Tạp chí Tin học và Điều khiển học, T. 18, S. 3 (2002), 279–284

CẢI TIẾN MỘT SỐ GIẢI THUẬT PHÂN TÍCH CÚ PHÁP TRONG XỬ LÝ NGÔN NGỮ TỰ NHIÊN

PHAN THỊ TƯƠI

Abstract. Earley and Chart algorithms are often used to parse natural language. However, these algorithms are limited when they are used to work with large grammar. This paper presents some improvements for Earley and top-down chart algorithms in natural language processing.

Tóm tắt. Để phân tích cú pháp cho ngôn ngữ tự nhiên, người ta thường dùng các giải thuật như Earley và biểu đồ. Tuy nhiên khi xử lý các văn phạm lớn thì các giải thuật nêu trên đều bị hạn chế. Bài báo này sẽ trình bày một số cải thiện cho giải thuật Earley và biểu đồ từ trên xuống trong xử lý ngôn ngữ tự nhiên.

1. MỞ ĐẦU

Vai trò của phân tích cú pháp trong xử lý ngôn ngữ tự nhiên là vô cùng quan trọng. Tuy nhiên không phải tất cả các giải thuật phân tích cú pháp cho ngôn ngữ lập trình đều có thể áp dụng cho ngôn ngữ tự nhiên, bởi vì văn phạm của ngôn ngữ tự nhiên là không tường minh. Ngay cả khi ta dùng các giải thuật như Tomita [10], Earley [1] và Chart [3], là những giải thuật phân tích cú pháp cho văn phạm không tường minh thì cũng gặp nhiều khó khăn khi áp dụng chúng cho xử lý ngôn ngữ tự nhiên, bởi vì muốn phân tích một chuỗi nhập vào là câu hoặc đoạn câu của ngôn ngữ tự nhiên thì bộ phân tích buộc phải kiểm tra từ vài chuỗi đến hàng chục, hàng trăm chuỗi từ loại khác nhau (từ loại được hiểu như các token trong ngôn ngữ lập trình), điều đó sẽ dẫn đến sự bùng nổ tổ hợp. Trong bài báo này chúnh tôi trình bày việc cải thiện giải thuật Earley và giải thuật biểu đồ từ trên xuống (top down chart parsing) cho phân tích cú pháp của ngôn ngữ tự nhiên.

2. MỘT SỐ GIẢI THUẬT EARLEY VÀ LR(k) CẢI TIẾN

Giải thuật LR là giải thuật phân tích cú pháp từ dưới lên còn được gọi là giải thuật bảng. Giải thuật này chỉ dùng cho văn phạm phi ngữ cảnh tường minh, Tomita đã cải tiến giải thuật này để giải quyết đụng độ bằng việc mô phỏng việc thực thi song song của bộ phân tích LR dựa trên nhiều bản sao chép của stack trạng thái. Tuy nhiên giải thuật của Tomita chỉ hiệu quả khi sự đụng độ ít. Còn giải thuật Earley lại cho phép làm việc với văn phạm không tường minh. Với văn phạm bất kỳ thì thời gian phân tích của Earley là n^3, n là chiều dài của chuỗi nhập, n^2 với văn phạm tường minh và n với hầu hết ngôn ngữ lập trình. Giải thuật LRE [8] là sự kết hợp của Earley và LR(k). LRE giống như Tomita, sử dụng bảng nhưng vẫn giữ được những ưu điểm của giải thuật Earley. Các giải thuật cải tiến nêu trên chỉ làm việc với một chuỗi nhập.

3. SƠ LƯỢC VĂN PHẠM TIẾNG VIỆT

Trước khi trình bày việc cải thiện giải thuật Earley và biểu đồ từ trên xuống tôi sẽ nêu sơ lược về văn phạm tiếng Việt. Khi nghiên cứu tiếng Việt chúng ta phải nghiên cứu hai thành phần là từ pháp và cú pháp. Từ pháp chuyên nghiên cứu các biến hình của từ và đặc tính ngữ pháp của các loại từ cũng như sự cấu tạo của từ , còn cú pháp nghiên cứu cách cấu thành các từ và câu từ các nhóm từ theo một quy tắc nhất định.

