

# **INE 5421 - Linguagem Formais e Compiladores**

## **Relatório e Manual do Trabalho 1**

Diego Almeida de Oliveira, 11100871

José Luis Bressan Ruas, 10202969

**\*\*Acesso ao programa via executável na pasta.**

**Estrutura de pastas do projeto:**

**./source/**

**./testes/**

**./executável**

### **Mini sumário**

- 1. Introdução**
- 2. Ambiente de Desenvolvimento e Tecnologias**
- 3. Implementação/Estrutura do programa**
- 4. Aprofundamento sobre as funcionalidades**
- 5. Manual do Usuário**
- 6. Plano de testes**

## **1 Introdução**

O sistema permite como entrada uma Expressão Regular e uma Gramática Regular e através destas entradas é possível utilizar todas as funcionalidades descritas no capítulo 3, funcionalidades estas oriundas da descrição do trabalho. Seguimos todos os algoritmos visto em aula, a explicação sobre a implementação deles pode ser vista também no capítulo 3.

Sobre usabilidade, tentamos inserir uma mensagem de feedback para cada ação efetuada no programa, mandando uma mensagem de aviso se a entrada for uma gramática vazia, e também avisando se a funcionalidade foi executada com sucesso e aonde ela foi executada. Também deixamos claro todos os passos ao executar uma operação que tenha AFs intermediários.

## **2 Ambiente de Desenvolvimento e Tecnologias**

A implementação do programa foi feita em C++, utilizando a IDE do Qt Creator em ambiente Linux (Ubuntu). Todas as bibliotecas necessárias para a implementação são do Qt. A interface gráfica foi feita também no ambiente da IDE.

### 3 Implementação

Sem entrar em muitos detalhes de nomes de classes, funções, etc, abaixo é explicado como alguns componentes importantes do trabalho foram implementados:

#### 1. AF

Um AF é representado por uma tabela de transições, no qual esta tabela é uma instância de uma classe de tabelas da ferramenta Qt Creator.

#### 2. GR

A GR é representada basicamente por suas produções que são armazenadas na forma de um vetor de strings, e disponibilizadas ao usuário em um campo de texto editável fornecido pelo Qt Creator

#### 3. ER

Assim como a GR, a ER é representada por uma string que é disponibilizada ao usuário em um campo de texto editável.

#### 4. Árvore Costurada

A árvore costurada foi toda implementada do zero, de forma que ficasse de acordo com a árvore usada pelo algoritmo de De Simone

### 4 Aprofundamento sobre as funcionalidades

Faremos uma abordagem top-down, por exemplo ao clicar no botão quais algoritmos são usados, qual a entrada e saída e para alguns casos somente o que faz. Seguem as explicações:

#### 1. Intersecção

Basicamente é a aplicação das propriedades de união e complemento.

$$AF1 \cap AF2 \equiv AF3 \Leftrightarrow \neg(\neg AF1 \cup \neg AF2) \equiv AF3$$

**Algoritmos necessários:** União, Complementação, Determinização e Completo.

**Entrada:** AF1 e AF2

**Saída:**  $AF1 \cap AF2$

**Saídas para o usuário:** Esta propriedade resulta em 6 AFs, iniciais, intermediários e final, de M1 a M6.

M1 = AF1

M2 = AF2

M3 =  $\neg AF1$

M4 =  $\neg AF2$

$$M5 = \neg AF1 \cup \neg AF2$$

$$M6 = \neg(\neg AF1 \cup \neg AF2)$$

## 2. Complementação

**Algoritmos necessários:** Determinização e Completo.

**Entrada:** AF

**Saída:** AF complementado

**Saída AFs para o usuário:** MX = Complementação(MX)

**Obs.:** MX, pois esta funcionalidade pode ser aplicada em qualquer uma das 9 tabelas/AFs, M1 a M9.

1.  $AF_i = \text{Determinização}(AF)$
2.  $AF2 = \text{Completo}(AF_i)$
3. Para todos os estados de AF2 verifica e seta:  
if(estado == final) estado = não final : estado == final

## 3. Equivalência

**Algoritmos necessários:** ER Gera AF. GR Gera AF. Determinização, Completo, Complementação, União, Vazio.

$$AF1 \equiv AF2 \Leftrightarrow AF1 \subseteq AF2 \wedge AF1 \supseteq AF2$$

$$\Leftrightarrow \neg(\neg AF1 \cup AF2) \equiv \emptyset \wedge \neg(\neg AF2 \cup AF1)$$

