Example Solutions to Exercises on Topics of ICS141

Kazuo Sugihara
Department of Information and Computer Sciences
University of Hawaii at Manoa
Honolulu, Hawaii, U.S.A.

August 15, 2013

Abstract

This handout presents example solutions to exercises on topics covered in ICS141 (such as proving techniques, mathematical induction, finite series and asymptotic notations). It does not intend to overview all topics addressed in ICS141. Instead, it aims at showing students examples for them learning (a) how to solve problems anticipated in theoretical computer science, (b) how to present solutions to the problems and (c) how to typeset the solutions by using LaTeX.

1 Review of Asymptotic Notations and Finite Series

There are four asymptotic notations discussed in ICS141 [1, 2].

(a) **Upper Bound:** "Big-Oh" notation $f(n) = O(g(n))$ or $f(n) \in O(g(n))$

(b) **Lower Bound:** "Big-Omega" notation $f(n) = \Omega(g(n))$ or $f(n) \in \Omega(g(n))$

(c) **Tight Bound:** "Theta" notation $f(n) = \Theta(g(n))$ or $f(n) \in \Theta(g(n))$

(d) **Dominated Upper Bound:** "little-oh" notation $f(n) = o(g(n))$ or $f(n) \in o(g(n))$

For a given algorithm, we always attempt to derive a tight bound in the Θ notation. A few examples of proving a tight bound are given below.

Definition 1. $f(n) = O(g(n))$ iff $\exists c \in \mathbb{R}\ \exists N \in \mathbb{Z}^+$ [$\forall n \in \mathbb{Z}^+$ [$n \geq N \rightarrow f(n) \leq c|g(n)|$]]

Definition 2. $f(n) = \Omega(g(n))$ iff $\exists c \in \mathbb{R}\ \exists N \in \mathbb{Z}^+$ [$\forall n \in \mathbb{Z}^+$ [$n \geq N \rightarrow c|g(n)| \leq f(n)$]]

Definition 3. $f(n) = \Theta(g(n))$ iff $f(n) = O(g(n)) \wedge f(n) = \Omega(g(n))$

By Definition 3, one way to prove a tight bound is to prove two assertions (i.e., an upper bound and a lower bound) separately. To prove each of the two bounds requires to give concrete values of the two constants c and N satisfying the corresponding inequality.

Definition 3 can be rephrased as follows.

Definition 4.
$f(n) = \Theta(g(n))$ iff $\exists c_1 \in \mathbb{R}\ \exists c_2 \in \mathbb{R}\ \exists N \in \mathbb{Z}^+$ [$\forall n \in \mathbb{Z}^+$ [$n \geq N \rightarrow c_1|g(n)| \leq f(n) \leq c_2|g(n)|$]]

This definition suggests another way to prove a tight bound that is to give concrete values of the three constants c_1, c_2 and N satisfying the two inequalities in Definition 4.

Ex.1 *Prove or disprove* $2x^2 + x - 7 = \Theta(x^2)$.
(Section 3.2 Ex.30 in p.217 of the Rosen's textbook [1] for ICS141)

Theorem 1. $2x^2 + x - 7 = \Theta(x^2)$

Proof. Let $c_1 = 2$, $c_2 = 3$ and $N = 7$.
$c_1|x^2| = 2x^2 \leq 2x^2 + x - 7$ for all $x \geq N$ since $7 \leq x$ for all $x \geq N$
$2x^2 + x - 7 < 2x^2 + x < 2x^2 + x^2 = 3x^2 = c_2x^2$ for all $x \geq N$ □

Ex.2 *Prove or disprove* $x \log x = \Theta(x^2)$.