3.1. Từ loại

Từ trong tiếng Việt được chia thành các loại: danh từ, động từ, thời vị từ, số từ, tính từ, đại

từ, phó từ, giới từ, liên từ, trợ từ và thán từ. Với mỗi loại từ trên lại có thể được chia nhỏ hơn, mang nghĩa khác nhau tùy thuộc vào ngữ cảnh. Ví dụ: danh từ có 27 loại, được ký hiệu bắt đầu bằng chữ N, như: N20, N51,... Động từ có 22 loại bắt đầu bằng V: V45, V20,... Tính từ có 16 loại được bắt đầu bằng A. Chi tiết về chia từ loại được trình bày ở [5, 6].

3.2. Từ tổ

Hai hay nhiều hơn hai thực thể ở trong câu có quan hệ với nhau về ý nghĩa và cú pháp thì được gọi là từ tổ. Trong mỗi từ tổ bao giờ cũng có từ trung tâm. Tùy thuộc từ trung tâm mà ta phân từ tổ thành từ tổ động từ, từ tổ danh từ, từ tổ số từ, từ tổ thời vị từ. Mỗi loại từ tổ lại có từ một đến nhiều dạng khác nhau. Từ cấu trúc từ tổ ta có thể xây dựng văn phạm để kiểm tra cú pháp của chúng.

4. CẢI THIỆN PHƯƠNG PHÁP EARLEY CHO XỬ LÝ NGÔN NGỮ TỰ NHIÊN

Khi bộ phân tích cú pháp muốn xác định xem chuỗi nhập là câu hoặc đoạn câu của ngôn ngữ tự nhiên có đúng cú pháp không thì nó phải phân tích các chuỗi từ loại tương ứng của câu, đoạn câu. Trong trường hợp xấu nhất bộ phân tích phải kiểm tra tất cả các chuỗi mới có thể kết luận về cú pháp của câu nhập.

Ví dụ 1. Câu cần phân tích: *Cán bộ công chức quy định tại pháp lệnh này là công dân Việt Nam.*

Câu trên được bộ phân tích từ vựng xử lý và xuất ra các chuỗi từ loại tương ứng:

N20 N20 N61 L22 N51 D71 H10 N20 N13
N20 N20 V48 L22 N51 D71 H10 N20 N13

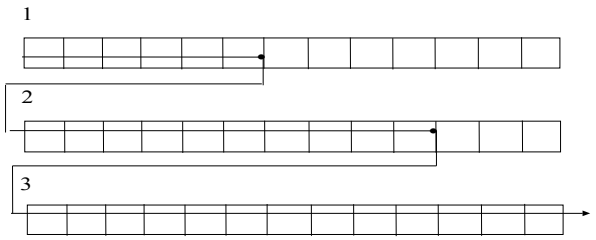
Ở đây chỉ có từ quy định là có hai từ loại: N61 và V48. Do đó việc phân tích cú pháp cho câu trên không có gì khó khăn.

Ví dụ 2. Đoạn câu cần phân tích: *trong biên chế và hưởng lương từ ngân sách.*

Có 12 chuỗi từ loại được xuất ra:

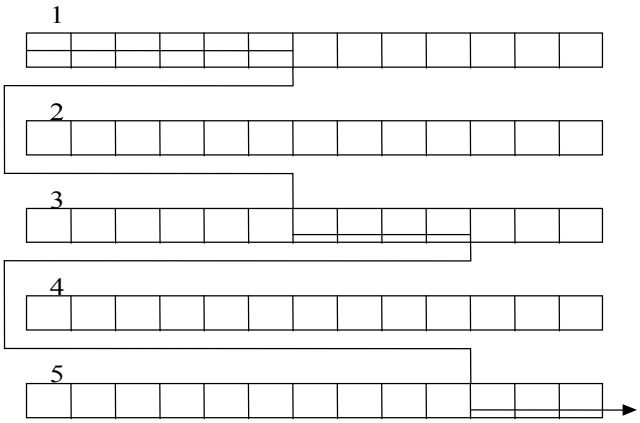
A11 N22 L10 V43 N23 F10 N23 N50
A11 N22 L10 V43 N23 F11 N23 N50
A11 V40 L10 V43 N23 F10 N23 N50
A11 V40 L10 V43 N23 F11 N23 N50
F10 N22 L10 V43 N23 F10 N23 N50
F10 N22 L10 V43 N23 F11 N23 N50
F10 V40 L10 V43 N23 F23 N23 N50
F10 V40 L10 V43 N23 F10 N23 N50
F11 N22 L10 V43 N23 F10 N23 N50
F11 N22 L10 V43 V23 F11 N23 N50
F10 V40 L10 V43 N23 F10 N23 N50
F10 V40 L10 V43 N23 F11 N23 N50

