## Chương 6

# MẠNG TÍNH TOÁN

## 6.1. MỞ ĐẦU

Mạng tính toán là một dạng biểu diễn tri thức có thể dùng biểu diễn các tri thức về các vấn đề tính toán và được áp dụng một cách có hiệu quả để giải quyết các vấn đề này. *Mỗi mạng tính toán là một mạng ngữ nghĩa chứa các biến và những quan hệ có thể cài đặt và sử dụng được cho việc tính toán*. Có thể nói rằng mạng tính toán là một sự tổng quát hoá của kiểu dữ liệu trừu tượng có khả năng tự xây dựng các hàm dùng cho việc tổng hợp thành các chương trình.

Trong chương nầy chúng ta xét một mạng tính toán gồm một tập hợp các biến cùng với một tập các quan hệ (chẳng hạn các công thức) tính toán giữa các biến. Trong ứng dụng cụ thể mỗi biến và giá trị của nó thường gắn liền với một khái niệm cụ thể về sự vật, mỗi quan hệ thể hiện một sự tri thức về sự vật.

Cách biểu diễn tri thức tính toán dưới dạng các đối tượng nầy rất tự nhiên và gần gũi đối với cách nhìn và nghĩ của con người khi giải quyết các vấn đề tính toán liên quan đến một số khái niệm về các đối tượng, chẳng hạn như các tam giác, tứ giác, hình bình hành, hình chữ nhật....

## 6.2. MẠNG TÍNH TOÁN

### 6.2.1. Các quan hệ

Cho M = {x1,x2,...,xm} là một tập hợp các biến có thể lấy giá trị trong các miền xác định tương ứng D1,D2,...,Dm. Đối với mỗi quan hệ R $\subseteq$ D1xD2x...xDm trên các tập hợp D1,D2,...,Dm ta nói rằng quan hệ này liên kết các biến x1,x2,...,xm, và ký hiệu là R(x1,x2,...,xm) hay vắn tắt là R(x) (ký hiệu x dùng để chỉ bộ biến < x1,x2,...,xm >). Quan hệ R(x) xác định một (hay một số) ánh xạ :

$$fR_{,u,v} : Du \rightarrow Dv,$$

trong đó u,v là các bộ biến và u $\subseteq$ x, v$\subseteq$ x; Du và Dv là tích của các miền xác định tương ứng của các biến trong u và trong v.

Ta có thể thấy rằng quan hệ R(x) có thể được biểu diễn bởi một ánh xạ fR,u,v với u $\cup$ v = x, và ta viết :

$$fR,u,v : u \rightarrow v,$$

hay vắn tắt là:

$$f : u \rightarrow v.$$

Đối với các quan hệ dùng cho việc tính toán, cách ký hiệu trên bao hàm ý nghĩa như là một hàm: ta có thể tính được giá trị của các biến thuộc v khi biết được giá trị của các biến thuộc u.

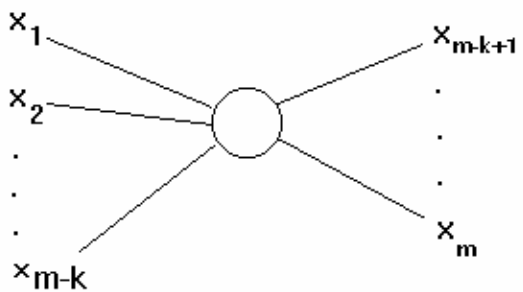
Trong phần sau ta xét các quan hệ xác định bởi các hàm có dạng:

$$f : u \rightarrow v,$$

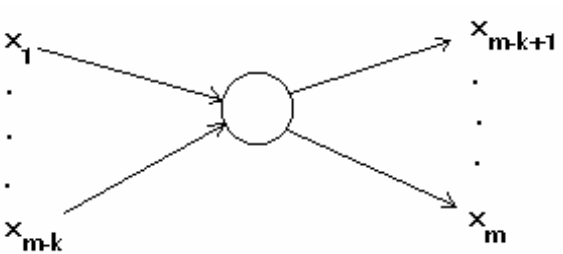
trong đó u $\cap$ v = $\varnothing$ (tập rỗng). Đặc biệt là các ***quan hệ đối xứng*** có hạng (rank) bằng một số nguyên dương k. Đó là các quan hệ mà ta có thể tính được k biến bất kỳ từ m-k biến kia (ở

đây x là bộ gồm m biến < x1,x2,...,xm >). Ngoài ra, trong trường hợp cần nói rõ ta viết u(f) thay cho u, v(f) thay cho v. Đối với các quan hệ không phải là đối xứng có hạng k, không làm mất tính tổng quát, ta có thể giả sử quan hệ xác định duy nhất một hàm f với tập biến vào là u(f) và tập biến ra là v(f); ta gọi loại quan hệ này là quan hệ không đối xứng xác định một hàm, hay gọi vắn tắt là ***quan hệ không đối xứng***.

Ta có thể vẽ hình biểu diễn cho các quan hệ đối xứng và các quan hệ không đối xứng (xác định một hàm) như trong hình 6.1 và 6.2.




**Hình 6.1**. Quan hệ đối xứng có hạng k




**Hình 6.2**. Quan hệ không đối xứng có hạng k

**Nhận xét**

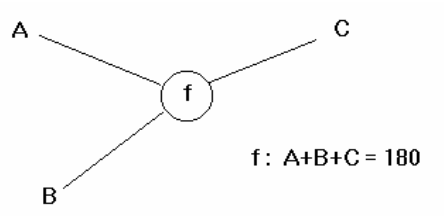
1/ Một quan hệ không đối xứng hạng k có thể được viết thành k quan hệ không đối xứng có hạng 1.

2/ Nếu biểu diễn một quan hệ đối xứng có hạng k thành các quan hệ đối xứng có hạng là 1 thì số quan hệ có hạng 1 bằng :

$$C_m^{m-k+1} = C_m^{k-1}$$

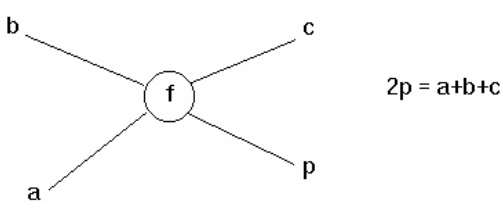
Dưới đây là một vài ví dụ về các quan hệ (tính toán) và mô hình biểu diễn tương ứng.

**Ví dụ 1**: Quan hệ f giữa ba góc A, B, C trong tam giác ABC cho bởi hệ thức: A+B+C = 180 (đơn vị: độ).




Quan hệ f giữa ba góc trong một tam giác trên đây là một quan hệ đối xứng có hạng 1.

**Ví dụ 2**: quan hệ f giữa nửa chu vi p với các độ dài của ba cạnh a, b, c:



**Ví dụ 3**: Quan hệ f giữa n biến x1, x2, ..., xn được cho dưới dạng một hệ phương trình tuyến tính có nghiệm. Trong trường hợp này f là một quan hệ có hạng k bằng hạng của ma trận hệ số của hệ phương trình.

### 6.2.2. Mạng tính toán và các ký hiệu

Như đã nói ở trên, ta sẽ xem xét các mạng tính toán bao gồm một tập hợp các biến M và một tập hợp các quan hệ (tính toán) F trên các biến. Ta gọi một mạng tính toán một cách vắn tắt là một *MTT*, và trong trường hợp tổng quát có thể viết:

$$M = \{x_1, x_2, ..., xn\},$$

$$F = \{f_1, f_2, ..., fm\}.$$

Đối với mỗi $f \in F$, ta ký hiệu M(f) là tập các biến có liên hệ trong quan hệ f. Dĩ nhiên M(f) là một tập con của M: $M(f) \subseteq M$. Nếu viết f dưới dạng:

$$f : u(f) \rightarrow v(f)$$

thì ta có $M(f) = u(f) \cup v(f)$.

## Ví dụ 4

Trong ví dụ 1 ở trên, ta có $M(f) = \{A,B,C\}$.

Trong ví dụ 2 ở trên, ta có $M(f) = \{a,b,c,p\}$.

Trong ví dụ 3 ở trên, ta có $M(f) = \{x_1, x_2, ..., xn\}$.

**Ví dụ 5** : Mạng tính toán cho một hình chữ nhật.

Việc tính toán trên một hình chữ nhật liên quan đến một số giá trị của hình chữ nhật như sau :

$b_1$, $b_2$ : hai cạnh của hình chữ nhật;

d : đường chéo của hình chữ nhật;

s : diện tích của hình chữ nhật;

p : chu vi của hình chữ nhật;

trong đó mỗi biến đều có giá trị là thuộc tập các số thực dương. Giữa các biến ta đã biết có các quan hệ sau đây:

$f_1$ : $s = b_1 * b_2$;

$f_2$ : $p = 2 * b_1 + 2 * b_2$;

$f_3$ : $d^2 = b_1^2 + b_2^2$;

Các quan hệ này đều là các quan hệ đối xứng có hạng 1.

Như vậy tập biến và tập quan hệ của mạng tính toán này là :

$M = \{b_1, b_2, d, s, p\}$,

$F = \{f_1, f_2, f_3\}$.

## 6.3. VẤN ĐỀ TRÊN MẠNG TÍNH TOÁN

Cho một mạng tính toán (M,F), M là tập các biến và F là tập các quan hệ. Giả sử có một tập biến $A \subseteq M$ đã được xác định (tức là tập gồm các biến đã biết trước giá trị), và B là một tập biến bất kỳ trong M.

Các vấn đề đặt ra là

1. Có thể xác định được tập B từ tập A nhờ các quan hệ trong F hay không? Nói cách khác, ta có thể tính được giá trị của các biến thuộc B với giả thiết đã biết giá trị của các biến thuộc A hay không?

2. Nếu có thể xác định được B từ A thì quá trình tính toán giá trị của các biến thuộc B như thế nào?

3. Trong trường hợp không thể xác định được B, thì cần cho thêm điều kiện gì để có thể xác định được B.

Bài toán xác định B từ A trên mạng tính toán (M,F) được viết dưới dạng:

$$A \rightarrow B,$$

trong đó A được gọi là giả thiết, B được gọi là mục tiêu tính toán (hay tập biến cần tính) của vấn đề. Trường hợp tập B chỉ gồm có một phần tử b, ta viết vắn tắt bài toán trên là $A \rightarrow b$.

## Định nghĩa 2.1

Bài toán $A \rightarrow B$ được gọi là ***giải được*** khi có thể tính toán được giá trị các biến thuộc B xuất phát từ giả thiết A. Ta nói rằng một dãy các quan hệ $\{f_1, f_2, ..., fk\} \subseteq F$ là một ***lời giải*** của bài toán $A \rightarrow B$ nếu như ta lần lượt áp dụng các quan hệ fi (i=1,...,k) xuất phát từ giả thiết A thì sẽ tính được các biến thuộc B. Lời giải $\{f_1, f_2, ..., fk\}$ được gọi là ***lời giải tốt*** nếu không thể bỏ bớt một số bước tính toán trong quá trình giải, tức là không thể bỏ bớt một số quan hệ trong lời giải. Lời giải được gọi là ***lời giải tối ưu*** khi nó có số bước tính toán ít nhất, tức là số quan hệ áp dụng trong tính toán là ít nhất.

Việc tìm lời giải cho bài toán là việc tìm ra một dãy quan hệ để có thể áp dụng tính ra được B từ A. Điều nầy cũng có nghĩa là tìm ra được một quá trình tính toán để giải bài toán.

Trong quá trình tìm lời giải cho bài toán chúng ta cần xét một dãy quan hệ nào đó xem có thể tính thêm được các biến từ

một tập biến cho trước nhờ dãy quan hệ nầy hay không. Do đó chúng ta đưa thêm định nghĩa sau đây.

**Định nghĩa 2.2**

Cho D = $\{f_1, f_2, ..., fk\}$ là một dãy quan hệ của mạng tính toán (M,F), A là một tập con của M. Ta nói dãy quan hệ D là ***áp dụng được*** trên tập A khi và chỉ khi ta có thể lần lượt áp dụng được các quan hệ $f_1, f_2, ..., fk$ xuất phát từ giả thiết A.

***Nhận xét*** : Trong định nghĩa trên, nếu đặt : $A_0 = A$, $A_1 = A_0 \cup M(f_1)$, . . . , $Ak = Ak_{-1} \cup M(fk)$, và ký hiệu Ak là **D(A)**, thì ta có D là một lời giải của bài toán $A \rightarrow D(A)$. Trong trường hợp D là một dãy quan hệ bất kỳ (không nhất thiết là áp dụng được trên A), ta vẫn ký hiệu D(A) là tập biến đạt được khi lần lượt áp dụng các quan hệ trong dãy D (nếu được). Chúng ta có thể nói rằng D(A) là sự mở rộng của tập A nhờ áp dụng dãy quan hệ D.

**Thuật toán tính D(A)**

*Nhập :* Mạng tính toán (M,F),

$A \subseteq M$,

dãy các quan hệ $D = \{f_1, f_2, ..., fm\}$.

*Xuất :* D(A).

*Thuật toán :*

1. $A' \leftarrow A$;
2. for i=1 to m do
   if fi áp dụng được trên A' then
   $A' \leftarrow A' \cup M(fi)$;
3. $D(A) \leftarrow A'$

## 6.4. GIẢI QUYẾT VẤN ĐỀ

### 6.4.1. Tính giải được của bài toán

Trong mục này chúng ta nêu lên một khái niệm có liên quan đến tính giải được của vấn đề trên một mạng tính toán : bao đóng của một tập hợp biến trên một mạng tính toán.

### Định nghĩa 3.1

Cho mạng tính toán (M,F), và A là một tập con của M. Ta có thể thấy rằng có duy nhất một tập hợp B lớn nhất $\subseteq$ M sao cho bài toán $A \rightarrow B$ là giải được, và tập hợp B được gọi là ***bao đóng*** của A trên mô hình (M,F). Một cách trực quan, có thể nói bao đóng của A là sự mở rộng tối đa của A trên mô hình (M,F). Ký hiệu bao đóng của A là $\overline{A}$, chúng ta có kiểm tra dễ dàng các tính chất liên quan đến bao đóng trong mệnh đề dưới đây.

***Mệnh đề 3.1:*** Cho A và B là hai tập con của M. Ta có:

(1) $\overline{A} \supseteq A$.

(2) $\overline{\overline{A}} = \overline{A}$.

(3) $A \subseteq B \Rightarrow \overline{A} \subseteq \overline{B}$

(4) $\overline{A \cap B} \subseteq \overline{A} \cap \overline{B}$

(5) $\overline{A \cup B} \supseteq \overline{A} \cup \overline{B}$

Đối với tính giải được của bài toán, ta có thể dễ dàng kiểm chứng mệnh đề sau:

***Mệnh đề 3.2***

(1) Bài toán $A \to B$ là giải được khi và chỉ khi các bài toán $A \to b$ là giải được với mọi $b \in B$.

(2) Nếu $A \to B$ và $B \to C$ là các bài toán giải được thì bài toán $A \to C$ cũng giải được. Hơn nữa, nếu $\{f_1, f_2, ..., fm\}$ và $\{g_1, g_2, ..., gp\}$ lần lượt là lời giải của bài toán $A \to B$ và bài toán $B \to C$ thì $\{f_1, f_2, ..., fm, g_1, g_2, ..., gp\}$ là một lời giải của bài toán $A \to C$.

(3) Nếu bài toán $A \to B$ là giải được và B' là một tập con của B thì $A \to B'$ cũng là một bài toán giải được. Hơn nữa, nếu $\{f_1, f_2, ..., fm\}$ là một lời giải của bài toán $A \to B$ thì đó cũng là một lời giải của bài toán $A \to B'$.

Từ khái niệm bao đóng đã nói ở trên ta cũng có các định lý sau:

**Định lý 3.1**. Trên một mạng tính toán (M,F), bài toán $A \to B$ là giải được khi và chỉ khi $B \subseteq \overline{A}$

Từ định lý này, ta có thể kiểm tra tính giải được của bài toán $A \to B$ bằng cách tính bao đóng của tập A rồi xét xem B có bao hàm trong $\overline{A}$ hay không.

***Mệnh đề 3.3***: Cho một dãy quan hệ $D = \{f_1, f_2, ..., fk\} \subseteq F$, $A \subseteq M$. Đặt :

$A_0 = A$, $A_1 = A_0 \cup M(f_1)$, ..., $Ak = Ak_{-1} \cup M(fk)$. Ta có các điều sau đây là tương đương :

(1) Dãy D áp được trên A.

(2) Với mọi i=1, ..., k ta có:

Card (M(fi) \ $Ai_{-1}$) ≤ r(fi) nếu fi là quan hệ đối xứng,

M(fi) \ $Ai_{-1}$ ⊆ v(fi) nếu fi là quan hệ không đối xứng.

(ký hiệu Card (X) chỉ số phần tử của tập X).

**Ghi chú** : Dựa vào mệnh đề 3.3 ta có một thuật toán để kiểm tra tính áp dụng được của một dãy quan hệ D trên một tập biến A.

**Định lý 3.2.** Cho một mạng tính toán (M,F), A, B là hai tập con của M. Ta có các điều sau đây là tương đương:

(1) $B \subseteq \overline{A}$.

(2) Có một dãy quan hệ D = {$f_1$, $f_2$, ..., fk} ⊆ F thỏa các điều kiện :

(a) D áp được trên A.

(b) D(A) ⊇ B.

*Chứng minh* : Giả sử có (1), tức là $B \subseteq \overline{A}$. Khi đó bài toán A → B là giải được. Do đó có một dãy quan hệ {$f_1$, $f_2$, ..., fk} ⊆ F sao cho khi ta lần lượt áp dụng các quan hệ fi (i=1,...,k) xuất phát từ giả thiết A thì sẽ tính được các biến thuộc B. Dễ dàng thấy rằng dãy {$f_1$, $f_2$, ..., fk} nầy thỏa các điều kiện (2).

Đảo lại, giả sử có (2). Với các điều kiện có được bởi (2) ta thấy {fi} là lời giải của vấn đề $Ai_{-1}$ → Ai, với mọi i = 1,2, ..., k. Từ mệnh đề 3.2 suy ra bài toán $A_0$ → Ak là giải được. Do đó bài toán A → B cũng giải được, suy ra $B \subseteq \overline{A}$ theo định lý 3.1.

**Nhận xét**

1. Dãy quan hệ {$f_1$, $f_2$, ..., fk} trong định lý trên là một lời giải của vấn đề A → B trên mạng tính toán (M,F).

2. Trong lời giải $\{f_1, f_2, ..., fk\}$ ta có thể bỏ bớt những fi nào mà $M(fi) \subseteq Di_{-1}(A)$, với $Di_{-1} = \{f_1, f_2, ..., fi_{-1}\}$.

