搭建高性能计算环境（四）、应用软件的安装之VASP

1，将需要的软件包上传vasp.5.2.12.tar.gz、vasp.5.lib.tar.gz、benchmark.Hg.tar.gz。

2，创建vasp目录并解压软件包。

```
mkdir /opt/vasp
cd /opt/vasp
tar xvf ~/vasp.5.lib.tar.gz
tar xvf ~/vasp.5.2.12.tar.gz
```

3,编译lib

```
cd vasp.5.lib/
cp makefile.linux_ifc_P4 makefile
修改makefile
make
```

修改后的makefile：

```
.SUFFIXES: .inc .f .F
#-----------------------------------------------------------------------
# Makefile for Portland Group F90/HPF compiler
# the makefile was tested only under Linux on Intel platforms
# however it might work on other platforms as well
#
# this release of vasp.4.lib contains lapack v2.0
# this can be compiled with pgf90 compiler if the option -O1 is used
#
# Mind: one user reported that he had to copy preclib.F diolib.F
#  dlexlib.F and drdatab.F to the directory vasp.4.4, compile the files
#  there and link them directly  into vasp
#  for no obvious reason these files could not be linked from the library
#
#-----------------------------------------------------------------------

# C-preprocessor
CPP     = gcc -E -P -C $*.F >$*.f
FC=ifort

CFLAGS = -O3
FFLAGS = -O3 -ip -no-prec-div -funrool-loops -xHost -heap-arrays 64
FREE   =  -FR

DOBJ =  preclib.o timing_.o derrf_.o dclock_.o  diolib.o dlexlib.o drdatab.o


#-----------------------------------------------------------------------
# general rules
#-----------------------------------------------------------------------

libdmy.a: $(DOBJ) lapack_double.o linpack_double.o lapack_atlas.o
        -rm libdmy.a
        ar vq libdmy.a $(DOBJ)

# files which do not require autodouble
```

```
lapack_min.o: lapack_min.f
	$(FC) $(FFLAGS) $(NOFREE) -c lapack_min.f
lapack_double.o: lapack_double.f
	$(FC) $(FFLAGS) $(NOFREE) -c lapack_double.f
lapack_single.o: lapack_single.f
	$(FC) $(FFLAGS) $(NOFREE) -c lapack_single.f
lapack_atlas.o: lapack_atlas.f
	$(FC) $(FFLAGS) $(NOFREE) -c lapack_atlas.f
linpack_double.o: linpack_double.f
	$(FC) $(FFLAGS) $(NOFREE) -c linpack_double.f
linpack_single.o: linpack_single.f
	$(FC) $(FFLAGS) $(NOFREE) -c linpack_single.f


.c.o:
	$(CC) $(CFLAGS) -c $*.c
.F.o:
	$(CPP)
	$(FC) $(FFLAGS) $(FREE) $(INCS) -c $*.f
.F.f:
	$(CPP)
.f.o:
	$(FC) $(FFLAGS) $(FREE) $(INCS) -c $*.f
```

4，编译vasp

```
cd  ../vasp.5.2/
cp makefile.linux_ifc_P4 makefile
修改makefile
make
```

修改后的makefile：

```
.SUFFIXES: .inc .f .f90 .F
#-----------------------------------------------------------------------
# Makefile for Intel Fortran compiler for Pentium/Athlon/Opteron
# bases systems
# we recommend this makefile for both Intel as well as AMD systems
# for AMD based systems appropriate BLAS and fftw libraries are
# however mandatory (whereas they are optional for Intel platforms)
#
# The makefile was tested only under Linux on Intel and AMD platforms
# the following compiler versions have been tested:
#  - ifc.7.1  works stable somewhat slow but reliably
#  - ifc.8.1  fails to compile the code properly
#  - ifc.9.1  recommended (both for 32 and 64 bit)
#  - ifc.10.1 partially recommended (both for 32 and 64 bit)
#             tested build 20080312 Package ID: l_fc_p_10.1.015
#             the gamma only mpi version can not be compiles
#             using ifc.10.1
#
# it might be required to change some of library pathes, since
# LINUX installation vary a lot
# Hence check ***ALL*** options in this makefile very carefully
#-----------------------------------------------------------------------
#
# BLAS must be installed on the machine
# there are several options:
# 1) very slow but works:
#   retrieve the lapackage from ftp.netlib.org
#   and compile the blas routines (BLAS/SRC directory)
#   please use g77 or f77 for the compilation. When I tried to
#   use pgf77 or pgf90 for BLAS, VASP hang up when calling
#   ZHEEV  (however this was with lapack 1.1 now I use lapack 2.0)
# 2) more desirable: get an optimized BLAS
```

