UNIVERSIDADE ESTADUAL DO PARANÁ - *CAMPUS APUCARANA*

Lissa Guirau Kawasaki

RELATÓRIO TÉCNICO

Linguagens Formais, Autômatos e Computabilidade

APUCARANA – PR

2024

Lissa Guirau Kawasaki

RELATÓRIO TÉCNICO

Linguagens Formais, Autômatos e Computabilidade

Trabalho apresentado à disciplina de Linguagens Formais, Autômatos e Computabilidade do curso de Bacharelado em Ciência da Computação.

Professor: Guilherme Henrique de Souza Nakahata.

APUCARANA – PR

2024

SUMÁRIO

INTRODUÇÃO

A área relacionada a Linguagens Formais, Autômatos e Computabilidade se destaca como fundamental para a compreensão dos princípios de processamento e manipulação da informação. Essa área engloba conceitos abstratos que permitem a análise de linguagem, de comportamento de máquinas e os limites da computação. As Linguagens Formais servem como a base para a construção de sistemas computacionais. Através de regras gramaticais pré definidas, elas nos permitem descrever com precisão a estrutura e o significado de sequências de símbolos, como palavras em um código de programação ou instruções em um sistema operacional. O conhecimento de análise de linguagens formais dá acesso ao desenvolvimento de compiladores, tradutores e ferramentas de análise sintática, componentes essenciais para a construção de softwares complexos.

Os autômatos servem como um modelo abstrato de máquinas que podem processar e reconhecer padrões em sequências de símbolos. Estas máquinas, guiadas por regras de transformação definidas, imitam o comportamento de sistemas informáticos reais, permitindo-nos analisar a sua capacidade de perceber e manipular informação. Entre os tipos de autômatos mais importantes encontramos os Autômatos Finitos Determinísticos (AFD), modelos simplificados que operam de forma previsível e determinística, seguindo um conjunto fixo de regras para cada sinal de entrada.

OBJETIVOS

Os AFDs se distinguem por sua simplicidade e eficiência, tornando-os ferramentas valiosas para diversas aplicações práticas. A criação de um programa que simula um Autômato Finito Determinístico (AFD) envolve uma série de objetivos práticos que visam aprofundar o entendimento teórico e a aplicação prática dos conceitos de teoria da computação. Através de sua estrutura, é possível utilizar de AFDS para validar entradas, analisar tokens, identificar a classificação de unidades em um código de programação como identificadores e operadores e implementar circuitos lógicos.

À vista disso, o objetivo principal do código gerado pode ser entendido como configurar uma implementação prática de um Autômato Finito Determinístico, reconhecendo diferentes linguagens formais, expressões regulares e palavras em uma linguagem de programação específica, visualizando transições, identificando estados e analisando comportamentos, dessa forma, quebrando a abstração dos conceitos essenciais e propondo o aprimoramento em diversas habilidades lógicas e de análise.

Ao dominar os conceitos de linguagens estruturadas, Autômatos e Continuidade, especialmente AFD, garantimos a compreensão de áreas complexas da computação e da própria linguagem, permitindo a compreensão dos fundamentos da Ciência da Computação, e abrindo possibilidades para a criação de novos sistemas computacionais.

MOTIVAÇÃO E RECURSOS UTILIZADOS

2.1 Motivação.

No mundo da Ciência da Computação, o Autômato Finito Determinístico (AFD) serve como um modelo abstrato de máquinas que podem processar e reconhecer padrões de acordo com sequências de sinais. Esses modelos são fundamentais na teoria da computação e têm aplicações importantes em diversos campos. A principal motivação para a criação de um simulador de AFD é a análise de sistemas informáticos e sua capacidade de perceber e manipular informação. Essas regras, incluindo a função de transição, mapeiam cada par de estado atual e símbolo de entrada para o próximo estado único. Essa certeza é o que distingue o AFD de outros tipos de autômatos, como os autômatos finitos não determinísticos (AFN), nos quais múltiplas transformações podem ser realizadas para um determinado estado e sinal de entrada.

2.2 Linguagem de Programação

Para o código, foi utilizado da linguagem de programação Python, uma linguagem que disponibiliza ferramentas eficientes e versáteis para o trabalho, a linguagem oferece uma gama de benefícios que impulsionam a produtividade e a resolução de problemas, sendo assim, a sua sintaxe clara e intuitiva e sua abundância de ferramentas torna ideal para a utilização em um programa que tem como seu principal objetivo a solução de problemas.

