Notas de Aula em Linguagens Formais e Autômatos: Linguagens Regulares

Maurício Marengoni

Centro Universitário SENAC
mauricio.marengoni@sp.senac.br
http://www.sp.senac.br

1 Autômato Finito

Existem vários tipos de **modelos computacionais** e, como todo modelo, eles são semelhantes a computadores reais em alguns aspectos e não tão semelhantes assim em outros aspectos. O modelo computacinal mais simples é o **autômato finito** ou a **máquina de estados finitos**.

Um autômato finito é um modelo de computador com uma memória extremamente limitada, na prática, sistemas deste tipo existem em vários locais, como um sistema de abrir e fechar portas automaticamente, ou uma escada rolante que funciona apenas quando existe algum usuário.

Exemplo: escada rolante automática. Estados: parada ou funcionando. Trasições: se está parada e chega usuário passa para funcionando; se está funcionando e chega usuário, continua funcionando; se está parada e não tem usuário continua parada; se está funcionando e não tem usuário vai para parada.

Table 1. Tabela resumindo as transições de estado da escada rolante automática.

Evento	Estado	
	Parada	Funcionando
Chega usuário	Funcionando	Funcionando
Sem usuário	Parada	Parada

Autômatos finitos e Cadeias de Markov (que são autômatos finitos probabilísticos) são ferramentas importantes para reconhecer padrões em dados. Um autômato finito pode ser representado graficamente por um **diagrama de estados** conforme indicado na Figura 2

Esta máquina possui três **estados** marcados por $\{1, 2, 3\}$, o **estado inicial** da máquina $\{1\}$ indicado pela $\rightarrow$ chegando ao estado de lugar algum, o **estado de aceite** ou **estado final** da máquina $\{2\}$, indicado pelo círculo duplo. As setas indicadas no diagrama mostram as **transições** possíveis na máquina. A saída de autômato finito é sempre uma condição de **aceite** ou **rejeite**. O processamento

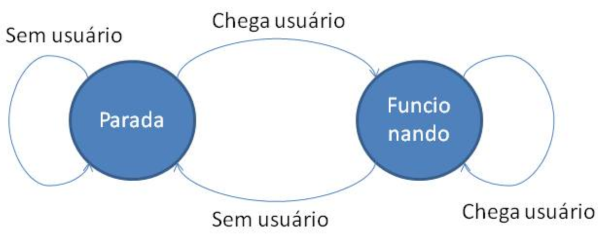

Fig. 1. Representação gráfica do exemplo de uma escada rolante automática.

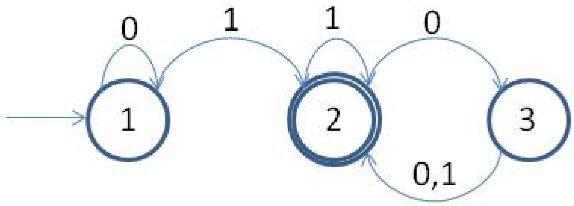

Fig. 2. Diagrama de estados do autômato finito M1.

de um autômato finito é simples, dada uma palavra na entrada da máquina, ele processa cada símbolo da palavra separadamente, fazendo transições entre estados da máquina. Se ao final da palavra o autômato estiver num estado de aceite, então o autômato aceita aquela palavra, caso contrário ele rejeita a palavra.

Exemplo: O que faz M1 com a palavra 01100?

1. Inicia a computação no estado 1.
2. Lê o primeiro símbolo da palavra, 0, e faz a transição indicada, neste caso fica no próprio estado 1.
3. Lê o segundo símbolo da palavra, 1, e faz a transição para o estado 2.
4. Lê o terceiro símbolo da palavra, 1, e faz a transição, permanecendo no estado 2.
5. Lê o quarto símbolo da palavra, 0, e faz a transição para o estado 3.
6. Lê o quinto símbolo da palavra, 0, e faz a transição para o estado 2.
7. Como não existem mais símbolos a serem lidos e o autômato encontra-se num estado de aceite, a palavra é aceita pelo autômato.

Definição Formal: um **autômato finito** é uma 5-tupla $(Q, \Sigma, \delta, q_0, F)$ onde:

- Q é o conjunto finito de estados da máquina.
- Σ é o conjunto finito de símbolos chamado de alfabeto.
- $\delta : Q$ x $\Sigma \rightarrow Q$ é a função de transição da máquina.
- q_0 é o estado inicial da máquina.
- $F \subseteq Q$ é o conjunto finito de estados de aceite da máquina.

Pode-se então descrever qualquer autômato definindo a 5-tupla indicada acima, no caso de M1 tem-se: $M1 = (Q, \Sigma, \delta, q_0, F)$, onde

- $Q = \{1, 2, 3\}$
- $\Sigma = \{0, 1\}$
- $q_0 = \{1\}$
- $F = \{2\}$
- δ é definida como:

Table 2. Tabela com a função de transição da máquina M1.

