Chương II

NGÔN NGỮ VÀ BIỂU DIỄN NGÔN NGỮ

Nội dung chính : Chương này trình bày quan niệm hình thức về ngôn ngữ và khái niệm về các công cụ dùng để mô tả một tập hữu hạn ngôn ngữ có hiệu quả - đó là văn phạm và ôtômát. Đây là những công cụ có định nghĩa toán học chặt chẽ được nghiên cứu kỹ càng và đã trở thành một thành phần chủ yếu của lý thuyết ngôn ngữ hình thức.

Mục tiêu cần đạt: Sau chương này, mỗi sinh viên cần nắm vững các khái niệm sau :

- Cấu trúc ngôn ngữ tự nhiên cũng như ngôn ngữ lập trình.
- Các phép toán cơ bản trên chuỗi, ngôn ngữ
- Cách thức biểu diễn ngôn ngữ
- Cách phân loại văn phạm theo quy tắc của Noam Chomsky
- Xác định các thành phần của một văn phạm.
- Mối liên quan giữa ngôn ngữ và văn phạm.

Kiến thức cơ bản: Để tiếp thu tốt nội dung của chương này, sinh viên cần có một số các kiến thức liên quan về chuỗi, ký hiệu, từ trong các ngôn ngữ tự nhiên như tiếng Việt, tiếng Anh; cấu trúc cú pháp của các chương trình máy tính viết bằng một số ngôn ngữ lập trình cơ bản như Pascal, C…

Tài liệu tham khảo :

[1] John E. Hopcroft, Jeffrey D.Ullman – *Introduction to Automata Theory, Languages and Computation* – Addison – Wesley Publishing Company, Inc – 1979 ***(trang 1 – trang 12).***

[2] Hồ Văn Quân – *Giáo trình lý thuyết ôtômát và ngôn ngữ hình thức* – Nhà xuất bản Đại học quốc gia Tp. Hồ Chí Minh – 2002 ***(trang 8 – trang 18).***

[3] The Chomsky Hierarchy : http://en.wikipedia.org/wiki/Chomsky_hierarchy

I. TỔNG QUAN VỀ NGÔN NGỮ

Các ngôn ngữ lập trình (như Pascal, C, ...) lẫn ngôn ngữ tự nhiên (như tiếng Việt, tiếng Anh, ...) đều có thể xem như là tập hợp các câu theo một cấu trúc quy định nào đó. Câu của ngôn ngữ, trong tiếng Việt như "*An là sinh viên giỏi*" hay trong Pascal là một đoạn chương trình bắt đầu bằng từ khóa *program* cho đến dấu chấm câu kết thúc chương trình, đều là một chuỗi liên tiếp các từ, như "*An*", "*giỏi*" hay "*begin*", "*if*", "*x2*", "*215*", tức các chuỗi hữu hạn các phần tử của một bộ chữ cái cơ sở nào đó. Ta có thể xem chúng như là các ký hiệu cơ bản của ngôn ngữ.

Từ nhận xét đó, ta dẫn tới một quan niệm hình thức về ngôn ngữ như sau (theo từ điển): *Ngôn ngữ, một cách không chính xác là một hệ thống thích hợp cho việc biểu thị các ý nghĩ, các sự kiện hay các khái niệm, bao gồm một tập các ký hiệu và các quy tắc để vận dụng chúng.*

Định nghĩa trên chỉ cung cấp một ý niệm trực quan về ngôn ngữ chứ không đủ là một định nghĩa chính xác để nghiên cứu về ngôn ngữ hình thức. Chúng ta bắt đầu xây dựng định nghĩa này bằng các khái niệm mà mọi ngôn ngữ đều đặt nền tảng trên đó.

1.1. Bộ chữ cái (alphabet)

Một bộ chữ cái (bộ ký hiệu) là một tập hợp không rỗng, ký hiệu là Σ. Các phần tử của một bộ chữ cái Σ được gọi là các ký hiệu (symbol).

