COMP 7991/8991: Big Data Computing

Xiaofei Zhang

Spring, 2019

E-mail: xiaofei.zhang@memphis.edu
Office Hours: 10:30-11:30 Tu
Office: DH318

Web: eCourseware
Class Hours: 14:40–16:05 TR
Class Room: DH107

Course Description

This course provides an introduction to big data infrastructure for analytics. The focus is algorithm design and "thinking at scale": we will cover data analytic techniques as applied to text, graphs, and relational data. Most of the course will be taught in a combination of MapReduce and Spark, two representative dataflow abstractions for large-scale data analysis, although we will introduce alternative abstractions such as bulk-synchronous parallel and streaming models as well.

One might break down the "big data" stack in three levels. At the bottom resides the execution infrastructure, which is responsible for coordinating computations across a cluster (examples include MapReduce and Spark). In the middle resides analytics infrastructure, which implements data mining and machine learning algorithms on top of the execution infrastructure (an example would be MLlib in Spark). At the top are the tools data scientists use to generate insights, built on top of the analytics infrastructure. This course focuses on the middle part – by the end of the course, you will be able to implement basic data mining and machine learning algorithms that can operate at scale. Of course, effective algorithm design requires understanding the execution infrastructure and what the algorithms are used for , so we will cover the broader context as well.

Required Materials

- Course notes available on Github, as well as the reading list.

Prerequisites

Prerequisites: Algorithms & Data structures, Database system, Java/Python programming

Course Objectives

1. Understand the landscape of big data computing
2. Be able to implement basic data mining and machine learning algorithms that can operate at scale

Course Structure

Lecture

Week	Topic	Highlights
1	Introduction	What's this course about? Why big data? The datacenter is the computer and other "big ideas" The MapReduce programming model
2,3	MapReduce Algorithm Design	MapReduce physical execution MapReduce design patterns Intermediate aggregation and combiners Partitioning, grouping, sorting, and monoids
4,5	From MapReduce to Spark	Evolution of dataflow abstractions MapReduce, Pig, Dryad, Spark, Flink, etc.
6	Analyzing Text	Language models and machine translation Inverted indexing and search
7	Analyzing Graph	Graph representations Parallel breadth-first search PageRank and random walks Issues and challenges with dataflow abstractions
8,9	Analyzing Relational Data	OLTP vs. OLAP Data warehousing and data lakes ETL SQL-on-Hadoop Relational data processing with MapReduce and Spark Optimizations for relational processing Row vs. column stores Vectorized processing
10,11	Data Mining	Supervised machine learning: binary classification Logistic regression, gradient descent, stochastic gradient descent Production machine learning pipelines Hashing: minhash, random projections, etc. Clustering: k-means, Gaussian mixture models
12	Real-time Data Analytics	Stream processing issues and models Probabilistic data structures Integrating batch and stream processing

Assessments & Grading

7000-level Sections

- 3 assignments: weight 60% (This score will be calculated by the average of the top three scored assignments.)
- 1 final project: weight 25%
- 1 Paper presentation: weight 15%

8000-level Sections

- 4 assignments: weight 60% (This score will be calculated by the average of the top four scored assignments)
- 1 final project: weight 25%
- 1 Paper presentation: weight 15%

Note: The assignments are all hand-on programming tasks. 8000-level sections will have more difficult assignments than the 7000-level sections. In addition, a list of suggested topics for the final project will be posted on the course website. Students enrolled in the 8000-level sections will have to work on more challenging problems for the final project.

Grading Scale

We will calculate final letter grades in two different ways; then each student will receive the higher of the two letter grades. One way is a fixed grading scale, with the following cutoffs:

$A \geq 90\%$ $A- \geq 82\%$ $B+ \geq 74\%$ $B \geq 66\%$ $B- \geq 58\%$ $C+ \geq 50\%$ $C \geq 42\%$

The other way is a curve, with the following percentages of students receiving each grade:

$A : 18\%$ $A- : 18\%$ $B+ : 18\%$ $B : 18\%$ $B- : 18\%$ $C+ : 5\%$ $C : 5\%$

However, we will feel free to give an F to any student who clearly did not put effort into the course (or an A+ to any student with truly exceptional performance).

Course Policies

Plagiarism or cheating behavior in any form is unethical and detrimental to proper education and will not be tolerated. All work submitted by a student (projects, programming assignments, lab assignments, quizzes, tests, etc.) is expected to be a student's own work. The plagiarism is incurred when any part of anybody else's work is passed as your own (no proper credit is listed to the sources in your own work) so the reader is led to believe it is therefore your own effort. Students are allowed and encouraged to discuss with each other and look up resources in the literature, but appropriate references must be included for the materials consulted, and appropriate citations made when the material is taken verbatim.

If plagiarism or cheating occurs, the student will receive a failing grade on the assignment and (at the instructor's discretion) a failing grade in the course. The course instructor may also

decide to forward the incident to the Office of Student Conduct for further disciplinary action. For further information on U of M code of student conduct and academic discipline procedures, please refer to: http://www.memphis.edu/studentconduct/misconduct.htm