随笔档案

2014年11月 (10)

最新评论

1. Re:搭建高性能计算环境（四）、应用软件的安装之VASP

非常感谢楼主的详细安装信息，按照楼主的方法成功的安装了MS，但是在编译VSAP时报错。报错信息如下：./preprocess base.f90 -DHOST=\"LinuxIFC\" -DCACHE.....

--安风琴

2. Re:搭建高性能计算环境（四）、应用软件的安装之VASP

非常感谢楼主，我找了好久，没想到楼主这里写的这么详细！一定要顶上去！！

--huoxing487

3. Re:搭建高性能计算环境（一）、Linux操作系统的安装和配置

博主，你真辛苦，这里的10篇笔记对我们做计算的人来说，是极大的帮助。
只是第一，二篇的图片不能显示。

--coffeetan

阅读排行榜

1. 搭建高性能计算环境（四）、应用软件的安装之VASP(989)
2. 搭建高性能计算环境（三）、安装intel编译器和mpi(909)
3. 搭建高性能计算环境（九）、应用软件的安装之gaussian 09(596)
4. 搭建高性能计算环境（八）、应用软件的安装之gromacs(565)
5. 搭建高性能计算环境（五）、应用软件的安装之Amber12(528)

评论排行榜

1. 搭建高性能计算环境（四）、应用软件的安装之VASP(2)
2. 搭建高性能计算环境（一）、Linux操作系统的安装和配置(1)

