DS-563 / CS-543: Algorithmic Techniques for Taming Big Data

Boston University

Spring 2024

Course Description: Growing amounts of available data lead to significant challenges in processing them efficiently. In many cases, it is no longer possible to design feasible algorithms that can freely access the entire data set. Instead of that, we often have to resort to techniques that allow for reducing the amount of data such as sampling, sketching, dimensionality reduction, and core sets. Apart from these approaches, the course will also explore scenarios in which large data sets are distributed across several machines or even geographical locations and the goal is to design efficient communication protocols or MapReduce algorithms.

The course will include a final project and programming assignments in which we will explore the performance of our techniques when applied to publicly available data sets.

Instructor: Krzysztof Onak (konak@bu.edu)
Office Hours: Monday 3–5pm, CDS 1443 (or the adjacent purple, northeast corner)

TA: Rathin Desai (rathin@bu.edu)
Office hours: Friday 1–3pm, CDS 15th floor, the yellow, southwest corner

Lectures: Tuesday and Thursday 3:30–4:45pm, CDS 264
Discussion Section B1: Wednesday 1:25–2:15pm, IEC B07
Discussion Section B2: Wednesday 2:30–3:20pm, IEC B07

Learning Objectives

- **Big Data Techniques:** In the classic computing setting, the computer has unconstrained access to the entire data set while it computes a solution to a given problem. The main goal of this course is to develop various ways of thinking about processing big data that address scenarios in which this data access is constrained due to significantly limited computational resources. The content of the course is divided into a few parts, each devoted to a different approach to processing big data. Within each part, we will explore a number of computational scenarios and corresponding algorithms. We will also mention or explore the limitations of many of the techniques.
- **Developing mathematical toolkit:** We will develop and learn a number of mathematical tools, which will be used for analyzing algorithms and developing lower bounds for them. In particular, since many big data algorithms involve randomness, we will introduce many tools for analyzing random variables.

- **Rigorous analysis of algorithms:** Lectures, discussion sections, and homework will encourage students to learn and practice the art of rigorous analysis of algorithms. This involves both using mathematical proofs to prove the correctness and efficiency of algorithm, and constructing counterexamples for failed attempts.
- **Turning theoretical algorithms into working implementations:** We will discuss and explore ways of converting theoretical ideas and algorithms with provable guarantees into efficient implementations. For instance, many theoretical analyses introduce impractically large constant factors. To address this, we will consider running experiments to determine much smaller and more practical constants.

Course Information and Tools

- Course website: `https://onak.pl/ds563` (or `https://onak.pl/cs543`)
- Piazza (announcements and discussions): `https://piazza.com/bu/spring2024/ds563cs543`
- Gradescope code: 6G4V6G

Prerequisites

- **Discrete mathematics, logic, and proofs:** Familiarity with basic discrete mathematics is required. Knowledge of basic logic and ability to conduct mathematical proofs are required. These topics are covered in the DS math sequence (DS-120, DS-121, DS-122) and CS-131, as well as many discrete math textbooks, including Stein, Drysdale, Bogart "Discrete Mathematics for Computer Scientist."
- **Algorithms:** Familiarity with basic topics in algorithms and computation complexity (commensurate with DS-320, CS-330, EC-330, or equivalent) is required. These topics are covered in many textbooks, including Cormen, Leiserson, Rivest, Stein "Introduction to Algorithms," and Kleinberg, Tardos "Algorithm Design."
- **Linear algebra, probability, and calculus:** Familiarity with basics of linear algebra, probability, and calculus is required. These topics are covered in the DS math sequence (DS-120, DS-121, DS-122) and multiple other courses across the BU campus (including CS-132, CS-237, MA-115, MA-242, EK-381). Some ot these topics are covered in textbooks such as Strang "Introduction to Linear Algebra," Lay, Lay, McDonald "Linear Algebra and Its Applications," and Pishro-Nik "Introduction to Probability, Statistics, and Random Processes."
- **Programming:** Fluency with programming and basic data structures is required (commensurate with DS-110, CS-111, EK-125, or equivalent). Any programming language can be used for programming assignments but recommended suitable programming languages include Python, C++, Java, and Rust.

Self–Assessment Questionnaire. Since advanced courses offered through the Faculty of Computing & Data Sciences are meant to be open to students from various disciplines, we provide a questionnaire to assist students with self–assessment and placement. See the appendix.