Thí dụ 2.1: Bộ chữ cái Latinh {A, B, C, ... , Z, a, b, c, ... , z}
Bộ chữ cái Hylạp {α, β, γ, …, φ}
Bộ chữ số thập phân {0, 1, 2, ... , 9}
Bộ ký hiệu Moene { . , / , - }
Bộ bit nhị phân { 0, 1}

1.2. Ký hiệu và chuỗi

Một ký hiệu (symbol) là một thực thể trừu tượng mà ta sẽ không định nghĩa được một cách hình thức.

Chẳng hạn : Các chữ cái (a, b, c, ...) hoặc con số (0, 1, 2, ...) là các ký hiệu.

Một chuỗi (string) hay từ (word) trên bộ chữ cái Σ là một dãy hữu hạn gồm một số lớn hơn hay bằng không các ký hiệu của Σ, trong đó một ký hiệu có thể xuất hiện vài lần.

Chẳng hạn : . a, b, c là các ký hiệu còn abcac là một từ.
. ε, 0, 1011, 00010, ... là các từ trên bộ chữ cái Σ = {0, 1}

Độ dài của một chuỗi w, ký hiệu |w| là số các ký hiệu tạo thành chuỗi w.

Chẳng hạn: Chuỗi abca có độ dài là 4 , ký hiệu : |abca | = 4

Chuỗi rỗng (ký hiệu ε) là chuỗi không có ký hiệu nào, vì vậy | ε | = 0.

Chuỗi v được gọi là **chuỗi con** của w nếu v được tạo bởi các ký hiệu liền kề nhau trong chuỗi w.

Chẳng hạn: Chuỗi 10 là chuỗi con của chuỗi 010001

Tiền tố của một chuỗi là một chuỗi con bất kỳ nằm ở đầu chuỗi và **hậu tố** của một chuỗi là chuỗi con nằm ở cuối chuỗi. Tiền tố và hậu tố của một chuỗi khác hơn chính chuỗi đó ta gọi là tiền tố và hậu tố thực sự.

Chẳng hạn: Chuỗi abc có các tiền tố là a, ab, abc và các hậu tố là c, bc, abc

Chuỗi nối kết (ghép) từ hai chuỗi con là một chuỗi tạo được bằng cách viết chuỗi thứ nhất sau đó là chuỗi thứ hai (không có khoảng trống ở giữa).

Chẳng hạn : Nối kết chuỗi Long và Int là chuỗi LongInt.

Sự đặt cạnh nhau như vậy được sử dụng như là một toán tử nối kết. Tức là, nếu w và x là hai chuỗi thì wx là sự nối kết hai chuỗi đó. Chuỗi rỗng là đơn vị của phép nối kết, vì ta có εw = wε = w với mọi chuỗi w.

Ta viết $v^0 = \varepsilon$; $v^1 = v$; $v^2 = vv$ ... hay tổng quát $v^i = vv^{i-1}$ với $i > 0$.

Chuỗi đảo ngược của chuỗi w, ký hiệu w^R là chuỗi w được viết theo thứ tự ngược lại, nghĩa là nếu $w = a_1 a_2 ... a_n$ thì $w^R = a_n a_{n-1} ... a_1$. Hiển nhiên : $\varepsilon^R = \varepsilon$

1.3. Ngôn ngữ (Languages)

Một ngôn ngữ (hình thức) L là một tập hợp các chuỗi của các ký hiệu từ một bộ chữ cái Σ nào đó.

Tập hợp chứa chuỗi rỗng (ký hiệu {ε}) và tập hợp rỗng ∅ cũng được coi là ngôn ngữ. Chú ý rằng hai ngôn ngữ đó là khác nhau: ngôn ngữ ∅ không có phần tử nào trong khi ngôn ngữ {ε} có một phần tử là chuỗi rỗng ε.

Tập hợp tất cả các chuỗi con kể cả chuỗi rỗng trên bộ chữ cái cố định Σ, ký hiệu là Σ^* cũng là một ngôn ngữ. Mỗi ngôn ngữ trên bộ chữ cái Σ đều là tập con của Σ^*. Chú ý rằng Σ^* vô hạn đếm được với mọi Σ khác ∅, vì ta có thể liệt kê tất cả các chuỗi con của nó theo thứ tự độ dài tăng dần, khi có cùng độ dài thì liệt kê theo thứ tự từ điển.

