Приложение 2.
Среды параллельного программирования MPI

В данном приложении будут рассмотрены некоторые вопросы, связанные с установкой и использованием пакетов mpich, mpich2, Intel MPI Library, mvapich, ScaMPI и IBGOLD. Найти дистрибутивы этих пакетов в сети не представляется сложным – достаточно набрать название в любой поисковой системе, например, google, yandex или rambler.

Существующие на сегодня различные варианты реализации MPI не ограничиваются только теми, которые обсуждаются в данном приложении. На практике широко используются такие пакеты, как LAM, OpenMPI, mpich-gm, MPI/PRO, NT-mpich и некоторые другие. Для большей части из них процедура установки и настройки почти полностью совпадает с пакетами, приведенными в данном разделе, а незначительные отличия можно легко найти из описания, которое есть практически во всех дистрибутивах.

Еще раз повторим, что не рекомендуется использовать готовые пакеты mpich из дистрибутивов операционных систем, так как настроить их сложнее, а качество сборки не всегда удовлетворительное.

mpich

Идеология mpich предполагает, что обмен сообщениями будет идти через некое абстрактное устройство `device`. При сборке необходимо явно указать, какое устройство будет использовано. Наиболее часто используется устройство `ch_p4`, которое позволяет обмениваться сообщениями через TCP/IP. Запуск процессов задачи на узлах происходит через `rsh` или `ssh`.

Вариантом устройства ch_p4 является ch_p4mpd. В этом варианте для запуска программы на узлах и контроля запущенных процессов, в принципе, могут использоваться специальные программы-мониторы mpd. Увы, качество работы этого варианта оставляет желать лучшего, поэтому мы не рекомендуем его использовать.

Из других устройств можно отметить ch_shmem, при использовании которого обмен будет происходить только через общую память. Для кластера использовать не рекомендуется.

Для сборки mpich, распакуйте дистрибутив командой

```
tar xfvz mpich-1.X.X.tar.gz
```

Войдите в созданный каталог и наберите

```
./configure --prefix=/opt/mpich -with-device=ch_p4
```

Вместо /opt/mpich можно указать иной каталог, например, /usr/local. Если предполагается использование компиляторов, отличных от gcc/g77, то укажите к ним пути:

```
-c++=CXXPATH -cc=CCPATH -fc=FCPATH -f90=F90PATH\
-clinker=CLINK -c++linker=CXXLINK\
-flinker=FLINK -f90linker=F90LINK
```

Здесь CXXPATH, CCPATH, FCPATH и F90PATH – пути к компиляторам C++, C, Фортран и Фортран-90 соответственно, а CLINK, CXXLINK, FLINK и F90LINK – пути к соответствующим линкерам. Например, в случае с компилятором Intel нужно указать опции:

```
-c++=icc -cc=icc -fc=ifort -f90=ifort -clinker=icpc
-c++linker=icpc -flinker=ifort -f90linker=ifort
```

Для задания дополнительных ключей компиляторам можно указать опции -cflags= , -c++flags= , -fflags= , -f90flags.

Если нужно явно указать программу, которая будет использоваться для доступа на узлы (по умолчанию это /usr/bin/ssh), то задайте ключ rsh=RSHCOMMAND.

Если configure отработает без ошибок, можно запустить команду make. В противном случае постарайтесь разобраться, почему не отработала configure. Например, это может быть вызвано отсутствием каких-либо

компиляторов или библиотек. Отключить ненужные компиляторы можно опциями

`--disable-cxx, --disable-f77, --disable-f90.`

После успешной отработки `make`, наберите `make install` для установки всего программного комплекса.

Теперь можно компилировать программы командами `mpicc` (Си), `mpicxx` (Си++), `mpif77` (Фортран) и `mpif90` (Фортран-90). Откомпилированные программы можно запускать командой `mpirun`. Например, следующая команда

```
mpirun -np 10 ./a.out
```

запустит программу `a.out` из текущего каталога на 10 процессорах. На каких именно процессорах? Чтобы указать это явно, потребуется создать файл со списком узлов (по одному узлу в строке файла) или исправить глобальный список `$INSTALL_DIR/share/machinefile.LINUX`. Если создается отдельный файл, то его надо указать команде `mpirun` с помощью ключа `-machinefile`.

Не забудьте о том, что сама параллельная программа и ее входные файлы должны быть расположены на сетевом диске либо скопированы на вычислительные узлы.

mpich2

Процесс сборки этого программного комплекса в целом аналогичен сборке mpich, но есть и некоторые отличия, на которых мы остановимся.

- По умолчанию привязки к Си++, Фортрану и Фортрану-90 отключены, чтобы их добавить, используйте опции
  `--enable-f77, --enable-f90 и --enable-cxx.`
- Устройства в mpich2 отличаются от устройств в mpich. По умолчанию используется устройство `c3`, которое, как и `ch_p4` использует передачу сообщений через TCP/IP. Используя опции

`--with-device=c3:shm` или `--with-device=c3:ssm`, можно указать, что общаться надо через общую память или через общую память на SMP и через TCP/IP на разных машинах, соответственно.

