Teoria da Computação

Relação entre AFD e AFN

Thiago Alves

Introdução

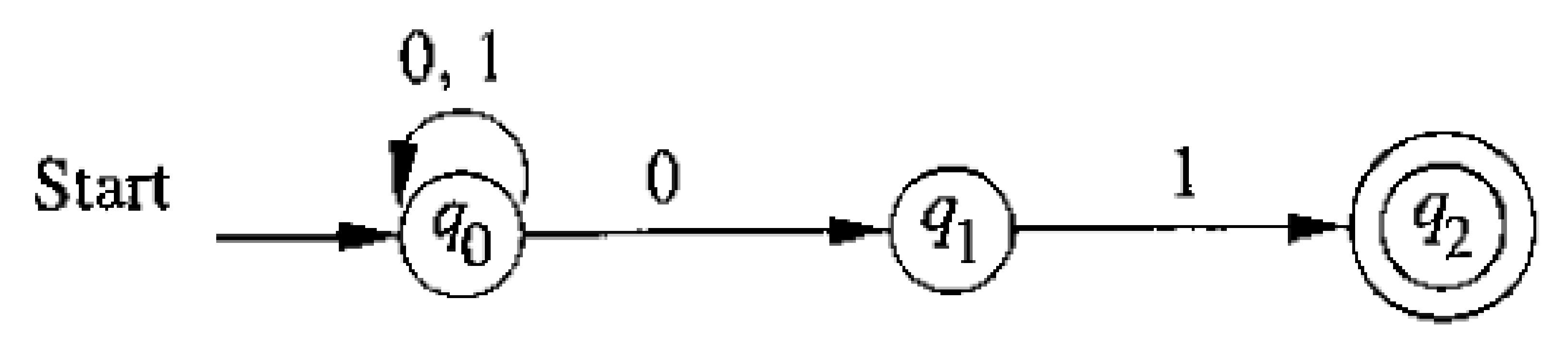
- Será que os AFN são mais poderosos que os AFD?
- Ou seja, será que existe um AFN tal que não existe um AFD que reconheça a mesma linguagem?

Introdução

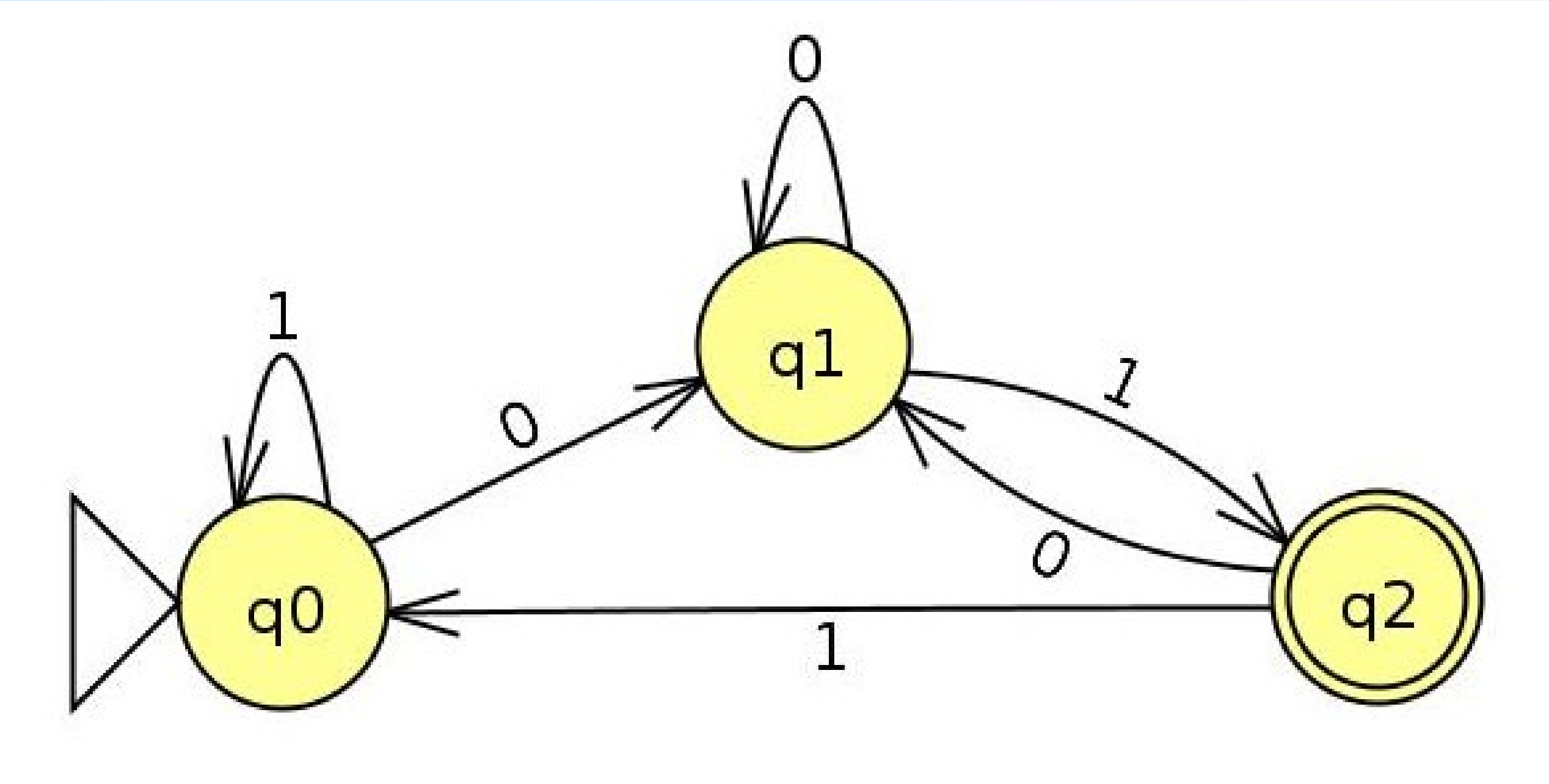
- Vamos comparar os AFNs com os AFDs tentando construir um AFDs equivalentes aos AFNs vistos
- São equivalentes se aceitam as mesmas strings

