Ollscoil na hÉireann
The National University of Ireland

Coláiste na hOllscoile, Corcaigh
University College, Cork

Mid-Term Examination 2010

CS4407 Analysis of Algorithms

Prof. G. Provan

Attempt all questions

Total marks: 100

60 minutes

Please answer all questions
Points for each question are indicated by [xx]

1. [25] Write the most efficient algorithm you can think of (in C, Java, pseudo-code) for the following:
   a. Given an array of n integers A[n], find the median (the number that divides A[n] into two equal halves).
   b. What is the running time in terms of big-oh, big-theta, or big-omega? Explain your answer.

SOLUTION:
A simple sorting algorithm (like mergesort or heapsort) will take Order of $O(n\lg_2 n)$ time.

Step	**Running Time**
Sort n elements using heapsort	$O(n\log_2 n)$
Return the k^{th} smallest element	$O(1)$
Total running time	$O(n\log_2 n)$

This can also be solved by using a divide-and-conquer approach, and then computing the recurrence.

2. **[25]** Solve the following recurrence relation using repeated substitution. Do an inductive proof to show your formula is correct.

$T(0) = 1$
$T(n+1) = 2*T(n)$

SOLUTION:
$T(n) = 2*T(n-1) = 2*[2*T(n-2)] = 2^2 T(n-2)$
$= 2^k T(n-k) = 2^n$

Induction Proof:
Prove base case: $T(0) = 2^n = 2^0 = 1$.
Assume $T(k)= 2^k$ and prove that $T(k+1) = 2^{k+1}$.
$T(k+1) = 2*T(k) = 2* 2^k = 2^{k+1}$.
QED.

3. **[25]** Use the Master Theorem to compute the complexity of the MergeSort algorithm by defining a suitable recurrence.

Consider a recurrence of the form:

$$T(n) = a\,T(n/b) + f(n) \qquad \textbf{1)}$$

where a and b are constants subject to

$$a \geq 1 \qquad b > 1$$

and the function $f(n)$ is asymptotically positive.

The Master Theorem gives the solutions to recurrences of the form specified in **1)** in the table below (3 cases).

	Case 1	**Case 2**	**Case 3**
If *f(n)* is	$f(n) = O\left(n^{\log_b(a)-\varepsilon}\right)$	$f(n) = \Theta\left(n^{\log_b(a)}\lg^k n\right)$	$f(n) = \Omega\left(n^{\log_b(a)+\varepsilon}\right)$
Under the conditions	$\varepsilon > 0$		$\varepsilon > 0$ $a\,f(n/b) \leq c\,f(n)$ $c < 1$

Then the order of the solution is	$T(n) = \Theta\left(n^{\log_b(a)}\right)$	$T(n) = \Theta\left(n^{\log_b(a)} \lg^{k+1} n\right)$	$T(n) = \Theta\left(f(n)\right)$

First, we must define the recurrence for MergeSort. Recall that MergeSort divides the array by half and sorts the resulting sublists. The functional form for integer comparison is cn for an array of length n, and so we obtain

T(n) = 2T(n/2) + cn.

By comparing the above to equation 1) of the recipe, we can identify

a = 2
b = 2
f(n) = cn.

To determine to which case this problem belongs, evaluate

$\text{Log}_b(a) = \text{Log}_2(2) = 1.$

Case 1: Does the following hold? $f(n) = O\left(n^{\log_b(a)-\varepsilon}\right)$

$f(n) = cn \neq O(n^{(1-\varepsilon)})$
Case 1 does not hold.

Case 2: Does the following hold? $f(n) = \Theta\left(n^{\log_b(a)} \lg^k n\right)$

$\Theta(n^1 \lg^k n) = \Theta(n \lg^k n) \Rightarrow k=0$ since *f(n) = cn.*
Hence Case 2 holds.

Therefore, $f(n) = \Theta(n^{(\log b(a)} \log^{k+1} n)$ and is of the form $\Theta(n \lg n)$.

4. **[25]** A string that contains only 0's, 1's and 2's is called a ternary string.
   a. State an algorithm to output the number of ternary strings that contain two consecutive symbols that are the same. For example, for n = 3, here are the strings: 001, 000, 100, 002, 200, 110, 011, 111, 112, 211, 220, 022, 122, 221, 222.
   b. Define a recurrence relation for the number of ternary strings that contain two consecutive symbols that are the same.

Set of all ternary strings = S*. The total number of length n ternary strings is 3^n.
Set of strings without consecutive symbols that are the same = S-.
Set of strings S with consecutive symbols = S* - S-.

Use induction to define recurrence for S-.

- length 1: 3 strings
- length 2: 3*2 strings (each length 1 string has 2 possible suffixes that work)
- length 3: 3*2*2 strings (2 suffixes for each length 2 string)
  ...
- length n: $3 * (2^{n-1})$ strings

So the number containing consecutive symbols that are the same is $3^n - (3 * (2^{n-1}))$

$T(n) = 3^n - (3 * (2^{n-1}))$