- В mpich2 есть встроенная поддержка InfiniBand (на данный момент только через Mellanox Verbs API), включить ее можно следующим набором ключей:
  `--with-device=ch3:ib -with-ib=vapi`
  `--with-ib-path=PATH_TO_MELLANOX_IB`.
- Можно использовать поддержу Myrinet, используя драйвер GM и библиотеку GASNet. Для этого надо указать опции
  `--with-device=ch3:gasnet -with-gasnet-conduit=gm`
  `--with-gm=GM_INSTALL_DIR`.

Перед началом работы пользователь должен создать в домашнем каталоге файл `.mpd.conf`, содержащий строку '`secretword=<pass>`', где `<pass>` – это некоторое "секретное" слово (ни в коем случае не пароль в систему!). После этого надо установить права доступа на него командой: `chmod 600 .mpd.conf`.

В mpich2 программы-мониторы `mpd` используются в обязательном порядке. Для запуска мониторов необходимо создать файл `/etc/mpd.hosts` со списком всех узлов. Затем надо запустить команду `mpdboot -n N`, где `N` – число узлов+1 (один `mpd` должен работать на головном узле). Можно также запускать `mpd` вручную или автоматически на каждом узле. Проверить работоспособность мониторов можно командой `mpdtrace`.

Теперь уже можно запускать задачи, что делается либо старой программой `mpirun`, либо с помощью новой программы `mpiexec`. Список узлов команде `mpiexec` можно передать ключом `-machinefile mfile`. В файле `mfile` должны быть перечислены узлы в формате: `<node:ncpu>`, где `node` – имя узла, `ncpu` – число процессоров на нем. Ключом `-np N` указывается общее число процессоров. Альтернативный способ указать

узлы для запуска – это указать ключи `-hosts N host1 ... hostN`, где `N` равно числу процессов, а `host1 ... hostN` описывают нужные узлы.

Intel MPI Library

Данная реализация MPI основана на mpich2. Её ключевой особенностью является прозрачная поддержка различных коммуникационных сред. Это значит, что в среде с гетерогенной коммуникационной средой можно будет запускать параллельное приложение даже без перекомпиляции. Например, к кластеру на InfiniBand можно подключить ещё несколько узлов через Ethernet или Myrinet и приложения сразу смогут их использовать.

Это достигается за счёт использования динамического выбора типа коммуникации. Доступные варианты: через общую память, через Ethernet или через любое устройство, поддерживающее RDMA (Remote Direct Memory Access) через библиотеку DAPL. Такая библиотека поставляется с большинством высокоскоростного оборудования, такого как InfiniBand и Myrinet. Несмотря на возможность динамической поддержки работы приложения с различными коммуникационными сетями, одновременное использование процессоров с различным набором команд не допускается.

Intel MPI Library полностью реализует стандарт MPI-2 и поддерживает работу программ на архитектурах x86, x86_64 и процессорах семейства Itanium. Поддерживаются компиляторы Intel C/C++/F77/F90 версий 8.1 и выше, а также любые компиляторы, совместимые с форматом GNU.

Установка производится запуском скрипта `install.sh` из каталога, где был распакован архив с дистрибутивом. Для успешной инсталляции необходимо иметь лицензию, которую для удобства установки пакета нужно сохранить в файле. В процессе работы инсталлятор попросит указать каталог для установки (по умолчанию это `/opt/intel/mpi/версия`), а также путь к файлу с лицензией и набор

компонент для установки. Для успешной работы необходимо наличие на узлах и сервере интерпретатора python версии не ниже 2.2.

После установки Intel MPI Library команды компиляции и запуска будут доступны только с указанием полных путей, что, конечно, неудобно. Для комфортной работы на 32-битных процессорах необходимо выполнить скрипт `<путь_к_intelmpi>/bin/mpivars.sh`, если вы работаете в bash или zsh или `<путь_к_intelmpi>/bin/mpivars.csh`, если вы работаете в csh или tcsh. На 64-битных процессорах вместо `<путь_к_intelmpi>/bin/` пишите `<путь_к_intelmpi>/bin64/`. Чтобы не выполнять эти действия каждый раз, пропишите их запуск в файлах `/etc/profile` и/или `/etc/csh.cshrc` соответственно. Во многих дистрибутивах Linux есть каталог `/etc/profile.d`, в котором хранится набор скриптов, выполняющихся bash и tcsh при запуске. Вы можете не «захламлять» `/etc/profile` и `/etc/csh.cshrc`, а просто сделать жёсткие ссылки указанных выше скриптов в этот каталог. Символические ссылки, скорее всего, не сработают в силу специфической обработки каталога `/etc/profile.d`.