Qua các định lý trên, ta thấy rằng việc xác định bao đóng của một tập biến trên mô hình tính toán là cần thiết. Dưới đây là thuật toán cho phép xác định bao đóng của tập hợp $A \subseteq M$. Trong thuật toán này chúng ta thử áp dụng các quan hệ $f \in F$ để tìm dần những biến thuộc M có thể tính được từ A; cuối cùng sẽ được bao đóng của A.

**Thuật toán 3.1.** Tìm bao đóng của tập $A \subseteq M$ :

*Nhập :* Mạng tính toán (M,F),
$A \subseteq M$.

*Xuất :* $\overline{A}$

*Thuật toán :*

1. $B \leftarrow A$;
2. **Repeat**
   $B1 \leftarrow B$;
   **for** $f \in F$ **do**
   **if** ( f đối xứng **and** $Card(M(f) \setminus B) \leq r(f)$) **or** ( f không đối xứng **and** $M(f) \setminus B \subseteq v(f)$ ) **then**
   **begin**
   $B \leftarrow B \cup M(f)$;
   $F \leftarrow F \setminus \{f\}$; // *loại f khỏi lần xem xét sau*
   **end;**
   **Until** $B = B1$;
3. $\overline{A} \leftarrow B$;

**Ghi chú** : Trên đây ta đã nêu lên đặc trưng cho tính giải được của bài toán trên một mạng tính toán và chỉ ra thuật toán

để kiểm tra khi nào bài toán là giải được. Ngoài ra chúng ta sẽ còn nêu lên cách để kiểm định giả thiết của bài toán; và trong trường hợp bài toán chưa đủ giả thiết có thể bổ sung thêm nếu được.

### 6.4.2. Lời giải của bài toán

Ở trên ta đã nêu lên cách xác định tính giải được của bài toán. Tiếp theo, ta sẽ trình bày cách tìm ra lời giải cho bài toán $A \rightarrow B$ trên mạng tính toán (M,F). Trước hết từ nhận xét sau định lý 3.2 ta có mệnh đề sau đây:

**Mệnh đề 3.4:** Dãy quan hệ D là một lời giải của bài toán $A \rightarrow B$ khi và chỉ khi D áp dụng được trên A và $D(A) \supseteq B$.

Do mệnh đề trên, để tìm một lời giải ta có thể làm như sau: Xuất phát từ giả thiết A, ta thử áp dụng các quan hệ để mở rộng dần tập các biến có giá trị được xác định; và quá trình này tạo ra một sự lan truyền tính xác định trên tập các biến cho đến khi đạt đến tập biến B. Dưới đây là thuật toán tìm một lời giải cho bài toán $A \rightarrow B$ trên mạng tính toán (M,F).

**Thuật toán 3.2:** Tìm một lời giải cho bài toán $A \rightarrow B$:

*Nhập :* Mạng tính toán (M,F),
tập giả thiết $A \subseteq M$,
tập biến cần tính $B \subseteq M$.

*Xuất :* lời giải cho bài toán $A \rightarrow B$

*Thuật toán :*

1. Solution $\leftarrow$ empty; // Solution *là dãy các quan hệ sẽ //áp dụng*

2. **if** $B \subseteq A$ **then**
**begin**
Solution_found ← true; // *biến* Solution_found =
// true *khi bài toán là giải được*
**goto** 4;
**end**
**else**
Solution_found ← false;
3. **Repeat**
Aold ← A;
Chọn ra một $f \in F$ chưa xem xét;
**while not** Solution_found **and** (chọn được f) **do**
**begin**
**if** ( f đối xứng **and** $0 < \text{Card}(M(f) \setminus A) \leq r(f)$ ) **or**(
f không đối xứng **and** $\varnothing \neq M(f) \setminus A \subseteq v(f)$ )
**then**
**begin**
$A \leftarrow A \cup M(f)$;
Solution ← Solution ∪ {f};
**end;**
**if** $B \subseteq A$ **then**
Solution_found ← true;
Chọn ra một $f \in F$ chưa xem xét;
**end;** { while }
**Until** Solution_found **or** (A = Aold);

4. **if not** Solution_found **then**
Bài toán không có lời giải;
**else**
Solution là một lời giải;

**Ghi chú**

1. Về sau, khi cần trình bày quá trình giải (hay bài giải) ta có thể xuất phát từ lời giải tìm được dưới dạng một dãy các quan hệ để xây dựng bài giải.
2. Lời giải (nếu có) tìm được trong thuật toán trên chưa chắc là một lời giải tốt. Ta có thể bổ sung thêm cho thuật toán ở trên thuật toán để tìm một lời giải tốt từ một lời giải đã biết nhưng chưa chắc là tốt. Thuật toán sẽ dựa trên định lý được trình bày tiếp theo đây.

**Định lý 3.3**. Cho $D = \{f_1, f_2, ..., f_m\}$ là một lời giải của bài toán $A \rightarrow B$. ứng với mỗi $i = 1,...,m$ đặt $D_i = \{f_1, f_2, ..., f_i\}$, $D_0 = \varnothing$. Ta xây dựng một họ các dãy con $S_m, S_{m-1}, ..., S_2, S_1$ của dãy D như sau :

$S_m = \varnothing$ nếu $D_{m-1}$ là một lời giải,
$S_m = \{f_m\}$ nếu $D_{m-1}$ không là một lời giải,
$S_i = S_{i+1}$ nếu $D_{i-1} \cup S_{i+1}$ là một lời giải,
$S_i = \{f_i\} \cup S_{i+1}$ nếu $D_{i-1} \cup S_{i+1}$ không là một lời giải,

với mọi $i = m - 1, m - 2, ..., 2, 1$.

Khi đó ta có:

(1) $S_m \subseteq S_{m-1} \subseteq ... \subseteq S_2 \subseteq S_1$.

(2) $D_{i-1} \cup S_i$ là một lời giải của bài toán $A \rightarrow B$ với mọi $i = m, ..., 2, 1$.

(3) Nếu $S'_i$ là một dãy con thật sự của $S_i$ thì $D_{i-1} \cup S'_i$ không phải là một lời giải của bài toán $A \rightarrow B$ với mọi i.

(4) $S_1$ là một lời giải tốt của bài toán $A \rightarrow B$.

Từ định lý 3.3 trên ta có một thuật toán tìm lời giải tốt từ một lời giải đã biết sau đây:

**Thuật toán 3.3.** tìm một lời giải tốt từ một lời giải đã biết.

*Nhập :* Mạng tính toán (M,F),
lời giải $\{f_1, f_2, ..., f_m\}$ của bài toán $A \rightarrow B$.

*Xuất :* lời giải tốt cho bài toán $A \rightarrow B$

*Thuật toán :*

1. $D \leftarrow \{f_1, f_2, ..., f_m\}$;
2. **for** i=m **downto** 1 **do**
   **if** $D \setminus \{fi\}$ là một lời giải **then**
   $D \leftarrow D \setminus \{f_i\}$;
3. D là một lời giải tốt.

Trong thuật toán 3.3 có sử dụng việc kiểm tra một dãy quan hệ có phải là lời giải hay không. Việc kiểm tra này có thể được thực hiện nhờ thuật toán sau đây:

**Thuật toán kiểm tra lời giải cho bài toán**

*Nhập :* Mạng tính toán (M,F),
bài toán $A \rightarrow B$,
dãy các quan hệ $\{f_1, f_2, ..., f_m\}$.

*Xuất :* thông tin cho biết $\{f_1, f_2, ..., f_m\}$ có phải là lời giải của bài toán $A \rightarrow B$ hay không.

*Thuật toán :*

1.**for** i=1 **to** m **do**
**if** ( $f_i$ đối xứng **and** Card $(M(f_i) \setminus A) \leq r(f_i)$ ) **or**
( fi không đối xứng **and** $M(f_i) \setminus A \subseteq v(f_i)$ ) **then**
$A \leftarrow A \cup M(f_i)$;

2.**if** $A \supseteq B$ **then**
$\{f_1, f_2, ..., f_m\}$ là lời giải

**else**

$\{f_1, f_2, ..., f_m\}$ không là lời giải;

Ở trên ta đã có một thuật toán tổng quát để tìm lời giải tốt cho bài toán khi đã biết trước một lời giải. Tuy nhiên trong ứng dụng cụ thể ta thường gặp các quan hệ đối xứng có hạng một hơn là các quan hệ đối xứng có hạng lớn hơn 1. Trong trường hợp như thế ta có thể áp dụng một thuật toán khác để tìm một lời giải tốt từ một lời giải biết trước với mức độ tính toán ít hơn. Theo thuật toán nầy, ta lần lượt xem xét các quan hệ trong tập lời giải đã biết và chọn ra các quan hệ để đưa vào một lời giải mới sao cho trong lời giải mới này không thể bớt ra bất kỳ một quan hệ nào.

**Thuật toán 3.4.** Tìm một lời giải tốt từ một lời giải đã biết không chứa quan hệ đối xứng hạng > 1.

*Nhập :* Mạng tính toán (M,F),

Lời giải $\{f_1, f_2, ..., f_m\}$ của bài toán $A \rightarrow B$,

*Điều kiện* : $f_i$ không phải là quan hệ đối xứng có hạng lớn hơn 1.

*Xuất :* lời giải tốt cho bài toán $A \rightarrow B$

*Thuật toán :*

1. NewSolution $\leftarrow \varnothing$; // *đầu tiên tập lời giải mới*

// *chưa có quan hệ nào.*

$A_0 \leftarrow A$;

**for** i=1 **to** m **do** $A_i = A_{i-1} \cup M(f_i)$;

2. // *Dò theo chỉ số i từ 0 tìm i đầu tiên sao cho* $A_i \supseteq B$.

$i \leftarrow 0$;

**while not** $(A_i \supseteq B)$ **do**

i ← i + 1;

3. **if** i = 0 **then** **goto** 8;
4. m ← i;
5. // *Ghi nhận fm trong lời giải mới*.
   NewSolution ← { fm } ∪ NewSolution;
6. // *Dò theo chỉ số i từ 1 đến m – 1 tìm i đầu tiên (nếu có)* // *sao cho ta có thể áp dụng fm trên Ai để tính ra được* // *B.*
   i_found ← false;
   i ← 1;
   **while not** i_found **and** (i ≤ m –1) **do**
   **if** (($f_m$ đối xứng **and** Card ($M(f_m) \setminus Ai$) ≤ $r(f_m)$)
   **or** ($f_m$ không đối xứng **and** $M(f_m) \setminus Ai \subseteq v(f_m)$ )
   **and** ($B \subseteq M(f_m) \cup Ai$) **then**
   i_found ← true
   **else**
   i ← i + 1;
7. **if** i_found **then**
   **begin**
   B ← $(B \cup M(f_m)) \cap Ai$;
   **goto** 2;
   **end**;
8. NewSolution là một lời giải tốt của bài toán A → B.

**Ví dụ :** Bây giờ ta xét một ví dụ cụ thể để minh họa cho các thuật toán trên.

Cho tam giác ABC có cạnh a và hai góc kề là β, γ được cho trước.

Tính diện tích S của tam giác.

Để tìm ra lời giải cho bài toán trước hết ta xét mạng tính toán của tam giác. Mạng tính toán này gồm :

1. Tập biến M = {a, b, c, $\alpha$, $\beta$, $\gamma$, ha, hb, hc, S, p, R, r, ...},
trong đó a, b, c là ba cạnh; $\alpha$, $\beta$, $\gamma$ là ba góc tương ứng với ba cạnh; ha, hb, hc là ba đường cao; S là diện tích tam giác; p là nửa chu vi; R là bán kính đường tròn ngoại tiếp tam giác; r là bán kính đường tròn nội tiếp tam giác...

2. Các quan hệ

$f_1$ : $\alpha + \beta + \gamma = 180$

$f_2$ : $\frac{a}{\sin\alpha} = \frac{b}{\sin\beta}$

$f_3$ : $\frac{c}{\sin\gamma} = \frac{b}{\sin\beta}$

$f_4$ : $\frac{a}{\sin\alpha} = \frac{c}{\sin\gamma}$

$f_5$ : $p = (a+b+c)/2$

$f_6$ : $S = a.ha / 2$

$f_7$ : $S = b.hb / 2$

$f_8$ : $S = c.hc / 2$

$f_9$ : $S = a.b.\sin\gamma / 2$

$f_{10}$ : $S = b.c.\sin\alpha / 2$

$f_{11}$ : $S = c.a.\sin\beta / 2$

$f_{12}$ : $S = \sqrt{p(p-a)(p-b)(p-c)}$

v.v ...

3. Yêu cầu tính : S (diện tích của tam giác).

Theo đề bài ta có giả thiết là : A = {a, $\beta$, $\gamma$}, và tập biến cần tính là B = {S}.

Áp dụng thuật toán tìm lời giải (thuật toán 3.2) ta có một lời giải cho bài tính là dãy quan hệ sau:

$\{f_1, f_2, f_3, f_5, f_9\}$.

Xuất phát từ tập biến A, lần lượt áp dụng các quan hệ trong lời giải ta có tập các biến được xác định mở rộng dần đến khi S được xác định :

$\{a, \beta, \gamma\} \xrightarrow{f_1} \{a, \beta, \gamma, \alpha\} \xrightarrow{f_2} \{a, \beta, \gamma, \alpha, b\} \xrightarrow{f_3}$
$\{a, \beta, \gamma, \alpha, b, c\} \xrightarrow{f_5} \{a, \beta, \gamma, \alpha, b, c, p\} \xrightarrow{f_9} \{a, \beta, \gamma, \alpha, b, c, p, S\}$.

Có thể nhận thấy rằng lời giải này không phải là lời giải tốt vì có bước tính toán thừa, chẳng hạn là $f_5$. Thuật toán 3.3 hay thuật toán 3.4 sẽ lọc ra từ lời giải trên một lời giải tốt là $\{f_1, f_2, f_9\}$:

$\{a, \beta, \gamma\} \xrightarrow{f_1} \{a, \beta, \gamma, \alpha\} \xrightarrow{f_2} \{a, \beta, \gamma, \alpha, b\} \xrightarrow{f_9}$
$\{a, \beta, \gamma, \alpha, b, S\}$.

Theo lời giải này, ta có quá trình tính toán như sau :

bước 1: tính $\alpha$ (áp dụng $f_1$).

bước 2: tính b (áp dụng $f_2$).

bước 3: tính S (áp dụng $f_9$).

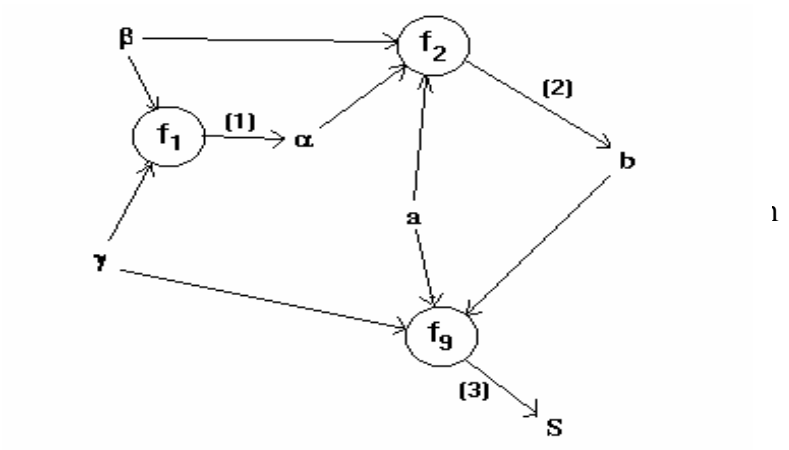
Quá trình tính toán (gồm ba bước) này có thể được diễn đạt một cách rõ ràng trên sơ đồ mạng hình 6.3.

### 6.4.3. Lời giải tối ưu của bài toán

Liên quan đến lời giải tối ưu cho bài toán, ta có thể dễ dàng chứng minh mệnh đề dưới đây dựa vào tính thứ tự tốt của tập hợp các số tự nhiên.

**Mệnh đề 3.3**. Nếu bài toán A → B là giải được thì sẽ tồn tại một lời giải tối ưu cho bài toán.

Ngoài ra, ta có thể áp dụng thuật toán $A^*$ (thuật toán heuristic) để tìm ra một lời giải tối ưu trong trường hợp bài toán là giải được.




**Hình 6.3** Sơ đồ thể hiện một mạng tính toán

### 6.4.4. Kiểm định giả thiết cho bài toán

Xét bài toán A → B trên mạng tính toán (M,F). Trong mục này chúng ta xem xét giả thiết A của bài toán xem thừa hay thiếu, và trong trường hợp cần thiết ta tìm cách điều chỉnh giả thiết A.

Trước hết ta cần xét xem bài toán có giải được hay không. Trường hợp bài toán giải được thì giả thiết là đủ. Tuy nhiên có thể xảy ra tình trạng thừa giả thiết. Để biết được bài toán có thật sự thừa giả thiết hay không, ta có thể dựa vào thuật toán tìm sự thu gọn giả thiết sau đây:

**Thuật toán 3.5.** Tìm một sự thu gọn giả thiết của bài toán.

*Nhập :* Mạng tính toán (M,F),

Bài toán $A \rightarrow B$ giải được,

*Xuất :* Tập giả thiết mới $A' \subseteq A$ tối tiểu theo thứ tự $\subseteq$.

*Thuật toán :*

**Repeat**

$A' \leftarrow A$;

**for** $x \in A$ **do**

**if** $A \setminus \{x\} \rightarrow B$ giải được **then**

$A \leftarrow A \setminus \{x\}$;

**Until** $A = A'$;

**Ghi chú** : Trong thuật toán trên nếu tập giả thiết mới A' thật sự bao hàm trong A thì bài toán bị thừa giả thiết và ta có thể bớt ra từ giả thiết A tập hợp các biến không thuộc A', coi như là giả thiết cho thừa.

Trường hợp bài toán $A \rightarrow B$ là không giải được thì ta nói giả thiết A thiếu. Khi đó có thể điều chỉnh bài toán bằng nhiều cách khác nhau để cho bài toán là giải được. Chẳng hạn ta có thể sử dụng một số phương án sau đây:

**Phương án 1** : Tìm một A' $\subseteq$ M \ ($\overline{A} \cup$ B) tối tiểu sao cho bao đóng của tập hợp A'$\cup$ A chứa B.

**Phương án 2** : Khi phương án 1 không thể thực hiện được thì ta không thể chỉ điều chỉnh giả thiết để cho bài toán là giải được. Trong tình huống này, ta phải bỏ bớt kết luận hoặc chuyển bớt một phần kết luận sang giả thiết để xem xét lại bài toán theo phương án 1.

