# Compiladores
# Análise Léxica
# Autômatos Finitos

Prof. Dr. Luiz Eduardo G. Martins
(adaptado por Profa Dra Ana Carolina Lorena)
UNIFESP

# Autômatos Finitos

- Um Autômato Finito é uma máquina de estados finitos que permite reconhecer, por meio de um conjunto de estados e transições dirigidas, pela ocorrência de símbolos de um alfabeto, se uma determinada cadeia de caracteres *(string)* pertence ou não a uma linguagem regular
- Os Autômatos Finitos podem ser usados para descrever o processo de reconhecimento de padrões em cadeias de entrada (identificação de *tokens*)
- Portanto, são muito úteis na construção do Analisador Léxico

# Autômatos Finitos

- O núcleo do analisador léxico é uma implementação de um autômato finito
  - máquina de estados finitos que aceita símbolos de uma **palavra**
  - ao final da **palavra** indica se ela é válida para a gramática ou não
  - autômato é definido para cada conjunto de símbolos que deve ser reconhecido
- Cada *token* pode ser descrito por uma lista ou por uma expressão regular

# Autômatos Finitos: definição

- Definição Formal

Autômato é descrito por uma quíntupla

$$M = (K, \Sigma, \delta, s, F)$$

# Autômatos Finitos: definição

- Definição Formal

Autômato é descrito por uma quíntupla

$$M = (K, \Sigma, \delta, s, F)$$

$K$ conjunto (finito) de estados

$\Sigma$ alfabeto (finito) de entrada

$\delta$ conjunto de transições

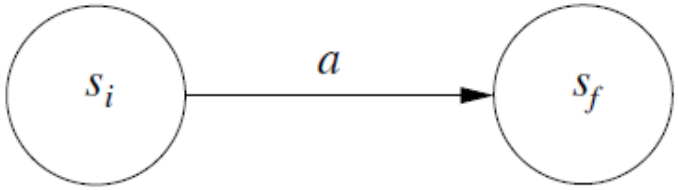
$s$ estado inicial, $s \in K$

$F$ conjunto de estados finais, $F \subseteq K$

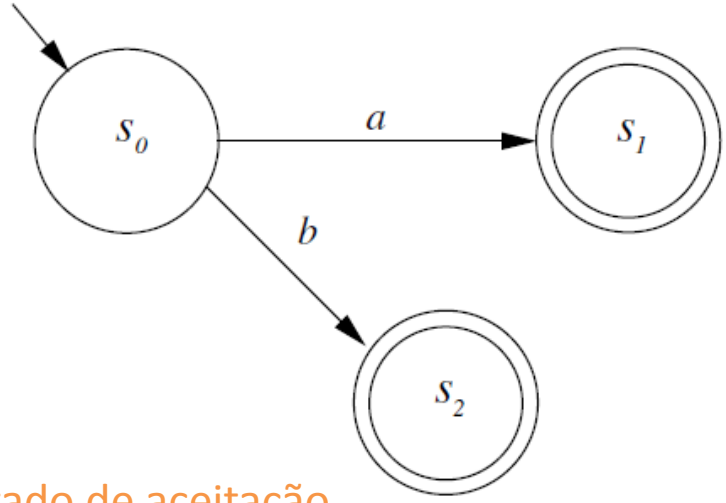
# Autômatos Finitos: representação

- Representação Gráfica

Estados e transição




Estados inicial e finais



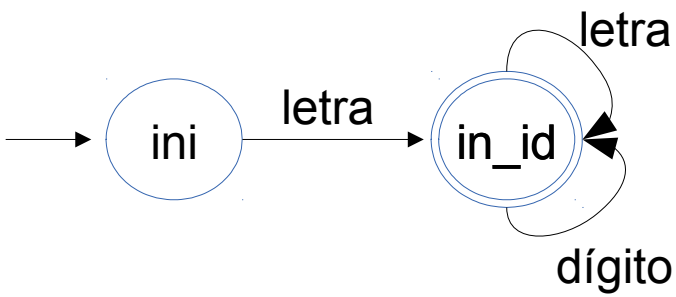

**Representação Formal:**

*K = {so, s1, s2}*
*∑ = {a, b}*
*δ = {(so, {a}, s1),(so,{b}, s2)}*
*s = so*
*F = {s1, s2 }*

estado final = estado de aceitação
(um ou mais)

