УДК 519.6

СИСТЕМА АНАЛИЗА ПРОИЗВОДИТЕЛЬНОСТИ ПАРАЛЛЕЛЬНЫХ ПРОГРАММ НА КЛАСТЕРНЫХ УСТАНОВКАХ

С. А. Жуматий[1]

В статье описывается система анализа эффективности параллельных программ. Основное назначение системы — облегчить адаптацию параллельных программ к современным кластерным установкам, ускорить процесс поиска узких мест параллельной программы.

Ключевые слова: вычислительные кластеры, эффективность программ, анализ программ, управление вычислительными ресурсами.

1. Введение. Кластерные вычислительные установки приобретают все бóльшую популярность. Многие вычислительные задачи, счет которых ранее занимал недели и месяцы, а то и вовсе был невозможен из-за малых ресурсов персональных компьютеров, теперь выполняется на вычислительных кластерах за приемлемое время.

Однако, запуская задачу на кластере, пользователь нередко получает совсем не ту скорость работы, которую ожидал. В чем причина низкой эффективности параллельной программы? Ответов на этот вопрос может быть много (см. рис. 1) — не хватает оперативной памяти, малая производительность коммуникационной сети, неудачное распараллеливание и др. Но что именно послужило причиной неэффективной работы конкретной программы? Оптимизировать программу “вслепую” очень сложно; этот процесс может занять длительное время.

Рис. 1. Причины снижения эффективности программ на кластерных системах

На основе опыта эксплуатации суперкомпьютерного комплекса НИВЦ МГУ была разработана методика анализа эффективности функционирования программно-аппаратных сред высокопроизводительных вычислительных систем с кластерной архитектурой. На базе предложенной методики реализован программный комплекс ParCon, призванный облегчить работу пользователей и администраторов кластеров. Данный комплекс позволяет определять узкие места в конфигурации программно-аппаратных сред кластерных систем как в процессе выполнения, так и после завершения работы параллельных программ.

Программный комплекс ParCon состоит из двух компонент — системы управления прохождением заданий Cleo и системы мониторинга Antmon. Основываясь на данных мониторинга и информации о программе, полученной от системы Cleo, ParCon предоставляет пользователю отчет о работе программы. На рис. 2 представлена общая схема работы ParCon.

Имея данные, предоставленные ParCon, пользователь может анализировать профиль выполнения своей задачи. Большинство проблем будет сразу видно по данным загрузки процессора, сети, использо-

[1] Научно-исследовательский вычислительный центр, Московский государственный университет им. М. В. Ломоносова, 119992, Москва; e-mail: serg@parallel.ru

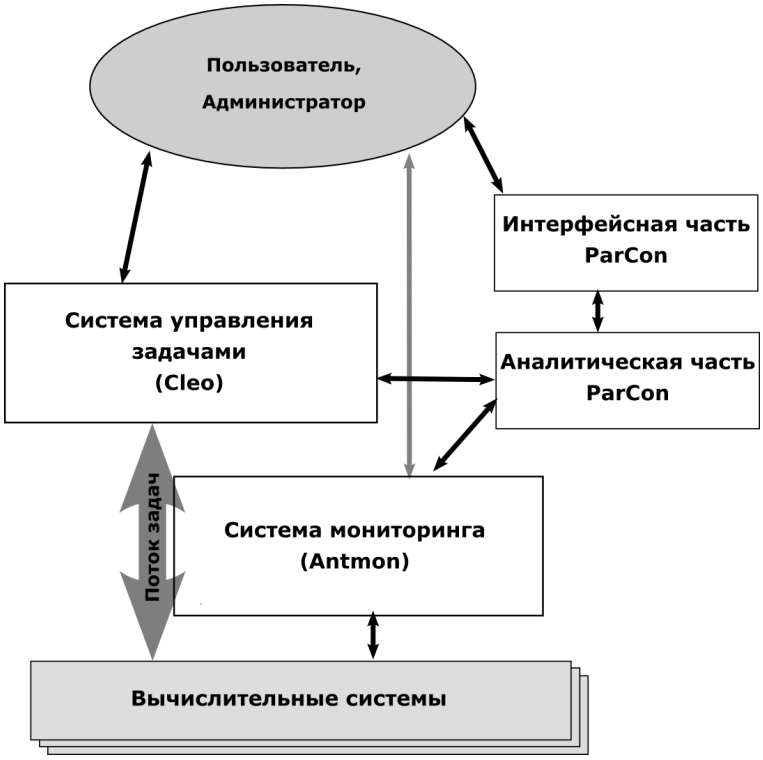

Рис. 2. Архитектура комплекса ParCon

вания оперативной памяти и другим данным. Такая система не позволяет исследовать тонкую структуру программы, но помогает выбрать правильный путь ее оптимизации.

