MAC 4722 - Linguagens, Autômatos e Computabilidade

Rodrigo Augusto Dias Faria - NUSP 9374992
Departamento de Ciência da Computação - IME/USP

4 de junho de 2016

Lista 6

L6.1 (Sipser 3.06) No Teorema 3.21 mostramos que uma linguagem é Turing-reconhecível sse algum enumerador a enumera. Por que não usamos o seguinte algoritmo mais simples para a direção de ida da prova? Tal qual anteriormente, $s_1, s_2, \ldots$ é uma lista de todas as cadeias em Σ^*.

$E =$ "Ignore a entrada.

1. Repita o que se segue para $i = 1, 2, 3, \ldots$

2. Rode M sobre s_i.

3. Se ela aceita, imprima s_i."

Resposta: O algoritmo indica que devemos rodar M sobre todas as cadeias possíveis de Σ^*. A mudança sugerida no enunciado faz com que M execute em uma cadeia s_i por vez e, caso M aceite s_i, o enumerador E imprime-na. Ocorre que M pode entrar em *loop* para uma cadeia s_k qualquer e, por conseguinte, nenhuma outra cadeia subsequente a s_k será impressa por E e, portanto, a linguagem de M será diferente do conjunto de cadeias listadas por E.

L6.2 (Sipser 3.08) Dê descrições a nível de implementação de máquinas de Turing que decidem as linguagens abaixo sobre o alfabeto $\{0, 1\}$.

a. $\{w \mid w$ contém o mesmo número de 0s e 1s$\}$

Resposta: Vamos chamar de M_a a MT que decide a linguagem em **a.** Note que o primeiro passo deve aceitar a cadeia vazia pois, neste caso, o número de 0s e 1s é igual a zero.

$M_a =$ "Sobre a cadeia de entrada w:

1. Se w é a cadeia vazia, *aceite*, caso contrário, vá para o passo 2.

2. Faça uma varredura na fita e marque o primeiro 0 que ainda não esteja marcado. Se nenhum 0 desmarcado foi encontrado, *rejeite*. Retorne a cabeça para a extremidade esquerda da fita.

3. Faça uma varredura na fita e marque o primeiro 1 que ainda não esteja marcado. Se a varredura não encontrou nenhum 1 desmarcado, *rejeite*. Retorne a cabeça para a extremidade esquerda da fita.
4. Faça uma nova varredura na fita. Se um 0 ou um 1 desmarcado for encontrado, mova a cabeça para a extremidade esquerda da fita e retorne ao passo 2, caso contrário, *aceite*.

b. $\{w \mid w$ contém duas vezes mais 0s que 1s$\}$

Resposta: Vamos chamar de M_b a MT que decide a linguagem em **b.** A estratégia utilizada aqui é similar à linguagem do item **a.**

$M_b =$ "Sobre a cadeia de entrada w:

1. Repita por duas vezes o próximo estágio:
2. Faça uma varredura na fita e marque o primeiro 0 que ainda não esteja marcado. Se nenhum 0 desmarcado foi encontrado, *rejeite*. Retorne a cabeça para a extremidade esquerda da fita.
3. Faça uma varredura na fita e marque o primeiro 1 que ainda não esteja marcado. Se a varredura não encontrou nenhum 1 desmarcado, *rejeite*. Retorne a cabeça para a extremidade esquerda da fita.
4. Faça uma nova varredura na fita. Se um 0 ou um 1 desmarcado for encontrado, mova a cabeça para a extremidade esquerda da fita e retorne ao passo 1, caso contrário, *aceite*.

c. $\{w \mid w$ **não** contém duas vezes mais 0s que 1s$\}$

Resposta: Vamos chamar de M_c a MT que decide a linguagem em **c.** A estratégia utilizada aqui é aplicar a MT obtida em **b.** como uma subrotina.

$M_c =$ "Sobre a cadeia de entrada w:

1. Rode w na máquina M_b.
2. Se M_b aceita w, *rejeite*, senão *aceite*.

L6.3 (Sipser 3.14) Um ***autômato com fila*** é como um autômato com pilha, exceto que a pilha é substituída por uma fila. Uma ***fila*** é uma fita que permite que símbolos sejam escritos somente na extremidade esquerda e lidos somente da extremidade direita. Cada operação de escrita (denominá-la-emos *empurrar*) adiciona um símbolo na extremidade esquerda da fila e cada operação de leitura (denominá-la-emos *puxar*) lê e remove um símbolo na extremidade direita. Como com um `AP`, a entrada é colocada numa fita de entrada de somente-leitura separada, e a cabeça sobre a fita de entrada pode mover somente da esquerda para a direita. A fita de entrada contem uma célula com um símbolo em branco após a entrada, de modo que essa extremidade da entrada possa ser detectada. Um autômato com fila aceita sua entrada entrando num estado especial de aceitação em qualquer momento. Mostre que uma linguagem pode ser reconhecida por um autômato com fila determinístico **sse** a linguagem é Turing-reconhecível.

Resposta: TODO

L6.4 (Sipser 3.16) Mostre que a coleção de linguagens Turing-reconhecíveis é fechada sob a operação de união, concatenação, estrela e intersecção.

