COSC 302: Analysis of Algorithms — Spring 2018
Prof. Darren Strash
Colgate University

**Problem Set 2 — Order of Growth and Asymptotic Analysis**

**Due by 4:30pm Friday, Feb. 9, 2018 as a single pdf via Moodle (either generated via LaTeX, or concatenated photos of your work). Late assignments are not accepted.**

This is an *individual* assignment: collaboration (such as discussing problems and brainstorming ideas for solving them) on this assignment is highly encouraged, but the work you submit must be your own. Give information only as a tutor would: ask questions so that your classmate is able to figure out the answer for themselves. It is unacceptable to share any artifacts, such as code and/or write-ups for this assignment. If you work with someone in close collaboration, you must mention your collaborator on your assignment.

*Suggested practice problems (not to be turned in): 3.1-2, 3.1-3, 3.1-5, 3.2-1, 3.2-4, 3-3 (a)*

1. Describe a $\Theta(n \log n)$-time algorithm that, given an array $A$ of $n$ integers and another integer $x$, determines whether or not there exist two elements in $A$ whose sum is exactly $x$. (*Hint: do not use divide-and-conquer.*)

   Prove that your algorithm is correct by giving an invariant that holds throughout your algorithm, and arguing that it holds at initialization, throughout execution, and describe why this invariant implies correctness.

   **Solution:** Assume integers are given in an array $A[1..n]$. First sort using merge sort in $\Theta(n \lg n)$ time. Then for each index $i$ (from $i = 1$ to $n$), we compute $y = x - A[i]$, and perform a binary search for $y$ in $A[i+1..n]$. If we find $y$, then we have values $A[i]$ and $y$ that sum to $x$. Otherwise, we evaluate all $i$'s and do not find integers that sum to $x$.

   **Invariant:** For each integer $z$ in $A[1..i-1]$, there is no value $y \neq z$ in $A[1..n]$ such that $z + y = x$.

   **Initialization:** There is trivially no integer before $A[1]$; therefore there is no integer to sum to $x$ with any integer in $A[1..n]$.

   **Maintenance**: Suppose that before the $(i-1)$-th iteration of the **for** loop, that no integer in $A[1..i-2]$ sums to $x$ with any integer in $A[1..n]$. During the $(i-1)$-th iteration, if there is no integer $y = x - A[i-1]$ in $A[i..n]$, then no $z$ in $A[1..i-1]$ sums to $x$ with any $y$ in $A[i..n]$. Further, $A[i-1]$ does not sum up to $x$ with any value in $A[1..i-2]$ by our inductive assumption. Thus at the beginning of the next loop, no integer $z$ in $A[1..i-1]$ sums up to $x$ with any $y$ in $A[1..n]$.

   **Termination**: Suppose we reach the beginning of loop $i = n+1$. Then since (by maintenance) the invariant still holds, we have that there is no $y$ in $A[1..n]$ and $z \neq y$ in $A[1..n]$ such that $x = y + z$) and correctly return false. Otherwise, we return true when we find values $z = A[i]$ and $y$ in $A[i+1..n]$ such that $y + z = x$. Thus, the algorithm is correct at termination.

2. Problem 3.1-1 in CLRS.

**Solution:**

*Proof.* Since $f(n)$ is an asymptotically nonnegative function, by definition there exists some $n_1 > 0$ such that $f(n) \geq 0$ for all $n \geq n_1$. By the same token, for some $n_2 > 0$, $g(n) \geq 0$ for all $n \geq n_2$.

Note then that

$$\begin{aligned} 0 &\leq \frac{1}{2}(f(n) + g(n)) && \text{when } n \geq \max\{n_1, n_2\} (\text{since } f(n) \geq 0,\ g(n) \geq 0) \\ &\leq \max\{f(n), g(n)\} && \text{average of two values is at most the maximum value} \\ &\leq f(n) + g(n). \end{aligned}$$

