SYLLABUS

****

YEAR COURSE OFFERED: 2023
SEMESTER COURSE OFFERED: Fall
DEPARTMENT: Computer Science
COURSE NUMBER: 4397 (6397 for grad section)
NAME OF COURSE: Big Data Processing Systems
JOINT SECTION: Yes (6397)
FORMAT: Face to face
NAME OF INSTRUCTOR: Carlos Ordonez

****

The information contained in this class syllabus is subject to change without notice. Students are expected to be aware of any additional course policies presented by the instructor during the course.

****

Learning Objectives

The course has two major themes: big data with an algorithm/systems focus. On one hand, students taking this course will learn theory, algorithms and programming mechanisms to store and analyze data at large scale: with block-based algorithms, without memory limitations, exploiting external data structures, file systems, operating system runtime management in data science languages, taking advantage of parallel processing, large RAM and distributed storage. On the other hand, students will learn algorithms to analyze data, going from queries, computing descriptive statistics, to computing fundamental machine learning models and analyzing graphs. Systems Programming angle: students will gain an understanding how algorithms, data structures and mechanisms interact and work. Programming assigments will combine C and C++ (new generation data systems, stream management systems, non-relational data) and Python (the glue language where analytic libraries are assembled together).

Permanent course number

Undergrad section:
This course has been restructured and revised to become a permanent course in 2024, valuable for undergrad students pursuing data science or students wanting to learn more about software infrastructure to analyze data. I propose two possibilities:

1) 4380, which could be interpreted as a continuation of 3380 (but going well beyond relational DBMS and SQL)
2) 4339, which could be interpreted as the undergrad section of 6339

Important observation: COSC3380 is the flagship CS course focusing only on database systems. I revised the catalog description and syllabus many years ago. This is a well established course, which in my opinion every undergrad should take. The advanced course, continuing COSC3380 was COSC6340, which died due to lack of interest of MS students, lack of PhD students taking it and a decreasing trend on "systems" in our department. Therefore, a revision is needed.

Grad section:
Two possibilities:

1) This course can become 6340.. Despite my specialty on SQL, parallel database systems, transaction processing, and in-DBMS analytics an advanced course focusing solely on Database Systems is not viable because of AI and Data Science (Python and non-relational systems).
2) This course would replace 6339, with more "systems" content and less "data analytics" (taught in many other courses). So this course is similar to 6339 as it is taught today (by Edgar Gabriel and Feng Yan).

Therefore, I believe an advanced course combining "big data" and "systems programming" is valuable for our students, where 6339 and 6340 are integrated.

Major Assignments/Exams

- 80%: 2 programming projects using a combination of C, C++ and Python programming languages. The projects will require getting familiarized with the internal subsystems of a modern data system (database, big data), parsing and evaluating queries (SQL) as well as understanding the code evaluation runtime & storage of a data science language (e.g. Python, R). This project will be developed by teams of 2 students.
- 20%: midterm exam covering fundamental theory and systems programming topics. The exam is in-class and individual.

Differences between undergrad and grad sections.

Experience teaching 3380 (undergrad DB systems) and 6340 (grad Database Systems), 3320 (theoretical algorithms and data structures) and 4315/6345 (Programming Languages) shows most undergrad students tend to be better C/C++ developers than grad students. On the other hand, some grad students tend to be better at theory/math concepts and they are generally more agile with Python and Data Science concepts. Therefore, projects for undegrads will have deeper systems programming and less theory (more C/C++), whereas grad students projects will have more theory (math).

Reading

Since processing big data, machine learning, graphs and database systems are dynamic fields, references and textbooks have changed, especially with the current AI revolution. We will use a well-known data mining textbook, whose data processing content still remains relevant. Students will also get familiarized with popular websites of big data and large-scale data processing, some from academia, some from industry. The course will require reading papers from mainstream CS conferences and journals, focusing on large-scale data processing like IEEE Big Data, ACM SIGMOD, Dawak, DOLAP, Distributed and Parallel Databases, Intelligent Information Systems, IEEE TKDE, ACM TKDD. About 25% of these papers are coauthored by the instructor and his students.

Textbook: J. Han and M. Kamber. Data Mining: Concepts and Techniques. Morgan Kaufmann, San Francisco, 2nd edition, 2006.

All papers can be accessed from DBLP, with direct links to the PDF inside UH:
https://dblp.uni-trier.de/.

Open-access versions PDFs of the instructor's papers are available at:
http://www2.cs.uh.edu/~ordonez/co_research_proceedings.html
http://www2.cs.uh.edu/~ordonez/co_research_journal.html

List of topics

The semester has 14 weeks. The following topics take 12 weeks of lectures. The 2 additional weeks are a buffer to make adjustments and the midterm exam (around week 10).

