Análise léxica

Autômatos finitos

Prof. Edson Alves

Faculdade UnB Gama

Reconhecedores

- Um reconhecedor é um programa que identifica, respondendo `"Sim"` ou `"Nao"`, se uma cadeia é ou não uma sentença válida de uma determinada linguagem
- Uma estratégia para a compilação de expressões regulares em reconhecedores é o uso de diagramas de transição generalizados, denominados autômatos finitos
- Um autômato finito pode ser determinístico ou não-determinístico: no segundo caso, podem existir duas ou mais transições com o mesmo rótulo partindo de um mesmo estado
- Autômatos determinísticos podem resultar em reconhecimentos mais rápidos do que os não-determinísticos, porém em geral são muito maiores, no que diz respeito ao número de estados e transições
- É possível representar expressões regulares em ambos tipos de autômatos finitos

Autômatos finitos não-determinísticos

Definição de AFN

Um autômato finito não-determinístico (AFN) é um modelo matemático que consiste em

1. um conjunto de estados S,
2. um alfabeto Σ de símbolos de entrada,
3. uma função de transição que mapeia pares (estado, símbolo) em um conjunto de estados,
4. um estado s_0, denominado estado inicial ou de partida, e
5. um conjunto F de estados de aceitação (ou estados finais).

Grafo de transições

- Um AFN pode ser representado por meio de um grafo direcionado e rotulado, denominado grafo de transição
- Em um grafo de transição, os nós representam os estados e as arestas definem a função de transição
- Os rótulos das arestas são os símbolos associados à transição, e a direção da aresta parte do estado atual para o próximo estado
- Grafos de transição se assemelham aos diagramas de transição, com duas diferenças fundamentais
- A primeira diferença é que um mesmo rótulo pode estar associado a duas ou mais arestas partindo de um mesmo estado
- A segunda é que o símbolo ϵ pode rotular uma aresta

Grafo de transição para a linguagem $(a \mid b)^*abb$

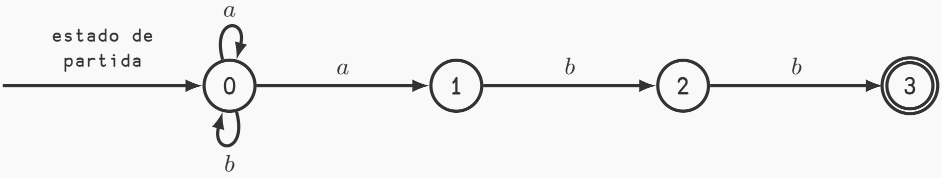


Tabela de transições

- Uma alternativa para a implementação da função de transição é a tabela de transições
- Em uma tabela de transições cada linha representa um estado e cada coluna representa um rótulo
- Se necessário, é necessário adicionar uma coluna para o rótulo ϵ
- A entrada da tabela posicionada na línha i, coluna c, contém o conjunto de estados que podem suceder o estado i quando o caractere c for lido na entrada
- De fato, a tabela de transições corresponde à representação do grafo de transições como uma matriz de adjacências
- Outra alternativa é representar o grafo por meio de uma lista de adjacências

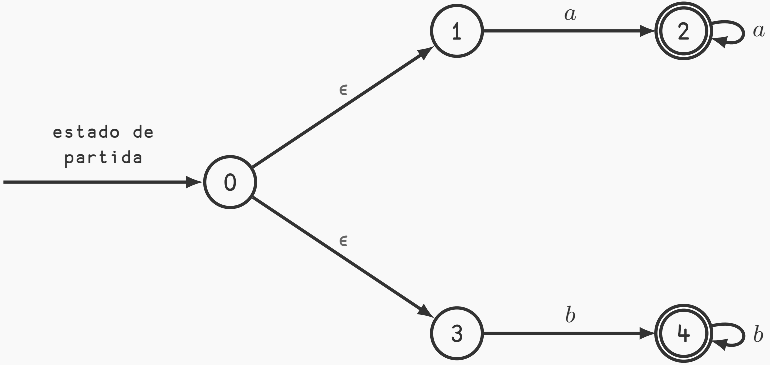
Tabela de transições do AFN da linguagem $(a \mid b)^*abb$

