Министерство образования и науки Российской Федерации
Федеральное агентство по образованию

Федеральное государственное образовательное учреждение
высшего профессионального образования
«ЮЖНЫЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

В.Н. Дацюк, О.В. Дацюк

Система управления заданиями на вычислительных кластерах

Ростов-на-Дону 2011

Общая характеристика системы PBS

При запуске задания на вычислительных узлах кластера необходимо, указывать список узлов, на которых будет выполняться задание. Этот список формируется либо непосредственно в командной строке запускающей команды, либо в командной строке указывается имя файла, содержащего список узлов. Такой запуск задания называется прямым запуском. Прямой запуск заданий имеет множество недостатков. Во-первых, он не позволяет буферизовать задания, во-вторых,в таком подходе на пользователя возлагается обязанность определять свободные в данный момент узлы и из этого набора выделять узлы для запуска своего задания. Но даже, если пользователь определил каким-то образом список свободных на данный момент узлов, то может оказаться, что пока он готовил свой файл со списком, ситуация на кластере уже изменилась. Зачастую, это приводит к тому, что какие-то узлы оказываются перегруженными процессами, а какие-то в это время простаивают. Очевидно, что наиболее разумным подходом является автоматизация выделения узлов каждому заданию. Как правило, эта функция возлагается на диспетчерские системы.

Для управления заданиями на высокопроизводительных вычислительных системах используется различные диспетчерские системы, назначение которых предоставить вычислительные ресурсы для задачи и осуществлять контроль над процессом выполнения задания. Как правило, все диспетчерские системы построены таким образом, что они устанавливают признак «занято» для тех узлов, на которых уже выполняется какое-то задание и, если для вновь поступившего задания нет свободных ресурсов, то оно буферизуется и ставится в очередь. Кроме того, диспетчерские системы позволяют проводить некоторую политику лимитов. К таким лимитам относятся – количество одновременно запущенных одним пользователем заданий; максимальное количество узлов, которое может одновременно захватить один пользователь; максимальное время решения задания и т. д. Наиболее широкое распространение на сегодняшний день получили различные реализации системы PBS (Portable Batch System). Имеются как коммерческие реализации, например, Altair® PBS Professional™ 7.0 , так и Open Source продукты – OpenPBS, Torque.

Основные характеристики системы:

- система может обслуживать множество потоков очередей, разделенных как по архитектуре вычислительных узлов, так и по требуемым для задачи ресурсам (времени решения задачи, оперативной памяти и т.д.);
- позволяет выделять необходимые для задачи ресурсы (время решения, требуемую память и т.д.);
- устанавливает предельные лимиты на ресурсы;
- задает лимиты по умолчанию;
- ведет учет выполненных заданий.

PBS состоит из четырех основных модулей, каждый из которых может устанавливаться на одном или нескольких вычислительных узлах, обслуживаемых системой :

- одного или нескольких серверов заданий;
- одного или нескольких планировщиков заданий;
- исполнительных серверов - по одному на каждый вычислительный узел;

- набора команд администратора для управления системой и набора команд пользователя для управления своими заданиями.

Конфигурация PBS на кластерах ЦКП ВВ ЮФУ

В настоящее время системой PBS в суперкомпьютерном центре ЮФУ обслуживаются следующие вычислительные ресурсы:

- **TP** — Linux-кластер, состоящий из 16 вычислительных узлов, соединенных скоростной коммуникационной сетью DDR Infiniband (скорость передачи данных около 1400 Мб/сек, латентность 3.1 мксек). Каждый вычислительный узел представляет собой компьютер с двумя 4-х ядерными процессорами Intel Xeon E5345 2.33GHz и оперативной памятью 16Гбайт. Компилиция и отладка программ выполняется на управляющем компьютере rsufs.
- **IBMX** — Linux-кластер, состоящий из 13 вычислительных узлов, соединенных скоростной коммуникационной сетью DDR Infiniband . Каждый вычислительный узел представляет собой компьютер с одним 2-х ядерных процессором Intel Xeon 5160 с тактовой частотой 3.0 Ггц и оперативной памятью 8Гбайт. Производительность каждого вычислительного узла на тесте Linpack составляет 21 Gflops, а всего кластера в целом 252 Gflops.
- **WSD** - кластер из 13-ти рабочих станций DELL с двух ядерными процессорами Intel Core 2 Duo E6750 @ 2.66GHz, оперативной памятью 4 Гб и коммуникационной сетью Gigabit Ethernet. Не рекомендуется запускать параллельные программы с интенсивным обменом данных.

