

Caio Brighenti
COSC 302 - Analysis of Algorithms – Spring 2019
Assignment 1

1. Simply put, an algorithm is a way to solve a problem. In other words, it is a specific, well-defined step-by-step process to solve not just a single problem, but a category of similar problems. Algorithms must take some form of input, and must correctly use this input to solve its defined category of problems.
2. An example of a problem that cannot be solved is an algorithm that for any given input problem could output an efficient algorithmic solution to it. A similar scenario is NP-complete problems – neither an algorithm nor a human has determined whether there are efficient solutions to this class of problems. There are plenty of problems for which we do not know efficient solutions, such as the Halting problem. An example of a non-algorithmic solution an algorithmically solvable problem would be the tallest student problem. We could simply look at the class and see quickly which student is the tallest, without systematically going one by one and comparing to the currently tallest – we have an innate sense of things like this, and can process multiple pieces of information simultaneously, something a computer cannot do.
3. The book defines analyzing an algorithm as predicting the resources that the algorithm requires, such as memory space, communication bandwidth, or disk storage. Most commonly, it refers to analyzing the rate of growth of the algorithm's runtime, also known as time complexity.
4. Each of the following are things that can be simplified when analyzing an algorithm
 - 1 - Not specifying which case we are referring to and assuming worst
 - 2 - Assuming each line is a single step
 - 3 - Assuming the running time does not depend on the hardware
 - 4 - Considering only the runtime with respect to the input and not other variables
 - 5 - Assuming that the biggest term in the running time dominates
 - 6 - Assuming that each step takes the same amount of time
 - 7 - Assuming that functions with the same dominating term are in the same category
 - 8 - Ignoring the coefficients of the highest ordered term
 - 9 - Not considering the actual runtime in a unit of time
 - 10 - Assuming that the underlying hardware will execute a process step-by-step and not run in parallel
 - 11 - Not clarifying if the complexity is different for the best case
 - 12 - Not calculating the precise total of computer steps but just the order of growth
 - 13 - Ignoring OS overhead like fetching things in memory or context swapping
 - 14 - Considering only the time complexity and not the space complexity
5. The running time of an algorithm is a function that counts the number of steps the algorithm must execute as a function of the size of the input. If we're asked to calculate, we must ask whether we want the worst case, best case, or average case because there are different. If none of these are specified, then by convention we look at the worst case of the algorithm.
6. The order of growth results from a simplification of the runtime. The order of growth relates to how the running time grows as a function of n , but we simplify and keep only the largest, dominating term, and we place all functions with the same simplified order of growth into a category. We can say that running time relates to an algorithm, while order of growth represents a category of algorithms.

7. We typically use big-O more than other asymptotic notations for two reasons. Firstly, it provides us with information about the upper bound of the algorithm, meaning the maximum time it can take to run. This information is more important than the lower bound, as we would like to know what the slowest our algorithm can run is. Secondly, it is easier to determine big-O than Theta, as it provides information about only one bound, not both the upper and lower. Thus, big-O is used for its generality – it is less specific than the other notations but provides us with the information we need to know.
8. We say that an efficient algorithm is anything better than an exponential runtime, so typically polynomial or better. Of course, this measure is subjective, and it is possible that although we have an efficient solution for a problem, we might not have the most efficient solution.
9. Not necessarily. If $f(n)$ is equal to $g(n)$, then the first part of the claim would be true, but $f(n)$ would not be asymptotically smaller than $g(n)$.
10. Show by definition

$$(a) \ f(n) \in \omega(g(n)) \implies f(n) \in \Omega(g(n))$$

By definition, if we have $f(n) \in \omega(g(n))$ then we have:
 $g(n) \in o(f(n))$, and consequently:
 $0 \leq c_1 g(n) < f(n)$

The definition of $f(n) \in \Omega(g(n))$ is $0 \leq c_1 g(n) \leq f(n)$. Since for any (x, y) , $x < y \implies x \leq y$, then if we have:

$$0 \leq c_1 g(n) < f(n)$$

we must also have:

$$0 \leq c_1 g(n) \leq f(n)$$

Thus, the definition $f(n) \in \Omega(g(n))$ is met, and so the claim must be true.

$$(b) \ f(n) \in \Theta(g(n)) \iff g(n) \in \Omega(f(n))$$

If we have $f(n) \in \Theta(g(n))$, then by definition we have:

$$0 \leq c_1 g(n) \leq f(n) \leq c_2 g(n)$$

We then rearrange and combine each side of the inequality as follows. First, the left side:

$$0 \leq c_1 g(n) \leq f(n) \quad \text{definition} \quad (1)$$

$$0 \leq g(n) \leq \frac{1}{c_1} f(n) \quad \text{divide by } c_1 \quad (2)$$

Then, the right side:

$$f(n) \leq c_2 g(n) \quad \text{definition} \quad (3)$$

$$\frac{1}{c_2} f(n) \leq g(n) \quad \text{divide by } c_2 \quad (4)$$

Combining these two inequalities, we have:

$$0 \leq \frac{1}{c_2} f(n) \leq g(n) \leq \frac{1}{c_1} f(n)$$

As $\frac{1}{c_1}$ and $\frac{1}{c_2}$ are both constants, then this meets the definition of $g(n) \in \Theta(f(n))$. Thus, $f(n) \in \Theta(g(n)) \implies g(n) \in \Theta(f(n))$ for all $n > n_0$.

