THE UNIVERSITY COLLEGE OF THE CARIBOO
DEPARTMENT OF COMPUTING SCIENCE

WORK TERM REPORT
FALL 2004

Westgrid Job Submission Scripts Manual

Performed at
TRIUMF
Vancouver, BC
by
Daniel Graves

Supervised by
Mina Nozar and
Art Olin

Last Modified Jan 13, 2005

In partial fulfillment of the requirement of the
Bachelor of Science: Computing Science
Co-op Degree Program

Table of Contents

Executive Summary

TRIUMF is a subatomic physics research laboratory where I worked with the project TWIST. There were a variety of tasks I completed. I modified and improved the existing TWIST job submission scripts and added new ones for the super-cluster called Westgrid. I also updated the TWIST job submission script manual to reflect the changes in the improved scripts. The scripts increase the efficiency of TWIST jobs on Westgrid. Submission and resubmission were also made more uniform and more user-friendly. I also did programming work in Fortran and C++ employing high-level emulation to mimic library routines from a software product called hbook for handling histograms. There was also work with databases and dynamic web pages including a shift schedule database, a Westgrid job database, and a Westgrid disk space database. The co-op work-term was enjoyable and an excellent learning experience that I recommend to co-op students interested in physics.

1.0 Introduction

TRIUMF stands for TRI-University Meson Facility and is a subatomic physics research laboratory that employs 5 cyclotrons to create intense particle beams. A cyclotron accelerates particles to extremely high momentum. The largest cyclotron at TRIUMF is the biggest cyclotron in the world. Many scientists come to perform experiments on these beams of very high intensity.

The project TWIST (TRIUMF Weak-Interaction Symmetry Test) is a high-precision experiment that studies the decay of polarized muons. TWIST strives to achieve three to ten times higher precision of the standard model parameters. TWIST requires massive computing power to generate and analyze Monte Carlo simulations and analyze physics data. TWIST uses Linux cluster technology for their computation needs. Much of my work at TWIST involved improving and increasing functionality of the job submission scripts.

Other job duties included work with PostgreSQL databases, dynamic web pages, and C++ and Fortran 90/95 programming. I wrote a number of documents describing the workings of the databases, dynamic web pages, programs, and scripts that I wrote. The main document that I wrote is the TWIST job submission manual and is the document that will be presented in the report. I made two presentations of the new TWIST job submission scripts that I wrote as well.

A description of the work I completed during the co-op work term is the following section: Summary of Job Duties. The section following is the Westgrid job submission manual I wrote to aid users that need to submit jobs to Westgrid. The manual is based on the format of the previous manual for submitting jobs to the Westgrid super-cluster.

2.0 Summary of Job Duties

I've learned a bit about the physics of the experiment and the detector but my tasks have been focused mostly with programming scripts, writing C++ and Fortran90 programs, designing databases, and writing dynamic web pages in PHP (PHP: Hypertext Preprocessor) and PERL (Practical Extraction and Reporting Language) on the Linux platform. A good chunk of my work has been with Westgrid which is a 504 node IBM (International Business Machines Corporation) 3.06Ghz Xeon dual-processor super-cluster. I've completed work on some scripts written in PERL that submit TWIST jobs to the Westgrid super-cluster. These jobs include Monte Carlo generation, analysis of Monte Carlo, and analysis of data sets taken from the detector. The analysis is done using software called MOFIA written by TWIST. MOFIA is based on an older software program called KOFIA developed by TRIUMF.

The changes I've made to the way jobs are submitted to Westgrid allow for more efficient use of the super-cluster computing resources and provide a more user-friendly mechanism for job submission and resubmission than the previous job submission scripts TWIST used. Some of the benefits of the new job submission scripts include increased automation and better handling of failed jobs. An example of this is with automatic checking for jobs that have finished but produced incomplete data. The scripts handle failed jobs better by providing a mechanism to resubmit jobs that either fail or vanish from the job queue. Another benefit is that it can also email the operator using sendmail if a job abnormally terminates or fails for whatever reason. Sendmail is a Linux tool that manages mail between users on different machines. It sends the electronic mail to the appropriate machine in the appropriate user's mail box.

