NOVAS TÉCNICAS PARA ANALISADORES SINTÁTICOS TIPO MATRIZ DE TRANSIÇÃO

– Estevam Gilberto De Simone –

TESE SUBMETIDA AO CORPO DOCENTE DA COORDENAÇÃO DOS PROGRAMAS DE PÓS-GRADUAÇÃO DE ENGENHARIA DA UNIVERSIDADE FEDERAL DO RIO DE JANEIRO COMO PARTE DOS REQUISITOS NECESSÁRIOS PARA A OBTENÇÃO DO GRAU DE DOUTOR EM CIÊNCIAS (D.Sc.).

Aprovada por:

José Lucas Mourão Rangel Neto
(Presidente)

Jayme Szwarcficter

Sergio Roberto Teixeira

Sueli Mendes dos Santos

Valdemar Setzer

RIO DE JANEIRO

DEZEMBRO DE 1980

SIMONE, *ESTEVAM GILBERTO DE*

Novas Técnicas para Analisadores Sintáticos Tipo Matriz de Transição [Rio de Janeiro] 1980

X, 157 p. 29,7 cm (COPPE-UFRJ, D.Sc., Engenharia de Sistemas e Computação, 1980)

Tese - Univ. Fed. Rio de Janeiro, Fac. Engenharia.

1. Compiladores I. COPPE/UFRJ II Título (série)

— Ã *Estevam e Elvira*

AGRADECIMENTOS

- Aos meus alunos da COPPE e Instituto de Matemática da UFRJ;
- Aos meus colegas professores da área de Computação e do Departamento de Ciência da Computação;
- Ao Lucas, não apenas pelos cursos, pela orientação e pela paciência mas, principalmente, por ter emprestado a este trabalho um pouco do brilho de sua inteligência;
- À Ligia, à Beti, ao Zé Carlos e à Cecília pela ajuda importante e desinteressada;
- Aos professores Celio Guimarães, Jayme Szwarcfiter, Sergio Teixeira, Sueli Mendes dos Santos e Valdemar Setzer por acederem em participar desta banca;
- à COPPE, a FINEP e a CAPES pelo apoio recebido.
- À Stavna, pelo fantástico trabalho de datilografia, desenho, composição gráfica (e até correção de erros de português e conteudo)!

RESUMO

O método de análise sintática por matrizes de transição proposto por |*Gries, 68*| é totalmente reelaborado. É proposta uma definição equivalente para a classe de gramáticas reconhecida pelo método. É fornecido um algorítmo construtor mais eficiente para as tabelas de controle do analisador. É alterada a definição das funções que controlam o analisador e fornecido seu novo algorítmo. Estas novas funções são representáveis em tabelas que permitem determinar seus pontos inacessiveis e são fornecidos algorítmos de compactação das tabelas que permitem reduzir grandemente o espaço ocupado, preservado o formato matricial.

Paralelamente é desenvolvido um algorítmo de recuperação de erros que permite uma recuperação sofisticada e precisa, sem utilizar informação adicional às tabelas do analisador, sem degradar o desempenho do analisador para programas corretos e corrigindo de alguma forma as imprecisões na detecção do erro pelo analisador.

Demonstra-se que as tabelas originais de um analisador de gramáticas por matrizes de transição são menores que suas correspondentes para gramáticas SLR (1) e que o analisador por matrizes de transição é mais rápido que o correspondente analisador SLR(1).

ABSTRACT

Transition matrices have been used in parsing, since the method was proposed by |Gries, 68|. We present here a completely new vèrsion of the method, with a new equivalent definition for the class of grammars accepted.

New, more efficient algorithms are presented both for parsing and for the construction of the parsing tables. The control functions are redefined, and allow the determination of inaccesible points and table compression.

In parallel, an algorithm for error recovery is present ed which results in a sophisticate and accurate recovery, which need no information besides the one in the parsing tables, and which introduces no degradation of the performance in the parser.

It is shown that the original tables of the transition matrix parser are smaller than the corresponding SLR(1) parsers, and also faster.

ÍNDICE

ÍNDICE DOS PARÁGRAFOS

CAPÍTULO I

— INTRODUÇÃO —

De todas as fases de um compilador a análise sintática é, sem dúvida, o problema melhor resolvido. Entretanto, talvez pela própria variedade de soluções distintas, é um campo aberto a polêmicas já clássicas: analisadores sintáticos ascendentes contra descendentes, analisadores controlados por tabelas ou por programas, geração automática de analisadores contra programação manual, gramáticas SLR contra gramáticas LL contra gramáticas LALR contra gramáticas LR contra gramáticas de precedência contra gramáticas recursivo-descendentes, e muitas mais. É realmente um tanto arriscado aventurar-se a pesquisar neste terreno.

Este trabalho é uma recomposição do método de análise sintática por matrizes de transição, proposto por |*Gries, 68*|. De fato, restou daquele método apenas o essencial: a classe de gramáticas, ditas "*gramáticas de matrizes de transição*" (GMT).

Não se encontra na literatura, referências posteriores a 1972 para analisadores GMT. Entre nós, todavia, o método ainda é bastante popular, sendo utilizado atualmente para a feitura de compiladores comerciais ou didáticos. A origem deste trabalho está no desenvolvimento do primeiro gerador de analisadores sintáticos feito na COPPE/UFRJ. Inicialmente voltado apenas para GMT esse gerador — o NHÃO NHÃO — deu origem a geradores de analisadores para as principais classes de gramáticas. O NHÃO NHÃO, em uso desde 1978, já permitiu a geração de tabelas de controle para uma dezena de linguagens distintas, usando matrizes de transição.

O trabalho está dividido nas seguintes secções principais:

a) no Capítulo II é apresentado o estritamente necessário de teoria de linguagens, relações binárias, grafos e gramáticas de precedência de operadores, visando fixar os conceitos e a notação;

b) no Capítulo III faz-se uma revisão do método original com suas principais propriedades;

c) o Capítulo IV é o centro do trabalho onde o método é reformulado, são demonstrados os teoremas principais, é definido o novo conjunto de relações, é redefinida a classe de gramáticas e são propostos o novo analisador sintático e o novo construtor de analisadores sintáticos; além disso é apresentado o método de compactação das tabelas;

d) no Capítulo V, é feita uma revisão bibliográfica de recuperação de erros e apresentado o algorítmo recuperador;

e) o capítulo VI é dedicado a avaliação do trabalho e às conclusões.

Visando criar um sistema de referência sobre o próprio texto este foi paragrafado sendo os parágrafos numerados em romano e claramente delimitados. A notação utilizada está definida no Capítulo II ou na sua primeira ocorrência. O texto faz uso intensivo dessa notação menos pela afinidade do autor ao rigorismo que por sua capacidade de síntese.

CAPÍTULO II

— FUNDAMENTOS —

2.1 – FORMALISMO E DEFINIÇÕES BÁSICAS

São introduzidos aqui os conceitos necessários ao desenvolvimento da teoria nos capítulos posteriores. Paralelamente às definições será estabelecida sua notação. Serão adotadas a simbologia e a estruturação teórica de |*Aho & Ullman, 72*|.

■ Notação CONJUNTOS (I)

Um conjunto X formado pelos elementos $x_1, x_2, \ldots, x_n$ será denotado

(i) $X = \{x_1, x_2, \ldots, x_n\}$

O número de elementos desse conjunto será

(ii) $\#(X)$.

■

■ Definição GRAMÁTICA LIVRE DE CONTEXTO (GLC) (II)

Uma gramática G é dita livre de contexto ou, equivalentemente $G \in \mathcal{C}_{GLC}$, quando G é uma quádrupla (N, Σ, P, S) com:

(i) N, um conjunto finito de símbolos denominados não-terminais;

(ii) Σ, um conjunto finito de símbolos denominados terminais;

(iii) $N \cap \Sigma = \Phi$;

(iv) P, um conjunto finito de pares denominados produções onde o primeiro elemento é um não-terminal e o segundo é uma sequência não-vazia de terminais e/ou não-terminais. Ou seja,

$$P = \{(A,\alpha) \mid A \in N \wedge \alpha \in (N \cup \Sigma)^+\}$$

Uma produção $(A, \alpha) \in P$ será denotada $(A \rightarrow \alpha)$;

(v) $S \in N$, um não-terminal especial denominado símbolo inicial.

