FACULDADE DE CIÊNCIAS E TECNOLOGIA

DEPARTAMENTO DE MATEMÁTICA E COMPUTAÇÃO

CIÊNCIA DA COMPUTAÇÃO

JUAN CARDOSO DA SILVA

LUIZ FILIPE MIGUEL MONGE

VINÍCIUS VEDOVOTTO

**Trabalho de LFA – Parte 2**

Relatório e Manual do usuário.



PRESIDENTE PRUDENTE

2019

Sumário

[Sumário 2](#_Toc20448781)

[Introdução 3](#_Toc20448782)

[Estrutura e Interface 4](#_Toc20448783)

[Implementação do RegEx 17](#_Toc20448784)

[Implementação da Gramática 20](#_Toc20448785)

[Implementação do Autômato 23](#_Toc20448786)

# Introdução

Na segunda parte do trabalho, foi pedido implementação de funcionalidades que permitissem a exportação de gramática, expressões regulares e autômatos, como também a conversão entre gramática-autômato, e autômatos finitos e determinísticos. Utilizando a mesma base do trabalho 1.

# Estrutura e Interface

Agora com a importação e exportação adicionada, na pasta base do projeto existe novos tipos de arquivos, os arquivos em xml com extensão .jff do Jflap para leitura das exportações a serem realizadas.

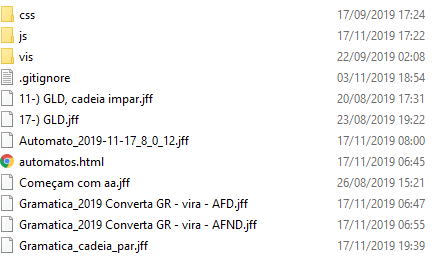


Figura 1 - exemplos de arquivos .jff na pasta base

Não tem muito o que fala em quesito de estruturação de arquivo, apenas as exportações e um novo tipo de arquivo .js chamado gramar-importer, que será discutido na parte de implementações.

**Interface**

Ao entrar na grammar.html, existe um novo botão chamado exportar e importar, ele abre um modal que contem mais dois botões exportar e importar, exportar uma gramática resulta em um arquivo com nome, data e hora de quando a exportação foi realizada como na figura 3.



Figura 2 - imgem mostrando as novas funcionalidades

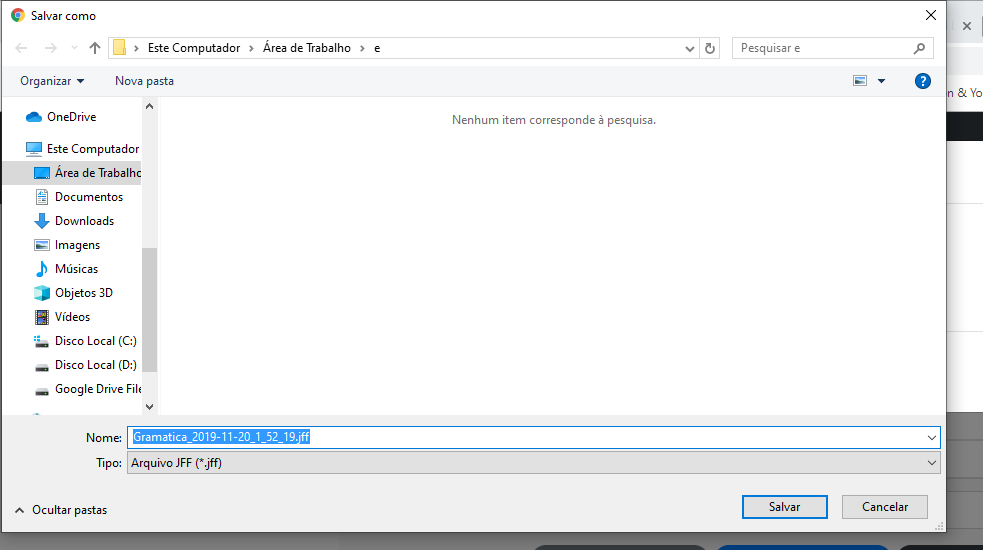


Figura 3 – exemplo de como o arquivo é nomeado ao ser exportado, é possível renomear ele caso queira.