I've also done work on Fortran 90 and C++ programming. TWIST's analysis software called MOFIA included the ability to compute time zero values for each wire in each plane. I wrote a Fortran 90 program based on MOFIA's code that computes these time zero values. This involved using routines from hbook libraries to make histograms for each wire and fit a function to each of these histograms. The next phase was to upgrade to root libraries by writing a high-level emulation library for the hbook routines to link with newer root routines. Root is a graphical data analysis package for analyzing histograms. Root also includes a library of routines for creating, handling and storing histograms and was written in object-oriented C++. hbook is an older library than root for creating, handling and storing histograms. A separate and older graphical analysis package called PAW is used for analyzing hbook histograms. PAW was written in Fortran. Root, hbook, and PAW were all made by CERN (European Council for Nuclear Research).

Some work in databases, web-servers and server processes has been done as well on the Linux platform. This includes using OpenSSL, which is an open-source implementation of SSL (Secure Sockets Layer) in Linux to create keys and certificates for the apache web server.

I also performed PostgreSQL database configuration and syslog configuration for

PostgreSQL logs files. The PostgreSQL database was for a new web and database server posed to replaced the older production web and database server. My work involved configuring and starting up the database and copying the database contents over to the new machine. The procedure for copying the database contents over to the new machine involved doing a full database backup and restore. A full database backup in PostgreSQL is simply an SQL (Structured Query Language) dump of the database contents. The SQL dump can be then used to restore the database on the new machine. I ran into a few problems because of incompatibilities issues in the SQL dumped by the older version of PostgreSQL. As a result I manually changed the dumped SQL script to conform to the newer version of PostgreSQL.

I learned about OpenPBS (Portable Batch System) which was originally developed by NASA (National Aeronautics and Space Administration) for managing parallel computing jobs on a cluster of Linux computers. I used programs provided by OpenPBS in my Westgrid submission scripts to submit jobs and view the status of by debug jobs.

One of the first of many databases I worked on was the shift schedule database which uses PosgreSQL. A web-page interface written in PHP allowed a password-protected shift administrator account to change the shift schedule. It also allows shift takers to view the shift schedule. I have created two other databases for Westgrid using a hash database file. Both of these databases had a PERL website interface. One database summarized the status of the hard disk data partitions on the 504 nodes on the Westgrid super-cluster. It highlights nodes with low free disk space and nodes that are down. A second database summarizes the status of Westgrid jobs categorized by the owner of the jobs. It tells you the number of jobs in the queue, the number of jobs running and the number of nodes currently used by the user. By clicking on the user, a detailed table of all the user's jobs currently running are displayed giving information such as the resources the job is currently using and the host name of the node on which the job is running.

On the physics side, I've done some shift work doing TWIST (TRIUMF Weak Interaction Symmetry Test) runs with a muon beam (a muon is a sub-atomic particle). The runs I've done have mostly been for determining beam characterization. One of the purposes of beam characterization is to reconstruct the beam accurately in Monte Carlo simulation using the GEANT (GEneration ANd Tracking) package. Monte Carlo is a software package for creating simulations of real-world events such as the effects of forest fires or in the context of TWIST, the path particles take through the TWIST detector.

3.0 Westgrid Job Submission Manual

3.1 Overview

The Westgrid job submission manual is located in ~e614/public_html/private/sysman/geant_mofia_setup.sxw on the computer tw04.triumf.ca A pdf version is also available. The URL (Uniform Resource Locator) is http://twist.triumf.ca/private/sysman/westgrid/geant_mofia_setup.pdf

