Data Analysis Lab

The purpose of this lab is to provide an insider look in the field of Data Analysis, using tools such as Computing Cluster to work with large sets of experimental data, such as gravitational wave data. The main goal of the lab is to detect a signal injected in Gaussian noise. For this purpose the student will have to make usage of computing resources. The lab yields to a basic understanding of:

- time series data with stationary noise and signals
- statistical concepts (moments, median, higher moments)
- template placement, mismatch statistics, ROC curve
- sensitivity computational cost analysis
- paralelization with a computing cluster using Condor

The practical work is divided into four exercises. A guide complemented with useful links and documentation is given. In order to proceed to the next exercise, the student is asked to demonstrate solutions to short set of problems.

The lab is targeted to Bachelor or Master students in Physics. The student should have knowledge of:

- C programming language
- scripting language (Python is particularly useful)
- basic statistical working knowledge (mean, variance, higher moments), time series analysis

Exercise 2: Using Condor

As you will notice in the next exercises, data analysis is computationally expensive. However, the analysis can be splitted into many independent jobs. We will introduce HTCondor in this exercise. HTCondor is a tool to organise many independent computing jobs using a large number of computers.

In order to use Condor one has to write a so-called submit file, where the next information has to be included:

- *Executable*, the program you want to execute.
- *Universe*, for our purposes only the vanilla universe is needed. You can look for more information related with Condor universes on the guide.
- *Arguments*, command line arguments needed for your executable to run.
- *Output*, the standard output is redirected to this file. That is the output of your program that should appear on your terminal.
- *Input*, standard input will be also in a file. That is the input to your program that you would write on your terminal.
- *Error*, the same with the standard error messages that would appear at your terminal.
- *Log*, file where the logs of Condor are stored.
- *Queue*

You will probably need Condor's manual by side, there are several examples of submission files from page 17. Afterwards, you submit the job to the cluster using the command *condor_submit* followed by the submission file. To check the status of your job you can use the command *condor_q*.

Furthermore, complex jobs usually have dependencies; for instance, computation B could depend on the result of computation A. Condor uses DAGs to describe such jobs. A dag file contains the relation between jobs, in the example above job B would be the children of job A, which would be the parent. In a diamond-shaped dag where jobs B and C depend on the result of job A, while job D depends both on B and C, you would need the following dag file specifying the submission files:

```
Job A A.subfile # A submission file
Job B B.subfile
Job C C.subfile
Job D D.subfile
PARENT A CHILD B C
PARENT B C CHILD D
```

Goal

Learn how to run simple programs in a cluster using Condor.

Guidance

1. Submit a simple job. Write a very simple C or Python program to test Condor. Submit it to the cluster once and learn how to submit it many times.

2. Write a simple diamond-shaped DAG and submit it to the cluster. For example, job A should create two files for jobs B and C Afterwards job B reads B.input and produces B.output, likewise for job C. Finally, job D merges outputs of jobs B and C.

Useful links

`http://research.cs.wisc.edu/htcondor/manual/`