GROWTH RATES OF SEQUENCES

Many functions or sequences go to infinity when their input goes to infinity. We have seen that n^2, $\ln n$, and 2^n all diverge to infinity. We often need to know how fast a sequence grows. One way to do this is to calculate many terms of the sequences and compare them.

n	n^2	$\ln n$	2^n
1	1	0	2
10	100	2.303	1024
100	10000	4.605	$\approx 10^{30}$
1000	1000000	6.908	$\approx 10^{301}$
10000	10^8	9.210	$\approx 10^{3010}$
100000	10^{10}	11.513	$\approx 10^{30103}$
1000000	10^{12}	13.816	$\approx 10^{301030}$
10000000	10^{14}	16.118	$\approx 10^{3010300}$
100000000	10^{16}	18.421	$\approx 10^{30103000}$

It appears that 2^n grows faster than n^2, which grows faster than $\ln n$. However, this is not a mathematically precise argument. Also, we can never be sure how many terms must be calculated before a pattern become apparent. Instead, we can compare two sequences by examining the ratio of their terms as n grows large.

Definition 1. A sequence $f(n)$ **grows faster than** $g(n)$ (or g grows slower than f) if

$$\lim_{n\to\infty} \frac{f(n)}{g(n)} = \infty \text{ or } \lim_{n\to\infty} \frac{g(n)}{f(n)} = 0.$$

In this case we write $f \gg g$ or $g \ll f$.

Sequences $f(n)$ and $g(n)$ **grow at the same rate** if for some L, $0 < L < \infty$,

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

Example. Compare the following pairs of sequences and determine which grows faster.

a. $\ln n$ and n

b. $\ln n$ and $\ln \ln n$

c. e^n and n^p, $p > 1$

Solution. We find the limit of the ratio of functions, using L'Hopital's Rule as necessary.

$$\lim_{n\to\infty} \frac{\ln n}{n} = \lim_{x\to\infty} \frac{\frac{1}{n}}{1} = 0$$

$$\lim_{n\to\infty} \frac{\ln \ln n}{\ln n} = \lim_{n\to\infty} \frac{\frac{1}{n \ln n}}{1/n} = \lim_{n\to\infty} \frac{1}{\ln n} = 0$$

$$\lim_{n\to\infty} \frac{e^n}{n^p} = \lim_{x\to\infty} \frac{e^n}{pn^{p-1}} = \lim_{n\to\infty} \frac{e^n}{p(p-1)n^{p-2}} = ... = \infty$$

Thus we see $\ln \ln n \ll \ln n \ll n$, and $n^p \ll e^n$. Thus any power function grows slower than e^n (or any exponential function with base more than 1). Many common functions can be ordered by their growth rates, as follows.

Theorem 2.

$$\ln \ln n \ll \ln n \ll \underset{0<p<1}{n^p} \ll n \ll n \ln n \ll \underset{p>1}{n^p} \ll e^n \ll n! \ll n^n \ll e^{n^2}$$

One complication is the factorial function, which is used in Taylor series and many counting problems. Since it is nondifferentiable, it cannot be used directly. However, **Stirling's approximation**, $n! \approx \sqrt{2\pi n}\left(\frac{n}{e}\right)^n$, can be used for these calculations.

Additional sequences can be inserted into this order. For example, additional exponential sequences $(1.1)^n \ll 2^n \ll e^n \ll 3^n$ could be inserted in place of e^n. The polynomial sequences $n^2 \ll n^3 \ll n^4 \ll n^5$ could be inserted. Not every sequence fits into this order; one could construct a sequence that crosses more than one of these sequences infinitely many times. However, these are the most common sequences used in calculus.

Sequences can be very complicated. However, often an approximation is all we need. The following notation is called **big-O notation**.

Definition 3. Let f and g be sequences. We say $g \in \mathcal{O}(f)$ if $|g(n)| \leq c|f(n)|$ for some c and sufficiently large n.

If $g \in \mathcal{O}(f)$, we say g is $\mathcal{O}(f)$. This means that g grows at most as fast as f.

Example. The sequences n^3, n^2, and $\ln n$ are all in $\mathcal{O}(n^4)$, but e^n is not.

