

## 实验三 MPI

### 3.1 实验目的

安装 MPICH、学会多节点（用多台虚拟机）并行编程。

### 3.2 MPI介绍

MPI (Message Passing Interface) 是目前发展较快、使用面较广的一个公共的消息传递库，顾名思义，就是在现有机器的软硬件通信基础上，实现了并行应用程序中各并行任务之间的相互通信、协调和同步功能，并对这些并行任务进行管理。MPI 是在众多专家经多次会谈、综合考虑了 NX/2、Express、Vertex、p4、PARMACS 和 PVM 的优点的前提条件下，于 1993 年形成的一个标准，可谓后起之秀。它具有可移植性、高效性、灵活性和易用性的特点，不仅适用于具有分布式内存的大型机、工作站机群，也可用于具有共享内存的大型机。MPI 标准的常用实现有 OpenMPI 和 MPICH。

### 3.3 实验内容

以下实验步骤已在 Windows 11+VMware Workstation17+CentOS 7 下验证通过

首先检查虚拟机有否安装 gcc 开发软件包，若没有，可以通过 yum 安装：  
#yum -y install gcc gcc-c++ kernel-devel //安装 gcc、c++编译器以及内核文件

#### 1. MPICH 的安装（用 OpenMPI 也可以）

##### 1.1 安装

下载 MPICH 包（版本自选），解压、解包到某子目录。

```
#wget http://www.mpich.org/static/downloads/3.4/mpich-3.4.tar.gz
```

```
#tar -xzf mpich-3.4.tar.gz (假设当前目录为/root)
```

```
#cd mpich-3.4
```

假设要将 mpich 安装到/opt/mpich-3.4 目录下：

```
#!/configure --prefix=/opt/mpich-3.4
```

如果提示 Fortran 相关错误，可以选择输入：./configure --prefix=/opt/mpich-3.4

```
--with-device=ch4:ofi -disable-fortran
```

（或者你装一下 Fortran 有关的库）

注：/opt/mpich-3.4 是安装路径，configure.log 保存屏幕显示。

如果 configure 过程中，出现错误：no ch4 netmod selected，则可以按照提示加上 --with-device=ch4:ofi 重新运行 configure。

```
输入：make && make install
```

##### 1.2 设置检索路径

使得 MPI 程序编译和运行时能找到相关的命令、头文件、库文件。若是超级用户安装，则可以统一添加一条辅助检索路径。方法为修改用户目录下的：

~/.bashrc 文件，在适当（最后）位置添加：

```
export MPICH=/opt/mpich-3.4
```

```
export PATH=$MPICH/bin:$PATH
```

```
export INCLUDE=$MPICH/include:$INCLUDE
```

```
export LD_LIBRARY_PATH=$MPICH/lib:$LD_LIBRARY_PATH
```

运行以下命令使修改生效:

```
# source ~/.bashrc
```

注: 可通过命令 `env | grep -w PATH` 来查看环境变量修改是否成功。可运行目录 `examples` 下的例子程序查看是否安装成功

在安装时可能缺少一些支持软件, 可以通过 `yum install **` 进行安装, 例如:

```
*** C++ compiler and preprocessor
```

```
checking for g++... no
```

```
checking for c++... no
```

```
checking for gpp... no
```

```
.....
```

```
configure: error: C++ preprocessor "/lib/cpp" fails sanity check
```

```
See 'config.log' for more details
```

是由于没有编译器, 因此可以网上查找软件的名称, 然后安装。如下指令可以完成安装该编译器

```
# yum install gcc-c++
```

### 1.3 放权 (用 ssh 或者 rsh 命令能互相登陆)

在每台虚拟机创建文件 `/etc/hosts` 文件中输入主机名和 IP 对应, 例如:

```
192.168.62.129 master
```

```
192.168.62.221 slave1
```

```
192.168.62.221 slave2
```

在每台虚拟机创建文件 `/etc/hosts.equiv` 文件 (定义了哪些计算机和用户可以不用提供口令就在本地计算机上执行远程命令, 如 `rexec`, `rcp`, `rlogin` 等等), 放权。例如:

```
master
```

```
slave1
```

```
slave2
```

放权测试: 能 ssh 到自己和其他机器, 并能从其他机器 ssh 到自己。

### 1.4 单节点上测试 MPICH 安装是否成功

```
#cd examples
```

```
#mpirun -np 4 ./cpi
```

```
[root@master examples]# mpirun -np 4 ./cpi
Process 0 of 4 is on master
Process 3 of 4 is on master
Process 2 of 4 is on master
Process 1 of 4 is on master
pi is approximately 3.1415926544231239, Error is 0.0000000008333307
wall clock time = 0.073888
[root@master examples]# _
```