Ngoài ra tập hợp tất cả các chuỗi sinh ra từ bộ chữ cái Σ, ngoại trừ chuỗi rỗng ε, được ký hiệu là Σ^+. Dễ thấy:

$$\Sigma^+ = \Sigma^* - \{\varepsilon\} \qquad \text{hay} \qquad \Sigma^* = \Sigma^+ + \{\varepsilon\}$$

Thí dụ 2.2 : $\Sigma = \{a\}$ thì $\Sigma^* = \{\varepsilon, a, aa, aaa, ...\}$

$\Sigma^+ = \{a, aa, aaa, ...\}$

$\Sigma = \{0, 1\}$ thì $\Sigma^* = \{\varepsilon, 0, 1, 00, 01, 10, 11, 000, ...\}$

$\Sigma^+ = \{0, 1, 00, 01, 10, 11, 000, ...\}$

1.4. Các phép toán trên ngôn ngữ

Từ các ngôn ngữ có trước, ta có thể thu được các ngôn ngữ mới nhờ áp dụng các phép toán trên ngôn ngữ. Trước hết, vì ngôn ngữ là một tập hợp, nên mọi phép toán trên tập hợp như: hợp (union), giao(intersection) và hiệu (difference) ... đều có thể áp dụng lên các ngôn ngữ. Ngoài ra, còn có thêm một số phép toán thường gặp khác như sau :

Phép phần bù (complement) của một ngôn ngữ L trên bộ chữ cái Σ được định nghĩa như sau :

$$\overline{L} = \Sigma^* - L$$

với chú ý khái niệm bù của ngôn ngữ được định nghĩa dựa trên Σ^*

Phép nối kết (concatenation) của hai ngôn ngữ L_1 trên bộ chữ cái Σ_1 và L_2 trên bộ chữ cái Σ_2 được định nghĩa bởi :

$$L_1L_2 = \{w_1w_2 \mid w_1 \in L_1 \text{ và } w_2 \in L_2\} \text{ trên bộ chữ cái } \Sigma_1 \cup \Sigma_2$$

Ký hiệu L^i được mở rộng để dùng cho phép nối kết nhiều lần (còn gọi là phép lũy thừa trên chuỗi) trên cùng một tập ngôn ngữ L, tổng quát : $L^i = LL^{i-1}$. Theo định nghĩa, ta có một trường hợp đặc biệt : $L^0 = \{\varepsilon\}$, với mọi ngôn ngữ L.

Phép bao đóng (closure) : Trong nhiều trường hợp, người ta muốn thành lập một ngôn ngữ bằng cách nối kết các chuỗi (với số lượng bất kỳ) lấy trong một ngôn ngữ L cho trước, các phép toán đó như sau :

Bao đóng (Kleene) của ngôn ngữ L, ký hiệu L^* được định nghĩa là hợp của mọi tập tích trên L :

$$L^* = \bigcup_{i=0}^{\infty} L^i$$

Bao đóng dương (positive) của ngôn ngữ L, ký hiệu L^+ được định nghĩa là hợp của mọi tích dương trên L :

$$L^+ = \bigcup_{i=1}^{\infty} L^i$$

Chú ý rằng : $L^+ = lL^* = L^*L$

$L^* = L^+ \cup \{\varepsilon\}$

Thí dụ 2.3 : Cho ngôn ngữ L = { a, ba } thì

L^2 = { aa, aba, baa, baba, … }

L^3 = { aaa, aaba, abaa, ababa, baaa, baaba, babaa, bababa, … }

L^* = { ε, a, ba, aa, aba, baa, baba, aaa, aaba, abaa, ababa, baaa, baaba, … }

II. VẤN ĐỀ BIỂU DIỄN NGÔN NGỮ

Như đã định nghĩa ở trên, một ngôn ngữ L trên một bộ chữ cái Σ là một tập con của tập Σ^*. Vậy vấn đề đặt ra là đối với một ngôn ngữ L, làm sao có thể chỉ rõ các chuỗi có thuộc vào L hay không ? Đó chính là vấn đề biểu diễn ngôn ngữ .

