Máquinas de Turing: uma introdução

Nelma Moreira
Armando Matos
Departamento de Ciência de Computadores
Faculdade de Ciências, Universidade do Porto
email: {nam,acm}@ncc.up.pt

1996
Revisão: Maio 2001

1 Máquinas de Turing

Começamos por definir um tipo básico de máquina de Turing. Como veremos, existem muitos outros modelos, todos equivalentes a este.

Uma **máquina de Turing** (MT) é constituída por um controlo finito (conjunto finito de estados), uma fita dividida em células, e uma cabeça de leitura/escrita (RW) que actua sobre uma célula da fita de cada vez.

Supomos que a fita é infinita para a esquerda e para a direita. No início, supomos que os dados – sequência finita de símbolos de um alfabeto – se encontram escritos na fita, um símbolo em cada célula, e as restantes células da fita contém um caracter especial da fita, designado por caracter "branco".

$\ldots$	$\bullet$	a_1	$\ldots$	a_i	$\ldots$	a_n	$\bullet$	$\ldots$

$$\uparrow$$

$$\boxed{\text{controlo finito}}$$

Num passo de computação (movimento), dependendo do símbolo lido na fita pela cabeça e do estado do controlo finito, a máquina

1. muda de estado
2. escreve um símbolo na célula que está debaixo da cabeça
3. move a cabeça para a esquerda ou para a direita

Formalmente, uma máquina de Turing (MT) é um tuplo

$$M = (S, \Sigma, \Gamma, \delta, s_0, b, F)$$

onde

S é um conjunto finito de estados

Γ é o conjunto finito de *símbolos da fita*

Σ é um subconjunto de Γ que não incluí $\bullet$, é o conjunto dos *símbolos de entrada*

δ é a *função de transição*, função parcial de $S \times \Gamma$ em $S \times \Gamma \times \{\rightarrow, \leftarrow\}$

s_0 é o estado inicial

b é um símbolo de Γ, designado por *branco*

$F \subseteq S$ é o conjunto de estados finais

Uma *descrição instantânea* (ID) ou *configuração* é representada por xsy, onde s é o estado corrente, $x \in \Gamma^\star$ é a sequência de símbolos à esquerda da cabeça, sendo o último símbolo de x o que está imediatamente à esquerda da cabeça, e $y \in \Gamma^\star$ é a sequência à direita da cabeça (o primeiro símbolo de y é o que está debaixo da cabeça). Supomos que x e y não contêm "brancos" desnecessários, isto é, xy é a sequência de caracteres da fita desde ou o caracter não-branco mais à esquerda ou o símbolo que está sobre a cabeça, conforme o que for mais à esquerda, e analogamente para o seu extremo direito.

A relação $\overset{M}{\vdash}$, um *movimento*, entre duas configurações é definida da seguinte forma. Seja

$$X_1 \cdots X_{i-1} s X_i \cdots X_n$$

uma configuração. Seja $\delta(s, X_i) = (s', Y, D)$. Se $i = 1$ e $D = \leftarrow$ então

$$sX_1 \cdots X_n \overset{M}{\vdash} s'bYX_{i+1} \cdots X_n$$

Se $i = 1$, $D = \leftarrow$ e $Y = b$ então

$$sX_1 \cdots X_n \overset{M}{\vdash} s'X_{i+1} \cdots X_n$$

Se $i > 1$ e $D = \leftarrow$ então,

$$X_1 \cdots X_{i-1} s X_i \cdots X_n \overset{M}{\vdash} X_1 \cdots X_{i-2} s' X_{i-1} Y X_{i+1} \cdots X_n$$

Analogamente se $D = \rightarrow$ tem-se que

$$X_1 \cdots X_{i-1} s X_i \cdots X_n \overset{M}{\vdash} X_1 \cdots X_{i-1} Y s' X_{i+1} \cdots X_n$$

Diz-se que $xsy \overset{M^k}{\vdash} x's'y'$ se $x's'y'$ resulta de xsy em k movimentos e $xsy \overset{M^\star}{\vdash} x's'y'$ se existe $k > 0$ tal que $xsy \overset{M^k}{\vdash} x's'y'$.

Uma palavra $x \in \Sigma^\star$ é **aceite** por uma máquina de Turing M se com palavra x na fita e no estado inicial s_0, a máquina usando a função de transição δ, M entra num **estado final**. **Note** que mal a máquina atinja um estado final a computação pára, independentemente de ter lido ou não todos os símbolos de entrada. Por outro lado, se para um estado não final e um símbolo lido pela cabeça, a função de transição δ não estiver definida a máquina pára sem aceitar os dados (a palavra x). Pode ainda acontecer que a máquina nunca pare ... caso em que também não aceita a palavra x. Formalmente,

A *linguagem aceite por* M, é

$$L(M) = \{x \mid \ x \in \Sigma^\star \text{ e } s_0 x \overset{M^\star}{\vdash} x_1 s x_2 \text{ para algum } s \in F, \text{ e } x_1, x_2 \in \Gamma^\star\}$$

