CS/Math 240: Introduction to Discrete Mathematics

Reading 10 : Asymptotic Analysis

Author: Dieter van Melkebeek (updates by Beck Hasti and Gautam Prakriya)

In the last reading, we analyzed the running times of various algorithms. We expressed the number of steps an algorithm makes as a function of the input size. In this reading, we introduce asymptotic analysis. Which is a tool for comparing the the running times of different algorithms.

While analyzing the running time of algorithms in the last reading, we were interested in the number of elementary steps performed as a function of the input size. For example when analyzing the powering algorithms in class, we chose the to count the number of multiplications. But this was by no means the only meaningful choice of an elementary step. We could have instead counted the number of recursive calls made or the number of lines of code executed. Each choice gives us a different function. But these functions only differ by constant factors. The dependence on the input size still remains the same. This suggests that when comparing running times, we should not be concerned with constant factor differences.

Additionally, when comparing running times, we are concerned only with how the algorithms behave on large inputs. This is because, differences between the complexities of algorithms may not be noticeable on small inputs, and any computer could run the two algorithms with different, but still small running times in a short period of time. But when the inputs get larger, the difference is more pronounced and the slower algorithm may not have a chance of finishing before the end of the universe, even on a supercomputer, whereas a slow computer could still execute the faster algorithm in a few days, or perhaps even seconds.

Asymptotic analysis allows us to achieve both these goals.

10.1 Asymptotic Analysis

In asymptotic analysis, we are interested in the relative growth of two functions f, g from $\mathbb{N}$ to $\mathbb{R}_{>0}$, where by $\mathbb{R}_{>0}$ we mean the positive real numbers. We think of one or both of these functions as running times of programs. If f represents the running time of a program P, $f(n)$ tells us how many elementary operations P takes on input (of length) n.

Now let's forget about the fact that f and g represent running times of programs on various inputs, and set down some notation and terminology that is generally applicable.

We start with a very restrictive notion of equivalence.

Definition 10.1. *Given $f : \mathbb{N} \to \mathbb{R}_{>0}$ and $g : \mathbb{N} \to \mathbb{R}_{>0}$, we say $f(n)$ and $g(n)$ are* asymptotically equivalent *if*

$$(\forall \epsilon \in \mathbb{R}_{>0})(\exists N \in \mathbb{N})(\forall n \geq N)\ \ 1 - \epsilon \leq \frac{f(n)}{g(n)} \leq 1 + \epsilon. \tag{10.1}$$

We use the notation $f(n) \sim g(n)$ to denote that $f(n)$ and $g(n)$ are asymptotically equivalent.

In words, as n increases, the ratio of $f(n)$ and $g(n)$ approaches 1. Another way to formulate Definition 10.1 is $\lim_{n\to\infty} f(n)/g(n) = 1$.

Definition 10.1 captures one of the goals we have set for ourselves. It characterizes the behavior of two functions as their inputs get large.

Example 10.1: Let F_n denote the n-th fibonacci number. Recall that fibonacci numbers are defined as follows:

$$F_1 = 1, F_2 = 1$$

and for $n > 2$

$$F_n = F_{n-1} + F_{n-2}.$$

Notice that this is slightly different from the stairs problem in Hw 6. The difference is that the foundation rule corresponds to $n = 1$ and $n = 2$ here , instead of $n = 0$, $n = 1$ as in the stairs problem.

When defined as above, the solution to the recurrence for F_n works out to be:

$$F_n = \frac{\varrho^n - (1-\varrho)^n}{\sqrt{5}}$$

where $\varrho = (1+\sqrt{5})/2$.

$F_n \sim \frac{\varrho^n}{\sqrt{5}}$. Consider $f(n) = F_n$ and $g(n) = \frac{\varrho^n}{\sqrt{5}}$.

Now let's look at the ratio of f and g.

$$\frac{f(n)}{g(n)} = \frac{\frac{\varrho^n-(1-\varrho)^n}{\sqrt{5}}}{\frac{\varrho^n}{\sqrt{5}}} = \frac{\varrho^n - (1-\varrho)^n}{\varrho^n} = 1 - \frac{(1-\varrho)^n}{\varrho^n} = 1 - \left(\frac{1-\varrho}{\varrho}\right)^n. \tag{10.2}$$

Notice that $(1-\varrho)/\varrho < 1$, so raising it to a high power yields a number that is close to zero. Thus, as n increases, the term $[(1-\varrho)/\varrho]^n$ gets closer and closer to zero, which means (10.2) gets closer and closer to one.

