- *Ví dụ*:
Frame MáyTínhCáNhân;
{ Là một dạng riêng của MáyTínhĐiệnTử;
  KiếnTrúc {8 bits, 16 bits, 32 bits};
  SốỔĐĩa {1, .., 5};
  MànHình Thủ tục xác định tính năng của màn hình;
}

Ngoài ra, người ta còn có thể biểu diễn tri thức bằng *bộ ba liên hợp OAV* (Object - Attribute – Value), *mạng nơron (ANN* - Artificial Neural Network), …

III.3. Xử lý tri thức tất định bằng phương pháp suy diễn logic

III.3.1. Các cơ chế lập luận với tri thức tất định

- *Suy diễn* (Deduction, diễn dịch): modus ponens, modus tollens, các kiểu suy diễn: tiến và lùi.
- *Qui diễn* (Abduction, tương tự): Cho trước: $A \rightarrow B$.
  . $A \sim A' \Rightarrow A' \rightarrow B$
  . $B \sim B' \Rightarrow A \rightarrow B'$
  . $A \sim A'$ & $B \sim B' \Rightarrow A' \rightarrow B'$
- *Qui nạp* (Induction): hoàn toàn và không hoàn toàn

Sau đây ta sẽ xét các thuật toán và thuật giải nhằm chứng minh tự động các biểu thức lôgic mệnh đề.

Bài toán A (BTA):Xét bài toán lôgic mệnh đề sau có hằng đúng hay không:

$$\bigwedge_{1 \le i \le m} GT_i \Rightarrow \bigvee_{1 \le j \le m} KL_j$$

với $\{GT_i , KL_j\}$ là các biểu thức lôgic mệnh đề bất kỳ.

Tại sao ta chỉ xét bài toán dạng trên đây (*bài tập)* ?

III.3.2. Thuật toán Vương Hạo (Wong Havard) giải BTA

* *Ý tưởng*: Áp dụng chiến lược "*Chia để trị*" nhằm tách bài toán xuất phát thành các bài toán con dạng "VÀ" đơn giản hơn. Bài toán ban đầu sẽ được giải khi và chỉ khi mọi bài toán con sơ cấp giải được.

* Trước tiên, ta đưa vế *trái VT* (hay vế *phải VP*) về dạng *chuẩn hội (*hay *chuẩn tuyển* tương ứng) bằng cách:

. Thay các phép toán tương đương $\leftrightarrow$ (nếu có) bởi các phép toán kéo theo $\rightarrow$

. Thay các phép toán $\rightarrow$ bởi các phép toán $\neg, \vee$

. Dùng các luật *De Morgan* để bỏ các dấu $\neg$ của nguyên một nhóm mệnh đề

. Dùng các *luật phân phối* (nếu chưa gặp dạng chuẩn cần tìm) của *phép tuyển đối với phép hội* (hay của *phép hội đối với phép tuyển* tương ứng).

Trong thuật toán Wong, sau khi đưa VT về dạng chuẩn hội và VP về dạng chuẩn tuyển, ta sẽ sử dụng các qui tắc sau (hãy chứng minh tại sao ta có quyền sử dụng chúng ? *Bài tập*).

- Thủ tục *Chuyển*(VT,VP): thay các dấu $\wedge$ các nhóm chính bên VT và các dấu $\vee$ các nhóm chính bên VP bởi dấu phẩy; chuyển vế các mệnh đề chính ở dạng phủ định $\neg$p từ vế này sang vế kia và bỏ đi dấu $\neg$ (chỉ còn p). Nói cách khác, ta có thể xem dấu phẩy (,) bên VT là phép hội và dấu phẩy (,) bên VP là phép tuyển.

. <u>*Ví dụ 3*</u>: Kiểm tra tính hằng đúng của biểu thức lôgic mệnh đề sau:

$VT \equiv (\neg a) \wedge (\neg b \vee c) \quad \Rightarrow \quad (a \wedge b) \vee (\neg b) \vee c \equiv VP$

Các *nhóm chính* trong: vế trái VT là $(\neg a)$ và $(\neg b \vee c)$, vế phải VP là $(a \wedge b)$, $(\neg b)$ và (c).

$$VT \equiv (\neg a) \wedge (\neg b \vee c) \quad \Rightarrow \quad (a \wedge b) \vee (\neg b) \vee c \equiv VP$$

Chuyển

$$\neg a, \neg b \vee c \quad \Rightarrow \quad a \wedge b, \neg b, c$$

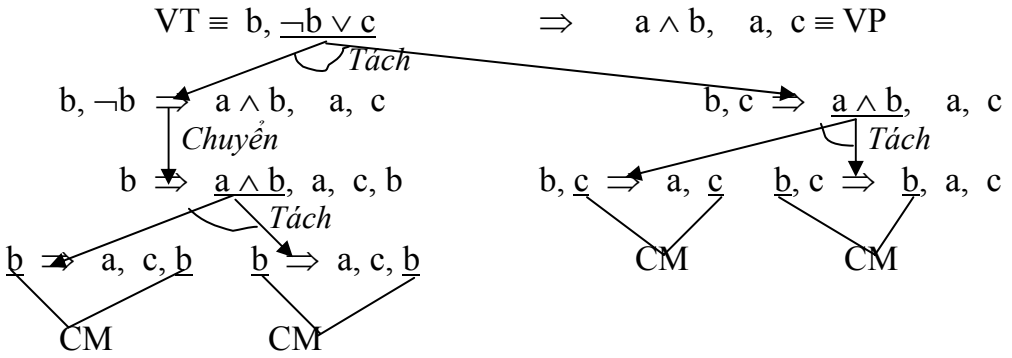
Chuyển

$$VT \equiv b, \neg b \vee c \quad \Rightarrow \quad a \wedge b, \quad a, \ c \equiv VP$$

- *Hàm lôgic Tách*(VT,VP): tách mỗi *nhóm tuyển $\vee$ chính trong VT* (hay *mỗi nhóm hội $\wedge$ chính trong VP)* thành *nhiều bài toán con dạng VÀ*. Mỗi bài toán con gồm một mệnh đề trong nhóm chính và giữ nguyên các nhóm chính khác. Nếu tách được thì thủ tục Tách(VT,VP) nhận giá trị *true*; nếu ngược lại, nó nhận trị *false*. Với ví dụ 3 trên đây, ta có sơ đồ biến đổi như *hình III.3.2.1* sau đây.