1. New hardware: 64 bit architectures, larger RAM, accelerators, GPU, multicore CPU, non-volatile memory, SSD. Contrasting parallel processing: multi-core versus multi-node architectures
2. Advanced data structures and external algorithms to read and write large files. Algorithms and data layout for relational tables (block-based I/O), text files (stemming, IDF) and big data (logs, csv, JSON). External search and sort algorithms (low memory footprint). Buffer management. Advanced Hashing: functions, distributed data partitioning, managing collisions on I/O blocks.
3. Advanced SQL queries: partitioning tables, recursive queries, aggregate UDFs, OLAP cube functions.
4. Algorithms used in machine learning on large data sets: fast descriptive statistics, classical machine learning (regression, PCA/SVD, HMMs) dense/sparse matrix multiplication, stochastic

gradient descent, data summarization, neural nets, (tensors, basic neural net functions, new net architectures).

5. Graphs: adjacency matrix storage, path problems, transitive closure, graph summarization, clique enumeration.
6. Consistency versus performance: ACID transaction data processing requirements, ETL. Distributed data partitioning, managing collisions on I/O blocks. and how different data systems relax them. Locking vs timestamping.
7. Advanced C and C++ programming: byte-level storage of simple data types, pointer manipulation for large objects, binary I/O, thread synchronization, coding standards, compiling large projects, debugging runtime errors
8. Advanced Python programming: modern libraries like NumPy, PyTorch, Pandas, TensorFlow, Cython
9. Latest trends. File systems (single node, distributed; ext3, ext4, HDFS, Lustre). Common file formats to exchange big data (CSV, JSON), displacing XML. Cloud computing: disaggregated storage, containers, edge computing, embedded systems.

Pre-Requisites and Overlap with other courses

Pre-requisites (undergrad): 3337 (Data Science I), 3360 (Operating Systems) and 3380 (Database Systems). 3360 (OS) can be taken concurrently (talk to instructor).

Pre-requisites (grad): students should have data science and systems programming background: (1) a data science, statistics or data mining course, and (2) advanced programming courses with C or C++, including: Algorithms, Database systems, Networking or Operating Systems. Programming experience with C and C++, SQL and Unix (any Linux distribution, Windows is insufficient unfortunately) is required. But the instructor will give a quick review of advanced C++, Linux and SQL advanced programming aspects.

Course overlap: This course has a minor overlap with most grad courses. The grad course is closest to 6339: 6339 is currently a ``high-level'' and ``tools" course to analyze big data, from a high-level language (SQL, Python) with machine learning models and graphs. In contrast, this new "Systems" course will go deeper into how memory, storage is managed and how ML/graph algorithms are modified to scale to large data size and how they can work in parallel. On the other hand, this course will replace 6340 as it appears in the catalog today. The main difference with 6340 (discontinued) is that all review/fundamental DB topics have been removed

and programming is mostly C++; most fundamental DB topics from 6340 are expected to be covered in 3380 (or equivalent). In short, this new course can become either 6339 or 6340.

Instructor Background and Expertise

Brief academic bio (taken from my web page):

Education:

- B.Sc. in Applied Mathematics (Actuarial Science), UNAM University, Mexico, 1992.
- M.S. in Computer Science, UNAM University, Mexico, 1996.
- Ph.D. in Computer Science, Georgia Institute of Technology, USA, 2000.

Research goal: developing scalable, interoperable, parallel algorithms available as libraries and programs to analyze big data. In my lab we develop programs in C++, Python and SQL, mostly in Unix. I currently work on adapting and optimizing a broad spectrum of algorithms in machine learning and graphs to work in modern data science languages. I conduct research on both multi-node processing with distributed memory/storage, as well as multi-threaded processing on multi-core CPUs and GPUs. I work on many science and engineering problems, but most of my applied research has been in medicine and healthcare.

General policies

- Projects may be individual or team-based, but individual requires instructor permission.
- All questions about grading, programming, exam difficulty will be answered during the 1st week of class.
- We will setup an agile message communication system (e.g. discord, slack), instead of email. We will make an effort to answer within 24 hours.
- Source code quality: TAs and instructor will check code clarity, indentation, comments and overall modularity.
- Academic honesty: Source code will be automatically checked for plagiarism. Source code cannot be shared among teams. Source code obtained from the Internet must be disclosed in documentation.With the growing AI trend, students must disclose using mainstream AI tools like ChatGPT (Microsoft/OpenAI) and Bard (Google).
- Resubmission of programs with errors or incomplete requirements will incur a penalty between 10% and 20%, depending on the magnitude of changes.

- If for any reason a student cannot take the midterm exam on the assigned date, the makeup exam will be an oral exam (short questions) in the instructor office or lab.
- Face to face lectures will not be recorded. Online lectures may be recorded.
- Disruptive noises during class are unacceptable (whispering to next student, phone calls, typing on laptop, eating).
- Arriving late in the classroom is discouraged (i.e. more than 10 mins late). Students are asked to wait until 15 minutes have passed to enter classroom in batch.