Estado	Símbolo de entrada	
	a	b
0	{ 0, 1 }	{ 0 }
1	-	{ 2 }
2	-	{ 3 }

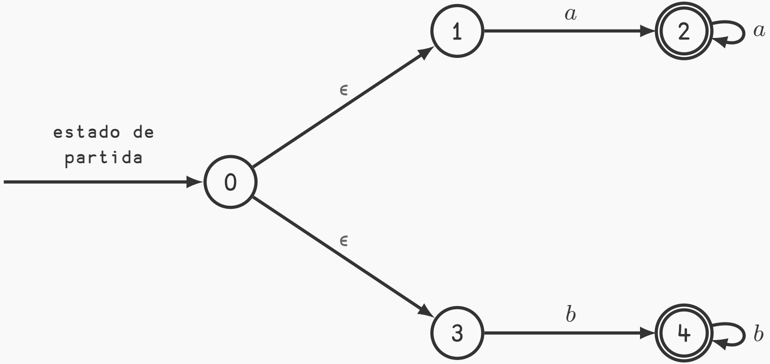
Caminhos

- Um caminho em um grafo de transições é uma sequência de arestas de transição
- Os rótulos das arestas, quando concatenados, formam uma cadeia s
- Caso o símbolo ϵ seja o rótulo de uma ou mais arestas de um caminho, na concatenação dos rótulos este símbolo é descartado
- Um AFN aceita uma cadeia de entrada s se, e somente se, existe um caminho no grafo de transições que parte do estado inicial e que termina em algum estado de aceitação
- Pode existir mais de um caminho que leva a um estado de aceitação
- A linguagem definida por um AFN é o conjunto de cadeias que são aceitas

AFN da linguagem $aa^* \mid bb^*$



AFN da linguagem $aa^* \mid bb^*$



Caminho que aceita a cadeia aa

$$0 \xrightarrow{\epsilon} 1 \xrightarrow{a} 2 \xrightarrow{a} 2$$

Autômatos finitos determinísticos

Definição de AFD

Um autômato finito determinístico (AFD) é um caso especial de AFN no qual

1. nenhum estado possui um transição rotulada pelo símbolo ϵ (denominada transição-ϵ); e
2. para cada estado s existe no máximo uma transição rotulada com o caractere c partindo de s.

Observação: em um AFD, cada entrada da tabela de transições contém um único estado, o que simplifica o processo de verificação de aceitação de uma cadeia.

Pseudocódigo para verificação de cadeias por meio de um AFD

Input: Uma cadeia de entrada x terminada por um `EOF`

Output: `"Sim"`, caso a cadeia seja uma sentença válida da linguagem, ou `"Nao"`, caso contrário

$s \leftarrow s_0$
$c \leftarrow \textsc{próximoCaractere}(\,)$

while $c \neq$ `EOF` **do**
    $s \leftarrow \textsc{transição}(s, c)$
    $c \leftarrow \textsc{próximoCaractere}(\,)$

if $s \in F$ **then**
    return `"Sim"`
else
    return `"Não"`

Conversão de um AFN em um AFD

- Dado um AFN, é possivel determinar um AFD que reconheça a mesma linguagem
- A ideia central da conversão de um AFN para um AFD é fazer com que cada estado do AFD corresponda a um conjunto de estados do AFN
- Seja N um AFN e D um AFD
- O primeiro passo para a conversão é construir uma tabela de transições $Dtrans$ para D
- Cada estado do AFD corresponderá a um conjunto de estados do AFN e $Dtrans$ será construída de forma a simular, "em paralelo", todas as possíveis transições de N para uma dada entrada
- Para esta tarefa, são necessárias algumas operações que envolvem um estado s de N e um conjunto T de estados de N

