Linguagens, autômatos e gramáticas

Teoria da computação

Prof. Allan Rodrigo Leite

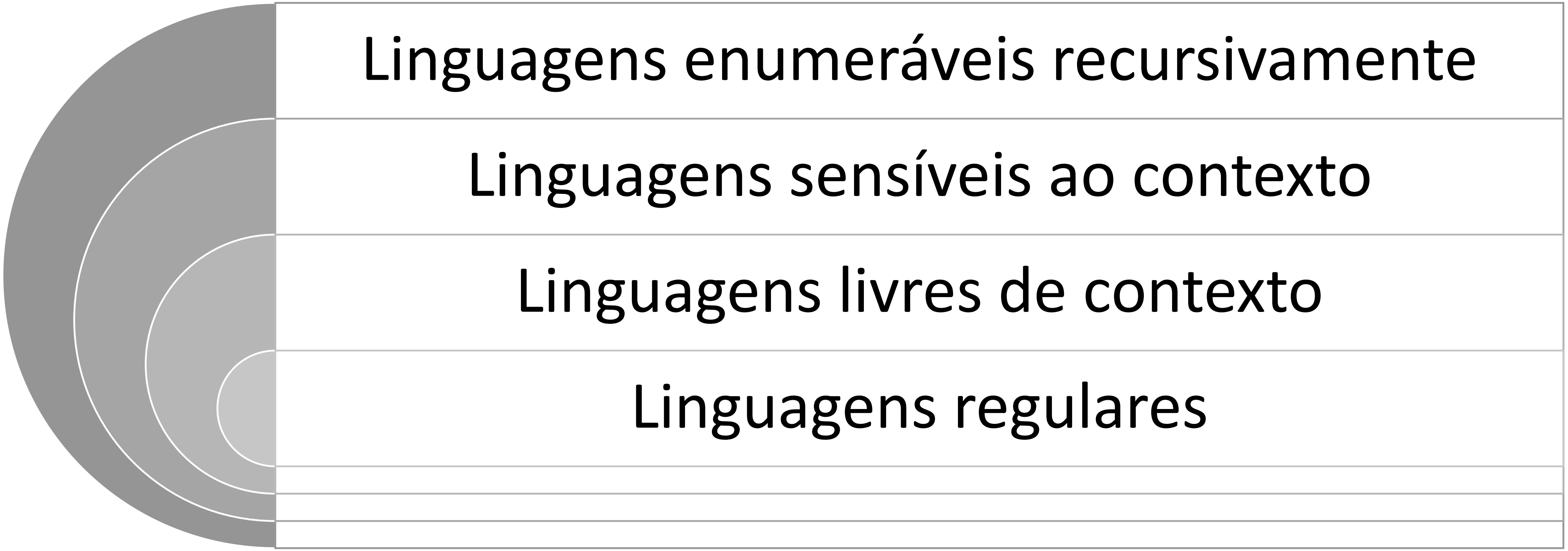
Linguagens, autômatos e gramáticas

- Linguagem é um conceito fundamental na teoria da computação
  - Deve ser capaz de expressar problemas de forma precisa
  - Permite o desenvolvimento formal ao estudo da computabilidade

- Exemplo
  - Analisar a existência de um algoritmo que determine se uma palavra pertence ou não à linguagem (objeto de estudo)
  - Determinar se uma palavra pertence ou não (solução)
  - Uma palavra (entrada)
  - Linguagem (definição do problema)

Linguagens, autômatos e gramáticas

- Hierarquia de Chomsky (1959)



Alfabeto

- Alfabeto
  - Representa um conjunto finito de símbolos ou caracteres
  - Conjunto infinito não é um alfabeto
  - Conjunto vazio é um alfabeto

- Exemplos de alfabeto
  - `{ a, b, c }`
  - Ø (conjunto vazio)

- Não são exemplos de alfabeto
  - `{ a,b,aa, ab, ba, bb, aaa, ... }`
  - N (conjunto dos números naturais)

Símbolos e palavras

- Cadeia de símbolos
  - Sequência de zero ou mais símbolos (do conjunto) justapostos

- Palavra
  - Cadeia de símbolos finita
  - Comprimento ou tamanho de uma palavra w é representado por $|w|$
  - Ou seja, $|w|$ corresponde ao número de símbolos que compõe a palavra w

Símbolos e palavras

- Palavra (cont.)
  - ε representa uma cadeia vazia ou palavra vazia
  - Σ representa um alfabeto
  - Σ^* representa todas as palavras do alfabeto Σ
  - Σ^+ é o conjunto não vazio de palavras do alfabeto Σ, isto é, $\Sigma^* - \{ \varepsilon \}$

- Exemplo
  - abcb é uma palavra sobre o alfabeto { a, b, c }
  - Se $\Sigma = \{ a, b \}$, então
    - $\Sigma^+ = \{ a, b, aa, ab, ba, bb, aaa, ... \}$
    - $\Sigma^* = \{ \varepsilon, a, b, aa, ab, ba, bb, aaa, ... \}$
  - |abcb| = 4

Símbolos e palavras

- Palavra (cont.)
  - Prefixo é qualquer sequência inicial de símbolos de uma palavra
  - Sufixo é qualquer sequência final de símbolos de uma palavra
  - Qualquer prefixo ou sufixo de uma palavra é uma subpalavra

- Exemplo
  - Em relação à palavra abcb, temos
    - ε, a, ab, abc, abcb são os prefixos
    - ε, b, cb, bcb, abcb são os sufixos

Linguagem formal

- Linguagem formal
  - Descreve o conjunto de palavras de um alfabeto
  - Linguagem formal ou L sobre um alfabeto Σ ($L \subseteq \Sigma^*$)

- Exemplo
  - Dado alfabeto $\Sigma = \{ a, b \}$, exemplos de palíndromos são:
    - $\{ \varepsilon, a, b, aa, bb, aaa, aba, bab, bbb, aaaa, ... \}$
    - O exemplo acima representa as palavras desta linguagem

Linguagem formal

- Concatenação de palavras
  - Operação binária definida sobre uma linguagem
  - Associa a cada par de palavras uma palavra formada pela justaposição da primeira com a segunda

- Operação de concatenação seguem as propriedades:
  - Associatividade
    - $v(wt) = (vw)t = vwt$
  - Elemento neutro à esquerda ou direita
    - $\varepsilon w = w = w\varepsilon$

Linguagem formal

- Concatenação de palavras (cont.)
  - Suponha o alfabeto $\Sigma = \{ a, b \}$
  - Para as palavras $v = baaaa$ e $w = bb$, temos:
    - $vw = baaaabb$
    - $v\varepsilon = v = baaaa$
  - Suponha a linguagem L de palíndromos sobre o alfabeto $\Sigma = \{ a, b \}$
    - A concatenação das palavras aba e bbb resulta em $ababbb$
    - $ababbb$ não é palíndromo
    - Portanto, a operação de concatenação não é fechada sobre L

Linguagem formal

- Concatenação sucessiva de palavras
  - Representada por um expoente após a palavra
  - Dada a palavra p, uma concatenação sucessiva é dada por p^n onde n representa o número de concatenações sucessivas
  - $p^0 = \varepsilon$
  - $p^n = pp^{n-1}$, para $n > 0$

