Ghabriel Nunes e Thiago Barbato

Relatório da aplicação de manipulação de LR

1. Utilização

A aplicação foi desenvolvida utilizando JavaScript para a parte lógica e HTML/CSS para a interface gráfica. Para executar a aplicação, basta abrir o arquivo “index.html” no navegador.

A utilização é bem simples. Todas as funcionalidades do programa são acessíveis através da barra lateral à esquerda, enquanto que na seção principal os autômatos gerados são exibidos em forma de tabela de transição. A barra lateral é composta de:

- Botões “open” e “save”: para leitura e gravação em disco, respectivamente, da workspace, isto é, do conjunto de expressões regulares e autômatos que estão sendo exibidos.
- Campo de texto e botão “Add”:para adicionar expressões regulares ao workspace. A tecla “enter” também pode ser utilizada ao invés do botão “Add”. Se a expressão digitada for válida, ela será adicionada à lista de expressões e seu autômato finito determinístico correspondente será exibido na seção principal. Caso ela seja inválida, um erro será exibido.
- Lista de expressões regulares: uma lista com todas as ER presentes no workspace, juntamente com uma checkbox para cada uma. A checkbox é utilizada para seleção de expressões, possibilitando deleção (para qualquer quantidade positiva de expressões), minimização (para uma expressão de cada vez), intersecção (para duas expressões) e teste de equivalência (para duas expressões). Além disso, a forma como cada ER é mostrada depende de como foi adicionada:
    - Para ER's digitadas pelo usuário, a ER é mostrada do jeito que foi digitada.
    - Para ER's geradas a partir de uma minimização, é mostrado “[MIN] *expr*”, onde *expr* é a ER cujo autômato foi minimizado.
    - Para ER's geradas a partir de uma intersecção, é mostrado “[∩] {*expr1*, *expr2*}” onde *expr1* e *expr2* são as ER's cuja intersecção foi calculada.
    - Para ER's geradas a partir de um complemento (a partir de um teste de equivalência), é mostrado “[NOT] *expr*”, onde *expr* é a ER cujo autômato foi complementado.
- Botão de deleção: só é visível se pelo menos uma expressão estiver selecionada. Remove todas as ER's selecionadas do workspace, assim como seus respectivos autômatos.
- Botão de minimização: só é visível se exatamente uma expressão estiver selecionada. Se a ER selecionada possuir o prefixo [MIN], é exibido um erro dizendo que ela já está minimizada. Caso contrário, a forma minimizada da ER selecionada é adicionada ao workspace. O algoritmo de minimização utilizado será explicado mais adiante.
- Botão de intersecção: só é visível se exatamente duas expressões estiverem selecionadas. A intersecção das ER's selecionadas é adicionada ao workspace. O algoritmo de intersecção utilizado será explicado mais adiante.
- Botão de teste de equivalência: só é visível se exatamente duas expressões estiverem selecionadas. Quatro autômatos são adicionados ao workspace (M1 representa o autômato da primeira ER e M2 o autômato da segunda):
    - Complemento de M1, isto é, ¬M1
    - Complemento de M2, isto é, ¬M2
    - Resultado da operação M1∩¬M2
    - Resultado da operação M2∩¬M1

Além disso, o resultado do teste (se são equivalentes ou não) é exibido embaixo do botão de teste de equivalência.

2. Algoritmos utilizados

2.1. Minimização

Para a minimização de AFD's, inicialmente os estados inúteis são removidos utilizando o algoritmo padrão mostrado em aula, isto é:

- Remover os estados mortos
  - Para todo estado q do autômato:
    - Se nenhum estado aceitador é alcançável a partir de q:
      - Remova q do autômato.
- Remover os estados inacessíveis
  - Construa A = {q | q é um estado alcançável a partir de q0}
  - Para todo estado q do autômato:
    - Se q não pertence a A:
      - Remova q do autômato.