# Autômatos Finitos: exemplo

- identificador = letra(letra|dígito)*



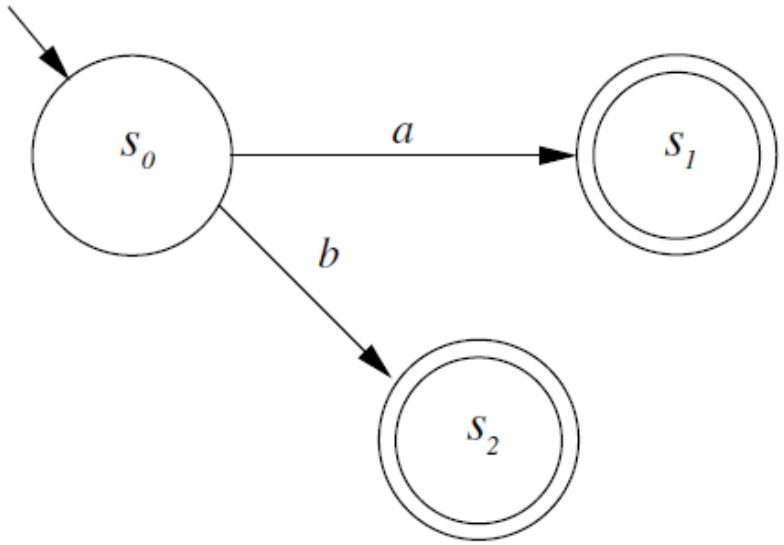

**Exercício 1**:

apresente a representação formal do AFD ao lado

# Autômatos Finitos: representação

- Representação Tabular

Estados inicial e finais



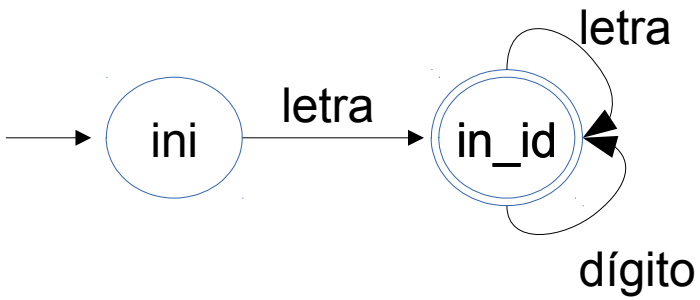

| | $s_0$ | $s_1$ | $s_2$ |
|---|---|---|---|
| $a$ | $s_1$ | — | — |
| $b$ | $s_2$ | — | — |

Estado inicial: $s_0$

Estados finais: $s_1$ , $s_2$

# Autômatos Finitos: exemplo

- identificador = letra(letra|dígito)*




**Representação Formal:**

*K = {ini, in_id}*
*∑ = {letra, dígito}*
*δ = {(ini, {letra}, in_id),(in_id,{letra}, in_id), (in_id,{dígito}, in_id)}*
*s = ini*
*F = {in_id}*

**Exercício 2**:

apresente a representação
por tabela do AFD acima

# Autômatos Finitos: tipos

- Tipos de Autômatos
  1. Determinísticos (AFD ou DFA)
  2. Não-Determinísticos (AFND ou NFA)

# Autômatos Finitos Determinísticos

- Determinísticos
  - Estado seguinte é univocamente determinado pelo estado corrente e pelo caractere de entrada
  - Não há alternativas para a transição a partir de um estado com o mesmo símbolo de entrada
  - Não há transições pela cadeia vazia

