MC448: Projeto e Análise de Algoritmos I

Profs. Cid C. de Souza e Ricardo Dahab – 1ª Prova – (10/04/2008)

Nome:	
RA:	**Turma:**

Questão	frac	Peso	Nota
1			
2			
3			
4			
Total		10,0	

Observação: o peso das questões será decidido pelo docente da seguinte forma: as duas questões que você responder melhor terão peso 3 e as demais terão peso 2. Portanto, uma mesma questão pode ter peso 2 para um aluno e peso 3 para um outro aluno.

Instruções: A duração da prova é de 110 minutos. Não é permitido usar qualquer material de consulta durante a prova. **Questões mal justificadas serão consideradas erradas !** O termo **ordenado** é usado aqui para denotar vetores ou seqüências em **ordenação não decrescente**.

1. Seja k um número inteiro positivo. Deseja-se ordenar um vetor A de tamanho kn cujos elementos são números inteiros no intervalo $[0, (n^2 - 1)]$. Supondo k constante, descreva em **alto-nível** um algoritmo que ordena o vetor A em tempo linear em n. Além desta descrição você deve argumentar porque o seu algoritmo está correto e tem a complexidade desejada.

2. Abaixo encontra-se um pseudo-código do algoritmo AjustaHeap vista em sala de aula. Seja $T(n)$ a complexidade de **pior caso** deste algoritmo. Uma vez que o algoritmo é recursivo, Ana Saab Tudor, ex-aluna de MC448, descreveu esta complexidade através da recorrência

$$T(n) \leq T(\frac{2n}{3}) + c,$$

sendo c uma constante. Responda os itens abaixo:

(i) Explique com auxílio de desenhos qual o raciocínio usado por Ana para chegar a esta fórmula.

(ii) Resolva esta recorrência da forma que achar melhor, mostrando que $T(n) \in O(g(n))$, sendo $g(n)$ a função vista em aula para a compexidade do AjustaHeap. Se usar o Teorema Master, diga exatamente em qual dos seus casos sua recorrência se encaixa e por quê.

AjustaHeap(A, i, n)
▷ **Entrada:** Vetor A de n números inteiros com estrutura de heap, exceto, talvez, pela subárvore de raiz i.
▷ **Saída:** Vetor A com estrutura de heap.
1. **se** $2i \leq n$ e $A[2i] \geq A[i]$
2. **então** maximo $:= 2i$ **se não** maximo $:= i$
3. **se** $2i + 1 \leq n$ e $A[2i + 1] \geq A[\text{maximo}]$
4. **então** maximo $:= 2i + 1$
5. **se** maximo $\neq i$ **então**
6. $t := A[\text{maximo}]$; $A[\text{maximo}] := A[i]$; $A[i] := t$
7. AjustaHeap(A, maximo, n)

3. Um vetor $V[0..n-1]$, de n **elementos distintos**, inicialmente **ordenado**, sofreu uma rotação circular de k posições se o vetor $(V[k \mod n], V[k+1 \mod n], \ldots, V[k+n-1 \mod n]$ é o vetor V ordenado. Neste caso dizemos que V é um vetor *girado*. Projete **por indução** um algoritmo que dado um vetor girado $V[0..n-1]$, contendo n elementos distintos, encontre o **menor** elemento de V em tempo $O(\log n)$. Note que o valor k não é dado para o algoritmo.

Além da prova por indução, sua resposta deverá ter um pseudo-código do algoritmo, a fórmula de recorrência que descreve a sua complexidade e a resolução desta recorrência mostrando que ela é $O(\log n)$. Se usar o Teorema Master, diga em qual dos casos a recorrência se encaixa e por quê.

4. A Professora Jenny All inventou um novo algoritmo para ordenação de vetores que ela diz ser muito eficiente. Um pseudo-código do algoritmo NewSort projetado por ela é dado abaixo.

NewSort(A, i, j)
$\triangleright\ q$: vetor auxiliar de tamanho 5;
00. $n := (j - i + 1)$;
01. **se** $(n < 4)$ **então** BUBBLESORT(A, i, j);
02. **se não**
03. $k := n$ div 4; $r := n$ mod 4; $q[1] := 1$;
04. **para** $i = 2$ **até** $(4 - r + 1)$ **faça** $q[i] = q[i-1] + k$;
05. **para** $i = (4 - r + 2)$ **até** 5 **faça** $q[i] = q[i-1] + k + 1$;
06. NewSort$(A, q[1], q[3] - 1)$;
07. NewSort$(A, q[2], q[4] - 1)$;
08. NewSort$(A, q[3], q[5] - 1)$;
09. NewSort$(A, q[1], q[3] - 1)$;
10. NewSort$(A, q[2], q[4] - 1)$;
11. NewSort$(A, q[1], q[3] - 1)$;
12. **fim-se**

(i) Preencha a tabela abaixo para os valores de n indicados. Interprete o significado do vetor q.

n	k	r	$q[1]$	$q[2]$	$q[3]$	$q[4]$	$q[5]$
8							
9							
10							
11							

(ii) Mostre a corretude do algoritmo NewSort. Em particular, explique os objetivos da execução de cada uma das chamadas recursivas.

(iii) Encontre a fórmula de recorrência que representa a complexidade do algoritmo NewSort e resolva-a usando o método de sua preferência. Se usar o Teorema Master, diga exatamente em qual dos seus casos sua recorrência se encaixa e por quê.

(iv) Como se compara a eficiência do algoritmo da Professora Jenny All com aquelas dos algoritmos INSERTIONSORT e MERGESORT ?

(v) Comente a seguinte frase: *"A complexidade assintótica de pior caso do NewSort vai melhorar se, na linha 01, trocarmos o* BUBBLESORT *pelo* HEAPSORT*."*