### 6.4.5. Định lý về sự phân tích quá trình giải

Xét bài toán A $\rightarrow$ B trên mạng tính toán (M,F). Trong các mục trên chúng ta đã trình bày một số phương pháp để xác định tính giải được của bài toán, tìm ra một lời giải tốt cho bài toán.

Trong mục nầy ta nêu lên một cách xây dựng quá trình giải từ một lời giải đã biết. Đối với một lời giải, rất có khả năng một quan hệ nào đó dẫn tới việc tính toán một số biến thừa, tức là các biến tính ra mà không có sử dụng cho các bước tính phía sau. Do đó, chúng ta cần xem xét quá trình áp dụng các quan hệ trong lời giải và chỉ tính toán các biến thật sự cần thiết cho quá trình giải theo lời giải. Định lý sau đây cho ta một sự phân tích tập các biến được xác định theo lời giải và trên cơ sở đó có thể xây dựng quá trình tính toán các biến để giải quyết bài toán.

**Định lý 3.4**. Cho $\{f_1, f_2, ..., f_m\}$ là một lời giải tốt cho bài toán A $\rightarrow$ B trên một mạng tính toán (M,F). Đặt :

$A_0 = A$, $A_i = \{f_1, f_2, ..., f_i\}(A)$, với mọi i = 1,...,m.

Khi đó có một dãy $\{B_0, B_1, ..., B_{m-1}, B_m\}$, thỏa các điều kiện sau đây:

(1) $B_m = B$.

(2) $B_i \subseteq A_i$ , với mọi i = 0,1,...,m.

(3) Với mọi i = 1,...,m, {fi} là lời giải của bài toán $B_{i-1} \rightarrow B_i$ nhưng không phải là lời giải của bài toán $G \rightarrow B_i$ , trong đó G là một tập con thật sự tùy ý của $B_{i-1}$.

*Chứng minh* : Ta xây dựng dãy $\{B_0, B_1, ..., B_{m-1}, B_m\}$ bằng cách đặt: B = m = B, và ứng với mỗi i < m, đặt:

$$B_i = (B_{i+1} \cap A_i) \cup A_i',$$

với $A_i'$ là tập có ít phần tử nhất trong $A_i \setminus B_{i+1}$ sao cho $f_{i+1}$ áp dụng được trên tập hợp $(B_{i+1} \cap A_i) \cup A_i'$. Thật ra, $A_i'$ có được xác định như sau:

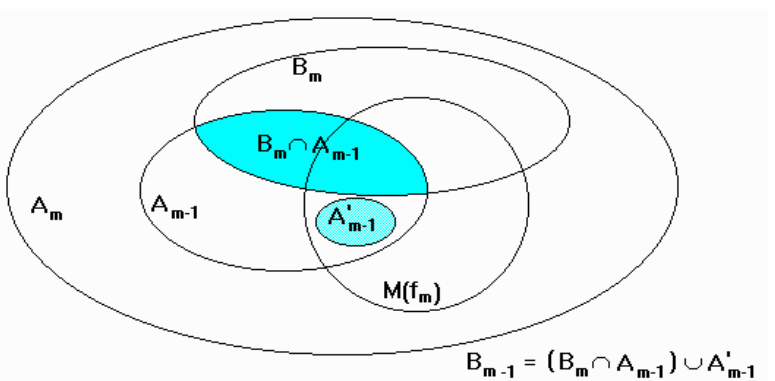
$A_i' = u(f_{i+1}) \setminus B_{i+1}$ nếu $f_{i+1}$ không đối xứng,

và nếu $f_{i+1}$ đối xứng thì

$A_i'$ = một tập con gồm $\max(0,t_i)$ phần tử của tập hợp $(M(f_{i+1})\setminus B_{i+1}) \cap A_i$

trong đó $t_i = card(M(f_{i+1})) - r(f_{i+1}) - card(M(f_{i+1}) \cap B_{i+1} \cap A_i)$.

Với cách xây dựng này ta có thể kiểm tra được rằng dãy $\{B_0, B_1, ..., B_{m-1}, B_m\}$ thỏa mãn các điều kiện ghi trong định lý.□



**Ghi chú**

(**1**) Từ định lý trên ta có quá trình tính toán các biến để giải bài toán $A \rightarrow B$ như sau:

bước 1: tính các biến trong tập $B_1 \setminus B_0$ (áp dụng $f_1$).

bước 2: tính các biến trong tập $B_2 \setminus B_1$ (áp dụng $f_2$).

v.v...

bước m: tính các biến trong tập $B_m \setminus B_{m-1}$ (áp dụng $f_m$).

(**2**) Từ chứng minh của định lý trên, ta có thể ghi ra một thuật toán để xây dựng dãy các tập biến $\{B_1', ..., B_{m-1}', B_m'\}$ rời nhau cần lần lượt tính toán trong quá trình giải bài toán ($B_i' = B_i \setminus B_{i-1}$) gồm các bước chính như sau:

- xác định các tập $A_0, A_1, ..., A_m$ .
- xác định các tập $B_m, B_{m-1}, ..., B_1, B_0$ .
- xác định các tập $B_1', B_2', ..., B_m'$ .

## Ví dụ

Giả sử trên một mạng tính toán ta có $\{f_1, f_2, f_3\}$ là một lời giải tốt cho bài toán $A \rightarrow B$, trong đó :

$A = \{a_1, b_1, b_2\}$,
$B = \{b_3\}$,

$f_1$, $f_2$ là các quan hệ đối xứng có hạng 2, $f_3$ là quan hệ đối xứng có hạng 1, và

$M(f_1) = \{a_1, b_1, c_1, d_1\}$,
$M(f_2) = \{a_1, c_1, b_2, d_2, e_2\}$,
$M(f_3) = \{b_1, e_2, b_3\}$.

Dựa theo sự phân tích quá trình giải trong định lý trên ta có :

$A_0 = A$,
$A_1 = \{a_1, b_1, b_2, c_1, d_1\}$,
$A_2 = \{a_1, b_1, b_2, c_1, d_1, d_2, e_2\}$,
$A_3 = \{a_1, b_1, b_2, c_1, d_1, d_2, e_2, b_3\}$,
$B_3 = B$,
$B_2 = \{b_1, e_2\}$,
$B_1 = \{b_1, a_1, c_1, b_2\}$,
$B_0 = \{a_1, b_1, b_2\}$,

và các tập biến cần lần lượt tính toán trong bài giải cho bài toán là :

$B_1' = \{c_1\}$,
$B_2' = \{e_2\}$,
$B_3' = \{b_3\}$.

Từ đó ta có quá trình tính toán theo lời giải trên như sau:

tính $c_1$ (áp dụng $f_1$),
tính $e_2$ (áp dụng $f_2$),
tính $b_3$ (áp dụng $f_3$).

## 6.5. ỨNG DỤNG TRONG CÁC PHẢN ỨNG HÓA HỌC

Ngoài ứng dụng đã nêu trong các ví dụ ở các mục trên về việc giải các bài toán về tam giác và tứ giác, mạng tính toán có

thể được áp dụng trong việc biểu diễn và giải một số bài toán trên các phản ứng hóa học. Chúng ta biết rằng trong hóa học, việc xem xét các phản ứng hóa học là một trong những vấn đề quan trọng. Về mặt tri thức người ta đã biết được nhiều chất và các phản ứng hóa học có thể chuyển hóa từ một số chất này thành các chất khác. Tạm thời bỏ qua một số điều kiện phản ứng, ta có thể xem tri thức đó như một mạng tính toán mà mỗi phản ứng là một quan hệ của mạng. Ví dụ như phản ứng điều chế clo từ axít clohidric và đioxit mangan :

$MnO_2 + HCl \rightarrow MnCl_2 + Cl_2\uparrow + H_2O$

Phản ứng trên có thể được xem như một quan hệ cho chúng ta có được các chất $Cl_2$, $MnCl_2$, $H_2O$ từ các chất $MnO_2$, HCl.

Trong mục này ta dùng mạng tính toán để giải hai bài toán sau:

**1.** Cho một số chất, hỏi có điều chế được một vài chất nào đó không?

**2.** Tìm các phương trình phản ứng để biểu diễn dãy các biến hóa, chẳng hạn như các dãy :

$$Zn \rightarrow ZnO \rightarrow ZnSO_4$$

$$S \rightarrow SO_2 \rightarrow SO_3 \rightarrow H_2SO_4$$

Kiến thức ta cần có đối với 2 bài toán trên là tập hợp tất cả các chất cùng với các phản ứng hóa học được phân loại thành các nhóm phản ứng khác nhau. Dưới đây chúng ta liệt kê một số nhóm phản ứng hóa học được lưu trữ trong phần kiến thức của chương trình.

❖ **Một số phản ứng liên quan đến khí clo**

1/ Natri nóng chảy cháy trong clo cho phản ứng tạo thành natri clorua:

$Na + Cl_2 \rightarrow NaCl$

2/ Bột sắt nóng chảy trong clo cho phản ứng:

$Fe + Cl_2 \rightarrow FeCl_3$

3/ Nung đỏ dây đồng cho vào khí clo, ta có phản ứng:

$Cu + Cl_2 \rightarrow CuCl_2$

4/ Clo tác dụng với nước:

$Cl_2 + H_2O \rightarrow HCl + HclO$

5/ Điều chế clo từ axít clohidric và đioxit mangan:

$MnO_2 + 4HCl = MnCl_2 + Cl_2\uparrow + 2H_2O$

6/ Điều chế clo bằng axit clohidric và Kali pemanganat:

$HCl + KMnO_4 \rightarrow KCl + MnCl_2 + H_2O + Cl_2\uparrow$

7/ Điện phân dung dịch đậm đặc muối ăn trong nước:

$NaCl + H_2O \rightarrow Cl_2\uparrow + H_2\uparrow + NaOH$

❖ **Một số phản ứng khác**

8/ $K + Cl_2 \rightarrow KCl$

9/ $Na + Cl_2 \rightarrow NaCl$

10/ $Al + Cl_2 \rightarrow AlCl_3$

11/ $Ca + Cl_2 \rightarrow CaCl_2$

12/ $FeCl_2 + Cl_2 \rightarrow FeCl_3$

13/ $Cl_2 + KBr \rightarrow KCl + Br_2$

14/ $Cl_2 + NaBr \rightarrow NaCl + Br_2$

❖ **Các phản ứng liên quan đến hidro clorua HCl**

15/ Cho natri clorua tinh thể tác dụng với axit sunfuric đậm đặc, đun nóng (phương pháp sunfat), tùy theo nhiệt độ ta có các phản ứng sau đây :

$NaCl + H_2SO_4 \xrightarrow{t^o} NaHSO_4 + HCl\uparrow$

$NaCl + H_2SO_4 \xrightarrow{t^o} Na_2SO_4 + HCl\uparrow$

16/ Phản ứng điều chế HCl bằng phương pháp tổng hợp :

$H_2 + Cl_2 \rightarrow HCl$

❖ **Một số phản ứng khác**

17/ $KCl + H_2SO_4 \rightarrow KHSO_4 + HCl$

18/ $KCl + H_2SO_4 \rightarrow K_2SO_4 + HCl$

19/ $CaCl_2 + H_2SO_4 \rightarrow Ca(HSO_4)_2 + HCl$

20/ $CaCl_2 + H_2SO_4 \rightarrow CaSO_4 + HCl$

❖ **Các phản ứng của axit clohidric và muối clorua**

21/ $Zn + HCl \rightarrow ZnCl_2 + H_2\uparrow$

22/ $Fe + HCl \rightarrow FeCl_2 + H_2\uparrow$

23/ $Al + HCl \rightarrow AlCl_3 + H_2\uparrow$

24/ $Ca + HCl \rightarrow CaCl_2 + H_2\uparrow$

25/ $Na + HCl \rightarrow NaCl + H_2\uparrow$

26/ $K + HCl \rightarrow KCl + H_2\uparrow$

27/ $Mg + HCl \rightarrow MgCl_2 + H_2\uparrow$

28/ $CuO + HCl \rightarrow CuCl_2 + H_2O$

29/ $ZnO + HCl \rightarrow ZnCl_2 + H_2O$

30/ $MgO + HCl \rightarrow MgCl_2 + H_2O$

31/ $MnO_2 + HCl \rightarrow MnCl_2 + Cl_2\uparrow + H_2O$

32/ $Al_2O_3 + HCl \rightarrow AlCl_3 + H_2O$

33/ $Al(OH)_3 + HCl \rightarrow AlCl_3 + H_2O$

34/ $NaOH + HCl \rightarrow NaCl + H_2O$

35/ $KOH + HCl \rightarrow KCl + H_2O$

36/ $Zn(OH)_2 + HCl \rightarrow ZnCl_2 + H_2O$

37/ $Cu(OH)_2 + HCl \rightarrow CuCl_2 + H_2O$

38/ $Cl_2 + KOH \rightarrow KClO_3 + KCl + H_2O$

39/ $Cl_2 + NaOH \rightarrow NaClO + NaCl + H_2O$

40/ $AgNO_3 + HCl \rightarrow HNO_3 + AgCl\downarrow$

41/ $CaCO_3 + HCl \rightarrow CaCl_2 + H_2O + CO_2\uparrow$

42/ $MgCO_3 + HCl \rightarrow MgCl_2 + H_2O + CO_2\uparrow$

43/ $BaSO_3 + HCl \rightarrow BaCl_2 + H_2SO_3$

44/ $AgNO_3 + NaCl \rightarrow NaNO_3 + AgCl\downarrow$

45/ $AgNO_3 + KCl \rightarrow KNO_3 + AgCl\downarrow$

46/ $H_2S + HClO_3 \rightarrow HCl + H_2SO_4$

❖ **Nước Javen :** Dẫn clo vào dung dịch NaOH:

47/ $Cl_2 + NaOH \rightarrow NaCl + NaClO + H_2O$

(natri hipoclorit)

tương tự ta cũng có :

48/ $Cl_2 + KOH \xrightarrow{t^o} KCl + KClO + H_2O$

❖ **Kali clorat $KClO_3$**

Clo đi vào dung dịch kiềm đun nóng đến $100^{oC}$ sẽ cho phản ứng :

49/ $Cl_2 + KOH \xrightarrow{t^o} KCl + KClO_3 + H_2O$

Kali clorat bị phân hủy khi đung nóng theo các phương trình :

50/ $KClO_3 \xrightarrow{t^o} KCl + O_2 \uparrow$

51/ $KClO_3 \xrightarrow{t^o} KCl + KClO_4$

(Kali peclorat)

❖ **Clorua vôi :** Clo tác dụng với vôi:

52/ $Ca(OH)_2 + Cl_2 \rightarrow CaOCl_2 + H_2O$

❖ **Các phản ứng của Brom và Iot :**

53/ $Al + Br_2 \rightarrow AlBr_3$

54/ $Al + I_2 \rightarrow AlI_3$

55/ $H_2 + Br_2 \xrightarrow{t^o} HBr$

56/ $H_2 + I_2 \xrightarrow{t^o} HI$

(hidro iotua)

57/ $HI \xrightarrow{t^o} H_2 + I_2$

Dựa trên các thuật toán tìm lời giải trên mạng tính toán ta có thể giải được các bài toán như trong các ví dụ sau:

**Ví dụ 1**: Viết phương trình phản ứng biểu diễn các biến hóa sau :

$$Zn \rightarrow ZnO \rightarrow ZnSO_4$$

*Giải :* Trên cở sở dò tìm các phản ứng (xem là các quan hệ của mạng tính toán các chất hóa học) đã biết ta có thể tìm thấy được các phản ứng sau đây :

$$Zn + O_2 \rightarrow ZnO$$
$$ZnO + H_2SO_4 \rightarrow ZnSO_4 + H_2O$$

**Ví dụ 2 :** Từ lưu huỳnh (S) và nước ($H_2O$) ta có thể điều chế được axit sunfuaric ($H_2SO_4$) không ?

*Giải :* Áp dụng các thuật toán tìm lời giải cho mạng tính toán các chất hóa học, dò theo các phản ứng liên quan đến lưu huỳnh và nước ta tìm ra được quá trình điều chế như sau :

$$H_2O \xrightarrow{\text{điện phân}} H_2\uparrow + O_2\uparrow$$
$$S + O_2 \rightarrow SO_2$$
$$SO_2 + O_2 \rightarrow SO_3$$
$$SO_3 + H_2O \rightarrow H_2SO_4$$

Tương tự như hai ví dụ trên, với tri thức gồm các phản ứng hóa học đã biết dưới dạng một mạng các chất hóa học chúng ta cũng có thể dễ dàng giải các bài toán sau đây:

**Ví dụ 3**: Viết các phương trình phản ứng để thực hiện các biến hóa theo các sơ đồ sau đây :

$$ZnS \rightarrow SO_2 \rightarrow H_2SO_4$$

$ZnS \rightarrow ZnO \rightarrow ZnCl_2$

**Ví dụ 4**: Hoàn thành các phương trình phản ứng sau đây:

$Mg + H_2SO_4 \rightarrow \ldots$
$Fe(OH)_3 + H_2SO_4 \rightarrow \ldots$
$K_2CO_3 + H_2SO_4 \rightarrow \ldots$
$Ba(NO_3)_2 + H_2SO_4 \rightarrow \ldots$

**Ví dụ 5**: Viết phương trình phản ứng theo các sơ đồ sau:

$FeS + ? \rightarrow FeCl_2 + ?$
$CuSO_4 + ? \rightarrow ? + Na_2SO_4$

**Ví dụ 6**: Từ muối NaCl và nước ($H_2O$) ta có thể điều chế được axit clohidric (HCl) và NaOH không?

# Chương 7

# HỆ HỌC

## 7.1. MỞ ĐẦU

Các chương trước đã thảo luận về biểu diễn tri thức và các kỹ thuật suy diễn. Trong trường hợp này giả định đã có sẵn tri thức và có thể biểu diễn tường minh tri thức. Tuy vậy trong nhiều tình huống, sẽ không có sẵn tri thức như:

- Kỹ sư phần mềm cần thu nhận tri thức từ chuyên gia lĩnh vực.
- Cần biết các luật mô tả lĩnh vực cụ thể.
- Bài toán không được biểu diễn tường minh theo luật, sự kiện hay các quan hệ.

