CSCE5150
Analysis of Computer Algorithms

Dr. Xuan Guo

1

Plan for Today

- Course Organization
- Course Overview
- Review asymptotic analysis

2

Instructor Information

Dr. Xuan Guo (/ʃuɛn/ /gʊo/)

Contact: Message me on Canvas

Office: NTDP F290

Office hours: Mondays 10:00 – 11:30 AM or Wednesdays by appointment

3

Teaching Assistant Information

TA: Arun Kunwar
Office: E247
Office Hours: Thursdays 1 pm – 4 pm
E-mail: arunkunwar@my.unt.edu

TA: Shiva Ebrahimi
Office: F296
Office Hours: Mondays 8 am - 11 am
E-mail: shivaebrahimi@my.unt.edu

4

Course Webpage and Schedule

Course webpage can be found at: CANVAS

The lecture schedule is:
Thursday 10:00PM - 12:50PM @ NTDP B155

5

Recommended Textbooks

- Cormen, Thomas, Charles Leiserson, Ronald Rivest, and Clifford Stein. **Introduction to Algorithms**. 3rd ed. MIT Press, 2009. ISBN: 9780262033848.

6

Course Outcomes

- know the fundamentals of computer algorithms.
- know how to analyze a computer algorithm.
- know how to frame a problem and specify its solution with an appropriate algorithm.
- have a good understanding of computer programming, data structures, and computer algorithms.
- understand the key ideas behind divide and conquer, greedy algorithm, dynamic programming, graph algorithms, and backtracking.
- understand the theories behind branch and bound, and NP-Completeness.

7

Course Content

Topics include:

- Worst Case Analysis and Growth of Functions (Chapters 1, 2, 3)
- Recurrence Relations (Chapter 4)
- Divide and Conquer (Chapter 4)
- Lower Bounds (Chapter 8)
- Greedy Algorithms (Chapter 16)
- Dynamic Programming (Chapter 15)
- Graph and Network Flows (Chapters 23, 24, 25, 26)
- Approximation Algorithms for NP hard problems, vertex cover, TSP, Set Cover (Chapter 35)
- Introduction to Linear Programming (Chapter 29: tentative)

8

Prerequisites

CSCE 4110, or equivalent. Meanwhile, the following topics will be helpful for this course, Computer Programming (C, C++) and Data Structures.

- You should be familiar with classical data structures and algorithms
  - heaps, trees, graphs, etc.
  - sorting, shortest paths, graph search, etc.
- You should know how to program without help in C++
  - design a program, compile, run experiments, etc.

9

Tentative Schedule

Week #	Description	Reading material	Assignments
1-2	Analysis of algorithms	chap. 1 - 4	
2-3	Divide-and-conquer	chap. 4	HW#1&2
3-4	Greedy Algorithms	chap. 16	HW#3
5-6	Dynamic Programming	chap. 15	HW#4&5
6-8	Graph and Network Flow	chap. 23 - 26	HW#6
9	Midterm Exam		
10-11	Backtracking, Branch-and-bound	chap. 35	HW#7
12-13	NP-completeness, Approximation Algorithms		HW#8
14-15	Probabilistic Algorithms		HW#9
15	Review		HW#10
16	Final Exam Week		

10

Grading

Assignments	30%
Midterm Exam	35%
Final Exam	35%
In-class Quizzes (Optional)	10%

11

Grading scale

Tentative grading scale (based on 100 points)

A 90-100 points

B 80-89 points

C 70-79 points

D 60-69 points

F below points

- No absolute grading scale; appropriate letter grade cutoffs set by instructor at the end of semester.

12

Grading – Assignment (30%)

- There will be at most ten homework assignments
- Assignment grade = average of the N-1 highest-graded homework assignments
  - everybody may drop one
- Assignment is due at the end of the day
- Written and programming exercises (C++)
- All programming will be in C/C++ and must compile on a University Unix/Linux machine. No credit will be given for programs that do not compile.

13

C/C++

- This course does not teach C++ programming
  - You will use C++ to demonstrate your knowledge in this course
- A primer lecture covers:
  - Basics of C++
- Other on-line tutorials
  - online tutorials: http://www.cplusplus.com/

14

Important Notes

- All assignments, shall be turned in electronically using Canvas.
- A late penalty of 10% will be applied to all late assignments for up to 3 calendar days. Assignments that are not turned in 3 days after the due date will not be accepted.
- All holidays and weekends will be counted as calendar days
- Missed assignments will be granted zero points. Illness and severe circumstances may be accommodated only with solid evidence. Email instructor before the due date and provide solid evidence whenever possible.