Theorem 2. $x \log x = \Theta(x^2)$ *is false.*

Proof. We disprove the tight bound by proving $x \log x = o(x^2)$.
$$\lim_{x\to\infty} \frac{x \log x}{x^2}$$
$$= \lim_{x\to\infty} \frac{\log x}{x}$$
$$= \lim_{x\to\infty} \frac{\frac{d}{dx} \log x}{\frac{d}{dx} x} \qquad \text{by L'Hôspital's Rule}$$
$$= \lim_{x\to\infty} \frac{\frac{1}{x}}{1} \qquad \text{since } \frac{d}{dx} \log x = \frac{1}{x}$$
$$= \lim_{x\to\infty} \frac{1}{x}$$
$$= 0$$
Thus, by the definition of the little-oh notation, $x \log x = o(x^2)$ holds. Since $f(n) = \Omega(g(n))$ and $f(n) = o(g(n))$ cannot hold simultaneously, the tight bound $x \log x = \Theta(x^2)$ does not hold. □

Ex.3 *Prove or disprove* $\sum_{i=1}^{n} i(i-1) = \Theta(n^3)$.

Theorem 3. $\sum_{i=1}^{n} i(i-1) = \Theta(n^3)$

Proof. We first simplify the summation.

$$\sum_{i=1}^{n} i(i-1)$$
$$= \sum_{i=1}^{n} (i^2 - i)$$
$$= \sum_{i=1}^{n} i^2 - \sum_{i=1}^{n} i$$
$$= \frac{n(n+1)(2n+1)}{6} - \frac{n(n+1)}{2}$$

by closed formulas for the finite series in Section 2.4 Table 2 (p.166) of the Rosen's Textbook [3]

$$= \frac{2n^3 + 3n^2 + n - 3(n^2+n)}{6}$$
$$= \frac{2n^3 - 2n}{6}$$
$$= \frac{n^3 - n}{3}$$

Next, we show an upper bound $O(n^3)$ as follows.

$\frac{n^3 - n}{3} \leq \frac{n^3}{3} = \frac{1}{3}n^3 = c|n^3|$ for all $n \geq N$ where $c = \frac{1}{3}$ and $N = 1$.

Next, we show a lower bound $\Omega(n^3)$ as follows.

$\frac{n^3 - n}{3} = \frac{n^3}{3} - \frac{n}{3} = \frac{n^3}{6} + \frac{(n^3 - 2n)}{6} > \frac{n^3}{6}$ for all $n \geq 2$ since $n^3 > 2n$ for all $n \geq 2$

Thus, $\frac{n^3 - n}{3} \geq c\ |n^3|$ holds for all $n \geq N$, where $c = \frac{1}{6}$ and $N = 2$.

Therefore, the tight bound $\sum_{i=1}^{n} i(i-1) = \Theta(n^3)$ holds. □

Ex.4 *Prove or disprove* $\ln(x^2 + 1) = \Theta(\lg x)$.

Theorem 4. $\ln(x^2 + 1) = \Theta(\lg x)$

Proof. We first convert the natural logarithm $\ln$ to $\lg$, i.e., the base of logarithm is converted from e to 2 by using the following fact (See Appendix 2 Theorem 3 $\log_a x = \frac{\log_b x}{\log_b a}$ in p.A-8 [4]).

$\ln(x^2 + 1) = \frac{\lg(x^2+1)}{\lg e}$ where e is the base of the natural logarithm called the *Napier constant*

Next, we prove an upper bound.

$\frac{\lg(x^2+1)}{\lg e} \leq \frac{\lg(x^2+x^2)}{\lg e}$ for all $x \geq 1$ since $1 \leq x^2$ for all $x \geq 1$

$= \dfrac{\lg(2x^2)}{\lg e}$ for all $x \geq 1$

$= \dfrac{\lg 2 + \lg x^2}{\lg e}$ for all $x \geq 1$ since $\log_b xy = \log_b x + \log_b y$ for all $b > 1$, $x > 0$ and $y > 0$

$= \dfrac{\lg 2 + 2\lg x}{\lg e}$ for all $x \geq 1$ since $\log_b x^r = r \log_b x$ for all $b > 1$, $x > 0$ and $r \in \mathbb{R}$

$\leq \dfrac{\lg x + 2\lg x}{\lg e}$ for all $x \geq 2$ since $\lg 2 \leq \lg x$ for all $x \geq 2$

$= \dfrac{3\lg x}{\lg e}$ for all $x \geq 2$

$= c_2 \,|\lg x|$ for all $x \geq N$, where $c_2 = \dfrac{3}{\lg e}$ and $N = 2$

Thus, the upper bound $\ln(x^2 + 1) = O(\lg x)$ holds.

