Fourth laboratory.

Iván Alejandro Cruz Tole.

08.10.2018

The following consists in two parts. Part one is made up of some exercises from the third edition of *Introduction to Algorithms*, by Cormen, Leiserson, Rivest and Stein, while the second part's exercises are taken from the 2006's *Algorithms* (Dasgupta, Papadimitriou and Vazirani) edition.

1 Part one.

1.1 Suppose we are comparing implementations of insertion sort and merge sort on the same machine. For inputs of size n, insertion sort runs in $8n^2$ steps, while merge sort runs in $64n$ lg n steps. For which values of n does insertion sort beat merge sort?

The inequality to work with is: $8n^2 < 64n \lg n$, which in turn leads to: $1 < \lg n^{\frac{8}{n}}$, which is true for $n \leq 43$. This is more or less clear in observance of the graph of $n^{\frac{8}{n}}$, for it shows that the function of n's value is bigger than 2 for those values of n - i.e. The logarithm's value is bigger than 1 for those values of n.

1.2 What is the smallest value of n such that an algorithm whose running time is $100n^2$ runs faster than an algorithm whose running time is 2^n on the same machine?

The inequality $100n^2 < 100n^2$ is unsatisfied for $n = 14$, and since both functions at both sides of the inequality are increasing, it is unsatisfied for every smaller value of n. Yet, one can satisfy it by making $n = 15$, so that's the answer.

1.3 For each function $f(n)$ and time t in the following table, determine the largest size n of a problem that can be solved in time t, assuming that the algorithm to solve the problem takes $f(n)$ microseconds.

	1 Second	1 Minute	1 Hour	1 Day	1 Month	1 Year	1 Century
$\lg n$	$2^{1\times10^6}$	$2^{6\times10^7}$	$2^{3.6\times10^9}$	$2^{8.64\times10^{10}}$	$2^{2.592\times10^{12}}$	$2^{3.1536\times10^{13}}$	$2^{3.15576\times10^{15}}$
$\sqrt{n}$	1×10^{12}	3.6×10^{15}	1.29×10^{19}	7.46×10^{21}	6.72×10^{24}	9.95×10^{26}	9.96×10^{30}
n	1×10^{6}	6×10^{7}	3.6×10^{9}	8.64×10^{10}	2.59×10^{12}	3.15×10^{13}	3.16×10^{15}
$n\lg n$	62746	2801417	133378058	2755147513	71870856404	797633893349	6.86×10^{13}
n^2	1000	7745	60000	293938	1609968	5615692	56176151
n^3	100	391	1532	4420	13736	31593	146679
2^n	19	25	31	36	41	44	51
$n!$	9	11	12	13	15	16	17

1.4 Let $p(n) = \sum_{i=0}^{d} a_i n^i$ where $a_d > 0$, be a degree-d polynomial in n,and let k be a constant. Use the definitions of the asymptotic notations to prove the following properties:

a. If $k \geq d$ then $p(n) = O(n^k)$.

b. If $k \leq d$ then $p(n) = \Omega(n^k)$.

c. If $k = d$ then $p(n) = \Theta(n^k)$.

d. If $k > d$ then $p(n) = o(n^k)$.

e. If $k < d$ then $p(n) = \omega(n^k)$.

Proposition (a) is false: The "biggest" possible term in the summation is $a_k n^k$, and such expression's most significant term (recall that a_k is a degree-k polynomial in n) is cn^{k^2}, for some $c \neq 0$, which is, in turn, $\Theta(n^{2k})$. This allows us to assert that $p(n) = O(n^{2k})$.

For proposition (b), however, it is different: $p(n)$'s first term is, at least, $a_k n^k$, and since it is provided that the polynomial's value is always bigger than 0, $a_k > 0$, thus providing the existence of some c both smaller than a_k and bigger than 0, and thus providing that, from some n_0 on, $cn^k \leq a_k n^k \leq p(n)$.

