CS 341 Advanced Topics in Algorithms
Problem Sheet 0 (not for credit)

Source: Brassard & Bratley, Fundamentals of Algorithms, Prentice-Hall, 1996.

Problem 3.3. Consider two algorithms A and B that take time in $O(n^2)$ and $O(n^3)$, respectively. Could there exist an implementation of algorithm B that would be more efficient (in terms of computing time) than an implementation of algorithm A on *all* instances? Justify your answer.

Problem 3.4. What does $O(1)$ mean? $\Theta(1)$?

Problem 3.5. Which of the following statements are true? Prove your answers.

1. $n^2 \in O(n^3)$
2. $n^2 \in \Omega(n^3)$
3. $2^n \in \Theta(2^{n+1})$
4. $n! \in \Theta((n+1)!)$

Problem 3.6. Prove that if $f(n) \in O(n)$ then $[f(n)]^2 \in O(n^2)$.

Problem 3.7. In contrast with Problem 3.6, prove that $2^{f(n)} \in O(2^n)$ does not necessarily follow from $f(n) \in O(n)$.

Problem 3.8. Consider an algorithm that takes a time in $\Theta(n^{\lg 3})$ to solve instances of size n. Is it correct to say that it takes a time in $O(n^{1.59})$? In $\Omega(n^{1.59})$? In $\Theta(n^{1.59})$? Justify your answers. (Note: $\lg 3 \approx 1.58496\ldots$)

Problem 3.9. Prove that the O notation is reflexive: $f(n) \in O(f(n))$ for any function $f: \mathbb{N} \to \mathbb{R}^{\geq 0}$.

Problem 3.10. Prove that the O notation is transitive: it follows from

$$f(n) \in O(g(n)) \text{ and } g(n) \in O(h(n))$$

that $f(n) \in O(h(n))$ for any functions $f, g, h: \mathbb{N} \to \mathbb{R}^{\geq 0}$.

Problem 3.11. Prove that the ordering on functions induced by the O notation is not total: give explicitly two functions $f, g: \mathbb{N} \to \mathbb{R}^{\geq 0}$ such that $f(n) \notin O(g(n))$ and $g(n) \notin O(f(n))$. Prove your answer.

Problem 3.14. Let $f(n) = n^2$. Find the error in the following "proof" by mathematical induction that $f(n) \in O(n)$.

- *Basis*: The case $n = 1$ is trivially satisfied since $f(1) = 1 \leq cn$, where $c = 1$.
- *Induction step*: Consider any $n > 1$. Assume by the induction hypothesis the existence of a positive constant c such that $f(n-1) \leq c(n-1)$.

$$\begin{aligned} f(n) &= n^2 = (n-1)^2 + 2n - 1 = f(n-1) + 2n - 1 \\ &\leq c(n-1) + 2n - 1 = (c+2)n - c - 1 < (c+2)n \end{aligned}$$

Thus we have shown as required the existence of a constant $\hat{c} = c + 2$ such that $f(n) \leq \hat{c}n$. It follows by the principle of mathematical induction that $f(n)$ is bounded above by a constant times n for all $n \geq 1$ and therefore that $f(n) \in O(n)$ by definition of the O notation.

Problem 3.15. Find the error in the following "proof" that $O(n) = O(n^2)$. Let $f(n) = n^2$, $g(n) = n$ and $h(n) = g(n) - f(n)$. It is clear that $h(n) \leq g(n) \leq f(n)$ for all $n \geq 0$. Therefore, $f(n) = \max(f(n), h(n))$. Using the maximum rule, we conclude

$$O(g(n)) = O(f(n) + h(n)) = O(\max(f(n), h(n))) = O(f(n)).$$