Vinicius F. da Silva

Compiladores - Uma abordagem técnica

Contents

List of Figures

List of Tables

Part I

Ecosystem-base Impacts of Climate Change

1

Conceitos básicos

Author Name

Vinicius Francisco da Silva

CONTENTS

“nobreak

1.1 Introdução

“nobreak

1.2 Definição

De maneira bem simples, um compilador é um programa de computador que recebe um arquivo de entrada denominado “programa fonte” realiza uma série

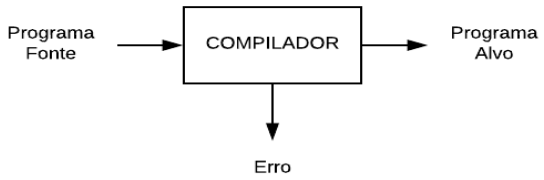

FIGURE 1.1
Figure caption goes here.

de análises e caso haja sucesso em todas as análises é gerado um outro arquivo correspondente denominado "programa alvo", caso contrário é gerado erro.

1.2.1 Programa fonte:

O programa fonte são arquivos escritos na linguagem de programação x, que será compilador em um compilador próprio para a linguagem x.

Todos os elementos da linguagem presentes no código fonte são denominados "símbolos" que são elementos únicos e indivisíveis. Qualquer outro símbolo que não pertence a linguagem é considerado inválido. Estes símbolos fazem parte do alfabeto (Sigma) da linguagem, na **imagem 2** temos o exemplo de um conjunto de símbolos (Alfabeto) da linguagem C.

IMAGEM 2

O alfabeto Σ é um conjunto de n símbolos de uma linguagem x. Este conjunto de símbolos possuí todos os elementos válidos da linguagem, ou seja, aquilo que a linguagem permite está presente neste conjunto.

Para reforçar os símbolos são elementos indivisíveis da linguagem, que fazem parte do conjunto denominado alfabeto, representado por Σ. Fazendo uma analogia, um símbolo da linguagem de programação é como uma letra de uma linguagem comum que falamos, que fazem parte do alfabeto.

Na implementação do compilador, os símbolos da linguagem são estruturas que possuem uma série de informações como: lexema, token, classe, endereço, padrão de formação e descrição.

IMAGEM 3

Nesta sessão serão definidos os lexemas, tokens, padrão de formação e descrição. As classes e os endereços serão abordados nos CAPÍTULOS 7 E 8.

1.2.1.1 Token

Um token é um elemento do alfabeto Σ. Pode-se dizer que um token é um valor correspondente ao símbolo pertencente ao alfabeto. Representa uma unidade léxica.

IMAGEM 4

1.2.1.2 Lexema

Um lexema é definido com as formas que um token pode ser escrito na linguagem. Tome o identificador como exemplo (id). O token do símbolo é id, porém o id pode ser escrito (nomeado) de várias formas diferentes. Estas formas são considerados os lexemas válidos para o token id. Um outro exemplo é o token constante, que pode ser representado por números na faixa de $-\infty$ á $+\infty$. Estas representações são os lexemas do token constante.

Dentre os símbolos do alfabeto o único que não é utilizado é o espaço em branço, está presente no alfabeto da linguagem mas não possui importância para a análise do compilador, sendo descartado nas primeiras fases do compilador. Pode-se considerar o espaço em branco apenas como um delimitador léxico.

IMAGEM 5

1.2.1.3 Padrão de formação de lexema

É uma expressão regular que define a formação dos lexemas de um token.

Exemplos na linguagem Pascal:

1.2.1.4 Descrição do símbolo

A descrição de símbolos é uma rotulação feita para identificar os símbolos em uma linguagem de programação. Na implementação do compilador esta informação não é utilizada, porém ela é de extrema importância para a sep-

TABLE 1.1
Now we are engaged (a_g^a) $\left(a_g^a\right)$ in a great civil war, testing whether that nation, or any nation so conceived.

Token	**Lexema**	**Padrão de formação**
constante	0,-2,20,...	(-digito+ ∪ digito+)
<	<	<
if	If,IF,if,iF	(i ∪ I)(f ∪ F)
constante	"a", "String", "cadeia de caracteres"	"(Símbolos de Σ)"
while	While,WHILE,while	(W ∪ w)(h ∪ H)(i ∪ I)(l ∪ L)(e ∪ E)

aração e identificação dos símbolos da linguagem.

Estes rótulos são:

- Identificadores
- Constantes numéricas
- Constantes literais
- Palavras Reservadas
- Operador relacional

Exemplos:

TABLE 1.2
Now we are engaged (a_g^a) $\left(a_g^a\right)$ in a great civil war, testing whether that nation, or any nation so conceived.

