IDMA Course Material

– Some Notes on Functions and Asymptotic Analysis –

1 Functions

In this section, we recall the concept of **functions** and introduce some relevant functions for the course.

- A **power function** is of the form
$$f(x) = x^a$$
where the exponent a is a constant. The expression is well-defined for all $x \neq 0$ when $a \in \{1, 2, 3, \dots\}$ but it is sometimes necessary to restrict to $x > 0$ for other values of a (for instance, for $x^{1/2}$, which we recall is more often denoted $\sqrt{x}$).

- An **exponential function** is of the form
$$f(x) = b^x$$
where the base $b > 0$ is a constant. The expression is well-defined for all $x \in \mathbb{R}$.

- A **logarithmic function** is of the form
$$f(x) = \log_b(x)$$
where the base $b > 1$ is a constant. The expression is well-defined for all $x > 0$. Logarithmic functions are the inverse functions of exponential functions with the same base. Thus,
$$b^x = y \Longleftrightarrow \log_b(y) = x \; .$$

Theorem 1 (Properties of logarithms)**.** *For all $a, b, c > 0$ and all $r \in \mathbb{R}$ we have*

$$\log_c(ab) = \log_c(a) + \log_c(b) \tag{1}$$
$$\log_b(a^r) = r \log_b(a) \tag{2}$$
$$\log_b(a) = \frac{\log_c(a)}{\log_c(b)} \tag{3}$$

where, in each equation above, logarithm bases are not 1.

Proof sketch. The easiest way to prove this kind of statements (and to avoid having to memorize them) is to just go back to the definition. For instance, since by definition $a = c^{\log_c(a)}$, $b = c^{\log_c(b)}$, and $ab = c^{\log_c(ab)}$, just playing around with these expressions we see that

$$c^{\log_c(ab)} = a \cdot b = c^{\log_c(a)} \cdot c^{\log_c(b)} = c^{\log_c(a)+\log_c(b)} ,$$

and hence $\log_c(ab) = \log_c(a) + \log_c(b)$ as claimed. The other properties are proven in a very similar way. $\square$

- The **absolute value** is given by

$$|x| = \begin{cases} x & \text{if } x \geq 0 \\ -x & \text{if } x < 0 \end{cases}$$

- **Floor** is the function $\lfloor x \rfloor$, which rounds a real number x to the largest integer less than or equal to x. Thus,

$$\left\lfloor \frac{1}{2} \right\rfloor = 0 \qquad \left\lfloor -\frac{1}{2} \right\rfloor = -1 \qquad \lfloor \pi \rfloor = 3 \qquad \lfloor 7 \rfloor = 7$$

  We can define the **ceiling** function in a similar way; $\lceil x \rceil$ is the smallest integer larger than or equal to x.

In computer science, it is often convenient to combine other functions with the floor and ceiling functions. In order to perform calculations in such situations, it is advantageous to notice that

$$\lfloor x \rfloor = n \iff n \leq x < n+1$$

and

$$\lceil x \rceil = n \iff n-1 < x \leq n.$$

Example 2. We have that

$$\lceil \log_2 x \rceil = n$$

when

$$n-1 < \log_2 x \leq n,$$

which gives

$$2^{n-1} < 2^{\log_2 x} \leq 2^n$$

or just

$$2^{n-1} < x \leq 2^n$$

Thus, we see that $\lceil \log_2 x \rceil = n$ exactly when $2^{n-1} < x \leq 2^n$.

2 Asymptotic Growth of Functions

2.1 Collection of Definitions

In this section we collect the definitions regarding asymptotic growth of functions. Further explanations are available in the CLRS textbook.

Definition 3. We say that a function $f : \mathbb{R}^+ \to \mathbb{R}$ is *asymptotically positive* if there exists some $x_0 \in \mathbb{R}^+$ such that for all $x \geq x_0$ it holds that $f(x) > 0$.

We will also apply the above definition for functions that are defined on some subset of the positive reals. A common choice of such a subset will be positive integers $\mathbb{Z}^+ = \{1, 2, 3, \ldots\}$ or natural numbers $\mathbb{N} = \{0, 1, 2, \ldots\}$. (We note in passing that sometimes the notation $\mathbb{N}^+$ is used to mean the same as $\mathbb{Z}^+$—hopefully this should not be too confusing.)