Exemplo 1.1 *A máquina de Turing definida por,*

$$M = (\{s_0, s_1, s_2, s_3\}, \{0,1\}, \{0,1,\bullet,X\}, \bullet, s_0, \{s_4\})$$

$$\begin{array}{lll}
\delta(s_0,0) = (s_2, X, \rightarrow) & \delta(s_0,1) = (s_1, X, \rightarrow) & \delta(s_0,X) = (s_0, X, \rightarrow) \\
\delta(s_0,\bullet) = (s_4, \bullet, \leftarrow) & \delta(s_2,0) = (s_2, 0, \rightarrow) & \delta(s_2,1) = (s_3, X, \leftarrow) \\
\delta(s_2,X) = (s_2, X, \rightarrow) & \delta(s_1,1) = (s_1, 1, \rightarrow) & \delta(s_1,0) = (s_3, X, \leftarrow) \\
\delta(s_1,X) = (s_1, X, \rightarrow) & \delta(s_3,0) = (s_3, 0, \leftarrow) & \delta(s_3,1) = (s_3, 1, \leftarrow) \\
\delta(s_3,X) = (s_3, X, \leftarrow) & \delta(s_3,\bullet) = (s_0, \bullet, \rightarrow). &
\end{array}$$

aceita a linguagem

$$L = \{x \in \{0,1\}^\star \mid x \textit{ tem igual número de 1's e de 0's}\}$$

No estado s_0 se a cabeça da máquina lê 0 (1), a máquina entra no estado s_2 (s_1), substitui o 0 (1) por X – indicando que o "apagou"– e procura um 1 (0), deslocando-se para a direita. No estado s_0 se encontra um caracter "branco", $\bullet$, entra no estado final s_4 e pára (significa que aceitou os dados!). Se no estado s_2 (s_1) encontra um 1 (um 0) substituí-o por um símbolo de fita X -indicando que o "apagou"– e desloca-se para a esquerda, entrando no estado s_3 até encontrar um "branco". Quando encontra um "branco" passa para o estado s_0. E repete o ciclo enquanto houverem 0's ou/e 1's. Termina com sucesso no estado s_4. Caso contrário, termina num estado não final. Esta máquina pára sempre!

Exemplo 1.2 *A máquina de Turing definida por*

$$M = (\{s_0, s_1, s_2, s_3, s_4\}, \{0,1\}, \{0,1,X,Y,\bullet\}, \delta, s_0, \bullet, \{s_4\})$$

onde,

$$\begin{array}{ll}
\delta(s_0,0) = (s_1, X, \rightarrow) & \delta(s_0,Y) = (s_3, Y, \rightarrow) \\
\delta(s_1,0) = (s_1, 0, \rightarrow) & \delta(s_1,1) = (s_2, Y, \leftarrow) \\
\delta(s_1,Y) = (s_1, Y, \rightarrow) & \delta(s_2,0) = (s_2, 0, \leftarrow) \\
\delta(s_2,X) = (s_0, X, \rightarrow) & \delta(s_2,Y) = (s_2, Y, \leftarrow) \\
\delta(s_3,Y) = (s_3, Y, \rightarrow) & \delta(s_3,\bullet) = (s_4, \bullet, \rightarrow)
\end{array}$$

aceita a linguagem

$$L = \{0^n 1^n \mid \ n \geq 1\}$$

Verifique!

Exemplo 1.3 *Sendo $L = \{ww^r \mid w \in \{0,1\}^\star\}$ onde w^r corresponde ao inverso de w , a máquina de Turing a seguir apresentada aceita L.*

$$M = (\{s_0, s_1, s_2, s_4, s_5, s_6, s_7\}, \{0,1\}, \{0,1,X,\bullet\}, \delta, s_0, \bullet, \{s_6\})$$

$$\begin{array}{lll}
\delta(s_0,0) = (s_1, X, \rightarrow) & \delta(s_0,1) = (s_2, X, \rightarrow) & \delta(s_0,X) = (s_0, X, \rightarrow) \\
\delta(s_0,\bullet) = (s_6, \bullet, \rightarrow) & \delta(s_1,0) = (s_1, 0, \rightarrow) & \delta(s_1,1) = (s_1, 1, \rightarrow) \\
\delta(s_1,\bullet) = (s_3, \bullet, \leftarrow) & \delta(s_2,0) = (s_2, 0, \rightarrow) & \delta(s_2,1) = (s_2, 1, \rightarrow) \\
\delta(s_2,\bullet) = (s_4, \bullet, \leftarrow) & \delta(s_3,0) = (s_5, \bullet, \leftarrow) & \delta(s_3,1) = (s_7, 1, \rightarrow) \\
\delta(s_3,X) = (s_7, \bullet, \rightarrow) & \delta(s_4,0) = (s_7, 1, \rightarrow) & \delta(s_4,1) = (s_5, \bullet, \leftarrow) \\
\delta(s_4,X) = (s_7, X, \rightarrow) & \delta(s_5,0) = (s_5, 0, \leftarrow) & \delta(s_5,1) = (s_5, 1, \leftarrow) \\
\delta(s_5,X) = (s_0, X, \rightarrow) & &
\end{array}$$

Se $x \in L$ a máquina pára no estado s_6.

