Propriedades de Linguagens Regulares III

INF05005 - Linguagens Formais e Autômatos

Instituto de Informática
Universidade Federal do Rio Grande do Sul
Porto Alegre, Brasil
http://www.inf.ufrgs.br

Direitos Autorais

Todas as imagens utilizadas nesta apresentação foram extraídas de ou inspiradas em **Menezes, Paulo Fernando Blauth. "Linguagens formais e autômatos."Porto Alegre: Instituto de Informática da UFRGS, 2005**.
O seu uso é puramente para fins didáticos.

Relembrando...

- Linguagens Regulares são fechadas para as operações de união, concatenação, complemento e intersecção

Relembrando...

- Linguagens Regulares são fechadas para as operações de união, concatenação, complemento e intersecção
- Existem algoritmos para definirmos se uma dada LR é vazia, finita ou infinita

Relembrando...

- Linguagens Regulares são fechadas para as operações de união, concatenação, complemento e intersecção
- Existem algoritmos para definirmos se uma dada LR é vazia, finita ou infinita
- Podemos decidir se duas LRs são iguais

Otimização de AF

- Uma consequência do teste de igualdade de LRs é a possibilidade de substituição de um dado AFD/algoritmo por outro que aceita a mesma linguagem

Otimização de AF

- Uma consequência do teste de igualdade de LRs é a possibilidade de substituição de um dado AFD/algoritmo por outro que aceita a mesma linguagem
- Seguindo esta ideia, podemos imaginar que exista um AFD mais "otimizado" que possamos usar em lugar de algum autômato mais complexo

Otimização de AF (cont.)

- Neste contexto, "otimizado" se refere a um número menor de estados

Otimização de AF (cont.)

- Neste contexto, "otimizado" se refere a um número menor de estados
- Com este propósito, precisamos de um algoritmo para:

Otimização de AF (cont.)

- Neste contexto, "otimizado" se refere a um número menor de estados
- Com este propósito, precisamos de um algoritmo para:
  1. Unificar estados equivalentes

Otimização de AF (cont.)

- Neste contexto, "otimizado" se refere a um número menor de estados
- Com este propósito, precisamos de um algoritmo para:
  1. Unificar estados equivalentes
  2. Eliminar estados inúteis

Algoritmo de Minimização

- Requisitos:

Algoritmo de Minimização

- Requisitos:
  1. AF determinístico

Algoritmo de Minimização

- Requisitos:
  1. AF determinístico
  2. Somente estados alcançáveis a partir do estado inicial

Algoritmo de Minimização

- Requisitos:
  1. AF determinístico
  2. Somente estados alcançáveis a partir do estado inicial
  3. Função programa total

- Pode ser dividido no seguintes passos:

Algoritmo de Minimização (cont.)

- Pode ser dividido no seguintes passos:
  1. Teste de equivalência de estados

Algoritmo de Minimização (cont.)

- Pode ser dividido no seguintes passos:
  1. Teste de equivalência de estados
  2. Unificação de estados equivalentes

Algoritmo de Minimização (cont.)

- Pode ser dividido no seguintes passos:
  1. Teste de equivalência de estados
  2. Unificação de estados equivalentes
  3. Exclusão de estado inúteis

Algoritmo de Minimização (cont.)

- Pode ser dividido no seguintes passos:
  1. Teste de equivalência de estados
  2. Unificação de estados equivalentes
  3. Exclusão de estado inúteis
  4. Construção do AF minimizado

Equivalência de Estados

- Seja $M = (\Sigma, Q, \delta, q_0, F)$ um AFD que reconhece uma dada LR

Equivalência de Estados

- Seja $M = (\Sigma, Q, \delta, q_0, F)$ um AFD que reconhece uma dada LR
- Dois estados p e q são equivalentes se, para toda palavra $w \in \Sigma^*$, $\delta^*(p, w) \in F$ sse $\delta^*(q, w) \in F$

Equivalência de Estados

- Seja $M = (\Sigma, Q, \delta, q_0, F)$ um AFD que reconhece uma dada LR
- Dois estados p e q são equivalentes se, para toda palavra $w \in \Sigma^*$, $\delta^*(p, w) \in F$ sse $\delta^*(q, w) \in F$
- Se p e q não são equivalentes, ele são ditos distinguíveis

Equivalência de Estados

- Seja $M = (\Sigma, Q, \delta, q_0, F)$ um AFD que reconhece uma dada LR
- Dois estados p e q são equivalentes se, para toda palavra $w \in \Sigma^*$, $\delta^*(p, w) \in F$ sse $\delta^*(q, w) \in F$
- Se p e q não são equivalentes, ele são ditos distinguíveis
- Ou seja, a equivalência de estados é baseada no comportamento observável (palavras aceitas/rejeitadas a partir dos estados)

Equivalência de Estados (cont.)