- Exemplo
  - $w^3 = www$
  - $w^1 = w$
  - $a^n = aaaaa \; ... \; a$ (o símbolo a repetido n vezes)

Linguagem formal

- Concatenação de linguagens
  - A concatenação também vale para as linguagens
  - L_1 = { a, bc }
  - L_2 = { aa, cb, bb }
  - L_1 o L_2 = { x o y | x ∈ L_1 e y ∈ L_2 }
  - L_1 o L_2 = { a o aa, a o cb, a o bb, bc o aa, ... }
  - L_1 o L_2 = { aaa, acb, abb, bcaa, ... }

Linguagem formal

- Demais operações sobre linguagens
  - Como uma linguagem é definida como um conjunto, são permitidas as operações: união, intersecção, diferença e complemento

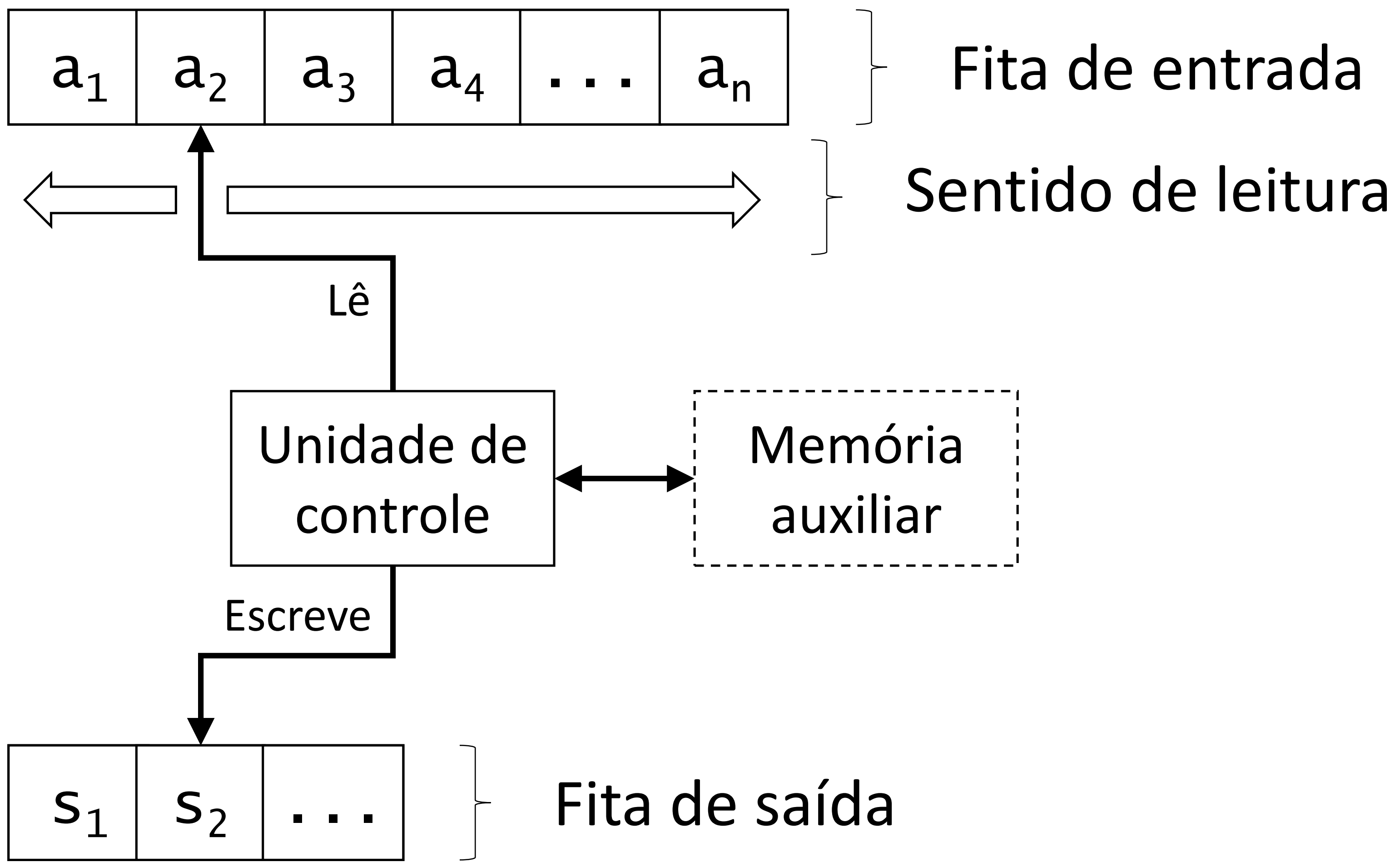
- Exemplo
  - $L_1 = \{a,b,aa,ab,abb,aab,aaa\}$ e $L_2 = \{a,aa,aaa,aaaa\}$
  - União: $L_1 \cup L_2 = \{a,b,aa,ad,abb,aab,aaa,aaaa\}$
  - Intersecção: $L_1 \cap L_2 = \{a,aa,aaa\}$
  - Diferença: $L_1 - L_2 = \{b,ab,abb,aab\}$

Autômato

- Modelo abstrato de um computador digital composto por:
  - Fita ou cadeia de entrada
  - Fita de saída ou cadeia resultante
  - Memória auxiliar utilizada para armazenamento temporário de símbolos
  - Unidade de controle

