# CS 240: Data Structures and Data Management
# Module 1 Study Guide

Taylor J. Smith — Spring 2017

## Key Concepts

- An **algorithm** is a step-by-step process for carrying out a series of computations to produce a **solution** for a given **instance** of a **problem**.
- **Order notation:**
  - $f(n) \in O(g(n))$ if $\exists c > 0,\ \exists n_0 > 0$ such that $\forall n \geq n_0$, $0 \leq f(n) \leq cg(n)$.
  - $f(n) \in \Omega(g(n))$ if $\exists c > 0,\ \exists n_0 > 0$ such that $\forall n \geq n_0$, $0 \leq cg(n) \leq f(n)$.
  - $f(n) \in \Theta(g(n))$ if $\exists c_1, c_2 > 0,\ \exists n_0 > 0$ such that $\forall n \geq n_0$, $0 \leq c_1 g(n) \leq f(n) \leq c_2 g(n)$.
  - $f(n) \in o(g(n))$ if $\forall c > 0,\ \exists n_0 > 0$ such that $\forall n \geq n_0$, $0 \leq f(n) < cg(n)$.
  - $f(n) \in \omega(g(n))$ if $\forall c > 0,\ \exists n_0 > 0$ such that $\forall n \geq n_0$, $0 \leq cg(n) < f(n)$.
- **Growth rates:**

$$\Theta(1) \subset \Theta(\log(n)) \subset \Theta(n) \subset \Theta(n\log(n)) \subset \Theta(n\log^k(n)) \subset \Theta(n^k) \subset \Theta(k^n) \text{ for } k > 1$$

$$\text{If } L = \lim_{n\to\infty} \frac{f(n)}{g(n)} \text{ then } f(n) \in \begin{cases} o(g(n)) & \text{if } L = 0 \\ \Theta(g(n)) & \text{if } 0 < L < \infty \\ \omega(g(n)) & \text{if } L = \infty \end{cases}$$

- **Relationships:**
  - $f(n) \in \Theta(g(n)) \Leftrightarrow g(n) \in \Theta(f(n))$
  - $f(n) \in O(g(n)) \Leftrightarrow g(n) \in \Omega(f(n))$
  - $f(n) \in o(g(n)) \Leftrightarrow g(n) \in \omega(f(n))$
  - $f(n) \in \Theta(g(n)) \Leftrightarrow f(n) \in O(g(n))$ and $f(n) \in \Omega(g(n))$

## Suggested Readings

- **Sedgewick:** 2.1 (Implementation and Empirical Analysis), 2.2 (Analysis of Algorithms), 2.4 (Big-Oh Notation), 2.6 (Examples of Algorithm Analysis), 2.7 (Guarantees, Predictions, and Limitations)
- **CLRS:** 3.1 (Asymptotic notation)
- **Goodrich/Tamassia:** 1.2 (Asymptotic Notation), 1.3 (A Quick Mathematical Review)

## Additional Readings

- **Sedgewick:** 2.5 (Basic Recurrences)
- **CLRS:** 3.2 (Standard notations and common functions), Chapter 4 (Recurrences)

## Practice Questions

### Sedgewick

2.20. Prove that $O(1)$ is the same as $O(2)$.

2.21. Prove that we can make any of the following transformations in an expression that uses the $O$-notation:

$$\begin{aligned} f(n) &\to O(f(n)), \\ cO(f(n)) &\to O(f(n)), \\ O(cf(n)) &\to O(f(n)), \\ f(n) - g(n) = O(h(n)) &\to f(n) = g(n) + O(h(n)), \\ O(f(n))O(g(n)) &\to O(f(n)g(n)), \\ O(f(n)) + O(g(n)) &\to O(g(n)) \qquad \text{if } f(n) = O(g(n)). \end{aligned}$$

2.22. Show that $(n+1)(H_n + O(1)) = n\ln(n) + O(n)$. (*Note:* $H_n = \sum_{i=1}^{n}(1/i)$.)

2.23. Show that $n\ln(n) = O(n^{3/2})$.

2.40. Solve the recurrence

$$C_n = C_{n/2} + n^2, \qquad \text{for } n \geq 2 \text{ with } C_1 = 0,$$

when $n$ is a power of 2.

2.41. Solve the recurrence

$$C_n = C_{n/\alpha} + 1, \qquad \text{for } n \geq 2 \text{ with } C_1 = 0,$$

when $n$ is a power of $\alpha$.

2.42. Solve the recurrence

$$C_n = \alpha C_{n/2}, \qquad \text{for } n \geq 2 \text{ with } C_1 = 1,$$

when $n$ is a power of 2.

2.47. Give the average number of comparisons used by the linear search algorithm in the case that $\alpha n$ of the searches are successful, for $0 \leq \alpha \leq 1$.

2.48. Estimate the probability that at least one of $m$ random 10-digit numbers matches one of a set of $n$ given values, for $m = 10$, 100, and 1000 and $n = 10^3$, $10^4$, $10^5$, and $10^6$.

