Fundação Universidade Federal de Mato Grosso do Sul
Faculdade de Computação
Curso de Graduação Ciência da Computação

Trabalho de LFA

Edvaldo R. Wassouf Júnior
Renato Alves
Thiago Santos
Mário Carvalho

Campo Grande - MS
12/06/2018

INTRODUÇÃO

Um compilador é um programa que traduz um código de uma linguagem-fonte para um código em uma linguagem-alvo. Para isso, os compiladores basicamente realizam duas etapas: análise e síntese. Na fase de análise, o código na linguagem-fonte é analisada do ponto de vista léxico, sintático e semântico, enquanto na síntese, um código equivalente é gerado na linguagem-alvo.

GRAMÁTICA LIVRE DE CONTEXTO

A gramática livre de contexto, em teoria de linguagem formal, é uma gramática formal onde todas as regras de produções são da forma Onde é um símbolo não terminal, e é uma cadeia de terminal e/ou não terminais.

OBJETIVO

O objetivo deste trabalho é alterar o analisador sintático para aumentar o conjunto de palavras geradas pela gramática da linguagem C-.

O presente trabalho consiste em alterar a gramática fornecida pela professora e fazer as seguintes:

Expressões aritméticas envolvendo operadores binários (− e /), operadores de incremento e decremento (++ e −−), operadores de atribuição (+ =, − =, ∗ =, / =), sendo que algumas dessas alterações devem ser feitas no léxico, e declaração de variáveis globais do tipo básico.

GRAMÁTICA FORNECIDA

A linguagem C- é aquela gerada pela seguinte gramática $G = (V, T, P, S)$ em que P é constituída pelas seguintes produções:

```
S          -> Function S_
S_         -> Function S_
            | epsilon
Function   -> Type Function_
Type       -> void
            | int
            | float
Function_  -> main() { B }
            | id() { B }
B          -> C B
            | epsilon
C          -> id = E ;
            | while (E) C
            | { B }
E          -> T E_
E_         -> + T E_
            | epsilon
T          -> F T_
T_         -> * F T_
            | epsilon
F          -> ( E )
            | id
            | num
```

Os conjuntos de variáveis V e de terminais T são dados abaixo:

```
V = {S, S_, Function, Function_, Type, C, B, E, T, E_, T_, F}

T= {ELSE, FLOAT, FOR, IF, INT, MAIN, VOID, WHILE, ID, NUM, OP_ATRIB, OP_ADIT, OP_MULT,
    OP_REL, ABRE_PARENT, FECHA_PARENT,PONTO_VIRG, VIRG, ABRE_CHAVES, FECHA_CHAVES, FIM,
    INVALIDO}
```

Fundação Universidade Federal de Mato Grosso do Sul
Faculdade de Computação
Curso de Graduação Ciência da Computação

GRAMÁTICA FINAL ALTERADA

Parte 1	Parte 2
S -> void S_ | int S_ | float S_ | S_ S_ -> id X | main() { D B } | epsilon Y -> id X | main() { D B } X -> , L ; S |; S | = num P |() {DB} S | epsilon Y_ -> id X P -> ; S | , Y_ S Type -> void | int | float D -> Type L ; D | epsilon L -> id L_ L_ -> , L | = num Z_	Z_ -> , | epsilon B -> C B | epsilon C -> id C_ | while (E) C | { B } C_ -> = E; | ++; | --; | += E ; | -= E ; | /= E ; | *= E ; E -> TE' E'-> +TE' | -TE' | epsilon T -> FT' T'-> * FT' | / FT' | epsilon F -> (E) | id F_ | num | ++ id | -- id F_ -> ++ | -- | epsilon

CONSIDERAÇÕES FINAIS

Para realizar o seguinte trabalho, foi necessário aplicar os conhecimentos obtidos na disciplina de LFA, ministrada pela professora Dra. Edna, como por exemplo, tirar recursão a esquerda, colocar a gramática em LL(1), retirar ambiguidade, etc. Fica claro, portanto, que a disciplina ministrada foi extremamente satisfatória e suficiente para realizar o presente trabalho.

O trabalho se encontra no seguinte link do Github :
https://github.com/MarioDeAraujoCarvalho/trabalho-lfa/