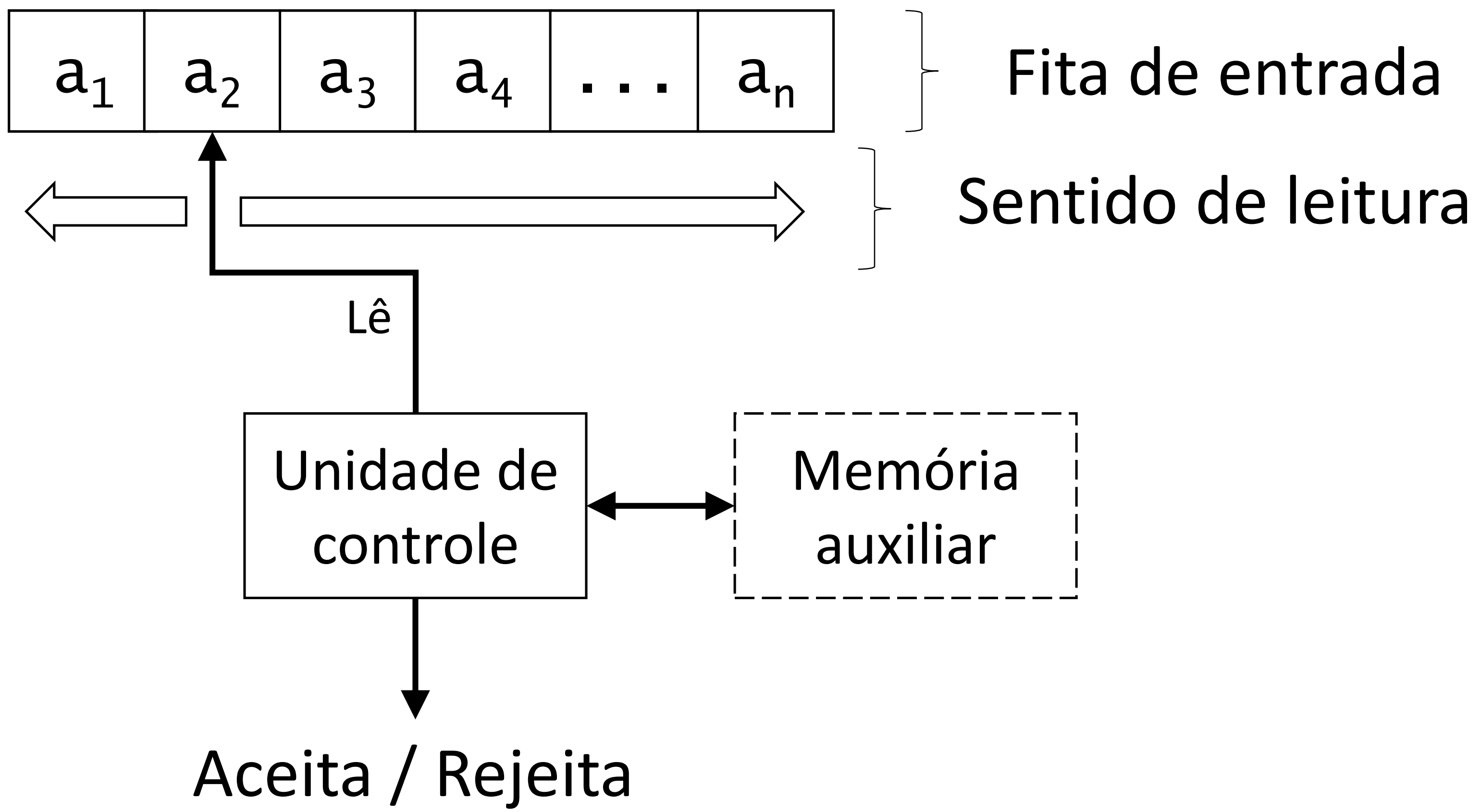
- A cadeia a ser tratada fica armazenada na fita de entrada onde a unidade de controle realiza operações como
  - Ler um símbolo da fita de entrada por vez
  - Mudar o estado conforme as funções de transição determinadas
  - Escrever na memória ou na fita de saída

Autômato



a1 a2 a3 a4 ... an
Fita de entrada
Sentido de leitura
Lê
Unidade de controle
Memória auxiliar
Escreve
s1 s2 ...
Fita de saída

Autômato



a1
a2
a3
a4
. . .
an
Fita de entrada
Sentido de leitura
Lê
Unidade de controle
Memória auxiliar
Aceita / Rejeita

Autômato finito determinístico (AFD)

- Autômato finito
    - Reconhecedor de linguagens simples que:
        - Não possui memória auxiliar
        - Não altera a fita (apenas a utiliza para leitura de símbolos)
        - A unidade de controle desloca-se em apenas um sentido
        - A fita tem um comprimento pré-determinado
    - Basicamente consiste de um sistema de estados finitos

- Autômato finito determinístico
    - Cada movimento resulta em uma única transição

Autômato finito determinístico (AFD)

- Implementa um sistema de estados finitos
  - Modelo matemático de sistema com entradas e saídas discretas
  - Número finito e predefinido de estados
  - Podem ser definidos antes de iniciar o processamento

- Estado
  - Baseia-se em somente informações do passado
  - Necessárias para determinar as ações para a próxima entrada

- Transição
  - Ação que realiza uma mudança de estado

Autômato finito determinístico (AFD)

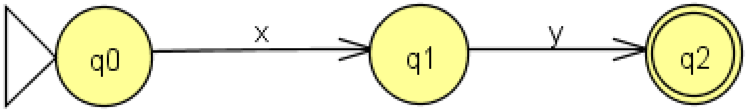
- Definição formal: $M = < Q,\Sigma,\delta,q_0,F >$
  - M: identificador do autômato
  - Q: conjunto de estados finitos
  - Σ: alfabeto de entrada (ou conjunto finito de símbolos)
  - δ: função de transição definida por $\delta: Q \times \Sigma \rightarrow Q$
  - q_0: estado inicial ($q_0 \in Q$)
  - F: conjunto de estados finais ($F \in Q$)

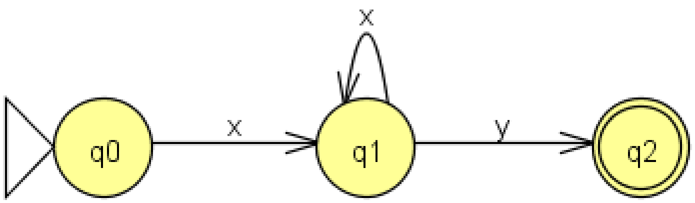
Autômato finito determinístico (AFD)

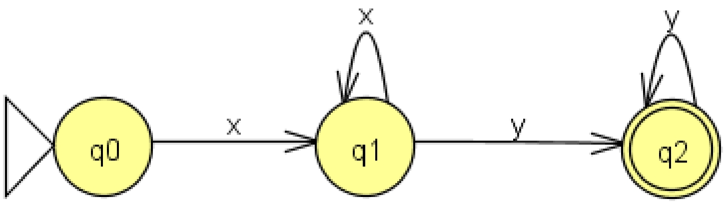
- Crie linguagens com o alfabeto $\Sigma = \{ x, y \}$ que aceitem as palavras:
  1. somente xy
     - Aceita: xy
     - Não aceita: x, y, yx, xyx, ...
  2. que comecem com uma sequencia não vazia de x e terminem com y
     - Aceita: xy, xxy, xxxy, ...
     - Não aceita: y, yxy, yy, xyxy, ...
  3. com uma sequencia não vazia de xy
     - Aceita: xy, xyxy, xyxyxy, ...
     - Não aceita: x, y, yx, xyx, ...
  4. que comecem com uma sequencia (¬Ø) de x e terminem com sequencia (¬Ø) de y
     - Aceita: xy, xxy, xxyy, xyy ...
     - Não aceita: y, yxy, yy, xyxy, ...
  5. que termine com xy
     - Aceita: xy, xxy, yxy, xyxy, ...
     - Não aceita: x, y, yx, xyx, xyy, ...

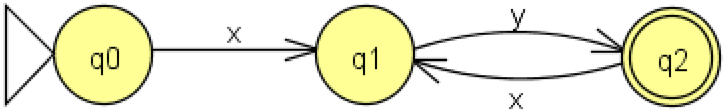
Autômato finito determinístico (AFD)

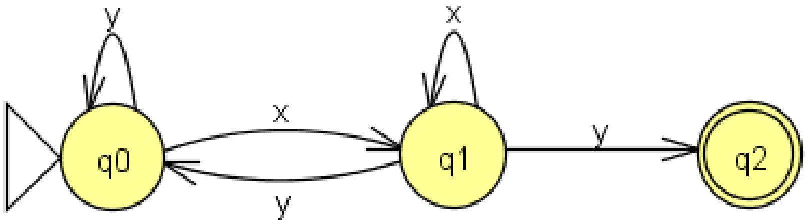
- Crie linguagens com o alfabeto $\Sigma = \{ x , y \}$ que aceitem as palavras:

1.


2.