Therefore, for $n_0 = \max\{n_1, n_2\}$, $c_1 = 1/2$ and $c_2 = 1$, we have that

$$0 \leq c_1(f(n) + g(n)) \leq \max\{f(n), g(n)\} \leq c_2(f(n) + g(n)),$$

and by definition $\max\{f(n), g(n)\} = \Theta(f(n) + g(n))$. □

3. Problem 3-2 (Big Oh, Big Omega, and Big Theta only) in CLRS.

| $A$ | $B$ | $O$ | $\Omega$ | $\Theta$ | Reason |
|---|---|---|---|---|---|
| $\lg^k n$ | $n^\epsilon$ | yes | no | no | Proved in worksheet 3, problem 7. |
| $n^k$ | $c^n$ | yes | no | no | Exponential function grows faster than any polynomial. Can prove by contradiction. |
| $\sqrt{n}$ | $n^{\sin n}$ | no | no | no | $n^{\sin n}$ alternates between $1/n$ and $n$ and is therefore neither upper bounded nor lower bounded by $\sqrt{n}$ |
| $2^n$ | $2^{n/2}$ | no | yes | no | $2^{n/2} = \sqrt{2^n}$, which grows strictly slower than $2^n$ |
| $n^{\lg c}$ | $c^{\lg n}$ | yes | yes | yes | $c^{\lg n} = c^{\frac{\log_c n}{\log_c 2}} = (c^{\log_c n})^{\frac{1}{\log_c 2}} = n^{\lg c}$ |
| $\lg(n!)$ | $\lg(n^n)$ | yes | yes | yes | Proved in worksheet 2, problem 3. |

4. Problem 3-4 (parts c,d,e,g) in CLRS.

**Solution:**

(a) Conjecture: $f(n) = O(g(n))$ implies $\lg(g(n)) = O(\lg(f(n)))$ where $\lg(g(n)) \geq 1$ and $f(n) \geq 1$ for all sufficiently large $n$.

**True:**

*Proof.* First note that, by definition of Big Oh, there exist positive constants $c_1$ and $n_1$ such that $0 \leq f(n) \leq c_1 g(n)$ for all $n \geq n_1$. We further note that

$$0 \leq f(n) \leq c_1 g(n) \leq \max\{1, c_1\} g(n)$$

for all $n \geq n_1$. (This is important to ensure $\lg(g(n))$ is defined later on.)

Furthermore, let $n_2$ be the value such that $\lg(g(n)) \geq 1$ and $f(n) \geq 1$ for all $n \geq n_2$. Then,

$$f(n) \leq \max\{1, c_1\}g(n) \text{ for } n \geq n_1,$$

and

$$\lg(f(n)) \leq \lg(\max\{1, c_1\}g(n)) \text{ for } n \geq \max\{n_1, n_2\},$$

where $n \geq n_2$ is required to ensure that $\log(f(n))$ and $\lg(\max\{1, c_1\}g(n))$ are defined. Finally, we have that

$$\begin{aligned}
\lg(f(n)) &\leq \lg(\max\{1, c_1\}g(n)) && \text{for } n \geq max\{n_1, n_2\} \\
&= \lg(\max\{1, c_1\}) + \lg(g(n)) \\
&\leq \lg(\max\{1, c_1\})\lg(g(n)) + \lg(g(n)) && (\text{since } \lg(g(n)) \geq 1 \text{ when } n \geq n_2) \\
&= (\lg(\max\{1, c_1\}) + 1)\lg(g(n)).
\end{aligned}$$

Therefore, for all $n_0 = \max\{n_1, n_2\}$ and $c = \max\{1, c_1\} + 1$, we have that $\lg(f(n)) \leq c\lg(g(n))$. Note also that since $f(n) \geq 1$ for all $n \geq n_2$, we also have that

$$\begin{aligned}
\lg(f(n)) &\geq \lg(1) && n \geq n_2 \\
&= 0,
\end{aligned}$$

which also holds true for all $n \geq \max\{n_1, n_2\} \geq n_2$.
Putting it all together, we have that $n_0 = \max\{n_1, n_2\}$ and $c = \lg(\max\{1, c_1\}) + 1$, we have that $0 \leq \lg(f(n)) \leq c\lg(g(n))$, and by definition $\lg(f(n)) = O(\lg(g(n)))$. □

(b) Conjecture: $f(n) = O(g(n))$ implies $2^{f(n)} = O(2^{g(n)})$.
**Sometimes True:** The conjecture is true for $f(n) = n$ and $g(n) = 2n$, since $2^n = O(2^{2n}) = O(4^n)$. However, it is false when $f(n) = 2n$ and $g(n) = n$, since $4^n \neq O(2^n)$.

(c) Conjecture: $f(n) = O(f(n)^2)$.
**Sometimes True:** The conjecture is true for $f(n) = n$, since $n = O(n^2)$. However, it is false when $f(n) = \frac{1}{n}$, since $\frac{1}{n} \neq O\left(\frac{1}{n^2}\right)$.

(d) Conjecture $f(n) = O(f(n/2))$.
**Sometimes True:** The conjecture is true for $f(n) = n$, since $n = O(n/2) = O(n)$. However, it is false when $f(n) = 2^n$, since $2^n \neq O(2^{n/2}) = O(\sqrt{2}^n)$.