В процессе разработки системы ParCon учитывался целый ряд требований, которым должна удовлетворять подобная система. Крайне важно, чтобы сбор данных мониторинга не оказывал серьезного влияния на работу пользовательских программ. Если процесс мониторинга будет потреблять слишком много ресурсов, то вреда для пользователя будет куда больше, чем пользы. Поэтому использование процессорного времени системой мониторинга на вычислительных узлах не должно превышать 3–5 %, а объем передачи вспомогательных данных по сети — 1 %.

Весь комплекс должен быть масштабируемым, обеспечивать возможность работы с большинством современных кластеров. Сейчас в России только несколько уникальных кластеров насчитывают более 500 процессоров. Поэтому достаточным требованием к масштабируемости комплекса будем считать стабильную работу на сотнях процессоров (до 500). Это относится и к системе мониторинга, и к системе управления задачами, и к комплексу в целом.

Кроме масштабируемости, не менее важна и переносимость. Поскольку Linux распространен в качестве операционной системы при построении кластеров, очевидным требованием является поддержка этой ОС и минимальная зависимость от плохо переносимых или непереносимых программных пакетов.

В силу того, что аппаратное и программное обеспечения кластеров может быть самым различным, то еще одним важным требованием к комплексу становится расширяемость. Комплекс должен поддерживать возможность добавления новых параллельных сред, а также добавления мониторинга любых параметров кластера.

Программный комплекс ParCon реализован под ОС Linux и способен работать на любых аппаратных архитектурах, на которых работает эта операционная система. Комплекс требует для работы пакеты perl (версии не ниже 5.6.0) и rrd, свободно распространяемых для ОС Linux.

2. Техническая реализация. Как уже было сказано выше, комплекс ParCon включает в себя систему управления прохождением заданиями Cleo и систему мониторинга. Комплекс разработан для работы под ОС Linux. Остановимся подробнее на реализации компонентов комплекса.

2.1. Система управления прохождением заданий Cleo. Система управления прохождением заданиями Cleo была создана специально для кластерных установок. Она содержит средства для планирования запуска задач, управления политикой использования ресурсов кластера пользователями, а также

для контроля выполнения задач.

Система управления заданиями Cleo включает в себя:

— головную программу-демон;

— программы-демоны на вычислительных узлах;

— динамически загружаемые компоненты;

— клиентские программы, позволяющие посылать запросы программе-демону и получать информацию о состоянии системы;

— документацию и информационные материалы.

Головная программа-демон запускается на хост-компьютере кластера. На нем осуществляется работа пользователей. Именно эта программа-демон отвечает за состояние очередей, планирование запуска задач, процедуры запуска, завершения и отслеживания состояния задач. С головной программой общаются клиентские программы, посылая ей запросы на операции с задачами и очередью.

Система Cleo поддерживает разделы кластера — группы процессоров. Они могут быть вложены друг в друга. На этих разделах может быть создано несколько очередей, которые могут быть независимы или организованы в иерархию. В случае иерархии все процессы дочерних очередей принадлежат также родительским очередям. При использовании родительской очередью процессоров дочерней в последнюю ставится формальная задача, занимающая эти процессоры.

Для того чтобы действия пользователей не противоречили их правам в системе Cleo, при общении клиентской программы с головной используется механизм авторизации. Права пользователя в системе Cleo определяются в ее конфигурационном файле и могут быть изменены в процессе работы системы без ее перезапуска. Права пользователя в системе могут быть отличны от его прав на компьютере; например, рядовой пользователь может быть администратором Cleo и, наоборот, суперпользователь на компьютере может иметь ограниченные права в Cleo или даже вообще не иметь их. Права пользователей могут быть разными в разных очередях. Например, можно создать очередь для пользователей-студентов и не разрешать им ставить задачи в другие очереди.

