CSC 226

Algorithms and Data Structures: II

Tianming Wei
twei@uvic.ca
ECS 466

Lectures and Labs

Tianming Wei

- E-mail: twei@uvic.ca
- Voice: 250-472-5758
- Office: ECS 466
- Office hours:
  - Tue: 1:30 pm – 3:00 pm
  - Fri: 1:30 pm – 3:00 pm
- Lectures
  - TWF 9:30 am – 10:20 am
  - HSD A240

- Labs
  - Instructor: TBA
  - Labs start week of May 13, 2019
  - B01 W 1:30 – 2:20 pm ECS 342
  - B02 W 2:30 – 3:20 pm ECS 342
  - B03 W 3:30 – 4:20 pm ECS 342
  - B04 Th 1:30 – 2:20 pm ECS 342
  - B05 Th 2:30 – 3:20 pm ECS 342
  - Check UVic website
- Register for labs!

- Course Web pages
  - Official Webpage on the Department Website
  - Detailed Course Website on ConneX
    - https://connex.csc.uvic.ca/portal/site/

Administrative Officer Announcements

- CSC Undergraduate Officer is Irene Statham
  E-mail: cscadvisor@uvic.ca   Office: ECS 512
- Any student who has registered in CSC 226 and **does not** have the required pre-requisites and no waiver **must drop the class**. Otherwise: student will be dropped and a pre-requisite drop is recorded on the student's record.
- Taking the course more than twice:

  you must request, in writing, permission from the Chair of the Department and the Dean of the Faculty to be allowed to stay registered in the class (University Rule). The letter should be given to Irene Statham, Undergraduate Advisor. Otherwise: student will be dropped from class.
- Always use and check your UVic e-mail account and use CSC 226 as part of the subject line.
- Do not send messages from other accounts (such messages are filtered and discarded).

MAY 2018

Computer Science

S	M	T	W	T	F	S
		1	**1**	**2**	**3**	**4**
5 **WEEK 1** **NO LAB →**	**6** *Summer term classes begin*	**7**	**8**	**9**	**10**	**11**
12 **WEEK 2** **LAB 1 →**	**13**	**14**	**15**	**16**	**17**	**18** *Last day for 100% reduction/add courses*
19 **WEEK 3** **LAB 2 →**	**20** **Victoria Day**	**21**	**22 Quiz 1**	**23**	**24**	**25**
26 **WEEK 4** **LAB 3 →**	**27**	**28**	**29**	**30**	**31 Due: Ass 1** *Last day for paying summer term fees*	

JUNE 2018

Computer Science

S	M	T	W	T	F	S
					31 **Due: Ass 1**	1
2 WEEK 5 LAB 4 →	3	4	5 **Quiz 2**	6	7	8 *Last day for 50% reduction of tuition*
9 WEEK 6 LAB 5 →	10	11	12	13	14 **Due: Ass 2**	15
16 WEEK 7 LAB 6 →	17	18	19	20	21 **Midterm**	22
23 WEEK 8 LAB 7 →	24	25	26	27	28	29

JULY 2018

Computer Science

S	M	T	W	T	F	S
30 WEEK 9 NO LAB →	1 Reading Break	2 Reading Break	3 Quiz 3 *Last day to withdraw*	4	5	6
7 WEEK 10 LAB 8 →	8	9	10	11	12 Due: Ass 3	13
14 WEEK 11 LAB 9 →	15	16	17 Quiz 4	18	19	20
21 WEEK 12 LAB 10 →	22	23	24	25	26 Due: Ass 4	27
28 WEEK 13 NO LAB →	29	30	31 Quiz 5			

AUGUST 2018

Computer Science

S	M	T	W	T	F	S
			31	1	2 *Last day of classes*	3
4	5 BC Day	6 *Exams begin for summer term courses*	7	8	9	10
11	12	13	14	15	16 *Exams end for summer term courses*	17
18	19	20	21	22	23	24
25	26	27	28	29	30	31