Course Requirements

Apart from active participation, the the course requirements include theoretical homework problem sets, three experimental programming assignments, and a final project. The overall grade will be based on the following factors:

- class participation: 10%
- theoretical homework: 22.5%
- programming assignments: 22.5%
- project proposal: 5%
- final project: 40%

Class participation. The course requires active class participation. It is important to attend both lectures and discussion sections and talk to other students about any missed material. It is also highly recommended to come to office hours to discuss any material that one finds challenging and to actively participate in Piazza discussions.

In particular, we will experiment this semester with the following approach to "measuring" active participation. Every student receives a card with their name at the beginning of the semester. Cards will be collected in person from students, who contribute to lectures and discussion sections and "virtually" from students who contribute to Piazza discussions. From time to time, perhaps every other week or every week, cards will be returned to students in a lecture or discussion section. We will keep track of how many times we collected cards for each student.

The class participation grade contribution includes 2% for timely signing up for a final project presentation slot.

Programming assignments. The course will feature three programming assignments in which students will implement algorithms covered in class and apply them to data sets of their choice. Collaboration here is not allowed (except for discussing high–level ideas), i.e., students are required to implement algorithms and run experiments on their own.

Final project. Possible final projects ideas include but are not limited to

- implementing an algorithm not covered in class and testing its practical performance on real–world data,
- creating an open–source implementation of one of the algorithms with easy to follow documentation,
- developing a new algorithm with good theoretical or practical guarantees.

The outcome of a project will be a short technical report, describing obtained results and conclusions. As opposed to programming assignments, students are allowed to work in teams of 2 or 3. A list of potential projects topics will be provided, but students are encouraged to develop their own ideas. These projects have to be approved by the instructor.

LaTeX. All submitted materials solutions have to be typed. It is strongly suggested to use LaTeX.

Late submissions. You may submit your homework one day late, but your grade may be reduced by 10%.

Grade cutoffs. I will determine grade cutoffs after all assignments and exams have been graded. Grade cutoffs will take into account my assessment of the difficulty level of the assignments and exams, and my assessment of what is expected for each letter grade.

Code of conduct

Homework collaboration policy. You are allowed to collaborate on your homework with up to three of your classmates. However the assignments you hand in should be written up by yourself and represent your own work and thoughts. In particular, you are allowed to discuss ideas with them in person, but as a rough rule, nobody should leave the room with anything written down. If you really understand the discussion, you should be able to reconstruct it on your own.

Your must list your collaborator's names on the top of your assignment. If you don't work with anyone, you must write "Collaborators: none."

Academic code of conduct. You have to adhere to BU's academic conduct policy:
`https://www.bu.edu/academics/policies/academic-conduct-code/`
Additionally, this webpage has great examples of what is and what is not acceptable:
`https://www.bu.edu/cs/undergraduate/undergraduate-life/academic-integrity/`

Generative AI. Using generative AI tools such as ChatGPT or Google Bard is not allowed for homework, including both theory and programming assignments, with the exception of looking up syntax of the programming language you are using.

It is allowed to use it as a search tool to learn more about subjects being covered. If you use it for your final project, you have to strictly delineate what was contribution was and what was created created using generative tools. The main ideas and conclusions from your project have to be yours and you have to be able to defend them. You also are personally responsible for everything delivered.

See also the CDS GAIA policy:
`https://www.bu.edu/cds-faculty/culture-community/gaia-policy/`

Materials

There is no textbook. A good list of resources on many of the topics covered in this class—including books, surveys, lectures notes, and presentations—can be found at

`https://sublinear.info/index.php?title=Resources`

Lecture recording

This course will not provide lecture or discussion section recordings. However, we may allow some students to record lectures as a disability accommodation. Sharing these recordings without permission of class participants is not allowed and they should be deleted when the course completes.

If this lecture recording policy makes you uncomfortable, please discuss it with the instructor.

Reasonable accommodations

If you are a student with a disability or believe you might have a disability that requires accommodation, please contact the Office for Disability Services at 617-353-3658 or `access@bu.edu`. Please also notify the instructor about any accommodation that you may require as soon as possible. We may not be able to provide some accommodations if we do not learn about them sufficiently early.

Laptop and Cellphone Policy

Using laptops, cellphones, tablets, and other similar electronic devices is generally not allowed. If you want to use your laptop or tablet for taking notes, you have to email a copy of your notes to the TF after the class. You are not allowed to use your device for other purposes, such as replying to emails or browsing the web.