- Với mỗi bài toán con, nếu mỗi vế của nó *có một mệnh đề đơn đứng độc lập giống nhau thì nó được chứng minh (hằng đúng)*. *Bài toán xuất phát chỉ <u>được chứng minh</u> (hằng đúng) khi <u>mọi</u> bài toán con của nó được chứng minh*. Nói cách khác, *bài toán xuất phát <u>không hằng đúng</u> nếu có <u>tồn tại</u> một bài toán con của nó*

không hằng đúng (không thể áp dụng qui tắc tách cũng như chuyển được nữa và hai vế không có chung một mệnh đề đơn đứng độc lập nào cả).



(Hình III.3.2.1)

Vậy bài toán nêu ra trong ví dụ 3 là hằng đúng.

* ***Thuật toán Vương Hạo***

```
{       VT = VP = ∅ ;
        for  i = 1 to m  do { ChuẩnHội(GTi);      VT ← VT U {GTi }; }
        for  i = 1 to n  do { ChuẩnTuyển(KLi);  VP ← VP U {KLi }; }
        P ← {(VT,VP)};
        while  (P≠∅)  do
        {   (VT,VP) ← get(P);
            if  (VT ∩ VP = ∅) then
            {   Chuyển(VT,VP);
                if  (VT ∩ VP = ∅)  then
                    if  (not(Tách(VT,VP)))
                        then  exit("Không thành công"); // BT không hằng đúng
            }
        }
        write("Thành công");                          // BT hằng đúng
}
```

* *Nhận xét*:

. Thuật toán Wong *dừng sau một số hữu hạn bước* và cho ra thông báo "Thành công" nếu và chỉ nếu từ các giả thiết GT_1, ..., GT_m có thể suy ra một trong các kết luận KL_1, ..., KL_n.

. Nếu số các phép toán liên kết $\vee$ trong GT_i và $\wedge$ trong KL_j là M thì thuật toán sẽ sinh ra từ M đến 2^M dòng (VT,VP) $\in$ P (bùng nổ tổ hợp nếu M khá lớn !).

* *Ví dụ 4 (biểu thức hằng đúng)*: Có thể chứng minh rằng:

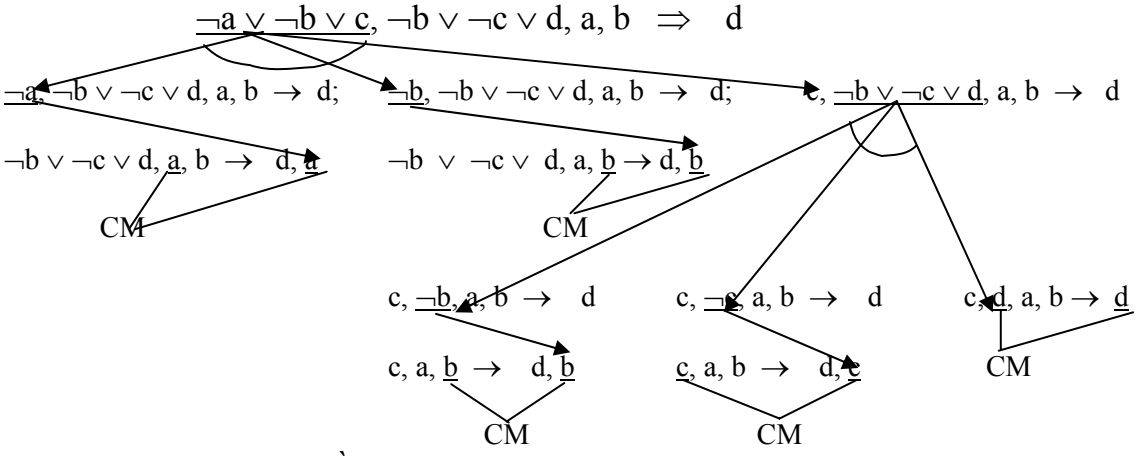
$$[(a \wedge b \rightarrow c) \wedge (b \wedge c \rightarrow d) \wedge a \wedge b] \Rightarrow d \ ?$$

Đưa VT về dạng chuẩn hội và VP về dạng chuẩn tuyển:

VT ≡ {¬a ∨ ¬b ∨ c, ¬b ∨ ¬c ∨ d, a, b}: dạng chuẩn hội

VP ≡ {d}: dạng chuẩn tuyển

Ta có thể *biểu diễn quá trình giải* của thuật toán Wong thông qua *đồ thị suy diễn* hay *đồ thị lời giải* như sau*:*



(*Đồ thị lời giải trong thuật toán Wong)*

* *Ví dụ 5 (biểu thức không hằng đúng)*: Kiểm tra tính hằng đúng của biểu thức:

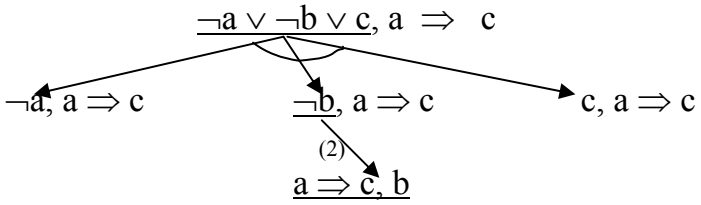
$$[(a \wedge b \rightarrow c) \wedge a] \Rightarrow c \ ? \qquad (1)$$

Đưa VT về dạng chuẩn hội và VP về dạng chuẩn tuyển:

VT ≡ {¬a ∨ ¬b ∨ c, a}: dạng chuẩn hội

VP ≡ {c}: dạng chuẩn tuyển

Đồ thị lời giải:



BT con *(2)* sai vì không thể tách, chuyển và không có mệnh đề chung. Do đó bài toán *(1)* xuất phát cũng sai.