There are three quad-processor machines that control user access to the super-cluster. They are called nunatak1, nunatak2, and nunatak3. Users who wish to run jobs on Westgrid must log onto one of these machines first (e.g. nunatak2.westgrid.ca). All machines have access to a global file system where the e614 home directory (/global/home/e614) is located. The GEANT and/or MOFIA Submission script is located on the e614 home directory in the following location: /global/home/e614/bin/tbsub.pl

tbsub.pl submits jobs by passing environment variables as parameters. It loads the default environment variable parameters from the template file located in /global/home/e614/rundb-dev/exe/e614_EnvTmpFile The environment variable parameters used can be changed by passing arguments directly to tbsub.pl

Email notifications are implemented into tbsub.pl It is possible in certain circumstances for the script running on Westgrid to send a notification email. The email notification system used sendmail on nunatak2.westgrid.ca thus if the network connection between the local node and nunatak2.westgrid.ca is down, email notifications are impossible. Email is currently only used when PBS aborts a job.

The following job directory structure is needed to setup jobs for submission:

/global/home/e614/tbsroot/<jobname>/common/

- <jobname> an arbitrary name for the job
- within /common, you need the following files:

– **e614.com**	--command file
– **gb.sh**	--sets up environment variables and calls 'e614.com'

/global/home/e614/tbsroot/<jobname>/queued/

- this is where all the ffcard files must go
- to create ffcards, use: /global/home/e614/e614soft/triumf/geant/devel/run/mkffcards.pl

How to use 'mkffcards.pl'
From 'mkffcards.pl --help'

```
Usage: mkffcards.pl ffcard-file firstrun nruns
Or:    mkffcards.pl ffcard-file firstrun nruns CARD[INDEX] PARMIN
```

PARMAX ...

Produces files mcrun<N>.ffcards from the given template with appropriate run numbers and random seeds.

If CARD arguments are given, also varies linearly the given parameters between their limits from the first to the last run.

EXAMPLES:

To prepare ffcards for 100 "identical" GEANT runs 6000 to 6099:

```
mkffcards.pl e614.ffcards 6000 100
```

To prepare ffcars files for 11 runs 6100 to 6110 changing gas degrader setting in 10% steps from 0 to 100%:

```
mkffcards.pl e614.ffcards 6100 11 GABS[2] 0. 1.
```

IMPORTANT: the script uses the same format for a variable parameter as the original ffcard file has, including the number of digits after the decimal point. For the example above to work correctly the original e614.ffcards should give GABS[2] with at least one digit after the point (e.g. 0.0, not just 0.). The number from the original file is discarded, but the format is copied.

Westgrid Example:

If the ffcard file (e614.ffcards) is in /global/home/e614/tbsroot/<jobname>, and . Is /global/home/e614/tbsroot/<jobname>/queued then to create 1000 ffcard files starting with run number 8000, execute:

```
'~/e614soft/triumf/geant/devel/run/mkffcards.pl ../e614.ffcards 8000 1000'
```

3.2 Directory Structure

```
/global/home/e614/tbsroot/genXXX/
  |⇒ common
  |⇒ queued
  |⇒ submitted
  |⇒ errorlogs
  |⇒ scripts
  |⇒ running
  |⇒ finished
  |⇒ to_be_deleted_in_future
  \⇒ unreachable
```

The common directory contains files common to a gen or set. This includes environment variable output files which contain the environment variables used by the script and is generated at submission time. One environment variable file for GEANT submission unless the data is a set and one environment variable file for each MOFIA analysis. These files are important for script resubmission because the environment variables in these files will be loaded at resubmission time.

The queued directory contains all the ffcards to be submitted if a GEANT job is selected to be run. All of the ffcards are moved from queued to submitted once the GEANT jobs enter the PBS queue. The submitted directory then contains all the ffcards for GEANT jobs that have been submitted. When the script does a self check of the MC and if the check returns TRUE (meaning the MC is good), then the ffcard for that run in the submitted directory is removed.

The errorlogs directory contains all log files for jobs that failed (log file names are in the format <WG_JobNumber>.log) It also contains log files for jobs where the MC check returned false (log file names are in the format <JobName>.log).

The scripts directory contains all the scripts used to run jobs on Westgrid. These scripts are necessary for resubmission of both GEANT and MOFIA jobs.

