Cluster Setup

Here, I provide some tips for system and software installations and configurations of a Rocks cluster. This vignette completes the other manual file for running the whole MVR procedure in R.

You might encounter problems on different system following these steps. It is recommended to ask help from your system administrator.

1. Some system requirements

• The following assumes that your cluster has already been installed under Unix/Linux and setup. Basically, you need to install in your system (with correct configurations) the PVM or Open MPI libraries of communications protocol for parallel networking of computers (PVM) or parallel programming of computers (MPI). You might want to ask your system administrator to do this, otherwise you must login as root to have administrator privileges. PVM (or MPI) must be installed on all nodes or they can be exported from a network file system. You also want to use ssh for safely connecting between each node. You can install them directly from the Yum repositories, or download the latest source codes from their official websites:
PVM : from netlib at http://www.netlib.org/pvm3/index.html
MPI : from http://www.open-mpi.org/.

When installing, target the download into a master node shared directory with slaves nodes, e.g. ~/state/partition1/apps/R/. From the UNIX command line of each node, install by typing:

```
yum -y install pvm
```

• R (>= 2.13.0) must be installed on all nodes.
You can download it from http://cran.r-project.org

• The following R packages are also required:
"rpvm" : if R interface to PVM (Parallel Virtual Machine).
You can download it only from http://cran.r-project.org
or "Rmpi" : if R Interface to MPI (Message-Passing Interface).
You can download it from http://cran.r-project.org or
http://www.stats.uwo.ca/faculty/yu/Rmpi

"snow" : Simple Network Of Workstations.
You can download it from http://cran.r-project.org.
Tutorials on how to use it can also be found at:
http://www.stat.uiowa.edu/~luke/R/cluster/cluster.html
and http://www.sfu.ca/~sblay/R/snow.html

2. Some additional system settings and installations

• Assuming the above installed on your system, first thing to do is to set some user specific environment variables. From the UNIX command line of the master node, create two environmental variables "NUMCPU", and "HOSTS", containing respectively the number and hostnames (or IP addresses) of the slave nodes (machines) for later adding to the virtual machine. The master node host name should (by default) be stored in the environment variable "HOSTNAME".

Set the environment variables by adding the following lines in your startup script file (.bashrc, .cshrc, .bash_profile, etc...):

```
NUMCPU=`cat /proc/cpuinfo | grep processor | wc -l`
HOSTS=`tail -n4 /etc/hosts | awk '{print $3}'`
```

and export some paths (I found the last two exports not required):

```
export PVM_ROOT=/usr/share/pvm3
export PVM_ARCH=LINUX64
export PVM_RSH=/usr/bin/ssh
#export R_SNOW_LIB=/usr/lib64/R/library
#export R_LIBS=/usr/lib64/R/library
```

• Note that there are dependencies to take care of prior to installing R: Perl, R-core, R-devel, libRmath, and libRmath-devel. See R manual "R Installation and Administration".

• The R package snow suggests "rsprng" for creating separate Stream of Parallel RNG (SPRNG) per node, and distributing the stream states to the nodes. This requires installing the "gmp" and "gmp-devel" Yum packages beforehand. From the UNIX command line of each node, install them in the master shared folder from the yum repositories:

```
yum -y install gmp gmp-devel
```

Next, download sprng from http://sprng.cs.fsu.edu/ and compile the source files. The following steps assume you've done so.

3. Using and testing the cluster in R

Assuming the above is done, here are the steps to create/configure the virtual machine and use the cluster in R.

==========
• 3.1 Starting the PVM
==========

Starting a workstation cluster depends explicitly on the underlying communication mechanism. A snow cluster is started by calling the makeCluster() function (see next). Details of the call currently vary slightly depending on the type of cluster (PVM or MPI) .PVM and MPI clusters may also need some preliminary configurations to start the PVM or MPI systems.