推荐排行榜

1. 搭建高性能计算环境（四）、应用软件的安装之VASP(1)
2. 搭建高性能计算环境（十）、应用软件的安装之Wien2k(1)
3. 搭建高性能计算环境（九）、应用软件的安装之gaussian 09(1)

```
#
# the two most reliable packages around are presently:
# 2a) Intels own optimised BLAS (PIII, P4, PD, PC2, Itanium)
#     http://developer.intel.com/software/products/mkl/
#   this is really excellent, if you use Intel CPU's
#
# 2b) probably fastest SSE2 (4 GFlops on P4, 2.53 GHz, 16 GFlops PD,
#     around 30 GFlops on Quad core)
#   Kazushige Goto's BLAS
#   http://www.cs.utexas.edu/users/kgoto/signup_first.html
#   http://www.tacc.utexas.edu/resources/software/
#
#-----------------------------------------------------------------------

# all CPP processed fortran files have the extension .f90
SUFFIX=.f90

#-----------------------------------------------------------------------
# fortran compiler and linker
#-----------------------------------------------------------------------
FC=ifc
# fortran linker
FCL=$(FC)


#-----------------------------------------------------------------------
# whereis CPP ?? (I need CPP, can't use gcc with proper options)
# that's the location of gcc for SUSE 5.3
#
#  CPP_    =  /usr/lib/gcc-lib/i486-linux/2.7.2/cpp -P -C
#
# that's probably the right line for some Red Hat distribution:
#
#  CPP_    =  /usr/lib/gcc-lib/i386-redhat-linux/2.7.2.3/cpp -P -C
#
#  SUSE X.X, maybe some Red Hat distributions:

CPP_ =  ./preprocess <$*.F | /usr/bin/cpp -P -C -traditional >$*$(SUFFIX)

#-----------------------------------------------------------------------
# possible options for CPP:
# NGXhalf             charge density   reduced in X direction
# wNGXhalf            gamma point only reduced in X direction
# avoidalloc          avoid ALLOCATE if possible
# PGF90               work around some for some PGF90 / IFC bugs
# CACHE_SIZE          1000 for PII,PIII, 5000 for Athlon, 8000-12000 P4, PD
# RPROMU_DGEMV        use DGEMV instead of DGEMM in RPRO (depends on used BLAS)
# RACCMU_DGEMV        use DGEMV instead of DGEMM in RACC (depends on used BLAS)
# tbdyn                 MD package of Tomas  Bucko
#-----------------------------------------------------------------------

CPP     = $(CPP_)  -DHOST=\"LinuxIFC\" \
          -DCACHE_SIZE=12000 -DPGF90 -Davoidalloc -DNGXhalf \
#          -DRPROMU_DGEMV  -DRACCMU_DGEMV

#-----------------------------------------------------------------------
# general fortran flags  (there must a trailing blank on this line)
# byterecl is strictly required for ifc, since otherwise
# the WAVECAR file becomes huge
#-----------------------------------------------------------------------

FFLAGS =  -O3 -ip -no-prec-div -funroll-loops -xHost -heap-arrays 64

#-----------------------------------------------------------------------
# optimization
# we have tested whether higher optimisation improves performance
```

```
# -axK  SSE1 optimization,  but also generate code executable on all mach.
#       xK improves performance somewhat on XP, and a is required in order
#       to run the code on older Athlons as well
# -xW   SSE2 optimization
# -axW  SSE2 optimization,  but also generate code executable on all mach.
# -tpp6 P3 optimization
# -tpp7 P4 optimization
#-----------------------------------------------------------------------

# ifc.9.1, ifc.10.1 recommended
OFLAG=

OFLAG_HIGH = $(OFLAG)
OBJ_HIGH =
OBJ_NOOPT =
DEBUG  = -FR -O0
INLINE = $(OFLAG)

#-----------------------------------------------------------------------
# the following lines specify the position of BLAS  and LAPACK
# VASP works fastest with the libgoto library
# so that's what we recommend
#-----------------------------------------------------------------------

# mkl.10.0
# set -DRPROMU_DGEMV  -DRACCMU_DGEMV in the CPP lines
BLAS=-L/opt/intel/mkl/lib/intel64 -lmkl_core -lmkl_sequential -lmkl_intel_lp64

# even faster for VASP Kazushige Goto's BLAS
# http://www.cs.utexas.edu/users/kgoto/signup_first.html
# parallel goto version requires sometimes -libverbs
#BLAS=  /opt/libs/libgoto/libgoto.so

# LAPACK, simplest use vasp.5.lib/lapack_double
LAPACK= ../vasp.5.lib/lapack_double.o

# use the mkl Intel lapack
#LAPACK= -lmkl_lapack

#-----------------------------------------------------------------------

LIB  = -L../vasp.5.lib -ldmy \
     ../vasp.5.lib/linpack_double.o $(LAPACK) \
     $(BLAS)

# options for linking, nothing is required (usually)
LINK    =

#-----------------------------------------------------------------------
