# [基础准备：需要预先准备好下述文件]

1. BCC\_CSM源码：BCCCSM1.1m/
2. BCC\_CSM初始场：INIDATA\_1.1m/
3. 控制实验脚本：piControl\_t106/
4. 历史实验脚本：T106historical/

上述文件在服务器上的位置

|  |  |
| --- | --- |
| 166.111.68.173 | /share2/BCC/BCC\_CSM1.1m |

# [BCC\_CSM编译安装]

以下所有路径和操作流程均为166.111.68.173机器，以控制实验为例

1. 进入piControl\_t106，脚本调用关系如下图。1ctl106csm.run是一键编译脚本，其调用的6个csh脚本会分别产生一个可执行文件。submit.run是提交作业脚本。

cpl5.1.setup.csh

mom.setup.csh

bccam2.0.2.setup.csh

bccavim1.setup.csh

sis.setup.csh

esmf.setup.csh

1ctl106csm.run

submit.run

piControl\_t106

1. 配置信息。在运行1ctl106csm.run前，需要对机器信息和实验信息进行单独配置。
2. 实验信息配置。

在1ctl106csm.run中

代码路径：setenv MODEL\_PATH /home/dingnan/BCC\_CSM1.1m/BCCCSM1.1m

初始场路径：setenv DATA\_PATH /share2/BCC/BCC\_CSM1.1m/INIDATA\_1.1m/

建立算例的脚本路径：setenv WORK\_PATH /home/dingnan/BCC\_CSM1.1m/piControl\_t106

实验运行时间（天或月）：setenv System\_Time\_Type day # month/day

实验运行时间（结合上面，表示运行几天或几个月）：setenv Integration\_Time 3

b）机器信息配置。

(i) 在1ctl106csm.run，132行设置操作系统类别，本例OS 设置为Linux

(ii) 在BCCCSM1.1m/models/bld 下面，修改Macros.Linux。修改内容为mpi， netcdf的路径。

1. 实验编译( **Intel mpi 5.0 + Intel compiler 15.0** )。

原则上编译只需要：./1ctl106csm.run

本例编译中会报错，需要些许修正，如下：

1. 在编译atm, lnd和ice时会出现找不到timemanger.mod的情况。

解决方法：将esmf编译好的\*.mod, 全部拷贝到对应的 OBJ/atm, OBJ/lnd , OBJ/ice的路径下。

1. Ice编译报错：ld: cannot find –limf

在BCCCSM1.1m/models/ice/sis/bin路径下，修改mkmf.template.sunway文件中的mpi，编译器和 libimf.so的路径。

1. Ice编译报错：undefined reference to `for\_cpystr' 等很多undef

* 连接过程由ld \*.o –o ice 改成 mpiifort \*.o -o ice
* 然后，新的错误为：

---------------------------------------------------------------------------------------------------------------------------------------

/usr/lib/../lib64/crt1.o: In function `\_start':

(.text+0x12): undefined reference to `\_\_libc\_csu\_fini'

/usr/lib/../lib64/crt1.o: In function `\_start':

(.text+0x19): undefined reference to `\_\_libc\_csu\_init'

/lib64/libpthread.so.0: undefined reference to `errno@GLIBC\_PRIVATE'

/lib64/libpthread.so.0: undefined reference to `h\_errno@GLIBC\_PRIVATE'

/lib64/libpthread.so.0: undefined reference to `\_\_libc\_dl\_error\_tsd@GLIBC\_PRIVATE'

/usr/lib/../lib64/librt.so: undefined reference to `\_\_vdso\_clock\_gettime@GLIBC\_PRIVATE'

/lib64/libc.so: undefined reference to `\_dl\_out\_of\_memory@GLIBC\_PRIVATE'

/usr/lib/../lib64/librt.so: undefined reference to `\_\_fortify\_fail@GLIBC\_PRIVATE'

/home/software/intel/parallel\_studio\_xe\_2015\_update2/impi/5.0.3.048/intel64/lib/libmpi.so: undefined reference to `vmsplice@GLIBC\_2.5'

/lib64/libpthread.so.0: undefined reference to `\_\_resp@GLIBC\_PRIVATE'