Definition 4 (Asymptotic notation)**.** Let f and g be asymptotically positive functions.

- We say that *$f(x)$ is $O(g(x))$* if there exists a constant $c > 0$ and x_0 such that
$$f(x) \leq cg(x)$$
for all $x \geq x_0$.
- We say that *$f(x)$ is $\Theta(g(x))$* if f is $O(g(x))$ and $g(x)$ is $O(f(x))$.
- We say that *$f(x)$ is $o(g(x))$* if for any constant $c > 0$ we can find x_0 such that
$$f(x) < cg(x)$$
for all $x \geq x_0$.

Asymptotic notation is a very, very convenient tool, that we will need to get well acquainted with. At the end of the day, though, it is important to understand that it is no mysterious mathematical voodoo, but just a convenient way of expressing common-sense facts. As we have discussed in class, a good way to think about this is that we want to *focus only on the highest-order term of the function, and shave off the constant factor in front of this term.* This description is, of course, not the formal definition, but it is the intuition what the formal definition tries to make precise.

As an example, suppose we have two functions $f(n) = 2n^3 - 10n^2 - 5n$ and $g(n) = n^3 + 8n^2$. In this case, the notation $f(n) = \Theta(g(n))$ is just a way to say that when n gets large enough, then these two functions behave essentially in the same way except for constant factors (namely, they will be scaling like n^3 within a constant factor of 2). Note that this matches our intuitive understanding: the highest-order term in $f(n)$ with the constant factor shaved off is n^3, and for $g(n)$ the highest-order term with constant factor shaved off is also n^3, so asymptotically the two functions are the same. This is what $f(n) = \Theta(g(n))$ means.

As another example, if $h(n) = 1000n^2 + 1000000n$, then $h(n) = o(f(n))$ (for our function $f(n) = 2n^3 - 10n^2 - 5n$ above) is just convenient notation for the fact that although $h(n)$ will be much larger than $f(n)$ for small values of n, as n goes to infinity $h(n)$ will become vanishingly small compared to $f(n)$.

One can use the definition of big-O to show that big-Θ can equivalently be defined in the following manner.

Definition 5 (Second definition of big-Θ)**.** Let f and g be asymptotically positive functions. We say that $f(x)$ *is* $\Theta(g(x))$ if there exist constants $c_1, c_2 > 0$ and x_0 such that

$$c_1 g(x) \leq f(x) \leq c_2 g(x)$$

for all $x \geq x_0$.

For purposes of intuition, it can be useful to think *informally* of the above defined asymptotic notions as being analogous to comparison of numbers. Specifically,

$$\begin{array}{lll} f(x) \text{ is } O(g(x)) & \text{is like} & \text{“}f \leq g\text{”} \\ f(x) \text{ is } o(g(x)) & \text{is like} & \text{“}f < g\text{”} \\ f(x) \text{ is } \Theta(g(x)) & \text{is like} & \text{“}f = g\text{”} \end{array}$$

One must be careful and only use the above analogy to build intuition as some properties that hold for comparison of numbers do not carry over for functions. For instance, if a and b are numbers, then we have that $a \leq b$ or $b \leq a$. For functions, however, we can have a situation where f is not $O(g)$ and g is not $O(f)$. Can you think of such an example?

Both little-o and big-O give upper bounds. Intuitively, little-o gives a strict upper bound while big-O gives an upper bound that is potentially not strict. More formally, we have the following.

Theorem 6. *Let $f, g : \mathbb{R}^+ \to \mathbb{R}$ be asymptotically positive functions such that $f(x)$ is $o(g(x))$. Then we have that*

1. *$f(x)$ is $O(g(x))$ and*

2. *$g(x)$ is* not *$O(f(x))$.*

2.2 Collection of Rules

Theorem 7. *Let $f, g, h : \mathbb{R}^+ \to \mathbb{R}$ be asymptotically positive functions.*

Rules for big-O:

(B1) **Addition:** *If $f(x)$ and $g(x)$ are both $O(h(x))$ then $f(x)+g(x)$ is $O(h(x))$.*[1]

(B2) If $c > 0$ is a constant then c is $O(\log_a(x))$ for all $a > 1$.