Do vậy cần phát triển các hệ thống có thể học từ tập các ví dụ. Có hai tiếp cận cho hệ thống học là học từ ký hiệu và học từ dữ liệu số. Học từ ký hiệu bao gồm việc hình thức hóa, sửa chữa các luật tường minh, sự kiện và các quan hệ. Học từ dữ liệu số được áp dụng cho những hệ thống được mô hình dưới dạng số liên quan đến các kỹ thuật nhằm tối ưu các tham số. Học theo dạng số bao gồm mạng neural nhân tạo, thuật giải di truyền, bài toán tối ưu truyền thống. Các kỹ thuật học theo số không tạo ra CSTT tường minh

## 7.2. CÁC HÌNH THỨC HỌC

Có thể phân chia các loại học như sau:

### 7.2.1. Học vẹt

Hệ tiếp nhận các khẳng định của các quyết định đúng. Khi hệ tạo ra một quyết định không đúng, hệ sẽ đưa ra các luật hay quan hệ đúng mà hệ đã sử dụng. Hình thức học vẹt nhằm cho phép chuyên gia cung cấp tri thức theo kiểu tương tác.

### 7.2.2. Học bằng cách chỉ dẫn

Thay vì đưa ra một luật cụ thể cần áp dụng vào tình huống cho trước, hệ thống sẽ được cung cấp bằng các chỉ dẫn tổng quát. Ví dụ: "gas hầu như bị thoát ra từ van thay vì thoát ra từ ống dẫn". Hệ thống phải tự mình đề ra cách biến đổi từ trừu tượng đến các luật khả dụng.

### 7.2.3. Học bằng quy nạp

Hệ thống được cung cấp một tập các ví dụ và kết luận được rút ra từ từng ví dụ. Hệ liên tục lọc các luật và quan hệ nhằm xử lý từng ví dụ mới.

### 7.2.4. Học bằng tương tự

Hệ thống được cung cấp đáp ứng đúng cho các tác vụ tương tự nhưng không giống nhau. Hệ thống cần làm thích ứng đáp ứng trước đó nhằm tạo ra một luật mới có khả năng áp dụng cho tình huống mới.

### 7.2.5. Học dựa trên giải thích

Hệ thống phân tích tập các lời giải ví dụ ( và kết quả) nhằm ấn định khả năng đúng hoặc sai và tạo ra các giải thích dùng để hướng dẫn cách giải bài toán trong tương lai.

### 7.2.6. Học dựa trên tình huống

Bấy kỳ tính huống nào được hệ thống lập luận đều được lưu trữ cùng với kết quả cho dù đúng hay sai. Khi gặp tình huống mới, hệ thống sẽ làm thích nghi hành vi đã  lưu trữ với tình huống mới.

### 7.2.7. Khám phá hay học không giám sát

Thay vì có mục tiêu tường minh, hệ khám phá liên tục tìm kiếm các mẫu và quan hệ trong dữ liệu nhập. Các ví dụ về học không giám sát bao gồm gom cụm dữ liệu, học để nhận dạng các đặc tính cơ bản như cạnh từ các điểm ảnh.

## 7.3. CÂY ĐỊNH DANH

Cây định danh là một công cụ khá phổ biến trong nhiều dạng ứng dụng, với cơ chế rút trích các luật nhân quả xác định các mẫu dữ liệu.

### 7.3.1. Thí dụ về thế giới thực thu gọn

Tưởng tượng có các dữ liệu về bài toán đánh giá độ rám nắng tại một nơi nghỉ mát. Có người vui vì ngăm đen, nhưng cũng có người rát vì rộp da. Dữ liệu quan sát trên tám người được ghi lại theo bảng sau đây.

| TT | Tên người | Màu tóc | Chiều cao | Cân nặng | Dùng thuốc? | Kết quả |
|---|---|---|---|---|---|---|
| 1 | Hoa | Đen | Tầm thước | Nhẹ | Không | Bị rám |
| 2 | Lan | Đen | Cao | Vừa phải | Có | Không |
| 3 | Xuân | Râm | Thấp | Vừa phải | Có | Không |
| 4 | Hạ | Đen | Thấp | Vừa phải | Không | Bị rám |
| 5 | Thu | Bạc | Tầm thước | Nặng | Không | Bị rám |
| 6 | Đông | Râm | Cao | Nặng | Không | Không |
| 7 | Mơ | Râm | Tầm thước | Nặng | Không | Không |
| 8 | Đào | Đen | Thấp | Nhẹ | Có | Không |

**Bảng 7.1**. Số liệu quan sát về hiện tượng rám nắng.

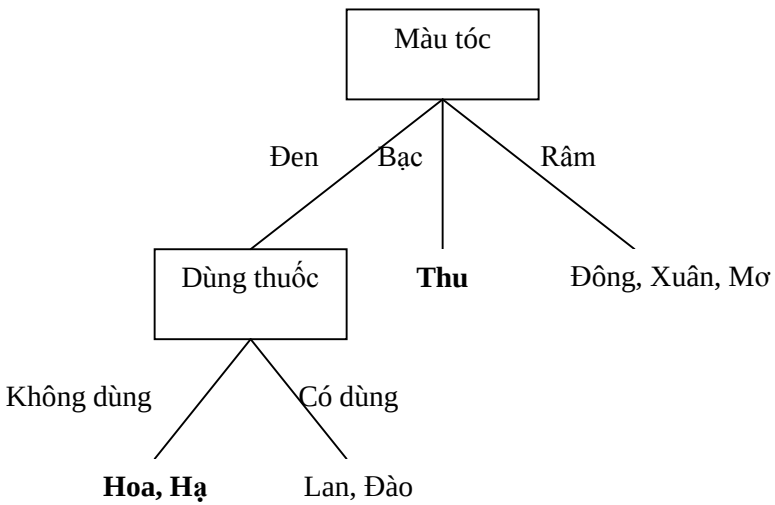
Nếu có ba dữ liệu khác nhau về màu tóc, cân nặng, chiều cao và người ta có thể dùng thuốc hoặc không thì sẽ có 3*3*3*2 = 54 tổ hợp. Một người mới được chọn thì xác suất khớp được với mẫu là 8/54 = 0.15. Xác suất này cao hơn thực tế do còn nhiều thuộc tính và nhiều giá trị khác mà thuộc tính có thể nhận.

Do vậy nếu so sánh các thuộc tính của đối tượng chưa biết với thuộc tính của các đối tượng thống kê thì không chính xác. Một số nhận xét về việc giải bài toán này:

- Người ta có thể xử lý dữ liệu như không gian các đặc tính, nhưng nếu không rõ thuộc tính nào là quan trọng hơn thì cũng khó tìm thấy đối tượng khớp nhất.
- Có thể dùng không gian các thế hệ để cô lập các thuộc tính liên quan và thuộc tính không liên quan. Tuy nhiên thường không tìm thấy lý do thuyết phục để tin được mô hình phân

loại có thể được diễn tả như tổ hợp các giá trị của tập các thuộc tính. Mặt khác cũng có thể các mẫu bị nhiễu, không thực chính xác như thế giới thực.

- Người ta dùng thủ tục phân loại chính xác các mẫu . Khi thủ tục đã học mẫu với số lượng mẫu "khá đủ", thủ tục có thể nhận ra đối tượng chưa biết.




**Hình 7.1.** Cây định danh (*Người có tên ghi đậm là người bị rám nắng*).

Một cách phù hợp cho phép thực hiện các thủ tục thử nghiệm các thuộc tính là sắp xếp các thử nghiệm trên **cây định danh**. Do cây định danh thuộc loại cây quyết định, đặc tả của nó như đặc tả cây quyết định.

❖ **Định nghĩa 7.1 : Cây định danh (Identification tree)**

*Cây định danh có thể hiện như cây quyết định, trong đó mỗi tập các kết luận được thiết lập ngầm định bởi một danh sách đã biết.*

Cây định danh dùng để xác định người bị rám nắng với kiểm tra đầu tiên là màu tóc. Nếu thấy tóc đen người ta kiểm tra có dùng thuốc không? Ngược lại, nếu tóc bạc hay râm, người ta không cần kiểm tra. Nói chung việc chọn tiến hành loại kiểm tra nào là phụ thuộc vào kết quả của lần kiểm tra trước. Xem ví dụ thể hiện trong hình 7.1.

Mỗi đối tượng đưa vào định danh đi xuống theo một nhánh cây, tùy theo các giá trị thuộc tính của nó. Cây định danh như hình vẽ có thể phân loại người trong cơ sở dữ liệu "rám nắng" vì mỗi người ứng với một nút lá trên cây định danh. Nút này dùng cho một hay nhiều người. Trong hình cho thấy người bị rám nắng được đánh dấu bằng tên in đậm hơn.

Người ta có thể đưa ra một cây, có các nút lá ứng với mỗi người, không liên quan gì đến thuộc tính rám nắng. Liệu nó được dùng phân loại không?

So sánh cây định danh và cây tổng quát, người ta thấy cây thứ nhất là cây định danh, có vẻ liên quan đến sự rám nắng nhiều hơn cây thứ hai. Cây định danh tỏ ra đã biết rằng màu tóc và phần lộ ra ánh nắng liên quan trực tiếp đến tính rám nắng.

Làm sao có thể chương trình hóa để đến được cùng một kết luận mà không cần biết trước màu tóc và việc dùng thuốc liên quan đến đặc tính của da? Một trong các giải đáp là phát biểu của Occam.

### ❖ Phát biểu Occam dùng cho các cây định danh

*Thế giới vốn đơn giản. Do vậy cây định danh gồm các mẫu là cái thích hợp nhất để định danh các đối tượng chưa biết một cách chính xác.*

Theo phát biểu này, cây định danh đầu tiên nhỏ hơn cây sau nên nó phù hợp hơn.

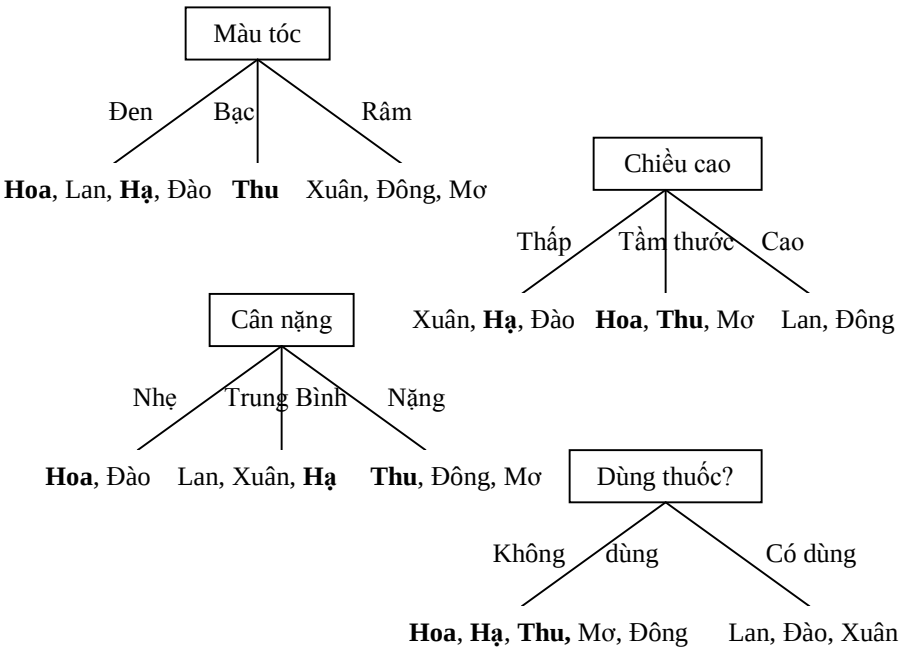
### 7.3.2. Phân loại đối tượng theo các thuộc tính

Nếu như ta tìm kiếm cây định danh nhỏ nhất khi cần có rất nhiều thử nghiệm thì thực là không thực tế. Chính vì vậy mà cũng nên dừng lại ở thủ tục xây dựng những cây định danh nhỏ, dù rằng nó không phải là nhỏ nhất. Người ta chọn thử nghiệm cho phép chia cơ sở dữ liệu các mẫu thành các tập con. Trong đó nhiều mẫu cùng chung một loại. Đối với mỗi tập có nhiều loại mẫu, dùng thử nghiệm khác để chia các đối tượng không đồng nhất thành các tập chỉ gồm đối tượng đồng nhất.

Xét ví dụ thể hiện ở hình 7.2. Cơ sở dữ liệu "rám nắng" có thể được chia nhỏ theo bốn thử nghiệm ứng với bốn thuộc tính:

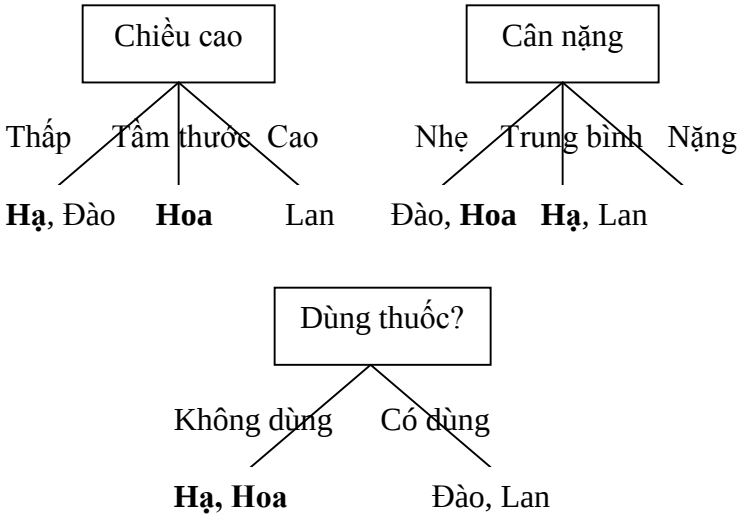
- Thử nghiệm theo cân nặng là tồi nhất nếu người ta đánh giá thử nghiệm theo các tập đồng nhất, có cùng tính chất rám nắng. Sau khi dùng thử nghiệm này, những mẫu rám nắng nằm đều ở các tập.
- Thử nghiệm theo chiều cao có vẻ tốt hơn. Có hai người trong một tập đồng nhất. Hai tập kia có lẫn cả người rám và không rám nắng.
- Thử nghiệm về việc dùng thuốc thu được ba đối tượng trong một tập đồng nhất gồm những người không rám nắng.

- Thử nghiệm thao màu tóc là tốt nhất. Trong tập đồng nhất rám nắng có một người là Thu, và tập đồng nhất không rám nắng có ba người là Đông, Mơ và Xuân.




**Hình 7.2.** Bốn cách phân chia cơ sở dữ liệu theo bốn thuộc tính khác nhau

Theo các thử nghiệm này người ta sử dụng trước tiên thử nghiệm về màu tóc. Thử nghiệm này có một tập không đồng nhất ứng với màu tóc, lẫn lộn người rám nắng và không rám nắng. Bốn người Hoa, Lan, Hạ và Đào được chia nhỏ ra.




**Hình 7.3.** Ba cách phân chia tiếp theo đối với bốn người thuộc tập không đồng nhất

Sau lần chia này người ta nhận thấy trong ba cách chia, cách chia theo việc dùng thuốc cho phép tách bốn đối tượng thành hai tập đồng nhất.

### 7.3.3. Độ lộn xộn của tập hợp

Đối với cơ sở dữ liệu  thực, không phải bất kỳ thử nghiệm nào cũng có thể cho ra tập đồng nhất. Với cơ sở dữ liệu này người ta cần đo mức độ lộn xộn của dữ liệu, hay độ không đồng nhất trong các tập con được sinh ra. Công thức đo lý thuyết thông tin về độ lộn xộn trung bình:

$$\sum_{b} \frac{n_b}{n_t} \sum_{c} - \frac{n_{bc}}{n_b} \log 2 \frac{n_{bc}}{n_b}$$

trong đó *nb* là số mẫu trong nhánh *b*, *nt* là tổng số các mẫu, và *nbc* là số mẫu trong nhánh *b* của lớp *c*.

Thực chất người ta quan tâm đến số các mẫu tại cuối nhánh. Yêu cầu *nb* và *nbc* là cao khi thử nghiệm sinh ra các tập không đồng nhất, và là thấp khi thử nghiệm sinh ra các tập hoàn toàn thống nhất.

Độ lộn xộn tính bằng $\sum_{c} -\frac{n_{bc}}{n_b} \log 2 \frac{n_{bc}}{n_b}$

Dù công thức chưa cho thấy "sự lộn xộn", nhưng người ta dùng nó để đo thông tin. Để thấy được các khía cạnh quan tâm, giả sử có một tập gồm các phần tử của hai lớp A và B. Nếu số phần tử của hai lớp là cân bằng, thì độ lộn xộn là 1 và giá trị cực đại về độ lộn xộn được tính theo:

$$-1/2\log_2 1/2 - 1/2\log_2 1/2 = 1/2 + /2 = 1$$

Mặt khác nếu phần tử thuộc chỉ một trong A, B, độ lộn xộn là 0.

$$-1\log_2 1 - 0\log_2 0 = 0$$

Độ lộn xộn bằng 0 khi tập là hoàn toàn thống nhất, và bằng 1 khi tập hoàn toàn không đồng nhất. Độ đo lộn xộn có giá trị từ 0 đến 1.

Bằng công cụ này, người ta có thể tính được độ lộn xộn trung bình của các tập tại cuối các nhánh sau lần thử nghiệm.

Độ lộn xộn trung bình= $\sum_{b} \frac{nb}{nt}$*độ lộn xộn trong tập nhánh *b*.

Trong thí dụ trước, thử nghiệm về màu tóc đã chia cơ sở dữ liệu thành ba phần. Tập tóc đen có hai người rám nắng và hai người không rám nắng. Tập tóc bạc chỉ có một người rám nắng.

Trong tập tóc râm, cả ba người không hề rám. Độ lộn xộn trung bình được tính cho kết quả 0.5.

$$4/8(-2/4\log_2 2/4 - 2/4 \log_2 2/4) + 1/8*0 + 3/8*0 = 0.5$$

Thực hiện tính toán tương tự:

| Thử nghiệm theo ... | Độ lộn xộn |
|---|---|
| Màu tóc | 0.50 |
| Chiều cao | 0.69 |
| Cân nặng | 0.94 |
| Dùng thuốc | 0.61 |

**Bảng 7.2**. Lựa chọn cách phân chia cơ sở dữ liệu theo độ đo về sự lộn xộn.

Do thử nghiệm theo màu tóc gây ra ít lộn xộn nhất, người ta dùng nó làm thử nghiệm đầu tiên. Tương tự, sau khi làm thử nghiệm này người ta làm thử nghiệm về việc dùng thuốc, do tính toán:

| Thử nghiệm theo ... | Độ lộn xộn |
|---|---|
| Chiều cao | 0.5 |
| Cân nặng | 1.0 |
| Dùng thuốc | 0.0 |

**Bảng 7.3**. Lựa chọn tiếp theo.

Vậy để tạo ra cây định danh, người ta dùng thủ tục SINH được trình bày như sau:

❖ **Thủ tục SINH dùng cây định danh:**

*Cho đến khi mỗi nút lá được ghi tên các phần tử của tập mẫu đồng nhất, thực hiện:*

*1. Chọn nút lá ứng với tập mẫu không đồng nhất.*

*2. Thay thế nút này bằng nút thử nghiệm cho phép chia tập không đồng nhất thành các tập không đồng nhất, dựa theo tính toán độ lộn xộn.*

### 7.3.4. Chuyển cây sang luật

Một khi đã dựng được cây định danh, nếu muốn chuyển các tri thức sang dạng luật thì cũng đơn giản. Người ta đi theo các nhánh của cây, từ gốc đến các nút lá, lấy các thử nghiệm làm giả thiết và phân loại nút lá làm kết luận.

