Instituto Superior Técnico
Lic. Engenharia Informática e de Computadores (Alameda)

Teoria da Computação

27 de Maio de 2015 Teste 2A Duração: 1h30

Grupo I (3+1+3 valores)

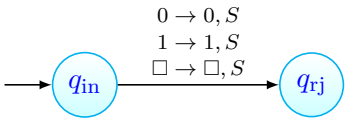
Considere as linguagens $P_A = \{M : M \text{ é máquina tal que } L_{\text{ac}}(M) \text{ é finita}\}$ e $L_A = \{M_1\$M_2 : M_1 \text{ e } M_2 \text{ são máquinas tais que } L_{\text{ac}}(M_1) \cap L_{\text{ac}}(M_2) \text{ é finita}\}$.

a) Use o teorema de Rice para demonstrar a indecidibilidade de P_A.

Resolução: Segundo o teorema de Rice, se $L \subseteq \mathcal{M}$, então L é indecidível desde que satisfaça três requisitos: (1) $L \neq \emptyset$, (2) $L \neq \mathcal{M}$, e (3) se $M_1 \in L$ e $L_{\text{ac}}(M_1) = L_{\text{ac}}(M_2)$ então $M_2 \in L$.

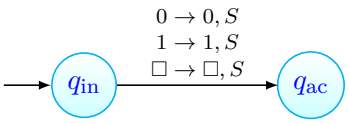
Dado que $P_A \subseteq \mathcal{M}$, basta verificar que a linguagem P_A satisfaz os três requisitos.

(1) Considere-se a máquina $M_\emptyset$, com alfabeto $\Gamma = \{0, 1, \square\}$, representada graficamente por:



A máquina $M_\emptyset$ rejeita todos os *inputs* pelo que $L_{\text{ac}}(M_\emptyset) = \emptyset$, que é um conjunto finito. Logo, temos $M_\emptyset \in P_A$ e portanto $P_A \neq \emptyset$.

(2) Considere-se a máquina A, com alfabeto $\Gamma = \{0, 1, \square\}$, representada graficamente por:



A máquina A aceita todos os *inputs* pelo que $L_{\text{ac}}(A) = \{0, 1\}^*$, que é um conjunto infinito. Logo, temos $A \notin P_A$ e portanto $P_A \neq \mathcal{M}$.

(3) se $M_1 \in P_A$ e $L_{\text{ac}}(M_1) = L_{\text{ac}}(M_2)$, então sabemos que $L_{\text{ac}}(M_1)$ é finita, e portanto também $L_{\text{ac}}(M_2)$ é finita, o que permite concluir que $M_2 \in P_A$.

Pelo teorema de Rice concluimos que P_A é indecidível.

b) É possível usar o teorema de Rice para demonstrar que L_A é indecidível?

Resolução: Não é possível usar (directamente) o teorema de Rice para demonstrar a indecidibilidade de L_A, pois $L_A \notin \mathcal{M}$, ou seja, cada palavra da linguagem L_A não é a representação de uma máquina de Turing (é, isso sim, um par de representações de máquinas de Turing).

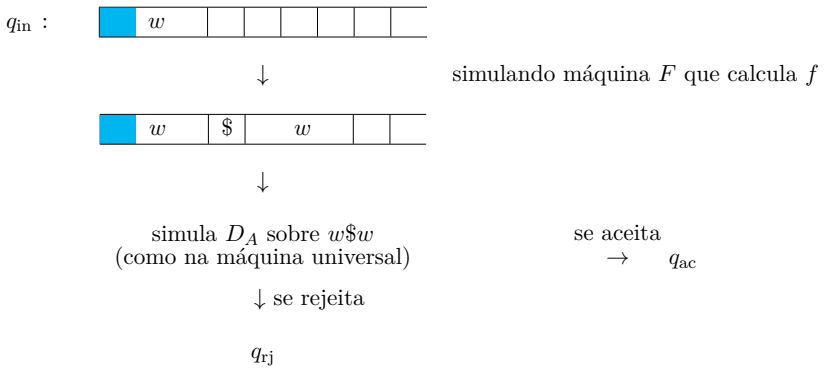
c) Mostre que $P_A \leq L_A$ e conclua, justificadamente, que L_A é indecidível.

Resolução: Considere-se a função $f : \{0,1\}^* \to \{0,1,\$\}^*$ definida por $f(w) = w\$w$. A função é total, por definição, e computável (basta copiar o *input* após inserir \$). Para mostrar que $P_A \leq L_A$ basta verificar que $w \in P_A$ se e só se $f(w) \in L_A$.

