Chương V

VĂN PHẠM PHI NGỮ CẢNH

Nội dung chính : Trong chương này, ta sẽ nghiên cứu một loại văn phạm khá quan trọng, gọi là văn phạm phi ngữ cảnh (CFG) và lớp ngôn ngữ mà chúng mô tả - ngôn ngữ phi ngữ cảnh (CFL). CFL, cũng như tập hợp chính quy, có nhiều ứng dụng thực tế rất quan trọng, đặc biệt trong việc biểu diễn ngôn ngữ lập trình. Chẳng hạn, CFG dùng hữu ích để mô tả các biểu thức số học trong các dấu ngoặc lồng nhau hay những cấu trúc khối trong ngôn ngữ lập trình (cấu trúc khối **begin-end**). Sau khi định nghĩa văn phạm phi ngữ cảnh, một số cách biến đổi văn phạm phi ngữ cảnh nhằm giản lược nó và đưa nó về một trong những dạng chuẩn sẽ được trình bày. Cuối chương, bổ đề bơm cho ngôn ngữ CFL và một số tính chất nhằm xác định tập ngôn ngữ này cũng sẽ được giới thiệu.

Mục tiêu cần đạt: Cuối chương, sinh viên cần phải nắm vững:

- Khái niệm CFG, xác định các thành phần của một CFG.
- Nhận dạng được lớp ngôn ngữ mà một văn phạm CFG đặc tả.
- Xây dựng các luật sinh cho một CFG đặc tả một lớp ngôn ngữ.
- Các bước giản lược văn phạm CFG không chứa các giá trị vô ích.
- Chuẩn hóa CFG về các dạng chuẩn Chomsky hoặc Greibach.
- Ứng dụng bổ đề bơm cho CFL để chứng tỏ một ngôn ngữ không là ngôn ngữ phi ngữ cảnh.
- Xác định một ngôn ngữ có thuộc lớp ngôn ngữ phi ngữ cảnh hay không theo các tính chất của CFL.
- Kiểm tra tính rỗng, hữu hạn hoặc vô hạn của một CFL.

Kiến thức cơ bản: Để tiếp thu tốt nội dung của chương này, trước hết sinh viên cần hiểu rõ cấu trúc cú pháp của một số ngôn ngữ lập trình cấp cao như Pascal, C; nắm vững lý thuyết đồ thị và cây; phương pháp chứng minh phản chứng và sự phân cấp các lớp văn phạm theo Noam Chomsky; …

Tài liệu tham khảo :

[1] John E. Hopcroft, Jeffrey D.Ullman – *Introduction to Automata Theory, Languages and Computation* – Addison – Wesley Publishing Company, Inc – 1979 ***(Chapter 4 : Context – Free Grammars).***

[2] V.J. Rayward-Smith – *A First course in Formal Language Theory (Second Editor)* – McGraw-Hill Book Company Europe – 1995 (***Chapter 5: Context-Free Languages)***

[3] From Wikipedia, the free encyclopedia – *Context-Free Grammar:*
http://en.wikipedia.org/wiki/Context-free_grammar

I. VĂN PHẠM PHI NGỮ CẢNH (CFG : Context Free Grammar)

Xuất xứ của văn phạm phi ngữ cảnh là sự mô tả thông qua các ngôn ngữ tự nhiên. Ta có thể viết các quy tắc cú pháp để diễn tả câu "**An là sinh viên giỏi**" như sau :

< câu đơn > → < chủ ngữ > < vị ngữ >
< chủ ngữ > → < danh từ >
< vị ngữ > → < động từ > < bổ ngữ >
< bổ ngữ > → < danh từ > < tính từ >
< danh từ > → **An**
< danh từ > → **sinh viên**
< động từ > → **là**
< tính từ > → **giỏi**

Các từ trong dấu móc nhọn như < câu đơn >, < chủ ngữ >, < vị ngữ >, ... là các phạm trù cú pháp, cho ta vai trò của các bộ phận hợp thành câu. Ta thấy một câu sinh ra qua các bước triển khai dần dần theo các quy tắc cú pháp. Đây cũng chính là dạng của các luật sinh trong văn phạm phi ngữ cảnh. Và như vậy, văn phạm phi ngữ cảnh cũng có thể chọn làm mô hình cho các văn phạm của các ngôn ngữ tự nhiên.

Tuy nhiên, trong khoa học máy tính, với nhu cầu biểu diễn các ngôn ngữ lập trình, văn phạm phi ngữ cảnh CFG còn được thiết kế thành một dạng tương đương gọi là văn phạm BNF (**B**ackus - **N**aur **F**orm)**.** Đây cũng là văn phạm CFG với những thay đổi nhỏ về dạng thức và một số ký hiệu viết tắt mà các nhà khoa học máy tính thường ứng dụng trong việc diễn tả cú pháp của các ngôn ngữ lập trình cấp cao (như ALGOL, PASCAL, ...). Trong dạng thức của văn phạm BNF, ký hiệu ::= được dùng thay cho ký hiệu →. Chẳng hạn, để định nghĩa một biểu thức số học (expression) bao gồm các danh biểu (identifier) tham gia vào các phép toán +, * hoặc biểu thức con lồng trong dấu ngoặc đơn , ta viết :

<expression> ::= <expression> + <expression>
<expression> ::= <expression> * <expression>
<expression> ::= **(** <expression> **)**
<expression> ::= <identifier>

Việc nghiên cứu các văn phạm phi ngữ cảnh đã tạo nên một cơ sở lý luận vững chắc cho việc biểu diễn ngôn ngữ lập trình, việc tìm kiếm các giải thuật phân tích cú pháp vận dụng trong chương trình dịch và cho nhiều ứng dụng khác về xử lý chuỗi. Chẳng hạn, nó rất hữu ích trong việc mô tả các biểu thức số học với nhiều dấu ngoặc lồng nhau hoặc cấu trúc khối trong ngôn ngữ lập trình mà biểu thức chính quy không thể đặc tả.

1.1. Định nghĩa

Văn phạm phi ngữ cảnh là một tập hợp hữu hạn các *biến* (còn gọi là các *ký hiệu chưa kết thúc*), mỗi biến biểu diễn một ngôn ngữ. Ngôn ngữ được biểu diễn bởi các biến được mô tả một cách đệ quy theo thuật ngữ của một khái niệm khác gọi là *ký hiệu kết thúc*. Quy tắc quan hệ giữa các biến gọi là luật sinh. Mỗi luật sinh có dạng một biến ở vế trái sinh ra một chuỗi có thể gồm biến lẫn các ký hiệu kết thúc trong văn phạm.

Văn phạm phi ngữ cảnh (CFG) là một hệ thống gồm bốn thành phần, ký hiệu là văn phạm **G** (V, T, P, S), trong đó :

. V là tập hữu hạn các biến (hay ký tự chưa kết thúc)

. T là tập hữu hạn các ký tự kết thúc, $V \cap T = \varnothing$

. P là tập hữu hạn các luật sinh mà mỗi luật sinh có dạng $A \rightarrow \alpha$ với A là biến và α là chuỗi các ký hiệu $\in (V \cup T)^*$

. S là một biến đặc biệt gọi là ký hiệu bắt đầu văn phạm.

Thí dụ 5.1 : Văn phạm G ({S, A, B}, {a, b}, P, S), trong đó P gồm các luật sinh sau:

$S \rightarrow AB$
$A \rightarrow aA$
$A \rightarrow a$
$B \rightarrow bB$
$B \rightarrow b$

Quy ước ký hiệu:

- Các chữ in hoa A, B, C, D, E, ... và S ký hiệu các biến (S thường được dùng làm ký hiệu bắt đầu).
- Các chữ nhỏ a, b, c, d, e, ...; các chữ số và một số ký hiệu khác ký hiệu cho các ký hiệu kết thúc.
- Các chữ in hoa X, Y, Z là các ký hiệu có thể là ký hiệu kết thúc hoặc biến.
- Các chữ Hi-lạp α, β, γ, ... biểu diễn cho chuỗi các ký hiệu kết thúc và biến.

Ta sẽ biểu diễn văn phạm một cách tóm tắt bằng cách chỉ liệt kê các luật sinh của nó. Nếu $A \rightarrow \alpha_1$, $A \rightarrow \alpha_2$, ... , $A \rightarrow \alpha_k$ là các luật sinh của biến A trong văn phạm nào đó, ta sẽ ghi ngắn gọn là $A \rightarrow \alpha_1 \mid \alpha_2 \mid ... \mid \alpha_k$

Thí dụ 5.2 : Văn phạm trong Thí dụ 5.1 trên có thể viết gọn là :

$S \rightarrow AB$
$A \rightarrow aA \mid a$
$B \rightarrow bB \mid b$

Câu hỏi :

Bạn nghĩ gì về lớp ngôn ngữ có thể được sinh bởi văn phạm trong ví dụ trên ? Cơ chế nào có thể được sử dụng cho văn phạm để phát sinh ngôn ngữ ?

1.2. Dẫn xuất và ngôn ngữ

Dẫn xuất: Để định nghĩa ngôn ngữ sinh bởi văn phạm CFG G (V, T, P, S), ta dẫn nhập khái niệm *dẫn xuất*. Trước hết ta giới thiệu hai quan hệ $\Rightarrow_G$ và $\Rightarrow^*_G$ giữa hai chuỗi trong tập $(V \cup T)^*$. Nếu $A \rightarrow \beta$ là một luật sinh trong văn phạm và α, γ là hai chuỗi bất kỳ trong tập $(V \cup T)^*$ thì $\alpha A\gamma \Rightarrow_G \alpha\beta\gamma$, hay ta còn nói luật sinh $A \rightarrow \beta$ áp dụng vào chuỗi $\alpha A\gamma$ để thu được chuỗi $\alpha\beta\gamma$, nghĩa là $\alpha A\gamma$ sinh trực tiếp $\alpha\beta\gamma$ trong văn phạm G. Hai chuỗi gọi là quan hệ nhau bởi $\Rightarrow_G$ nếu chuỗi thứ hai thu được từ chuỗi thứ nhất bằng cách áp dụng một luật sinh nào đó.

Giả sử $\alpha_1, \alpha_2, ..., \alpha_m$ là các chuỗi thuộc $(V \cup T)^*$ với $m \geq 1$ và :

$$\alpha_1 \Rightarrow_G \alpha_2, \alpha_2 \Rightarrow_G \alpha_3, \ldots, \alpha_{m-1} \Rightarrow_G \alpha_m$$

thì ta nói $\alpha_1 \Rightarrow^*_G \alpha_m$ hay α_1 dẫn xuất ra α_m trong văn phạm G.

Như vậy, $\Rightarrow^*_G$ là bao đóng phản xạ và bắc cầu của $\Rightarrow_G$. Nói cách khác, $\alpha \Rightarrow^*_G \beta$ nếu β được dẫn ra từ α bằng không hoặc nhiều hơn các luật sinh của P. Chú ý rằng $\alpha \Rightarrow^*_G \alpha$ với mọi chuỗi α.

Thông thường nếu không có nhầm lẫn ta sẽ dùng các ký hiệu $\Rightarrow$ và $\Rightarrow^*$ thay cho ký hiệu $\Rightarrow_G$ và $\Rightarrow^*_G$. Nếu α dẫn ra β bằng i bước dẫn xuất thì ta ký hiệu $\alpha \Rightarrow^i \beta$.

Ngôn ngữ sinh bởi văn phạm phi ngữ cảnh

Cho văn phạm CFG G(V, T, P, S), ta định nghĩa :

$$L(G) = \{w \mid w \in T^* \text{ và } S \Rightarrow^*_G w\}$$

Nghĩa là, một chuỗi thuộc L(G) nếu:

1) Chuỗi gồm toàn ký hiệu kết thúc.
2) Chuỗi được dẫn ra từ ký hiệu bắt đầu S.

Ta gọi L là ngôn ngữ phi ngữ cảnh (CFL) nếu nó là L(G) với một CFG G nào đó. Chuỗi α gồm các ký hiệu kết thúc và các biến, được gọi là một dạng câu sinh từ G nếu $S \Rightarrow^* \alpha$. Hai văn phạm G_1, G_2 được gọi là tương đương nếu $L(G_1) = L(G_2)$

Thí dụ 5.3 : Xét văn phạm G (V, T, P, S), trong đó :

$V = \{S\}$, $T = \{a, b\}$, $P = \{S \rightarrow aSb, S \rightarrow ab\}$.

Bằng cách áp dụng luật sinh thứ nhất n -1 lần và luật sinh thứ hai 1 lần, ta có:

$$S \Rightarrow aSb \Rightarrow aaSbb \Rightarrow a^3Sb^3 \Rightarrow \ldots \Rightarrow a^{n-1}b^{n-1} \Rightarrow a^nb^n$$

Vậy, L(G) chứa các chuỗi có dạng a^nb^n, hay $L(G) = \{a^nb^n \mid n \geq 1\}$.

1.3. Cây dẫn xuất

Để dễ hình dung sự phát sinh ra các chuỗi trong văn phạm phi ngữ cảnh, ta thường diễn tả một chuỗi dẫn xuất qua hình ảnh một cây. Một cách hình thức, ta định nghĩa như sau:

Định nghĩa : Cho văn phạm G (V, T, P, S). Cây dẫn xuất (hay cây phân tích cú pháp) của G được định nghĩa như sau :

i) Mỗi nút (đỉnh) có một nhãn, là một ký hiệu $\in (V \cup T \cup \{\varepsilon\})$

ii) Nút gốc có nhãn là ký hiệu bắt đầu S.

iii) Nếu nút trung gian có nhãn A thì $A \in V$

iv) Nếu nút n có nhãn A và các đỉnh $n_1, n_2, ..., n_k$ là con của n theo thứ tự từ trái sang phải có nhãn lần lượt là $X_1, X_2, ..., X_k$ thì $A \rightarrow X_1X_2 ... X_k$ là một luật sinh trong tập luật sinh P.

v) Nếu nút n có nhãn là từ rỗng ε thì n phải là nút lá và là nút con duy nhất của nút cha của nó.

