

# Minnesota CLEO III Monte Carlo Farm

## Abstract

This note describes the hardware and software used for the Condor-based Minnesota CLEO III Monte Carlo production facility. The farm scripts provide an easy and reliable means to generate large Monte Carlo samples on a networked cluster of computers. Application of these scripts at other sites should be straightforward.

## Contents

<b>1</b>	<b>Monte Carlo Farm Scripts</b>	<b>2</b>
1.1	Condor . . . . .	2
1.2	Perl Scripts . . . . .	3
1.2.1	mc_start . . . . .	3
1.2.2	mc_congealer . . . . .	4
1.2.3	mc_archiver . . . . .	4
1.2.4	mc_check . . . . .	5
<b>2</b>	<b>Using the Farm Scripts</b>	<b>5</b>
2.1	Obtaining the scripts . . . . .	5
2.2	Installing the MC Farm Scripts . . . . .	5
2.2.1	Installing Condor . . . . .	5
2.2.2	Installing the Farm Scripts . . . . .	6
2.2.3	Additional Configuration Issues . . . . .	9
2.2.4	CLEO III Monte Carlo Code . . . . .	9
2.2.5	Installing the Perl CPAN Modules Used by the Farm . . . . .	9
2.3	Running the Farm . . . . .	9
2.4	Starting and Stopping Specific Farm Processes . . . . .	12
2.5	Adding or Removing Worker Machines . . . . .	12
2.6	Constants Server Management . . . . .	13
2.7	CLEO III Code Management . . . . .	13
2.8	Random Trigger Events . . . . .	13
2.9	Monitoring the Farm Status Through the Web Interface . . . . .	14

<b>3</b>	<b>Minnesota Monte Carlo Farm Hardware</b>	<b>16</b>
<b>4</b>	<b>Performance</b>	<b>16</b>
<b>5</b>	<b>Portability Issues</b>	<b>16</b>
<b>6</b>	<b>Conclusions and Future Plans</b>	<b>16</b>

## **1 Monte Carlo Farm Scripts**

The Minnesota Monte Carlo farm (MN Farm) scripts are intended to be a simple yet robust solution to generating Monte Carlo on a medium-sized cluster of machines.

The University of Minnesota group generated a large fraction of the Monte Carlo samples used for CLEO II analyses. The software used for that job was a mixture of sh and tcsh scripts.

For CLEO III, the Minnesota Monte Carlo farm scripts were completely rewritten in order to take advantage of new technology designed for GRID computing. The scripts utilize the Condor parallel processing software [1] written at the University of Wisconsin–Madison. The Condor software manages the worker machines and handles the distribution of jobs to worker machines. The scripts are written in Perl, which offers greater functionality than bash and tcsh, yet retains the portability of those scripting languages.

While Condor is capable of truly distributed GRID computing on widely distributed processors on desktops or otherwise, the MN Farm scripts were not intended for that purpose. A cluster of computers with their disks NFS (or otherwise) mounted on a local network is optimal for generating Monte Carlo at a single site. This is probably adequate for all but some of the largest of experiments, which are considering GRID-based Monte Carlo generation.

The main CLEO III MN Farm scripts and web interface were written by Alex Smith. Alexander Scott wrote the graphical user interface and has taken over responsibility for maintaining and modifying the scripts.

### **1.1 Condor**

Condor is a parallel processing software package developed for running CPU-intensive computations using a large number of computers, which may be distributed at remote locations and may be running different platforms. The MN Farm is built around the Condor package. The code checkpointing features of Condor, which allow a job to be stopped and resumed where it left off, are not utilized in the MN Farm scripts. Code checkpointing is not practical in this application, as it would require the cleo3 code to be compiled with the Condor libraries. Furthermore, it is not necessary in a farm

consisting primarily of dedicated worker machines, which are rarely removed from the Condor pool. If a machine is removed, the job running on that machine is killed and resubmitted to another machine.

The authors of Condor provide excellent documentation and support. Extensive documentation about the features and installation of Condor may be found at the Condor web site [1].

## 1.2 Perl Scripts

Most of the MN Farm is written in the Perl scripting language. The MN Farm Perl scripts were written to be portable across UNIX-like platforms (TRU64 UNIX, Linux, Solaris, etc.). Furthermore, the farm is capable of running on a several different platforms at once, as long as the Monte Carlo code is supported on each platform. Presently, the scripts have been written to run on TRU64 UNIX and Linux machines. When the port of CLEO III code to Linux is complete, Linux nodes will be added to the farm.