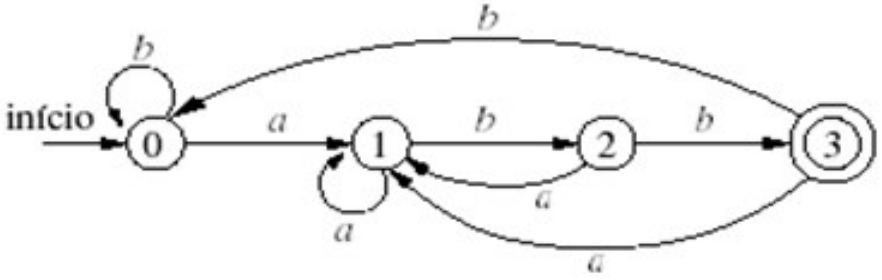



FIGURA 3.28 DFA aceitando $(a|b)^*abb$.

# Autômatos Finitos Não Determinísticos

- Não-Determinísticos
  - Pode existir mais de uma transição (aresta) saindo do mesmo estado com o mesmo símbolo
  - Pode existir transição de estado sem a ocorrência de nenhum símbolo de entrada
    - Transição pela cadeia vazia (anotada por ε)

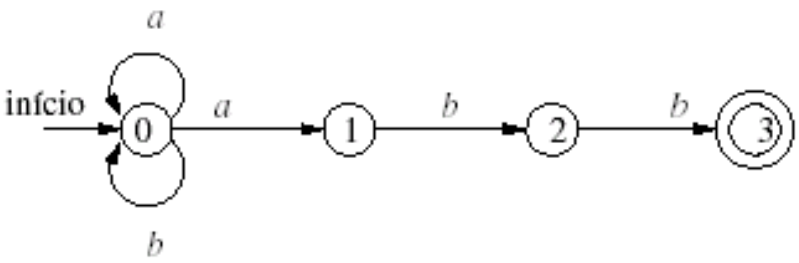



FIGURA 3.24 Um autômato finito não determinista.

| ESTADO | *a* | *b* | $\epsilon$ |
|---|---|---|---|
| 0 | {0,1} | {0} | Ø |
| 1 | Ø | {2} | Ø |
| 2 | Ø | {3} | Ø |
| 3 | Ø | Ø | Ø |

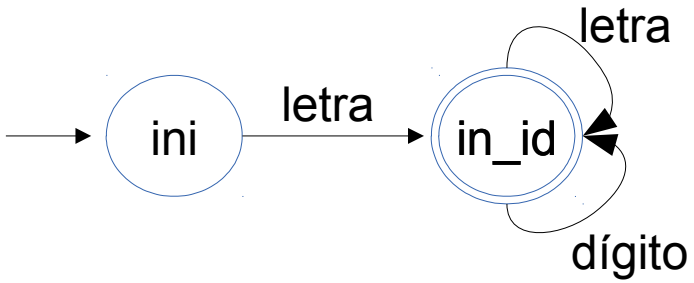
FIGURA 3.25 Tabela de transição para o NFA da Figura 3.24.

# Compiladores: Autômatos Finitos

- Aceitação das cadeias de entrada pelos autômatos
  - Um autômato finito *aceita* (consome) a cadeia de entrada se e somente se houver algum caminho no grafo de transições: do estado inicial para um dos estados finais (estados de aceitação)

# Autômatos Finitos: exemplo

- identificador = letra(letra|dígito)*




**Representação Formal:**

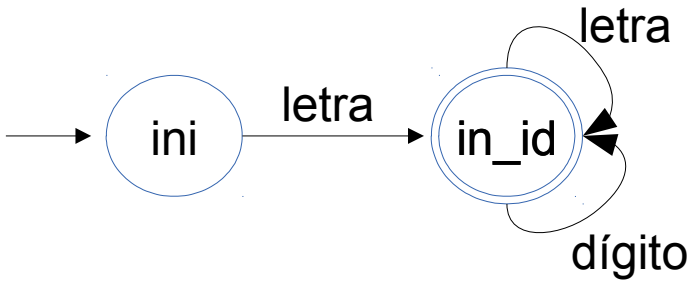
*K = {ini, in_id}*
*∑ = {letra, dígito}*
*δ = {(ini, {letra}, in_id),(in_id,{letra}, in_id), (in_id,{dígito}, in_id)}*
*s = ini*
*F = {in_id}*