2.3 Estrutura de Dados.

De maneira geral, o código utiliza listas, dicionários e variáveis para representar o AFD e suas regras de transição. As listas utilizadas são “Estados”, que armazena os nomes dos estados do AFD, “Símbolos”, que armazena os símbolos válidos que podem ser utilizados na linguagem e “Estados_Finais”, que armazena os estados que representam a aceitação de uma palavra; o dicionário utilizado é a função_Transição, que armazena as regras de transição do AFD. Cada chave do dicionário é uma tupla (estado

atual, símbolo), e o valor associado ao próximo estado para qual a máquina se move após ler o símbolo. Um valor “None”, representado por um “X” indica que não há transição definida para a combinação de estado e símbolo.

A função principal do programa é definida por “executar_AFD", essa função é responsável por interagir com o usuário para realizar o processamento das palavras, validando a entrada e simulando a execução do AFD, assim, imprimindo a validade da palavra inserida. Fora da função, acontece a solicitação ao usuário para que preencha manualmente as regras de transição do AFD.

1. **Estado inicial:** O usuário fornece o estado inicial do autômato.
2. **Estados finais:** O usuário lista os estados finais.
3. **Conjunto de estados:** O usuário informa todos os estados possíveis do autômato.
4. **Símbolos de entrada:** O usuário lista todos os símbolos que o autômato pode processar.
5. **Transições:** O usuário define as transições para cada combinação de estado e símbolo, preenchendo a função de transição.

Para cada combinação de estado e símbolo, o usuário informa o próximo estado para o qual a máquina se move.

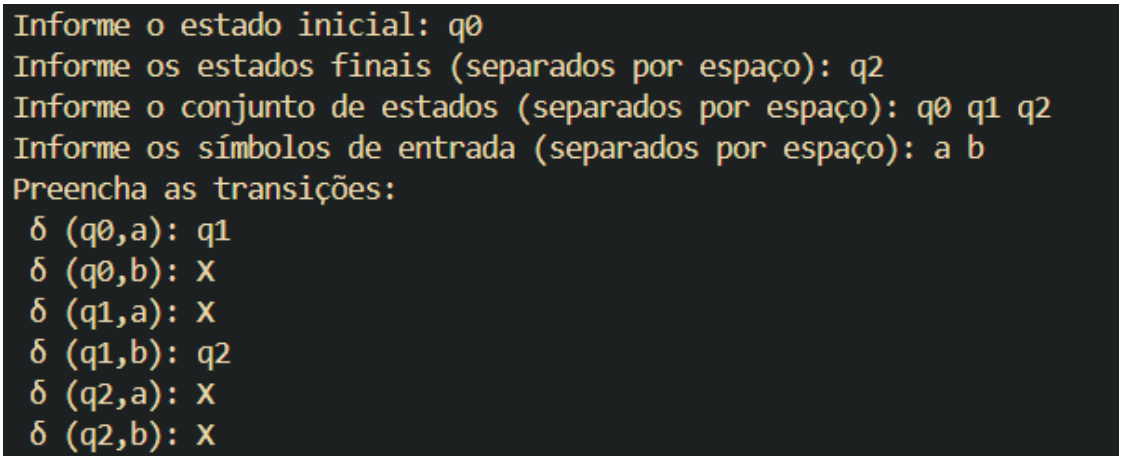
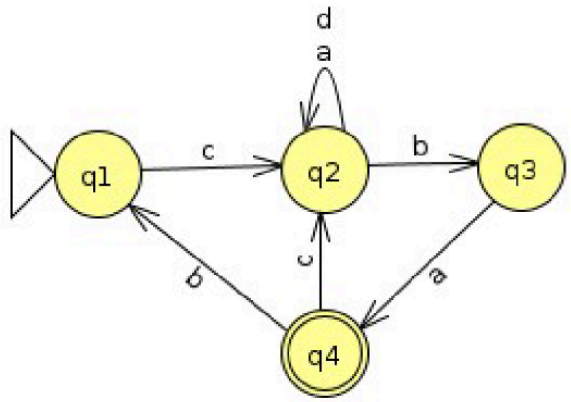


Figura 1, exemplo de uso.

