Máquinas de Turing com JFLAP

Filipe Borba Matheus Bittencourt

25 de maio de 2017

1 Máquinas de fita única

(a) $L = \{a^i b^j c^k \mid i, j, k \in \mathbb{N} \text{ e } i \times j = k\}$

Para cada 'a', trocamos ele por 'A', depois para cada 'b' testamos se temos um 'c' correspondente, fazendo isso para todos os 'a's. Ao final da marcação de letras C, as letras B são desmarcadas e a próxima letra A é marcada, e assim por diante. Se a multiplicação não apresentar seu resultado correto, lembrando que todas as letras precisam aparecer pelo menos uma vez, a máquina rejeitará a entrada.

(b) $L = \{\#x_1\#x_2\#\ldots\#x_n \mid x_i \in \{0,1\}^* \text{ e } x_i \neq x_j \text{ para cada } i \neq j\}$

A máquina compara x_i e x_j, $\forall i \neq j$, exceto pela palavra vazia, garantida no início do procedimento. Após esta garantia, um x_i é fixado e comparado com $x_{i+1}, x_{i+2}, \cdots, x_n$. Não havendo subpalavras iguais nesta iteração a próxima subpalavra, x_{i+1}, é fixada e comparada com os elementos posteriores, separados pelo #, até que este não seja mais encontrado na entrada, significando o fim da mesma.

2 Máquinas com múltiplas fitas

(a) $L = \{ww \mid w \in \{0,1\}^*\}$

Para computar essa linguagem usando 2 fitas, movemos o primeiro símbolo para a segunda fita, e comparamos as duas fitas, se elas possuirem os mesmos símbolos, aceitamos a *string*, senão retrocedemos a primeira fita até o novo primeiro símbolo, copiamos esse símbolo para a segunda fita, e repetimos o processo de comparação das duas fitas. Repetimos esses passos até tenhamos duas *strings* semelhantes nas duas fitas, aceitando a *string*, ou até que a primeira fita esteja vazia, rejeitando a *string*.

(b) $L = \{a^n b^n c^n \mid n \geq 0\}$

Usando 2 fitas, copiamos todos os 'a's para a segunda fita, retrocedemos o cabeçote da segunda fita até o início dela. Depois, para cada 'b' na primeira fita checamos se existe um 'a' na segunda fita. Retrocedemos o cabeçote da segunda fita, e repetimos o processo para os 'c's. Se chegarmos ao final da primeira fita aceitamos a *string*.

3 Máquinas em bloco

(a) $L = \{0^{2^n} \mid n \geq 0\}$

Se tirarmos o primeiro zero de uma *string* w tal que $|w| = 2^n$, então sobrará um número ímpar de zeros, e sempre que dividirmos esse número de zeros por 2 (truncando o resultado da divisão) obteremos um número ímpar também. Se dividirmos $n-1$ vezes por 2, o resultado será sempre 1. Prova:

$$\lfloor \frac{2^n - 1}{2^{n-1}} \rfloor = 1 \tag{1}$$

$$\lfloor \frac{2^n}{2^{n-1}} - \frac{1}{2^{n-1}} \rfloor = 1 \tag{2}$$

$$\lfloor 2 - \frac{1}{2^{n-1}} \rfloor = 1 \tag{3}$$

sendo $n \geq 1$, então $0 < \frac{1}{2^{n-1}} \leq 1$ portanto 3 está correto.

Tendo isso em mente, para computar essa linguagem, apagamos o primeiro '0' da fita , trocamos o próximo '0' da fita por 'x' e pulamos um '0', marcando apenas metade dos zeros. Se nessa passada pela fita marcando os '0's tivermos um número par de '0's, rejeitamos a *string*, senão continuamos o processo até que toda a *string* seja marcada por 'x's.

(b) Um somador binário

Assumindo que a entrada dessa máquina seja algo como "1001+1010", para computar a soma, caminhamos até o final da fita, apagamos o número que está lá, caminhamos até o bit menos significativo do primeiro número, que ainda não foi marcado, e marcamos o valor já adicionado, que se for 0, marcamos 'a', e 1 marcamos 'b'. Se a soma desses dois bits gerar um *carry* continuamos somando nos números adjacentes, porém sem marcá-los, até que não tenha mais esse *carry*. Repetimos esse processo até que o segundo número esteja vazio e depois trocamos todos os 'a's e 'b's por '0's e '1's, respectivamente.