Trường hợp xấu nhất bộ phân tích phải kiểm tra hết 12 chuỗi từ loại. Một số trường hợp khác có thể số chuỗi từ loại còn nhiều hơn, bởi vì bộ phân tích kiểm tra không thành công ở một chuỗi thì nó buộc phải kiểm tra lại từ đầu chuỗi tiếp theo. Đó chính là nguyên nhân làm cho giải thuật Earley bị hạn chế. Hình 1 minh họa quá trình phân tích đoạn câu của giải thuật Earley.



Hình 1. Kiểm tra cú pháp cho câu có nhiều chuỗi từ loại bằng giải thuật Earley.

Trong thực tế các chuỗi từ loại đều có các chuỗi con giống nhau, chẳng hạn trong các ví dụ 1 và 2. Lợi dụng đặc điểm này, chúng tôi đã cải thiện giải thuật Earley như sau: nếu bộ phân tích thất bại khi đang kiểm tra một chuỗi, thì nó sẽ so trùng các chuỗi còn lại với đoạn vừa kiểm tra thành công và sẽ tiếp tục quy trình phân tích ở vị trí của một chuỗi khác có chuỗi con dài nhất trùng với đoạn đã phân tích. Quá trình này được lặp lại cho tới khi bộ phân tích "đi" hết một chuỗi nào đó. Lúc đó câu nhập được xác nhận là đúng cú pháp. Ngược lại khi đi đến chuỗi cuối cùng mà vẫn không phân tích thành công thì bộ phân tích sẽ kết luận rằng câu nhập vào không đúng cú pháp. Quá trình phân tích cú pháp của giải thuật Earley cải thiện được minh họa ở hình 2.



Hình 2. Quá trình phân tích của giải thuật Earley cải thiện

Giải thuật Earley cải thiện:

Nhập: Văn phạm phi ngữ cảnh $G = (N, \Sigma, P, S)$ và các chuỗi từ loại của câu cần phân tích cú pháp: $W_1, W_2, ..., W_m \in \Sigma^*$; Kích thước các chuỗi bằng kích thước câu nhập.

Xuất: Danh sách các tập thực thể $I_0, I_1, ..., I_n$.

Phương pháp: Thực hiện phân tích trên W_1:

Trước tiên ta tạo I_0:

(1) Nếu $S \to \alpha$ là luật sinh thuộc P, thì thêm $[S \to \bullet\alpha, 0]$ vào I_0. Tiếp tục bước 2 và 3 cho tới khi không còn thực thể mới chưa cho vào I_0.

(2) Nếu $[B \to \gamma\bullet, 0]$ ở trong I_0, thì thêm vào I_0 thực thể $[A \to \alpha B \bullet \beta, 0]$ cho tất cả các thực thể có dạng $[A \to \alpha \bullet B\beta, 0]$ ở trong I_0.

(3) Giả sử $[A \to \alpha \bullet B\beta, 0]$ là thực thể ở trong I_0, thêm vào I_0 tất cả các thực thể $[B \to \bullet\gamma, 0]$, nếu tồn tại các luật sinh có dạng $B \to \gamma$ ở trong P và các thực thể đó chưa có ở trong I_0.

Khởi động các trị : $m_1 := 1$; $m_2 := 1$; $l := 1$;

Xây dựng I_j, khi đã có các $I_0, I_1, ..., I_{j-1}$.

(4) Nếu mỗi $[B \to \alpha \bullet a\beta, i]$ trong I_{j-1}, mà $a = a_j$ thì thêm $[B \to \alpha a \bullet \beta, i]$ vào I_j và thực hiện các bước 5, 6 cho tới khi không còn thực thể mới nào chưa có trong I_j.