Đối với các ngôn ngữ hữu hạn, để biểu diễn chúng một cách đơn giản ta chỉ cần liệt kê tất cả các chuỗi thuộc vào chúng.

Chẳng hạn : $L_1 = \{\varepsilon\}$

L_2 = { a, ba, aaba, bbbbb }

Tuy nhiên, trong trường hợp các ngôn ngữ là vô hạn, ta không thể liệt kê tất cả các chuỗi thuộc ngôn ngữ được mà phải tìm cho chúng một cách biểu diễn hiệu quả khác.

Trong những trường hợp không phức tạp lắm, người ta thường xác định các chuỗi bằng cách chỉ rõ một đặc điểm chủ yếu chung cho các chuỗi đó. Đặc điểm này thường được mô tả qua một phát biểu hay một tân từ.

Chẳng hạn : $L_3 = \{ a^i \mid i$ là một số nguyên tố $\}$

$L_4 = \{ a^i b^j \mid i \geq j \geq 0 \}$

$L_5 = \{ w \in \{ a, b\}^* \mid$ số a trong w = số b trong w $\}$

Song, trong phần lớn các trường hợp, người ta thường biểu diễn ngôn ngữ một cách tổng quát thông qua một văn phạm hay một ôtômát. Văn phạm là một cơ chế cho phép sản sinh ra mọi chuỗi của ngôn ngữ, trong khi ôtômát lại là cơ chế cho phép đoán nhận một chuỗi bất kỳ có thuộc ngôn ngữ hay không. Về mặt hình thức, cả văn phạm và ôtômát đều là các cách biểu hiện khác nhau của cùng một quan niệm.

Thí dụ 2.4 : Cho L là một ngôn ngữ trên bộ chữ cái Σ = {a, b} được định nghĩa như sau:

i) $\varepsilon \in L$

ii) Nếu $X \in L$ thì $aXb \in L$

iii) Không còn chuỗi nào khác thuộc L

Định nghĩa đệ quy trên cho ta một cách sản sinh ra các chuỗi thuộc ngôn ngữ L như sau : Do (i) nên ta có chuỗi đầu tiên trong L là **ε**. Xem đó là X thì theo (ii) ta lại có được chuỗi thứ hai aεb hay **ab**. Áp dụng lặp đi lặp lại quy tắc (ii) ta lại tìm được các chuỗi: **aabb**, rồi lại **aaabbb**, … Cứ như thế có thể phát sinh tất cả các chuỗi thuộc ngôn ngữ L. Bằng cách áp dụng (một số hữu hạn) quy tắc phát sinh như trên, ta có thể phát sinh bất kỳ chuỗi nào trong ngôn ngữ.

Dễ dàng nhận thấy : $L = \{a^i b^i \mid i \geq 0\}$

Trong giáo trình này, chúng ta sẽ tập trung nghiên cứu hai dạng hệ phát sinh dùng để biểu diễn ngôn ngữ, như đã nói ở trên, là văn phạm và ôtômát. Bằng cách ấn định các dạng khác nhau vào các quy tắc phát sinh, người ta cũng định nghĩa nhiều loại văn phạm và ôtômát khác nhau, từ đơn giản đến phức tạp, nghiên cứu các ngôn ngữ sản sinh hay đoán nhận bởi chúng và mối liên quan giữa chúng với nhau.

III. VĂN PHẠM VÀ SỰ PHÂN LỚP VĂN PHẠM

Với mục đích sản sinh (hay đoán nhận) ngôn ngữ, văn phạm được dùng như một cách thức hiệu quả để biểu diễn ngôn ngữ.

3.1. Định nghĩa văn phạm cấu trúc (Grammar)

Theo từ điển, văn phạm, một cách không chính xác, là một tập các quy tắc về cấu tạo từ và các quy tắc về cách thức liên kết từ lại thành câu.

