

**A Practical Guide to Performing Molecular Dynamics Simulations
on the University of Tennessee Linux Cluster**

Document prepared by:

David Keffer
Computational Materials Research Group
The University of Tennessee Knoxville
<http://clausius.engr.utk.edu/cmrg/index.html>
dkeffer@utk.edu

Robert McNeil
Office of Information Technology
The University of Tennessee Knoxville
rmcneil1@utk.edu

date begun: June 8, 2006
last updated: June 8, 2006

This document can be found online at
http://clausius.engr.utk.edu/pdf/newtonusers_cmrg.pdf

Introduction

The University of Tennessee is now maintaining a cluster with 84 processors, named Newton. The connections between processors are fast enough for our molecular simulation codes that we can use this as a real parallel machine. However, this machine is much smaller than the massively parallel machines like Cheetah at ORNL. Therefore, it cannot be used for massive data production. Rather, it can function for three specific purposes. First, the cluster can be used to develop codes that will later be run on many nodes at ORNL. Second, the cluster can be used for algorithm development, which requires relatively small systems. Third, the cluster can be used to run Gaussian, which does not exist on the ORNL machines. (As of June 8, 2006, Gaussian does not exist on this UT cluster, but hopefully we will have it installed soon.)

I. Accessing the Machine

The name of the machine is *newton*. The internet address is `newton.usg.utk.edu`.

You must access *newton* via ssh. A free version of ssh for use on windows can be downloaded from www.ssh.com.

Your username is the same as your UT network ID.

II. Home Directory

Your home directory is `/home/username`. For example, for user *dkeffer*, the home directory is `/home/dkeffer`.

III. .cshrc file

You will need to create a `.cshrc` file. This file should be located in your home directory. Using `vi`, put the following two lines in your `.cshrc` file

```
setenv PATH ${PATH}:\nalias mpirun /usr/local/topspin/mpi/mpich/bin/mpirun_ssh
```

IV. FORTRAN Code Changes

In our group, we write ANSI standard FORTRAN. The code that runs on the massively parallel machines should compile and run without change on Newton. The one exception is system specific calls like *flush*. Some machines use *flush_* and some machines use *flush*. Cheetah at ORNL, for example, uses *flush_*. Newton used *flush*. Therefore, in order to compile on Newton, all instances of *flush_* must be changed to *flush*. (I moved one of our recent MD codes from Cheetah to Newton and got the code to compile and execute making only this single change.)

IV. Compilation

We assume that we are using a code written in FORTRAN 90 and using the MPI library of message passing subroutines. We also assume that you are using `csh`.

You can only compile on node `compute-1-0`. Therefore, you must first ssh to `compute-1-0` before you compile.

ssh compute-1-0

The compiler is the Intel FORTRAN compiler, located at

/opt/intel/fce/9.0/bin/ifort

The appropriate MPI header file, mpif.h is located in

/usr/local/topspin/mpi/mpich/include/mpif.h

The appropriate MPI object files are located at

/usr/local/topspin/mpi/mpich/lib64/

It turns out that the MPI library calls some routines that call the pthread library located in

/usr/lib64/libpthread.so

Therefore, you will have a makefile like that shown in Figure 1.

V. Execution in your Home Directory

As of June 8, 2006, there is no mandatory job queueing system on Newton. These notes will be changed when a job queueing system become available.

For ordinary jobs with relatively little file reading and writing, we will execute in our home directory. This is the easiest thing to do. In order to execute jobs on Newton in your home directory, you should follow the procedure given below.

1. Identify idle nodes. Visit <http://newton.usg.utk.edu/ganglia/> and identify which nodes are idle.
2. Create a pool file. This file is located in your home directory and is named pool, for example /home/dkeffer/pool. This pool file contains the names of the nodes that you will use for your job. Try to get nodes that are (i) idle and (ii) all on the same switch (See section VI. Machine Architecture below.) An example of a good pool file is given in Figures 2.a. and 2.b.
3. make sure you are back on the master node, *newton*, because mpirun jobs can only be started on *newton*.
4. Create a *working directory*, such as /home/dkeffer/work
5. Copy your executable and input files to the work directory
6. In the work directory, create a file, named *jobscript*, that contains a script that will get mpirun into the work directory and launch the executable. A sample jobscript file is given in Figure 3.
7. Run job by typing

```
mpirun -np N -hostfile /home/dkeffer/pool jobscript &
```

where N is the number of processors and matches the number of entries in the pool file. For the example pool files given in Figures 2.a. and 2.b., N=8.

