CS60001 Advances in Algorithms, Autumn 2008– 09

Class test 1

Maximum marks: 40 | September 12, 2008 | Total time: 1 hour

Roll no: ________________ **Name:** ____

[*Write your answers in the question paper itself. Be brief and precise. Answer* <u>*all*</u> *questions.*]

1. Let A be an array of n integers $a_0, a_1, \ldots, a_{n-1}$ (negative integers are allowed). Denote, by $A[i \ldots j]$, the subarray $a_i, a_{i+1}, \ldots, a_j$ for $i \leqslant j$. Also let $S_{i,j}$ denote the sum $a_i + a_{i+1} + \cdots + a_j$. Your task is to find out the maximum value of $S_{i,j}$ over all allowed indices i, j. Call this maximum value S. For example, for the array $1, 3, -7, 2, -1, 5, -1, -2, 4, -6, 3$, this maximum sum is $S = S_{3,8} = 2+(-1)+5+(-1)+(-2)+4 = 7$. This example illustrates that the maximum sum may come from a subarray containing negative elements.

Let us also allow $j < i$ in the notation $A[i \ldots j]$. In this case, $A[i \ldots j]$ denotes the *empty* subarray (that is, a subarray that ends before it starts) with sum $S_{i,j} = 0$. Indeed, if all the elements of A are negative, then one returns 0 as the maximum subarray sum.

(a) Design a naive algorithm that computes $S_{i,j}$ for all the pairs i, j with $0 \leqslant i \leqslant j \leqslant n-1$, and obtains the maximum of these computed sums. Your program must run in $O(n^2)$ time. Write a pseudocode for your algorithm. Also supply an argument that your algorithm has $O(n^2)$ running time. **(10)**

Our plan is to arrive at an $O(n)$-time dynamic-programming algorithm to solve the maximum subarray sum problem.

(b) For $j \geqslant 0$, define E_j to be the maximum of all the values $S_{i,j}$ for $i = 0, 1, \ldots, j$. Thus, E_j represents the maximum subarray sum over all subarrays ending at index j. If no such subarray has positive sum, we take $E_j = 0$ (this corresponds to the empty suffix). We also take $E_{-1} = 0$. Prove that $E_j = \max(0, E_{j-1} + a_j)$ for $j \geqslant 0$. **(5)**

(c) Let $S_{-1} = 0$. For $j \geqslant 0$, define $S_j = \max_{i',j'} \left(\{S_{i',j'} \mid 0 \leqslant i' \leqslant j' \leqslant j\} \cup \{0\} \right)$. Our task is to compute $S_{n-1} = S$. Prove that $S_j = \max(S_{j-1}, E_j)$ for $j \geqslant 0$. **(5)**

(d) Describe an $\mathrm{O}(n)$-time algorithm for the computation of the maximum S. Write a pseudocode for your algorithm and also justify that your algorithm runs in linear time. Inefficient management of extra space will be penalized. **(10)**

(e) Suppose that the minimum sum $s = \min_{i,j} \left(\{S_{i,j} \mid 0 \leqslant i \leqslant j \leqslant n-1\} \cup \{0\} \right)$ is to be computed. Propose an $\mathrm{O}(n)$-time algorithm for this minimum subarray sum problem. **(5)**

(f) Modify the algorithm of Part (d) so that the indices i, j, for which $S_{i,j}$ is maximized, are computed (along with the maximum sum S). Your modification should continue to run in $O(n)$ time. **(10)**