HP Superdome 服务器用户使用指南

中国科学技术大学 超级运算中心 李会民

2008 年 5 月

目录

1 HP Superdome 服务器概述 **4**

2 用户登录与文件传输 **4**

3 串行及 OpenMP 程序编译 **6**

3.1 C/C++ 程序的编译 6

3.1.1 HP aC++/HP C 编译器简介 6

3.1.2 输入输出文件后缀与类型的关系 7

3.1.3 aCC 编译举例 8

3.1.4 重要编译选项 9

3.1.5 C/C++ 程序编译举例 12

3.1.6 相关资料 12

3.2 Fortran 程序的编译 12

3.2.1 HP Fortran 编译器简介 12

3.2.2 输入输出文件后缀与类型的关系 13

3.2.3 重要编译选项 13

3.2.4 Fortran 程序编译举例 18

3.2.5 相关资料 18

3.3 OpenMP 程序的编译与运行 18

4 MPI 并行环境：HP MPI **20**

4.1 MPI 并行程序的编译 20

4.2 MPI 并行程序的运行 20

4.3 MPI 并行程序调试 21

4.4 相关资料 21

5 数学函数库：HP MLIB **22**

5.1 MLIB 主要内容 22

5.2 MLIB 目录内容 22

5.3 链接 MLIB 22

5.4 相关资料 25

6 作业管理系统 **26**

6.1 提交作业：bsub 26

6.1.1 提交到特定队列：bsub -q 26

6.1.2 指明所需要的 CPU 数：bsub -n 27

6.1.3 运行 MPI 作业：bsub -a hpmpi 27

6.1.4 运行共享内存作业：bsub -n 27

6.1.5 指明输出、输出文件运行：bsub -i -o -e 27

6.2 终止作业：bkill 27

6.3 挂起作业：bstop 28

6.4 继续运行被挂起的作业：bresume 28

6.5 设置作业最先运行：btop 28

6.6 设置作业最后运行：bbot 28

6.7 修改排队中的作业选项：bmod 29

6.8 查看作业的排队和运行情况：bjobs 29

6.9 查看运行中作业的屏幕正常输出：bpeek 29

6.10 查看各节点的运行情况：lsload 30

6.11 查看各节点的空闲情况：bhosts 30

6.12 查看队列情况：bqueues 30

6.13 查看用户信息：buser 31

6.14 相关资料 32

7 技术支持 **33**

1 HP Superdome 服务器概述

中国科学技术大学超级运算中心的共享内存的每台峰值计算能力为每秒 1920 亿次 HP Integrity Superdome 服务器（以下简称 Superdome 服务器）于 2003 年建成，共四台，每台的具体参数为：

- 32 颗 1.5GHz Itanium 2 Madison 64 位处理器，其中 01 - 03 号机为每台共享 64GB 内存，04 号为共享 128GB 内存
- 存储：SAN 高速网络连接 4TB 容量的 HP EVA8000
- 峰值计算能力：每秒 3840 亿次
- 操作系统：HP-UX
- 编译器：HP aC/C++ Fortran 编译器
- 数学函数库：HP MLIB
- 并行环境：HP MPI，支持 MPI 程序；支持共享内存的程序，比如 OpenMP
- 作业管理：Platform LSF

本指南主要将对在 Superdome 服务器上进行编译以及运行作业做一基本介绍，详细信息请参看相应的指南，超算中心也将为用户需要提供必要的技术支持。

2 用户登录与文件传输

Superdome 服务器操作系统为 HP UX 11i V2 B.11.23，用户需以 ssh 或 telnet 方式登录上此系统后进行编译、运行等操作（建议采用 ssh 协议，用户在 MS Windows 下可利用 putty[1] 等支持 ssh 协议的软件进行登录），用户数据则可以利用 ftp和 sftp 协议进行传输（建议在客户端设置使用安全的 sftp 协议）。

用户登录进来后默认的 shell 为 /usr/bin/sh，可以利用 **rlogin** 登录到管理节点(hpsadmin)上运行 **chsh** 命令修改为自己喜欢的 shell（bash 安装在 /usr/local/bin/bash），修改密码可以在登录节点(hpsuser)上运行 **yppasswd** 命令（利用 **passwd** 命令在登录节点上修改密码无效）。

[1]putty 下载：http://scc.ustc.edu.cn/download/putty.exe

HP-UX 为 UNIX 系统的一种，可以用 **man**␣命令（注意 ␣ 表示空格）或者命令加 -h 或 –help 等选项来查看该命令的详细用法，详细信息请参考 HP-UX 手册，也可参考一般 UNIX 和 Linux 手册。

HP-UX 相关资料：http://scc.ustc.edu.cn/docs/doc-main.php?id=superdome

3 串行及 OpenMP 程序编译

Superdome 服务器上可运行 C/C++、Fortran 的串行程序，以及与 OpenMP 和 MPI 结合的并行程序。编译时，用户只需要在登录节点上以相应的编译命令和选项进行编译即可。当前安装的编译环境为：

- C/C++ 编译器：HP aC++ Compiler for Linux
- Fortran 编译器：HP Fortran Compiler for Linux
- MPI 并行环境：HP MPI

本节主要介绍串行程序和 OpenMP 程序的编译，MPI 的编译将在后面介绍。

3.1 C/C++ 程序的编译

3.1.1 HP aC++/HP C 编译器简介

HP aC++/HP C 编译器是针对 HP-UX 平台的高性能的高级编译器，可用于开发复杂且要进行大量计算的程序，也可对 Fortran 程序进行语言间调用，当前安装的版本为 A.06.10。HP aC++ 编译器支持 C++ 编程语言的 ISO/IEC 14882 标准，安装目录在 /opt/aCC。HP ANSI C 编译器支持 ANSI 编程语言 C 的 ISO 9899:1990 标准。

