COP5615 – Fall 2019

Project 4.1 – Twitter Engine

Project 4.1 Report

I. Team Members:

- Himanshu Gupta (UFID: 6699-9903)
- Anushri Jain (UFID: 8764-6425)

II. Introduction:

The goal of this project is to implement a Twitter-like engine and a client tester/simulator. The client part (send/receive tweets) and the engine (distribute tweets) were simulated in separate OS processes.

III. Functionalities Implemented:

- Log-in/out
- Create/Delete a account
- Tweets/Re-tweets
- Subscribe/Unsubscribe to an account
- Search by hashtags/mention
- Live push to the connected users
- Additionally, we can also export the social graph of whole network to a .txt file which can be visualized via graphviz or social network visualize.

IV. System Architecture:

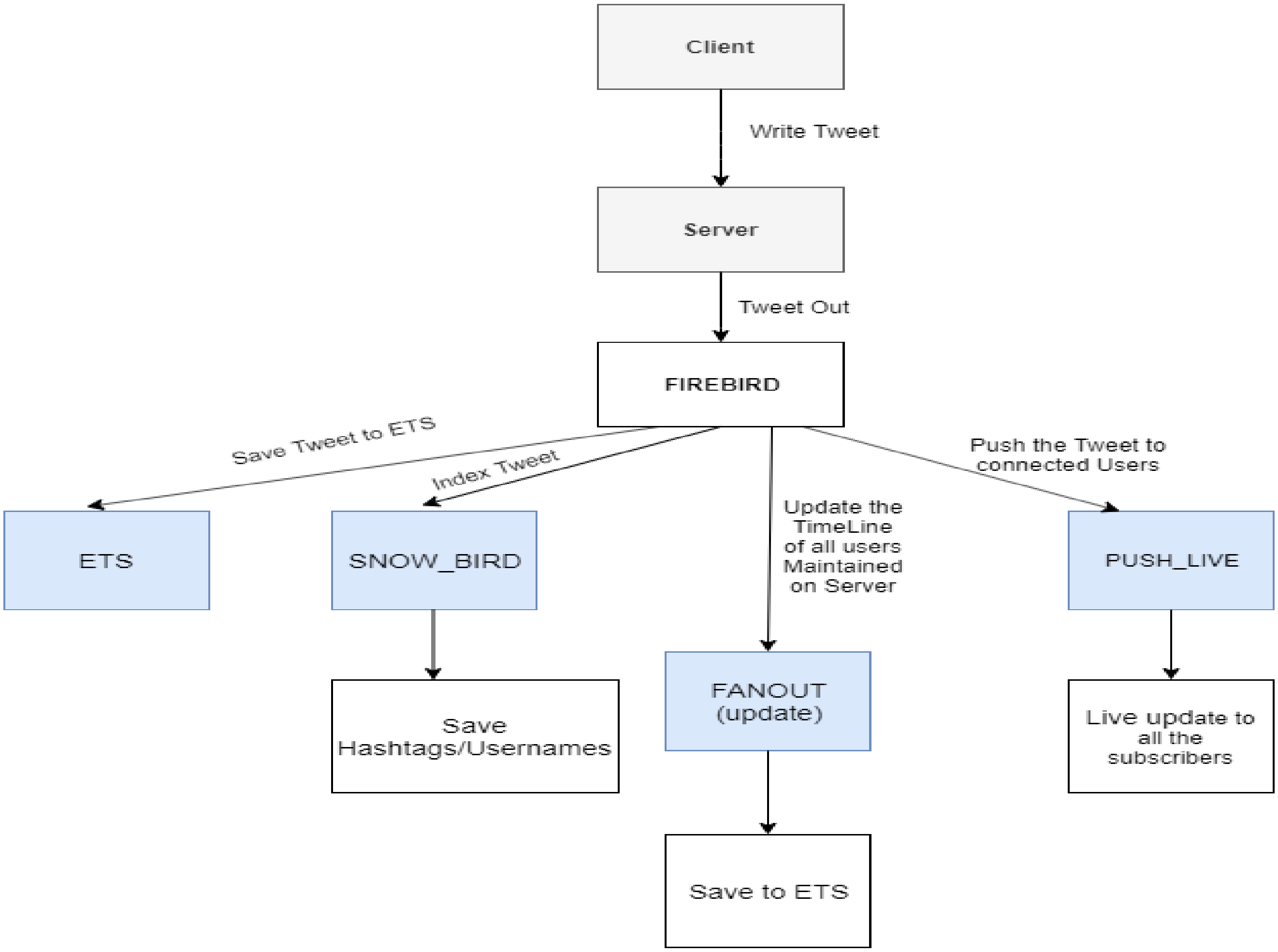
We have the following modules in the system.

- **Server:** This is the server part of our system. It exposes the methods for writing tweets, registering account, search, login, logout etc. The client only interacts with server.
- **Fire Bird:** This exposes the logic for pushing the tweets to live users. Fire Bird also update the timeline of the users.
- **Snow Bird:** This is used for indexing the tweets(by hashtags and mentions). This also helps in searching by hashtags and mentions.