III.3.3. Thuật toán Robinson giải BTA

* *Ý tưởng*: Sử dụng:

. phương pháp *chứng minh phản chứng*:

$[a \rightarrow b$ *đúng*$] \equiv [a \wedge \neg b$ *sai* hay *mâu thuẫn*$]$

. *nguyên lý hợp giải*:

$(\neg a \vee b) \wedge (a \vee c) \Rightarrow b \vee c.$

(V*ì sao? Hãy chứng minh*).

- Ta thường gặp các trường hợp riêng của nguyên lý này:

. nguyên lý *modus ponens*:

$a \wedge (\neg a \vee b) \Rightarrow b$

. nguyên lý *modus tollens:*

$(\neg b \wedge (a \rightarrow b)) \Rightarrow \neg a$

- Để chứng minh từ các giả thiết GT_1, ..., GT_m suy ra một trong các kết luận KL_1, ..., KL_n, ta chỉ cần *lấy phủ định của KL_1, ..., KL_n đưa về cùng với các giả thiết*:

$$\mathbf{P \equiv (\bigwedge_{1 \le i \le m} GT_i) \wedge (\bigwedge_{1 \le j \le m} \neg KL_j)}$$

Nếu *suy ra được mâu thuẫn* từ tập các mệnh đề P thì quá trình chứng minh kết thúc và kết luận *BTA là hằng đúng*.

- Để suy ra được mâu thuẫn, Robinson đã đưa ra nguyên lý hợp giải trên đây để *bổ sung thêm càng ngày càng nhiều các biểu thức mệnh đề trung gian mới* cho đến khi nào trong P *có 2 mệnh đề đơn đứng độc lập là phủ định của nhau thì mâu thuẫn xảy ra*.

* ***Thuật toán Robinson*** *(giải BTA)*

```
{       P = ∅;
        for  i = 1 to m  do { GTi ← ChuẩnHội(GTi);     P  ← P U GTi; }
        for  i = 1 to n do {NotKLi ← ChuẩnHội(¬KLi); P ← P U NotKLi;}
        if  (MâuThuẫn(P)) then exit("Thành công");
        P1 = ∅;
        while  (P≠ P1  and not(MâuThuẫn(P)))  do {P1 =  P; HợpGiải(P);}
        if  (MâuThuẫn(P))
        then exit("Thành công");
        else  exit("Không thành công");
```

}

trong đó, *hàm lôgic MâuThuẫn*(P) và thủ tục *HợpGiải*(P) có nội dung như sau:

Function MâuThuẫn*(P): boolean*

{ **for each** p ∈ P **do**
  for each q ∈ P and q ≠ p **do**
    if (p = ¬ q or q = ¬ p) **then** return(*True*);
  return(*False*);
}

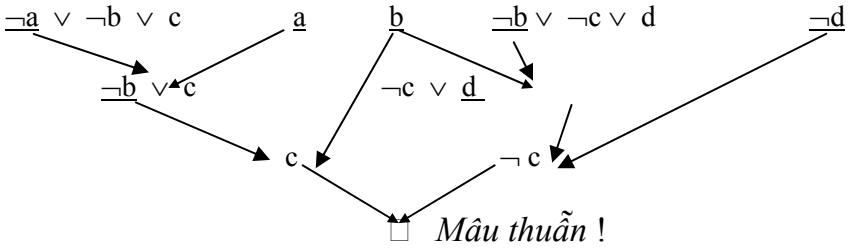
Procedure HợpGiải*(P)*

{ **for each** p ∈ P **do**
  for each q ∈ P and q ≠ p **do**
    if (p = ¬ a ∨ b and q = a ∨ c) **then** P ← P U {b ∨ c};
}

* *Ví dụ 6 (biểu thức hằng đúng)*: Xét bài toán trong *ví dụ 4*, ta có:

P ≡ {¬a ∨ ¬b ∨ c, ¬b ∨ ¬c ∨ d, a, b, ¬d}

Để tiện theo dõi, ta biểu diễn quá trình thực hiện thuật toán qua *đồ thị hợp giải* như sau:



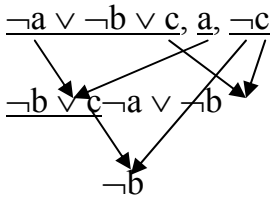
* *Ví dụ 7 (biểu thức không hằng đúng)*: Kiểm tra tính hằng đúng của biểu thức:

[(a ∧ b → c) ∧ a] ⇒ c ?

Lấy phủ định của kết luận và đưa về dạng chuẩn hội, ta có:

P ≡ {¬a ∨ ¬b ∨ c, a, ¬c}

Đồ thị hợp giải:



Không thấy mâu thuẫn cũng không còn cách nào hợp giải mà có thể sinh ra mệnh đề mới nữa. Vậy bài toán xuất phát sai.