To show that $f(n) \sim g(n)$, consider $\epsilon \in \mathbb{R}_{>0}$, and pick an N such that $[(1-\varrho)/\varrho]^N < \epsilon$. Since 1 doesn't change with n, and $[(1-\varrho)/\varrho]^n$ gets closer to zero as n increases, we get that $f(n)/g(n)$ is bounded between $1-\epsilon$ and $1+\epsilon$ for all $n \geq N$. Thus, $f(n) \sim g(n)$. ⊠

Example 10.2: Consider $f(n) = 12n^2 + 10^9 n + \frac{1}{n}$ and $g(n) = 12n^2$. Then $f(n) \sim g(n)$.

To see this, observe that as n increases, $12n^2$ grows much faster than the other terms in $f(n)$. Thus, when n is large, $10^9 n$ becomes negligible compared to $12n^2$ even though the coefficient in front of n may look like a giant number. A common expression for this is that $12n^2$ *dominates* $10^9 n$. Similarly, $12n^2$ dominates $\frac{1}{n}$.

Now we argue more formally. We can write

$$\frac{f(n)}{g(n)} = \frac{12n^2 + 10^9 n + \frac{1}{n}}{12n^2} = 1 + \frac{10^9}{12} \cdot \frac{1}{n} + \frac{1}{12} \cdot \frac{1}{n^3}.$$

Since both the terms $10^9/12n$ and $1/12n^3$ decrease towards zero as n increases, we can find an N for which Definition 10.1 is satisfied using the same argument as in the previous example.

Also notice that $f(n) \not\sim n^2$. The ratio of $f(n)$ and n^2 approaches 12 as n goes to infinity, and the ratio won't get close to one. ⊠

Example 10.3: $F_n \not\sim 12n^2$.

The Fibonacci sequence grows much faster than the quadratic $12n^2$. Let $f(n) = F_n$ and $g(n) = 12n^2$. For small values of n, we actually have $f(n) < g(n)$, but that quickly changes and $f(n)$ starts growing much faster than $g(n)$. One way to see this is that $f(n)$ increases by a factor of ϱ if n increases by 1, whereas $12n^2$ increases by a factor that is close to 1 (just write out $12(n+1)^2/12n^2$ to see this) if n increases by 1. ⊠

We see from the examples above that Definition 10.1 accomplishes some of our goals, but not all of them. It allows us to capture the behavior of functions as their inputs grow. However, as we saw in Example 10.2, there is still dependence on constant factors. In particular, we saw $f(n) \sim 12n^2$, but $f(n) \not\sim n^2$. Since $f(n)$, $12n^2$, and n^2 grow like quadratics, we would like them to be similar, but asymptotic equivalence doesn't allow that. The next definition fixes this shortcoming of asymptotic equivalence.

Definition 10.2 (Big Theta). *Given $f : \mathbb{N} \to \mathbb{R}_{>0}$ and $g : \mathbb{N} \to \mathbb{R}_{>0}$, we say $f(n)$* is Theta of *$g(n)$ if*

$$(\exists c, d \in \mathbb{R}_{>0})(\exists N \in \mathbb{N})(\forall n \geq N) \ \ c \leq \frac{f(n)}{g(n)} \leq d. \tag{10.3}$$

We write $f(n) = \Theta(g(n))$ to denote that $f(n)$ is Theta of $g(n)$.

Definition 10.2 no longer requires f and g to approach each other in the limit. Instead, it requires that their ratio be bounded above and below by some constants as n gets large. This allows us to ignore constant factors because the Θ notation "absorbs" them.

Let's see how the examples we used for asymptotic equivalence change. First, if $f(n) \sim g(n)$, then certainly $f(n) = \Theta(g(n))$. To see this, pick $\epsilon = 1/2$ (we can pick any $\epsilon > 0$ to make this argument work; $\epsilon = 1/2$ is just an example). Since $f(n) \sim g(n)$, there is an N such that for all $n \geq N$, $1/2 \leq f(n)/g(n) \leq 3/2$. So take that N, and pick $c = 1/2$ and $d = 3/2$ to show $f(n) = \Theta(g(n))$.