- Teste é feito para cada par de estados distintos $\{p, q\}$

Equivalência de Estados (cont.)

- Teste é feito para cada par de estados distintos $\{p, q\}$
- Usa-se uma tabela de distinções para identificar estados equivalentes

q_1					
q_2					
...					
q_n					
d					
	q_0	q_1	...	q_{n-1}	q_n

Equivalência de Estados (cont.)

- Marcação da tabela de distinções ocorre com auxílio de listas de dependências para cada célula da tabela:

Equivalência de Estados (cont.)

- Marcação da tabela de distinções ocorre com auxílio de listas de dependências para cada célula da tabela:

  1. Para todo estado $q \in F$, **marcam-se** os pares $\{p, q\}$ onde $p \notin F$

Equivalência de Estados (cont.)

- Marcação da tabela de distinções ocorre com auxílio de listas de dependências para cada célula da tabela:

  1. Para todo estado $q \in F$, **marcam-se** os pares $\{p, q\}$ onde $p \notin F$

  2. Para cada par $\{p, q\}$ não marcado no passo anterior, supondo $\delta(p, a) = r$ e $\delta(q, a) = s$ para $a \in \Sigma$

Equivalência de Estados (cont.)

- Marcação da tabela de distinções ocorre com auxílio de listas de dependências para cada célula da tabela:

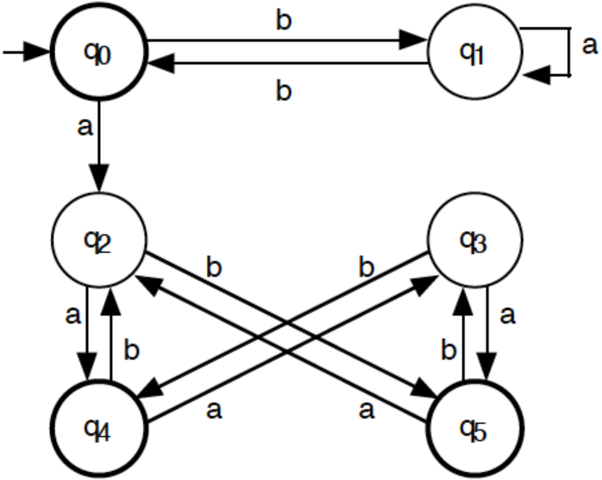
1. Para todo estado $q \in F$, **marcam-se** os pares $\{p, q\}$ onde $p \notin F$
2. Para cada par $\{p, q\}$ não marcado no passo anterior, supondo $\delta(p, a) = r$ e $\delta(q, a) = s$ para $a \in \Sigma$
   - Se $\{r, s\}$ **não está marcado**: incluir $\{p, q\}$ na lista de dependências de $\{r, s\}$

Equivalência de Estados (cont.)

- Marcação da tabela de distinções ocorre com auxílio de listas de dependências para cada célula da tabela:

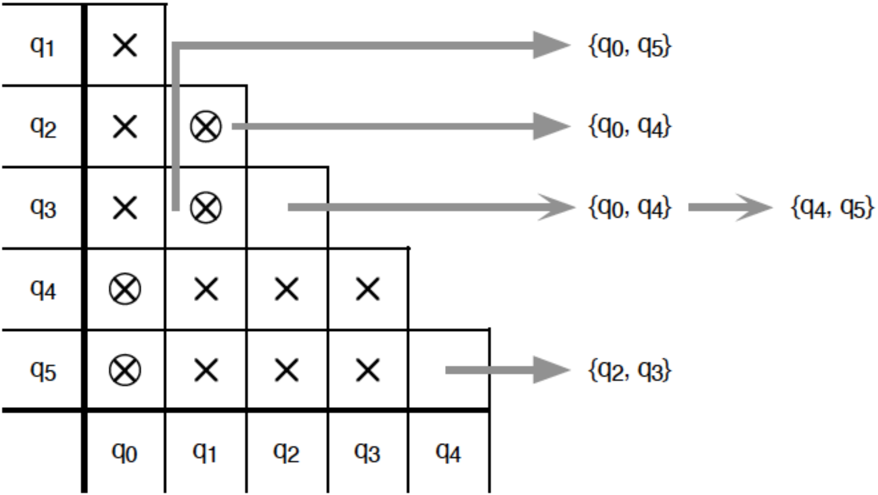
  1. Para todo estado $q \in F$, **marcam-se** os pares $\{p, q\}$ onde $p \notin F$
  2. Para cada par $\{p, q\}$ não marcado no passo anterior, supondo $\delta(p, a) = r$ e $\delta(q, a) = s$ para $a \in \Sigma$
     - Se $\{r, s\}$ **não está marcado**: incluir $\{p, q\}$ na lista de dependências de $\{r, s\}$
     - Se $\{r, s\}$ **está marcado**: (i) marcar $\{p, q\}$; (ii) se $\{p, q\}$ possui uma lista de dependências, marcar todos os pares da lista (e, recursivamente, aqueles pares das listas de dependências destes pares)