Ví dụ

Các tri thức trong thí dụ về rám nắng khi nghỉ mát ở phần trên được viết thành luật:

- IF Tóc đen
  Người đó dùng thuốc
  THEN Không sao cả
- IF Người tóc đen
  Không dùng thuốc
  THEN Họ bị rám nắng
- IF Người tóc bạc
  THEN Bị rám nắng
- IF Người tóc râm
  THEN Không sao cả

#### 7.3.4.1. Lược bỏ giả thiết không cần thiết trong luật

Sau khi thu được các luật chuyển từ cây định danh, có thể bỏ đi các luật không cần thiết để đơn giản tập các luật. Người ta kiểm tra giả thiết nào có thể bỏ đi mà không thay đổi tác dụng của luật đối với mẫu.

Ví dụ

Xét luật đầu tiên trong các luật trên:

IF Tóc đen
Người đó dùng thuốc
THEN Không sao cả

Giả thiết có hai phần. Nếu bỏ đi phần đầu, còn điều kiện về "dùng thuốc". Theo các mẫu, người dùng thuốc chỉ có Lan, Xuân và Đào. Không ai trái với phần kết luận cả, tức không ai bị rám nắng. Do vậy người ta bỏ phần giả thiết đầu, thu được:

IF Người đó dùng thuốc
THEN Không sao cả

Để suy lý dễ dàng người ta thường đưa ra **bảng ngẫu nhiên.** Sở dĩ gọi vậy vì kết quả tùy thuộc vào thuộc tính.

| **Loại người** | **Không sao** | **Bị rám** |
|---|---|---|
| Người có tóc đen | 2 | 0 |
| Người tóc không đen | 1 | 0 |

**Bảng 7.4**. Bảng ngẫu nhiên theo loại tóc đối với những người dùng thuốc.

Trong bảng người ta thấy số người dùng thuốc có tóc đen, không đen và số người bị rám nắng, không rám. Bảng cho thấy tri thức về màu của tóc không quyết định vì đến việc họ bị rám nắng.

Quay lại luật trên nếu bỏ đi phần giả thiết thứ hai "dùng thuốc", người ta thấy trong số bốn người tóc đen là Hoa, Lan, Hạ và Đào, có hai người dùng thuốc mà vẫn bị rám nắng. Còn bảng ngẫu nhiên cho biết:

| Dùng thuốc ? | Không sao | Bị rám |
|---|---|---|
| Có dùng | 2 | 0 |
| Không dùng | 0 | 2 |

**Bảng 7.5**. Bảng ngẫu nhiên theo việc dùng thuốc đối với những người tóc đen.

Như vậy việc dùng thuốc có tác dụng đối với những người tóc đen. Các mẫu người tóc đen không rám nắng khi và chỉ khi họ dùng thuốc. Bởi vậy ý định bỏ giả thiết này không thực hiện.

## Ví dụ

Luật thứ hai trong tập các luật:

IF Người tóc den
Không dùng thuốc
THEN Họ bị rám nắng

Tương tự như thí dụ trên, người ta dự tính bỏ đi giả thiết đầu trong hai giả thiết. Bảng ngẫu nhiên cho thấy:

| **Loại người** | **Không sao** | **Bị rám** |
|---|---|---|
| Người có tóc đen | 0 | 2 |
| Người tóc không đen | 2 | 1 |

**Bảng 7.6.** Bảng ngẫu nhiên theo loại tóc đối với những người không dùng thuốc.

Như vậy giả thiết này là quan trọng. Thiếu nó người ta không thể đảm bảo việc khớp để kết luận rằng không bị rám nắng. Nếu xét tiếp giả thiết còn lại, trong số bốn người tóc đen có hai bị rám và hai không. Bảng ngẫu nhiên cho thấy:

| **Dùng thuốc ?** | **Không sao** | **Bị rám** |
|---|---|---|
| Có dùng | 2 | 0 |
| Không dùng | 0 | 2 |

**Bảng 7.7**. Bảng ngẫu nhiên theo việc dùng thuốc đối với những người tóc đen.

Nội dung bảng cho thấy không thể bỏ đi giả thiết này. Luật này không cần đơn giản hơn.

Ví dụ

Xét hai luật còn lại. Chúng có một giả thiết. Việc bỏ giả thiết đi là không được. Các bảng ngẫu nhiên cũng cho các tri thức như vậy.

### 7.3.4.2. Lược bỏ luật thừa

Phần trên đã đơn giản hóa các luật riêng rẽ. Nhìn tổng thể, chúng cần được tính giản nữa. Các luật không cần thiết cần được bỏ đi. Quả thật có luật trong số bốn luật thu được sẽ bị bỏ đi.

Ví dụ

Bốn luật thu gồm có:

- IF Người tóc đen
  không dùng thuốc
  THEN Họ bị rám nắng
- IF Người đó dùng thuốc
  THEN Không sao cả
- IF Người tóc bạc
  THEN Bị rám nắng
- IF Người tóc râm
  THEN Không sao cả

Hai luật có kết luận "rám nắng" và hai luật khẳng định "không sao cả". Người ta có thể thay thế hai luật khẳng định "rám nắng" bằng một luật. Gọi là **luật mặc định.** Luật mặc định là luật được dùng chỉ khi không có luật nào. Do có hai kết luận, có hai khả năng của luật mặc định:

- IF Không có luật nào
  THEN Người đó bị rám nắng
- IF Không có luật nào
  THEN Không sao cả

Chắc chắn chỉ dùng hai luật này là không thể được! Cả hai luật đều ứng với hai luật khác. Cần chọn luật mặc định là luật quét được kết luận chung trong tập mẫu, tức "không sao cả". Thí dụ này chọn:

- IF Người tóc đen
  Không dùng thuốc
  THEN Họ bị rám nắng
- IF Người tóc bạc
  THEN Bị rám nắng

- IF Không có luật nào
  THEN Không sao cả

Một cách khác chọn luật mặc định không dựa vào số mẫu thu được, mà dựa vào số các giả thiết trong các luật, người ta có tập các luật sau:

- IF Người đó dùng thuốc
  THEN Không sao cả
- IF Người tóc râm
  THEN Không sao cả
- IF Không có luật nào
  THEN Người đó bị rám nắng

Tóm lại, để chuyển cây định danh về tập các luật, thực hiện thủ tục tên là CAT sau:

❖ **Dùng thủ tục CAT cho phép tạo nên các luật từ cây định danh**

- *Tạo một luật từ mỗi nhánh gốc - lá của cây định danh.*
- *Đơn giản hóa mỗi luật bằng cách khử các giả thiết không có tác dụng đối với kết luận của luật.*

*Thay thế các luật có chung kết luận bằng luật mặc định. Luật này được kích hoạt khi không có luật nào hoạt động. Khi có nhiều khả năng, dùng phép may rủi để chọn luật mặc định.*

## 7.4. THUẬT GIẢI ILA

Thuật giải ILA (Inductive Learning Algorithm) được dùng để xác định các luật phân loại cho tập hợp các mẫu học. Thuật giải này thực hiện theo cơ chế lặp, để tìm luật riêng đại diện cho

tập mẫu của từng lớp. Sau khi xác định được luật, ILA loại bỏ các mẫu liên quan khỏi tập mẫu, đồng thời thêm luật mới này vào tập luật. Kết quả có được là một danh sách có thứ tự các luật chứ không là một cây quyết định. Các ưu điểm của thuật giải này có thể được trình bày như sau:

- Dạng các luật sẽ phù hợp cho việc khảo sát dữ liệu, mô tả mỗi lớp một cách đơn giản để dễ phân biệt với các lớp khác.
- Tập luật được sắp thứ tự, riêng biệt – cho phép quan tâm đến một luật tại thời điểm bất kỳ. Khác với việc xử lý luật theo phương pháp cây quyết định, vốn rất phức tạp trong trường hợp các nút cây trở nên khá lớn.

### 7.4.1. Xác định dữ liệu

1. Tập mẫu được liệt kê trong một bảng, với mỗi dòng tương ứng một mẫu, và mỗi cột thể hiện một thuộc tính trong mẫu.
2. Tập mẫu có *m* mẫu, mỗi mẫu gồm *k* thuộc tính, trong đó có một thuộc tính quyết định. Tổng số *n* các giá trị của thuộc tính này chính là số lớp của tập mẫu.
3. Tập luật R có giá trị khởi tạo là $\phi$.
4. Tất cả các cột trong bảng ban đầu chưa được đánh dấu (kiểm tra).

### 7.4.2. Thuật giải ILA

*Bước 1*: Chia bảng *m* mẫu ban đầu thành *n* bảng con. Mỗi bảng con ứng với một giá trị của thuộc tính phân lớp của tập tập mẫu.

(* thực hiện các bước 2 đến 8 cho mỗi bảng con*)

*Bước 2*: Khởi tạo bộ đếm kết hợp thuộc tính j, j=1.

*Bước 3*: Với mỗi bảng con đang khảo sát, phân chia danh sách các thuộc tính theo các tổ hợp phân biệt, mỗi tổ hợp ứng với j thuộc tính phân biệt.

*Bước 4*: Với mỗi tổ hợp các thuộc tính, tính số lượng các giá trị thuộc tính xuất hiện theo cùng tổ hợp thuộc tính trong các dòng chưa được đánh dấu của bảng con đang xét (mà đồng thời không xuất hiện với tổ hợp thuộc tính này trên các bảng còn lại). Gọi tổ hợp đầu tiên (trong bảng con) có số lần xuất hiện nhiều nhất là *tổ hợp lớn nhất*.

*Bước 5*: Nếu tổ hợp lớn nhất bằng ϕ, tăng *j* lên 1 và quay lại bước 3.

*Bước 6*: Đánh dấu các dòng thoả tổ hợp lớn nhất của bảng con đang xử lý theo lớp.

*Bước 7*: Thêm luật mới vào tập luật R, với vế trái là tập các thuộc tính của tổ hợp lớn nhất (kết hợp các thuộc tính bằng toán tử AND) và vế phải là là giá trị thuộc tính quyết định tương ứng.

*Bước 8*: Nếu tất cả các dòng đều đã được đánh dấu phân lớp, tiếp tục thực hiện từ bước 2 cho các bảng con còn lại. Ngược lại (nếu chưa đánh dấu hết các dòng) thì quay lại bước 4. Nếu tất cả các bảng con đã được xét thì kết thúc, kết quả thu được là tập luật cần tìm.

### 7.4.3. Mô tả thuật giải ILA

ILA là một thuật giải khá đơn giản rút trích các luật dẫn từ một tập mẫu. Mỗi mẫu được mô tả dưới dạng một tập xác định các thuộc tính, mỗi thuộc tính ứng với một vài giá trị nào đó.

Để minh họa thuật giải ILA, chúng ta sử dụng tập mẫu cho trong bảng 7.8, gồm có 7 mẫu ($m$=7), 3 thuộc tính ($k$=3), và thuộc tính quyết định (phân lớp) có hai giá trị là {yes, no} ($n$=2). Trong ví dụ này, "Size", "Color" và "Shape" là các thuộc tính với các nhóm giá trị {small, medium, large}, {red, blue, green}, và {brick, wedge, sphere, pillar}.

| Mẫu số | Size | Color | Shape | Decision |
|---|---|---|---|---|
| 1 | medium | blue | brick | yes |
| 2 | small | red | wedge | no |
| 3 | small | red | sphere | yes |
| 4 | large | red | wedge | no |
| 5 | large | green | pillar | yes |
| 6 | large | red | pillar | no |
| 7 | large | green | sphere | yes |

**Bảng 7.8.** Tập mẫu học cho bài toán phân lớp đối tượng

Do $n$ = 2, bước đầu tiên ta chia tập mẫu thành hai bảng con như trong bảng 7.9.

| ***Bảng con 1*** | | | | |
|---|---|---|---|---|
| **Mẫu số cũ mới** | **Size** | **Color** | **Shape** | **Decision** |
| 1 1 | medium | blue | brick | yes |

| | | | | |
|---|---|---|---|---|
| 3 2 | small | red | sphere | yes |
| 5 3 | large | green | pillar | yes |
| 7 4 | large | green | sphere | yes |
| ***Bảng con 2*** | | | | |
| **Mẫu số cũ mới** | **Size** | **Color** | **Shape** | **Decision** |
| 2 1 | small | red | wedge | no |
| 4 2 | large | red | wedge | no |
| 6 3 | large | red | pillar | no |

**Bảng 7.9.** Chia thành hai bảng con theo thuộc tính Decision

Áp dụng bước 2 của thuật giải vào bảng con thứ nhất trong bảng 7.9. Với $j$=1, danh sách các tổ hợp thuộc tính gồm có {Size}, {Color}, và {Shape}.

Với tổ hợp {Size}, giá trị thuộc tính "medium" xuất hiện trong bảng con thứ nhất nhưng không có trong bảng con thứ hai, do đó giá trị tổ hợp lớn nhất là "medium". Bởi vì các giá trị thuộc tính "small" và "large" xuất hiện trong cả hai bảng con, nên không được xét trong bước này. Với tổ hợp {Size}, giá trị thuộc tính "medium" chỉ bằng 1, ta xét tiếp cho tổ hợp {Color} thì giá trị tổ hợp lớn nhất là bằng 2, ứng với thuộc tính "green", còn thuộc tính "blue" là bằng 1. Tương tự như vậy, với tổ hợp {Shape}, ta có "brick" xuất hiện một lần, và "sphere" hai lần. Đến cuối bước 4, ta có tổ hợp {Color} với thuộc tính "green" và {Shape} với thuộc tính "sphere" đều có số lần xuất hiện lớn nhất là 2. Thuật toán mặc định chọn trường hợp thứ nhất để xác

định luật tổ hợp lớn nhất. Dòng 3 và 4 được đánh dấu đã phân lớp, ta có luật dẫn như sau:

Rule 1: **IF color** là **green THEN decision** là **yes**

Ta tiếp tục thực hiện từ bước 4 đến 8 cho các mẫu còn lại (chưa đánh dấu) trong bảng con này (tức dòng 1 và 2). Áp dụng tương tự như trên, ta thấy giá trị thuộc tính "medium" của {Size}, "blue" của "Color", "brick" và "sphere" của {Shape} đều xuất hiện một lần. Bởi vì số lần xuất hiện này giống nhau, thuật giải áp dụng luật mặc định chọn trường hợp đầu tiên. Ta có thêm luật dẫn sau:

Rule 2: **IF size** là **medium THEN decision** là **yes**

Đánh dấu cho dòng 1 trong bảng con thứ nhất. Tiếp tục áp dụng bước 4 đến 8 trên dòng còn lại (tức dòng 2). Giá trị thuộc tính "sphere" của {Shape} xuất hiện một lần, ta có luật dẫn thứ ba:

Rule 3: **IF shape** là **sphere THEN decision** là **yes**

Dòng 2 được đánh dấu. Như vậy, tất cả các dòng trong bảng con 1 đã được đánh dấu, ta chuyển qua xử lý tiếp bảng con 2. Thuộc tính "wedge" của {Shape} xuất hiện hai lần trong dòng 1 và 2 của bảng con này. Đánh dấu các dòng này với luật dẫn thứ tư như sau:

Rule 4: **IF shape** là **wedge THEN decision** là **no**

Với dòng còn lại (tức dòng 3) của bảng con 2, ta có thuộc tính {Size} với giá trị "large" có xuất hiện trong bảng con 1. Do đó, theo thuật giải, ta loại bỏ trường hợp này. Tương tự như vậy cho giá trị "red" của {Color} và "pillar" của {Shape}. Khi đó, ILA tăng *j* lên 1, và khởi tạo các tổ hợp 2 thuộc tính là {Size và

Color}, {Size và Shape}, và {Color và Shape}. Các tổ hợp thứ nhất và thứ ba thoả mãn điều kiện không xuất hiện trong bảng con 1 với các cặp thuộc tính hiện có của dòng này. Theo luật mặc định, ta chọn luật theo trường hợp thứ nhất. Đánh dấu dòng này, ta có thêm luật dẫn thứ 5.

Rule 5:

**IF size** là **large AND color** là **red THEN decision** là **no**

Bởi vì lúc này tất cả các dòng trong bảng con hai cũng đầu đã được đánh dấu phân lớp, đồng thời không còn bảng con nào chưa xét, thuật giải kết thúc.

### 7.4.4. Đánh giá thuật giải

Số lượng các luật thu được xác định mức độ thành công của thuật giải. Đây chính là mục đích chính của các bài toán phân lớp thông qua một tập mẫu học. Một vấn đề nữa để đánh giá các hệ học quy nạp là khả năng hệ thống có thể phân lớp các mẫu được đưa vào sau này.

Thuật giải ILA được đánh giá mạnh hơn hai thuật giải khá nổi tiếng về phương pháp học quy nạp trước đây là ID3 và AQ, đã thử nghiệm trên một số tập mẫu như Balloons, Balance, và Tic-tac-toe (lấy từ Kho Dữ liệu Máy học và Giả thuyết - Đại học California Irvine).

## Chương 8

# KẾT HỢP CƠ SỞ TRI THỨC VÀ CƠ SỞ DỮ LIỆU

## 8.1. MỞ ĐẦU

Trong phần này chúng tôi trình bày cách kết hợp cơ sở tri thức (CSTT) với cơ sở dữ liệu (CSDL) nhằm suy diễn thông tin từ CSDL bằng các luật suy diễn trong CSTT. Trước hết, chúng tôi nêu ra dạng của các luật suy diễn, cách suy diễn thông tin từ các luật có trong CSTT và dữ liệu trong CSDL, cách quản trị CSTT. Trong các phần sau, chúng tôi sẽ trình bày cách kết hợp CSTT với CSDL để khai thác dữ liệu và suy diễn thông tin dựa trên tri thức và hiểu biết của chuyên gia về dữ liệu.

## 8.2. KẾT HỢP CSDL VÀ CSTT

### 8.2.1. Dạng luật trong CSTT

Chúng tôi gọi các luật của CSTT kết hợp với CSDL là các luật suy diễn dữ liệu, các luật này có dạng tổng quát như sau:

**H: - $G_1$ & $G_2$ … & $G_k$**

trong đó: **H** được gọi là phần đầu hay kết luận của luật. **$G_1$** & **$G_2$** … & **$G_k$** là phần thân của luật. Các **$G_k$** là đích con hay tiền đề của luật.