Token	**Lexema**	**Padrão de formação**	**Descrição**
constante	0,-2,20,...	(-digito+ ∪ digito+)	Constante numérica
constante	"Cadeia de caracteres", ...	("Símbolos de Σ")*	Constante literal
for	For, for, FOR, ...	(f ∪ F)(o ∪ O)(r ∪ R)	Palavra reservada
identificador	a,ab,n,x, ...	(∪ d)(∪ d ∪ l)*	Identificador
if	if,If,IF	(i ∪ I)(f ∪ F)	Palavra reservada

1.2.2 Programa alvo

É um programa gerado pelo compilador, correspondente ao programa passado na entrada como mostra a **Figura 1**. Pode-se dizer que em quase todos os compiladores que o programa alvo é um programa Assembly de uma máquina específica.

IMAGEM 5

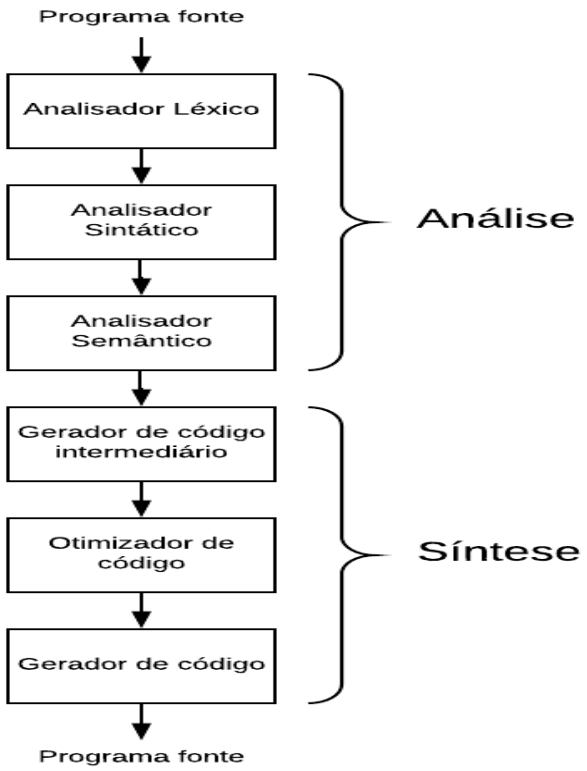

FIGURE 1.2
Figure caption goes here.

1.3 Estrutura

O compilador é dividivo em duas partes: A análise e a síntese.

1.3.1 Análise

É a parte que o compilador recebe o programa fonte e faz uma série de análises léxicas, sintáticas, e semânticas. É através da análise que caso o código fonte tenha erro o mesmo é gerado e tratado. É uma etapa que certifica que o arquivo fonte recebido na entrada está pronto para ser "tranformado" em código alvo.

A etapa de análise é dividida em 3: Analisador Léxico, Analisador Sintático, Analisador Semântico.

1.3.1.1 Analisador Léxico

Responsável por percorrer o código fonte realizando a identificação dos lexemas (ação semântica) e o atribuindo a um símbolo com seu token correspondente. É através do analisador léxico que comentários, espaços em branco são descartados.

IMAGEM 7

Fazendo uma alusão, podemos dizer que o processo do analisador léxico é como o processo de verificação de palavras em um texto, frase, poema, etc ...

Vemos o exemplo abaixo, utilizando a linguagem de programação C e a língua portuguesa.
IMAGEM 7 E 8

1.3.1.2 Analisador Sintático

É responsável por agrupar os tokens dos símbolos em estruturas hierárquicas.

Esta estrutura hierárquica denominado árvore de parse é fruto da derivação de uma gramática livre de contexto. Este processo de agrupamento tem a função de realizar a análise gramatical que é simplesmente conferir o que foi escrito no programa fonte com o que está na gramática da linguagem.

Fazendo alusão podemos dizer que o agrupamento de tokens na análise sintática é um agrupamento de palavras na linguagem falada. Não é possível verificar a coerência gramatical com apenas uma palavra. É necessário agrupar os elementos para que esta analise seja feita.

Tome parte de uma gramática livre de contexto GLC abaixo e parte da gramática da língua portuguesa.

Linguagem C

G:

S → int **Atribuicao** | char **Atribuicao** | byte **Atribuicao**
Atribuicao → id = **Valores**;
Valores → const | id

Língua Portuguesa

G:

Oracao → **Sujeito Predicado**
Sujeito → artigo substantivo
Predicado → **VerboLigativo PredicativoSujeito**
VerboLigativo → verbo
PredicativoSujeito → adjetivo

Part II

Climate Risks in Public Health and Epidemiology

Part III

Socio-economic Impacts of Climate Adaptation and Resilience