Fall’19 CSCE 629

Analysis of Algorithms

What is an algorithm?

In mathematics and computer science, an **algorithm** (/ˈælgərɪðəm/ (listen)) is a set of instructions, typically to solve a class of problems or perform a computation. Algorithms are unambiguous specifications for performing calculation, data processing, automated reasoning, and other tasks.

Can you name a few algorithms?

Multiplication: back to grade school

- Long multiplication algorithm

```
           2 0 1 9
  ×          8 2 6
  --------------------
         1 2 1 1 4      (2019×6)
         4 0 3 8 0      (2019×20)
  +  1 6 1 5 2 0 0      (2019×800)
  --------------------
     1 6 6 7 6 9 4
```

Unambiguous set of instructions (above),

solving a problem (multiplying two non-negative integers)

Principal questions to ask

0. What problem to solve?

   be as precise as possible

1. Is the algorithm correct?

2. How much resource it costs?

   (time, space, …)

3. Can we do better?

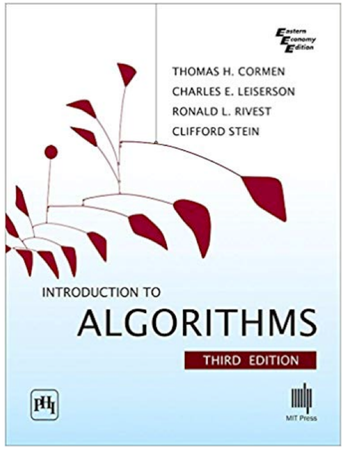
DESIGN

ANALYSIS

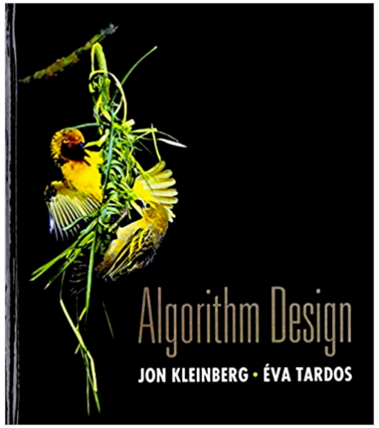
Logistics

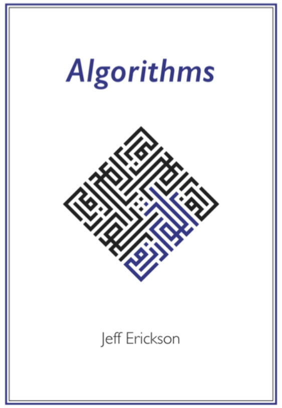
- Instructor: Prof. Fang Song @ HRBB 427B
- Email: fang.song "AT" tamu.edu. Start your subject line with "f19-629". Use Piazza for quick response. Sign up at: piazza.com/tamu/fall2019/csce629602/home
- Lectures: M/W/F 10:20 –11:10am @ HRBB 113
- Office hours: M 1 – 3 pm and by appointment (cancelled today)
- TA: Abhishek Das abkds@tamu.edu; Th 1 - 3 pm @ HRBB 526 (starting week 2)
- Texts: finish reading materials before class

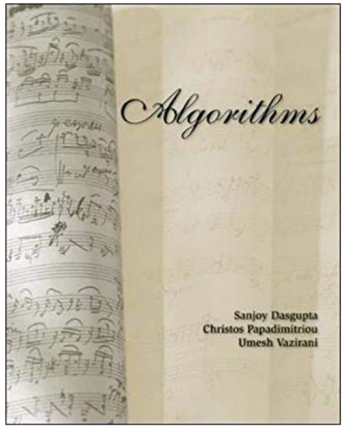
Required: CLRS, 3rd ed.
E-book available at TAMU library



Supplementary







Prerequisite & main topics

- CSCE 411 or equivalent
  - Basic data structures and algorithms: sorting, graph traverse, …
  - Math maturity: basics of combinatorics, linear algebra, probability

Comfortable with READING & WRITING mathematical **proofs**

  - Study review materials and HW 1 to get you up to speed
  - Uncertain? Come talk with me. Not a good idea to take it if not ready
- This course: continuation on advanced materials
  - Mostly standard
  - Selected topics at the end: approximation algorithms, quantum algorithms, …

