Rough COMP4600 notes

Brendan McKay
Research School of Computer Science
Australian National University

July 16, 2014

Definitions of asymptotic notations

Definitions of the basic asymptotic notations vary between sources. Here are the preferred definitions for COMP4600.

In all cases, we say that something is true *"if n is sufficiently large"* if there is some constant n_0 such that it is true whenever $n \geq n_0$.

Assume $f(\,)$ and $g(\,)$ are functions from the positive integers to the real numbers and $g(n) > 0$ for sufficiently large n.

"$\boldsymbol{f(n) = O(g(n))}$" means:

For *some* constant $c > 0$, $|f(n)| \leq c\,g(n)$ if n is sufficiently large.

"$\boldsymbol{f(n) = \Omega(g(n))}$" means:

For *some* constant $c > 0$, $f(n) \geq c\,g(n)$ if n is sufficiently large.

"$\boldsymbol{f(n) = \Theta(g(n))}$" means:

For *some* constants $0 < c_1 < c_2$, $c_1\,g(n) \leq f(n) \leq c_2\,g(n)$ if n is sufficiently large.

"$\boldsymbol{f(n) = o(g(n))}$" means:

For *any* constant $c > 0$, $|f(n)| \leq c\,g(n)$ if n is sufficiently large.

This is equivalent to:

$$\frac{f(n)}{g(n)} \to 0 \text{ as } n \to \infty.$$

Simplifying expressions

In simplifying expressions in terms of asymptotic notation, the most important thing is to know the relative growth rate of different classes of functions. The main classes, listed in decreasing order of growth rates are:

- **Factorial functions.** Stirling's formula tells that $n! = \Theta\big(n^{n+1/2}e^{-n}\big)$.
- **Increasing exponential functions.** These are functions like a^n for $a > 1$ and e^{cn} for $c > 0$. Note that these examples are the same, since $e^{cn} = (e^c)^n$ and $a^n = e^{(\log a)n}$.

  Relative growth rates are elementary: a^n grows faster than b^n if $a > b$.
- **Positive powers.** These are functions like n^c for $c > 0$. Relative growth rates are elementary: n^c grows faster than n^d if $c > d$.
- **Increasing logarithmic functions.** These are functions like $\big(\log n)^c$ for $c > 0$. It doesn't matter what the base of the logarithm is, since $\log_a n = \log n / \log a$. That is, logarithms to different bases only differ from each other by a constant.
- **Iterated logarithmic functions.** These are functions like $\log\log n$.
- **Constant functions.** Like $f(n) = 3$.
- **Decreasing logarithmic functions.** These are functions like $\big(\log n)^c$ for $c < 0$.
- **Negative powers.** These are functions like n^c for $c < 0$.
- **Increasing exponential functions.** These are functions like a^n for $0 < a < 1$ and e^{cn} for $c < 0$.

There are plenty of other classes. Consider 2^{n^2} and $e^{\sqrt{n}}$.

To simplify a sum, reject the terms that grow slower than the largest terms. Eg, $2e^{n/3} + n^4 + \log n = \Theta(e^{n/3})$ since the first term grows faster than the others (and the constant 2 has no effect inside the Θ function).

A common case is a polynomial: $5n^6 - 4n^4 + 3n^3 + 17 = \Theta(n^6)$. However, beware in case the largest terms cancel or approximately cancel: $(n+1)^7 - n^7 = \Theta(n^6)$, since the terms with n^7 cancel out when the function is expanded.

Some cancellations are trickier: $\sqrt{n+1} - \sqrt{n} = \Theta(n^{-1/2})$, $\log(n+1) - \log n = \Theta(n^{-1})$, and $\log(2n) - \log n = \log 2 = \Theta(1)$.

To simplify quotients, first simply both the numerator and denominator using Θ.

$$\frac{e^n + n^9}{\log n - 1} = \frac{\Theta(e^n)}{\Theta(\log n)} = \Theta(e^n / \log n).$$

Caution about adding too many terms

If we have some functions $f_i(n)$ such that $f_i(n) = O(g(n))$ for each i, then the sum of a constant number of them is also $O(g(n))$. For example, $f_1(n) + f_2(n) + f_3(n) = O(g(n))$.

This is not true if a non-bounded number of functions are added. Define

$$F(n) = f_1(n) + f_2(n) + \cdots + f_n(n).$$

Then it is not (in general) true that $F(n) = O(g(n))$, or even that $F(n) = O(ng(n))$. Two examples:

(i) If $f_i(n) = i^2 n$, then $f_i(n) = O(n)$ for each i, but $F(n) = \Theta(n^4)$.

(ii) Define $f_i(n) = i^2$ when $n \le i$, and $f_i(n) = 0$ if $n > i$. Note that for each i, $f_i(n)$ is exactly zero for large n. However, $F(n) = n^2$.

