

## STEP 1: Setting up the Environment

(NOTES: YOU ONLY NEED TO DO THIS STEP ONCE.)

Add the following lines into your C shell startup script (i.e. ~/.cshrc):

```
% cat - >> ~/.cshrc
setenv PVM_ROOT /usr/local/pvm
setenv PVM_ARCH ` $PVM_ROOT/lib/pvmgetarch `
^D
```

where “^D” is Ctrl-D and % is the command prompt. After you modify your C shell startup script, it will take effect the next time you login. To make it take effect without a new login, you can run the following command:

```
% source ~/.cshrc
```

Next generate the key for ssh:

```
% ssh-keygen
Generating 2048-bit dsa key pair
 11 o.oOoo.oOo.o
Key generated.
2048-bit dsa, lou@rumba, Wed Jul 23 2003 19:00:42 -0800
Passphrase : *****
Again      : *****
Private key saved to /export/home2/lou/.ssh2/id_dsa_2048_a
Public key saved to /export/home2/lou/.ssh2/id_dsa_2048_a.pub
```

After you generate a pair of keys with your passphrase, you need to create two files, ~/.ssh2/identification and ~/.ssh2/authorization:

```
% cat - > ~/.ssh2/identification
IdKey id_dsa_2048_a
^D
% cat - > ~/.ssh2/authorization
Key id_dsa_2048_a.pub
^D
```

## STEP 2: Start PVM Daemon

Login to one of the workstation on DECF. The server, [decf.berkeley.edu](http://decf.berkeley.edu), are not allowed to run parallel jobs as stated in the DECF user policy. Let's assume that you are on a host named “rumba”.

```
% ssh-agent tcsh
% ssh-add
Need passphrase for /export/home2/lou/.ssh2/id_dsa_2048_a (2048-bit
dsa, lou@rumba, Wed Jul 23 2003 19:00:42 -0800)
Enter passphrase: *****
Adding identity: /export/home2/lou/.ssh2/id_dsa_2048_a.pub
```

Now, you can ssh to other hosts without entering your password.

Take a look at the /etc/hosts file and pick some of the available hosts. For example, rumba, flamenco, mazurka are the hosts you would like to use. Then, run the pvm daemon by:

```
% pvm
pvm> add flamenco mazurka
add flamenco mazurka
2 successful

          HOST      DTID
flamenco  80000
mazurka   c0000

pvm> conf
3 hosts, 1 data format
          HOST      DTID      ARCH      SPEED      DSIG
rumba    40000  SUN4SOL2    1000  0x0658eb59
flamenco 80000  SUN4SOL2    1000  0x0658eb59
mazurka  c0000  SUN4SOL2    1000  0x0658eb59

pvm> quit
%
```

The add command is used to start slave PVM daemons on other hosts. The conf command is used to show several important information for your system. For example, DTID is a number that indicate the host. We will explain its use later.

NOTES: Please make sure that you are not adding a host which you never login. The first time you login a host, ssh will ask whether you want to store its hostkey. In this case, PVM may fail to start its slave daemon.

For performance reason, you want to make sure that there is no runaway process, or that the load of a machine is not heavy. The following is one way to find out the load and runaway process:

```
pvm> spawn -3 ->>top.txt /usr/local/bin/top
spawn -3 ->>top.txt /usr/local/bin/top
[1]
3 successful
t80001
tc0001
t40003
[1] finished
pvm> quit
% grep '[0-9][0-9]\.[0-9]' top.txt
[1:t80003] 17276 be144-32 1 0 19 241M 53M cpu/0 503:14
49.45% hmmscore
% rm top.txt
```

The spawn command send three (i.e. “-3”) commands out. Each command is /usr/local/bin/top and their output is concatenate into a file named “top.txt”. The grep command gives you the process (e.g. hmmscore) that use more than 10% of CPU time. The “[1:t80003]” in our example tells us which host this process is running on. The first few characters should match the first few characters of TDID. In this case, it is the host “flamenco” which is identified to have a running process or runaway process. Therefore, you want to remove it from PVM. To do so, run pvm again:

```
% pvm
```

```

pvm> delete flamenco
delete flamenco
1 successful
          HOST  STATUS
flamenco  deleted

pvm> quit
%
```

Now, PVM is running and you are ready to continue with the next step.

## STEP 3: Run MCNP in Parallel

The default setup on DECF has pointed the MCNP cross-section library to the right location. Unless you know what you are doing, there is no necessity to customize these setting. To run MCNP in parallel, the command syntax is:

```
% mcnp n=inp tasks 2x1
```

Where `inp` is the input filename and “2” is the number of processors you allocated with PVM. You can use “i=” instead of “n=”. I prefer to use “n=” here. You can also put other MCNP options such as “r=” for specifying runtpe and “C” for continuous run in the command line.

## TIPS for Running MCNP in Parallel

It is recommand to use 10 hosts for PVM because MCNP4C has two threads on the master host when the number of hosts is less than 10. However, some workstations on DECF may be occupied by other users so that you may not have 10 hosts when you run your problem. In this case, you can check whether there are runaway processes that the system administrator can kill.

The PVM version of MCNP does not work perfectly. If a particle being tracked takes too much computer time, it will slowly kill other PVM processes. Sometime, it can be avoided by the controlling the number of particles rendezvous each time and the number of secondary particles (from either variance reduction techniques or physical events):

1. The particle per cycle in the “kcode” card of a kcode problems can be used to adjust the number of particles rendezvous each time.
2. The 2<sup>nd</sup> and 5<sup>th</sup> entry of prdmp card can be used to control the number of particles rendezvous in a non-kcode problem.
3. When variance reduction methods are used, avoid to generate too many particles from one single particle. An explosion of particles will make one pvm task much longer than the other. Thus, the master pvm host think that one host was dead. In this case, the master host starts to kill the mcnp process on all slave hosts.
4. The number of secondary photon induced by neutrons can be controlled. For details, please check your MCNP manual.

## STEP 4: Cleaning up MCNP Runaway Process on PVM

Often times, when MCNP is not terminated gracefully, it stops running but the process does not exit. These runaway processes do not release the allocated memory until it is terminated. To do so, you can use the “ps” command to find out these process after you finish running MCNP.

```
% pvm
pvm> ps -ax
ps -ax
          HOST      TID      FLAG 0x  COMMAND
          rumba    (cons)   4/c  -
          rumba    t40009   6/c,f  mcnp.pvm
          mazurka  tc0010   6/c,f  mcnp.pvm
pvm>
```

To kill these runaway MCNP processes, you can use “kill” to terminate an individual runaway process or “reset” command to kill all processes run by pvm.

```
% pvm
pvm> reset
pvm>
```

When you are done with all your calculation, please stop PVM by running a halt command.

```
% pvm
pvm> halt
halt
Terminated
%
```

Sometimes, pvm daemon is not stopped gracefully. For example, people shut down the computer by unplugging it from the power. There will be a pvmd file left behind. You need to remove the lock file before you can start pvm on this computer. It is located at /tmp directory. Its name is pvmd followed by your uid. (uid means User ID.) You can find out your uid by:

```
% echo $uid
1320
```

You can remove these files by:

```
% rm /tmp/pvm[dl].$uid
```

This will remove the pvmd.1320 and pvml.1320 for that particular machine. To remove these files on other hosts, you have to login to these machines and repeat this command.

You may also find some temporary files created by MCNP when the program is running. These files will not be removed if your MCNP does not stop gracefully. These files are stored under your home directory and their names begin with “tmp.” You may want to clean them up too.