Để hiểu rõ hơn khái niệm này, ta xét ví dụ cây minh họa cấu trúc cú pháp của một câu đơn trong ngôn ngữ tiếng Việt "*An là sinh viên giỏi*" ở thí dụ 1.5 của chương 1. Xuất phát từ nút gốc theo dần đến nút lá, ta nhận thấy các từ ở những nút lá của cây như "*An*", "*sinh viên*", "*giỏi*", … là những từ tạo thành câu được sản sinh. Ta gọi đó là các **ký hiệu kết thúc** bởi vì chúng không còn phát sinh thêm nút nào trên cây và câu được hoàn thành. Trái lại, các nút trong của cây như "*câu đơn*", "*chủ ngữ*", "*danh từ*", … sẽ không có mặt trong dạng câu sản sinh, chúng chỉ giữ vai trò trung gian trong việc sinh chuỗi, dùng diễn tả cấu trúc câu. Ta gọi đó là các **ký hiệu chưa kết thúc**.

Quá trình sản sinh câu như trên thực chất là sự diễn tả thông qua cấu trúc cây cho một quá trình phát sinh chuỗi. Các chuỗi được phát sinh bắt đầu từ một ký hiệu chưa kết thúc đặc biệt, sau mỗi bước thay thế một ký hiệu chưa kết thúc nào đó trong chuỗi thành một chuỗi lẫn lộn gồm các ký hiệu kết thúc và chưa, cho đến khi không còn một ký hiệu chưa kết thúc nào nữa thì hoàn thành. Quá trình này chính là phương thức phát sinh chuỗi của một văn phạm, được định nghĩa hình thức như sau:

Định nghĩa : Văn phạm cấu trúc G là một hệ thống gồm bốn thành phần xác định như sau **G (V, T, P, S)**, trong đó:

. **V** : tập hợp các biến (**v**ariables) hay các ký hiệu chưa kết thúc (non terminal)

. **T** : tập hợp các ký hiệu kết thúc (**t**erminal) (với $V \cap T = \varnothing$)

. **P** : tập hữu hạn các quy tắc ngữ pháp được gọi là các luật sinh (**p**roduction), mỗi luật sinh được biểu diễn dưới dạng $\alpha \rightarrow \beta$, với α, β là các chuỗi $\in (V \cup T)^*$.

. **S** $\subset$ V: ký hiệu chưa kết thúc dùng làm ký hiệu bắt đầu (**s**tart)

Người ta thường dùng các chữ cái Latinh viết hoa (A, B, C, ...) để chỉ các ký hiệu trong tập biến V; các chữ cái Latinh đầu bảng viết thường (a, b, c, ...) dùng chỉ các ký hiệu kết thúc thuộc tập T. Chuỗi các ký hiệu kết thúc thường được biểu diễn bằng các chữ cái Latinh cuối bảng viết thường (x, y, z, ...).

Nhận xét : Bằng quy ước này chúng ta có thể suy ra các biến, các ký hiệu kết thúc và ký hiệu bắt đầu của văn phạm một cách xác định và duy nhất bằng cách xem xét các luật sinh. Vì vậy, để biểu diễn văn phạm, một cách đơn giản người ta chỉ cần liệt kê tập luật sinh của chúng.

Từ văn phạm, để sinh ra được các câu (từ), ta định nghĩa khái niệm "dẫn xuất" như sau :

Nếu $\alpha \rightarrow \beta$ là một luật sinh thì $\gamma\ \boldsymbol{\alpha}\ \delta \Rightarrow \gamma\ \boldsymbol{\beta}\ \delta$ gọi là một **dẫn xuất trực tiếp**, có nghĩa là áp dụng luật sinh $\alpha \rightarrow \beta$ vào chuỗi $\gamma\ \boldsymbol{\alpha}\ \delta$ để sinh ra chuỗi $\gamma\ \boldsymbol{\beta}\ \delta$.