O botão de importar carrega uma gramática e criar as linhas das tabelas conforme a gramática escrita.

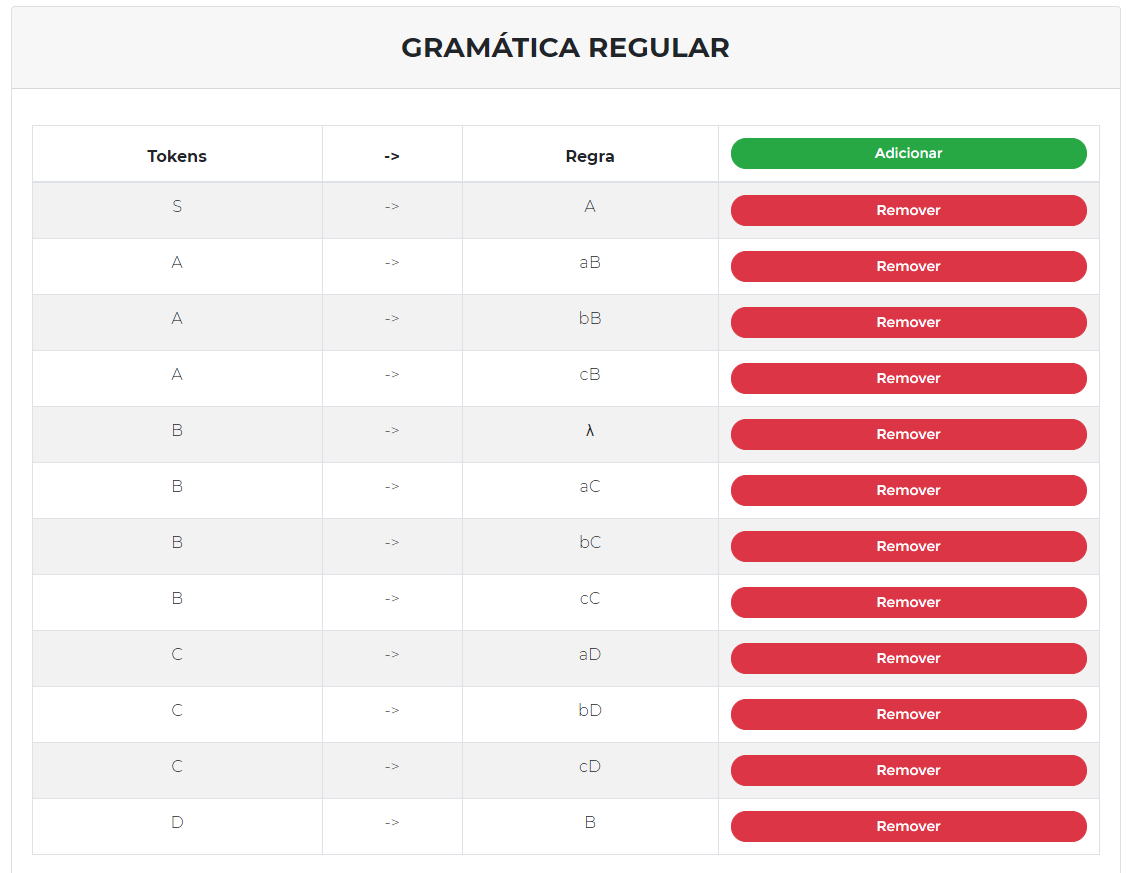


Figura 4 - Exemplo de uma gramática lida, permite cadeias ímpares.

Os mesmos botões existem para os autômatos, mas com estilos diferentes, o botão com a seta pra cima, serializa o autômato e cria um arquivo para salvar, enquanto o botão para importar é o botão com a seta para baixo.