Программы-агенты работают на вычислительных узлах и обеспечивают корректность работы параллельных программ — проверяют, все ли процессы программы запустились, завершились ли они корректно при завершении программы и т.п. Кроме того, по факту доступности программы-демона головная программа определяет, доступен ли вычислительный узел. Если узел недоступен, он автоматически блокируется, а программы, запущенные на нем, завершаются.

Динамически загружаемые компоненты позволяют изменять логику работы Cleo, не перегружая систему и не останавливая работу уже запущенных параллельных программ. В качестве такого компонента реализован планировщик задач — модуль, определяющий очередность запуска задач из очереди. Система допускает использование и нескольких планировщиков, переключаясь с одного на другой при необходимости.

Общение пользователей и администраторов с системой осуществляется через программы-клиенты, а также через файлы состояния системы. Через файлы состояния можно только получать данные о задачах и очередях, но нельзя влиять на состояние системы. С помощью программ-клиентов можно добавлять или удалять задачи из очереди, менять их приоритет, изменять состояние системы, получать расширенные данные о ее состоянии.

Схема работы Cleo представлена на рис. 3.

Для удобства пользователей реализована программа *mpirun*, которая имеет такие же ключи, как и программа mpirun, входящая в состав большинства реализаций широко распространенной библиотеки MPI. Вместо запуска задачи mpirun из поставки Cleo просто ставит эту задачу в очередь с использованием всех указанных в ключах параметров. Кроме стандартных ключей mpirun в версии этой программы в Cleo реализованы опциональные, специфичные для Cleo ключи, такие как указание приоритета, лимита времени счета, названия конкретной очереди и т.п.

Командой *tasks* пользователь может посмотреть состояние очереди. С помощью этой же команды с ключом *-d* можно удалить из очереди любую свою задачу, как ожидающую запуска, так и работающую в данный момент.

Кроме команд, доступных из командной строки, можно воспользоваться web-интерфейсом к системе [2]. Web-интерфейс использует файлы состояния для отображения текущего статуса очередей и задач. Посредством web-интерфейса нельзя добавлять или удалять задачи, но можно получить полную информацию об очередях и задачах.

Cleo включает в себя средства для управления политиками использования ресурсов кластера. Администратор может устанавливать квоты на время, которое должно быть выделено задаче пользователя, а

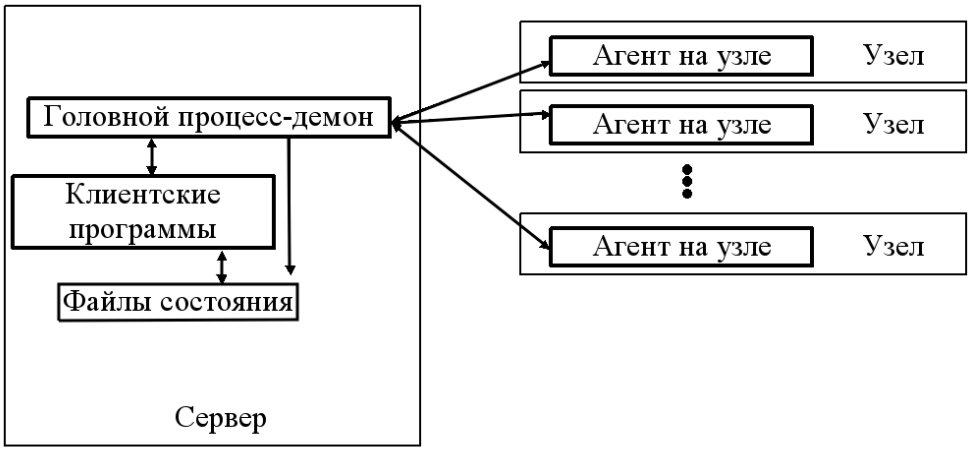

Рис. 3. Архитектура системы Cleo

также на количество процессоров, которые пользователь может использовать под задачу. Независимо от этих параметров можно установить квоту на количество одновременно используемых процессоров. Квоты устанавливаются как для всех пользователей (по умолчанию), так и для каждого в отдельности. Кроме квот на время и количество процессоров можно вводить квоты на количество одновременно запущенных задач и количество задач, стоящих в очереди.