/usr/lib/../lib64/libm.so: undefined reference to `\_\_get\_cpu\_features@GLIBC\_PRIVATE'

首先确定该路径的是否正确，发现实际应该连接的是/usr/lib64/crt1.o，最终的连接命令为：

Mpiifort \*.o -o ice -L/home/software/netcdf-4.0.1.intel11/lib -lnetcdf

-L/home/software/intel/parallel\_studio\_xe\_2015\_update2/impi/5.0.3.048/intel64/lib -lmpi

-L/home/software/intel/parallel\_studio\_xe\_2015\_update2/composer\_xe\_2015.2.164/compiler/lib/intel64/

-limf -L/usr/lib64

**现在未找到默认的ld 链接是在哪里定义的，所以编译过程需要分步手动执行。**

1. ocn编译报错同ice
2. cpl编译报错同ice

# [BCC\_CSM运行]

1. 所有的可执行文件在/home/dingnan/BCC\_CSM1.1m/piControl\_t106/output.pit106/ 对应的文件夹下。

所有的编译的.o文件在/home/dingnan/BCC\_CSM1.1m/piControl\_t106/OBJ

1. poe\_sw.cmdfile（/home/dingnan/BCC\_CSM1.1m/piControl\_t106/output.pit106/目录下）文件中写明了每个可执行文件用多少核，及可执行文件的路径。

---------------------------------------------------------------------------------------------------------------------------------

0-39 /home/dingnan/BCC\_CSM1.1m/piControl\_t106/output.pit106/atm/atm

40-43 /home/dingnan/BCC\_CSM1.1m/piControl\_t106/output.pit106/lnd/lnd

44-51 /home/dingnan/BCC\_CSM1.1m/piControl\_t106/output.pit106/ice/ice

52-83 /home/dingnan/BCC\_CSM1.1m/piControl\_t106/output.pit106/ocn/ocn

84-84 /home/dingnan/BCC\_CSM1.1m/piControl\_t106/output.pit106/cpl/cpl

---------------------------------------------------------------------------------------------------------------------------------

表示 atm 使用40核，lnd使用4核，ice使用8核，ocn使用32核，cpl使用1核

1. 运行命令：

mpiexec -machinefile host -env atm\_dir /home/dingnan/BCC\_CSM1.1m/piControl\_t106/ output.pit106/atm -env atm\_in /home/dingnan/BCC\_CSM1.1m/piControl\_t106/output.pi t106/atm/atm.stdin -env atm\_out /home/dingnan/BCC\_CSM1.1m/piControl\_t106/output. pit106/atm/atm.log -n 40 ./atm/atm : -env lnd\_dir /home/dingnan/BCC\_CSM1.1m/piCo ntrol\_t106/output.pit106/lnd -env lnd\_in /home/dingnan/BCC\_CSM1.1m/piControl\_t10 6/output.pit106/lnd/lnd.stdin -env lnd\_out /home/dingnan/BCC\_CSM1.1m/piControl\_t 106/output.pit106/lnd/lnd.log -n 4 ./lnd/lnd : -env cpl\_dir /home/dingnan/BCC\_CS M1.1m/piControl\_t106/output.pit106/cpl -env cpl\_in /home/dingnan/BCC\_CSM1.1m/piC ontrol\_t106/output.pit106/cpl/cpl.stdin -env cpl\_out /home/dingnan/BCC\_CSM1.1m/p iControl\_t106/output.pit106/cpl/cpl.log -n 1 ./cpl/cpl : -env ice\_dir /home/ding nan/BCC\_CSM1.1m/piControl\_t106/output.pit106/ice -env ice\_in /home/dingnan/BCC\_C SM1.1m/piControl\_t106/output.pit106/ice/ice.stdin -env ice\_out /home/dingnan/BCC \_CSM1.1m/piControl\_t106/output.pit106/ice/ice.log -n 8 ./ice/ice : -env ocn\_dir /home/dingnan/BCC\_CSM1.1m/piControl\_t106/output.pit106/ocn -env ocn\_in /home/din gnan/BCC\_CSM1.1m/piControl\_t106/output.pit106/ocn/ocn.stdin -env ocn\_out /home/d ingnan/BCC\_CSM1.1m/piControl\_t106/output.pit106/ocn/ocn.log -n 32 ./ocn/ocn

