**HOMEWORK 2 – KARAN KADAKIA – UIN: 655641760**

**Explanation of working of project**

* Dblp database is parsed using the project XMLParser and the cs authors present in <inproceedings> and <article> tags are extracted. Cs authors are recognized using prof\_name\_list.txt
* The extracted authors are saved in a .txt file
* This .txt file has all the single author names as well as the authors who have collaborated, but there is no information regarding the count of publications or the collaborations.
* The above count is calculated using the MapRed java project
* In MapRed the input is passed through a Mapper and Reducer and the reducer gives out a <key, value> pair of <author/authors, no of publications>
* Now this output is fed to a python script that converts the output format to DOT format
* This DOT format can be pasted on website: <http://viz-js.com/>. Details are mentioned below.

**How to run the XMLParser**

* dblp.xml and dblp.dtd should be saved in the XMLParser folder in order for the project to run Link: <https://dblp.uni-trier.de/xml/>
* Delete the author\_pairs.txt before running the program
* Import the Java Project in the IDE
* Make sure the prof\_name\_list.txt file(File is present in XMLParser) is saved in the XMLParser folder. This file contains a list of the cs professor which is used to parse the dblp.xml file
* When test cases are run output file changes, so run test before running the actual project and then delete the output file authors.txt
* Run the project
* Output is saved programatically in author\_pairs.txt file

**How to run the MapReduce code**

* Import the folder MapRed in IDE
* Go to > Run configurations
* Provide the location of input and output file location in the Arguments
* Input file is author\_pairs.txt which is present in the folder MapRed
* Run the project
* Output file is generated in a folder named output.txt named "part-r-00000.txt"
* Output file is currently present in the project delete it before running the project

**Example of the output format**

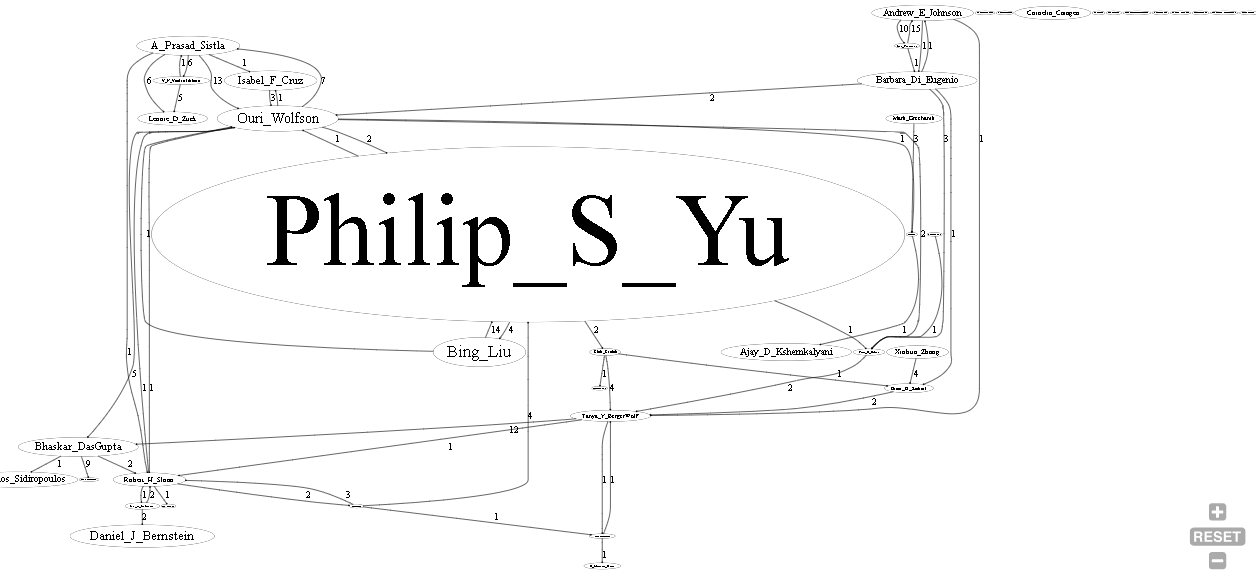
* If the output has single authors without any collaboration then it comes out as below:
* Luc Renambot 15
* If the authors have collaborated then output is:
* Mark Grechanik,Ugo Buy 3
* Actual results attached in the file below:



**How to perform Visualization**

* Open the folder named "Visualization\_code"
* This folder contains the text file named "part-r-00000.txt" which is the output of runnning MapReduce
* This output is converted to a DOT format using the python script named "dotcode\_script.py"
* The output generated by the python script is saved in the file named "di\_output.txt"
* The DOT format output(present in Visualization\_code/di\_output.txt) can be copy pasted on the website http://viz-js.com/ and the visualization can be seen on the right side

**Visualization snapshot**

.

**Conceptual explanation how map reduce works**

Overall steps in map reduce:

1. Map tasks (Splits and Mapping)
2. Reduce tasks (Shuffling, Reducing)

* The Hadoop job client submits the job and configuration to the JobTracker which then assumes the responsibility of distributing the software/configuration to the slaves, scheduling tasks and monitoring them, providing status and diagnostic information to the job-client.
* It is the responsibility of job tracker to coordinate the activity by scheduling tasks to run on different data nodes
* Execution of individual task is then to look after by task tracker, which resides on every data node executing part of the job.
* Framework views the input to the job as a set of <key, value> pairs and produces a set of <key, value> pairs as the output of the job
* A MapReduce job usually splits the input data-set into independent chunks which are processed by the map tasks in a completely parallel manner.
* The number of maps is usually driven by the total size of the inputs, that is, the total number of blocks of the input files.
* Hadoop sorts the framework of the map which are then fed to the the reduce task
* Reducer has 3 primary phases: shuffle, sort and reduce.
* The framework groups Reducer inputs by keys (since different mappers may have output the same key) in this stage.
* The shuffle and sort phases occur simultaneously while map-outputs are being fetched they are merged.
* In this phase the reduce(WritableComparable, Iterable<Writable>, Context) method is called for each <key, (list of values)> pair in the grouped inputs.

**Steps involved in setting up AWS EMR**

* Setup the EC2 security key pair
* S3 bucket setup, contains input file and Mapreduce JAR file, output is created here
* JAR file is created by exporting the project as a runnable JAR file in eclipse IDE
* Create a EMR cluster, steps are: select the s3 bucket, select the EC2 key pair, create cluster, add step, provide input output arguments in the step and run it
* Output can be seen in the s3 bucket