Lecture 5: Asymptotics

(Tuesday, Feb 10, 1998)

Read: Chapt. 3 in CLR. The Limit Rule is not really covered in the text. Read Chapt. 4 for next time.

Asymptotics: We have introduced the notion of $\Theta()$ notation, and last time we gave a formal definition. Today, we will explore this and other asymptotic notations in greater depth, and hopefully give a better understanding of what they mean.

Θ-Notation: Recall the following definition from last time.

Definition: Given any function $g(n)$, we define $\Theta(g(n))$ to be a set of functions:

$$\Theta(g(n)) = \{f(n) \quad | \quad \text{there exist strictly positive constants } c_1, c_2, \text{ and } n_0 \text{ such that } 0 \leq c_1 g(n) \leq f(n) \leq c_2 g(n) \text{ for all } n \geq n_0\}.$$

Let's dissect this definition. Intuitively, what we want to say with "$f(n) \in \Theta(g(n))$" is that $f(n)$ and $g(n)$ are *asymptotically equivalent*. This means that they have essentially the same growth rates for large n. For example, functions like $4n^2$, $(8n^2+2n-3)$, $(n^2/5+\sqrt{n}-10\log n)$, and $n(n-3)$ are all intuitively asymptotically equivalent, since as n becomes large, the dominant (fastest growing) term is some constant times n^2. In other words, they all grow *quadratically* in n. The portion of the definition that allows us to select c_1 and c_2 is essentially saying "the constants do not matter because you may pick c_1 and c_2 however you like to satisfy these conditions." The portion of the definition that allows us to select n_0 is essentially saying "we are only interested in large n, since you only have to satisfy the condition for all n bigger than n_0, and you may make n_0 as big a constant as you like."

An example: Consider the function $f(n) = 8n^2 + 2n - 3$. Our informal rule of keeping the largest term and throwing away the constants suggests that $f(n) \in \Theta(n^2)$ (since f grows quadratically). Let's see why the formal definition bears out this informal observation.

We need to show two things: first, that $f(n)$ does grows asymptotically at least as fast as n^2, and second, that $f(n)$ grows no faster asymptotically than n^2. We'll do both very carefully.

Lower bound: $f(n)$ grows asymptotically at least as fast as n^2: This is established by the portion of the definition that reads: (paraphrasing): "there exist positive constants c_1 and n_0, such that $f(n) \geq c_1 n^2$ for all $n \geq n_0$." Consider the following (almost correct) reasoning:

$$f(n) = 8n^2 + 2n - 3 \geq 8n^2 - 3 = 7n^2 + (n^2 - 3) \geq 7n^2 = 7n^2.$$

Thus, if we set $c_1 = 7$, then we are done. But in the above reasoning we have implicitly made the assumptions that $2n \geq 0$ and $n^2 - 3 \geq 0$. These are not true for all n, but they are true for all sufficiently large n. In particular, if $n \geq \sqrt{3}$, then both are true. So let us select $n_0 \geq \sqrt{3}$, and now we have $f(n) \geq c_1 n^2$, for all $n \geq n_0$, which is what we need.

Upper bound: $f(n)$ grows asymptotically no faster than n^2: This is established by the portion of the definition that reads "there exist positive constants c_2 and n_0, such that $f(n) \leq c_2 n^2$ for all $n \geq n_0$." Consider the following reasoning (which is almost correct):

$$f(n) = 8n^2 + 2n - 3 \leq 8n^2 + 2n \leq 8n^2 + 2n^2 = 10n^2.$$

This means that if we let $c_2 = 10$, then we are done. We have implicitly made the assumption that $2n \leq 2n^2$. This is not true for all n, but it is true for all $n \geq 1$. So, let us select $n_0 \geq 1$, and now we have $f(n) \leq c_2 n^2$ for all $n \geq n_0$, which is what we need.

From the lower bound, we have $n_0 \geq \sqrt{3}$ and from the upper bound we have $n_0 \geq 1$, and so combining these we let n_0 be the larger of the two: $n_0 = \sqrt{3}$. Thus, in conclusion, if we let $c_1 = 7$, $c_2 = 10$, and $n_0 = \sqrt{3}$, then we have

$$0 \leq c_1 g(n) \leq f(n) \leq c_2 g(n) \qquad \text{for all } n \geq n_0,$$

and this is exactly what the definition requires. Since we have shown (by construction) the existence of constants c_1, c_2, and n_0, we have established that $f(n) \in n^2$. (Whew! That was a lot more work than just the informal notion of throwing away constants and keeping the largest term, but it shows how this informal notion is implemented formally in the definition.)

