GitHub's Big Data Adaptor: An Eclipse Plugin

Ali Sajedi Badashian
University of Alberta
Edmonton, Canada
alisajedi@ualberta.ca

Vraj Shah
Indian Institute of Technology
Indore, India
vraj@ualberta.ca

Eleni Stroulia
University of Alberta
Edmonton, Canada
stroulia@ualberta.ca

ABSTRACT

The data of *GitHub*, the most popular code-sharing platform, fits the characteristics of "big data" (Volume, Variety and Velocity). To facilitate studies on this huge *GitHub* data volume, the GHTorrent web-site publishes a MYSQL dump of (some) *GitHub* data quarterly. Unfortunately, developers using these published data dumps face challenges with respect to the time required to parse and ingest the data, the space required to store it, and the latency of their queries. To help address these challenges, we developed a data adaptor as an Eclipse plugin, which efficiently handles this dump. The plugin offers an interactive interface through which users can explore and select any field in any table. After extracting the data selected by the user, the parser exports it in easy-to-use spreadsheets. We hope that using this plugin will facilitate further studies on the *GitHub* data as a whole.

Keywords

GitHub, Mining software repositories, Eclipse Plugin, Software tools, Big data

1. INTRODUCTION AND MOTIVATION

As the most popular code-sharing platform, *GitHub* is the flagship site for open-source software development and version control, as well as social coding. In recent years, many big companies migrated (parts of) their code base to *GitHub*[7]. The web site includes more than 6 million users and 16 million project repositories. Since *GitHub* enables technical and social interactions among developers, in the last few years it has been the subject of substantial study and research[4, 11, 1, 12, 13].

The data in *GitHub* fits the characteristics of "big data" (Volume, Variety and Velocity). The *GitHub* API offers access to *GitHub* data on demand, but also imposes serious constraints to this access. For example, users are allowed a small quota of API invocations per hour and cannot exceed that limit, or else the server's availability would suffer from excessive numbers of simultaneous requests. Therefore, one can only use the *GitHub* APIs for very limited studies.

To facilitate large-scale studies of *GitHub* and to make dealing with this huge data volume easier, GHTorrent [6] publishes a quarterly MYSQL *GitHub* data dump. However, this is in the form of huge ".sql" file that requires a long time to be imported in MySQL (at least several weeks on a regular desktop, provided that it does not time out or run short of memory or disk space). Moreover, even after the data has been imported, SQL does not always meet the needs of analyzing big data with billions of records. In our previous *GitHub* study on the 2013 version of *GitHub* data set[12], we discovered that regarding data analysis and filtering of this huge data set, SQL is extremely slow or fails to respond. It took us about a week on a regular machine[1] to import less than one third of the data dump[2]. This is why we parsed the data and obtained the necessary data and stored as delimiter-separated formats to be imported by analytical tools like R Studio, SPSS and Microsoft Excel.

Typically, quantitative data analysis of big data sets involves the use of analytical software tools like *R Studio*, *Matlab*, *SPSS* or *Lisrel*. Unfortunately, these tools are incapable of handling data dumps, and instead; they require, as input, tabular (spreadsheet readable) files, containing a record per line, such as ".tsv" and ".csv" files. Once imported in these analytical software systems, various statistical tests and aggregate analyses can be performed. On the other hand, these file formats can also be easily imported in SQL tools, provided that they are not too big, in order to enable the users to query on specific properties and examine their values or to select interesting subsets of the data. Instead of struggling to import the whole data dump to SQL tools, it would be a great relief if researchers could extract the parts of the dump they need, and then import to SQL software.

As a result, the best way to deal with the *GitHub* data is to parse it, which is a tedious task and needs to be aware of the format of the dump, database schema and handling of special characters in the data dump. We built a tool to address exacly this task. It is implemented as an Eclipse plugin, which makes it easy to use and available on demand for the developers. It is dynamic and, based on the input from the user, it can extract all or some parts of the data. It parses the records and converts them to ".csv" and ".tsv" files. Finally, it is fast and efficient. We believe that using this tool will dramatically save developers' time and effort on data preparation in their *GitHub* studies.

