Lessons learned about Hbase

Portland State UNIVERSITY

- HBase stores data as Byte Arrays and itt also sorts the rows lexicographically based on the row key
- Installation of native Hbase on Windows and Mac was challenging for us and also we had to install a whole bunch of softwares like Hadoop, Cygwin, SSH, Java, etc.
- Hortonworks Sandbox requires VM and it needs more than 8 GB RAM. Go for this option if you have the necessary infrastructure to support this
- Google cloud shell on Google Bigtable Console does not support importtsv.
- Connection to Hbase inside the sandbox from Java had challenges identifying correct credentials(host, port for hortonworks sandbox etc.)
- We found Python's Happybase as a good package to talk to Hbase from Python through the thrift server and was easy to use
- All of the above points are with respect to installation and use of Hbase on one system and if anyone would like to use the clustering approach, we haven't explored this part of Hbase.

Lessons Learned

- Start Earlier.
- Don't Assume too much from test data.
- Writing something further down the pipeline does you no good if you don't get that far.
- Figure out what format you want for certain before splitting tasks.
- Algorithms look much more elegant in pseudocode than in real code.
- Don't try to process hundreds of MB of data in couchDB on a seven year old Macbook and expect it to do anything other than get hot.
- Don't use couchDB.

Key Properties of System

Cassandra does not support Distinct, Aggregation, Join queries are not supported in Cassandra.

Write speed of Cassandra is very high. We were able to load 75MB of data in seconds.

Cassandra supports different platforms. We used Python because data manipulation is flexible in Python.

Lessons Learned

- **Use SQL (or a graph database like Neo4j).**
- Creating views takes a long time, which slows the development process.
- Having views can decrease query runtime.
- Views have limitations in functionalities. Graph traversal in CouchDB is flagrantly awful. We basically have two options for making it less painful
  - add all relevant data upon creation, meaning that populating the database also has to figure out co-authors
  - use a different NoSQL database that actually supports graph traversal
- Python is awesome.
  - Extremely rapid development due to dynamic typing.
  - Lots of available libraries and ease of installation - doing `pip install couchdb` worked right out of the box. Additionally, native dictionaries are wonderful when dealing with JSON.

What did we learn, what advice can we give to others?

Don't wait until the last minute to do your project.

Database management is a very complex and open-ended discipline, so it can take work to get the results you are looking for.

Finally, you have to work for what you want. And that could mean learning new things along the way!