Next, we prove a lower bound.

$\ln(x^2 + 1)$

$\geq \ln x^2$ for all $x \geq 2$ since $\ln x$ is a monotonically increasing function

$= 2 \ln x$ for all $x \geq 2$ since $\log_b x^r = r \log_b x$ for all $b > 1$, $x > 0$ and $r \in \mathbb{R}$

$= \dfrac{2\lg x}{\lg e}$ for all $x \geq 2$ by the base conversion

$= c_1 \,|\lg x|$ for all $x \geq N$, where $c_1 = \dfrac{2}{\lg e}$ and $N = 2$

Thus, the lower bound $\ln(x^2 + 1) = \Omega(\lg x)$ holds.

Therefore, the tight bound $\ln(x^2 + 1) = \Theta(\lg x)$ holds. □

Ex.5 ***Prove or disprove*** $\displaystyle\sum_{i=1}^{n} \frac{1}{2(i+1)} = \Theta(n \lg n)$.

Theorem 5. $\displaystyle\sum_{i=1}^{n} \frac{1}{2(i+1)} = \Theta(n \lg n)$

Proof. We first simplify the summation.

$$\sum_{i=1}^{n} \frac{1}{2(i+1)} = \frac{1}{2} \sum_{i=1}^{n} \frac{1}{i+1} = \frac{1}{2} \sum_{j=2}^{n+1} \frac{1}{j}$$

Next, we prove the following lemma.

Lemma 1. $\displaystyle\sum_{j=2}^{n} \frac{1}{j} = \Theta(\lg n)$

Proof for Lemma 1. We first prove an upper bound.

$$\sum_{j=2}^{n} \frac{1}{j}$$
$$\leq \int_{1}^{n} \frac{1}{x} dx$$
$$= \ln n - \ln 1$$
$$= \ln n \qquad \text{since } \ln 1 = 0$$
$$= \frac{\lg n}{\lg e} \qquad \text{by the base conversion}$$
$$= c_2 \, |\lg n| \qquad \text{for all } n \geq N, \text{ where } c_2 = \frac{1}{\lg e} \text{ and } N = 2$$

Next, we prove a lower bound.

$$\sum_{j=2}^{n} \frac{1}{j}$$
$$\geq \int_{2}^{n} \frac{1}{x} dx$$
$$= \ln n - \ln 2$$
$$> \ln n - 1 \qquad \text{since } \ln 2 < 1$$
$$= \frac{\lg n}{\lg e} - 1 \qquad \text{by the base conversion}$$
$$> \frac{\lg n}{2} - 1 \qquad \text{since } 1 < \lg e < 2$$
$$= \frac{\lg n}{4} + (\frac{\lg n}{4} - 1)$$
$$\geq \frac{\lg n}{4} \qquad \text{for all } n \geq 16 \qquad \text{since } \lg n \geq 4 \text{ for all } n \geq 16$$
$$= c_1 \, |\lg n| \qquad \text{for all } n \geq N, \text{ where } c_1 = \frac{1}{4} \text{ and } N = 16$$

Therefore, the tight bound holds. □

Let $c_1 = \frac{1}{4}$, $c_2 = \frac{2}{\lg e}$ and $N = 16$. By Lemma 1, the following holds for all $n \geq N$.