(B3) $\log_a(x)$ is $O(x^b)$ for all $a > 1$ and $b > 0$.

(B4) x^a is $O(x^b)$ for $a \leq b$.

(B5) a^x is $O(b^x)$ for $0 < a \leq b$.

(B6) x^a is $O(b^x)$ for all a and all $b > 1$.

Rules for little-o:

(L1) **Linear combination:** *If $f(x)$ is $o(g(x))$ then $c_1 g(x)+c_2 f(x)$ is $\Theta(g(x))$, where $c_1 > 0$ and $c_2 \in \mathbb{R}$ are constants.*

(L2) If $c > 0$ is a constant then c is $o(\log_a(x))$ for all $a > 1$.

(L3) $\log_a(x)$ is $o(x^b)$ for all $a > 1$ and $b > 0$.

(L4) x^a is $o(x^b)$ if $a < b$.

(L5) a^x is $o(b^x)$ if $0 < a < b$.

(L6) x^a is $o(b^x)$ for all a and all $b > 1$.

Mixed rules:

(M1) **Transitivity:**
If $f(x)$ is $O(g(x))$ and $g(x)$ is $O(h(x))$ then $f(x)$ is $O(h(x))$.
If $f(x)$ is $o(g(x))$ and $g(x)$ is $o(h(x))$ then $f(x)$ is $o(h(x))$.
If $f(x)$ is $\Theta(g(x))$ and $g(x)$ is $\Theta(h(x))$ then $f(x)$ is $\Theta(h(x))$.

[1]This addition rule has to be applied with care in order not to get absurd results. For instance, by carelessly using it over and over again we can derive the amazing "fact" that

$$n = \underbrace{1+1+\cdots+1}_{n \text{ times}} = O(1) ,$$

since certainly $1 = O(1)$, and then it also holds that $1 + 1 = O(1)$, et cetera. The mistake here is that the addition rule can only be used a constant number of times in any asymptotic analysis. This will be important later for, e.g., the asymptotic analysis of recursive algorithms. If in doubt, just go back to the formal definition.

(M2) **Constant:**
If $c > 0$ is a constant then $cf(x)$ is $O(f(x))$.
If $c > 0$ is a constant then $cf(x)$ is $\Theta(f(x))$.

(M3) **Multiplication:**
If $f(x)$ is $O(g(x))$, then $h(x)f(x)$ is $O(h(x)g(x))$.
If $f(x)$ is $o(g(x))$, then $h(x)f(x)$ is $o(h(x)g(x))$.
If $f(x)$ is $\Theta(g(x))$, then $h(x)f(x)$ is $\Theta(h(x)g(x))$.

(M4) **Chaining:**
If $f(x)$ is $o(g(x))$ and $g(x)$ is $O(h(x))$ then $f(x)$ is $o(h(x))$.
If $f(x)$ is $O(g(x))$ and $g(x)$ is $o(h(x))$ then $f(x)$ is $o(h(x))$.
If $f(x)$ is $o(g(x))$ and $g(x)$ is $\Theta(h(x))$ then $f(x)$ is $o(h(x))$.
If $f(x)$ is $\Theta(g(x))$ and $g(x)$ is $o(h(x))$ then $f(x)$ is $o(h(x))$.

Example 8. Let us use the rules above to show that

$$f(x) = \frac{1}{2}x^3 + \frac{1}{2}x^2$$

and

$$g(x) = x^3$$

have the same asymptotic order (in other words, f is $\Theta(x^3)$). First, note that by *(L4)* we have that x^2 is $o(x^3)$. Then, using *(L1)*, we conclude that $\frac{1}{2}x^3 + \frac{1}{2}x^2$ is $\Theta(x^3)$ thus showing that f and g have the same asymptotic order.

Example 9. All logarithms have the same asymptotic order. This is seen from the general rule

$$\log_b(x) = \frac{\log_a(x)}{\log_a(b)},$$

which shows that two logarithms are equal up to multiplication with a constant. It then follows from (*M2*) that all logarithms have the same asymptotic order.

Example 9 tells us that we do not need to worry about the base of logarithms when dealing with asymptotic growth. We will therefore simply write $\log x$ when the base is irrelevant. Another comment in this context is that for a computer scientist (at least for one of the theoretical sort) the most natural base for logarithms is 2, so, e.g., in more advanced textbooks or research papers log denotes logarithms to the base 2 unless otherwise stated.