HP aC++/C 编译系统主要包含下面的内容：

- aCC：编译 C++ 程序
- cc：编译 C 程序
- c89：编译符合 C89 标准的 C 程序
- c99：编译符合 C99 标准的 C 程序

另外一些 HP aC++ 命令为：

- C++filt：名字的识别编码器，它实现名字识别编码算法（编码函数名、类名及参数和名字）
- ld：链接器，链接可执行文件和创建共享库

HP aC++ 运行时库：

- C++ 标准库：
  - /usr/lib/hpux32/libstd.so（32 位共享版本）
  - /usr/lib/hpux32/libstd.a（32 位存档版本）
  - /usr/lib/hpux64/libstd.so（64 位共享版本）
  - /usr/lib/hpux64/libstd.a（64 位存档版本）
- C++ 支持库：
  - /usr/lib/hpux##/libCsup.so
  - /usr/lib/hpux##/libstd.so
  - /usr/lib/hpux##/libstd_v2.so
  - /usr/lib/hpux##/librwtool.so
  - /usr/lib/hpux##/librwtool_v2.so
  - /usr/lib/hpux##/libstream.so

其中 ## 为 32 或 64，还有对应的 .a 文件，为 HP-UX 核心系统的一部分。

- C++ 标准库：
  - /usr/lib/hpux32/libstream.so（32 位共享版本）
  - /usr/lib/hpux32/libstream.a（32 位存档版本）
  - /usr/lib/hpux64/libstream.so（64 位共享版本）
  - /usr/lib/hpux64/libstream.a（64 位存档版本）

这些库函数的头文件存放在 /opt/aCC/include。

3.1.2 输入输出文件后缀与类型的关系

文件的后缀与类型的关系见表 1。

表 1: 输入文件后缀与类型的关系

文件名	解释	动作
filename.c	C 源文件	传给编译器
filename.C	C++ 源文件	传给编译器
filename.a filename.so	库文件	传递给链接器
filename.i	预处理文件	传递给标准输出
filename.o	目标文件	传递给链接器
filename.s	汇编文件	传递给汇编器

3.1.3 aCC 编译举例

- 编译并指定生成的可执行文件名：

  aCC -o prog prog.C

- 编译成可调试的可执行文件：

  aCC -g prog.C

- 只编译成目标文件而不链接：

  aCC -c prog.C

- 链接目标文件成可执行文件：

  aCC file1.o file2.o file3.o

- 编译成 O2 级别优化的可执行文件，并显示详细编译信息：

  aCC -O2 -v prog.C

- 编译生成共享库：

  aCC +z -c prog.C

  aCC -b -o mylib.sl prog.o

  第一行编译将位置无关代码(PIC)编译进目标文件，第二行将可执行代码编译进共享库 mylib.sl。

3.1.4 重要编译选项

与一般编译器以 - 开始表示选项不同，HP aCC/C 编译器有部分独有的选项以 + 开始。

- -AC89：以 ANSI C89 兼容模式编译。
- -AC99：以 ANSI C99 兼容模式编译。
- -Ae：使 aC++ 作为 ANSI C 编译器，并增加了对 HP C 语言的扩展。
- -Ag++：以 GNU C++ 兼容方式编译。
- -Agcc：以 GNU C 兼容方式编译。
- -b：生成动态链接库而不是可执行文件。
- -c：仅编译成目标文件（.o 文件）。
- -C：禁止对注释块中的进行预处理，需要与 -tp 联用。
- +d：编译为可调试程序时禁止展开内联函数，无法在内联函数内设置断点调试时非常有用。
- +DDdata_model：data_model 可以为 32 和 64，分别表示编译成 32 位(ILP32：int、long、pointer 为 32 位)和 64 位(LP64：int 为 32 位，long、pointer 为 64 位)的可执行文件，默认为 32 位。注意：如果不需要 64 位寻址空间，不要编译成 64 位，64 位的会比 32 位的慢。
- -Dname[=def]：指定预处理中的一个符号名。
- +DOosname：编译成针对操作系统版本优化的可执行文件，对当前系统无需指定，或者用 +DO11.23。
- +DSmodel：编译成针对硬件系统指令调度版本优化的可执行文件，对当前系统无需指定，或者用 +DSitanium2。
- -dynamic：生成动态帮定的可执行文件。
- -exec：所有生成的目标文件都被用于生成可执行文件。

- +e 和 -ext：+e 和 ext 等价，表示支持 64 位整数（long long 和 unsigned long long）和 #assert、#unassert 预处理指令。
- -fast：平衡编译时间与最大化整个程序的速度的优化级别，与 +Ofast 等价。
- +FPmode：设置程序启动时如何捕获浮点异常的运行环境，大写字符启动诱捕，小写禁止诱捕，具体请查看手册。
- -g：包含最少调试信息。
- -g0：包含全部调试信息。
- -g1：与 -g 类似产生最少调试信息，但会用某种算法以降低重复信息。
- +[no]gprof：是否准备用 gprof 进行程序概要分析目标文件。
- -H：显示编译时头文件的调用顺序。
- +help：显示命令的详细用法。
- -ipo：进行过程间优化(Interprocedural Optimizations-IPO)。
- -I<头文件目录>：指明头文件的搜索路径。
- -I-：与 -I<头文件目录>结合可以指明头文件的搜索顺序而不是使用默认的搜索顺序。
- +inline_level num：指明内联函数的展开级别，可以为 0 - 9。
- -L<库目录>：指明库的搜索路径。
- -lname：指明所需链接的库名，如库名为 libxyz.a，则可用 -lxyz 指定。
- +noeh：禁止程序的例外处理。如打开此选项，程序中的 throw 和 try 会被报为错误。
- +[no]objdebug：+objdebug 选项使得调试信息存储在目标文件而不是可执行文件中，HP WDB 调试器可以读取目标文件构造调试信息。+noobjdebug 将调试信息存储在目标文件中并链接到可执行文件中，HP WDB 调试器可以读取可执行文件构造调试信息。+objdebug 为默认选项，对大程序来说，可以快速链接并减小可执行文件的大小。