Thí dụ 5.4 : Xét văn phạm G ({S, A}, {a, b}, P, S), trong đó P gồm:

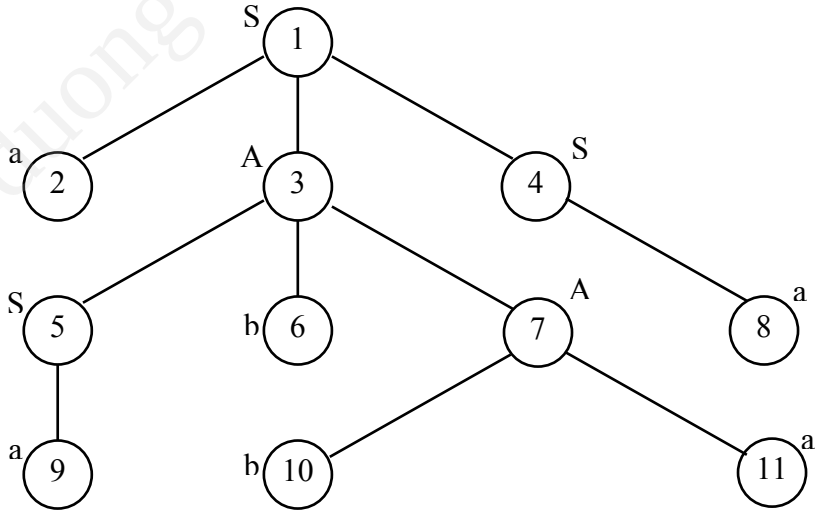
$S \rightarrow aAS \mid a$

$A \rightarrow SbA \mid SS \mid ba$

Một cây dẫn xuất từ văn phạm có dạng như hình 5.1 sau :

Ta thấy, nút 1 có nhãn S và các con của nó lần lượt là a, A, S (chú ý $S \rightarrow aAS$ là một luật sinh). Tương tự, nút 3 có nhãn A và các con của nó là S, b, A (từ luật sinh $A \rightarrow SbA$). Nút 4, 5 có cùng nhãn S và có nút con nhãn a (luật sinh $S \rightarrow a$). Cuối cùng nút 7 có nhãn A và có các nút con b, a (luật sinh $A \rightarrow ba$).

Trên cây dẫn xuất, nếu ta đọc các lá theo thứ tự từ "trái sang phải" thì ta có một dạng câu trong G. Ta gọi chuỗi này là chuỗi sinh bởi cây dẫn xuất.



Hình 5.1 - *Cây dẫn xuất từ văn phạm*

Một cây con (subtree) của cây dẫn xuất có nút gốc nhãn là A còn được gọi là A-cây con (hoặc A-cây). Cây con cũng giống như cây dẫn xuất, chỉ khác là nhãn của nút gốc không nhất thiết phải là ký hiệu bắt đầu S.

Thí dụ 5.5 : Xét văn phạm và cây dẫn xuất trong Hình 5.1. Đọc các lá theo thứ tự từ trái sang phải ta có chuỗi **aabbaa**, trong trường hợp này tất cả các lá đều là ký hiệu kết thúc, nhưng nói chung cũng không bắt buộc như thế, lá có thể có nhãn là ε hoặc biến. Ta thấy dẫn xuất $S \Rightarrow^* aabbaa$ bằng chuỗi dẫn xuất :

$$S \Rightarrow aAS \Rightarrow aSbAS \Rightarrow aabAS \Rightarrow aabbaS \Rightarrow aabbaa$$

A-cây có nút đỉnh là 3 tạo ra chuỗi con **abba** theo chuỗi dẫn xuất :

$$S \Rightarrow SbA \Rightarrow abA \Rightarrow abba$$

Câu hỏi :

Các cây dẫn xuất được sinh từ những chuỗi dẫn xuất khác nhau cho cùng một chuỗi nhập có là những cây dẫn xuất khác nhau không ?

1.4. Quan hệ giữa dẫn xuất và cây dẫn xuất

ĐỊNH LÝ 5.1 : Nếu G (V, T, P, S) là một văn phạm phi ngữ cảnh thì $S \Rightarrow^* \alpha$ nếu và chỉ nếu có cây dẫn xuất trong văn phạm sinh ra α.

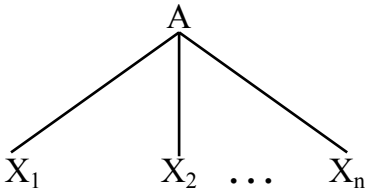
Chứng minh

Ta chứng minh rằng với biến A bất kỳ, $A \Rightarrow^* \alpha$ nếu và chỉ nếu có một A-cây sinh ra α.

***Nếu**: Giả sử α được sinh bởi A-cây, ta chứng minh quy nạp theo số nút trung gian của cây dẫn xuất rằng $A \Rightarrow^* \alpha$.*

Nếu có 1 nút trung gian thì cây phải có dạng như hình sau :

Khi đó $X_1X_2 \ldots X_n$ là chuỗi α và $A \rightarrow \alpha$ là một luật sinh trong P theo định nghĩa cây dẫn xuất.



Hình 5.2(a) - *A-cây với một nút trong*

Giả sử kết quả đúng tới k -1 nút trung gian (k > 1)

Ta chứng minh kết quả cũng đúng với k nút.

Xét α được sinh ra bởi A-cây có k nút trung gian. Rõ ràng các nút con của nút gốc không phải tất cả đều là lá, ta gọi chúng từ trái sang phải là X_1, X_2, ..., X_n thì chắc chắn rằng $A \rightarrow X_1X_2 \ldots X_n$ là một luật sinh. Xét nút X_i bất kỳ :

- Nếu X_i không là nút lá thì X_i phải là một biến và X_i - cây con sẽ sinh ra một chuỗi α_i nào đó.

- Nếu X_i là nút lá, ta đặt $\alpha_i = X_i$. Dễ thấy rằng nếu $j < i$ thì các α_j ở bên trái α_j, do vậy chuỗi đọc từ lá vẫn có dạng $\alpha = \alpha_1\alpha_2 \ldots \alpha_n$. Mỗi X_i - cây con phải có ít nút

trung gian hơn cây ban đầu, vì thế theo giả thiết quy nạp, với mỗi đỉnh i không phải là lá thì $X_i \Rightarrow^* \alpha_i$.

Vậy $A \Rightarrow X_1X_2 \ldots X_n \Rightarrow^* \alpha_1X_2 \ldots X_n \Rightarrow^* \alpha_1\alpha_2X_3 \ldots X_n \Rightarrow^* \ldots \Rightarrow^* \alpha_1\alpha_2 \ldots \alpha_n = \alpha$

Hay ta có $A \Rightarrow^* \alpha$ *. Chú ý rằng đây chỉ là một trong nhiều cách dẫn xuất ra* α.

Chỉ nếu *: Ngược lại, giả sử* $A \Rightarrow^* \alpha$ *ta cần chỉ ra một A - cây sinh ra* α.

Nếu $A \Rightarrow^* \alpha$ *bằng một bước dẫn xuất thì* $A \rightarrow \alpha$ *là một luật sinh trong P và có cây dẫn xuất sinh ra* α *như trong hình trên.*

Giả sử kết quả đúng tới k-1 bước dẫn xuất

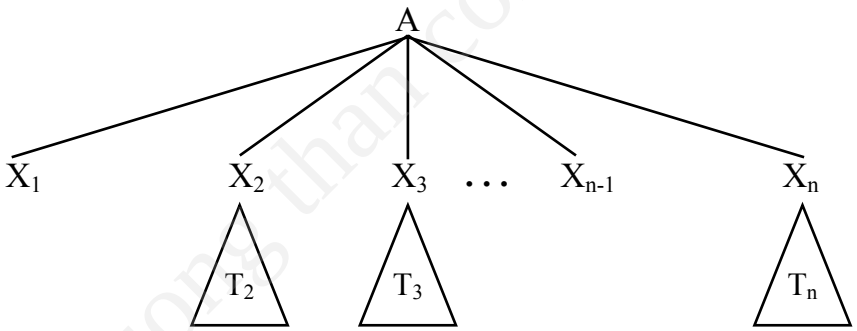
Xét $A \Rightarrow^* \alpha$ *bằng k bước dẫn xuất, gọi bước đầu tiên là* $A \rightarrow X_1X_2 \ldots X_n$.

Rõ ràng, một ký hiệu trong α *phải được dẫn ra từ một biến* X_i *nào đó. Vì vậy, ta có thể viết* $\alpha = \alpha_1\alpha_2 \ldots \alpha_n$, *trong đó mỗi* $1 \le i \le n$ *thoả mãn :*

- $\alpha_i = X_i$ *nếu* X_i *là ký hiệu kết thúc.*
- $X_i \Rightarrow^* \alpha_i$ *nếu* X_i *là một biến.*

Nếu X_i *là biến thì dẫn xuất của* α_i *từ* X_i *phải có ít hơn k bước. Vì vậy, theo giả thiết quy nạp ta có* X_i *- cây sinh ra* α_i, *đặt cây này là* T_i

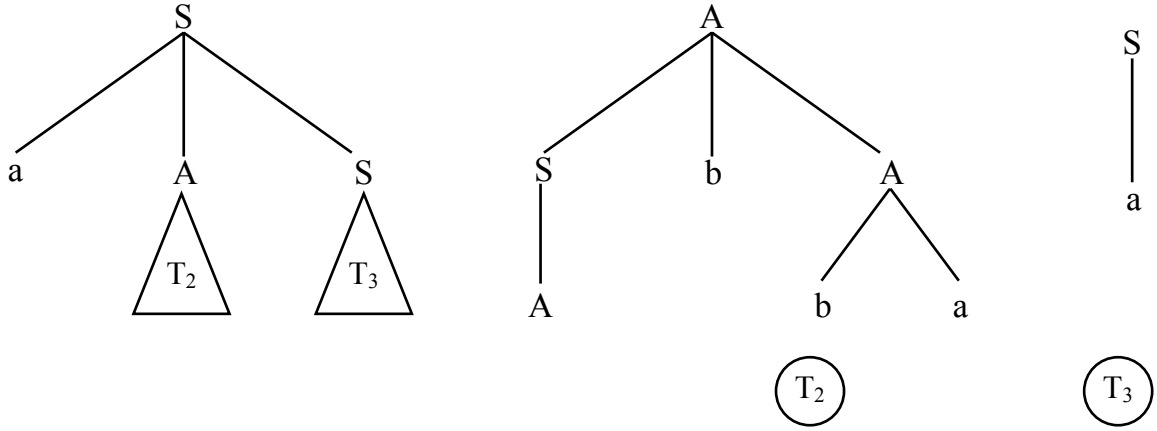
Bây giờ ta dựng A - cây có n lá $X_1X_2 \ldots X_n$. *Mỗi* X_i *không là ký hiệu kết thúc ta thay bằng cây* T_i *tương ứng. Cuối cùng, ta có cây dẫn xuất sinh ra có dạng như sau :*



Hình 5.2(b) - *A-cây*

Thí dụ 5.6 : Xét chuỗi dẫn xuất $S \Rightarrow^*$ **aabbaa** cho văn phạm ở Thí dụ 5.4.

Bước đầu tiên trong dẫn xuất đó là S → aAS. Theo dõi các bước suy dẫn sau đó, ta thấy biến A được thay bởi SbA, rồi trở thành abA và cuối cùng thành abba, đó chính là kết quả của cây T_2 (A - cây). Còn biến S thì được thay bởi a và đó là kết quả của cây T_3 (S -cây). Ghép nối lại, ta được cây dẫn xuất mà kết quả là chuỗi **aabbaa** như dưới đây.



Hình 5.3 - *Ghép nối các cây dẫn xuất*

1.5. Dẫn xuất trái nhất, dẫn xuất phải nhất

Nếu tại mỗi bước dẫn xuất, luật sinh được áp dụng vào biến bên trái nhất thì ta gọi đó là *dẫn xuất trái nhất* (leftmost) hay dẫn xuất trái. Tương tự, nếu biến bên phải nhất được thay thế ở mỗi bước dẫn xuất, đó là *dẫn xuất phải nhất* (rightmost) hay dẫn xuất phải. Nếu chuỗi w $\in$ L(G) với CFG G thì w sẽ có ít nhất một cây dẫn xuất ra nó và tương ứng với các cây này, w chỉ có duy nhất một dẫn xuất trái nhất và duy nhất một dẫn xuất phải nhất. Dĩ nhiên, w có thể có nhiều dẫn xuất trái (phải) nhất vì nó có thể có nhiều cây dẫn xuất.

Thí dụ 5.7 : Xét cây dẫn xuất ở Hình 5.1

. Dẫn xuất trái nhất của cây :

$S \Rightarrow a\mathbf{A}S \Rightarrow a\mathbf{S}bAS \Rightarrow aab\mathbf{A}S \Rightarrow aabba\mathbf{S} \Rightarrow aabbaa.$

. Dẫn xuất phải nhất tương ứng là :

$S \Rightarrow aA\mathbf{S} \Rightarrow a\mathbf{A}a \Rightarrow aSb\mathbf{A}a \Rightarrow a\mathbf{S}baa \Rightarrow aabbaa.$

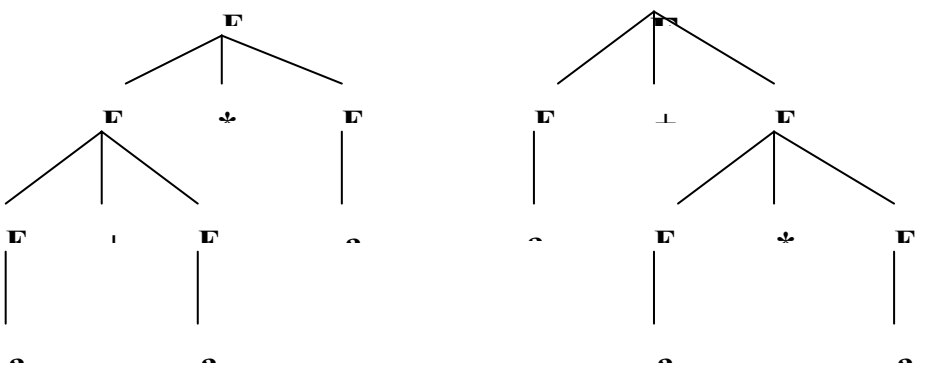
1.6. Văn phạm mơ hồ

Một văn phạm phi ngữ cảnh G có nhiều hơn một cây dẫn xuất cho cùng một chuỗi w, thì G được gọi là *văn phạm mơ hồ* (ambiguity). Dĩ nhiên, cũng có thể nói rằng văn phạm G là mơ hồ nếu có một chuỗi w được dẫn ra từ ký hiệu bắt đầu S với hai dẫn xuất trái hoặc hai dẫn xuất phải.