Introdução

- Como seria um AFD equivalente ao AFN abaixo?

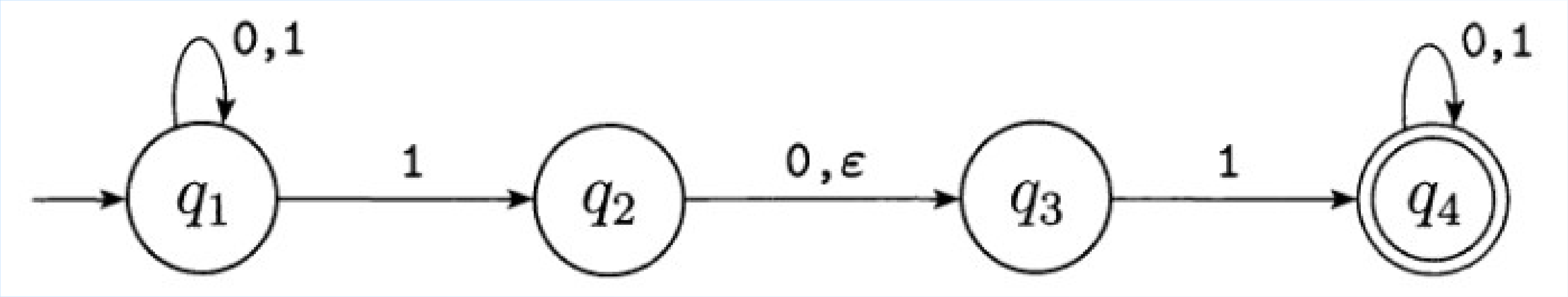


Introdução

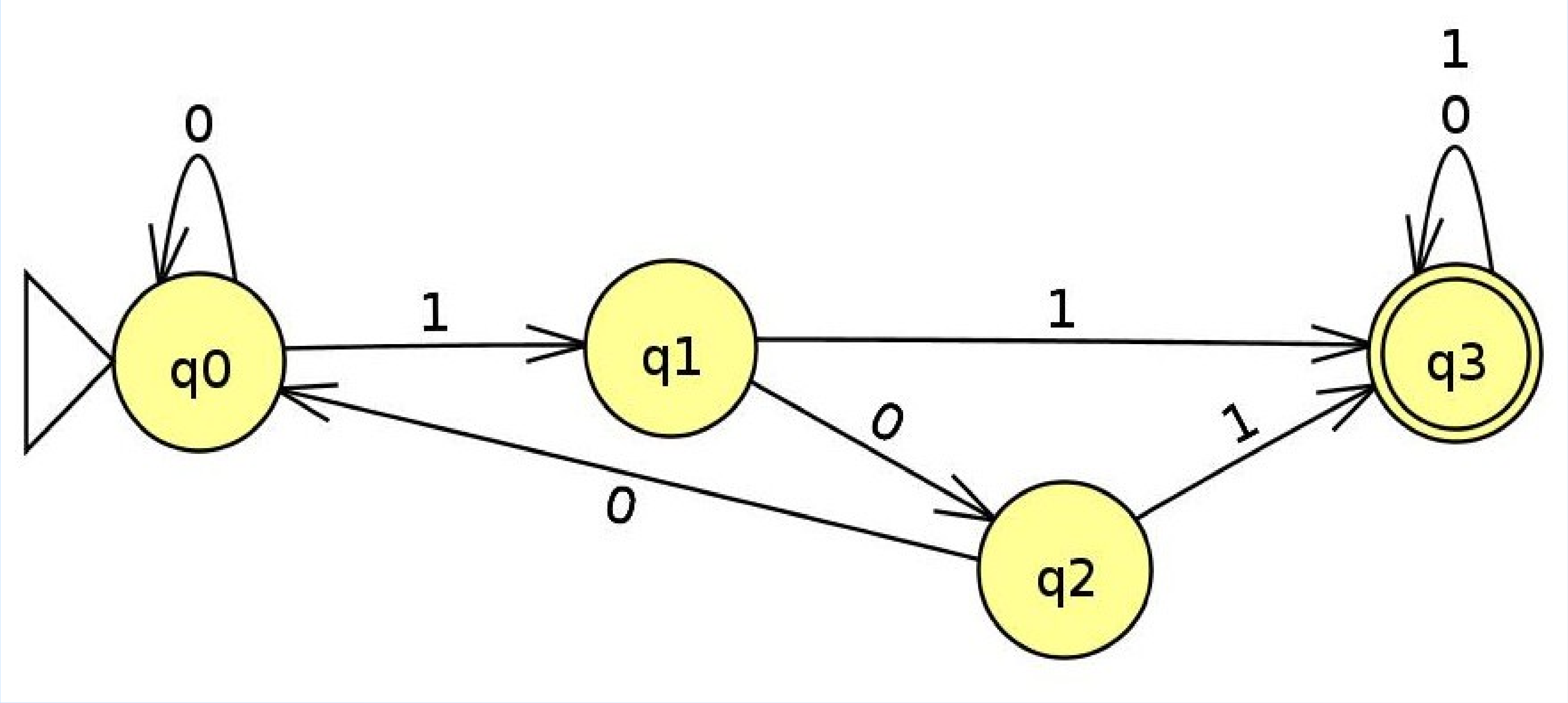


Introdução

- Como seria um AFD equivalente ao AFN abaixo?



Introdução



Introdução

- Mostrar que para todo AFD A, existe um AFN N tal que L(A)=L(N)
- Mostrar que para todo AFD N, existe um AFD A tal que L(A)=L(N)
- Um dos dois é fácil de mostrar
  - Qual?

AFD → AFN

- Um AFD pode ser convertido em um AFN que aceita a mesma linguagem
- Seja um AFD A com seus estado, alfabeto, estado inicial, estados finais e função de transição δ_D
- Como podemos converter em um AFN?

AFD → AFN

- Só precisamos nos preocupar com a função de transição
- Se $\delta_D(q, a) = p$
  - Definimos a função de transição do AFN como $\delta_N(q, a) = \{p\}$
- O AFN está sempre em um conjunto contendo exatamente um estado
  - O estado em que o AFD está depois de processar a mesma string de entrada

AFN → AFD

- A cada transição, o AFN pode estar em vários estados ao mesmo tempo
- Como simular isso em um AFD?

AFN → AFD

- A cada transição, o AFN pode estar em vários estados ao mesmo tempo