Omitiremos, doravante, a menção "*livre de contexto*".

■ <u>Notação</u> CONJUNTOS GRAMATICAIS (III)

(i) NÃO TERMINAIS

$N = \{A, B, C, ..., T\}$ ou $N = \{$PROGRAMA, BLOCO, ...$\}$

(ii) TERMINAIS

$\Sigma = \{a, b, c, ..., t\}$ ou $\Sigma = \{$if, then, ...$\}$

(iii) TERMINAIS OU NÃO-TERMINAIS

$(N \cup \Sigma) = \{U, V, X, ..., Z\}$

(iv) SEQUENCIAS DE TERMINAIS

$(\Sigma)^* = \{u, v, x, ... z\}$

(v) SEQUENCIAS DE TERMINAIS E/OU NÃO-TERMINAIS

$(N \cup \Sigma)^* = \{\alpha, \beta, \gamma ..., \tau\}$

(vi) SEQUÊNCIA VAZIA

ε

___________ ■

■ <u>Definição</u> DERIVAÇÃO E REDUÇÃO (IV)

Sejam G = (N, C, P, S) e $\beta A\gamma \in (N \cup \Sigma)^*$.

Diremos que $\beta A\gamma$ *"deriva diretamente"* $\beta\alpha\gamma$ em G, se (A → a) E P. Indica-se por:

(i) $\beta A\gamma \underset{G}{=>} \beta\alpha\gamma$

Diremos ainda nesse caso que $\beta\alpha\gamma$ *"reduz-se diretamente"* a $\beta A\gamma$ em G e indicamos por:

(ii) $\beta A\gamma \underset{G}{<=} \beta\alpha\gamma$

Por extensão, diremos que a *"deriva em zero ou mais* PUA-*sos"* - ou simplesmente *"deriva"* - 6 se existirem sequencias β, γ, ... tais que

$$\alpha \underset{G}{=>} \beta \underset{G}{=>} \gamma \underset{G}{=>} \cdots \underset{G}{=>} \delta$$

e indicamos

(iii) $\alpha \overset{*}{\underset{G}{=>}} \delta$

Analogamente, 6 *"reduz-se"* a a em G e

(iv) a $\overset{*}{\underset{G}{<=}}$ 6

Quando nenhuma produção for aplicada ainda teremos e

Quando pelo menos uma produção for aplicada diremos que

(vii) $\alpha \underset{G}{\overset{+}{\Longrightarrow}} \delta$ e (viii) $a \underset{G}{\overset{+}{\Longleftarrow}} 6$

Omitiremos o nome da gramática quando não causar dúvida.

Definição FORMA SENTENCIAL E SENTENÇA (V)

(i) se $G = (N, C, P, S)$ e $S \Rightarrow a \Rightarrow \beta \Rightarrow \ldots \Rightarrow \gamma$ então $\alpha, \beta, \ldots, \gamma$ serão formas sentenciais de G.

(ii) se $S \underset{G}{\overset{+}{\Longrightarrow}} u$ então u será sentença produzida por G.

Definição LINGUAGEM (VI)

Se $G = (N, C, P, S)$ então a linguagem produzida por G será

$$L(G) = \{u \in \Sigma^* \mid S \underset{G}{\overset{*}{\Longrightarrow}} u\}$$

Definição GRAMÁTICA REDUZIDA (VII)

Uma gramática $G = (N, \Sigma, P, S)$ será dita reduzida se, simultaneamente:

(i) $L(G) \neq \Phi$ (caso particular de (iii) abaixo)

(ii) $(A \to \alpha) \in P \Rightarrow A \neq \alpha$

(iii) $A \in N \Rightarrow L(G') \neq \Phi$, $G' = (N, \Sigma, P, A)$

(iv) $A \in N \Rightarrow S \overset{*}{\underset{G}{\Longrightarrow}} \alpha A \beta$

■

■ Definição GRAMÁTICA UNICAMENTE INVERSÍVEL (VIII)

Uma gramática reduzida $G = (N, \Sigma, P, S)$ será dita unicamente inversível se não possuir duas produções com mesmo lado direito.

■

■ Definição DERIVAÇÃO (REDUÇÃO) DIREITA (ESQUERDA) (IX)

Se $G = (N, \Sigma, P, S)$ e $S \Rightarrow \alpha Au \Rightarrow \beta Bv \Rightarrow \ldots \Rightarrow \gamma Cx$ então S deriva γCx por derivação mais a direita ou, simplesmente, derivação direita e

(i) $S \overset{*}{\underset{DIR}{\Longrightarrow}} \gamma Cx$

Ainda neste caso,

(ii) $S \overset{*}{\underset{DIR}{\Longleftarrow}} \gamma Cx$ (redução direita)

Se $S \Rightarrow uA\alpha \Rightarrow vB\beta \Rightarrow \ldots \Rightarrow xC\gamma$ então S deriva $xC\gamma$ por derivação esquerda e

(iii) $S \overset{*}{\underset{ESQ}{\Longrightarrow}} xC\gamma$

e, ainda,

(iv) $S \overset{*}{\underset{ESQ}{\Leftarrow}} xC\gamma$ (redução esquerda)

Omitiremos a indicação direita/esquerda quando for suficientemente claro.

Definição RELAÇÃO BINÁRIA (X)

Sejam A, B conjuntos. Uma relação R é qualquer subconjunto de (AxB).

(i) $R \subseteq \{(a,b) \mid a \in A \wedge b \in B\} = (A x B)$

(i i) $(a,b) \in R \iff aRb$

Definição RELAÇÕES BINÁRIAS SOBRE CONJUNTOS GRAMATICAIS (XI)

(i) $FIRSTNT \subseteq (N x N)$

$A \; FIRSTNT \; B \iff (A \rightarrow Ba) \in P$

(ii) $LASTNT \subseteq (N x N)$

$A \; LASTNT \; B \iff (A \rightarrow \alpha B) \in P$

(iii) $NT.TERM \subset (N x \Sigma)$

$A \; NT.TERM \; a \iff (B \rightarrow \alpha A a \beta) \in P$

(iv) $TERM.NT \; (\Sigma x N)$

$a \; TERM.NT \; A \iff (B \rightarrow \alpha a A \beta) \in P$

(v) $SYMB \subset (N x N)$

$A \; SYMB \; B \iff (A \rightarrow B) \in P$

(vi) $FIRSTTERM \subseteq (N \times \Sigma)$

$A \; FIRSTTERM \; a \iff (A \rightarrow Ba\alpha) \in P, \quad B \in (N \cup \{\varepsilon\})$

(vii) LASTTERM $\subseteq$ $(N x \Sigma)$

A LASTTERM a $\Leftrightarrow$ $(A \rightarrow \alpha a B) \in P$, $B \in (N \cup \{\varepsilon\})$

Definição OPERAÇÕES SOBRE RELAÇÕES (XII)

Sejam $R \subseteq (AxB)$, $S \subseteq (AxB)$, $T \subseteq (BxC)$, $V \subseteq (AxA)$

(i) SOMA

$(R + S) = \{(a,b) \mid aRb \vee aSb\} \subseteq (AxB)$

(ii) PRODUTO

$(RT) = \{(a,c) \mid aRb \wedge bTc\} \subseteq (AxC)$

(iii) POTÊNCIA

$(V^0) = \{(a,a) \mid a \in A\} \subseteq (AxA)$

$(V^n) = VV^{n-1}$, $\forall n > 0$

(iv) TRANSPOSIÇÃO

$(V^{-1}) = \{(a,b) \mid bVa\} \subseteq (AxA)$

(v) FECHAMENTO TRANSITIVO

$(V^+) = V + V^2 + V^3 + \dots \subseteq (AxA)$

(vi) FECHAMENTO TRANSITIVO REFLEXIVO

$(V^*) = V^0 + V^+ \subseteq (AxA)$

Definição REPRESENTAÇÃO DE RELAÇÕES POR GRAFOS (XIII)

(Baseado em |*Hunt*, 77|)

Seja $R \subseteq (AxB)$ e $G = (V,X)$ um grafo direcionado onde V é o conjunto finito de vértices e X é o conjunto finito de arestas (u, v), com $u, v \in V$.