Thí dụ 5.8 : Xét văn phạm G với các luật sinh như sau :

$$E \rightarrow E + E \mid E * E \mid (E) \mid a$$

Văn phạm này sinh ra các chuỗi biểu thức số học với 2 phép toán + và * . Với chuỗi **a + a * a**, ta có thể vẽ đến hai cây dẫn xuất khác nhau như sau :



(a) **(b)**

Hình 5.4 - *Các cây dẫn xuất khác nhau cho cùng chuỗi nhập*

Điều này có nghĩa là biểu thức **a + a * a** có thể hiểu theo hai cách khác nhau: thực hiện phép cộng trước hay phép nhân trước ? Để khắc phục sự mơ hồ này, ta có thể :

- Hoặc quy định rằng các phép cộng và nhân luôn luôn được thực hiện theo thứ tự từ trái sang phải (trừ khi gặp ngoặc đơn). Ta viết văn phạm G_1 không mơ hồ tương đương như sau :

$$E \rightarrow E + T \mid E * T \mid T$$
$$T \rightarrow (E) \mid a$$

- Hoặc quy định rằng khi không có dấu ngoặc đơn ngăn cách thì phép nhân luôn luôn được ưu tiên hơn phép cộng. Ta viết văn phạm G_2 không mơ hồ tương đương như sau :

$$E \rightarrow E + T \mid T$$
$$T \rightarrow T * F \mid F$$
$$F \rightarrow (E) \mid a$$

II. GIẢN LƯỢC CÁC VĂN PHẠM PHI NGỮ CẢNH

Thường thì một văn phạm phi ngữ cảnh có thể còn chứa đựng một vài yếu tố thừa, vô ích. Chẳng hạn như theo các đặc tính trên, có những ký hiệu không thực sự tham gia vào quá trình dẫn xuất ra câu, hoặc sẽ có những luật sinh dạng $A \rightarrow B$ làm kéo dài chuỗi dẫn xuất một cách không cần thiết. Vì vậy, việc giản lược văn phạm phi ngữ cảnh là nhằm loại bỏ những yếu tố vô ích đó mà không làm giảm bớt khả năng sản sinh ngôn ngữ của văn phạm.

Nếu L là một CFL, nó có thể tạo ra văn phạm CFG với các đặc tính sau :

1) Mỗi biến và mỗi ký hiệu kết thúc của G đều xuất hiện trong dẫn xuất của một số chuỗi trong L.
2) Không có luật sinh nào dạng $A \rightarrow B$, mà trong đó A, B đều là biến.

Hơn nữa, nếu $\varepsilon \notin L$ thì không cần luật sinh $A \rightarrow \varepsilon$. Thực tế, nếu $\varepsilon \notin L$, ta có mọi luật sinh trong G đều có một trong hai dạng :

$A \rightarrow BC$ hoặc $A \rightarrow a\alpha$ (α là chuỗi các biến hoặc ε)
$A \rightarrow a$

Hai dạng đặc biệt này gọi là dạng chuẩn Chomsky và dạng chuẩn Greibach.

2.1. Các ký hiệu vô ích

Một ký hiệu X gọi là có ích nếu có một dẫn xuất $S \Rightarrow^* \alpha X \beta \Rightarrow^* w$ với các chuỗi α, β bất kỳ và $w \in T^*$. Ngược lại X gọi là vô ích.

Vậy, có 2 đặc điểm cho ký hiệu có ích:

- X phải dẫn ra một chuỗi ký hiệu kết thúc.
- X phải nằm trong dẫn xuất từ S.

Tuy nhiên 2 dấu hiệu trên không đủ để đảm bảo X có ích vì X có thể nằm trong dạng câu chứa một biến nhưng từ đó không có ký hiệu kết thúc được sinh ra.

BỔ ĐỀ 1: *(Dùng loại bỏ các biến không dẫn ra chuỗi ký hiệu kết thúc)*

Cho CFG G (V, T, P, S) với L(G) ≠ ∅, có một CFG G' (V', T', P', S) tương đương sao cho mỗi A ∈ V' tồn tại w ∈ T* để A ⇒* w.

Chứng minh

Mỗi biến A với luật sinh $A \rightarrow w$ trong P thì rõ ràng A ∈ V'. Nếu $A \rightarrow X_1X_2 \ldots X_n$ là một luật sinh, trong đó mỗi X_i hoặc là ký hiệu kết thúc hoặc là một biến đã có sẵn trong V' thì một chuỗi các ký hiệu kết thúc có thể được dẫn ra từ A bằng dẫn xuất bắt đầu $A \Rightarrow X_1X_2 \ldots X_n$, vì vậy A ∈ V'. Tập V' có thể tính được bằng cách lặp lại giải thuật trên. P' là tập tất cả các luật sinh mà các ký hiệu của nó thuộc $V' \cup T$.

Giải thuật tìm V' như sau:

```
Begin
(1)  OLDV:= ∅;
(2)  NEWV:= {A | A → w với w ∈ T*};
(3)  While  OLDV ≠ NEWV do
       begin
(4)      OLDV := NEWV;
(5)      NEWV := OLDV ∪ {A | A → α với α ∈ (T ∪ OLDV)*}
     end;
(6)  V' := NEWV;
end;
```

Rõ ràng rằng nếu biến A được thêm vào V' tại bước (2) hoặc (5) thì A sẽ dẫn ra được chuỗi ký hiệu kết thúc. Ta chứng minh rằng nếu A dẫn ra được một chuỗi ký hiệu kết thúc thì A được thêm vào tập NEWV.

Dùng chứng minh quy nạp theo độ dài của dẫn xuất $A \Rightarrow^* w$.

Nếu độ dài bằng 1 thì $A \rightarrow \alpha$ là một luật sinh trong P. Vậy A được đưa vào V' tại bước (2).

Giả sử kết quả đúng tới k -1 bước dẫn xuất (k >1)

Nếu $A \Rightarrow X_1X_2 \ldots X_n \Rightarrow^* w$ bằng k bước thì ta có thể viết $w = w_1w_2 \ldots w_n$, trong đó $X_i \Rightarrow^* w_i$, với $1 \le i \le n$ bằng ít hơn k bước dẫn xuất. Theo giả thiết quy nạp thì các

biến X_i này được thêm vào V'. Khi X_i cuối cùng được thêm vào V' thì vòng lặp (3) vẫn tiếp tục lặp một lần nữa và A sẽ được thêm vào V' tại (5).

Ta chứng minh L(G') = L(G) :

Chọn V' là tập hợp tại (6) và P' là tập tất cả các luật sinh mà các ký hiệu của nó thuộc (V' ∪ T) thì chắc chắn rằng có tồn tại văn phạm G' (V', T, P', S) thoả mãn tính chất: nếu A ∈ V' thì A $\Rightarrow^*$ w với w nào đó thuộc T^*. Hơn nữa, mỗi luật sinh của P' đều là luật sinh của P nên ta có L(G') ⊆ L(G).

Ngược lại giả sử một từ w ∈ L(G) - L(G') thì một dẫn xuất bất kỳ của w phải liên quan đến các biến thuộc V – V' hoặc luật sinh thuộc P – P' (các dẫn xuất này đưa ra các biến thuộc V – V'), nhưng do không có biến nào trong V – V' dẫn đến chuỗi kết thúc, điều này dẫn đến mâu thuẫn.

Vậy L(G') = L(G).

Hay có thể nói 2 ngôn ngữ được cho từ 2 văn phạm G và G' là tương đương nhau, hay nói cách khác: nếu có một văn phạm G thì luôn luôn có một văn phạm G' tương ứng mà trong đó mỗi biến của G' đều cho ra ký hiệu kết thúc.

BỔ ĐỀ 2: *(Dùng loại bỏ các biến không được dẫn ra từ ký hiệu bắt đầu văn phạm)*

Nếu G (V, T, P, S) là CFG thì ta có thể tìm được CFG G' (V', T', P', S) tương đương sao cho mỗi X ∈ V' ∪ T' tồn tại α, β ∈ (V' ∪ T')* để S $\Rightarrow^*$ αXβ.

Chứng minh

Tập V' ∪ T' gồm các ký hiệu xuất hiện trong dạng câu của G được xây dựng bởi giải thuật lặp như sau :

. Đặt V' = {S}; T' = ∅;

. Nếu A ∈ V' và A → $\alpha_1 | \alpha_2 | ... | \alpha_n$ là các luật sinh trong P thì thêm tất cả các biến của $\alpha_1, \alpha_2, ..., \alpha_n$ vào V' và các ký hiệu kết thúc của $\alpha_1, \alpha_2, ..., \alpha_n$ vào T'.

. Lặp lại giải thuật cho đến khi không còn biến hoặc ký hiệu kết thúc nào được thêm vào nữa.

Dễ thấy, X ∈ V' ∪ T' thì tồn tại α, β ∈ (V' ∪ T')* để S $\Rightarrow^*$ αXβ, trong đó P' là tập hợp tất cả các luật sinh của P chỉ chứa các ký hiệu thuộc (V' ∪ T').

Ta dễ dàng chứng minh L(G') = L(G) .

ĐỊNH LÝ 5.2: Mỗi ngôn ngữ phi ngữ cảnh (CFL) không rỗng được sinh ra từ một văn phạm phi ngữ cảnh (CFG) không có ký hiệu vô ích.

Chứng minh

Đặt L = L(G) là CFL không rỗng.

Đặt G_1 là kết quả của việc áp dụng bổ đề 1 vào G và G_2 là kết quả của việc áp dụng bổ đề 2 vào G_1.

Giả sử G_2 có ký hiệu vô ích là X. Theo bổ đề 2 ta có $S \Rightarrow^*_{G2} \alpha X \beta$. Vì tất cả các ký hiệu trong G_2 đều có trong G_1 nên theo bổ đề 1: $S \Rightarrow^*_{G1} \alpha X \beta \Rightarrow^*_{G1} w$ với w là chuỗi ký hiệu kết thúc. Vì vậy không có ký hiệu nào trong dẫn xuất $\alpha X \beta \Rightarrow^*_{G1} w$ bị loại bỏ bởi bổ đề 2, vậy X dẫn ra ký hiệu kết thúc trong G_2 . Suy ra X là ký hiệu có ích (mâu thuẫn).

Vậy văn phạm G_2 không có ký hiệu vô ích nào.

Thí dụ 5.9 : Xét văn phạm có các luật sinh sau :

$$S \rightarrow AB \mid a$$
$$A \rightarrow a$$

Áp dụng bổ đề 1, ta thấy không có ký hiệu kết thúc được nào dẫn ra từ B nên ta loại bỏ B và luật sinh $S \rightarrow AB$. Tiếp tục, áp dụng bổ đề 2 cho văn phạm :

$$S \rightarrow a$$
$$A \rightarrow a$$

Ta thấy chỉ có S xuất hiện trong dạng câu. Vậy ({S}, {a}, {S → a}, S) là văn phạm tương đương với văn phạm đã cho và không có ký hiệu vô ích.

Câu hỏi :

Bạn hãy cho nhận xét về thứ tự áp dụng Bổ đề 1 và Bổ đề 2 trong quá trình loại bỏ các ký hiệu vô ích trong văn phạm ?

2.2. Luật sinh ε

Một luật sinh có dạng $A \rightarrow \varepsilon$ gọi là luật sinh ε.

Ta xét đến việc loại bỏ các luật sinh này. Nếu $\varepsilon \in L(G)$ thì không thể loại được tất cả các luật sinh ε, nhưng nếu $\varepsilon \notin L(G)$ thì có thể. Phương pháp loại bỏ dựa trên việc xác định liệu một biến A có dẫn xuất $A \Rightarrow^* \varepsilon$ hay không ? Nếu có, ta gọi A là biến rỗng (nullable). Ta có thể thay thế mỗi luật sinh $B \rightarrow X_1X_2 \ldots X_n$ bằng tất cả các luật sinh được định dạng bởi việc xóa bỏ tập hợp con các biến X_i rỗng, nhưng không bao gồm luật sinh $B \rightarrow \varepsilon$, ngay cả khi tất cả các X_i đều là biến rỗng.

ĐỊNH LÝ 5.3 : Nếu L = L(G) với CFG G (V, T, P, S) thì L - { ε } là L(G') với CFG G' không có ký hiệu vô ích và không có luật sinh ε.

Chứng minh

Ta có thể xác định tập hợp các biến rỗng (nullable) của G bằng giải thuật lặp như sau : Bắt đầu, nếu $A \rightarrow \varepsilon$ là một luật sinh thì A là biến rỗng. Kế tiếp, nếu $B \rightarrow \alpha$, trong đó α gồm toàn các ký hiệu là các biến rỗng đã được tìm thấy trước đó thì B cũng là biến rỗng. Lặp lại cho đến khi không còn biến rỗng nào được tìm thấy nữa.

Tập luật sinh P' được xây dựng như sau : Nếu $A \to X_1X_2 \ldots X_n$ là một luật sinh trong P thì ta thêm tất cả các luật sinh $A \to \alpha_1\alpha_2 \ldots \alpha_n$ vào P' với điều kiện :

1) Nếu X_i không là biến rỗng thì $\alpha_i = X_i$;
2) Nếu X_i là biến rỗng thì α_i là X_i hoặc ε;
3) Không phải tất cả α_i đều bằng ε.

Đặt G'' = (V, T, P', S). Ta sẽ chứng minh rằng với mọi $A \in V$ và $w \in T^*$, $A \Rightarrow^*_{G''} w$ nếu và chỉ nếu $w \neq \varepsilon$ và $A \Rightarrow^*_G w$.

Nếu: *Đặt $A \Rightarrow^i_G w$ và $w \neq \varepsilon$, ta chứng minh quy nạp rằng $A \Rightarrow^*_{G''} w$.*

Nếu i = 1 ta có $A \to w$ là một luật sinh trong P, và vì $w \neq \varepsilon$ nên luật sinh này cũng thuộc P'.