The MN Farm Perl scripts consist of four major modules, which are described below, plus several utility scripts, which will be described in Section 2.3.

### 1.2.1 mc\_start

The module `mc_start` (`mc_start.pl`) creates the Monte Carlo scripts and submits them to the Condor queues.

Specifically, the `mc_start` module does the following steps:

1. Check the farm prerequisites (routine `check_farm_prerequisites`). These include the following:
  - (a) Make sure the Condor master process is running on the Condor master node. Restart if not running.
2. Create a list of nodes in the farm and their local working areas to be used by the farm.
3. Clean out directories for this farm if they already exist.
4. Create the directory structure used by the farm, if it does not already exist.
5. Get information for each run (luminosity, number of events to generate, etc.) and create a master job list to be used by the farm.
6. Create a local archive of the setup files for this farm.
7. Create and submit job scripts to the Condor queue.
8. Start the congealer (`mc_congealer`) process on the farm master node.

9. Start the archive handler (`mc_archiver`) process on the archive node.
10. Start the web page farm monitor process (`mc_check`) on the farm master node.

The script `mc_start` exits after the farm jobs are all submitted.

A graphical user interface called with the command line `mc_farm` provides an easy means to do most of the tasks necessary to run the farm, such as: setting up the farm environment, configuring a Monte Carlo run, starting and stopping the constants server, and starting the farm.

### 1.2.2 `mc_congealer`

The module `mc_congealer` (`congealer.pl`) watches the jobs and collects output to archives as the jobs finish. If jobs fail, the congealer will resubmit them a configurable number of times.

The congealer loops over the following tasks:

1. check if the farm is done.
2. Check the status of all jobs in the master job list.
  - If the job has failed, create a Condor “rescue file” and resubmit the job. Save a copy of the log files for debugging purposes.
  - If the job is done, move the file to an output staging directory.
  - If there are no unfinished intervening jobs the file is moved to the archive volume area.
3. When enough files are in the archive volume area to make a complete volume (specified in the setup), the log files corresponding to these runs are tarred and gzipped. A flag is set to tell the archiver that the volume is ready to be transferred to the master site.
4. Update the master job list with the current job status.
5. See if congealer stop is requested. If so, archive the master job list and complete any open archives.

### 1.2.3 `mc_archiver`

The archiver (`archiver.pl`) watches for finished archives and sends finished volumes consisting of Monte Carlo and log files to the master site (in this case Cornell) using the `scp` command. The archiver will only transfer files during the time range set in the archiver setup. The archiver checks for free space on the remote site before transferring

the file. RSA encryption keys must be set up between the machine at the master site and the node on which the archiver is running.

The archiver loops over the following tasks:

- See if archiver stop is requested.
- Determine the index of the current finished volume.
- Check the time to see if it corresponds to the block of time when volume transfers are allowed.
- Fetch the name of the current volume from the status file created by the congealer.
- Secure copy files to master site (Cornell).
- Update archive status file, which is used by `mc_check`.

#### 1.2.4 `mc_check`

The `mc_check` process (`mc_check.pl`) creates and updates a web page from which the status of the farm may be monitored. This is especially convenient for monitoring from remote locations. This web page is discussed in greater detail in Section 2.9.

## 2 Using the Farm Scripts

### 2.1 Obtaining the scripts

When the port of the CLEO III code to Linux is complete, we plan to include the final MN Farm scripts in the CLEO III libraries, so they will be available with the CLEO III code distribution.

### 2.2 Installing the MC Farm Scripts

#### 2.2.1 Installing Condor

See instructions on the Condor web site [1] for installing Condor to your cluster if it is not already installed. This is the most challenging part of the farm script install, but is supported with extensive documentation and a helpful staff of developers who will actually respond to your email! The MN Farm scripts are designed to utilize the “Vanilla Universe” in Condor. At MN we installed condor under username “condor” so that root privileges are not required in order to manage condor.

### 2.2.2 Installing the Farm Scripts

Copy the scripts to some directory on your cluster, *YourScriptDirectory*.

The following lines must be added to the `.cshrc` script of the account from which the farm will be run:

```
setenv FARM_SCRIPTS YourScriptDirectory
source ${FARM_SCRIPTS}/.mcfarmrc
```

The script `.mcfarmrc` defines a bunch of commands used by the farm. You may see a list of defined command lines within this script. These commands may be used in place of the corresponding commands in the menus of `mc_farm`. The most useful of these are `mc_start farmname`, `mc_resume farmname`, and `mc_stop farmname`. These are discussed in Section 2.3.

