THUẬT TOÁN SUY DIỄN LÙI SỬ DỤNG BỘ NHỚ ĐỆM TOÀN CỤC CHO BÀI TOÁN SUY LUẬN LOGIC

Trần Thanh Lương

Trường Đại học Khoa học, Đại học Huế

Email: ttluong@hueuni.edu.vn

Ngày nhận bài: 16/5/2023; ngày hoàn thành phản biện: 31/5/2023; ngày duyệt đăng: 26/6/2023

TÓM TẮT

Bài toán suy luận logic là một trong những bài toán phổ biến của các hệ thống thông minh như: khai phá dữ liệu, suy luận tri thức, dự đoán/dự báo, xây dựng/kiểm tra mạch logic cho các linh kiện điện tử, … Hiện nay, có một số phương pháp giải quyết bài toán này như: thuật toán suy diễn tiến, thuật toán suy diễn lùi, phương pháp Vương Hạo, phương pháp Robinson, phương pháp hợp giải. Trong bài báo này, chúng tôi tập trung vào phương pháp suy diễn lùi và đề xuất một kỹ thuật sử dụng bộ nhớ đệm toàn cục để kiểm tra tính đúng, sai của các sự kiện trước khi chứng minh nhằm tăng tính hiệu quả cho phương pháp suy diễn lùi đối với bài toán suy luận logic. Bên cạnh đó, chúng tôi cũng nghiên cứu và xem xét việc loại bỏ những luật cũng như sự kiện không cần thiết trong quá trình chứng minh để làm tăng tốc độ suy diễn. Từ đó, đề xuất thuật toán suy diễn lùi sử dụng bộ nhớ đệm toàn cục cho bài toán suy luận logic.

Từ khóa: bài toán suy luận logic, bộ nhớ đệm toàn cục, thuật toán suy diễn lùi.

1. MỞ ĐẦU

Như chúng ta đã biết, không có một phương pháp tổng quát nào để giải quyết tất cả các bài toán. Một số phương pháp phù hợp cho lớp bài toán này nhưng không phù hợp đối với lớp bài toán khác. Do vậy, tùy theo từng bài toán, chúng ta có những phương pháp giải quyết khác nhau, trong đó phương pháp biểu diễn và phương pháp tìm kiếm trong không gian biểu diễn đó là một vấn đề hết sức quan trọng. Điều này quyết định đến tính khả thi trong giải quyết bài toán cũng như tốc độ thực thi bài toán nhanh hay chậm. Bài toán suy luận logic là một trong những bài toán phổ biến trong lĩnh vực trí tuệ nhân tạo với phương pháp biểu diễn tri thức bằng logic [6]. Cùng với sự phát triển về những thành tựu trong việc tạo ra các máy tính có tốc độ cao, chúng ta có thể chứng minh được các bài toán logic bằng phương pháp đơn giản nhất là lập bảng chân trị. Tuy nhiên, độ phức tạp của phương pháp này là quá lớn, $O(2^n)$, điều rất khó thực hiện khi

bài toán có số lượng các biến mệnh đề lớn [2].

Đối với bài toán suy luận logic, chúng ta sẽ biểu diễn tri thức của bài toán bằng logic hình thức và chứng minh sự kiện đã cho đúng hay sai. Để giải quyết bài toán này, chúng ta có nhiều phương pháp như: thuật toán suy diễn tiến, thuật toán suy diễn lùi, phương pháp Vương Hạo, phương pháp Robinson, phương pháp phân giải trên logic vị từ,… Trong bài báo này, chúng tôi đề cập đến bài toán suy luận logic trên cơ sở tri thức gồm có tập các sự kiện đúng ban đầu, tập các luật sản xuất và chứng minh một sự kiện đã cho đúng hay sai. Với bài toán này, hai phương pháp thông dụng thường được sử dụng là thuật toán suy diễn tiến và thuật toán suy diễn lùi. Thuật toán suy diễn tiến có ưu điểm là cho ra một khối lượng thông tin lớn từ các thông tin ban đầu và rất phù hợp khi bài toán có bản chất là thu thập thông tin rồi lấy ra điều cần suy diễn khi nó xuất hiện. Thuật toán suy diễn lùi có ưu điểm là tập trung vào đích cần chứng minh và xem xét những luật sản xuất liên quan đến vấn đề chứng minh. Khi tìm kiếm, thuật toán này chỉ tập trung tìm kiếm trên một phần cơ sở tri thức phù hợp với mục đích tìm kiếm [5]. Tuy nhiên, thuật toán suy diễn lùi hiện nay có một số nhược điểm nhất định như: việc chứng minh các sự kiện trung gian bị lặp lại cho dù sự kiện đó đã được chứng minh trước đó; việc chứng minh tất cả các sự kiện p_i trong luật sản xuất $p_1 \wedge p_2 \wedge \ldots \wedge p_k \rightarrow q$ để kết luận q là không cần thiết khi chúng ta đã chứng minh được một sự kiện p_i nào đó sai. Thật vậy, nếu một p_i đã được chứng minh sai thì $p_1 \wedge p_2 \wedge \ldots \wedge p_k$ sẽ mang giá trình sai và chúng ta suy ra được q sai; việc sử dụng tất cả các luật sản xuất có vế phải là q sẽ dẫn đến không tối ưu, nhất là khi đã có một luật sản xuất chứng minh được sự kiện q là đúng. Những nhược điểm này làm cho thời gian chứng minh của bài toán không tối ưu cũng như làm lãng phí bộ nhớ máy tính.