Exemplo: reconhecendo xtemp

→ ini → ind_in → in_id → in_id → in_id → in_id
x t e m p

# Autômatos Finitos: exemplo

- identificador = letra(letra|dígito)*




**Exercício 3**:
Mostre se o AFD ao lado aceita as seguintes cadeias como identificadores:

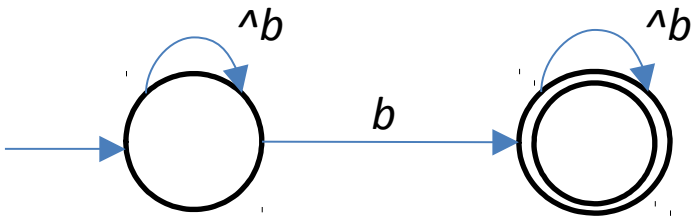
a) i
b) x_temp
c) i1

Exemplo: reconhecendo xtemp

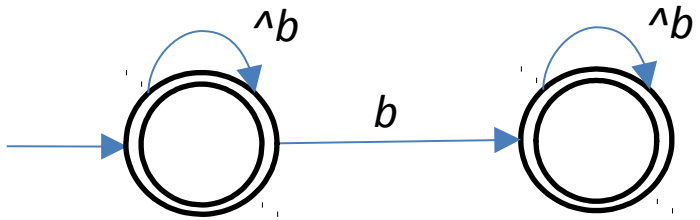
→ ini → ind_in → in_id → in_id → in_id → in_id
x t e m p

# Autômatos Finitos: exemplos

- Exemplos de AFDs $\sum = \{a, b, c, \ldots, z\}$
  - Reconhecimento de cadeias de caracteres que contêm exatamente um *b*




  - Reconhecimento de cadeias de caracteres que contém no máximo um *b*



# Autômatos Finitos: exemplos

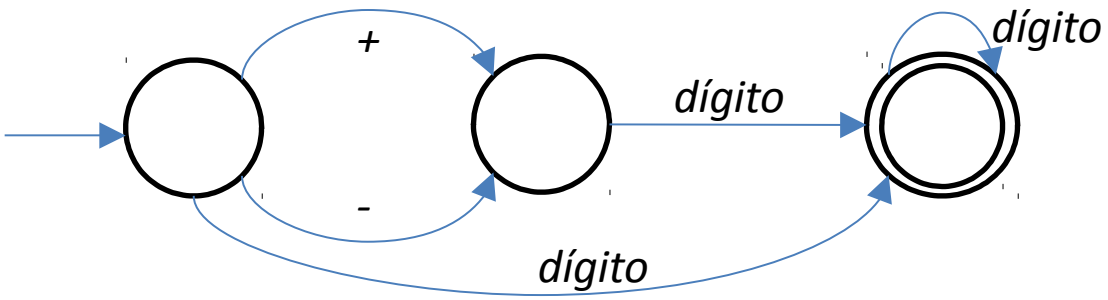
- Exemplos de AFDs
  - Reconhecimento de um número natural
    - Definições regulares

```
dígito = [0-9]
nat = dígito+
```

# Autômatos Finitos: exemplo

- Exemplos de AFDs
  - Reconhecimento de um número natural com sinal opcional
    - Definições regulares