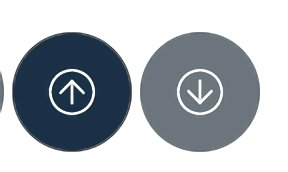


Figura 5 - botões para baixar e importar automatos

Implementação da exportação e importação

Gramática Regular e Autômatos Finitos

A implementação da importação da gramática segue uma mesma maneira que a dos autômatos ocorrem, mudando apenas o conteúdo do XML e algumas funções dentro do autômatos.js

A exportação ocorre ao criar uma string utilizando a mesma forma do XML, adicionando os valores das variáveis em cada parte da string para gerar uma string final do qual será armazenada dentro do arquivo utilizando a biblioteca chamada FileSaver, ela é responsável por gerar uma função que permite criar um arquivo de um formato e tipo específico.



Figura 6 - parte principal do algoritmo de exportação

A importação da gramática ocorre usando uma biblioteca chamar xmlToJson, q é responsável para de-serializar o XML em um objeto Json e gerar um objeto do qual podemos extrair as informações das gramáticas para a tabela.

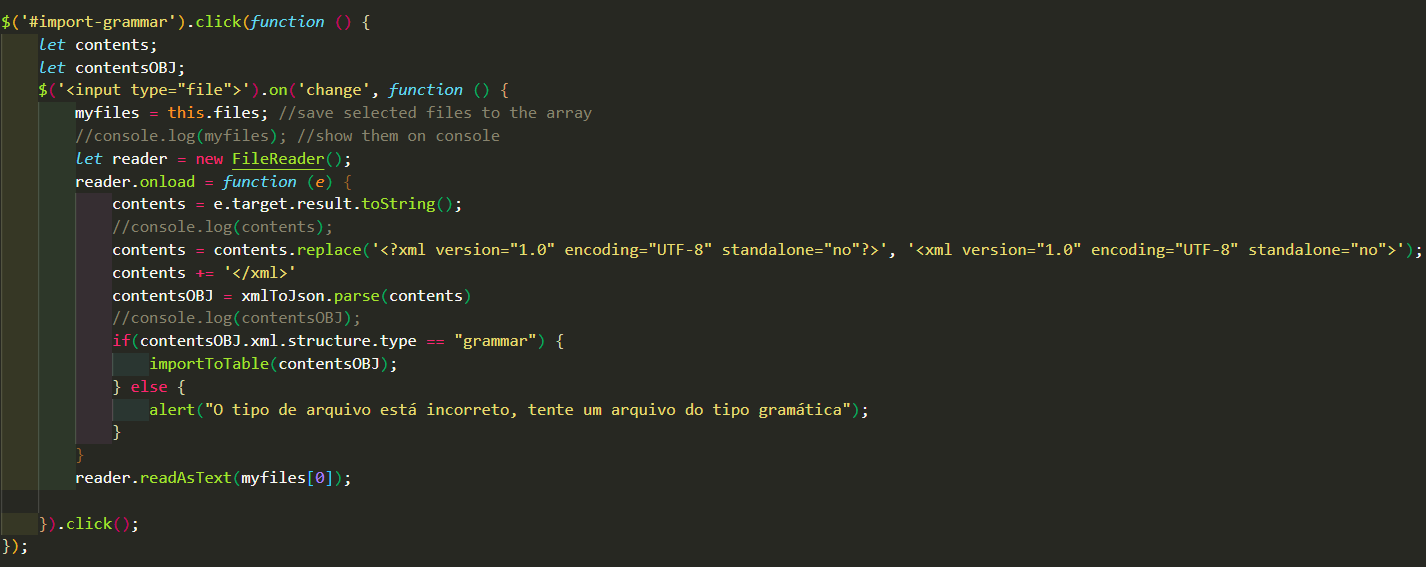


Figura 7 - importação da gramática

A importação dos autômatos funciona da mesma maneira, só que ao invés de gerar os resultados numa tabela, o resultado é criado em um canvas aonde ficam os estados, abaixo as funções da importação e exportação.