Exercício 1.4 *Descreve uma máquina de Turing que aceite a linguagem* $\{0^n10^n \mid n \geq 1\}$. $\diamond$

Se $x \in L(M)$ então M pára quando atinge um estado final. Caso contrário, M pode não parar ou parar num estado não final.

Uma linguagem diz-se **recursivamente enumerável** (r.e) ou **semi-decidível** se é aceite por uma máquina de Turing. A classe de linguagens recursivamente enumeráveis designa-se por $\mathcal{SD}$.

Note-se que sendo M uma máquina de Turing, o problema $x \in L(M)$? não é decidível (i.e computável). Se for verdade, sendo x a sequência de entrada (dados) de M, esta pára num número finito de movimentos (passos). Mas se $x \notin L(M)$, M pode não parar e portanto não se obtém resposta.

Designam-se por **recursivas** ou **decidíveis** as linguagens para as quais existe uma máquina de Turing que a reconhece, isto é, que pára para todos os dados e que pára num estado final se a palavra dada pertence à linguagem e pára num estado não final se a palavra dada não pertence à linguagem. A classe de linguagens recursivas designa-se por $\mathcal{D}$.

Exercício 1.5 *Mostra que* $\mathcal{D} \subseteq \mathcal{SD}$. $\diamond$

Contudo, existem linguagens que não são sequer *semi-decidíveis* ...

Podemos, também, considerar máquinas de Turing *reconhecedoras* de linguagens, isto é, máquinas

$$M = (S, \Sigma, \Gamma, \bullet, \delta, s_0, \bullet, s_a, s_r)$$

onde há só dois estados finais, um estado de aceitação, s_a e um estado de rejeição, s_r. Neste caso, as máquinas param para quaisquer dados. Como veremos, isto equivale à máquina de Turing calcular uma função total de $\Sigma^\star$ em $\{0, 1\}$.

1.1 Máquinas de Turing calculadoras de funções parciais

Uma máquina de Turing pode ser vista como uma calculadora de funções parciais dos inteiros nos inteiros:

$$f : \mathbb{N}^k \longrightarrow_p \mathbb{N}$$

Suponhamos que os inteiros estão codificados em unário. Então, um inteiro $i \geq 0$ é representado por 1^i. Se uma função tiver k argumentos $i_1, \ldots, i_k$, a sequência de entrada pode ser $1^{i_1}0\ldots01^{i_k}$. Se a máquina de Turing pára com a sequência 1^m na fita, então $f(i_1, \ldots, i_k) = m$.

Uma função parcial f diz-se **recursiva** se existir uma máquina de Turing que calcula a função para todos os valores em que está definida não terminando a execução caso contrário.

Uma função total f diz-se *recursiva total* se existir uma máquina de Turing que calcula a função para todos os valores do argumento, isto é, a MT pára para quaisquer argumentos.

Note-se que as funções recursivas totais correspondem a linguagens recursivas.

Exemplo 1.6 *Seja*

$$m \dot{-} n = \begin{cases} m - n & \text{se } m \geq n \\ 0 & \text{caso contrário} \end{cases}$$

A máquina de Turing $M = (\{s_0, s_1, \ldots, s_6\}, \{0,1\}, \{0,1,\bullet\}, \delta, s_0, \bullet, \emptyset)$ *onde*

$$\begin{array}{lll}
\delta(s_0,1) = (s_1,\bullet,\rightarrow) & \delta(s_1,1) = (s_1,1,\rightarrow) & \delta(s_1,0) = (s_2,0,\rightarrow) \\
\delta(s_2,0) = (s_2,0,\rightarrow) & \delta(s_2,1) = (s_3,0,\leftarrow) & \delta(s_3,1) = (s_3,1,\leftarrow) \\
\delta(s_3,0) = (s_3,0,\leftarrow) & \delta(s_3,\bullet) = (s_0,\bullet,\rightarrow) & \delta(s_2,\bullet) = (s_4,\bullet,\leftarrow) \\
\delta(s_4,0) = (s_4,\bullet,\leftarrow) & \delta(s_4,1) = (s_4,1,\leftarrow) & \delta(s_4,\bullet) = (s_6,1,\rightarrow) \\
\delta(s_0,0) = (s_5,\bullet,\rightarrow) & \delta(s_5,0) = (s_5,\bullet,\rightarrow) & \delta(s_5,1) = (s_5,\bullet,\rightarrow) \\
\delta(s_5,\bullet) = (s_6,\bullet,\rightarrow) & &
\end{array}$$

calcula a função $m \dot{-} n$, isto é com dados 1^m01^n calcula $1^{m \dot{-} n}$. A máquina vai sucessivamente apagando um 1 da representação de m e substituindo um 1 por 0 na representação de n, até que uma das seguintes condições ocorre:

- quando procura um 1 na representação de n, não o encontra. Neste caso, $m > n$. Então substitui $n+1$ $0's$ por um 1 e por n símbolos $\bullet$, deixando $m \dot{-} n$ 1s na fita.
- quando começa um ciclo não encontra nenhum 1. Neste caso, $m < n$. Então apaga toda a fita.