Now let's show why $f(n)$ is not in some other asymptotic class. First, let's show that $f(n) \notin \Theta(n)$. If this were true, then we would have to satisfy both the upper and lower bounds. It turns out that the lower bound is satisfied (because $f(n)$ grows at least as fast asymptotically as n). But the upper bound is false. In particular, the upper bound requires us to show "there exist positive constants c_2 and n_0, such that $f(n) \leq c_2 n$ for all $n \geq n_0$." Informally, we know that as n becomes large enough $f(n) = 8n^2 + 2n - 3$ will eventually exceed $c_2 n$ no matter how large we make c_2 (since $f(n)$ is growing quadratically and $c_2 n$ is only growing linearly). To show this formally, suppose towards a contradiction that constants c_2 and n_0 did exist, such that $8n^2 + 2n - 3 \leq c_2 n$ for all $n \geq n_0$. Since this is true for all sufficiently large n then it must be true in the limit as n tends to infinity. If we divide both side by n we have:

$$\lim_{n \to \infty} \left(8n + 2 - \frac{3}{n} \right) \leq c_2.$$

It is easy to see that in the limit the left side tends to ∞, and so no matter how large c_2 is, this statement is violated. This means that $f(n) \notin \Theta(n)$.

Let's show that $f(n) \notin \Theta(n^3)$. Here the idea will be to violate the lower bound: "there exist positive constants c_1 and n_0, such that $f(n) \geq c_1 n^3$ for all $n \geq n_0$." Informally this is true because $f(n)$ is growing quadratically, and eventually any cubic function will exceed it. To show this formally, suppose towards a contradiction that constants c_1 and n_0 did exist, such that $8n^2 + 2n - 3 \geq c_1 n^3$ for all $n \geq n_0$. Since this is true for all sufficiently large n then it must be true in the limit as n tends to infinity. If we divide both side by n^3 we have:

$$\lim_{n \to \infty} \left(\frac{8}{n} + \frac{2}{n^2} - \frac{3}{n^3} \right) \geq c_1.$$

It is easy to see that in the limit the left side tends to 0, and so the only way to satisfy this requirement is to set $c_1 = 0$, but by hypothesis c_1 is positive. This means that $f(n) \notin \Theta(n^3)$.

O-notation and Ω-notation: We have seen that the definition of Θ-notation relies on proving both a lower and upper asymptotic bound. Sometimes we are only interested in proving one bound or the other. The O-notation allows us to state asymptotic upper bounds and the Ω-notation allows us to state asymptotic lower bounds.

Definition: Given any function $g(n)$,

$$O(g(n)) = \{f(n) \mid \text{there exist positive constants } c \text{ and } n_0 \text{ such that } 0 \leq f(n) \leq cg(n) \text{ for all } n \geq n_0\}.$$

Definition: Given any function $g(n)$,

$$\Omega(g(n)) = \{f(n) \mid \text{there exist positive constants } c \text{ and } n_0 \text{ such that } 0 \leq cg(n) \leq f(n) \text{ for all } n \geq n_0\}.$$

Compare this with the definition of Θ. You will see that O-notation only enforces the upper bound of the Θ definition, and Ω-notation only enforces the lower bound. Also observe that $f(n) \in \Theta(g(n))$ if and only if $f(n) \in O(g(n))$ and $f(n) \in \Omega(g(n))$. Intuitively, $f(n) \in O(g(n))$ means that $f(n)$ grows asymptotically at the same rate or slower than $g(n)$. Whereas, $f(n) \in O(g(n))$ means that $f(n)$ grows asymptotically at the same rate or faster than $g(n)$.

For example $f(n) = 3n^2 + 4n \in \Theta(n^2)$ but it is not in $\Theta(n)$ or $\Theta(n^3)$. But $f(n) \in O(n^2)$ and in $O(n^3)$ but not in $O(n)$. Finally, $f(n) \in \Omega(n^2)$ and in $\Omega(n)$ but not in $\Omega(n^3)$.

The Limit Rule for Θ: The previous examples which used limits suggest alternative way of showing that $f(n) \in \Theta(g(n))$.

Limit Rule for Θ-notation: Given positive functions $f(n)$ and $g(n)$, if

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

for some constant $c > 0$ (strictly positive but not infinity), then $f(n) \in \Theta(g(n))$.

Limit Rule for O-notation: Given positive functions $f(n)$ and $g(n)$, if

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

for some constant $c \geq 0$ (nonnegative but not infinite), then $f(n) \in O(g(n))$.

Limit Rule for Ω-notation: Given positive functions $f(n)$ and $g(n)$, if

$$\lim_{n\to\infty} \frac{f(n)}{g(n)} \neq 0$$

(either a strictly positive constant or infinity) then $f(n) \in \Omega(g(n))$.

