GAMESS-2013May01 for UV2000

Webpage

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

Version

May 1, 2013

Tools for Compiling

- Intel Compiler 13.1.1.163

Needed files for Compiling

- gamess-2013May01.tar.gz (from GAMESS webpage)
- rungms.patch

Content of rungms.patch

```
--- rungms.orig    2013-05-18 01:15:11.000000000 +0900
+++ rungms    2013-08-21 13:28:19.759927571 +0900
@@ -60,9 +60,9 @@
 #       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 SCR=/work/users/$USER/scr.$$
+if (! -d $SCR) mkdir $SCR
+set GMSPATH=/local/apl/uv/gamess2013May01
 #
 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 'lked' step
@@ -92,16 +92,11 @@
    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
@@ -134,6 +129,15 @@
    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 $GMSPATH/gms-files.csh
@@ -265,53 +269,55 @@
 #                     NODE= physical enclosure (box/blade)
 #
 #       1. Sequential execution is sure to be on this very same host
-   if ($NCPUS == 1) then
+###   if ($NCPUS == 1) then
+###      set NNODES=1
+###      set HOSTLIST=(`hostname`)
+###   endif
+####
+####       2. This is an example of how to run on a multi-core SMP enclosure,
+####          where all CPUs (aka COREs) are inside a -single- NODE.
+####     At other locations, you may wish to consider some of the examples
+####     that follow below, after commenting out this ISU specific part.
+###   if ($NCPUS > 1) then
+###      switch (`hostname`)
+###        case se.msg.chem.iastate.edu:
+###        case sb.msg.chem.iastate.edu:
+###          if ($NCPUS > 2) set NCPUS=2
+###          set NNODES=1
+###          set HOSTLIST=(`hostname`:cpus=$NCPUS)
+###          breaksw
+###        case br.msg.chem.iastate.edu:
+###          if ($NCPUS > 4) set NCPUS=4
+###          set NNODES=1
+###          set HOSTLIST=(`hostname`:cpus=$NCPUS)
+###          breaksw
+###        case cd.msg.chem.iastate.edu:
+###        case zn.msg.chem.iastate.edu:
+###        case ni.msg.chem.iastate.edu:
+###        case co.msg.chem.iastate.edu:
+###        case pb.msg.chem.iastate.edu:
+###        case bi.msg.chem.iastate.edu:
+###        case po.msg.chem.iastate.edu:
+###        case at.msg.chem.iastate.edu:
+###        case sc.msg.chem.iastate.edu:
+###          if ($NCPUS > 4) set NCPUS=4
+###          set NNODES=1
+###          set HOSTLIST=(`hostname`:cpus=$NCPUS)
+###          breaksw
+###        case ga.msg.chem.iastate.edu:
+###        case ge.msg.chem.iastate.edu:
+###        case gd.msg.chem.iastate.edu:
+###          if ($NCPUS > 6) set NCPUS=6
+###          set NNODES=1
+###          set HOSTLIST=(`hostname`:cpus=$NCPUS)
+###          breaksw
+###        default:
+###          echo I do not know how to run this node in parallel.
+###          exit 20
+###      endsw
+###   endif
       set NNODES=1
-      set HOSTLIST=(`hostname`)
-   endif
-#
-#       2. This is an example of how to run on a multi-core SMP enclosure,
-#          where all CPUs (aka COREs) are inside a -single- NODE.
-#     At other locations, you may wish to consider some of the examples
```

```
-#     that follow below, after commenting out this ISU specific part.
-   if ($NCPUS > 1) then
-     switch (`hostname`)
-       case se.msg.chem.iastate.edu:
-       case sb.msg.chem.iastate.edu:
-         if ($NCPUS > 2) set NCPUS=2
-         set NNODES=1
-         set HOSTLIST=(`hostname`:cpus=$NCPUS)
-         breaksw
