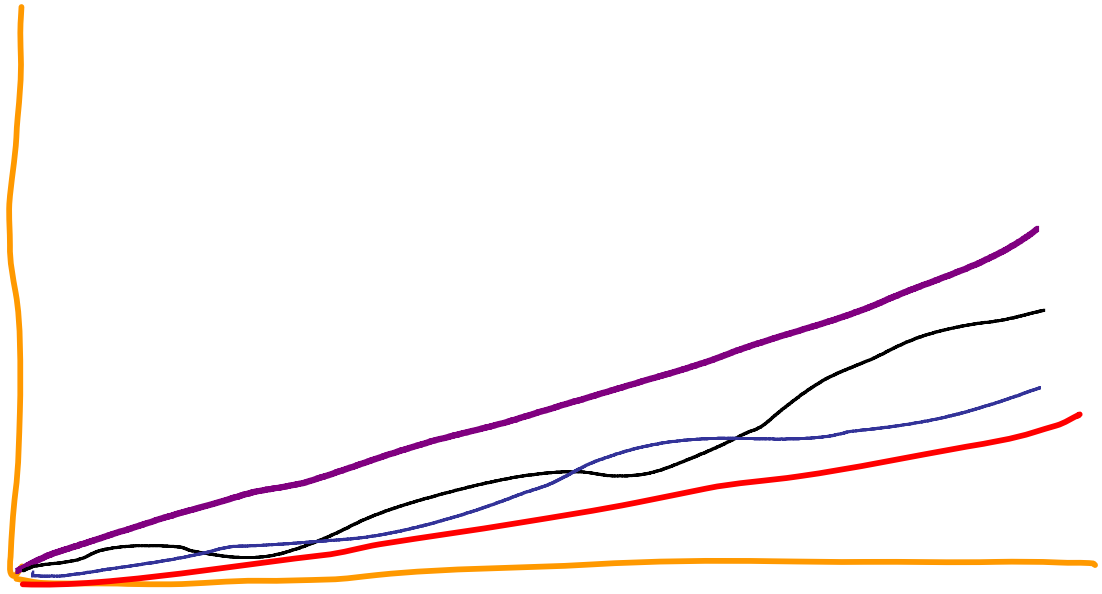
The growth of functions.

Runtime Growth Rates (I)

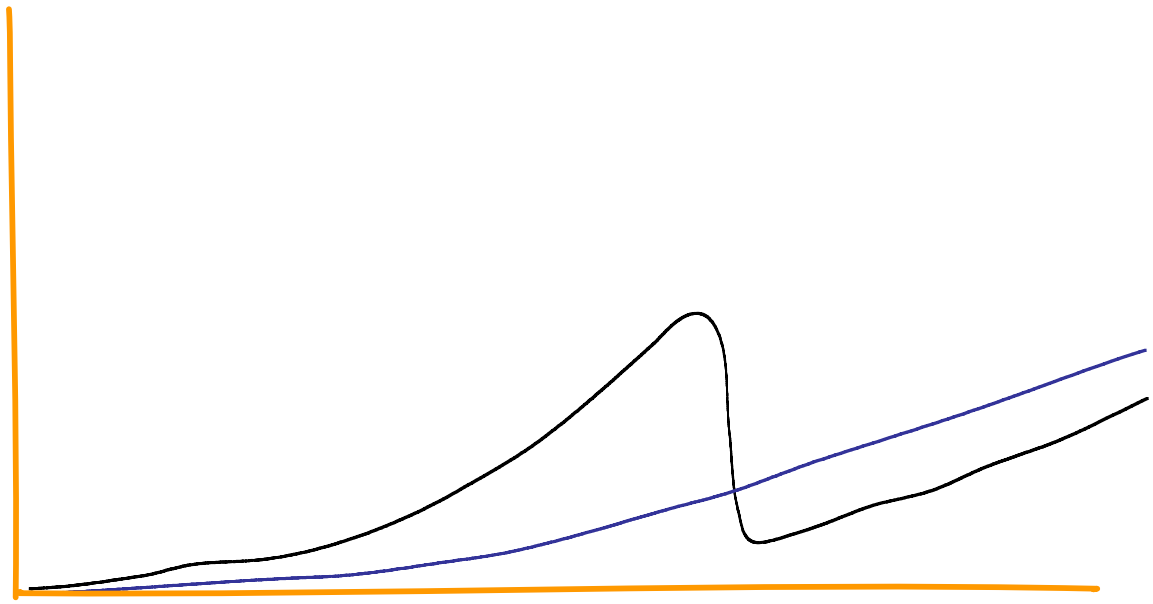
The runtimes of some (most?) algorithms are "clean" curves, though some do oscillate:



It can be useful when describing algorithms (or even problems) to discuss how their runtimes can be bound.

Runtime Growth Rates (II)

It is also possible that an algorithm has a runtime of T(n) only for "sufficiently large value s of n":



In this case, it can be useful to find that "crossover" value.

Asymptotic Analysis

There are FIVE different asymptotic measures of an algorithm.

- Big-O $O(\ )$
- Theta $\Theta(\ )$
- Omega $\Omega(\ )$
- little-o $o(\ )$
- little-omega $\omega(\ )$

These can all be used to discuss the relationship between how any two functions grow. We use them in a specific way when talking about (for example) the runtime of algorithms.

Big-O (or Big-Omicron)

$T(n) \in O(f(n))$ if and only if

$\exists n_0 \in \mathbf{Z}^+, c \in \mathbf{R}^+$ such that

$\forall n \in \mathbf{Z}^{\geq n_0}, T(n) \leq c \cdot f(n)$

We use this to state $f(n)$ as an upper bound on $T(n)$.

The fact that $0.5n^2 \in O(n^2)$ might be obvious, but what about the fact that $17n^2 \in O(n^2)$? While it is true that $42n \in O(n^2)$ would we ever use this fact in this way?

Let's look at some more involved examples…

Prove $\frac{1}{5}n^2 - 100n \in O(n^2)$

Prove $n^2 + 50n \in O(n^2)$

Big-O and Recurrences

The recurrence for MergeSort was:

$$T(0)=T(1)=1$$

$$T(n)=2T(n/2)+n$$

Let's show that $T(n) \in O(n \log n)$

Ω (Omega or Big-Omega)

$T(n) \in \Omega(f(n))$ if and only if

$\exists n_0 \in \mathbf{Z}^+, c \in \mathbf{R}^+$ such that

$\forall n \in \mathbf{Z}^{\geq n_0}, c \cdot f(n) \leq T(n)$

We use this to state f(n) as a lower bound on T(n).

Now, the fact that $17n^2 \in \Omega(n^2)$ should be obvious, but we can also say that $0.5n^2 \in \Omega(n^2)$? We can also say things like $42n^2 \in \Omega(n)$.

Let's look at some more involved examples…

Prove $n^2 + 50n \in \Omega(n^2)$

Prove $\frac{1}{5}n^2 - 100n \in \Omega(n^2)$

Θ (Theta)

If a function is both Ω and O of the same function class, then we say it is theta of that class.

For example, if we look at BubbleSort in more detail, we can show that it is in $\Omega(n^2)$ and $O(n^2)$ so we would call it $\Theta(n^2)$.

Formally, we have…

$T(n) \in \Theta(f(n))$ if and only iff

$\exists n_0 \in \mathbf{Z}^+, c_1, c_2 \in \mathbf{R}^+$ such that

$\forall n \in \mathbf{Z}^{\geq n_0}, c_1 \cdot f(n) \leq T(n) \leq c_2 \cdot f(n)$

Consider the following…

$T(n) = (n^2-n)/2$

Find n_0, c such that

$\forall n \in \mathbf{Z}^{\geq n_0}, c \cdot n^2 \leq T(n)$

to prove that this runtime is in $\Omega(n^2)$.

