FCT – UNESP Presidente Prudente

Linguagens Formais &

Autômatos

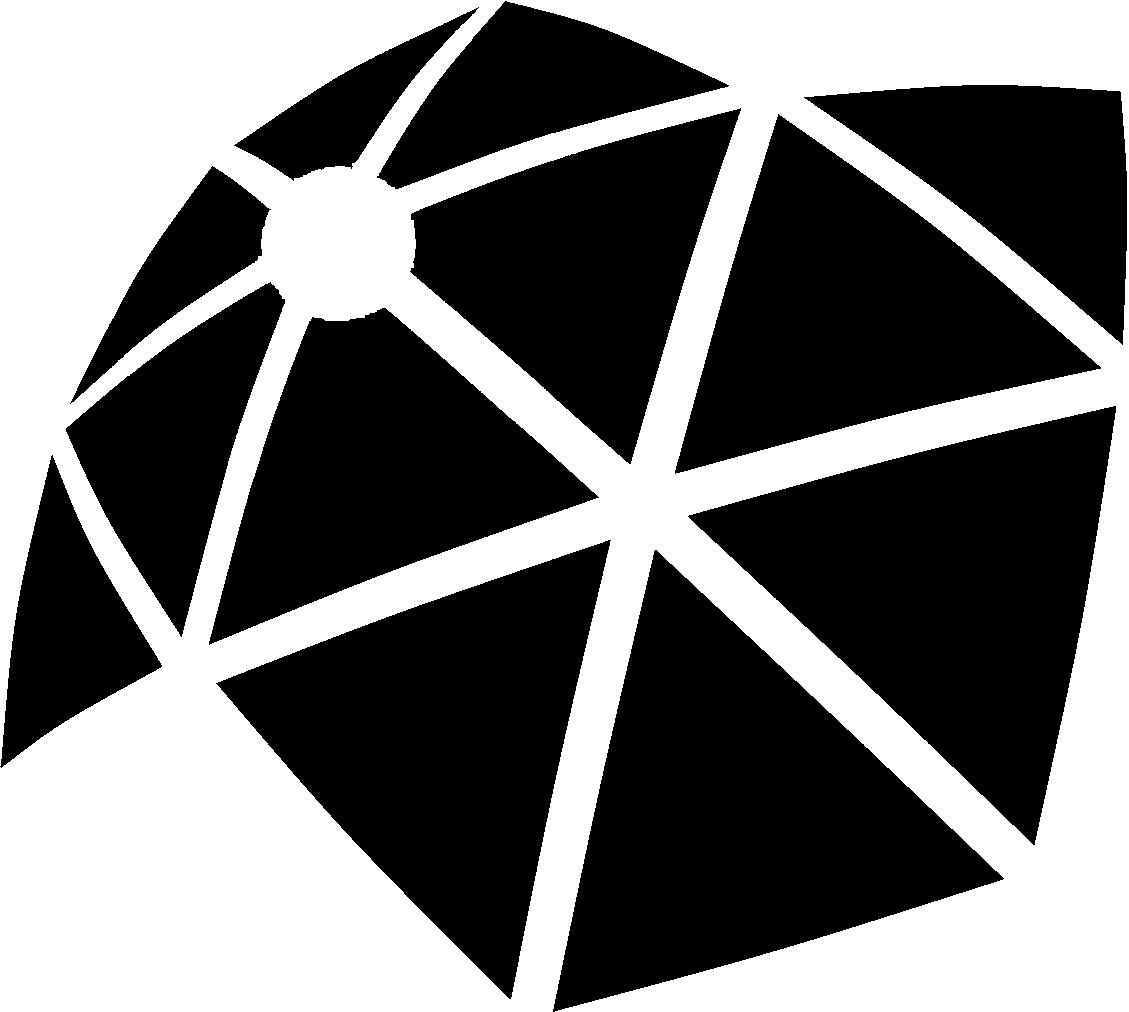
**Trabalho I**

Autômatos Finitos, Expressões Regulares e Gramática Regular

Professor: Celso Olivete Júnior.

Nome: Fabio da Silva Takaki e Lucas Martins Valladares Ribeiro

.



Presidente Prudente – SP

18 . 01 . 2017

1. **Introdução**

Fora proposto em sala e aula, a implementação da simulação de Autômatos Finitos, Gramáticas Regulares e Expressões Regulares. Foi utilizado para o desenvolvimento do trabalho a linguagem *JavaScript*, *HTML* e *CSS*. Para a criação e manipulação dos Autômatos Finitos, foi utilizado o plugin *Cytoscape*. Além disso, também utilizamos algumas ferramentas de ajuda no desenvolvimento como *node, gulp, sass e browsersync*. Por fim, para execução do trabalho abra o arquivo *index.html* dentro da pasta *app*.