Помимо введения квот администратор может временно приостанавливать прохождение новых задач определенных пользователей. Например, если определенной группе пользователей необходимо предоставить монопольный доступ к ресурсам кластера, то для всех других пользователей можно установить политику автоматического блокирования новых задач. Таким образом, задачи выделенной группы будут исполняться, а задачи остальных пользователей будут ставиться в очередь. После окончания работы в монопольном режиме все заблокированные задачи будут разблокированы и продолжат работу в обычном режиме.

Заблокированными могут быть не только задачи, но и вычислительные узлы, например для проведения профилактических работ. Это может быть сделано двумя способами. При первом способе узел блокируется немедленно и задачи, которые считались на нем, принудительно завершаются. При втором способе узлы блокируются только после того, как завершатся запущенные на них задачи. Каждый процессор на этих узлах блокируется немедленно после завершения задачи или сразу после выполнения команды блокировки, если он был свободен. Если узел не отвечает на запросы сервера в течении определенного времени, то он автоматически блокируется системой. Команды блокирования и разблокирования узлов доступны только администраторам системы.

В стандартную поставку системы Cleo входит базовый модуль планировщика запуска задач. Этот модуль запускает задачи по порядку приоритетов при наличии достаточного количества процессоров для запуска. Если задача не может быть запущена из-за введенных квот, она временно блокируется и могут быть запущены задачи, стоящие в очереди за ней. Перед каждым вызовом планировщика заблокированные задачи проверяются на соответствие квотам — если блокировка по квоте уже не актуальна, она снимается.

Для более полной загрузки вычислительных мощностей реализован алгоритм ускоренного запуска коротких задач — если все задачи, которые стоят в очереди перед задачей с маленьким лимитом времени работы, не могут быть запущены в данный момент времени из-за недостаточного количества свободных процессоров, то эта задача может быть запущена раньше них. Обязательное условие запуска такой задачи состоит в том, что она должна завершиться ранее, чем запланирована к запуску очередная задача, стоящая перед ней. Время запуска очередной задачи рассчитывается из лимитов времени уже запущенных задач. Благодаря такому алгоритму небольшие тестовые задачи могут быть пропущены быстрее, а вычислительные мощности кластера использованы более эффективно.

2.2. Система мониторинга Antmon. Каждый узел кластера имеет несколько параметров (сенсоров), которые должны отслеживаться. Учитывая количество узлов, общее число параметров может оказаться довольно велико. Большинство существующих систем мониторинга не очень хорошо реагируют на большое количество параметров, которые необходимо отслеживать, так как это приводит к повышен-

ной нагрузке на сеть, а также на процессоры на всех хостах и головном мониторе. В условиях работы вычислительного кластера такое поведение нежелательно.

Важный момент, который необходимо учитывать, — надежность самой системы мониторинга. Все существующие системы используют один головной сервер, и в случае возникновения проблем с ним (например, сбой сервера, обрыв связи и т.п.) администратор не получает информации о случившемся.

Созданная в рамках комплекса ParCon система мониторинга Antmon призвана решить обе упомянутые выше проблемы. Архитектура системы мониторинга разработана так, чтобы сбор сведений, которые необходимо отслеживать, проводился с минимальной нагрузкой на сеть и процессоры, а сбой головного сервера не приводил к отказу всей системы.

На всех узлах запускаются агенты, которые собирают необходимую информацию и отдают ее по требованию головным серверам. Агенты на узлах не имеют локальной конфигурации, поэтому настройка всей системы значительно упрощается — вся конфигурация осуществляется на головных серверах.

Для минимизации нагрузки на сеть был разработан свой протокол. При начальном соединении головного сервера и агента происходит перенумерация требуемых сенсоров; в дальнейшем все запросы к агенту на узле ссылаются на короткий номер сенсора.

Информация на узлах собирается агентами посредством модулей расширения. В отличие от большинства систем мониторинга модули расширения в рассматриваемой системе запускаются один раз и не завершаются после получения порции данных. Это исключает накладные расходы на запуск модулей при запросе данных и приводит к минимизации нагрузки на процессор.