To start a PVM-based cluster, you first need to start the PVM. I prefer to do this from within a R session using e.g. the "rpvm" R interface (see next). Alternatively, it can be done using the PVM console from the UNIX command line (for details, see: http://www.sfu.ca/~sblay/R/snow.html). From within an R session, load in the R package "snow" and start a pvmd3 process of a new local virtual machine on the master. pvmd3 is a daemon process which coordinates UNIX hosts in the virtual machine. It returns as soon as the pvmd is started and ready for work:

```
R> library(snow)
R> masterhost <- Sys.getenv("HOSTNAME")
R> .PVM.start.pvmd(hosts = masterhost)
```

To create a virtual machine of 1 master node with at least one slave node, and spawn the daemon process, add a vector of host names and add them to the virtual machine. From within an R session:

```
R> slavehosts <- unlist(strsplit(Sys.getenv("HOSTS"), split="\\\n"))
R> .PVM.addhosts(slavehosts)
```

==========
• 3.2 Initializing the cluster
==========

Once the PVM is running, retrieve relevant parameters needed for creating and intializing the cluster. From within an R session:

```
R> cpus <- as.numeric(Sys.getenv("NUMCPU"))
R> cl.type <- getClusterOption("type")
R> cl.nodes <- length(slavehosts) + 1
R> cl.cpus <- cl.nodes * cpus
R> cl.homo <- getClusterOption("homogeneous")
R> cl.script <- getClusterOption("useRscript")
R> cl.out <- paste(getwd(), "/your_file_name.log", sep="")
```

The following depends on whether this is not already done internally in some package using snow for configuring their cluster. If not, use the snow function makeCluster() to create and initialize the cluster :

```
R> cl <- makeCluster(spec=cl.cpus,
                     type=cl.type,
                     homogeneous=cl.homo,
                     useRscript=cl.script,
                     outfile = cl.out,
                     verbose=T)
```

where:
- spec = integer specifying the total number of CPU cores, counting the masternode.
- type = character vector specifying the cluster type ("SOCK", "PVM", "MPI"). homogeneous = logical to be set to 'FALSE' for inhomogeneous clusters (default to TRUE).
- useRscript = logical to be set to 'FALSE' if non-R script is used (default to TRUE).
- outfile = character vector of output log file name for the slavenodes.

==========
• 3.3 Initializing random number generation (optional)
==========
For random number generation, create separate Stream of Parallel RNG (SPRNG) per node. This is done from within the R session by distributing the stream states to the nodes like this:

```
R> clusterSetupSPRNG(cl=cl)
```

==========
• 3.4 Testing the cluster (optional)
==========
To test if the cluster is working, check the node names and machine types:

```
R> clusterCall(cl=cl, function() Sys.info()[c("nodename","machine")])
```

You're supposed to see the node names and machine types in an R object of type list of size "cl.cpus". Check the random number generation by running e.g.:

```
R> clusterCall(cl, runif, 3)
```

==========
• 3.5 Exporting/Evaluating
==========
The following depends on whether this is not already done internally in some package using snow for configuring their cluster. If not, you must assign the global values on the master of all (listable) R objects of the worspace to variables of the same names in the global environments of each node before using the cluster. To make sure not to forget anything, I usually export the whole global environment. Depending on how large is your workspace it can take up to a few minutes. This is done via the snow function clusterExport().

```
R> clusterExport(cl=cl, list=as.list(ls(.GlobalEnv)))
```

Also, you must do the same for required R libraries. This is done via clusterEvalQ() which evaluates a literal expression on each cluster node.

```
R> clusterEvalQ(cl=cl, expr=library("<libname_here>"))
```

==========
• 3.6 Stopping the cluster
==========
The following also depends on whether this is not already done internally in some package using snow for configuring their cluster. If not, to cleanly stop the

cluster, close any remaining connections between machines and ensure that all slave processes are shut down:

```
R> stopCluster(cl)
```

• 3.7 Shutting the PVM down

To shut the PVM down, first delete the host machine(s) (pointed to in hosts) from the existing configuration of machines making up the virtual machine, then the master.

```
R> .PVM.delhosts(slavehosts)
R> .PVM.delhosts(masterhost)
```

Next, shut down the entire PVM system including remote tasks, remote pvmds, the local tasks (including the calling task) and the local pvmd.
Note: this also kills the parent R session!

```
R> .PVM.halt()
```

That's it! Now you can start to enjoy the parallel computing.