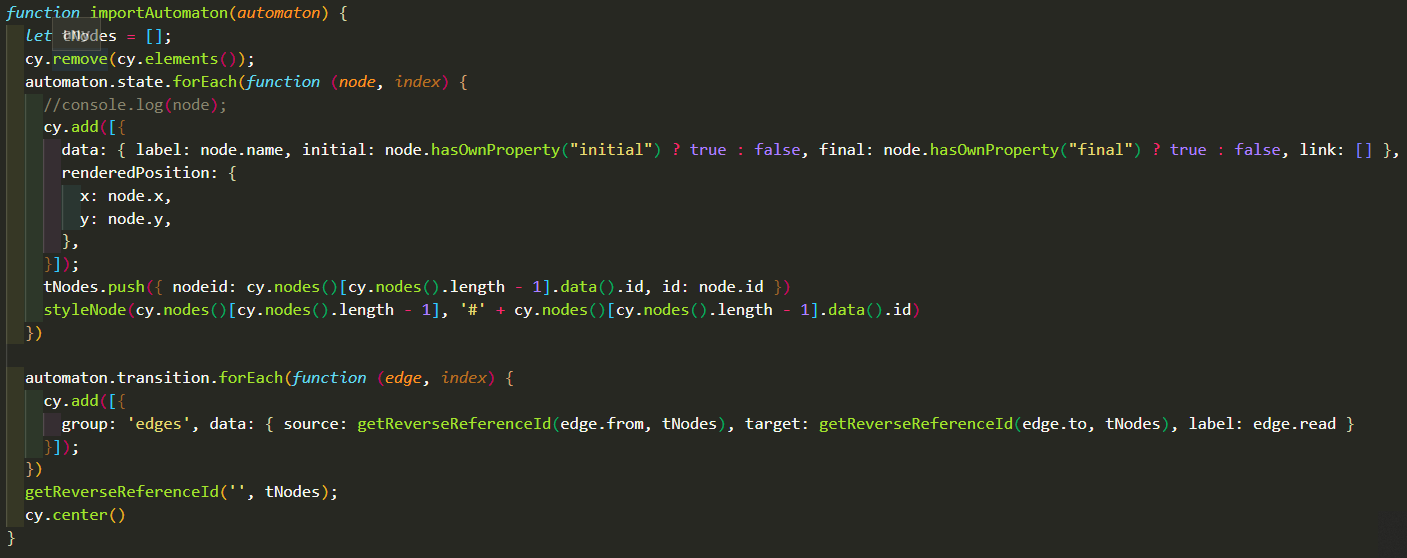


Figura 8 - Após ler o arquivo, essa função é chamada para realizar a importação

Conversão Gramática-Autômato

As conversão de gramática pra autômato é realizado na mesma lógica de conversão vista em aula, porem é utilizados objetos para gerar cada parte transitória da gramática para o autômato, são em geral três transições, gramática para tabela de produção do autômato, da tabela para uma lista com os nomes de cada autômato e na ultima parte a criação dos autômatos utilizando a biblioteca cytoscape junto com a lista e a tabela de objetos para gerara as transições e os estados.

A transformação do autômato para gramática é dada em uma serialização do autômato em XML e a biblioteca de parse para JSon e utilizar o objeto em Json para gerar um outro objeto que conterá as transições, variáveis e terminais da tabela, em seguida eles são colocados na tabela, é necessário exportar a gramática gerada para um arquivo .jff para testar depois na área de gramática.

Conversão Autômato Finito Não Determinístico - Autômato Finito Determinístico (AFND - AFD)

A conversão de autômato finito não determinístico para um autômato finito determinístico, também é realizado na mesma lógica de conversão vista em aula, usando objetos para estruturar os dados.

Os estados conforme foram encontrados ficaram salvos no array **reduce,** as arestas em **parseEdges** e o alfabeto em **alfabeto.** Após isso, fora alimentado os dados que seriam os alcances (target) dos estados para cara símbolo. Essas alterações foram inseridas nas propriedades de cada estado no array reduce. (Problema em inserir os novos estados)