Operações sobre os estados de um AFN

Operação	**Descrição**
FECHAMENTO-$\epsilon(s)$	Conjunto de estados do AFN atingíveis a partir do estado s somente por meio de transições-ϵ
FECHAMENTO-$\epsilon(T)$	Conjunto de estados do AFN atingíveis a partir de algum estado $s \in T$ somente por meio de transições-ϵ
MOVIMENTO(T, a)	Conjunto de estados do AFN para o qual existe uma transição partindo de $s \in T$ cujo rótulo é o símbolo da entrada a

Relação entre as operações sobre os estados de um AFN

- Antes mesmo de ver o primeiro símbolo da entrada, N pode estar em qualquer estado pertencente a $\textsc{fechamento-}\epsilon(s_0)$, onde s_0 é o estado inicial
- Seja T o conjunto de todos os estados atingíveis a partir de s_0 e que a seja o próximo símbolo da entrada
- Ao ver a, N pode seguir para qualquer estado em $\textsc{movimento}(T, a)$
- Se existem transições-ϵ, N pode estar em qualquer estado em $\textsc{fechamento-}\epsilon(M)$, onde $M = \textsc{movimento}(T, a)$, após ver a

$Estados$-D

- O conjunto de todos os estados de D é denominado $Estados$-D
- Cada estado de D corresponde a um conjunto de estados de AFN que poderiam ser atingidos em N após uma sequência de símbolos da entrada, incluindo as possíveis transições-ϵ antes ou depois dos símbolos serem vistos
- O estado de partida de D é $\textsc{fechamento-}\epsilon(s_0)$
- Os demais estados são adicionados segundo o algoritmo descrito a seguir
- Os estados de aceitação de D são aqueles cujo conjunto de estados de AFN que ele representa contém ao menos um estado de aceitação de N

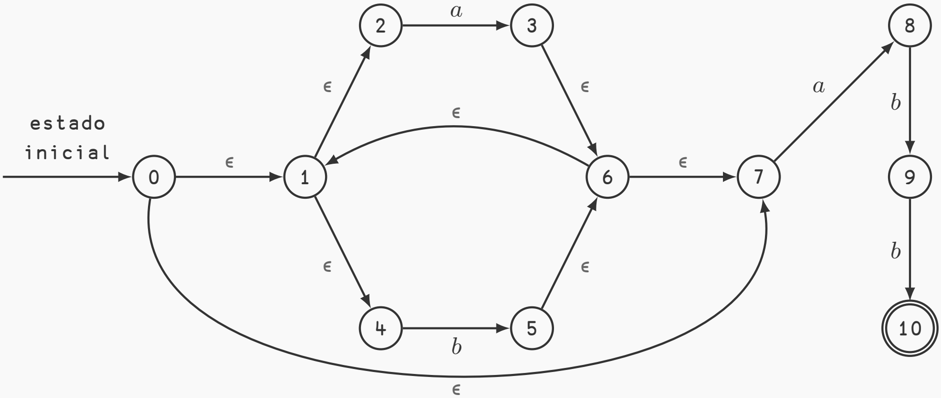
Construção de subconjuntos

$F \leftarrow$ FECHAMENTO-$\epsilon(s_0)$
Desmarque F
Inclua em F em $Estados$-D
while existe um estado $T \in Estados$-D não marcado **do**
    Marque T
    for cada símbolo de entrada a **do**
        $M \leftarrow$ MOVIMENTO(T, a)
        $U \leftarrow$ FECHAMENTO-$\epsilon(M)$
        if $U \notin Estados$-D **then**
            Desmarque U
            Inclua U em $Estados$-D
        $Dtrans[T, a] \leftarrow U$

Cálculo de fechamento-$\epsilon(T)$

Seja P uma pilha
Empilhe em P todos os estados em T
Inclua T em FECHAMENTO-$\epsilon(T)$
while P não estiver vazia **do**
    Desempilhe o topo t de P
    for cada estado u que é chegada de uma aresta partindo de t com rótulo ϵ **do**
        if $u \notin$ FECHAMENTO-$\epsilon(T)$ **then**
            Inclua u em FECHAMENTO-$\epsilon(T)$
            Empilhe u