- Como simular isso em um AFD?
- Os estados do AFD podem ser conjuntos de estados
- Cada estado que é um estadão simula o conjunto de estado no AFN

AFN → AFD

- *Construção de subconjuntos*
- O número de estados de um AFD pode ser exponencial no número de estado do AFN

Construção de Subconjuntos

- Seja um AFN com estados em Q, alfabeto Σ, função de transição δ_N, estado inicial q_0 e estado finais em F, construímos um AFD:
  - Estados 2^Q (Conjunto das partes de Q)
  - Alfabeto Σ
  - Estado inicial ε-closure($\{q_0\}$)
  - Estados finais: Conjuntos com algum membro em F

Observação

- Para facilitar o entendimento, podemos nomear os estados do AFD
  - Cada conjunto de estados do AFN pode ser rotulado com letras maiúsculas do alfabeto A, B, C, ...
- Cada conjunto de estados do AFN é apenas um elemento no AFD
  - Devem ser entendidos como um único símbolo

Construção de Subconjuntos

- A função de transição δ_D é definida por:

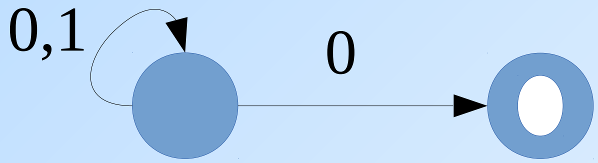
$\delta_D(\{q_1,\ldots,q_k\}, a)$ é o ε-closure da união dos $\delta_N(q_i, a)$, para i de 1 até k