* <u>*Nhận xét*</u>:

- Thủ tục Robinson sẽ *dừng sau hữu hạn bước* và cho ra thông báo "Thành công" nếu và chỉ nếu $GT_1 \wedge ... \wedge GT_m \Rightarrow KL_1 \vee ... \vee KL_n$.
- Để *chứng minh BTA không hằng đúng*, ta phải *hợp giải hết tất cả các khả năng cho đến khi không thể sinh ra thêm biểu thức mệnh đề nào mới* trong P.
- Cũng như thủ tục Wong H., thủ tục Robinson có *nhược điểm* là *tùy theo thứ tự lấy các cặp mệnh đề để hợp giải* có thể *xảy ra hiện tượng tràn bộ nhớ (do bùng nổ tổ hợp)* đối với các bài toán có kích thước lớn.
- Có thể *áp dụng một số heuristic* cho thuật toán trên để thu được *"thuật giải Robinson"*, chẳng hạn: khi áp dụng nguyên lý hợp giải, sau khi thêm vào tập P mệnh đề $(b \vee c)$, có thể bỏ hai mệnh đề $(\neg a \vee b)$ và $(a \vee c)$ ra khỏi P.

* <u>*Ví dụ 8*</u> *(biểu thức hằng đúng nhưng có thể kết luận sai nếu áp dụng heuristic không đúng)*: Kiểm tra tính hằng đúng của biểu thức:

$$[a \wedge (a \rightarrow b) \wedge (a \rightarrow c)] \Rightarrow c \qquad (3)$$

Lấy phủ định của kết luận và đưa về dạng chuẩn hội, ta có:

$$P \equiv \{\neg a \vee b, \neg a \vee c, a, \neg c\}$$

. Nếu hợp giải trước hai mệnh đề $\neg a \vee b$ và a để được b rồi loại chúng thì trong P còn lại:

$$P \equiv \{b, \neg a \vee c, \neg c\}$$

Hợp giải tiếp tục thì: $P \equiv \{b, \neg a\}$: không có mâu thuẫn cũng không thể hợp giải được nữa: bài toán sai chăng ? Thật ra, bài toán *(3)* vẫn đúng !

. Nếu hợp giải trước hai mệnh đề $\neg a \vee c$ và a để được c rồi loại chúng thì trong P còn lại:

$$P \equiv \{c, \neg a \vee b, \neg c\}$$

Khi đó mâu thuẫn xảy ra giữa c và $\neg c$. Vậy bài toán xuất phát đúng !

* <u>*Chú ý*</u>: Để tránh tình trạng kết luận sai như trên, khi *thay thuật toán Robinson thành "thuật giải Robinson*", ta nên chú ý rằng ngoài việc *thêm heuristic trên vào thủ tục hợp giải, ta phải bỏ đi thông báo exit("Không thành công")*. Nếu xuất hiện mâu thuẫn thì kết luận "Thành công" và dừng. Nếu *không xuất hiện mâu thuẫn* thì không được thông báo exit("Không thành công"), nghĩa là *không được quyền kết luận gì* khi tình trạng này xảy ra !

* ***<u>Bài toán B (BTB)</u>***

Input: - Tập các mệnh đề giả thiết: $GT = \{g_1, ..., g_n\} = \wedge_{1 \le i \le n} g_i$

- Tập RULE gồm *m* luật có dạng chuẩn r : $p_1 \wedge ... \wedge p_n \rightarrow q$
- Tập các mệnh đề kết luận: KL = $\{q_1, ..., q_k\} = \wedge_{1 \le i \le k} q_i$

Output: Thông báo "Thành công" nếu *mọi* $q_i \in KL$ có thể suy diễn từ GT nhờ sử dụng tập luật RULE.

- Để chứng minh các bài toán lôgic mệnh đề, người ta còn dùng phương pháp *suy diễn tiến* (hay *lùi*), bằng cách *xuất phát từ giả thiết* (hay *kết luận* tương ứng), dựa trên các nguyên lý suy diễn:

. *Modus ponens:* $\dfrac{A,\ A \rightarrow B}{B}$

. *Modus tollens:* $\dfrac{\neg B,\ A \rightarrow B}{\neg A}$

ta *thêm vào các mệnh đề mới được chứng minh đúng* (hay các mệnh đề *cần phải chứng minh thêm* tương ứng) cho đến khi thu được *kết luận* (hay *giả thiết* tương ứng) thì dừng.

III.3.4. Thuật toán suy diễn tiến giải BTB

* *Ý tưởng*: Quá trình *suy diễn tiến bắt đầu từ tập TrungGian các mệnh đề đúng xuất phát* (tập giả thiết*)* và nó *sẽ được "làm nở" dần bằng cách thêm vào các sự kiện mới đúng* nhờ các luật suy diễn trong RULE*, cho đến khi các kết luận cần chứng minh được phát hiện.*

*** *Thuật toán suy diễn tiến SDT***

```
{       TrungGian = GT;
        THỎA   = Lọc(RULE, TrungGian);
      // THỎA là tập các luật r có dạng p1∧ ... ∧pn → q mà pi∈TrungGian, ∀i=1..n
        while (KL ⊄ TrungGian and THỎA ≠ ∅) do
          {     r ← get(THỎA);      // r ∈ THỎA có dạng r: p1∧ ... ∧pn → q
                TrungGian ← TrungGian U {q};
                RULE ← RULE \ {r};
                THỎA = Lọc(RULE, TrungGian)
          }
        if (KL ⊂ TrungGian)
        then write("Thành công")
        else write("Không thành công");
}
```

* *Ví dụ 9*: Cho trước tập các sự kiện giả thiết: GT = {a, b}. Sử dụng tập RULE các luật:

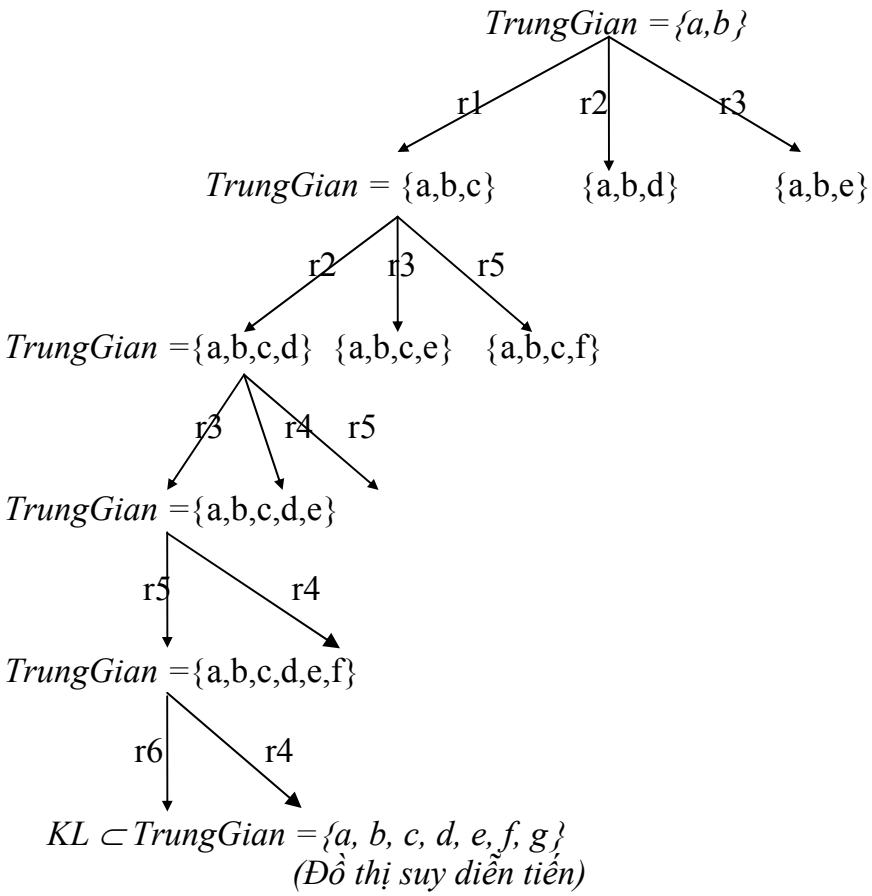
r1: $a \rightarrow c$

r2: $b \rightarrow d$

r3: $a \rightarrow e$
r4: $a \wedge d \rightarrow e$
r5: $b \wedge c \rightarrow f$
r6: $e \wedge f \rightarrow g$
Cần suy ra kết luận: KL = {g}.

. Ta có: TrungGian = {a, b}; RULE = {r1-r6}. Do đó THỎA = {r1,r2,r3}.
. Áp dụng luật r1: $a \rightarrow c$, ta được TrungGian = {a,b,c} và RULE={r2-r6}.
Do đó: THỎA = {r2, r3, r5}.
. Lúc này, KL = {g} $\not\subset$ {a, b, c} = TrungGian và THỎA $\neq \varnothing$.
Ta lại tiếp tục quá trình, bằng cách chọn r2 để thực hiện, ...

Ta minh họa quá trình trên bằng *đồ thị suy diễn tiến*, trong đó mỗi đỉnh ứng với tập TrungGian tại thời điểm đang xét và sẽ có một cung đi từ đỉnh *TrungGian* đến đỉnh *TrungGian* ***U*** *{q}* tương ứng với luật r đã được lọc và có dạng r : $p_1 \wedge ... \wedge p_n \rightarrow q$ mà $p_i \in$ TrungGian, $\forall i=1,.., n$.



(Đồ thị suy diễn tiến)

* <u>*Ví dụ 10*</u> *(biểu thức không hằng đúng)*: Kiểm tra tính hằng đúng của biểu thức:

$$[(a \wedge b \rightarrow c) \wedge a] \Rightarrow c \ ?$$

TrungGian = GT = {a}, KL = {c}, RULE = {r1: $a \wedge b \rightarrow c$ }.

Khi đó: THỎA = $\varnothing$ và KL $\not\subset$ TrungGian. Vậy bài toán trên không hằng đúng.

* <u>*Nhận xét*</u>:

.Với các thứ tự khác nhau để chọn r từ tập THỎA, ta sẽ có các cơ chế duyệt theo chiều sâu, theo chiều rộng, ... trên đồ thị.

. Nếu trong *BTB* thay tập kết luận KL = $\wedge_{1\le i\le k}\ q_i$ bởi: KL = $V_{1\le i\le k}\ q_i$ thì cần hiệu chỉnh thuật toán SDT ra sao ? (*Bài tập*)

. Để tránh các kết luận sai, với *mỗi biến mệnh đề p* xuất hiện trong biểu thức cần chứng minh, bằng các qui tắc biến đổi tương đương lôgic, chuyển về *cùng dạng p* hoặc *cùng dạng* $\neg p$.

III.3.5. Thuật toán suy diễn lùi giải BTB

* Với suy diễn lùi, để đưa ra kết luận B ta thử tìm tất cả các luật có dạng:

$A_1 \wedge ... \wedge A_n \rightarrow B$

Để có B, ta cần chứng minh A_1 ,... , A_n (các kết luận mới được thêm vào tập kết luận). Quá trình xác định A_i cũng diễn ra tương tự như đối với B. Nếu đến một lúc nào đó tìm thấy một A_{io} nào đó không thể dẫn xuất được từ các giả thiết thì ta quay lui sang luật khác sinh ra B và lại tiếp tục quá trình trên. Nếu không tìm được A_{io} như vậy (nghĩa là mọi A_i đều được dẫn xuất từ giả thiết) thì quá trình dẫn xuất ra B thành công.

Để thực hiện quá trình quay lui, ta sử dụng hai tập có cấu trúc ngăn xếp GOAL và VET: GOAL là tập lưu các mệnh đề cần phải chứng minh đến thời điểm đang xét và VET là tập lưu các luật đã sử dụng để chứng minh các đích (kể cả đích trung gian).

