Universidade Estadual de Santa Cruz – UESC

Relatórios de Implementações de Autômato de Pilha para GLC

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 busca simular um autômato pilha que reconhece a linguagem gerada por uma gramática livre de contexto. A execução do código recebe como entrada a palavra e retorna a Tabela de Parsing, suas Produções e o vetor referente a Árvore de Análise.

O autômato implementado ignora o token ' '.

Produções:

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-lo normalmente.

$ gcc proj2_a.c -o proj2_a

Ao executar é necessário digitar o nome do arquivo destino contendo as palavras, caso contrário resultará em erro.

Exemplo:

$ ./proj2_a examples/inputs.txt

Neste projeto pode-se adicionar num .txt todas as palavras separadas por uma quebra de linha.

Exemplo:

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

Testes

Caso a entrada dada seja incorreta o programa irá imprimir sua tabela até o momento de erro, então pulará a linha para ler a próxima palavra. Como outputs temos a tabela de parsing, as produções a árvore sintática (no formato 'palavra, posicao_no_vetor'), foi considerado o pior caso da produção 24, onde temos uma árvore n-ária de 4.

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

m(){ r(1); }

Palavra 1:

i	Qi	Token	Stack	Pi
0	Q0	m		-
1	Q1	m	M	P1
2	Q1	(	(	P6
2	Q1	(	)	-
2	Q1	)	{	-
2	Q1	{	C	-

Erro.

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

Palavra 2:

i	Qi	Token	Stack	Pi
0	Q0	m		-
1	Q1	m	M	P1
2	Q1	(	(	P6
2	Q1	(	)	-
2	Q1	)	{	-
2	Q1	{	C	-
3	Q1	=	=	P16
3	Q1	=	E	-
4	Q1	x	E	P11
5	Q1	+	X	P9
6	Q1	y	E	P12
7	Q1	)	)	P10
7	Q1	)	;	-
7	Q1	;	r	-
7	Q1	r	(	-
7	Q1	(	E	-
8	Q1	)	)	P7
8	Q1	)	;	-
8	Q1	;	}	-
8	Q1	}		-

Producoes: P1 P6 P16 P11 P9 P12 P10 P7

Arvore: 0, S 1, M 5, C 6, E 21, E 25, 0 85, E 86, X 87, E 341, x 345, + 349, y

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

Palavra 3:

i	Qi	Token	Stack	Pi
0	Q0	m		-
1	Q1	m	M	P1
2	Q1	(	(	P6
2	Q1	(	)	-
2	Q1	)	{	-
2	Q1	{	C	-
3	Q1	1	E	P21
4	Q1	-	X	P8
5	Q1	1	E	P13
6	Q1	)	)	P8
6	Q1	)	;	-
6	Q1	;	r	-
6	Q1	r	(	-
6	Q1	(	E	-
7	Q1	)	)	P8
7	Q1	)	;	-
7	Q1	;	}	-
7	Q1	}		-

Producoes: P1 P6 P21 P8 P13 P8 P8

Arvore: 0, S 1, M 5, C 6, E 21, E 22, X 23, E 25, 1 85, 1 89, - 93, 1

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

Palavra 4:

i	Qi	Token	Stack	Pi
0	Q0	m		-
1	Q1	m	M	P1
2	Q1	(	(	P6
2	Q1	(	)	-
2	Q1	)	{	-
2	Q1	{	C	-
3	Q1	(	(	P22
3	Q1	(	E	-
4	Q1	)	)	P8
4	Q1	)	{	-
4	Q1	{	C	-
5	Q1	1	E	P21
6	Q1	/	X	P8
7	Q1	x	E	P15
8	Q1	)	)	P9
8	Q1	)	;	-
8	Q1	;	}	-
8	Q1	}	;	-
8	Q1	;	r	-
8	Q1	r	(	-
8	Q1	(	E	-
9	Q1	)	)	P8
9	Q1	)	;	-
9	Q1	;	}	-

9 Q1 } -

Producoes: P1 P6 P22 P8 P21 P8 P15 P9 P8

Arvore: 0, S 1, M 5, C 6, E 21, E 22, C 25, 1 85, 1 89, E 90, X 91, E 357, 1 361, / 365, x

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

Palavra 5:

i	Qi	Token	Stack	Pi
0	Q0	n		-
1	Q1	n	N	P3
2	Q1	(	(	P4
2	Q1	(	)	-
2	Q1	)	{	-
2	Q1	{	C	-
3	Q1	0	E	P21
4	Q1	/	X	P7
5	Q1	y	E	P15
6	Q1	)	)	P10
6	Q1	)	;	-
6	Q1	;	r	-
6	Q1	r	(	-
6	Q1	(	E	-
7	Q1	)	)	P10
7	Q1	)	;	-
7	Q1	;	}	-
7	Q1	}	G	-
8	Q1	(	(	P5
8	Q1	(	)	-
8	Q1	)	{	-
8	Q1	{	C	-
9	Q1	=	=	P16
9	Q1	=	E	-

10	Q1	;	;	P10
10	Q1	;	r	-
10	Q1	r	(	-
10	Q1	(	E	-
11	Q1	)	)	P9
11	Q1	)	;	-
11	Q1	;	}	-
11	Q1	}	M	-
12	Q1	(	(	P6
12	Q1	(	)	-
12	Q1	)	{	-
12	Q1	{	C	-
13	Q1	1	E	P21
14	Q1	-	X	P8
15	Q1	x	E	P13
16	Q1	)	)	P9
16	Q1	)	;	-
16	Q1	;	r	-
16	Q1	r	(	-
16	Q1	(	E	-
17	Q1	)	)	P7
17	Q1	)	;	-
17	Q1	;	}	-
17	Q1	}		-

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

Arvore: 0, S 1, N 2, G 3, M 5, C 6, E 9, C 10, E 13, C 14, E 21, E 22, X 23, E 25, y 37, E 41, x 53, E 54, X 55, E 57, 0 85, 0 89, / 93, y 149, y 213, 1 217, - 221, x

Dificuldades enfrentadas

O relatório e código fonte sofreram atrasos por dificuldade ao implementar a árvore sintática, como este consiste num produto indispensável para conclusão do projeto, foi adiada a entrega até a conclusão deste item.

Link para download

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

https://github.com/MatBrands/Compiladores/tree/master/Proj2/Proj2a

Referências

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