Приложение 4.
Использование системы управления заданиями Cleo

Система управления заданиями Cleo контролирует запуск задач на многопроцессорных вычислительных установках, в том числе на кластерах. Она позволяет автоматически распределять вычислительные ресурсы между задачами, управлять порядком их запуска, временем работы, получать информацию о состоянии очередей. При невозможности немедленного запуска задач, они ставятся в очередь и ожидают, пока не освободятся нужные ресурсы.

Система позволяет работать с различными типами заданий, в том числе с последовательными и написанными с использованием различных версий MPI: MPICH, MPICH-gm, LAM, ScaMPI, MVAPICH и другими.

В качестве вычислительной единицы используется процессор. Система оперирует и понятием вычислительного узла, который может объединять несколько процессоров. В рамках одной очереди система считает все процессоры равноправными.

Сама система Cleo написана на языке perl, что позволяет использовать ее на различных платформах.

Общая структура системы

Система Cleo состоит из сервера (запускаемого с правами суперпользователя), управляющего заданиями, программ-мониторов, запускаемых на всех рабочих узлах, и набора клиентских программ, осуществляющих взаимодействие с сервером по TCP/IP либо через файлы состояния сервера.

В качестве клиентских программ в поставку входит так называемый основной клиент (программа `cleo-client`), который поддерживает полный набор команд сервера Cleo. Его используют

скриптовые программы, осуществляющие простые операции с сервером, например, постановку задания в очередь, снятие задания и другие.

Возможно написание произвольных клиентов, использующих протокол взаимодействия сервера и клиентских программ. Предусмотрена возможность анализировать файл состояния сервера для получения данных об очередях, задачах и т.п.

Иерархия очередей

Система очередей способна поддерживать несколько очередей одновременно. Все они организованы иерархически, когда одни очереди включают в себя другие. Это значит, что любой процессор может использоваться одновременно несколькими очередями.

Рассмотрим следующий пример:

<table>
<tr><td colspan="8">main</td></tr>
<tr><td colspan="4">Long</td><td colspan="4">short</td></tr>
<tr><td>p1:1</td><td>p1:2</td><td>p2:1</td><td>p2:2</td><td>p3:1</td><td>p3:2</td><td>p4:1</td><td>p4:2</td></tr>
</table>

Представлена иерархия из трех очередей над 4 вычислительными узлами, содержащими по 2 процессора каждый. Узлы имеют имена p1, p2, p3 и p4, а их процессоры соответственно p1:1, p1:2 для узла p1 и т.д.

Очередь main включает в себя все процессоры. Такая объемлющая очередь должна присутствовать всегда, даже если ей никогда не будут пользоваться (это имеет смысл, например, если она объединяет несколько несвязанных кластеров или разделов одного кластера, управление которыми осуществляется независимо).

В данном примере очередь main имеет две дочерних очереди – long, включающую в себя процессоры p1:1, p1:2, p2:1 и p2:2, и очередь short, включающую в себя процессоры p3:1, p3:2, p4:1 и p4:2.

Дочерние очереди могут пересекаться, но в этом случае информация о занятости процессоров из пересечения не передается между очередями одного уровня, поэтому такой режим нежелателен.

Система следит за тем, чтобы каждый процессор использовался одновременно только одной задачей (если явно не оговорен иной режим). Это достигается за счет того, что задачи очередей верхнего уровня автоматически попадают и в дочерние очереди. Таким образом, когда задача фактически идет на счет, она присутствует и помечается как считающаяся во всех очередях, чьи процессоры она занимает. Например, поставим в очередь short задачу на 4 процессора, а затем в очередь main задачу на 6 процессоров. Последняя задача автоматически попадет в очереди short и long. В очереди long она будет помечена как предзапущенная, то есть для нее будут зарезервированы процессоры, но на счет она отправлена не будет. В очереди short будет считаться первая задача, а вторая будет ждать окончания ее счета.

Как только первая задача досчитается, в очереди short будет предзапущена вторая задача. Очередь main (которой и принадлежит вторая задача) получит 8 процессоров для ее запуска. Так как заказано для нее было только 6 процессоров, то ровно 6 процессоров будет отобрано для счета. Остальные 2 процессора будут сняты с резервирования и могут быть использованы для запуска новых задач.

Описанную выше ситуацию можно условно представить в таком виде:

До постановки задач

main	
long	**short**

После постановки задач

Main	
задача2	
long	**short**
задача2	*задача1*
	задача2

Каждая из очередей может быть настроена независимо. Иными словами, в различных очередях могут быть заданы различные политики для пользователей, лимиты, стратегии планирования.

В качестве лимитов могут быть заданы количество процессоров на одну задачу, количество одновременно занятых процессоров (если пользователь одновременно запускает несколько задач), максимальное время счета задачи, максимальный приоритет задачи и другие.

Ограничения могут также задаваться и для отдельных пользователей, при этом такие ограничения могут быть как строже, так и мягче, чем для очереди в целом.