A seguir, os estados equivalentes são removidos. Para fazer isso, inicialmente são calculadas as partições de equivalência utilizando o algoritmo de Hopcroft. Esse algoritmo é apresentado a seguir:

- Entrada: um AFD M(K, Σ, δ, q0, F)
- Saída: conjunto P contendo as partições de equivalência de M
- Método:
  - Seja P = {F, K – F}
  - Seja W = {F}
  - Enquanto W não está vazio:
    - Seja A ∈ W
    - Remova A de W
    - Para cada c ∈ Σ:
      - Seja X = {q | q ∈ K e δ(q, c) ∈ A}
      - Para cada Y ∈ P:
        - Sejam I = X∩Y e D = Y–X
        - Se |I| > 0 e |D| > 0:
          - Substitua Y em P por I e D
          - Se Y ∈ W:
            - Substitua Y em W por I e D
          - Senão:
            - Se |I| <= |D|:
              - Adicione I a W
            - Senão:
              - Adicione D a W

Assim, o seguinte algoritmo é utilizado para remover partições equivalentes:

- Entrada: um AFD M(K, Σ, δ, q0, F)
- Saída: um AFD M'(K', Σ, δ', q0, F') | T(M') = T(M) e M' não possui estados equivalentes
- Método:
  - Explicite o estado de erro de M.
  - Seja P o conjunto de partições de equivalência de M, obtidas aplicando o algoritmo acima.
  - Para cada p ∈ P:

        - Enquanto |p| > 1:
            - Seja q ∈ p tal que q não seja q0
            - Remova q do autômato, substituindo as transições que vão para ele por transições que vão para outro estado em p.
    - Remova o estado de erro de M, indefinindo todas as transições que levam a ele.

A etapa “Explicite o estado de erro de M” significa substituir transições indefinidas por transições para um estado de erro φ.

Como exemplo, considere o seguinte autômato M, que já não possui estados mortos nem inacessíveis:

δ	a	b
→q0	q1	q2
*q1	q2	q3
*q2	q1	q3
q3	q0	-

Inicialmente, o estado de erro de M deve ser explicitado:

δ	a	b
→q0	q1	q2
*q1	q2	q3
*q2	q1	q3
q3	q0	φ
φ	φ	φ

A seguir, o conjunto P de partições de equivalência de M deve ser calculado. Inicialmente, temos:
P = {{q1, q2}, {q0, q3, φ}}
W = {{q1, q2}}
W não está vazio, então continuamos:
    A = {q1, q2}
    W = {}
    c = a:
        X = {q0, q1, q2}
        Y = {q1, q2}:
            I = {q1, q2}
            D = {}
            Como |D| = 0, pulamos a iteração.
        Y = {q0, q3, φ}
            I = {q0}
            D = {q3, φ}
            Como |I| > 0 e |D| > 0, continuamos:
                P = {{q1, q2}, {q0}, {q3, φ}}
                Como Y ∉ W e |I| <= |D|:
                    W = {{q0}}

c = b:
$X = \{q0\}$
$Y = \{q1, q2\}$:
$I = \{\}$
$D = \{q1, q2\}$
Como $|I| = 0$, pulamos a iteração.
$Y = \{q0\}$:
$I = \{q0\}$
$D = \{\}$
Como $|D| = 0$, pulamos a iteração.
$Y = \{q3, \varphi\}$:
$I = \{\}$
$D = \{q3, \varphi\}$
Como $|I| = 0$, pulamos a iteração.
W ainda não está vazio (pois $W = \{\{q0\}\}$), então continuamos:
$A = \{q0\}$
$W = \{\}$
c = a:
$X = \{q3\}$
$Y = \{q1, q2\}$:
$I = \{\}$
$D = \{q1, q2\}$
Como $|I| = 0$, pulamos a iteração.
$Y = \{q0\}$:
$I = \{\}$
$D = \{q0\}$
Como $|I| = 0$, pulamos a iteração.
$Y = \{q3, \varphi\}$:
$I = \{q3\}$
$D = \{\varphi\}$
Como $|I| > 0$ e $|D| > 0$, continuamos:
$P = \{\{q1, q2\}, \{q0\}, \{q3\}, \{\varphi\}\}$
Como $Y \notin W$ e $|I| <= |D|$:
$W = \{\{q3\}\}$
c = b:
$X = \{\}$
Todas as iterações de Y serão puladas pois $X \cap Y$ será sempre o conjunto vazio.

