Universidade Federal de Santa Catarina
Departamento de Informática e Estatística
Alunos: Paulo Barbato Fogaça de Almeida, Wesly Carmesini Ataide
Data: 07/12/2020

Para a elaboração deste trabalho usamos a linguagem Python.

Detalhes sobre a modelagem:

- Nos autômatos finitos, cada um dos estados é representado como um string. Para o conjunto de estados, o conjunto do alfabeto e o conjunto de estados finais, usamos o set do Python, que implementa justamente o conjunto, para facilmente evitar repetições. As transições são representadas com um dicionário, que a chave sendo um estado e o valor um outro dicionário em que as chaves são os símbolos do alfabeto e o valor um conjunto contendo os estados para os quais o autômato transita com o estado e o símbolo especificado. No caso do autômato finito determinístico, o conjunto pode ter apenas um elemento.
- Para as gramáticas regulares, novamente usamos os conjuntos para os não terminais e os terminais, e um dicionário para as produções, com a chave do dicionário sendo o lado esquerdo da produção e levando para um conjunto de produções possíveis.
- No caso das expressões regulares precisamos de uma abordagem mais complexa: temos uma classe ER que engloba várias definições, que são instanciadas nas classe DefReg. Como podemos ter referências entre as definições, fazemos que a classe ER trate de cuidar dessas referências, através de um dicionário que mantenha a DefReg de cada expressão.
- Quando estamos convertendo de uma ER para um AFD, precisamos criar uma árvore usando a expressão da definição. Para isso, usamos a classe Nodo, que especifica um nodo de tal árvore, incluindo seus filhos e parâmetros como firstpos e lastpos.
- Para as GLCs, novamente usamos os conjuntos para os não terminais e os terminais, e um dicionário para as produções, firsts, follows e para a tabela. Onde para as produções as chaves são os não terminais, para os firsts e follows podem ser não terminais ou terminais, e a tabela tem uma tuple (s, t), onde ‘s’ é um não terminal e ‘t’ é o um elemento do follow desse não terminal.

Detalhes de uso:

- Para se iniciar a aplicação o usuário deve rodar em python3 o ‘main.py’ contido no diretório de ‘Aplicacao’.
- Ao iniciar a aplicação será exibido no terminal os comandos presentes, sendo eles:
    - help
        - Exibe novamente esta lista de comandos
    - nomes
        - Exibe os nomes dos objetos criados e seus tipos
    - print <nome de objeto>
        - AFD ou AFND: Exibe a tabela de transições
        - GR ou GLC: Exibe as produções
        - ER: Exibe as expressões regulares
    - import <nome do objeto> <tipo> <nome do arquivo com extensão>
        - Faz a leitura do arquivo e criação do objeto com nome desejado
          os tipos são: ‘afd’, ‘afnd’, ‘gr’, ‘er’, ‘glc’
    - export <nome do objeto> <nome do arquivo com extensão>
        - Faz a escrita do arquivo do objeto desejado

        - Importante que o objeto tenha seu nome ajustado para o arquivo de saída ficar da forma especificada
    - editar <nome do objeto>
        - Exibe o menu de edição de objeto, com base no tipo de objeto.
    - metodos <nome do objeto>
        - Exibir o menu de métodos do objeto, com base no tipo de objeto.
    - sair
        - Para terminar o programa (objetos não exportados serão perdidos)

  Pode se executar os comandos sem passar argumentos, mas eles serão pedidos depois e devem ser entrados todos na mesma linha.

  Ex:
    - import afd1 afd afd1.txt
    - import
      afd1 afd afd1.txt

- O menu de editar possui as seguintes funções:
    - AF:
        - add estados <string: estado>
            - Adiciona <estado> aos estados
        - add alfabeto <string: simbolo>
            - Adiciona <símbolo> ao alfabeto
        - add transicao <string: estado antes> <string: simbolo> <string: estado depois>
            - Adiciona uma transição partindo <estado antes> por <símbolo> para <estado depois>
        - remove estados <string: estado>
            - Remove <estado> dos estados
        - remove alfabeto <string: simbolo>
            - Remove <símbolo> do alfabeto
        - remove transicao <string: estado antes> <string: simbolo> <string: estado depois>
            - Remove uma transição partindo <estado antes> por <símbolo> para <estado depois>
        - sair
            - Volta para o menu geral de comandos
        - computar <string: entrada>
            - Fala se <entrada> pertence a linguagem
        - print
            - Exibe a tabela de transições
    - GR:
        - add naoterminal <string: nao terminal>
            - Adiciona um <nao terminal> aos não terminais
        - add terminal <string: terminal>
            - Adiciona um <terminal> aos terminais
        - add producao <string: lado esquerdo> <string: lado direito>
            - Adiciona uma produção partindo de um símbolo <lado esquerdo> indo para <lado direito>
        - remove naoterminal <string: nao terminal>
            - Remove <nao terminal> dos não terminais
        - remove terminal <string: terminal>
            - Remove <terminal> dos terminais