Nếu các chuỗi $\alpha_1, \alpha_2, ..., \alpha_m \in \Sigma^*$ và $\alpha_1 \Rightarrow \alpha_2, \alpha_2 \Rightarrow \alpha_3, ..., \alpha_{m-1} \Rightarrow \alpha_m$ thì ta nói α_m có thể được dẫn ra từ α_1 thông qua chuỗi dẫn xuất $\alpha_1 \Rightarrow \alpha_2, \alpha_2 \Rightarrow \alpha_3, ..., \alpha_{m-1} \Rightarrow \alpha_m$ hay α_1 **dẫn xuất (gián tiếp)** ra α_m, viết tắt là $\alpha_1 \Rightarrow^* \alpha_m$.

Ngôn ngữ của văn phạm G (V, T, P, S) là tập hợp các chuỗi ký hiệu kết thúc $w \in T^*$ được sinh ra từ ký hiệu bắt đầu S của văn phạm bởi các luật sinh thuộc tập P, ký hiệu là L(G) :

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

Một ngôn ngữ có thể có nhiều cách đặc tả, do đó cũng có thể có nhiều văn phạm khác nhau sinh ra cùng một ngôn ngữ. Hai văn phạm sinh ra cùng một ngôn ngữ thì gọi là tương đương.

$$G_1 \text{ tương đương } G_2 \quad \Leftrightarrow \quad L(G_1) = L(G_2)$$

3.2. Sự phân cấp Chomsky trên văn phạm

Bằng cách áp đặt một số quy tắc hạn chế trên các luật sinh, Noam Chomsky đề nghị một hệ thống phân loại các văn phạm dựa vào tính chất của các luật sinh. Hệ thống này cho phép xây dựng các bộ nhận dạng hiệu quả và tương thích với từng lớp văn phạm. Ta có 4 lớp văn phạm như sau :

1) Văn phạm loại 0: Một văn phạm không cần thỏa ràng buộc nào trên tập các luật sinh được gọi là văn phạm loại 0 hay còn được gọi là ***văn phạm không hạn chế*** (Unrestricted Grammar)

2) Văn phạm loại 1: Nếu văn phạm G có các luật sinh dạng $\alpha \rightarrow \beta$ và thỏa $|\beta| \geq |\alpha|$ thì G là văn phạm loại 1 hoặc còn được gọi là ***văn phạm cảm ngữ cảnh CSG*** **(C**ontext-**S**ensitive **G**rammar)

Ngôn ngữ của lớp văn phạm này được gọi là ngôn ngữ cảm ngữ cảnh (CSL)

3) Văn phạm loại 2: Nếu văn phạm G có các luật sinh dạng $A \rightarrow \alpha$ với A là một biến đơn và α là một chuỗi các ký hiệu $\in (V \cup T)^*$ thì G là văn phạm loại 2 hoặc còn được gọi là ***văn phạm phi ngữ cảnh CFG*** **(C**ontext-**F**ree **G**rammar)

Ngôn ngữ của lớp văn phạm này được gọi là ngôn ngữ phi ngữ cảnh (CFL)

4) Văn phạm loại 3: Nếu văn phạm G có mọi luật sinh dạng **tuyến tính phải** (right-linear): $A \rightarrow wB$ hoặc $A \rightarrow w$ với A, B là các biến đơn và w là chuỗi ký hiệu kết thúc (có thể rỗng); hoặc có dạng **tuyến tính trái** (left-linear): $A \rightarrow Bw$ hoặc $A \rightarrow w$ thì G là văn phạm loại 3 hay còn được gọi là ***văn phạm chính quy RG*** **(R**egular **G**rammar)

Ngôn ngữ của lớp văn phạm này được gọi là ngôn ngữ chính quy (RL)

Ký hiệu : L_0, L_1, L_2, L_3 là các lớp ngôn ngữ sinh ra bởi các văn phạm loại 0, 1, 2, 3 tương ứng. Ta có : $L_3 \subset L_2 \subset L_1 \subset L_0$ và các bao hàm thức này là nghiêm ngặt.

Thí dụ 2.5 :

1. Xét văn phạm G :

   $V = \{S, A\}$, $T = \{a, b\}$ và tập $P = \{ S \rightarrow aS$
   $S \rightarrow aA$
   $A \rightarrow bA$
   $A \rightarrow b \}$

   Đây là văn phạm loại 3 (vì tập luật sinh có dạng tuyến tính phải).