The running directory contains a links to the local nodes where a GEANT job is running. If a link exists in this directory and no job is running on PBS (checked by using the PBS qstat -u e614) then tbsub_checkMC_and_resubmit.pl checks the job on the local node. If the MC check returns TRUE (MC is good) then the link is moved to the finished directory otherwise the job is resubmitted.

The finished directory contains links to the local nodes where a GEANT job finished running and the MC check return TRUE. All links in the finished directory always point to a local node with MC that was checked and the result returned TRUE.

The to_be_deleted_in_future directory contains links to the local nodes where a GEANT job finished running but the MC check returned FALSE. There are two different links that can appear here. The first link is created when a job is submitted to PBS (before it starts running – link name ends with 'queued' and this points to the script in the scripts directory). This link allows jobs that happen to vanish from PBS without being executed on a local node to be resubmitted. This link gets deleted when the job starts running. The second type of links in the to_be_deleted_in_future directory point to nodes with incomplete MC data for whatever reason. tbsub_checkMC_and_resubmit.pl will resubmit all jobs with a link in the to_be_deleted_in_future directory that are not in the PBS queue or running on a local node.

The unreachable directory only exists if any of the Westgrid nodes could not be reached (ssh time-out) by tbsub_checkMC_and_resubmit.pl when it checks MC. It contains links to the local nodes that are unreachable. When tbsub_checkMC_and_resubmit.pl is executed again, it checks to see if the unreachable directory contains links. If it does it adds those nodes to the list of nodes to be checked so that the nodes are checked again.

3.3 Environment Variable File

The template file is located /global/home/e614/rundb-dev/exe/e614_EnvTmpFile
It contains all the default values for the environment variables that get passed to the scripts. This is a template file because genXXX and analYYY are variables that get replaced at submission time with the appropriate values.

An output environment variable file is produced and placed in /global/home/e614/tbsroot/genXXX/common when jobs are submitted. This is for record-keeping and it is also used for ensuring a consistent state between the first submission and subsequent resubmissions. Thus is it important that these files don't overwritten or modified once they are produced. tbsub.pl checks to make sure it doesn't overwrite any of these environment variable files.

When the GEANT jobs for genXXX are submitted, one output file is produced in the <u>common</u> directory called genXXX-EnvFile . If MOFIA jobs are chained to the GEANT jobs or if MOFIA jobs are run on their own with the --mofia_only option, one output file per MOFIA analysis is produced in the <u>common</u> directory called genXXXanalYYY-EnvFile

During resubmission of GEANT jobs which is only done using tbsub_checkMC_and_resubmit.pl, the genXXX-EnvFile is read to load all the environment variables used for the first GEANT submission with tbsub.pl

During resubmission of MOFIA jobs which is only done using tbsub.pl with the --mofia_only option, the genXXXanalYYY-EnvFile is read to load all the environment variables used for the first MOFIA submission with tbsub.pl

Only a few environment variables can be overwritten by passing parameters to tbsub.pl during MOFIA resubmission: CHAIN_NUM, NO_EMAIL, and PBSPARAM. All other tbsub.pl parameters are ignored to ensure a consistent state during resubmission. When resubmitting either GEANT or MOFIA it is possible but not recommended to change the corresponding environment variable file in the <u>common</u> directory. Changing these files would result in an inconsistency between how GEANT or MOFIA are resubmitted and how GEANT or MOFIA were originally submitted.

3.4 Submitting a Job

Once the job directory is setup, call **/global/home/e614/bin/tbsub.pl** to submit a GEANT and/or MOFIA job. See *westgrid_scripts.sxw* for usage of tbsub.pl located in ~e614/public_html/private/sysman/westgrid/westgrid_script.sxw on tw04.triumf.ca The contents of the document are included in the appendix. A pdf version is also available. The URL (Uniform Resource Locator) is http://twist.triumf.ca/private/sysman/westgrid/westgrid_scripts.pdf

- Once 'tbsub.pl' is executed, it will submit jobs to qsub, creating directories running, finished, submitted, scripts, to_be_deleted_in_future and errorlogs

- A given ffcard for a run will be moved from /queued to /submitted and copied to to the output destination on the local node upon submission of the GEANT job

- A link to the script will be created at submission time in the to_be_deleted_in_future directory with the name run<run_number>-queued This link is used by tbsub_checkMC_and_resubmit.pl to resubmit jobs that vanish from the PBS queue without a trace. The link is removed once the job starts running.