Exemplo 1.7 *A seguinte máquina de Turing permite calcular a função* $f(n) = n + 1$*, supondo os inteiros codificados em "unário".*

$$M = (\{s_0, s_1\}, \{0,1\}, \{0,1,\bullet\}, \delta, s_0, \bullet, \{s_2\})$$

$\delta(s_0,1) = (s_0,1,\rightarrow)$	*move-se para a direita*
$\delta(s_0,\bullet) = (s_1,1,\leftarrow)$	*muda o primeiro branco por* 1
$\delta(s_1,1) = (s_1,1,\leftarrow)$	*move-se para a esquerda*
$\delta(s_1,\bullet) = (s_2,\bullet,\rightarrow)$	*pára no estado* s_2 *com a cabeça no primeiro dígito do resultado*

Exercício 1.8 *Descreve máquinas de Turing que permitam calcular as seguintes funções, supondo os inteiros codificados em "unário":*

(i) $f(n) = n^2$

(ii) $f(n,m) = n + m$

◇

Dum modo geral uma máquina de Turing M calcula uma função parcial f de $\Sigma^\star$ em $\Sigma^\star$ se dada como sequência de entrada $x \in \Sigma^\star$ a configuração final de M é $f(x)s\bullet$, para um estado final s.

1.1.1 Função característica duma linguagem

Seja $L \subseteq \Sigma^\star$ uma linguagem. A **função característica** da linguagem L é uma função $f : \Sigma^\star \rightarrow \{0,1\}$ tal que $f(x) = 1$ se $x \in L$ e $f(x) = 0$ se $x \notin L$.

Exercício 1.9 *Mostra que a função característica de uma linguagem é*

1. *recursiva total sse a linguagem é recursiva.*

2. *recursiva sse a linguagem é recursivamente enumerável.*

$\diamond$

1.2 Comparação com outros modelos de computação

Os modelos de computação podem ser caracterizados em termos do tipo de memória que possuem e como se tem acesso a ela. Em particular comparando as máquinas de Turing com os autómatos finitos e de pilha, temos o seguinte quadro:

Memória	Acesso	Modelo de Computação	Gramáticas	Linguagens
Finita	Fixo	Autómatos Finitos $AFD \equiv AFND \equiv AFND\epsilon$	Regulares	$\mathcal{R}$
Ilimitada	Primeiro elemento da Pilha	Autómatos de Pilha	Ind. de Contexto	$\mathcal{I_C}$
Ilimitada	Sequêncial (não restrito)	Máquinas de Turing $MTD \equiv MTND$	Tipo 0	$\mathcal{SD}$

com

$$\mathcal{R} \subset \mathcal{I_C} \subset \mathcal{SD}$$

Em resumo, todos os problemas que se podem resolver usando métodos computacionais (programas em linguagens de programação de uso geral) podem ser resolvidos por uma máquina de Turing ... é tudo uma questão de paciência... como se exemplifica nos exercícios a seguir propostos!

Exercício 1.10 *Descreve uma máquina de Turing que*

(a) *dada uma palavra x não vazia de $\{0,1\}^\star$ calcule o dobro do número binário por ela representado depois de eliminar os seus zeros não significativos.*

Inicial :

•	0	0	1	0	1	•	•	•

Final :

•	1	0	1	0	•	•	•	•

(b) *dados dois números x e y em unário, separados por um zero, calcule a sua soma em unário. Por exemplo se esses números fossem 3 e 2 as configurações da fita deveriam ser*

Inicial :

•	1	1	1	0	1	1	•	•

Final :

•	1	1	1	1	1	•	•	•

Sugestão: Procurar o 0 *e se a seguir não estiver um "branco", substituir o* 0 *por um* 1*. Se não encontrar o* 0 *os dados estão errados... A seguir localizar o último* 1 *e substituir por um "branco".*

(c) *dados dois números x e y em unário separados por um zero calcule o seu produto. Por exemplo se esses números fossem* 3 *e* 2 *as configurações da fita deveriam ser*

Inicial :

•	•	1	1	1	0	1	1	•	•

Final :

•	•	1	1	1	1	1	1	•	•

<u>*Uma solução:*</u>

$$M = (\{q_0, \ldots, q_{14}\}, \{1, 0\}, \{0, 1, \bullet, s, a\}, \delta, q_0, \bullet, \{q_{14}\})$$