Giả sử kết quả đúng tới i - 1 (i >1)

Xét $A \Rightarrow_G X_1X_2 \ldots X_n \Rightarrow^{i-1}_G w$. Ta viết $w = w_1w_2 \ldots w_n$ sao cho $\forall j$, $X_j \Rightarrow^* w_j$.

Nếu $w_j \neq \varepsilon$ và X_j là biến thì theo giả thiết quy nạp, ta có $X_j \Rightarrow^*_{G''} w_j$ (vì dẫn xuất $X_j \Rightarrow^* w_j$ có ít hơn i bước). Nếu $w_j = \varepsilon$ thì X_j là biến rỗng, vậy $A \to \beta_1\beta_2 \ldots\beta_n$ là một luật sinh trong P', trong đó $\beta_j = X_j$ nếu $w_j \neq \varepsilon$ và $\beta_j = \varepsilon$ nếu $w_j = \varepsilon$.

Vì $w \neq \varepsilon$ nên không phải tất cả β_j là ε. Vậy, ta có dẫn xuất :

$A \Rightarrow \beta_1\beta_2 \ldots\beta_n \Rightarrow^* w_1\beta_2 \ldots\beta_n \Rightarrow^* w_1w_2\beta_3 \ldots\beta_n \Rightarrow^* \ldots \Rightarrow^* w_1w_2 \ldots w_n = w$ trong G''.

Chỉ nếu*: Giả sử $A \Rightarrow^i_{G''} w$. Chắc chắn rằng $w \neq \varepsilon$ vì G'' không có luật sinh ε. Ta quy nạp theo i rằng $A \Rightarrow_G w$.*

Nếu i = 1: Ta thấy $A \to w$ là một luật sinh trong P', do đó cũng phải có luật sinh $A \to w$ trong P sao cho bằng việc loại bỏ các ký hiệu rỗng trong α, ta có w. Vậy, có tồn tại dẫn xuất $A \Rightarrow_G \alpha \Rightarrow^*_G w$, trong đó $\alpha \Rightarrow^* w$ liên quan đến các dẫn xuất ε từ các biến rỗng của α mà chúng ta đã loại bỏ khỏi w.

Giả sử kết quả đúng tới i - 1 (i >1)

Xét $A \Rightarrow_{G''} X_1X_2 \ldots X_n \Rightarrow^{i-1}_{G''} w$. Phải có luật sinh $A \to \beta$ trong P sao cho $X_1X_2 \ldots X_n$ tìm được khi loại bỏ các biến rỗng của β. Vậy $A \Rightarrow^*_G X_1X_2 \ldots X_n$ (chứng minh tương tự như ở trên). Ta viết $w = w_1w_2 \ldots w_n$ sao cho $\forall j$ ta có $X_j \Rightarrow^*_{G''} w_j$ (bằng ít hơn i bước). Theo giả thiết quy nạp $X_j \Rightarrow^*_{G''} w_j$ nếu X_j là biến. Nếu X_j là ký hiệu kết thúc thì $w_j = X_j$ và $X_j \Rightarrow^*_G w_j$ là hiển nhiên. Vậy $A \Rightarrow^*_G w$.

Cuối cùng ta áp dụng bổ đề 2 vào G'' ta thu được G' không có ký hiệu vô ích. Vì bổ đề 1 và bổ đề 2 không đưa ra thêm luật sinh mới nào nên G' không có chứa ký hiệu là biến rỗng hay ký hiệu vô ích.

Hơn nữa $S \Rightarrow^*_{G'} w$ nếu và chỉ nếu $S \Rightarrow^*_G w$. Vậy $L(G') = L(G) - \{\varepsilon\}$.

Thí dụ 5.10 : Loại bỏ luật sinh ε trong văn phạm sau :

$$S \to AB$$
$$A \to aA \mid \varepsilon$$
$$B \to bB \mid \varepsilon$$

Trước hết, ta xác định tập các biến rỗng trong văn phạm: A, B là các biến rỗng vì có các luật sinh $A \to \varepsilon$ và $B \to \varepsilon$. S cũng là biến rỗng vì có luật sinh $S \to AB$ với A, B đều là các biến rỗng.

$\Rightarrow$ Tập biến rỗng Nullable = {A, B, S}

Theo quy tắc xây dựng tập luật sinh P' trong định lý , ta có tập luật sinh mới như sau :

$S \rightarrow AB \mid A \mid B$
$A \rightarrow aA \mid a$
$B \rightarrow bB \mid b$

Lưu ý rằng văn phạm mới G' không sản sinh ra ε, trong khi G lại có sinh ra từ rỗng ε. Vậy muốn có một văn phạm thực sự tương đương với văn phạm G thì ta phải bổ sung thêm luật sinh $S \rightarrow \varepsilon$ vào tập luật sinh của G'. Ta có, văn phạm G' tương đương G.

2.3. Luật sinh đơn vị

Một luật sinh có dạng $A \rightarrow B$ với A, B đều là biến gọi là luật sinh đơn vị.

<u>ĐỊNH LÝ 5.4</u> : Mỗi CFL không chứa ε được sinh ra bởi CFG không có ký hiệu vô ích, luật sinh ε hoặc luật sinh đơn vị .

Chứng minh

Đặt L là CFL không chứa ε và L = L(G) với G (V, T, P, S) là một CFG nào đó.
Theo định lý 3 ta có thể giả sử G không có luật sinh ε. Xây dựng tập hợp mới P' gồm các luật sinh từ P như sau:

Đầu tiên đưa các luật sinh không là luật sinh đơn vị vào P'.

Sau đó, nếu có luật sinh đơn vị dạng $A \Rightarrow^* B$ với $A, B \in V$ thì thêm vào P' tất cả các luật sinh dạng $A \rightarrow \alpha$, với $B \rightarrow \alpha$ không phải là luật sinh đơn vị của P.

Chú ý rằng ta có thể dễ dàng kiểm tra có hay không $A \Rightarrow^*_G B$ vì G không có luật sinh ε và nếu $A \Rightarrow_G B_1 \Rightarrow_G B_2 \ldots \Rightarrow_G B_m \Rightarrow_G B$ (trong đó một vài biến nào đó có thể xuất hiện 2 lần) thì ta có thể tìm một chuỗi rút ngắn hơn $A \Rightarrow^*_G B$, vì vậy ta chỉ xét các luật sinh đơn vị không có biến lặp lại.
Bây giờ ta sửa lại văn phạm G' (V, T, P', S). Chắc chắn rằng nếu $A \rightarrow \alpha$ là một luật sinh trong P' thì $A \Rightarrow^*_G \alpha$. Vậy nếu có dẫn xuất trong G' thì có dẫn xuất trong G. Giả sử rằng $w \in L(G)$. Xét dẫn xuất trái của w trong G:

$$S \Rightarrow_G \alpha_0 \Rightarrow_G \alpha_1 \Rightarrow_G \ldots \Rightarrow_G \alpha_n = w.$$

Nếu $0 \le i < n$ thì nếu trong G có $\alpha_i \Rightarrow_G \alpha_{i+1}$ bằng luật sinh không là luật sinh đơn vị thì trong G' cũng có $\alpha_i \Rightarrow_{G'} \alpha_{i+1}$ không là luật sinh đơn vị. Giả sử $\alpha_i \Rightarrow_G \alpha_{i+1}$ bằng luật sinh đơn vị, nhưng bước dẫn xuất trước đó $\alpha_{i-1} \Rightarrow \alpha_i$ không phải bằng luật sinh đơn vị hoặc i = 0. Và chuỗi dẫn xuất trong G từ $\alpha_{i+1} \Rightarrow_G \alpha_{i+2} \Rightarrow_G \ldots \Rightarrow_G \alpha_j$ tất cả đều bằng luật sinh đơn vị, còn từ $\alpha_j \Rightarrow_G \alpha_{j+1}$ không là luật sinh đơn vị thì ta thấy tất cả các $\alpha_i, \alpha_{i+1}, \ldots, \alpha_j$ sẽ có cùng độ dài và vì chuỗi dẫn xuất là dẫn xuất trái nên các ký hiệu thay thế phải ở cùng một vị trí. Do vậy, tại vị trí này $\alpha_j \Rightarrow_G \alpha_{j+1}$ bằng một luật sinh nào đó thuộc P'- P hay có nghĩa là một luật sinh không thuộc văn phạm G. Điều này sinh ra mâu thuẫn. Vậy L(G) = L(G').

Ta còn có G' không có chứa luật sinh đơn vị (theo chứng minh trên) nên G cũng sẽ không chứa luật sinh đơn vị (do G ⇔ G').

Việc áp dụng bổ đề 1, bổ đề 2 để loại các ký hiệu vô ích không đưa ra thêm luật sinh nào chứng tỏ G không chứa ký hiệu vô ích.

Vậy, kết quả ta được một văn phạm thỏa điều kiện định lý.

Thí dụ 5.11 : Loại bỏ các luật sinh đơn vị trong văn phạm sau :

$$E \rightarrow E + T \mid T$$
$$T \rightarrow T * F \mid F$$
$$F \rightarrow (E) \mid a$$

Gọi tập $\Delta_A = \{B \mid A \Rightarrow^* B\}$, xét các biến trong văn phạm, ta có :

$\Delta_E = \{ E, T, F \}$ $\Delta_T = \{ T, F \}$ $\Delta_F = \{ F \}$

Vậy tập luật sinh mới, theo định lý sẽ chứa các luật sinh không là luật sinh đơn vị trong P, bổ sung thêm các luật sinh mới thay cho luật sinh đơn vị như sau :

$$E \rightarrow \mathbf{E + T} \mid T * F \mid (E) \mid a$$
$$T \rightarrow \mathbf{T * F} \mid (E) \mid a$$
$$F \rightarrow \mathbf{(E)} \mid \mathbf{a}$$

III. CHUẨN HÓA VĂN PHẠM PHI NGỮ CẢNH

Phần này sẽ giới thiệu hai định lý dùng chuẩn hóa CFG về một trong hai dạng chuẩn Chomsky và Greibach.

3.1. Dạng chuẩn Chomsky - CNF (Chomsky Normal Form)

ĐỊNH LÝ 5.5 : *(Dạng chuẩn Chomsky, hay CNF)*

Một ngôn ngữ phi ngữ cảnh bất kỳ không chứa ε đều được sinh ra bằng một văn phạm nào đó mà các luật sinh có dạng A → BC hoặc A → a, với A, B, C là biến còn a là ký hiệu kết thúc.

Chứng minh

Đặt G là CFG sinh ra CFL không chứa ε. CFG tương đương có dạng chuẩn Chomsky có thể xây dựng từ G theo giải thuật sau :

Bước 1 : *Thay thế tất cả các luật sinh có độ dài vế phải bằng 1 (luật sinh đơn vị dạng A → B, với A, B là biến)*

Theo định lý 4.4, ta có thể tìm được CFG tương đương $G_1(V, T, P, S)$ không có luật sinh đơn vị và luật sinh ε. Vậy nếu luật sinh mà vế phải chỉ có một ký hiệu thì đó phải là ký hiệu kết thúc và luật sinh này là luật sinh có dạng đúng trong định lý.

Bước 2 : *Thay thế các luật sinh có độ dài vế phải >1 và có chứa ký hiệu kết thúc.*

Xét luật sinh trong P có dạng $A \rightarrow X_1X_2 \ldots X_m$ (m >1). Nếu X_i là ký hiệu kết thúc a thì ta đưa thêm một biến mới C_a và luật sinh mới $C_a \rightarrow a$. Thay thế X_i bởi C_a, gọi tập các biến mới là V' và tập luật sinh mới là P'.

Xét CFG G_2 (V', T, P', S). Nếu $\alpha \Rightarrow_{G1} \beta$ thì $\alpha \Rightarrow^*_{G2} \beta$. Vậy $L(G_1) \subseteq L(G_2)$. *Ta chứng minh quy nạp theo số bước dẫn xuất rằng nếu* $A \Rightarrow^*_{G2} w$, *với* $A \in V$ *và* $w \in T^*$ *thì* $A \Rightarrow^*_{G1} w$.

Kết quả hiển nhiên với 1 bước dẫn xuất.

Giả sử kết quả đúng tới k bước dẫn xuất.

Xét $A \Rightarrow^*_{G2} w$ *bằng k +1 bước dẫn xuất.*

Bước đầu tiên có dạng $A \rightarrow B_1B_2 \ldots B_m$ $(m > 1)$. *Ta có thể viết* $w = w_1w_2 \ldots w_m$ *trong đó* $B_i \Rightarrow^*_{G2} w_i$, $1 \le i \le m$. *Nếu* B_i *là ký hiệu kết thúc* a_i *nào đó thì* w_i *là* a_i. *Theo cách xây dựng P' ta có luật sinh* $A \rightarrow X_1X_2 \ldots X_m$ *của P trong đó* $X_i = B_i$ *nếu* $B_i \in V$ *và* $X_i = a_i$ *nếu* $B_i \in V' - V$. *Với* $B_i \in V$, *ta đã biết rằng có dẫn xuất* $B_i \Rightarrow^*_{G1} w_i$ *bằng ít hơn k bước, do vậy theo giả thiết quy nạp* $X_i \Rightarrow^*_{G1} w_i$. *Vậy* $A \Rightarrow^*_{G1} w$.

Ta đã có kết quả là một CFL bất kỳ được sinh ra từ một CFG mà mỗi luật sinh có dạng $A \rightarrow a$ *hoặc* $A \rightarrow B_1B_2 \ldots B_m$ $(m \ge 2)$ *với A,* $B_1, \ldots, B_m$ *là các biến và a là ký hiệu kết thúc. Ta sửa* G_2 *bằng cách thêm vào P' một số luật sinh.*

Bước 3 : *Thay thế các luật sinh có độ dài vế phải > 2 ký hiệu chưa kết thúc.*

Xét luật sinh trong P'có dạng $A \rightarrow B_1B_2 \ldots B_m$ (m > 2) . Ta thay bằng tập hợp các luật sinh :

$$A \rightarrow B_1D_1$$
$$D_1 \rightarrow B_2D_2$$
$$\ldots$$
$$D_{m-3} \rightarrow B_{m-2}D_{m-2}$$
$$D_{m-2} \rightarrow B_{m-1}B_m$$

Đặt V'' là tập các biến mới, P'' là tập các luật sinh mới và văn phạm mới G_3 (V'', T, P'', S). Ta có G_3 chứa các luật sinh thoả mãn định lý.