- Once the job in the queue starts running, a symbolic link of the output destination will be created in /global/home/e614/tbsroot/<jobname>/running

- Once the GEANT job is finished running, the symbolic link from /running will be moved. The location it is moved to depends on the result from check MC. If check MC returns the MC is good then the link is moved to /global/home/e614/tbsroot/<jobname>/finished If check MC returns the MC is bad then then link is moved to /global/home/e614/tbsroot/<jobname>/to_be_deleted_in_future and waits to resubmitted by tbsub_checkMC_and_resumbit.pl

- If a script failed during execution or if the check MC returned that the MC is bad, the script's log file will be written to /global/home/e614/tbsroot/<jobname>/errorlogs

- tbsub.pl submits MOFIA jobs using either the links located in /global/home/e614/tbsroot/<jobname>/finished if MOFIA runs using MC or /global/scratch/twist/rawdata_symlinks/<jobname> if MOFIA runs using data

- When a MOFIA job is complete checktrees is automatically executed and either a goodlink or a badlink is created for that run.

3.5 Output

The destination of the output is stored locally on the 'ICE' machines for GEANT. To determine which machine the job ran on see the links in /global/home/e614/tbsroot/<jobname>/finished or /global/home/e614/tbsroot/<jobname>/to_be_deleted_in_future, or see qsub while the job is running.

The structure on the ICE machines is:
/data/twist/e614/tbsdata/<jobname>/run<run_number>/

- within /run<run_number>, you will find files similar to:

```
bat614 -> /global/home/e614/tbsroot/genXXX/common/bat614
e614.com ->/global/home/e614/tbsroot/genXXX/common/e614.com
gb.sh -> /global/home/e614/tbsroot/genXXX/common/gb.sh
mcrun<run_number>.ffcards
run<run_number>.dat
run<run_number>.hbook
tbslog.txt
```

For MOFIA execution the output is stored on /global/scratch/twist/systematics/mc/genXXX/analYYY/root/run<run_number> if the data is from GEANT. If the data is from a data set then the output is stored on /global/scratch/twist/systemeatics/data/genXXX/analYYY/root/run<run_number>
The directory format is the same regardless of where GEANT data or a set is used.

- within /run<run_number>, you will find files similar to:

```
tree<run_number>.root
mofialog.dat
<run_number>.root
run<run_number>log.txt
```

4.0 Conclusions

Working with TWIST has been very enjoyable learning experience. I've learned about computing science, physics, and team dynamics. In computing science I've explored new languages, database systems, Linux clusters, and emulation. Although, I'm not a physics major, I've learned about physics through experiment data runs and seminar talks. Because TWIST is a team effort, I've gained a clearer understanding of how teams work together in the real-world by watching the TWIST team operate. As a project leader in my fourth year computing science project course for my computing science major, I understand some of the problems that can face a team such as TWIST. Teamwork is a balance of making arguments and counter-arguments, and being respectful of one another. I found the team dynamics for TWIST to be fairly good and relatively exercised. Although no team can be perfect, I enjoyed being apart of the TWIST team in the time I spent at TRIUMF.

The tasks I did on my co-op work-term were very useful to me. I enjoyed the experience in creating databases and dynamic web-pages, and using Linux clusters. TWIST has provided an enjoyable learning experience. I especially enjoyed programming in the Linux environment and the variety of tasks I was assigned.

