Universidade Estadual de Santa Cruz – UESC

Relatório de Implementação da conversão de Árvore Abstrata para Notação Polonesa Normal e Reversa

Docente César Alberto Bravo Pariente

Discente Matheus Miranda Brandão

Matrícula 201820065

Disciplina Compiladores.

Curso Ciência da Computação

Semestre 2022.2

Ilhéus – BA
2022

Índice

GLC LL

O projeto consiste na implementação em C de um algoritmo que a partir de uma gramática livre de contexto faça a análise sintática utilizando o método de tabela parsing, gere a árvore compacta dessas produções, converta a árvore sintática compacta para abstrata e ao fim gere a notação polonesa normal e reversa.

O projeto foi dividido em: Automação de tabela parsing; Automação da árvore compacta; Árvore Abstrata; Notação polonesa. O analisador implementado ignora o token ' '.

As etapas de automação estão presentes no mesmo arquivo "parsing.c" que recebe como entrada a gramática a ser analisada e a palavra, caso a palavra não atenda aos critérios da gramática, retornará um aviso sobre o erro e onde ocorreu, sua tabela e produções até o momento, caso a palavra seja aceita retornará sua tabela parsing, suas produções, sua árvore sintática compactada e um arquivo txt contendo a palavra, o tamanho da árvore compacta e a árvore no formato (hash, token). Por fim cria um arquivo "ada_tree" com a árvore sintática de saída.

As etapas faltantes compõem o arquivo "proj3_b.c", que a partir da entrada do módulo anterior faz a conversão da árvore sintática para árvore abstrata utilizando o método de Backtracking, removendo todo o conteúdo sem valor semântico, ao fim percorre a árvore abstrata (Asa) em pré-ordem para gerar a notação polonesa normal e em pós-ordem para gerar a notação polonesa reversa.

Produções da gramática:

p_1: S -> M | GM | NGM

p_4: N -> n(){ C; r(E); }

p_5: G -> g(){ C; r(E); }

p_6: M -> m() { C; r(E); }

p_7: E -> 0 | 1 | x | y | (EXE)

p_{12}: X -> + | - | * | /

p_{16}: C -> h=E | i=E | j=E | k=E | z=E | (EXE) | w(E){ C; } | f(E){ C; } | o(E;E;E){ C; }

Compilando e Executando

Para a execução não é necessário o uso de nenhuma dependência, basta compila-los normalmente.

$ gcc parsing.c -o parsing

$ gcc proj3_b.c -o proj3_b

Ao executar o parse_generator é necessário passar como argumentos o arquivo contendo as palavras, caso contrário resultará em erro. Para executar diversas palavras em um arquivo deve-se separá-los por uma quebra de linha ('\n').

Exemplo:

$ ./parsing examples/inputs

Ao executar o proj3_b é necessário passar como argumentos o arquivo contendo a saída do parsing: "ada_tree".

Exemplo:

$ ./proj3_b ada_tree

Gramática

Representação para automação:

SM	X-
SGM	X*
SNGM	X/
Nn(){C;r(E);}	Ch=E
Gg(){C;r(E);}	Ci=E
Mm(){C;r(E);}	Cj=E
E0	Ck=E
E1	Cz=E
Ex	C(EXE)
Ey	Cw(E){C;}
E(EXE)	Cf(E){C;}
X+	Co(E;E;E){C;}

Caso a entrada dada seja incorreta o analisador sintático irá imprimir os tokens até o momento, então avisará sobre o erro, informará qual o token inesperado, então pulará para a próxima palavra. Como outputs temos as produções, o tamanho da árvore n-ária, a palavra analisada, as produções e a árvore sintática (no formato '[index | mapeamento | token]') e um arquivo para utilizar na criação da árvore abstrata.

Na árvore abstrata temos como outputs a palavra e sua árvore abstrata (no formato '[index | mapeamento | token]').

Na notação polonesa temos como outputs os tokens.

Para criação de palavras compatíveis com a linguagem gerada pela GLC foi utilizado o website "CFG Developer".

m(){ r(1); }

Palavra 1:

m(){ r(1); }

i	Qi	Token	Stack	Pi
0	Q0	m		-
1	Q1	m	M	P6
1	Q1	(	(	-
1	Q1	)	)	-
1	Q1	{	{	-
1	Q1		C	-

Error! Token = 'r' nao esperado.

