UNIVERSIDADE FEDERAL DA FRONTEIRA SUL
CIÊNCIA DA COMPUTAÇÃO

ANA ROZA JORGE NETA

DANIELE KAROLINE CARVALHO ROSA

TRABALHO FINAL

Construção de aplicação geradora de Autômato Finito Determinístico

CHAPECÓ

2023

ANA ROZA JORGE NETA

DANIELE KAROLINE CARVALHO ROSA

TRABALHO FINAL

Construção de aplicação geradora de Autômato Finito Determinístico

Trabalho Final da matéria de Linguagens Formais e Autômatos com o intuito de avaliação parcial.

CHAPECÓ

2023

SUMÁRIO

1 Introdução

O autômato finito é o primeiro modelo computacional de definição de linguagens que são definidas por mecanismo de reconhecimento, que pode ser encarado como um teste aplicado a cada caractere da palavra. A linguagem reconhecida pelo autômato finito é constituída por todas as palavras que passem no teste. Este teste é aplicado de forma incremental, percorrendo os símbolos da palavra um a um a partir do seu início, e a decisão final só surge após o percurso completo da palavra, conferindo a qualidade computacional dos autômatos finitos. [1]

Os autômatos finitos podem ser determinísticos ou não-determinísticos, apresentando ou não transições em vazio. A utilização do termo "determinístico" para designar esse tipo de autômato finito decorre do fato de que, enquanto houver símbolos na fita de entrada, será sempre possível determinar o estado seguinte a ser assumido pelo autômato, o qual será único em todas as situações. [2]

Um autômato finito determinístico — também chamado máquina de estados finita determinística (AFD) — é uma máquina de estados finita que aceita ou rejeita cadeias de símbolos gerando um único ramo de computação para cada cadeia de entrada.[1]

Um importante resultado da teoria dos autômatos refere-se à equivalência da classe dos autômatos finitos determinísticos com a dos não-determinísticos, ou seja, não determinismos em autômatos finitos em nada contribuem para ampliar a classe de linguagens por eles reconhecíveis. Conforme mostrado anteriormente, é sempre possível transformar autômatos finitos não-determinísticos que contenham transições em vazio em outros equivalentes, determinísticos, isentos de tais transições. Pode-se provar que cada conjunto regular é reconhecido por um autômato finito mínimo e único. O termo mínimo é empregado para designar um autômato finito que tenha o número mínimo possível de estados. [2]

A minimização do número de estados de um autômato finito é feita em duas etapas:

1. Eliminação de estados inacessíveis e inúteis;
2. Agrupamento e fusão de estados equivalentes.

Estados inacessíveis e inúteis podem ser eliminados através da aplicação dos algoritmos apresentados anteriormente, ficando então o autômato finito determinístico apto a ser reduzido à sua versão mínima através da fusão dos seus estados equivalentes.

O princípio desse método consiste em se efetuar o particionamento sistemático do conjunto de estados do autômato em grupos de estados sucessivamente mais abrangentes, até que cada grupo contenha o maior número de estados possível, o que caracteriza a respectiva classe de equivalência.

Uma vez esgotado o processo, ou seja, quando não mais for possível modificar os grupos construídos, escolhe-se um representante único para cada classe, descartando-se os demais estados da mesma classe, uma vez que se trata de estados equivalentes. [2]

Como os autômatos finitos representam a mesma classe de linguagens, sejam determinísticos ou não-determinísticos, é de se esperar que haja uma equivalência entre eles. Realmente esta equivalência existe. Assim, para cada AFND é possível construir um AFD equivalente. Assim, é possível utilizar a clareza da representação de um AFND, e para fins de facilidade de implementação, transformá-lo num AFD. [3]

2 Metodologia e Desenvolvimento

O objetivo do trabalho se baseia na construção de uma aplicação capaz de gerar um AFD livre de estados inalcançáveis e mortos.

Para obter os resultados, respostas e promover o desenvolvimento da aplicação geradora de autômato finito determinístico, utilizou-se a linguagem Python como cerne.

Entrada: arquivo com a relação de tokens e/ou GRs de uma linguagem.

Saída: Autômato Finito Determinístico (AFD) e mínimo sem a aplicação de classes de equivalência entre estados.

A aplicação executa a carga de tokens e Gramáticas Regulares (GR) a partir de um arquivo de texto.

Exemplo de arquivo de entrada:

---

se

entao

senao

<S> ::= a<A> | b<A> | c<A> | d<A>

<A> ::= a<A> | b<A> | c<A> | d<A> | ε

Para cada token e gramática, a aplicação gera o conjunto de transições rotuladas em um único AF durante o procedimento de carga.

No AF, apenas o estado inicial é compartilhado entre diferentes tokens/gramáticas. Os demais estados são exclusivos para as transições dos demais símbolos dos tokens e/ou estados das GRs. Ou seja, após o estado final, na carga, todas as demais transições têm um novo estado como destino.

O AF será indeterminístico quando ocorrer uma ou mais situações em que dois tokens ou sentenças definidas por GR iniciam pelo mesmo símbolo.

Determinização: Aplicar o teorema de determinização para obter o AFD.

Minimização: O AFD resultante deve ser submetido ao processo de minimização, contudo, sem aplicar a Classe de Equivalência. No AFD final os estados podem ser representados por números.

Estado de erro: Ao final da minimização, acrescentar um último estado final. Este será o estado de erro.

3 Discussão

Considerando um autômato, podendo ser determinístico ou não e apresentando transições em vazio ou não, é possível determinar o seu estado seguinte a partir da fita de entrada da cadeia. Levando-nos a comparação ou equivalência entre autômato determinístico e não determinístico, sendo possível a minimização do número de estados de um autômato em duas etapas: eliminação de estados inacessíveis e inúteis e agrupamento de estados equivalentes. Terminado o processo, chega-se a uma classe equivalente

4 Conclusão

Através deste trabalho, tivemos a oportunidade de explorar o campo da teoria dos Autômatos Finitos Não Determinísticos e aprimorar nossas habilidades em algoritmos e programação. O desenvolvimento do algoritmo de Determinização de um Autômato Finito Não Determinístico proporcionou um desafio empolgante, permitindo-nos aplicar os conceitos aprendidos em sala de aula e ampliar nossa compreensão sobre o assunto.

Durante o processo de pesquisa e implementação, fomos expostas a diversas questões e obstáculos, que nos incentivaram a aprofundar nosso conhecimento teórico e buscar soluções criativas. A análise minuciosa dos autômatos e a compreensão dos princípios subjacentes foram essenciais para o sucesso do projeto.

5 Referências Bibliográficas

1. Menez, P. Blauth, Nome. Matemática Discreta para Ciência da Computação. Departamento de Informática Teórica Instituto de Informática / UFRGS.
2. Ramos, Marcos Vinicius Videna. Linguagens Formais e Autômatos. Universidade Federal do Vale do São Francisco: 2008.
3. Scheffel, Roberto M. Apostila de Linguagens Formais e Autômatos. Universidade Federal do Sul de Santa Catarina.