Se (i) $V = V_1 \cup V_2$ e $V_1 \cap V_2 = \phi$

(ii) $(u, v) \in X$, $u \in V_1$, $v \in V_2 \iff uRv$ (*)

(iii) não houver outras arestas

então G representa R.

(*) *Note que chamamos pelo mesmo nome tanto um elemento do domínio (ou contradomínio) de uma relação quanto o nó a ele correspondente no grafo. A distinção pode ser feita pelo contexto.*

Definição OPERAÇÕES SOBRE RELAÇÕES REPRESENTADAS POR GRAFOS (XIV)

Sejam: $G_1 = (I_1 \cup O_1, X_1)$ representando $R \subseteq (AxB)$

$G_2 = (I_2 \cup O_2, X_2)$ representando $S \subseteq (AxB)$

$G_3 = (I_3 \cup O_3, X_3)$ representando $T \subseteq (BxC)$

$G_4 = (I_4 \cup O_4, X_4)$ representando W C (AxA)

Suporemos ainda que todos os conjuntos de nós I_1, I_2, I_3, I_4, O_1, O_2, O_3 e O_4 são disjuntos entre si.

(i) SOMA

$G = (I \cup O, X)$ representará R + S se:

(a) $I = I_1 = I_2$

(b) $O = O_1 = O_2$

(c) $(u, v) \in X_1 \Rightarrow (u,v) \in X$

(d) $(u, v) \in X_2 \Rightarrow (u,v) \in X$

(e) não há outras arestas

(ii) PRODUTO

$G = (I \cup C \cup O, X)$ representará RT se:

(a) $I = I_1$

(b) $O = O_3$

(c) $C = O_1 = I_3$

(d) $(u,v) \in X_1 \Rightarrow (u, v) \in X, \quad u \in I, \quad v \in C$

(e) $(u, v) \in X_3 \Rightarrow (u, v) \in X, \quad u \in C, \quad v \in O$

(f) não há outras arestas

(iii) TRANSPOSIÇÃO

$G = (I \cup O, X)$ representará W^{-1} se:

(a) $I = O_4$

(b) $O = I_4$

(c) $(u, v) \in X_4 \Rightarrow (v, u) \in X, \quad v \in I, \quad u \in O$

(d) não há outras arestas

(iv) FECHAMENTO TRANSITIVO REFLEXIVO

$G = (I \cup O, X)$ representará W^* se:

(a) $I = O$

(b) $(u, v) \in X_4 \Rightarrow (u, v) \in X$

(v) FECHAMENTO TRANSITIVO

Obtido por $W^+ = WW^*$

———————— ∎

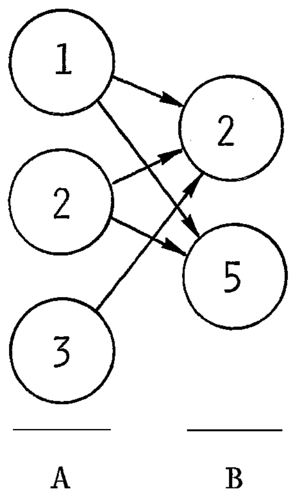
Exemplo

A = {1,2,3} R = {(1,2), (3,2), (2,5)} W = {(1,2),(2,3)}

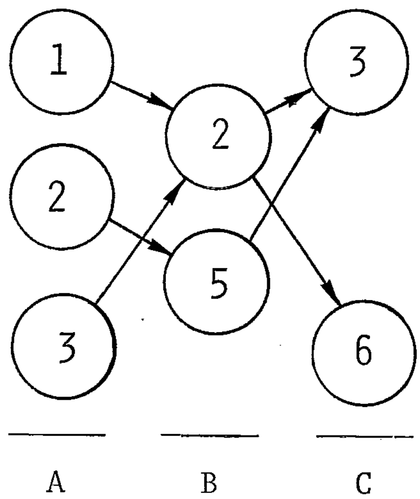
B = {2,5} S = {(2,2), (1,5)}

C = {3,6} T = {(2,3), (5,3), (2,6)}

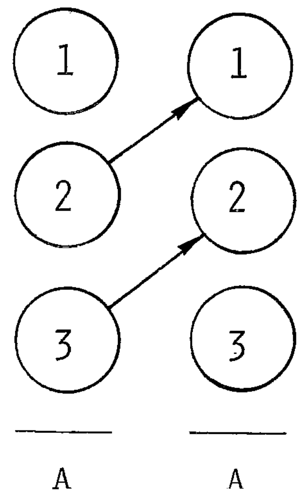
R + S



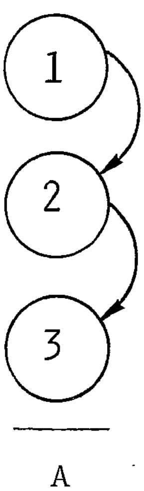
RT



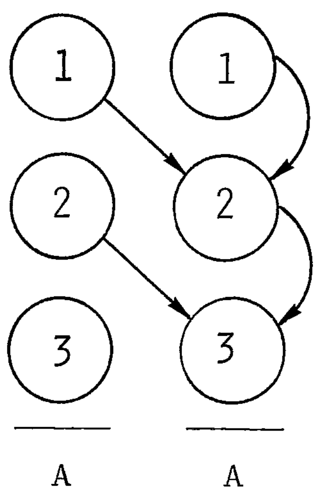
W^{-1}



W*



W^{+}



Notação ALGORÍTMO (XVI)

Os algorítmos constantes deste trabalho serão escritos em português, envolvendo alguma notação matemática e lógica, além da notação definida no próprio texto. Seu funcionamento será demonstrado teoricamente quando necessário.

Definição ANALISADOR SINTÁTICO DA GRAMÁTICA G (XVII)

É um algorítmo que admite a sentença x como entrada e fornece como saída:

(i) "ACEITO", se $x \in L(G)$, fornecendo ainda uma sequência ordenada das produções usadas em $S \overset{*}{\underset{G}{\Longrightarrow}} x$;

(ii) "REJEITADO", se $x \notin L(G)$.

2.2 - GRAMÁTICAS DE PRECEDÊNCIA DE OPERADORES

O presente trabalho guarda íntima relação com o estudo das gramáticas de precedência de operadores, que se justifica sua explicação detalhada aqui.

■ Definição: GRAMÁTICA DE OPERADORES (GO) (XVIII)

Uma gramática reduzida $G = (N, C, P, S)$ é dita gramática de operadores, ou equivalentemente da classe $\mathcal{C}_{GO}$, se

$$(A \rightarrow \alpha) \in P \Rightarrow \alpha \neq \beta BC\gamma$$

Ou seja, não há dois não-terminais adjacentes no lado direito de suas produções.

______ ■

■ Teorema 1 (XIX)

$$G = (N, \Sigma, P, S) \in \mathcal{C}_{GO} \wedge S \overset{*}{\underset{G}{\Longrightarrow}} \alpha \Rightarrow \alpha \neq \beta AB\gamma$$

Ou seja, não há forma sentencial em G com dois não-terminais adjacentes.

Demonstração: |*Floyd*, 63|

______ ■

A principal característica das GO é a possibilidade de se montar analisadores sintáticos baseados apenas em relações sobre os terminais da gramática. A subclasse importante das GO é a classe das gramáticas de precedência de operadores caracteriza da pelas seguintes relações.