# fft libraries:
# VASP.5.2 can use fftw.3.1.X (http://www.fftw.org)
# since this version is faster on P4 machines, we recommend to use it
#-----------------------------------------------------------------------

FFT3D   = fft3dfurth.o fft3dlib.o

# alternatively: fftw.3.1.X is slighly faster and should be used if available
#FFT3D   = fftw3d.o fft3dlib.o   /opt/libs/fftw-3.1.2/lib/libfftw3.a


#=======================================================================
# MPI section, uncomment the following lines until
#    general  rules and compile lines
# presently we recommend OPENMPI, since it seems to offer better
# performance than lam or mpich
#
```

```
# !!! Please do not send me any queries on how to install MPI, I will
# certainly not answer them !!!!
#=======================================================================
#-----------------------------------------------------------------------
# fortran linker for mpi
#-----------------------------------------------------------------------

FC=mpiifort
FCL=$(FC)

#-----------------------------------------------------------------------
# additional options for CPP in parallel version (see also above):
# NGZhalf               charge density   reduced in Z direction
# wNGZhalf              gamma point only reduced in Z direction
# scaLAPACK             use scaLAPACK (usually slower on 100 Mbit Net)
# avoidalloc          avoid ALLOCATE if possible
# PGF90               work around some for some PGF90 / IFC bugs
# CACHE_SIZE          1000 for PII,PIII, 5000 for Athlon, 8000-12000 P4, PD
# RPROMU_DGEMV        use DGEMV instead of DGEMM in RPRO (depends on used BLAS)
# RACCMU_DGEMV        use DGEMV instead of DGEMM in RACC (depends on used BLAS)
# tbdyn                 MD package of Tomas  Bucko
#-----------------------------------------------------------------------

#-----------------------------------------------------------------------

CPP    = $(CPP_) -DMPI  -DHOST=\"LinuxIFC\" -DIFC \
     -DCACHE_SIZE=12000 -DPGF90 -Davoidalloc -DNGZhalf \
     -DMPI_BLOCK=8000
#    -DRPROMU_DGEMV  -DRACCMU_DGEMV

#-----------------------------------------------------------------------
# location of SCALAPACK
# if you do not use SCALAPACK simply leave that section commented out
#-----------------------------------------------------------------------

#BLACS=$(HOME)/archives/SCALAPACK/BLACS/
#SCA_=$(HOME)/archives/SCALAPACK/SCALAPACK

#SCA= $(SCA_)/libscalapack.a  \
# $(BLACS)/LIB/blacsF77init_MPI-LINUX-0.a $(BLACS)/LIB/blacs_MPI-LINUX-0.a $(BLACS)/LIB/bla
csF77init_MPI-LINUX-0.a

SCA=

#-----------------------------------------------------------------------
# libraries for mpi
#-----------------------------------------------------------------------

LIB     = -L../vasp.5.lib -ldmy  \
      ../vasp.5.lib/linpack_double.o $(LAPACK) \
      $(SCA) $(BLAS)

# FFT: fftmpi.o with fft3dlib of Juergen Furthmueller
FFT3D   = fftmpi.o fftmpi_map.o fft3dfurth.o fft3dlib.o

# alternatively: fftw.3.1.X is slighly faster and should be used if available
#FFT3D   = fftmpiw.o fftmpi_map.o fftw3d.o fft3dlib.o  /opt/libs/fftw-3.1.2/lib/libfftw3.a

#-----------------------------------------------------------------------
# general rules and compile lines
#-----------------------------------------------------------------------
BASIC=   symmetry.o symlib.o   lattlib.o  random.o


SOURCE=  base.o     mpi.o      smart_allocate.o      xml.o  \
         constant.o jacobi.o   main_mpi.o  scala.o   \
```

```
         asa.o      lattice.o  poscar.o   ini.o  mgrid.o  xclib.o  vdw_nl.o  xclib_grad.o \
         radial.o   pseudo.o   gridq.o     ebs.o  \
         mkpoints.o wave.o     wave_mpi.o  wave_high.o  \
         $(BASIC)   nonl.o     nonlr.o    nonl_high.o dfast.o    choleski2.o \
         mix.o      hamil.o    xcgrad.o   xcspin.o    potex1.o   potex2.o  \
         constrmag.o cl_shift.o relativistic.o LDApU.o \
         paw_base.o metagga.o  egrad.o    pawsym.o   pawfock.o  pawlhf.o   rhfatm.o  paw.o
  \
         mkpoints_full.o       charge.o   Lebedev-Laikov.o  stockholder.o dipol.o    pot.o
\
         dos.o      elf.o      tet.o      tetweight.o hamil_rot.o \
         steep.o    chain.o    dyna.o     sphpro.o    us.o  core_rel.o \
         aedens.o   wavpre.o   wavpre_noio.o broyden.o \
         dynbr.o    rmm-diis.o reader.o   writer.o   tutor.o xml_writer.o \
         brent.o    stufak.o   fileio.o   opergrid.o stepver.o  \
         chgloc.o   fast_aug.o fock.o     mkpoints_change.o sym_grad.o \
         mymath.o   internals.o dynconstr.o dimer_heyden.o dvvtrajectory.o vdwforcefield.o
\
         hamil_high.o nmr.o    pead.o     mlwf.o     subrot.o   subrot_scf.o \