-       case br.msg.chem.iastate.edu:
-         if ($NCPUS > 4) set NCPUS=4
-         set NNODES=1
-         set HOSTLIST=(`hostname`:cpus=$NCPUS)
-         breaksw
-       case cd.msg.chem.iastate.edu:
-       case zn.msg.chem.iastate.edu:
-       case ni.msg.chem.iastate.edu:
-       case co.msg.chem.iastate.edu:
-       case pb.msg.chem.iastate.edu:
-       case bi.msg.chem.iastate.edu:
-       case po.msg.chem.iastate.edu:
-       case at.msg.chem.iastate.edu:
-       case sc.msg.chem.iastate.edu:
-         if ($NCPUS > 4) set NCPUS=4
-         set NNODES=1
-         set HOSTLIST=(`hostname`:cpus=$NCPUS)
-         breaksw
-       case ga.msg.chem.iastate.edu:
-       case ge.msg.chem.iastate.edu:
-       case gd.msg.chem.iastate.edu:
-         if ($NCPUS > 6) set NCPUS=6
-         set NNODES=1
-         set HOSTLIST=(`hostname`:cpus=$NCPUS)
-         breaksw
-       default:
-         echo I do not know how to run this node in parallel.
-         exit 20
-     endsw
-   endif
+      set HOSTLIST=(`hostname`:cpus=$NCPUS)
  #
  #       3. How to run in a single computer, namely the "localhost", so
  #          this computer needn't have a proper Internet name.
@@ -363,90 +369,90 @@
  #          names into the HOSTLIST string for the kickoff program,
  #          and to request the host name of the fast network adapters.
  #
-   if ($?PBS_JOBID) then
-#
-#         The IBM cluster has two Gigabit adapters in each 4-way SMP,
-#         while the AXP cluster is based on a Myrinet network.
-     if (`uname` == AIX)   set NETEXT=".gig,.gig2"
-     if (`uname` == Linux) set NETEXT=".myri"
-#
-#         repeated host names in the PBS host file indicate being assigned
-#         CPUs in the same SMP enclosure, which we must count up correctly.
-#         Fortunately PBS gives duplicate host names in a row, not scrambled.
-#         The number of hosts in the PBS node file (nmax) should equal the
-#         requested processor count, NCPUS.  We need to count duplicates
-#         in order to learn the number of SMP enclosures, NNODES, and how
-#         many CPUs inside each SMP were assigned (NSMPCPU).  For example,
-#         if we are assigned the host names "a a a b b c c c" we must build
-#         the string "a:cpus=3 b:cpus=2 c:cpus=3" so that ddikick.x will
-#         know the SMP structure of the assigned node names.  (C-shell handles
-#         variable substitution followed by colon gracefully by ${HOST}:cpus.)
```

```
-#
-     set HOSTLIST=()
-     set nmax=`wc -l $PBS_NODEFILE`
-     set nmax=$nmax[1]
-     if ($nmax != $NCPUS) then
-       echo There is processor count confusion
-       exit
-     endif
-#           1st host in the list is sure to be a new SMP enclosure
-     set MYNODE=`sed -n -e "1 p" $PBS_NODEFILE`
-     set MYNODE=`echo $MYNODE | awk '{split($0,a,"."); print a[1]}'`
-#           IPROC counts assigned processors (up to NCPUS),
-#           NNODES counts number of SMP enclosures.
-#           NSMPCPU counts processors in the current SMP enclosure
-     @ IPROC = 2
-     @ NNODES = 1
-     @ NSMPCPU = 1
-     set spacer1=":cpus="
-     set spacer2=":netext="
-     while($IPROC <= $nmax)
-       set MYPROC=`sed -n -e "$IPROC p" $PBS_NODEFILE`
-       set MYPROC=`echo $MYPROC | awk '{split($0,a,"."); print a[1]}'`
-       if($MYPROC != $MYNODE) then
-         set HOSTLIST = ($HOSTLIST $MYNODE$spacer1$NSMPCPU$spacer2$NETEXT)