Finally, we must show that this is two sided (\iff , not \implies). This is done simply by swapping $f(n)$ with $g(n)$ in the proof above. We arbitrarily selected $f(n) \in \Omega(g(n))$ as the starting point, and the math is identical if we do the opposite. Thus, the claim must be true.

11. Using limit prove $2^n \in o(n!)$

We use Stirling's approximation to rewrite the second term such that:

$$\lim_{n \rightarrow \infty} \frac{2^n}{n!} = \lim_{n \rightarrow \infty} \frac{2^n}{\sqrt{2\pi n} \left(\frac{n}{e}\right)^n} \quad (5)$$

$$= \lim_{n \rightarrow \infty} \left(\frac{2e}{n}\right)^n \frac{1}{\sqrt{2\pi n}} \quad (6)$$

As the right term in the limit is a fraction with a constant numerator and a denominator that goes to ∞ , then the term goes to 0 and can be eliminated. We then consider the left term. First, we let $y = \left(\frac{2e}{n}\right)^n$. Then,

$$\ln(y) = n \ln\left(\frac{2e}{n}\right) \quad (7)$$

$$\lim_{n \rightarrow \infty} \ln(y) = \lim_{n \rightarrow \infty} n \ln\left(\frac{2e}{n}\right) \quad (8)$$

$$= \lim_{n \rightarrow \infty} \frac{\ln \frac{2e}{n}}{\frac{1}{n}} \quad (9)$$

Now we consider a variable x such that $x = \frac{1}{n}$. Since we are taking the limit of n with respect to ∞ , then x will be going to 0. Thus, we now have a limit as x goes to 0.

$$\lim_{x \rightarrow 0} \frac{\ln 2ex}{x} = \quad (10)$$

$$= \frac{-\infty}{0} \quad (11)$$

$$= -\infty \quad (12)$$

This is because $\ln(x)$ approaches $-\infty$ as x approaches 0. Thus, we have:

$$\lim_{n \rightarrow \infty} \ln y = \lim_{n \rightarrow \infty} \frac{\ln \frac{2e}{n}}{\frac{1}{n}} = \lim_{x \rightarrow 0} \frac{\ln 2ex}{x} = -\infty \quad (13)$$

$$\lim_{n \rightarrow \infty} \ln y = -\infty \quad (14)$$

Finally, we use the property:

$$\lim_{n \rightarrow \infty} \ln y = a \implies \lim_{n \rightarrow \infty} y = e^a$$

to determine $\lim_{n \rightarrow \infty} y$, where $y = \left(\frac{2e}{n}\right)^n$ and $a = -\infty$.

$$\lim_{n \rightarrow \infty} y = e^a \quad (15)$$

$$= e^{-\infty} \quad (16)$$

$$= 0 \quad (17)$$

Finally, let us rewrite the original full limit and plug in the separate limit terms calculated.

$$\lim_{n \rightarrow \infty} \frac{2^n}{n!} = \lim_{n \rightarrow \infty} \left(\frac{2e}{n}\right)^n \frac{1}{\sqrt{2\pi n}} \quad (18)$$

$$= 0 \cdot 0 \quad (19)$$

$$\lim_{n \rightarrow \infty} \frac{2^n}{n!} = 0 \quad (20)$$

Thus, we have that $2^n \in o(n!)$.

12. Prove $f(n) \in o(g(n)) \implies f(n) + g(n) \in \Theta(g(n))$

Using the definitions of $o(n)$ and $\Theta(n)$ we can rewrite the claim as follows:

$$0 \leq f(n) < cg(n) \implies 0 \leq c_1g(n) \leq f(n) + g(n) \leq c_2g(n)$$

For all $n > n_0$, and where c , c_1 and c_2 are positive constants.

We start by assuming the left side of the claim is true. Thus, since $f(n) < cg(n)$, then it must be that $cg(n) + f(n) \leq 2g(n)$, and that $g(n) \leq g(n) + f(n)$. Thus, we have:

$$c(g(n)) + f(n) \leq 2g(n) \leq 2(g(n) + f(n)) \leq 4g(n) \quad (21)$$

$$2g(n) \leq 2(g(n) + f(n)) \leq 4g(n) \quad \text{eliminate left-most term} \quad (22)$$

$$1g(n) \leq g(n) + f(n) \leq 2g(n) \quad \text{divide by 2} \quad (23)$$

Thus, we have that $0 \leq c_1g(n) \leq f(n) + g(n) \leq c_2g(n)$, where $c_1 = 1$ and $c_2 = 2$, for all $n > n_0$. Therefore, the claim must be true.