Proposition (c) is, once again, false: If we have $p(n) = O(n^{2k})$ and $p(n) = \Omega(n^k)$ it doesn't follow that $p(n) = \Theta(n^k)$ - It is the contrary that follows, actually.

Proposition (d) is false, for reasons similar to those implicated in the proof of proposition (a)'s falsity.

Proposition (e) is true, for reasons similar to those related to proposition (b)'s veracity.

2 Part two.

2.1 In each of the following situations, indicate whether $f = O(g)$, or $f = \Omega(g)$, or both (in which case $f = \Theta(g)$). For each situation, take the first function as f and second as g.

a. $n - 100, n - 200$.

b. $n^{\frac{1}{2}}, n^{\frac{2}{3}}$.

c. $100n + \log n, n + (\log n)^2$.

d. $n \log n, 10n \log 10n$.

e. $\log 2n, \log 3n$.

f. $10 \log n, \log n^2$.

g. $n^{1.01}, n(\log n)^2$.

h. $\frac{n^2}{\log n}, n(\log n)^2$.

i. $n^{0.1}, (\log n)^{10}$.

j. $(\log n)^{\log n}, \frac{n}{\log n}$.

k. $\sqrt{n}, \log n^3$.

l. $n^{\frac{1}{2}}, 5^{\log_2 n}$.

m. $n2^n, 3^n$.

n. $2^n, 2^{n+1}$.

o. $n!, 2^n$.

p. $(\log n)^{\log n}, 2^{(\log_2 n)2}$.

q. $\sum_{i=1}^{n} i^k, n^{k+1}$.

Case (a): $f = \Theta(g)$. Case (b): $f = O(g)$. Case (c): $f = O(g)$. Case (d): $f = \Theta(g)$. Case (e): $f = \Theta(g)$. Case (f): $f = \Theta(g)$. Case (g): $f = O(g)$. Case (h): $f = \Omega(g)$. Case (i): $f = \Omega(g)$. Case (j): $f = \Omega(g)$. Case (k): $f = \Omega(g)$. Case (l): $f = O(g)$. Case (m): $f = O(g)$. Case (n): $f = \Theta(g)$. Case (o): $f = \Omega(g)$. Case (p): $f = \Omega(g)$. Case (q): $f = O(g)$.

2.2 Show that, if c is a positive real number, then $g(n) = 1 + c + c^2 + ... + c^n$ is:

a. $\Theta(1)$ **if** $c < 1$.

b. $\Theta(n)$ **if** $c = 1$.

c. $\Theta(c^n)$ **if** $c > 1$.

We can easily see how $g(n)$ is a geometric series, thus being easily solvable by means of a closed formula, namely $g(n) = \frac{c^{n+1}-1}{c-1}$. Now, if we're to go through this series' asymptotic behaviour, we're to take the following limit: $\lim_{n\to\infty} \frac{c^{n+1}-1}{c-1} = \frac{c^{\infty}-1}{c-1}$, and this last expression will behave differently depending on c's value, as we shall see:

a. If $c < 1$ then $\frac{c^{\infty}-1}{c-1}$ resolves to $\frac{0-1}{c-1} = 1$, for any number both positive and smaller than one will reduce to zero when raised to infinity. Hence, this series' value is precisely 1, which suffices for the series to be considered $\Theta(1)$.

b. If $c = 1$ then we have one of those ugly indeterminate forms to deal with: 1^{∞}. In consideration of both my health and yours, another approach proceeds: Notice that if we were to have $c = 1$ each of $g(n)$'s terms would

equal 1, so we'd have $g(n) = \sum_{i=0}^{n} c^i = \sum_{i=0}^{n} 1^i = \sum_{i=0}^{n} 1 = n + 1$, and from this the fact that $g(n) = \Theta(n)$ immediately follows.

c. If $c > 1$ then the arrive at a pretty boring conclusion: $g(n) = \infty$. This doesn't say much about the series' theta class, but there is still much to be said:

- $g(n) = c^n + g(n-1)$, $g(n)$ is an increasing function (for $c > 0$), and $g(0) = c^0 = 1$, so it will always hold that $g(n) > c^n$. Thus, $g(n) = \Omega(c^n)$.
- $c^{n+1} = g(n)(c-1) + 1$ (how so? Playing around with the previously introduced closed formula for the series), which in turn allows me to assert that $c^{n+1} > g(n)$ or $c^{n+1} > \sum_{i=0}^{n} c^i$, which are the same. From that last assertion, an essentially identical one: $c^n > \sum_{i=0}^{n-1} c^i$, from which follows that $c^n + c^n > \sum_{i=0}^{n-1} c^i + c^n$, i.e. $2c^n > g(n)$. Thus, $g(n) = O(c^n)$.

Finally, then: Since $g(n) = \Omega(c^n)$ and $g(n) = O(c^n)$, $g(n) = \Theta(c^n)$.

3 Extra: Solve $T(n) = 2T(n-2) + 2$, with $n = 2k$ and for $T(0) = 0$, and $T(0) = 1$ by Recursive Substitution and the Graphical Method.

Notice that $T(n) = T(2k) = 2T(2k-2) + 2$. Now, by means of substitution:

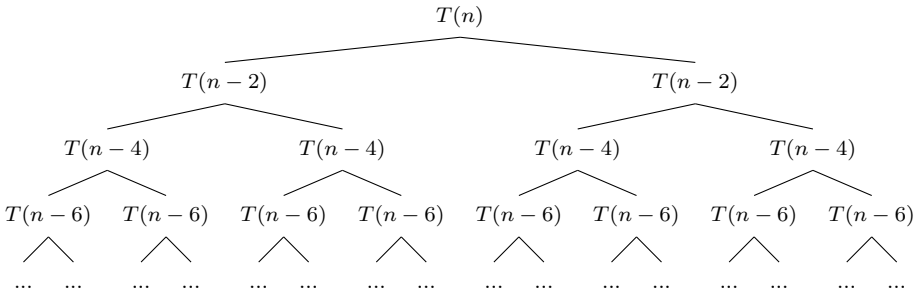
1. $T(2k) = 2T(2k-2) + 2$...
2. $T(2k) = 2[2T(2k-4) + 2] + 2 = 2[2T(2k-4)] + 4 + 2$...
3. $T(2k) = 2\{2[2T(2k-6) + 2]\} + 4 + 2 = 2\{2[2T(2k-6)]\} + 8 + 4 + 2$...

$\vdots$

k. $T(2k) = 2^k + 2^{k-1} ... + 2^3 + 2^2 + 2^1 = \sum_{i=1}^{k} 2^i = \sum_{i=0}^{k} 2^i - 2^0 = 2^{k+1} - 1 - 1$.

And that may be formally proved by induction. **Notice, though,** that the validity of the former conclusion hangs from the premise that $T(0) = 0$, for if it didn't, it wouldn't hold: We would have to add another term to the summation, and if it were that $T(0) = 1$ such term would be 2^k, and so our final answer would be $2^{k+1} + 2^k - 2$.

Another approach is the graphical one, through which we will arrive at the same conclusions. Consider the following tree:



Now, consider the following observations:

- Each inner node has a weight of two.
- Each leave has a weight of zero or one.
- Each level that is added down duplicates the amount of nodes the previous level had - e.g. level two has two nodes, while level one has one, and level three has four nodes, while level two has precisely two.
- The tree has exactly $k+1$ levels, for it goes as deep as to reach $T(0)$ nodes, which in turn will happen in k decrements of two units (remember that $n = 2k$).

And from those propositions we see how the net weight until the penultimate level is $\sum_{i=1}^{k} 2^i - 2^0 = 2^{k+1} - 1 - 1$, and the weight of the last level is either 0 or 2^k - Which is the number of leaves.