5.0 Recommendations

Overall the learning experience and environment was very good and beneficial. I applied for a co-op position at TRIUMF with the interest in learning more about physics and computing science and I enjoyed having a number of opportunities to accomplish that. I was given a tour of the TRIUMF cyclotron and the TWIST detector. A number of people in TWIST were very friendly in explaining some of the physics behind the projects they were working on. I was also given the opportunity to attend a number of physics talks. Although most of these talks were above my knowledge in physics, I was able to grasp and learn parts of what the presenters were trying to say to construct in my head a general understanding of what the talk was about. The opportunity to take shifts using the TWIST detector was very enjoyable and an excellent opportunity to learn about particle physics. I recommend any student who wishes to learn about physics to apply for a co-op position at TRIUMF.

The opportunity to have practical experience in Linux was exciting as well. The reason is that many people are turning to Linux as a free alternative to Windows because it provides many useful server and production tools for both commercial and non-commercial use. UNIX (UNiplexed Information and Computing Services) and it's different variants are very popular in many computing fields for its simple and stable design. Some of the topics I've learned during this co-op work term have sparked an interest in a few areas. One area I had an interest in were Linux compilers, Linux clusters and parallel computing, and CVS code repositories. Another area of interest was in compilers and both high-level and low-level emulation. An example of low-level or machine emulation is the JVM (Java Virtual Machine). The emulation I worked on at TWIST was called high-level emulation. I think it would interesting to learn more about both kinds of emulation in the future.

6.0 References

Altair Grid Technologies (2004). Portable Batch System. Retrieved Dec, 2004, from http://www.openpbs.org/

Cluster Resources (2004). Documentation. Retrieved Dec, 2004, from http://www.clusterresources.com/products/documentation.shtml

Litt, Steve (2001). PERL Regular Expressions. Retrieved Dec, 2004, from http://www.troubleshooters.com/codecorn/littperl/perlreg.htm

MacDonald, Rob (2004). TWIST Overview. Retrieved Dec 2004, from http://twist.triumf.ca/

O'Reilly Media (2004). Perl Documentation. Retrieved Dec, 2004, from http://www.perl.com/pub/q/documentation

O'Reilly Media (2004). Downloading the Latest Version of Perl. Retrieved Dec, 2004, from http://www.perl.com/download.csp

Westgrid (2004). Westgrid: Western Canada Research Grid. Retrieved Dec, 2004, from http://www.westgrid.ca/

Wikipedia (2004). Muon. Retrieved Dec, 2004, from http://en.wikipedia.org/wiki/Muon

7.0 Glossary

Certificate – An electronic document containing personal information that binds a public encryption key to specific person. The certificate is issued and electronically signed by a certificate authority. The certificate is used to confirm the public encryption key really belongs to a specific person.

Cluster – A group of networked computers, each with their own central processing unit, memory, operating system, and possibly a hard disk, working one unit. They can share a global storage space and use networking protocols to communicate such as the Internet's TCP/IP.

Compiler – A software program that parses high-level language source code to generate the equivalent machine code instructions.

CVS Code Repository – It stands for Concurrent Versioning System and is a database for source code that manages changes made to the source code and keeps track of incremental changes committed to the database.

Cyclotron – A cyclotron is a vacuumed cylindrical chamber that accelerates charged particles such as protons outward from the center of the cylinder in a spiral until the accelerated particles exit from various portals on the sides of the chamber.

Database – An organized collection of information. A database management system manages operations to an electronic database.

Detector – A device in physics that detects, and measures properties of particles such as angle and momentum

Dynamic Web-Page – A web page written in a language that generates the HTML code each time the page is requested.

Environment Variables – Linux operating system variables that are contained in the current shell.

Emulation – There are two kinds of emulation: low-level and high-level. Low-level emulation virtualizes the machine code instructions by translating these to the machine code instructions of the executing machine. High-level emulation involves linking an executable to an alternative library which has a subset of the operations implemented and mimics the operation of the original library.

Grid – A multi-user cluster that allows multiple independent jobs to be run across a cluster. A grid virtualizes the computing resources of the cluster to administer the computing resources to multiple users.

Hard-Link – A link in the file system that points directly to the physical inode rather than the file system location.