[1] A machine with Core i7 2.0GHz CPU and 8GB Ram

[2] Note that we disabled indexing and foreign key checks and increased cache and buffer sizes to speed up the process.

2. LITERATURE REVIEW

The huge size of the *big data* available on social networks demands new tools, engines and models to facilitate their analysis. Currently, several tools have been developed for analyzing social networks. For example, Facebook Data Infrastructure [10] provides batch-oriented support for its internal applications and external products by performing analytical queries for different users on Facebook data. Another example is Twitter API's Automatic Topic-focused Monitor (ATM) [9], which continuously monitors tweets on different topics in the twitter stream.

Focusing on technical computer-science aspects, there are services and sites that support analytical software-development inquiries. Stack Exchange Data Explorer [5] is an online service, which provides limited service to public user's queries regarding Stack Overflow data set. Codebook is a social-networking web service which is made to find connections between developers [2, 3]. It helps developers gain information about other teams and projects. CVSgrab [15] is yet another tool to visualize and analyze the evolution of information in software projects, including developers' joining/leaving the projects and comparison of file associations with developers.

The problem with most of these services is that they gather data in the form of limited meta data. They are incapable of providing the whole data set (or a huge selection) at once as it is needed for most quantitative analysis software and tools like *R Studio*.

As a repository of repositories [14, 7], *GitHub* enables the researchers to a wide range of analytical approaches and tools. To date, the most agile tool that facilitates studies on *GitHub* is LeanGHTorrent [7], which provides direct access to the *GitHub* data. This online service facilitates access to the (up to two weeks up-to-date) data set with a queryable interface. It enforces limitations on the number of repositories and query-result size (which usually leads to pagination). In other words, it enables one to collect limited information on a few projects. For the whole *GitHub* data set, however, it is recommended to use the GHTorrent offline mirrors [6] that are in the form of huge SQL data dumps published quarterly. While conducting our own studies on *GitHub* [12, 13], we discovered that researchers face many challenges – including time to extract the data, extensive effort and huge required space– while working with this data set.

All in all, despite the usefulness of the previous services in the area of software-repository mining, for the purpose of data browsing and validation of specific hypotheses, there is no support for studies aiming at analyzing substantial *GitHub*-data subsets for the purpose of correlation and regression testing in R, or, t-test in Matlab. Our tool fills this gap by enabling access to selective parts of the data set, with offline speed. These selective parts which are way smaller in size, can then be imported easily in SQL tools for querying.

3. THE *GitHub* DATA ADAPTOR

The *GitHub* Data Adaptor is an Eclipse plugin that imports the *GitHub* data dump; parses the data-dump records; extracts the fields dynamically selected by the user from the relevant tables; and exports the results in ".tsv" or ".csv" files. Its parser successfully handles all the special cases like special characters in the textual fields or *Null* values. In this section, we describe the tool in some detail.

3.1 Data Model and the Plugin Back-end

The data model currently includes 22 connected tables. In short, it contains information regarding the following items:

- Commits (commits, commit comments, commit parents)
- Issues (issues, issue comments, issue events and issue labels)
- Organization members
- Projects (projects, project members, project watchers, project forks, project commits, project labels and project milestones)
- Pull requests (pull requests, pull request comments, pull request commits and pull request history)
- Users (user and followers)

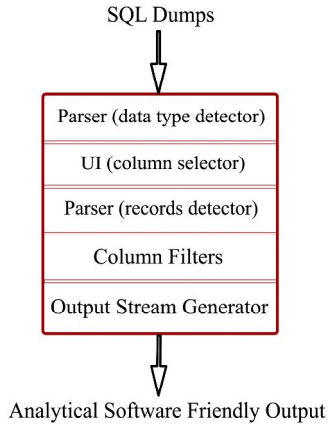

Figure 1: The high level model of the tool

The general model of our tool is shown in Figure 1. It includes five main components;

1. The parser detects all the ".SQL" files available in the input folder and reads their headers to identify available tables and their columns.
2. The UI provides these table names and their columns to the user and the user selects the needed ones. The user also determines the delimiter for the columns in the output file.
3. The parser then starts reading the data dump again. It extracts all the available records and their columns.
4. While extracting the records, it applies column projection (based on the specified columns in the second step) to reduce the size and get rid of unnecessary fields.
5. The output stream generator writes them in output, separated by the specified delimiter.

