Center for
Data Science

Lab 3: Map-Reduce

Lab 3 outline

1. Login to HPC
2. Set up github account permissions
3. Run starter code (word counting) directly and via Hadoop
4. Translate a SQL query to map-reduce
5. Computing item co-occurrence in shopping carts

NYU – HPC Clusters

- HPC Wiki: https://sites.google.com/nyu.edu/nyu-hpc
- HPC offers several clusters for different purposes.
  In this assignment, we'll use Dataproc (Google hosted)
  https://dataproc.hpc.nyu.edu/
- You can log in either via the web interface ("ssh in browser"),
  or using GCloud command-line tools

Setting up GitHub permissions on HPC

- We require using SSH authentication in this class. Do not use HTTPS authentication.
- Follow the steps mentioned in the link: https://docs.github.com/en/authentication/connecting-to-github-with-ssh/generating-a-new-ssh-key-and-adding-it-to-the-ssh-agent
- Add the new keys to your GitHub account by following the link at the end of the page: *Adding a new SSH key to your GitHub account.*

Running map-reduce jobs on your own

- If you want to develop and test on your own machine, you need **mrjob**:
  - **pip install mrjob**

    OR
  - **conda install mrjob**
- **mrjob** can run map-reduce jobs either on a Hadoop cluster or on a single machine.
- Use this to develop and debug!

Using HDFS on Dataproc

Uploading File: **hadoop fs -put <file>**

List files: **hadoop fs -ls**

Remove file (directory): **hadoop fs -rm (-r) <file or directory>**

Retrieve file from HDFS: **hadoop fs -get <file>**
hadoop fs -getmerge <file>

<output-path>

Running the word count demo

- Directly (for development/testing):
  - **cd Lab2/word_count/src/**
  - **python mr_wordcount.py ../book.txt**
- On the cluster:
  - **cd Lab2/**
  - **cd word_count/src/**
  - **bash run_mrjob.sh**
    how it works

← Open this file in an editor to see

- To get the results from HDFS:

  hadoop fs -get word_count

mrjob and Hadoop

- Read the word-count source carefully to see how mrjob works
- Read the shell scripts to see how to execute either locally or by Hadoop
- We provide the basic skeleton for the next parts, but you will need to write the mappers and reducers

First question: translating SQL

- You are given a dataset of movies and a SQL query to translate
- Edit **filter/src/mr_sql.py** to implement map and reduce
- Each call to the mapper will see one line of **movies.csv**
- You need to determine what the intermediate key/value structure is

Second question: item co-occurrence

- Here you are given a collection of grocery purchases
  - `(user id, date, item name)`
  - You can assume (user id, date) uniquely identifies a “basket” or shopping session
- Your task:

  For each item, find the other item that most frequently co-occurs across all shopping baskets

- mrjob allows you to write multi-stage pipelines

  map → reduce → map → reduce → …

Documenting your solution

- In basket/README.md, document your solution
- Describe your algorithm, and provide an analysis of the time and storage used by each step.
- Think carefully about your solution: is it as efficient as you can make it?

Tips

- Your program will produce an output file on HDFS
  - If the file already exists, your program will fail!
  - Get and remove the file between runs of your program
- Develop and test-run locally. MrJob makes this easy
- Use the HPC's job status monitor to track your job progress
  - https://dataproc.hpc.nyu.edu/jobhistory/
- Learn to parse the console output of mrjob and Hadoop