Inode – The physical address of the block where information resides on the hard disk.

JVM – It stands for Java Virtual Machine and implements low-level emulation of a non-existent byte-code machine. A JVM translates the byte-code to the executing machine's machine language.

Muon – A fundamental and semi-stable sub-atomic particle with a negative charge.

Parallel Computing – Strictly speaking a computer that shares multiple processors in either symmetric parallel processing or asymmetric parallel processing. Symmetric parallel processing is where processors are considered “equal” such that no one processor controls the instructions the other processor executes. Asymmetric parallel processing is where a master processor controls what the other processors execute.

Particle Beam – A stream made up of sub-atomic particles.

Public/Private Key Encryption – A public and private key are complement encryption n-bit sized binary numbers where only the private key can decrypt data encrypted with the public key. The private key is carefully guarded by the server computer. Public/private key encryption is used to exchange the shard key, user name and password so that secure communication can continue with the faster shared-key encryption.

Shared-Key Encryption – Encryption that involves two parties having the same secret n-bit sized binary number. Data is encrypted and decrypted with this secret n-bit sized binary number. Any computer with the shared-key can listen and/or intercept encrypted traffic.

Shell – A user environment with which the user can interact with the operating system.

SQL – It stands for Structured Query Language and is a standardized language for relation databases for data query, manipulation and definition.

Sub-Atomic Particle – A sub-atomic particle is a particle smaller than an atom.

Symbolic Link (Soft-link) – A file system entry that points to another file system entry.

8.0 Appendix

8.1 Overview

The following section includes a detailed description of the available parameters in the Westgrid job submission scripts. The scripts described here are tbsub.pl for Westgrid job submission of GEANT and/or MOFIA jobs and resubmission of MOFIA jobs; tbsub_checkMC_and_resubmit.pl for GEANT job resubmission; and tbdel.pl for removing jobs from the Westgrid job queue. The information provided here is available from by executing each script without any arguments. The contents here are in the manual *westgrid_scripts.sxw*

8.2 tbsub.pl

Purpose: To execute GEANT and/or MOFIA (replaces tbsub_geant.pl and tbsub_mofia.pl)

found in /global/home/e614/bin/tbusb.pl

Submits a GEANT & MOFIA jobs

Default environment file: /global/home/e614/rundb-dev/exe/e614_EnvTmpFile
All default parameters are located in the environment file. Use option --show_env to display all the environment variables used by GEANT/MOFIA without submitting any jobs.

Usage:

tbsub.pl genName [--help] [--show_env] [--env_file='env_file'] [--script='userscript'] [--geant_exe='absolute_path'] [--chain_num='number'] [--pbsopt='opt1 opt2 ...'] [--qos='opt'] [--copy_data] [--data_outpath='absolute_path'] [--data_dir='dirname'] [--hbook_dir='dirname'] [--log_dir='dirname'] [--mofia_outpath='absolute_path'] [--root_tree_dir='dirname'] [--kcm_path='absolute_path'] [--mofia_exe='absolute_path'] [--no_email] [--geant_only] [--mofia_only] [--mofia_inpath='absolute_path'] [--local]

