Foundations II: Data Structures and Algorithms

Instructor: Yusu Wang
Topic 1: Introduction and Asymptotic notation
Course Information

- Course webpage
  
  [http://www.cse.ohio-state.edu/~yusu/courses/2331](http://www.cse.ohio-state.edu/~yusu/courses/2331)

  CANVAS: [https://carmen.osu.edu](https://carmen.osu.edu)

- Office hours
  
  - Tue: 11:30 am – 12:30 pm; Thu: 3:45pm – 5:00pm

- Grading policy:
  
  - homework: 20%,
  - two midterms: 40%,
  - final: 40%
Exams

- Midterm 1
  - Oct. 7\(^{th}\) : 7 – 8:45pm, Location: KN250
- Midterm 2
  - Nov. 4\(^{th}\) : 7 – 8:45pm, Location: KN250
- Final exam:
  - Dec. 6\(^{th}\) (Friday), 6:00pm--7:45pm, Room DL113
Exams

- Midterm 1
  - Oct. 7\textsuperscript{th} : 7 – 8:45pm, Location: KN250

- Midterm 2
  - Nov. 4\textsuperscript{th} : 7 – 8:45pm, Location: KN250

- Final exam:
  - Dec. 6\textsuperscript{th} (Friday), 6:00pm--7:45pm, Room DL113

Mark you calendar.
Contact me by Aug. 28 if you have schedule conflict for any exam.
Notes

- All homework should be submitted before or in class on the due date. Late homework during the same day receives a 10% penalty. Late homework submitted the next day receives a 30% penalty.

- You may discuss homework with your classmates. It is very important that you write up your solutions individually, and acknowledge any collaboration / discussion with others.
More Information

- **Textbook**
  - *Introduction to Algorithms*
    - by *Cormen, Leiserson, Rivest and Stein* (third edition)

- **Others**
  - CSE 2331 course notes
  - *The Art of Computer Programming*
    - By *Donald Knuth*
Introduction

- What is an algorithm?
  - Step by step strategy to solve problems
  - Should terminate
  - Should return correct answer for any input instance
This Course

- Various issues involved in designing good algorithms
  - What do we mean by “good”?
    - Algorithm analysis, language used to measure performance
  - How to design good algorithms
    - Data structures, algorithm paradigms
  - Fundamental problems
    - Graph traversal, shortest path etc.
Asymptotic Notation
Pseudo-code

Function Test($A, n$)

for $i = 1$ to $2n-1$ do

    $key = A[i+1]$

    for $j = 1$ to $i$ do


        $key = A[j+1] \times 2$

    end

end

$A[i] = key$

return $key$
Running Time

- Depends on input size
- Depends on particular input
  - Worst case
    - $T(n) = \max \text{ time of algorithm on any input of size } n$
Pseudo-code

Function Test($A, n$)

for $i = 1$ to $2n-1$ do

\[
key = A[i+1]
\]

for $j = 1$ to $i$ do

\[
key = A[j+1] \times 2
\]

\[
A[i] = key
\]

return $key$

\[
T(n) = \sum_{i=2}^{n} (c_1 + c_2 i) = c_3 n^2 + c_4 n + c_5
\]
Asymptotic Complexity

- Why shall we care about constants and lower order terms?

- Should focus on the relative growth w.r.t. \( n \)
  - Especially when \( n \) becomes large!
  - E.g: \( 0.2n^{1.5} \) v.s \( 500n \)
\[ f(n) = O(g(n)) \]
**Big O Notation**

- $f(n) \in O(g(n))$ if and only if $\exists c > 0$, and $n_0$, so that $0 \leq f(n) \leq cg(n)$ for all $n \geq n_0$

- $f(n) = O(g(n))$ means $f(n) \in O(g(n))$ (i.e, at most)
  - We say that $g(n)$ is an *asymptotic upper bound* for $f(n)$.

- It also means: $\lim_{n \to \infty} \frac{f(n)}{g(n)} \leq c$

- E.g, $f(n) = O(n^2)$ if there exists $c, n_0 > 0$ such that $f(n) \leq cn^2$, for all $n \geq n_0$. 
More Examples

- $5n^2 + 6n + 8 = O(n^3)$ ?
- $100n^2 - 1000n = O(n^2)$ ?
- $\sqrt{6n^3 + 7n^2 + 3n} = O(n^{1.5})$ ?
- $2^n = O(n^2)$ ?
- $3\lg n = O(n)$ ?
- $3^n = O(2^n)$ ?
- $3^n = O(2^{2n})$
- $\log_3 n = O(\log_2 n)$ ?
- $\frac{n}{\lg n} = O(n)$
Prove that $\sqrt{6n^3 + 7n^2 + 3n} = O(n^{1.5})$

Note that

- $\sqrt{6n^3 + 7n^2 + 3n} \leq \sqrt{6n^3 + 7n^3 + 3n^3} \leq \sqrt{16n^3} = 4n^{1.5}$

By definition, it then follows that $\sqrt{6n^3 + 7n^2 + 3n} = O(n^{1.5})$
Omega \( \Omega \) Notation

- \( f(n) \in \Omega (g(n)) \) if and only if \( \exists c > 0 \), and \( n_0 \), so that \( 0 \leq cg(n) \leq f(n) \) for all \( n \geq n_0 \)

- \( f(n) = \Omega (g(n)) \) means \( f(n) \in \Omega (g(n)) \) (i.e, at least)
  - We say \( g(n) \) is an *asymptotic lower bound* of \( f(n) \).