Policy

- Final Exam: 30%
- Take-home mid-term exam: 25%. Week 7
- Participation: 5%.
  - Quizzes will be posted for some reading materials
- Homework: 40%
  - Weekly. Release Friday & due next Friday. No late homework accepted.
  - Collaboration on homework problems is encouraged, but you must write up solutions entirely on your own and clearly list who you worked with for each problem and any other source you have used other than the text (a person, a book, a research paper, a webpage, etc.).
  - All submissions must be type-set using LaTeX and submitted in PDF format.
  - More specifics on syllabus …

Policy

Academic Integrity

"An Aggie does not lie, cheat, or steal, or tolerate those who do."

http://aggiehonor.tamu.edu

- Academic accommodations
  - Contact me and the Disability Services (http://disability.tamu.edu)

How to succeed in this class?

- Study the reading materials in advance
- Start on assignments early
- Ask *a lot of* questions!
- Form study groups

Announcements

- Wednesday: LaTeX tutorial by Andrew Nemec
- Friday: lecture cancelled (QIP'19 @ Montreal)
- Homework 1 will be posted, stay tuned!

  (course webpage under "schedule")

To-do #0: course webpage

You've probably accomplished it already. Congrats!

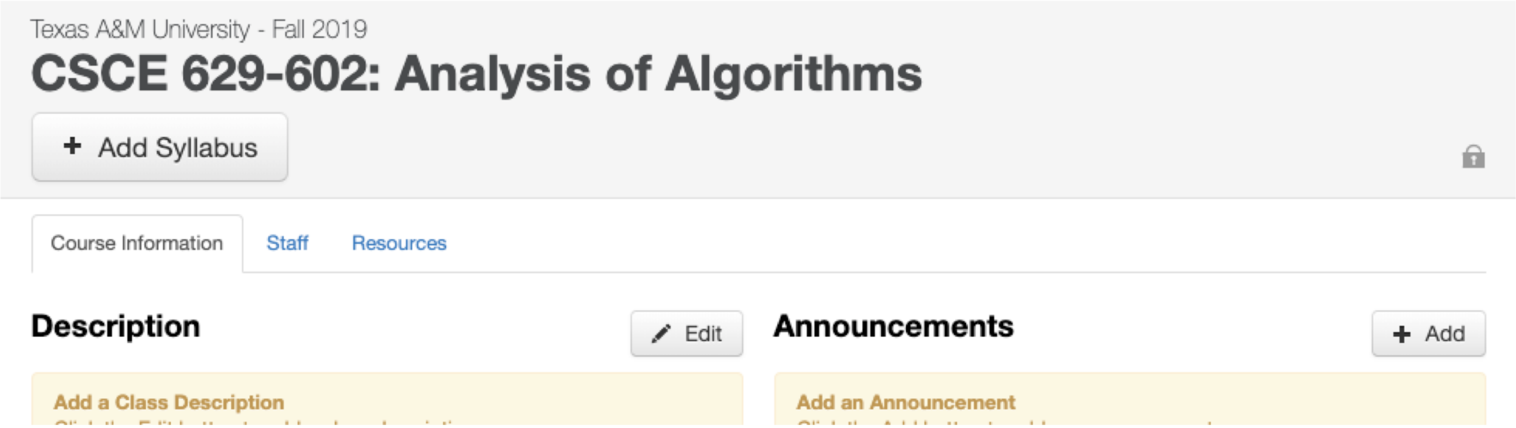
- Familiarize yourself with it
  - https://fangsong.info/teaching/f19_629_alg/

- "Schedule" page
  - Post reading materials and assignments

- "Resource" page contains useful and extended materials

Check regularly!