* ***<u>Thuật toán suy diễn lùi SDL</u>***

```
{1 if (KL ⊂ GT)
  then exit("Thành công");
  else {2 GOAL = ∅;
        VET = ∅;
        CMĐược = True;
        for each q ∈ KL do
                    GOAL = GOAL U {(q,0)};
        repeat
```

```
{3  (f,i) ← get(GOAL);
    if (f ∉ GT) then
    {4 Tìm_Luật(f, i, RULE, j);          // Tìm luật r_j : left_j →f
      if (j ≤ m)
      then { VET = VET U {(f,j)};
             for each t ∈left_j\GT do GOAL=GOAL U {(t,0)};
           }
      else {5 back = true;               // dùng biến này để quay lui
             while (f ∉ KL and back) do
             {6 repeat {(g,k)←get(VET);
                  // Lấy luật r_k:left_k→g ra khỏi VET
                  // để quay lui đến luật khác mà cũng →g
                        GOAL = GOAL \ left_k;
                        }
               until (f ∈ left_k);
                  // Bỏ hậu quả của việc chọn sai luật r và dẫn đến việc
                  // không CM được mệnh đề f được giả định cần CM
               Tìm_Luật(g, k, RULE, s); // Tìm luật r_s:left_s →g
               if (s ≤ m)
               then { for each t ∈ left_s \ GT do
                                GOAL = GOAL U {(t,0)};
                      VET = VET U {(g,s)}; back = false;
                    }
               else f = g;
             }6
             if (f ∈ KL and back) then CMĐược = False;
           }5
    }4
  }3
until (GOAL = ∅ or not(CMĐược));
if (not(CMĐược)) then exit("Không Thành công")
else exit("Thành công");
}2
}1
```

Trong thủ tục trên ta sử dụng thủ tục: *Tìm_Luật*(f, i, RULE, k) để tìm xem có luật r_k nào kể từ luật thứ i+1 trở đi mà cũng suy ra được f (r_k:$left_k$ →f). Nếu không có luật nào như thế thì qui ước lấy k = m+1.

* <u>*Ví dụ 11*</u>: Cho trước tập các sự kiện giả thiết GT = {a, b}. Sử dụng tập RULE các luật:

r1. $a \wedge b \rightarrow c$
r2. $a \wedge h \rightarrow d$
r3. $b \wedge c \rightarrow e$
r4. $a \wedge d \rightarrow m$
r5. $a \wedge b \rightarrow o$
r6. $o \wedge e \rightarrow m$

Cần suy ra KL = {m}.

Ban đầu GOAL = VET = ∅;

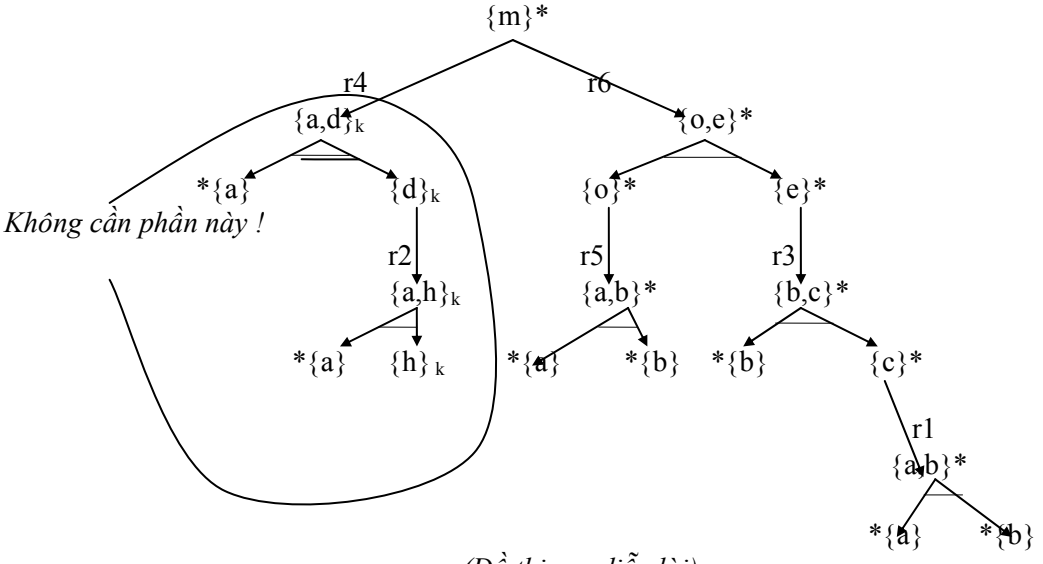
Áp dụng thủ tục *Tìm_Luật*(m, 0, RULE, j), ta được j = 4 (r4 là luật đầu tiên sinh ra m). Khi đó VET = {(m,4)}; GOAL = {(d,0)} (vì a ∈ GT nên chỉ cần xét (d,0)).