Компиляция программ производится обычными скриптами `mpicc/mpiCC/mpif77/mpif90`. Для запуска программ необходимо предварительно создать файл `$HOME/.mpd.conf`, с правами на чтение-запись владельцу и запрещением всего остальным (`0600`). В этом файле необходимо прописать строку '`secretword=<mpd_secret_word>`', где `mpi_secret_word` – это произвольный набор символов. Не пишите туда свой пароль! Убедитесь, что файл доступен на всех узлах кластера. Создайте файл `mpd.hosts` со списком всех узлов кластера, по одному в каждой строке.

Перед запуском пользовательской программы запустите команду `mpdallexit`, а затем `mpdboot -n <Nodes_Number>`, где `Nodes_Number` – число узлов, перечисленных в `mpd.hosts`. Для работы с узлами должен быть настроен беспарольный доступ по `rsh` или `ssh`. Если используется `ssh`, то `mpdboot` надо запускать с ключом `-r ssh`.

Собственно запуск программ пользователей осуществляется командой `mpiexec -n <Nodes_Number> <путь_к_программе>`. Если по какой-то причине нужно использовать определённую среду, то можно запретить автоматический выбор, указав `mpiexec` ключ `-genv I_MPI_DEVICE <device>`, где параметр `device` может принимать значения: `rdma` (использовать библиотеку DAPL), `shm` (общая память), `sock` (TCP/IP через Ethernet), `ssm` (`socket+shm`) или `rdssm` (`socket+shm+rdma`). Если среда была указана явно, то смена среды в динамике становится невозможной.

Дополнительная информация по Intel MPI Library доступна в Интернет на странице `http://www.intel.com/go/mpi`.

mvapich

Несмотря на то, что этот программный комплекс в свое время выделился из mpich, процесс его установки значительно отличается. Мы будем описывать версию gen2, как наиболее перспективную. mvapich ориентирован на работу с InfiniBand и его основное устройство (в терминах mpich) называется `ch_gen2`.

Для установки комплекса потребуются драйвер InfiniBand и библиотека `vapi`. Процесс установки начинается запуском скрипта `mvapich.make.gcc`. Предусмотрены варианты для компилятора Intel: `mvapich.make.icc` и `mvapich.make.ecc`, а также для компилятора Portland Group – `mvapich.make.pgi`. Можно воспользоваться и ручным методом с запуском `configure`, но в этом случае надо внимательно почитать руководство.

После завершения настройки можно запустить команду `make`, и, если сборка прошла успешно, то `make install`. Так как программы по умолчанию будут собираться с динамическими библиотеками mvapich, то нужно скопировать каталог с динамическими библиотеками на все узлы. После этого, необходимо на всех узлах дописать путь к этим библиотекам в файл `/etc/ld.so.conf` и выполнить команду `ldconfig`.

Программа `mpirun` будет присутствовать в числе установленных, но работать она не будет. Запуск программ осуществляется командой `mpirun_rsh`. Синтаксис запуска:

```
mpirun_rsh -np N host1 ... hostN program args...
```

где `N` – число процессов, `host1 ... hostN` – имена узлов, а `program` и `args` – запускаемая программа и ее аргументы соответственно.

ScaMPI (Scali MPI Connect)

Этот пакет разработан компанией Scali (`http://www.scali.no`) и поддерживает Ethernet, Myrinet, SCI и InfiniBand. Пакет является коммерческим. Процесс установки полностью автоматизирован, администратору надо лишь ответить на несколько вопросов в процессе установки. Это единственное хорошо работающее решение для сети SCI.

IBGOLD/OpenFabrics

Пакет IBGold был изначально разработан компанией Mellanox и распространялся бесплатно. В последствии развитие и поддержка IBGold была прекращена, а все исходные тексты переданы группе разработчиков OpenFabrics. В данный момент этой группой разработан пакет `OFED`, который включает в себя базовые драйверы к картам InfiniBand, обширный набор библиотек, open subnet manager и реализацию MPI на базе mvapich.

Установка пакета автоматизирована, но только для одного узла. Чтобы установить `OFED` на все узлы, следует запустить скрипт `install.sh` и выбрать пункт «Build OFED Software RPMs». Затем все пакеты из каталога `RPMS`, не содержащие в имени суффикса `-devel`, кроме пакета `opensm`, необходимо установить на всех узлах кластера. Не исключено, что драйверы InfiniBand из пакета `kernel-ib` будут конфликтовать с уже установленными драйверами штатного ядра. В этом случае требуется форсировать установку, указав команде `rpm` ключ `-replacefiles`. На одном из узлов необходимо установить пакет `opensm`. На управляющем

узле нужно установить все пакеты библиотек (`lib*.rpm`), пакет `mpich_mlx` и все пакеты с суффиксом `-devel`. Затем достаточно перезагрузить узлы или запустить сервисы `openibd` на всех узлах и `opensm` на том узле, где он установлен, после чего кластер должен быть готов к работе.