Computing for Data Sciences – 2017

PGDBA, First Year, First Semester, Indian Statistical Institute

Assignment 0

Discussed in class during 26–28 July 2017 | Submit by 9 August 2017

Problem 1

[not graded]

Background : In a list (one-dimensional array) of n integers, L, an element $L[i]$ is called a *one-dimensional peak* (1D-Peak) if $L[i-1] \leq L[i] \geq L[i+1]$. In the boundary cases, the first element $L[0]$ is a 1D-Peak if $L[0] \geq L[1]$, the last element $L[n-1]$ is a 1D-Peak if $L[n-2] \leq L[n-1]$, and in case the list contains a single element (that is, if $n = 1$), the element is by default a 1D-Peak.

Note that the definition guarantees the occurrence of at least one 1D-Peak in any list. Moreover, there may be multiple 1D-Peaks in a list, as a 1D-Peak is not necessarily the maximum element.

Task : Write two Python functions, `LinSearchPeak1D()` and `BinSearchPeak1D()`, each of which takes a list of integers as input, and outputs – (a) one 1D-Peak (any one) from the list, and (b) the total number of integer-to-integer comparisons required to find that 1D-Peak.

`LinSearchPeak1D()` should exploit the *linear search* approach, and `BinSearchPeak1D()` should exploit the *binary search* approach, as discussed in class. Try to notice which one is faster.

Problem 2

[not graded]

Background : In a matrix (two-dimensional array) of $n \times n$ integers, M, an element $M[i]$ is called a *two-dimensional peak* (2D-Peak) if $M[i-1][j] \leq M[i][j] \geq M[i+1][j]$ and $M[i][j-1] \leq M[i][j] \geq M[i][j+1]$. In the boundary cases, the elements on the sides are 2D-Peaks if they are greater than or equal to the three neighboring elements, the elements in the corners are 2D-Peaks if they are greater than or equal to the two neighboring elements, and in case the matrix contains a single element ($n = 1$), it is by default a 2D-Peak.

Note that the definition guarantees the occurrence of at least one 2D-Peak in any list. Moreover, there may be multiple 2D-Peaks in a list, as a 2D-Peak is not necessarily the maximum element.

Task : Write a Python function, `SearchPeak2D()`, which takes a list-of-lists of integers as input, and outputs – (a) one 2D-Peak (any one) from the matrix, and (b) the total number of integer-to-integer comparisons required to find that 2D-Peak. Use the best algorithm you can.

Properly acknowledge every source of information that you referred to, including discussions with your friends. Verbatim copy from any source is strongly discouraged, and plagiarism will be heavily penalized. It is strongly recommended that you write the codes completely on your own.