Algorithms: Design and Analysis - CS 412

Problem Set 02: Asymptotic Analysis

1. Let $f(n)$ and $g(n)$ be asymptotically non-negative functions. Using the basic definition of Θ-notation, prove that $max\{f(n), g(n)\} = \Theta(f(n) + g(n))$.

Proof. We can prove the above by showing that there exists constants $c_1, c_2, n_0 > 0$ such that $\forall n \geq n_0$, $0 \leq c_1(f(n) + g(n)) \leq max\{f(n), g(n)\} \leq c_2(f(n) + g(n))$.

As the functions are asymptotically non-negative, we can assume that for some $n_0 > 0$, $f(n) \geq 0$ and $g(n) \geq 0$. Therefore, $n \geq n_0$. Then

$$f(n) + g(n) \geq max\{f(n), g(n)\}$$

. Also, since $f(n) \leq max\{f(n), g(n)\}$, and $g(n) \leq max\{f(n), g(n)\}$,

$$f(n) + g(n) \leq 2max\{f(n), g(n)\}$$

$$\frac{1}{2}(f(n) + g(n)) \leq max\{f(n), g(n)\}$$

Then combining the above two inequalities, we get

$$0 \leq \frac{1}{2}(f(n) + g(n)) \leq max\{f(n), g(n)\} \leq f(n) + g(n) \text{ for } n \geq n_0$$

which follows from the definition of Θ-notation with $c_1 = \frac{1}{2}, c_2 = 1$.

Therefore, $max\{f(n), g(n)\} = \Theta(f(n) + g(n))$. $\square$

2. Prove or disprove the statements below.

(a) $(n + 1)^2 = n^2 + O(n)$

Expand. $(n + 1)^2 = n^2 + 2n + 1$.
Then $n^2 + 2n + 1 = n^2 + O(n)$ since $O(n)$ denotes a function which grows linearly with n, and $2n + 1$ is indeed $O(n)$. Hence proved that $(n + 1)^2 = n^2 + O(n)$.

(b) $(n+O(\sqrt(n)))(n+O(\log(n)))^2 = n^3 + O(\sqrt{n^5})$

Expand. $(n+O(\sqrt(n)))(n^2+O(n\log n)+O(\log^2 n))$
$= (n+O(\sqrt(n)))(n^2+O(n\log n))$
$= n^3 + O(n^{\frac{3}{2}}\sqrt{n}) + O(n^2\log n) + O(n^{\frac{3}{2}}\log n)$
$= n^3 + O(n^2\log n) + O(n^{\frac{3}{2}}\sqrt{n}) + O(n^{\frac{3}{2}}\log n)$
$= n^3 + O(n^{\frac{5}{2}}) + O(n^2\log n) + O(n^{\frac{3}{2}}\log n)$
Clearly, the dominating terms are n^3, followed by $O(n^{\frac{5}{2}})$, and $O(n^{\frac{5}{2}})$ reduces to $O(\sqrt{n^5})$.
Then we get: $n^3 + O(\sqrt(n^5))$ as the dominating terms. Hence proved that the statement is true.

(c) $\exp(O(1)) = O(e^n)$

The term $\exp(O(1))$ implies the constant e, which simplifies to $O(e)$. Obviously $O(e)$ is not equal to $O(e^n)$, since e^n is an exponential function, and e is a constant. Hence disproved.

(d) $n^{\log(n)} = O((\log n)^n)$

$O((\log n)^n) = O(n\log n)$. For large values of n, $n^{\log n}$ grows faster than $n\log n$. Hence disproved.

(e) $2^{2n} = O(2^n)$

Assume that c and n_0 are positive constants satisfying $0 \le 2^{2n} \le c2^n$ for all $n \ge n_0$. Then $2^{2n} = 2^n \cdot 2^n \le c2^n$, which implies that $c \ge 2^n$. The last inequality, however, does not hold regardless of a value we would choose for c, because 2^n becomes arbitrarily large as n gets large. Hence, disproved.

3. Prove that for any two functions $f(n)$ and $g(n)$, we have $f(n) = \Theta(g(n))$ if and only if $f(n) = O(g(n))$ and $f(n) = \Omega(g(n))$.

Proof. By the definition of Θ-notation, $f(n) = \Theta(g(n))$ whenever there exist positive constants c_1, c_2, and n_0 such that $0 \le c_1 g(n) \le f(n) \le c_2 g(n)$ for all $n \ge n_0$. This condition can be decomposed into the combination of inequalities $0 \le c_1 g(n) \le f(n)$ and $0 \le f(n) \le c_2 g(n)$, both holding for all $n \ge n_0$. From the former follows $f(n) = \Omega(g(n))$ and from the latter follows $f(n) = O(g(n))$.

For the proof of the opposite direction, suppose that $f(n) = \Omega(g(n))$ and $f(n) = O(g(n))$. The former means that there exist positive constants c_1 and n_1 such that $0 \le c_1 g(n) \le f(n)$ for all $n \ge n_1$, and the latter means that there exist positive constants c_2 and n_2 such that $0 \le f(n) \le c_2 g(n)$ for all $n \ge n_2$. Then, by letting $n_0 = \max\{n_1, n_2\}$ and merging both conditions, we get that $0 \le c_1 g(n) \le f(n) \le c_2 g(n)$ for all $n \ge n_0$. Thus, $f(n) = \Theta(g(n))$. $\square$

4. Prove that for $S \subseteq \mathbb{Z}$,

$$\sum_{k \in S} \Theta(f(k)) = \Theta\left(\sum_{k \in S} f(k)\right)$$

assuming both sums converge.

Proof. The question asks to prove that the sum of Θ of some function is equal to the Θ of the sum of that function.

Let $g_k = \Theta(f(k))$ for our ease, then by the definition, $\exists c_1, c_2, n_0 \; \forall n \geq n_0$ such that

$$\forall_{k \in S} \; g_k = \Theta(f(k)) \iff 0 \leq c_1 f_k \leq g_k \leq c_2 fk$$

.

Then we need to show that

$$\sum_{k \in S} g_k = \Theta\left(\sum_{k \in S} f(k)\right)$$

Then from the definition of Θ, we sum over k:

$$\begin{aligned}
\sum_{k \in S} a_k f_k \leq \sum_{k \in S} g_k \leq \sum_{k \in S} b_k f_k \\
\sum_{k \in S} min\{a_k\} f_k \leq \sum_{k \in S} g_k \leq \sum_{k \in S} max\{b_k\} f_k \\
min\{a_k\}\left(\sum_{k \in S} f_k\right) \leq \sum_{k \in S} g_k \leq max\{b_k\}\left(\sum_{k \in S} f_k\right) \; \forall_{n \geq n_0} \\
\implies \sum_{k \in S} g_k = \Theta\left(\sum_{k \in S} f(k)\right)
\end{aligned}$$

□

5. Is $2^{n+1} = O(2^n)$? Is $2^{2n} = O(2^n)$?

Let's determine positive constants c and n_0 such that $0 \leq 2^{n+1} \leq c2^n$ for all $n \geq n_0$. Since $2^{n+1} = 2 \cdot 2^n$, we can pick $c = 2$ and $n_0 = 1$. So $2^{n+1} = O(2^n)$.

Now let's assume that c and n_0 are positive constants satisfying $0 \leq 2^{2n} \leq c2^n$ for all $n \geq n_0$. Then $2^{2n} = 2^n \cdot 2^n \leq c2^n$, which implies that $c \geq 2^n$. The last inequality, however, does not hold regardless of a value we would choose for c, because 2^n becomes arbitrarily large as n gets large. Hence, $2^{2n} \neq O(2^n)$.