2. BÀI TOÁN SUY LUẬN LOGIC

2.1. Luật sản xuất

Luật sản xuất được hình thành trên cơ sở là các câu Horn dạng chuẩn và được sử dụng để biểu diễn tri thức phục vụ cho việc vận dụng thuật toán suy diễn tiến và thuật toán suy diễn lùi. Câu Horn dạng chuẩn có cú pháp là $p_1 \wedge p_2 \wedge \ldots \wedge p_k \rightarrow q$, trong đó $p_1, p_2, \ldots, p_k$ là các biến mạnh đề, q là một biến mệnh đề hoặc hằng false. Với câu Horn dạng chuẩn như trên, khi $n = 0$, câu Horn sẽ có dạng là $\rightarrow q$ và được gọi là một sự kiện q; khi $n > 0$, câu Horn có dạng $p_1 \wedge p_2 \wedge \ldots \wedge p_k \rightarrow q$ và được gọi là luật sản xuất, viết dưới dạng **if-then** như sau [3, 4]:

if p_1 **and** p_2 **and** … **and** p_k **then** q

2.2. Bài toán suy luận logic

Một trong những bài toán quan trọng trong logic mệnh đề là bài toán chứng minh

tính đúng đắn của phép suy diễn $P \to Q$ hoặc một sự kiện q nào đó. Đây cũng là bài toán suy luận logic thường gặp trong toán học cũng như trong thực tiễn cuộc sống. Để sử dụng các phương pháp suy diễn cho bài toán suy luận logic, chúng ta có thể phát biểu bài toán như sau:

Cho tập sự kiện giả thiết đúng $F = \{f_1, f_2, \dots, f_n\}$ và tập luật sản xuất $R = \{r_1, r_2, \dots, r_m\}$, trong đó $r_i, i = \overline{1,n}$ là các luật sản xuất. Chứng minh kết luận g đúng hay sai.

Để giải bài toán này, chúng ta có thể sử dụng thuật toán suy diễn tiến hoặc thuật toán suy diễn lùi. Trong đó, suy diễn tiến là quá trình suy diễn bắt đầu từ tập các sự kiện giả thiết đúng đã biết, vận dụng các luật thích hợp $p_1 \wedge p_2 \wedge \ldots \wedge p_k \to q$, sao cho $p_1, p_2, \dots, p_k$ là những sự kiện đã biết là đúng để rút ra những sự kiện q đúng mới và cứ tiếp tục như vậy cho đến khi có được sự kiện cần chứng minh là đúng (*chứng minh được sự kiện kết luận là đúng*) hoặc không có luật nào để sinh ra sự kiện đúng mới (*chứng minh sự kiện kết luận là sai*). Đối với suy diễn lùi, chúng ta bắt đầu từ sự kiện cần chứng minh q và thay vào đó là những sự kiện $p_1, p_2, \dots, p_k$ trong vế trái của luật $p_1 \wedge p_2 \wedge \ldots \wedge p_k \to q$ mà luật này có vế phải là sự kiện cần chứng minh. Chúng ta lần lượt chứng minh các sự kiện $p_1, p_2, \dots, p_k$. Nếu $p_1, p_2, \dots, p_k$ được chứng minh đúng thì ta kết luận q đúng, ngược lại đến một p_i nào đó mà không thể chứng minh đúng được từ các giả thiết thì chúng ta quay lui, sử dụng các luật khác sinh ra q để thực hiện việc chứng minh. Trong trường hợp không có luật nào để chứng minh được q chúng ta kết luận q sai [2, 5].