Definição RELAÇÕES DE PRECEDÊNCIA ENTRE OPERADORES |*Floyd*, 63| (XX)

(Formalização segundo |*Gries*, 71|)

Seja $G = (N, C, P, S) \in \mathcal{C}_{GO}$.

(i) relação $\gtrdot \subseteq (\Sigma \times \Sigma)$ (maior operadores)

$$a \underset{G}{\gtrdot} b \iff (A \to \alpha B b \beta) \in P$$
$$\wedge\ (C \to \gamma a D) \in P \qquad , D \in (N \cup \{\varepsilon\})$$
$$\wedge\ B \overset{*}{\underset{G}{\Longrightarrow}} \delta C$$

(ii) relação $\lessdot \subseteq (C \times C)$ (menor operadores)

$$a \underset{G}{\lessdot} b \iff (A \to \alpha a B \beta) \in P$$
$$\wedge\ (C \to D b y) \in P \qquad , D \in (N \cup \{\varepsilon\})$$
$$\wedge\ B \overset{*}{\underset{G}{\Longrightarrow}} C\delta$$

(iii) relação $\doteq \subseteq (\Sigma \times \Sigma)$ (igual operadores)

$$a \underset{G}{\doteq} b \iff (A \to \alpha a D b \beta) \in P \qquad , D \in (N \cup \{\varepsilon\})$$

Pode-se deduzir com facilidade |*Gries*, 71| que, conforme as definições de (X):

(iv) $a \underset{G}{} b \iff a\ (\text{LASTTERM})^{-1}\ (\text{LASTNT}^*)^{-1}\ \text{NT.TERM}\ b$

(v) $a \underset{G}{\lessdot} b \iff a\ \text{TERM.NT}\ \text{FIRSTNT}^*\ \text{FIRSTTERM}\ b$

(vi) $a \underset{G}{\doteq} b$ pode ser construída por inspeção na gramática.

A referência à gramática pode ser omitida na notação.

Definição GRAMÁTICAS DE PRECEDÊNCIA DE OPERADORES (GPO) (XXI)

Uma gramática unicamente inversível $G=(N, \Sigma, P, S) \in \mathcal{C}_{GO}$ será dita de precedência de operadores, ou equivalentemente, pertencente à classe $\mathcal{C}_{GPO}$, se:

(i) $\lessdot \cap \doteq = \phi$

$\wedge$ (ii) $\lessdot \cap \gtrdot = \phi$

$\wedge$ (iii) $\doteq \cap \gtrdot = \phi$

Vamos nos aproveitar desta classe de gramáticas para introduzir os conceitos e exemplificar o problema da análise sintática, conforme o trataremos posteriormente.

Interessa-nos obter do analisador sintático a árvore sintática tal como definida abaixo.

Definição ARVORE SINTÁTICA (XXII)

Seja $G = (N, \Sigma, P, S)$ e x uma sentença em $L(G)$

A árvore sintática de x em G será uma árvore construída da forma abaixo, supondo-se conhecida uma derivação $S \overset{*}{\underset{G}{\Longrightarrow}} x$:

(i) a raiz terá rótulo S;

(ii) a cada passo da derivação, para o nó cujo rótulo é o lado esquerdo da produção usada crie tantos filhos rotulados quantos forem os símbolos do lado direito dessa produção.

As folhas da árvore sintática serão terminais e os nós internos serão não-terminais.

_________ ∎

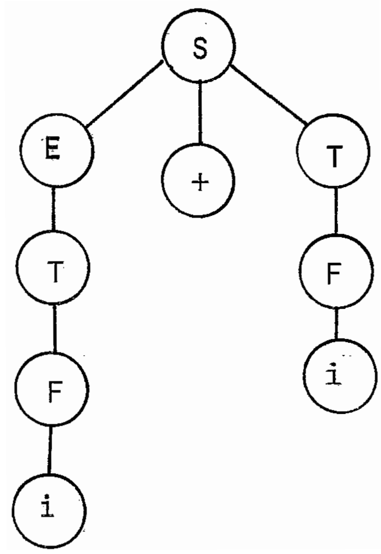
∎ Exemplo (XXIII)

Seja $G_0 = (\{E, T, F\}, \{+, *, (,), i\}, P, E)$ com $P = \{(E \to E + T), (E \to T), (T \to T * F), (T \to F), (F \to (E)), (F \to i)\}$.

Seja a derivação

$$S \underset{G_0}{\Longrightarrow} E+T \Longrightarrow T+T \Longrightarrow F+T \Longrightarrow i+T \Longrightarrow i+F \Longrightarrow i+i$$

A árvore sintática correspondente seria:



Se tomarmos outra derivação

$$S \underset{G_0}{\Longrightarrow} E + T \Longrightarrow E + F \Longrightarrow E + i \Longrightarrow T + i \Longrightarrow F + i \Longrightarrow i+i$$

ainda teremos, neste caso, a mesma árvore sintática.

<u>Definição</u> GRAMÁTICA AMBÍGUA (XXIV)

Uma gramática $G = (N, \Sigma, P, S)$ é dita ambígua, se:

(i) para $x \in L(G)$ existir mais de uma árvore sintática

Esta condição é equivalente a:

(ii) para $x \in L(G)$ existir mais de uma derivação direita (esquerda) de x em G.

Definição PARSE (XXV)

Seja $G = (N, \Sigma, P, S)$ com as produções de P numeradas 1,2, ..., p e uma derivação $S \overset{*}{\underset{G}{\Longrightarrow}} x$.

A sequência dos números das produções utilizadas na derivação é o parse de x em G.

Mais particularmente:

(i) o parse descendente de x em G é a sequência dos números das produções utilizados na derivação $S \overset{*}{\underset{ESQ}{\Longrightarrow}} x$;

(ii) o parse ascendente de x em G é a sequência dos números das produções utilizadas na redução $S \overset{*}{\underset{DIR}{\Longleftarrow}} x$;

Há correspondência biunívoca entre parse ascendente, parse descendente e árvore sintática.

Em todo o decorrer deste trabalho usaremos o parse como a sequência ordenada de produções prevista em XVII. (ii), a saída dos analisadores sintáticos. Como trataremos apenas analisadores ascendentes toda referência futura a parse significará parse ascendente.

Os métodos de análise que estudaremos não permitem a obtenção - ou, caso permitam, será ignorada - do parse da sentença. Veremos neste capítulo a definição de parse "*estrutural*" e no capítulo seguinte a definição de parse "*esparso*".

Para as gramáticas de precedência de operadores não se consegue obter analisadores sintáticos , tal como definidos em XVII, mas apenas analisadores sintáticos de sua gramática estrutural ("skeletal" em |*Aho & Ullman*, 72|). Esta gramática é obtida da gramática original pela substituição de todas as ocorrências de não-terminais pelo símbolo inicial. Isto é, em geral, suficiente para a construção de compiladores, sendo uma preocupação reduzir o tempo gasto com produções simples - do tipo $A \rightarrow B$ com $A, B \in N$ - que normalmente não carregam informação semântica.

Esta discussão sobre a necessidade de se eliminar produções simples tem gerado um sem número de "otimizadores" de analisadores sintáticos |*Anderson*, 73|, |*Aho & Ullman*, 73|, |*Pager*, 74|, |*Demers*, 75|, |*Backhouse*, 76|, |*LaLonde*, 76|, |*Soisalon - Soininen*,77|, |*Rushby*,77|, cada qual propondo métodos distintos de aperfeiçoar a análise sintática das classes de gramáticas, em particular a família LR, que produzem parses completos. Tanta efervecência no assunto deve-se ao alto lucro obtido pela análise sintática sem produções simples: |*Joliat*, 73| mostrou que sua eliminação eficiente dobra a velocidade da análise sintática e aumenta a velocidade do compilador, como um todo, em 15%.

Outra alternativa para o mesmo problema são os métodos recentes de análise sintática em gramáticas com lados direitos regulares |*De* Rema, 70|, |*LaLonde*, 75|, |*Simone*, 80| , que partem para a eliminação da causa primária das produções simples: a inadequação do mecanismo descritivo composto pelas gramáticas livres de contexto, para descrever as próprias linguagens livres de contexto.

