Análise de Algoritmos
Gabarito da Primeira Prova
Tópicos: Fundamentos de análise de algoritmos e algoritmos para ordenação

Instituto de Ciências Exatas, Universidade de Brasília

22 de abril de 2009

Prof. Mauricio Ayala-Rincón

Fundamentos: relações de recurrência

1. (4.0 pontos) Uma instância de uma classe de problemas hipotéticos pode ser resolvida computacionalmente como segue:

   (a) dividindo-a em duas instâncias de subproblemas de tamanho a metade da instância do problema original, as quais são resolvidas recursivamente;

   (b) combinando as duas soluções dessas duas instâncias para assim obter uma solução da instância do problema original.

   Supondo que o custo de dividir problemas em subproblemas é zero e que combinar as soluções de dois subproblemas tem um custo igual à suma dos tamanhos das instâncias dos subproblemas menos um, o trabalho realizado, denotado por U, é expresso pela relação de recurrência:

$$\begin{cases} U(n) = 2U(n/2) + (n-1) \\ U(1) = 0 \end{cases}$$

   A questão é se obteriamos alguma vantagem com uma técnica algorítmica para resolver essa classe de problemas baseada em:

   (a) dividir a entrada em quatro instâncias de subproblemas de tamanho um quarto do tamanho da entrada, as quais são resolvidas recursivamente;

   (b) combinar soluções de pares de instâncias de subproblemas de tamanho um quarto da entrada para assim obter soluções de duais instâncias de subproblemas de tamanho a metade da entrada;

   (c) combinar essas soluções de instâncias do problema de tamanho a metade da entrada, para assim obter uma solução da instância de entrada.

   Se supomos que combinar duas soluções de tamanho m tem custo $2m-1$, o custo total dos passos (b) e (c) de combinação de instâncias de subproblemas para uma entrada de tamanho n é de $(2(\frac{n}{2}-1))+(n-1)$. E nesse caso o trabalho total realizado ao utilizar essa técnica algorítmica, denotado por W, é expresso pela relação de recurrência:

$$\begin{cases} W(n) = 4W(n/4) + (2n-3) \\ W(1) = 0 \end{cases}$$

   $\Longrightarrow$ Concretamente a questão de qual das duas técnicas algorítmicas e melhor deve ser respondida segundo os passos a seguir:

(a) (1.5 pontos) **Calcule explícitamente** $U(n)$. Para poder comparar com a sua resposta do item (b), suponha que $n = 2^{2k} = 4^k$.

(b) (1.5 pontos) **Calcule explícitamente** $W(n)$. Supondo $n = 2^{2k} = 4^k$.

(c) (1.0 pontos) Forneça então, comparando os valores calculados para U e W, uma resposta precisa à questão.

$\Longrightarrow$ Observe que será necessário um cálculo preciso de $U(n)$ e $W(n)$, sempre que o comportamento assintótico das duas relações é exatamente o mesmo: $U(n), W(n) \in \Theta(n\,ln(n))$. E não será possível formular uma resposta adequada á questão baseada unicamente no comportamento assintótico dessas funções.

R/

(a) Espandindo a relação de recurrência, temos:

$$\begin{aligned} U(n) &= 2U(n/2) + (n-1) \\ &= 2^2U(n/2^2) + (n-2) + (n-1) \\ &= 2^3U(n/2^3) + (n-2^2) + (n-2) + (n-1) \\ &\dots \\ &= 2^{2k}U(n/2^{2k}) + (n-2^{2k-1}) + \ldots + (n-2) + (n-1) \\ &= n\,2k - \textstyle\sum_{i=0}^{2k-1} 2^i \\ &= n\,2k - 2^{2k} + 1 \\ &= n\,(2k-1) + 1 \end{aligned}$$

(b) Espandindo a relação de recurrência, temos:

$$\begin{aligned} W(n) &= 4W(n/4) + (2n-3) \\ &= 4^2U(n/4^2) + (2n-3\,4) + (2n-3) \\ &= 4^3U(n/4^3) + (2n-3\,4^2) + (2n-3\,4) + (2n-3) \\ &\dots \\ &= 4^kU(n/4^k) + (2n-3\,4^{k-1}) + \ldots + (2n-3\,4) + (2n-3) \\ &= 2n\,k - 3\textstyle\sum_{i=0}^{k-1} 4^i \\ &= 2n\,k - (4^k - 1) \\ &= n\,(2k-1) + 1 \end{aligned}$$

(c) Ambas as relações de recurrência resultam em $n(log_2 n - 1) + 1$. Dessa forma, pode-se concluir que nenhuma vantagem teriamos com a segunda abordagem.

