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 $n \to \infty$.
- As you double n, what does $T(n)$ do?? Double?? Square??

Theta (Θ) Notation (more details in future lectures)

- Drop lower order terms;
- Ignore leading constants
- Concentrates on the growth
- Tight bound on growth

Ω is lowerbound, O (read as Big-O) is upperbound

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

$$T(n) = c_1 n + c_2(n-1) + c_3 \sum_{i=1}^{n-1} t_i + c_4 \sum_{1}^{n-1} (t_i - 1)$$

$$\text{Worst case: } t_i = i \to O(n^2)$$

$$\text{Best case: } t_i = i \to O(n)$$

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 $O(n)$	10 seconds	20	40	80
Binary search $O(\lg n)$	7	8 seconds	9	10
Insertion Sort $O(n^2)$	20	80	320 seconds	1,280

For Binary search, the steps increase logarithmically, meaning that for each iteration, the solution is cut in half. This means that (starting with $n = 200$), after one iteration the size has been cut in half, which would give you the $n = 100$ problem. So the time would only increase with 1 iteration for doubling the size. The 1-second increase was an estimation for how long each iteration was.

Opening Questions - Average Case Runtime

3. How did we approach case runtime analysis of iterative algorithms previously? How can we improve on this? Insertion sort – more or less, the inner loop needs to walk half way down on every other iteration.

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 (there is only 1-way to get a sum of 7, get 1's on every roll. But there are multiple ways you could get a sum of 12, so this would be a random variable.)
- Z =in insertion sort, the number of swaps needed to move the ith item to its correct position in items 1 through $(i-1)$. Range$(Z) = [0, i-1]$ (assuming all equally likely)

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

$$E(X) = \sum_{s \in S} X(s)\, p(s)$$

- s is the outcome
- X is the random variable (assigning a number to a certain outcome.)
- p is the probability

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?

$$1\left(\frac{1}{6}\right)+2\left(\frac{1}{6}\right)+3\left(\frac{1}{6}\right)+4\left(\frac{1}{6}\right)+5\left(\frac{1}{6}\right)+6\left(\frac{1}{6}\right)=\frac{21}{6}=3.5$$

$$1\left(\frac{1}{21}\right)+2\left(\frac{2}{21}\right)+3\left(\frac{3}{21}\right)+4\left(\frac{4}{21}\right)+5\left(\frac{5}{21}\right)+6\left(\frac{6}{21}\right)=\frac{1+4+9+16+25+36}{21}=\frac{91}{21}=\frac{13}{3}=4.\bar{3}$$

5. Calculate the expected outcome when you roll a fair die twice and sum the results. Do this two different ways. There are 36 possible combinations of rolls. $2 = (\{1,1\}) = \frac{1}{36}, 3 = (\{1,2\};\{2,1\}\}) = \frac{2}{36}, \ldots$

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
- X_i is the random variable equal to the number of comparisons used to insert a_i into the proper position after the first $i-1$ elements have already been sorted. $1 \le X_i \le i-1$

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

$E(X) = E(X_2) + E(X_3) + \cdots + E(X_n)$ is the expected number of comparisons to complete the sort (our new average case runtime function).

6. Write equations for the following and simplify.

$$\sum_{j=1}^{n} j = \frac{(n)(n+1)}{2}$$

- $E(X_i)$Outcomes: $1, 2, 3, \ldots, (i-1)$, Probability for each: $\frac{1}{i-1}$

$$\begin{aligned}
E(X_i) &= \left(\frac{1}{i-1}\right)1 + \left(\frac{1}{i-1}\right)2 + \left(\frac{1}{i-1}\right)3 + \cdots + \left(\frac{1}{i-1}\right)(i-1) \\
&= \left(\frac{1}{i-1}\right)\sum_{j=1}^{i-1} j \\
&= \left(\frac{1}{i-1}\right)\left[\frac{(i)(i+1)}{2} - i\right] \\
&= \frac{(i)(i+1)}{2(i-1)} - \frac{i}{i-1}
\end{aligned}$$

- $E(X)$

$$\sum_{i=2}^{n} \left(\frac{i}{2}\right) = O(n^2)$$

7. What if the data is not random?