- +O[no]openmp：是否编译成 OpenMP 程序，默认为 +Onoopenmp。
- +O<级别>：设定优化级别，默认为 +O2，-O 与 +O2 相同，推荐使用。+O0 禁止任何优化；+O1 包括分支优化、无效代码去除、更快的注册分配、指令调度和窥孔优化优化；+O2 在 +O1 基础上增加对单个文件中全部函数的优化；+O3 为在 +O2 基础之上增加包含在一个文件中的所有子程序的优化；+O4 在 +O3 的基础上增加对整个程序的优化。
- +O[no]all：是否进行得到可能最大的性能的优化。
- +O[no]aggressive：是否进行激进优化。
- -O：等价于 +O2。
- +Ofast：与 -fast 等价。
- +Ofaster：比 +Ofast 更激进的优化，提升优化级别到 O4。
- +Oprofile=[use|collect]：进行基于程序概要分析的优化(Profile Based Optimization)。
- +O[no]info：是否显示优化信息。
- +O[no]report[=report_type]：是否显示优化报告，report_type 可为 loop、private 和 all。
- +p：禁止允许所有代码中的过时结构。
- -P：只有在命令行中命名的文件不引起其余阶段的预处理时使用。将产生后缀为 .i 的对应文件。
- -S：编译成汇编代码文件而不是目标文件。
- -tx,name：编译时用 name 处理 x，x 可以为 a、c、C、f、l、p、u、x，分别表示处理汇编等，比如 **aCC -ta,/users/sjs/myasmb file.s**。
- -Uname：去除预处理中的某个定义的符号。
- -v：显示详细编译信息。
- -V：显示详细编译器版本信息。

- -w：编译时不显示任何警告，只显示错误。
- +w：编译时显示任何有疑问的警告，不添加此选项的话，默认主要只显示可确定的警告。
- +wendian：设定源代码使用的是 little-endian 还是 big-endian。

建议仔细看看编译器手册中关于程序优化的部分，特别是 IPO 等部分，多加测试，选择适合自己程序的编译选项以提高性能。

3.1.5 C/C++ 程序编译举例

- **cc -o yourprog yourprog.c**
  将 C 程序 yourprog.c 编译为可执行程序 yourprog。
- **aCC -o yourprog yourprog.C**
  将 C++ 程序 yourprog.C 编译为可执行程序 yourprog。
- **cc -o yourprog-omp +Oopenmp yourprog.c**
  将 OpenMP 的 C 程序 yourprog-omp.c 编译为可执行程序 yourprog-omp。

3.1.6 相关资料

- 超算中心主页上的 HP aC++/C ANSI 编译器文档：http://scc.ustc.edu.cn/docs/doc-main.php?id=superdome
- HP aC++/C ANSI 编译器文档：http://docs.hp.com/en/dev.html#aC%2B%2B%20for%20Itanium-based%20Systems
- HP aC++/C ANSI 编译器论坛：http://forums11.itrc.hp.com/service/forums/categoryhome.do?categoryId=150

3.2 Fortran 程序的编译

3.2.1 HP Fortran 编译器简介

HP Fortran 编译器是一种针对 HP-UX 系统的高性能的高级编译器，支持 Fortran 77/90/95，可用于开发复杂且要进行大量计算的程序，也可对 C/C++ 程序进行语

言间调用。HP Fortran 编译器安装在 /opt/fortran90 下，Fortran 程序的编译命令为 **f90**。

3.2.2 输入输出文件后缀与类型的关系

输入文件的后缀与类型的关系见表 2。

表 2: 输入文件后缀与文件类型的关系

文件名	解释
filename.f90	自由格式的 Fortran 源代码
filename.f	固定格式的 Fortran 源代码
filename.F	固定格式的 Fortran 源文件，自动被 Fortran 编译器预处理后再被编译
filename.i90	.f90 源文件从预处理后产生的自由格式输出
filename.i	.f 源文件从预处理后产生的格式输出

3.2.3 重要编译选项

与一般编译器以 - 开始表示选项不同，HP Fortran 编译器有部分独有的选项为以 + 开始。

- +[no]autodbl：是否提升整型、逻辑型和实型的长度为 8 位，双精度和复数型为 16 位。默认为 +noautodbl。
- +[no]autodbl4：是否提升整型、逻辑型和实型的长度为 8 位，复数型为 16 位，但并不提升双精度和双精度复数型。默认为 +noautodbl4。
- -c：仅编译成目标文件（.o 文件）。
- +check=[all|none]：是否检查数组下标，默认为 +check=none。
- -cpp=[yes|no|default]：yes 对所有文件进行预处理，default 对所有 .F 文件进行预处理，no 对所有文件都不进行预处理。
- -[no]cpp_keep：保留或忽略对预处理结果的保留。如果源文件为 file.F 或 file.f，则输出为 file.i；如果源文件为 file.F90 或 file.f90，则输出为 file.i90。