Producoes: P1 P6

m(){ h=(x+y); r(0); }

Palavra 2:

m(){ h=(x+y); r(0); }

i	Qi	Token	Stack	Pi
0	Q0	m		-
1	Q1	m	M	P6
1	Q1	(	(	-
1	Q1	)	)	-
1	Q1	{	{	-
1	Q1		C	-
2	Q1	h	C	P16
2	Q1	=	=	-
2	Q1	(	E	-
3	Q1	(	E	P11
3	Q1	x	E	-
4	Q1	x	E	P9
5	Q1	+	X	P12
6	Q1	y	E	P10
6	Q1	;	;	-
6	Q1		r	-
6	Q1	(	(	-
6	Q1	0	E	-
7	Q1	0	E	P7
7	Q1	;	;	-
7	Q1		}	-
7	Q1		-	

Palavra aceita.

Producoes: P1 P6 P16 P11 P9 P12 P10 P7

Arvore sintatica: [0|0|S], [1|1|M], [2|13|m], [3|14|(], [4|15|)], [5|16|{], [6|17|C], [7|18|;],

[8|19|r], [9|20|(], [10|21|E], [11|22|)], [12|23|;], [13|24|}], [14|205|h], [15|206|=],

[16|207|E], [17|2485|(], [18|2486|E], [19|2487|X], [20|2488|E], [21|2489|)], [22|29833|x],

[23|29845|+], [24|29857|y], [25|253|0]

Arvore abstrata: [0 | 0 | m], [1 | 1 | =], [2 | 2 | r], [3 | 3 | h], [4 | 4 | +], [5 | 9 | x], [6 | 10 | y], [7 | 5 | 0]

Notacao ponolesa: m, =, h, +, x, y, r, 0, h, +, x, y

Notacao ponolesa reversa: h, x, y, +, =, 0, r, h, x, y, +, m

m(){ (1-1); r(1); }

Palavra 3:

m(){ (1-1); r(1); }

i	Qi	Token	Stack	Pi
0	Q0	m		-
1	Q1	m	M	P6
1	Q1	(	(	-
1	Q1	)	)	-
1	Q1	{	{	-
1	Q1		C	-
2	Q1	(	C	P21
2	Q1	1	E	-
3	Q1	1	E	P8
4	Q1	-	X	P13
5	Q1	1	E	P8
5	Q1	;	;	-
5	Q1		r	-
5	Q1	(	(	-
5	Q1	1	E	-
6	Q1	1	E	P8
6	Q1	;	;	-
6	Q1		}	-
6	Q1		-	

Palavra aceita.

Producoes: P1 P6 P21 P8 P13 P8 P8

Arvore sintatica: [0|0|S], [1|1|M], [2|13|m], [3|14|(], [4|15|)], [5|16|{], [6|17|C], [7|18|;], [8|19|r], [9|20|(], [10|21|E], [11|22|)], [12|23|;], [13|24|}], [14|205|(], [15|206|E],

[16|207|X], [17|208|E], [18|209|)], [19|2473|1], [20|2485|-], [21|2497|1], [22|253|1]

Arvore abstrata: [0 | 0 | m], [1 | 1 | -], [2 | 2 | r], [3 | 3 | 1], [4 | 4 | 1], [5 | 5 | 1]

Notacao ponolesa: m, -, 1, 1, 1, r, 1, 1, 1, 1

Notacao ponolesa reversa: 1, 1, 1, -, 1, r, 1, 1, 1, m

m(){ w(1) { (1/x); }; r(1); }

Palavra 4:

m(){ w(1) { (1/x); }; r(1); }

i	Qi	Token	Stack	Pi
0	Q0	m		-
1	Q1	m	M	P6
1	Q1	(	(	-
1	Q1	)	)	-
1	Q1	{	{	-
1	Q1		C	-
2	Q1	w	C	P22
2	Q1	(	(	-
2	Q1	1	E	-
3	Q1	1	E	P8
3	Q1		{	-
3	Q1		C	-
4	Q1	(	C	P21
4	Q1	1	E	-
5	Q1	1	E	P8
6	Q1	/	X	P15
7	Q1	x	E	P9
7	Q1	;	;	-
7	Q1		}	-
7	Q1	;	;	-
7	Q1		r	-
7	Q1	(	(	-
7	Q1	1	E	-

8 Q1 1 E P8

8 Q1 ; ; -

8 Q1 } -

8 Q1 -

Palavra aceita.