         force.o    pwlhf.o  gw_model.o optreal.o   davidson.o  david_inner.o \
         electron.o rot.o  electron_all.o shm.o    pardens.o  paircorrection.o \
         optics.o   constr_cell_relax.o   stm.o    finite_diff.o elpol.o    \
         hamil_lr.o rmm-diis_lr.o  subrot_cluster.o subrot_lr.o \
         lr_helper.o hamil_lrf.o   elinear_response.o ilinear_response.o \
         linear_optics.o linear_response.o   \
         setlocalpp.o  wannier.o electron_OEP.o electron_lhf.o twoelectron4o.o \
         ratpol.o screened_2e.o wave_cacher.o chi_base.o wpot.o local_field.o \
         ump2.o bse_te.o bse.o acfdt.o chi.o sydmat.o dmft.o \
         rmm-diis_mlr.o  linear_response_NMR.o

vasp: $(SOURCE) $(FFT3D) $(INC) main.o
        rm -f vasp
        $(FCL) -o vasp main.o  $(SOURCE)   $(FFT3D) $(LIB) $(LINK)
makeparam: $(SOURCE) $(FFT3D) makeparam.o main.F $(INC)
        $(FCL) -o makeparam  $(LINK) makeparam.o $(SOURCE) $(FFT3D) $(LIB)
zgemmtest: zgemmtest.o base.o random.o $(INC)
        $(FCL) -o zgemmtest $(LINK) zgemmtest.o random.o base.o $(LIB)
dgemmtest: dgemmtest.o base.o random.o $(INC)
        $(FCL) -o dgemmtest $(LINK) dgemmtest.o random.o base.o $(LIB)
ffttest: base.o smart_allocate.o mpi.o mgrid.o random.o ffttest.o $(FFT3D) $(INC)
        $(FCL) -o ffttest $(LINK) ffttest.o mpi.o mgrid.o random.o smart_allocate.o base.o
$(FFT3D) $(LIB)
kpoints: $(SOURCE) $(FFT3D) makekpoints.o main.F $(INC)
        $(FCL) -o kpoints $(LINK) makekpoints.o $(SOURCE) $(FFT3D) $(LIB)

clean:
        -rm -f *.g *.f *.o *.L *.mod ; touch *.F

main.o: main$(SUFFIX)
        $(FC) $(FFLAGS)$(DEBUG)  $(INCS) -c main$(SUFFIX)
xcgrad.o: xcgrad$(SUFFIX)
        $(FC) $(FFLAGS) $(INLINE)  $(INCS) -c xcgrad$(SUFFIX)
xcspin.o: xcspin$(SUFFIX)
        $(FC) $(FFLAGS) $(INLINE)  $(INCS) -c xcspin$(SUFFIX)

makeparam.o: makeparam$(SUFFIX)
        $(FC) $(FFLAGS)$(DEBUG)  $(INCS) -c makeparam$(SUFFIX)

makeparam$(SUFFIX): makeparam.F main.F
#
# MIND: I do not have a full dependency list for the include
# and MODULES: here are only the minimal basic dependencies
# if one strucuture is changed then touch_dep must be called
# with the corresponding name of the structure
#
base.o: base.inc base.F
```

```
mgrid.o: mgrid.inc mgrid.F
constant.o: constant.inc constant.F
lattice.o: lattice.inc lattice.F
setex.o: setexm.inc setex.F
pseudo.o: pseudo.inc pseudo.F
poscar.o: poscar.inc poscar.F
mkpoints.o: mkpoints.inc mkpoints.F
wave.o: wave.F
nonl.o: nonl.inc nonl.F
nonlr.o: nonlr.inc nonlr.F

$(OBJ_HIGH):
        $(CPP)
        $(FC) $(FFLAGS) $(OFLAG_HIGH) $(INCS) -c $*$(SUFFIX)
$(OBJ_NOOPT):
        $(CPP)
        $(FC) $(FFLAGS) $(INCS) -c $*$(SUFFIX)

fft3dlib_f77.o: fft3dlib_f77.F
        $(CPP)
        $(F77) $(FFLAGS_F77) -c $*$(SUFFIX)

.F.o:
        $(CPP)
        $(FC) $(FFLAGS) $(OFLAG) $(INCS) -c $*$(SUFFIX)
.F$(SUFFIX):
        $(CPP)
$(SUFFIX).o:
        $(FC) $(FFLAGS) $(OFLAG) $(INCS) -c $*$(SUFFIX)

# special rules
#-----------------------------------------------------------------------
# these special rules are cummulative (that is once failed
#   in one compiler version, stays in the list forever)
# -tpp5|6|7 P, PII-PIII, PIV
# -xW use SIMD (does not pay of on PII, since fft3d uses double prec)
# all other options do no affect the code performance since -O1 is used

fft3dlib.o : fft3dlib.F
        $(CPP)
        $(FC) -FR -lowercase -O2 -c $*$(SUFFIX)

fft3dfurth.o : fft3dfurth.F
        $(CPP)
        $(FC) -FR -lowercase -O1 -c $*$(SUFFIX)

fftw3d.o : fftw3d.F
        $(CPP)
        $(FC) -FR -lowercase -O1 -c $*$(SUFFIX)

wave_high.o : wave_high.F
        $(CPP)
        $(FC) -FR -lowercase -O1 -c $*$(SUFFIX)

radial.o : radial.F
        $(CPP)
        $(FC) -FR -lowercase -O1 -c $*$(SUFFIX)

symlib.o : symlib.F
        $(CPP)
        $(FC) -FR -lowercase -O1 -c $*$(SUFFIX)

symmetry.o : symmetry.F
        $(CPP)
        $(FC) -FR -lowercase -O1 -c $*$(SUFFIX)
```

```
wave_mpi.o : wave_mpi.F
        $(CPP)
        $(FC) -FR -lowercase -O1 -c $*$(SUFFIX)

wave.o : wave.F
        $(CPP)
        $(FC) -FR -lowercase -O1 -c $*$(SUFFIX)

dynbr.o : dynbr.F
        $(CPP)
        $(FC) -FR -lowercase -O1 -c $*$(SUFFIX)

asa.o : asa.F
        $(CPP)
        $(FC) -FR -lowercase -O1 -c $*$(SUFFIX)

broyden.o : broyden.F
        $(CPP)
        $(FC) -FR -lowercase -O2 -c $*$(SUFFIX)

us.o : us.F
        $(CPP)
        $(FC) -FR -lowercase -O1 -c $*$(SUFFIX)

LDApU.o : LDApU.F
        $(CPP)
        $(FC) -FR -lowercase -O2 -c $*$(SUFFIX)
```