De modo que estaremos tratando de analisadores sintáticos que, por natureza, gozam da propriedade de ignorar produções simples.

Vejamos, então, como opera o analisador sintático de gramáticas de precedência de operadores, descrito pelo algorítmo 1, a seguir.

Este algorítmo obtem o parse estrutural da sentença, sendo controlado por duas funções:

(i) $f \; : \; (C \times C) \longrightarrow \{\lessdot, \doteq, *, \text{"erro"})$

definido por:

$f(a, b) = \gtrdot \iff a \gtrdot b$

$f(a, b) = \doteq \qquad a \doteq b$

$f(a,b) = \lessdot \qquad a \lessdot b$

$f(a,b) =$ "erro", qualquer outro caso.

(ii) $g \; : \; C \cup \{S\})^{+} \longrightarrow \{1, 2, \ldots, p, \text{"stop"}\}$

definida por:

$g(\alpha) = i \qquad (i{:}S \to \alpha) \in P$

$g(a) =$ "stop", qualquer outro caso.

O algorítmo é uma versão do clássico algorítmo *"avança-reduz"* para análise sintática ascendente. A função f determina as ações de avançar um símbolo da entrada ou reduzir, indicando também a frase a ser reduzida. A função g indica o número da produção usada.

Algorítmo 1 (XXVI)

ANALISADOR SINTÁTICO PARA GRAMÁTICA ESTRUTURAL DE GPO

Passo 1: (Inicialização)

Coloque na pilha o marcador "#" ;

Faça a variável MEIO = " " ;

Leia um símbolo da sentença para ENTRADA;

Repita o passo 2 até receber ordem de parada.

Passo 2: (Escolha da ação)

Caso f(topo da pilha, ENTRADA) igual a:

$\lessdot$ ou $\doteq$: execute o passo 3;

$\gtrdot$: execute o passo 4;

"erro" : execute o passo 5;

Passo 3: (Avanço)

Se MEIO = "S" então coloque S na pilha e faça MEIO=" ";

Coloque ENTRADA na pilha;

Leia novo símbolo para ENTRADA;

Passo 4: (Redução)

Retire da pilha todos os símbolos até encontrar dois símbolos cuja relação entre sí seja $\lessdot$;

Concatene tais símbolos retirados, em ordem;

Se MEIO = "S", concatene também "S", à direita;

Seja SEQ o produto da concatenação;

Se G(SEQ) $\neq$ "stop" escreva G(SEQ) e faça MEIO = "S" senão: verifique se o topo da pilha é "#", se MEIO = "S" e ENTRADA = "#" ;

Caso as três condições ocorram escreva "ACEITO" e pare;

Caso contrário escreva "REJEITADO" e pare;

Passo 5: (Erro)

Escreva "REJEITADO" e pare;

___________ ■

O algorítmo se baseia na seguinte propriedade das gramáticas de precedência e é válida também para as GPO, na forma abaixo:

"Seja $X_1\ X_2\ \dots\ X_m$ uma forma sentencial e $a_1 a_2 \dots a_n$ os terminais dessa forma sentencial. A sequência a ser reduzida é a sequência mais à esquerda tal que

$$a_g \lessdot a_{g+1} \doteq a_{g+2} \doteq \dots \doteq a_k \gtrdot a_{k+1} \qquad "$$

A demonstração desta propriedade pode ser vista em |*Gries*, 71| .

O sinal " # " é usado como inicialização da pilha e final da sentença de entrada. Faz parte do conjunto de terminais da gramática aumentada, obtida a partir de G = (N, C, P, S) com $G_A = (N \cup \{S'\}, \Sigma \cup \{\#\}, P \cup \{(S' \rightarrow \# S \#)\}, S')$.

O algorítmo nada mais faz que varrer a sequência de entrada da esquerda para a direita, efetuando uma redução sempre que a situação acima ocorrer. As funções f e g são calculadas sobre a gramática estrutural de G_A.

Exemplo (XXVII)

(i) $G_0 = (\{E, T, F\}, \{+, *, (,), i\}, P, E)$

$P = \{(E \to E + T), (T \to T * F), (F \to (E)), (E \to T),$
$(T \to F), (F \to i)\}$

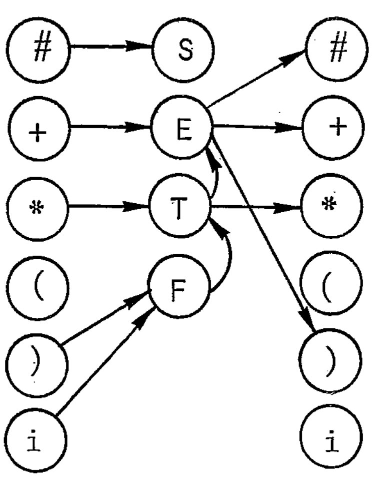
(ii) $G_{0A} = (\{S, E, T, F\}, \{\#, +, *, (,), i\}, P', S)$

$P' = \{(S \to \# E \#), (T \to T * F), (F \to (E)), (E \to T),$
$(E \to E + T), (T \to F), (F \to i)$

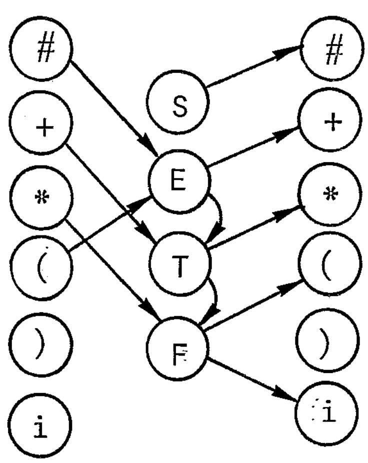
(iii) $G_{ESTR} = (\{S\}, \{\#, +, *, (,), i\}, P'', S)$

$P'' = \{$ 1 : $S \to \#S\#$
2 : $S \to S + S$
3 : $S \to S * S$
4 : $S \to (S)$
5 : $S \to i\ \}$

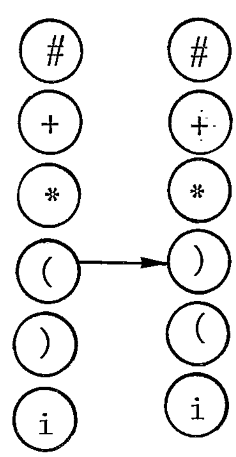
(iv) Cálculo das funções f e g



$\gtrdot_0$



$\lessdot_0$



$\overset{0}{=}$

F	#	+	*	(	)	i
#	?	$\lessdot$	$\lessdot$	$\lessdot$	?	$\lessdot$
+	$\gtrdot$	$\gtrdot$	$\lessdot$	$\lessdot$	$\gtrdot$	$\lessdot$
*	$\gtrdot$	$\gtrdot$	$\gtrdot$	$\lessdot$	$\gtrdot$	$\lessdot$
(	?	$\lessdot$	$\lessdot$	$\lessdot$	$\doteq$	$\lessdot$
)	$\gtrdot$	$\gtrdot$	$\gtrdot$	?	$\gtrdot$	?
i	$\gtrdot$	$\gtrdot$	$\gtrdot$	?	$\gtrdot$	?