Example 10.4: By the previous paragraph, $F_n = \Theta(\varrho^n/\sqrt{5})$ and $12n^2 + 10^9 n + 1/n = \Theta(12n^2)$.

In addition, we also have $12n^2 + 10^9 n + 1/n = \Theta(n^2)$ because we can pick $N = 1$, $c = 12$ and $d = 10^9 + 13$ in order to satisfy Definition 10.2. Another option is $c = 12$, $d = 13$, $N = 10^9$. This second choice illustrates that we only care about larger values of n in asymptotic analysis. It also allowed us to pick c and d that are closer to each other.

Finally, $F_n \neq \Theta(12n^2)$ because F_n grows exponentially, and we cannot bound the ratio of F_n and $12n^2$ by any constant. ⊠

We may also be interested in only one of the bounds in (10.3). For example, if we want to guarantee that a program terminates within a number of steps that is at most some function of the input, we only care about the upper bound. On the other hand, if we want to prove to someone that the number of steps is at least some function of the input, we only care about the lower bound. Thus, we make the following two definitions.

Definition 10.3 (Big Oh). *Given $f : \mathbb{N} \to \mathbb{R}_{>0}$ and $g : \mathbb{N} \to \mathbb{R}_{>0}$, we say $f(n)$* is Oh of *$g(n)$ if*

$$(\exists d \in \mathbb{R}_{>0})(\exists N \in \mathbb{N})(\forall n \geq N) \ \ \frac{f(n)}{g(n)} \leq d. \tag{10.4}$$

We write $f(n) = O(g(n))$ to denote that $f(n)$ is Oh of $g(n)$.

Definition 10.4 (Big Omega). *Given $f : \mathbb{N} \to \mathbb{R}_{>0}$ and $g : \mathbb{N} \to \mathbb{R}_{>0}$, we say $f(n)$* is Omega of *$g(n)$ if*

$$(\exists c \in \mathbb{R}_{>0})(\exists N \in \mathbb{N})(\forall n \geq N) \ \ c \leq \frac{f(n)}{g(n)}. \tag{10.5}$$

We write $f(n) = \Omega(g(n))$ to denote that $f(n)$ is Omega of $g(n)$.

Observe that if $f = \Theta(g)$, then $f = O(g)$ and $f = \Omega(g)$. This follows because conditions (10.4) and (10.5) are weaker than condition (10.3). Also, if both $f = O(g)$ and $f = \Omega(g)$ hold, then $f = \Theta(g)$ because combining the inequalities from (10.4) and (10.5) gives us the inequality (10.3).

Example 10.5: By the previous paragraph, we have $F_n = O(\varrho^n/\sqrt{5})$ and $F_n = \Omega(\varrho^n/\sqrt{5})$. We also get $12n^2 + 10^9 n + 1/n = O(12n^2)$ and $12n^2 + 10^9 n + 1/n = \Omega(12n^2)$, as well as $12n^2 + 10^9 n + 1/n = O(n^2)$ and $12n^2 + 10^9 n + 1/n = \Omega(n^2)$.

Since F_n grows way faster than $12n^2$, $F_n \neq O(12n^2)$, but it is true that $F_n = \Omega(12n^2)$. Along the same vein, $12n^2 = O(F_n)$ and $12n^2 \neq \Omega(F_n)$. ⊠

Remark : If $\lim_{n\to\infty} \frac{f(n)}{g(n)}$ exists, then we can figure out the relationship between f and g is using the limit:

1. If $\exists c > 0, \lim_{n\to\infty} \frac{f(n)}{g(n)} \leq c$ then $f(n) = O(g(n))$.
2. If $\exists c > 0, \lim_{n\to\infty} \frac{f(n)}{g(n)} \geq c$ then $f(n) = \Omega(g(n))$.
3. If $\exists c > 0, \lim_{n\to\infty} \frac{f(n)}{g(n)} = c$ then $f(n) = \Theta(g(n))$.

