CS 3100 – Algorithms

Assignment 1
Kishore Kothapalli

Due 28/Jan/2009 at the beginning of the class. Strictly no extensions, and no late submissions shall be allowed.

Part 1

Each question carries 5 marks.

Problem 1 *Given an array A, devise an efficient algorithm to find the elements that appear more than once in the array A.*

Problem 2 *Given an array A design an efficient algorithm to find two elements x, y of A so that $|x - y|$ is minimized.*

Problem 3 *Consider the following statement*
"If there exists an efficient algorithm for Problem 2 above, then there exists an efficient algorithm for the Problem 1 above"
Is the statement true ? Why or why not.

Problem 4 *In an array A define a pair (i, j) to be an inversion if $i < j$ and $A[i] > A[j]$. Answer the following questions.*

- *Given $A = (3, 5, 2, 9, 6, 1)$ find the array B consisting of elements of A but with the maximum number of inversions.*
- *What is the relationship between the number of inversions in an array A and the runtime of insertion sort on A. Justify your answer.*

Problem 5 *Let A be a sorted array of integers. You wish to find whether any query integer x is present in the array A or not.*
a) Write the binary search method for the purpose.
b) Use the master theorem to prove that binary search never examines more than log n +1 numbers.

Problem 6 *Consider using insertion sort to sort an array A. Suppose we use binary search to find the position of the element that we are trying to insert into the already sorted sequence. How does this affect the worst-case runtime of insertion sort? Justify.*

Part 2

Each question carries 2 marks.

Prove the following recurrences.

- If $C_N = C_{N-1} + N$ for $N \geq 2$ with $C_1 = 1$, then $C_N = \frac{N(N+1)}{2}$.
- If $C_N = C_{\frac{N}{2}} + 1$ for $N \geq 2$ with $C_1 = 1$, then $C_N = lg\ N$ (Approx).
- If $C_N = C_{\frac{N}{2}} + N$ for $N \geq 2$ with $C_1 = 1$, then $C_N = 2N$ (Approx).
- If $C_N = 2 * C_{\frac{N}{2}} + N$ for $N \geq 2$ with $C_1 = 1$, then $C_N = N\ lg\ N$ (Approx).
- If $C_N = 2 * C_{\frac{N}{2}} + 1$ for $N \geq 2$ with $C_1 = 1$, then $C_N = 2N$ (Approx).

Part 3

Each question carries 1 mark.
Prove or disprove.

- $f(n) = O(g(n))$ implies $2^{f(n)} = O(2^{g(n)}$.
- $f(n) + g(n) = \Theta(\min\{f(n), g(n)\}$.
- $f(n) \neq O(g(n))$ implies $g(n) = O(f(n))$.
- $f(n) + g(n) = O(\min\{f(n), g(n)\})$.

Part 4

Each question carries $\frac{1}{2}$ mark.

Solve the following recurrence relations.

- $T(n) = 3T(n/2) + n \log n$
- $T(n) = T(\sqrt{n}) + 1$
- $T(n) = 8T(n/3) + n^2$
- $T(n) = T(n) = T(n-2) + n$
- $T(n) = T(n-2) + 2 \log n$
- $T(n) = \sqrt{n}T(\sqrt{n}) + n$
- $T(n) = 5T(n/3) + n^{4/3}$
- $T(n) = T(8n/9) + n$
- $T(n) = T(n-1) + \log n$
- $T(n) = 4T(n/2) + n^2 \log n$