-         set MYNODE=$MYPROC
-         @ NSMPCPU = 0
-         @ NNODES++
-       endif
-       @ IPROC++
-       @ NSMPCPU++
-     end
-     set HOSTLIST = ($HOSTLIST $MYNODE$spacer1$NSMPCPU$spacer2$NETEXT)
-   endif
-#
-#       we have now finished setting up a correct HOSTLIST.
-#       uncomment the next two if you are doing script debugging.
-#--echo "The generated host list is"
-#--echo $HOSTLIST
-#
-#
-#       choose remote shell execution program.
-#    Parallel run do initial launch of GAMESS on remote nodes by the
-#    following program.  Note that the authentication keys for ssh
-#    must have been set up correctly.
-#    If you wish, choose 'rsh' using .rhosts authentication on next line.
-   setenv DDI_RSH ssh
-#
-   if($DDI_RSH == ssh) then
-     setenv DDI_RCP scp
-   else
-     setenv DDI_RCP rcp
-   endif
-
-#       One way to be sure that the master node of each subgroup
-#       has its necessary copy of the input file is to stuff a
-#       copy of the input file onto every single node right here.
-   if ($GDDIjob == true) then
-     @ n=2   # master in master group already did 'cp' above
-     while ($n <= $NNODES)
-       set host=$HOSTLIST[$n]
-       set host=`echo $host | cut -f 1 -d :` # drop anything behind a colon
-       echo $DDI_RCP $SCR/$JOB.F05 ${host}:$SCR/$JOB.F05
-            $DDI_RCP $SCR/$JOB.F05 ${host}:$SCR/$JOB.F05
-       @ n++
-     end
```

```
-   endif
+###    if ($?PBS_JOBID) then
+####
+####        The IBM cluster has two Gigabit adapters in each 4-way SMP,
+####        while the AXP cluster is based on a Myrinet network.
+###     if (`uname` == AIX)   set NETEXT=".gig,.gig2"
+###     if (`uname` == Linux) set NETEXT=".myri"
+####
+####        repeated host names in the PBS host file indicate being assigned
+####        CPUs in the same SMP enclosure, which we must count up correctly.
+####        Fortunately PBS gives duplicate host names in a row, not scrambled.
+####        The number of hosts in the PBS node file (nmax) should equal the
+####        requested processor count, NCPUS.  We need to count duplicates
+####        in order to learn the number of SMP enclosures, NNODES, and how
+####        many CPUs inside each SMP were assigned (NSMPCPU).  For example,
+####        if we are assigned the host names "a a a b b c c c" we must build
+####        the string "a:cpus=3 b:cpus=2 c:cpus=3" so that ddikick.x will
+####        know the SMP structure of the assigned node names.  (C-shell handles
+####        variable substitution followed by colon gracefully by ${HOST}:cpus.)
+####
+###     set HOSTLIST=()
+###     set nmax=`wc -l $PBS_NODEFILE`
+###     set nmax=$nmax[1]
+###     if ($nmax != $NCPUS) then
+###       echo There is processor count confusion
+###       exit
+###     endif
+####          1st host in the list is sure to be a new SMP enclosure
+###     set MYNODE=`sed -n -e "1 p" $PBS_NODEFILE`
+###     set MYNODE=`echo $MYNODE | awk '{split($0,a,"."); print a[1]}'`
+####          IPROC counts assigned processors (up to NCPUS),
+####          NNODES counts number of SMP enclosures.
+####          NSMPCPU counts processors in the current SMP enclosure
+###     @ IPROC = 2
+###     @ NNODES = 1
+###     @ NSMPCPU = 1
+###     set spacer1=":cpus="
+###     set spacer2=":netext="
+###     while($IPROC <= $nmax)
+###       set MYPROC=`sed -n -e "$IPROC p" $PBS_NODEFILE`