В соответствии с этим создано три очереди, по одной для каждой архитектуры с именами TP, IBMX, WSD. Внутри каждой из очередей дополнительного разбиения (например, по времени решения задачи) не сделано. Используется устанавливаемый по умолчанию планировщик FIFO (первый вошел первый вышел), сконфигурированный для эксклюзивного выполнения одного счетного процесса на каждом из узлов. PBS автоматически распределяет задания по свободным узлам заданной архитектуры.

Каждую из программ, запускаемую на кластере можно отнести к одному из четырех типов:

1. Обычная однопроцессорная последовательная программа занимает один узел и задействует одно ядро, и ни каким образом не может использовать дополнительные ядра.
2. Параллельная многонитевая OpenMP программа. Занимает один узел и задействует несколько ядер. По умолчанию захватывает все ядра узла. Не всегда использование программой всех ядер в узле позволяет получить максимальную производительность. Регулируется количество ядер занимаемых программой переменной окружения OMP_NUM_THREADS. Она может быть задана либо в конфигурационном файле пользователя .bashrc или в запускающем скрипте.
3. Параллельная многоузловая MPI программа. Захватывает несколько узлов, в каждом из которых может быть задействовано либо одно, либо несколько ядер (если на узле запускается несколько MPI процессов. Например, можно заказать 2 узла, но командой mpirun запустить 4 процесса).
4. Гибридная многоузловая многонитевая MPI+OpenMP программа. Захватывает несколько узлов, в каждом из которых может быть задействовано несколько ядер (путем выполнения многонитевого процесса). Для таких программ только

эмпирическим путем можно установить оптимальное число нитей на узле.

Работа с диспетчерской системой

Запуск исполнимой программы выполняется специальной командой **qsub,** с помощью запускающего скрипта (командного файла), в котором указываются требуемые задаче ресурсы (архитектура узлов, число процессоров, время решения) и команда запуска программы.

qsub -q <ARCH> <script_name>

где <ARCH> - имя очереди, в которую ставится задание (возможные значения IBMX, TP, WSD)

< script_name> - имя запускающего скрипта, который может быть создан любым текстовым редактором.

На самом деле команда qsub имеет множество опций, но практически все из них могут быть помещены внутрь запускающего скрипта. В том числе и опция -q.

Тогда команда запуска еще более упрощается:

qsub <script_name>

В простейшем случае для запуска однопроцессорной программы на кластере WSD нужно сформировать файл (например с именем wsd1) следующего содержания:

```
#!/bin/sh
#PBS -l nodes=1:WSD
#PBS -q WSD
cd $PBS_O_WORKDIR
./progname
```

В данном случае будет сформирована однопроцессорная задача для выполнения на кластере WSD программы с именем **progname** и с заказом времени по умолчанию 1 час. Конструкции « #PBS » распознаются командой qsub и устанавливают лимиты для задачи. Командой **cd** указывается путь к исполнимой программе. В данном случае подразумевается что запуск задания производится из рабочего каталога, в котором находится пользователь. Тогда запуск программы **progname** должен быть произведен командой:

qsub wsd1

Простая многонитевая программа запускается на одном узле и ее запуск ни чем не отличается от запуска обычной однопроцессорной программы ни в плане запускающего скрипта, ни в плане команды запуска. Разумеется, многонитевые программы имеет смысл запускать на кластерах, имеющих многоядерные процессоры (TP, IBMX, WSD).

