Algorithms, Week 2, Lec 1

Saif Ul Islam

Sep 7th, 2020

Contents

1 General Analysis Strategy

1. T(n): Maximum line taken by algorithm to solve any input of size n
2. T(n): Measure of goodness, how good or bad inidicated by function
3. The function will indicate how good is a algorithm
4. Conservative Definition - *Worst Case*.

What is *T(n)*? It indicates the maximum time it would take for a machine to run that algorithm - the worst case scenario?

1. Form of T(n) (independent from machine)
2. "Linear", "Cubic", "Quadratic" etc
3. Bounds of T(n), upper bound, lower bound
4. Large n is important, as n becomes larger and larger which algorithm is better

2 Running Time Analysis

1. The running time depends upon the input size, e.g., n
2. Different inputs of the same size may rsult in different running time

3. Criteria for measuring running time
4. Worst-Case Time (Maximum running time over legal inpout of size n)
5. Criteria Worst-case time
6. Let I denote an input instance
7. Let —I— denote its length
8. Let T(I) denote the running time of algorihtm on input I

Some measurements are as follows,

1. Average Case Time - the average running time over all inputs of size n
2. Let P(I) denote the probability of seeing this input
3. Average case time is the weighted sum of running times with weights being the probabilities

2.1 Example: 2-Dimension Maxima

The car selection problem can be modelled this way: For each car we assocaite (x, y) pair where,

1. x is the speed of the car
2. y is the negation of the price

2.1.1 Algorithm

MAXIMA(int n, Point P[1 ... n]) for i ¡- 1 to n do maximal ¡- true: for j ¡- 1 to n do if(i != j) and (P[i].x ¡= P[j].x) and (P[i].y ¡= P[j].y) then maximal ¡- false; break; if maximal == true continue;

2.1.2 Analysis

To do *Analysis*, just count the steps,