Приоритеты

В очереди задачи сортируются по приоритету. Приоритет – это целое число в диапазоне от 0 до 256. Приоритет по умолчанию задается в настройках очереди. Если он не задан, то он принимается равным 10. Задачи с более высоким приоритетом всегда ставятся в очередь выше задач с более низким приоритетом и будут запущены раньше, даже если они были поставлены в очередь позже по времени. Приоритет можно повышать и понижать после постановки задачи в очередь.

Пользователь может понизить приоритет своей задачи, а также повысить его до величины, не превышающей установленного для него лимита.

Ограничение времени счета

Для задач можно задать ограничение времени счета. Обычно оно задано по умолчанию, но можно его понизить, если это необходимо. По истечении указанного лимита задача будет принудительно снята со счета.

Система Cleo ориентируется на то, что задача будет считаться не дольше указанного лимита времени, и учитывает это при планировании времени старта других задач.

Стратегии выбора процессоров

При старте задачи для нее выделяются процессоры. В системе есть возможность управлять стратегией выбора процессоров для задач. Есть три встроенные стратегии:

`random`	Процессоры выбираются случайным образом.
`random_hosts`	Процессоры выбираются на случайно выбранных узлах, предпочтительно занимая все доступные процессоры на узле.
`hosts_alone`	Процессоры выбираются на случайно выбранных узлах, предпочтительно занимая лишь по одному процессору на узле.

Элемент случайности в выборе процессоров присутствует для достижения двух целей: лучшей сбалансированности нагрузки на процессоры и предотвращения блокировок, связанных с неудачным заказом конфигурации процессоров.

В системе есть возможность подключения внешних модулей для задания своих стратегий. В случае использования внешнего модуля, он должен быть описан в разделе `pe_sel_method` секции [`server`], где также будет описано его имя. Для подключения модуля в систему достаточно просто указать его имя вместо имени встроенного метода.

Политики пользователей

Как для всех очередей, так и для каждой очереди в отдельности можно задать список пользователей, которые имеют право ставить на счет свои задачи. Все остальные пользователи не будут иметь доступа к соответствующим очередям. Можно запретить доступ к очередям фиксированному списку пользователей, тогда для всех остальных доступ будет открыт.

Права доступа не наследуются очередями-потомками. В контексте нашего примера, если пользователю user1 разрешен доступ в очередь main, то ему не обязательно разрешен доступ в очередь short или long.

Для всех очередей и для каждой очереди в отдельности можно задать список пользователей-администраторов, которые в рамках соответствующих очередей получают возможность управлять лимитом времени работы задач, приоритетами, удалять любые задачи и выполнять другие административные действия.

Блокировки

Системой Cleo поддерживаются следующие виды блокировок:

- блокировка очереди на запуск задач;
- блокировка очереди на постановку новых задач;
- блокировка процессоров;
- отложенная блокировка процессоров;
- блокировка задач;
- автоблокировка пользователей;
- автоблокировка по времени;
- автоблокировка по ограничению ресурсов.

Блокировка очереди на запуск задач означает, что задачи, стоящие в очереди, не будут запущены на счет, пока блокировка не снята. При этом уже запущенные задачи со счета не снимаются. Любая очередь может быть в любой момент времени заблокирована на добавление новых задач. На работающих и уже стоящих в очереди задачах это никак не отражается.

Указанные блокировки могут быть установлены рекурсивно, то есть не только на конкретную очередь, но и на все ее дочерние очереди.

Блокировкой процессора можно запретить исполнение задач на конкретном процессоре или узле. Отложенная блокировка процессора активируется только после того, как процессор заканчивает выполнение пользовательской задачи. Когда это происходит, администратору высылается уведомление.

Автоматические блокировки срабатывают при наступлении определенных условий. Автоблокировка новых задач пользователя начинает действовать при постановке задачи в очередь. Автоблокировка по времени срабатывает в том случае, если задача не успевает завершиться к указанному времени. Автоблокировка по ресурсам вступает в силу, если запуск задачи пользователя нарушает наложенное ограничение, например, по числу одновременно занимаемых процессоров.

Схема запуска задач

Для запуска задач в системе Cleo предусмотрено несколько режимов, но предпочтительным является следующий. Задача запускается в псевдотерминале (используется пакет `empty-cleo`[1]) обычным для выбранной среды способом. Например, для MPICH или LAM будет запущена программа `mpirun` из соответствующего дистрибутива. При запуске программе передаются данные о процессорах, на которых надо запускаться.

На соответствующие узлы передается команда, по которой все новые процессы указанного пользователя запоминаются. В дальнейшем все процессы, порожденные от этих процессов, также запоминаются. При

[1] Основан на пакете empty, автор Михаил Захаров (zmey20000@yahoo.com)

завершении задачи все запомненные процессы принудительно завершаются через заданный промежуток времени.

Во время работы задачи можно подключиться к псевдотерминалу командой `cleo-empty` и выполнить любые интерактивные действия.

Запущенные задачи не зависят от сервера Cleo, т.к. работают в собственных псевдотерминалах. Это значит, что сервер может быть остановлен или перезапущен во время работы задач.