Ta tiếp tục quá trình và có bảng theo dõi sau:

GOAL (back)	(f,i)	CMĐ	j	$left_j$\GT	VET	(g,k)	l	$left_l$\GT	Quaylui
{(m,0)}	(m,0)	1	true	d	{(m,4)}				
{(d,0)}	(d,0)		2	h	{(d,2);(m,4)}				
{(h,0)}	(h,0)		7		{(m,4)}	(d,2)	7		true
∅	d				∅	(m,4)	6	o,e	
{(o,0),(e,0)}					{(m,6)}				false(thoát while 6)
{(e,0)}	o		5	∅	{(o,5),(m,6)}				
∅	e		3	c	{(e,3),(o,5),(m,6)}				
{(c,0)}	c		1	∅	{(c,1),(e,3),(o,5),(m,6)}				
∅									

(GOAL = ∅ : thoát vòng while$\{_2\}_2$; *CMĐược = True: Thành công !)*

Ta có thể biểu diễn quá trình suy diễn lùi trên đây thông qua *đồ thị* (VÀ/HOẶC) *suy diễn lùi* như sau:



(Đồ thị suy diễn lùi)

- *Ví dụ 12*: Cho trước tập các sự kiện giả thiết GT = {a}. Sử dụng tập RULE các luật:

r1. $a \wedge b \rightarrow c$
r2. $a \wedge h \rightarrow d$
r3. $b \wedge c \rightarrow e$
r4. $a \wedge d \rightarrow m$
r5. $a \wedge b \rightarrow o$
r6. $o \wedge e \rightarrow m$

Cần suy ra KL = {m}.

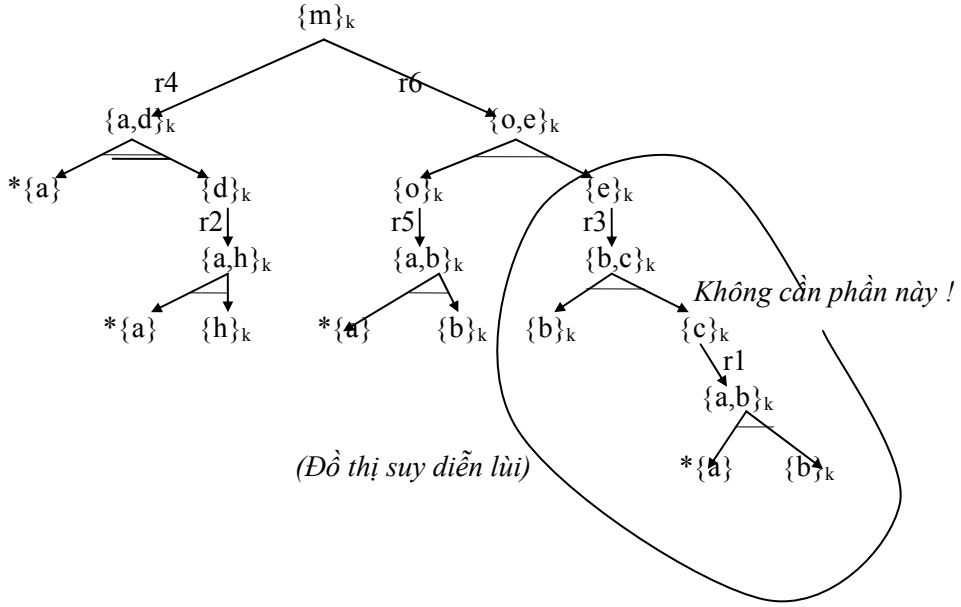
Ban đầu GOAL = VET = ∅. Áp dụng thủ tục *Tìm_Luật*(m, 0, RULE, j), ta được j = 4 (r4 là luật đầu tiên sinh ra m). Khi đó VET = {(m,4)}; GOAL = {(d,0)} (vì a ∈ GT nên chỉ cần xét (d,0)).

Ta tiếp tục quá trình, sẽ có bảng theo dõi sau:

GOAL (back)	(f,i)	CMĐ	j	$left_j$\GT	VET	(g,k)	l	$left_l$\GT	Quaylui
{(m,0)} (m,0)	1	true	d	{(m,4)}					
{(d,0)}	(d,0)		2	h	{(d,2);(m,4)}				
{(h,0)}	(h,0)		7		{(m,4)}	(d,2)	7		true
∅	d				∅	(m,4)	6	{o,e}	
{(o,0),(e,0)}					{(m,6)}			(thoát while 6)	false
{(e,0)}	o		5	b	{(o,5),(m,6)}				
{(b,0),(e,0)}	b		7	∅	{(m,6)}	(o,5)	7		true
∅	o				∅	(m,6)	7		
∅	m	false							

(f = m ∈ KL : thoát while $\{_6\}_6$; do đó: *CMĐược= false (*và GOAL = ∅): thoát vòng while$\{_2\}_2$; *CMĐược= false: Không thành công !)*

Đồ thị suy diễn lùi:



(Đồ thị suy diễn lùi)

* <u>*Nhận xét*</u>:

- Quá trình suy diễn lùi *tương tự như quá trình tìm đồ thị con lời giải trong đồ thị VÀ/HOẶC* biểu diễn tập luật.
- Để *tăng hiệu quả* của thủ tục suy diễn lùi có thể đưa vào 2 tập: tập ĐÚNG chứa các sự kiện đã được khẳng định là đúng và tập SAI chứa các sự kiện đã được khẳng định là sai. Mỗi khi lấy một sự kiện (f, i) nào đó ta cần kiểm tra trước $f \in$ ĐÚNG hay $f \in$ SAI ?

III.4. Xử lý tri thức bất định bằng phương pháp suy diễn logic

III.4.1. Các cơ chế lập luận với tri thức bất định và không chính xác

- *Khái niệm*: Trên thực tế, có nhiều mệnh đề được phát biểu không chính xác, mang tính bất định, *chúng không hẳn hoàn toàn đúng cũng không hoàn toàn sai*. Tính đúng - sai của chúng không được xác định và thể hiện rõ ràng như trong lôgic mệnh đề cổ điển. Ta có thể mở rộng tính đúng sai của mệnh đề *p* thông qua một *hàm μ đo độ đúng* của nó: *μ (p) $\in$ [0; 1]*.