- +DDdata_model：data_model 可以为 32 和 64，分别表示编译成 32 位（ILP32：int、long、pointer 为 32 位）和 64 位（LP64：int 为 32 位，long、pointer 为 64 位）的可执行文件，默认为 32 位。注意：如果不需要 64 位，不要编译成 64 位，64 位的会比 32 位的慢。
- +[no]Dlines：是否认为所有第一列为 d 或 D 开始的行在编译时被认为为声明行，默认为 +noDlines，编译时被认为为注释行。
- -Dname[=def]：指定预处理中的一个符号名。
- +DOosname：编译成针对操作系统版本优化的可执行文件，对当前系统无需指定，或者用 +DO11.23。
- +DSmodel：编译成针对硬件系统指令调度版本优化的可执行文件，对当前系统无需指定，或者用 +DSitanium2。
- -exec：所有生成的目标文件都被用于生成可执行文件。
- +externals=file：利用外部文件 file 指定被认为为外部函数的函数名。
- +[no]escape：是否将 \ 看作为 C 类似的转义符，默认为 +noescape。
- +[no]extend_source：是否扩展源文件的宽度到 254 个字符，默认固定格式的源文件宽度为 72，自由格式源文件的宽度为 132。
- +FPflags：设置程序启动时如何捕获浮点异常的运行环境，大写字符启动诱捕，小写禁止诱捕，具体请查看手册。
- +[no]fp_exceptions：是否允许浮点异常。
- -g：包含最少调试信息。
- +[no]gprof：是否准备用 gprof 进行程序概要分析目标文件。
- +i2：编译时将 4 字节的整数、逻辑常数、内部和用户变量为 2 字节。
- +i8：编译时将 4 字节的整数、逻辑常数、内部和用户变量为 8 字节。

- +[no]implicit_none：设定源码中没有指明类型的变量为未定义还是已定义，默认为 +noimplicit_none。建议在源代码中添加 implicit none 语句，以避免由于类型造成的错误。
- +O[no]info：是否显示优化信息。
- +O[no]inline：设置所有子程序是否都可以内联。
- -I<头文件目录>：指明头文件的搜索路径。
- -I-：与 -I<头文件目录>结合可以指明头文件的搜索顺序而不是使用默认的搜索顺序。
- -L<库目录>：指明库的搜索路径。
- -lname：指明所需链接的库名，如库名为 libxyz.a，则可用 -lxyz 指定。
- +langlvl=[90|default]：显示所有 Fortran 90 标准之外的扩展警告信息，默认为 +langlvl=default 允许扩展。
- +moddir=directory：设置 .mod 文件的存放路径，默认在当前目录下。
- +[no]onetrip：所有由计数控制的 DO 循环至少执行一次，默认为 +noonetrip。
- +[no]objdebug：+objdebug 选项使得调试信息存储在目标文件而不是可执行文件中，HP WDB 调试器可以读取目标文件构造调试信息。+noobjdebug 将调试信息存储在目标文件中并链接到可执行文件中，HP WDB 调试器可以读取可执行文件构造调试信息。+objdebug 为默认选项，对大程序来说，可以快速链接并减小可执行文件的大小。
- +O[no]openmp：是否编译成 OpenMP 程序，默认为 +Onoopenmp。
- +O<级别>：设定优化级别，默认为 +O2，-O 与 +O2 相同，推荐使用。+O0 禁止任何优化；+O1 包括分基本块优化、分支优化、指令调度优化；+O2 在 +O1 基础上增加着色注册器分配、感应变量和强度降低、通用子表达式去除、循环非变化代码动作、存取/复制优化、未使用变量去除、数据流分析、软件流水线操作、标量替换和求和规约优化等；+O3 为在 +O2 基础之上增加过程间的优化、包含克隆和内联和循环变换以提高内存性能，主要包括循环的融合与交

换；+O4 在 +O3 的基础上产生用户代码的媒介表示，并存到一个临时文件中，需与 +Oprofile=use 联用。

- +Ooptlevel=name1 [,name2...]：对特定函数降低优化级别到 optlevel。
- +O[no]all：是否进行得到可能最大的性能的优化。
- +O[no]aggressive：是否进行激进优化。
- -O：等价于 +O2。
- +[no]ppu：是否在外部可见符号的后面添加下划线，默认为 +noppu，如果编译时显示找不到函数名等，而对应的函数名后面有下划线，可以考虑添加 +ppu 选项看看。
- +[no]pipeline：是否进行流水线优化。
- +[no]prof：是否准备用 prof 进行程序概要分析目标文件。
- +Oprofile=[use| collect]：进行基于概要的优化(Profile Based Optimization)。
- +O[no]optimization：使用或者禁止预先定义的指示优化类别的字符串 optimization 进行优化。
- -p：产生用 prof 进行程序概要分析目标文件。
- +p：基于从数据库文件 flow.data 发现的程序概要分析数据进行优化。
- +pre_include=file：编译时预先包含文件 file。
- +pic=[short|long|no]：编译成可以加入共享库中的位置无关的代码。一般用 +pic=short 即可，但在出现数据链接表过长时需换用 +pic=long。+z 等价于 +pic=short，+Z 等价于 +pic=long。
- +r8：将 4 字节的实数、内部和用户变量变为 8 字节。
- -R4：设置实数和复数为 4 字节，等价于 +real_constant=single。
- -R8：设置实数和复数为 8 字节，等价于 +real_constant=double。

- +real_constant=[single|double]：设置默认的单精度实数和复数为单精度还是双精度，默认为+real_constant=single，此选项对于明确声明的变量或用 +autodbl 或 +autodbl4 编译选项指定的变量无效。
- +[no]save：在子程序中的本地变量的值是否保留。
- +[no]shared：设置由链接器生成文件被是否标记为共享，默认为 +shared。
- +source=[fixed|free|default]：设定源文件是固定格式还是自由格式的，默认为 +source=default ，此时格式由后缀决定：对 .f90 程序被认为为自由格式的，而后缀为 .f 和 .F 程序，则被认为为固定格式的。
- -S：编译成汇编代码文件而不是目标文件。
- -tx,path：编译时用 path 下的子进程处理 x，x 可以为 a、c、C、f、l、p、u、x，分别表示处理汇编等，比如 **f90 -ta,/users/sjs/myasmb file.s**。
- -Uname：去除预处理中的某个定义的符号。
- +[no]uppercase：将所有外部函数的名字都看作是大写还是小写的，默认为 +nouppercase。
- +usage：显示此编译命令的详细用法。
- -v：显示详细编译信息。
- -V：显示详细编译器版本信息。
- +version：详细显示编译器版本信息。
- +O[no]vectorize：是否对循环调用向量库，仅在 +O3 及以上优化级别时有效。
- +what：详细显示编译器版本信息，包括补丁号等。
- -w：编译时不显示任何警告，只显示错误。
- +w：编译时显示任何有疑问的警告，不添加此选项的话，默认主要只显示可确定的警告。
- +z：等价于 +pic=short。