Resposta: Sejam L_1 e L_2 linguagens Turing-reconhecíveis e M_1 e M_2 máquinas de Turing que reconhecem L_1 e L_2, respectivamente. Vamos mostrar que a classe de linguagens Turing-reconhecíveis é fechada sob as seguintes operações:

a. união

Vamos construir uma máquina de Turing M capaz de reconhecer a união $L_1 \cup L_2$ da seguinte forma:

$M =$ "Sobre a cadeia de entrada w:

1. Repita o seguinte para $i = 1, 2, 3, \ldots$

2. Rode as máquinas M_1 e M_2 sobre w por i passos. Se M_1 ou M_2 aceita w, *aceite*. Se M_1 e M_2 param e rejeitam w, então *rejeite*."

A máquina construída reconhece cadeias de $L_1 \cup L_2$, pois se M_1 ou M_2 aceitam w, em algum momento a máquina M aceitará w, já que a máquina vai, passo a passo, tentando reconhecer w em M_1 e M_2 simultaneamente. M poderá entrar em *loop* se M_1 ou M_2 entrar em *loop* e ambas rejeitarem w.

b. concatenação

Vamos construir uma máquina de Turing M capaz de reconhecer a concatenação L_1L_2. Seja w uma cadeia de comprimento n, tal que pode ser particionada em duas partes. Seja $p_i = x_i y_{n-1}$ cada uma dessas possíveis partições, onde $i = 0, 1, \ldots, n$ representa a quantidade de caracteres de w (a partir do início) na primeira parte da partição e n–i representa a quantidade de caracteres de w (a partir de $i + 1$) na segunda parte da partição. Vamos inserir na fita da máquina M cada uma das p_i partições separadas por um símbolo, digamos #, da forma $\#p_0\#p_1\# \ldots p_n\#$, e cada parte da partição p_i separada por um outro caractere, digamos β, da forma $x_i\beta y_{n-i}$, ou seja, a fita ficaria da forma $\#x_0\beta y_n\#x_1\beta y_{n-1}\# \ldots \#x_n\beta y_0$. O que faremos é rodar a máquina M_1, simultaneamente, em todas as primeiras partes x_i de cada partição p_i e a máquina M_2 em todas as segundas partes y_{n-i} de cada partição p_i. Se M_1 e M_2 aceitar alguma partição p_i (M_1 aceitando a primeira parte e M_2 aceitando a segunda), então M *aceita* w. Se para todo i, M_1 e M_2 rejeitar p_i (M_1 rejeitando a primeira parte ou M_2 rejeitando a segunda), então M *rejeita* w. A máquina M pode entrar em *loop* se $w \notin L_1L_2$ e M_1 ou M_2 entrar em *loop* na tentativa de reconhecimento das partições.

c. estrela

Vamos construir uma máquina de Turing M capaz de reconhecer a operação estrela L_1^*. Seja w uma cadeia de comprimento n, tal que pode ser particionada em m partes. Seja $p_i = w'_{i1}w'_{i2} \ldots w'_{im}$, para $i = 0, 1, \ldots$ cada uma dessas possíveis partições. Vamos inserir na fita da máquina M cada uma das p_i partições separadas por um símbolo, digamos #, da forma $\#p_0\#p_1\# \ldots p_n\#$, e cada parte da partição p_i separada por um outro caractere, digamos β, da forma $w'_{i1}\beta w'_{i2} \ldots$, ou seja, a fita ficaria da forma

$\#w'_{01}\beta w'_{02}\beta \dots w'_{0m}\#w'_{11}\beta w'_{12}\beta \dots \#$. O que faremos é rodar a máquina M_1 simultaneamente em todas as partes $w'_{i1}w'_{i2}\dots w'_{im}$ de cada partição p_i. Se M_1 aceitar alguma partição p_i (M_1 aceitando cada uma das partes de p_i), então M *aceita* w. Se para todo i, M_1 rejeitar p_i, então M *rejeita* w. A máquina M pode entrar em *loop* se $w \notin L_1^*$ e M_1 entrar em *loop* na tentativa de reconhecimento das partições.

d. intersecção

Vamos construir uma máquina de Turing M capaz de reconhecer a intersecção $L_1 \cap L_2$ da seguinte forma:

$M =$ "Sobre a cadeia de entrada w:

1. Repita o seguinte para $i = 1, 2, 3, \dots$

2. Rode as máquinas M_1 e M_2 sobre w por i passos. Se M_1 e M_2 aceitam w, *aceite*. Se M_1 ou M_2 para e rejeita w, então *rejeite*."

A máquina construída aceita cadeias de $L_1 \cap L_2$, pois se $w \in L_1 \cap L_2$, significa que $\exists w \mid w \in L_1$ e $w \in L_2$. Como a máquina testa w em M_1 e M_2, e só aceita caso w seja aceita por ambas, temos que a máquina M reconhece as cadeias de $L_1 \cap L_2$. M poderá entrar em *loop* se M_1 ou M_2 entrar em *loop* e rejeitar w.