Hơn nữa, nếu $A \Rightarrow^*_{G2} \beta$ thì $A \Rightarrow^*_{G3} \beta$, vậy $L(G_2) \subseteq L(G_3)$. Ngược lại cũng đúng tức là, $L(G_3) \subseteq L(G_2)$. Chúng ta cũng đã có $L(G_2) \subseteq L(G_1)$ và $L(G_1) \subseteq L(G_2)$. Vậy G_3 là văn phạm thoả mãn dạng chuẩn CNF.

Thí dụ 5.12 : Tìm văn phạm có dạng CNF tương đương văn phạm sau :

$$S \rightarrow A \mid ABA$$
$$A \rightarrow aA \mid a \mid B$$
$$B \rightarrow bB \mid b$$

Bước 1 : Thay thế các luật sinh có độ dài vế phải = 1 (luật sinh đơn vị)

Gọi tập $\Delta_A = \{B \mid A \Rightarrow^* B\}$, xét các biến trong văn phạm, ta có :

$$\Delta_S = \{ S, A, B \}$$

$\Delta_A = \{ A, B\}$
$\Delta_B = \{ B \}$

Vậy tập luật sinh mới, theo định lý sẽ chứa các luật sinh không là luật sinh đơn vị trong P, bổ sung thêm các luật sinh mới thay cho luật sinh đơn vị như sau :

$S \rightarrow aA \mid a \mid bB \mid b \mid ABA$
$A \rightarrow aA \mid a \mid bB \mid b$
$B \rightarrow bB \mid b$

Bước 2 : Thay thế các luật sinh có độ dài vế phải > 1 và có chứa ký hiệu kết thúc.

Ta thấy, a và b đều xuất hiện ở vế phải một số luật sinh, do đó ta tạo thêm 2 biến mới C_a, C_b và 2 luật sinh mới $C_a \rightarrow a$ và $C_b \rightarrow b$.

Văn phạm tương đương có tập luật sinh như sau :

$S \rightarrow C_aA \mid a \mid C_bB \mid b \mid ABA$
$A \rightarrow C_aA \mid a \mid C_bB \mid b$
$B \rightarrow C_bB \mid b$
$C_a \rightarrow a$
$C_b \rightarrow b$

Bước 3 : Thay thế các luật sinh có độ dài vế phải > 2

Chỉ còn duy nhất một luật sinh cần xét ở bước này : $S \rightarrow ABA$ và tập luật sinh mới được thay thế có dạng như sau :

$\mathbf{S \rightarrow C_aA \mid a \mid C_bB \mid b \mid AD_1}$
$A \rightarrow C_aA \mid a \mid C_bB \mid b$
$B \rightarrow C_bB \mid b$
$C_a \rightarrow a$
$C_b \rightarrow b$
$\mathbf{D_1 \rightarrow BA}$

Cuối cùng, ta sẽ thu được văn phạm có dạng chuẩn Chomsky như trên tương đương với văn phạm đã cho.

3.2. Dạng chuẩn Greibach GNF (Greibach Normal Form)

Ta gọi luật sinh với biến A ở bên trái là A - luật sinh.

BỔ ĐỀ 3 : *(Dùng thay thế các luật sinh trực tiếp)*

Cho G (V, T, P, S) là một CFG, đặt $A \rightarrow \alpha_1 B \alpha_2$ là luật sinh trong P và $B \rightarrow \beta_1 \mid \beta_2 \mid ... \mid \beta_r$ là các B - luật sinh; văn phạm G_1 (V, T, P_1, S) thu được từ G bằng cách loại bỏ luật sinh $A \rightarrow \alpha_1 B \alpha_2$ và thêm vào luật sinh $A \rightarrow \alpha_1\beta_1\alpha_2 \mid \alpha_1\beta_2\alpha_2 \mid ... \mid \alpha_1\beta_r\alpha_2$ thì $L(G) = L(G_1)$

Chứng minh

. Hiển nhiên $L(G_1) \subseteq L(G)$ vì nếu $A \rightarrow \alpha_1\beta_i\alpha_2$ được dùng trong dẫn xuất của G_1 thì ta dùng $A \Rightarrow_G \alpha_1 B \alpha_2 \Rightarrow_G \alpha_1\beta_i\alpha_2$

. Để chỉ ra $L(G) \subseteq L(G_1)$ ta cần chú ý rằng $A \rightarrow \alpha_1 B \alpha_2$ là luật sinh trong $P - P_1$ (có trong G và không có trong G_1). Bất cứ khi nào luật sinh $A \rightarrow \alpha_1 B \alpha_2$ được dùng trong dẫn xuất của G thì phải viết lại tại bước sau đó dùng luật sinh dạng $B \rightarrow \beta_i$. Hai bước dẫn xuất này có thể được thay thế bằng một bước dẫn xuất duy nhất, hay :

$$\begin{cases} A \rightarrow \alpha_1 B \alpha_2 \\ B \rightarrow \beta_i \end{cases} \quad \Leftrightarrow \quad A \Rightarrow_{G1} \alpha_1 \beta_i \alpha_2$$

Vậy $L(G) = L(G_1)$

BỔ ĐỀ 4 : *(Dùng loại bỏ luật sinh dạng đệ quy trái trong văn phạm)*

Đặt G (V, T, P, S) là CFG; $A \rightarrow A\alpha_1 \mid A\alpha_2 \mid ... \mid A\alpha_r$ là tập các A - luật sinh có A là ký hiệu trái nhất của vế phải (luật sinh đệ quy trái). Đặt $A \rightarrow \beta_1 \mid \beta_2 \mid ... \mid \beta_s$ là các A - luật sinh còn lại; G_1 ($V \cup \{B\}$, T, P_1, S) là CFG được tạo thành bằng cách thêm biến mới B vào V và thay các A - luật sinh bằng các luật sinh dạng:

1) $A \rightarrow \beta_i$
$A \rightarrow \beta_i B$ } **với $1 \le i \le s$**

2) $B \rightarrow \alpha_i$
$B \rightarrow \alpha_i B$ } **với $1 \le i \le r$**

thì $L(G) = L(G_1)$.

Chứng minh

Trong một chuỗi dẫn xuất trái, một chuỗi luật sinh dạng $A \rightarrow A\alpha_i$ phải kết thúc bằng $A \rightarrow \beta_j$. Tức là:

$A \Rightarrow A\alpha_{i1} \Rightarrow A\alpha_{i2}\alpha_{i1} \Rightarrow ... \Rightarrow A\alpha_{ip}\alpha_{ip-1}\ldots\alpha_{i1} \Rightarrow \beta_j\alpha_{ip}\alpha_{ip-1}\ldots\alpha_{i1}$

Chuỗi dẫn xuất trong G có thể thay bằng chuỗi dẫn xuất trong G_1 bởi :

$A \Rightarrow \beta_j B \Rightarrow \beta_j \alpha_{ip} B \Rightarrow \beta_j\alpha_{ip}\alpha_{ip-1}\ldots B \Rightarrow ... \Rightarrow \beta_j\alpha_{ip}\alpha_{ip-1}\ldots\alpha_{i2}B$
$\Rightarrow \beta_j\alpha_{ip}\alpha_{ip-1}\ldots\alpha_{i1}.$

Sự chuyển đổi ngược lại cũng có thể được.

Vậy $L(G) = L(G_1)$.

ĐỊNH LÝ 5.6 : *(Dạng chuẩn Greibach, hay GNF)*

Mỗi CFL bất kỳ không chứa ε được sinh ra bởi một CFG mà mỗi luật sinh có dạng $A \rightarrow a\alpha$ với A là biến, a là một ký hiệu kết thúc, và α là một chuỗi các biến (có thể rỗng).

Chứng minh

Bước 1: Đặt G là CFG sinh ra CFL không chứa ε. Xây dựng văn phạm tương đương G' có dạng chuẩn Chomsky.

Bước 2: Đổi tên các biến trong tập của G' thành $A_1, A_2, ..., A_m$ ($m \ge 1$) với A_1 là ký hiệu bắt đầu. Đặt $V = \{A_1, A_2, ..., A_m\}$.

Bước 3: *Thay thế các luật sinh sao cho nếu* $A_i \to A_j\gamma$ *là một luật sinh thì* $j > i$

Bắt đầu từ A_1 và tiến tới A_m, ta thay thế các A_k - luật sinh :

Nếu $A_k \to A_j\gamma$ là luật sinh với $j < k$: sinh ra một tập luật sinh mới bằng cách thay thế A_j bên vế phải của mỗi A_j - luật sinh theo bổ đề 3. Lặp lại không quá k - 1 lần ta thu được tập luật sinh dạng $A_k \to A_l\gamma$ với $l \geq k$.

Sau đó, các luật sinh với $l = k$ được thay thế theo bổ đề 4 bằng cách đưa vào các biến mới B_k.

Giải thuật cụ thể như sau:

```
Begin
(1)  For k := 1 to m do begin
(2)    for j := 1 to k-1 do
(3)      for Mỗi luật sinh dạng Ak → Ajα do
          begin
(4)         for Tất cả luật sinh Aj → β do
(5)            Thêm luật sinh Ak → βα;
(6)         Loại bỏ luật sinh Ak → Ajα
         end;
(7)    for Mỗi luật sinh dạng Ak → Akα do
         begin
(8)       Thêm các luật sinh Bk → α và Bk → αBk;
(9)       Loại bỏ luật sinh Ak → Akα
         end;
(10)   for Mỗi luật sinh Ak → β trong đó β không bắt đầu bằng Ak do
(11)      Thêm luật sinh Ak → βBk
    end;
end;
```

Bằng cách lặp lại bước xử lý trên cho mỗi biến nguồn, trong P chỉ chứa các luật sinh có dạng như sau :

1) $A_i \to A_j\gamma$ với $j > i$
2) $A_i \to a\gamma$ với $a \in T$
3) $B_k \to \gamma$ $\gamma \in (V \cup \{B_1, B_2, ..., B_{i-1}\})^*$

Bước 4: *Thay thế các* A_i *- luật sinh về đúng dạng.*

Gọi V' là tập biến mới phát sinh sau bước 3.

Chú ý rằng ký hiệu trái nhất của vế phải trong một luật sinh bất kỳ đối với biến A_m phải là một ký hiệu kết thúc, vì A_m là biến có chỉ số cao nhất. Ký hiệu trái nhất của vế phải của một A_{m-1}- luật sinh bất kỳ phải là A_m hoặc một ký hiệu kết thúc. Nếu là A_m, ta tạo ra tập luật sinh mới bằng cách thay thế A_m bởi chuỗi vế phải của các A_m-luật sinh theo bổ đề 3. Tiếp tục quá trình này cho các luật sinh từ A_{m-2}, ..., A_2, A_1 cho tới khi vế phải của tất cả các A_i - luật sinh có dạng bắt đầu bằng một ký hiệu kết thúc.

Bước 5: *Thay thế các* B_k *-luật sinh về đúng dạng.*

Bước cuối cùng, ta khảo sát các luật sinh với tập các biến mới B_1, B_2, ..., B_m.

Vì ta bắt đầu từ văn phạm đã có dạng chuẩn Chomsky, nên dễ dàng chứng minh quy nạp theo số lần áp dụng bổ đề 3 và bổ đề 4 rằng vế phải của mỗi A_i -luật sinh, với $1 \le i \le n$, bắt đầu bằng ký hiệu kết thúc hoặc A_jA_k với j, k nào đó. Vậy α (trong bước (7)) không khi nào có thể rỗng hoặc bắt đầu bằng một B_j khác, hay tất cả B_i - luật sinh đều có vế phải bắt đầu bằng ký hiệu kết thúc hoặc A_i. Một lần nữa, lại áp dụng bổ đề 3 cho mỗi B_i - luật sinh.

Ta thu được tập luật sinh trong văn phạm sau cùng thỏa đúng dạng chuẩn Greibach.

Thí dụ 5.13 : Tìm văn phạm có dạng GNF tương đương văn phạm G sau :

$$A_1 \rightarrow A_2A_1 \mid A_2A_3$$
$$A_2 \rightarrow A_3A_1 \mid a$$
$$A_3 \rightarrow A_2A_2 \mid b$$

Bước 1 : G thỏa dạng chuẩn CNF sinh ra CFL không chứa ε

Bước 2 : Ta có $V = \{A_1, A_2, ..., A_3\}$

Bước 3 : Thay thế các luật sinh sao cho nếu $A_i \rightarrow A_j \gamma$ là một luật sinh thì $j > i$.

Ta thấy trong tập luật sinh, các luật sinh cho A_1 và A_2 đã thỏa điều kiện $j > i$. Chỉ có luật sinh $A_3 \rightarrow A_2A_2$ cần sửa đổi. Áp dụng bổ đề 3 để thay thế luật sinh này, ta có:

$$A_3 \rightarrow A_3A_1A_2 \mid aA_2$$

Sau đó, dùng bổ đề 4 để loại bỏ đệ quy trái, ta được tập luật sinh mới có dạng như sau :

$$A_1 \rightarrow A_2A_1 \mid A_2A_3$$
$$A_2 \rightarrow A_3A_1 \mid a$$
$$A_3 \rightarrow aA_2 \mid b \mid aA_2B \mid bB$$
$$B \rightarrow A_1A_2 \mid A_1A_2\,B$$

Bước 4 : Thay thế các A_i -luật sinh về đúng dạng.