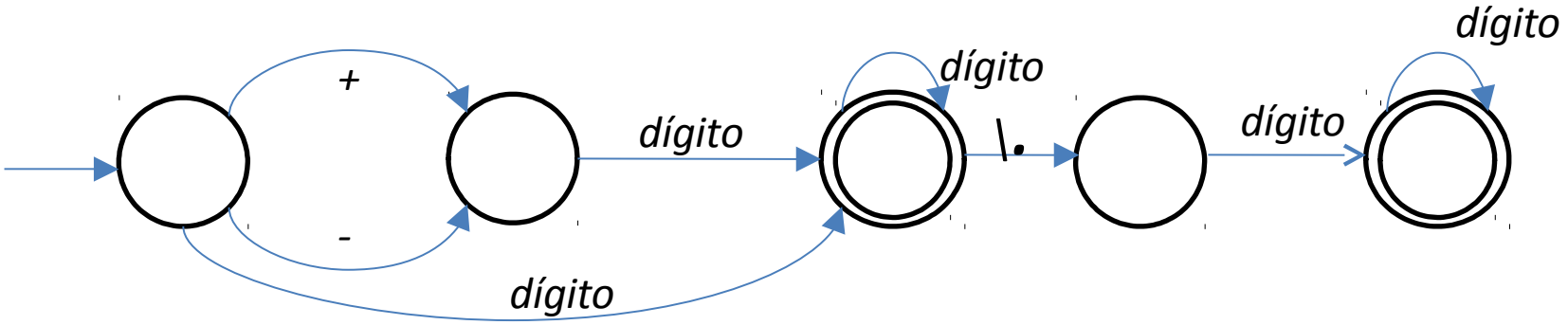
```
dígito = [0-9]
nat = dígito+
signedNat = (+|-)? nat
```



# Autômatos Finitos: exemplo

- Exemplos de AFDs
  - Reconhecimento de um número real
    - Definições regulares

```
dígito = [0-9]
nat = dígito+
signedNat = (+|-)? nat
real = signedNat (\.nat)?
```



# Autômatos Finitos: exemplo

- Reconhecimento de um número real com parte exponencial opcional
  - Definições regulares

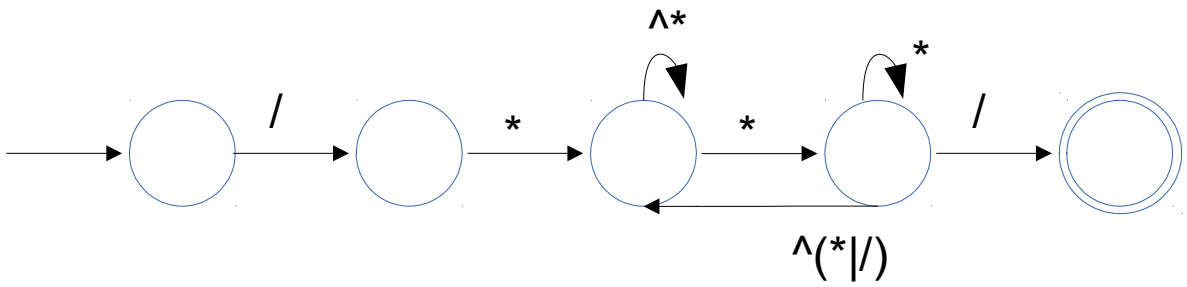
```
dígito = [0-9]
nat = dígito+
signedNat = (+|-)? nat
exp = E signedNat
real = signedNat (\.nat)? exp?
```

**Exercício 4**:

apresente o AFD para o reconhecimento de reais segundo as definições acima

# Autômatos Finitos: exemplo

- Reconhecimento de comentários
  - Em C: /* … (nenhuma ocorrência de */) … */
  - ER era complicada, mas AFD é fácil de escrever

