DSCI6007: Distributed and Scalable Data Engineering

Instructors:
* Alessandro Gagliardi
* Asim Jalis (Guest Lecturer)
* Mike Tamir

Class Location: 44 Tehama St, 3rd Floor, gU Classroom
Lab Time: 2-4 weekdays
Class Time: 4:00 to 5:20 PM M,T,Th,F
Office Hours: Wednesday by Appointment

Description of the Course

Distributed and Scalable Data Engineering teaches you how to work with distributed systems for collecting and analyzing large amounts of varied data quickly. It introduces a variety of (predominantly open source) data platforms including Postgres, Hadoop, Spark, and Kafka. We will focus on the Lambda Architecture as a method of integrating many of these technologies into an enterprise-level "big data" system.

Prerequisite

DSCI6003: Machine Learning and Data Analysis

Standards

By the end of this course, you will be able to:

- Write complex queries including joins and aggregate functions
- Define a normalized relational data model
- Process data in the cloud (*i.e.* EC2)
- Distribute embarrassingly parallel processing task across a cluster
- Use shell commands to search through files and reveal statistics

- Set up HDFS and move data in and out of HDFS
- Evaluate when it's appropriate to apply Lambda Architecture
- Define a fact-based graph schema
- Process data using Hadoop Map Reduce
- Ingest data into Spark, transform it, and write it back out again
- Use Spark to aggregate & process key-value pair
- Develop file schema and stage data for batch processing in HDFS
- Explain uses and limitations of Hive
- Employ MLlib to predict user behavior
- Develop and implement a DAG for batch processing
- Generalize batch layer methodology to a new problem
- Serve queries to the batch layer using Hive
- Configure and deploy a massively parallel processing RDBMS
- Use SparkSQL to query batch views
- Generalize serving layer methodology to a new problem
- Build a horizontally scalable queueing and streaming system using Kafka
- Deploy micro-batch stream processing for real-time analytics
- Identify use-cases for NoSQL
- Configure HBase to store realtime views
- Develop speed layer for realtime pageviews
- Generalize speed layer methodology to a new problem
- Produce a complete query-able lambda architecture
- Generalize the complete lambda architecture to a new problem

Structure of the Class

A typical class will follow this structure:

- **4:00pm-4:20pm: RAT**
  - The RAT is a readiness assessment to make sure you are prepared for class
  - Students will perform the quiz once individually then once as a team
- **4:20pm-5:20pm: Lecture**
  - Introduce the day's topic
  - Present the material necessary to complete the lab
- **2:00pm-4:00pm (the following day): Lab**
  - Students work on the exercise described in `lab.ipynb` or `lab.md`
  - Students should work on the lab individually ahead of time

RAT

RATs are intended to ensure that students comprehended the material consumed between classes. Students unsure of their comprehension should bring questions

to be addressed before the individual RAT. After each student has answered all the questions on the RAT individually, the class will split into teams, who will then review their answers and attempt to reach consensus. Misunderstandings are often better addressed by peers. It is important that all members of each team understand the solution provided by their team. Finally, the answers to the questions will be gone over by the class, hopefully resolving any final misunderstandings before proceeding with the projects.

Labs and Level Assessments

The RATs are meant to assess the first three levels of Bloom's Taxonomy, namely knowledge, comprehension, and analysis. The labs and level assessments are meant to develop the latter three levels: analysis, synthesis, and evaluation.

Presentations

When time allows, students will present their work to one another before class.

Resources

Much of the curriculum is adapted from Big Data by Nathan Marz with James Warren. Much of the technology has changed since that book was written but the basic principles are the same.

Spark and Hadoop are the technologies we will be using most throughout the course. There are many books on both (including some in our own library) that can help including Hadoop: the Definitive Guide, Learning Spark and Advanced Analytics with Spark. Berkeley's AMPLab also has a good tutorial for Spark.

Other technologies that will be used include PostgreSQL, StarCluster, Avro, Hive, HBase, Kafka, and Redshift.

Grading

Students will be graded according to their mastery of curriculum standards (see above). Mastery is rated on a scale from 1 to 4 (where 0 means not-yet-assessed):

1. Unsatisfactory
2. Needs Improvement
3. Satisfactory
4. Exceeds Expectations

Every student is expected to achieve at least a 3 on all standards.
We will be using Galvanize's Mastery Tracker which can be found at students.galvanize.com.

Course Requirements

Attendance

Students are required to attend every class. It is very important you attend each class. If you cannot, please let us know as early as possible.

Lab Exercises

Participation in and completion of lab exercises is a requirement for this course. Each unit includes exercises to provide practice applying techniques discussed in class and to reveal deficiencies in understanding in preparation for skills tests.

Level Assessments

Four level assessments are designed to assess your ability to generalize the techniques discussed in class to a new problem. In week 2, you will begin to develop this new architecture according to a system of your own design which you will continue to develop throughout the course.

Academic Integrity

As per the University's Academic Integrity Policy and Procedures: > The University expects that all students, graduate and undergraduate, will learn in an environment where they work independently in the pursuit of knowledge, conduct themselves in an honest and ethical manner and respect the intellectual work of others. Each member of the University community has a responsibility to be familiar with the definitions contained in, and adhere to, the Academic Integrity Policy. Students are expected to be honest in their academic work.

Violations of the Academic Integrity Policy include (but are not limited to):

1. **Cheating** – *i.e.* Don't read off of your neighbors exams
2. **Collusion** – Group work is encouraged *exept on evaluative exams*. When working together (on exercises, *etc.*) acknowledgement of collaboration is required.
3. **Plagiarism** – Reusing code presented in labs and lectures is expected, but copying someone else's solution to a problem is a form of plagiarism (even if you change the formatting or variable names).

Students who are dishonest in any class assignment or exam will receive an "F" in this course.

Tentative Schedule

1. Relational Databases and the Cloud
    1. Structured Query Language
    2. Relational Data Model
    3. Cloud Computing
    4. Cluster Computing
2. Files and File Systems
    1. Linux
    2. Hadoop Distributed File System
    3. Intro to Lambda Architecture
    4. Serialization Frameworks and Fact-Based Data Models
3. Map/Reduce and Spark
    1. Project Proposals
    2. Hadoop MapReduce
    3. Intro to Spark
    4. Spark II
4. Batch Layer
    1. Vertical Partitioning with HDFS
    2. Intro to Hive
    3. Distributed Machine Learning
    4. Generating Batch Views
5. Serving Layer
    1. Batch Layer Level Assessment
    2. Advanced Hive
    3. Intro to NoSQL and HBase
    4. Advanced HBase
6. Speed Layer
    1. MPP RDBMS
    2. Spark SQL and the Serving Layer
    3. Serving Layer Level Assessment
    4. Queueing with Kafka
7. Lambda Architecture
    1. Micro-batch Stream Processing with Spark
    2. Generating Realtime Views
    3. Speed Layer Level Asessment
    4. Advanced Lambda Architecture
8. Final: Summative Assessment