If $f(n) \ll g(n)$, what can we say about $g(n) + f(n)$? We see

$$\lim_{n\to\infty} \frac{g(n)+f(n)}{g(n)} = \lim_{n\to\infty}\left(1+\frac{f(n)}{g(n)}\right) = 1+0=1.$$

Thus $g(n)+f(n) \in \mathcal{O}(g(n))$.

The sequences n^2-n and n^2-3n+2 are in $\mathcal{O}(n^2)$. They all grow at the same rate, since they have the same leading term. This is what really matters, since when n gets large, all other terms are small compared to the leading term. Thus every polynomial grows at the same rate as its leading term. This also implies that every polynomial grows slower than e^n.

One use of this information is in evaluating the complexity of an algorithm. This is a major topic in computer science. Big-O notation is used to estimate how many operations an algorithm performs. It is typical to chop off the trailing terms and just report the leading term, which is a basic function like those in Theorem 2. Using a more efficient algorithm saves lots of time and money when the size of the input is large.

Example. Suppose you want to search an alphabetical list of names for a given name. One option, **sequential search**, is to just check every name in order. If the list has n names, we may use up to n comparisons.

An alternative is to go to the middle of the list and check this name. If it is not the one we are searching for, use either the first or second half of the list depending on how the given name compares to the middle name. Repeat this process with the smaller list until the name is found or there are no more names to check. This is called **binary search**. If there are at most $n=2^r$ names, binary search will use at most $r=\log_2 n$ comparisons. Thus the complexity of binary search is $\mathcal{O}(\log_2 n)$, which is better than $\mathcal{O}(n)$ for sequential search.

A closely related question is how quickly functions go to zero. Note that if $f(n) < g(n)$, then $\frac{1}{f(n)} > \frac{1}{g(n)}$. Reversing the chain of inequalities above shows the following, for n sufficiently large.

$$\frac{1}{n!} < \frac{1}{e^n} < \underset{p>1}{\frac{1}{n^p}} < \frac{1}{n\ln n} < \frac{1}{n} < \underset{0<p<1}{\frac{1}{n^p}} < \frac{1}{\ln n} < \frac{1}{\ln\ln n}$$

This is important for the comparison test for improper integrals and infinite series. It is also useful to determine how quickly infinite series converge.

Exercises.

1. Determine which sequence grows faster, or if they grow at the same rate.

a. n^{1000}, e^n

b. $\ln\ln n^2$, $\ln\ln\sqrt{n}$

c. $n!$, $e^{n\ln n}$

d. $\sqrt{n^2-6}$, $\sqrt[3]{1+n^6}$

2. Put the following sequences in order of their growth rate. Where do they fit in the order of sequences in Theorem 2?

a. $n^3 2^n$, $n^2 3^n$, $n^4 e^n$

b. $n(\ln n)^2$, $n^2\ln n$, $n\ln(n^2)$

c. $\ln(n\ln n)$, $\ln\ln 2^n$, $\ln\ln\ln n$

d. $e^{\sqrt{n}}$, n^n, e^{n^2}

3. Determine which sequences are in $\mathcal{O}(f)$ for the given f. (Explain.)

a. $n+3$, n^2+4n, n^2-n^3, $\sqrt{n^4+5}$; $f=n^2$

b. $\log_2 n$, $\log n$, $\sqrt{n}$, $\ln n^2$; $f=\ln n$

c. 3^n, $4^{n/2}$, n^n, $e^{n+\frac{1}{n}}$; $f=e^n$

d. $n^{\ln n}$, $(\ln n)^n$, $2n!$, $(2n)!$; $f=n!$

4. Show that if $f(n)$ and $g(n)$ grow at the same rate, then $f \in \mathcal{O}(g)$ and $g \in \mathcal{O}(f)$.

5. One common computer science problem is sorting data. The **quicksort** algorithm has complexity $\mathcal{O}(n^2)$, and the **merge sort** algorithm has complexity $\mathcal{O}(n\log_2 n)$. Determine which algorithm is better.

6. For the **Traveling Salesman Problem**, a brute force algorithm has complexity $\frac{(n-1)!}{2}$, and the **Held-Karp algorithm** has complexity $\mathcal{O}(n^2 2^n)$. Determine which algorithm is better.