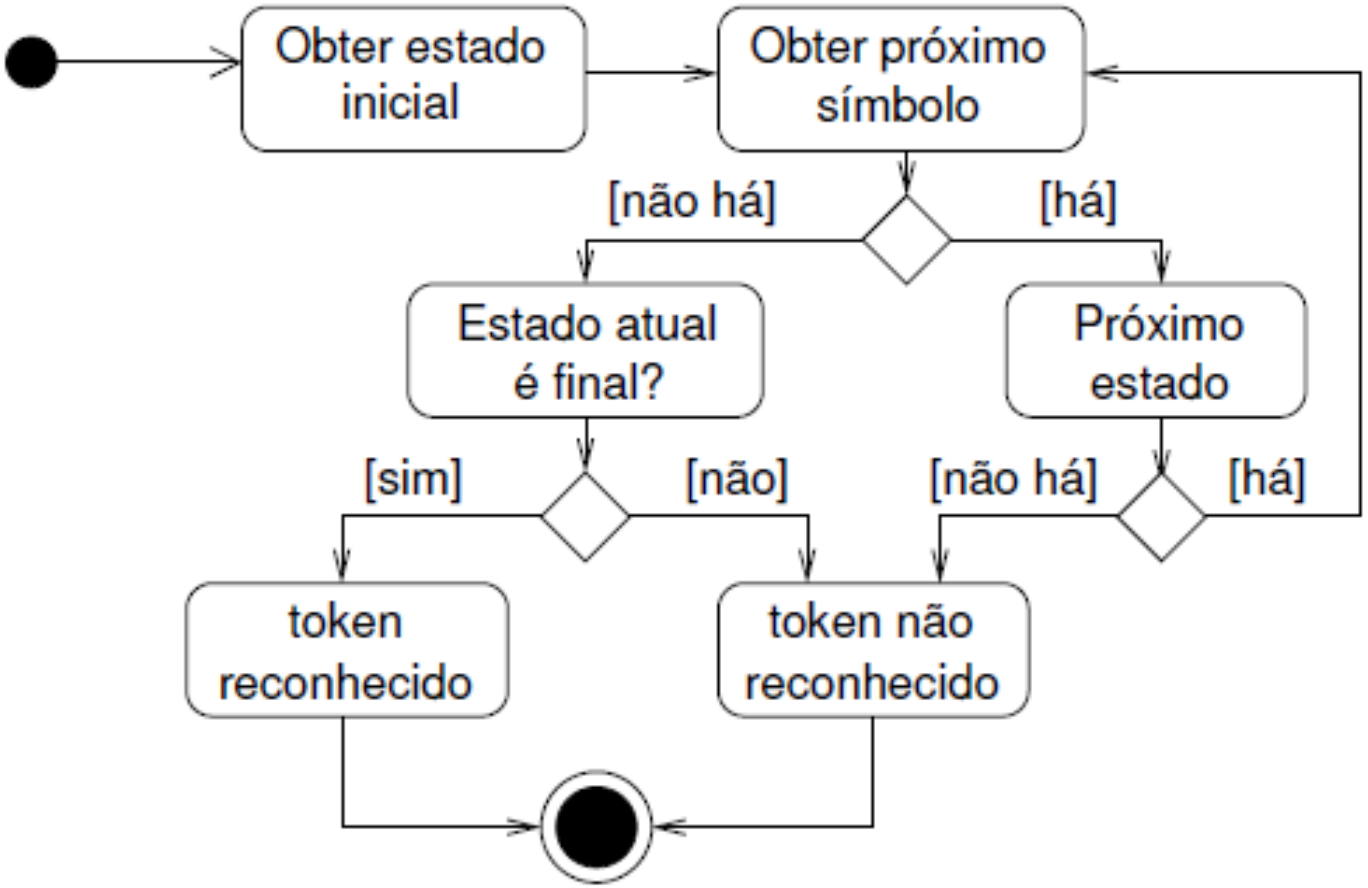


# Compiladores: Autômatos Finitos

- Ação típica da transição no Analisador Léxico: mover caractere da cadeia de entrada para uma cadeia que acumule os caracteres pertencentes a uma marca (lexema da marca)
- Ação típica do estado de aceitação no Analisador Léxico: retornar a marca reconhecida, com seus atributos (tipicamente lexema)
- Ação típica do Analisador Léxico em caso de erro: voltar para trás (retrocesso) na cadeia de entrada ou gerar marca de erro