**Ví dụ 1:** **parent**(X,Y): - **father**(X,Y) | **mother**(X,Y)

Với **father**, **mother**, **parent** là các vị từ **X**,**Y** là các biến. Mỗi vị từ **p**(**X**,**Y**,**Z**) ứng với một quan hệ **P**(**X**,**Y**,**Z**). Chúng tôi phân biệt hai loại vị từ, một là vị từ được suy và hai là vị từ nền.

- ***Vị từ được suy IDB*** (intensional predicate): là vị từ xuất hiện trong phần kết luận của luật. Vị từ được suy cũng có thể xuất hiện trong phần thân của luật.
- ***Vị từ nền EDB*** (extensional predicate): ứng với một quan hệ được lưu trữ trong CSDL. Mỗi vị từ ứng với một quan hệ. Một vị từ có thể nhận được giá trị đúng hay sai. Nếu p là vị từ nền và **P** là quan hệ nền thì **p**(**a**,**b**,**c**) với **a**, **b**, **c** là đối sẽ có giá trị đúng nếu bộ (**a**,**b**,**c**) sẽ tạo được trong tiến trình suy diễn.

Trong ví dụ trên, father và mother là các vị từ nền(EDB) còn parent là vị từ được suy(IDB). Ta cũng phân ra hai dạng luật suy diễn dữ liệu là:

- ***Luật không đệ qui***: Vị từ ở phần đầu không xuất hiện trong phần thân của luật.

**Ví dụ 2: sibling**(**X**,**Y**): - **parent**(**Z**,**X**) & **mother**(**Z**,**Y**).

- ***Luật đệ qui***: Vị từ ở phần đầu xuất hiện trong phần thân của luật.

**Ví dụ 3*:*** **ancestor**(**X**,**Y**): - **parent**(**X**,**Y**).

**ancestor**(**X**,**Y**): - **parent**(**X**,**Z**) & **ancestor**(**Z**,**Y**)

Trong hệ thống của mình, chúng tôi chỉ tập trung khảo sát các luật suy diễn không đệ qui cho CSTT.

### 8.2.2. Ý nghĩa của các phép toán logic trong các luật suy diễn

Trong các luật suy diễn dữ liệu có các phép toán logic ***and***(&), ***or***(|), ***not***(~) để kết nối các vị từ trong phần thân của luật. Các phép toán này được định nghĩa dựa trên các phép toán của đại số quan hệ và các câu lệnh SQL tương đương như sau.

#### 8.2.2.1. Phép toán AND

Phép **AND(&)** được xây dựng trên cơ sở phép kết và phép chiếu của đại số quan hệ.

Với luật **t**(**a**,**b**,**d**,**e**):- **r**(**a**,**b**,**c**) & **s**(**c**,**d**,**e**), quan hệ trong **T**(**a**,**b**,**d**,**e**) ứng với vị từ **t**(**a**,**b**,**d**,**e**):được tính theo cách sau:

$$\mathbf{T(a,b,d,e) = \Pi_{a,b,d,e}\ (\ R(a,b,c) \infty\ S(c,d,e)\ ).}$$

Nếu dùng câu SQL, ta có câu lệnh tương ứng:

**SELECT** **r**.**a**, **r**.**b**, **s**.**d**, **s**.**e**

**FROM** table **r**, table **s**
**WHERE** **r**.**c** = **s**.**c**.

Nếu một vị từ trong phần tiền đề của luật có dạng một biểu thức so sánh với luật **s**(**a**,**b**,**c**): - **r**(**a**,**b**,**c**) & (**b** >= 2). Quan hệ **S**(**a**,**b**,**c**) ứng với vị từ **s**(**a**,**b**,**c**) **ø** được suy **s**(**a**,**b**,**c**) được tính theo công thức sau:

$$\mathbf{s(a,b,c) = \delta_{b >= 2}\ (\ r\ (a,b,c)\ )}$$

trong đó $\delta$ là phép chọn theo điều kiện **b** >= 2, nếu dùng câu SQL,ta có câu lệnh tương ứng:

**SELECT** *

**FROM** table **r**

**WHERE** **r**.**b** >= 2

### 8.2.2.2. Phép toán OR

Phép toán **OR (|)** được xây dựng trên cơ sở phép hợp sau đây:

**t(a,b,c): - r(a,b,c) | s(a,b,c)**

Quan hệ **T**(**a**,**b**,**c**) trong **t**(**a**,**b**,**c**) được tính theo cách sau:

**T(a,b,c) = R(a,b,c) ∪ S(a,b,c)**

Nếu dùng SQL, ta có câu lệnh tương ứng :

**SELECT** *

**FROM** table1 **r**

**UNION SELECT** * **FROM** table2 **r** **INTO** table **t**

### 8.2.2.3. Phép toán NOT (~)

Phép **not(~)** được xây dựng trên cơ sở phép hiệu, ví dụ:

**t(a,b,c): - r(a,b,c) & (~s(a,b,c))**

Quan hệ được suy **T**(**a**,**b**,**c**) của vị từ **t**(**a**,**b**,**c**) được tính theo cách sau:

**t(a,b,c) = r(a,b,c) \ s(a,b,c)**

Nếu dùng SQL, ta có thể cài đặt như sau:

**SELECT** **a**, **b**, **e**

**FROM** table **r**

**WHERE** **a NOT IN** (**SELECT a FROM s**)

## 8.3. MÔ HÌNH SUY DIỄN

### 8.3.1. So khớp và đồng nhất biến

Mục đích là để so sánh các thành phần để tìm vị từ và sự kiện trong tiến trình suy diễn. Sau khi so khớp sẽ xảy ra tiến trình đồng nhất biến theo nghĩa thay thế biến bằng một giá trị cụ thể. Xét hai luật sau:

**r**1: **grandfather**(**X**,**Y**): -**father**(**X**,**Z**) & **parent**(**Z**,**Y**)

**r**2: **parent**(**X**,**Y**): - **father**(**X**,**Y**) | **mother**(**X**,**Y**)

Quan hệ **Father**(**A**,**B**), **Mother**(**A**,**B**) được mô tả bằng vị từ **father**(**A**,**B**), **mother**(**A**,**B**) là các quan hệ nền (hay **EDB**). Từ các vị từ ngoài, nhờ các luật suy diễn chúng ta có thể tạo sinh các quan hệ cho các vị từ được suy(**IDB**) là **Parent**(**A**,**B**) hay **Grandfather**(**A**,**B**) như trong ví dụ trên.

Có thể mô tả các luật suy diễn bằng đồ thị suy diễn. Ví dụ với hai luật trên ta có thể tạo đồ thị dạng cây suy diễn ở hình 8.1.

Trong tiến trình suy diễn để tạo quan hệ cho vị từ được suy **grandfather**(**X**,**Y**) chúng ta cần tạo các quan hệ cho các vị từ **father**(**X**,**Z**) và **parent**(**Z**,**Y**). Do **father**(**X**,**Z**) là quan hệ nền nên chỉ cần thẩm định **parent**(**Z**,**Y**) bằng cách so khớp và đồng nhất biến để có dạng sau:

**parent(Z,Y):- father(Z,Y) | mother(Z,Y)**

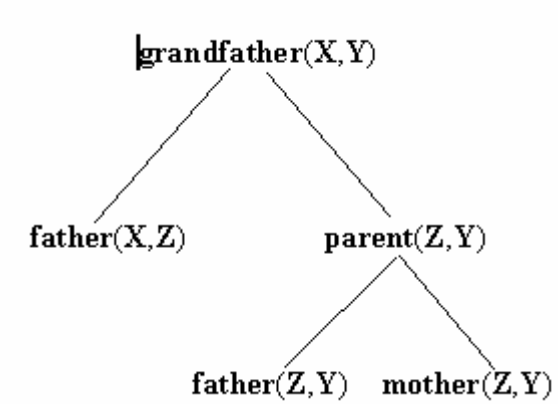
Với vị từ nền **father**(**Z**,**Y**) chúng ta sử dụng so khớp và đồng nhất biến theo thứ tự xuất hiện của đối trong vị từ và thứ tự xuất hiện trường trong quan hệ nền. Từ quan hệ **father**(**F,C**),

chúng ta có các đồng nhất biến sau cho **father**(**X**,**Z**) và **father**(**Z**,**Y**).

**FATHER (F C ) father( X Z ) father( Z Y )**

an son an son an son

loc vinh loc vinh loc vinh




**Hình 8.1**. Đồ thị suy diễn

### 8.3.2. Phân giải luật suy diễn không đệ qui

Nhìn chung có hai bước chính là:

- Tạo cây suy diễn theo các luật.
- Duyệt cây để thẩm định và tạo sinh dữ liệu cho vị từ được suy.

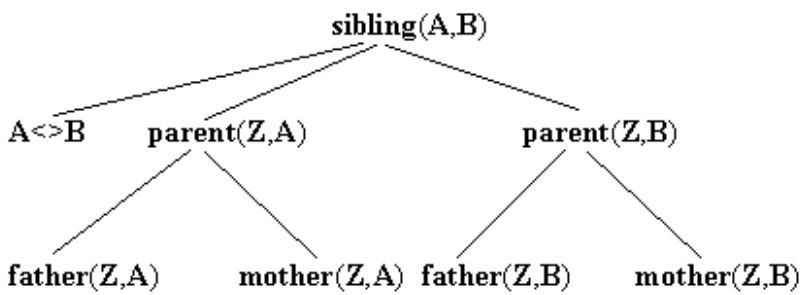
Với hai luật:

**r**1: **sibling**(**X**,**Y**):- **parent**(**X**,**Z**) & **parent**(**Z**,**Y**) & (**X**<>**Y**)

**r**2: **parent**(**X**,**Y**):- **father**(**X**,**Y**) | **mother**(**X**,**Y**)

Vị từ được suy **sibling**(**X**,**Y**) với các đích **sibling**(**A**,**B**) sẽ được thẩm định như sau:

- Tìm luật so khớp được với đích và thực hiện đồng nhất biến.
- Tạo cây con có gốc chính là đích của bài toán. Xử lý đệ qui với các đích con là các lá của cây con vừa mới tạo được. Nếu vị từ trong lá là vị từ nền thì không thể mở rộng được cây con.
- Kết quả ta đồ thị suy diễn ở hình 8.2:



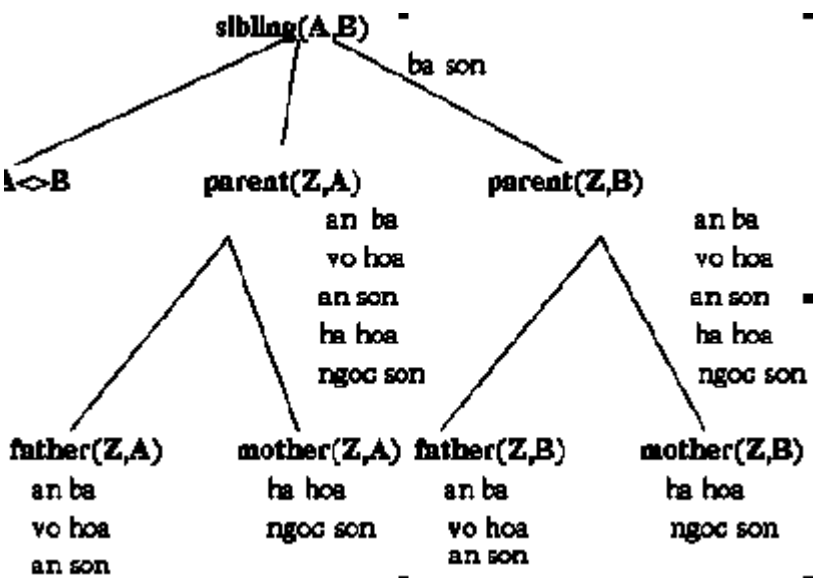

**Hình 8.2**. Đồ thị suy diễn

Sau khi đã tạo xong cây suy diễn, chúng ta bước sang suy diễn bằng cách duyệt cây để tạo sinh dữ liệu cho các vị từ được suy. Ý tưởng của thuật toán như sau:

- Tìm luật có phần đầu so khớp được với đích.
- Tạo dữ liệu cho các vị từ trong phần thân của luật.
- Thực hiện các phép toán logic của cơ sở dữ liệu suy diễn để tạo dữ liệu cho vị từ được suy.

Trong phần 4, chúng tôi sẽ trình bày chi tiết các thuật toán trên.

**Ví dụ**: với các vị từ nền **father**(**X**,**Y**) và **mother**(**X**,**Y**) sau đây, chúng ta có thể tạo dữ liệu cho quan hệ **sibling**(**X**,**Y**) thông qua đồ thị suy diễn ở hình 8.3.




**Hình 8.3**. Tạo ra các quan hệ IDB trên đồ thị suy diễn

## 8.4. VÍ DỤ MINH HỌA

Trong phần này, chúng tôi trình bày một ví dụ kết hợp CSTT và CSDL để suy diễn thông tin từ CSDL .

### 8.4.1. Phần Cơ sở dữ liệu

Xét một CSDL theo dõi kết quả học tập của sinh viên với các quan hệ nền sau.

a. **Nganhhoc**(manganh,tennganh): chứa các ngành học của trường.

b. **Sinhvien**(masv,manganh,holot,ten): chứa hồ sơ sinh viên.

c. **Moncoso**(manganh,mamon,tenmon,sotc): các môn cơ sở của từng ngành học và số tín chỉ của môn.

d. **Chuan**(manganh,mamon,dchuan): điểm chuẩn của môn cơ sở chính của ngành dùng để suy diễn.

e. **Diemcoso**(masv,mamon,dmon): điểm kết quả môn của sinh viên.

f. **Chuyennganh**(manganh,machnganh):các chuyên ngành của từng ngành.

g. **Chuyende**(machnganh,machde,tenchde,sotc): các chuyên đề của ngành.

h. **Diemchde**(masv,machde,dchde): điểm kết quả chuyên đề của sinh viên.

### 8.4.2. Phần Cơ sở tri thức

Phần CSTT bao gồm các luật suy diễn dữ liệu sau:

**a. Các luật suy diễn 1: Sinh viên giỏi ở giai đoạn cơ sở**

***r1:***

**IF** **diemcoso**(masv,mamon,dmon)

**AND** **chuan**(manganh, mamon, dmon,dchuan)

**THEN diemchinh**(masv,manganh,mamon,dmon,dchuan)

Tạo quan hệ **diemchinh** (masv, manganh, mamon, dmon, dchuan) chứa điểm môn quan trọng của ngành ở giai đọan của sinh viên và điểm chuẩn để suy diễn của môn đó.

***r2:***

**IF dchinh**(masv,manganh, mamon, dmon,dchuan)

**AND** (dmon> dchuan)

**THEN svgioicoso** (masv,manganh)

Tạo quan hệ **svgioicoso**(masv,manganh) chứa các sinh viên được đánh giá là giỏi ở giai đoạn cơ sơ theo quan điểm nếu điểm môn quan trọng của sinh viên lớn hơn điểm chuẩn.

**b. Các luật suy diễn 2: Sinh viên giỏi ở giai đoạn chuyên ngành**

***r3:*** **IF chuyende**(machnganh,machde,tenchde,sotchi)

**AND** (sotchi>5)

**THEN chdeqtrong**(machnganh, machde)

Tạo quan hệ **chdeqtrong**(machnganh,machde) chứa các chuyên đề quan trọng theo nghĩa có số tín chỉ lớn hơn 5.

***r4:*** **IF diemde**(masv,machde,dchde) **AND** (dchde>8)

**THEN svgioicde**(masv, machde)

Tạo quan hệ **svgioicde**(masv,machde) chứa các chuyên đề mà sinh viên đạt kết quả tốt.

***r5:*** **IF svgioicde**(masv,machde)

**AND chdeqtrong**(machnganh,machde)

**THEN svgioichng**(masv, machnganh)

Tạo quan hệ **svgioichng**(masv, machnganh) chứa các sinh viên được đánh giá là giỏi chuyên ngành được xác định bởi machnganh. Ở đây có thể dùng độ đo niềm tin vào để đánh giá.

**c. Các luật suy diễn 3: Sinh viên sẽ làm sẽ làm nghiên cứu sinh theo chuyên ngành nào sau khi tốt nghiệp**

***r6:*** **IF svgioicoso**(masv,manganh)

**AND svgioichng**(masv, machnganh)

**THEN svtheochnganh**(masv,machnganh)

Tạo quan hệ **svtheochnganh**(masv,machnganh) suy luận về chuyên ngành sinh viên sẽ làm nghiên cứu sinh. Luật này có độ đo chính xác của luật.

## 8.5. XÂY DỰNG ĐỘNG CƠ SUY DIỄN THÔNG TIN TỪ CSDL DỰA TRÊN CÁC LUẬT TRONG CSTT

### 8.5.1. Tổ chức dữ liệu

Chúng tôi dùng một đồ thị suy diễn **G** = (**E**,**V**) để chứa các luật suy diễn dữ liệu, trong đó **E** là các cạnh do các luật suy diễn dữ liệu tạo ra và **V** là tập các đỉnh ứng với các quan hệ **EDB** hau **IDB**. Đồ thị này được lưu trong các bảng sau:

**a) Bảng 1: EvidenceTable(RuleNo,Term)**

Thuộc tính Term của bảng này chứa dạng Postfix của phần giả thuyết của luật suy diễn dữ liệu.

Với dạng lưu trữ của luật

**r**1: **IF** ( NOT **a**) **AND** ( NOT **b** )

**THEN** sẽ là **a ~ b ~ &**

**b) Bảng 2: RuleTable(Ruleno, Evidence, Conclusion,Cfrule)**

Chứa luật suy diễn dữ liệu. Cfrule là độ tin cây của luật

**c) Bảng 3: NodeTable(NodeNo,NodeType, NodeDescription, Cfnode)**

Chứa thông tin của các node trên đồ thị suy diễn. Thuộc tính Nodetype có ba gía trị là 1 nếu là node lá, 2 nếu là nối mục tiêu trung gian và 3 nếu là mục tiêu cuối.