**Entrada:** ER e GR.

**Saída:** boolean, éEquivalente

**Saída AFs para o usuário:** AFs M1 a M9.

1.  $M1 = \text{ER Gera AF}$ ;  $AF1 = \text{ER}$ ; Obs.: Já é determinizado.
2.  $M2 = \text{GR Gera AF}$ ;  $AF2 = \text{GR}$ ;
3.  $M3 = \text{Complementação}(M1)$
4.  $M4 = \text{Complementação}(M2)$
5.  $M5 = M3 \cup M2$
6.  $M6 = M4 \cup M1$
7.  $M7 = \text{Complementação}(M5)$
8.  $M8 = \text{Complementação}(M6)$
9.  $M9 = M7 \cup M8$
10. éEquivalente = éVazio(M9)

## 4. Determinização

Determiniza o AF conforme o algoritmo visto em aula.

**Entrada:** AF

**Saída:** AF determinizado

**Saída AFs para o usuário:** MX = Determiniza(MX)

## 5. Minimização

Minimiza o AF conforme o algoritmo visto em aula

**Entrada:** AF

**Saída:** AF minimizado.

**Saída AFs para o usuário:**  $MX = \text{Minimiza}(MX)$

## 6. GR Gera AF

Cria uma AF usando algoritmo visto em aula, no qual a linguagem do AF corresponde à mesma linguagem da GR fornecida como entrada.

**Entrada:** GR

**Saída:** AF convertido da GR de entrada

**Saída AFs para o usuários:**  $MX | X = 1, 2, \dots, 9$  e MX é o autômato da interface de usuário que conterá o AF correspondente à conversão de GR para AF.

## 7. ER Gera AF

**Algoritmos necessários:**

Converte uma ER em um AF usando algoritmo De Simone de forma que AF reconhece sentenças cuja linguagem correspondem a mesma linguagem de ER.

**Entrada:** ER

**Saída:** AF

**Saída AFs para o usuário:**  $MX | X = 1, 2, \dots, 9$  e MX é o autômato da interface de usuário que conterá o AF correspondente à conversão de ER para AF.

**Obs:** uma ER que contém parênteses sofre uma troca de expressões dentro de parênteses por um id que representa um terminal temporário (exemplo,  $a(a|b)(b|c)$  vira  $a<1><2>$ ). A árvore costurada da expressão com os ids é criada, assim como a árvore costurada das expressões que foram substituídas por ids. Dessa forma os ids (que são nodos folhas) serão substituídos pelas raízes das árvores costuradas dos correspondentes ids

## 8. Buscar Padrão em Texto

**Algoritmos necessários:**

Algoritmo de De Simone para converter uma ER (que representa o padrão a ser buscado em um texto) em AF, e algoritmo que varre cada caractere do texto de forma que quando um caractere corresponde a uma transição válida do estado inicial do AF, uma busca por uma palavra válida (que pertence à linguagem/padrão do ER/AF) é realizada a partir daquele primeiro caractere encontrado. O processo se repete até que todo o texto seja analisado e zero ou mais padrões são encontrados.

**Entrada:** ER e Texto

**Saída:** padrões encontrados no texto

**Saída para usuário:** todos padrões encontrados e o texto com os padrões encontrados destacados em amarelo.

## 9. Salvar

Escreve a GR e a ER presentes na interface de usuário em um arquivo

## 10. Abrir

Abre um arquivo que contém uma GR e/ou uma ER e sobreescreve a GR e/ou ER da interface de usuário.

## 11. Editar GR e ER

Um campo de texto editável é disponibilizado para edição de GR e ER.

## 12. Editar AFs

Abrir, salvar, editar GR e ER; Editar AFs

**Outros (não são funcionalidades disponíveis ao usuário):**

### 1. União

Feito igual o algoritmo visto em aula.

Dado  $AF1 \cup AF2 \rightarrow AF3$  e seus respectivos estados iniciais  $s1$ ,  $s2$  e  $s3$ .