## 2. 在每台虚拟机上都安装 mpich

上述安装过程需要在每台虚拟机上都进行一遍。为节省配置和编译时间, 可

以通过 NFS 把 mpich-3.4 源代码目录共享到其它虚拟机的相同目录上, 假设上述安装时源代码目录是:

```
/root/mpich-3.4
```

则在其它虚拟机上, 可以直接安装:

```
#cd /root/mpich-3.4
```

```
#mkdir /opt/mpich-3.4
```

```
#make install
```

同样方法修改/etc/profile 和/etc/hosts、/etc/hosts.equiv。

### 3. 多节点上运行 MPI 示例程序

编写并运行自己的 MPI 源程序。简单地, 也可使用例子程序。

将例子程序(cpi.c)复制到当前目录, 命令如下:

```
#cp <安装 MPICH 的目录>/examples/cpi* .
```

编译 MPI 程序(以 cpi.c, fpi.f 为例)

```
#mpicc -o cpi cpi.c
```

运行 MPI 程序 (smpd 模式——同样的程序多数据流)。将可执行程序 (例如 cpi 程序) 远程拷贝 rcp 或者 scp 到其它节点的相应子目录中。如果用 NFS 共享, 就无需拷贝。

```
rcp cpi [用户名@]机器名 1:路径
```

```
rcp cpi [用户名@]机器名 2:路径
```

```
.....
```

```
rcp cpi [用户名@]机器名 n:路径
```

编辑机器表配置文件, 文件名是 hostlist (可以改), 文件的每一行为一台节点机器名, 其中第一行为本机名。节点名可重复出现, 表示在节点上将启动若干个进程运行, 节点名后面可说明进程数, 例如:

```
master:3
```

```
slave1:1
```

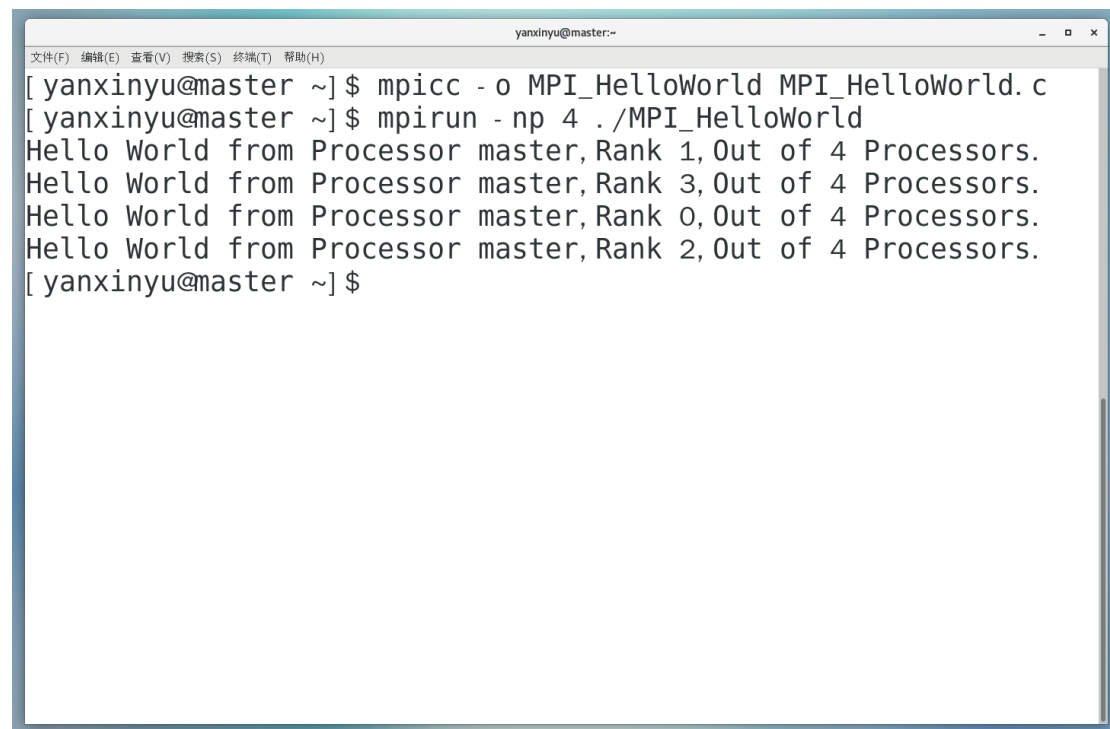
```
slave2:2
```

运行: mpirun -machinefile hostlist -np 4 cpi

```
[root@master examples]# mpirun -machinefile hostlist -np 4 ./cpi
Process 3 of 4 is on slave
Process 1 of 4 is on master
Process 2 of 4 is on master
Process 0 of 4 is on master
pi is approximately 3.1415926544231243, Error is 0.0000000008333312
wall clock time = 0.055360
[root@master examples]#
```