$$c_1 |\lg n| = \frac{1}{4} \lg n \leq \frac{1}{2} \sum_{j=2}^{n} \frac{1}{j} < \frac{1}{2} \sum_{j=2}^{n+1} \frac{1}{j} = \frac{1}{2} \sum_{j=2}^{n} \frac{1}{j} + \frac{1}{n+1} \leq \frac{1}{\lg e} \lg n + \frac{1}{n+1} < \frac{1}{\lg e} \lg n + 1$$
$$< \frac{1}{\lg e} \lg n + \frac{1}{\lg e} \lg n = \frac{2}{\lg e} \lg n = c_2 |\lg n| \text{ since } \frac{\lg n}{\lg e} > 1 \text{ for all } n \geq N.$$

Thus, the tight bound holds. □

2 Review of Proof by Induction

Consider Ex.18 (p.371) in Section 5.4 of the Rosen's textbook [5] for ICS141. The exercise is rephrased as follows.

> *" Prove* **by induction on** n *that the following recursive algorithm* `factorial` *correctly computes the factorial* $n!$ *that is inductively defined as* $0! = 1$ *and* $n! = \prod_{i=1}^{n} i$ *for* $n > 0$. *"*
>
> **function** `factorial`(n: **integer**): **integer**;
> // Given a nonnegative integer n, return the factorial n!.
> **if** $n = 0$ **then return** 1
> **else return** $n\times$ `factorial`$(n - 1)$

2.1 Partial Correctness

An algorithm is said to be *partially correct* if the algorithm produces a correct output when its execution terminates. The partial correctness will be discussed more rigorously in ICS241 and ICS311.

Informal Proof

Here is an informal proof that is sufficient in the context of ICS141.

Theorem 6. *The algorithm* `factorial` n *outputs* $n!$ *for every nonnegative integer* n *when the algorithm terminates.*

Proof. Suppose that the algorithm `factorial` terminates for every input. Let $factorial(n)$ denote its return value for input n. We prove $factorial(n) = n!$ by induction on n.

Basis: $n = 0$

The algorithm executes the **then** clause and returns 1. By the definition of $n!$, $0! = 1$. Thus, $factorial(0) = 0!$ holds.

Inductive Step: $n > 0$

Assume an induction hypothesis "$factorial(n) = n!$ for $n = k - 1$." Consider $n = k$. Since $n > 0$, the algorithm executes the **else** clause and returns $n \times factorial(n - 1)$. Since $n - 1 = k - 1$, the induction hypothesis can be applied to $factorial(n - 1)$. Hence, $factorial(n) = n \times factorial(n - 1) = n \times (n - 1)! = n!$ holds.

Thus, $factorial(n) = n!$ is true for all $n \geq 0$ when it terminates. Therefore, the algorithm correctly computes the factorial $n!$ whenever it terminates. □

2.2 Total Correctness

An algorithm is said to be *totally correct* if it satisfies the following two properties.

- *Partial Correctness:* The algorithm is partially correct.
- *Finite Termination:* For every input instance, the algorithm produces an output and terminates within a finite number of instructions executed.

Theorem 7. *For every input $n \geq 0$, the algorithm* `factorial` *always produces an output and terminates within a finite number of instructions executed.*

Proof. Since the algorithm `factorial` consists of only a single "**if-then-else**" statement and returns a value in each of the **then** and **else** clauses of "**if-then-else**" statement, it is obvious that the algorithm `factorial` always produces an output for every input $n \geq 0$.

We first show that the algorithm `factorial` invokes exactly n recursive calls.

Lemma 2. *For every input $n \geq 0$, the algorithm* `factorial` *invokes exactly n recursive calls.*

Proof for Lemma 2. Let $f(n)$ be the number of recursive calls invoked by `factorial`(n). By the construction of the algorithm `factorial`, we construct the following recurrence system on $f(n)$.

- **Initial Condition:** $f(0) = 0$
- **Recurrence Relation:** $f(n) = f(n-1) + 1$

By repeatedly applying the recurrence relation to a recurrence term in the right-hand side, it is easy to get the solution $f(n) = n$ and verify the solution by using a proof by induction. More techniques for solving recurrence systems will be discussed in ICS241.