AFN para a linguagem $(a \mid b)^*abb$



Convertendo o AFN em um AFD

- O estado inicial do AFD será o estado

$$\texttt{A} = \textsc{fechamento-}\epsilon(\texttt{0}) = \{\texttt{0, 1, 2, 4, 7}\}$$

- Veja que no cálculo de $\textsc{fechamento-}\epsilon(\texttt{0})$, o estado `0` é atingível por meio de um caminho vazio
- O alfabeto da linguagem é $\Sigma = \{\ a, b\ \}$
- Seguindo o algoritmo, o próximo estado será $\texttt{B} = \textsc{fechamento-}\epsilon(M)$, onde $M = \textsc{movimento}(A, a)$
- Segue que

$$M = \textsc{movimento}(A, a) = \textsc{movimento}(\{\texttt{0, 1, 2, 4, 7}\}, a) = \{\texttt{3, 8}\}$$

Convertendo o AFN em um AFD

- Daí

$$\texttt{B} = \textsc{fechamento-}\epsilon(\{\texttt{3, 8}\}) = \{\texttt{1, 2, 3, 4, 6, 7, 8}\}$$

- A transição entre estes dois estados deve ser registrada na tabela $Dtrans$:

$$Dtrans[\texttt{A}, a] = \texttt{B}$$

- O próximo estado `C` é computado a partir de $\textsc{movimento}(\texttt{A}, b) = \{\texttt{5}\}$:

$$\texttt{C} = \textsc{fechamento-}\epsilon(\{\texttt{5}\}) = \{\texttt{1, 2, 4, 5, 6, 7}\}$$

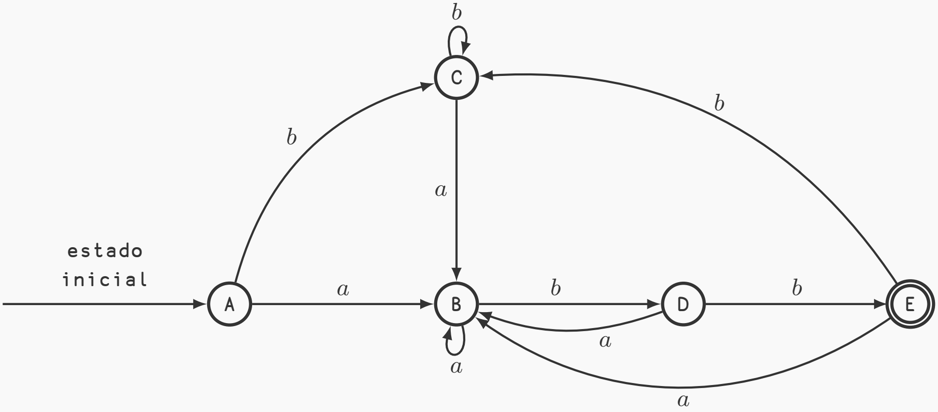
Convertendo o AFN em um AFD

- Assim, $Dtrans[\texttt{A}, b] = \texttt{C}$
- Seguindo o algoritmo, apenas mais dois estados distintos serão encontrado:

$$\texttt{D} = \{\texttt{1, 2, 4, 5, 6, 7, 9}\}$$
$$\texttt{E} = \{\texttt{1, 2, 4, 5, 6, 7, 10}\}$$

- Dentre os estados identificados, apenas `E` é estado de aceitação, pois contém o estado `10` de N
- No pior caso, seriam identificados 2^n novos estados, onde n é o número de estados do AFN

Resultado da conversão do AFN para um AFD



Referências

1. **AHO**, Alfred V, **SETHI**, Ravi, **ULLMAN**, Jeffrey D. *Compiladores: Princípios, Técnicas e Ferramentas*, LTC Editora, 1995.