Модульный принцип позволяет отслеживать не только те параметры, которые предусмотрены в системе изначально, но и любые другие — достаточно написать модуль, который будет получать эти данные и передавать в систему. Интерфейс системы с модулями весьма прост и хорошо описан. Модуль может быть написан практически на любом языке программирования и для его написания не требуется высокой квалификации программиста, достаточно вставить кусок кода, получающий данные, в код модуля-образца и откомпилировать полученный модуль.

В целях повышения надежности системы в архитектуре Antmon предусмотрено использование нескольких головных серверов. Они работают параллельно и могут даже разделять общую нагрузку — каждый сервер может обслуживать свой набор агентов или отдельных сенсоров. Для каждого сенсора прописывается список головных серверов, которые могут его обслуживать. Изначально сенсор обслуживается первым в списке сервером. Если этот сервер перестает функционировать, сенсор переходит под опеку следующего в списке сервера. В случае сбоя одного из головных серверов остальные серверы перераспределяют ответственность за сенсоры и производят реагирование на сбой — запускается определенная в конфигурации команда. Когда сбойный головной сервер возвращается в рабочее состояние, остальные серверы снова перераспределяют ответственность за сенсоры, а также выполняют соответствующую команду, например уведомление администратора.

Схема работы Antmon представлена на рис. 4.

2.3. Интеграция. Каждая из описанных выше систем способна работать независимо одна от другой, но в интеграции друг с другом они получают новое качество. Предполагается, что система мониторинга настроена на сбор и хранение информации об узлах кластера: загрузка процессора, сети, использование файла подкачки, жесткого диска и т.п. Эта информация является очень важной при анализе работы параллельных программ, но использовать ее, как правило, весьма тяжело, поскольку нет возможности привязать данные ко времени работы программы и использованным вычислительным узлам.

Опираясь на собранные системой мониторинга данные, а также на информацию, предоставляемую системой Cleo, комплекс ParCon генерирует наглядное представление информации о программе пользователя. Полученные данные отображаются таким образом, чтобы пользователь увидел взаимосвязь параметров, влияющих на эффективность его программы (например, загрузки процессора и нагрузки на сеть), а также смог оценить степень сбалансированности работы программы на всех использованных узлах.

Интерфейс к системе ParCon сделан в виде web-страницы, что упрощает работу с системой. Сам интерфейс крайне прост — пользователю достаточно ввести номер своей задачи и он получит необходимые данные о ней.

3. Примеры использования. Рассмотрим несколько примеров использования системы ParCon. Все эти примеры были получены на реальных задачах пользователей кластеров НИВЦ МГУ [2].

3.1. Обмены с жестким диском.

Одна из вычислительных задач показывала на кластере очень низкую производительность. После анализа профиля исполнения стало понятно, что она очень активно использует жесткий диск и это явля-