Further configuration of the farm is specified from a graphical user interface. This interface, shown in Fig. 1, is opened by typing `mc_farm` at the command line. Within this window are buttons to configure various aspects of the farm. The farm configuration is specified by filling in the menus in categories corresponding to three buttons. The “General parameters” menu, shown in Fig. 2, allows one to specify most of the specifics of an installation. The “Archive parameters” menu, shown in Fig. 3 allows one to specify the properties of the archive files and procedures. Finally, one may specify the constants database server setup using the “Farm constants” menu, shown in Fig. 4.

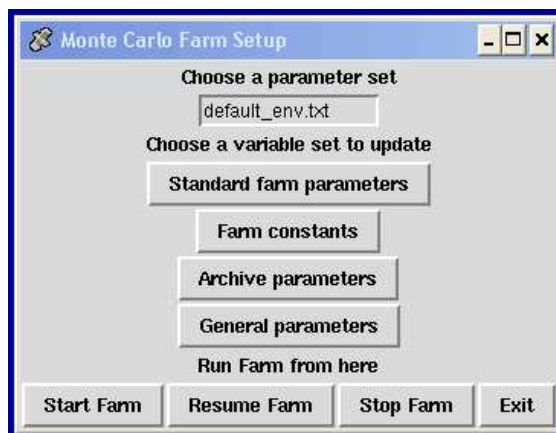


Figure 1: The Monte Carlo farm main control panel.

**Monte Carlo Farm Setup**

<b>\$C3_RUNFIL</b> /cleo3/runfil	<b>Area for trigger files</b> /data/cleo4/cleo3/c3lib/random_triggers
<b>Working area of farm</b> /data/cleo18/cleo3	<b>When to email farmer</b> Never
<b>Condor priority for first job</b> 0	<b>web page area</b> /home/hep/cleo3/public_html
<b>name of farm facility</b> MN	<b>web page name</b> /home/hep/cleo3/public_html/mcfarm_html
<b>Area where farm archives are located</b> /home/hep/cleo3/mcfarm/archive	<b>username when calling to LNS</b> c3mc
<b>Roster of machines to be added or removed</b> /home/hep/cleo3/scott_mn_farm/mn_farm/farm_machine_roster	<b>Nice priority of jobs</b> 15
<b>Master node for farm</b> mnheps.hep.umn.edu	<b>Output volums size</b> 5000
<b>Congealer node for farm</b> mnmc11.hep.umn.edu	<b>Minimum local space for output (Mb)</b> 1000
<b>Master node for condor</b> mnmc0.hep.umn.edu	<b>Minimum memory for QQ (Mb)</b> 100
<b>Max number of retries</b> 2	<b>Minimum free space in /data/cleo18/cleo3/output/ (Mb)</b> 100

Default      UPDATE      SAVE PARAMETERS      RETURN

Figure 2: Panel used for general configuration of the farm installation.

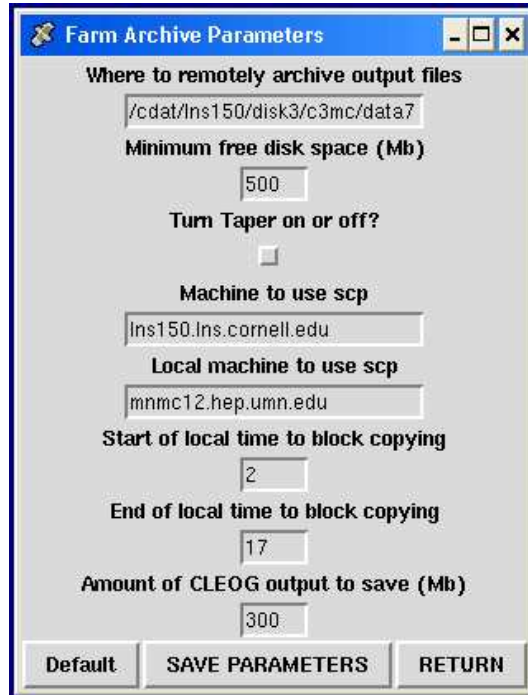


Figure 3: Panel used to configure the archiver parameters.