Se $w \in P_A$ então $w = M$ é a codificação de uma máquina de Turing tal que $L_{\text{ac}}(M)$ é finita. Logo, $f(w) = M\$M$ e $L_{\text{ac}}(M) \cap L_{\text{ac}}(M) = L_{\text{ac}}(M)$ é finita, pelo que $M\$M \in L_A$.

Reciprocamente, se $f(w) = w\$w \in L_A$ então $w = M$ é a codificação de uma máquina de Turing tal que $L_{\text{ac}}(M) \cap L_{\text{ac}}(M)$ é finita. Logo, $L_{\text{ac}}(M) = L_{\text{ac}}(M) \cap L_{\text{ac}}(M)$ é finita e portanto $M \in P_A$.

Assuma-se agora, por absurdo, que L_A fosse decidível. Nesse caso, existiria um classificador D_A tal que $L_{\text{ac}}(D_A) = L_A$. Então, poder-se-ia construir a máquina T, descrita por:



Facilmente, T é um classificador, pois a função f é total e D_A um classificador. Além disso, T aceita w se e só se D_A aceita $w\$w$ se e só se $w\$w = f(w) \in L_A$ se e só se $w \in P_A$. Mas então T decidiria a linguagem P_A, em contradição com a sua indecidibilidade, obtida na alínea **a)**.

Grupo II (5+3 valores)

Considere a linguagem S_A formada por todas as palavras da forma

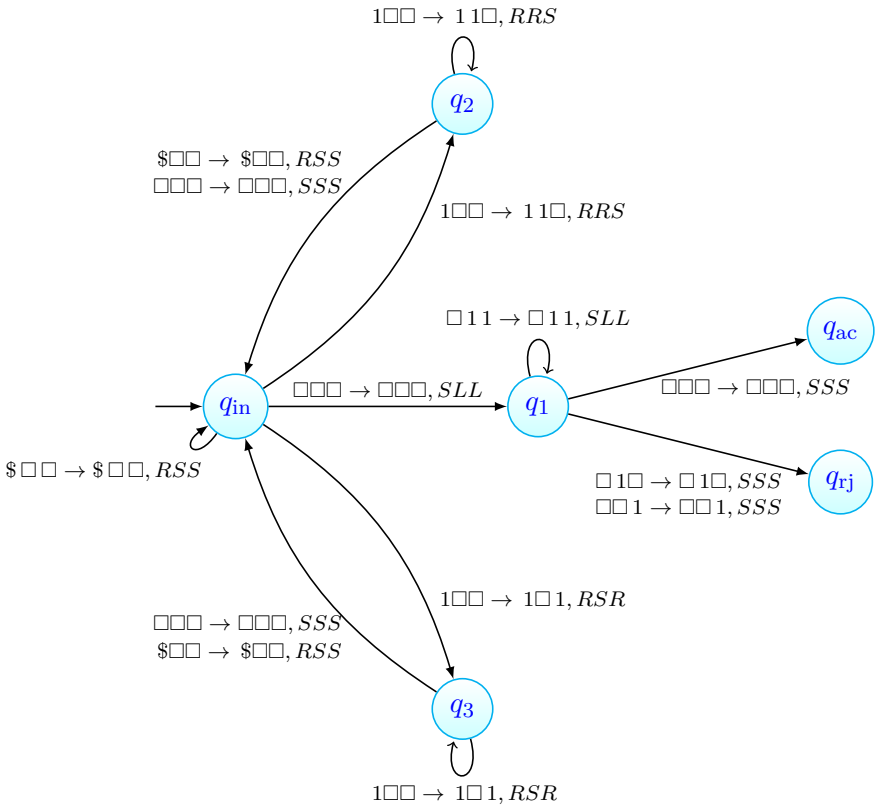
$$w_1\$w_2\$\ldots\$w_k, \text{ onde } k \in \mathbb{N} \text{ e } w_1, \ldots, w_k \in \{1\}^*$$

para as quais existe $I \subseteq \{1, \ldots, k\}$ tal que $\sum_{i \in I} w_i = \sum_{i \notin I} w_i$, sendo as somas tomadas sobre a representação em unário de naturais.