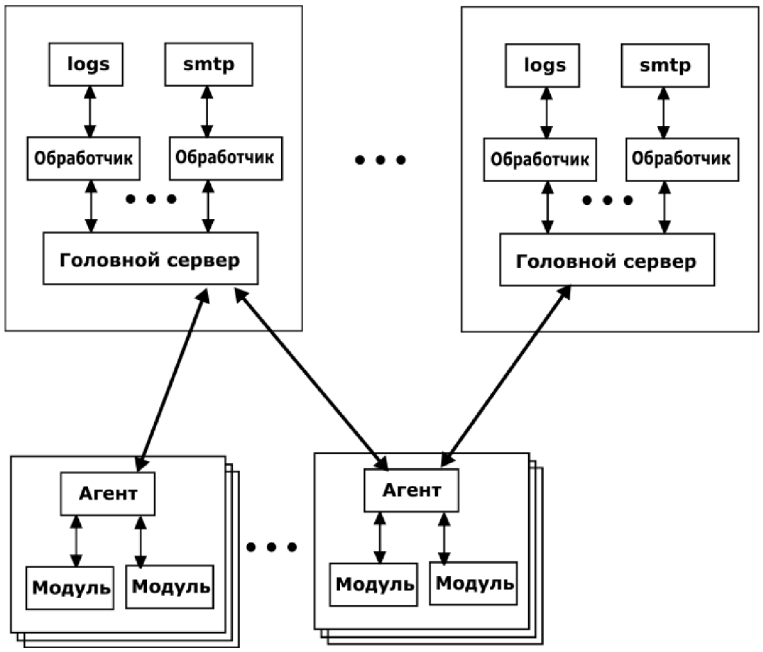

Рис. 4. Архитектура системы Antmon

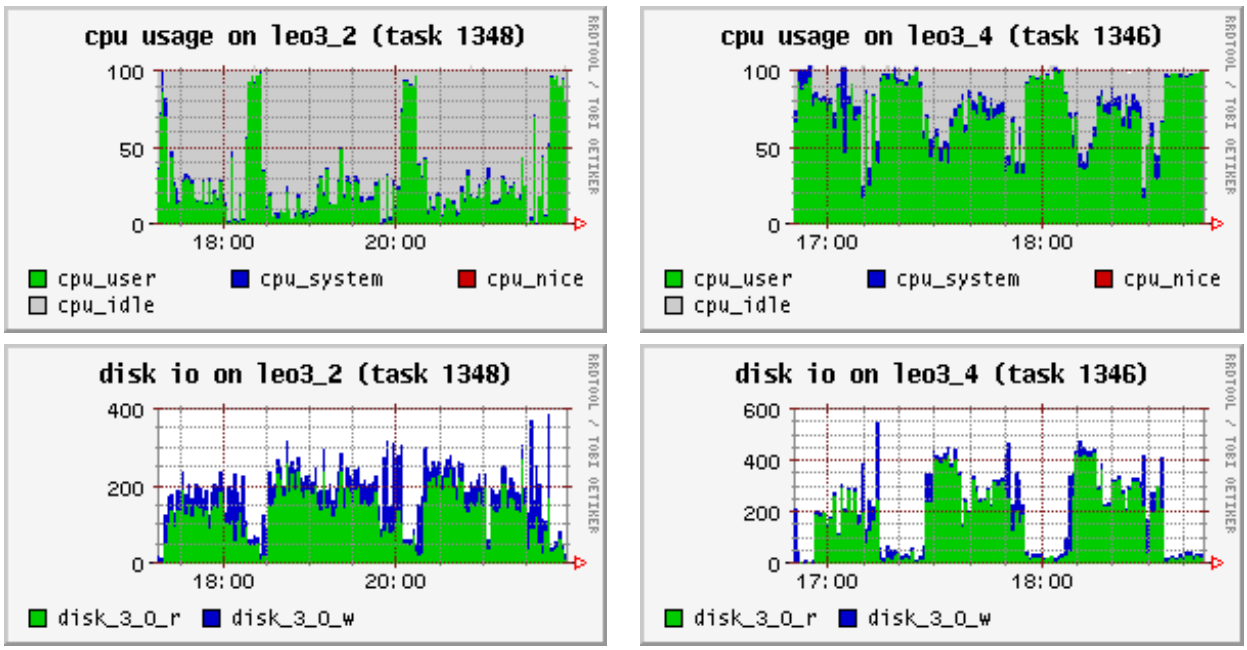

Рис. 5. Пример оптимизации задачи, активно использующей жесткий диск

ется ее узким местом. На рис. 5 слева представлены графики использования процессора и жесткого диска исходной задачей.

После небольшой переделки программа стала использовать два жестких диска вместо одного; за счет распараллеливания работы с ними был получен более чем двухкратный прирост производительности (рис. 5 справа).

3.2. Активное использование сети. Падение скорости работы программы может быть вызвано и недостаточной производительностью сети. На рис. 6 приведен пример задачи, использующей схему мастер-работники [1]. При этом на узлах-работниках (справа) загрузка процессора далеко не всегда максимальна (порядка 80 %). Из загрузки сети видно, что передаются большие объемы данных (используется 100MBit Ethernet) и значительное время уходит на прием-передачу сообщений мастером, в то время как работни-

ки ожидают. В этом случае можно либо оптимизировать прием-передачу сообщений, либо пересмотреть алгоритм распараллеливания задачи.