This problem does not occur if the functions $f_i(n)$ are all $O(g(n))$ with the same constants c and n_0. This is called *uniformity.*

Estimating summations

Summations can be arbitrarily difficult, but in this course we only need some simple cases of summations with positive terms. We will consider when the terms are increasing, but if they are decreasing just turn the sum around backwards.

Slowly increasing terms

By "slowly increasing" we mean increasing but not faster than some positive power of the index. Examples:

$$\sum_{k=1}^{n} k^8, \quad \sum_{k=1}^{n} \sqrt{k}\, \log k, \quad \sum_{k=1}^{n} \frac{(2k+1)^5 \log k}{k+3}.$$

In all such cases, the sum is $\Theta(nT)$ where T is the largest term and n is the number of terms. This is an upper bound, since all sums are at most equal to the largest term times the number of terms. It is a lower bound since the sum is at least equal to the sum of the largest half of the terms, which is at least equal to $n/2$ times the middle term. The "slowly increasing" property ensures that the middle term is only a constant different from the largest term.

The answers are:

$$\Theta(n^9), \quad \Theta(n^{3/2} \log n), \quad \Theta(n^5 \log n).$$

Exponentially increasing terms

This is when the terms (or most of the large terms) grow by at least a constant at each step. Examples:

$$\sum_{k=1}^{n} 3^k, \quad \sum_{k=1}^{n} e^{k/2} \log k, \quad \sum_{k=1}^{n} \frac{2^k}{k^2+1}.$$

In the first case, each term is 3 times larger than the previous term. In the second case, each term is at least $e^{1/2}$ times larger than the previous term.

The third case is slightly trickier, since the term initially decrease. However, the 2^k eventually overwhelms the k^2+1 and the terms start to increase. A little work shows that starting at the third term, each term is at least 20/17 times as large as the previous term. The first two terms contribute only a constant to the answer, so we can ignore them.

Note that we require the ratios to be bounded above 1, not just greater than 1. That is, there needs to be a constant $c > 1$ such that the growth from one term to the next is at least c.

In such cases, the value of the sum is $\Theta(T)$ where T is the largest term. Obviously it is at least as large as the largest term (we are assuming all the terms are positive). The sum is at most a constant times the largest term on account of the sum

$$\sum_{k=0}^{\infty} c^k = \frac{1}{1-c}, \quad \text{if } 0 < c < 1.$$

Other cases

The only other case which is commonplace in computer science is the harmonic sum:

$$\sum_{k=1}^{n} \frac{1}{k} = \Theta(\log n).$$

It can be proved by approximating the sum by an integral.

Solving a recursion by guessing+induction

Suppose we have to solve a recurrence, such as

$$T(n) = 5T(n/3) + \Theta(n^3),$$

and we don't want to use the Master Theorem.

In a real-life problem, the term $5T(n/3)$ will probably be some combination of $T(\lfloor n/3 \rfloor)$ and $T(\lceil n/3 \rceil)$, but we will ignore this subtlety until later.

Guessing the solution

We might be clever enough to guess the solution just by looking at the recurrence. If so, proceed to the next section. If we aren't clever enough, one way to guess is by repeated application of the recurrence to itself. Since this is just a guessing calculation, we will write the term $\Theta(n^3)$ in the recurrence as cn^3.

$$\begin{aligned}
T(n) &= 5T(n/3) + cn^3 \\
&= 5\big(5T(n/3^2) + c(n/3)^3\big) + cn^3 \\
&= 5^2T(n/3^2) + (1 + 5/3^3)cn^3 \\
&= 5^2\big(5T(n/3^3) + c(n/3^2)^3\big) + (1 + 5/3^3)cn^3 \\
&= 5^3T(n/3^3) + \big(1 + 5/3^3 + (5/3^3)^2\big)cn^3,
\end{aligned}$$

and so on. Evidently the general form is

$$T(n) = 5^kT(n/3^k) + \big(1 + 5/3^3 + (5/3^3)^2 + \cdots + (5/3^3)^{k-1}\big)cn^3$$

for any k. Also note that

$$1 + 5/3^3 + (5/3^3)^2 + \cdots = \Theta(1)$$

since $5/3^3 < 1$, so for any k we have

$$T(n) = 5^kT(n/3^k) + \Theta(n^3).$$

Now choose k so that $n/3^k$ is tiny, say about 1 or 2. This happens when $k \approx \log_3 n$. In that case $5^k \approx 5^{\log_3 n} = n^{\log_3 5} \approx n^{1.465}$ (see note at the end). Also $T(n/3^k) = \Theta(1)$, since we chose k to make $n/3^k$ into a tiny number. So finally we have

$$T(n) = \Theta(n^{1.465}) + \Theta(n^3) = \Theta(n^3).$$

Recall that we were only doing a rough calculation to guess what the answer might be. Our guess is $T(n) = \Theta(n^3)$.

Proving the guess by induction

Next we use induction to formally prove the guess is correct. Upper and lower bounds must be done separately. Let's start with the upper bound: we will prove $T(n) \le cn^3$ for some c.