3. THUẬT TOÁN SUY DIỄN LÙI

Thuật toán suy diễn lùi là một trong những phương pháp khá phổ biến và hiệu quả dùng để giải quyết bài toán suy luận logic. Với bài toán suy luận logic được phát biểu: cho tập sự kiện giả thiết đúng $F = \{f_1, f_2, \dots, f_n\}$ và tập luật sản xuất $R = \{r_1, r_2, \dots, r_m\}$. Chứng minh kết luận g đúng hoặc sai.

Gọi T là tập các sự kiện cần chứng minh tại thời điểm đang xem xét. Đầu tiên, chúng ta khởi tạo $T = \{g\}$ là sự kiện cần chứng minh. Lấy trong T ra một sự kiện p để chứng minh, đặt $S(p) = \{r \in R | right(r) = p\}$ là tập các luật sản xuất trong R sao cho vế phải của luật này bằng p. Cứ mỗi luật $r \in S(p)$, chúng ta cập nhật lại T bằng cách loại bỏ khỏi T sự kiện bên vế phải, thêm vào T các sự kiện bên vế trái của luật r, $T = T \setminus right(r) \cup left(r)$ và tiến hành chứng minh bằng đệ quy các sự kiện trong T. Nếu tất cả các sự kiện trong T đều đúng thì chúng ta có p đúng (*các sự kiện đều suy dẫn về sự kiện cần chứng minh thuộc giả thiết*), ngược lại chúng ta quay lui, lấy luật sản xuất khác từ $S(p)$ và tiếp tục chứng minh bằng đệ quy các sự kiện sinh ra từ luật này. Nếu tất cả các luật này được sử dụng mà tồn tại sự kiện cần chứng minh không thuộc giả thiết thì p được kết luận là sai [2, 5]. Thuật toán suy diễn lùi được thể hiện như sau:

```
Input:
       Tập sự kiện đúng F = {f1, f2, ..., fn},
       Tập quy tắc R = {r1, r2, ..., rm},
       Kết luận cần chứng minh g.
Output:
       Khẳng định kết luận g đúng hoặc sai.
Method:
bool backwardChaining(g)
{
  T = {g};
  if (T ⊆ F)
    return true;
  else
  {
    p = get(T);
    S(p) = {r ∈ R | p = right(r)};
    if (S(p) = ∅)
       return false;
    else
    {
       for (r ∈ S(p))
       {
         result = true;
         T = T \ {right(ri)};
         T = T ∪ {left(ri)};
         for (q ∈ T \ F)
           return result and backwardChaining(q);
       }
    }
  }
}
```

Có thể thấy rằng, thuật toán suy diễn lùi là một thuật toán đệ quy quay lui. Để chứng minh một kết luận *g* nào đó là đúng hay không, chúng ta phải chứng minh tất cả

những sự kiện được sử dụng để sinh ra g. Quá trình này tiếp tục như vậy cho đến khi tất cả các sự kiện cần chứng minh thuộc tập sự kiện đúng ban đầu F (*khẳng định g đúng*), hoặc không thể tìm ra luật để có thể triển khai tiếp các sự kiện đang cần chứng minh (*khẳng định g sai*). Thuật toán suy diễn lùi nêu trên có một số các nhược điểm như sau:

(i) Bản chất của thuật toán là đệ quy quay lui, trong quá trình chứng minh, một số sự kiện được chứng minh là đúng hoặc sai nhưng không lưu lại kết quả, do đó không sử dụng lại được kết quả chứng minh trước đó. Điều này dẫn đến việc phải chứng minh lặp lại nhiều lần các sự kiện cần chứng minh gây ra lãng phí về thời gian và tốn kém về không gian bộ nhớ.