Foi feito igual o algoritmo visto em aula, é criado um estado inicial novo  $s3$ , que possui as transições de  $s1$  e  $s2$ ,  $s3$  é final se  $s1$  ou  $s2$  for final. Os outros estados e transições de  $AF1$  e  $AF2$  são copiados para  $AF3$ .

**Entrada:**  $AF1$  e  $AF2$

**Saída:**  $AF1 \cup AF2$

**Saída para o usuário:** Ao verificar um  $AF(M1$  a  $M9)$  resultante de uma união, pode-se perceber que o resultado não é modificado, não é determinizado e nem minimizado.

### 2. Completo

Transforma um AF incompleto em completo.

**Entrada:** AF

**Saída:** o AF da entrada completo

**Saída para o usuário:** Ao verificar um  $AF(M1$  a  $M9)$  resultante dessa operação, pode-se perceber que o resultado não possui transições indefinidas, e possivelmente um novo estado foi adicionado.

### 3. Vazio

Verifica se a linguagem de um AF é vazia

**Entrada:** AF

**Saída:** boolean (é vazia ou não).

## 5 Manual do usuário

Vale ressaltar que assumimos que o usuário colocará as entradas corretamente, isto é, a GR, ER e os AF's são criados de acordo com o enunciado do trabalho e com as definições vistas em aula, e não são inseridos dados incorretos. Não é feita uma validação da corretude do dado entrado pelo usuário. Se por exemplo o usuário entrar com uma ER com mais “abre parênteses” do que “fecha parênteses”, o programa terá um comportamento indeterminado. Outro caso é que todas as transições dos estados de um AF precisam conter algum valor inserido, seja um não-terminal ou pelo menos um ‘-’.

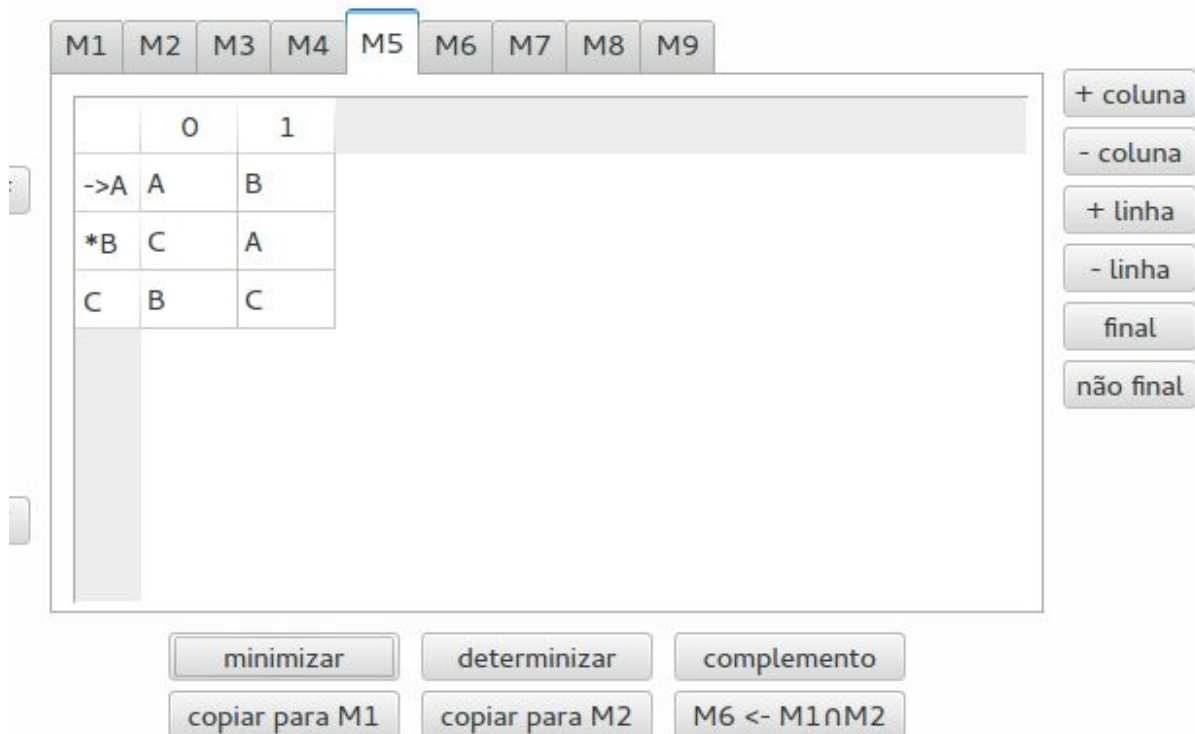
As explicações para as funcionalidades serão agrupadas por comportamento na interface.

