Project Proposal
Google Summer of Code 2020

Upgrading the Ganga user interface to use a relational database for persistent storage

CERN-HSF

Mentors:

- Ulrik Egede
- Alex Richards
- Mark Smith

Ratin Kumar
Email: ratin.kumar.2k@gmail.com
Phone: +91-9896225424

Index

Introduction

Personal Information

Full Name	Ratin Kumar
Institute	2nd Year B.Tech Student Computer Engineering National Institute of Technology Kurukshetra
Email	ratin.kumar.2k@gmail.com ratin_11822004@nitkkr.ac.in
Phone	+91-9896225424

Full Name	**Ratin Kumar**
Blog	https://medium.com/@ratin.kumar.2k
Github	https://github.com/DumbMachine
Skype	Ratin Kumar
Timezone	Indian Standard Time (GMT +0530)
Address	208, Hostel-6, National Institute of Technology Kurukshetra, Haryana, India -136119

About Me

I am Ratin Kumar, a 2nd year undergraduate Computer Engineering student at National Institute of Technology Kurukshetra. I have experience in programming with multiple languages such as Python, C/C++, Java, JavaScript, with Python being my favorite. My major interests lie in the field of machine learning. My most significant work to date is a machine learning algorithm that extracts key frames from long videos and uses object locations from each key frame as a means to search frames from database of videos. A research paper detailing the specifics of this approach will be released soon. *Project Autonate* is a tool I have been working on which will provide the world with an open source solution that helps bring automation to the mundane task of data annotation by actively learning from annotated data.

Besides programming, road cycling and stalking stocks are the hobbies that tack up most of my time.

Why GSoC with CERN-HSF?

I have been always passionate about projects which link sciences with programming and this is the main inspiration for me to work with CERN. The idea of getting a chance to help scientists, who are influencing the future, really excites me and also aligns with the ambition to help humanity with the help of science and code.
Being able to analyze and perform operations on data is very important for it to consumable. Thus being a part of Ganga and helping to increase the speed and optimize the operations would be really great.

Preparation I have done:

- Completed the qualification task. The code resides here (I'm aware that this link is currently private, I plan to make the repository public before deadline)
- I have run extensive tests to help me understand the project better and determine best techniques. For more information about it, refer to this respository (I plan to make a edit readme file with details about my experiments and their outcomes/conclusions)

Synopsis:

[About database here]

[About Singularity]

In experiments like LHC(Large Hadron Collider), a very large amount of data (in order of petabytes) is generated. This huge amount of data is then processed using a collection of powerful computers at multiple computing sites by distributing the data in small chunks and processing them individually at remote distributed computing network and then finally collecting the result. These multiple sites are interconnected by a grid. These type of Grids can be accessed by a toolkit called Ganga.

Ganga is an open source iPython based interface tool to the computing grid which leverage the power of distributed computing grid and provide scientists an interface supported by a powerful backend where they can submit their computation intensive programs to Ganga as a batch job. After submitting the job, Ganga processes the program somewhere on the grid, it keeps track of status of the job and after completion of job it gives back output to the user. It can also provide job statistics and job errors, if any.

[Aim of this project]

The aim will be to create a new model, most likely based on a relational database, that can serve as the persistent system for Ganga. It should be able to support caching of information in memory in an efficient way and should scale to the use cases within science over the next decade. A more responsive user interface that can transfer information to and from the persistent storage in a faster and more efficient way. A caching in memory that will allow Ganga to keep a fixed size in memory.

Benefits to Community

This project will replace the old xml-based metadata system with a database, which will be assisted by a caching server, that will allow for a more responsive user interface. The containerized database will also allow for remote access to job metadata, provided user has access to the host of container. (ask for clarification from mentors here). The project will enhance the process of large scale computing of batch jobs at CERN and other similar organizations.

Implementation Details

Project Goals

Objectives

- Add support in *ganga* to save/retrieve job metadata from database instance running inside of a singularity container.
- Add support for usage of cache database, to allow efficient retrieval of data.
- Test the new database-based persistent storage.