$\delta(q_0, 1) = (q_1, 1, \leftarrow)$	$\delta(q_1, \bullet) = (q_2, s, \rightarrow)$	$\delta(q_2, \bullet) = (q_3, \bullet, \leftarrow)$	$\delta(q_2, s) = (q_2, s, \rightarrow)$
$\delta(q_2, 1) = (q_2, 1, \rightarrow)$	$\delta(q_2, 0) = (q_2, 0, \rightarrow)$	$\delta(q_2, a) = (q_2, a, \rightarrow)$	$\delta(q_3, 1) = (q_4, \bullet, \leftarrow)$
$\delta(q_3, 0) = (q_9, 0, \leftarrow)$	$\delta(q_4, 1) = (q_4, 1, \leftarrow)$	$\delta(q_4, 0) = (q_5, 0, \leftarrow)$	$\delta(q_5, s) = (q_8, s, \rightarrow)$
$\delta(q_5, 1) = (q_6, a, e)$	$\delta(q_5, a) = (q_5, a, e)$	$\delta(q_6, \bullet) = (q_7, 1, \rightarrow)$	$\delta(q_6, s) = (q_6, s, e)$
$\delta(q_6, 1) = (q_6, 1, e)$	$\delta(q_7, s) = (q_7, s, \rightarrow)$	$\delta(q_7, 1) = (q_7, 1, \rightarrow)$	$\delta(q_7, 0) = (q_5, 0, e)$
$\delta(q_7, a) = (q_7, a, \rightarrow)$	$\delta(q_8, \bullet) = (q_3, \bullet, e)$	$\delta(q_8, 1) = (q_8, 1, \rightarrow)$	$\delta(q_8, 0) = (q_8, 0, \rightarrow)$
$\delta(q_8, a) = (q_8, 1, \rightarrow)$	$\delta(q_9, s) = (q_{10}, \bullet, \rightarrow)$	$\delta(q_9, 1) = (q_9, 1, e)$	$\delta(q_10, \bullet) = (q_{11}, \bullet, \rightarrow)$
$\delta(q_{10}, 1) = (q_{10}, \bullet, \rightarrow)$	$\delta(q_{10}, 0) = (q_{10}, \bullet, \rightarrow)$	$\delta(q_{10}, a) = (q_{10}, \bullet, \rightarrow)$	$\delta(q_{11}, \bullet) = (q_{12}, \bullet, e)$
$\delta(q_{12}, \bullet) = (q_{12}, \bullet, e)$	$\delta(q_{12}, 1) = (q_{13}, 1, e)$	$\delta(q_{13}, 1) = (q_{13}, 1, e)$	$\delta(q_{13}, \bullet) = (q_{14}, \bullet, d)$

Esta máquina depois de efectuar o produto coloca a cabeça no primeiro dígito desse valor em unário (estados q_{11} a q_{14}).

(d) $\star$ *calcule a soma em binário de dois números em binário dados.*

(e) $\star$ *calcule o triplo em binário de um número em binário dado.*

(f) $\star$ *calcule o produto em binário de dois números em binário dados.*

(g) $\star$ *simule o "funcionamento"de um autómato finito determinístico de alfabeto $\{0, 1\}$. Suponha que a descrição do autómato é dada na parte inicial da fita. Cada estado é representado ou por $[X]$ ou por $[X0]$ conforme é ou não final, sendo X um número em unário.*

Inicial :	$\bullet$	[	1	0	]	0	[11]	[10]	1	[10]	[11]	0	[11]	[11]	1	[10]	0	0	1	0	1	0	$\bullet$

Final :	$\bullet$	[	1	0	]	0	[11]	[10]	1	[10]	[11]	0	[11]	[11]	1	[10]	$<$	1	$>$	:	1	1	$\bullet$

onde [11] *corresponde a* | [| 1 | 1 |] | *e analogamente para os outros estados.*

Cada transição (Est, a, Est') é representada por $[X]a[X']$ em que $[X]$, e $[X']$ representam Est e Est' respectivamente, com $a \in \{0, 1\}$. As transições de um mesmo estado estão em posições consecutivas. Se o autómato encravar a máquina deve escrever na fita $< 0 >$. Senão escreve $< 0 >$:X ou $< 1 >$:X conforme a sequência é ou não aceite, sendo X o número do estado do autómato depois de ler a palavra. A palavra encontra-se na fita imediatamente após a descrição do autómato e deve ser também nessa posição que o resultado é escrito depois de apagar a palavra.

Notar que na figura cada caracter ocupa uma cela distinta na fita, tendo-se marcado apenas as duas primeiras celas.

$\diamond$

1.3 Variações sobre as Máquinas de Turing

Várias modificações da máquina de Turing básica são possíveis sem que o modelo de computação obtido seja nem mais nem menos poderoso.

Entre elas considerem-se as seguintes restrições/extensões:

1. Alfabeto.

(a) Codificação de um alfabeto noutro com pelo menos dois símbolos. Dê exemplos e conclua que o alfabeto não reduz a potência. Em particular podemos considerar máquinas de Turing em que o alfabeto tem apenas dois símbolos, para além do branco.

2. Aridade das funções

   (a) Codificação de pares de inteiros em inteiros. Considere a função bijectiva p de $\mathbf{N} \times \mathbf{N}$ em $\mathbf{N}$
$$p(x, y) = \frac{(x+y)(x+y+1)}{2} + x$$

   (b) Codificações bijectivas de $\mathbf{N}^n$ em $\mathbf{N}$ usando a função p definida atrás.

Conclusão: As funções de aridade maior que 1 podem ser codificadas em funções de aridade 1.

