PROJECT 1: BIG-DATA CONTENT RETRIEVAL, STORAGE AND ANALYSIS FOUNDATIONS OF DATA-INTENSIVE COMPUTING

Purpose:

1. To understand the components and core technologies related to content retrieval, storage and data-intensive computing analysis
2. To explore designing and implementing data-intensive (Big-data) computing solutions using MapReduce (MR) programming model
3. Using Hadoop 2 for HDFS and MR infrastructure
4. To visualize the data using appropriate tools,
5. Create a detailed documentation of the experience.

Problem Statement:

In order to aggregate interesting data and also to keep up with the "trends" we will aggregate data from Twitter. (Why Twitter?) We will collect Twitter data for different ranges of dates (week-range, month-range). Aggregated raw data needs to be cleansed to some extend before analyzing it using Big-data methods.(i) We want to find out the most trending words.(ii) We are also interested in finding out co-occurring hash tags, (iii) We want to cluster the tweeters by the number of followers they have. We will need three clusters where the average number of followers is low, medium and high respectively, where the actual average value for each cluster will depend on your data size. This information may be used for marketing and for targeting the ads/messages. (iv) For the most popular hash tag, we want to develop a connected network of sender→ receiver pattern (all kinds allowed tweeter→ follower, reply to etc.), label the network for determining the shortest path between nodes of this network (This is a hypothetical problem… hoping we will get something like six degrees (of separation) of Kevin Bacon.)

Preparation:

1. Review your Java/Python language skills by working on the sample application that will be given to you in the recitation. (1 day)
2. Understand the MR model and modeling your data as <key, value> store. (1 day)
3. You must have a clear understanding of a client-server system operation and also three-tier application development. (1 day)
4. Install Hadoop 2 on your machine. And test it out. See the handouts by your TAs for instructions on this.
5. Run the *wordcount* program on the virtual machine/on your laptop and make sure you understand the code for the Mapper, Reducer, and the main MR job configuration. 91 day)
6. Think about the starting point and the algorithms for the solutions for each of the problems mentioned above. (1 day)
7. Explore ways of aggregating twitter data, what topic you want to focus on, try some of the existing APIs for twitter; keep collecting data. This is a first and ongoing step throughout your project duration and beyond. (more than 1 day.. all the time)
8. Think about visualization methods you will use: prototype them. (2 days): JSP, Javascript, Jquery, JSON, Processing, Gephi, Tableau…R.. We prefer web-enabled visualization.

Application Architecture: Shown below: Self-explanatory.

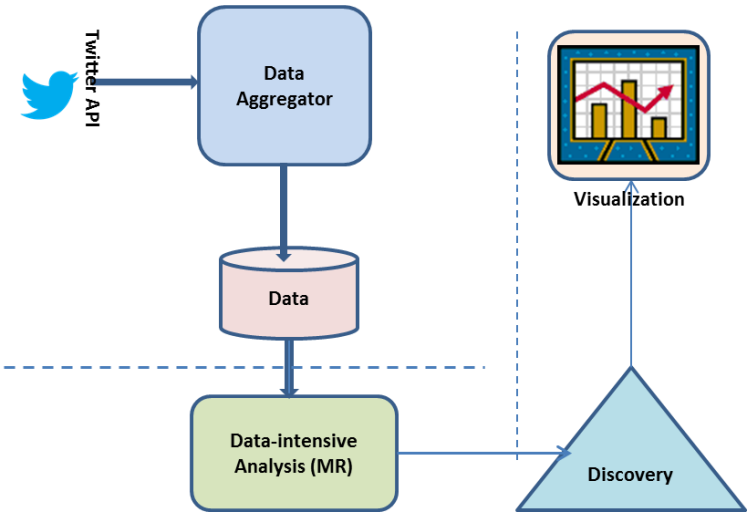

Figure 1: System Architecture for Data-intensive Analyzer

Project Implementation Details and Steps:

1. Data aggregator: Twitter Developer API is available at https://dev.twitter.com/docs You can it to understand the details of what is available. However there are many aggregators already available online. You are allowed to use any of these. (Tweepy; twitter streaming API or Twitter4J)
2. Clean the unwanted details from the tweets and save them in a set of regular files in a designated directory. How many tweets to collect? Initially as you will develop the prototype with smaller Big-data (!) so that you can get the aggregator working without any problem. Then you will scale it up to getting a large set. What is considered a large set? I would say 200,000 – 500,000 tweets, depends on the topic you choose.
3. Also each document may have many tweets.
4. Now design and implement the various MR workflows to extract various information from the data. (i) **simple wordcount** (ii) trends (iii)#tag counts (iv)@xyz counts etc. Once you get the wordcount running, it is just the matter of parsing for the right information.
5. My conjecture is that for twitter data in-mapper combiner will perform better than external combiner. Don't worry about the partitioner.
6. Render the discovered knowledge using appropriate visualization methods/tools.
7. The basic MR algorithms can be improved for performance using the knowledge about the data. For example, typically it is not a best practice to have more than two #tags in a tweet. So once you locate 2 #tags, you can move on to the next tweet.
8. Document the complete process and all the MR source code and the screen shots of visualization.
9. Continue on to the **word-co-occurrence algorithm,** both "pairs" and "stripes" approach.
10. Use each tweet as the context for determining the NEIGHBORS in the algorithm.
11. Next we move on to clustering: use **Mapreduced version of K-means** discussed in [1]. There are other discussion in the literature, If you use some others please cite them in your report.

12. K-means in MR is different from the last two problems since multiple iterations of MR are required and also the state (value) of the centroids will have to be saved between these iterations.
13. You could use "Counters" of Hadoop framework that are actually meant for diagnostics and statistics about the job itself. So don't be very careful not about usage of counters. If misused they may lead to performance issues.
14. The next problem of network creation requires careful design of the <key, value>. For example <node, <fromNodes:cost, toNodes:cost>> etc. Once the nodes have been extracted that represents the networks using MapReduce. You will apply the **MR version of shortest path algorithm** to label the edges of the network/graph. Labeled graph will be output.

Project Deliverables:

1. A report providing all the details of the project:
    a. Data format and source
    b. Amount of data collected and details
    c. Aggregator details, from the scratch or used exiting one, modified etc.
    d. MR (mapper, reducer pseudo code) programs
    e. Configuration of the cluster used (if one was used)
    f. Outputs: different outputs, and visualizations
2. Tar bundle and a README so that I can run the program repeating your MR deployment and execution.
3. Use a similar report format as the Project 1. Report should have pseudo code of all the algorithms used in the format of algorithms discussed in Lin and Dyer's text.

   More details will be given as necessity arises.

Grading: 4 problems 4 X 20 =80, 10 points for the report, 10 points for creativity

Creativity: Besides the basic minimum you can do so much with twitter data. You can use other fields twitter feed offers and do something.

Submission Details: (-50% if you email me the project2.zip)
submit_cse487 files separated by space
submit_cse587 files separated by space

Due date: 4/6/2015 by midnight. Keep submitting parts as and when they are ready. DO NOT wait till the last minute. 3/15, 3/22, 3/29, 4/5 (for the 4 problems, 1 day for readme/report etc. submit on due date)

References

1. G. S. Linoff, Mapreduce and K-means clustering, Data Miner Blog, http://blog.data-miners.com/2008/02/mapreduce-and-k-means-clustering.html, last viewed 2014.