qsub <script_name>

По умолчанию многонитевая программа порождает столько нитей, сколько имеется ядер на узле. В том случае, если требуется управлять количеством порождаемых нитей, то в скрипт добавляется строка:

```
#!/bin/sh
#PBS -l nodes=1:WSD
#PBS -q WSD
#PBS -v OMP_NUM_THREADS=2
```

```
cd $PBS_O_WORKDIR
./progname
```
В данном случае будет порождено 2 нити, которые задействуют 2 ядра
В этих скриптах предельное время решения устанавливается по умолчанию и для кластеров ЦКП оно равно одному часу. Для заданий требующих для своего решения большего времени нужно указывать время выполнения задания:

```
#!/bin/sh
#PBS -l walltime=300:20:00
#PBS -l nodes=1:IBMX
#PBS -q IBMX
#PBS -v OMP_NUM_THREADS=2
cd $PBS_O_WORKDIR
./progname
```
Здесь для решения задачи заказано 300 часов и 20 мин. По истечении заказанного времени задача будет принудительно завершена. Предельный лимит времени установлен равным 336 часам (две недели). Он может быть изменен администратором. Или изменен даже для уже выполняющегося задания.

Для запуска параллельной MPI-программы на 4-х узлах кластера IBMX используется тот же самый формат команды, но содержимое скрипта должно быть другим (скрипт ib4)
```
#!/bin/sh
#PBS -l walltime=30:00:00
#PBS -l nodes=4:IBMX
#PBS -q IBMX
cd $PBS_O_WORKDIR
mpirun -np 4 progname
```
Здесь заказано время счета 30 часов на 4-х узлах кластера IBMX. Запуск выполняется командой:

```
qsub ib4
```

Наконец приведем пример скрипта для запуска гибридных программ:
```
#!/bin/sh
#PBS -l walltime=30:00:00
#PBS -l nodes=4:IBMX
#PBS -q IBMX
cd $PBS_O_WORKDIR
mpirun -np 4 progname
```

В данном случае, если программа с именем progname распараллелена с использованием технологии MPI по узлам и с использованием технологии OpenMP по ядрам внутри узла, то на каждом из 4-х узлов будет запущен один многонитевый процесс (по количеству ядер в узле). Такой режим работы программы является наиболее естественным для кластеров с многоядерными процессорами. Если требуется ограничиться одной нитью на каждом узле, то следует добавить строку:
```
#PBS -v OMP_NUM_THREADS=1
```
К сожалению, такая конструкция работает только при использовании OpenMPI. При

использовании MVAPICH (основная коммуникационная библиотека по умолчанию) значения переменных не передаются на удаленные узлы и эта инструкция сработает только на первом узле из выделенных системой. В этом случае регулировать количество нитей можно только задавая переменную OMP_NUM_THREADS в конфигурационном файле пользователя .bashrc:
export OMP_NUM_THREADS=1

При запуске программы через команду **qsub** заданию присваивается уникальный целочисленный идентификатор, который представляет собой номер запущенного задания. По этому номеру можно отслеживать прохождение задания, снимать задание со счета или из очереди, перемещать его в очереди относительно других своих заданий.

Результат работы программы будет записан в файл ***по окончании выполнения*** программы и помещен в тот каталог, из которого было запущено задание. Имя выходного файла формируется автоматически следующим образом:
`<имя скрипта>.o<номер задания>`, а в файл
`<имя скрипта>.e<номер задания>` - будет записываться стандартный канал диагностики (ошибок).
Имя выходного файла можно изменить с помощью специальной опции команды **qsub**.

Если требуется просмотр результатов по ходу выполнения задания, то можно использовать механизм перенаправления вывода в команде запуска программы. Причем, для того, чтобы выходной файл каждого задания имел уникальное имя можно использовать внутренние переменные PBS.
Пример:

```
#!/bin/sh
#PBS -l walltime=30:00:00
#PBS -l nodes=4:IBMX
#PBS -q IBMX
cd $PBS_O_WORKDIR
mpirun -np 4 progname > $PBS_JOBID
```

В данном случае результаты будут записываться в файл, имя которого сформируется из идентификатора задания. Этот файл можно будет просматривать в процессе выполнения программы, либо с помощью команд:

```
cat имя_файла ,
less имя_файла
```

либо интерактивно по мере заполнения файла:

```
tail -f имя_файла .
```

Замечания:

а) программа не должна быть интерактивной, т.е. содержать ввод с клавиатуры;
б) в случае, если у системы возникают проблемы с поиском пути, куда должны быть записаны выходные файлы, то они остаются в системном каталоге /var/spool/PBS/undelivered/ на том узле кластера, где непосредственно решалась задача;
в) в файл диагностики выдаются сообщения об ошибках.