3. Movimentos da cabeça

   (a) Máquinas com movimentos $\{\leftarrow, \rightarrow\}$ (esquerda/direita)

   (b) Máquinas com movimentos $\{\leftarrow, -, \rightarrow\}$ (esquerda/não mexe/direita)

4. Tamanho da fita

   (a) Máquinas com uma fita duplamente infinita

   (b) Máquinas com uma fita semi-infinita

5. Número e tipo de fitas

   (a) Máquinas com 1 fita RW

   (b) Máquinas com k fitas RW

   (c) Máquinas com uma fita de escrita de saída (W) e uma fita RW de trabalho.

   (d) Máquinas com uma fita de leitura (R), com os dados e k fitas RW de trabalho (eventualmente uma fita (W) de saída) (*MT off-line*).

Vamos apenas analisar a equivalência dos seguintes modelos: 3a e 3b;4b e4a; 5a e 5b.

Máquinas com movimentos $\{\leftarrow, -, \rightarrow\}$ Exceptuando a terceira componente da definição de δ estas máquinas são definidas como as MT básicas.

Se M_1 é uma MT com movimentos $\{\leftarrow, \rightarrow\}$ também é uma MT com movimentos $\{\leftarrow, -, \rightarrow\}$. Inversamente, suponhamos que $M_2 = (S_2, \Sigma_2, \Gamma_2, \delta_2, s_2, \bullet, F_2)$ é uma máquina com movimentos $\{\leftarrow, -, \rightarrow\}$. Vamos construir uma MT M_1, apenas com movimentos $\{\leftarrow, \rightarrow\}$, tal que se L é aceite por M_2 então, L é aceite por M_1. Quando a cabeça de M_2 fica parada, M_1 deve entrar num estado especial e movimentar-se uma célula para direita e outra para a esquerda, voltando à mesma célula de tal modo que simule o facto de M_2 não se mexer. Para tal, os estados de M_1 vão ser da forma $[s, X]$ ou $[s, Y]$ para cada estado $s \in S_2$, sendo os estados $[s, X]$ usados sempre que a cabeça de M_2 se movimente e os estados $[s, Y]$ usados quando a cabeça de M_2 não se mova.

Formalmente, tem-se $M_1 = (S_1, \Sigma_1, \Gamma_1, \delta_1, [s_2, X], \bullet, F_1)$ onde

$S_1 = \{[s, X], [s, Y] \mid s \in S_2\}$, $F_1 = F_2 \times \{X, Y\}$

$\Sigma_1 = \Sigma_2$, $\Gamma_1 = \Gamma_2$

δ_1 define-se por, $\forall a \in \Gamma_1,\ s \in S_2$,:

1. $\delta_1([s, X], a) = ([s', X], a', \rightarrow)$, se $\delta_2(s, a) = (s', a', \rightarrow)$
2. $\delta_1([s, X], a) = ([s', X], a', \leftarrow)$ se $\delta_2(s, a) = (s', a', \leftarrow)$
3. $\delta_1([s, X], a) = ([s', Y], a', \rightarrow)$ se $\delta_2(s, a) = (s', a', -)$
4. $\delta_1([s, Y], a) = ([s, X], a, \leftarrow)$

Máquinas com uma fita semi-infinita Uma MT com uma fita semi-infinita é definida como a máquina de Turing básica com a excepção de que existe uma célula mais à esquerda e portanto δ está limitada nos seus movimentos para a esquerda. No início supomos que as n células mais à esquerda contém a sequência de entrada ("dados") e as restantes o caracter de fita "branco".

a_1	a_2	...	a_i	...	a_n	$\bullet$	...

$\uparrow$

$\boxed{\text{controlo finito}}$

Seja $X_1 \cdots X_{i-1} s X_i \cdots X_n$ uma configuração. Se $\delta(s, X_i) = (s', Y, D)$ e $i = 1$ então $D = \rightarrow$ obrigatoriamente.

A noção de configuração e da relação $\overset{M}{\vdash}$ são definidas analogamente, supondo-se a não existência de "brancos" desnecessários.

Exercício 1.11

Formaliza as noções associadas a uma MT com uma fita semi-infinita. $\diamond$

Teorema 1.12 *L é uma linguagem aceite por uma MT com uma fita duplamente infinita se e só se é aceite por uma MT com uma fita semi-infinita.*

Dem. ($\Leftarrow$) Uma MT M_2 com uma fita duplamente infinita pode simular facilmente uma MT com uma fita semi-infinita, M_1. Basta que M_2 marque a célula à esquerda da cabeça no início e depois simule a MT M_1.

($\Rightarrow$) Seja $M_2 = (S_2, \Sigma_2, \Gamma_2, \delta_2, s_2, \bullet, F_2)$ uma MT com fita duplamente infinita. Vamos construir M_1 com uma fita semi-infinita para a direita. Podemos imaginar que a fita de M_1 é dividida ao meio em duas pistas, de modo a que cada célula pode conter dois símbolos. Uma pista S correspondente às células da fita de M_2 à direita da célula onde inicialmente se encontra a cabeça e a outra pista I correspondente às células à esquerda dessa célula. Por exemplo:

Fita de M_2:

...	a_{-2}	a_{-1}	a_0	a_1	a_2	...