8. As of June 8, 2006, there are a couple warnings that are generated N times. They are:

*Warning: No xauth data; using fake authentication data for X11 forwarding.
/usr/X11R6/bin/xauth: error in locking authority file /home/dkeffer/.Xauthority*

You can ignore these warnings.

VI. Machine Architecture

As of June 8, 2006, newton has 36 nodes. They are named

compute-1-0
compute-1-1
...
compute-1-31
compute-2-0
compute-2-1
...
compute-2-3

The 32 nodes named compute-1-0 through compute-1-31 have two processors each. The 4 nodes named compute-2-0 through compute-2-3 have four processors each. The 18 nodes spanning from compute-1-0 through compute-1-17 are on one infiniband switch. The 18 nodes spanning from compute-1-18 through compute-1-31 and compute-2-0 through compute-2-3 are on a second infiniband switch. At this time it is not recommended to run jobs that span switches. Keep your jobs on nodes that are either all on the first switch or all on the second switch.

VII. More Pertinent Information

It appears that if one of your codes crashes or is killed during execution, that some processes will continue to run. Therefore it is in everyone's best interest if you manually check after a crash to make sure that all your running programs are dead. To do this, you must

- (1) ssh into each node that you were running on
- (2) check your processes by typing `ps -u dkeffer`
- (3) kill your remaining jobs
to kill a bunch of processes named mddriver at the same time, type
`killall mddriver`
Sometimes jobs resist being killed. In that case, type
`kill -9 process id`
where the process id is reported in the first column of the `ps -u dkeffer` command.

```

.SUFFIXES: .f

COMP = /opt/intel/fce/9.0/bin/ifort

MPI_INCLUDE_DIR = /usr/local/topspin/mpi/mpich/include

OPT = -O3 -I $(MPI_INCLUDE_DIR)

MPI_LIB_DIR = /usr/local/topspin/mpi/mpich/lib64
MPI_LIB_1 = $(MPI_LIB_DIR)/libfmpich_i.so
MPI_LIB_2 = $(MPI_LIB_DIR)/libmpichf90_i.so
MPI_LIB_3 = $(MPI_LIB_DIR)/libmpichf90nc_i.so
MPI_LIB_4 = $(MPI_LIB_DIR)/libmpichfsup_i.so
MPI_LIB_5 = $(MPI_LIB_DIR)/libmpich_i.so
MPI_LIB = $(MPI_LIB_5) $(MPI_LIB_1) $(MPI_LIB_2) $(MPI_LIB_3) $(MPI_LIB_4)

PTHREAD_LIB = /usr/lib64/libpthread.so

LIB = $(PTHREAD_LIB) $(MPI_LIB)

TARGETS = mddriver

OBJS = md_mix_v70.o transport_d.o md_mpi_extras.o pbc_multi.o linkedcell.o \
self_d_corr.o transport_dmut.o onsagerL.o onsagerL_corr.o adriver.o \
md_ewald_real.o md_ewald_reciprocal.o erfc.o xinvmat.o \
make_molecules.o droulet.o intra_001.o intra_002.o intra_003.o \
ewald_real_correct.o ewald_recip_correct.o pair_corr_f.o w_cluster.o \
h3o_stats.o profiles.o restart.o

mddriver: $(OBJS)
$(COMP) -o mddriver $(OPT) $(OBJS) $(LIB)

clean:
rm -f *.o $(TARGETS) *.mod

all: $(TARGETS)

.f.o:
$(COMP) $(OPT) $(INCCH) -c $<

```

Figure 1. Newton makefile.

```
compute-2-0.local  
compute-2-0.local  
compute-2-0.local  
compute-2-0.local  
compute-2-1.local  
compute-2-1.local  
compute-2-1.local  
compute-2-1.local
```

Figure 2.a. Newton pool file example A.. This example is intended to be executed for a job that will run on a total of 8 processors, four of which are located on node compute-2-0 and four of which are located on node compute-2-1.

```
compute-1-18.local  
compute-1-18.local  
compute-1-19.local  
compute-1-19.local  
compute-1-20.local  
compute-1-20.local  
compute-1-21.local  
compute-1-21.local
```

Figure 2.b. Newton pool file example B.. This example is intended to be executed for a job that will run on a total of 8 processors, two each on nodes 1-18, 1-19, 1-20 and 1-21.

```
cd /home/dkeffer//work  
mddriver
```

Figure 3. Newton job script file for execution in your home directory. This script file changes the directory to the working directory, */home/dkeffer/work*, and launches the executable, *mddriver*.