Then, use this property find the asymptotic relation between:

$$f(n) = n + \log n \text{ and } g(n) = 100n + \log^2 n$$

First, we find the relationship between each of the terms in $f(n)$ and $g(n)$. We do this by first plotting n vs $\log(n)$, and then $100n$ versus $\log^2 n$. The plots are included below:



On the left is the plot of n versus $\log(n)$, and n clearly dominates asymptotically. Similarly, on the right $100n$ dominates $\log^2(n)$. Thus, using the property we just proved, it must be that:

$$f(n) \in \Theta(n)$$

and

$$g(n) \in \Theta(100n) = \Theta(n)$$

Thus, since both of the functions are linear, then it must be that $f(n) \in \Theta(g(n))$, or that $g(n) \in \Theta(f(n))$.

13. Show order of growth between n^3 and $\log n^{\log n}$

We proceed by solving the limit $L = \lim_{n \rightarrow \infty} \frac{n^3}{(\log n)^{\log n}}$

First, we define a variable y such that $y = \log n$. As $\log n$ will go to infinity as n goes to infinity, we can instead the limit as y approaches infinity. Additionally, if $y = \log n$, then we can rewrite n^3 as follows.

$$\log n = y \quad (24)$$

$$n = 2^y \quad (25)$$

$$n^3 = (2^y)^3 = 2^{3y} \quad (26)$$

Thus, we rewrite and solve the limit as follows.

$$L = \lim_{y \rightarrow \infty} \frac{2^{3y}}{y^y} \quad (27)$$

$$= \lim_{y \rightarrow \infty} \frac{8^y}{y^y} \quad (28)$$

$$= \lim_{y \rightarrow \infty} \left(\frac{8}{y}\right)^y \quad (29)$$

We now define a variable Y such that $Y = (\frac{8}{y})^y$. Thus, we have that $\ln Y = y \ln \frac{8}{y}$. We then take the limit with respect to $\ln Y$ as y approaches infinity.

$$\lim_{y \rightarrow \infty} \ln Y = \lim_{y \rightarrow \infty} y \ln \frac{8}{y} \quad (30)$$

$$= \lim_{y \rightarrow \infty} \frac{\ln \frac{8}{y}}{\frac{1}{y}} \quad (31)$$

$$= \frac{-\infty}{0} \quad (32)$$

$$= -\infty \quad (33)$$

This is because $\ln(x)$ goes to negative infinity as x approaches 0, and both $\frac{8}{y}$ and $\frac{1}{y}$ will approach 0, as in both cases the numerator is a constant and the denominator goes to infinity.

Finally, we use the property:

$$\lim_{n \rightarrow \infty} \ln y = a \implies \lim_{n \rightarrow \infty} y = e^a$$

To determine the limit $\lim_{y \rightarrow \infty} Y$, where $a = -\infty$.

$$\lim_{y \rightarrow \infty} Y = e^a \quad (34)$$

$$= e^{-\infty} \quad (35)$$

$$= 0 \quad (36)$$

Thus, rewriting the original limit and combining with this result, we have the following.

$$\lim_{y \rightarrow \infty} \frac{n^3}{(\log n)^{\log n}} = \lim_{n \rightarrow \infty} (\frac{8}{y})^y = 0 \quad (37)$$

$$\lim_{y \rightarrow \infty} \frac{n^3}{(\log n)^{\log n}} = 0 \quad (38)$$

Thus, we have that $(\log n)^{\log n} \in \Theta(n^3)$.

14. Rank the following functions by order of growth

I accomplished this problem by first guessing a potential order based on intuition (for instance, n^2 obviously grows slower than 2^n). Then, I used a graphic calculator to plot each of the functions I thought to be close to each other, rearranging my initial based on the result. I plotted each function at least once, but I used the transitive property to assume that if $g_1 = \Omega(g_2)$ and $g_2 = \Omega(g_3)$ then $g_1 = \Omega(g_3)$. For instances where the graphs were unclear, I calculated the limit between the two functions. Eventually, I obtained the following order, from largest to smallest.

(a) 2^{2n+1}

(b) 2^{2n}

(c) $(n+1)!$

(d) $n!$

(e) e^n

(f) $n2^n$

(g) 2^n

(h) $(\frac{3}{2})^n$

(i) $(\log n)^{\log n}, n^{\log \log n}$

(j) n^3

(k) $(\log n)!$

(l) $n^2, 4^{\log n}$

- (m) $n \log n, \log(n!)$
- (n) $n, 2^{\log n}$
- (o) $\sqrt{2}^{\log n}$
- (p) $(\log n)^2$
- (q) $2^{\sqrt{2 \log n}}$
- (r) $\ln n$
- (s) $\sqrt{\log n}$
- (t) $\ln \ln n$
- (u) $n^{\frac{1}{\log n}}, 1$