+###       set MYPROC=`echo $MYPROC | awk '{split($0,a,"."); print a[1]}'`
+###       if($MYPROC != $MYNODE) then
+###         set HOSTLIST = ($HOSTLIST $MYNODE$spacer1$NSMPCPU$spacer2$NETEXT)
+###         set MYNODE=$MYPROC
+###         @ NSMPCPU = 0
+###         @ NNODES++
+###       endif
+###       @ IPROC++
+###       @ NSMPCPU++
+###     end
+###     set HOSTLIST = ($HOSTLIST $MYNODE$spacer1$NSMPCPU$spacer2$NETEXT)
+###   endif
+####
+####       we have now finished setting up a correct HOSTLIST.
+####       uncomment the next two if you are doing script debugging.
+####--echo "The generated host list is"
+####--echo $HOSTLIST
+####
+####
+####       choose remote shell execution program.
+####    Parallel run do initial launch of GAMESS on remote nodes by the
+####    following program.  Note that the authentication keys for ssh
+####    must have been set up correctly.
+####    If you wish, choose 'rsh' using .rhosts authentication on next line.
+###   setenv DDI_RSH ssh
```

```
+####
+###   if($DDI_RSH == ssh) then
+###      setenv DDI_RCP scp
+###   else
+###      setenv DDI_RCP rcp
+###   endif
+###
+####        One way to be sure that the master node of each subgroup
+####        has its necessary copy of the input file is to stuff a
+####        copy of the input file onto every single node right here.
+###   if ($GDDIjob == true) then
+###      @ n=2   # master in master group already did 'cp' above
+###      while ($n <= $NNODES)
+###        set host=$HOSTLIST[$n]
+###        set host=`echo $host | cut -f 1 -d :` # drop anything behind a colon
+###        echo $DDI_RCP $SCR/$JOB.F05 ${host}:$SCR/$JOB.F05
+###             $DDI_RCP $SCR/$JOB.F05 ${host}:$SCR/$JOB.F05
+###        @ n++
+###      end
+###   endif

 #
 #       Just make sure we have the binaries, before we try to run
@@ -465,9 +471,10 @@
 #
   if ($DDI_VER == new) then
     set echo
-     $GMSPATH/ddikick.x $GMSPATH/gamess.$VERNO.x $JOB \
-         -ddi $NNODES $NCPUS $HOSTLIST \
-         -scr $SCR < /dev/null
+      $GMSPATH/ddikick.x $GMSPATH/gamess.$VERNO.x $JOB \
+          -ddi $NNODES $NCPUS $HOSTLIST \
+          -scr $SCR < /dev/null
+        $GMSPATH/ddikick.x dplace -s1 $GMSPATH/gamess.$VERNO.x $JOB -ddi $NNODES $NCPUS ${HOSTLIST} -scr
$SCR < /dev/null
     unset echo
   else
     set path=($GMSPATH $path)
@@ -1447,13 +1454,13 @@
 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
 #
```

Procedure of Compiling

```
#!/bin/csh -f
umask 022
set file_gamess=/home/users/${USER}/build/gamess2013May01/gamess-2013May01.tar.gz
set work=/work/users/${USER}
set gamess=gamess2013May01
set patch_rungms=/home/users/${USER}/build/gamess2013May01/ccuv/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 "10.2.5.035\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 "sockets\r"
expect "Do you want to try LIBCCHEM"
send "no\r"
expect eof
EXPECT
#------------------------------------------------------------------------
cd ${work}/${gamess}/ddi
sed -e 's/MAXCPUS  = 16/MAXCPUS  = 1024/' -e 's/MAXNODES = 256/MAXNODES = 2/' compddi > compddi1024
csh ./compddi1024 >& compddi.log
mv ddikick.x ../
cd ${work}/${gamess}
./compall >& compall.log
./lked >& lked.log
#------------------------------------------------------------------------
chmod -R o-rwx source object
find . -name "src" | xargs chmod -R o-rwx
#------------------------------------------------------------------------
patch -p0 < ${patch_rungms}
```