Por exemplo, $111\$1\$11111\$1\$1111 \in S_A$ (com $I = \{1, 5\}$) pois verifica-se $w_1 + w_5 = w_2 + w_3 + w_4$, isto é $3 + 4 = 1 + 5 + 1$.
Por outro lado, $111\$1111\$11 \notin S_A$.

a) Demonstre que $S_A \in$ **NP** (pode usar máquinas de Turing multifita e os resultados conhecidos sobre a sua complexidade).

Resolução: Considere-se a máquina de Turing não-determinística N_A com alfabeto $\Gamma = \{1, \$, \square\}$ e três fitas bidireccionais, representada graficamente por:



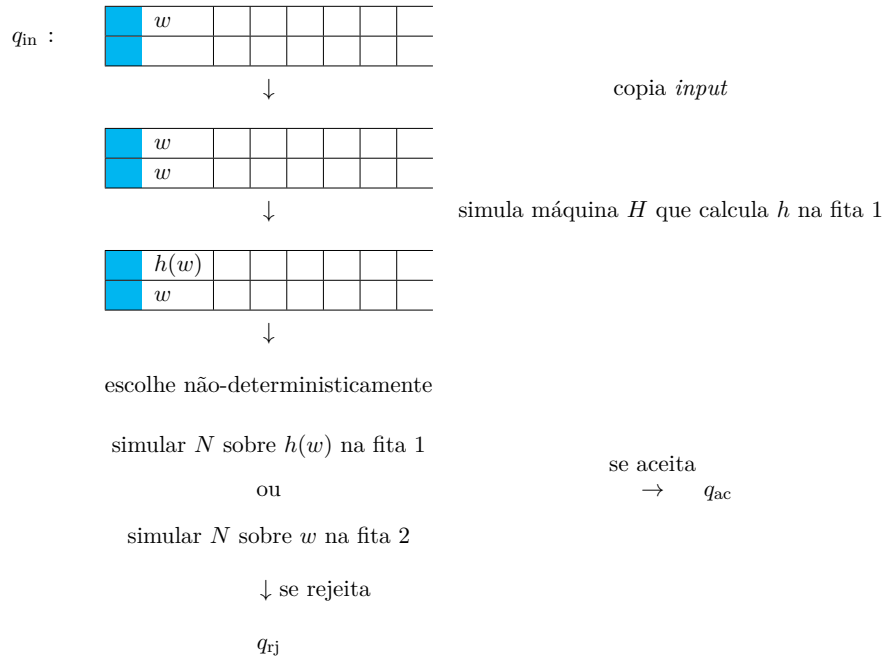
A máquina N_A copia, não-deterministicamente, cada um dos w_i do *input*, para a fita 2 ou a fita 3, onde vão sendo adicionados (o conjunto de índices I da definição corresponde precisamente aos índices i cujo w_i a máquina

escolhe copiar para a fita 2). Após processar todo o input, a máquina compara os valores das fitas 2 e 3, aceitando se forem iguais e rejeitando no caso contrário.

Analisando o comportamento de N_A, têm-se $n+1$ passos para percorrer o *input* até ao final, mais um passo para transitar para q_1 ao iniciar a comparação das fitas 2 e 3, e no pior caso, em que de facto os valores são iguais, um máximo de $\frac{n}{2}+1$ passos de comparação, pelo que $ntime_{N_A}(n) \leq n+1+1+\frac{n}{2}+1 = \mathcal{O}(n)$. Sabendo que a mesma computação poderia ser realizada numa única fita com uma desaceleração quadrática, podemos concluir que $S_A \in \mathbf{NTIME}(n^2) \subseteq \mathbf{NP}$.

b) Use a alínea **a)** para demonstrar que se $L \leq_p S_A$ então $L \cup S_A \in \mathbf{NP}$.

Resolução: Da alínea **a)** temos uma máquina não determinística N, de tempo polinomial, que decide S_A. Se, adicionalmente, sabemos que $L \leq_p S_A$ então existe uma função total $h : \Sigma^* \to \{1, \$\}^*$ computável em tempo polinomial (onde Σ é o alfabeto subjacente à linguagem L) tal que $w \in L$ se e só se $h(w) \in S_A$ para cada $w \in \Sigma^*$. Considere-se então a máquina não-determinística N', com 2 fitas, descrita por:



As computações da máquina terminam sempre, já que h é total e N decide S_A.