Ở bước này, ta có thể thấy tất cả các A_3 - luật sinh đã có dạng chuẩn. Tiếp tục, áp dụng bổ đề 3 để thay thế các A_3 - luật sinh vào A_2, A_1, thu được tập luật sinh mới như sau:

$$A_1 \rightarrow aA_2A_1A_1 \mid bA_1A_1 \mid aA_2BA_1A_1 \mid bBA_1A_1 \mid aA_1 \mid aA_2A_1A_3 \mid bA_1A_3 \mid aA_2BA_1A_3 \mid bBA_1A_3 \mid aA_3$$
$$A_2 \rightarrow aA_2A_1 \mid bA_1 \mid aA_2BA_1 \mid bBA_1 \mid a$$
$$A_3 \rightarrow aA_2 \mid b \mid aA_2B \mid bB$$
$$B \rightarrow A_1A_2 \mid A_1A_2\,B$$

Bước 5 : Thay thế các B_k - luật sinh về đúng dạng.

$$B \rightarrow aA_2A_1A_1A_2 \mid bA_1A_1A_2 \mid aA_2BA_1A_1A_2 \mid bBA_1A_1A_2 \mid aA_1A_2 \mid aA_2A_1A_3A_2 \mid bA_1A_3A_2 \mid aA_2BA_1A_3A_2 \mid bBA_1A_3A_2 \mid aA_3A_2 \mid aA_2A_1A_1A_2B \mid bA_1A_1A_2B \mid aA_2BA_1A_1A_2B \mid bBA_1A_1A_2B \mid aA_1A_2\,B \mid aA_2A_1A_3A_2B \mid bA_1A_3A_2B \mid aA_2BA_1A_3A_2B \mid bBA_1A_3A_2B \mid aA_3A_2B$$

Cuối cùng, ta thu được văn phạm có dạng GNF với 39 luật sinh.

IV. TÍNH CHẤT CỦA NGÔN NGỮ PHI NGỮ CẢNH

Cũng như lớp ngôn ngữ chính quy, có một vài tính chất giúp xác định một ngôn ngữ có thuộc lớp ngôn ngữ phi ngữ cảnh hay không ?

4.1. Bổ đề bơm đối với CFL

(Dùng chứng minh một ngôn ngữ không là ngôn ngữ phi ngữ cảnh)

Cho L là một CFL bất kỳ, tồn tại một số n chỉ phụ thuộc vào L sao cho nếu $z \in L$ và $|z| \geq n$ thì ta có thể viết $z = uvwxy$ sao cho:

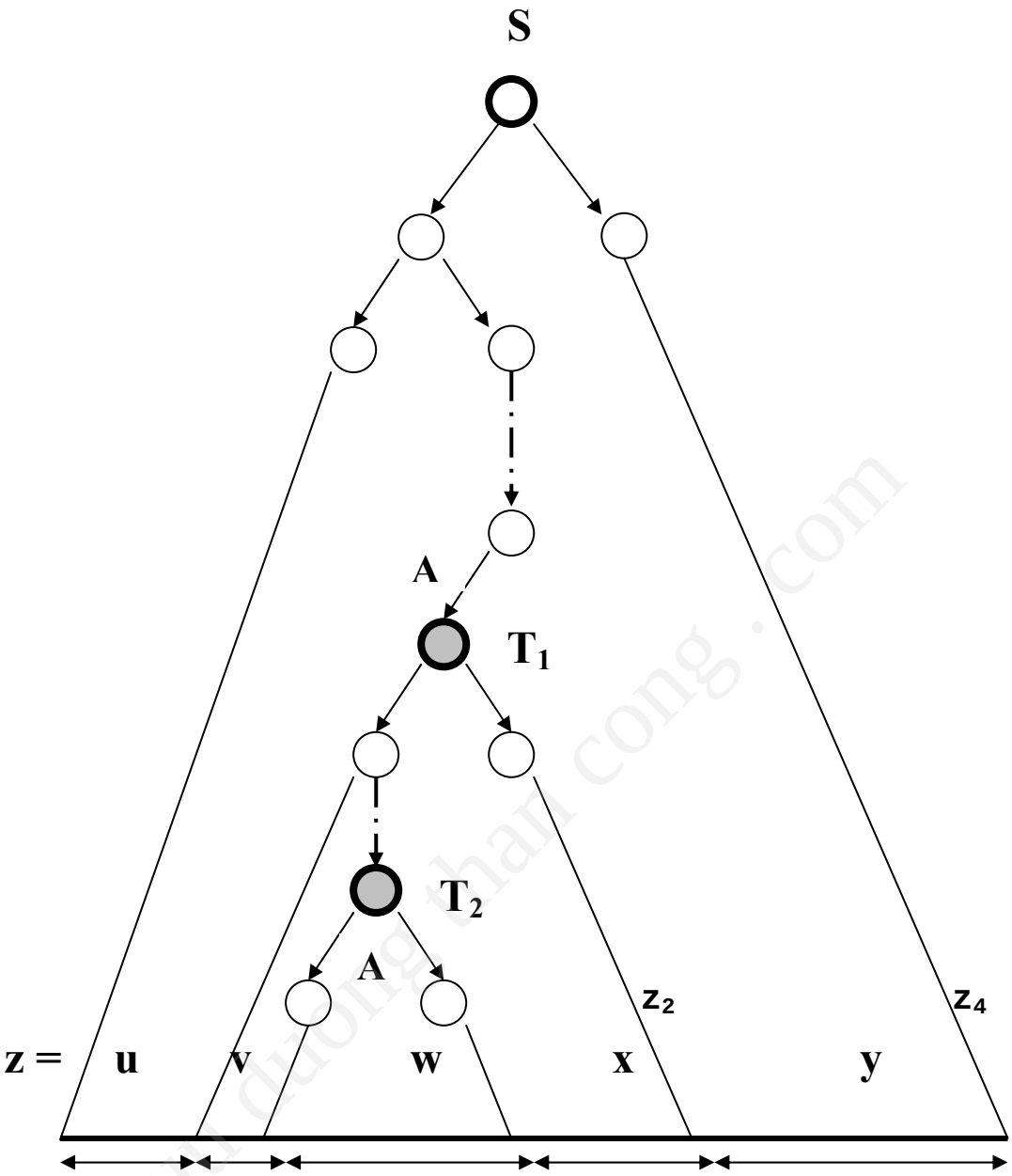
1) $|vx| \geq 1$

2) $|vwx| \leq n$ và

3) $\forall i \geq 0 : uv^iwx^iy \in L$

Chứng minh

Đặt G là văn phạm có dạng chuẩn CHOMSKY sinh L - {ε}. Chú ý rằng nếu $z \in L(G)$ và cây dẫn xuất không có đường đi dài hơn i thì chuỗi sinh ra từ văn phạm có độ dài không dài hơn 2^{i-1}.



Hình 5.5 - *Các bước dẫn xuất trong chứng minh Bổ đề bơm*

Giả sử G có k biến, ta đặt $n = 2^k$. Nếu $z \in L(G)$ và $|z| \geq n$ thì $|z| > 2^{k-1}$, vậy có một đường đi nào đó trên cây dẫn xuất có độ dài lớn hơn hoặc bằng k+1. Như vậy đường đi đó sẽ có ít nhất k+2 đỉnh, hay có ít nhất k+1 biến trên đường đi (chỉ có nút lá mới có thể không là biến), suy ra phải có biến xuất hiện hai lần, hơn nữa ta phải có:

1) Có hai đỉnh v_1 và v_2 có cùng nhãn là A

2) Đỉnh v_1 gần gốc hơn v_2

3) Phần đường đi từ v_1 tới lá có độ dài nhiều nhất là k+1 (đi từ lá lên tới gốc theo đường đi, chỉ có lá mới có thể là ký hiệu kết thúc vì vậy trong k+2 đỉnh đầu tiên phải có ít nhất k+1 biến và phải có ít nhất hai biến trùng nhau)

Xét cây con T_1 có đỉnh là v_1 biểu diễn dẫn xuất của chuỗi con có độ dài không quá 2^k (vì trong cây con T_1 không có đường đi nào có độ dài vượt quá k+1). Đặt z_1 là

chuỗi sinh ra từ cây T_1. Ta gọi T_2 là một cây con có nút gốc là v_2, rõ ràng T_2 là cây con của T_1. Giả sử T_2 sinh ra chuỗi z_2 thì ta có thể viết $z_1 = z_3z_2z_4$. Hơn nữa z_3 và z_4 không thể đồng thời bằng ε vì luật sinh đầu tiên trong cây dẫn xuất của T_1 là $A \rightarrow BC$ với biến B, C nào đó. Cây con T_2 phải thuộc vào cây con sinh bởi bút biến B hoặc cây con sinh bởi nút biến C. Ta có :

$A \Rightarrow^*_G z_3Az_4$ và $A \Rightarrow^*_G z_2$ trong đó $| z_3z_2z_4 | \leq 2^k = n$.

Vậy $A \Rightarrow^*_G z_3^i z_2 z_4^i$, $\forall i \geq 0$.

Hiển nhiên chuỗi $z = uz_3z_2z_4y$, với các chuỗi u, y nào đó.

Nếu đặt $z_3 = v$, $z_2 = w$ và $z_4 = x$, thì ta sẽ hoàn thành việc chứng minh.

Ứng dụng bổ đề bơm

Thí dụ 5.14 : Chứng minh $L = \{a^ib^ic^i \mid i \geq 1\}$ không phải là ngôn ngữ phi ngữ cảnh.

Chứng minh

Giả sử L là ngôn ngữ phi ngữ cảnh, khi đó có tồn tại số n (theo bổ đề bơm).

Xét chuỗi $z = a^nb^nc^n$ với $| z | \geq n$, ta có thể viết $z = uvwxy$ thoả mãn bổ đề.

Ta thấy vx nằm trong $a^nb^nc^n$ và $| vwx | \leq n$, vậy vx không thể chứa cả ký hiệu a và ký hiệu c (do sau ký hiệu a bên phải nhất n+1 vị trí mới đến vị trí của c bên trái nhất). Nếu vx chỉ có chứa ký hiệu a, thì chuỗi uwy (trường hợp uv^iwx^iy với $i = 0$) sẽ có chứa số ký hiệu b và c ít hơn số ký hiệu a vì $| vx | \geq 1$. Vậy uwy không có dạng $a^jb^jc^j$. Tương tự cho các trường hợp chuỗi vx chỉ chứa ký hiệu b hay c. Còn nếu trong vx có chứa ký hiệu a và b thì chuỗi uvy sẽ có chứa số ký hiệu c lớn hơn a và b, nên nó cũng không thể thuộc L. Cũng tương tự cho trường hợp vx chứa hai ký hiệu b và c. Cuối cùng, ta suy ra chuỗi $uv^iwx^iy \notin L$, vì các ký hiệu a, b, c trong chúng không thể bằng nhau với mọi i. Mà theo giả thiết của bổ đề bơm, chuỗi này phải thuộc L, mâu thuẫn.

Vậy L không thể là CFL.

Thí dụ 5.15 : Chứng minh $L = \{a^ib^jc^id^j \mid i, j \geq 1\}$ không phải là ngôn ngữ phi ngữ cảnh.

Chứng minh

Giả sử L là ngôn ngữ phi ngữ cảnh, khi đó có tồn tại số n (theo bổ đề bơm).

Xét chuỗi $z = a^nb^nc^nd^n$ với $| z | \geq n$, ta có thể viết $z = uvwxy$ thoả mãn bổ đề.

Ta thấy vì vx nằm trong $a^nb^nc^nd^n$ và $| vwx | \leq n$, nên vx không thể chứa ít nhất hai ký hiệu khác nhau. Hơn nữa, nếu vx có chứa hai ký hiệu khác nhau, thì chúng phải là hai ký hiệu liên tiếp đứng cạnh nhau, chẳng hạn a và b. Nếu vx chỉ có chứa ký hiệu a, thì chuỗi uwy sẽ có số ký hiệu a ít hơn số ký hiệu c nên không thuộc L, mâu thuẫn. Tương tự với trường hợp chuỗi vx chỉ chứa ký hiệu b, c hoặc d. Bây giờ giả sử chuỗi vx có chứa a và b thì vwy vẫn có số ký hiệu a ít hơn c. Mâu thuẫn tương tự cũng xuất hiện khi chuỗi vx có chứa b và c hoặc c và d. Vì chỉ có thể có một trong các trường hợp này nên ta có thể kết luận rằng L không thể là CFL.

<u>Câu hỏi</u> :

Hãy so sánh các yếu tố ràng buộc trong phát biểu Bổ đề bơm cho ngôn ngữ phi ngữ cảnh và Bổ đề bơm cho ngôn ngữ chính quy ?

4.2. Tính chất đóng của CFL

ĐỊNH LÝ 5.7 : CFL đóng với phép hợp, phép nối kết và phép bao đóng Kleene.

Chứng minh

Đặt L_1 và L_2 là hai CFL sinh bởi các CFG G_1 (V_1, T_1, P_1, S_1) và G_2 (V_2, T_2, P_2, S_2). Vì các biến có thể đổi tên mà không ảnh hưởng tới ngôn ngữ sinh ra nên ta có thể xem tập V_1 và V_2 là rời nhau. Ta cũng giả sử các biến mới S_3, S_4, $S_5 \notin V_1$ hoặc V_2

. Đối với $L_1 \cup L_2$: Xây dựng văn phạm G_3 ($V_1 \cup V_2 \cup \{S3\}$, $T_1 \cup T_2$, P_3, S_3),
trong đó $P_3 = P_1 \cup P_2 \cup \{S_3 \rightarrow S_1 \mid S_2\}$.

Nếu $w \in L_1$ thì dẫn xuất $S_3 \Rightarrow_{G3} S_1 \Rightarrow^*_{G1} w$ là một dẫn xuất trong G_3 (vì mỗi luật sinh trong G_1 cũng là luật sinh trong G_3). Tương tự mỗi chuỗi trong L_2 có dẫn xuất trong G_3 bắt đầu bằng $S_3 \Rightarrow S_2$. Vậy $L_1 \cup L_2 \subseteq L(G_3)$.