- *Mô hình xác suất*: Xét P là tập các mệnh đề đóng kín đối với các phép toán $\neg$, $\wedge$ (do đó cả $\vee$) và ánh xạ (độ đo xác suất) Pr: $P \to [0, 1]$ thỏa các tính chất:
  - $\Pr(p) + \Pr(\neg p) = 1, \forall p \in P$
  - $\Pr(p \wedge q) = \Pr(p).\Pr(q), \forall p, q \in P$
  - Từ đó: $\Pr(p \vee q) = \Pr(p) + \Pr(q) - \Pr(p \wedge q)$ *(Chứng minh ? Bài tập)*

- <u>*Mô hình khái luật*</u>:

Biểu diễn mệnh đề:

- Nếu p thì q bởi ***luật*** bình thường: $p \to q$
- Nếu p thì *thông thường (nói chung, hầu như, …)* q bởi ***khái luật***: $p \to q$ **!**

III.4.2. Phân bố khả xuất của khái luật và các phép toán nối kết

* <u>*Phân bố khả xuất (PBKX) của khái luật*</u>:

- PBKX của một cặp sự kiện đối ngẫu nhau p và $\overline{p}$ (hay $\neg$p): $\pi(p)$, $\pi(\overline{p})$ thỏa: *max $(\pi(p), \pi(\overline{p})) = 1$*

- *PBKX của một sự kiện p* trong khái luật là bộ: $\begin{pmatrix} \pi(p) \\ \pi(\neg p) \end{pmatrix}$

  Nếu ta không thể khẳng định được p là tuyệt đối đúng thì:

  $\pi(p) = 1$, nhưng $\pi(\overline{p}) = \lambda$ $(\lambda \in [0, 1))$

- *PBKX của một khái luật* $p \to q$! được biểu diễn bởi ma trận:

$$\begin{pmatrix} \pi(p \to q) & \pi(\neg p \to q) \\ \pi(p \to \neg q) & \pi(\neg p \to \neg q) \end{pmatrix}$$

Bài tập

1. +Bằng các phép biến đổi tương đương trong lôgic mệnh đề, hãy chứng minh:
   a. Ta luôn có thể đưa một luật suy diễn ở dạng *chuẩn Horn*:
      $p_1 \wedge .... \wedge p_n \quad \rightarrow \quad q_1 \vee .... \vee q_m$
   về dạng *chuẩn sơ cấp*:
      $p_1 \wedge .... \wedge p_n \quad \rightarrow \quad q$
   b. Nguyên lý *chứng minh phản chứng*:
      (a $\rightarrow$ b *đúng*) $\equiv$ (a $\wedge \neg$ b *sai* hay *mâu thuẫn*)
   c. Nguyên lý *hợp giải.*
   d. Nguyên lý *modus ponens và tollens.*
   e. Trong việc chứng minh tự động tính hằng đúng của các biểu thức lôgic mệnh đề, tại sao người ta thường xét bài toán dạng BTA ?
   f. Quy tắc *Chuyển* và *Tách* trong thủ tục Vương Hạo.
   g. Phương pháp *Qui diễn (Abduction*, tương tự): Cho trước: A $\rightarrow$ B.
      . A ~ A' $\Rightarrow$ A' $\rightarrow$ B
      . B ~ B' $\Rightarrow$ A $\rightarrow$ B'
      . A ~ A' & B ~ B' $\Rightarrow$ A' $\rightarrow$ B'

2. +Hãy đưa về dạng *chuẩn hội* và *chuẩn tuyển* lần lượt cho vế trái (VT) và vế phải (VP) của các biểu thức lôgic mệnh đề sau:
   a. $[(a \wedge b \rightarrow c \vee d) \wedge (c \rightarrow e) \wedge a] \Rightarrow [b \rightarrow e]$
   b. $[(a \wedge b \rightarrow c \wedge d) \wedge (c \rightarrow e) \wedge a] \Rightarrow [b \wedge c \rightarrow e]$
   c. $[(a \vee b \rightarrow c \wedge d) \wedge (c \rightarrow e) \wedge a] \Rightarrow [d \wedge c \vee b]$
   d. $[(a \vee b \rightarrow c \vee d) \wedge (c \wedge b \rightarrow e) \wedge a] \Rightarrow [b \wedge c \rightarrow e]$
   e. $[b \wedge c \wedge \neg e] \Rightarrow [(a \wedge b \wedge (\neg c \vee \neg d)) \vee (c \wedge \neg e) \vee \neg a]$
   f. $[a \rightarrow (b \rightarrow c)] \Leftrightarrow [(a \rightarrow b) \rightarrow c]$
   g. $\{[(a \wedge b \rightarrow c \vee d) \wedge (c \rightarrow e) \wedge a] \vee [(a \wedge b \rightarrow e) \wedge a]\} \Rightarrow [b \rightarrow e]$
   h. $\{[(a \wedge b \rightarrow c \wedge d) \wedge (c \rightarrow e) \wedge a] \vee [(a \wedge b \rightarrow e \vee d) \wedge a]\} \Rightarrow [b \wedge c \rightarrow e]$

3. *Nếu trong bài toán *BTB* thay tập kết luận KL = $\wedge_{1 \le i \le k} q_i$ bởi KL = $V_{1 \le i \le k} q_i$ thì cần hiệu chỉnh thuật toán *suy diễn tiến* SDT ra sao ?

4. +*Kiểm tra tính hằng đúng* của các biểu thức lôgic trong bài tập 2 chương III (có thể *áp dụng vài heuristics để rút gọn hay chuyển đổi tương đương* cho *việc giải trở nên đơn giản hơn*) bằng các phương pháp:
   i) biến đổi *lôgic*
   ii) phương pháp *suy diễn tiến*
   iii) phương pháp *suy diễn lùi*
   iv) thuật toán *Vương Hạo (Wong Havard)*
   v) thuật toán *Robinson.* Nếu *thay thuật toán Robinson bằng "thuật giải Robinson"* thì các kết quả có luôn trùng nhau không ? Tại sao?