The University of British Columbia
Department of Computer Science

Computer Science 420
Advanced Algorithm Design and Analysis

General Information
January 2015

Please read this handout carefully, and save it for future reference.

Instructor: David Kirkpatrick
Office: ICICS/CS X839; Phone: 2-4777
email: kirk@cs.ubc.ca
Course homepage: http://people.cs.ubc.ca/~kirk/cs420/

Meeting times

- Lectures: DMP 301, Tuesday and Thursday 9:30–10:50 am
- Tentative Office Hours (ICICS/CS X839): Tuesday 16:00-17:00, Wednesday 09:00-10:00 and Friday 08:30-09:30, or by appointment.
- Group Office Hour (ICICS/CS 146): Wednesday 15:30-17:00.
- Tentative TA Consultation Sessions (Demco Learning Centre): Monday, 13:30-15:30 (Ehsan) and Tuesday 11:30-13:30 (Alireza)

Course objectives

This course covers advanced topics in algorithm design (including associated data structures) and analysis (including both efficiency and correctness), building on the introductory treatment of the same material provided by CPSC 320. The emphasis is on fundamental problems and the formulation and analysis of their associated algorithms and data structures, including important applications, history, recent developments and open questions.

The objective of the course is to expose students to basic techniques in algorithm design and analysis, fundamental problem domains, and applications. Students completing the course should have acquired an appreciation of issues of algorithm design that arise in both theoretical and applied branches of our discipline, and be prepared to understand and critically evaluate new developments.

Although specific problems (and lectures) cut across several of these dimensions, the material can be classified in terms of the following issues:

- algorithm design techniques: divide and conquer, prune and search, greedy methods, dynamic programming, randomized algorithms, local improvement, on-line algorithms, preprocessing, approximation techniques, heuristic algorithms...
- analysis techniques: worst, average and expected case analysis, recurrences, probabilistic analysis, amortization, competitive analysis, relativistic analysis (reductions), time, space and I/O complexity, resource tradeoffs, approximation bounds...

- advanced data structures: heaps and priority queues, disjoint set structures, hash tables, dynamic data structures, higher dimensional search structures, ...
- intrinsic complexity analysis: models of computation, information theoretic bounds, adversary arguments, hardness/completeness arguments...
- problem settings: searching and sorting, graph algorithms, geometric algorithms, text/string algorithms, number theoretic algorithms, algebraic computation...
- other computational frameworks: restricted domains, parallel computation, distributed computation, error tolerant models, quantum computation...

Student participation in the classroom discussion, and in the choice of topics, is strongly encouraged.

Background

CPSC 320 is a prerequisite. Like CPSC 320 much of the material of this course is formal (mathematical) in nature. The course is intended to be interesting, intellectually challenging and fun. If you have any concerns about your background, please contact the instructor.

Text

Much, but by no means all, of the lecture material will be drawn from the text:

> Kleinberg, J. and Tardos, E., **Algorithm Design**, Addison-Wesley Publishing Company, 2005, ISBN 0-321-29535-8.

Additional material will be drawn from:

> Cormen, T.H., Leiserson, C.E., Rivest, R.L. and Stein, C., **Introduction to Algorithms** (third edition), McGraw Hill, 2009, ISBN 0-262-03384-4.

available on-line through the UBC Library (check out Books 24x7).

We will also make extensive use, including occasional reading assignments, of an excellent collection of very comprehensive course notes (**Erickson'sNotes**) prepared by Jeff Erickson at UIUC (http://www.cs.uiuc.edu/~jeffe/teaching/algorithms/). These resources will be supplemented by material from recent journal/conference papers.

Other good general references include:

- Sanjoy Dasgupta, Christos Papadimitriou and Umesh Vazirani, Algorithms, McGraw Hill Book Company, 2008, ISBN 0-07-352340-2.
- Michael Garey and David Johnson, Computers and Intractability: a Guide to the theory of NP-Completeness, W.H. Freeman & Company, 1979, ISBN 0-7167-1044-5.
- Donald E. Knuth, The Art of Computer Programming, Volume 1, Third edition: Fundamental Algorithms, Addison-Wesley Publishing company, 1997, ISBN 0-201-89683-4.
- Donald E. Knuth, The Art of Computer Programming, Volume 2, Third edition: Seminumerical Algorithms, Addison-Wesley Publishing company, 1998, ISBN 0-201-89684-2.
- Donald E. Knuth, The Art of Computer Programming, Volume 3, Second edition: Sorting and Searching, Addison-Wesley Publishing company, 1998, ISBN 0-201-89685-0.

- Donald E. Knuth, The Art of Computer Programming, Volume 4a: Combinatorial Algorithms, Part 1, Addison-Wesley Publishing company, 2011, ISBN 0-201–3804-8.

You will also find a huge amount of related material (including texts, course notes, etc.) on the web, some of which you might find useful.

Lectures

The primary source of examinable material will be that covered in lectures and assignments. This will be supplemented by assigned readings from the texts and other references/handouts. You should do the assigned readings for each lecture prior to attending that lecture, and then review the readings again, along with the posted lectures slides and any additional notes you take, after the lecture.