Equivalência de Estados (cont.)



Equivalência de Estados (cont.)

	q_0	q_1	q_2	q_3	q_4
q_1	×				
q_2	×				
q_3	×				
q_4		×	×	×	
q_5		×	×	×	



q1
q2
q3
q4
q5
q0
q1
q2
q3
q4
{q0, q5}
{q0, q4}
{q0, q4}
{q4, q5}
{q2, q3}

Unificação de Estados Equivalentes

- Todos os pares de estados da tabela que não ficarem marcados são equivalentes

Unificação de Estados Equivalentes

- Todos os pares de estados da tabela que não ficarem marcados são equivalentes
- Tais pares podem ser unificados, gerando um novo estado que representa os dois originais (que são eliminados do AFD)

Unificação de Estados Equivalentes

- Todos os pares de estados da tabela que não ficarem marcados são equivalentes
- Tais pares podem ser unificados, gerando um novo estado que representa os dois originais (que são eliminados do AFD)
- No nosso exemplo, q_{23} representa a unificação dos estados q_2 e q_3 e q_{45} representa a unificação dos estados q_4 e q_5

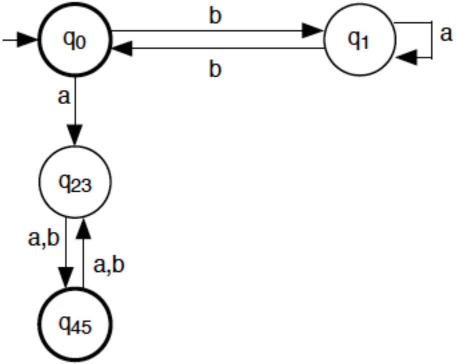
Exclusão de Estados Inúteis

- Um estado q é dito inútil se:
  - $q \notin F$
  - Nenhum estado $p \in F$ pode ser atingido a partir de q

Exclusão de Estados Inúteis

- Um estado q é dito inútil se:
  - $q \notin F$
  - Nenhum estado $p \in F$ pode ser atingido a partir de q
- Todo estado que atender a estes requisitos deve ser excluído do AFD

Construção do AF Minimizado



Autômato Mínimo

- O algoritmo de minimização **sempre** gera um autômato mínimo $M_{min} = (\Sigma, Q_{min}, \delta_{min}, q_{min}, F_{min})$ tal que $|Q_{min}| \leq |Q|$ e $ACEITA(M_{min}) = ACEITA(M)$

Autômato Mínimo

- O algoritmo de minimização **sempre** gera um autômato mínimo $M_{min} = (\Sigma, Q_{min}, \delta_{min}, q_{min}, F_{min})$ tal que $|Q_{min}| \leq |Q|$ e $ACEITA(M_{min}) = ACEITA(M)$
- Além disso, o autômato mínimo é ÚNICO para cada LR L (a menos de isomorfismo)

Autômato Mínimo

- O algoritmo de minimização **sempre** gera um autômato mínimo $M_{min} = (\Sigma, Q_{min}, \delta_{min}, q_{min}, F_{min})$ tal que $|Q_{min}| \leq |Q|$ e $ACEITA(M_{min}) = ACEITA(M)$

- Além disso, o autômato mínimo é ÚNICO para cada LR L (a menos de isomorfismo) $\Rightarrow$ Para toda LR L existe um AF M que possui um número de estados menor ou igual a qualquer outro AF equivalente.

Autômato Mínimo (cont.)

- Também podemos usar o algoritmo de minimização para testar equivalência de LR:

Autômato Mínimo (cont.)

- Também podemos usar o algoritmo de minimização para testar equivalência de LR:
  - Se os estados iniciais dos autômatos forem equivalentes, então as autômatos aceitam a mesma LR