# Compiladores: Autômatos Finitos

- Atividades para o reconhecimento de *token*



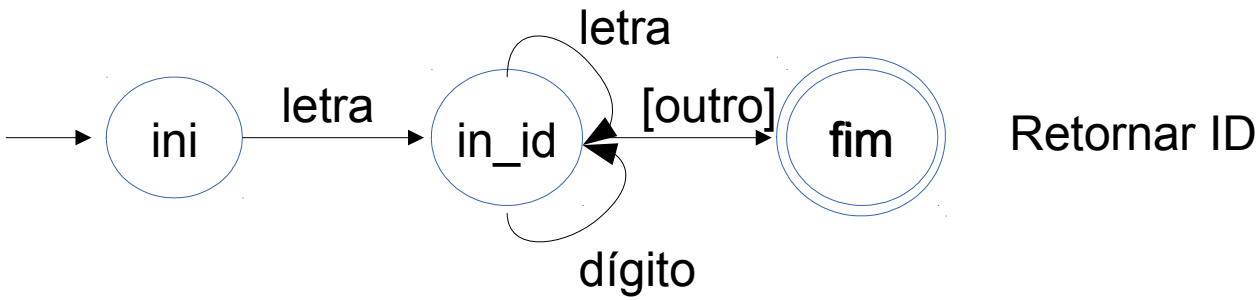
# Compiladores: Autômatos Finitos

- Algoritmo para o Analisador Léxico

```
RECONHECE_STRING(AFD, string)
    s ← ESTADO_INICIAL(AFD)
    while TEM_SIMBOLOS(string)
        c ← PROXIMO_SIMBOLO(string)
        if EXISTE_PROXIMO_ESTADO(AFD, s,c)
            s ← PROXIMO_ESTADO(AFD, s, c)
        else
            return false
    if ESTADO_FINAL(AFD, s)
        return true
    else
        return false
```

# Compiladores: observações

- Exemplo: reconhecer identificador

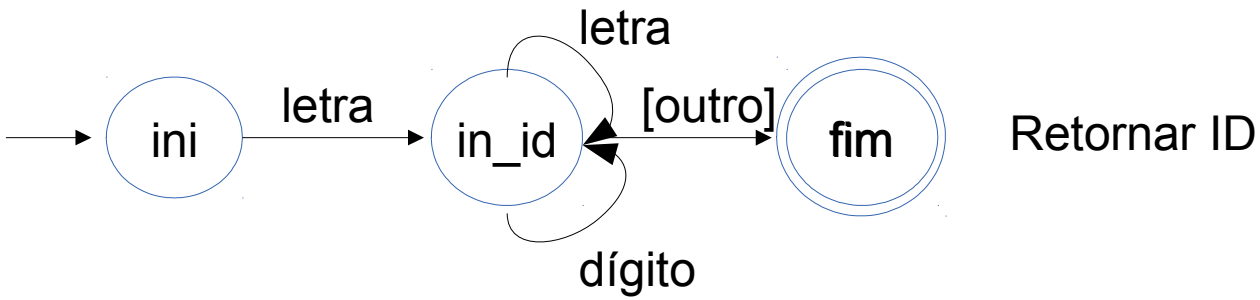



Acomoda o fato de que um delimitador deve ser visto para que realmente a marca seja reconhecida

O caractere representado por outro está entre colchetes pelo fato de que ele não deve ser consumido (colocado no lexema), mas sim devolvido à cadeia de entrada (lembre-se do exemplo x=10)