## 8.5.2. Quản trị CSTT

**Chúng tôi xây dưng bộ quản trị CSTT bao gồm các chức năng thêm, xoá, sửa luật. Với bộ quản trị CSTT này, chúng ta có thể tiến hành soạn thảo các luật suy diễn dữ liệu.**

### 8.5.2.1. Các chức năng quản trị CSTT

**a) Hàm tạo dữ liệu cho các bảng EvidenceTable**

Function **CreaPostFix**( **Ruleno**, **Cfrule**:string): table

**Begin**

ExpArray = ChangeToPostFixArray(Cfrule);

**For** ( each Term of ExpArray ) **do**

WriteDataToTableEvidence(Ruleno, Term);

**Return** (EvidenceTable);

**End**;

### b) Hàm tạo dữ liệu cho các bảng RuleTable

Function **CreaRule**( Ruleno, RuleEvidence, RuleConclusion, Cfrule:string): table

**Begin**

WriteDataToTableRule(Ruleno,RuleEvidence, RuleConclusion, Cfrule);

**Return** (RuleTable);

**End**;

### c) Hàm tạo dữ liệu cho các bảng NodeTable

Function **CreaNode**( Ruleno, Cfrule:string): table

**Begin**

ExpArray = CreateAllNodesOfNet (EvidenceTable, RuleTable);

**For** ( each Nodeno of ExpArray) **do**

**Begin**

NodeType = AssignNodeType(Rule,Nodeno);
WriteDataToTableNode(NodeInfo,NodeType);

**End**;

**Return** (NodeTable);

**End**;

## 8.5.2.2. Các chức năng thêm, xóa, sửa các luật trong CSTT

### a) Thêm luật vào CSTT

```
Function    AddRule(Ruleno, RuleEvidence, RConclusion: string;
            Cfrule: real; EvidenceTable, RuleTable,NodeTable:
            table) : BOOL
Begin
    EvidenceTable = CreaPostFix( Ruleno, Cfrule);
    RuleTable = CreaRule( Ruleno, RuleEvidence,
    RuleConclusion, Cfrule:string);
    NodeTable = CreaNode( EvidenceTable, RuleTable);
      If ( NOT Validate(Ruleno))  then
      Begin
        DeleteRule(Ruleno, EvidenceTable, RuleTable,
        NodeTable);
            return FALSE;
      End;
  return TRUE;
End;
```

### b) Hàm xử lý xóa luật khỏi CSTT

```
Function DeleteRule(Ruleno: string;
EvidenceTable,RuleTable,NodeTable: table) : BOOL
Begin
  DeleteRuleInEvidenceTable( Ruleno);
  DeleteRuleInRuleTable(Ruleno);
```

```
    NodeTable = CreaNode(EvidenceTable, RuleTable);
    If ( NOT Validate(Ruleno) Then
Begin
        UnDeleteRule(Ruleno, EvidenceTable, RuleTable,NodeTable);
          return FALSE;
     End;
     return TRUE;
End;
```

### c) Hàm sửa luật suy diễn trong CSTT

```
Function EditRule(Ruleno: string;
EvidenceTable,RuleTable,NodeTable: table): BOOL
Begin
     RuleInfo= ReadInfoFromTable(Ruleno,
     RuleTable);
  Edit(RuleInfo);
    If(DeleteRule(Ruleno,EvidenceTable,RuleTabl, NodeTable)
Then
    Begin
          AddRule(Ruleno,RuleInfo.RuleEvidence,
          RuleInfo.RConclusion,RuleInfo.Cfrule,EvidenceTable,
          RuleTable, NodeTable);
                return TRUE;
         End;
```

**return** TRUE;

**End**;

### d) Hàm kiểm tra tính hợp lệ của CSTT

Function **ValidateRule**(Ruleno: string; EvidenceTable, RuleTable, NodeTable:table):BOOL

**Begin**

**If** ( HasLoopInNet(EvidenceTable, RuleTable, NodeTable))

**return** FALSE;

**return** TRUE;

**End**;

## 8.5.3. Một số thuật toán để trong động cơ suy diễn cho các luật suy diễn dữ liệu

### a) Hàm tính trị của một nút

Function **EvaluateNode**(NodeName:string): real;

**Begin**

**If** ( NodeName is a Terminator) **Then**

**Begin**

**Do case**

**Case** SimpleNode: Ask for the Cfnode;

**Case** CompoundNode: Evaluate the Node

Expression;

Create CfNode;

```
            Endcase
          Return (CfNode);
    End;
    If ( NodeName had Cfnode) Then
        return ( Cfnode);
        MatchingRuleArray = FindMatchRules(NodeName);
        Ruleno = GetNextRuleFromArray( MatchingRuleArray);
        Val1 = EvaluateRule(Ruleno);
    For ( each rule of MatchingRuleArray )do
    Begin
       Ruleno = GetNextRuleFromArray(MatchingRuleArray);
   Val2 = EvaluateRule(Ruleno);
   Val1 = CombinesRules (Val1,Val2);
          End;
    return(Val1);
End;
```

### b) Hàm tính trị của một luật

```
Function EvaluateRule(Ruleno:string): string;
Begin
    PostFixArray = CreateTermFromEvidenceTable(Ruleno);
    InitStack();
    For ( each element of PostfixArray ) do
```

```
Begin
    X = GetNextTermFromArray(PostFixArray);
    If ( IsOperand(X) ) Then PushStack(X);
    If ( IsOperator(X) ) Then
    Begin
        y = PopStack();
        Val1 = (IsNode(Y))?EvaluateNode(Y):Y ;
        Do case
            Case IsAnd(X):
                Y = PopStack();
                Val2 = EvaluateNode(Y);
                Val = ANDCombine(Val1, Val2);
                                        PushStack(Val);
            Case IsOr(X):
                Y = PopStack();
                Val2 = EvaluateNode(Y);
    Val = ORCombine(Val1, Val2);
                PushStack(Val);
            Case IsNot(X):
                Val = NotEvaluate(Val1);
    PushStack(Val);
  EndCase;
End;
```

**Return** (( PopStack() * Cfrule )

**End**;

### c) Các hàm phân giải các phép toán AND, OR, NOT

Các hàm **ANDCombine( ); ORCombine( ); NOTEvaluate( )** nhằm thức hiện các phép toán **AND**, **OR**, **NOT** nhằm tạo sinh ra quan hệ ứng với vị từ kết qủa như đã nêu trong mục 8.2.2.1, 8.2.2.2, và 8.2.2.3.

**Chương 9**

# HỆ THỐNG MỜ CHO CÁC BIẾN LIÊN TỤC

## 9.1. MỞ ĐẦU

Các chuyên gia sử dụng các lập luận một cách tự nhiên để giải quyết các bài toán. Các tri thức này thường là các tri thức không rõ ràng và rất khó diễn tả bằng các hệ thống logic truyền thống.

Từ những năm 1920, Lukasiewicz đã nghiên cứu cách diễn đạt toán học khái niệm mờ. Năm 1965, Lotfi Zadeh đã phát triển lý thuyết khả năng và đề xuất hệ thống logic mờ (fuzzy logic). Hiện nay, logic mờ đang được ứng dụng rộng rãi trong rất nhiều lĩnh vực, đặc biệt là các hệ thống điều khiển mờ. Chương này trình bày các khái niệm cơ bản về logic mờ, lập luận mờ và hệ thống điều khiển mờ tiêu biểu.

## 9.2. CÁC KHÁI NIỆM CƠ BẢN

### 9.2.1. Tập rõ và hàm thành viên

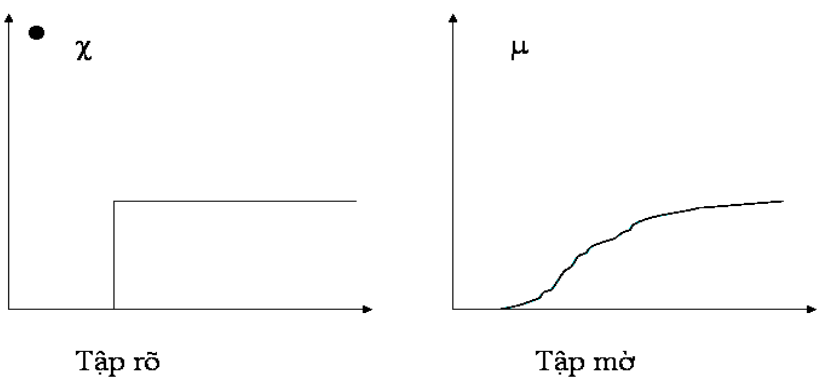
Tập rõ crisp set) là tập hợp truyền thống theo quan điểm của Cantor (crisp set). Gọi A là một tập hợp rõ, một phần tử x có thể có $x \in A$ hoặc $x \notin A$, Có thể sử dụng hàm $\chi$ để mô tả khái niệm thuộc về. Nếu x

$\in$A, $\chi(x) = 1$, ngược lại nếu $x \notin A$, $\chi(x) = 0$. Hàm $\chi$ được gọi là hàm đặc trưng của tập hợp A.

### 9.2.2. Tập mờ và hàm thành viên

Khác với tập rõ, khái niệm thuộc về được mở rộng nhằm phản ánh mức độ x là phần tử của tập mờ A. Một tập mờ fuzzy set): A được đặc trưng bằng hàm thành viên $\mu$ và cho x là một phần tử $\mu(x)$ phản ánh mức độ x thuộc về A.

- Ví dụ: Cho tập mờ High
- Lan cao 1.5m, $\mu$(Lan)=0.3
- Hùng cao 2.0 m, $\mu$(Hùng)=0.9



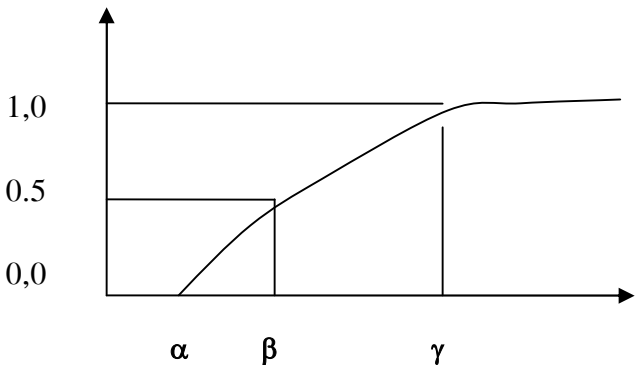

**Hình 9.1.** Đường biểu diễn của hàm đặc trưng và hàm thành viên

### 9.2.3. Các dạng của hàm thành viên

Các hàm thành viên của tập mờ có ba dạng cơ bản là: dạng tăng, dạng giảm và dạng chuông.

**a) Dạng S tăng**

$$\mu(x) = S(x, \alpha, \beta, \gamma) = \begin{cases} 0 \text{ nếu } x <= \alpha \\ 2((x - \alpha)/(\gamma - \alpha))^2 \text{ nếu } \alpha < x <= \beta \\ 1 - [2((x - \alpha)/(\gamma - \alpha))^2] \text{ nếu } \beta < x < \gamma \\ 1 \text{ nếu } x >= \gamma \end{cases}$$



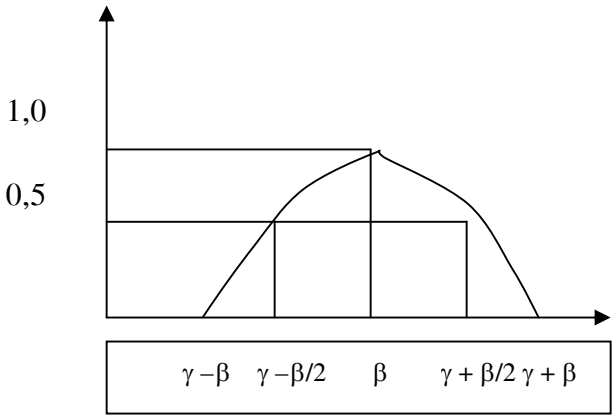

**Hình 9.2**. Hàm S tăng

**b) Dạng S giảm**

$\mu(x) = 1 - S(x, \alpha, \beta, \gamma)$

**c) Dạng hình chuông**

$$\Pi(x; \gamma, \beta) = \begin{cases} \mathbf{S(x; \gamma - \beta, \gamma - \beta/2; \gamma) \text{ if } x <= \gamma} \\ \mathbf{S(x; \gamma, \gamma + \beta/2; \gamma + \beta) \text{ if } x > \gamma} \end{cases}$$




**Hình 9.3**. Hàm dạng chuông

### 9.2.4. Các phép toán trên tập mờ

Cho ba tập mờ A, B , C với $\mu_A(x)$, $\mu_B(x)$,$\mu_C(x)$

- $C = A \cap B$: $\mu_C(x) = \min(\mu_A(x), \mu_B(x))$
- $C = A \cup B$: $\mu_C(x) = \max(\mu_A(x), \mu_B(x))$
- $C = \neg A$ : $\mu_C(x) = 1 - \mu_A(x)$

## 9.3. CÁC HỆ THỐNG MỜ

❖ **Hàm thành viên cho các biến rời rạc**

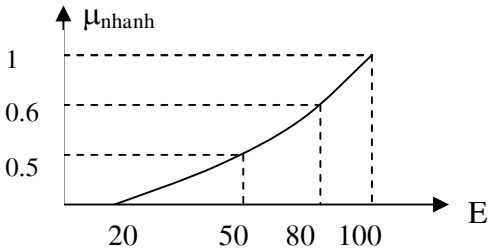
Cho tập vũ trụ E = Tốc độ = { 20,50,80,100 } đơn vị là km/g.

a. Xét tập mờ F = Nhanh xác định bởi hàm thành viên

$\mu_{\text{nhanh:}}$ E ——> **[ 0,1 ]**

x1 ——> $\mu_{nhanh}(X)$

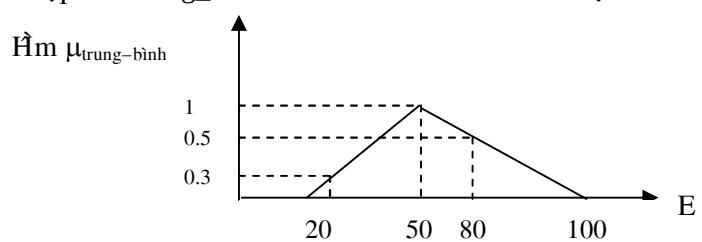
Khi ta gán $\mu_{nhanh}(20) = 0$ nghĩa là tốc độ 20 km/g được xem như là không nhanh.




theo nguyên tắc đó tập mờ nhanh = { (20,0), (50,0.5), (80,0.6), 100, 1) } hay vắn tắt hơn Nhanh = { 0,0.5,0.6,1 }

Vậy hàm thành viên đánh giá mức độ đúng của các tốc độ trong tập vũ trụ E với khái niệm nhanh. Hàm này có tính chủ quan và do kinh nghiệm hay do thực nghiệm.

b. Xét tập mờ trung_bình với hàm thành viên xác định như sau:




thì tập Trung Bình = { 0.3,1,0.5,0 }

❖ Hàm thành viên trong không gian các biến liên tục

Chẳng hạn như các tập mờ Nhanh và Trung bình ở trên có thể định nghĩa như là các hàm

$\mu_{nhanh}(x) = (x/100)^2$

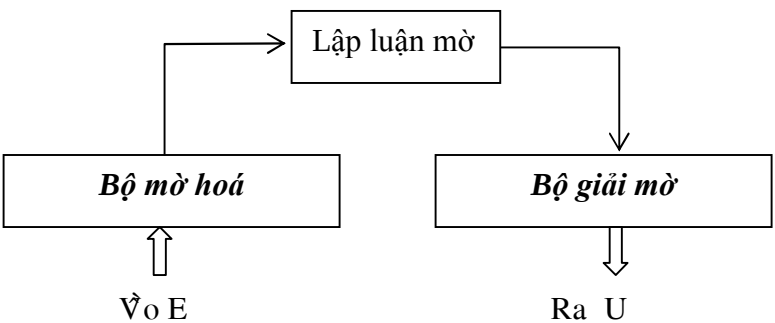
$\mu_{trung–bình}(x) = \{ 0$ if x<=20

$(x–20)/30$ if 20<=x<=50

$(100–x)/50$ if 50<=x<=100 }

Trong phần sau chỉ xét các hàm thành viên có biến liên tục

## 9.4. NGUYÊN LÝ XỬ LÝ CÁC BÀI TOÁN MỜ




**Hình 9.4**. Hệ thống mờ

Các dữ liệu nhập qua bộ mờ hoá để biến thành các trị mờ. Sau đó các giá trị mờ được đưa vào bộ lập luận mờ. Các kết quả là các giá trị mờ ứng với phần kế luật. Bộ giải mờ sẽ biến đổi trị mờ trở lại trị rõ.

### 9.4.1. Bài toán 1

Dữ liệu Input là các giá trị rõ.

**Ví dụ:** Xét bài toán mờ xác định bởi các luật sau:

***Luật 1***: **if** x **is** $A_1$ **and** y **is** $B_1$ **Then** z **is** $C_1$

***Luật 2***: **if** x **is** $A_2$ **or** y **is** $B_2$ **Then** z **is** $C_2$

Vào: trị $x_0$, $y_0$

Ra : trị $z_0$ tương ứng

Bài toán được giải quyết như sau:

Ứng với tập mờ $A_1$ ta có hàm thành viên $\mu_{A1}(x)$

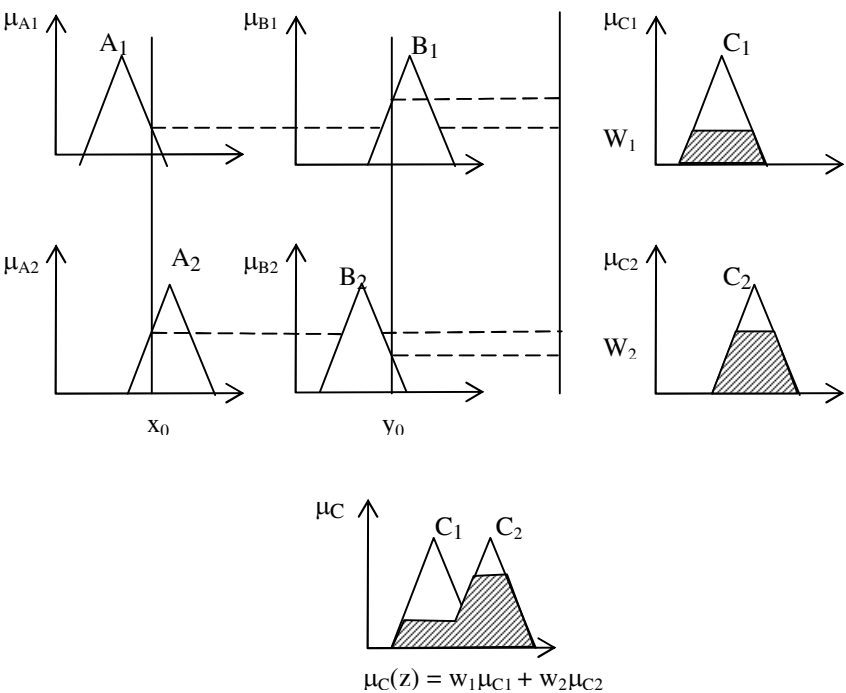
Ứng với tập mờ $A_2$ ta có hàm thành viên $\mu_{A2}(x)$

Ứng với tập mờ $B_1$ ta có hàm thành viên $\mu_{B1}(y)$

Ứng với tập mờ $B_2$ ta có hàm thành viên $\mu_{B2}(x)$

Ứng với tập mờ $C_1$ ta có hàm thành viên $\mu_{C1}(x)$

Ứng với tập mờ $C_2$ ta có hàm thành viên $\mu_{C2}(x)$

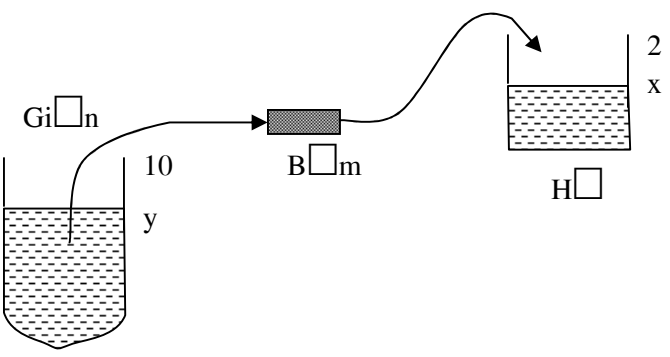


Vấn đề là khi cho các giá trị Input $x = x_0$ và $y = y_0$ hãy tìm hàm thành viên của các luật theo hình vẽ $W_1$ là min của hai giao điểm, $W_2$ là max của hai giao điểm: chúng gọi là trọng các luật. Khi đó hàm thành viên của kết luận là:

$$\mu_C(z) = \sum W_{i}\mu_{K1i}(z) \quad i = 1\ldots N$$

với $\mu_{K1i}(z)$ là hàm thành viên của kết luận cho luật thứ i. Từ đó suy ra trị Output $z_0$ là hệ thống mờ trên.