“Exam #1”

After revisiting sorting a little, we will continue on with the remaining asymptotic definitions in this deck as well as limits but there will not be questions about things past this slide within this set on the first exam.

Note: We will still be covering some new things before the first exam, it’s just the later things beyond this within this particular slide set that will not appear on the first exam.

Little-o

$T(n) \in o(f(n))$ if and only if

$\forall c \in \mathbf{R}^+$, $\exists n_0 \in \mathbf{Z}^+$, such that

$\forall n \in \mathbf{Z}^{\geq n_0}$, $T(n) < c \cdot f(n)$

Note that in this definition we are saying that the runtime grows *slower* than the given function $f(n)$.

$17n \in o(n^2)$

$3n^2 \notin o(n^2)$

$0.5n^2 \notin o(n^2)$

Little-o Proof

Show that $3n \in o(n^2)$

Need $\forall c \in \mathbf{R}^+$, $\exists n_0 \in \mathbf{Z}^+$, such that

$$\forall n \in \mathbf{Z}^{\geq n_0}, 3n < c \cdot n^2$$

Choose a generic particular c>0.

Show $\exists n_0 \geq 1$ *st* $\forall n \in \mathbf{Z}^{\geq n_0}, 3n < c \cdot n^2$

Let's build that n_0!

want $3n < cn^2$ true

want $3 < cn$ true

want $3/c < n$ true

OK, let $n_0 = 3/c + 1$

Let's make sure this really does work by plugging it in and proving the ∀...

Little-omega

$T(n) \in \omega(f(n))$ if and only if

$\forall c \in \mathbf{R}^+$, $\exists n_0 \in \mathbf{Z}^+$, such that

$\forall n \in \mathbf{Z}^{\geq n_0}$, $c \cdot f(n) < T(n)$

Note that in this definition we are saying that the runtime grows *faster* than the given function $f(n)$.

$0.5n^2 \in \omega(n)$

$3n^2 \notin \omega(n^2)$

$0.5n^2 \notin \omega(n^2)$

Other uses…

In a recurrence, we could make use of this by writing something such as:

$$T(n) = 2T(n/2)+\Theta(n)$$

In fact, we briefly saw this type of use when we discussed the runtime of MergeSort.

In this case we are saying that there is another cost to add, and that cost is dominated by a function of n.

We might use this if there are several different linear algorithms that we might choose to call, but we don't want to decide which yet.

Problems -vs- Algorithms

We have been discussing how to classify the runtime of algorithms.

It is also be possible to classify an entire problem.

For example, we would prove that a certain problem is in little-omega of n if we could prove that ***no*** linear-time algorithm could ever be able to solve it correctly on all inputs.

> This is very different from just proving that a specific linear-time algorithm does not work.

Some properties…

All of this relationships are transitive relationships.

Big-Omicron, Big-Omega, and Theta relationships are all reflexive.

Theta is the only relationship that is symmetrical!

Limits

There is another way of showing the relationship between the growth of two functions; limits.

If we let c = $\lim_{n\to\infty} \frac{f(n)}{g(n)}$

- if c>0 and c≠∞ that means

$f(n) \in \Theta(g(n))$

- if c≥0 and c≠∞ that means

$f(n) \in O(g(n))$

- if c>0 that means

$f(n) \in \Omega(g(n))$

Is $\frac{n^2-n}{2}$ in each class of (n^2)?

Is $\log n$ in each class of (n)?

L'Hôpital's rule would probably come in handy here…

Use limits to determine which runtime grows at a faster rate.

n -or- $n\log n$

$\log n$ -or- squareroot(n)

n^{500} -or- 2^n

Limits Recap

$$\lim_{n \to \infty} \frac{f(n)}{g(n)} = ?$$