

## Webpage

<http://www.msg.ameslab.gov/GAMESS/GAMESS.html>

## Version

May 1, 2012

## Tools for Compiling

- Intel Compiler 12.1.2.273
- Intel MPI 4.0.2.003

## Necessary Files for Compiling

- gamess-2012May01.tar.gz (from GAMESS webpage)
- rungms.patch

## Content of rungms.patch

```
--- rungms.orig 2012-08-14 16:33:58.481954000 +0900
+++ rungms 2012-08-14 16:34:15.780440000 +0900
@@ -56,10 +56,10 @@
# both Sun Grid Engine (SGE), and Portable Batch System (PBS).
# See also a very old LoadLeveler "ll-gms" for some IBM systems.
#
-set TARGET=sockets
-set SCR=/scr/$USER
-set USERSCR=~$USER/scr
-set GMSPATH=/u1/mike/gamess
+set TARGET=mpi
+set SCR=/work/users/$USER/scr.$$
+if (! -d $SCR) mkdir $SCR
+set GMSPATH=/local/apl/pg/gamess2012May01
#
set JOB=$1 # name of the input file xxx.inp, give only the xxx part
set Verno=$2 # revision number of the executable created by 'lkd' step
@@ -89,16 +89,10 @@
    uniq $TMPDIR/machines
endif
if ($SCHED == PBS) then
- set SCR=/scratch/$PBS_JOBID
+# set SCR=/scratch/$PBS_JOBID
    echo "PBS has assigned the following compute nodes to this run:"
    uniq $PBS_NODEFILE
endif
-#
-echo "Available scratch disk space (Kbyte units) at beginning of the job is"
-df -k $SCR
-echo "GAMESS temporary binary files will be written to $SCR"
-echo "GAMESS supplementary output files will be written to $USERSCR"
-
# this added as experiment, February 2007
# its intent is to detect large arrays allocated off the stack
limit stacksize 8192
@@ -131,6 +125,15 @@
    endif
endif

+set dir=`dirname $JOB`
+set USERSCR=`cd $dir; pwd`
+
+#
+echo "Available scratch disk space (Kbyte units) at beginning of the job is"
```

```

+df -k $SCR
+echo "GAMESS temporary binary files will be written to $SCR"
+echo "GAMESS supplementary output files will be written to $USERSCR"
+
# define many environment variables setting up file names.
# anything can be overridden by a user's own choice, read 2nd.
source $GMSPATH/gms-files.csh
@@ -583,13 +586,13 @@

#
if ($NCPUS == 1) then
- echo "-n $NPROCS -host `hostname` /home/mike/games/gamess.$VERNO.x" >> $PROCFILE
+ echo "-n $NPROCS -host `hostname` $GMSPATH/games/gamess.$VERNO.x" >> $PROCFILE
else
if ($NNODES == 1) then
# when all processes are inside a single node, it is simple!
# all MPI processes, whether compute processes or data servers,
# are just in this node. (note: NPROCS = 2*NCPUS!)
- echo "-n $NPROCS -host `hostname` /home/mike/games/gamess.$VERNO.x" >> $PROCFILE
+ echo "-n $NPROCS -host `hostname` $GMSPATH/games/gamess.$VERNO.x" >> $PROCFILE
else
# For more than one node, we want PPN compute processes on
# each node, and of course, PPN data servers on each.
@@ -602,7 +605,7 @@
while ($n <= $NNODES)
set host=`sed -n -e "$n p" $HOSTFILE`
set host=${host[1]}
- echo "-n $PPN2 -host $host /home/mike/games/gamess.$VERNO.x" >> $PROCFILE
+ echo "-n $PPN2 -host $host $GMSPATH/games/gamess.$VERNO.x" >> $PROCFILE
@ n++
end
endif
@@ -666,7 +669,7 @@
setenv I_MPI_DEBUG 0
setenv I_MPI_STATS 0
setenv I_MPI_FABRICS dapl
- setenv I_MPI_DAT_LIBRARY libdat2.so
+ setenv I_MPI_DAT_LIBRARY libdat2.so.2
# in case someone wants to try the "tag matching interface",
# an option which unfortunately ignores the WAIT_MODE in 4.0.2!
#--setenv I_MPI_FABRICS tmi
@@ -758,7 +761,7 @@
# Now, at last, we can actually kick-off the MPI processes...
#
echo "MPI kickoff will run GAMESS on $NCPUS cores in $NNODES nodes."
- echo "The binary to be executed is /home/mike/games/gamess.$VERNO.x"
+ echo "The binary to be executed is $GMSPATH/games/gamess.$VERNO.x"
echo "MPI will run $NCPUS compute processes and $NCPUS data servers,"
echo " placing $PPN of each process type onto each node."
echo "The scratch disk space on each node is $SCR, with free space"
@@ -790,7 +793,7 @@
setenv I_MPI_HYDRA_ENV all
setenv I_MPI_PERHOST $PPN2
mpiexec.hydra -f $PROCFILE -n $NPROCS \
- /home/mike/games/gamess.$VERNO.x < /dev/null
+ $GMSPATH/games/gamess.$VERNO.x < /dev/null
unset echo
breaksw
case default:
@@ -1040,6 +1043,7 @@
echo Files used on the master node $master were:
ls -lF $SCR/$JOB.*
rm -f $SCR/$JOB.F*
+rm -f $SCR/$JOB.nodes.mpd
#