- It also means:
  \[
  \lim_{n \to \infty} \frac{f(n)}{g(n)} \geq c
  \]
Illustration

\[ f(n) = \Omega(g(n)) \]
More Examples

- $5n^2 + 6n + 8 = \Omega(n^3)$?
- $5n^2 + 6n + 8 = \Omega(n^2)$?
- $\sqrt{6n^3 - 7n^2 + 3n} = \Omega(n^{1.5})$?
- $2^n = \Omega(n^2)$?
- $3\log n = \Omega(n)$?
- $3^n = \Omega(2^n)$?
- $\log_3 n = \Omega(\log_2 n)$?
- $2^{\log_3 n} = \Omega(2^{\log_2 n})$
Theta $\Theta$ Notation

- Combine lower and upper bound
- Means tight: of the same order

$f(n) \in \Theta(g(n))$ if and only if $\exists c_1, c_2 > 0$, and $n_0$, such that $c_1 g(n) \leq f(n) \leq c_2 g(n)$ for any $n \geq n_0$

- $f(n) = \Theta(g(n))$ means $f(n) \in \Theta(g(n))$
  - We say $g(n)$ is an asymptotically tight bound for $f(n)$.
- It also means:
  - $c_1 \leq \lim_{n \to \infty} \frac{f(n)}{g(n)} \leq c_2$
Prove that $\sqrt{6n^3 - 7n^2 + 3n} = \Theta(n^{1.5})$

First note that:
- $\sqrt{6n^3 - 7n^2 + 3n} \leq \sqrt{6n^3 + 3n^3} \leq \sqrt{9n^3} = 3n^{1.5}$
- $\Rightarrow \sqrt{6n^3 - 7n^2 + 3n} = O(n^{1.5})$

Second note that
- $\sqrt{6n^3 - 7n^2 + 3n} \geq \sqrt{6n^3 - 7n^2}$
- For $n \geq 7$, $\sqrt{6n^3 - 7n^2} \geq \sqrt{5n^3} = \sqrt{5} n^{1.5}$
- It then follows that $\sqrt{6n^3 - 7n^2 + 3n} = \Omega(n^{1.5})$

Putting the upper and lower bound together, we thus have $\sqrt{6n^3 - 7n^2 + 3n} = \Theta(n^{1.5})$
\[ f(n) = O(g(n)) \quad f(n) = \Omega(g(n)) \quad f(n) = \Theta(g(n)) \]
Remarks -- 1

- \( f(n) = \Theta(g(n)) \) if and only if \( f(n) = O(g(n)) \), and \( f(n) = \Omega(g(n)) \).
  - Insertion sort algorithm as described earlier has time complexity \( \Theta(n^2) \).

- Transpose symmetry:
  - If \( f(n) = O(g(n)) \), if and only if \( g(n) = \Omega(f(n)) \)

- Transitivity:
  - If \( f(n) = O(g(n)) \) and \( g(n) = O(h(n)) \), then \( f(n) = O(h(n)) \).
  - Same for \( \Omega \) and \( \Theta \)
Remarks -- 2

- It is convenient to think that
  - Big-O is smaller than or equal to
  - Big-Ω is larger than or equal to
  - Big-Θ is equal

- However,
  - These relations are modulo constant factor scaling
  - Not every pair of functions have such relations

If \( f(n) \notin O(g(n)) \), this does not imply that \( f(n) = \Omega(g(n)) \).
Remarks -- 3

- Provide a unified language to measure the performance of algorithms
  - Give us intuitive idea how fast we shall expect the alg.
  - Can now compare various algorithms for the same problem

- Constants hidden!
  - $O(n \ lg \ lg \ n), \ 2n \ lg \ n$

- $T(n) = n^2 + O(n)$
  means $T(n) = n^2 + h(n), \ and \ h(n) = O(n)$
Pitfalls

- $O(n) + O(n) = O(n)$
  - OK

- $T(n) = n + \sum_{i=1}^{k} O(n)$
  
  $= n + O(n)$
  
  ?