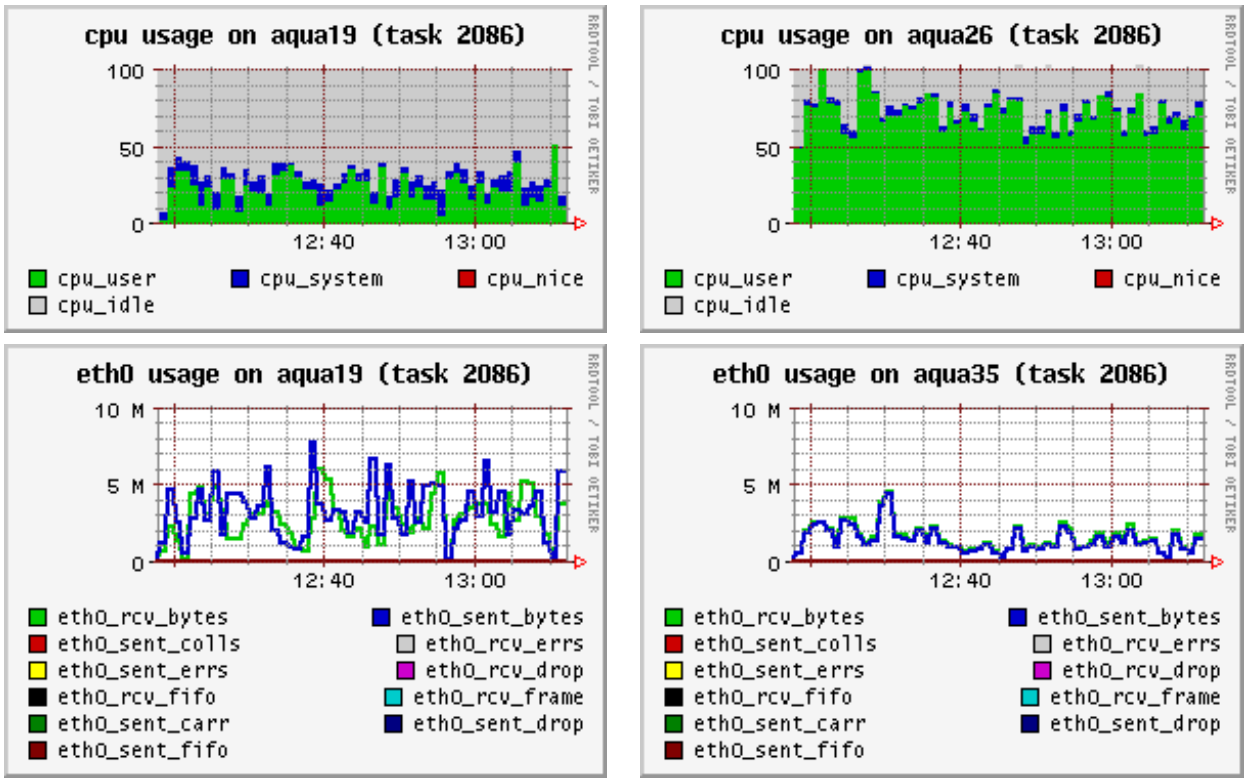

Рис. 6. Пример обнаружения неэффективного распределения нагрузки в программе

3.3. Недостаточный объем памяти. Не всегда удается точно оценить объем оперативной памяти, необходимый для работы даже одной задачи. На рис. 7 приведен пример профиля задачи, которая пыталась использовать больше оперативной памяти, чем было доступно. Из-за этого использование процессора упало вдвое. Бóльшая часть времени уходила на ожидание подкачки нужной страницы в память, что и иллюстрирует рис. 7. На левом графике показана загрузка процессора, а на правом графике — число подкачиваемых страниц.

В подобных случаях эффективность программы может падать более чем в два раза, и важно обнаружить это как можно раньше. Система ParCon способна не только показать причину замедления программы как во время ее работы, так и после ее завершения, но и предупредить системного администратора о том, что на соответствующих узлах подкачка слишком велика. В ближайшее время система сможет оповещать не только администратора, но и запустившего задачу пользователя о проблемах, возникающих во время ее работы.