In fact, this is the process how the tool performs a projection on the columns in the tables of the data dump, and. makes it readable for analytical tools.

3.2 Installation and User Interaction

In order to install the plugin, first, download the "jar" file from the online repository [3] and put it in *plugins* and *features* folder in the "Eclipse" path (and restart Eclipse if it is running). A menu item called "*GitHub* Dump" will be added to the Eclipse menu bar (shown with a red ellipse in Figure 2), which is the plugin trigger. By clicking on this menu item, a three-page wizard appears.

The plugin user-interaction is guided by a three-step wizard. **(step 1)** After obtaining from the user the path in the

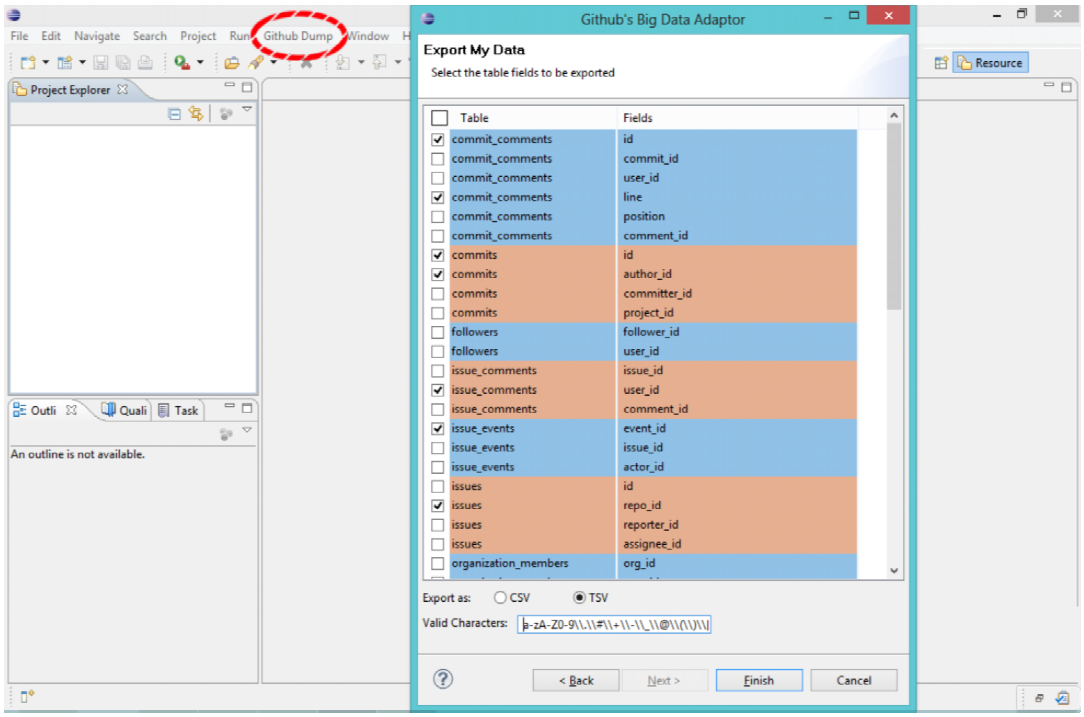

Figure 2: The main plugin page

local file system where the input data dump can be found, it automatically detects all the available tables and their fields and shows them in the second page which is shown in Figure 2. Anticipating the possibility that the tables included in the data dump may change –the providers may, in the future. add additional information, for example regarding *code branches*– our plugin is capable of dynamically detecting the tables.