3.


4.


5.


http://www.jflap.org/tutorial

Autômato finito determinístico (AFD)

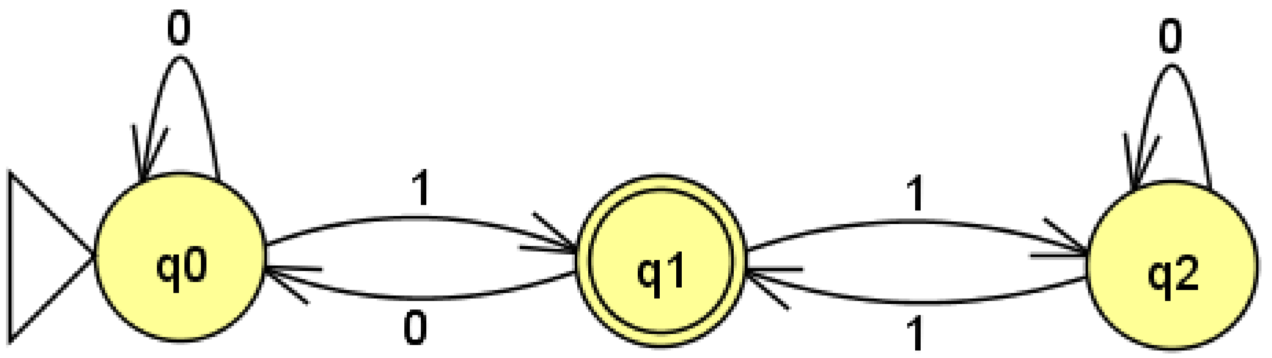
- Exemplo 1
  - Crie uma linguagem L com o alfabeto $\Sigma = \{ 0, 1 \}$
  - As palavras contidas em L devem terminar com um número ímpar de 1

  - Aceita
    - 1, 01, 111, 0111, 010111

  - Não aceita
    - 0, 10, 011, 0011, 1011

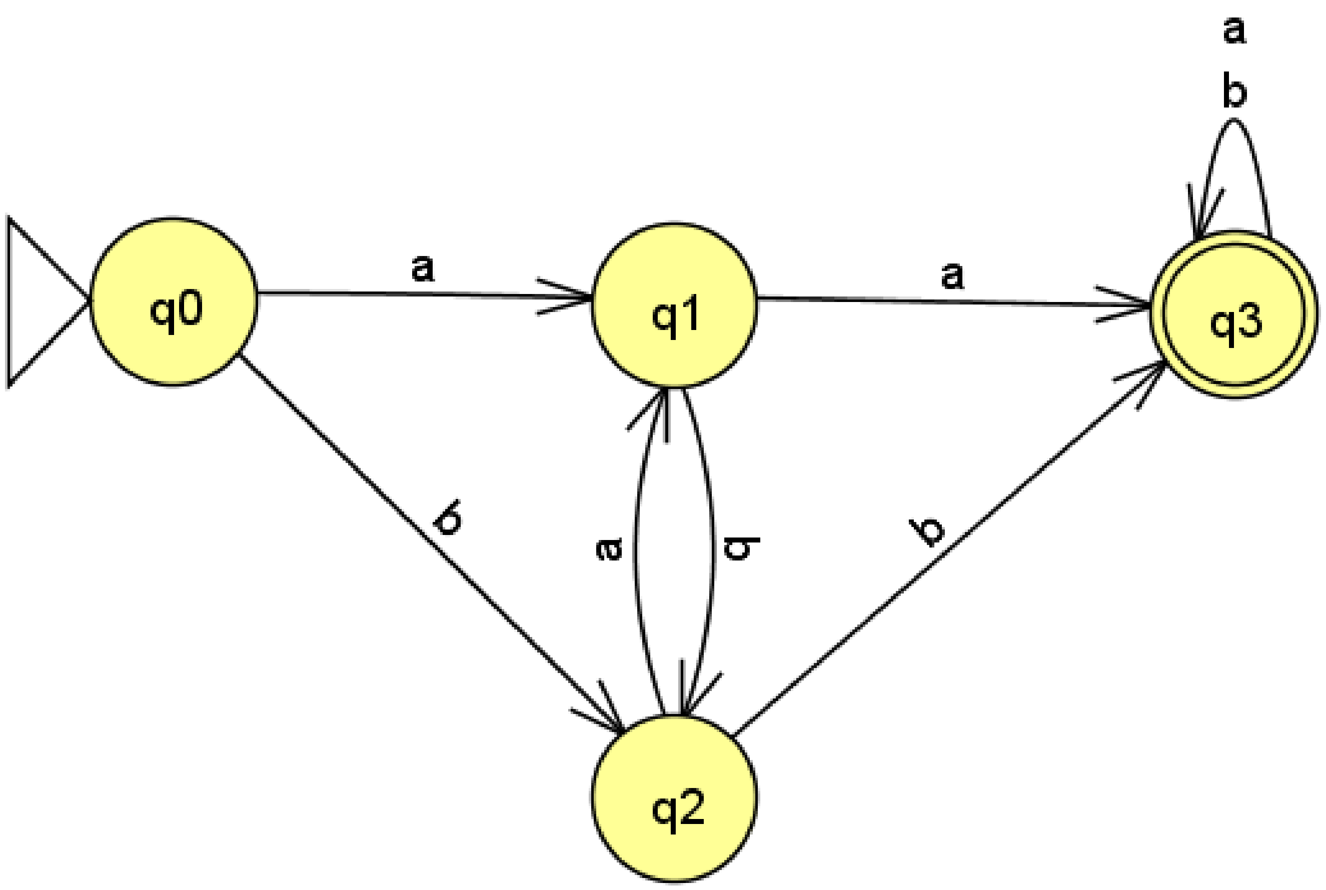
Autômato finito determinístico (AFD)

- Exemplo 1 (cont.)
  - AFD < $Q, \Sigma, \delta, q_0, F$ >
  - Q: { q_0, q_1, q_2 }
  - Σ: { 0, 1 }
  - δ: funções de transição
    - $\delta(q_0,0) = q_0$
    - $\delta(q_0,1) = q_1$
    - $\delta(q_1,0) = q_0$
    - $\delta(q_1,1) = q_2$
    - $\delta(q_2,0) = q_2$
    - $\delta(q_2,1) = q_1$
  - q_0: estado inicial
  - F: { q_1 }



Autômato finito determinístico (AFD)