Ngược lại, nếu $[B \to \alpha \bullet a\beta, i]$ trong I_{j-1} mà $a \neq a_j$ thì thực hiện các bước sau:

a. Gán $m_2 := j - 1$;

b. So trùng chuỗi $W_{lm} = a_{m1}a_{m1+1}...a_{m2}$ với các chuỗi $W_{l+1}, W_{l+2}, ..., W_m$ để tìm một chuỗi con dài nhất trùng với chuỗi W_{lm}, gọi là W_{hm} thuộc chuỗi W_h.

c. Gọi a'_i là ký hiệu đứng ngay bên phải ký hiệu cuối dãy W_{hm} của chuỗi W_h.

d. Gán $m_2 := j$; $m_1 := m_2$; $l := h$.

e. Quay về bước (4) để tạo I_j, với ký hiệu nhập được đọc là a'_j ở vị trí j của chuỗi W_h.

(5) Giả sử $[A \to \alpha\bullet, i]$ là thực thể ở trong I_j, kiểm tra I_j cho tất cả các thực thể có dạng $[B \to \alpha \bullet A\beta, k]$ với mỗi thực thể tìm thấy ta thêm $[B \to \alpha A \bullet \beta, k]$ cho I_j.

(6) Giả sử $[A \to \alpha \bullet \beta,\ i]$ là thực thể trong I_j, với tất cả các luật sinh có dạng $B \to \gamma$ trong P, ta thêm $[B \to \bullet\gamma,\ j]$ vào I_j. Chúng ta nhận thấy rằng thực thể nào có ký hiệu kết thúc ở bên

phải của dấu • sẽ không phải là thực thể mới trong bước 2, 3, 5, 6. Giải thuật được minh họa ở hình 2, giải thuật này được thực hiện bằng chương trình trong [5].

Lưu ý: các dòng in nghiêng là phần cải thiện giải thuật Earley của chúng tôi.

5. CẢI THIỆN PHƯƠNG PHÁP BIỂU ĐỒ TRONG XỬ LÝ NGÔN NGỮ TỰ NHIÊN

Phần này trình bày việc cải thiện giải thuật phân tích cú pháp biểu đồ từ trên xuống. Giải thuật này hiệu quả hơn so với các giải thuật hiện nay cho các văn phạm hợp lý. Nó kết hợp ưu điểm của hai giải thuật phân tích cú pháp biểu đồ từ dưới lên (bottom up chart parsing) và phân tích từ trên xuống (top down parsing). Phương pháp này không bị quay lui và không xét các từ loại mà chúng không thể dùng để tạo ra câu đúng.

Để thuận lợi cho việc giải thích ý tưởng cải thiện giải thuật biểu đồ, chúng tôi tóm tắt giải thuật biểu đồ [9] từ trên xuống như sau.

5.1. Giải thuật trình bày cung từ trên xuống

Để thêm một cung $S \rightarrow C_1...\bullet C_i...C_n$ vào cuối của vị trí j thì với mỗi luật sinh của văn phạm có dạng $C_i \rightarrow X_1...X_k$ ta thêm vào một cung mới $C_i \rightarrow \bullet X_1...X_k$ một cách đệ quy từ vị trí j đến j.

5.2. Giải thuật phân tích sơ đồ từ trên xuống

Bắt đầu với mỗi luật sinh của văn phạm có dạng $S \rightarrow X_1...X_k$, thêm vào một cung có tên $S \rightarrow \bullet X_1X_2...X_k$ bằng giải thuật trình bày cung. Thực hiện các bước phân tích sau đây cho tới khi không còn ký hiệu nhập:

a. Nếu bảng rỗng, hãy tìm các từ loại của ký hiệu nhập kế tiếp và thêm chúng vào bảng.
b. Chọn một thành phần từ bảng (gọi nó là C).
c. Dùng giải thuật mở rộng, kết hợp C với từng cung hoạt động trên biểu đồ. Một thành phần mới được thêm vào bảng.
d. Với bất kỳ một cung mới nào được tạo ra ở bước c, hãy thêm chúng vào biểu đồ bằng giải thuật trình bày cung.