Fita de M_1:

a_0	a_1	a_2	...
$\triangleright$	a_{-1}	a_{-2},	...

onde a_0 é a célula que inicialmente está debaixo da cabeça de M_2. A primeira célula da fita de M_1 contém o símbolo $\triangleright$, na "pista" inferior, para indicar que é a célula mais à esquerda. O controlo finito de M_1 indicará se a cabeça de M_2 está debaixo duma célula correspondente à parte superior ou inferior da fita de M_1. Enquanto a cabeça de M_2 estiver à direita da célula inicial, a cabeça de M_1 "trabalhará" na parte superior da sua fita e movimentar-se-á como a de M_2. Se a cabeça de M_2 estiver à esquerda da célula inicial, a cabeça de M_1 actuará na parte infeior da fita e movimentar-se-á em sentido contrário ao da cabeça de M_2. Note-se que a noção de pistas é apenas conceptual, não havendo nenhuma alteração na definição de MT básica.

Formalmente seja $M_1 = (S_1, \Sigma_1, \Gamma_1, \delta_1, s_1, [\bullet, \bullet], F_1)$ onde,

$$S_1 = \{[s, S], [s, I] \mid s \in S_2\} \cup \{s_1\}$$
$$\Sigma_1 = \{[a, \bullet] \mid a \in \Sigma_2\}$$
$$\Gamma_1 = \{[X, Y] \mid X \in \Gamma_2, Y \in \Gamma_2 \cup \{\triangleright\}\}$$
$$F_1 = \{[s, S], [s, I] \mid s \in F_2\}$$

A função $\delta_1 : S_1 \times \Gamma_1 \longrightarrow_p S_1 \times \Gamma_1 \times \{\leftarrow, \rightarrow\}$ é definida por:

1. Para cada $a \in \Sigma_1 \cup \{\bullet\}$,
$$\delta_1(s_1, [a, \bullet]) = ([s, S], [X, \rhd], \rightarrow) \qquad \text{se } \delta_2(s_2, a) = (s, X, \rightarrow)$$

2. Para cada $a \in \Sigma_1 \cup \{\bullet\}$,
$$\delta_1(s_1, [a, \bullet]) = ([s, I], [X, \rhd], \rightarrow) \qquad \text{se } \delta_2(s_2, a) = (s, X, \leftarrow)$$

3. Para cada $[X, Y] \in \Gamma_1$, com $Y \neq \rhd$ e $m =\rightarrow$ ou $m =\leftarrow$,
$$\delta_1([s, S], [X, Y]) = ([s', S], [Z, Y], m) \qquad \text{se } \delta_2(s, X) = (s', Z, m)$$

4. Para cada $[X, Y] \in \Gamma_1$, com $Y \neq \rhd$ e se $m =\rightarrow$ então $\bar{m} =\leftarrow$ e se $m =\leftarrow$ então $\bar{m} =\rightarrow$,
$$\delta_1([s, I], [X, Y]) = ([s', I], [X, Z], m) \qquad \text{se } \delta_2(s, Y) = (s', Z, \bar{m})$$

5.
$$\begin{aligned} \delta_1([s, S], [X, \rhd]) = \delta_1([s, I], [X, \rhd]) &= ([s', P], [Y, \rhd], \rightarrow) \\ & \text{se } \delta_2(s, X) = (s', Y, m) \end{aligned}$$

   onde $P = S$ se $m =\rightarrow$ e $P = I$ se $m =\leftarrow$

$\square$

Máquinas com k fitas RW

Uma máquina de Turing com k fitas, $k > 1$, consiste num controlo finito com k cabeças, uma para cada uma das k fitas. Vamos considerar que cada fita é duplamente infinita. Num movimento, dependendo do estado do controlo finito e dos símbolos lidos em cada uma das k fitas pelas k cabeças, a máquina

1. muda de estado
2. escreve um símbolo em cada uma das células que estão debaixo de cada cabeça
3. move cada cabeça, independentemente, para a esquerda ou para a direita ou não mexe

Inicialmente a sequência de entrada encontra-se na primeira fita e todas as outras estão em branco. Os restantes conceitos são análogos aos duma MT com uma só fita.

Exercício 1.13 *Descreve formalmente uma MT com k fitas e as noções de configuração, relação $\overset{M}{\vdash}$ e de linguagem aceite por uma MT de k fitas.* $\diamond$

Teorema 1.14 *Se uma linguagem L é aceite por uma MT com k fitas, então L é aceite por uma MT com uma única fita.*

Dem. Suponhamos que L é aceite por uma MT com k fitas M_1. Podemos construir uma MT, M_2 com uma só fita e com $2k$ pistas, duas pistas para cada fita. Uma pista contém o conteúdo da fita correspondente de M_1 e a outra está em branco excepto para a célula actualmente lida pela cabeça correspondente de M_1. Esta célula contém um indicador, que supomos ser o símbolo X. Por exemplo se $k = 2$, a fita de M_2 podia ser

cabeça 1	$\bullet$	X	$\dots$	$\bullet$	$\dots$	$\bullet$	
fita 1	a_1	a_2	$\dots$	a_i	$\dots$	a_m	
cabeça 2	$\bullet$	$\bullet$	$\dots$	X	$\dots$	$\bullet$	
fita 2	b_1	b_2	$\dots$	b_i	$\dots$	b_m	

se em M_1 a cabeça da fita 1 estivesse no símbolo a_2 e e a da fita 2 no símbolo b_i.