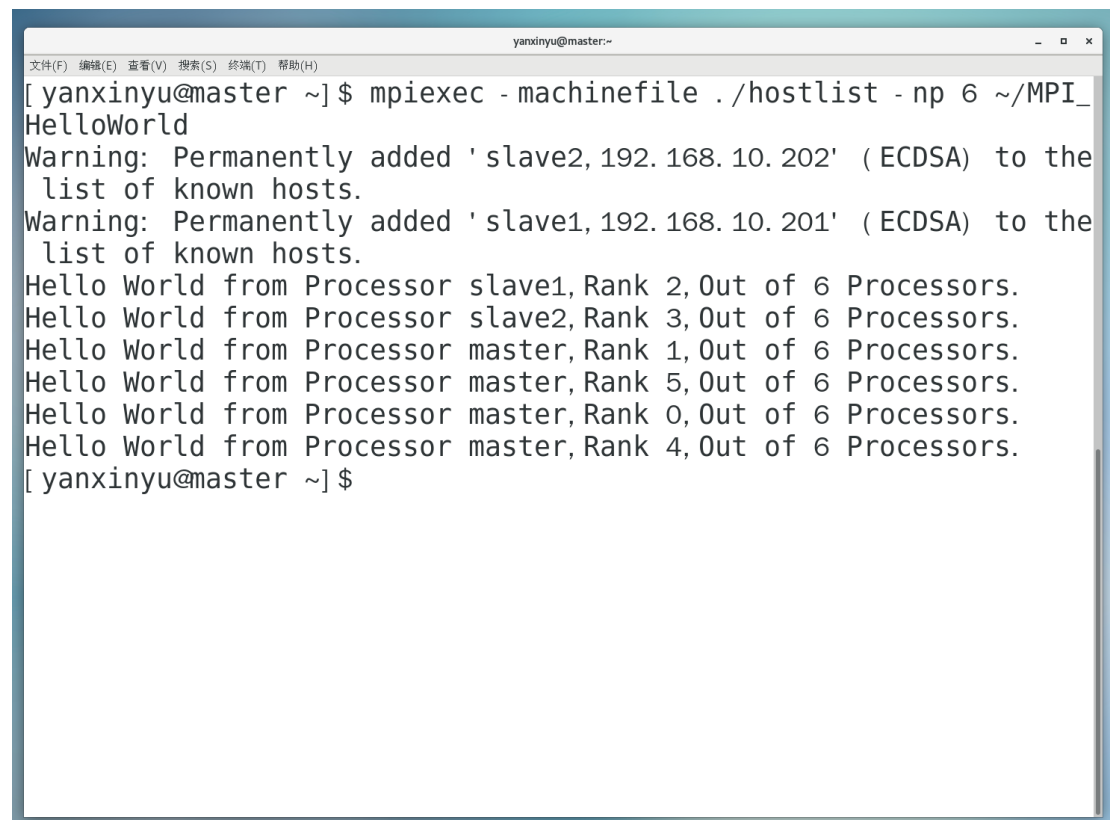
其他一些效果图，可以观察里面的命令

单机 HelloWorld:



```
yanxinyu@master:~  
文件(F) 编辑(E) 查看(V) 搜索(S) 终端(T) 帮助(H)  
[yanxinyu@master ~]$ mpicc -o MPI_HelloWorld MPI_HelloWorld.c  
[yanxinyu@master ~]$ mpirun -np 4 ./MPI_HelloWorld  
Hello World from Processor master, Rank 1, Out of 4 Processors.  
Hello World from Processor master, Rank 3, Out of 4 Processors.  
Hello World from Processor master, Rank 0, Out of 4 Processors.  
Hello World from Processor master, Rank 2, Out of 4 Processors.  
[yanxinyu@master ~]$
```

多机 HelloWorld:



```
yanxinyu@master:~  
文件(F) 编辑(E) 查看(V) 搜索(S) 终端(T) 帮助(H)  
[yanxinyu@master ~]$ mpiexec -machinefile ./hostlist -np 6 ~/MPI_HelloWorld  
Warning: Permanently added 'slave2,192.168.10.202' (ECDSA) to the list of known hosts.  
Warning: Permanently added 'slave1,192.168.10.201' (ECDSA) to the list of known hosts.  
Hello World from Processor slave1, Rank 2, Out of 6 Processors.  
Hello World from Processor slave2, Rank 3, Out of 6 Processors.  
Hello World from Processor master, Rank 1, Out of 6 Processors.  
Hello World from Processor master, Rank 5, Out of 6 Processors.  
Hello World from Processor master, Rank 0, Out of 6 Processors.  
Hello World from Processor master, Rank 4, Out of 6 Processors.  
[yanxinyu@master ~]$
```