(ii) Việc chứng minh tất cả các sự kiện trong tập $T\backslash F$ là không cần thiết khi có một sự kiện $q \in T\backslash F$ được chứng minh là sai. Thật vậy, vì tất cả các sự kiện trong tập $T\backslash F$ được phối hợp với nhau theo toán tử $\wedge$ nên khi một sự kiện $q \in T\backslash F$ được chứng minh là sai thì sự kiện g (*được suy diễn từ hội của các sự kiện trong $T\backslash F$*) cũng mang giá trị là sai.

(iii) Việc sử dụng tất cả các luật trong $S(p)$ sẽ dẫn đến không tối ưu, đặc biệt là khi đã có một luật để chứng minh được sự kiện p là đúng thì không cần phải dùng những luật khác để chứng minh lại p nữa, bởi vì việc chứng minh lại p bởi các luật khác chỉ làm mất thêm thời gian mà không thể thay đổi kết quả của p.

4. THUẬT TOÁN SUY DIỄN LÙI SỬ DỤNG BỘ NHỚ ĐỆM TOÀN CỤC

Những hạn chế nêu trên làm cho thuật toán suy diễn lùi có tốc độ thực hiện chậm, đặc biệt là lãng phí trong việc chứng minh lại những sự kiện đã chứng minh đúng hoặc sai. Trong phần này, chúng tôi đề xuất thuật toán suy diễn lùi sử dụng bộ nhớ đệm toàn cục, đồng thời bổ sung thêm một số kỹ thuật nhằm loại bỏ những thao tác chứng minh không cần thiết cho phép chứng minh sự kiện kết luận g nhanh hơn so với thuật toán suy diễn hiện có. Cụ thể như sau:

- Để khắc phục được nhược điểm (i) về việc chứng minh lại sự kiện đã được chứng minh là đúng hoặc sai của thuật toán suy diễn lùi ở trên, chúng tôi sử dụng thêm hai bộ nhớ đệm toàn cục gồm: $GlobalTrue$ để lưu lại những sự kiện đúng hoặc đã được chứng minh là đúng; $GlobalFalse$ để lưu lại những sự kiện đã được chứng minh là sai. Hai bộ nhớ đệm toàn cục này làm nhiệm vụ kiểm tra những sự kiện đã được chứng minh là đúng hoặc sai để không phải chứng minh lại các sự kiện này nhằm làm giảm số lần chứng minh các sự kiện. Do có sử dụng $GlobalTrue$ để lưu lại các sự kiện đã được chứng minh đúng (*bao gồm các sự kiện đã được khẳng định trong F*) nên tập các sự kiện cần chứng minh tại vòng lặp $T\backslash F$ được thay bởi $T\backslash GlobalTrue$ đã rút gọn được nhiều sự kiện cần phải chứng minh.

- Để khắc phục nhược điểm (ii) về việc phải chứng minh tất cả các sự kiện trong tập T (*các sự kiện trong T được liên kết với nhau qua toán tử* $\wedge$) khi có một sự kiện $q \in T$ được chứng minh sai, chúng ta cho dừng vòng lặp **foreach** ($q \in T$) và trả về kết quả chứng minh là sai, việc chứng minh các sự kiện còn lại khác không có ý nghĩa vì không làm thay đổi kết quả này. Tương tự như thế, để khắc phục nhược điểm (iii) về việc phải sử dụng tất cả các luật để chứng minh sự kiện, chúng ta cho dừng vòng lặp **foreach** ($r \in S(g)$) khi có một luật nào đó chứng minh sự kiện là đúng mà không cần sử dụng các luật khác (*chúng ta có thể sử dụng một trong các luật trong* $S(g)$ *dùng để chứng minh g đúng, khi tất cả các luật trong* $S(g)$ *không thể chứng minh g đúng thì mới kết luận là* g *sai*).

Với những ý tưởng như trên, chúng tôi đề xuất thuật toán suy diễn lùi sử dụng bộ nhớ đệm toàn cục như sau:

```
Input:
       Tập sự kiện đúng F = {f₁, f₂, ..., fₖ},
       Tập quy tắc R = {r₁, r₂, ..., rₙ},
       Kết luận cần chứng minh g.
Output:
       Khẳng định kết luận g đúng hoặc không đúng.
Method:
bool backwardChainingUseGlobalCaching(g)
{
  if (g ∈ GlobalTrue)
    return true;
  if (g ∈ GlobalFalse)
    return false;
  S(g) = {r ∈ R | g = right(r)};
  if (S(g) = ∅)
  {
    GlobalFalse = GlobalFalse ∪ {g};
    return false;
  }
  T = ∅;
  result = true;
  foreach (r ∈ S(g))
  {
    result = true;
    T = T ∪ {left(r)} \ GlobalTrue;
    foreach (q ∈ T)
    {
       rs = backwardChainingUseGlobalCaching(q);
```

```
        if (rs = true)
          GlobalTrue = GlobalTrue ∪ {q};
        else
        {
          GlobalFalse = GlobalFalse ∪ {q};
          result = false;
          break;
        }
      }
      if (result = true)
        break;
    }
    return result;
}
```