Formalmente, ter-se-á $\Sigma_2 \subseteq (\{X, \bullet\} \times \Sigma_1)^k$.

O controlo finito de M_2 guardará para além do estado de M_1 o número de indicadores (símbolos X) à direita da cabeça de M_2 (n_i). Cada movimento de M_1 é simulado percorrendo a fita de M_2 da esquerda para a direita, para saber quais os símbolos actualmente lidos pelas k cabeças de M_1 e depois da direita para a esquerda, para actualizar esses valores e movimentar convenientemente os "indicadores". Inicialmente supõe-se que a cabeça de M_2 está na célula mais à esquerda que contém um indicador. Então, a cabeça de M_2 percorre a fita para a direita, e para cada célula com indicador "memoriza" o símbolo lido e diminui n_i. Esta memorização pode ser feita considerando que os estados de M_2 guradam também estes símbolos, i.e cada estado de M_2 corresponde a uma combinação de um estado de M_1 k-símbolos de Γ_1 e o número de indicadores. Quando $n_i = 0$ então M_2 tem informação suficiente para determinar o movimento de M_1. Então, a cabeça de M_2 percorre a fita da esquerda para a direita, e sempre que encontra um indicador modifica o símbolo lido e movimenta o indicador de acordo com o movimento da cabeça correspondente em M_1. Quando já actualizou todos os indicadores, M_2 muda o estado correspondente a M_1. Se o novo estado de M_1 é um estado final, o de M_2 também o é. $\square$

Exercício 1.15 *Para $k = 2$ formaliza a máquina M_2 construída na demonstração anterior.* $\diamond$

Exercício 1.16 *Estima o número de movimentos efectuados por M_2 para cada movimento de M_1. Analisa a eficiência do uso de MTs com k fitas, $k > 1$, em relação a MTs só com uma fita.* $\diamond$

Exercício 1.17 *Descreve uma MT com 2 fitas que aceite a seguinte linguagem:*

$$L = \{wcw^r \mid w \in \{0,1\}^\star\}$$

e compare-a com a descrita no exemplo 1.3. Sugestão: Copiar os dados para a segunda fita e comparar os símbolos, em cada fita movimentando as cabeças em direções opostas. $\diamond$

1.4 Máquinas de Turing não determinísticas

Uma máquina de Turing é *não determinística* se dado um estado e um símbolo lido, existirem várias hipóteses para o movimento seguinte, isto é,

$$\delta \subseteq (S \times \Gamma) \times (S \times \Gamma \times \{\leftarrow, \rightarrow\})$$

Uma MT não determinística aceita a sequência de entrada se existe uma sequência de escolhas de movimentos que conduza a um estado de aceitação (analogamente se definem os conceitos de função calculada ou linguagem reconhecida). Em contraste, as MT definidas anteriormente denominam-se *determinísticas.*

Contudo a adição de não determinismo não torna estas MTs um modelo de computação mais poderoso que as MT determinísticas.

Teorema 1.18 *Se uma linguagem L é aceite por uma máquina de Turing não determinística, M_1, então L é aceite por alguma máquina de Turing determinística, M_2.*

Dem. Para cada estado e símbolo de fita de M_1, existe um número finito de escolhas para o movimento seguinte. Estas escolhas podem ser numeradas 1, 2, ... Seja r o número máximo de escolhas de um par estado-símbolo. Então qualquer sequência finita de escolhas pode ser representada por uma sequência de dígitos de 1 a r.

Vamos construir M_2 com 3 fitas. A primeira contém a sequência de entrada. A segunda, gera sequências de dígitos de 1 a r dum modo sistemático. Por exemplo, por ordem crescente de comprimento e por ordem lexicográfica se de igual comprimento. Para cada sequência gerada na segunda fita, M_2 copia a sequência de entrada para a terceira fita e simula M_1 na terceira fita, usando a sequência da segunda fita para decidir os movimentos de M_1. Se M_1 atinge um estado de aceitação, então M_2 aceita. Se nenhuma sequência de escolhas conduz a M_1 aceitar, M_2 também não aceita. □

Exercício 1.19 *Formaliza a geração de sequências de dígitos entre* 1 *e* r*, usada na demonstração do teorema anterior. Mostra como é que esta sequência pode codificar os movimentos de* M_1*.* ◇

Bibliografia [HMU00, Cap. 8-9], [Pap94, Cap. 2],[Dew93, Cap. 28]

Referências

[Dew93] A. K. Dewdney. *The (New) Turing Omnibus – 66 Excursions in Computer Science.* W. H. Freeman and Company, 1993.

[HMU00] John E. Hopcroft, Rajeev Motwani, and Jeffrey D. Ullman. *Introduction to Automata Theory, Languages and Computation.* Addison Wesley, 2nd edition, 2000.

[Pap94] Christos H. Papadimitriou. *Computational Complexity.* Addison Wesley, 1994.