Facilmente, se a máquina aceita o *input* w é porque N aceita $h(w)$ na fita 1 ou w na fita 2. No primeiro caso, sabemos que $h(w) \in S_A$ e portanto $w \in$

L. No segundo caso, sabemos que $w \in S_A$. Em qualquer das situações, $w \in L \cup S_A$.

Por outro lado, se $w \in L \cup S_A$ há duas possibilidades. Se $w \in L$ então $h(w) \in S_A$ e escolhendo simular N sobre $h(w)$ na fita 1 há um caminho de computação que leva ao estado q_{ac}. Se $w \in S_A$, escolhendo simular N sobre w na fita 2 há de novo um caminho de computação que leva ao estado q_{ac}.

Conclui-se que $L_{\text{ac}}(N') = L \cup S_A$. Resta verificar que a máquina é de tempo polinomial. Tem-se, de facto, que $ntime_{N'}(n) = \mathcal{O}(n)+time_H(n)+\max(ntime_N(n), ntime_N(|h(n)|))$, correspondendo ao tempo necessário para copiar o *input* w, mais o tempo necessário para calcular $h(w)$, mais o máximo entre o tempo necessário para simular N na primeira fita ou na segunda.

Sejam p, q polinómios tais que $ntime_N(n) = \mathcal{O}(p(n))$ e $time_H(n) = \mathcal{O}(q(n))$. Desenvolvendo a expressão obtida, tem-se

$$\begin{aligned} ntime_{N'}(n) &\leq \mathcal{O}(n) + \mathcal{O}(q(n)) + \max(\mathcal{O}(p(n)), \mathcal{O}(p(space_H(n)))) \\ &\leq \mathcal{O}(n + q(n) + \max(p(n), p(time_H(n)))) \\ &= \mathcal{O}(n + q(n) + \max(p(n), p(q(n)))), \end{aligned}$$

pelo que $L \cup S_A \in \mathbf{NP}$.

Grupo III (1+1+3 valores)

a) Defina as classes de complexidade **PSPACE** e **EXPSPACE**.

Resolução: Tem-se que $\mathbf{PSPACE} = \bigcup_{c\in\mathbb{N}} \mathbf{SPACE}(n^c)$ e $\mathbf{EXPSPACE} = \bigcup_{c\in\mathbb{N}} \mathbf{SPACE}(2^{n^c})$, onde $\mathbf{SPACE}(f(n))$ é o conjunto de todas as linguagens que podem ser decididas por um classificador D tal que $space_D(n) = \mathcal{O}(f(n))$.

b) Enuncie o teorema de hierarquia espacial.

Resolução: Uma função f diz-se *construtível no espaço* se $\log(n) = \mathcal{O}(f(n))$ e o cálculo de $f(n)$ (em binário) a partir de n (em unário) é computável em espaço $\mathcal{O}(f(n))$.

O teorema de hierarquia espacial afirma que, se f é uma função construtível no espaço, então existe uma linguagem $L \in \mathbf{SPACE}(f(n))$ que não pode ser decidida por nenhuma máquina cujo espaço seja $o(f(n))$.

c) Use o teorema de hierarquia espacial para mostrar que

$$\mathbf{PSPACE} \subsetneq \mathbf{EXPSPACE}.$$

Resolução: É fácil verificar que a função $f(n) = 2^n$ é construtível no espaço. Em particular, basta notar que para o *input* 1^n o *output* pretendido é 10^n, e portanto computável em espaço linear.

Pelo teorema de hierarquia espacial, existe $L \in \mathbf{SPACE}(2^n) \subseteq \mathbf{EXPSPACE}$ que não pode ser decidida por nenhuma máquina cujo espaço seja $o(2^n)$. Resta-nos verificar que $n^c = o(2^n)$ para qualquer $c \in \mathbb{N}$.

Na verdade, usando sucessivamente a regra de l'Hôpital, tem-se imediatamente que

$$\lim_{n\to\infty} \frac{n^c}{2^n} = \lim_{n\to\infty} \frac{c.n^{c-1}}{ln(2).2^n} = \lim_{n\to\infty} \frac{c(c-1).n^{c-2}}{ln(2)^2.2^n} = \dots$$

$$= \lim_{n\to\infty} \frac{c!}{ln(2)^c.2^n} = \frac{c!}{ln(2)^c}. \lim_{n\to\infty} \frac{1}{2^n} = 0.$$

Por outro lado, como $n^c = o(2^n)$ então também $n^c = \mathcal{O}(2^n)$, o que garante que $\mathbf{PSPACE} \subseteq \mathbf{EXPSPACE}$. A linguagem L obtida acima pelo teorema de hierarquia espacial garante que a inclusão é estrita, isto é, $\mathbf{PSPACE} \subsetneq \mathbf{EXPSPACE}$.