genName	= gen/set name to compute in the format genXXX or setXXX (setXXX is allowed only if --mofia_only is specified)
--help	= displays this usage screen
--show_env	= displays all environment variables defined in the environment variable file without submitting any jobs
--env_file	= file to load the environment variables, default is /global/home/e614/rundb-dev/exe/e614_EnvTmpFile
--chain_num	= number of jobs to chain to a Westgrid job (GEANT and/or MOFIA jobs can be chained)
--pbsopt	= options to pass on to pbs

```
    --qos              = default is 'normal', can also pass 'hi_prio' and 'debug'
                         (if 'debug' walltime will be set to 9.59min)
    --no_email = disables email except when jobs are aborted by PBS

GEANT specific options
    --geant_only       = specifies that only GEANT jobs are submitted
    --geant_exe        = absolute path to GEANT executable
    --script           = GEANT script to execute on the local node including the
                         absolute path
    --copy_data        = copy generated data to global disks (disabled)

    Following options should only be specified if --copy_data was specified
        --data_outpath = path to copy your job data, default is
                         "/global/scratch/twist/mc/$genName"
        --data_dir     = directory name to store data relative to "$data_outpath",
                         default is 'data'
        --hbook_dir    = directory name to store hbooks relative to "$data_outpath",
                         default is 'hbook'
        --log_dir      = directory name to store logs relative to "$data_outpath",
                         default is 'log'

MOFIA specific options
    --mofia_only       = specifies that only MOFIA jobs are submitted
    --mofia_exe        = absolute path to mofia executable
    --mofia_outpath    = path to the MOFIA output directory, default is
                         "/global/scratch/twist/systematics"
    --root_tree_dir    = directory name to store root trees relative to
                         "$mofia_outpath/mc|data/$genName", default is 'root'
    --kcm_path         = absolute path to kcm file for analysis, default is
                         "/global/home/e614/tbsroot/$genName/common/$genName"
                         ."anal1.kcm"

    Options used only when --mofia_only is specified (only one of these options can be
used)
        --local  = specify that the MC data is on the local node, this is
automatically selected if genName is in the form genXXX
        --mofia_inpath         = the absolute path of where the datalinks are on the
                                 global  disk, this is automically selected if genName is in
                                 the form setXXX, default is
                                 "/global/scratch/twist/rawdata_symlinks/$genName"

Notes:  1) Do not include trailing slashes '/' at end of paths.
        2) "$genName" is the only required argument when MOFIA execution is
        chained to GEANT.
        3) The default arguments come from the default environment variable file:
        /global/home/e614/rundb-dev/exe/e614_EnvTmpFile
        4) KCM file for MOFIA analysis must have a name with the format
        genXXXanalYYY.kcm
```

5) GEANT_ONLY, MOFIA_ONLY and GEANT & MOFIA jobs can be chained

Examples:

Standard submission to generate GEANT data:

```
tbsub.pl gen2 --geant_only
```

Standard submission to generate GEANT data and perform MOFIA analysis for anal1 on the same node:

```
tbsub.pl gen2
```

Standard submission to generate GEANT data, but changing walltime to 3 hours, using queue ice_N, and specifying disk usage to 1gb:

```
tbsub.pl gen2 --pbsopt='-l nodes=1,walltime=3:00:00,file=1gb -q ice_N' --
geant_only
```

Standard submission/resubmission for MOFIA analysis anal2 on gen2 on local nodes (automatically gets link information from '/global/home/e614/tbsroot/gen2/finished'):

```
tbsub.pl gen2 --mofia_only --
kcm_path='/global/home/e614/tbsroot/gen2/common/gen2anal2.kcm'
```

Standard submission/resubmission for MOFIA analysis anal2 on set2 on global disks (automatically gets link information from '/global/scratch/twist/rawdata_symlinks/set2'):

```
tbsub.pl set2 --mofia_only –
kcm_path='/global/home/e614/tbsroot/set2/common/set2anal2.kcm'
```

8.3 tbsub_checkMC_and_resubmit.pl

Purpose: To check failed/crashed GEANT jobs and resubmit all jobs where the MC check returned FALSE (to be used only with tbsub.pl)

found in /global/home/e614/bin/tbusb_checkMC_and_resubmit.pl

NOTE: To be used only with tbsub.pl
Checks the status of crashed/failed jobs and resubmits if necessary

```
Usage: tbsub_checkMC_and_resubmit.pl genXXX
  eg. tbsub_checkMC_and_resubmit.pl gen93
```

8.4 tbdel.pl

Purpose: Deletes all jobs of a particular set or gen from PBS using qdel

found in /global/home/e614/bin/tbdel.pl

Removes all PBS jobs in relating to $genName

Usage:

```
tbdel.pl genName
```

genName = Name of gen or set (in the format genXXX or setXXX) to remove from PBS

Example

```
tbdel.pl gen2
```