Если бинарная программа находится непосредственно в рабочем каталоге, то для генерации PBS скрипта можно воспользоваться специальной командой **qpbs ,** которая в диалоговом режиме сформирует PBS скрипт и запишет его с указанным именем.

Команды управления заданиями

Помимо описанной кратко команды **qsub**, имеется набор команд для управления заданиями. Подробное их описание можно посмотреть с помощью команды **man** :
qdel - удаление задания ;
qhold - поставить запрет на исполнение задания ;
qmove - переместить задание;
qmsg - послать сообщение заданию;
qrls - убрать запрет на исполнение, установленный командой qhold;
qselect - выборка заданий;
qsig - посылка сигнала (в смысле ОС UNIX) заданию;
qstat - выдача состояния очередей (наиболее полезны команды qstat -a и qstat -q);
qsub - постановка задания в очередь ;
pestat - выдача состояния всех вычислительных узлов (нет описания) ;
xpbs - графический интерфейс для работы с системой PBS (требуется X-сервер) ;
xpbsmon - графическая программа выдачи состояния вычислительных ресурсов.

Запуск программ из стандартных пакетов

Запуск через диспетчерскую систему программ из стандартных пакетов, таких как ANSYS имеет свою специфику. Дело в том, что они имеет собственный механизм запуска, управляемый через командную строку. Поэтому для них нет другого способа, как писать специальные запускающие скрипты, которые «на лету» формируют запускающий скрипт для команды qsub, внутри которого будет сформирована правильная команда запуска программы. Такие скрипты написаны для запуска программ ansys и cfx5solve.
Для запуска программы ansys110 следует использовать команду:
qansys JOB QUE NODES CORES Tlimit
Здесь
JOB — имя основного файла данных без расширения (подразумевается, что он имеет расширение .dat);
QUE — имя очереди, (по умолчанию TP)
NODES — количество узлов (по умолчанию 1)
CORES — количество ядер на каждом узле (по умолчанию 1)
Tlimit — требуемое для задания время (по умолчанию 336 часов).
Параметры позиционные, поэтому начиная с некоторого все остальные могут быть пропущены.
Замечание: программа ansys требует указания лицензии в командной строке. На сегодняшний день для нее есть два типа лицензий:

1. Academic Research (aa_r), их 5 штук, но они позволяют работать не более чем с двумя процессами,хоть на одном узле, хоть на двух.
2. Multiphysics (ane3fl) не имеет ограничений по количеству процессов, но она одна единственная. Под этой лицензией может выполняться одновременно только одно задание.

Поэтому, если произведение NODES*CORES <= 2, то задействуется лицензия aa_r, в противном случае ane3fl. Задание формируется и отправляется на счет, но если оказывается, что лицензия занята, то оно вылетает по ошибке.

Аналогичный синтаксис имеет команда для запуска программы cfx5solve

qcfx JOB QUE NODES CORES Tlimit

Здесь

JOB — имя основного (definition) файла данных без расширения (подразумевается, что он имеет расширение .def);

QUE — имя очереди (по умолчанию TP)

NODES — количество узлов (по умолчанию 1)

CORES — количество ядер на каждом узле (по умолчанию 1)

Tlimit — требуемое для задания время (по умолчанию 336 часов).

Для этой команды с жесткими лицензионными ограничениями не сталкивались.

Обе команды должны запускаться из каталога, содержащего файлы данных.

Кроме этого, написано множество команд для запуска других стандартных пакетов.

