Rio de Janeiro, 15 de Abril de 2013.

PROVA 1 DE PROJETO E ANÁLISE DE ALGORITMOS

PROFESSOR: EDUARDO SANY LABER

DURAÇÃO: 2:50h

1. (3.0pt) Considere o algoritmo abaixo que recebe como parâmetros uma lista de inteiros L e um inteiro k e devolve a posição do k-ésimo menor elemento da lista.

Proc (L:lista,k:inteiro)
Se $|L| = 1$ Return L
Senão
    p ← Copperfield(L, $\lceil |L|/5 \rceil$) (*)
    Crie uma lista S com todos elementos de L que são menores que p (**)
    Crie uma lista B com todos elementos de L que são maiores que p (***)
    Se $k = \lceil |L|/5 \rceil$ Return p
    Se $|S| \geq k$ Return Proc(S,k)
    Se $|S| < k - 1$ Return Proc(B,$k - |L|/5$)
Fim Se

Assuma que a função Copperfield(L, i) na linha (*) devolve o i-ésimo menor elemento da lista L com complexidade de tempo $g(|L|)$.

a) Como e com que complexidade podemos implementar os passos das linhas (**) e (***)?

b) Seja $T(n)$ a complexidade de pior caso do algoritmo acima. Encontre uma equação de recorrência para $T(n)$. Assuma que $T(n)$ é não decrescente em n.

c) Assuma que $g(n) = O(n)$. Encontre uma função $f(n)$ tal que $T(n) = \Theta(f(n))$

2. (2.0pt) Considere o algoritmo abaixo que recebe um vetor A de números inteiros indexados de 1 a n.

Proc(A:vetor de n inteiros)
$i \leftarrow 1$
$j \leftarrow 1$
While $i \leq n$ **do**
    If A[i]==10
        While $j < i$ do
            j++
        End While
    End If
    $i++$
End While

a) Seja k o número de vezes em que 10 aparece no vetor A e seja $s_1 < s_2 < \ldots < s_k$ a sequência das posições em que o número 10 aparece no vetor A. Como exemplo se $A = [1, 6, 10, 2, 10, 3]$ então $k = 2$, $s_1 = 3$ e $s_2 = 5$. Se for conveniente assuma que $s_0 = 0$. Além disso, seja $T(n)$ a complexidade de pior caso do algoritmo acima. Escreva $T(n)$ como um somatório utilizando a sequência definida.

b) Seja $T(n)$ a complexidade de pior caso do algoritmo acima. Encontre uma função $f(n)$ tal que $T(n) = \Theta(f(n))$. Se não conseguir, exiba funções $g(n)$ e $h(n)$ tal que $T(n) = \Omega(g(n))$ e $T(n) = O(h(n))$. Quanto mais rápido o crescimento de $g(n)$ e mais lento o crescimento de $h(n)$, maior a pontuação.

3. (2.0pt) Seja S um conjunto de n elementos, onde cada elemento $s \in S$ tem uma prioridade $p(s)$ pertencente ao conjunto dos U menores inteiros positivos. Considere o seguinte procedimento

Para i=1,...,n^2

$x \leftarrow Rand(1, U)$

Modifique a prioridade do elemento de S com menor prioridade para x (*)

Fim Para

Assuma que a função $RAND(1, U)$ retorna em tempo constante um inteiro aleatório entre 1 e U. Além disso, caso haja mais de um elemento com prioridade mínima na linha (*), qualquer um pode ser escolhido.

a) Analise a complexidade do algoritmo acima quando S esta armazenado como um heap binário de mínimo. Note que após cada mudança de prioridade devemos restaurar a ordenação do heap.

b) Assuma agora que $1 \leq U \leq 5$. Como poderiamos armazenar S de modo a melhorar a complexidade do procedimento? Explique como seria o procedimento para modificar a prioridade do menor elemento para a nova estrutura de armazenamento proposta. Qual seria a complexidade obtida?

4. (2.0pt) Seja P uma cadeia de m caracteres indexada de 1 a m. A permutação circular P^s de P é obtida fazendo com que o sufixo de tamanho s de P seja o prefixo de P^s e o prefixo de tamanho $m - s$ de P seja o sufixo de P^s. Como exemplo, se $P = adfag$ então $P^2 = agadf$.

Dizemos que uma cadeia P ocorre em uma cadeia T de tamanho n, indexada de 1 a n, se existe algum $k \in \{1, \ldots, n-m+1\}$ tal que a cadeia P é idêntica a subcadeia de T que começa na posição k. Como exemplo $P = adfag$ ocorre em $T = dafadfagfa$. Dizemos que uma cadeia P ocorre circularmente em uma cadeia T se existe alguma permutação circular de P que ocorre em T.

Escreva o pseudo-código de um algoritmo que recebe como entrada uma cadeia P de tamanho m e uma cadeia T de tamanho n e responde SIM se P ocorre circularmente em T e responde NÃO, caso contrário. Analise em função de n e m a complexidade de tempo do algoritmo.

5. (2.0pt) Seja A um vetor de n inteiros indexado de 1 a n que satisfaz a seguinte propriedade: $A[i] < A[i+1]$ para $i = 1, \ldots, n-1$. Escreva o pseudo-código de um algoritmo que recebe como entrada o vetor A e responde SIM se existe algum índice j tal que $A[j] = j + 3$ e responde NÃO, caso contrário. Analise em função de n a complexidade de tempo do algoritmo.

Fatos úteis

1. A soma dos termos de uma progressão geométrica de n termos cujo primeiro termo é a_1 e a razão é q é igual a $\frac{a_1(q^n-1)}{q-1}$

2. A soma dos termos de uma progressão aritmética de n termos cujo primeiro termo é a_1 e o último é a_n é igual a $\frac{(a_1+a_n)n}{2}$