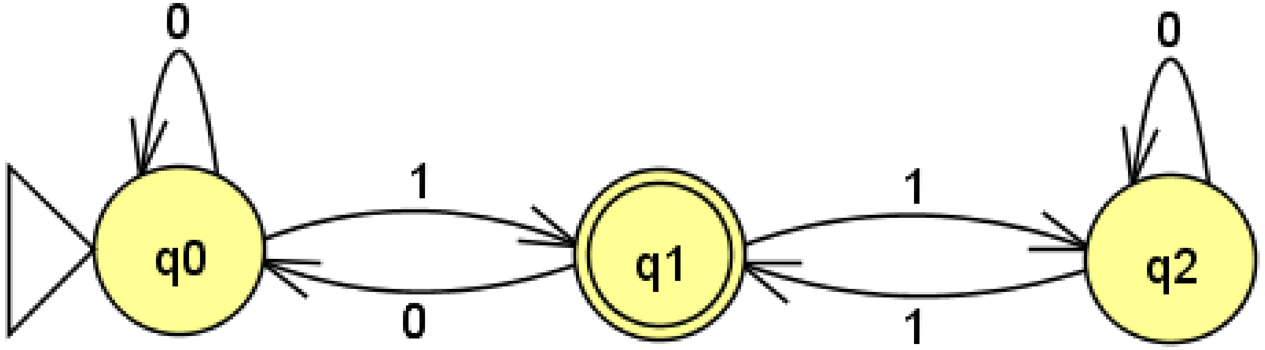
- Exemplo 2
  - AFD < $Q, \Sigma, \delta, q_0, F$ >
  - Q: { q_0, q_1, q_2, q_3 }
  - Σ: { a, b }
  - δ: funções de transição
    - $\delta(q_0, a) = q_1$
    - $\delta(q_0, b) = q_2$
    - $\delta(q_1, a) = q_3$
    - $\delta(q_1, b) = q_2$
    - $\delta(q_2, a) = q_1$
    - $\delta(q_2, b) = q_3$
    - $\delta(q_3, a) = q_3$
    - $\delta(q_3, b) = q_3$
  - q_0: estado inicial
  - F: { q_3 }



Autômato finito determinístico (AFD)

- Função de transição estendida
  - Representa uma sequência de transições dada uma cadeia (parcial ou não)
  - $\delta^*: Q \times \Sigma^* \rightarrow Q$

- Exemplo
  - Já que $\delta(q_0,0) = q_0$, $\delta(q_0,1) = q_1$ e $\delta(q_1,1) = q_2$, então:
  - $\delta^*(q_0,011) = q_2$



Autômato finito determinístico (AFD)

- Linguagem e autômato
  - Uma linguagem associada a um autômato refere-se ao conjunto de todas as cadeias aceitas pelo autômato
  - $L(M) = \{ w \in \Sigma^* : \delta^*(q_0, w) \in F \}$

- Equivalência entre autômatos
  - Dois autômatos finitos M_1 e M_2 são equivalentes se reconhecerem uma mesma linguagem
  - $L(M_1) = L(M_2)$

Linguagem regular

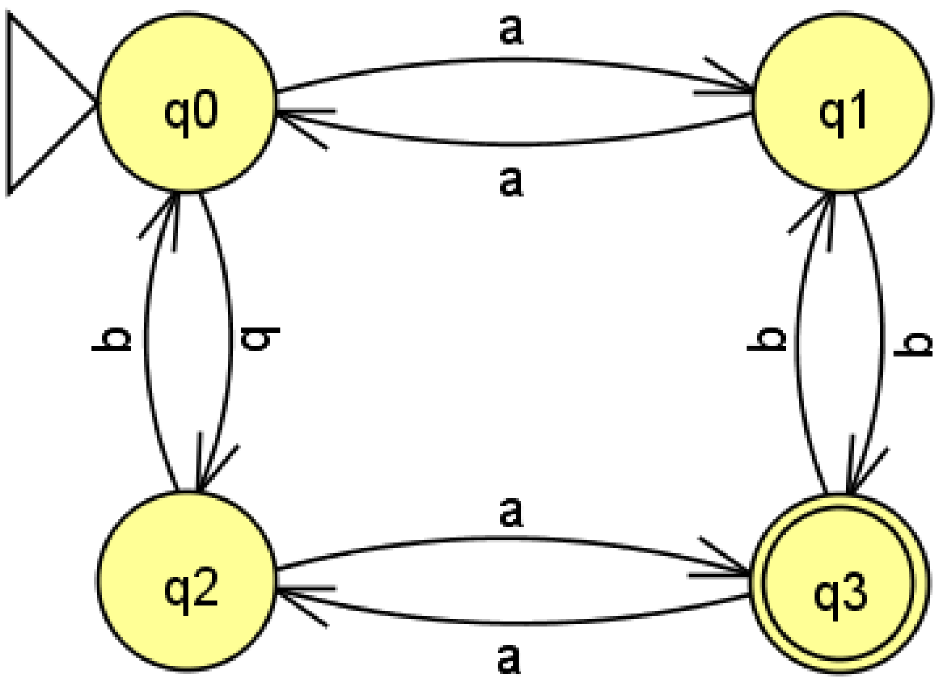
- Uma linguagem `L` é dita como regular se e somente se existir um autômato finito determinístico `M`
  - `L = L(M)`
- É uma linguagem mais simples, utilizada para desenvolver algoritmos de reconhecimento ou de geração de pouca complexidade, grande eficiência e de fácil implementação
- Formas de estudo de linguagens regulares
  - Operacional ou reconhecedor: uso de autômatos finitos
  - Axiomático ou gerador: gramática regular
  - Denotacional: expressão regular

Linguagem regular

- Exemplo
  - Considere a linguagem L $= \{\ w \mid w$ possui um número ímpar de a e b $\}$
  - M-impar $< Q, \Sigma, \delta, q_0, F >$
  - Q: ???
  - Σ: $\{ a, b \}$
  - δ: ???
  - q_0: estado inicial
  - F: ???

Linguagem regular

- Exemplo (cont.)
  - Considere a linguagem $L = \{ w \mid w$ possui um número ímpar de a e b $\}$
  - M-impar $< Q, \Sigma, \delta, q_0, F >$
  - Q: $\{ q_0, q_1, q_2, q_3 \}$
  - Σ: $\{ a, b \}$
  - δ: funções de transição
    - $\delta(q_0, a) = q_1$
    - $\delta(q_0, b) = q_2$
    - $\delta(q_1, a) = q_0$
    - $\delta(q_1, b) = q_3$
    - $\delta(q_2, a) = q_3$
    - $\delta(q_2, b) = q_0$
    - $\delta(q_3, a) = q_2$
    - $\delta(q_3, b) = q_1$
  - q_0: estado inicial
  - F: $\{ q_3 \}$

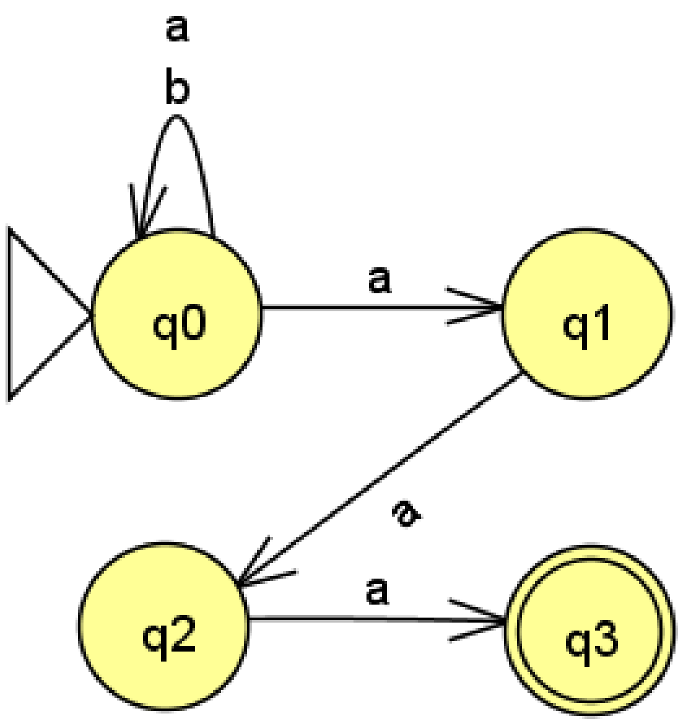


Autômato finito não determinístico (AFN)