        - remove producao <string: lado esquerdo> <string: lado direito>
            - Remove uma produção partindo de um símbolo <lado esquerdo> indo para <lado direito>
        - sair
            - Volta para o menu geral de comandos
        - derivar <int: profundidade>
            - Exibe as produções de tamanho <profundidade>
        - print
            - Exibe as produções
    - ER:
        - add <string: nome da definicao> <string: expressao>
            - Adiciona uma <definição> com <expressão>
        - remove <string: nome da definicao>
            - Remove uma <definição>
        - sair
            - Volta para o menu geral de comandos
        - print
            - Exibe as expressões
    - GLC:
        - add naoterminal <string: nao terminal>
            - Adiciona um <nao terminal> aos não terminais
        - add terminal <string: terminal>
            - Adiciona um <terminal> aos terminais
        - add producao <string: lado esquerdo> <string: lado direito>
            - Adiciona uma produção partindo de um símbolo <lado esquerdo> indo para <lado direito>
        - remove naoterminal <string: nao terminal>
            - Remove <não terminal> dos não terminais
        - remove terminal <string: terminal>
            - Remove <terminal> dos terminais
        - remove producao <string: lado esquerdo> <string: lado direito>
            - Remove uma produção partindo de um símbolo <lado esquerdo> indo para <lado direito>
        - sair
            - Volta para o menu geral de comandos
        - print
            - Exibe as produções

- O menu de métodos possui as seguintes funções:
    - AFD:
        - ajustar(NomeEstados)
            - Transforma o AFD ajustando o nome dos estados para ficar de 0 a nEstados
        - minimizar
            - Transforma o AFD o minimizando
        - converter(ParaGR)
            - Cria uma nova GR partindo do AFD
        - computar
            - Recebe uma <string: entrada> e informa se ela pertence a linguagem
        - interseccao

            - Cria um autômato da intersecção deste com outro
        - uniao
            - Cria um autômato da união deste com outro
        - print
            - Exibe a tabela de transições
        - sair
            - Volta para o menu geral de comandos
    - AFND:
        - ajustar(NomeEstados)
            - Transforma o AFD ajustando o nome dos estados para ficar de 0 a nEstados
        - computar
            - Recebe uma <string: entrada> e informa se ela pertence a linguagem
        - determinizar
            - Cria um novo AFD da determinização deste
        - print
            - Exibe a tabela de transições
        - sair
            - Volta para o menu geral de comandos
    - GR:
        - ajustar(NomeEstados)
            - Transforma a GR ajustando o nome dos estados para ficar de S a até a letra do alfabeto
        - derivar
            - Exibe as produções de tamanho <int: profundidade>
        - converter(ParaAFND)
            - Cria um novo AFND da partindo desta GR
        - print
            - Exibe as produções
        - sair
            - Volta para o menu geral de comandos
    - ER:
        - converter(ParaAFD) <string: nome da er (opcional)>
            - Cria um novo AFD da partindo desta ER ou da especificada caso tenho sido especificado o nome
        - print
            - Exibe as expressões
        - sair
            - Volta para o menu geral de comandos
    - GLC:
        - inuteis (remover simbolos inuteis)
            - Transforma a GLC removendo os símbolos inúteis
        - epsilon (remover epsilon producoes)
            - Transforma a GLC removendo as epsilon produções
        - recesq (remover recursao esquerda)
            - Transforma a GLC removendo a recursão esquerda
        - unitarias (remover producoes unitarias)
            - Transforma a GLC removendo as produções unitárias
        - fatoracao <int: profundidade maxima>

        - Transforma a GLC fatorando em até onde a recursão de para cada produção só vai até <profundidade máxima>
    - analisar <string: entrada>
        - Reconhecimento da sentença <entrada> através da tabela de análise
    - chomsky
        - Transforma a GLC para a Forma Normal de Chomsky
    - print
        - Exibe as produções
    - sair
        - Volta para o menu geral de comandos

Testes Entrega 1:

1. teste1conversaoGRAFD.txt - Converter uma GR em AFD
    1.1. import gr1 gr gr1.txt
    1.2. metodos gr1
    1.3. converter
    1.4. afnd1
    1.5. sair
    1.6. metodos afnd1
    1.7. determinizar
    1.8. afd1
    1.9. sair
    1.10. metodos afd1
    1.11. ajustar
    1.12. print
    1.13. sair
    1.14. export afd1 teste1conversaoGRAFD.txt
    1.15. sair

2. teste1conversaoGRAFND.txt - Converter uma GR em AFND
    2.1. import gr1 gr gr1.txt
    2.2. metodos gr1
    2.3. converter
    2.4. afnd1
    2.5. sair
    2.6. metodos afnd1
    2.7. ajustar
    2.8. sair
    2.9. export afnd1 teste1conversaoGRAFND.txt
    2.10. sair

3. teste1edicaoGR.txt - Editar GR
    3.1. import gr1 gr gr1.txt
    3.2. editar gr1
    3.3. print
    3.4. remove naoterminal U
    3.5. print
    3.6. add producao S aD

3.7. print
3.8. sair
3.9. export gr1 teste1edicaoGR.txt
3.10. sair

4. teste1conversaoAFGR.txt - Converter AF pra GR
   4.1. import afd1 afd afd1.txt
   4.2. metodos afd1
   4.3. converter
   4.4. grnova
   4.5. sair
   4.6. metodos grnova
   4.7. print
   4.8. ajustar
   4.9. print
   4.10. derivar
   4.11. 4
   4.12. sair
   4.13. export grnova teste1conversaoAFGR.txt
   4.14. sair

5. teste1uniao.txt - Uniao de 2 AFs
   5.1. import afd1 afd afd1.txt
   5.2. import afd2 afd afd2.txt
   5.3. metodos afd1
   5.4. uniao
   5.5. afd2
   5.6. afdu
   5.7. sair
   5.8. print afdu
   5.9. metodos afdu
   5.10. ajustar
   5.11. print
   5.12. sair
   5.13. export afdu teste1uniao.txt
   5.14. sair