   Chẳng hạn, một dẫn xuất từ S có dạng :

   $S \Rightarrow aS \Rightarrow aaS \Rightarrow aaaA \Rightarrow aaabA \Rightarrow aaabbA \Rightarrow aaabbbA \Rightarrow aaabbbb = a^3 b^4$

   Hay văn phạm sinh ra ngôn ngữ $L(G_3) = \{a^+b^+\} = \{a^n b^m \mid n, m \geq 1 \}$

2. Xét văn phạm G :

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

   Đây là văn phạm loại 2.

   Chẳng hạn, một dẫn xuất từ S có dạng :

   $S \Rightarrow aSb \Rightarrow aaSbb \Rightarrow aaaSbbb \Rightarrow aaaabbbb = a^4 b^4$

   Hay văn phạm sinh ra ngôn ngữ $L(G_2) = \{a^n b^n \mid n \geq 1\}$

3. Xét văn phạm G :

   $V = \{S, B, C\}$, $T = \{a, b, c\}$ và tập $P = \{ S \rightarrow aSBC$

$S \rightarrow aBC$
$CB \rightarrow BC$
$aB \rightarrow ab$
$bB \rightarrow bb$
$bC \rightarrow bc$
$cC \rightarrow cc$ }

Đây là văn phạm loại 1.

Chẳng hạn, một dẫn xuất từ S có dạng :

$S \Rightarrow aSBC \Rightarrow aaBCBC \Rightarrow aabCBC \Rightarrow aabBCC \Rightarrow aabbCC \Rightarrow aabbcC \Rightarrow aabbcc = a^2b^2c^2$

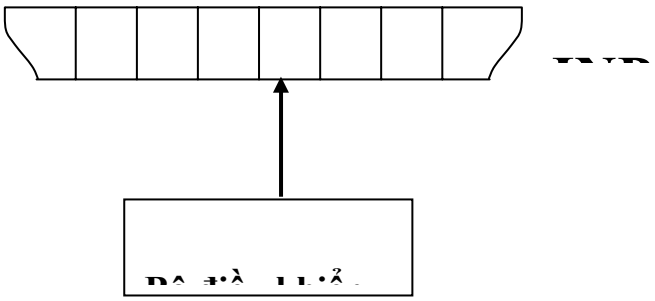
Hay văn phạm sinh ra ngôn ngữ $L(G_1) = \{a^nb^nc^n \mid n > 0\}$.

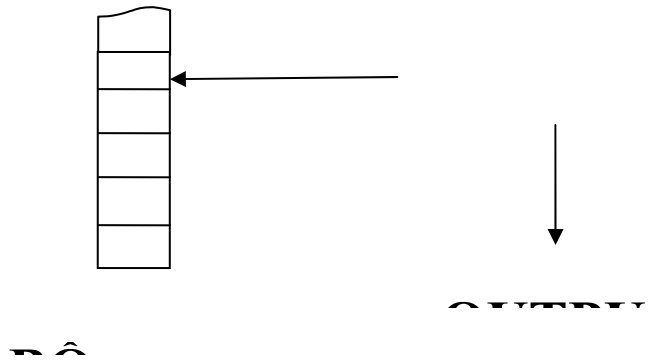
IV. CƠ CHẾ ÔTÔMÁT

4.1. Định nghĩa ôtômát

Ngoài các văn phạm, người ta còn sử dụng một phương tiện khác để xác định ngôn ngữ là ôtômát. Ôtômát, dịch nghĩa là máy tự động, được hiểu là các "máy" trừu tượng có cơ cấu và hoạt động rất đơn giản nhưng có khả năng đoán nhận ngôn ngữ. Với một chuỗi bất kỳ, sau một số bước làm việc, ôtômát sẽ cho câu trả lời chuỗi đó có thuộc ngôn ngữ hay không. Để có được quá trình tự động như vậy, con người thường phải lập trình sẵn cho nó một "lộ trình" thực hiện, và các máy chỉ cần hoạt động theo đúng lộ trình này. Một trong số những máy tự động này điển hình mạnh nhất có thể nói chính là máy tính số ngày nay. Tuy hoạt động theo kiểu "máy", song thực chất mỗi bước làm việc của ôtômát là một sự thay thế ký hiệu, nghĩa là một bước dẫn xuất như đã nói ở trên.