Next, we show that an argument n of the algorithm `factorial` eventually reaches 0, since the argument n is decrement by 1 every time a recursive call is made. The algorithm `factorial`(0) obviously terminates without any more recursive call.

Hence recursion of `factorial`(n) always terminates after it invokes n recursive calls. □

Since the algorithm `factorial` consists of only a single "**if-then-else**" statement, it executes at most a constant number of instructions in each recursive call. Hence, the total number of instructions executed by `factorial` is $\Theta(n)$ for input n. Thus, it becomes obvious that the algorithm `factorial` terminates within a finite number of instructions executed. □

The following corollary follows Theorems 6 and 7.

Corollary 1. *The algorithm* `factorial` *is totally correct with respect to the initial assertion $n \geq 0$ and the final assertion $factorial = n!$.*

3 A Summary of Concepts and Techniques in ICS141

Although everything in ICS141 is important as prerequisites of computer science courses, here is a summary of mathematical concepts and techniques in ICS141 that are especially crucial to the design and analysis of algorithms.

Section # (7th Ed.)	Topics	Major Usage
Sections 1.7–1.8	Proof Methods	Design & Analysis
Sections 2.1–2.2	Sets	Modeling & Design
Section 2.3	Functions	Modeling & Design
Section 2.4	Sequences, Summation, Finite Series	Modeling & Analysis
Sections 3.2–3.3	Asymptotic Notations	Analysis
Sections 4.1–4.6	Basics of Number Theory	Modeling & Design
Section 2.6	Matrices	Modeling & Design
Sections 5.1–5.2	Proof by Induction	Design & Analysis
Section 5.3	Inductive Definitions, Recursive Definitions	Modeling & Design & Analysis
Section 5.4	Recursive Algorithms	Design
Section 5.5	Hoare Logic for Verification	Design & Analysis
Sections 6.1,6.3,6.5	4 Basic Counting Methods (Permutation, Combination, Sample, Selection),	Analysis
Sections 6.2,6.4	Binomial Coefficients, Pigeonhole Principle	Analysis
Sections 7.1–7.2	Basics of Probability Theory	Analysis
Section 7.3	Bayes' Theorem	Analysis
Section 7.4	Expected Value, Variance	Analysis

References

[1] Kenneth H. Rosen, *Discrete Mathematics and Its Applications*, 7th ed., New York, NY: McGraw-Hill, 2012, Chapter 3, Section 3.2, pp.204–218.

[2] Kazuo Sugihara. (2007, Oct. 1). ICS141 Lecture Notes #11. Dept. of Information and Computer Sciences, University of Hawaii at Manoa. Honolulu, HI. [Online]. Available: http://pearl.ics.hawaii.edu/∼sugihara/courses/ics141f07/notes/10-01n11.html#S3

[3] Kenneth H. Rosen, *Discrete Mathematics and Its Applications*, 7th ed., New York, NY: McGraw-Hill, 2012, Chapter 2, Section 2.4, p.166.

[4] Kenneth H. Rosen, *Discrete Mathematics and Its Applications*, 7th ed., New York, NY: McGraw-Hill, 2012, Appendix 2, p.A-8.

[5] Kenneth H. Rosen, *Discrete Mathematics and Its Applications*, 7th ed., New York, NY: McGraw-Hill, 2012, Chapter 5, Section 5.4, p.371.

[6] Kenneth H. Rosen, *Discrete Mathematics and Its Applications*, 7th ed., New York, NY: McGraw-Hill, 2012, Chapter 5, Section 5.5, pp.372–377.

[7] Kazuo Sugihara. (2007, Oct. 22). ICS141 Lecture Notes #17. Dept. of Information and Computer Sciences, University of Hawaii at Manoa. Honolulu, HI. [Online]. Available:
http://pearl.ics.hawaii.edu/∼sugihara/courses/ics141f07/notes/10-22n17.html#S2