Возможности дополнения и расширения

Статус очередей можно получать не только посредством команд серверу, но и из файлов состояния. Именно так работает пакет `qs-web`, позволяющий представлять состояние очередей или задач в любом удобном виде, например, как web-страницу или в текстовом виде.

Система сделана расширяемой за счет использования модулей. В данный момент реализованы следующие типы модулей:

- модули стратегий распределения процессоров;
- модули, вызываемые при запуске или завершении задачи;
- планировщики задач.

Постановка заданий в очередь и удаление задач

Запуск задач осуществляется командой `mpirun`:

```
mpirun -np N [-q queue][-maxtime|-l lim][-p pri][-stdin
                        file]
    [-stdout file] [-stderr file] prog [prog_args]  ,
```

либо той же командой в таком варианте:

```
mpirun -np N [-q queue][-maxtime|-l lim][-p pri][-stdin
                        file]
    [-stdout file] [-stderr file] -f batch_file   .
```

Ключи, указанные в скобках, являются необязательными, а `prog` и `prog_args` – это исполняемый файл задачи и ее аргументы соответственно. Второй вариант запуска предполагает наличие файла `batch_file`, в котором перечислены программы с аргументами. В данном случае происходит последовательный запуск перечисленных программ в рамках одной задачи.

Ниже описаны значения ключей `mpirun`.

`-np`	– количество процессоров.
`-q`	– название очереди, в которую ставится задача.
`-maxtime`	– лимит времени счета в минутах.
`-l`	– лимит времени счета в секундах.
`-p`	– приоритет задачи в очереди.
`-stdin`	– файл для стандартного ввода.
`-stdout`	– файл для стандартного вывода.
`-stderr`	– файл для стандартного потока ошибок.

Например, команда вида:

```
mpirun -np 15 -q users -maxtime 10 my_test -dummy -i 50
```

поставит задачу `'my_test'` с параметрами `'-dummy -i 50'` в очередь `users` с лимитом работы в `10` минут и запросом на `15` процессоров.

Следующая команда:

```
mpirun -np 15 my_test2 /tmp/my_test.tmp -n 10
```

поставит задачу `'my_test2'` с параметрами `'/tmp/my_test.tmp -n 10'` в очередь по умолчанию с запросом на `15` процессоров.

Если ключ `-maxtime` (или `-l`) не задан, то будет взято значение из переменной окружения `QS_TIMELIMIT`. Если оно пустое, или данная переменная не определена, то будет использовано значение, заданное в пользовательском файле `~/.qconf` параметром `def_time`. Если в

пользовательском файле этого параметра нет, то он будет взят из глобального файла конфигурации.

Указывая ключ `-maxtime`, пользователь может ускорить запуск своей задачи, так как дает системе возможность планировать время счета этой задачи и, как следствие, возможность выполнить ее “досрочно”, если будет такая возможность.

При постановке задачи в очередь запоминаются все переменные окружения, во время запуска они будут установлены в эти значения.

После запуска задачи ее стандартный вывод будет перенаправлен в файл, имя которого определяется настройками по умолчанию, либо индивидуальными настройками пользователя, либо ключом `-stdout`. Обычно этот файл создается в том же каталоге, откуда задача была запущена, и имеет имя `<имя_задачи>.out-<номер>`, где `номер` – это номер задачи в очереди.

Удалить задачу из очереди или досрочно снять задачу со счета можно командой `tasks` с ключом `-d`. Например:

```
/home/usr2> tasks -d 2141
```

удалит задачу с номером `2141` из очереди.

По окончании работы задачи создается файл отчета (его имя обычно аналогично имени файла вывода задачи, но суффикс `out` меняется на `rep`), в котором указываются полное имя задачи, аргументы, время счета, имя файла вывода, код возврата и, если задача завершилась аварийно, причина останова.

Во время работы задачи на узлах специально для нее создается временный каталог, который будет очищен по окончании работы. В этом каталоге целесообразно создавать временные рабочие файлы. Узнать полный путь к нему можно во время работы программы из переменной окружения `TEMP_DIR`.

Просмотр состояния очереди

Посмотреть состояние очереди можно командой `tasks`:

```
tasks [-q queue][-r][-l][-t][-f][-o][-m mask][-u userlist]
                    [-b][-d id ... id|-v]
```

Указанные в скобках ключи являются необязательными, а ниже приведено их краткое описание.

`-q`	– название очереди,
`-r`	– показывать очереди рекурсивно,
`-l`	– показывать дополнительную информацию,
`-t`	– показывать лимит времени по умолчанию для пользователя,
`-f`	– учитывать чужие задачи,
`-o`	– учитывать свои задачи,
`-m mask`	– использовать маску для выборки задач,
`-u list`	– использовать список пользователей для выборки задач,
`-b`	– показывать информацию о заблокированных узлах,
`-v`	– показывать состояние очереди (по умолчанию),
`-d`	– удаление задачи.

Параметры `-f, -o` указывают выборку задач. По умолчанию в выборку попадают все задачи, если не указано иначе в файле конфигурации.

Более полная информация по системе управления заданиями Cleo, включая дистрибутив и документацию, доступна на следующей странице: `http://parcon.parallel.ru/cleo.html`.