Ngược lại, nếu $w \in L(G_3)$ thì dẫn xuất $S_3 \Rightarrow^*_{G3} w$ phải bắt đầu bằng $S_3 \rightarrow S_1$ hoặc $S_3 \rightarrow S_2$. Tức là dẫn xuất có dạng $S_3 \Rightarrow_{G3} S_1 \Rightarrow^*_{G3} w$ hoặc $S_3 \Rightarrow_{G3} S_2 \Rightarrow^*_{G3} w$. Trong trường hợp thứ nhất, do V_1 và V_2 rời nhau nên chỉ có các ký hiệu của G_1 xuất hiện trong dẫn xuất $S_1 \Rightarrow^*_{G3} w$. Vì trong các luật sinh của P_3 chỉ có chứa các ký hiệu thuộc G_1 và nằm trong tập luật sinh P_1, nên ta có thể kết luận chỉ có những luật sinh thuộc P_1 được dùng trong dẫn xuất $S_1 \Rightarrow^*_{G3} w$. Vì thế $S_1 \Rightarrow^*_{G1} w$ và $w \in L_1$. Tương tự cho trường hợp dẫn xuất $S_3 \Rightarrow_{G3} S_2$, ta cũng có $w \in L_2$. Vậy $L(G_3) \subseteq L_1 \cup L_2$, và vì thế $L(G_3) = L_1 \cup L_2$.

Vậy ta đã chứng minh xong $L(G_3) = L_1 \cup L_2$, hay $L_1 \cup L_2$ là CFL.

. Đối với L_1L_2 : Xây dựng văn phạm G_4 ($V_1 \cup V_2 \cup \{S_4\}$, $T_1 \cup T_2$, P_4, S_4) ,
trong đó $P_4 = P_1 \cup P_2 \cup \{S_4 \rightarrow S_1S_2\}$.

Chứng minh tương tự như trên ta có $L(G_4) = L_1L_2$, vậy L_1L_2 cũng là CFL.

. Đối với L_1^* : Xây dựng văn phạm G_5 ($V_1 \cup \{S_5\}$, T_1, P_5, S_5),
trong đó $P_5 = P_1 \cup \{ S_5 \rightarrow S_1S_5 \mid \varepsilon\}$.

Ta cũng dễ dàng chứng minh được $L(G_5) = (L(G_1))^*$.

ĐỊNH LÝ 5.8 : CFL không đóng với phép giao

Chứng minh

Ta đã biết ngôn ngữ $L_1 = \{a^ib^ic^i \mid i \geq 1\}$ không là CFL. Ta có thể chứng minh :

. $L_2 = \{a^i b^i c^j \mid i \geq 1 \text{ và } j \geq 1\}$ là CFL vì L_2 được sinh bởi văn phạm :

$S \rightarrow AB$
$A \rightarrow aAb \mid ab$
$B \rightarrow cB \mid c$

. $L_3 = \{a^i b^j c^j \mid i \geq 1 \text{ và } j \geq 1\}$ cũng là CFL vì L_3 được sinh từ văn phạm :

$S \rightarrow CD$
$C \rightarrow aC \mid a$
$D \rightarrow bDc \mid bc$

Tuy nhiên $L_2 \cap L_3 = L_1$ không phải là CFL.
Vậy CFL không đóng với phép giao.

Hệ quả: CFL không đóng với phép lấy phần bù.

Chứng minh

Giả sử CFL đóng với phép lấy phần bù, vậy với L_1, L_2 là hai CFL bất kỳ, theo quy luật DeMorgan ta có $L_1 \cap L_2 = \overline{\overline{L_1} \cup \overline{L_2}}$ nên $L_1 \cap L_2$ là CFL hay CFL cũng đóng với phép giao. (Điều này mâu thuẫn với định lý 6.6)

Câu hỏi :

Hãy so sánh các tính chất đóng của lớp ngôn ngữ phi ngữ cảnh với lớp ngôn ngữ chính quy ?

V. CÁC GIẢI THUẬT QUYẾT ĐỊNH CFL

Có một vài câu hỏi về CFL mà chúng ta cần phải trả lời. Chẳng hạn, liệu một ngôn ngữ phi ngữ cảnh cho trước là rỗng, hữu hạn hay vô hạn hay một chuỗi nào đó liệu có thuộc ngôn ngữ này không ? Tuy nhiên, cũng có những câu hỏi về CFL mà không có giải thuật nào để có thể trả lời. Chẳng hạn, liệu hai CFG thì có tương đương nhau, hay phần bù của một CFL có là CFL hay không, hoặc một CFG cho trước nào đó có phải là văn phạm mơ hồ ? Trong phần này, chúng ta chỉ đưa ra giải thuật cho một số các câu hỏi có thể trả lời.

5.1. Giải thuật xác định ngôn ngữ phi ngữ cảnh

ĐỊNH LÝ 5.9 : Tồn tại giải thuật để xác định CFL là: rỗng, hữu hạn, vô hạn.

Chứng minh

Với văn phạm G (V, T, P, S) :

. Để kiểm tra L(G) có rỗng hay không, ta dùng **bổ đề 5. 1**: Rõ ràng L(G) không rỗng khi và chỉ khi S sinh ra một chuỗi ký hiệu kết thúc nào đó.

. Để kiểm tra L(G) hữu hạn hay vô hạn, ta dùng **định lý 5. 5** để tìm văn phạm tương đương G' (V', T, P', S) có dạng chuẩn CHOMSKY và không có ký hiệu vô ích sinh ra L(G) - {ε}. L(G) hữu hạn khi và chỉ khi L(G') hữu hạn.

Để kiểm tra tính hữu hạn của CFG có dạng chuẩn CHOMSKY, ta chỉ cần vẽ đồ thị có hướng với mỗi đỉnh trên đồ thị là một biến thuộc văn phạm và cạnh từ A đến B nếu và chỉ nếu có luật sinh $A \to BC$ hoặc $A \to CB$ với biến C bất kỳ. Khi đó, ngôn ngữ sinh ra là hữu hạn nếu và chỉ nếu đồ thị không có chu trình. Vì :

Nếu đồ thị có chu trình, giả sử chu trình là $A_0, A_1, \ldots, A_n, A_0$ thì sẽ có chuỗi dẫn xuất: $A_0 \Rightarrow \alpha_1 A_1 \beta_1 \Rightarrow \alpha_2 A_2 \beta_2 \ldots \Rightarrow \alpha_n A_n \beta_n \Rightarrow \alpha_{n+1} A_0 \beta_{n+1}$, trong đó α_i, β_i là chuỗi các biến và $|\alpha_i \beta_i| = i$. Vì không có ký hiệu vô ích nên $\alpha_{n+1} \Rightarrow^* w$ và $\beta_{n+1} \Rightarrow^* x$ với mọi chuỗi w, x là các chuỗi ký hiệu kết thúc và độ dài tổng cộng ít nhất bằng n+1. Vì $n \geq 0$, nên w và x không thể đồng thời bằng ε.

Kế tiếp, cũng do văn phạm không có chứa ký hiệu vô ích nên ta có thể tìm được các chuỗi y, z sao cho $S \Rightarrow^* yA_0z$ và chuỗi ký hiệu kết thúc v sao cho $A_0 \Rightarrow^* v$. Vậy ∀i ta có :

$$S \Rightarrow^* yA_0z \Rightarrow^* ywA_0xz \Rightarrow^* yw^2A_0x^2z \Rightarrow^* \ldots \Rightarrow^* yw^iA_0x^iz \Rightarrow^* yw^ivw^iz.$$

Vì $|wx| > 0$, nên chuỗi yw^ivw^iz không thể bằng yw^jvw^jz nếu $i \neq j$. Vậy văn phạm sinh ngôn ngữ vô hạn.

Ngược lại, giả sử đồ thị không có chu trình. Ta gọi hạng của biến A là độ dài lớn nhất của đường đi bắt đầu từ A. Vì không có chu trình nên A sẽ có hạng hữu hạn. Nếu $A \to BC$ là một luật sinh thì hạng của B và C phải nhỏ hơn hạng của A. Ta chứng minh quy nạp theo r (hạng của A) rằng không có chuỗi ký hiệu kết thúc nào có độ dài lớn hơn 2^r

Với r = 0: hạng của A bằng 0, vậy không có cạnh từ A. Do đó, tất cả các A - luật sinh đều có dạng $A \to a$, hay A dẫn ra chuỗi có độ dài $l = 2^0$.

Xét r > 0: nếu ta dùng luật sinh $A \to a$ thì dẫn ra chuỗi chỉ có độ dài 1, nếu dùng luật sinh $A \to BC$ thì vì B, C có hạng í hơn hoặc bằng r -1 nên theo giả thiết quy nạp B, C dẫn ra chuỗi có độ dài ngắn hơn 2^{r-1} . Vậy BC không thể dẫn ra chuỗi có độ dài lớn hơn 2^r. Giả sử S có hạng là r_0 thì các chuỗi do S sinh ra có độ dài không quá 2^{r0} . Vì thế suy ra ngôn ngữ là hữu hạn.

Thí dụ 5.16 : Xét văn phạm G chứa các luật sinh sau :

$$S \to AB$$
$$A \to BC \mid a$$
$$B \to CC \mid b$$
$$C \to a$$

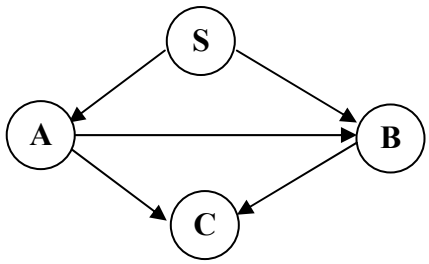
Ta thấy văn phạm G có các luật sinh đã thỏa dạng chuẩn Chomsky.

. Để kiểm tra tính rỗng của văn phạm, ta áp dụng Bổ đề 5.1 lên tập biến V để tìm tập biến mới mới V_1 chỉ chứa các biến có khả năng dẫn ra chuỗi ký hiệu kết thúc trong văn phạm :

Ta có : $V_1 = \{ A, B, C, S \}$ vì $A \to a$, $B \to b$, $C \to a$ và $S \to AB$

Hay S $\in$ V_1 có nghĩa là S có thể sinh ra các chuỗi ký hiệu kết thúc. Vậy ngôn ngữ sinh bởi văn phạm G : L(G) không rỗng.

. Để kiểm tra tính hữu hạn của văn phạm, ta vẽ đồ thị có hướng tương ứng với các luật sinh trong văn phạm như sau :



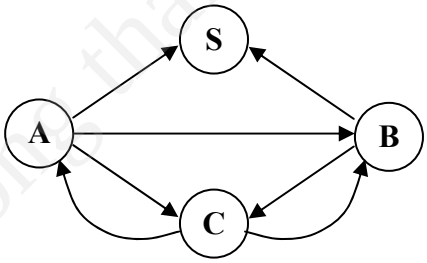
Hình 5.6 - *Đồ thị có hướng tương ứng*

Rõ ràng, ta thấy đồ thị không có chu trình. Hạng của S, A, B, C lần lượt là 3, 2, 1 và 0. Chẳng hạn, một đường đi dài nhất từ S là S $\rightarrow$ A $\rightarrow$ B $\rightarrow$ C. Vậy văn phạm này là hữu hạn, nó sinh ra hữu hạn chuỗi và độ dài các chuỗi không lớn hơn $2^3 = 8$.

Thực tế, chuỗi dài nhất dẫn xuất được từ S là :

S $\Rightarrow$ AB $\Rightarrow$ BCB $\Rightarrow$ CCCB $\Rightarrow$ CCCCC $\Rightarrow^*$ aaaaa ,với độ dài chuỗi là 5.

Nếu ta thêm vào văn phạm một luật sinh mới : C $\rightarrow$ AB, thì đồ thị có hướng tương ứng lúc đó có dạng như sau :



Hình 5.7 - *Đồ thị có hướng tương ứng văn phạm bổ sung*

Đồ thị mới này có nhiều chu trình, chẳng hạn A $\rightarrow$ B $\rightarrow$ C $\rightarrow$ A. Vậy ta phải tìm được một dẫn xuất dạng A $\Rightarrow^*$ $\alpha_3 A \beta_3$, cụ thể là A $\Rightarrow$ BC $\Rightarrow$ CCC $\Rightarrow$ CABC, trong đó α_3 = C và β_3 = BC. Vì C $\Rightarrow^*$ a và BC $\Rightarrow^*$ ba nên A $\Rightarrow^*$ aAba.

Mặt khác, S $\Rightarrow^*$ Ab và A $\Rightarrow^*$ a, suy ra : S $\Rightarrow^*$ $a^i a(ba)^i b$, $\forall i$. Vậy ngôn ngữ sinh từ văn phạm mới là vô hạn.

5.2. Giải thuật thành viên (Membership)

<u>ĐỊNH LÝ 5.10</u> : Tồn tại giải thuật để xác định với một CFL nào đó sinh ra từ CFG G(V, T, P, S) và một chuỗi x bất kỳ thì x có thuộc L(G) hay không ?

Chứng minh

Có một vài giải thuật được đề nghị cho bài toán thành viên này. Sau đây trình bày một giải thuật theo vòng lặp đơn giản, ta gọi là giải thuật CYK (Cocke-Younger-Kasami) với thời gian tỷ lệ với $|x|^3$.

Giả sử văn phạm G (V, T, P, S) đã có dạng chuẩn Chomsky và $|x| = n \geq 1$. Trước hết, ta phải xác định với mỗi i, j và mỗi biến A, phải chăng $A \Rightarrow^* x_{ij}$, trong đó x_{ij} là một chuỗi con của chuỗi x tính từ vị trí thứ i và có độ dài j.

Ta chứng minh quy nạp theo độ dài j :

- Với j = i : ta có $A \Rightarrow^* x_{ij}$ khi và chỉ khi $A \rightarrow x_{ij}$ là một luật sinh.
- Với j > i : ta có $A \Rightarrow^* x_{ij}$ khi và chỉ khi có một luật sinh dạng $A \rightarrow BC$ và số k, $1 \leq k < j$ sao cho B dẫn xuất ra k ký hiệu đầu tiên của x_{ij} và C dẫn xuất ra j – k ký hiệu cuối của x_{ij}. Có nghĩa là :

$$B \Rightarrow^* x_{ik} \text{ và } C \Rightarrow^* x_{i+k,\, j-k}$$

Vì cả k và j – k đều nhỏ hơn j, nên theo giả thiết quy nạp ta đã xác định được liệu hai chuỗi dẫn xuất này có tồn tại hay không ? Thế thì cũng có thể c]xác định được liệu $A \Rightarrow^* x_{ij}$ hay không ?