```

```

# Clean/Rescue any files created by the VB2000 plug-in
if (-e $SCR/$JOB.V84) mv $SCR/$JOB.V84 $USERSCR
@@ -1051,13 +1055,13 @@
if (-e $SCR/$JOB.molf) mv $SCR/$JOB.molf $USERSCR
if (-e $SCR/$JOB.mkl) mv $SCR/$JOB.mkl $USERSCR
if (-e $SCR/$JOB.xyz) mv $SCR/$JOB.xyz $USERSCR
-ls $SCR/${JOB}.*.cube > $SCR/${JOB}.lis
+(ls $SCR/${JOB}.*.cube > $SCR/${JOB}.lis) >& /dev/null
if (! -z $SCR/${JOB}.lis) mv $SCR/${JOB}.*.cube $USERSCR
rm -f $SCR/${JOB}.lis
-ls $SCR/${JOB}.*.grd > $SCR/${JOB}.lis
+(ls $SCR/${JOB}.*.grd > $SCR/${JOB}.lis) >& /dev/null
if (! -z $SCR/${JOB}.lis) mv $SCR/${JOB}.*.grd $USERSCR
rm -f $SCR/${JOB}.lis
-ls $SCR/${JOB}.*.csv > $SCR/${JOB}.lis
+(ls $SCR/${JOB}.*.csv > $SCR/${JOB}.lis) >& /dev/null
if (! -z $SCR/${JOB}.lis) mv $SCR/${JOB}.*.csv $USERSCR
rm -f $SCR/${JOB}.lis
#
@@ -1091,37 +1095,37 @@
# We have inherited a file of unique node names from above.
# There is an option to rescue the output files from group DDI runs,
# such as FMO, in case you need to see the other group's outputs.
-if ($TARGET == mpi) then
- set nnodes=`wc -l $HOSTFILE`
- set nnodes=$nnodes[1]
- @ n=1
- set master=`hostname`
- # burn off the .local suffix in our cluster's hostname
- set master=$master:r
- while ($n <= $nnodes)
- set host=`sed -n -e "$n p" $HOSTFILE`
- # in case of openMPI, unwanted stuff may follow the hostname
- set host=$host[1]
- if ($host != $master) then
- echo Files used on node $host were:
- #-----FMO rescue-----
- #--if ($GDDIjob == true) then
- #-- echo "===== OUTPUT from node $host is ====="
- #-- ssh $host -l $USER "cat $SCR/$JOB.F06*"
- #--endif
- #-----FMO rescue-----
- ssh $host -l $USER "ls -l $SCR/$JOB.*"
- ssh $host -l $USER "rm -f $SCR/$JOB.*"
- endif
- @ n++
- end
-# clean off the last file on the master's scratch disk.
- rm -f $HOSTFILE
- #
- if ($?_MPI_STATS) then
- if ($!_MPI_STATS > 0) mv $SCR/stats.txt ~/$JOB.$NCPUS.stats
- endif
-endif
+###if ($TARGET == mpi) then
+### set nnodes=`wc -l $HOSTFILE`
+### set nnodes=$nnodes[1]
+### @ n=1
+### set master=`hostname`
+### # burn off the .local suffix in our cluster's hostname
+### set master=$master:r
+### while ($n <= $nnodes)
+### set host=`sed -n -e "$n p" $HOSTFILE`
+### # in case of openMPI, unwanted stuff may follow the hostname
+### set host=$host[1]