5. KẾT LUẬN

Thuật toán suy diễn lùi được sử dụng khá phổ biến trong các bài toán suy luận logic. Bản chất thuật toán này là đệ quy quay lui, không lưu lại các kết quả trung gian cho nên việc chứng minh các sự kiện bị lặp lại cũng như sử dụng các luật không tối ưu. Trên cơ sở phân tích những nhược điểm của thuật toán suy diễn lùi, bài báo đã đề xuất kỹ thuật sử dụng bộ nhớ đệm toàn cục để lưu trữ những sự kiện đã được chứng minh đúng và những sự kiện đã được chứng minh sai nhằm tránh việc chứng minh lặp lại những sự kiện này gây ra lãng phí về mặt thời gian và không gian bộ nhớ. Bên cạnh đó, bài báo cũng đưa ra một số kỹ thuật nhằm hạn chế việc xem xét các sự kiện, các luật không mang lại ý nghĩa trong quá trình chứng minh.

TÀI LIỆU THAM KHẢO

[1]. Erdani Y. (2012). Developing Backward Chaining Algorithm of Inference Engine in Ternary Grid Expert System. International Journal of Advanced Computer Science and Applications. Vol 3, No. 9, pp. 241-245.

[2]. Hoàng Thị Lan Giao, Đoàn Thị Hồng Phước, Trần Thanh Lương (2018). Giáo trình trí tuệ nhân tạo. Nhà xuất bản Đại học Huế.

[3]. Horn, A. (1951). On Sentences Which are True of Direct Unions of Algebras. Journal of Symbolic Logic. Vol. 16, No. 1, pp. 14–21.

[4]. Makowsky J. A. (1987). Why Horn Formulas Matter in Computer Science: Initial Structures and Generic Examples. Journal of Computer and System Sciences. Vol. 34, pp. 266-292.

[5]. Ronald J. Brachma and Hector J. Levesqu (2004). Knowledge Representation and Reasoning. Morgan Kaufmann Publishers Inc.

[6]. Stuart J. Russell and Peter Norvig (2021). Artificial Intelligence: A Modern Approach (4th Edition). Pearson Publisher.

BACKWARD CHAINING ALGORITHM USING GLOBAL CACHING FOR LOGICAL REASONING PROBLEMS

Tran Thanh Luong

University of Sciences, Hue University

Email: ttluong@hueuni.edu.vn

ABSTRACT

The logical reasoning problem is one of the common problems in intelligent systems such as data mining, knowledge inference, machine learning prediction, building and testing logic circuits for electronic components, etc. Today, we have many methods to solve this problem, such as the forward chaining algorithm, the backward chaining algorithm, Wang's method, Robinson's method, and the resolution method. In this paper, we present the backward chaining algorithm and propose a technique that uses global caching to check the correctness of the propositional variables before proving them. This technique aims to increase the efficiency of the backward chaining algorithm for logical reasoning problems. In addition, we also study and consider removing unnecessary rules and facts from the proof process to increase the speed of inference. Therefore, we propose a backward chaining algorithm using global caching for logical reasoning problems.

Keywords: logical reasoning problem, backward chaining algorithm, global caching.

Trần Thanh Lương, sinh năm 1979 tại Quảng Trị. Ông tốt nghiệp đại học năm và thạc sĩ ngành Tin học vào năm 2001 và 2005. Năm 2016, Ông nhận học vị tiến sĩ ngành Khoa học máy tính tại Trường Đại học Khoa học, Đại học Huế. Từ năm 2021 đến nay, Ông công tác tại Trường Đại học Khoa học, Đại học Huế.

Lĩnh vực nghiên cứu: logic mô tả, web ngữ nghĩa, học máy.