Eficiência dos algoritmos baseados no esquema "dividir para conquistar"

Uma recorrência associada ao esquema "dividir para conquistar"

Suponhamos que uma implementação do esquema genérico "dividir para conquistar" tem a seguinte forma

```
     função f(problema de tamanho n):
       if n=1:
         return <solução imediata>
       else:
(1)      divide-se o problema em b partes de tamanho n/b
         fazem-se a chamadas recursivas do tipo
(2)          f(problema de tamanho n/b)
(3)      combinam-se as a soluções
         ... no resultado que é retornado
```

Para simplificar vamos supor que n é potência de b, seja $n = b^m$. Admitindo que a divisão (1) e a combinação (3) são executadas num tempo de ordem $O(n^k)$, temos a seguinte recorrência para a definição de um majorante do tempo de execução

$$t(n) = at(n/b) + cn^k \tag{1}$$

Para simplificar supomos que $t(1) = c$. Para estudar a solução desta recorrência, começemos por considerar o caso $n = b^2$. Temos

$$\begin{aligned} t(n) &= at(n/b) + cn^k && (2)\\ &= a(at(n/b^2) + c(n/b)^k) + cn^k && (3)\\ &= a^2t(1) + acb^k + cb^{2k} && (4)\\ &= a^2c + acb^k + cb^{2k} && (5) \end{aligned}$$

No caso geral $n = b^m$, obtemos

$$t(n) = c(a^m + a^{m-1}b^k + \cdots + a^0b^{mk}) = ca^m \sum_{i=0}^{m} \left(\frac{b^k}{a}\right)^i$$

Note-se que nesta expressão a, b e k são constantes; a variável é m com $n = b^m$, ou seja $m = \log_b n$. Casos a considerar: $a > b^k$, $a = b^k$, $a < b^k$.

Caso 1. $a > b^k$

Trata-se de uma soma geométrica. Sendo $r = b^k/a$ é $r < 1$ e temos

$$t(n) = ca^m \frac{r^{m+1} - 1}{r - 1} < \frac{ca^m}{1 - b^k/a} \in O(a^m)$$

Mas $a^m = a^{\log_b n} = b^{(\log_b a)(\log_b n)} = n^{\log_b a}$.

Caso 2. $a = b^k$

Neste caso os termos da soma são iguais e $t(n) = c(m+1)a^m \in O(ma^m)$. Como $m = \log_b n$, é

$$n^k = b^{(\log_b n)(log_b a)} = a^{\log_b n}$$

donde

$$ma^m = (\log_b n)a^{\log_b n} = (\log_b n)n^k \in O(n^k \log n)$$

Caso 3. $a < b^k$

Procedendo como no primeiro caso e sendo $r = b^k/a$, temos $r > 1$ e

$$t(n) = ca^m \frac{b^{k(m+1)}/a^{m+1} - 1}{b^k/a - 1} \in O(b^{km}) = O(n^k)$$

Em resumo, temos

Teorema 1 *A solução de uma recorrência com a equação geral da forma $t(n) = at(n/b) + cn^k$ onde c e k são inteiros positivos, a e b são inteiros com $a \geq 1$, $b \geq 2$ tem a seguintes ordens de grandeza*

$$\begin{cases} t(n) \in O(n^{\log_b a}) & \text{se } a > b^k \\ t(n) \in O(n^k \log n) & \text{se } a = b^k \\ t(n) \in O(n^k) & \text{se } a < b^k \end{cases}$$

O resultado mais forte que resulta de substituir em cima a ordem de grandeza $O(\cdot)$ por $\Theta(\cdot)$ é também válido.

Utilizaremos este resultado diversas vezes neste curso.

Exercício Considere os algoritmos (a) "mergesort" e (b) "pesquisa binária"; mostre como o resultado anterior pode ser utilizado em cada um desses algoritmos para encontrar a correspondente ordem de grandeza do tempo de execução.