1. **Construção do Dígrafo**

Para a construção do grafo, foi criado ferramentas em um formulário HTML em que há inputs do tipo radio, em que você poderá selecionar qual ferramentas utilizar.

As opções são:

* Estados: Cria novos estados dentro do canvas ao lado.
* Estado Inicial: Ao clicar em um estado criado, o mesmo se transforma em um Estado Inicial (forma de triângulo).
* Estado Final: Ao clicar em um estado criado, o mesmo se transforma em um Estado Final (forma de losango). Caso o estado clicado seja um estado Inicial, a forma é alterada para estrela para representar um estado inicial E final.
* Mudar Elemento da Transição: Ao clicar em uma transição (aresta), é possível mudar o elemento da transição.
* Deletar: Ao clicar em qualquer elemento criado, seja ele transição ou estado, o mesmo é deletado.

1. **Algoritmos**

Os algoritmos somente acontecerá após a construção do Dígrafo.

**Simulação direta, step-by-step e múltiplas entradas**

Foi utilizado um algoritmo desenvolvido pela dupla baseado na busca em profundidade realizando a técnica de *backtracking* para a verificação dos autômatos finitos.

Para utilização dos testes, a Simulação direta e o Múltiplas entradas tem seus próprios *inputs*, no qual o usuário pode inserir a *string* a ser testada. Sabendo isso, após a construção do autômato finito e ao inserir cada letra da string no input do algoritmo desejado, é executado automaticamente a simulação, colorindo a borda do input em VERDE caso sucesso. Caso contrário, a borda do input será colorida em VERMELHO.

Para a execução do step-by-step, no qual também tem seu próprio input, o usuário poderá inserir a string que deseja testar. Diferentemente da Simulação direta e Múltiplas entradas, é preciso clicar no botão ‘Iniciar’ para executar o algoritmo. Assim, aparecerá passo a passo, em uma janela, o caminho que o algoritmo percorre para testar se a entrada digitada é válida. Por fim, depois da execução step-by-step, se a entrada for válida, é colorido a borda de VERDE, caso contrário, em VERMELHO.

**Expressões regulares**

Foi utilizado a função de teste do Regexp nativa do Javascript, no qual há dois inputs de texto a serem preenchidos: a expressão regular e a string a ser testada.

Após o preenchimento da expressão regular, a cada letra digitada no input da string a ser testada, é executado a verificação se a entrada é válida ou não. Se caso válida, a borda é colorida em VERDE, caso contrário, em VERMELHO.

**Gramáticas regulares**

A implementação da aceitação ou não de gramáticas regulares, é semelhante ao algoritmo usado nas simulações em Autômatos. Elas são: GLUE e GLUD. Utiliza-se um backtracking e é baseado na busca em produndidade.

O usuário insere as regras que deseja, sendo que a primeira regra será a inicial. As entradas das regras são múltiplas e dinâmicas. Ao inserir as regras desejas, ele insere abaixo a string que desejea testar. Há dois botões para a escolha do algoritmo desejado. Caso a string seja aceita, a borda aparente é VERDE, caso contrário VERMELHO.

**Máquina de Mealy e Moore**

A implementação de aceitação ou não dos autômatos da Máquina de Mealy e Moore é também semelhante ao algoritmo usado nas simulações em Autômatos. A diferença é que não existe estado final e as duas máquinas imprimem saídas no momento da execução do algoritmo. A diferença entre as duas máquinas é que a de Mealy imprime as saídas no momento da transição entre estados, enquanto a de Moore imprime as saídas conforme a entrada em um estado. Portanto, na manipulação dos autômatos, ao criar um estado, na máquina de Moore, é pedido que o usuário digite a saída desejada que será impressa quando o algoritmo passar pelo estado criado. Já na máquina de Mealy, ao criar a transição é pedido o símbolo da transição e a saída que será impressa quando for realizada a transição.

Nas duas máquinas é possível o teste direto e com múltiplas entradas.

**Conversões**

A implementação das conversões foram aplicadas conforme os algoritmos propostos em aula.

Em ER para AF, o usuário insere no campo a expressão regular desejada e ao clicar em converter, o autômato é desenhado no quadro à direita.

Na conversão GR para AF, o usuário ao clicar em adicionar regra, o mesmo cria as regras desejadas da gramática. Assim, ao clicar em converter será desenhado o autômato finito no quadro à direita.

Na conversão AF para ER, ao ter desenhado o autômato finito no quadro à direita, o usuário clica em converter e é impresso a expressão regular abaixo do botão converter.

Por fim na conversão de AF para GR, ao ter desenhado o autômato finito no quadro à direita, o usuário clica em converter e é impresso a gramática abaixo do botão converter.

1. **Conclusão**
2. **Referências Bibliográficas**