To-do #1: Piazza



- Enroll on Piazza
  - https://piazza.com/tamu/fall2019/csce629602
  - Will use extensively for discussion and communication
- Post a public note introducing yourself
  - Bio, interest and strength, etc.
- Due Friday August 30, 11:59pm

To-do #2: get to know each other

- It's very helpful to form a study group
- Suggestion: Use W/F lecture time to mingle!
  - In addition to Piazza posts

Now the real stuff

1. Overview of algorithm design techniques

2. Overview of algorithm analysis

3. Growth of functions, asymptotic notations

Algorithmic techniques

- ☆Reduction: a general principle
  - Convert a problem to something you already know how to solve
- Brute force
- Divide and conquer
  - Decompose a problem into smaller sub-problems and compose the solutions
- Dynamic programming
  - Memorize solutions to sub-problems that occur repeatedly. Trace space for time.
- Greediness
  - Make a local optimal choice for subproblems
- Randomization
- … creativity

Algorithm analysis

- Correctness
  - Specify and check pre-conditions P & post-conditions Q for each procedure (esp. recursive ones). $(P \,\&\, P \to Q) \Rightarrow Q$
  - Loop invariants for iterative algorithms. Argue invariants hold at loop entry and loop exist, and use invariants to infer correctness.
  - Termination in finite steps. Some non-negative measure of the alg. decreases.
- Resource analysis
  - Recurrences. $T(n) = 2T(n/2) + 3n$. Recursion tree + induction, master theorem
  - Amortized analysis. Average of a sequence of operations < worst single op.
  - Probabilistic analysis.
  - Experimentation.
- Model of computation
  - E.x. Random-access machine (RAM). Unit cost per instruction and memory access.

Asymptotic notations

- $O(\cdot), \Omega(\cdot), \Theta(\cdot), o(\cdot), \omega(\cdot)$
  - Characterize alg. behaviors (rep. by functions on integers) as problem size grows.
  - Usually a good indicator of which algorithm is preferable (except for small inputs)
- Defining $O(\cdot)$. Upper bounds

  We write $f(n) = O(g(n))$ if there exist constants $c > 0, n_0 > 0$, such that $0 \le f(n) \le cg(n)$ for all $n \ge n_0$.

- $O(g(n))$ as a set

  $O(g(n)) \coloneqq \{f(n)$: if there exist constants $c > 0, n_0 > 0$, such that $0 \le f(n) \le cg(n)$ for all $n \ge n_0\}$

  - $2n^2 = O(n^3)$. $c = 1, n_0 = 2$. $2n^2 \in O(n^3)$
  - $f(n) = n^3 + O(n^2)$ means $f(n) = n^3 + h(n)$ for some $h(n) \in O(n^2)$
  - "=" not the usual sense.

Sort by asymptotic order of growth

1. $n \log n$
2. $\sqrt{n}$
3. $\log n$
4. n^2
5. 2^n
6. n
7. $n!$
8. $n^{1,000,000}$
9. $n^{1/\log n}$
10. $\log n!$

Ascending order: if f appears before g, then $f = O(g)$; "=" indicates $f = \Theta(g)$

$$9,3,2,6,1 = 10,4,8,5,7$$

Summary

Notation	… means …	Think…	E.g.	Lim $f(n)/g(n)$
$f(n)=O(n)$	$\exists\ c>0, n_0>0, \forall\ n > n_0$: $0 \leq f(n) < cg(n)$	Upper bound	$100n^2 = O(n^3)$	If it exists, it is $< \infty$
$f(n)=\Omega(g(n))$	$\exists\ c>0, n_0>0, \forall\ n > n_0$: $0 \leq cg(n) < f(n)$	Lower bound	$n^{100} = \Omega(2^n)$	If it exists, it is > 0
$f(n)=\Theta(g(n))$	both of the above: $f=\Omega(g)$ **and** $f = O(g)$	Tight bound	$\log(n!) = \Theta(n \log n)$	If it exists, it is > 0 and $< \infty$
$f(n)=o(g(n))$	$\forall\ c>0, n_0>0, \forall\ n > n_0$: $0 \leq f(n) < cg(n)$	Strict upper bound	$n^2 = o(2^n)$	Limit exists, $=0$
$f(n)=\omega(g(n))$	$\forall\ c>0, n_0>0, \forall\ n > n_0$: $0 \leq cg(n) < f(n)$	Strict lower bound	$n^2 = \omega(\log n)$	Limit exists, $=\infty$