# [BCC\_CSM运行时错误]

1. **control run 脚本错误**

|  |  |
| --- | --- |
| **编译器版本** | **Intel 15.0** |
| **Mpi版本** | **Intel 5.0** |
| **Netcdf** | **4.0** |
| **错误现场** | **(shr\_msg\_iFatal error in PMPI\_Comm\_rank: Invalid communicator, error stack:**  **PMPI\_Comm\_rank(122): MPI\_Comm\_rank(comm=0x5b, rank=0x7fffffffacbc) failed**  **PMPI\_Comm\_rank(75).: Invalid communicator**  **Fatal error in PMPI\_Comm\_rank: Invalid communicator, error stack:**  **PMPI\_Comm\_rank(122): MPI\_Comm\_rank(comm=0x5b, rank=0x7fffffffacbc) failed**  **PMPI\_Comm\_rank(75).: Invalid communicator**  **Fatal error in PMPI\_Comm\_rank: Invalid communicator, error stack:**  **PMPI\_Comm\_rank(122): MPI\_Comm\_rank(comm=0x5b, rank=0x7fffffffacbc) failed** |
| **错误保留** | **/home/dingnan/BCC\_CSM1.1m/piControl\_t106/output.pit106/run.log** |
| **出错原因** | **脚本设置错误。** |
| **解决办法** | **1.**  **原始代码：**    **修改为：**    **结果：**  **编译通过，运行时报错：**    **2.**  **原始代码修改为：（mpirecv 和mpisend是模式中对mpi\_recv和mpi\_send的封装）**  **call mpirecv (procname(i), mpi\_max\_processor\_name, mpichar, i, 1, mpicom)**  **call mpisend (procname(iam), mpi\_max\_processor\_name, mpichar, 0, 1, mpicom)**  **结果：**  **编译lnd报错:undef:mpirecv**  **under:mpisend**  **3.**  **原始代码修改为：**  **call shr\_msg\_recv\_c (procname(i),mpi\_max\_processor\_name, SHR\_MSG\_TID\_LND, SHR\_MSG\_TAG\_C2L)**  **call shr\_msg\_send\_c (procname(i), mpi\_max\_processor\_name, SHR\_MSG\_TID\_LND, SHR\_MSG\_TAG\_L2C)**  **结果：**  **编译通过，运行时卡死。shr\_msg\_recv\_c shr\_msg\_send\_c是在csm\_share中自己定义一套通信函数。**  **4.**  **将intel mpi5.0 换成mvapich2.2**  **报错同1**  **5.**  **在4的基础上，把allgather 部分的recv buffer 写成MPI\_IN\_PLACE**  **6.**  **在4的基础上，使用send 和recv**    **7. 通过加入print定位程序中断位置，追查对改mpi\_recv的buffer定义，发现当前配置是CAM2+nocarbon，但正确的配置应该是CAM3carbon，返回修改编译脚本。** |
| **问题是否解决** | **问题解决。** |

1. **control run MPI\_gather 发送和接受buffer不能相同。**

|  |  |
| --- | --- |
| **编译器版本** | **Intel 15.0** |
| **Mpi版本** | **Intel 5.0** |
| **Netcdf** | **4.0** |
| **错误定位** | **models/lnd/bccavim1.0.1/src/main/spmdMod.F90， 类似的还有ice，ocn。** |
| **出错原因** | **Mpi\_allgather不能自己收自己发的，即recv buffer 和 send buffer不能是同一个。** |
| **解决办法** | **将gather改成send recv：**  **原来为：**    **修改为：** |
| **问题是否解决** | **问题解决。** |