RESULTADO

Com os procedimentos realizados anteriormente, o pleno funcionamento completo do código foi alcançado conforme o esperado - Para melhor visualização dos resultados, é utilizado do seguinte autômato:



	a	b	c	d
q1	X	X	q2	X
q2	q2	q3	X	q2
q3	q4	X	X	X
q4	X	q1	q2	X

Primeiramente, é realizada a entrada do usuário, utilizando da definição formal de Autômatos Finitos Determinísticos.

```
Informe o estado inicial: q1
Informe os estados finais (separados por espaço): q4
Informe o conjunto de estados (separados por espaço): q1 q2 q3 q4
Informe os símbolos de entrada (separados por espaço): a b c d
```

Figura 1.

A definição formal é definida por uma quíntupla (Q, Σ, δ, q0, F), onde:

Q: Conjunto finito de estados.

Σ: Alfabeto finito de símbolos de entrada.

δ: Função de transição que mapeia pares de um estado e um símbolo para um novo estado.

q0: Estado inicial do AFD.

F: Conjunto de estados finais.

Em seguida, é realizado o preenchimento das transições, também pelo usuário.

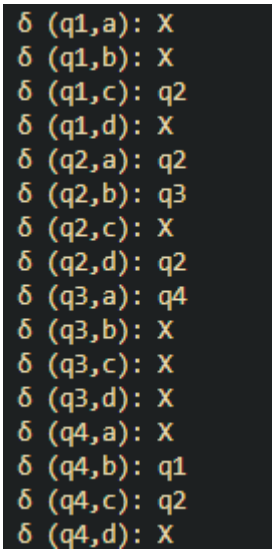


Figura 2.

Concluindo, o programa pede ao usuário que insira uma palavra a ser verificada, dessa forma, retornando a validade da palavra inserida baseada nas informações previamente preenchidas.

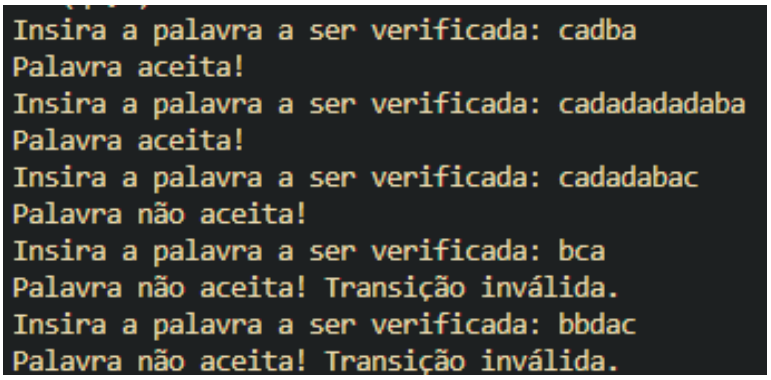


Figura 3.

CONCLUSÃO

A partir dos testes realizado acima, tomamos as seguintes conclusões:

O programa foi capaz de concluir sua proposta de forma satisfatória, ao utilizarmos o Python em sua aplicação, foi possível visualizar de forma desconstruída a lógica de programação por trás da implementação de um AFD, tendo sucesso em exemplificar o funcionamento e reconhecer linguagens especificadas pelo usuário.

Ao longo do trabalho, foi possível explorar de forma prática e lógica a Autômatos Finitos Determinísticos, ultrapassando a barreira de conceitos abstratos. Assim como dito anteriormente, os Autômatos Finitos Determinísticos e a teoria das linguagens formais desempenham um papel crucial na ciência da computação, especialmente na modelagem e análise de sistemas computacionais. AFDs são utilizados para representar e verificar o comportamento de programas de software e hardware, garantindo que eles funcionem conforme o esperado. Através do projeto, foi possível confirmar sua essencialidade na área e seus estudos.

REFERÊNCIAS

HOPCROFT, J. E.; ULLMAN, J. D.; MOTWANI, R. Introdução à Teoria de Autômatos, Linguagens e Computação. Editora Campus, 2002

MENEZES, P. B. Linguagens Formais e Autômatos. 6 ed.Editora Artmed, 2011.