### Determinização, complemento, minimização:



Ao clicar em um desses botões acima ele irá efetuar a operação descrita no botão para o Autômato Finito que está selecionado na interface. Exemplo: Na figura a baixo a operação será efetuada no AF M5.

### Autômato Finito



	0	1
->A	A	B
*B	C	A
C	B	C

Buttons on the right: + coluna, - coluna, + linha, - linha, final, não final.

Buttons at the bottom: minimizar, determinizar, complemento, copiar para M1, copiar para M2, M6 <- M1 ∩ M2.

Os botões da direita da figura acima servem para editar o Autômato que está sendo visualizado, na figura seria o M5 que seria editado. Para adicionar linhas, remover linhas, final e não final você deve clicar no label delas no caso no A,B ou C e depois clicar na ação que você deseja, para as colunas a mesma coisa, clicar em 0 ou 1 e depois no botão da ação.

**Intersecção:**

M6 <- M1∩M2

Ao clicar no botão da intersecção, ele fará a intersecção do M1 com M2 e colocará o resultado em M6, e em M3 a M5 colocará os AFs intermediários gerados nessa operação.

copiar para M1

copiar para M2

Os botões copiar para M1 e copiar para M2, vem como um kit para o usuário poder fazer a intersecção de AFs que foram gerados em lugares do M3 ao M9. Exemplo: Se você deseja fazer a intersecção do M5 com o M6, basta você clicar em M5 depois clicar no botão copiar para M1, clicar em M6 e clicar no botão copiar para M2, e finalmente clicar no botão da intersecção.

**Salvar:**

Para salvar, basta clicar em File->Salvar. Somente a ER e a GR são salvas em arquivos. O arquivo tem extensão .tf e é estruturado da seguinte forma:

```
<Gramatica Regular>
S->aA
A->a/aA
<Expressao Regular>
aaa*
```

**Abrir:**

Para abrir GR e/ou ER, basta clicar em File->Abrir. Qualquer arquivo que siga a estrutura e a extensão (.tf) definida acima pode ser aberto usando a função de abrir. Um detalhe importante é que o arquivo não precisa conter ambas GR e ER, somente uma delas é necessária. Dessa forma, se por exemplo o arquivo contiver somente uma ER, então a ER presente na interface será substituída pela ER do arquivo aberto, e a GR da interface não será modificada.

**GR Gera AF e ER Gera AF:**

GR->AF

ER->AF

Ao clicar em um desses botões ele irá gerar um AF da entrada para Gramática ou para Expressão regular na tabela que estiver selecionada, conforme foi explicado anteriormente.

As entradas para GR e ER podem ser vistas na figura abaixo.

As entradas para GR e ER podem ser vistas na figura abaixo.

**Gramática Regular**

A->OA|1B  
B->OC|1A|&  
C->OB|1C

GR->AF

**Expressão Regular**

(O\*(1(O1\*O)\*1))\*O\*101\*(001)\*

ER->AF

GR ≡ ER ?

### Equivalência de GR e ER:

GR ≡ ER ?

Ao clicar no botão da equivalência, ele verificará se são ou não equivalentes. O Resultado será mostrado no Feedback de Atividades(figura a baixo), que será explicado em seguida. Os AFs gerados para verificar a equivalência são 9 e eles serão colocados de M1 a M9, e no feedback de atividades será dito qual operação cada um deles representa.

### Feedback de atividades:

Ao clicar em qualquer botão das funcionalidades citadas anteriormente uma mensagem de feedback será exibida no Feedback de atividades.

**Feedback de atividades**

Autômato M5 minimizado.

Não são equivalentes

M1 = ER; M2 = GR;  
M3 = ¬M1; M4 = ¬M2;  
M5 = ¬M1UM2; M6 = ¬M2UM1;  
M7 = ¬M5; M8 = ¬M6; M9 = M7UM8

### Busca em texto:

Para acessar essa funcionalidade basta clicar na aba Busca em texto. A Utilização é bastante simples, basta inserir um texto em que você deseja buscar no campo Texto como pode ser visto na figura a baixo, uma Expressão regular em seu respectivo campo para o que você deseja buscar e então apertar no botão buscar padrões. O resultado aparecerá no Feedback de atividades e o no campo Texto o resultado será marcado em amarelo.



