Chapter 14-[Big-O](https://mfleck.cs.illinois.edu/building-blocks/updates-fa2017/big-o.pdf)

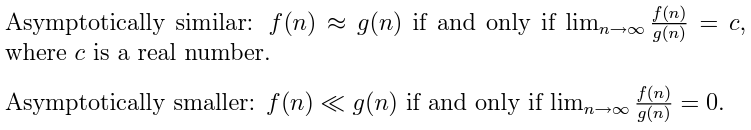
Saturday, January 7, 2023

10:35 PM

***Program Runtime and Asymptotic Relationships:***

To determine the efficiency of a program, we need to model the runtime of the program. However, we are lazy, so we just chuck a big value at it and see its behavior.

To compare 2 function's efficiency, we take the ratio of the 2 functions.



(Note that not all functions can be evaluated this way, since some functions will oscillate and others just don’t exist when put into a ratio, for those there are other methods)

***Primitive Functions Growth Rate:***

Higher-order polynomials grow faster than lower-order ones:



Exponentials grow faster than normal high order polynomials:



The base of the exponent also matters:



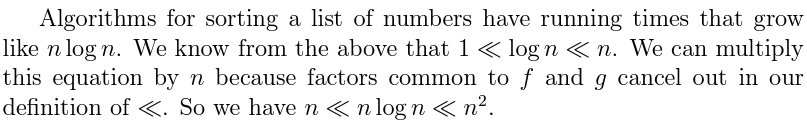
A tricky one that's not too obvious:



(since they both have n terms, but n! have bigger terms multiplied together)

Slow growing functions are better for computer programs.

(The log function, for example)



***Dominant Term Method:***

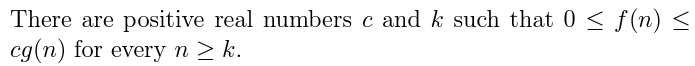
Most of the time we can just manipulate relationships at a high level and look at the primitive functions (the dominant expressions), and ignore the parts that grow slower.

*(Note that a constant multiple difference matters)*

***Big-****O*:

For messy functions, we use big O to compare their running times.

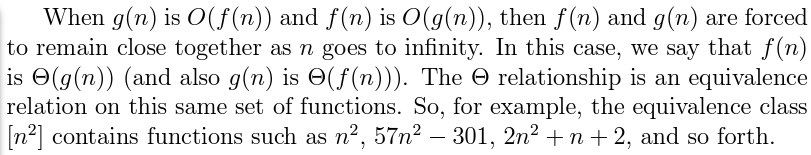
Instead of a simple ratio, we say that a function f(n) is ***O(***g(n)***)*** iff



In most cases, g(n) is a simple reference function, and this helps us understand how slow/fast the function in question f(n) runs.

*(Note that the function f can have the same/similar dominant term, regardless of a constant multiple difference, and the big o relationship still holds)*

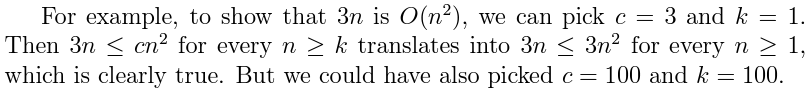
***The Equivalence Relationship:***



***Showing the Big-O Relationship:***

To show that a big-O relationship holds, we just need to find an *n* and *k* that work for the 2 functions.

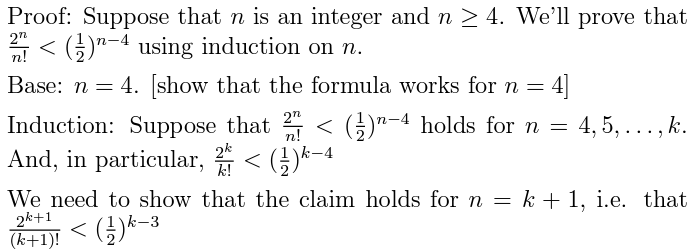
(Note that you can pick **any** *n* and *k,* as long as it works)



*(if you don't want to use overkill values every time, or if you don't know if a value is overkill enough, then pick a c value first, then plug it into the inequality. Then you can simplify the equation and start plugging and chugging)*

***Proving a Primitive Function Relationship:***





*(Note that when it comes to proving inequalities, you should write out explicitly the full inductive hypothesis and conclusion, like how both the assumed inequality and inequality to be proved are written out above)*