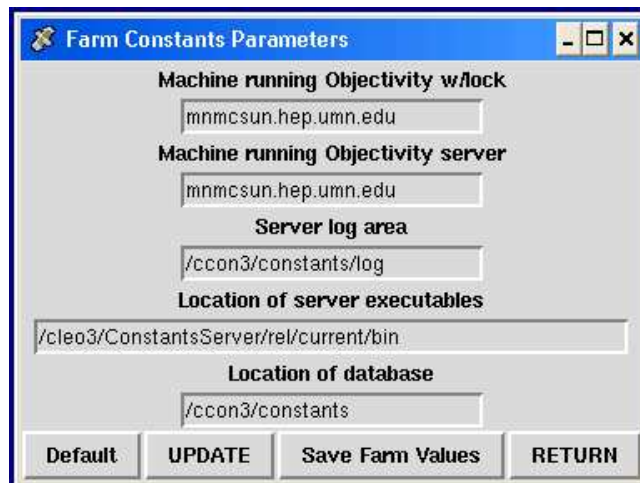


Figure 4: Panel used to configure the constants database server.

### 2.2.3 Additional Configuration Issues

RSA encryption keys must be set up between the machine at the master site and the node on which the archiver is running. This will allow the archiver machine to secure copy (`scp`) volumes without entering a password.

### 2.2.4 CLEO III Monte Carlo Code

The CLEO III libraries must be installed on the cluster to be used. Section 2.7 discusses a utility used to update the CLEO III code on the local machines.

For the first installation, there are likely to be other issues that come up in this process. It is a real pain right now, but promises to be easier in the future after the Offline package is integrated into Autoconf. A summary of what was done to install the CLEO III code on the MN Farm cluster can be found in CBX note [2].

### 2.2.5 Installing the Perl CPAN Modules Used by the Farm

The farm uses a couple Perl CPAN modules, which must be installed on your cluster. Install the modules `NFSLock`, `Tk` and `localtime` which are available from the official Perl web site or its mirrors [3].

## 2.3 Running the Farm

Most routine operations for the farm are accomplished through the graphical user interface, which is started with the command: `mc_farm [farmname]`. If a new farm is being started, the *farmname* argument should be omitted. If a running or previously defined farm is to be modified the name of that farm should be specified as a command line argument.

After starting `mc_farm`, a main control panel will be displayed, as shown in Fig. 1. From this panel, the farm may be started, stopped, or resumed. A new farm may be configured by selecting the “Standard Farm Parameters” button. This opens a screen similar to that shown in Fig. 5. In this window, one may specify the physics to be generated. After entering the new values, one must hit the “save parameters” and “update” buttons to save the parameters to a file and update other parameters which depend on these.

These commands and some additional ones may also be run from the UNIX command line. They are defined in `.mcfarmrc`.

- `mc_start farmname`  
Start a new farm. (May also be done from through `mc_farm`.)
- `mc_resume farmname`  
Resume farm running after a stop. (May also be done from through `mc_farm`.)

- `mc_restart_failed farmname`  
Restart only failed jobs.
- `mc_restart_all farmname`  
Restart all jobs that have not finished (both failed and queued).
- `mc_stop farmname`  
Stop the farm. (May also be done from through `mc_farm`.)

Monte Carlo Farm Setup

**Farm Name**  
bbbar\_1

**Series**  
1

**MC Type**  
bbbar

**CLEO3 Code release**  
cleo3\_MCPS2\_Jan30\_2002

**Data Set**  
data7

**First Run to Generate**  
112259

**Last Run to Generate**  
113216

**Run QQ Within CLEOG?**

**Event Generator to Use if Generating Within CLEOG**  
BBbarGeneratorProd

**QQ Executable to use if NOT running within CLEOG**  
/cleo/bin/c3qq

**QQ Control file to use if NOT running within CLEOG**  
/cleo3/Offline/rel/cleo3\_MCPS2\_Jan30\_2002/src/MCInfo/data/decay.dec

**Standard Decay File (decay.dec)**  
/cleo3/Offline/rel/cleo3\_MCPS2\_Jan30\_2002/src/MCInfo/data/decay.dec

**User Decay File (udecay.dec)**  
/cleo3/Offline/rel/cleo3\_MCPS2\_Jan30\_2002/data/mcevnts\_data7.asc

**Multiple of Luminosity to Generate**  
1.0

**Type of Run Information File**  
SPECIFIC

**Run Information File**  
/cleo3/Offline/rel/cleo3\_MCPS2\_Jan30\_2002/data/mcevnts\_data7.asc

**Cross Section of Process (if using RUNINFO)**  
1.0

**Advanced Options**  
Farm Configuration

Default UPDATE SAVE PARAMETERS RETURN

Figure 5: The Monte Carlo farm configuration panel.