? = "erro"

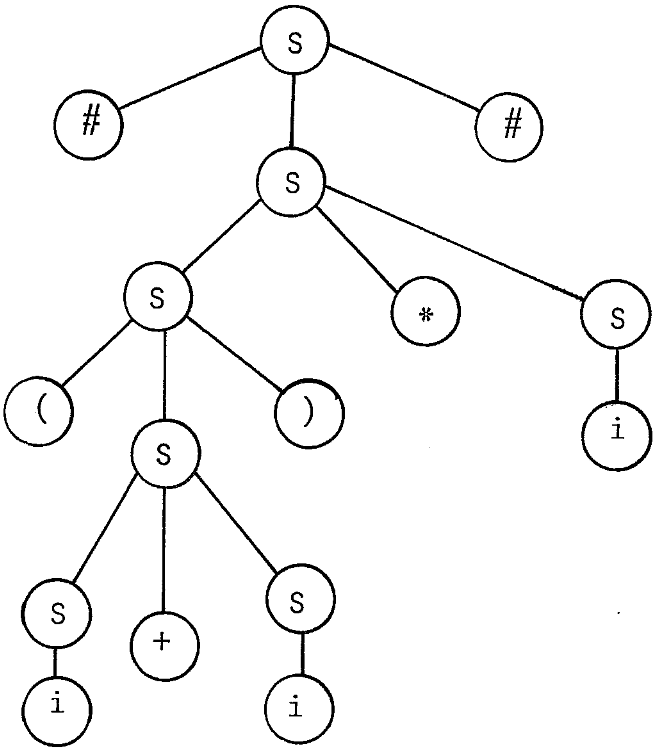
G	
1	#S#
2	S+S
3	S*S
4	(S)
5	i
"stop"	qq.outro

(v) Execução

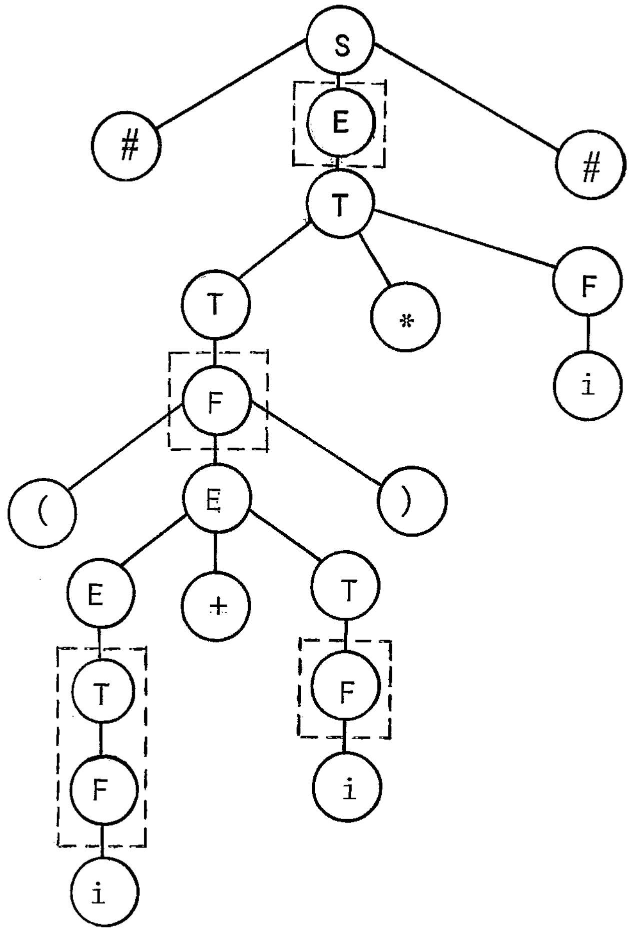
PILHA	MEIO	ENTRADA	PARSE ESTRUTURAL
#		(i+i)*i#	
# (		i+i)* i #	
# (i		+i) * i #	5
# (	S	+i) * i #	
# (S+		i) * i #	
# (S+i		) * i#	5
# (S+	S	) * i #	2
# (	S	) * i #	
# (S)		* i #	4
#	S	* i #	
# S*		i #	
# S*i		#	5
# S*	S	#	3
# S		#	ACEITO

(vi) Árvores Sintáticas

ESTRUTURAL



ORIGINAL



CAPÍTULO III

— MATRIZES DE TRANSIÇÃO —

Propriamente, começaremos aqui. Este capítulo será a revisão do conhecido sobre esta sub-classe das gramáticas livres de contexto. |*Gries, 68*| propôs a denominação *"transition matrix"* para a técnica de análise sintática que então formalizou. Essa *"matriz de transição"* guarda semelhanças com a *"tabela de transições"* de um automato finito. Tal semelhança é fatal à nomenclatura, Mais importante, porém, que o título da ferramenta é a definição dessa classe de gramáticas, mais ampla que a classe de precedência de operadores, baseada de certa forma em conceitos análogos. Como a classe de gramática de operadores, e sua nomenclatura, já estão também estabelecidas contentamo-nos em manter o título inadequado: matrizes de transição.

O fundamental deste capítulo foi extraído de |*Samelson & Bauer, 60*|, |*Conway, 63*| e, principalmente, |*Gries, 68*|. As demonstrações devidas são ali encontradas.

3.1 - GRAMÁTICA DE OPERADORES ESTENDIDA

O objetivo desta primeira modificação na gramática dada é reduzir o comprimento dos lados direitos das produções a, no máximo, três símbolos.

Note que a transformação descrita a seguir só é possí vel se partirmos de uma gramática de operadores.

O processo se dará pela adição à gramática de novos não-terminais, denominados – outra vez com infelicidade – por |*Gries, 68*| de *"não-terminais estrelados"*, e de novas produções intermediárias, denominadas *"produções estreladas"*. A origem do nome vem do uso de um "*" para identificá-los. Preferimos, com maior clareza, sublinhá-los.

■ Definição (XXVIII)

GRAMÁTICA DE OPERADORES ESTENDIDA (GOE)

Seja $G = (N, C, P, S) \in \mathcal{C}_{GO}$. A gramática $G' = (N', C', P', S') = (N \cup \underline{N} \cup \{S'\}, \Sigma \cup \{\#\}, P', S')$ será sua gramática de operadores estendida se G' for construída conforme o algorítmo 2, a seguir.

■

■ Algorítmo 2 (XXIX)

CONSTRUÇÃO DA GRAMÁTICA DE OPERADORES ESTENDIDA

Passo 1 : Construa $G' = (N', C', P', S')$ de modo que $G' = (N \cup \{S'\}, C \cup \{\#\}, P \cup \{S' \to \# S \#\}, S')$ com $S' \notin N$ e $\# \notin C$. Chamaremos esta gramática obtida no Passo 1 de gramática de operadores aumentada

Passo 2 : Se $(A \to a\alpha) \in P'$, $\alpha \in (N' \cup \Sigma')^*$
então
(i) acrescente a produção $\underline{V} \to a$, se já não existir

alguma produção com lado direito a; caso exista, seja $\underline{V}$ o seu lado esquerdo;

(ii) substitua cada produção da forma
$(B \to a\beta) \in P'$ por $(B \to \underline{V}\beta)$;

Passo 3 : Se $(A \to Ba\alpha) \in P'$, $\alpha \in (N' \cup \Sigma')^*$
então

(i) acrescente a produção $\underline{V} \to Ba$, se já não existir alguma produção com lado direito Ba; caso exista, seja $\underline{V}$ o seu lado esquerdo;

(ii) substitua cada produção da forma
$(C \to Ba\beta) \in P'$ por $(C \to \underline{V}\beta)$;

Passo 4 : Se $(A \to \underline{U}a\alpha) \in P'$, $\alpha \in (N' \cup \Sigma')^*$
então

(i) acrescente a produção $\underline{V} \to \underline{U}a$, se já não existir alguma produção com lado direito $\underline{U}a$; caso exista seja $\underline{V}$ seu lado esquerdo;

(ii) substitua cada produção da forma
$(C \to Ua\beta) \in P'$ por $(C \to \underline{V}\beta)$;

Passo 5 : Se $(A \to UBa\alpha) \in P'$, $\alpha \in (N' \cup \Sigma')^*$
então

(i) acrescente a produção $\underline{V} \to \underline{U}Ba$, se já não existir alguma produção com lado direito $\underline{U}Ba$; caso exista seja $\underline{V}$ seu lado esquerdo;

(ii) substitua cada produção da forma
$(C \to \underline{U}Ba\beta) \in P'$ por $(C \to \underline{V}\beta)$;

▮

Note que, a cada passo, $\underline{V}$ é um novo não-terminal criado quando uma nova produção for acrescentada e $\underline{N}$ é o conjunto desses não-terminais estrelados. Vamos convencionar que a numeração das produções originais seja (1, 2, ..., p), que o Passo 1 crie uma nova produção numerada 0 e que as demais produções acrescidas sejam numeradas (p + 1, p + 2, ...). Note que as substituições de produções não alteram sua numeração.

