**CS430 Lecture 02 Activities**

Asymptotic Analysis – To simplify comparing the resource usage of different algorithms for the same problem

* ignore machine dependent constants; look at the growth of T(n) as n-> infinity
* as you double n, what does T(n) do?? double?? square??

Theta Notation (more details in future lectures)

* Drop lower order terms;
* Ignore leading constants
* Concentrates on the growth

1. For your best case, average case, worst case T(n) functions from above give the asymptotic function (Theta notation)

2. Given the problem sizes and worst case runtime for one of the problem sizes, and what you know about each algorithm, predict the missing runtimes.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | n=100 | n=200 | n=400 | n=800 |
| Linear search | 10 seconds |  |  |  |
| Binary search |  | 8 seconds |  |  |
| Insertion Sort |  |  | 320 seconds |  |

Opening Questions - Average Case Runtime

3. How did we approach average case runtime analysis of iterative algorithms previously? How can we improve on this?

Expectation of a Random Variable

A random variable is a variable that maps an outcome of a random process to a number. Examples:

* Flipping a coin, If heads X=1., if tails X=0
* Y=sum of 7 rolls of a fair die
* Z=in insertion sort, the number of swaps needed to move the ith item to its correct position in items 1 thru (i-1)

|  |  |
| --- | --- |
| The expected value of a random variable X is sum over all outcomes of the value of the outcome times the probability of the outcome. |  |

4. What is the expected outcome when you roll a fair die once? What about a loaded die where the probability of a side coming up is the value of the side divided by 21?

5. Calculate the expected outcome when you roll a fair die twice and sum the results. Do this two different ways.

Now let’s use expectation of a random variable to improve our average case runtime for insertion sort (similar for bubble sort or selection sort).

* Sort *n* distinct elements using insertion sort
* *Xi* is the random variable equal to the number of comparisons used to insert *ai* into the proper position after the first *i-1* elements have already been sorted. *1<=Xi<=i-1*

E(Xi) is expected number of comparisons to insert *ai* into the proper position after the first *i-1* elements have been sorted.

E(X) = E(X2)+E(X3)+ . . . +E(Xn) is the expected number of comparisons to complete the sort (our new average case runtime function).

6. Write equations for the following and simplify.

E(Xi)

E(X)

7. What if the data is not random?