These are not if and only if statements. We could have $f(n) = O(g(n))$ even if $\lim_{n\to\infty} \frac{f(n)}{g(n)}$ doesn't exist. But, if the limit exists, the above gives us a convenient way of figuring out the relationship between f and g. Here are some additional facts:

- If $\lim_{n\to\infty} \frac{f(n)}{g(n)} = 0$ then $f(n) = O(g(n))$ and $f(n) \neq \Omega(g(n))$.
- If $\lim_{n\to\infty} \frac{f(n)}{g(n)} = \infty$ then $f(n) = \Omega(g(n))$ and $f(n) \neq O(g(n))$.

Practice Problem: For each of the following pairs, relate the functions defined by the expressions as tightly as possible using O ,Ω Θ and $\sim$: (assume the logarithms are with base 2.)

- $n^3 + 15\log n$ and $n^3 + 16n^2 + 32$.
- $n \log n$ and n^2
- $\log(n^2+1) + n^2$ and n^2
- 2^n and 4^n
- $15n^4$ and 2^n
- $2^{\log^2 n}$ and $n^{\log n}$

Example 10.6: Let $f(n) = \log(n!)$ and $g(n) = n\log n$ ($n! = n \cdot (n-1)\cdots 1$). Assume that the logarithms are with base 2.

Then $f(n) = \Theta(g(n))$.

We prove this by showing $f(n) = O(g(n))$ and $f(n) = \Omega(g(n))$.

By basic properties of the logarithm, we have:

$$\log(n!) = \log(n) + \log(n-1) + \ldots \log(1).$$

Every term on the right hand side is at most $\log(n)$ and there are n terms in the sum on the right hand side. This tells us $\log(n) + \log(n-1) + \ldots \log(1) \leq n \cdot \log n$.

Therefore $\forall n \geq 1, \log(n!) \leq n \cdot \log n$. Taking $d = 1$ and $N = 1$ in the definition of Big -Oh, we have $\log(n!) = O(n \log n)$.

We now show that $\log(n!) = \Omega(n \log n)$. We have $\log(n!) = \log(n) + \log(n-1) + \ldots \log(1)$. Observe that $\log(n/2) = \log(n) - 1$. So the first half of the terms in the sum $\log(n), \log(n-$

$1)\ldots\log(\lfloor n/2\rfloor+1)$ are all at least $\log(n)-1$. Which implies $\log(n!)\geq n/2\cdot(\log(n)-1)$. This is spelled out in more detail below.

Consider the largest $\lfloor n/2\rfloor+1$ terms of this sum.

$$\log(n!)=\log(n)+\log(n-1)+\ldots\log(1)\geq\log(n)+\log(n-1)+\ldots\log((n-\lfloor n/2\rfloor).$$

$$\log(n)+\log(n-1)+\ldots\log(n-\lfloor n/2\rfloor)=\log(n)+\log(n-1)+\ldots\log(\lceil n/2\rceil).$$

This is because $n=\lfloor n/2\rfloor+\lceil n/2\rceil$. (The ceiling function $\lceil x\rceil$ is the smallest integer $\geq x$. Therefore, if n is even $\lceil n/2\rceil=n/2$ and if n is odd, $\lceil n/2\rceil=\frac{n+1}{2}$.)

Now, regardless of whether n is even/ odd $\lceil n/2\rceil\geq n/2$, which implies $\log(\lceil n/2\rceil)\geq\log(n/2)=\log(n)-1$.

Therefore, each of the $\lfloor n/2\rfloor+1$ terms in the sum $\log(n)+\log(n-1)+\ldots\log(\lceil n/2\rceil)$ is at least $\log(n)-1$.

Which implies $\log(n)+\log(n-1)+\ldots\log(\lceil n/2\rceil)\geq(\lfloor n/2\rfloor+1)\cdot(\log(n)-1)$.

Since $\lfloor n/2\rfloor+1\geq\frac{n}{2}$,

$$\log(n)+\log(n-1)+\ldots\log(\lceil n/2\rceil)\geq\frac{n}{2}\cdot(\log(n)-1).$$

The above implies that $\forall n\geq 1,\log(n!)\geq\frac{n}{2}\cdot(\log(n)-1)$. To show $\log(n!)=\Omega(n\log n)$, it suffices to prove $\frac{n-1}{2}\cdot(\log(n)-1)=\Omega(n\log n)$ (why ?).

We leave this as an exercise for the reader.

$\boxtimes$