■ <u>Exemplo</u> (XXX)

$G = (N , \Sigma, P, S) \in \mathcal{C}_{GO}$

$N = \{S, A, B, C, D\}$ $C = \{a, b, c, d\}$

$P = \{1 : S \to aAb\ , 2 : A \to bcB\ , 3 : B \to C,\ 4 : C \to Dd\ ,$
$5 : D \to d\}$

Após o Passo 1:

$N' = \{S', S, A, B, C, D\}$ $C' = \{\#, a, b, c, d\}$

$P' = \{0 : S' \to \#S\#\ ,\ 1 : S \to aAb\ , 2 : A \to bcB,\ 3 : B \to C\ ,$
$4 : C \to Dd\ , 5 : D \to d\ \}$

Executando-se os passos seguintes:

$0 : S' \to \#S\#$		$0 : S' \to \underline{\#}\ S\#$		$0 : S' \to \underline{\#}\ S\#$
$1 : S \to aAb$		$1 : S \to \underline{a}\ Ab$		$1 : S \to \underline{a}\ Ab$
$2 : A \to bcB$	$\xrightarrow{P_2}$	$2 : A \to \underline{b}\ cB$	$\xrightarrow{P_3}$	$2 : A \to \underline{b}\ cB$
$3 : B \to C$		$3 : B \to C$		$3 : B \to C$
$4 : C \to Dd$		$4 : C \to Dd$		$4 : C \to \underline{Dd}$
$5 : D \to d$		$5 : D \to \underline{d}$		$5 : D \to \underline{d}$
		$6 : \underline{\#} \to \#$		$6 : \underline{\#} \to \#$

$7 : \underline{a} \rightarrow a$

$8 : \underline{b} \rightarrow b$

$9 : \underline{d} \rightarrow d$

$7 : \underline{a} \rightarrow a$

$8 : \underline{b} \rightarrow b$

$9 : \underline{d} \rightarrow d$

$10 : \underline{Dd} \rightarrow Dd$

$0 : S' \rightarrow \underline{\#}\, S\#$

$1 : S \rightarrow \underline{a}\, Ab$

$2 : A \rightarrow \underline{bc}\, B$

$3 : B \rightarrow C$

$4 : C \rightarrow \underline{Dd}$

$5 : D \rightarrow \underline{d}$

$6 : \underline{\#} \rightarrow \#$

$7 : \underline{a} \rightarrow a$

$8 : \underline{b} \rightarrow b$

$9 : \underline{d} \rightarrow d$

$10 : \underline{Dd} \rightarrow Dd$

$11 : \underline{bc} \rightarrow \underline{b}\, c$

$\overset{P_4}{\Longrightarrow}$ $\overset{P_5}{\Longrightarrow}$

$0 : S' \rightarrow \underline{\#S\#}$

$1 : S \rightarrow \underline{aAb}$

$2 : A \rightarrow \underline{bc}\, B$

$3 : B \rightarrow C$

$4 : C \rightarrow \underline{Dd}$

$5 : D \rightarrow \underline{d}$

$6 : \underline{\#} \rightarrow \#$

$7 : \underline{a} \rightarrow a$

$8 : \underline{b} \rightarrow b$

$9 : \underline{d} \rightarrow d$

$10 : \underline{Dd} \rightarrow Dd$

$11 : \underline{bc} \rightarrow \underline{b}\, c$

$12 : \underline{\#S\#} \rightarrow \underline{\#}\, S\#$

$13 : \underline{aAb} \rightarrow \underline{a}\, Ab$

Propriedades das GOE (XXXI)

(i) após o passo 2 as produções são das formas

$(A \rightarrow B)$, $(A \rightarrow Bca)$, $(A \rightarrow Ua)$, $(U \rightarrow a)$

(ii) após o passo 3 as produções são das formas

$(A \rightarrow B)$, $(A \rightarrow \underline{U}\alpha)$, $(\underline{U} \rightarrow a)$, $(U \rightarrow Ba)$

(iii) após o passo 4 as produções são das formas

$(A \rightarrow B)$, $(A \rightarrow \underline{U}Ba\alpha)$, $(U \rightarrow a)$, $(U \rightarrow Ba)$

$(U \rightarrow \underline{U_1}a)$, $(A \rightarrow \underline{U})$

(iv) em qualquer GOE temos apenas 7 formas de produção:

$(A \to B)$, $(A \to \underline{U})$, $(A \to \underline{U}B)$

$(\underline{U} \to a)$, $(\underline{U} \to Ba)$, $(\underline{U} \to \underline{U_1}a)$, $(\underline{U} \to \underline{U_1}Ba)$

(v) cada não-terminal estrelado aparece como lado esquerdo de uma Única produção; o lado direito correspondente aparece como lado direito apenas dessa produção.

(vi) se os novos não-terminais estrelados forem numerados conforme a ordem de aparecimento no algorítmo, então

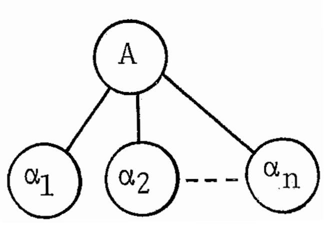
$(\underline{U_i} \to \underline{U_j}\alpha) \in P' \Rightarrow i > j$

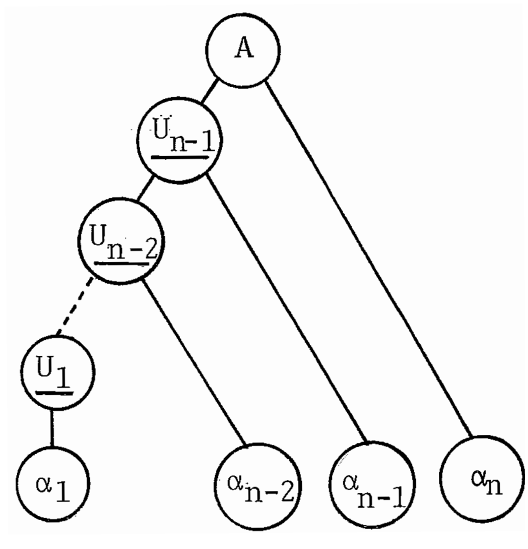
(vii) para cada produção $(A \to \alpha)$ na gramática original existe um Único conjunto de produções

$(\underline{U_1} \to \alpha_1)$, $(\underline{U_2} \to \underline{U_1}\alpha_2)$, ... , $(\underline{U_n} \to \underline{U_{n-1}}\,\alpha_n)$ tal que

$\alpha = \alpha_1\alpha_2 \ldots \alpha_n$

(viii) para cada árvore sintática de uma forma sentencial y na gramática de operadores pode-se construir uma Única árvore sintática da sentença #y# na gramática de operadores estendida, substituindo-se o ramo $(A \to \alpha)$ pelas produções descritas no ítem anterior





∎ Teorema 2 (XXXII)

Se uma GOE é não-ambígua então a GO da qual deriva é também não-ambígua.

Demonstração: decorrência das propriedades XXXI (iv), (v), (vi), (vii) e (viii) e devida a |*Gries, 68*|.

∎

3.2 – ANÁLISE SINTÁTICA EM GOE

Iremos descrever a seguir o método de construção de analisadores sintáticos para GOE, baseado em relações de precedência, sendo portanto um reconhecedor ascendente que lê a sentença da esquerda para a direita. Pelo fato de propormos um analisador determinístico estaremos também definindo condições suficientes para que uma GOE seja não-ambígua.