- +Z：等价于 +pic=long。

建议仔细看看编译器手册中关于程序优化的部分，特别是 IPO、PGO 和 HLO 部分，多加测试，选择适合自己程序的编译选项以提高性能。

3.2.4 Fortran 程序编译举例

- **f90 -o yourprog yourprog.f**
  将固定格式的 Fortran 程序 yourprog.f 编译为可执行程序 yourprog。
- **f90 -o yourprog yourprog.f90**
  将自由格式的 Fortran 程序 yourprog.f90 静态编译为可执行程序 yourprog。
- **f90 -o yourprog-omp +Oopenmp yourprog.f90**
  将 OpenMP 的自由格式 Fortran 程序 yourprog-omp.f90 编译为可执行程序 yourprog-omp。

3.2.5 相关资料

- 超算中心主页上的 HP Fortran 编译器文档：http://sfc.ustc.edu.cn/docs/doc-main.php?id=superdome
- HP Fortran 编译器文档：http://docs.hp.com/en/dev.html#Fortran
- HP Fortran 编译器论坛：http://forums11.itrc.hp.com/service/forums/categoryhome.do?categoryId=150

3.3 OpenMP 程序的编译与运行

HP aC++/C 和 Fortran 编译器支持 OpenMP 并行，只需要利用编译命令结合 +Oopenmp 编译选项进行编译即可，比如：

- **cc -o yourprog-omp +Oopenmp yourprog.c**
  将 OpenMP 的 C 程序 yourprog-omp.c 编译为可执行程序 yourprog-omp。
- **f90 -o yourprog-omp +Oopenmp yourprog.f90**
  将 OpenMP 的自由格式 Fortran 程序 yourprog-omp.f90 编译为可执行程序 yourprog-omp。

OpenMP 的运行一般是在运行前通过设置环境变量 **OMP_NUM_THREADS** 来控制进程数，比如在 bash 中利用 **export OMP_NUM_THREADS=2** 设置或在 C shell 中利用 **setenv OMP_NUM_THREADS 2** 设置来用 2 个进程运行。

4 MPI 并行环境：HP MPI

Superdome 服务器上安装的 MPI 的并行环境为 HP MPI，安装在 /opt/mpi 下。HP MPI 建立在 MPICH1 之上，用法等基本等同于 MPICH。

4.1 MPI 并行程序的编译

HP MPI 包含的编译命令主要为：**mpicc**、**mpiCC** 和 **mpif90**，对于并行程序，源文件类型和编译命令的对应关系如下：

- **mpicc -o yourprog-mpi yourprog-mpi.c**
  将 C 语言的 MPI 程序 yourprog-mpi.c 编译为可执行程序 yourprog-mpi。
- **mpiCC -o yourprog-mpi yourprog-mpi.C**
  将 C++ 语言的 MPI 程序 yourprog-mpi.C 编译为可执行程序 yourprog-mpi。
- **mpif90 -o yourprog-mpi yourprog-mpi.f**
  将固定格式的 Fortran 语言的 MPI 程序 yourprog-mpi.f 编译为可执行程序 yourprog-mpi。
- **mpif90 -o yourprog-mpi yourprog-mpi.f90**
  将自由格式的 Fortran 语言的 MPI 程序 yourprog-mpi.f90 编译为可执行程序 yourprog-mpi。

HP MPI 中的编译命令实际上是调用 HP aC++/C 和 Fortran 进行编译，具体优化选项等，请参看 HP aC++/C 和 Fortran 的手册。

4.2 MPI 并行程序的运行

在 Superdome 服务器上，MPI 并行程序需结合作业调度系统 LSF 的作业提交命令 **bsub** 来运行，基本用法为 **bsub -q queue1 -o log -e err_file -a hpmpi -n 4 ./yourprog-mpi**，必须添加 -a hpmpi 选项以表示使用 HP MPI 并行环境，详细用法将在作业调度部分进行说明。

4.3 MPI 并行程序调试

服务器缺乏专业的并行程序调试工具，并行程序的调试一般来说只能利用打印语句来逐步定位错误，建议利用尽量少的进程数来调试以方便进行追踪。

4.4 相关资料

- 超算中心主页 HP MPI 文档：http://scc.ustc.edu.cn/docs/doc-main.php?id=superdome
- HP MPI 文档：http://docs.hp.com/en/dev.html#Developer%20Tools%20and%20Libraries
- HP MPI 论坛：http://forums11.itrc.hp.com/service/forums/categoryhome.do?categoryId=150
- MPICH1 主页文档：http://www-unix.mcs.anl.gov/mpi/mpich1/download.html