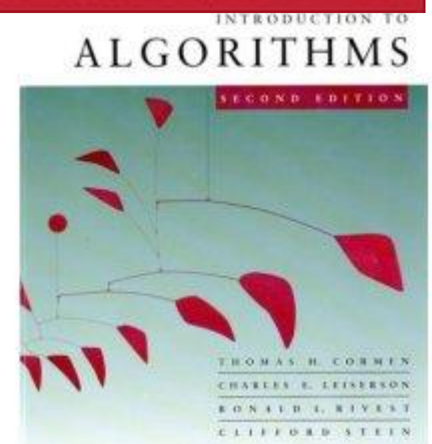
ENGL 135 final exam - Date to be confirmed by UVic

Books

- **Required Textbook**
  R. Sedgewick and K. Wayne
  Algorithms, Fourth Edition
  Addison-Wesley, Toronto, 2011
  ISBN: 0-321-57351-X

- http://algs4.cs.princeton.edu/home/

- **Optional Textbook (online)**
  T.H. Cormen, C.E. Leiserson, R.L. Rivest, C. Stein. *Introduction to Algorithms*. MIT Press (2001), 2nd edition.



Evaluation

Assignments	30%
Quizzes	5%
Midterm	25%
Final	40%

- Marks will be posted in the Connex Gradebook
- Midterm exam will be in-class, 50 min, closed books, closed notes, no calculators, no gadgets
  Friday, June 21, 2019
- The final exam will be three hours, closed books, closed notes, no calculators, no gadgets scheduled by the registrar
  - Do NOT make travel plans until the schedule is out!

Assignment Schedule

Assignment	Due Date (Tentative)
A1	May 31, 2019
A2	June 14, 2019
A3	July 12, 2019
A4	July 26, 2019

Quiz Schedule

Quiz	**Date**
Q1	May 22, 2019
Q2	June 5, 2019
Q3	July 3, 2019
Q4	July 17, 2019
Q5	July 31, 2019

Assignments

- **Programming: work in the labs or at home**
  - ➢use your favorite Java environment
  - ➢Textbook's booksite has supplemental classes and data
    - http://algs4.cs.princeton.edu/home/
- **Cheating: zero-tolerance policy**
  - ➢first time fail assignment, second time fail course

Lecture Notes

- **Acknowledgments**
  - ➢ I use slides from past offerings of this course, prepared by Dr. Rich Little and Dr. Ulrike Stege (thank you both!!!).
- Consider posted lecture slides as *additional* information
- **Note**
  - ➢ NOT all materials required for the midterm and final exams are on the lecture slides

Questions?

- Regarding questions on lectures, assignments, algorithms, data structures, programming, Java, etc. consult in the following order:
  - Study group, book, book website
  - ConneX web page
  - Lab instructor
  - Instructor

Prerequisites - CSC 225

- Pseudocode, Counting Operations
- Recursion, Induction Proofs
- Big-Oh Analysis (1.4)
- Abstract Data Types (1.3)
- Quadratic Sorting Algorithms (2.1)
- Merge-sort (2.2)
- Quicksort (2.3)
- Priority queues, Heap-sort (2.4)
- Selection algorithms (2.5)
- Trees, Binary Search Trees (3.1, 3.2)
- Radix-sort (5.1)
- Hashing (3.4)
- Graph theory (4.1)
- Graph ADT (4.1)
- BFS, DFS (4.1)
- Strongly Connected Components (4.2)
- Topological sort (4.2)
- Transitive closure (4.2)

Fundamental Algorithms

- Selection (Linear Median)
- Quicksort, Heapsort
- Linear search, Binary Search, Hash search
- Tree traversals
- Graph traversals
- Depth first search, breadth first search

Algorithm Design Techniques

- **Algorithm Design Techniques**
  - Greedy algorithms
    - Local optimums lead to global optimum
  - Divide and conquer
    - Recursively subdivide problem
  - Dynamic programming
    - Incrementally build complete solution
  - Backtracking
    - Technique for finding all solutions

Learning Outcomes for CSC 226

1. Understand the fundamental algorithm design paradigms and data structures
2. Apply mathematical techniques and tools to analyze running times and correctness of algorithms
3. Compare and choose the most appropriate paradigm/data structure to solve a problem
4. Correctly implement the best solution to a given problem

Course Division – CSC 226

The course will have four modules:

1. Topics from Sorting and Discrete Math
2. Advanced Graph Algorithms
3. Text-Processing Algorithms
4. Algorithms for *Hard* Problems

Course Topics – CSC 226

- Introduction and asymptotic review (1.4)
- Sorting revisited (2.2)
- Discrete and Combinatorial Math
- Balanced Binary Search Trees (3.3)
- Undirected graphs (4.1)
- Directed graphs (4.2)
- Minimum spanning trees (4.3)
- Union-find (1.5)

- Shortest path algorithms (4.4)
- Network flow (6.4)
- Longest Common Subsequence
- Tries (5.2)
- Substring search (5.3)
- Data compression (5.5)
- Planar graph algorithms (6.5)
- Coping with intractability (6.6)

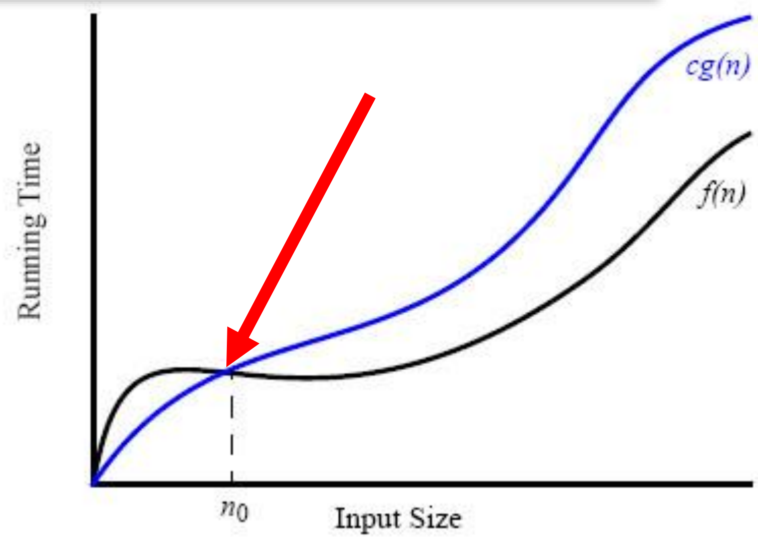
Asymptotic Notation Review

- Big-Oh
- Big-Omega
- Big-Theta
- Little-oh
- Little-omega

Formal Definition of Big-Oh Notation

Let f: IN→IR and g: IN→IR. f(n) is O(g(n)) if and only if
there exists a real constant $c > 0$
and an integer constant $n_0 > 0$
such that $f(n) \le c \cdot g(n)$ for all $n \ge n_0$.
IN: non-negative integers
IR: real numbers

- We say
  - f(n) is *order* g(n)
  - f(n) is *big-Oh* of g(n)
  - $f(n) \in O(g(n))$
- Visually, this says that the f(n) curve must eventually fit under the cg(n) curve.



Theorem

- **R1:** If $d(n)$ is $O(f(n))$, then $ad(n)$ is $O(f(n))$, $a > 0$
- **R2:** If $d(n)$ is $O(f(n))$ and $e(n)$ is $O(g(n))$, then $d(n) + e(n)$ is $O(f(n) + g(n))$
- **R3:** If $d(n)$ is $O(f(n))$ and $e(n)$ is $O(g(n))$, then $d(n)e(n)$ is $O(f(n)g(n))$
- **R4:** If $d(n)$ is $O(f(n))$ and $f(n)$ is $O(g(n))$, then $d(n)$ is $O(g(n))$
- **R5:** If $f(n) = a_0 + a_1 n + \ldots + a_d n^d$, d and a_k are constants, then $f(n)$ is $O(n^d)$
- **R6:** n^x is $O(a^n)$ for any fixed $x > 0$ and $a > 1$
- **R7:** $\log(n^x)$ is $O(\log n)$ for any fixed $x > 0$
- **R8:** $\log^x n$ is $O(n^y)$ for any fixed constants $x > 0$ and $y > 0$

Names of Most Common Big Oh Functions

- Constant $O(1)$
- Logarithmic $O(\log n)$
- Linear $O(n)$
- Linearithmic $O(n \log n)$
- Quadratic $O(n^2)$
- Polynomial $O(n^k)$, k is a constant

