step.

7. Faster(Bob, Pat) from 1 and 4-5. 0 = {x / Bob, y / Pat}
8. Faster(Pat, Steve) from 2 and 5-6. 0 = {x / Bob, y / Pat, z / Steve}
9. Faster(Bob, Steve) from 3 and 7-8. 0 = {x / Bob, y / Pat}

Thus, KB entails Faster(Bob, Steve) using forward chaining.

## ➔ ##LOGIC BẬC NHẤT

- tên riêng được coi là hằng (const). VD: Lan, John

- biểu diễn: verb(verb's main subject, O).

VD: Cháu(x,y): x là cháu của y

- lượng từ 'với mọi' (thường đi kèm với dấu kéo theo):  $\forall x. P$

Ex: Sinh viên CNTT thì thông minh  $\rightarrow \forall x. \text{Sinh-viên}(x, \text{CNTT}) \rightarrow \text{thông-minh}(x)$

- lượng từ ‘tồn tại’ (thường đi kèm với dấu giao “ $\wedge$ ”):  $\exists x$ . P. chỉ cần có tồn tại 1 cái gì đó thì xài dc

\*Hợp giải logic bậc nhất: tương tự như “chứng minh” của Logic (mục e)

$\begin{array}{c} \forall x, P(x) \rightarrow Q(x) \\ P(A) \\ \hline Q(A) \end{array}$	<p>Tam đoạn luận: Mọi người đều chết Socrates là người Socrates chết</p>
$\begin{array}{c} \forall x, \neg P(x) \vee Q(x) \\ P(A) \\ \hline Q(A) \end{array}$	<p>Tương đương theo định nghĩa của phép Suy ra</p>
$\begin{array}{c} \neg P(A) \vee Q(A) \\ P(A) \\ \hline Q(A) \end{array}$	<p>Thay A vào x, vẫn đúng khi đó Hợp giải Mệnh đề</p>

thế A vào x<sup>(\*)</sup> xong hợp giải, VD như hợp giải  $\neg P(x) \vee P(A)$  thì sẽ thành 1

→ Thế nào là 1 phép thế đúng đắn?

VD 1:

<input type="checkbox"/>	Chứng minh rằng $(P(x) \Rightarrow Q(x))$ và $P(A)$ suy dẫn logic $\exists z. Q(z)$
1.	$\neg P(x) \vee Q(x)$ Tiền đề
2.	$P(A)$ Tiền đề
3.	$\neg Q(z)$ Kết luận
4.	$\neg P(z)$ 1, 3 $\theta = \{x/z\}$
5.	<del>False</del> 2, 4 $\theta = \{x/z, z/A\}$

VD 2:

<input type="checkbox"/> Cho trước ( $P(x) \Rightarrow Q(x)$ ) và $P(A)$ và $P(B)$ , tìm $z$ sao cho $Q(z)$ là đúng	
1. $\neg P(x) \vee Q(x)$	Tiền đề
2. $P(A)$	Tiền đề
3. <del><math>P(B)</math></del>	Tiền đề
4. $\neg Q(z)$	Kết luận
5. <del><math>\neg P(z)</math></del>	1, 4 $\theta = \{x/z\}$
6. False	2, 5 $\theta = \{x/z, z/A\}$
7. False	3, 5 $\theta = \{x/z, z/B\}$

VD 3:

a) Art là cha của Bob và Bud

Bob là cha của Cal và Coe

Ông nội là cha của cha  $\rightarrow \forall x,y,z. F(x,y) \wedge F(y,z) \Rightarrow G(x,z) \rightarrow$  lấy phủ định

Hỏi: Art có là ông của Coe?

 $\rightarrow$  giả sử Art KHÔNG là ông Coe $\rightarrow !G(Art, Coe)$ Tip: thường hội chuẩn với kết luận trước

1	$F(Art, Bob)$	Tiền đề
2	$F(Art, Bud)$	Tiền đề
3	$F(Bob, Cal)$	Tiền đề
4	$F(Bob, Coe)$	Tiền đề
5	$\neg F(x,y) \vee \neg F(y,z) \vee G(x,z)$	Tiền đề
6	$\neg G(Art, Coe)$	Kết luận
7	$\neg F(Art,y) \vee \neg F(y, Coe)$	5,6; theta={x/Art, z/Coe}
8	$\neg F(Bob, Coe)$	1,7; theta={x/Art, z/Coe, y/Bob}
9	False (dpcm)	4,8; theta={x/Art, z/Coe, y/Bob}

b) Art là cha của Bob và Bud

Bob là cha của Cal và Coe

Ông nội là cha của cha  $\rightarrow \forall x,y,z. F(x,y) \wedge F(y,z) \Rightarrow G(x,z)$

Ai là cháu của Art?

