Universidade Federal de Santa Catarina
INE5421- Linguagens Formais e Compiladores
Alunos: Lucas Machado da Palma 12200640
Luiz Henrique Urias de Sousa 12200647

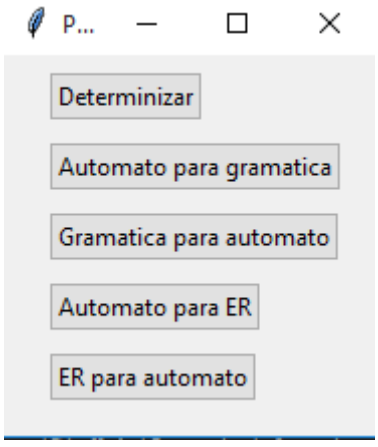
Trabalho I - Algoritmos Básicos sobre Linguagens Regulares

Este trabalho consiste na implementação dos mecanismos básicos sobre as linguagens regulares. São eles:

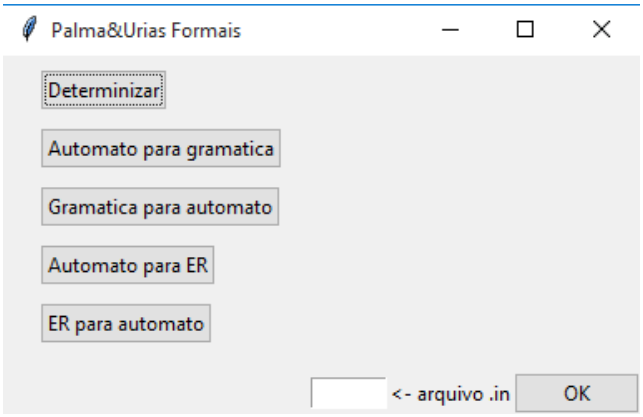
1. Determinização de um automato não determinístico (com e sem transição vazia)
2. Conversão de um automato determinístico em uma gramática
3. Conversão de uma gramática em um automato não determinístico
4. Conversão de uma expressão regular para um automato não deterministico
5. Conversão de um automato (deterministico ou não) em uma expressão regular.

Para tanto, foi utilizada a linguagem de programação Python 3. É necessário ter o sistema operacional Linux para executar o programa, seguem os passos para a execução:

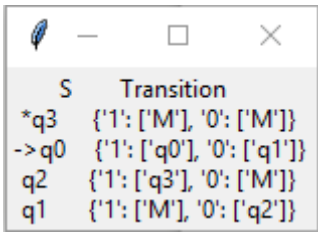
1 - Instalar Python3 em seu SO.
no Ubuntu :
sudo apt-get install python3
2 - Instalar os pacotes necessário para a interface gráfica.
no Ubuntu :
sudo apt-get install python3-tk
3 - Executar o programa.
no Ubuntu :
- ctrl + t para abrir o terminal
- caminhe até a pasta dos algoritmos do trabalho
- execute o comando: python3 gui.py



Após aparecer a tela inicial, basta clicar no que deseja fazer. O procedimento é desta forma: ao clicar em alguma das opções aparecerá um campo para inserir o arquivo de entrada do programa, como este:



A entrada deve seguir o seguinte formato: “teste1AfndAfd.in” após isso, clique em ok e será gerado um arquivo “.out” no mesmo diretório do arquivo “.in” (diretório testes), além disso, aparecerá uma tela com o resultado da entrada:



Foi utilizado um arquivo para determinização, e este foi o resultado.

Após isso, esta tela pode ser fechada e então pode-se utilizar outra opção (ou a mesma) com as devidas entradas corretas. Por exemplo, após fechar a tela, digite o nome do arquivo de entrada que deseja testar e então selecione a opção correspondente, em seguida clique em ok.

Formatos de entrada

Cada arquivo de entrada deve ser terminado com um “.in”, cada arquivo (automato, gramática ou expressão) segue um formato diferente. Os formatos são estes:

1. Automato.in:
   K = {q0, q1, q2, q3, M}
   E = (0,1)
   S = afnd1.json
   I = q0
   F = {q3}
   onde, K = o conjunto dos estados do automato
   E = alfabeto do automato
   S = o nome do arquivo .json que será utilizado para informar as transições, o formato dele será especificado abaixo
   I = o estado inicial
   F = o conjunto de estados finais

1.1 O arquivo .json:
{“estado”: {“alfabeto”: [“estadoAlcandado”, “estadoAldancado2”], “alfabeto2”: [“estadoAlcandado”]}, {“estado2”: {“alfabeto”: [“estadoAlcandado”,], “alfabeto2”: [“estadoAlcandado”]}}. Por exemplo:
{"q0": {"0": ["q1", “q2”], "1": ["q0"]}, "q1": {"0": ["q2"], "1": ["M"]}, "q2": {"0": ["M"], "1": ["q3"]}, "q3": {"0": ["M"], "1": ["M"]}, "M": {"0": ["M"], "1": ["M"]}}

Note que as strings são sempre entra aspas duplas e um alfabeto pode levar a um ou mais estados. Obs: M é o estado morto. Note também que é sempre {“estado”: {“alfabeto”: [….] }}, ...

Obs: o nome dos estados deve condizer com os nomes informados em K.

2. Grammar.in:

T = {a,b}
N = {A,B,S}
P = G1.json
I = S
onde, T = o conjunto dos terminais
N = o conjunto dos não terminais
P = o nome do arquivo .json que será utilizado para informar as produções, o formato dele será especificado abaixo
I = o estado inicial da gramática

2.2. O arquivo .json:

{“naoTerminal”: [“terminal”,”terminanaoTerminal”], “naoTerminal2” : [“...”], ...}
Este é um exemplo:
{"S": ["aA","bB","a","b"], "A": ["aA","bA","a"], "B": ["bB","aB","b"]}

Note que neste caso é diferente do .json do automato. O nome dos não terminais e terminais deve condizer com os nomes de N e T, respectivamente.

3. Expressao.in: a expressão propriamente dita, por exemplo: (a.((a|b)*).a)