ACH2043 – Introdução à Teoria da Computação

Exercício-Programa: Minimização de AFDs

Objetivo

Desenvolver um programa que minimize autômatos finitos determinísticos (AFDs) através da remoção de seus estados inacessíveis, inúteis e equivalentes. Seu programa deve receber um arquivo-texto contendo a especificação de um AFDs, e gerar um arquivo de saída contendo a especificação de um AFD equivalente minimizado.

Em outras palavras, dada uma especificação de um AFD, gere a especificação de um AFD minimizado quando remove se os estados inúteis e/ou a combinação de estados equivalentes.

Uma descrição do método de miminização de estados de AFDs e de sua implementação encontra-se em:

www.each.usp.br/lauretto/ACH2043_2017/Minimizacao_AFD.pdf

Chamada do programa e especificações dos arquivos

A chamada do programa será nas formas:

- Implementacão em C:

  minimiza.exe <FileInput> <FileOutput>

- Implementacão em Java:

  java minimiza <FileInput> <FileOutput>

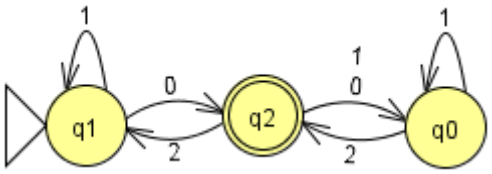
O arquivo <FileInput> deverá conter a descrição do AFD a ser minimizado, e estará organizado da seguinte forma:

- A primeira linha será um cabeçalho contendo os campos:
  n s q0,
  onde **n** é o número de estados, **s** é o número de símbolos do alfabeto e **q0** é o estado inicial. Todos esses parâmetros são inteiros maiores ou iguais a zero.

- A segunda linha conterá um vetor binário **F** com n posições, onde **F[i]** = 1 se e somente se o estado **i** for de aceitação.

- As demais linhas corresponderão à matriz de transição **Delta**.
  Essa matriz terá n linhas e s colunas, e a posição **Delta[i, j]** conterá o estado para o qual o autômato deverá ir, dado o estado **i** e o símbolo **j**.
  Quando não houver transição sobre o estado **i** com o símbolo **j**, **Delta[i, j]** = -1.

O exemplo abaixo ilustra como representar o AFD do diagrama.



Arquivo*:

```
3 3 1
0 0 1
-1  0  2
 2  1 -1
 0  0  1
```

*Espaços adicionais nas linhas 3, 4 e 5 foram inseridos apenas para melhor visualização da matriz Delta.

Observações:

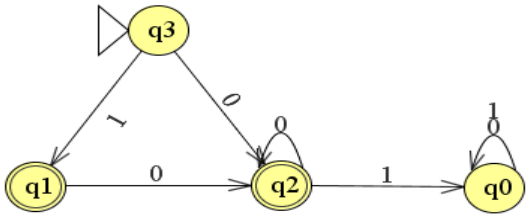
- Em cada linha, os campos serão separados por espaços (preferencialmente) ou tabulação.
- Os estados serão indexados de 0 a n-1, e os símbolos do alfabeto serão indexados de 0 a s-1.

Outros exemplos serão apresentados adiante.

O arquivo <FileOutput> será a saída de seu programa, e deverá conter a descrição do AFD minimizado. O formato desse arquivo é idêntico àquele do arquivo <FileInput>.

Os exemplos 1 e 2 ilustram dois AFDs antes e após a chamada do programa de minimização.

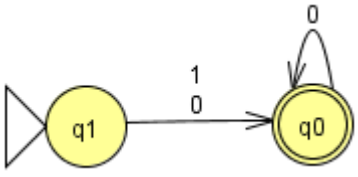
Exemplo 1:



Arquivo:

```
4 2 3
0 1 1 0
0  0
2 -1
2  0
2  1
```

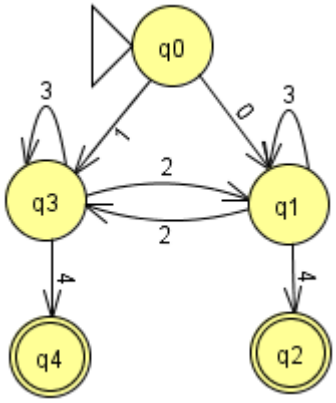
Autômato após a minimização:



Arquivo:

```
2 2 1
1  0
0 -1
0  0
```

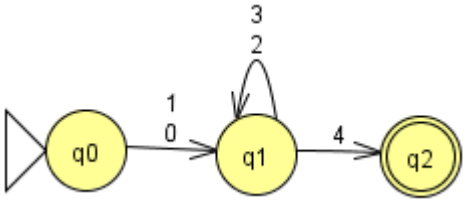
Exemplo 2:



Arquivo:

```
5 5 0
0 0 1 0 1
 1  3 -1 -1 -1
-1 -1  3  1  2
-1 -1 -1 -1 -1
-1 -1  1  3  4
-1 -1 -1 -1 -1
```

Autômato após a minimização:



Arquivo:

```
3 5 0
0 0 1
 1  1 -1 -1 -1
-1 -1  1  1  2
-1 -1 -1 -1 -1
```

Note que, por simplicidade, os índices dos estados nos AFDs minimizados não necessariamente possuem associação com os estados dos AFDs originais. No Exemplo 1, o estado q1 no AFD simplificado corresponde ao estado q3 no AFD original; e o estado q0 no AFD simplificado corresponde à classe de equivalência {q1,q2} do AFD original.

Condições da entrega:

- O trabalho poderá ser feito em duplas, devidamente identificados na primeira linha do código-fonte.
- O prazo para entrega e 09/10/2017.
- A entrega deverá ser feita através do Tidia 4.0, na ferramenta "Escaninho". O projeto deverá ser entregue em um diretório compactado (formato ZIP), nomeado da seguinte forma: d_<NUSP1>_<NUSP2>.zip, onde NUSP1 e NUSP2 são os números USP dos integrantes da dupla.
- Esse diretório deverá conter:
  - O programa-fonte (em Java ou C) com nome minimiza.c ou minimiza.java
  - A respectiva versão compilada, de nome minimiza.exe ou minimiza.class.
  - Arquivo-texto readme.txt, contendo as instruções para compilação
  - Demais módulos/classes necessários (se houver)
- O código-fonte deverá ser compilável no gcc ou no JDK (comando javac).

- Não inclua excesso de mensagens (algumas poucas são admissíveis), e principalmente, **não inclua instruções de pausa do tipo: "Tecle <enter> para continuar"**. Seu programa será testado por um testador automatizado, e essas condições inviabilizarão a correção adequada do mesmo – e portanto você terá nota próxima de zero.
- Não é necessário que os dois alunos da dupla enviem o mesmo EP, basta um envio por dupla.
- Dúvidas a respeito das especificações ou a respeito da implementacão do trabalho serão sanadas até cinco dias antes da data da entrega. Dúvidas encaminhadas após este prazo serão ignoradas.
- Além da correção do programa, será considerada a qualidade da documentação do código fonte.
- Se houver evidência de plágio entre trabalhos de grupos distintos, os mesmos serão desconsiderados.

Material suplementar:

- Na página da disciplina, encontram-se links para os seguintes materiais:
  - Arquivo **exemplos.zip**, contendo arquivos de exemplos de entrada e saída. Os arquivos com extensão .jff estão no formato do JFlap, para melhor visualização.
  - Arquivo **programas.zip**, contendo:
    - Um módulo em C denominado **afd_util.c**, seu respectivo header (**afd_util.h**)

      Este módulo contém a estrutura de dados básica para implementação de AFDs, bem como algumas funções úteis (leitura e escrita de AFDs em arquivos, alocação e liberação das estruturas, etc). Sua função *main* faz a conversão de formatos de arquivos, conforme explicado abaixo.

      O uso desse módulo não é obrigatório. Você pode aproveitá-lo conforme sua conveniência, seja usando-o diretamente como base para sua implementação do EP, ou adaptando/traduzindo parte das funções em sua própria implementação.
    - Um executável denominado **converte_afd.exe**, que é obtido pela compilação direta do módulo **afd_util.c** com o seguinte comando no prompt do DOS:

      ```
      gcc -Wall -O3 -o converte_afd.exe afd_util.c -D _STANDALONE_
      ```

      Este programa serve para converter AFDs entre o formato do JFlap e o formato texto especificado neste enunciado. É útil para:

      - Criar seus próprios exemplos no JFlap e convertê-los para o formato texto;
      - Converter a saída de seu programa

      Para detalhes sobre o uso do programa, leia os comentários no módulo **afd_util.c**.