5 数学函数库：HP MLIB

Superdome 服务器上安装的数学函数库主要有 HP 高性能数学库(MLIB)，用户可以直接调用，以提高性能、加快开发。

5.1 MLIB 主要内容

MLIB 主要包含六个组成部分：

- VECLIB：包含基本线性代数库(BLAS)和傅立叶变换程序(FFT)
- LAPACK：线性代数库
- ScaLAPACK：可扩展性线性代数库
- SuperLU：包含分布式离散线性系统求解库(Sparse Linear System)
- SOLVERS：包含直接离散线性系统求解库(Direct Sparse Linear System Solvers)和图划分程序(Graph Partitioning)，可用于对称多处理架构系统。
- VMATH：向量数学库(Vector Math Routines)，包含 C、C++ 和 Fortran 90 常用的一些标量数学函数。分为两个库：4 字节整数的 VMATH 和 8 字节整数的 VMATH8。

5.2 MLIB 目录内容

MLIB 的主要目录内容如下见表 3。

5.3 链接 MLIB

下面以 VECLIB 为例介绍在程序中链接 MLIB 库，其余库类似。

有几种方式可以链接 VECLIB，通常可以在 f90、cc、c89 命令中利用 -lveclib 选项直接链接，默认链接 32 位的库。当含有 -aarchive_shared 选项时链接存档版本的库(.a)。如果此库不存在，则尝试链接共享库(.so)。如果未含有 -aarchive_shared 和 -ashared_archive，将默认链接共享库。

表 3: MLIB 目录内容

目录	解释
<MLIB_dir>	MLIB 主目录：/opt/mlib
<MLIB_dir>/lib/hpux32/liblapack.#	32 位 LAPACK 库
<MLIB_dir>/lib/hpux32/libscalapack.#	32 位 ScaLAPACK 库
<MLIB_dir>/lib/hpux32/libsolvers.#	32 位 SOLVERS 库
<MLIB_dir>/lib/hpux32/libsuperlu_dist.#	32 位 SuperLU 库
<MLIB_dir>/lib/hpux32/libveclib.#	32 位 VECLIB 库
<MLIB_dir>/lib/hpux32/libvmath.#	32 位 VMATH
<MLIB_dir>/lib/hpux64/liblapack#	64 位 LAPACK 库
<MLIB_dir>/lib/hpux64/liblapack8#	64 位 8 字节整数 LAPACK 库
<MLIB_dir>/lib/hpux64/libscalapack#	64 位 ScaLAPACK 库
<MLIB_dir>/lib/hpux64/libscalapack8#	64 位 8 字节整数 ScaLAPACK 库
<MLIB_dir>/lib/hpux64/libsolvers#	64 位 SOLVERS 库
<MLIB_dir>/lib/hpux64/libsolvers8#	4 位 8 字节整数 SOLVERS 库
<MLIB_dir>/lib/hpux64/libsuperlu_dist#	分布式 64 位 SuperLU 库
<MLIB_dir>/lib/hpux64/libsuperlu_dist8#	分布式 64 位 8 字节整数 SuperLU 库
<MLIB_dir>/lib/hpux64/libveclib#	64 位 VECLIB 库
<MLIB_dir>/lib/hpux64/libveclib8#	64 位 8 字节整数 VECLIB 库
<MLIB_dir>/lib/hpux64/libvmath#	64 位 VMATH
<MLIB_dir>/lib/hpux64/libvmath8#	64 位 8 字节整数 VMATH
<MLIB_dir>/share/man/man3.Z	man 在线文档
<MLIB_dir>/include	所需要的头文件

其中 # 可以为 a、so，分别对应静态和共享链接时所需要的库文件。

- 基本链接方式：
  - f90 [options] file . . . -Wl,-aarchive_shared -lveclib
  - cc [options] file . . . -Wl,-aarchive_shared -lveclib -lcl -lm
  - aCC [options] file . . . -CWl,-aarchive_shared -lveclib -lcl -lm
- 指明库路径的链接方式：
  - f90 [options] file . . . /opt/mlib/lib/[hpux32|hpux64]/libveclib.a
  - cc [options] file . . . /opt/mlib/lib/[hpux32|hpux64]/libveclib.a -lcl -lm
  - aCC [options] file . . . /opt/mlib/lib/[hpux32|hpux64]/libveclib.a -lcl -lm

  如果用 -libveclib.so 代替 -libveclib.a，则将使用共享版本的库。
- 与 Wl,-aarchive_shared,L/opt/mlib/lib/[hpux32|hpux64] 结合使用：
  - f90 [+DD32|+DD64] [options] file . . . Wl,-aarchive_shared,L/opt/mlib/lib/[hpux32|hpux64] -lveclib
  - cc [+DD32|+DD64] [options] file . . . Wl,-aarchive_shared,L/opt/mlib/lib/[hpux32|hpux64] -lveclib -lcl -lm
  - aCC [+DD32|+DD64] [options] file . . . Wl,-aarchive_shared,L/opt/mlib/lib/[hpux32|hpux64] -lveclib -lcl -lm
- 设置 LDOPTS 环境变量包含 -aarchive_shared,L/opt/mlib/lib/[hpux32|hpux64] 以链接，比如 export LDOPTS="-aarchive_shared,-L/opt/mlib/lib/hpux32"，然后用 -lveclib 选项链接：
  - f90 [options] file . . . -lveclib
  - cc [options] file . . . -lveclib -lcl -lm
  - aCC [options] file . . . -lveclib -lcl -lm
- 利用 +DD64 选项链接 VECLIB8 库生成 64 位可执行程序，并可结合 +i8、+autodbl 和 +autodbl4：
  - f90 +DD64 +i8 [options] file . . . Wl,-aarchive_shared -lveclib8

- cc +DD64 [options] file . . . Wl,-aarchive_shared -lveclib8 -lcl -lm
- aCC +DD64 [options] file . . . Wl,-aarchive_shared -lveclib8 -lcl -lm