Obs: Todos os caracteres de um texto, incluindo o espaço, são caracteres usados na busca, porém padrões que contenham caracteres especiais usados pela ER (isto é, ()\*?|. ) não poderão ser buscados como parte do padrão em um texto.

Geral

Busca em texto

Padrão em Texto

Expressão Regular

(A casa)?possui

Texto

A casacasa possui: 1 quarto e 1 banheiro.

buscar padrão

Feedback de atividades

Padrões encontrados:  
possui

Padrões encontrados:  
possui

## 6 Plano de testes

ER -> AF:

### ER

&  
a  
a\*  
a?  
ab  
a|b

### Nome do arquivo

conversao\_ER->AF\_1.tf  
conversao\_ER->AF\_2.tf  
conversao\_ER->AF\_3.tf  
conversao\_ER->AF\_4.tf  
conversao\_ER->AF\_5.tf  
conversao\_ER->AF\_6.tf

a|b|c|d|e|f  
 abcdef  
 a\*b\*c\*d\*e\*f\*  
 a?b?c?d?e?f?  
 (abcdef)\*  
 (ab|ac)\*a?|(ba?c)\*  
 (1(01\*0)\*10\*)|0  
 ((ba)|(a(ba)\*a))\*(b|a(ba)\*)  
 0(0\*(1(01\*0)\*1)\*)\*

conversao\_ER->AF\_7.tf  
 conversao\_ER->AF\_8.tf  
 conversao\_ER->AF\_9.tf  
 conversao\_ER->AF\_10.tf  
 conversao\_ER->AF\_11.tf  
 conversao\_ER->AF\_12.tf  
 conversao\_ER->AF\_13.tf  
 conversao\_ER->AF\_14.tf  
 conversao\_ER->AF\_15.tf

GR -> AF

### GR

S->&  
 S->a  
 S->aS  
 S->aS|&  
 S->aS|bS|cS|dS  
 S->aS|aS|aS|aS

### Nome do arquivo

conversao\_GR->AF\_1.tf  
 conversao\_GR->AF\_2.tf  
 conversao\_GR->AF\_3.tf  
 conversao\_GR->AF\_4.tf

S->aB|aD|bA|bC|a|b|&  
 A->aB|bA|a  
 B->bB|aA|b  
 C->aD|bC|b  
 D->aC|bD|a

conversao\_GR->AF\_5.tf

F->aA|cC|bA|bC|&  
 S->aA|cC|bA|bC  
 A->bS|cD|b|c  
 C->bS|aE|b|a  
 D->aA|bA|bC  
 E->cC|bC|bA

conversao\_GR->AF\_6.tf

S->aA  
 A->bA|aB  
 B->bB|aB|cC  
 C->cC|bD|aE  
 D->bD|aE  
 E->aE|bE|a

conversao\_GR->AF\_7.tf

S->aA|bE|cD|a|b|c

conversao\_GR->AF\_8.tf

A->aS|bC|cB

B->bE|b

C->cD|c

D->bC

E->cB

MINIMIZAÇÃO:

**GR**

S->aB|aC|bA|bD|&

A->aB|bB

B->aA|bB

C->aC|bD|&

D->aD|bC|&

**Nome do arquivo**

minimizacao\_1.tf

S->aA|aC|aD|bA|bB|bC

minimizacao\_2.tf

A->bA|bB|&

B->aA|bB|&

C->aC|aD|&

D->aD|bC|&

S->aA|bB|&

minimizacao\_3.tf

A->aS|bC|bE

B->aA|aC|&

C->aB

D->aE|&

E->aS|aD

F->aA|cC|bA|bC|&

minimizacao\_4.tf

S->aA|cC|bA|bC

A->bS|cD|b|c

C->bS|aE|b|a

D->aA|bA|bC

E->cC|bC|bA

S->aB|aD|bA|bC|a|b|&

minimizacao\_5.tf

A->aB|bA|a

B->bB|aA|b

C->aD|bC|b

D->aC|bD|a

DETERMINIZAÇÃO:

**GR****Nome do arquivo**

S->&

determinizacao\_1.tf

S->aS|aS|aS|aS|aS|aS|bS|&

determinizacao\_2.tf

S->aS|aA|aB|aC|aD|aE

determinizacao\_3.tf

A->b

B->b

C->b

D->b

E->b

S->aB|aC|bA|bD|&

determinizacao\_4.tf

A->aB|bB

B->aA|bB

C->aC|bD|&

D->aD|bC|&

S->aA|aC|aD|bA|bB|bC

determinizacao\_5.tf

A->bA|bB|&

B->aA|bB|&

C->aC|aD|&

D->aD|bC|&

S->aA|bB|&

determinizacao\_6.tf

A->aS|bC|bE

B->aA|aC|&

C->aB

D->aE|&

E->aS|aD

F->aA|cC|bA|bC|&

determinizacao\_7.tf

S->aA|cC|bA|bC

A->bS|cD|b|c

C->bS|aE|b|a

D->aA|bA|bC

E->cC|bC|bA

S->aB|aD|bA|bC|a|b|&

determinizacao\_8.tf

A->aB|bA|a

B->bB|aA|b

C->aD|bC|b

D->aC|bD|a

## PADRÃO EM TEXTO

### Padrões buscados:

Ele (não )?disse:( não)\*

(aranha arranha)|(arranha a aranha)

O peito do pé de Pedro é (azul|preto)

### Texto:

"A aranha arranha a rã. A rã arranha a aranha. Nem a aranha arranha a rã. Nem a rã arranha a aranha.

O tempo perguntou pro tempo quanto tempo o tempo tem. O tempo respondeu pro tempo que o tempo tem tanto tempo quanto tempo o tempo tem.

O peito do pé de Pedro é preto. Quem disser que o peito do pé de Pedro é preto, tem o peito do pé mais preto do que o peito do pé de Pedro.

O peito do pé de Pedro é azul. Quem disser que o peito do pé de Pedro é azul, tem o peito do pé mais azul do que o peito do pé de Pedro.

Ele disse: não não não!"

## EQUIVALÊNCIA:

**Nome do arquivo:** eq\_1.tf

**Resultado:** São equivalentes.

**GR**

S->b

**ER**

b

**Nome do arquivo:** eq\_2.tf

**Resultado:** São equivalentes.

**GR**

S->bS|&

**ER**

b\*

**Nome do arquivo:** eq\_3.tf

**Resultado:** São equivalentes.

**GR**

A->aA|a

**ER**

aa\*

**Nome do arquivo:** eq\_4.tf

**Resultado:** São equivalentes.

**GR**

A → aB|a

B → aC|a

C → a

**ER**

a|aa|aaa

**Nome do arquivo:** eq\_5.tf

**Resultado:** São equivalentes.

**GR**

A → 0A|1B

B → 0C|1A

C → 0B|1C|&

**ER**

$(0^*(1(01^*0)^*1)^*)^*0^*101^*(001^*)^*$

**Nome do arquivo:** neq\_1.tf

**Resultado:** Não são equivalentes.

**GR**

A → aA|a

**ER**

bb\*

**Nome do arquivo:** neq\_2.tf

**Resultado:** Não são equivalentes.

**GR**

A → aB|a

B → a

**ER**

a|aa|aaa

**Nome do arquivo:** neq\_3.tf

**Resultado:** Não são equivalentes.

**GR**

A → 0A|1B

B → 0C|1A|&

C → 0B|1C

**ER**

$(0^*(1(01^*0)^*1)^*)^*0^*101^*(001^*)^*$

INTERSECÇÃO

**Nome do arquivo:** intersec\_1.tf

**GR**

A->0A|1B

B->0C|1A

C->0B|1C|&

**ER**

$(0^*(1(01^*0)^*1)^*)^*0^*101^*(001^*)^*$

**Nome do arquivo:** intersec\_2.tf

**GR**

A->&

**ER**

aa\*

**Nome do arquivo:** intersec\_3.tf

**GR**

A->aA|&

**ER**

(aa)\*

COMPLEMENTO:

**Nome do arquivo:** comp\_1.tf

**GR**

A->aA|&

**Nome do arquivo:** comp\_2.tf

**GR**

A->0A|1B

B->0C|1A|&

C->0B|1C

