Ata de exercício
MO417 - Análise de Algoritmos

Daniel Cason

2 de julho de 2010

Questão 34.1-5 Mostre que um algoritmo de complexidade polinomial que faz no máximo um número *constante* de chamadas para uma uma rotina de complexidade polinomial executa em tempo polinomial, mas se a rotina é chamada um número *polinomial* de vezes é possível que o tempo total de execução do algoritmo seja exponencial.

Se o algoritmo tem complexidade polinomial de tempo então o número de instruções realizadas para uma instância (ou entrada) de tamanho n é limitado superiormente por $c \cdot n^p$, sendo c e p constantes positivas.

Sejam $f^{(1)}, f^{(2)}, \ldots$ as várias rotinas adotadas por este algoritmo, todas elas polinomiais. Assim, seja m_i a entrada para a i-ésima rotina e $f^{(i)}(m_i) \leq c_i \cdot m_i^{q_i}$ o limite superior do número de operações realizadas, assim como do volume de dados gerado por ela.

No primeiro caso chama-se um número constante de vezes (digamos k) rotinas polinomiais. Assim, o custo $T(n)$ do algoritmo pode ser dado por:

$$\begin{aligned}
T(n) &= c \cdot n^p + \sum_{i=1}^{k} f^{(i)}(m_i) \\
&\leq c \cdot n^p + \sum_{i=1}^{k} c_i \cdot m_i^{q_i} \\
&\leq c \cdot n^p + \sum_{i=1}^{k} c_i \cdot n^{\prod_{j=1}^{i} q_j} \\
&\leq c \cdot n^p + \sum_{i=1}^{k} c_i \cdot n^{\prod_{j=1}^{k} q_j} \\
&\leq c \cdot n^p + max_i\{c_i\} \cdot k \cdot n^{\prod_{j=1}^{k} q_j} \\
&\leq c^* \cdot n^{p^*}
\end{aligned}$$

ou seja, $T(n)$ é polinomial.

Se é feito um número polinomial de chamadas a uma rotina polinimial é possível que o custo total da função não seja mais polinomial. Isto pode ser demonstrado através de um exemplo.

Seja um algoritmo que recebe como entrada um polinômio de grau n,

$$P_n(x) = a_n x^n + a_{n-1} x^{n-1} + \cdots + a_i x + a_0$$

unicamente determinado pelas n constantes $a_n, a_{n-1}, \ldots, a_1, a_0$. Além do polinômio passa-se a potência ao qual ele será elevado, que é um valor inteiro positivo. Assim, a sua entrada é da ordem $O(n)$:

PolinomialPow($P_n(x)$,n):

1: $P' \leftarrow P$
2: $i \leftarrow 0$
3: **while** $i < n$ **do**
4: $\quad P' \leftarrow P' \cdot P'$
5: $\quad i \leftarrow i + 1$
6: **end while**

A linha 4 realiza o produto de dois polinômios $P_a(x)$ e $P_b(x)$ com graus a e b, através de $a \cdot b$ multiplicações, resultando em um polinômio $P_c(c)$ com grau $c = a + b$. Esta rotina é polinomial em relação ao tamanho da entrada $a + b$, assim como o algoritmo apresentado, que executa o laço das linhas 3 a 6 n vezes.

A cada iteração do laço das linhas 3 a 6 o grau de P' é dado por $2^i \cdot n$. Ao início da primeira iteração $P' = P$, segundo a atribuição da linha 1, e o grau de $P' = n = 2^0 \cdot n$. Ao final de cada iteração, o grau de P' é o dobo do grau de P' ao início da mesma iteração.

O custo do algoritmo é dado por:

$$\begin{aligned} T(n) &= c_1 + c_2 \sum_{i=0}^{n-1} (\text{ grau de } P' \text{ na iteração } i)^2 \\ &= c_1 + c_2 \sum_{i=0}^{n-1} (2^i n)^2 = c_1 + c_2 n^2 \sum_{i=1}^{n} 2^i \\ &\geq c_1 + c_2 n^2 (2^n) \end{aligned}$$

que é $\Omega(2^n)$, ou seja, não polinomial.