Producoes: P1 P6 P22 P8 P21 P8 P15 P9 P8

Arvore sintatica: [0|0|S], [1|1|M], [2|13|m], [3|14|(], [4|15|)], [5|16|{], [6|17|C], [7|18|;],

[8|19|r], [9|20|(], [10|21|E], [11|22|)], [12|23|;], [13|24|}], [14|205|w], [15|206|(],

[16|207|E], [17|208|)], [18|209|{], [19|210|C], [20|211|;], [21|212|}], [22|2485|1],

[23|2521|(], [24|2522|E], [25|2523|X], [26|2524|E], [27|2525|)], [28|30265|1],

[29|30277|/], [30|30289|x], [31|253|1]

Arvore abstrata: [0 | 0 | m], [1 | 1 | w], [2 | 2 | r], [3 | 3 | 1], [4 | 4 | /], [5 | 9 | 1], [6 | 10 | x], [7 | 5 | 1]

Notacao ponolesa: m, w, 1, /, 1, x, r, 1, 1, /, 1, x

Notacao ponolesa reversa: 1, 1, x, /, w, 1, r, 1, 1, x, /, m

n() { (0/y); r(y); } g() { i=y; r(x); } m() { (1-x); r(0); }

Palavra 5:

n() { (0/y); r(y); } g() { i=y; r(x); } m() { (1-x); r(0); }

i	Qi	Token	Stack	Pi
0	Q0	n		-
1	Q1	n	N	P4
1	Q1	(	(	-
1	Q1	)	)	-
1	Q1		{	-
1	Q1		C	-
2	Q1	(	C	P21
2	Q1	0	E	-
3	Q1	0	E	P7
4	Q1	/	X	P15
5	Q1	y	E	P10
5	Q1	;	;	-
5	Q1		r	-
5	Q1	(	(	-
5	Q1	y	E	-
6	Q1	y	E	P10
6	Q1	;	;	-
6	Q1		}	-
6	Q1		G	-
7	Q1	g	G	P5
7	Q1	(	(	-
7	Q1	)	)	-
7	Q1		{	-

7	Q1		C	-
8	Q1	i	C	P17
8	Q1	=	=	-
8	Q1	y	E	-
9	Q1	y	E	P10
9	Q1		r	-
9	Q1	(	(	-
9	Q1	x	E	-
10	Q1	x	E	P9
10	Q1	;	;	-
10	Q1		}	-
10	Q1		M	-
11	Q1	m	M	P6
11	Q1	(	(	-
11	Q1	)	)	-
11	Q1		{	-
11	Q1		C	-
12	Q1	(	C	P21
12	Q1	1	E	-
13	Q1	1	E	P8
14	Q1	-	X	P13
15	Q1	x	E	P9
15	Q1	;	;	-
15	Q1		r	-
15	Q1	(	(	-
15	Q1	0	E	-
16	Q1	0	E	P7

16 Q1 ; ; -

16 Q1 } -

16 Q1 -

Palavra aceita.

Producoes: P3 P4 P21 P7 P15 P10 P10 P5 P17 P10 P9 P6 P21 P8 P13 P9 P7

Arvore sintatica: [0|0|S], [1|1|N], [2|2|G], [3|3|M], [4|13|n], [5|14|(], [6|15|)], [7|16|{], [8|17|C], [9|18|;], [10|19|r], [11|20|(], [12|21|E], [13|22|)], [14|23|;], [15|24|}], [16|205|(], [17|206|E], [18|207|X], [19|208|E], [20|209|)], [21|2473|0], [22|2485|/], [23|2497|y], [24|253|y], [25|25|g], [26|26|(], [27|27|)], [28|28|{], [29|29|C], [30|30|;], [31|31|r], [32|32|(], [33|33|E], [34|34|)], [35|35|;], [36|36|}], [37|349|i], [38|350|=], [39|351|E], [40|4213|y], [41|397|x], [42|37|m], [43|38|(], [44|39|)], [45|40|{], [46|41|C], [47|42|;], [48|43|r], [49|44|(], [50|45|E], [51|46|)], [52|47|;], [53|48|}], [54|493|(], [55|494|E], [56|495|X], [57|496|E], [58|497|)], [59|5929|1], [60|5941|-], [61|5953|x], [62|541|0]

Arvore abstrata: [0 | 0 | |], [1 | 1 | n], [2 | 2 | |], [3 | 5 | g], [4 | 6 | m], [5 | 3 | /], [6 | 4 | r], [7 | 7 | 0], [8 | 8 | y], [9 | 9 | y], [10 | 11 | =], [11 | 12 | r], [12 | 23 | i], [13 | 24 | y], [14 | 25 | x], [15 | 13 | -], [16 | 14 | r], [17 | 27 | 1], [18 | 28 | x], [19 | 29 | 0]