其余 LAPACK、ScaLAPACK、分布式 SuperLU、SOLVERS 和 VMATH，类似可以分别通过 -llapack、-lveclib、-lscalapack、-lsuperlu_dist、-lsolvers、-lvmath 或 -llapack8、-lveclib8、-lscalapack8、-lsuperlu_dist8、-lsolvers8 lvmath8 等链接，具体请查看 HP MLIB 手册。

5.4 相关资料

- 超算中心主页的 HP MLIB 资料：http://scc.ustc.edu.cn/docs/doc-main.php?id=superdome
- HP MLIB 文档：http://docs.hp.com/en/dev.html#Developer%20Tools%20and%20Libraries
- HP MLIB 论坛：http://forums11.itrc.hp.com/service/forums/categoryhome.do?categoryId=150

6 作业管理系统

Superdome 服务器利用 Platform 公司的 LSF 进行资源和作业管理，所有需要运行的作业均必须通过作业提交命令 **bsub** 提交，提交后可利用相关命令查询作业状态等。为了利用 **bsub** 提交作业，需要在 **bsub** 中指定各选项和需要执行的程序。

注意：

- 不要在登录节点直接运行（编译除外），以免影响其余用户的正常使用
- 如果不通过作业调度系统直接在计算节点上运行将会被监护进程直接杀掉

6.1 提交作业：bsub

用户需要利用 **bsub** 提交作业，其基本格式为 **bsub** [**options**] **command** [**arguments**]，其中 options 和 arguments 分别为可设置队列、CPU 数等的选项和作业本身所需要的参数，下面将给出常用的几种提交方式。

6.1.1 提交到特定队列：bsub -q

Superdome 服务器为共享架构，共四台，每台含有 32 颗 Itanium2 CPU，其中 01 - 03 号机为 64GB 内存，04 号机 128GB 内存。利用 -q 选项可以指定提交到哪个队列，现有的队列分为两类：

- queue1 - queue7：分别对应几个重点学科，只有属于此重点学科的用户才可使用对应的队列，具体对应关系，请注意登录后的提示。
- idle：级别很低的队列，任何用户可使用，作业运行临时占用各学科空闲资源，当被占学科需使用时，将自动挂起，建议运行小作业，idle 队列的具体策略会根据运行情况进行调整，详细信息可以利用 **bqueues** 查看。

比如想提交到 queue1 队列运行串行程序 executable1，可以：

bsub -q queue1 executable1

如果提交成功，将显示类似下面的输出：

```
Job <79722> is submitted to default queue <queue1>.
```

其中 79722 为此作业的作业号，以后可利用此作业号来进行查询及终止等操作。

6.1.2 指明所需要的 CPU 数：bsub -n

利用 -n 选项指定可指定所需要的 CPU 数（Superdome 服务器最大允许运行的单个作业 CPU 数为 16），比如下面指定利用 4 颗 CPU（由 -n 4 指定）运行 MPI（由 -a hpmpi 指定）程序：

bsub -q queue1 -a hpmpi -n 4 executable-mpi1

6.1.3 运行 MPI 作业：bsub -a hpmpi

如果需要运行 MPI 作业，需要添加 -a hpmpi 选项，并用 -n 选项指定所需的 CPU 数，比如下面指定利用 16 颗 CPU 运行 MPI 程序 executable-mpi1：

bsub -q long -a hpmpi -n 16 executable-mpi1

6.1.4 运行共享内存作业：bsub -n

Superdome 服务器为共享内存架构，可以运行共享内存的程序，比如 OpenMP 程序，只要设置 OpenMP 所需要的环境变量 **OMP_NUM_THREADS** 与 -n 指定的 CPU 数相同即可（Superdome 服务器最大允许运行的单个作业 CPU 数为 16），比如下面指定利用 4 颗 CPU（由 -n 4 指定）运行 OpenMP 程序：

bsub -q queue1 -n 4 executable-omp1

6.1.5 指明输出、输出文件运行：bsub -i -o -e

作业的输入文件、正常屏幕输出到的文件和错误屏幕输出的文件可以利用 -i、-o 和 -e 选项来分别指定，运行后可以通过查看指定的这些输出文件来查看运行状态。比如指定 executable1 的输入、正常和错误屏幕输出文件分别为：executable1.input、executable1.log 和 executable1.err：

bsub -q queue1 -i executable1.input -o executable1.log -e executable1.err executable1

6.2 终止作业：bkill

利用 **bkill** 命令可以终止某个运行中或者排队中的作业，比如：

bkill 79722

运行成功后，将显示类似下面的输出：

```
Job <79722> is being terminated
```

6.3 挂起作业：bstop

利用 **bstop** 命令可临时挂起某个作业以让别的作业先运行，例如：

bstop 79727

运行成功后，将显示类似下面的输出：

```
Job <79727> is being stopped.
```

此命令可以将排在队列前面的作业临时挂起，以让后面的作业先运行。建议不要随便对运行中的作业进行挂起操作，如果运行中的作业不再想继续运行，请用 **bkill** 终止。

6.4 继续运行被挂起的作业：bresume

利用 **bresume** 命令可继续运行某个挂起某个作业，例如：

bresume 79727

运行成功后，将显示类似下面的输出：

```
Job <79727> is being resumed.
```

6.5 设置作业最先运行：btop

利用 **btop** 命令可最先运行排队中的某个作业，例如：

btop 79727

运行成功后，将显示类似下面的输出：

```
Job <79727> has been moved to position 1 from top.
```

6.6 设置作业最后运行：bbot

利用 **bbot** 命令可设定最后运行排队中的某个作业，例如：

bbot 79727

运行成功后，将显示类似下面的输出：

```
Job <79727> has been moved to position 1 from bottom.
```

6.7 修改排队中的作业选项：bmod

利用 **bmod** 命令可修改排队中的某个作业的选项，比如想将排队中的运行作业号为 79727 的作业的执行命令修改为 executable2 并且换到 queue1 队列，可以：

bmod -Z "executable2" -q "queue1" 79727

```
Parameters of job <79727> are being changed.
```