5.3. Giải thuật mở rộng để thêm thành phần C vào vị trí từ p_1 đến p_2

Thực hiện các bước sau:

a. Thêm C vào sơ đồ từ p_1 đến p_2.
b. Với bất kỳ cung hoạt động có dạng $X \rightarrow X_1... \bullet C...X_n$ từ vị trí p_0 đến p_1, thêm cung mới $X_1 \rightarrow X_1...C \bullet ...X_n$ từ p_0 đến p_2.
c. Với bất kỳ cung hoạt động nào có dạng $X \rightarrow X_1...X_n \bullet C$ từ vị trí p_0 đến p_1, thêm một thành phần mới của X từ p_0 đến p_2 vào trong bảng.

Mặc dù ưu điểm như vậy nhưng khi áp dụng phân tích cú pháp cho văn phạm lớn (có từ vài nghìn đến vài chục nghìn luật) thì giải thuật này cũng bị hạn chế về tốc độ. Trong xử lý ngôn ngữ tự nhiên chúng ta thường gặp các văn phạm lớn, vì vậy chúng tôi đã thêm vào một số xử lý về tổ chức dữ liệu khi thực hiện giải thuật nhằm khắc phục nhược điểm trên.

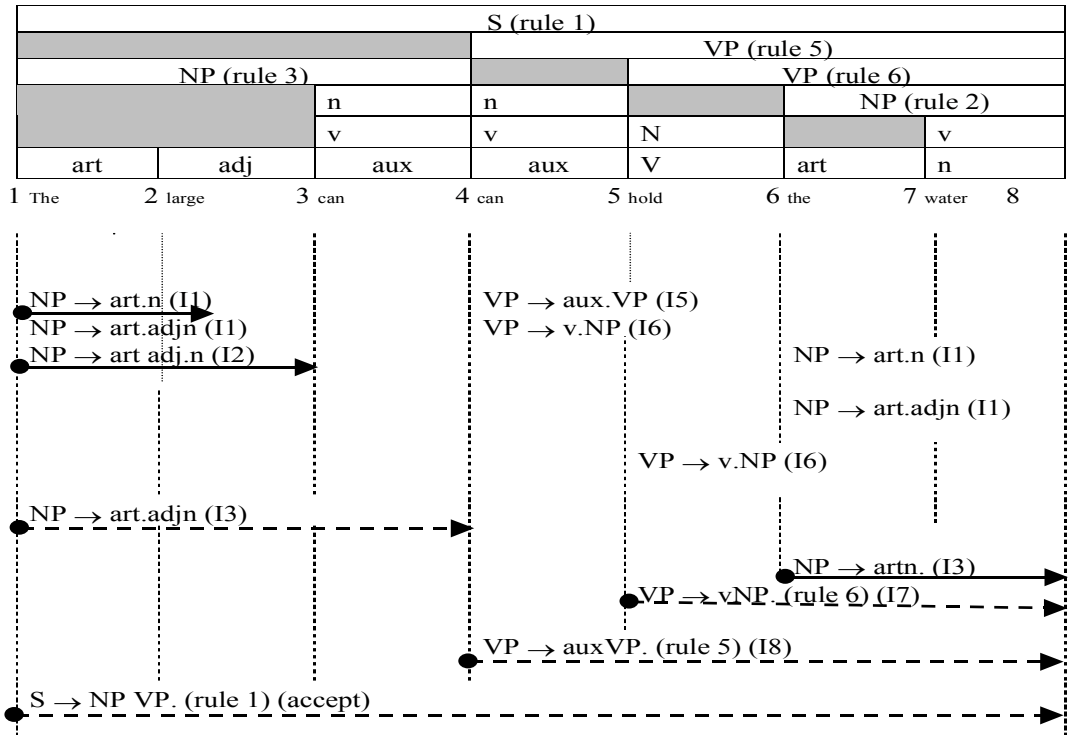
1) Số hóa toàn bộ các ký hiệu kết thúc và không kết thúc của văn phạm (để tiết kiệm bộ nhớ và so trùng nhanh).
2) Xây dựng giải thuật tìm kiếm theo phương pháp "băm" cho từng tập thực thể I_j.
3) Tính trước các thực thể do các ký hiệu không kết thúc tạo nên.
4) Ghi nhớ địa chỉ của các thực thể $I_0, I_1, ..., I_n$ (được tạo từ các hành vi chuyển dịch dấu • về bên phải, qua các ký hiệu văn phạm). Nhờ các tác vụ trên, nên bộ phân tích không cần lưu chứa một lần nữa dữ liệu đã được tính toán trước đó, mà chỉ cần lưu địa chỉ của tập thực thể tương ứng, do đó tiết kiệm được bộ nhớ đáng kể và bảo đảm tốc độ phân tích. Ví dụ 3 sẽ minh họa quá trình phân tích một câu bằng giải thuật biểu đồ từ trên xuống.

