Analisador Sintático para a Linguagem C- utilizando Autômato com Pilha: Projeto de Implementação

Gustavo Zanzin Guerreiro Martins[1]

[1]Departamento Acadêmico de Computação – Universidade Tecnológica Federal do Parná (UTFPR) Caixa Postal 86812-460 – 87301-899 – Campo Mourão – PR – Brazil

gustavozanzin@alunos.utfpr.edu.br

Resumo. *Este trabalho apresenta um analisador sintático para expressões aritméticas na linguagem de programacão C-. Tal projeto teve como base a tabela "Action Go-to" do algoritmo SLR(1) da gramática geradora da linguagem das expressões, com o auxílio do software JFLAP. E finalmente, foi realizada a implementação do autômato utilizando a biblioteca automata-lib da linguagem de programação Python.*

1. Descrição do Trabalho

O objetivo deste trabalho foi implementar um analisador sintático por meio de um autômato não determinístico com pilha para a linguagem de programação *C-*. O programa recebe como entrada uma lista de marcas (*tokens*) simplificadas (operações aritméticas) da linguagem *C-* e, como saída, reconhece se as expressões aritméticas inseridas na entrada foram aceitas ou não.

2. Descrição da Linguagem

Por conta de ser uma versão mais simplificada da linguagem de programação *C*, a linguagem *C-* foi a alternativa encontrada para desenvolver tal projeto sem a complexidade que seria a implementação desse mecanismo para reconhecer outros tipos de expressões e outras linguagens.

3. Descrição do Processo de Análise Sintática

No constexto da computação, o processo de Análise Sintática consiste na leitura e identificação de marcas (*tokens*) geradas pelo processo de Análise Léxica a fim de verificar a corretude da estrutura da gramática de uma determinada linguagem.

4. Implementação

Neste processo foi utilizado o software JFLAP a fim de compreender melhor o funcionamento da gramática simplificada de expressões aritméticas formadas pela linguagem de programação C- e como se dava seu comportamento ao utilizá-la em um automato com pilha e com o algoritmo SLR(1). Com isso, foi observado a tabela "Action Go-to" (Figura 1) gerada pelo JFLAP com base no SLR(1) na qual indica qual é o comportamento que o autômato deve ter de acordo com determinadas entradas e determinados símbolos que se encontram em um dado momento no topo da pilha, fazendo com que seja gerado, ao final do processo, a informação que aquela determinada expressão de entrada faz parte ou não da linguagem.

Além disso, para a implementação do analisador sintático (ou *Parser*) foi utilizada a linguagem de programação *Python* com o auxílio da biblioteca *automata-lib*.

	(	)	*	+	-	/	=	d	i	m	n	u	$	E	F	S	T	V
0	s1								s7		s8			2	3	4	5	6
1	s1								s7		s8			9	3		5	10
2				s11	s12								r1					
3		r9	r9	r9	r9	r9							r9					
4													acc					
5		r7	s13	r7	r7	s14							r7					
6		r10	r10	r10	r10	r10	s15						r10					
7								s16										
8												s17						
9		s18		s11	s12													
10		r10	r10	r10	r10	r10							r10					
11	s1								s7		s8				3		19	10
12	s1								s7		s8				3		20	10
13	s1								s7		s8				21			10
14	s1								s7		s8				22			10
15	s1								s7		s8			23	3		5	10
16		r11	r11	r11	r11	r11	r11						r11					
17										s24								
18		r5	r5	r5	r5	r5							r5					
19		r3	s13	r3	r3	s14							r3					
20		r6	s13	r6	r6	s14							r6					
21		r4	r4	r4	r4	r4							r4					
22		r8	r8	r8	r8	r8							r8					
23				s11	s12								r2					
24		r12	r12	r12	r12	r12							r12					

Figure 1. Tabela Action Go-to.