- Definição formal: $M = < Q,\Sigma,\delta,q_0,F >$
  - M: identificador do autômato
  - Q: conjunto de estados finitos
  - Σ: alfabeto de entrada (ou conjunto finito de símbolos)
  - δ: função de transição definida por $\delta\colon Q \times \Sigma \rightarrow 2^Q$
  - q_0: estado inicial ($q_0 \in Q$)
  - F: conjunto de estados finais ($F \in Q$)
- 2^Q representa um subconjunto de Q

Autômato finito não determinístico (AFN)

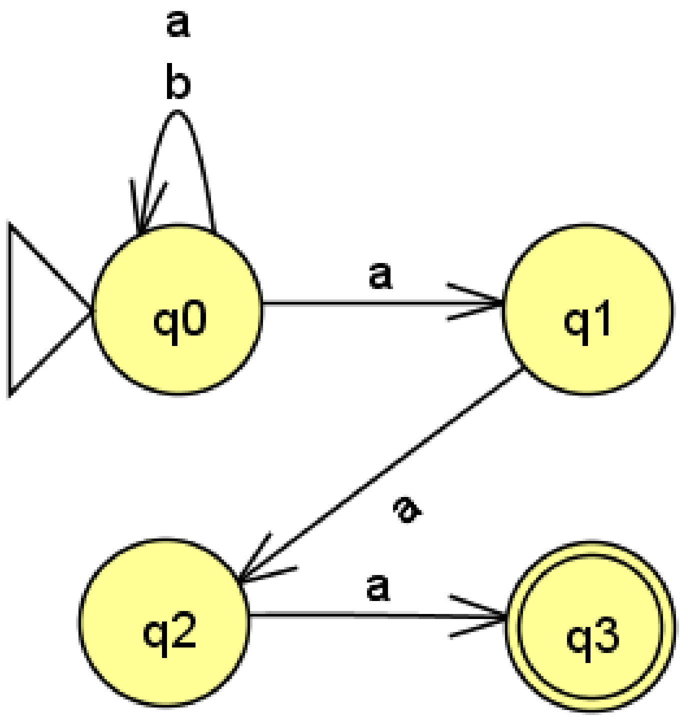
- Exemplo
  - Considere a linguagem $L = \{ w \mid w$ possui aaa como sufixo $\}$ dado alfabeto $\Sigma = \{a,b\}$
  - M: $\{\{q_0,q_1,q_2,q_3\},\{a,b\},\delta,\underset{Q}{q_0},\{q_3\}\}$
  - δ: função de transição
    - $\delta(q_0,a) = \{q_0,q_1\}$
    - $\delta(q_0,b) = \{q_0\}$
    - $\delta(q_1,a) = \{q_2\}$
    - $\delta(q_2,a) = \{q_3\}$



Autômato finito não determinístico (AFN)

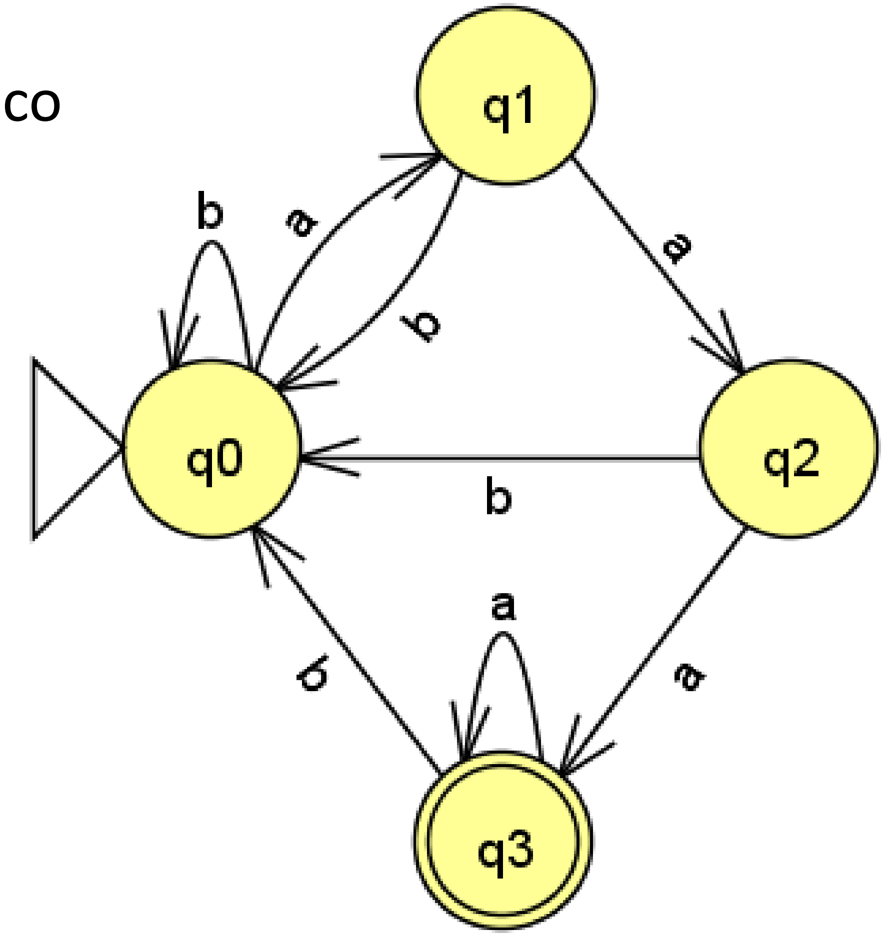
- Equivalência entre AFD e AFN
  - É possível transformar um AFN em um AFD para que a linguagem lida pelo AFN seja do tipo regular
  - Consiste em transformar as funções $\delta\colon Q \times \Sigma \rightarrow 2^Q$, onde $|2^Q| = 1$

- Exemplo
  - Transforme o autômato abaixo em determinístico
  - M: $\{\{q_0, q_1, q_2, q_3\}, \{a, b\}, \delta, q_0, \{q_3\}\}$
  - δ: função de transição
    - $\delta(q_0, a) = \{q_0, q_1\}$
    - $\delta(q_0, b) = \{q_0\}$
    - $\delta(q_1, a) = \{q_2\}$
    - $\delta(q_2, a) = \{q_3\}$