Símbolos	Estados		
	1	2	3
0	1	3	2
1	2	2	2

1.1 Definição de Computação

Uma definição formal é necessária para transformar as noções adquiridas em descrição precisa e livre de ambiguidades. Matematicamente a definição formal de uma computação em um autômato finito é dada por: seja $M = (Q, \Sigma, \delta, q_0, F)$ um aotômato finito e seja $w = w_1 w_2 \ldots w_n$ uma palavra onde $w_i \in \Sigma, 1 \leq i \leq n$. Dizemos que M aceita w se existir uma sequencia de estados $e_0, e_1, \ldots, e_p \in Q$ que atendam às seguintes condições:

1. $e_0 = q_0$, isto é, a computação inicia no estado inicial.
2. $\delta(e_i, w_{i+1}) = e_{i+1}$ para todo $0 \leq i \leq n - 1$, a sequencia de estados pela qual o autômato passa é dada pela função de transição, e
3. $e_p \in F$, o estado final da sequencia esta no conjunto de estados de aceite.

Dizemos que M **reconhece uma linguagem** A se A é formada por palavras que são aceitas por M, isto é, $A = \{w | M aceita w\}$.

Definição: uma linguagem é chamada de **linguagem regular** se existir um autômato finito que reconhece a linguagem.

1.2 Projetando autômatos finitos

Uma forma de se projetar um autômato finito é procurar se colocar no lugar da máquina e, uma vez verificando o símbolo que está sendo lido verificar qual o estado em que devemos estar, desta forma podemos determinar não apenas o número de estados que uma máquina deve ter, mas também a forma como a máquina se comporta e faz as transições entre estados.

Exemplo: construir um autômato finito que aceite palavras que possuam um número par de 0s.

1. Assumindo que o alfabeto que esta sendo considerado seja dado por $\Sigma = \{0, 1\}$, para qualquer palavra formada usando este alfabeto so existem duas possibilidades, ou a palavra tem um número impar de 0s ou a palavra tem um número par de 0s. Logo existem apenas dois estados nesta máquina conforme indicaso na Figura 3:

Fig. 3. Estados possíveis para a máquina.

2. Uma vez definidos os estados que a máquina possui temos que verificar qual é o estado inicial da máquina e quais são os estados finais da máquina. Neste caso, no inicio do processamento a máquina ainda não viu símbolo algum, portanto a máquina leu zero 0s, um número par de zeros, portanto o estado **par** é o estado inicial da máquina. Como desejamos uma máquina que aceite apenas palavras que contenham um número par de 0s, o conjunto de estados finais, neste caso, possui um único estado, que é o estado **par**. Conforme apresentado na Figura 4

Fig. 4. Estados inicial e final para a máquina.

3. Em seguida podemos verificar o que ocorre em cada um dos estados quando a máquina le um dos símbolos do alfabeto. Inicialmente vamos verificar o que ocorre se, num determinado estado, a máquina le o símbolo 1. Note que neste caso o número de 0s lidos pela máquina não se altera, portanto, para o símbolo 1 a máquina permanece no estado onde ela se encontra, conforme indicado na Figura 5.

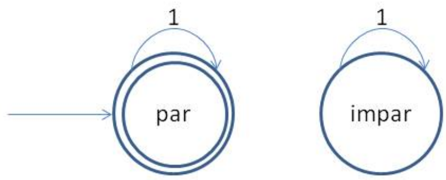

Fig. 5. Transição em cada estado para o símbolo 1.

4. Finalmente verificamos o que ocorre em cada estado para o símbolo 0. Se a máquina, até o momento leu um número par de 0s e ela lê mais um 0, então ela deve passar para o estado que indica um número impar de 0s lidos, e vice versa, conforme indicado na Figura 6.

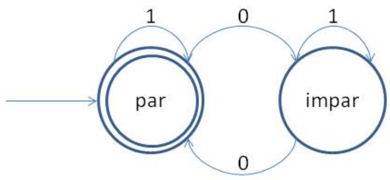

Fig. 6. Transição em cada estado para o símbolo 0.

5. Como não há mais símbolos no alfabeto, a máquina esta completa e deve atender as especificações da linguagem. Faça alguns testes voce mesmo para verificar o funcinamento da máquina. O que mudaria se quisessemos um número impar de 0s? E se quisessemos um número par de 1s?

1.3 Operações Regulares

2 Não Determinismo

2.1 Equivalência entre AFD e AFN

3 Expressões Regulares

3.1 Equivalência com Autômatos

4 Linguagens Não Regulares e o Lema do Bombeamento

References

1. Sipser, M.: Introdução à Teoria da Computação, Thomson, 2a edição americana, 2007.
2. Sipser, M.: Introduction to the Theory of Computation, Thomson, 2th edition, 2006.
3. Menezes, P.B.: Linguagens Formais e Autômatos, Série Livros Didáticos Instituto de Informática de UFRGS, Sagra Luzzatto 2005.
4. Vieira, N.J.: Introdução aos Fundamentos da Computação, Linguagens e Máquinas, Scott, Thomson, 2006.
5. Solow, D.: How to Read and Do Proofs, an introduction to mathematical thought processes, John Wiley and Sons, 2nd edition, 1990.