5，添加环境变量

在/etc/profile文件末尾添加如下内容，需要重新登录后生效

```
export PATH=/opt/vasp/vasp5.2:$PATH
```

6，测试

准备测试文件vasp.Hg，进入算例目录，输入：

```
mpirun -np 4 vasp
```

如能正常计算完毕，说明软件安装成功。

补充软件下载地址(仅供测试使用)：

链接: http://pan.baidu.com/s/1qWv4k7i 密码: rsgh

分类: HPC

标签: HPC, vasp

iDove
关注 - 1
粉丝 - 5
+加关注

1 0

« 上一篇：搭建高性能计算环境（三）、安装intel编译器和mpi
» 下一篇：搭建高性能计算环境（五）、应用软件的安装之Amber12

posted @ 2014-11-17 18:20 iDove 阅读(990) 评论(2) 编辑 收藏

评论列表

#1楼 2016-08-29 12:09 huoxing487

非常感谢楼主，我找了好久，没想到楼主这里写的这么详细！一定要顶上去！！

支持(1) 反对(0)

#2楼 2016-08-31 14:49 安风琴

非常感谢楼主的详细安装信息，按照楼主的方法成功的安装了MS，但是在编译VSAP时报错。报错信息如下:
./preprocess <base.F | /usr/bin/cpp -P -C -traditional >base.f90 -DHOST=\"LinuxIFC\" -DCACHE_SIZE=12000 -DPGF90 -Davoidalloc -DNGXhalf
ifort -FR -lowercase -assume byterecl -O2 -ip -ftz -c base.f90
ifort: command line remark #10148: option '-lowercase' not supported
base.f90(1): error #5082: Syntax error, found '/' when expecting one of: <LABEL> <END-OF-STATEMENT> ; TYPE INTEGER REAL COMPLEX BYTE CHARACTER CLASS DOUBLE ...
----------------------------------^
base.f90(20): error #5145: Invalid blank/tab
include it implicitly at the start of every compilation. It must
--^
base.f90(29): error #5143: Missing mandatory separating blank

---------------^
base.f90(32): error #5145: Invalid blank/tab
/* We do not support C11 <threads.h>. */
-------------------------------------^
base.f90(1): catastrophic error: Could not recover from previous syntax error
compilation aborted for base.f90 (code 1)
make: *** [base.o] 错误 1
还希望能得到楼主的帮助，多谢了!

支持(0) 反对(0)

刷新评论 刷新页面 返回顶部

注册用户登录后才能发表评论，请 登录 或 注册，访问网站首页。