## 2.4 Starting and Stopping Specific Farm Processes

Commands are defined to start and stop the individual farm processes without starting or stopping the entire farm. This is particularly useful in the case when a process dies and needs to be restarted.

- `mc_stop_congealer farmname`  
Stop the congealer. Wait for the process to die before returning.
- `mc_start_congealer farmname`  
Check to see if the congealer is running. If not running, start it.
- `mc_stop_archiver farmname`  
Stop the archiver. Wait for the process to die before returning.
- `mc_start_archiver farmname`  
Check to see if the archiver is running. If not running, start it.
- `mc_stop_webpage farmname`  
Stop the web page update. Wait for the process to die before returning.
- `mc_start_webpage farmname`  
Check to see if the web page is running. If not running, start it.

## 2.5 Adding or Removing Worker Machines

If nodes have already been set up as possible members of the Condor pool of machines, they may be added or removed from the farm during a run using the command lines described below. If a machine is stopped during the middle of a run, that job will be killed and resubmitted to another machine.

- `mc_start_node farmname`  
Add node to the farm.
- `mc_stop_node farmname`  
Remove node from the farm.

## 2.6 Constants Server Management

- `mc_update_constants` *farmname*

This utility steps the user through the update of the constants database. The update uses the database archives from LNS. Here are the steps that the constants update steps one through:

1. Choose which database archive to download from Cornell. The contents of the archive directories are shown.
2. Choose an action. The options are:
  - (a) Fetch database archives
  - (b) Unpack database archives
  - (c) Install database archives
  - (d) Update constants server executables
  - (e) Exit

The script will prompt the user for the required information to update the constants and/or server executables. The executables should always be updated when the constants are updated.

- `mc_start_constants` *farmname*

Start or restart the Constants and Runstatistics database servers on the Solaris server.

## 2.7 CLEO III Code Management

- `mc_update_code` *farmname*

This utility leads the user through the steps required to update the CLEO III code to a new release. It is possible that this script will require updates over time in order to stay current as the library structure and build procedures change.

## 2.8 Random Trigger Events

Events from random triggers are merged with generated Monte Carlo events in order to simulate electronic noise. These events are collected from the same data set as the Monte Carlo being generated. They can be found in the area `$C3_MERGE_DIR` at Cornell. They should be placed in the directory specified in the farm setup, discussed in Sec. 2.3.

## 2.9 Monitoring the Farm Status Through the Web Interface

A web page is automatically generated for each running farm. This allows one to monitor the farm locally or from a remote location. The status web page is divided into four sections. The first section displays the number of events generated and indicates the status of the farm processes, constants servers, and free area on the output disk. The second section displays the current size, location, and transfer status of the Monte Carlo farm output archives. Examples of these sections are shown in Fig. 6. The third section, shown in Fig. 7, is a summary of the farm processor pool. The status, local directory, free disk space, and memory is displayed for each machine. Finally, the status of each Monte Carlo job is displayed, including the corresponding luminosity, the number of events being generated, the number of retries, the processor node, and the clock time per event. A link to the log files for each job is also provided. An example of the job summary web page is shown in Fig. 8.

---

# CLEO III Monte Carlo farm **tau\_d17\_7a**

Last updated **Mon Jan 20 12:47:32 CST 2003**

---

## 403944 events generated

---

### Messages:

- \* Files are being collected to /data/cleo18/cleo3/output/tau\_d17\_7a which has 20.2 GB free
  - \* Congealer is running on mnmc0.hep.umn.edu
  - \* Archiver is running on mnmc12.hep.umn.edu
  - \* Constants client (nsd) is running on mnmc0.hep.umn.edu
  - \* AllConstantsServer is running on mnmcsun.hep.umn.edu
  - \* RunStatisticsDBServer is running on mnmcsun.hep.umn.edu
  - \* osagent is running on mnmcsun.hep.umn.edu
  - \* Objectivity lockserver (oolockserver) is running on mnmcsun.hep.umn.edu
- 

### Status of MC Archives:

Archive Name	Status	Location
tau_d17_7a_VOLUME_1	Not yet complete-- 2560.168/5000 MB	mnmc12.hep.umn.edu:/data/cleo18/cleo3/output/tau_d17_7a/tau_d17_7a_VOLUME_1

---

Figure 6: The Monte Carlo farm summary and archive status web pages.