Algoritmos de ordenação

2. (4.0 pontos) O algoritmo *binary-insertionsort* é um aprimoramento do algorimto *insertionsort* para o qual na $(j-1)$-ésima iteração, se insere a chave que ocupa a j-ésima posição da lista de entrada, L, no segmento inicial, $L[1..j-1]$, previamente ordenado, usando o método de inserção binária em troca do método de inserção seqüencial utilizado por *insertionsort*.

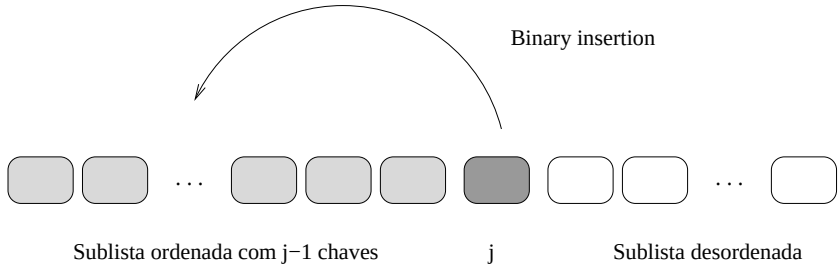

Figure 1: Inserção binária da j-ésima chave

(a) (1.0 ponto) **Explique** porque o número de comparações pior-caso, denotado por W, para ordenar listas de comprimento n com esse algoritmo pode ser expresso pela fórmula:

$$W(n) = \sum_{i=2}^{n} \lceil log_2(i) \rceil = \sum_{j=1}^{\lfloor log_2(n) \rfloor} j\ 2^{j-1} \ + \ (n - 2^{\lfloor log_2(n) \rfloor}) \lceil log_2(n) \rceil$$

(b) (2.0 pontos) **Calcule** $W(n)$ para qualquer $n \in \mathbb{N}$ e

(c) (1.0 pontos) **verifique** que o algoritmo tem complexidade em $\Theta(n\ log_2(n))$.

/R

(a) Quanto à primeira igualdade

$$W(n) = \sum_{i=2}^{n} \lceil log_2(i) \rceil$$

sempre que da segunda até a última chave realizamos uma inserção binária em listas de tamanho o índice da chave menos um. E a inserção binária em listas de tamanho j realiza no pior-caso $\lceil log_2(i) \rceil$ comparações.

Quanto à segunda igualdade

$$\sum_{i=2}^{n} \lceil log_2(i) \rceil = \sum_{j=1}^{\lfloor log_2(n) \rfloor} j\ 2^{j-1} \ + \ (n - 2^{\lfloor log_2(n) \rfloor}) \lceil log_2(n) \rceil$$

esta pode ser interpretada como uma suma de áreas como na figura 2

Os retângulos claros representam o primeiro termo da suma e o retângulo oscuro o segundo, para o caso de $n = 10$.

Para explicar o primeiro termo da suma, observe que para todos os $i \in \mathbb{N}$ tais que $2 \leq i \leq n$ e $2^{j-1} < i \leq 2^j$, para um $1 \leq j \leq \lfloor log_2(n) \rfloor$, temos que

$$\sum_{i=2^{j-1}+1}^{2^j} \lceil log_2(i) \rceil = j 2^{j-1}$$

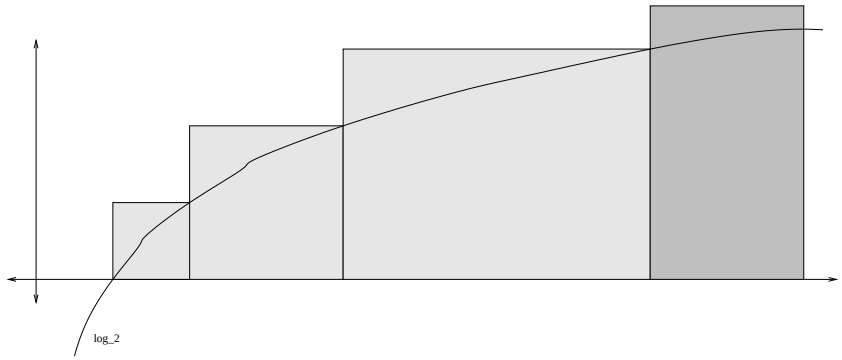

Figure 2: Custo da inserção binária como suma de áreas

Para explicar o segundo termo da suma, observe que para todos os i tais que $2^{\lfloor log_2 n \rfloor} < i \leq n$ vale que $\lceil log_2(i) \rceil = \lceil log_2(n) \rceil$. Assim, o retângulo oscuro tem uma área de base $n - 2^{\lfloor log_2 n \rfloor}$ vezes altura $\lceil log_2(n) \rceil$.