$k$ should not depend on $n$! It can be an arbitrarily large constant …
Yet Another Notation

- Small $o$ notation:

- $f(n) = o(g(n))$ means for any $c > 0$, $\exists n_0$ s.t. for all $n \geq n_0$, $0 \leq f(n) < cg(n)$. In other words,

$$\lim_{n \to \infty} \frac{f(n)}{g(n)} = 0$$

- Examples:
  - Is $n - \lg n = o(n)$?
  - Is $n = o(n \lg n)$?
Yet Another Notation

- Small $\omega$ notation:

  $f(n) = \omega(g(n))$ means for any $c > 0$, $\exists n_0$ s.t. for all $n \geq n_0$, $0 \leq cg(n) < f(n)$. In other words,

  $$\lim_{n \rightarrow \infty} \frac{f(n)}{g(n)} = \infty \quad \text{or} \quad \lim_{n \rightarrow \infty} \frac{g(n)}{f(n)} = 0$$

- Examples:
  - Is $n - \lg n = \omega(n)$?
  - Is $n = \omega(n / \lg n)$?
Summary Asymptotic Notation

\[ f(n) \in O(g(n)) \] if there exists \( c, n_0 > 0 \) such that:
\[ f(n) \leq cg(n) \quad \text{for all } n \geq n_0. \]

\[ f(n) \in \Omega(g(n)) \] if there exists \( c, n_0 > 0 \) such that:
\[ f(n) \geq cg(n) \quad \text{for all } n \geq n_0. \]

\[ f(n) \in \Theta(g(n)) \] if there exists \( c_1, c_2, n_0 > 0 \) such that:
\[ c_1g(n) \leq f(n) \leq c_2g(n) \quad \text{for all } n \geq n_0. \]
Another View

\[ f(n) \in O(g(n)) \text{ if there exists } c > 0 \text{ such that:} \]
\[ \lim_{n \to \infty} \frac{f(n)}{g(n)} \leq c. \]

\[ f(n) \in \Omega(g(n)) \text{ if there exists } c > 0 \text{ such that:} \]
\[ \lim_{n \to \infty} \frac{f(n)}{g(n)} \geq c. \]

\[ f(n) \in \Theta(g(n)) \text{ if there exists } c_1, c_2 > 0 \text{ such that:} \]
\[ c_1 \leq \lim_{n \to \infty} \frac{f(n)}{g(n)} \leq c_2. \]
Some Special Cases

If \( \lim_{n \to \infty} \frac{f(n)}{g(n)} = 0 \), then

\[ f(n) \in O(g(n)) \text{ but } f(n) \notin \Theta(g(n)). \]

If \( \lim_{n \to \infty} \frac{f(n)}{g(n)} = \infty \), then

\[ f(n) \in \Omega(g(n)) \text{ but } f(n) \notin \Theta(g(n)). \]

If \( \lim_{n \to \infty} \frac{f(n)}{g(n)} = c > 0 \ (c \neq \infty) \), then

\[ f(n) \in \Theta(g(n)). \]
Example

- Prove that $3 \ln n = O(n)$
  
  - This is because $\lim_{n \to \infty} \frac{3 \ln n}{n} = 0$
- Review some basics
  - Chapter 3.2 of CLRS
  - See the board
More Examples

- $n^3$ vs. $(1.0001)^n$
- $n^3$ vs. $(1.0001)^{\frac{n}{2}}$
- $n^a$ vs. $b^n$
- $\ln(n^4)$ vs. $\log_3 2n$
- $2 \lg n$ vs. $\sqrt{n}$
- $\lg n$ vs. $n^\epsilon$
- $n \lg n$ vs. $n \sqrt{\lg n}$
- $n$ vs. $\log_2 3^n$
- $n$ vs. $2^{\log_2 n+4}$
- $n \log_2 n$ vs. $2^{2 \log_2 n}$
More Examples

- $\lg n^2$ vs. $(\lg n)^2 - 100 \lg n$
- $10^{10}$ vs. $\ln n$
- $3^{\log_2 n}$ vs. $0.5n$
- $\lg \sqrt{2n^2 + n \lg n}$ vs. $\lg (n^3 - 100n)$
- $n - \sqrt{n \lg n}$ vs. $\sqrt{2n^2 + n \lg n}$

In general, we can safely ignore low order terms.
Hierarchy

- $\Theta(n^n)$
- $\Theta(3^n)$
- $\Theta(2^n)$
- $\Theta(n^3)$
- $\Theta(n^2)$
- $\Theta(n \log(n))$
- $\Theta(n)$
- $\Theta(n^{0.5})$
- $\Theta(n^{0.1})$
- $\Theta((\log(n))^2)$
- $\Theta(\log(n))$
- $\Theta(1)$
Rank the following functions by order of growth:

- $2^{n-4}$,
- $(n + 3) \log(3n^2 - 4n)$,
- $n\sqrt{n} + 5$,
- $n \ln n$,
- $3 \cdot 2^n$,
- $\frac{4n}{\log n}$,
- $\sqrt{n} + \log n$
Summary

- Algorithms are useful

- Example: sorting problem
  - Insertion sort

- Asymptotic complexity
  - Focus on the growth of complexity w.r.t input size