W ainda não está vazio (pois $W = \{\{q3\}\}$), então continuamos:
$A = \{q3\}$
$W = \{\}$
c = a:
$X = \{\}$
Todas as iterações de Y serão puladas pois $X \cap Y$ será sempre o conjunto vazio.
c = b:
$X = \{q1, q2\}$
$Y = \{q1, q2\}$:
$I = \{q1, q2\}$
$D = \{\}$

Como $|D| = 0$, pulamos a iteração.

Y = {q0}:

I = {}

D = {q0}

Como $|I| = 0$, pulamos a iteração.

Y = {q3}:

I = {}

D = {q3}

Como $|D| = 0$, pulamos a iteração.

Y = {φ}:

I = {}

D = {φ}

Como $|I| = 0$, pulamos a iteração.

W está vazio, então terminamos. O conjunto de partições de equivalência de M é:

P = {{q1, q2}, {q0}, {q3}, {φ}}

Agora, continuando o algoritmo principal, cuja parte restante é relembrada aqui:

- Para cada p ∈ P:
  - Enquanto $|p| > 1$:
    - Seja q ∈ p tal que q não seja q0
      - Remova q do autômato, substituindo as transições que vão para ele por transições que vão para outro estado em p.
  - Remova o estado de erro de M, indefinindo todas as transições que levam a ele.

Temos:

p = {q1, q2}:

Como $|p| > 1$, continuamos:

q = {q1}

O estado q1 é removido do autômato, substituindo as seguintes transições:

δ(q0, a) = q1

δ(q2, a) = q1

Pelas seguintes transições:

δ(q0, a) = q2

δ(q2, a) = q2

Como $|p| = 1$, encerramos o loop.

p = {q0}:

Como $|p| = 1$, não entramos no loop.

p = {q3}:

Como $|p| = 1$, não entramos no loop.

p = {φ}:

Como $|p| = 1$, não entramos no loop.

Por fim, o estado de erro de M é removido, substituindo a transição:

δ(q3, b) = φ

Por:

δ(q3, b) = –

E, assim, temos como resultado o seguinte autômato, que está minimizado:

δ	a	b
→q0	q2	q2
*q2	q2	q3
q3	q0	-

2.2. Intersecção

Para a intersecção entre AFD's, foi utilizado o seguinte algoritmo:

- Entrada: dois AFD's $M1(K1, \Sigma 1, \delta 1, q01, F1)$ e $M2(K2, \Sigma 2, \delta 2, q02, F2)$
- Saída: um AFD $M'(K', \Sigma', \delta', q0', F') \mid T(M') = T(M1) \cap T(M2)$
- Método:
  - $K' = K1 \times K2$
  - $\Sigma' = \Sigma 1 \cup \Sigma 2$
  - $\delta'((q, q'), a) = (\delta 1(q, a), \delta 2(q', a)), \ \forall q \in K1, q' \in K2, a \in \Sigma'$
  - $q0' = (q01, q02)$
  - $F' = F1 \times F2$

Como exemplo, considere os seguintes autômatos:
M1:

δ	a
→*q0	q1
q1	q0

M2:

δ	a
→*q2	q3
q3	q4
q4	q2

Assim, temos que:

- $K' = K1 \times K2 = \{(q0, q2), (q0, q3), (q0, q4), (q1, q2), (q1, q3), (q1, q4)\}$
- $\Sigma' = \Sigma 1 \cup \Sigma 2 = \{a\}$
- $q0' = (q01, q02) = (q0, q2)$
- $F' = F1 \times F2 = \{(q0, q2)\}$
- δ' é representado na tabela abaixo:

δ	a
→*(q0,q2)	(q1, q3)
q0, q3	(q1, q4)
q0, q4	(q1, q2)
q1, q2	(q0, q3)
q1, q3	(q0, q4)
q1, q4	(q0, q2)

Este AFD reconhece a intersecção de M1 e M2.