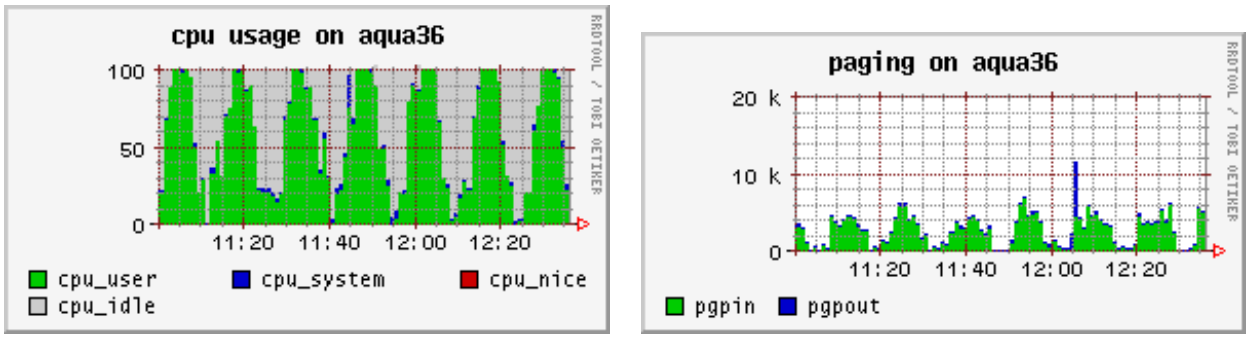

Рис. 7. Пример задачи, активно использующей подкачку страниц в память

3.4. Использование сетевых дисков. Еще одна причина возможного замедления работы про-

грамм — это работа с сетевым диском. На рис. 8 показан пример задачи, активно использующей сетевой диск. Как видно из левого графика, загрузка процессора упала до 50–60 %. При этом хорошо видно, что в то же время резко возросла нагрузка на сеть (средний график) и что эта нагрузка обусловлена использованием NFS (правый график). В таких случаях необходимо либо хранить временны́е данные на локальном диске (если по окончании счета их можно уничтожить), либо сокращать объем данных, хранимых на диске (сохранять данные в упакованном виде, например).

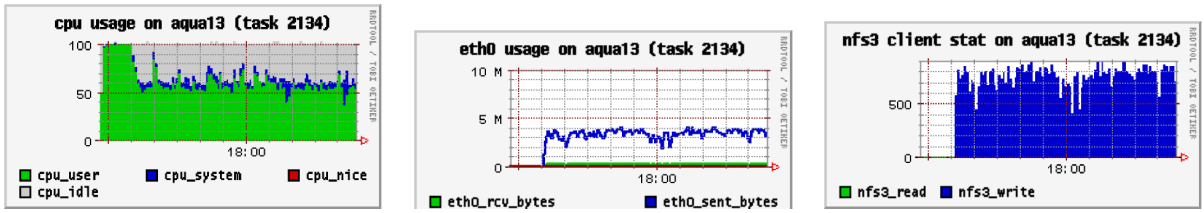

Рис. 8. Пример задачи, интенсивно использующей сетевой диск

4. Заключение. Программный комплекс ParCon прошел успешную апробацию и внедрен в ряде организаций, в частности, в Институте вычислительной математики РАН, в Южно-Уральском и Самарском государственных университетах. На его основе в течение нескольких лет построено администрирование и сопровождение всех суперкомпьютерных систем НИВЦ МГУ, обслуживающих более 250 процессоров.

Комплекс разработан в помощь как пользователю, так и администратору вычислительного кластера. Его применение позволяет более эффективно использовать ресурсы кластера. Пользователь может с помощью ParCon выявить факторы, мешающие работе параллельных программ на самых начальных этапах их отладки и использования на кластере, а администратор имеет возможность увидеть узкие места кластера в целом.

В данный момент работа над совершенствованием комплекса продолжается. Ведется разработка модулей, позволяющих автоматически сигнализировать об аномальном поведении параллельных программ. Дополняются наборы динамически загружаемых компонентов.

Страничка проекта в сети — http://parcon.parallel.ru.

Система разработана при поддержке РФФИ (грант № 05–07–90206).

СПИСОК ЛИТЕРАТУРЫ

1. *Воеводин В.В., Воеводин Вл.В.* Параллельные вычисления. СПб.: БХВ, 2002.
2. Официальный сайт суперкомпьютерного комплекса НИВЦ МГУ: http://parallel.ru/cluster/
3. Официальный сайт системы ParCon: http://parcon.parallel.ru/

Поступила в редакцию
07.06.2005