6.8 查看作业的排队和运行情况：bjobs

利用 **bjobs** 可以查看作业的运行情况，比如有哪些作业在运行，哪些在排队，某个作业运行在哪个节点上，以及为什么没有运行等，例如：

bjobs

```
JOBID USER STAT QUEUE  FROM_HOS EXEC_HOST JOB_NAME   SUBMIT_TIME
79726 hmli RUN  queue1 hpsuser  2*hpsd01  *executab1 Apr 27 19:20
79727 hmli PEND idle   hpsuser            *executab2 Apr 27 19:20
```

上面显示作业 79726 在运行，在 hpsd01 上运行 2 个进程；而作业 79727 处于排队中，尚未运行，查看未运行的原因可以利用：

bjobs -l 79727：

```
Job Id <79727>, User <hmli>, Project <default>, Status <PEND>,
Queue <idle> , Command <executab2>
Sun Apr 27 14:15:07: Submitted from host <hpsuser>,
CWD <$HOME>, Requested Resources <type==any && swp>35>;
PENDING REASONS:
SCHEDULING PARAMETERS:
            r15s r1m r15m  ut pg   io  ls   it  tmp  swp  mem
loadSched -   0.7 1.0  - 4.0  -   -    -    -    -    -
loadStop  -   1.5 2.5  - 8.0  -   -    -    -    -    -
```

6.9 查看运行中作业的屏幕正常输出：bpeek

利用 **bpeek** 命令可查看运行中作业的屏幕正常输出，例如：

bpeek 79727

```
<< output from stdout >>
---------------------------------------

Radius(nm): 300.000
```

如果在运行中用 -o 和 -e 分别指定了正常和错误屏幕输出，则可以通过直接查看指定的文件的内容来查看屏幕输出。

6.10 查看各节点的运行情况：lsload

利用 **lsload** 命令可查看当前各节点的运行情况，例如：

lsload

HOST_NAME	status	r15s	r1m	r15m	ut	pg	ls	it	tmp	swp	mem
hpsuser	ok	1.0	1.0	1.0	0%	1.7	0	174	56G	1878M	762M
hpsadmin	ok	1.0	1.0	1.0	0%	0.0	0	53035	57G	2044M	650M
hpsd01	ok	17.0	17.0	17.1	50%	0.0	0	9480	54G	4096M	50G
hpsd03	ok	22.2	21.9	22.0	65%	157.9	0	52864	54G	4096M	58G
hpsd02	ok	27.8	28.5	29.1	90%	11.1	0	53024	26G	4096M	45G
hpsd04	ok	30.5	30.0	28.8	88%	339.2	0	53024	50G	4096M	99G

ut 列表示利用率。

6.11 查看各节点的空闲情况：bhosts

利用 **bhosts** 命令可查看当前各节点的空闲情况，例如：

bhosts

HOST_NAME	STATUS	JL/U	MAX	NJOBS	RUN	SSUSP	USUSP	RSV
hpsadmin	closed	–	2	0	0	0	0	0
hpsd01	ok	–	32	16	16	0	0	0
hpsd02	closed	–	32	32	32	0	0	0
hpsd03	ok	–	32	21	21	0	0	0
hpsd04	ok	–	32	30	30	0	0	0
hpsuser	ok	–	2	0	0	0	0	0

STATUS 列中的 ok 表示可以接收新作业，closed 表示已经被占满。

6.12 查看队列情况：bqueues

利用 **bqueues** 可以查看现有队列信息，例如：

bqueues

QUEUE_NAME	PRIO	STATUS	MAX	JL/U	JL/P	JL/H	NJOBS	PEND	RUN	SUSP
queue1	53	Open:Active	32	8	–	–	88	72	16	0
queue2	43	Open:Active	16	–	–	–	0	0	0	0
queue3	43	Open:Active	16	–	–	–	32	16	16	0
queue4	43	Open:Active	16	–	–	–	48	32	16	0
queue5	43	Open:Active	16	–	–	–	13	0	13	0
queue6	43	Open:Active	16	–	–	–	14	0	14	0
queue7	43	Open:Active	16	–	–	–	16	0	16	0
normal	30	Closed:Active	–	6	–	–	0	0	0	0
idle	10	Open:Active	8	8	–	–	8	0	8	0

其中，主要列的含义为：

- QUEUE_NAME：队列名
- PRIO：优先级，数字越大优先级越高
- STATUS：状态。Open:Active 表示已激活，可使用；Closed:Active 表示已关闭，不可使用
- MAX：队列对应的最大 CPU 数，- 表示无限，以下类似
- JL/U：单个用户同时可以运行的 CPU 数
- NJOBS：排队、运行和被挂起的总作业所占的 CPU 数
- PEND：排队中的作业所需的 CPU 数
- RUN：运行中的作业所占的 CPU 数
- SUSP：被挂起的作业所占的 CPU 数

6.13 查看用户信息：buser

利用 **buser** 可以查看用户信息，例如：

busers hmli

USER/GROUP	JL/P	MAX	NJOBS	PEND	RUN	SSUSP	USUSP	RSV
hmli	–	–	40	32	8	0	0	0

6.14 相关资料

- 超算中心主页的 LSF 资料：http://scc.ustc.edu.cn/docs/doc-main.php?id=superdome
- Platform 公司 LSF 资料：http://www.platform.com/Products/platform-lsf-family

7 技术支持

中国科学技术大学超级运算中心主页为：http://scc.ustc.edu.cn。超级运算中心将为用户提供相应的技术支持，如有需要请联系：

李会民

- 电话：0551-3602248
- 电邮：hmli@ustc.edu.cn