- **Hydrator:** This takes a tweet ID and outputs the tweet presentable to the client. Basically, it adds the data and meta data to tweet id.
- **Timeline Manager:** For every user whether they are online or offline, we maintain a timeline, so that when they login after a period of inactivity, we give them the latest version of the timeline. It consists of various functions to maintain the timeline.
- **Social Graph:** This maintains the list of the user's followers and the followers whom the user is itself following.
- **Accounts:** It maintains the details of the account.

V. Tweet/Re-tweet flow:

The Block Diagram is as follows:



Flow of Tweet/Retweet

When the client writes a tweet or re-tweet, following happens:

- Client calls server with his username and the tweet data.
- The Server will call the Fire Bird module to process the tweet
- Fire Bird Module will perform the following function:
    - It will generate a unique id for tweet and save the tweet in ets table.
    - Fire Bird will invoke the Snow Bird module which will search the tweet for hashtags and mentions and index them
    - Then, Fire Bird will update the timeline of the users which is stored in ets table.
    - Then, the tweet will be pushed to the connected(online) users.

VI. Unit Test Cases:

We have written 18 test cases in total covering all the functionalities. You can run the test by running the following command:

mix test

VII. Steps for running the project:

Run the following command in sequence:

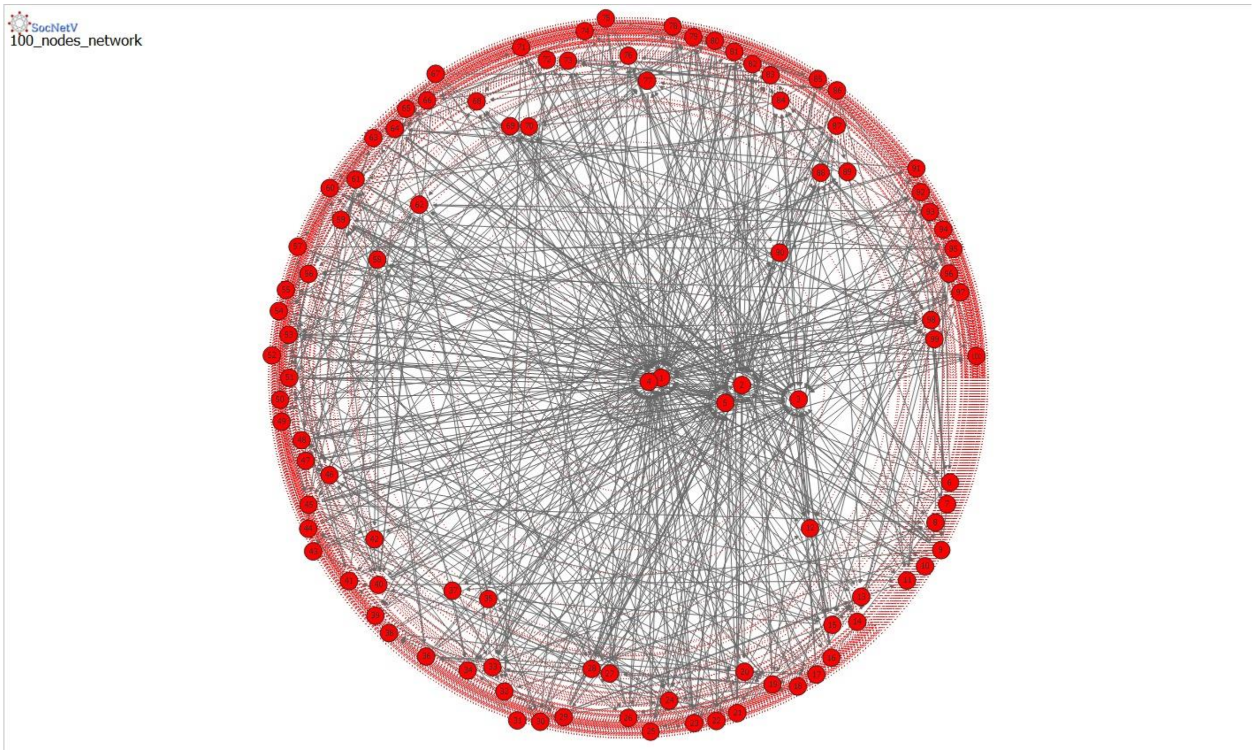
- mix deps.get
- mix compile
- mix run project4.exs <num_clients> <num_tweets>

VIII. Results:

We obtain the following system performance metrics:

(num_Users, num_tweets)	**Total Time taken to converge**	**Avg Number of Tweets Received by Users**	**Avg Time Taken To Tweet/Retweet**	**Avg. Time for Search**
50, 10	2406 ms	47	20 ms	2 ms
100, 10	2484 ms	73	22.134 ms	2 ms
100, 100	28438 ms	552	23.125 ms	2 ms
1000, 10	53734 ms	485	108.855 ms	15.7 ms

IX. Visualization Diagram of 100_nodes_network:


SocNetV
100_nodes_network