Improving reproducibility in science by adding provenance tracking to the EcoData Retriever.

Personal Details:-

Name:- Prabh Simran Singh Baweja
3rd Year, B.Tech in Computer Science and Engineering, IIIT Hyderabad, India.
Email Id:- prabhsimransingh.baweja@gmail.com
Website:- web.iiit.ac.in/~prabhsimransingh.baweja
Phone Number:- +91-8790616675.

Abstract:

The EcoData Retriever is a Python based tool for automatically downloading, cleaning up, and restructuring ecological data. Most of the steps are either related to data are either done manually, or using one-off scripts. Due to this, the process is not reproducible. The EcoData Retriever does most of this work, but is not able to keep track of the work that has been completed and therefore fails to support full reproducible workflows.

Technical Details:

The project would be divided in three parts :-

1. **Deciding on the usage of Provenance library or building our own functionality:-**

We will look at various open source provenance libraries, and check if any one of them can be integrated with the EcoData Retriever. If this fails, then we will have to implement the whole thing from scratch using a suitable database. This part of the project would set the environment for the future activities to take place.

2. **Implementation of the first part:-**
   At this stage, we would have a clear picture of what needs to be implemented.

   A). If we are able to find a provenance library that can be integrated, then it's various features need to be taken care of while integration. We can use various indexing techniques to store the pointers to the files that have been downloaded. The file pointers can be sorted according to their names, and then the pointers can point to the original files. This will reduce the time of search when we are check if a new file has already been downloaded or not.

   B.) The second option is to implement it from scratch. In that case, we can design an algorithm which checks if the new file has already been downloaded. If not, this algorithm enables the storage of pointers to the file accordingly. We can use the above indexing techniques while implementing this part as well.

3. **Rerun the exact data retrieval and processing pipeline:-**

The basic docker that has been built will be useful for this part. Whenever we want to reproduce a previous version, this would be helpful to us. This would store all the data of the previous versions so that we can easily access different releases of the code, whenever required.

Schedule of Deliverables:-

Before 27th April:-

1) Get familiar with the code base of EcoData Retriever.
2) Intensive Research on provenance libraries.
3) Discussion about the libraries with the mentors.

28th April - 25th May (Community Bonding Period):-

1) Discussion about the libraries with the mentors.
2) Finalizing the library.
3) If library not found, discussing about the implementation from scratch.

25th May - 7th June (Integration):-

1) Start to work on integration of the library with the Retriever.
2) Decide which indexing technique to use.
3) Decide the features of the provenance library that need to be integrated.
4) If library not found, design an algorithm for the same.

8th June - 12th June (Testing):-

1) Testing the integration that has been done till now.
2) Bugs can be easily detected and rectified in the early stages of development of the code.

13th June - 21st June (Integration):-

1) Integrating the various features that have been decided above.
2) If implementing from scratch, develop the algorithm by adding those features discussed above.

22nd June - 26th June (Testing):-

1) Testing those features on different scripts of EcoData Retriever.

26th June - 5th July (Integration):-

1) On-going integration of the features of provenance library with the Retriever coupled with hashing of the files to reduce the space complexity.

6th July - 19th July (Integration):-

1) Implementing various indexing techniques to store pointers to the files that have performed various tasks.
2) Completing the integration part of the library.

20th July - 26th July (Testing):-

1) Rigorous testing of the code with different scripts.
2) Checking if the library has been integrated accurately or the algorithm functions properly.
3) Checking if the indexing techniques work without any errors.

27th July - 2nd August (Buffer Period):-

1) Correction of errors, if any.
2) Improvisation of the code after careful scrutiny, resulting in smaller time and space complexity.
3) Performing various tests to confirm 100% accuracy.

3rd August - 16th August (Rerun the Retriver):-

1) Storing all the data vis-à-vis a particular version in the docker.
2) Release all the code related to a particular version as and when required.

17th August - 19th August (Testing):-

1) Check carefully if everything you have written works perfectly.
2) In case of errors, solve those bugs.

Future Works:-

The deadlines I have set for myself will be sufficient to complete this project. I am equally eager to contribute to the other projects that EcoData Retriever is working on. I wish to be in a long and creative relationship with Software Carpentry, through constant communication with the mentors, and this project would just be a start to it.

After all, once you start loving the work you do, you no longer feel that you are working.

Open Source Development Experience:-

I have been an active contributor to the open source community for over a year now. I designed a few games and solved some bugs for CodeCombat.

1) http://codecombat.com/play/level/sword-loop.
2) https://github.com/codecombat/codecombat/issues/399

Apart from that, I am an ardent member of the Open Source Sevelopment Group in my university.

Academic Experience:-

I am a third year student at International Institute of Information Technology, Hyderabad, India pursuing my B.Tech in Computer Science and Engineering.

Some of the courses that I have taken are:-

Information Retrieval and Extraction, Data Structures, Database Systems, Algorithms, Statistical Methods in Artificial Intelligence.

Some relevant projects:-

1) **Wikipedia Search Engine:-**

Developed an efficient search engine using Wikipedia data (42 GB). Search results will be obtained in less than 1 sec. Languages Used:- Python, C++. (~1k LOC)

2) **Database Query Execution:-**

Execute a single user DBMS that can execute certain SQL Queries using DBMS page structures and relations.

3) **Payment Gateway Interface:-**

Analyzed requirements, designed schema and created a database using MySQL for an online payment gateway. A query interface was also created using Web2py.

Why this project?

Contributing to the open source community has always been exciting for me. This project attracts me because it provides me a way to use my expertise at an organisational level, having recently worked on developing a search engine for Wikipedia. Given my skillset and previous projects, I find this to be the most relevant and interesting project. I have also started contributing to this project, researching libraries.

Also, the experience of working with this organisation would be invaluable to me in the long run. I hope to be an active member of this organisation even after the termination of this project.

Contributions:-

Check for and use system proxies for downloading files:-

https://github.com/weecology/retriever/pull/278.