Uma característica essencial deste método é que se baseará em relações de precedência entre não-terminais estrelados e terminais, não sendo capaz de detectar uma redução quando a produção usada for simples. Portanto, fornecerá um parse esparso.

As definições seguintes de "*frase*" e "*frase prima*" pretendem substituir a definição usual de "*handle*" em analisadores ascendentes e contornam o problema teórico da formalização do parse de modo mais sofisticado que a gramática estrutural vista no capítulo 2 para as gramáticas de precedência de operadores.

Definição FRASE (XXXIII)

Diremos que $\gamma \in (N' \cup \Sigma')^*$ é uma frase se $S \overset{*}{\Longrightarrow} \beta A \delta \overset{*}{\Longrightarrow} \beta \gamma \delta$ e $A \overset{*}{\Longrightarrow} \gamma$

Definição FRASE PRIMA (XXXIV)

Diremos que $\gamma \in (N' \cup \Sigma')^*$ é uma frase prima se

(i) γ é frase;

(ii) $\gamma = \alpha X \beta$, $X \in (\underline{N} \cup C)$;

(iii) $\gamma = \alpha_1 \alpha_2 \cdots \alpha_n \Rightarrow$ α_i não é frase, qq. i

---

A definição é feita de tal forma que as frases primas serão o objeto das reduções, restando solucionar as reduções advindas das produções simples.

Teorema 3 (XXXV)

As frases primas em GOE são de 6 formas:

$\underline{U}$, $\underline{U}a$, a, $\underline{U}A$, Aa, $\underline{U}Aa$.

Demonstração : consequência da propriedade XXXI.(iv) e da definição de frase prima.

Vamos agora definir uma classe de gramáticas - subcon junto das GOE – para as quais será sempre possível obter um a nalisador sintático ascendente determinístico, do tipo que nos interessa.

■ Definição RELAÇÃO PRECEDE (XXXVI)

$$\text{PRECEDE} \subseteq (N' \cup C') \times (N' \cup C')$$
$$X \text{ PRECEDE } Y \iff S' \overset{*}{\Longrightarrow} \alpha X Y \beta$$

■

■ Definição (XXXVII)

GRAMÁTICAS DE MATRIZES DE TRANSIÇÃO (GMT)

Uma gramática $G = (N', \Sigma', P', s') \in \mathcal{C}_{GOE}$ será dita gramática de matriz de transição (GMT) ou, equivalentemente, $G \in \mathcal{C}_{GMT}$, se:

(i) para qualquer par $(A,B) \in (N \cup \{S'\})^2$ se $A \overset{*}{\Longrightarrow} B$, então essa derivação é Única;

(ii) para cada par $(U, a) \in (N \times C')$ no máximo uma das três seguintes condições é verdadeira:

(a)' $(A \to U) \in P' \wedge (A \text{ PRECEDE } a) \wedge (A \text{ Único})$;

(b) $(\underline{V} \to Ua) \in P'$;

(c) $(V \to a) \in P' \wedge (U \text{ PRECEDE } \underline{V})$;

(iii) para cada tripla $(U, A, a) \in (\underline{N} \times N \times C')$ no máximo uma das três seguintes condições é verdadeira:

(a) $(B \to Uc) \in P' \wedge (B \text{ PRECEDE } a)$
$\wedge (C \overset{*}{\Longrightarrow} A) \wedge (B, C \text{ Únicos})$;

(b) $(V \to \underline{U}\ Ca) \in P'$ $\wedge (C \overset{*}{\Longrightarrow} A)$ $\wedge$ (C Único) ;

(c) $(\underline{V} \to Ca) \in P'$ $\wedge (\underline{U}$ PRECEDE $\underline{V})$

$\wedge (C \overset{*}{\Longrightarrow} A)$ $\wedge$ (C único) ;

São consequências da definição:

(i) cada condição imposta corresponde a uma das seis possíveis formas de frase prima em GOE;

(ii) a GOE (e, em consequência de XXXI. (v) também a gramática de operadores original) não necessita ser unicamente inversível admitindo a existência de duas produções $(A \to B)$ e $(C \to B)$, desde que respeitada a condição de derivação Única. Para as produções não simples é necessário que a gramática seja UI, em consequência das condições (i).a e (ii.a) da definição de GMT.

Teorema 4 (XXXVIII)

$G \in \mathcal{C}_{GMT} \Rightarrow$ G é não-ambígua.

Demonstração: |*Gries, 68*|.

Algorítmo 3 (XXXIX)

CONSTRUTOR DE ANALISADOR SINTÁTICO PARA GMT

Passo 1 : Verifique se $A \overset{*}{\Longrightarrow} B$ então a derivação é Única;

Passo 2 : Construa a GOE através do Alg.2;

Passo 3 : Construa a relação PRECEDE $\subset (N \cup \Sigma) \times (\Sigma \cup C')$;

Passo 4 : Para cada par (U, a) verifique se apenas uma das condições (i).a, (i).b e (i).c prevalece.
Assim sendo construa a quádrupla
(U, a, X, j)
onde X é o lado esquerdo da produção usada
j é o número dessa produção;

Passo 5 : Para cada tripla (U, A, a) verifique se apenas uma das condições (ii).a, (ii).b e (ii).c ocorre.
Assim sendo construa a 5-tupla
(U, A, a, X, j)

Passo 6 : Se mais de uma condição ocorrer nos passos 4 ou 5 ou se não for verificada a condição do passo 1 indique que $G \notin \mathcal{C}_{GMT}$.

■

É importante que se faça uma verificação sumária da complexidade do Alg. 3, visando sua comparação com o algorítmo a ser proposto no capitulo seguinte:

(i) o Passo 1, conforme veremos no próximo capítulo, é redutível ao problema de determinação da existência de dois caminhos distintos entre um par de nós de um digrafo esparso. Esse algorítmo é de $O(n)$, onde n é o número de não-terminais envolvidos em produções simples;

(ii) o Passo 2 é o Alg.2 para a construção da GOE. Esse algorítmo é de complexidade $O(pn)$, onde p é o número de produções e n é a soma dos comprimentos dos lados direitos. Como

$P \leqslant n$, temos complexidade $O(n^2)$.

(iii) o Passo 3 constroi a relação PRECEDE. Esta relação pode ser expressa como PRECEDE $= \lessdot + \gtrdot + \doteq$, onde estas são as relações de |*Wirth & Weber*, 66|. Conforme |*Hunt*, 77|, que detalharemos no próximo capítulo, é calculável por algorítmo de complexidade $O(n^2)$, onde n é a soma dos números de terminais e não-terminais;

(iv) o Passo 4 percorre todas as produções para cada par $(\underline{U}, a) \in (\underline{N} \times C')$, tornando o algorítmo de complexidade $O(pmn)$, com $p = \#(P')$, $m = \#(\underline{N})$, $n = \#(\Sigma')$. Como $p \leq C_1 n$ e $m \leq C_2 n$, para constantes C_1 e C_2, teremos complexidade $O(n^3)$;

(v) o Passo 5, por analogia ao Passo 4, terá complexidade $O(pmnr)$, com $p = \#(P')$, $m = \#(\underline{N})$, $n = \#(C')$, $r = \#(N)$. Por conseguinte $O(n^4)$;

(vi) em conclusão, temos um algorítmo $O(n^4)$.

Resta-nos apresentar o analisador sintático para GMT. Alguns problemas ainda não estão resolvidos, tais como as reduções da forma $A \overset{*}{\Leftarrow} B$ e o armazenamento das triplas obtidas pelo construtor. Seguiremos precisamente a solução de |*Gries*, *68*| que advoga a tradução das triplas em matriz de controle (a "*matriz de transição*") acoplada a um conjunto de subprogramas.

A geração da matriz e dos subprogramas é algorítmica mas não será necessário descrevê-la formalmente aqui bastando lembrar:

(i) como a matriz será bidimensional $(\underline{N} \times C)$, todas as tuplas que contiverem o par $(\underline{U}, a)$ serão tratadas pelo mesmo