### CLRS

3.1-1. Let $f(n)$ and $g(n)$ be asymptotically nonnegative functions. Using the basic definition of $\Theta$-notation, prove that $\max(f(n), g(n)) = \Theta(f(n) + g(n))$.

3.1-2. Show that for any real constants $a$ and $b$, where $b > 0$, $(n+a)^b = \Theta(n^b)$.

3.1-3. Explain why the statement, "The running time of algorithm A is at least $O(n^2)$," is meaningless.

3.1-4. Is $2^{n+1} = O(2^n)$? Is $2^{2n} = O(2^n)$?

3-2. Indicate, for each pair of expressions $(A, B)$ in the table below, whether $A$ is $O$, $o$, $\Omega$, $\omega$, or $\Theta$ of $B$. Assume that $k \geq 1$, $\epsilon > 0$, and $c > 1$ are constants. Your answer should be in the form of the table with "yes" or "no" written in each box.

| $A$ | $B$ | $O$ | $o$ | $\Omega$ | $\omega$ | $\Theta$ |
|---|---|---|---|---|---|---|
| $\log^k(n)$ | $n^\epsilon$ | | | | | |
| $n^k$ | $c^n$ | | | | | |
| $\sqrt{n}$ | $n^{\sin(n)}$ | | | | | |
| $2^n$ | $2^{n/2}$ | | | | | |
| $n^{\log(c)}$ | $c^{\log(n)}$ | | | | | |
| $\log(n!)$ | $\log(n^n)$ | | | | | |

3-3a. Rank the following functions by order of growth; that is, find an arrangement $g_1, g_2, \ldots, g_{30}$ of the functions satisfying $g_1 = \Omega(g_2)$, $g_2 = \Omega(g_3)$, ..., $g_{29} = \Omega(g_{30})$. Partition your list into equivalence classes such that $f(n)$ and $g(n)$ are in the same class if and only if $f(n) = \Theta(g(n))$.

| | | | | | |
|---|---|---|---|---|---|
| $\log(\log^*(n))$ | $2^{\log^*(n)}$ | $(\sqrt{2})^{\log(n)}$ | $n^2$ | $n!$ | $(\log(n))!$ |
| $\left(\frac{3}{2}\right)^n$ | $n^3$ | $\log^2(n)$ | $\log(n!)$ | $2^{2^n}$ | $n^{1/\log(n)}$ |
| $\ln\ln(n)$ | $\log^*(n)$ | $n \cdot 2^n$ | $n^{\log\log(n)}$ | $\ln(n)$ | $1$ |
| $2^{\log(n)}$ | $(\log(n))^{\log(n)}$ | $e^n$ | $4^{\log(n)}$ | $(n+1)!$ | $\sqrt{\log(n)}$ |
| $\log^*(\log(n))$ | $2^{\sqrt{2\log(n)}}$ | $n$ | $2^n$ | $n\log(n)$ | $2^{2^{n+1}}$ |

(*Note:* $\log^*(n)$ denotes the iterated logarithm function, or the number of times the logarithm must be applied to $n$ before the result is less than or equal to 1. In other terms, $\log^*(n) = \min(i \geq 0 \mid \log^i(n) \leq 1)$.)

3-4. Let $f(n)$ and $g(n)$ be asymptotically positive functions. Prove or disprove each of the following conjectures.

(a) $f(n) = O(g(n))$ implies $g(n) = O(f(n))$.

(b) $f(n) + g(n) = \Theta(\min(f(n), g(n)))$.

(c) $f(n) = O(g(n))$ implies $\log(f(n)) = O(\log(g(n)))$, where $\log(g(n)) \geq 1$ and $f(n) \geq 1$ for all sufficiently large $n$.

(d) $f(n) = O(g(n))$ implies $2^{f(n)} = O(2^{g(n)})$.

(e) $f(n) = O((f(n))^2)$.

(f) $f(n) = O(g(n))$ implies $g(n) = \Omega(f(n))$.

(g) $f(n) = \Theta(f(n/2))$.

(h) $f(n) + o(f(n)) = \Theta(f(n))$.

4.1-1. Show that the solution of $T(n) = T(\lceil n/2 \rceil) + 1$ is $O(\log(n))$.

4.1-5. Show that the solution of $T(n) = 2T(\lfloor n/2 \rfloor + 17) + n$ is $O(n\log(n))$.

**Goodrich/Tamassia**

R-1.4. Show that $\log^3(n)$ is $o(n^{1/3})$.

R-1.9. Consider the following recurrence equation, defining $T(n)$, as

$$T(n) = \begin{cases} 4 & \text{if } n = 1 \\ T(n-1) + 4 & \text{otherwise.} \end{cases}$$

Show, by induction, that $T(n) = 4n$.

R-1.19. Show that $(n+1)^5$ is $O(n^5)$.

R-1.22. Show that $n^2$ is $\omega(n)$.

R-1.23. Show that $n^3\log(n)$ is $\Omega(n^3)$.