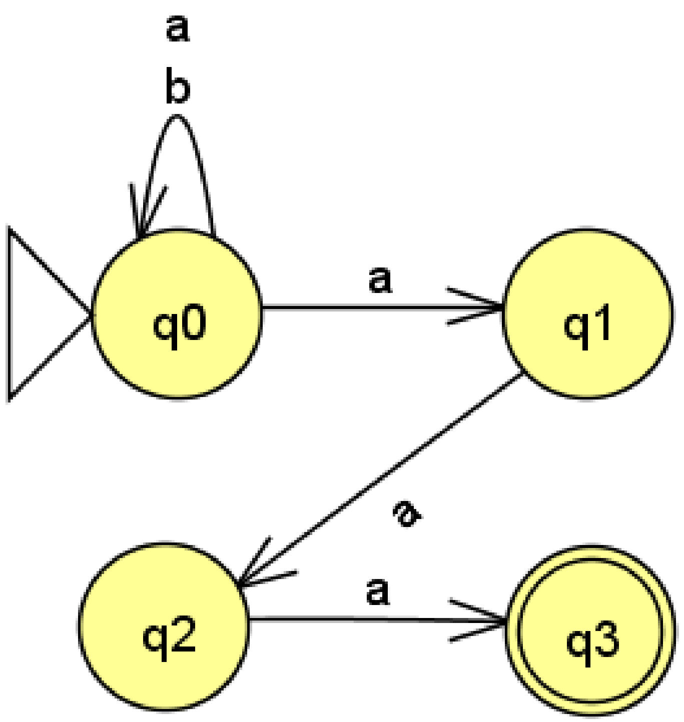
Autômato finito não determinístico (AFN)

- Exemplo (cont.)
  - Transforme o autômato abaixo em determinístico
  - M: $\{\{q_0, q_1, q_2, q_3\}, \{a, b\}, \delta, q_0, \{q_3\}\}$
  - δ: função de transição
    - $\delta(q_0, a) = q_1$
    - $\delta(q_0, b) = q_0$
    - $\delta(q_1, a) = q_2$
    - $\delta(q_1, b) = q_0$
    - $\delta(q_2, a) = q_3$
    - $\delta(q_2, b) = q_0$
    - $\delta(q_3, a) = q_3$
    - $\delta(q_3, b) = q_0$



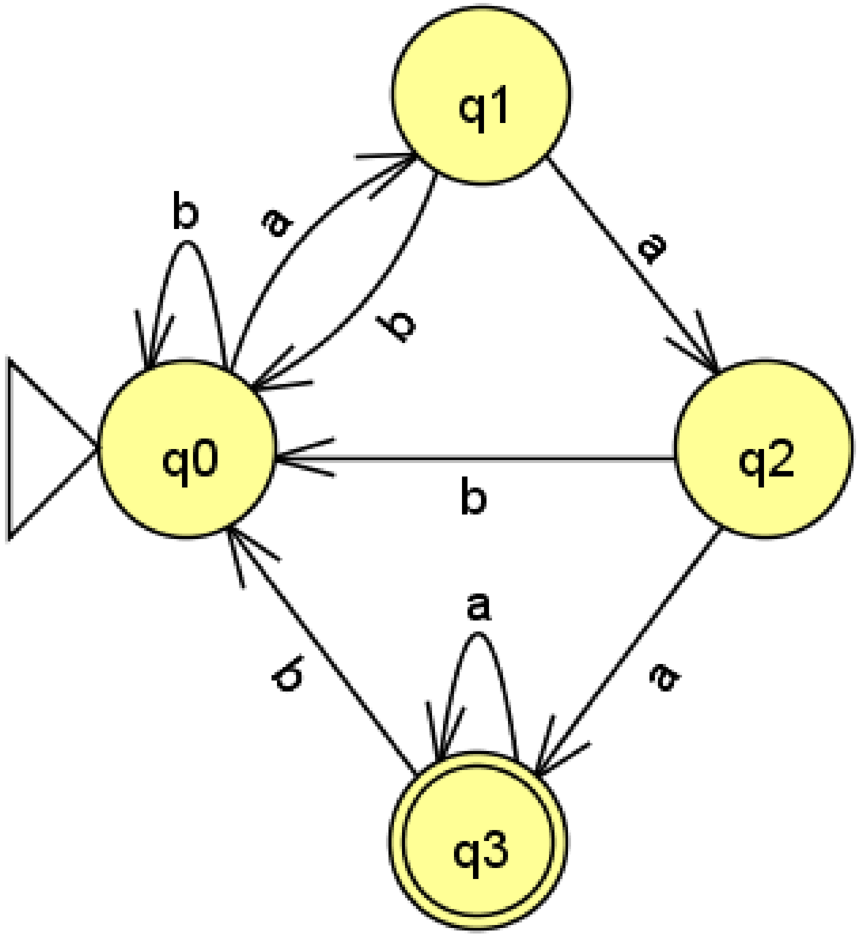
Autômato finito não determinístico (AFN)

- Exemplo (cont.)



AFN

≡



AFD

Expressão regular

- Modo de descrever conjuntos regulares
  - Amplamente utilizado em compiladores, editores, processadores de texto, sistemas operacionais, protocolos, etc.
  - Se `r` é uma expressão regular, então `L(r)` é uma linguagem regular

- Uma expressão regular sobre um alfabeto Σ é definido como:
  - ϕ: representa uma linguagem vazia
  - λ: representa uma linguagem exclusivamente com a palavra vazia, isto é, `{ λ }`
  - `x`: símbolo do alfabeto Σ contendo a palavra `{ x }`
  - Se `r` e `s` são expressões regulares e denotam as linguagens R e S, então:
    - `(r)` é uma expressão regular
    - `(r+s)` é uma expressão regular e denota a linguagem R ∪ S
    - `(r.s)` é uma expressão regular e denota a linguagem RS = { uv | u ∈ R e v ∈ S }
    - r^* é uma expressão regular e denota a linguagem R^*

Expressão regular

- Precedência de operadores
  - Concatenação sucessiva (*)
  - Concatenação (.)
  - União (+)

- Exemplo
  - Dada linguagem $L = \{ a^n b^m \mid n \geq 0, m \geq 0 \}$
  - $L = \{\lambda, a, b, aa, bb, ab, aab, ...\}$
  - $ER = a^* b^*$

Expressão regular

- Mais exemplos
  - aa: somente a cadeia aa
  - ba^*: cadeias que iniciam com b seguida por zero ou mais a
  - $(a+b)^*$: todas as cadeias sobre $\{a,b\}$
  - $(a+b)^*aa(a+b)^*$: todas as cadeias contendo aa como subcadeia
  - $a^*ba^*ba^*$: todas as cadeias contendo dois b
  - $(a+b)^*(aa+bb)$: todas as cadeias que terminam com aa ou bb
  - $(a+\lambda)(b+ba)^*$: todas as cadeias que não possuem dois a consecutivos

Gramática

- Linguagem de programação
  - Definida pelo conjunto de todos os programas (isto é, palavras)

- Linguagem de propósitos gerais
  - Conjunto de todos os programas é infinito
  - Não é uma definição adequada para implementação em computador

- Propósito da gramática
  - Maneira de especificar de forma finita linguagens (eventualmente) infinitas

Gramática

- Gramática é um conjunto finito de regras
  - As regras geram palavras quando aplicadas sucessivamente
  - O conjunto de todas as palavras geradas pela gramática define a linguagem
- Gramáticas para linguagens naturais
  - Exemplo: português
  - A ideia é a mesma para linguagens artificiais como Java, Python, C
- Gramáticas também são usadas para definir semântica
  - Mas em geral são usados outros formalismos