**Ví dụ:** Giải bài toán điền khiển tự động mờ cho hệ thống bơm nước lấy nước từ giếng. Trong khi hồ hết nước và trong giếng có nước thì máy bơm tự động bơm. Biến ngôn ngữ x, y, z.




| | H.Đầy | H.Lưng | H.Cạn |
|---|---|---|---|
| N.Cao | 0 | B.Vừa | B.Lâu |
| N.Vừa | 0 | B.Vừa | B.HơiLâu |
| N.Ít | 0 | 0 | 0 |

Với biến ngôn ngữ Hồ có các tập mờ hồ đầy (H.Đầy), hồ lưng (H.Lưng) và hồ cạn (H.Cạn).

Với biến ngôn ngữ Giếng có các tập mờ nước cao (N.Cao), nước vừa (N.Vừa), nước ít (N.Ít).

Với biến ngôn ngữ kết luận xác định thời gian bơm sẽ có các tập mờ bơm vừa (B.Vừa), bơm lâu (B.Lâu), bơm hơi lâu(B.HơiLâu).

Các tập mờ trên được xác định bởi các hàm thành viên sau:

Hàm thành viên của Hồ nước:

- H.Đầy(x) = x/2 0<=x<=2
- H.Lưng(x) = { x if 0<=x<=1
  (2–x) if 1<=x<=2 }
- H.Cạn(x) = 1–x/2 0<=x<=2

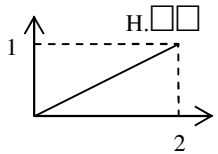
Hàm thành viên cho giếng:

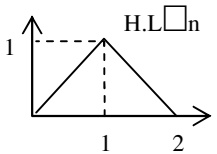
- N.Cao(y) = y/10 0<=y<=10
- N.Vừa(y) = { y/5 if 0<=y<=5
  (10–y)/5 5<=y<=10 }
- N.Ít(y) = 1–y/10 0<=y<=10

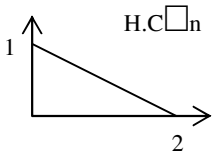
Hàm thành viên của kết luận cho từng luật:

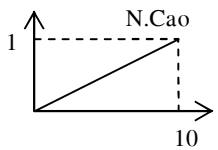
- B.Vừa(z) = { z/15 if 0<=z<=15
  30–z)/15 if 15<=z<=30 }
- B.lâu(z) = z /30 0<=z<=30
- B.Hơi lâu(z) = {z/20 if 0<=z<=20
  1+0.05(z–20) if 20<=z<=30 }

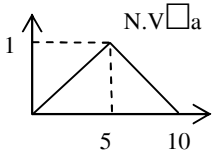
**trong đó x chỉ độ sâu của Hồ (0<=x<=2), y chỉ độ sâu của Giếng (0<=y<=10) và z chỉ thời gian bơm (0<=z<=30).**

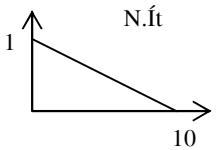


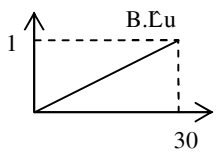


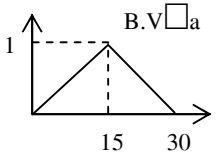


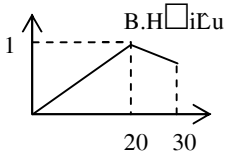













Từ bảng trên ta có các luật:

- ***Luật 1***: **if** x **is** H.Lưng **and** y **is** N.Cao **Then** z **is** B.Vừa
- ***Luật 2***: **if** x **is** H.Cạn **and** y **is** N.Cao **Then** z **is** B.Lâu
- ***Luật 3***: **if** x **is** H.Lưng **and** y **is** N.Vừa **Then** z **is** B.Vừa
- ***Luật 4***: **if** x **is** H.Cạn **and** y **is** N.Vừa **Then** z **is** B.Hơi lâu

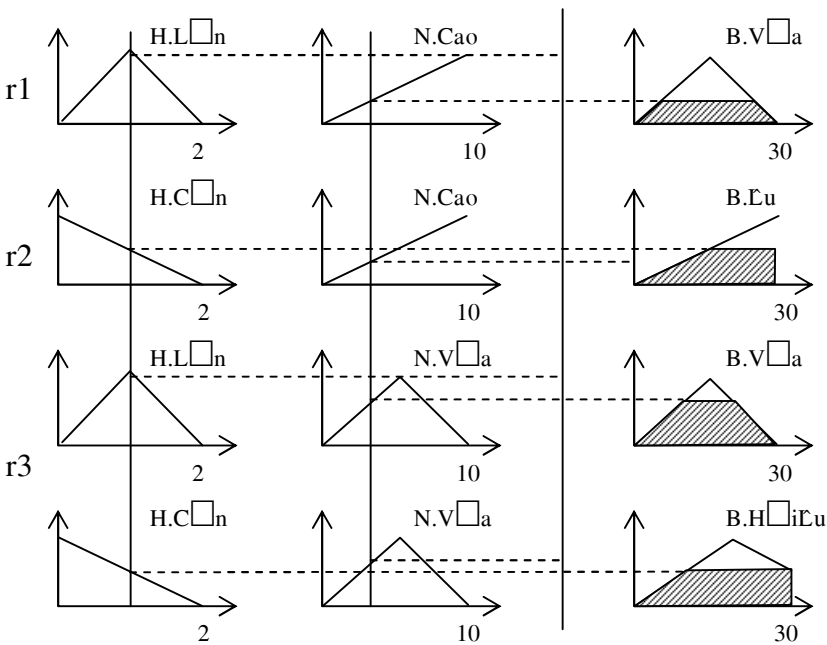
Bây giờ nếu ta nhập trị Input x0 = 1 (Độ cao của nước trong hồ ), y0 = 3 (Độ cao của nước trong giếng)

$\mu_{H.L\square ng}(x_0) = 1$
$\mu_{N}Cao(y_0) =$
$\Rightarrow W_1 = \min\{1,3/10\}$

$\mu_{H.C\square n}(x_0)=0.5$
$\mu_{N\ V\square a}(y_0)=3/5$
$\Rightarrow W_2 = \min\{0.5,3/5\} = 0.5$

$\mu_{H.L\square ng}(x0)=1$
$\mu_{N\ V\square a}(y0)=3/5$
$\Rightarrow W_3 = \min\{1,3/5\} = 3/5$

$\mu_{H.C\square n}(x_0)=0.5$
$\mu_{N\ V\square a}(y_0)=3/5$
$\Rightarrow W_4 = \min\{0.5,3/5\} = 0.5$




Các Wi gọi là các trọng số của luật thứ i

Theo lý thuyết hàm thành viên của kết luận cho bởi công thức:

$\mu_C(z) = \Sigma Wi_{\mu K1i}(Z)$  i = 1 …N

$\mu_C(z)$ = $W_1$.B.Vừa(z) + $W_2$.B.Lâu(z) + $W_3$.B.Vừa(z) + $W_4$.B.Hơi Lâu(z)

$\mu_C(z)$ = 3/10.B.Vừa(z) + 0.5.B.Lâu(z) + 3/5.B.Vừa(z) + 0.5.B.HơiLâu(z)

Bước tiếp theo là ta phải giải mờ từ hàm thành viên của kết luận bằng cách tính trọng tâm của hàm $\mu_C(z)$

Moment $\mu_C(z)$ là $\int_0^{30} z.\mu_c(z)dz = 17.12$

và $\int_0^{30} \mu_c(z)dz = 2.3$

Vậy Defuzzy(z) =17.12/2.3=8.15

Do đó nếu mực nước trong hồ và giếng là 1m và 3m thì thời gian cần bơm là 8 phút và 15 giây.

### 9.4.2. Bài toán 2

## Khi các trị Input là các tập mờ thì bài toán trên được giải quyết như thế nào?

Xét ví dụ sau:

*Luật* : **If** trời nắng **Then** mở khẩu độ nhỏ

*Sự kiện* : Trời rất nắng

*Kết luận* : Mở khẩu độ bao nhiêu?

Trong trường hợp này trị Input là một tập hợp mờ Rất Nắng trong trường hợp biến liên tục nó được xác định qua hàm thành viên của nó.

❖ Các gia tử trên tập mờ

Cho F là tập mờ trong tập vũ trụ E

Ta có các tập mờ phát sinh từ F như sau:

Very F =$F^2$

More or less F=$F^{1/2}$

**Plus F= $F^{1.25}$**

**Ví dụ:** nếu F ={0,0.1,0.5,1 } thì very F=$F^2$ = {0,0.01,0.25,1 }

Để giải quyết bài toán 2 ta xét mô hình sau:

Cho bài toán mờ xác định bởi các quy luật

***Luật 1*** : **if** x **is** $A_1$ **and** y **is** $B_1$ **Then** z **is** $C_1$

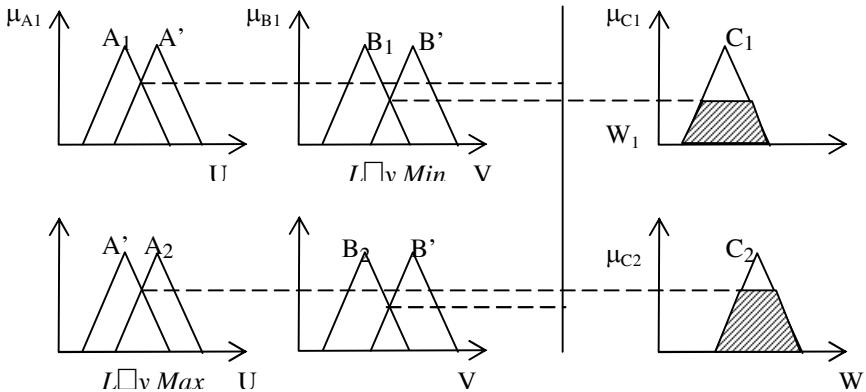
***Luật 2*** : **if** x **is** $A_2$ **and** y **is** $B_2$ **Then** z **is** $C_2$

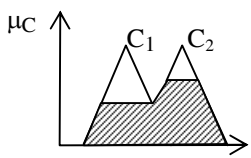
*Input* **: x = A' và y = B'**

(A' có thể là very A, more or less A, plus A … Cũng vậy cho B)

***Kết luận*** : Trị rõ của Output là bao nhiêu?

Giả sử hàm thành viên của các luật trên có dạng:




*Hàm $\mu_C$ của kết luận* $\mu_C = w_1\mu_{C1} +$

Trong luật 1 ta tính trị min của giao điểm của đồ thị $A_1$ và A' với giao điểm của đồ thị $B_1$ và B' trị min này làm trọng$W_1$ cho luật 1.

Tương tự cho luật 2 nhưng lần này ta lấy max (vì toán tử or) ta tìm được trọng $W_2$

Khi ấy hàm thành viên của Kết luận sẽ là:

$$\mu_C(z) = \Sigma Wi_{\mu K1i}(z) \quad i = 1 \ldots N$$

**Cuối cùng dùng công thức mờ ta được trị rõ.**

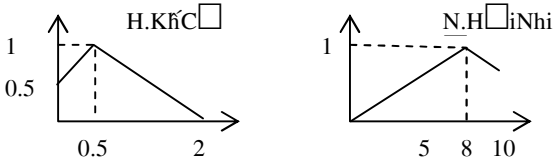
**Ví dụ:** Trong bài toán 1 nếu ta cho dữ liệu Input là các tập mờ như:

x **is** H.Khá Cạn (Hồ khá cạn)

y **is** N.Hơi nhiều (Nước trong giếng hơi nhiều)

Giả sử các tập mờ này được xác định bởi các hàm thành viên là:

- $\mu_{H.KhaCaïn}(x)$ = { x+0.5 if 0<=x<=0.5
  ( 2–x)/1.5 if 0.5<=x<=2 }
- $\mu_{N.}$HơiNhiều(x) = {y if 0<=y<=8
  1+0.25(8–y) if 8<=y<=10 }

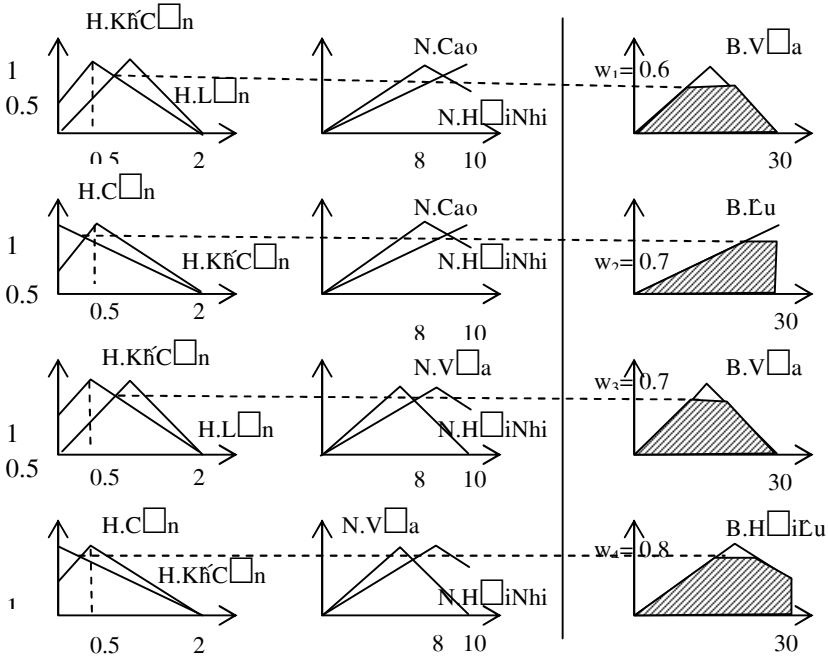



Tìm các trọng Wi cho từng luật (lấy min của các giao điểm)

Xây dựng hàm thành viên của kết luận

$\mu_C(z) = W_1$.B.Vừa(z) + $W_2$.B.Lâu(z) + $W_3$.B.Vừa(z) + $W_4$.B.HơiLâu(z)

Cuối cùng là giải mờ để tìm trị rõ $z_0$




*Tóm lại :* ***Muốn giải các bài toán mờ ta có các bước:***

**B1:** Xác định các luật mờ của bài toán

**B2:** Xác định các hàm thành viên của các tập mờ có trong luật

**B3:** Tìm các trọng Wi của từng luật

**B4:** Nhập trị Input và tìm hàm thành viên cho kết luận

$\mu_C(z) = \Sigma_{iWi\mu K1i}(z)$

*B5:* ***Giải mờ để được giá trị rõ***

**Chú thích:**

1. Hàm thành viên cho kết luận có thể tính bằng công thức:

a) $\mu_C(z) = \Sigma_{iWi\mu K1i}(z) \quad \forall x \in E$

b) $\mu_C(z) = \Sigma_{iMin}(Wi_{,\mu K1i}(z)) \quad \forall x \in E$

c) $\mu_C(z) = \Sigma_{iMax}(Min(Wi_{,\mu K1i}(z)) \quad \forall x \in E$

2. Giải mờ ta có thể áp dụng một trong hai phương pháp sau:

a) Tìm trọng tâm

b) Tìm trị trung bình

Defuzzy(z) =$(\sum_i \alpha_i.Wi)/ \sum_i \alpha_i$

trong đó $\alpha_i$ là khoảng tin cậy của đồ thị của luật thứ i

Wi là trọng số của luật thứ i

# TÀI LIỆU THAM KHẢO


[1]. Bạch Hưng Khang, Hoàng Kiếm. *Trí tuệ nhân tạo, các phương pháp và ứng dụng*. Nhà xuất bản Khoa học và Kỹ thuật, 1989.

[2]. Đỗ Trung Tuấn. *Trí tuệ nhân tạo*. Nhà xuất bản Giáo dục, 1998

[3]. Đỗ Trung Tuấn. *Hệ chuyên gia*. Nhà xuất bản Giáo dục. 1999

[4]. Đỗ Phúc. *Các đồ án môn học Cơ sở Tri thức*. Khoa công nghệ thông tin - Đại học Khoa học tự nhiên, 1998.

[5]. Rich Elaine. *Artificial Intelligence*. Addison Wesley, 1983.

[6]. John Durkin. *Expert Systems-design and development.* Prentice Hall International, Inc, 1994.

[7]. Adrian A. Hopgood, *Knowledge-based systems for engineers and scientists*. CRC Press, 1993.

[8]. Stuart Russell & Peter Norvig. *Artificial Intelligence – a modern approach*. Prentice Hall, 1995

[9]. Kurt Sundermeyer. *Knowledge-based systems*. Wissenschafs Verlag, 1991.

[10]. Mehmet R. Tolun & Saleh M. Abu-Soud. *An Inductive Learning Algorithm for Production Rule Discovery*. IEEE.

[11]. Patrick Henry Winston. *Artificial Intelligence*. Addison Wesley, 1992.