Readings

- Lecture Notes Chapter 5: Running Time and Growth Functions

Review: Runtime Analysis

When analyzing algorithms, we can analyze the best case, average case, and worst case running times:

**Best Case Analysis:** This is when we analyze the runtime of the algorithm on the set of inputs in which it performs the fastest. This isn’t super useful because algorithms can be modified to make the best case performance trivial by hardcoded the solution/what to return for a specific input. In these situations, the best case performance is effectively meaningless.

**Worst Case Analysis:** This is when we analyze the runtime of the algorithm on the set of inputs with which it performs the slowest. This is useful because the worst case running time of an algorithm gives an upper bound on the running time for any input. In other words, the worst case running time provides a guarantee that the algorithm can never take any longer. Thus, unless otherwise specified, in CIS 121, we ask you to perform worst case analysis since it is the cleanest method of analysis.

**Average Case Analysis:** This is when we analyze the runtime of the algorithm on the “average” input. This doesn’t surface too much, as what constitutes an “average” input is usually not given to us and finding the “average” input requires a probability distribution of all possible inputs to the algorithm.

Problems

**Problem 1: True or False**

1. A Big-$O$, Big-$\Theta$, Big-$\Omega$ bound for an algorithm correspond to its worst-case, average-case, and best-case runtime, respectively.

2. For any two functions, $f(n)$ and $g(n)$, either $f(n) \in O(g(n))$ or $g(n) \in O(f(n))$.

3. $f(n) \in O(g(n))$ if and only if $g(n) \in \Omega(f(n))$.

4. If $f(n) \in O(g(n))$, then $f(n) \in o(g(n))$.

5. If $f(n) \in o(g(n))$, then $f(n) \in O(g(n))$.

**Problem 2**

Prove that $3n^2 + 100n = \Theta(5n^2)$.

**Problem 3**

Prove using induction that $n \log n = \Omega(n)$.

**Problem 4**

Prove that $\log(n!) = \Theta(n \log n)$.