→ gs ko ai là cháu của Art = Art ko là ông của bất kỳ ai

→  $\forall t. !G(Art, t)$

1	F(Art, Bob)	Tiền đề
2	F(Art, Bud)	Tiền đề
3	F(Bob, Cal)	Tiền đề
4	F(Bob, Coe)	Tiền đề
5	$!F(x,y) \vee !F(y,z) \vee G(x,z)$	Tiền đề
6	$!G(Art, t)$	Kết luận
7	$!F(Art,y) \vee !F(y,t)$	5,6; {x/Art, z/t}
8	$!F(Bob,t)$	1,7; {y/Bob, x/Art, z/t}
9	$!F(Bud,t)$	2,7; {y/Bud, x/Art, z/t} → có thể xài lại 7 (bỏ bước này cũng dc)
10	false	3,8; {t/Cal, y/Bob, x/Art, z/t}
➔ Cal là cháu của Art		

b) Art là cha của Bob và Bud

Bob là cha của Cal và Coe

Ông nội là cha của cha →  $\forall x,y,z. F(x,y) \wedge F(y,z) \Rightarrow G(x,z)$

Hỏi: Các cặp ông cháu?

→  $\forall x,z. G(x,z)$  (lưu ý về ngữ nghĩa mà dùng lại x và z, chứ ko thêm biến mới)

1	F(Art, Bob)	Tiền đề
2	F(Art, Bud)	Tiền đề
3	F(Bob, Cal)	Tiền đề
4	F(Bob, Coe)	Tiền đề

5	$\neg F(x,y) \vee \neg F(y,z) \vee G(x,z)$	Tiền đề
6	$\neg G(x,z)$	Kết luận
7	$\neg F(x,y) \vee \neg F(y,z)$	5,6
8	$\neg F(Bob,z)$	1,7; {x/Art, y/Bob}
9	$\neg F(Bud,z)$	2,7; {x/Art, y/Bud}
10	false	3,8; {x/Art, y/Bob, z/Cal}
11	false	4,8; {x/Art, y/Bob, z/Coe}
<b>➔ 2 cặp ông cháu Art-Cal và Art-Coe</b>		

//BT:

Cho các câu sau:

1. Jack sở hữu một con chó.
2. Ai sở hữu một con chó là người yêu động vật.
3. Người nào yêu động vật thì không giết động vật.
4. Jack giết Tuna hoặc Curiosity giết Tuna
5. Tuna là một con mèo.
6. Mọi con mèo đều là động vật.

Hãy sử dụng các vị từ sau đây biểu diễn các câu trên về dạng logic bậc nhất.

D(x): "x là con chó"	O(x, y): "x sở hữu y"
L(x): "x là người yêu động vật"	A(x): "x là động vật"
K(x, y): "x giết y"	C(x): "x là con mèo"

Từ các câu trên, hãy chứng minh xem Curiosity có giết Tuna hay không?

$$1. \exists x. D(x) \wedge O(Jack, x)$$

$$= D(A) \wedge O(Jack, A) \text{ (Skolem - thay tên mới cho tất cả lượng từ } \exists)$$

\*Lưu ý: khi đi vào bảng phải tách D(A) và O(Jack,A) riêng vì chúng có dấu “ $\wedge$ ”

$$2. \forall x. (\exists y. D(y) \wedge O(x, y)) \rightarrow L(x)$$

$$= \forall x. \neg(\exists y. D(y) \wedge O(x, y)) \vee L(x)$$

$$= \forall x. \forall y. \neg D(y) \vee \neg O(x, y) \vee L(x)$$

$$= \neg D(y) \vee \neg O(x, y) \vee L(x) \text{ (luật 5 - bỏ } \forall)$$

$$3. \forall x. L(x) \rightarrow (\forall y. A(y) \rightarrow \neg K(x, y)) \rightarrow \text{vì là lượng từ với mọi nên dùng kéo theo}$$

$$= \forall x. \neg L(x) \vee (\forall y. \neg A(y) \vee \neg K(x, y))$$

$\neg L(x) \vee \neg A(y) \vee \neg K(x,y)$  (luật 5 - bỏ  $\forall$ )

4.  $K(\text{Jack}, \text{Tuna}) \vee K(\text{Curiosity}, \text{Tuna})$

5.  $C(\text{Tuna})$

6.  $\forall x. C(x) \rightarrow A(x)$

$\neg C(x) \vee A(x)$

7.  $K(\text{Curiosity}, \text{Tuna})$

GOAL

1	D(A)	Tiền đề
2	O(Jack, A)	Tiền đề
3	$\neg D(y) \vee \neg O(x, y) \vee L(x)$	Tiền đề
4	$\neg L(x) \vee \neg A(y) \vee \neg K(x,y)$	Tiền đề
5	$K(\text{Jack}, \text{Tuna}) \vee K(\text{Curiosity}, \text{Tuna})$	Tiền đề
6	C(Tuna)	Tiền đề
7	$\neg C(x) \vee A(x)$	Tiền đề
8	$\neg K(\text{Curiosity}, \text{Tuna})$	Kết luận
9	$K(\text{Jack}, \text{Tuna})$	5, 8
10	$A(\text{Tuna})$	6, 7 {x/Tuna}
11	$\neg L(\text{Jack}) \vee \neg A(\text{Tuna})$	4, 9 {x/Jack, y/Tuna}
12	$\neg L(\text{Jack})$	10, 11
13	$\neg D(y) \vee \neg O(\text{Jack}, y)$	3, 12 {x/Jack}
14	$\neg D(A)$	2, 13 {y/A}
15	•	14, 1