Notacao ponolesa: x, r, g, 1, x, 0, -, 0, r, m, y, x, r, x, m, -, 1, x, 0, r, 0, n, /, 0, y, y, r, y, g, =, i, y, x, r, x, m, -, 1, x, 0, r, 0

Notacao ponolesa reversa: 0, y, y, /, y, r, n, i, y, x, =, x, r, g, 1, x, 0, -, 0, r, m, 0, y, y, /, y, r, n, i, y, x, =, x, r, g, 1, x, 0, -, 0, r, m

m(){w(x){f(y){k=(1+(1*0));};};r(0);}

Palavra 6:

m(){w(x){f(y){k=(1+(1*0));};};r(0);}

i	Qi	Token	Stack	Pi
0	Q0	m		-
1	Q1	m	M	P6
1	Q1	(	(	-
1	Q1	)	)	-
1	Q1	{	{	-
1	Q1	w	C	-
2	Q1	w	C	P22
2	Q1	(	(	-
2	Q1	x	E	-
3	Q1	x	E	P9
3	Q1	{	{	-
3	Q1	f	C	-
4	Q1	f	C	P23
4	Q1	(	(	-
4	Q1	y	E	-
5	Q1	y	E	P10
5	Q1	{	{	-
5	Q1	k	C	-
6	Q1	k	C	P19
6	Q1	=	=	-
6	Q1	(	E	-
7	Q1	(	E	P11
7	Q1	1	E	-

8	Q1	1	E	P8
9	Q1	+	X	P12
10	Q1	(	E	P11
10	Q1	1	E	-
11	Q1	1	E	P8
12	Q1	*	X	P14
13	Q1	0	E	P7
13	Q1	)	)	-
13	Q1	;	;	-
13	Q1	}	}	-
13	Q1	;	;	-
13	Q1	}	}	-
13	Q1	;	;	-
13	Q1	r	r	-
13	Q1	(	(	-
13	Q1	0	E	-
14	Q1	0	E	P7
14	Q1	;	;	-
14	Q1	}	}	-
14	Q1		-	

Palavra aceita.

Producoes: P1 P6 P22 P9 P23 P10 P19 P11 P8 P12 P11 P8 P14 P7 P7

Arvore sintatica: [0|0|S], [1|1|M], [2|13|m], [3|14|(], [4|15|)], [5|16|{], [6|17|C], [7|18|;], [8|19|r], [9|20|(], [10|21|E], [11|22|)], [12|23|;], [13|24|}], [14|205|w], [15|206|(], [16|207|E], [17|208|)], [18|209|{], [19|210|C], [20|211|;], [21|212|}], [22|2485|x], [23|2521|f], [24|2522|(], [25|2523|E], [26|2524|)], [27|2525|{], [28|2526|C], [29|2527|;], [30|2528|}], [31|30277|y], [32|30313|k], [33|30314|=], [34|30315|E], [35|363781|(],

[36|363782|E], [37|363783|X], [38|363784|E], [39|363785|)], [40|4365385|1],

[41|4365397|+], [42|4365409|(], [43|4365410|E], [44|4365411|X], [45|4365412|E],

[46|4365413|)], [47|52384921|1], [48|52384933|*], [49|52384945|0], [50|253|0]

Arvore abstrata: [0 | 0 | m], [1 | 1 | w], [2 | 2 | r], [3 | 3 | x], [4 | 4 | f], [5 | 9 | y], [6 | 10 | =], [7 | 21 | k], [8 | 22 | +], [9 | 45 | 1], [10 | 46 | *], [11 | 93 | 1], [12 | 94 | 0], [13 | 5 | 0]

Notacao ponolesa: 1, 1, 0, *, +, =, f, m, 1, *, 1, 0, r, 0, x, f, y, =, k, +, 1, *, 1, 0

Notacao ponolesa reversa: x, y, k, 1, 1, 0, *, +, =, f, w, 0, r, x, y, k, 1, 1, 0, *, +, =, f, m

Link para download

Código fonte e exemplos encontram-se para download no seguinte link:

https://github.com/MatBrands/Compiladores/tree/master/Proj3/Proj3b

Referências

https://web.stanford.edu/class/archive/cs/cs103/cs103.1156/tools/cfg/