Ví dụ 3. Thực hiện phân tích cú pháp cho câu: *the large can can hold the water*

Cho văn phạm G với tập luật sinh:

(1) S→ NP VP (2) NP → art adj n
(3) NP → art n (4) NP → adj n
(5) VP → aux VP (6) VP → v NP

Hình 3 là các bước phân tích với các $I_1, I_2, ..., I_8$ ở bên cạnh các cung để biểu diễn cho địa chỉ của các tập thực thể.



Hình 3. Quá trình phân tích câu: *the large can can hold the water*
bằng giải thuật biểu đồ đi xuống cải thiện

Thực hiện theo bước b, ta có:

Tập •S: S → • NP VP
Tập •NP: NP → • adj n
NP→ • art n
NP→ • art adj n
Tập •VP: VP→ • aux VP
VP→ • v NP

Theo giải thuật chúng ta tính được các tập thực thể:

I_0 (S) I_1 (art) I_2 (adj) I_3 (n)
I_5 (aux) I_6 (v) I_7 (NP) I_8 (VP)

Lưu ý: Ví dụ ở trạng thái 7, triển khai cung NP thì máy tính không tính lại các cung NP → art n và NP → art adj n mà nó chỉ cần chỉ đến địa chỉ của I_1. Tương tự như vậy cho các trường hợp khác.

6. KẾT LUẬN

Qua thực tế khi thực hiện các đề tài nghiên cứu trong lĩnh vực xử lý ngôn ngữ tự nhiên, chúng tôi thấy có thể áp dụng các giải thuật đã có, song phải cải thiện để các giải thuật này đáp ứng được các yêu cầu đề ra khi đi sâu vào thực nghiệm, những cải thiện này góp phần làm cho các giải thuật

khá nổi tiếng như Earley, LR, biểu đồ càng hoàn thiện và hiệu quả hơn trong xử lý ngôn ngữ tự nhiên.

TÀI LIỆU THAM KHẢO

[1] Alfred V. Aho, Jeffred D.Ullman, *The Theory of Parsing, Translation, and Compiling*, Vol.1, Parsing, Prentice - Hall, Inc., 1972 (320–330).
[2] Alfred V. Aho, Ravisethi, Jeffrey D. Ullman, *Compiler Principles, Techniques, and Tools*, AddisonWesley Publishing Company, 1986.
[3] James Allen, *Natural Language Understanding*, the Benjamin / Commings Publishing company, Inc., 1995.
[4] John Aycook and Nigel Horspool, *Faster Generalized LR parsing,* aycook, nigelh @csc.uvic.ca.
[5] Phan Thị Tươi, "Trợ giúp bắt lỗi chính tả tiếng Việt tự động bằng máy tính (giai đoạn 1)", đề tài cấp thành phố, Trường Đại học Bách khoa TPHCM, 1998.
[6] Phan Thị Tươi, "Trợ giúp bắt lỗi chính tả tiếng Việt tự động bằng máy tính (giai đoạn 2)", đề tài cấp thành phố , Trường Đại học Bách khoa TPHCM 2001.
[7] Phan Thị Tươi, *Trình biên dịch*, NXB Đại học Quốc gia TPHCM, 2001.
[8] Philippe McLean Q., R. Nigel Horspool, *A faster Earley parser,* email: pmclean@csc.uvic.ca, nigelh@csc.uvic.ca.
[9] R. Nige Horspool, *Recursive Ascentdescent Parsing*, P.O. Box 1700, Victoria, B.C Canada, 1991.
[10] Tomita Masaru, *An efficient augmented context - free parsing algorithm,* Computational Linguistics **13** (1–2), 1987 (31–46).

Nhận bài ngày 5 - 1 - 2002

Trường Đại học Bách khoa - ĐHQG TPHCM