```

```

##### if ($host != $master) then
#####     echo Files used on node $host were:
#####     #-----FMO rescue-----
#####     #--if ($GDDIjob == true) then
#####     #-- echo "===== OUTPUT from node $host is ====="
#####     #-- ssh $host -l $USER "cat $SCR/$JOB.F06*"
#####     #--endif
#####     #-----FMO rescue-----
#####     ssh $host -l $USER "ls -l $SCR/$JOB.*"
#####     ssh $host -l $USER "rm -f $SCR/$JOB.*"
#####     endif
#####     @ n++
##### end
##### clean off the last file on the master's scratch disk.
##### rm -f $HOSTFILE
##### #
##### if ($?I_MPI_STATS) then
#####     if ($I_MPI_STATS > 0) mv $SCR/stats.txt ~/$JOB.$NCPUS.stats
#####     endif
#####endif
#####
##### IBM SP cleanup code...might need to be something other than 'rsh'.
#####

```

## Procedure of Compiling

```

#!/bin/csh -f
umask 022
set file_gamess=/home/users/${USER}/build/gamess2012May01/gamess-2012May01.tar.gz
set work=/work/users/${USER}
set gamess=gamess2012May01
set patch_rungms=/home/users/${USER}/build/gamess2012May01/ccpg/rungms.patch
#-----
cd ${work}
if (-d ${gamess}) then
    mv ${gamess} ${gamess}-erase
    rm -rf ${gamess}-erase &
endif
#-----
tar xzf ${file_gamess}
mv gamess ${gamess}
cd ${work}/${gamess}
expect <<EXPECT
spawn ./config
expect "After the new window is open"
send "\r"
expect "please enter your target machine name:"
send "linux64\r"
expect "GAMESS directory?"
send "\r"
expect "GAMESS build directory?"
send "\r"
expect "Version?"
send "\r"
expect "Please enter your choice of FORTRAN:"
send "ifort\r"
expect "Version?"
send "12\r"
expect "hit <return> to continue after digesting this message."
send "\r"
expect "hit <return> to continue to the math library setup."
send "\r"
expect "Enter your choice of 'mkl' or 'atlas' or 'acml' or 'none':"
send "mkl\r"
expect "MKL pathname?"
send "/opt/intel/mkl\r"

```

```
expect "MKL version (or 'skip')?"
send "skip\r"
expect "hit <return> after you have digested this warning."
send "\r"
expect "please hit <return> to compile the GAMESS source code activator"
send "\r"
expect "please hit <return> to set up your network for Linux clusters."
send "\r"
expect "communication library ('sockets' or 'mpi')?"
send "mpi\r"
expect "Enter MPI library (impi, mvapich2, mpt, sockets):"
send "impi\r"
expect "Please enter your impi's location:"
send "/opt/intel/impi/4.0.2.003\r"
expect eof
EXPECT
#-----
cd ${work}/${games}/ddi
./compddi >& compddi.log
cd ${work}/${games}
./compall >& compall.log
./lked >& lked.log
#-----
chmod -R o-rwx source object
find . -name "src" | xargs chmod -R o-rwx
#-----
patch -p0 < ${patch_rungms}
```