This limit rule can be applied in almost every instance (that I know of) where the formal definition can be used, and it is almost always easier to apply than the formal definition. The only exceptions that I know of are strange instances where the limit does not exist (e.g. $f(n) = n^{(1+\sin n)}$). But since most running times are fairly well-behaved functions this is rarely a problem.

You may recall the important rules from calculus for evaluating limits. (If not, dredge out your old calculus book to remember.) Most of the rules are pretty self evident (e.g., the limit of a finite sum is the sum of the individual limits). One important rule to remember is the following:

L'Hôpital's rule: If $f(n)$ and $g(n)$ both approach 0 or both approach ∞ in the limit, then

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

where $f'(n)$ and $g'(n)$ denote the derivatives of f and g relative to n.

Polynomial Functions: Using the Limit Rule it is quite easy to analyze polynomial functions.

Lemma: Let $f(n) = 2n^4 - 5n^3 - 2n^2 + 4n - 7$. Then $f(n) \in \Theta(n^4)$.

Proof: This would be quite tedious to do by the formal definition. Using the limit rule we have:

$$\lim_{n\to\infty} \frac{f(n)}{n^4} = \lim_{n\to\infty} \left(2 - \frac{5}{n} - \frac{2}{n^2} + \frac{4}{n^3} - \frac{7}{n^4}\right) = 2 - 0 - 0 + 0 - 0 = 2.$$

Since 2 is a strictly positive constant it follows from the limit rule that $f(n) \in \Theta(n^2)$.

In fact, it is easy to generalize this to arbitrary polynomials.

Theorem: Consider any asymptotically positive polynomial of degree $p(n) = \sum_{i=0}^{d} a_i n^i$, where $a_d > 0$. Then $p(n) \in \Theta(n^d)$.

From this, the informal rule of “keep the largest term and throw away the constant factors” is now much more evident.

Exponentials and Logarithms: Exponentials and logarithms are very important in analyzing algorithms. The following are nice to keep in mind. The terminology $\lg^b n$ means $(\lg n)^b$.

Lemma: Given any positive constants $a > 1$, b, and c:

$$\lim_{n\to\infty} \frac{n^b}{a^n} = 0 \qquad \lim_{n\to\infty} \frac{\lg^b n}{n^c} = 0.$$

We won’t prove these, but they can be shown by taking appropriate powers, and then applying L’Hôpital’s rule. The important bottom line is that polynomials always grow more slowly than exponentials whose base is greater than 1. For example:

$$n^{500} \in O(2^n).$$

For this reason, we will try to avoid exponential running times at all costs. Conversely, logarithmic powers (sometimes called *polylogarithmic functions*) grow more slowly than any polynomial. For example:

$$\lg^{500} n \in O(n).$$

For this reason, we will usually be happy to allow any number of additional logarithmic factors, if it means avoiding any additional powers of n.

At this point, it should be mentioned that these last observations are really asymptotic results. They are true in the limit for large n, but you should be careful just how high the crossover point is. For example, by my calculations, $\lg^{500} n \leq n$ only for $n > 2^{6000}$ (which is much larger than input size you’ll ever see). Thus, you should take this with a grain of salt. But, for small powers of logarithms, this applies to all reasonably large input sizes. For example $\lg^2 n \leq n$ for all $n \geq 16$.

Asymptotic Intuition: To get a intuitive feeling for what common asymptotic running times map into in terms of practical usage, here is a little list.

- $\Theta(1)$: Constant time; you can’t beat it!
- $\Theta(\log n)$: This is typically the speed that most efficient data structures operate in for a single access. (E.g., inserting a key into a balanced binary tree.) Also it is the time to find an object in a sorted list of length n by binary search.
- $\Theta(n)$: This is about the fastest that an algorithm can run, given that you need $\Theta(n)$ time just to read in all the data.
- $\Theta(n \log n)$: This is the running time of the best sorting algorithms. Since many problems require sorting the inputs, this is still considered quite efficient.
- $\Theta(n^2), \Theta(n^3), \ldots$.: Polynomial time. These running times are acceptable either when the exponent is small or when the data size is not too large (e.g. $n \leq 1,000$).
- $\Theta(2^n), \Theta(3^n)$: Exponential time. This is only acceptable when either (1) your know that you inputs will be of very small size (e.g. $n \leq 50$), or (2) you know that this is a worst-case running time that will rarely occur in practical instances. In case (2), it would be a good idea to try to get a more accurate average case analysis.
- $\Theta(n!), \Theta(n^n)$: Acceptable only for really small inputs (e.g. $n \leq 20$).

Are their even bigger functions. You betcha! For example, if you want to see a function that grows inconceivably fast, look up the definition of Ackerman’s function in our book.