(step 2) In the next step, the user can select the desired fields from different tables to be extracted. There are options like *select all / deselect all*, the output format and valid characters to be read. Then, after hitting "Finish", the extracting process begins. This process may take from seconds for extracting a few columns from a few tables, up to a couple of hours for extracting the whole data set. This time is not comparable with huge amounts of time needed for SQL manipulation of the data set (even with the first step which is SQL importing, and, as discussed previously may take several days for a regular machine).

(step 3) Then, in order to facilitate the tracking of the progress, the third window shows the progress and summary of the extracted data (Figure 3).

The source code of the plugin is available in our online repository [3], with detailed instructions on how to install and run the program. In addition, a screencast of its usage is available at [4]

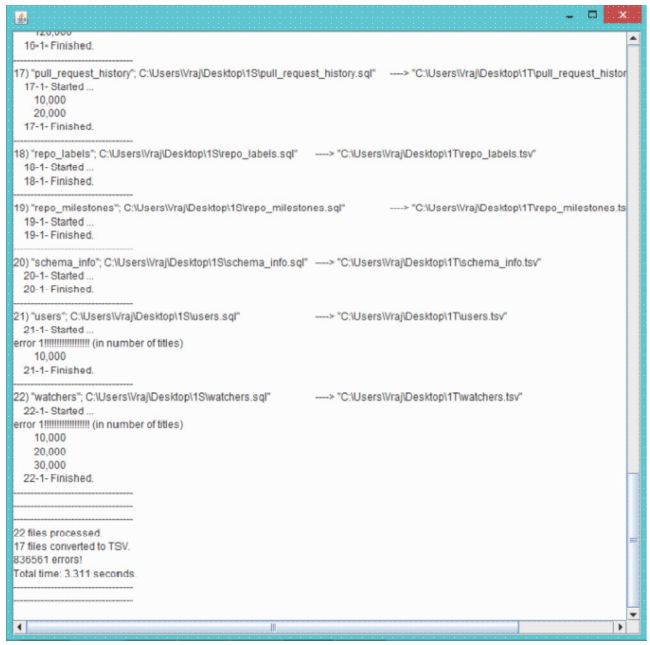

Figure 3: The results summary page

4. LIMITATIONS

There most important limitation to the usage of our tool is that it is based on the quarterly published GHTorrent data set [6] and does not contain all the *GitHub* contributions to date. When the complete *GitHub* data set (or huge parts of it) is required, the GHTorrent data set is the most convenient way to accomplish this goal, sacrificing up to three months of recent data.

Then, in order to run the plugin, first we need to download the desired data dump from the GHTorrent website in local storage and split it to separate dumps. Currently we

[3]https://github.com/pvn25/vrajplugin
[4]http://youtu.be/XBZbrG5oUHQ

are addressing this limitation using MySQLDumpSplitter[5], which is another Java open-source program. We are currently integrating this program within our plugin.

Finally, our plugin does not support querying of rows; it just enable the users to eliminate extra columns through projections, and, thus produce much smaller data sets containing only the needed columns. Supporting users' queries on selecting the rows is in our plans for next version. Non-SQL approaches [8] may be adopted for this manner. However, making it queryable may need higher levels of Ram, but the tool in the current stage does not have specific requirements on Ram or CPU.

5. CONCLUSIONS AND FUTURE WORK

Our plugin facilitates studies of *GitHub*; since the SQL dumps that are published quarterly are huge (more than 100GB), we developed a convenient tool for manipulating these data sets. Using our plugin helps users avoid spending days on importing MySQL or manually parsing the new data set. It provides the offline, flexible and scalable mirror of the data under their control where they write code (*e.g.,* in Eclipse).

We have developed an Eclipse plugin that provides detailed, selective access to *GitHub* data quarterly published as huge GHTorrent dumps [6]. The tool is friendly, fast and efficient; it generates the transformed results in a few minutes for a small number of tables, up to a couple of hours for extracting all tables. Our previous experience on different *GitHub* data sets [12, 13] shows that this is a great time saving for further research on *GitHub*.