- Exponential $O(2^n)$
- Exponential $O(a^n)$, a is a constant and $a > 1$

Most Common Functions in Algorithm Analysis Ordered by Growth

$$1 \quad \log\log n \quad \log \mathrm{n} \quad \sqrt{\mathrm{n}}$$

$$n \quad n\log n \quad \mathrm{n}^2 \quad \mathrm{n}^{2.31}$$

$$n^3 \quad n^k, \text{ for } k > 3$$

$$2^n \quad 3^n \quad n! \quad n^n$$

Big-Omega Notation

Let f: IN→IR and g: IN→IR. f(n) is $\Omega(g(n))$
if and only if

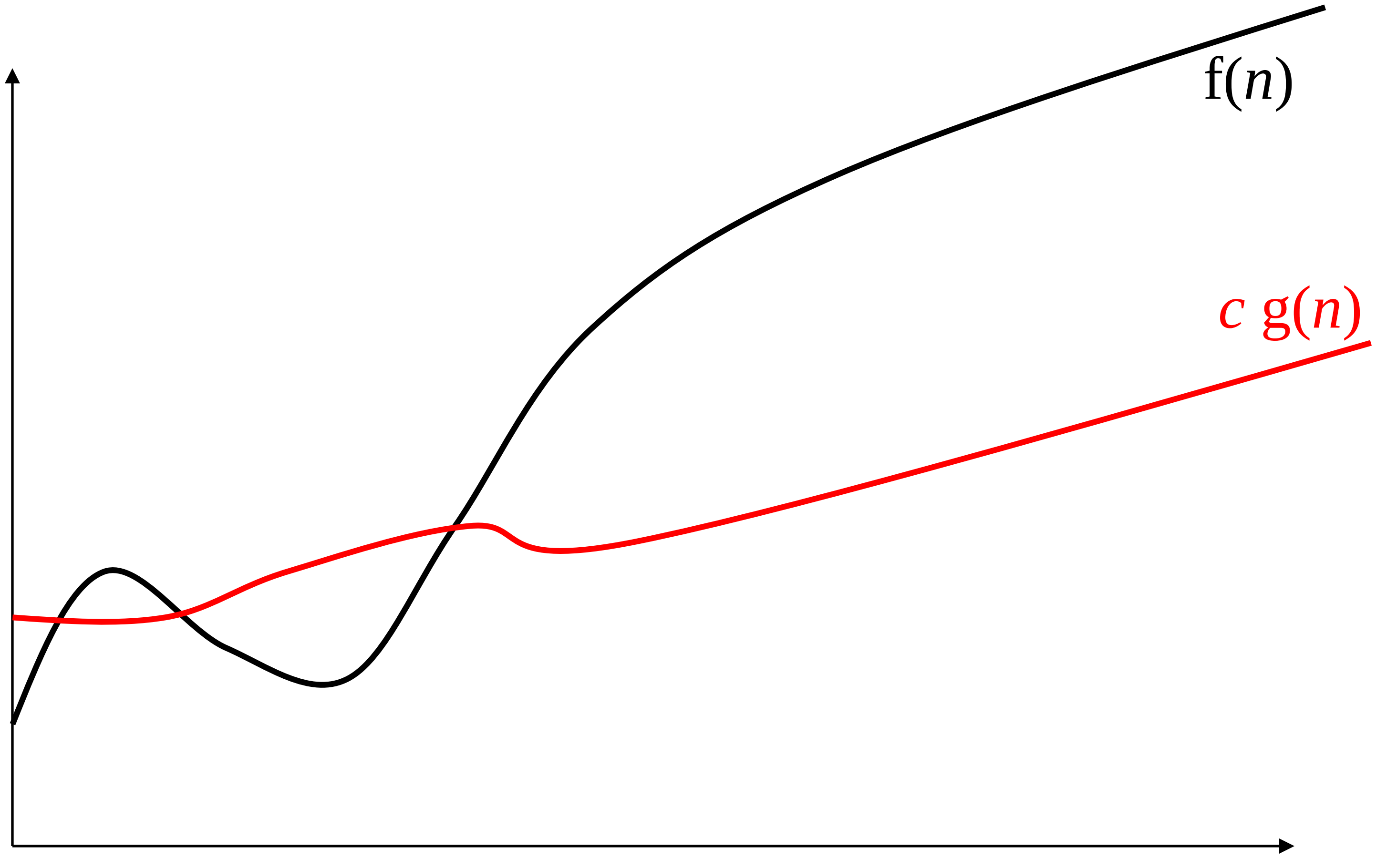
there exists a real constant $c > 0$

and an integer constant $n_0 > 0$

such that $f(n) \geq c \cdot g(n)$ for all $n \geq n_0$.