# Compiladores: observações

- Exemplo: reconhecer identificador

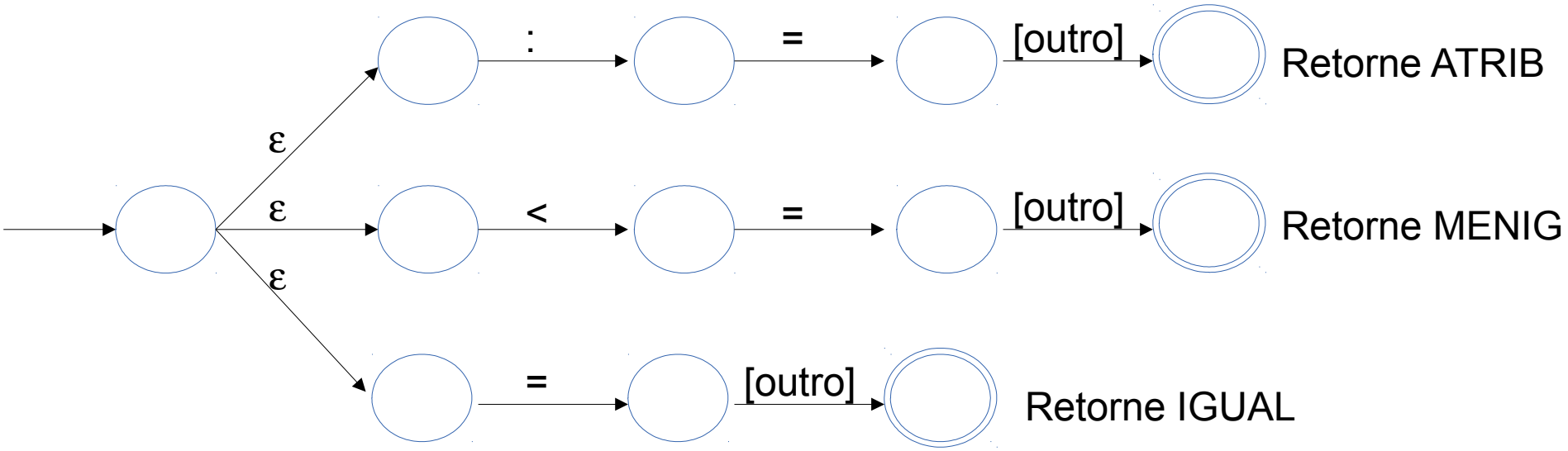



- Esse AFD, em contraste com o inicial apresentado, também expressa o princípio da subcadeia mais longa
  - Continua a casar letras e dígitos até encontrar um delimitador
  - O anterior poderia parar antes

# Compiladores: observações

- Tipicamente há muitas marcas, e cada marca é reconhecida por seu AFD próprio
- É necessário juntá-los em um único AFD
- Para tal, é mais simples primeiro gerar um AFND juntando os AFDs individuais e depois convertê-lo em um AFD equivalente

# Compiladores: exemplo




Vantagem é que preserva os autômatos originais, apenas adicionando um estado inicial que os conecta

Transição com cadeia vazia: pode ocorrer sem consultar a cadeia de entrada e sem consumir caracteres

# Exercício para próxima aula

- Considere o programa Flex para reconhecer identificadores, números e símbolos especiais da aula anterior:
  a) Apresente sua definição de identificador, número e símbolo especial (em forma escrita)
  b) Apresente as expressões regulares equivalentes às definições anteriores
  c) Apresente o AFD de cada uma dessas expressões

# Bibliografia

- Bibliografia consultada

Capítulo 2 LOUDEN, K. C. **Compiladores: princípios e práticas.** São Paulo: Pioneira Thompson Learning, 2004

RICARTE, I. **Introdução à Compilação.** Rio de Janeiro: Editora Campus/Elsevier, 2008.

AHO, A. V.; LAM, M. S.; SETHI, R. e ULLMAN, J. D. **Compiladores: princípios, técnicas e ferramentas.** 2ª edição – São Paulo: Pearson Addison-Wesley, 2008