Tentative Schedule

The course will consists of 28 lectures (with two lecture dedicated to final project presentations and discussions), and will cover the following topics that include sampling, sketching, dimensionality reduction techniques, and modern distributed parallel computation.

Lecture	Date	Topics
Section 1: Data projections		
Lecture 1	Jan 18	Course overview. Frequency estimation (CountMin sketch).
Lecture 2	Jan 23	Heavy hitters.
Lecture 3	Jan 25	Estimating the number of distinct elements.
	Jan 29	Due: theoretical homework
Lecture 4 Lecture 5	Jan 30 Feb 1	Adversarially robust streaming algorithms.
Lecture 6	Feb 6	Compressed graph representations with applications (graph sketches).
Lecture 7	Feb 8	Data dimensionality reduction (Johnson–Lindenstrauss Lemma).
	Feb 12	Due: programming assignment
Lecture 8	Feb 13	Data dimensionality reduction for clustering.
Lecture 9 Lecture 10	Feb 15 Feb 20	Finding similar data points (nearest–neighbor search).
	Feb 26	Due: theoretical homework
Section 2: Selection of representative subsets		
Lecture 11	Feb 22	Simple geometric problems. Clustering via core sets.
Lecture 12	Feb 27	Clustering via core sets.
Lecture 13	Feb 29	Diversity maximization via core sets.
	Mar 4	Due: final project proposal
Section 3: Sampling from probability distributions		
Lecture 14	Mar 5	Estimation of distributions and their properties.
Lecture 15	Mar 7	Verification of a distribution's uniformity.
Lecture 16	Mar 19	Verification of other properties. Access methods beyond sampling.
	Mar 20	Due: programming assignment (note Wednesday due date)
Section 4: Querying and sampling subsets of data sets		
Lecture 17	Mar 21	Estimation of data parameters and approximate verification of properties.
Lecture 18	Mar 26	Efficient local sparse graph exploration techniques. Estimating graph parameters.
Section 5: Distributed computation		
Lecture 19	Mar 28	MapReduce and the Massively Parallel Computation model. Sample MPC algorithms.
	Apr 1	Due: theoretical homework
Lecture 20	Apr 2	Clustering on MPC.
Lecture 21 Lecture 22	Apr 4 Apr 9	Graph algorithms on MPC.
Lecture 23	Apr 11	Limitations of distributed algorithms.
Section 6: Closing lectures		
	Apr 15	Due: programming assignment
Lecture 24 Lecture 25	Apr 16 Apr 18	Overview of additional topics not covered in detail.
Lecture 26	Apr 23	The lecturer's favorite open problems.
Lecture 27 Lecture 28	Apr 25 Apr 30	Project presentations and discussions.
	May 9	Due: final project report

Appendix: Self–Assessment Questionnaire

Programming:

- Do you know how to write simple programs?
- Can you write a program that reads a database of dates of birth and salaries represented as a text file, in which each line is of the form "name;date of birth;salary"? The program should create an appropriate representation of the database in memory. Sample input file:

```
Alice;1991-09-01;$123456.78
Bob;1993-09-03;$98765.43
```

- Can you write a program that simulates tossing an unbiased coin 10,000 times and reports 10 most common subsequences of five consecutive coin tosses?

Algorithms:

- What is binary search?
- What are (balanced) binary search trees?
- How much time does it take to sort a sequence of n real numbers?
- What is Depth–First Search and Breadth–First Search?
- What are hash tables?

Mathematics:

- Can you conduct simple mathematical proofs?
- Can you prove that
$$\sum_{i=1}^{n} i^2 = \frac{n(n+1)(2n+1)}{6}$$
for all positive integers n?
- What is the rank of a matrix? What is the rank of this matrix?
$$\begin{pmatrix} 3 & 2 & -1 & 5 & 3 \\ 0 & 11 & 3 & 1 & 1 \\ -6 & 7 & 5 & -9 & -5 \\ 0 & 9 & 4 & -4 & 0 \end{pmatrix}$$
- What is the dot product of two vectors?
- Do you know (and can prove) Markov's inequality?
- Do you know (and can prove) Chebyshev's inequality?
- If X and Y are random real variables, when does $E[X+Y] = E[X] + E[Y]$?
- If X and Y are random real variables, when does $E[X \cdot Y] = E[X] \cdot E[Y]$?