Lectures typically begin with a brief review of the previous lecture, both to highlight important issues and to provide a smooth transition to the next topic. While questions are welcome (and encouraged) at any time, this is a particularly good opportunity to stand back and look at "the big picture".

It is assumed that you will attend all lectures; if you are unable to attend a lecture because of sickness or similar reasons, make sure you review the lecture slides promptly, get notes/handouts from a classmate and, after reading these over, bring any questions to the instructor. If you are out of class for an extended period of time because of sickness, see the instructor immediately upon your return in order to determine how best to catch up.

Assignments

There will be regular (a total of 8 is planned; see the course webpage for a tentative schedule) homework assignments, consisting of a mixture of "textbook problems", questions arising from in-class discussions, and some exposure to new material or open problems. Emphasis will be placed on clarity and"style" as well as correctness of solutions; for this reason you should plan on putting about 25% of your effort on assignments into producing a clear writeup. In addition, please note:

- Assignment questions will form the basis of several examination questions. Sample solutions will be posted.
- *Assignments are due at the start of class on the due date.* In order to permit prompt distribution of sample solutions and return of marked assignments, *late assignments will not be accepted*.
- An important aspect of doing assignments in this course is reaching some understanding of *what you know as well as you do not know*. Students are encouraged to recognize and acknowledge their inability to solve certain problems. A clear acknowledgement that you have attempted an assignment question but failed to come up with a complete or satisfactory answer (including a *brief* description of the approach(es) that you have tried and an assessment of why your results are unsatisfactory) will receive 25% (or more) of the full credit for the associated question. Erroneous submissions whose deficiencies are not clearly acknowledged will receive less than 25% credit.
- All assignments *must* begin with a clearly identifiable statement detailing *all* sources of information (beyond the lecture notes and course text) used in preparing the assignment solutions, including other books, web sources and discussions with fellow

students. *If you have used no additional sources this must be explicitly stated.* (Assignments without this statement will not be graded.)

Plagiarism and collaboration

The "default" assumption is that *students will work on assignments independently*. Students who complete assignments with the aid of collaborators or other sources (e.g. other textbooks) *must* (i) acknowledge this fact (including the name(s) of other sources) at the start of their homework submission (see above), (ii) produce an independent writeup (copied submissions are *not* permitted), (iii) be prepared to explain their solutions in further detail, if asked, and (iv) be prepared to have the assignment grade adjusted accordingly. *Collaborating in groups of size greater than three is not permitted.*

Plagiarism (the submission of work of another person as your own) and other antiintellectual behaviour will not be tolerated. Your attention is directed to the "Student Discipline" section of the University Calendar as well as the Department Policy on "Plagiarism and Collaboration", available at

`https://www.cs.ubc.ca/our-department/administration/policies/collaboration`

If you have any questions concerning what constitutes acceptable behaviour, these should be directed to the instructor.

Examinations

There will be *three* 90 minute term examinations (tentatively scheduled for Wednesdays February 04, March 04 and March 25, from 5:30-7:00pm). There is no final examination planned.

If you miss an examination because of sickness or similar reasons, visit the Student Health Service or your physician, and then see the instructor. Do *not* write an examination if there is a medical factor that might impair your performance.

Final grade

The exact formula by which your grade in this course will be calculated will not be determined until the end of the course. However, the following is a *rough* approximation:

- 25% for homework;
- 70% for the term examinations (20% for the first and 25% for each of the remaining two);
- 5% for class participation

Tentative (and somewhat ambitious) lecture outline

Lectures 1–2: Introduction; review (and preview) of basic concepts through a case study: finding extrema of a set (and related questions/issues).

Lecture 3: Dictionary problem, review of general structures; exploiting restricted key spaces. and access patterns, hashing.

Lecture 4: Dictionary problem: x-fast and y-fast tries

Lecture 5: Dictionary problem: hashing; basic assumptions; universal hashing; perfect hashing

Lecture 6: Dictionary problem: construction of universal hash functions; applications (closest pair problem)

Lecture 7: Dictionary problem: exploiting non-uniform access patterns; optimal BSTs

Lecture 8: Dictionary problem: adapting to unknown or changing access patterns; list-structured dictionaries; applications in text compression

Lecture 9: Dictionary problem: adapting to unknown or changing access patterns; tree-structured dictionaries; splay trees

MIDTERM I

Lecture 10: Graphs: representations, review of basic properties and algorithms

Lecture 11: minimum-cost paths; single source

Lecture 12: minimum-cost paths; all pairs

Lecture 13: minimum-cost paths; dynamic programming approaches; applications

Lecture 14: minimum-cost paths; motion planning; robustness

Lecture 15: graph matching

Lecture 16: network flows

MIDTERM II

Lecture 17: Reductions and relative complexity

Lecture 18: P, NP and NP-hardness

Lecture 19: establishing NP-hardness of some familiar problems

Lecture 20: approximation algorithms

Lecture 21: coping with hardness: restricted domains; heuristic algorithms

MIDTERM III

Lectures 22-26: Other topics: algebraic computations; string problems; computational geometry