Với cách thực hiện như thế, ta sẽ xác định được phải chăng $S \Rightarrow^* x_{1n}$, nhưng $x_{1n} = x$, vậy $x \in L(G)$ khi và chỉ khi $S \Rightarrow^* x_{1n}$.

Sau đây trình bày giải thuật CYK theo giải thuật trên, trong đó V_{ij} là tập hợp tất cả các biến A sao cho $A \Rightarrow^* x_{ij}$. Chú ý rằng ta có thể giả thiết $1 \leq i \leq n - j + 1$, bởi vì lúc đó chuỗi con x_{ij} với độ dài j mới thực sự tồn tại.

Bước (1) và (2) xử lý trường hợp j = i. Vì văn phạm G đã cho sẵn, cho nên bước (2) chiếm mộ thời gian cố định. Vậy các bước (1) và (2) chiếm thời gian O(n). các vòng lặp For ở các dòng (3) và (4) làm cho các bước từ (5) đến (7) lặp lại nhiều nhất là n^2 lần (do i, j ≤ n). Bước (5) mỗi lần thực hiện cũng chiếm một khoảng thời gian cố định. Vậy tổng thời gian để thực hiện bước (5) là $O(n^2)$. Vòng lặp For ở dòng (6) làm cho bước (7) lặp lại n lần hoặc ít hơn. Vì bước (7) cũng chiếm một thời gian cố định, nên các bước (6) và (7) gộp lại chiếm thời gian O(n). Vì các bước này được thực hiện $O(n^2)$, nên tổng thời gia thực hiện cho bước (7) là $O(n^3)$. Vậy thời gian thực hiện toàn bộ giải thuật là ở cấp $O(n^3)$.

Giải thuật CYK:

```
    Begin
(1)     For i := 1 to n do
(2)         Vij := { A | A → a là một luật sinh và a là ký hiệu thứ i trong x }
(3)     For j := 2 to n do
(4)         for i := 1 to n – j + 1 do
                begin
(5)                 Vij := 0;
(6)                 for k := 1 to j - 1 do
(7)                     Vij := Vij ∪ { A | A → BC là một luật sinh, B ∈ Vik
                                                  và C ∈ Vi+k, j–k }
                end;
```

End;

Thí dụ 5.17 : Cho văn phạm G có dạng chuẩn Chomsky chứa các luật sinh sau :

$S \rightarrow AB \mid BC$
$A \rightarrow BA \mid a$
$B \rightarrow CC \mid b$
$C \rightarrow AB \mid a$

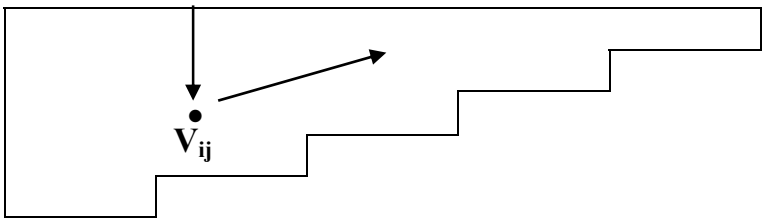
Xét chuỗi nhập x = **baaba.**

Bảng các V_{ij} cho ở hình 5.8 dưới đây. Dòng đầu tiên trong bảng được cho bởi các bước (1) và (2) trong giải thuật. Vì $x_{11} = x_{41} = b$ nên $V_{11} = V_{41} = \{B\}$ vì B là biến duy nhất dẫn xuất ra b, còn $x_{21} = x_{31} = x_{51} = a$, suy ra $V_{21} = V_{31} = V_{51} = \{A, C\}$ vì A và C có các luật sinh với a bên vế phải.

		b	*a*	*a*	*b*	*a*
				i →		
		1	2	3	4	5
j ↓	1	*B*	*A, C*	*A, C*	*B*	*A, C*
	2	*S, A*	*B*	*S, C*	*S, A*	
	3	∅	*B*	*B*		
	4	∅	*S, A, C*			
	5	*S, A, C*				

Hình 5.8 – *Bảng các V_{ij}*

Để tính V_{ij} với j > i, ta phải thực hiện vòng lặp For ở bước (6) và (7). Ta phải đối chiếu V_{ik} với $V_{i+k,\, j-k}$ với k = 1, 2, …, j - 1 để tìm biến D trong V_{ik} và biến E trong $V_{i+k,\, j-k}$ sao cho DE là vế phải của một hay nhiều luật sinh. Các vế trái của những luật sinh đó được đưa vào trong V_{ij}. Quá trình đối chiếu đó diễn ra bằng cách giảm dần giá trị cột i, đồng thời tăng dần lên theo đường chéo qua V_{ij} về phía phải như các chiều mũi tên vẽ trong hình 5.9.



Hình 5.9 – *Quá trình tính các V_{ij}*

Chẳng hạn, để tính V_{24}, đầu tiên ta đối chiếu $V_{21} = \{A, C\}$ với $V_{33} = \{B\}$. Ta có V_{21} $V_{33} = \{AB, CB\}$. Vì có các luật sinh S → AB và C → AB nên S và C được đưa vào

V_{24}. Tiếp đến ta lại xét V_{22} V_{42} = {A}{S, A} = {BS, BA}. Vì có luật sinh A → BA, vậy ta đưa thêm A vào V_{24}. Cuối cùng ta xét V_{23} V_{51} = {B}{A, C} = {BA, BC} gặp lại các vế phải đã xét, vậy không thể thêm gì vào V_{24}. Vậy V_{24} = {S, AC}.
Cuối cùng, vì S ∈ V_{15} , vậy chuỗi baaba là thuộc ngôn ngữ sinh ra bởi văn phạm đã cho.

Tổng kết chương V: Việc mô tả ngôn ngữ phi ngữ cảnh bằng phương tiện văn phạm phi ngữ cảnh tỏ ra rất hữu hiệu, cũng tương tự như việc sử dụng văn phạm BNF trong định nghĩa các ngôn ngữ lập trình. Trong chương này, chúng ta đã khảo sát tương đối cặn kẽ các phương tiện mô tả ngôn ngữ của văn phạm CFG thông qua các giải thuật tối ưu biến, giản lược, quy chuẩn và các tính chất của lớp ngôn ngữ mà nó mô tả. Câu hỏi đặt ra tiếp theo là có hay không một lớp ôtômát tương ứng để nhận dạng lớp ngôn ngữ phi ngữ cảnh. Như chúng ta đã thấy, lớp ngôn ngữ phi ngữ cảnh thực sự chứa lớp ngôn ngữ chính quy trong đó, nên ôtômát hữu hạn không thể nhận biết tất cả ngôn ngữ phi ngữ cảnh. Một cách trực quan, ôtômát hữu hạn có bộ nhớ bị hạn chế nghiêm ngặt, trong khi việc nhận dạng CFL có thể yêu cầu phải lưu trữ một lượng thông tin khá lớn. Khả năng cho sự mở rộng này sẽ được chúng ta xét đến trong nội dung của chương tiếp theo.

BÀI TẬP CHƯƠNG V

5.1 . Hãy mô tả ngôn ngữ sinh bởi các văn phạm sau :

a) $S \rightarrow aS \mid Sb \mid aSb \mid c$

b) $S \rightarrow SS \mid a \mid b$

c) $S \rightarrow SaS \mid b$

d) $S \rightarrow aSS \mid b$

e)
$$\begin{cases} S \rightarrow aA \mid bB \mid c \\ A \rightarrow Sa \\ B \rightarrow Sb \end{cases}$$

f)
$$\begin{cases} S \rightarrow AB \\ A \rightarrow Sc \mid a \\ B \rightarrow dB \mid b \end{cases}$$

g)
$$\begin{cases} S \rightarrow TT \\ T \rightarrow aTa \mid bTb \mid c \end{cases}$$

5.2. Hãy chỉ ra một văn phạm phi ngữ cảnh sinh ra tập hợp :

a) Tập hợp các chuỗi đọc xuôi đọc ngược như nhau trên bộ chữ cái $\Sigma = \{0,1\}$.

b) Tập hợp chuỗi các dấu ngoặc đúng trong biểu thức số học.

c) Tập hợp $\{a^i b^i c^j \mid i, j \geq 0\}$

d) Tập hợp $\{a^m b^n \mid m, n > 0\}$

e) Tập hợp $\{a^i c a^j \mid i \geq j \geq 0\}$

f) Tập hợp $\{a^j b^j c^i d^i \mid i, j \geq 1\}$

5.3. Cho văn phạm G với các luật sinh sau :

$$\begin{cases} S \rightarrow aB \mid bA \\ A \rightarrow a \mid aS \mid bAA \\ B \rightarrow b \mid bS \mid aBB \end{cases}$$

Với chuỗi **aaabbabbba** , hãy tìm:

a) Dẫn xuất trái nhất.

b) Dẫn xuất phải nhất.

c) Cây dẫn xuất.

d) Văn phạm này có là văn phạm mơ hồ không ?

5.4. Cho văn phạm G với các luật sinh sau :

$$\begin{cases} E \rightarrow T \mid E + T \mid E - T \\ T \rightarrow F \mid T \times F \mid T / F \\ F \rightarrow a \mid b \mid c \mid (E) \end{cases}$$

Hãy vẽ cây dẫn xuất sinh ra các chuỗi nhập sau :

a) $a - (b \times c / a)$

b) $a \times (b - c)$

c) (a + b) / c

5.5. Cho văn phạm : $S \rightarrow aSbS \mid bSaS \mid \varepsilon$

a) Chứng tỏ văn phạm này là văn phạm mơ hồ .

b) Xây dựng dẫn xuất trái (phải) và cây dẫn xuất tương ứng cho chuỗi abab.

c) Văn phạm này sinh ra ngôn ngữ gì ?

5.6. Chứng tỏ văn phạm sau đây là mơ hồ :

$$\begin{cases} S \rightarrow \text{If } \mathbf{b} \text{ then S else S} \\ S \rightarrow \text{If } \mathbf{b} \text{ then S} \\ S \rightarrow \mathbf{a} \end{cases}$$

Trong đó a, b, if, then, else là các ký hiệu kết thúc và S là biến.

5.7. Chứng tỏ văn phạm sau đây là mơ hồ :

$$\begin{cases} S \rightarrow aBS \mid aB \mid bAS \mid bA \\ A \rightarrow bAA \mid a \\ B \rightarrow aBB \mid b \end{cases}$$

Hãy đề nghị một văn phạm không mơ hồ tương đương ?

5.8. Tìm CFG không có chứa ký hiệu vô ích tương đương với văn phạm:

a)
$$\begin{cases} S \rightarrow A \mid a \\ A \rightarrow AB \\ B \rightarrow b \end{cases}$$

b)
$$\begin{cases} S \rightarrow AB \mid CA \\ A \rightarrow a \\ B \rightarrow BC \mid AB \\ C \rightarrow aB \mid b \end{cases}$$

5.9. Tìm văn phạm tương đương với văn phạm sau không có chứa ký hiệu vô ích, luật sinh ε và luật sinh đơn vị :

a) $S \rightarrow aSbS \mid bSaS \mid \varepsilon$

b)
$$\begin{cases} S \rightarrow A \mid B \\ A \rightarrow aB \mid bS \mid b \\ B \rightarrow AB \mid Ba \\ C \rightarrow AS \mid b \end{cases}$$

c)
$$\begin{cases} S \rightarrow ABC \\ A \rightarrow BB \mid \varepsilon \\ B \rightarrow CC \mid a \\ C \rightarrow AA \mid b \end{cases}$$

d)
$$\begin{cases} S \rightarrow A \mid B \\ A \rightarrow C \mid D \\ B \rightarrow D \mid E \\ C \rightarrow S \mid a \mid \varepsilon \\ D \rightarrow S \mid b \end{cases}$$

$E \to S \mid c \mid \varepsilon$

5.10. Tìm văn phạm chỉ có chứa một luật sinh ε duy nhất $S \to \varepsilon$ tương đương với văn phạm sau :

$$\begin{cases} S \to AB \\ A \to SA \mid BB \mid bB \\ B \to b \mid aA \mid \varepsilon \end{cases}$$

5.11. Biến đổi các văn phạm sau đây về dạng chuẩn CHOMSKY:

a) $\begin{cases} S \to bA \mid aB \\ A \to bAA \mid aS \mid a \\ B \to aBB \mid bS \mid b \end{cases}$

b) $\begin{cases} S \to aAB \mid BA \\ A \to BBB \mid a \\ B \to AS \mid b \end{cases}$

c) $\begin{cases} S \to adAda \mid aSa \mid aca \\ A \to bAb \mid bdSdb \end{cases}$

d) $S \to 0S1 \mid 01$

e) $S \to \#S \mid [\, S \supset S] \mid p \mid q$

5.12. Biến đổi các văn phạm sau đây về dạng chuẩn GREIBACH:

a) G ({S, A}, {0, 1}, P, S) với các luật sinh :

$$\begin{cases} S \to AA \mid 0 \\ A \to SS \mid 1 \end{cases}$$

b) G ($\{A_1, A_2, A_3\}$, {a, b}, P, A_1) với các luật sinh :

$$\begin{cases} A_1 \to A_2A_3 \\ A_2 \to A_3A_1 \mid b \\ A_3 \to A_1A_2 \mid a \end{cases}$$

c) G ($\{A_1, A_2, A_3, A_4\}$, {a, b}, P, A_1) với các luật sinh :

$$\begin{cases} A_1 \to A_2A_3 \mid A_3A_4 \\ A_2 \to A_3A_2 \mid a \\ A_3 \to A_4A_4 \mid b \\ A_4 \to A_2A_3 \mid a \end{cases}$$

5.13. Chứng minh rằng các ngôn ngữ sau không phải là CFL:

a) $L = \{a^i b^j c^k \mid i < j < k\}$

b) $L = \{a^i b^j \mid j = i^2\}$

c) $L = \{a^i \mid i$ là số nguyên tố $\}$

d) $L = \{a^n b^n c^n d^n \mid n \geq 0\}$

BÀI TẬP LẬP TRÌNH

5.14. Viết chương trình loại bỏ các ký hiệu vô ích trong một CFG.

5.15. Viết chương trình chuẩn hóa một CFG thành dạng chuẩn CHOMSKY (CNF).

5.16. Viết chương trình chuẩn hóa một CFG thành dạng chuẩn GREIBACH (GNF).