$\delta_D(\{q_1,\ldots,q_k\}, a) =$

$\varepsilon\text{-closure}(\bigcup_{i \in \{1,\ldots,k\}} \delta_N(q_i, a))$

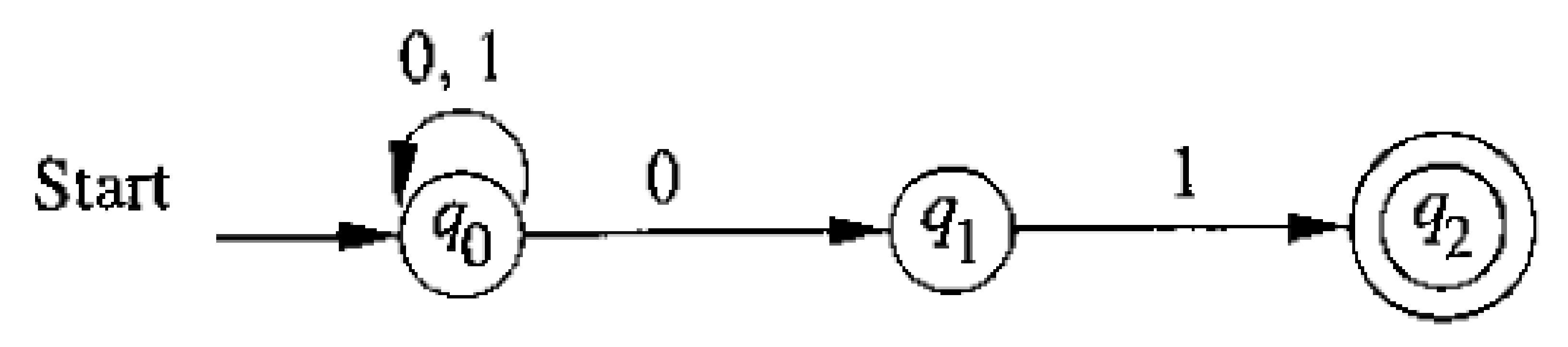
Construção de Subconjuntos

- Exemplo



Construção de Subconjuntos

- Exemplo



Construção de Subconjuntos

- Claramente podemos remover os estados que não são acessíveis a partir do inicial
- Nem sempre a construção converte um AFN em um AFD com número exponencial de estados com relação ao número de estado no AFN

Construção de Subconjuntos

- Podemos começar apenas com o conjunto de estados que representa o estado inicial do AFD
- Depois construir as transições dos acessíveis a partir desses e assim por diante

Prova da Construção

- Intuição: depois de ler uma string w, o AFD vai estar em um estado que é o conjunto de estados em que o AFN estaria depois de ler w
- Mostrar por indução em |w| que

$\delta^*_N(q_0, w) = \delta^*_D(\varepsilon\text{-closure}(\{q_0\}), w)$

- Base: $w = \varepsilon$: $\delta^*_N(q_0, \varepsilon) = \delta^*_D(\{q_0\}, \varepsilon) = \varepsilon\text{-closure}(\{q_0\})$

Indução

- Hipótese de Indução
  - Para toda string x tal que |w|>|x|>= 0, $\delta^*_N(q_0, x) = \delta^*_D(\varepsilon\text{-closure}(\{q_0\}), x) = \{p_1, \ldots, p_k\}$
- Passo de Indução
  - Seja w = xa
  - $\delta^*_N(q_0, xa) =$
  - $\varepsilon\text{-closure}(U_{p \in \delta N^*(q0,\, x)} \delta_N(p, a))$

Indução

- Passo de Indução
  - $\delta^*_D(\varepsilon\text{-closure}(\{q_0\}), xa) =$
  - $\delta_D(\delta^*_D(\varepsilon\text{-closure}(\{q_0\}), x), a) =$
  - $\delta_D(\delta^*_N(q_0, x), a) =$
  - $\varepsilon\text{-closure}(\cup_{p \in \delta_N^*(q0,\ x)} \delta_N(p, a))$
  - Logo, $\delta^*_N(q_0, w) =$
    $\delta^*_D(\varepsilon\text{-closure}(\{q_0\}), w)$

Resumo

- AFD's e AFN's são equivalentes
- Construir um AFN para uma linguagem é mais simples que construir um AFD
  - Além disso, pode ter bem menos estados que o AFD
- AFD é mais simples de implementar

Resumo

- Uma linguagem é regular se e somente se é aceita por um AFD
- Todo AFD tem um AFN equivalente
- Todo AFN tem um AFD equivalente

Teorema

- Uma linguagem é regular se e somente se é aceita por um AFN
  - Se uma linguagem é regular então tem um AFD. Se tem um AFD então tem um AFN
  - Se uma linguagem é aceita por um AFN então possui um AFD equivalente. Se possui um AFD então é regular.