1. History run中断

|  |  |
| --- | --- |
| **编译器版本** | **Intel 15.0** |
| **Mpi版本** | **Intel 5.0** |
| **Netcdf** | **4.0** |
| **错误现象** | **先是出现NaN，联系辛老师和房老师更换成了新的atm，ice和mom的初始场，虽然没有了NaN，但无报错中断。** |
| **出错原因** | **模式不稳定，原则上模式的restart文件是可以从其他地方control run的输出的，但是有时候出现模式的漂移现场。** |
| **解决办法** | **将history run的脚本里的模式的状态和时间，改成随实验变化的。如下：**  **在模式的总编译控制脚本中SETBLD从auto改为true，并将BASETIME设置成1850-01-01** |
| **问题是否解决** | **问题解决。** |

# [性能工具使用说明]

1. 基础准备：
2. MPIprofiling：可采集kernel或component级别的mpi通信操作，包括每个mpi操作的总时间，总调用次数，总通信量。
3. 优化方案搜索代码

上述文件在服务器上的位置

|  |  |  |
| --- | --- | --- |
| 166.111.68.173 | MPIprofiling | /home/dingnan/soft/profile-dn-hour-kernel-old |
|  | mpiP | /home/dingnan/soft/mpip-fixedfortran/ |
|  | 优化方案搜索 | /home/dingnan/mydata/search |

1. MPIprofiling使用
2. 适当修改Makefile(MPI库函数位置）

make

1. 编译MPI程序时链接libmpit.a库，并在相应代码位置进行插装。

call PROFILE\_XXX\_INIT()

call PROFILE\_XXX\_FINISH()

1. 编译运行应用程序后，在提交作业的目录会有mpi\_record.0,mpi\_record.1,…生成，后缀的0,1表示进程号, 每个文件都对应记录该进程的MPI time。
2. 统计数据profile结果并建模

./run-profile.sh

run-profile.sh中需要修改2处：

A. 文件第4行

parallel=(12 96)，是采样点的并行度。

B. 文件第17行

sh getdata.sh ${parallel[i]} /home/dingnan/mydata/tianhe-1A/dingnan/profile\_kernel\_B1850f09g16\_${parallel[i]} Bf09g16

其中：/home/dingnan/mydata/tianhe-1A/dingnan/profile\_kernel\_B1850f09g16\_${parallel[i]} 是mpi\_record.\* 的目录，用户依据自己目录的情况，修改该路径。

Bf0916是在当前脚本目录建立的一个文件夹，为进行本次统计数据后，数据的输出目录。用户可以自行更改该文件夹的名字。

1. run-profile运行后输出的文件：

Model\_Error.csv： kernel和component的绝对误差和相对误差。

model\_parameters.csv：模型的参数值。

12-144-20000.csv：是搜索算法的输入，表示该文件是用12和144核作为采样点建模，输出从1到20000核每个component的模型预测运行时间。

|  |  |  |
| --- | --- | --- |
| 166.111.68.173 | 插装的例子 | /home/dingnan/mydata/profileTool/SourceMods.kernel |
|  | 编译命令的例子 | /home/dingnan/mydata/profileTool/Macros.profile.kernel和Makefile.profile.kernel |
|  | 自动化建模 | /home/dingnan/mydata/modeling/run-profile.sh  run-profile.sh中修改并行度，和mpi\_record.进程号的路径。 |
|  | 针对BCC的例子 | 在/home/dingnan/BCC\_CSM1.1m/BCCCSM1.1m/models 下  grep –rn “PROFILE”./\* 可以看到插装位置，编译命令的修改 |

1. 优化方案搜索

clang++ -Wall -Wextra --std=c++14 final\_rec.cc

./a.out <96.txt

最后给出每个component的进程数，和component之间的布局

1. 可能遇到的问题：
2. 当使用intel编译器impi 5.0 一下的版本时，编译MPIprofiling会报错，原因为impi 5.0比较以前的版本，将mpi.h中一些sendbuf和recvbuf 定义为了const。如果报错，则将MPIprofiling中的mpiwapper.c里的定义和mpi.h中的定义改成一样的即可。对mpiP也是同样的道理。
3. 优化方案如果编译报错，一般是编译器的版本太低，请先查看编译器版本，最好是gcc 5.1或以上。搜索代码可在本机运行，如果是mac系统，可直接使用其中的clang++进行编译。