Tasks

1. Create database system for use in persistent storage
    - Design the model for data stored in the database.
    - Implement data saving and retrieval.
        - Implement functions to save job metadata in database.
            - Determine the most efficient way of converting job's metadata to string representation, allowing for efficient storage in the database.
            - Develop method to store the obtained string representation and save it in the database, using the database's apis.
            - Integrate the conversion strategy and storing method.
        - Implement functions to retrieve job metadata from database.

       - Determine the most efficient way of converting job's metadata, in string format, back to a ganga job object.
       - Develop method to retrieve job's metdata from database.
       - Integrate the retrieval and type conversion.
   - Test the new model on local backend
     - Test the new database system by performing set conversions and checking for correct output.
     - Perform tests for various corner cases that can arise.
   - Implement error handling mechanism
     - Intentional try to convert corrupted/incorrect job metadata to create proper logs and raise the appropriate errors.
   - Write comprehensive documentation for the code written in Task 1.

2. Create cache support for database
   - Design a model for LRU based cache to store metadata information and prevent trashing.
   - Implement the cache handler in ganga.
   - Test the cache for cases:
     - retrieval of new data
     - retrieval of data present in the cache
     - retrieval of data just removed from cache
   - Test the cache system.
   - Write comprehensive documentation for the code written in Task 2.
3. Combine Task 1 and Task 2
   - Combine database system with cache
   - Host the system using singularity container, to allow easy installation of the system.
   - Test the saving/retrieval of data from ganga process to database instance inside of singularity container.

Deliverables

Timeline

Duration	Task
March 31	**Deadline for submitting Project Proposal**
March 31 - April 27	**Application Review Period**
April 27 - May 18	**Official Community Bonding Period**
May 18 - June 12	**Official Coding Period Start Begin Task 1:**
June 12 - June 15	Time period for any unexpected delay. **Finish Task 1**
June 15 - June 19	**Phase 1 evaluation** Submit git repository of Code with documentation for Task 1
June 19 - July 10	**Begin Task 2 : Implement Active Learning**
July 10 - July 13	Time period for any unexpected delay. **Finish Task 2**
July 13 - July 17	**Phase 2 evaluation** Submit git repository of Code with documentation for Task 2
July 17 - August 7	**Begin Task 3 : Complete Integration**
August 7 - August 10	**Phase 3 evaluation** Submit git repository of Code with documentation for Task 3
August 10 - August 17	**Finish Task 3 Final Submission** Submit git repository of final code with complete documentation.

Additional Information Regarding Timeline

- The timeline gives a rough idea of my planned project work. Best efforts will be made to follow the proposed schedule. I believe that I will be able to achieve all the milestones for this as it aligns with my interest and trying to do something innovative.
- I've no other commitments during summer and hence, will be able to dedicate 48 hours to 60 hours a week. During the last month of the project, my college will begin and I'll be denoting around 28-30 hours a week.
- Each week, time will be divided for implementation of features and documentation of these features. All the features and documentation will be written simultaneously (in the same week). I would like to spend weekdays on implementation and use weekends for testing, writing documentation and researching about the next weeks implementation.
- Weekends will be mostly dedicated to testing, bug fixing, and blog writing. Fortnightly blogs will be maintained at https://medium.com/@ratin.kumar.2k and will include highlights of the development process and also methods used to overcome hurdles.

Working Schedule

I’ll be working full-time on the code on weekdays. On weekends, I’ll be focusing on documentation, testing and bug fixing. My awake hours would usually be in between 6:30 AM IST (1:00 AM UTC) to 2 AM IST the next day (8:30 PM UTC) and I’m comfortable working anytime during this period but I can easily tune my working hours if circumstances ask for it. Except for a few days of traveling (which I’ll be informing in advance to my mentor), I’ll be having no other absences. In case of emergencies, I'll be informing my mentor.

I'll be working from either my house or hostel, both the places have access to good internet. I’m very flexible with my schedule and timezone variation (with my mentor) won’t be an issue. I’m comfortable with any form of communication that suits my mentor.

Future Prospects:

(ask mentors for support here)

If selected I will continue to provide patches, bug fixes and support the project.

References:

This is a work in progress.

1. GANGA Toolkit
2. LHC
3. GANGA GPI
4. Pymongo, Mongodb
5. Redis
6. Singularity