15

Grading: In-class Quizzes (Optional)

- In-class Quizzes (10%)
  - during class on TBD
  - <= 30 mins
- There will be at most ten quizzes

16

Grading: Exams

- Midterm Exam (35%)
  - during class on TBD, most likely the 8th week
- Final (35%)
  - Date and time: TBD, most likely 8:00 am – 10:00 am, May 9

17

Academic Integrity

Academic Integrity is defined in the UNT Policy on Student Standards for Academic Integrity. Any suspected case of Academic Dishonesty will be handled in accordance with the University Policy and procedures. Possible academic penalties range from a verbal or written admonition to a grade of F in the course. Further sanctions may apply to incidents involving major violations. You will find the policy and procedures at: http://vpaa.unt.edu/academic-integrity.htm.

18

Academic Integrity

- Each topic discussed in class will have associated assignment.
- Students may discuss assignment problems and approaches with each other but must write their solutions individually.
- Students may not copy assignment from any source (e.g., other students, the Internet).
- No collaboration is allowed in quizzes and exams.

19

Religious Observance

In accordance with state law, a Student absent due to the observance of a religious holiday may take examinations or complete assignments scheduled for the days missed, including those missed for travel, within a reasonable time after the absence. Students should notify the instructor in each course of the date of the anticipated absence as early in the semester as possible. Only holidays or holy days observed by a religion whose place of worship is exempt from property taxation under Section 11.20 of the Tax Code may be included. A student who is excused under this provision may not be penalized for the absence, but the instructor may appropriately respond if the student fails satisfactorily to complete the assignment or examination.

Check http://policy.unt.edu/policy/15-2-5 for more information.

20

Disability Accommodations

The University of North Texas makes reasonable academic reasonable accommodation for students with disabilities. Students seeking reasonable accommodation must first register with the Office of Disability Accommodation (ODA) to verify their eligibility. If a disability is verified, the ODA will provide you with a reasonable accommodation letter to be delivered to faculty to begin a private discussion regarding your specific needs in a course. You may request reasonable accommodations at any time; however, ODA notices of reasonable accommodation should be provided as early as possible in the semester to avoid any delay in implementation. Note that students must obtain a new letter of reasonable accommodation for every semester and must meet with each faculty member prior to implementation in each class. Students are strongly encouraged to deliver letters of reasonable accommodation during faculty office hours or by appointment. Faculty members have the authority to ask students to discuss such letters during their designated office hours to protect the privacy of the student. For additional information see the Office of Disability Accommodation website at http://www.unt.edu/oda. You may also contact them by phone at 940.565.4323.

21

Seeking help

- Your first point of contact is to contact our TAs
- Typically, you will contact that for any issues on the Homework assignments and the grading issues of quizzes and exams
- Any clarification on the lecture content should be directed to the course Instructor

22

Questions?

23

Why Learn Algorithms?

- Algorithms are the heart and sole of computing!
- Algorithmic computing now plays a key role in nearly everything – proficiency opens many doors.

- Algorithmic prowess differentiates a true "computer scientist" from a run-of-the-mill "programmer" and provides the foundation for a long-term career in computing that can thrive as technology changes.
- Programming ≠ Algorithmic Problem Solving

24

A Good Algorithm (or Data Structure)…

- Always terminates and produces correct output.
  - A “close enough” answer is sometimes fine.
  - Some types of randomized algorithms can fail, but only with miniscule probability.
- Makes efficient use of computational resources.
  - Minimizes running time, memory usage, processors, bandwidth, power consumed, heat produced.
- Is simple to describe, understand, analyze, implement, and debug.

25

Expressing Algorithms

We need some way to express the sequence of steps comprising an algorithm.

In order of increasing precision, we have English, pseudocode, and real programming languages. Unfortunately, ease of expression moves in the reverse order.

I prefer to describe the ideas of an algorithm in English, moving to pseudocode to clarify sufficiently tricky details of the algorithm.

26

What is this course about?

- Designing fast algorithms
  - Divide and Conquer
  - Greedy
  - Dynamic programming
  - Graph search and traversal
  - Back-tracking
  - Branch-and-bound
- Proving that no fast algorithms are likely possible
  - Reductions & NP-completeness
- What to do if no fast algorithms are likely possible
  - Approximation algorithms
  - Randomized algorithms

27

What is this course about?

- How do we know which paradigm is right for a given problem?
  - A very interesting question!
  - Subject of much ongoing research…
    - Sometimes, you just know it when you see it…
- How do we analyze an algorithm?
  - Proof of correctness
  - Proof of running time
    - We'll try to prove the algorithm is efficient in the worst case
    - In practice, average case matters just as much (or even more)
- The course is theoretical in nature
  - You'll be working with abstract notations, proving correctness of algorithms, analyzing the running time of algorithms, designing new algorithms, and proving complexity results.

28

What is this course about?

- At the very least…
- This will help you prepare for your technical job interview!

- Real Microsoft interview question:
  - Given an array, find the total number of inversions of it.
    - If (i < j) and (A[i] > A[j]), then pair (i, j) is called an inversion of an array A. We need to count all such pairs in the array.
  - Greedy? Divide & conquer? …?

29