3 Sequences

We will think of a **sequence** as an infinite, linearly ordered collection of numbers. An example would be

$$1, 4, 9, 16, 25, 36, 49, 64, 81, 100, \ldots \tag{4}$$

or

$$0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, \ldots \tag{5}$$

We will often use the notation (a_n) to describe the elements of a sequence and refer to a_n as the nth element of the sequence. The number n is an index and we often start a sequence at index 0 or 1.

We can succinctly specify a sequence via a function f that is defined on, say, the positive integers $\{1, 2, 3, 4 \ldots\}$. In that case, we can write an explicit expression for a sequence by setting

$$a_n = f(n).$$

For example, the sequence in (4) is defined by the function $f(x) = x^2$ and we would write the sequence as

$$b_n = n^2 \text{ for } n \geq 1.$$

In a similar way, we can explicitly specify sequence (5) as follows:

$$c_n = n - 2\left\lfloor \frac{n}{2} \right\rfloor \text{ for } n \geq 0.$$

We can also define sequences **recursively**. A recursive definition works by explicitly specifying some of the first elements in a sequence and then defining the rest by referring back to previous elements one or more indices away. One example would be to define

$$\begin{aligned} c_0 &= 0 \\ c_1 &= 1 \\ c_n &= c_{n-2} \text{ for } n \geq 2 \end{aligned}$$

The above recovers the sequence (5) since we can use the rule $c_n = c_{n-2}$ to convince ourselves that c_2 must be equal to $c_0 = 0$ and that c_3 must be equal to $c_1 = 1$ etc.

Example 10. We define a sequence f_n by setting $f_0 = 1$ and

$$f_n = nf_{n-1} \text{ for } n \geq 1.$$

The first elements of the sequence are

$$1, 1, 2, 6, 24, 120, 720, 5040, \ldots$$

This sequence is referred to as the **factorial sequence** and we usually write f_n as $n!$ (which is pronounced "n-factorial"). Note that

$$n! = 1 \cdot 2 \cdot 3 \cdots n.$$

A special class of recursively defined sequences consists of **series** (sums of sequences). A series is defined from some sequence (a_n) by summing its elements. If n_0 is the first index of a sequence (a_n), we define the series (s_n) recursively as

$$\begin{aligned} s_{n_0} &= a_{n_0} \\ s_n &= s_{n-1} + a_n \text{ for } n > n_0 \end{aligned}$$

We will often write

$$s_n = a_{n_0} + a_{n_0+1} + \cdots + a_n$$

or

$$s_n = \sum_{k=n_0}^{n} a_k$$

to define a series.

A **summation formula** is the explicit expression of a series.

Example 11. Let (a_n) be the constant sequence 1, i.e. $a_n = 1$, $n \geq 1$. The series of (a_n) is then given by the sequence

$$1, 2, 3, 4, 5, 6, 7, 8, 9, \ldots$$

and we have obtained our first summation formula:

$$\sum_{k=1}^{n} 1 = n \tag{6}$$

The derivation and proof of summation formulas are very connected to the concept of **mathematical induction**, which we will return to later in the course. For now, we postulate some commonly used summation formulas without proofs:

Theorem 12.

$$\begin{aligned}
\sum_{k=1}^{n} k &= \frac{n^2+n}{2} \\
\sum_{k=1}^{n} k^2 &= \frac{2n^3+3n^2+n}{6} \\
\sum_{k=1}^{n} c^k &= \frac{c^{n+1}-c}{c-1}
\end{aligned}$$

The last formula is valid for all $c \neq 1$.

When $c = 1$ in the last summation formula, one can use formula from (6) instead.

We will get back to later how to prove expressions as in Theorem 12 formally, but note that already now you have enough information to establish that they are at least asymptotically correct, i.e., that $\sum_{k=1}^{n} k = \Theta(n^2)$, for instance. (Exercise: How can you prove this? What about the other expressions above?)