(b) Calcula-se $\sum_{j=1}^{\lfloor log_2(n) \rfloor} j\ 2^{j-1}$. Por brevidade, seja $l = \lfloor log_2(n) \rfloor$.

$$\begin{aligned}
\sum_{j=1}^{l} j\ 2^{j-1} &= \tfrac{1}{2} \sum_{j=1}^{l} j 2^j \\
&= \tfrac{1}{2} \sum_{j=1}^{l} j(2^{j+1} - 2^j) \\
&= \tfrac{1}{2} \left(\sum_{j=1}^{l} j 2^{j+1} - \sum_{j=1}^{l} j 2^j \right) \\
&= \sum_{j=1}^{l} j 2^j - \frac{1}{2} \sum_{j=0}^{l-1} (j+1) 2^{j+1} \\
&= \sum_{j=1}^{l} j 2^j - \sum_{j=0}^{l-1} j 2^j - \sum_{j=0}^{l-1} 2^j \\
&= \left(\sum_{j=1}^{l} j 2^j - \sum_{j=1}^{l-1} j 2^j \right) - (2^l - 1) \\
&= (l 2^l) - (2^l - 1) \\
&= (l-1) 2^l + 1
\end{aligned}$$

Assim, $\sum_{j=1}^{\lfloor log_2(n) \rfloor} j\ 2^{j-1} = (\lfloor log_2(n) \rfloor - 1) 2^{\lfloor log_2(n) \rfloor} + 1$ e obtemos

$$W(n) = (\lfloor log_2(n) \rfloor - 1) 2^{\lfloor log_2(n) \rfloor} + 1 + (n - 2^{\lfloor log_2(n) \rfloor}) \lceil log_2(n) \rceil$$

(c) Observe que o termo dominante na expressão anterior é $n \lceil log_2(n) \rceil$, sempre que $(\lfloor log_2(n) \rfloor - \lceil log_2(n) \rceil) 2^{\lfloor log_2(n) \rfloor}$ é menor ou igual que zero. Assim, conclui-se que $W(n) \in \Theta(n \, log_2(n))$.

3. (2.0 pontos) Segundo o límite inferior pior-caso para algoritmos de ordenação por comparação de chaves, nenhum dos algoritmos estudados em aula (*maxsort, bubblesort, insertionsort, binary-insertionsort, mergesort, quicksort, heapsort*) é ótimo.

   (a) (1.0 ponto) Qual é o límite inferior pior-caso para algoritmos de ordenação por comparação de chaves?

   (b) (1.0 ponto) **Descreva** em palavras um método ótimo para ordenar cinco chaves.

/R

(a) O límite inferior pior caso para algoritmos de ordenação por comparação de chaves para ordenar listas de tamanho n é de

$$\lceil log_2(n) \rceil$$

(b) Sejam a, b, c, d, e as cinco chaves da lista. Compare $a : b$ e $c : d$. Suponha o resultado da comparação é $a > b$ e $c > d$ (no caso contrário renomee chaves). Compare $a : c$. Suponha o resultado é $a > c$ (caso contrário renomee chaves). A situação é $a > c > d$ e $a > b$. Usando o método de inserção binária, insera e em $a > c > d$ com duas comparações e logo insera b na lista ordenada resultante das chaves a, c, d e e com duas comparações.

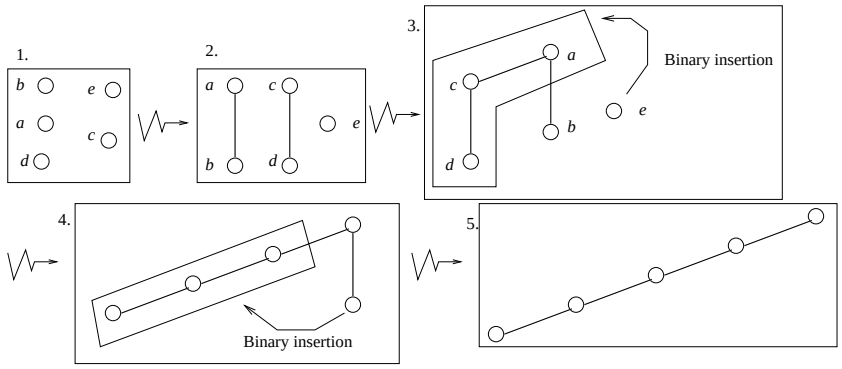

Figure 3: Ordenação ótima de cinco chaves