Teoria da Computação

Aula Teórica 23

Máquinas de Turing

António Ravara

Departamento de Informática, Faculdade de Ciências e Tecnologia,
Universidade Nova de Lisboa

3 de Junho de 2019

Uma Máquina que constroi uma árvore binária

A máquina seguinte coloca o valor $(((2,3),5),7)$ na célula de memória M_1.

start	`push 7`	*s1*
s1	`push 5`	*s2*
s2	`push 3`	*s3*
s3	`push 2`	*s4*
s4	`cons`	*s5*
s5	`cons`	*s6*
s6	`cons`	*s7*
s7	`store 1`	*end*

Uma Máquina que inverte uma sequência de símbolos

- Dada por exemplo a sequência $[x, y, z]$, quer-se obter $[z, y, x]$.
- Considera-se que a sequência dada está na posição M_2 de memória, e que o resultado vai ser colocado na posição M_1.
- Seja *start* o estado inicial e *end* o estado final.
- Assumindo que no início a posição de memória M_1 contém `null`, o programa consiste num ciclo que começa no estado *start*, remove o primeiro elemento da lista em M_2, e adiciona-o na cabeça da lista em M_1.
  O ciclo termina quando a lista em M_2 está vazia (contém `null`).

Uma Máquina que inverte uma sequência de símbolos

A relação de transição (como um "programa *Assembler*")

- Inicialmente, $M[1] = \mathit{null}$ e $M[2] = (X, (Y, \mathtt{null}))$.
- A pilha começa vazia (`null`) e o estado inicial é *start*.
- Obtém-se então

$$(\langle \mathtt{null} \rangle \langle [X, Y] \rangle, \mathtt{null}, \mathit{start}) \longrightarrow^* (\langle [Y, X] \rangle \langle \mathtt{null} \rangle, [\mathtt{null}], \mathit{end})$$

start	`load 2`	*ch*
ch	`?null`	*end*
ch	`?cons`	*s1*
s1	`store 2`	*s2*
s2	`load 1`	*s3*
s3	`load 2`	*s4*
s4	`left`	*s5*
s5	`cons`	*s6*
s6	`store 1`	*s7*
s7	`load 2`	*s8*
s8	`right`	*s9*
s9	`store 2`	*start*

A Máquina a inverter uma sequência de símbolos

$$(\langle \texttt{null} \rangle \langle [X, Y] \rangle, \texttt{null}, \mathit{start}) \xrightarrow{\texttt{load 2}}$$
$$(\langle \texttt{null} \rangle \langle [X, Y] \rangle, [[X, Y]], \mathit{ch}) \xrightarrow{\texttt{?cons}}$$
$$(\langle \texttt{null} \rangle \langle [X, Y] \rangle, [[X, Y]], \mathit{s1}) \xrightarrow{\texttt{store 2}}$$
$$(\langle \texttt{null} \rangle \langle [X, Y] \rangle, \texttt{null}, \mathit{s2}) \xrightarrow{\texttt{load 1}}$$
$$(\langle \texttt{null} \rangle \langle [X, Y] \rangle, [\texttt{null}], \mathit{s3}) \xrightarrow{\texttt{load 2}}$$
$$(\langle \texttt{null} \rangle \langle [X, Y] \rangle, [[X, Y], \texttt{null}], \mathit{s4}) \xrightarrow{\texttt{left}}$$
$$(\langle \texttt{null} \rangle \langle [X, Y] \rangle, [X, \texttt{null}], \mathit{s5}) \xrightarrow{\texttt{cons}}$$
$$(\langle \texttt{null} \rangle \langle [X, Y] \rangle, [[X]], \mathit{s6}) \xrightarrow{\texttt{store 1}}$$
$$(\langle [X] \rangle \langle [X, Y] \rangle, \texttt{null}, \mathit{s7}) \xrightarrow{\texttt{load 2}}$$
$$(\langle [X] \rangle \langle [X, Y] \rangle, [[X, Y]], \mathit{s8}) \xrightarrow{\texttt{right}}$$
$$(\langle [X] \rangle \langle [X, Y] \rangle, [[Y]], \mathit{s8}) \xrightarrow{\texttt{store 2}}$$
$$(\langle [X] \rangle \langle [Y] \rangle, \texttt{null}, \mathit{start}) \xrightarrow{\texttt{load 2}}$$
$$(\langle [X] \rangle \langle [Y] \rangle, [[Y]], \mathit{ch}) \xrightarrow{\texttt{?cons}}$$
$$(\langle [X] \rangle \langle [Y] \rangle, [[Y]], \mathit{s1}) \xrightarrow{\texttt{store 2}}$$
$$(\langle [X] \rangle \langle [Y] \rangle, \texttt{null}, \mathit{s2}) \xrightarrow{\texttt{load 1}}$$
$$(\langle [X] \rangle \langle [Y] \rangle, [[X]], \mathit{s3}) \xrightarrow{\texttt{load 2}}$$
$$(\langle [X] \rangle \langle [Y] \rangle, [[Y], [X]], \mathit{s4}) \xrightarrow{\texttt{left}}$$
$$(\langle [X] \rangle \langle [Y] \rangle, [Y, [X]], \mathit{s5}) \xrightarrow{\texttt{cons}}$$
$$(\langle [X] \rangle \langle [Y] \rangle, [[Y, X]], \mathit{s6}) \xrightarrow{\texttt{store 1}}$$
$$(\langle [Y, X] \rangle \langle [Y] \rangle, \texttt{null}, \mathit{s7}) \xrightarrow{\texttt{load 2}}$$
$$(\langle [Y, X] \rangle \langle [Y] \rangle, [[Y]], \mathit{s8}) \xrightarrow{\texttt{right}}$$
$$(\langle [Y, X] \rangle \langle [Y] \rangle, [\texttt{null}], \mathit{s9}) \xrightarrow{\texttt{store 2}}$$
$$(\langle [Y, X] \rangle \langle \texttt{null} \rangle, \texttt{null}, \mathit{start}) \xrightarrow{\texttt{load 2}}$$
$$(\langle [Y, X] \rangle \langle \texttt{null} \rangle, [\texttt{null}], \mathit{start}) \xrightarrow{\texttt{?null}}$$
$$(\langle [Y, X] \rangle \langle \texttt{null} \rangle, [\texttt{null}], \mathit{end})$$

Linguagem de uma *Stack-Based Turing Machine*

Suponha-se que se faz inicialmente o carregamento da palavra para a pilha de dada SBTM (que estava vazia).

Definição

Uma dada SBTM aceita a linguagem L sobre o alfabeto Σ se, sempre que inicia a sua computação com uma palavra $w \in \Sigma^*$ no topo da pilha, a sua execução termina:

- com `true` no topo da pilha, se $w \in L$; e
- com `false` se $w \notin L$.

Exemplo

Uma SBTM que aceita a linguagem definida pela expressão regular $(\texttt{open}\,\texttt{close})^*\,\texttt{end}$ sobre o alfabeto $\Sigma = \{\texttt{open}, \texttt{close}, \texttt{end}\}$.

s0	?null	*s3*
s0	?end	*s1*
s0	?open	*s2*
s0	?close	*s3*
s1	store 1	*s4*
s4	?null	*s5*
s4	?end	*s3*
s4	?open	*s3*
s4	?close	*s3*
s3	push false	*end*
s5	push true	*end*
s2	store 1	*s6*
s6	?null	*s3*
s6	?end	*s3*
s6	?open	*s3*
s6	?close	*s7*
s7	store 1	*s0*