The features under construction are optional filters, joining two of the tables (e.g., users and projects), counting or summing values (e.g., number of watchers per project), extracting limited number of records and the capability of renaming the column titles. In addition, online run-time storage space utilization can eliminate additional step of splitting the original dump. Finally, we plan to run a user study to evaluate the popularity, usefulness and adoption of our plugin between *GitHub* researchers.

6. REFERENCES

[1] K. Aggarwal, A. Hindle, and E. Stroulia. Co-evolution of project documentation and popularity within github. In *Proceedings of the 11th Working Conference on Mining Software Repositories*, pages 360–363. ACM, 2014.

[2] A. Begel and R. DeLine. Codebook: Social networking over code. In *Software Engineering-Companion Volume, 2009. ICSE-Companion 2009. 31st International Conference on*, pages 263–266. IEEE, 2009.

[3] A. Begel, Y. P. Khoo, and T. Zimmermann. Codebook: discovering and exploiting relationships in software repositories. In *Software Engineering, 2010 ACM/IEEE 32nd International Conference on*, volume 1, pages 125–134. IEEE, 2010.

[5]https://github.com/verace/mysqldumpsplitter

[4] C. Casalnuovo, P. Devanbu, A. Oliveira, V. Filkov, and B. Ray. Assert use in github projects. In *Proceedings of the 2015 International Conference on Software Engineering*, 2015.

[5] S. Exchange. Stack exchange data explorer. `http://data.stackexchange.com/stackoverflow/query/new`.

[6] G. Gousios. The ghtorent dataset and tool suite. In *Proceedings of the 10th Working Conference on Mining Software Repositories*, pages 233–236. IEEE Press, 2013.

[7] G. Gousios, B. Vasilescu, A. Serebrenik, and A. Zaidman. Lean ghtorrent: Github data on demand. In *Proceedings of the 11th Working Conference on Mining Software Repositories*, pages 384–387. ACM, 2014.

[8] I. Keivanloo, C. Forbes, A. Hmood, M. Erfani, C. Neal, G. Peristerakis, and J. Rilling. A linked data platform for mining software repositories. In *Mining Software Repositories (MSR), 2012 9th IEEE Working Conference on*, pages 32–35. IEEE, 2012.

[9] R. Li, S. Wang, and K. C.-C. Chang. Towards social data platform: Automatic topic-focused monitor for twitter stream. *Proceedings of the VLDB Endowment*, 6(14):1966–1977, 2013.

[10] A. Menon. Big data@ facebook. In *Proceedings of the 2012 workshop on Management of big data systems*, pages 31–32. ACM, 2012.

[11] B. Ray, D. Posnett, V. Filkov, and P. Devanbu. A large scale study of programming languages and code quality in github. In *Proceedings of the 22nd ACM SIGSOFT International Symposium on Foundations of Software Engineering*, pages 155–165. ACM, 2014.

[12] A. Sajedi, A. Esteki, A. GholiPour, A. Hindle, and E. Stroulia. Involvement, contribution and influence in github and stack overflow. In *Proceedings of the 2014 Conference of the Center for Advanced Studies on Collaborative Research*, CASCON '14, Markham, Toronto, Canada, 2014. ACM.

[13] A. Sajedi, A. Hindle, and E. Stroulia. Social bug triaging. In *To be appeared in the 29th IEEE International Conference on Software Maintenance and Evaluation*. ICSME 2015, Bremen, Germany, 2015. IEEE.

[14] S. K. Sowe, L. Angelis, I. Stamelos, and Y. Manolopoulos. Using repository of repositories (rors) to study the growth of f/oss projects: A meta-analysis research approach. In *Open Source Development, Adoption and Innovation*, pages 147–160. Springer, 2007.

[15] L. Voinea and A. Telea. Mining software repositories with cvsgrab. In *Proceedings of the 2006 international workshop on Mining software repositories*, pages 167–168. ACM, 2006.