Lecture 2 Engr. Sidra mudassar

Algorithm Design

• Algorithmic is a branch of computer science that consists of designing and analyzing computer algorithms

• The “design” pertain to

• The description of algorithm at an abstract level by means of a pseudo language, and

• Proof of correctness that is, the algorithm solves the given problem in all cases.

• The “analysis” deals with performance evaluation (complexity analysis).

Algorithm Performance Analysis

• Does the program efficiently use primary and secondary storage? • Is the program running time acceptable for the task?

• Space Complexity

• The space complexity of a program is the measure of the amount of memory that it needs to run to completion

• Time complexity

• The Time complexity of a program is the measure of the amount of computer time it needs to run to completion

How to analyze

•Running time depends upon •Single vs multi processor •Read/write speed to memory •32 bit vs 64 bit

•Input

If we talk about time complexity it means its depend upon on input Input is rate of growth of time taken

Performance Estimation

• How to determine which algorithm is better?

• We need some mechanism to predict the performance with out actually executing the program

• Mechanism should be independent of the compiler and underlying hardware

Algorithm Analysis

• The complexity of an algorithm is a function *g*(*n*) that gives bound of the number of operation (or

running time) performed by an algorithm when the input size is *n*

• There are two interpretations of bound. • **Worst-case** **Complexity**

• values of ithe input where the maximum is reached. Big O is used to calculate the worst case complexity

The runn ng time for any given size input will be lower than the upper bound except possibly for some

for an algorithm

• **Average-case** **Complexity**

• The running time for any given size input will be the average number of operations over all problem

instances for a given size

• **Best-case** **complexity**

• The best case of the algorithm is the function is defined by the minimum number of steps taken on any

instance of size n.

Big-Oh(O)

• Determining the exact step count of the program can be very difficult task

• Because of the in exactness of the definition of the step, exact step count is not very useful for comparative purposes

• E.g. which one is better • 45n+3 or 100n+10

• We use some asymptotic notation for measure of growth

Some useful Big-O estimates

• O(1): constant

• O(log(n)): Logarithm • O(n): Linear

• O(nlog(n)): Log Linear • O(n2): Quadratic

• O(n3):cubic

• O(2n):exponential