1. Квантово-химические пакеты

Gaussian03:

```
qgauss JOB QUE NODES CORES TLim
        JOB   - имя com-файла без расширения
        QUE   - IBMX | TP | WSD
        NODES – число узлов (по умолчанию 1 )
        CORES – число ядер на узле (по умолчанию 1 )
        TLim  - лимит времени (по умолчанию 335 часов )
```

GAMESS:

```
qgamess JOB QUE NODES TLim Mail
        JOB   - имя inp-файла без расширения
        QUE   - IBMX | TP | WSD
        NODES - число узлов (по умолчанию 1 )
        TLim  - лимит времени (по умолчанию 336 часов )
        Mail  - почтовый адрес (default $USER )
```

ORCA:

```
qorca3 JOB QUE NCPUS TLim
        JOB   - имя файла с заданием
        QUE   - IBMX | TP | WSD
        NCPUS - число узлов (по умолчанию 1 )
        TLim  - лимит времени ( по умолчанию 336 часов )
```

OPENMX:

```
qopenmx JOB QUE NODES CORES TLim
        JOB   -  имя dat-файла без расширения
        QUE   - IBMX | TP | WSD
        NODES - число узлов (по умолчанию 1 )
```

CORES - число ядер на узле (по умолчанию 1)
TLim - лимит времени (по умолчанию 336 часов)

Firefly (PC-GAMESS):

```
qfirefly JOB QUE NODES TLim Mail
```
JOB - имя inp-файла без расширения
QUE - IBMX | TP | WSD
NODES - число узлов (по умолчанию 1)
TLim - лимит времени (по умолчанию 336 часов)
Mail - почтовый адрес (по умолчанию $USER)

2. Квантово-физические пакеты

FDMNES:

```
qfdm15 JOB QUE NCPUS TLim Mail
```
JOB - имя задания (произвольное)
QUE - IBMX | TP | WSD
NCPUS - число узлов (по умолчанию 1)
TLim - лимит времени (по умолчанию 336 часов)
Mail - почтовый адрес (по умолчанию $USER)

FEFF84:

```
qfeff JOB QUE TLim Mail
```
JOB - имя задания (произвольное)
QUE - IBMX | TP | WSD
TLim - лимит времени (по умолчанию 336 часов)
Mail - почтовый адрес (по умолчанию $USER)

SPRKKR:

```
qkkr JOB QUE NCPUS TLim
```
JOB - имя inp-файла без расширения
QUE - IBMX | INFINI | WSD | LINUX
NCPUS - число узлов (по умолчанию 1)
TLim - лимит времени (по умолчанию 336 часов)

VASP:

```
qvasp5 JOB QUE NCPUS TLim Mail
```
JOB - имя задания (произвольное)
QUE - IBMX | TP | WSD
NCPUS - число узлов (по умолчанию 1)

TLim - лимит времени (по умолчанию 336 часов)
Mail - почтовый адрес (по умолчанию $USERS)

IBINIT:

```
qabinit EXE JOB QUE NODES TLim Mail
```
EXE - имя программы для выполнения (default abinit)
JOB - имя входного файла без расширения
QUE - queue for execution: IBMX | TP | WSD
NCPUS - число узлов (по умолчанию 1)
TLim - лимит времени (по умолчанию 336 часов)
Mail - почтовый адрес (по умолчанию $USERS)

QUANTUM ESPRESSO:

```
qesp EXE JOB QUE NODES TLim Mail
```
EXE - имя программы для выполнения (default pw.x)
JOB - имя входного файла без расширения
QUE - queue for execution: IBMX | TP | WSD
NODES - число узлов (по умолчанию 1)
NCORE - число процессов на узле (по умолчанию 1)
NB - number of band group
NT - number of task group
ND - number of processors for linear algebra
TLim - лимит времени (по умолчанию 336 часов)
Mail - почтовый адрес (по умолчанию $USERS

YAMBO:

```
qyambo EXE JOB QUE NODES NCORE TLim Mail
```
EXE - имя программы для выполнения (default yambo)
JOB - имя задания (произвольное)
QUE - queue for execution: IBMX | TP | WSD
NODES - число узлов (по умолчанию 1)
CORES - число ядер в процессоре (по умолчанию 1)
TLim - лимит времени (по умолчанию 336 часов)
Mail - почтовый адрес (по умолчанию $USERS)

С вопросами по работе с диспетчерской системы обращаться по телефону: 219-97-13
email: dvn@sfedu.ru