Gramática

- Componentes de uma gramática
  - Terminais: conjunto finito de símbolos que formam as palavras da linguagem, isto é, o alfabeto
  - Variáveis ou não terminais: conjunto finito de variáveis onde cada uma representa uma linguagem, isto é, um conjunto de palavras
  - Símbolo de início: variável utilizada para representar a linguagem sendo definida
  - Regras de produção: conjunto finito de produções ou regras que representam a definição recursiva de uma linguagem
    - Lado esquerdo (LHS): representa o símbolo de produção
    - Lado direito (RHS): representa o corpo da produção

Gramática

- Definição de gramática de Chomsky, gramática irrestrita ou gramática
  - $G = (V, T, P, S)$
  - V: conjunto finito de símbolos variáveis ou não-terminais
  - T: conjunto finito de símbolos terminais disjunto de V
  - P: relação finita de produções conforme $(V \cup T)^+ \rightarrow (V \cup T)^*$
    - Chamado de regras de produção
  - S: elemento distinguido de V que representa o símbolo inicial ou variável inicial

- Representação de uma regra de produção (α, β)
  - $\alpha \rightarrow \beta$
  - Abreviação de:
    - $\alpha \rightarrow \beta_1, \alpha \rightarrow \beta_2, \ldots, \alpha \rightarrow \beta_n$
    - $\alpha \rightarrow \beta_1 \mid \beta_2 \mid \ldots \mid \beta_n$

Gramática

- Derivação
  - Aplicação de uma regra de produção é denominada derivação
  - Aplicação sucessiva de regras de produção
  - Determina o fecho transitivo da relação de derivação
  - Permite derivar palavras da linguagem

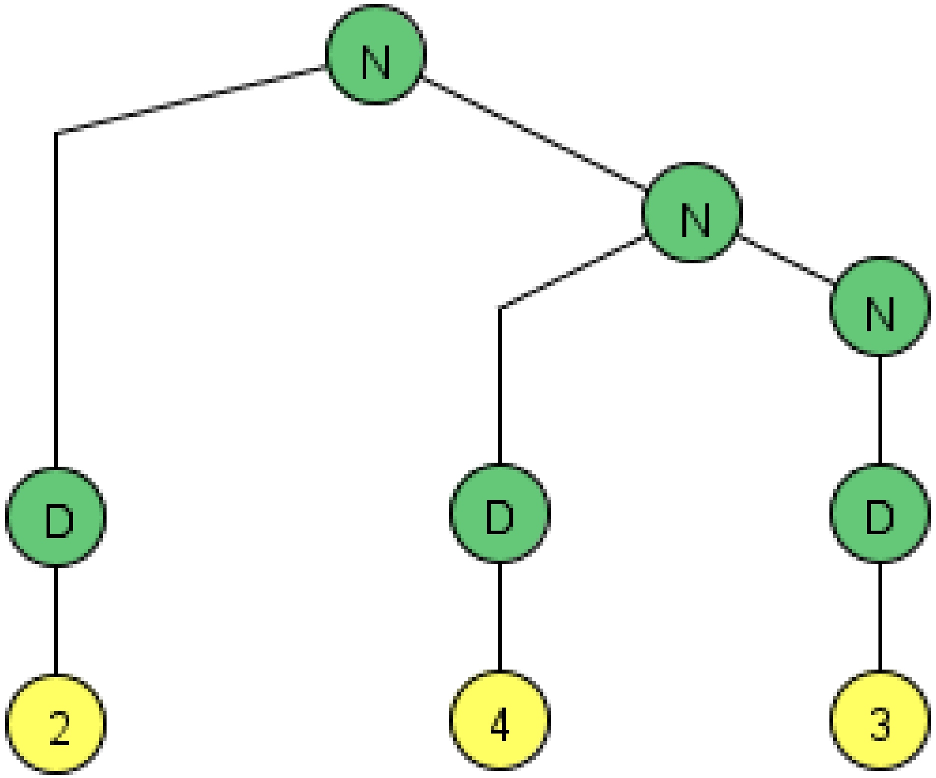
- Exemplo de derivação
  - G = < V, T, P, N >
  - V = {N,D}
  - T = {0,1,2,...,9}
  - P = regras de produção
    - N → D
    - N → DN
    - D → 0 | 1 | ... | 9

Gramática

- Exemplo de derivação (cont.)
  - $G = < V, T, P, N >$
  - $V = \{N, D\}$
  - $T = \{0, 1, 2, ..., 9\}$
  - $P = \{N \rightarrow D, N \rightarrow DN, D \rightarrow 0|1|...|9\}$

- Derivação de 243

1) N: $N \rightarrow DN$
2) DN: $D \rightarrow 2$
3) 2N: $N \rightarrow DN$
4) 2DN: $D \rightarrow 4$
5) 24N: $N \rightarrow D$
6) 24D: $D \rightarrow 3$
7) 243



Gramática

- Gramática
  - Permite definir tanto linguagens regulares como não regulares
  - Convenções
    - `A,B,C,...,S,T` para símbolos variáveis
    - `a,b,c,...,s,t` para símbolos terminais
    - `u,v,w,x,y,z` para palavras de símbolos terminais

- Gramática regular
  - Adiciona restrições nas regras de produção
  - Existem diferentes formas de restringir as regras de produção
  - Gramáticas lineares é um exemplo

Gramática regular

- Gramática linear à direita (GLD)
  - Regras de produção $A \rightarrow wB$ ou $A \rightarrow w$

- Gramática linear à esquerda (GLE)
  - Regras de produção $A \rightarrow Bw$ ou $A \rightarrow w$

- Gramática linear unitária à direita (GLUD)
  - Como na gramática linear à direita, porém, $|w| \leq 1$

- Gramática linear unitária à esquerda (GLUE)
  - Como na gramática linear à esquerda, porém, $|w| \leq 1$

Gramática regular

- Exemplo 1
  - $G = < \{S\},\{x,y\},P,S >$
  - $S = \{S \rightarrow xyS,\ S \rightarrow x\}$

- Exemplo 2
  - $G = < \{S,A,B\},\{a,b\},P,S_1 >$
  - $S = \{S \rightarrow Aab,\ A \rightarrow Aab \mid B,\ B \rightarrow a\}$

- Exemplo 3
  - $G = < \{S,A,B\},\{a,b\},P,S >$
  - $S = \{S \rightarrow A,\ A \rightarrow aB \mid \lambda,\ B \rightarrow Ab\}$

Gramática regular

- Exemplo 1 – é uma GLD
  - $G = < \{S\},\{x,y\},P,S >$
  - $S = \{S \rightarrow xyS,\ S \rightarrow x\}$

- Exemplo 2 – é uma GLE
  - $G = < \{S,A,B\},\{a,b\},P,S_1 >$
  - $S = \{S \rightarrow Aab,\ A \rightarrow Aab \mid B,\ B \rightarrow a\}$

- Exemplo 3 – não é uma gramática linear
  - $G = < \{S,A,B\},\{a,b\},P,S >$
  - $S = \{S \rightarrow A,\ A \rightarrow aB \mid \lambda,\ B \rightarrow Ab\}$

Linguagens regulares

- Exemplos de linguagens regulares
    - Autômatos finitos determinísticos
    - Autômatos finitos não determinísticos
    - Expressões regulares
    - Gramáticas regulares