IN: non-negative integers
IR: real numbers

f(n) is Ω(g(n))



Big-Omega Notation

Let f: IN→IR and g: IN→IR.

f(n) is Ω(g(n))

if and only if

g(n) is O(f(n))

IN: non-negative integers
IR: real numbers

Big-Theta Notation

Let f: $\mathbb{N} \rightarrow \mathbb{R}$ and g: $\mathbb{N} \rightarrow \mathbb{R}$.

f(n) is Θ(g(n))

if and only if

f(n) is O(g(n)) and f(n) is Ω(g(n)).

Big-Theta Notation

Let $f: \mathbb{N} \to \mathbb{R}$ and $g: \mathbb{N} \to \mathbb{R}$.

$f(n)$ is $\Theta(g(n))$ if and only if

there exists $c_1, c_2 > 0$ and $n_0 > 0$ such that

$$c_1 g(n) \leq f(n) \leq c_2 g(n)$$

for all $n \geq n_0$.

Little-Oh Notation

Let f: IN→IR and g: IN→IR.

f(n) is o(g(n))

if and only if

for any constant $c > 0$ there is a constant $n_0 > 0$ such that $f(n) \le c \cdot g(n)$ for $n \ge n_0$.

Note: Analogous to "$f(n) < g(n)$".

Little-Oh Notation

Let f: IN→IR and g: IN→IR.

f(n) is o(g(n)) if and only if

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

Ex: $n \log n$ is $o(n^2)$ (Hint: l'Hopital's Rule)

Little-Omega Notation

Let $f: \mathbb{N} \to \mathbb{R}$ and $g: \mathbb{N} \to \mathbb{R}$.

$f(n)$ is $\omega(g(n))$

if and only if

$g(n)$ is $o(f(n))$.

Little-Omega Notation

Let f: IN→IR and g: IN→IR.

f(n) is ω(g(n)) if and only if

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

Ex: $2n^2$ is $\omega(n)$

Notation	Name	Description	Definition	Limit
$f(n) \in O(g(n))$	Big Oh	f is bounded above by a constant factor of g	$\exists c > 0, \exists n_0 > 0$ s.t. $f(n) \leq cg(n), \forall n \geq n_0$	$\lim_{n\to\infty} \frac{f(n)}{g(n)} < \infty$
$f(n) \in o(g(n))$	Little Oh	f is dominated by g asymptotically	$\forall c > 0, \exists n_0 > 0$ s.t. $f(n) \leq cg(n), \forall n \geq n_0$	$\lim_{n\to\infty} \frac{f(n)}{g(n)} = 0$
$f(n) \in \Omega(g(n))$	Big Omega	f is bounded below by a constant factor of g	$\exists c > 0, \exists n_0 > 0$ s.t. $f(n) \geq cg(n), \forall n \geq n_0$	$\lim_{n\to\infty} \frac{f(n)}{g(n)} > 0$
$f(n) \in \omega(g(n))$	Little Omega	f dominates g asymptotically	$\forall c > 0, \exists n_0 > 0$ s.t. $f(n) \geq cg(n), \forall n \geq n_0$	$\lim_{n\to\infty} \frac{f(n)}{g(n)} = \infty$
$f(n) \in \Theta(g(n))$	Big Theta	f is bounded below and above by a constant factor of g	$\exists c_1, c_2 > 0, \exists n_0 > 0$ s.t. $c_1 g(n) \leq f(n) \leq c_2 g(n), \forall n \geq n_0$	$0 < \lim_{n\to\infty} \frac{f(n)}{g(n)} < \infty$