---

**Status of the Condor Pool:**

Machine	State	Local dir.	Disk Free (GB)	Memory (MB)
mnhepq.hep.umn.edu	Unclaimed/Idle	/data/cleo23/condor/execute	8.9	256
mnheps.hep.umn.edu	Claimed/Busy	/scratch/mnheps/condor/execute	7.7	256
mnhept.hep.umn.edu	Unclaimed/Idle	/data/cleo12/condor/execute	6.9	192
mnhepv.hep.umn.edu	Claimed/Busy	/data/cleo11/condor/execute	20.9	256
mnmc0.hep.umn.edu	Claimed/Busy	/data/cleo4/condor/execute	8.2	1024
mnmc1.hep.umn.edu	Unclaimed/Idle	/data/cleo6/condor/execute	28.1	192
mnmc4.hep.umn.edu	Claimed/Busy	/scratch/mnmc4/condor/execute	54.1	256

Figure 7: The Monte Carlo farm processor pool status web page.

---

**Status of the farm jobs:**

Run	Lum.	N(QQ)	Status	Retries	N(P2)	t/evt	Node
<a href="#">122245</a>	1.035	1312	Done/Volume	0	1312	19.5	mnmc0.hep.umn.edu
<a href="#">122246</a>	1.344	1700	Done/Volume	0	1700	4.7	mnmc5.hep.umn.edu
<a href="#">122247</a>	1.809	2280	Done/Volume	0	2280	4.9	mnhepq.hep.umn.edu
<a href="#">122249</a>	1.051	1333	Done/Volume	0	1333	9.9	mnheps.hep.umn.edu
<a href="#">122251</a>	0.959	1209	Done/Volume	0	1209	11.8	mnhepv.hep.umn.edu
<a href="#">122252</a>	1.252	1588	Done/Volume	0	1588	5.3	mnmc6.hep.umn.edu
<a href="#">122253</a>	1.406	1775	Done/Volume	0	1775	6.8	mnmc4.hep.umn.edu
<a href="#">122256</a>	0.851	1077	Done/Volume	0	1077	8.0	mnmc8.hep.umn.edu
<a href="#">122263</a>	0.286	361	Done/Volume	1	361	14.4	mnmc9.hep.umn.edu

Figure 8: The Monte Carlo farm job status web page.

### 3 Minnesota Monte Carlo Farm Hardware

Presently the MN Farm consists of 22 nodes with Alpha processors running TRU64 Digital UNIX. The farm includes a Solaris machine on loan from Cornell, whose sole purpose is to serve constants to the worker nodes in the farm.

### 4 Performance

The Monte Carlo farm has been used for Monte Carlo production for CLEO III events since the fall of 2001. The scripts have proved to be remarkably easy to configure and run at the University of Minnesota site.

After the port of the CLEO III code to Linux is complete, Linux nodes will replace most of the current UNIX nodes at the University of Minnesota. At that point, new Linux farms will go into operation at other CLEO institutions. The MN Farm scripts have been designed to be easily installed and configured for operation at new sites.

### 5 Portability Issues

The scripts have been written to be portable to different UNIX or Linux type operating systems.

In writing the scripts, any system-dependent commands have been wrapped in commands defined in the module `farm_utilities.pm`. Presently only Linux or UNIX type systems are supported in the scripts. Since the CLEO III Monte Carlo code does not yet run on Linux, only TRU64 UNIX has been tested.

### 6 Conclusions and Future Plans

The MN Farm scripts have worked great for their goal of generating CLEO III Monte Carlo samples on the twenty-four processors of the University of Minnesota UNIX cluster. They are also suited to applications of comparable scale at other institutions.

Since Condor handles most of the management issues for the individual machines, one could easily handle clusters of a hundred or so machines. The scripts have not been written to handle truly distributed grid computing, and rely on machines having access to each others disks (through NFS, for example).

With a modest effort, one could expand on the present scripts to create a more powerful and general tool. Items in this “wish list” include:

- Farm physics configuration requests using a web-based form. This would allow the end user to specify the physics configuration directly rather than relying on communication with the operator. This system would be less prone to errors.

- Generalization of the farm scripts to run Monte Carlo from any experiment. An elegant solution could be written using the object-oriented capability of Perl.

This additional functionality may be added to the scripts as we prepare to generate CLEO-c Monte Carlo.

## References

- [1] The Condor Project, <http://www.cs.wisc.edu/condor/>.
- [2] "How to Set Up and Run CLEO III Code at a Remote Site", A. Smith, CBX 03-5 (2003).
- [3] The Perl Scripting Language, <http://www.perl.com/>.