As we will discover in the exercises, one can often go a long way by combing the four summation formulas above with the general sum rules:

Theorem 13.

$$\begin{aligned}
\sum_{k=1}^{n}(a_k+b_k) &= \sum_{k=1}^{n} a_k + \sum_{k=1}^{n} b_k \\
\sum_{k=1}^{n} c a_k &= c\sum_{k=1}^{n} a_k \\
\sum_{k=1}^{n} a_k &= \sum_{k=1}^{m-1} a_k + \sum_{k=m}^{n} a_k
\end{aligned}$$

4 Asymptotic Growth of Sequences

We can treat the asymptotic behaviour of sequences completely analogously to functions by simply replacing the functions $f(x)$, $g(x)$ with sequences (a_n), (b_n) in Section 2 in all definitions and rules.

We are going to be particularly interested in situations where a sequence (a_n) describes the worst-case running time of an algorithm on an input of size n. It is very common that such sequences are of asymptotic order

$$\Theta(1), \Theta(\log n), \Theta(n), \Theta(n \log n), \Theta(n^2), \Theta(n^2 \log n), \Theta(n^3), \Theta(2^n)$$

(although this is of course not an exhaustive list of possibilities). During the course, we will repeatedly see that if one can replace an algorithm with another of lower asymptotic order, a significant improvement can be achieved.

An essential challenge when analyzing the efficiency of algorithms is to find the asymptotic order of recursively defined sequences and series, in particular. For now, let us only note (as we already touched on above) that our sum rules imply that $\sum_{k=1}^{n} k$ is of the same asymptotic order as n^2 and that $\sum_{k=1}^{n} k^2$ is of same asymptotic order as n^3. We will also make use of the following relation:

Theorem 14. $\log(n!)$ *is* $\Theta(n \log n)$

Proof. As mentioned previously, it is irrelevant what base we choose for the logarithm. We will therefore choose to use $\log_2$. We want to show that $\log_2(n!)$ is $O(n \log_2 n)$ and that $n \log_2 n$ is $O(\log_2(n!))$.

The first proof is easy since we have that

$$n! = 1 \cdot 2 \cdots n \leq n \cdot n \cdots n \leq n^n$$

and thus,

$$\log_2(n!) \leq \log_2(n^n) = n \log_2 n \ ,$$

where the inequality follows from the fact that the logarithm is an increasing function and the equality is a fundamental property of logarithms. This shows that $\log_2(n!)$ is $O(n \log_2 n)$ (with $c = 1$ in the definition).

For the second proof, we need to be a bit more resourceful. When $1 \leq i \leq n$, we have that $i - 1 \geq 0$ and $n - i \geq 0$. Thus,

$$(i-1)(n-i) \geq 0 \ .$$

Expanding, we get

$$i \cdot n - i^2 - n + i \geq 0$$

or

$$i \cdot n - i^2 + i \geq n \ ,$$

which can be written as

$$i(n - i + 1) \geq n \ .$$

This means that all products of the form $1 \cdot n$, $2 \cdot (n-1), 3 \cdot (n-2), \ldots, n \cdot 1$ are greater than or equal to n. Using this fact, and playing around with the laws of logarithms, we obtain

$$\begin{aligned}
n\log_2 n &= \log_2 n + \log_2 n + \cdots + \log_2 n \\
&\leq \log_2(1\cdot n) + \log_2(2\cdot(n-1)) + \cdots + \log_2(n\cdot 1) \\
&= [\log_2 1 + \log_2 n] + [\log_2 2 + \log_2(n-1)] + \cdots + [\log_2 n + \log_2 1] \\
&= 2\log_2 1 + 2\log_2 2 + \cdots + 2\log_2 n \\
&= 2\sum_{i=1}^{n} \log_2 i \\
&= 2\log_2(1\cdot 2\cdots n) \\
&= 2\log_2(n!)\ ,
\end{aligned}$$

where we have rearranged the terms at the third equality sign but otherwise just used the standard rules for logarithms. This shows that $n\log_2 n$ is $O(\log_2(n!))$ (with $c = 2$ in the definition). $\square$

We end the notes with the following result that we will state without a proof:

Theorem 15. *The series*

$$\sum_{k=1}^{n} \frac{1}{k}$$

is $\Theta(\log n)$.

Credits: *This is a lightly edited version of notes prepared for the DMA 2019 course by Laura Mancinska, who was in turn inspired by notes written by Søren Eilers.*