Nói chung, một mô hình ôtômát thường bao gồm những thành phần chủ yếu như sau :





***Hình 2.1 -** Mô hình chung cho một ôtômát*

Chuỗi nhập cần xác định sẽ được lưu trữ trên băng input. Tại mỗi thời điểm, ứng với trạng thái hiện thời, đọc vào một ký tự nhập trên băng input, có thể kết hợp với việc xem xét ký hiệu tương ứng trong Bộ nhớ, Bộ điều khiển của ôtômát sẽ quyết định bước chuyển đến trạng thái kế tiếp.

Các loại ôtômát tương ứng với từng lớp văn phạm sẽ được giới thiệu lần lượt trong những chương tiếp theo.

4.2. Phân loại các ôtômát

Dựa theo hoạt động của ôtômát, thông thường người ta chia ôtômát thành hai dạng sau:

Ôtômát đơn định (Deterministic Automata) : Là một ôtômát mà tại mỗi bước di chuyển chỉ được xác định duy nhất bởi cấu hình hiện tại. Sự duy nhất này thể hiện tính đơn định, nghĩa là hàm chuyển của ôtômát dạng này luôn là đơn trị.

Ôtômát không đơn định (Non - deterministic Automata) : Là một ôtômát mà tại mỗi bước di chuyển, nó có một vài khả năng để chọn lựa. Sự chọn lựa này thể hiện tính không đơn định, nghĩa là hàm chuyển của ôtômát dạng này là đa trị.

BÀI TẬP CHƯƠNG II

2.1. Chứng minh hoặc bác bỏ : $L^+ = L^* - \{\varepsilon\}$.

2.2. L^+ hay L^* có thể bằng $\varnothing$ không ? Khi nào thì L^+ hay L^* là hữu hạn ?

2.3. Hãy cho biết các thứ tự cho phép liệt kê các phần tử của các ngôn ngữ sau :

a) $\{a, b\}^*$

b) $\{a\}^*\{b\}^*\{c\}^*$

c) $\{w \mid w \in \{a, b\}^+$ và số a bằng số b trong w$\}$

2.4. Một chuỗi hình tháp có thể định nghĩa là một chuỗi đọc xuôi hay ngược đều như nhau, hoặc cũng có thể định nghĩa như sau :

1) ε là chuỗi hình tháp.

2) Nếu a là một ký hiệu bất kỳ thì a là một chuỗi hình tháp.

3) Nếu a là một ký hiệu bất kỳ và X là một chuỗi hình tháp thì aXa là một chuỗi hình tháp.

4) Không còn chuỗi hình tháp nào ngoài các chuỗi cho từ (1) đến (3).

Hãy chứng minh quy nạp rằng 2 định nghĩa trên là tương đương.

2.5. Các chuỗi ngoặc đơn cân bằng được định nghĩa theo 2 cách :

Cách 1 : Một chuỗi w trên bộ chữ cái { **(** , **)** } là cân bằng khi và chỉ khi :

a) w chứa cùng một số '**)**' và '**(**'

b) Mọi tiền tố của w chứa số các '**(**' ít nhất bằng số các '**)**'.

Cách 2 :

a) **(** là chuỗi ngoặc đơn cân bằng

b) Nếu w là một chuỗi ngoặc đơn cân bằng, thì **(**w**)** là chuỗi ngoặc đơn cân bằng.

c) Nếu w và x là các chuỗi ngoặc đơn cân bằng, thì wx là chuỗi ngoặc đơn cân bằng.

d) Không còn chuỗi ngoặc đơn cân bằng nào khác với trên.

Hãy chứng minh bằng quy nạp theo độ dài chuỗi rằng 2 định nghĩa trên là tương đương.