Compiladores e Linguagens Formais: Fundamentos Teóricos e Aplicação Prática

Autor: Guilherme Teodoro Zormann
Instituição: UNIC – Universidade de Cuiabá
Palavras-chave: compiladores; linguagens formais; autômatos; teoria da computação; análise léxica.

Introdução

Os compiladores são ferramentas essenciais no mundo da computação, responsáveis por traduzir programas escritos em linguagens de alto nível para uma forma que o computador consiga entender. Apesar de parecer apenas um processo técnico, há uma base teórica muito importante por trás disso, sustentada pelos conceitos de linguagens formais e autômatos. Segundo Garcia e Haeusler (2017), compreender esses conceitos é fundamental para entender como as linguagens de programação são estruturadas e processadas. Cada etapa de um compilador — desde a leitura do código até sua tradução final — depende de regras formais que garantem que o programa seja interpretado corretamente. Dessa forma, estudar linguagens formais e autômatos não é apenas algo teórico, mas uma parte essencial do funcionamento de qualquer compilador moderno.

Objetivo

Este artigo tem como objetivo mostrar de forma simples como os compiladores estão diretamente ligados aos conceitos de linguagens formais e autômatos, tomando como base o livro *Linguagens Formais e Autômatos* de Garcia e Haeusler (2017). Além da parte teórica, é apresentado um exemplo prático em Python que demonstra como um autômato finito determinístico (AFD) pode ser usado para reconhecer padrões básicos — tarefa que faz parte da análise léxica em um compilador.

Metodologia

A construção deste trabalho se baseou na leitura e interpretação do conteúdo apresentado por Garcia e Haeusler (2017), que explicam de forma detalhada os tipos de linguagens formais (regulares, livres de contexto, sensíveis ao contexto e recursivamente enumeráveis) e os autômatos que as reconhecem (finito, com pilha e máquina de Turing). Com base nessas definições, foi desenvolvido um exemplo prático em Python para representar o comportamento de um autômato finito. Esse tipo de modelo é usado na fase de análise léxica, que identifica e separa os símbolos de um código-fonte. Durante o desenvolvimento, utilizou-se o Git para

versionar o código, registrando as modificações e garantindo o controle individual do projeto.

Resultado

Durante o processo de compilação, as linguagens regulares são as mais usadas na análise léxica, pois permitem reconhecer padrões simples e repetitivos, como números e operadores. Essa etapa pode ser representada por um autômato finito determinístico (AFD), que percorre o código verificando se cada parte pertence à linguagem definida. A seguir, temos um exemplo prático e bem simples em Python que mostra como esse reconhecimento acontece. O código identifica números e operadores matemáticos em uma pequena expressão:

```
entrada = "3 + 7 - 2"
for caractere in entrada.split():
    if caractere.isdigit():
        print(f"Token: NÚMERO ({caractere})")
    elif caractere in ['+', '-', '*', '/']:
        print(f"Token: OPERADOR ({caractere})")
    else:
        print(f"Token inválido: {caractere}")
```

Saída esperada:

```
Token: NÚMERO (3)
Token: OPERADOR (+)
Token: NÚMERO (7)
Token: OPERADOR (-)
Token: NÚMERO (2)
```

Conclusão

A teoria das linguagens formais e dos autômatos, apresentada por Garcia e Haeusler (2017), é a base de praticamente todo o funcionamento dos compiladores. Com ela, é possível entender como um programa reconhece estruturas, verifica regras e transforma comandos escritos por humanos em instruções que a máquina consegue executar. O exemplo em Python demonstra de forma simples essa ligação entre teoria e prática, mostrando que os conceitos estudados não ficam apenas no papel. O uso de ferramentas como o Git também mostra a importância de práticas modernas no desenvolvimento de software, mesmo em trabalhos acadêmicos. Assim, o estudo das linguagens formais e dos autômatos não é apenas parte da teoria da computação — é um passo essencial para compreender o que realmente

acontece “por baixo dos panos” quando um programa é compilado.

Referência

GARCIA, Alex de Vasconcellos; HAEUSLER, Edward Hermann. *Linguagens Formais e Autômatos*. Cuiabá: UNIC, 2017.