Induction has to get started with small values, so the first thing is note that $T(n) \le cn^3$ is true for tiny n (say $1 \le n \le 3$) if c is large enough.

Next assume that $T(m) \le cm^3$ for all $m < n$. Let d be some constant so that the $\Theta(n^3)$ term in the recurrence is at most dn^3. Now apply the recurrence:

$$\begin{aligned}
T(n) &\le 5T(n/3) + dn^3 \\
&\le 5c(n/3)^3 + dn^3, \quad \text{by the induction hypothesis} \\
&= \Big(\frac{5}{27}c + d\Big)n^3 \\
&\le cn^3, \quad \text{if } c \text{ is large enough (for example if } c \ge 2d).
\end{aligned}$$

We have found that the induction gets started at small values if c is large enough, and the induction step works if c is large enough. So the whole induction proof works if c is large enough; that is, we proved that always $T(n) \le cn^3$ for some c.

Proving the lower bound is almost the same, but must be done separately since the constants are different.

Start by noting $T(n) \geq c'n^3$ for tiny n if c' is *small* enough. Then assume the $\Theta(n^3)$ term in the recurrence is at least $d'n^3$ for some $d' > 0$. The induction step becomes

$$\begin{aligned} T(n) &\geq 5T(n/3) + d'n^3 \\ &\geq 5c'(n/3)^3 + d'n^3, \quad \text{by the induction hypothesis} \\ &= \Big(\frac{5}{27}c' + d'\Big)n^3 \\ &\geq c'n^3, \quad \text{if } c' \text{ is small enough (for example if } c' \leq d'\text{).} \end{aligned}$$

So the whole proof works fine if c' is small enough.

Trickier induction proofs

Sometimes the proofs by induction don't work exactly as above. Consider the example

$$T(n) = 9T(n/3) + \Theta(n).$$

By repeated application we can guess that the solution is $T(n) = \Theta(n^2)$. Let's try to prove the upper bound

$$T(n) \leq cn^2$$

for some c using the induction hypothesis that $T(m) \leq cm^2$ for all $m < n$. As before, let $d > 0$ be some constant so that the $\Theta(n)$ term in the recurrence is at most dn. Now apply the recurrence:

$$\begin{aligned} T(n) &\leq 9T(n/3) + dn \\ &\leq 9c(n/3)^2 + dn, \quad \text{by the induction hypothesis} \\ &= cn^2 + dn. \end{aligned}$$

The problem is that the last line is not less than cn^2 no matter how large c is. The proof has failed.

The solution in such cases is to change the induction hypothesis by subtracting a lower order term. Let's try the induction hypothesis $T(m) \leq cm^2 - dm$, where d is the same value as before. It is still true for tiny m when c is large enough.

$$\begin{aligned} T(n) &\leq 9T(n/3) + dn \\ &\leq 9\big(c(n/3)^2 - d(n/3)\big) + dn, \quad \text{by the induction hypothesis} \\ &= cn^2 - 2dn \\ &\leq cn^2 - dn. \end{aligned}$$

Now the induction step has succeeded. Incidentally, I subtracted dm in the induction hypothesis after some trial and error; the idea is to subtract something smaller than the main term and adjust it until the proof works.

The truth about floor and ceiling

Suppose the real recurrence is actually

$$T(n) = 3T(\lfloor n/3 \rfloor) + 2T(\lceil n/3 \rceil) + \Theta(n^3)$$

and we want to be super-finicky about the rigour of the proof.

Nearly always, the same guess will still work. The thing to note is that $\lfloor n/3 \rfloor$ and $\lceil n/3 \rceil$ are not much different from $n/3$. In fact,

$$\frac{n-2}{3} \le \left\lfloor \frac{n}{3} \right\rfloor \le \frac{n}{3} \le \left\lceil \frac{n}{3} \right\rceil \le \frac{n+2}{3}.$$

The induction step for the upper bound now becomes:

$$\begin{aligned} T(n) &\le 3T(\lfloor n/3 \rfloor) + 2T(\lceil n/3 \rceil) + dn^3 \\ &\le 3c(n/3)^3 + 2c((n+2)/3)^3 + dn^3, \quad \text{by the induction hypothesis} \\ &= \left(\frac{5}{27}c + d\right)n^3 + \frac{4cn^2}{9} + \frac{8cn}{9} + \frac{16c}{27}. \end{aligned}$$

Now we can assume $n \ge 3$ since $n \le 2$ can taken care of by the initial starting step of the induction. For $n \ge 3$ and sufficiently large c (for example, $c \ge 2d$), the expression on the right is smaller than cn^3, which completes the induction step.

Note on logarithms

A useful fact to remember is that $x^{\log y} = y^{\log x}$ for all $x, y > 0$, regardless of the base of the logarithm. For example, we used it to write $5^{\log_3 n} = n^{\log_3 5}$.