Средство тестирования выделяемых узлов кластера и освобождения их ресурсов в процессе обработки потока заданий*

И.А. Сидоров, А.Г. Феоктистов

Институт динамики систем и теории управления им. В.М. Матросова СО РАН

В настоящее время процесс проведения крупномасштабных вычислительных экспериментов в гетерогенных распределенных вычислительных средах (ГРВС) обуславливает необходимость решения широкого спектра проблем, связанных с обеспечением надежности и отказоустойчивости вычислительных процессов. В их числе мониторинг и профилактика ресурсов ГРВС, обнаружение отказов программно-аппаратных средств и их диагностика, сбор и анализ статистической информации, техническая поддержка прикладного программного обеспечения (ПО) и другие проблемы [1]. Для решения данных проблем применяются разнообразные программные средства как на уровне самого приложения, так и на уровне ГРВС [2]. В частности, системы мониторинга и диагностики программно-аппаратных ресурсов среды, поддерживающие экспертный анализ данных и прогнозирование отказов, зачастую позволяют выявлять многие критические состояния ресурсов в процессе проведения вычислительных экспериментов и своевременно реагировать на них в интерактивном или автоматическом режимах. Однако существует ряд нерешенных в полной мере актуальных проблем, связанных с неполным освобождением ресурсов узлов ГРВС после завершения выполняемых в них заданий.

При обработке потока заданий, порождаемого приложением пользователя ГРВС, ресурсы узлов среды выделяются системой управления прохождением заданий (СУПЗ) на определенный период времени для вычислительных процессов, связанных с выполнением заданий. Зачастую некоторая доля выделяемых ресурсов остается частично занятой после завершения заданий. Например, в их число могут входить: процессоры, обслуживающие ненужные фоновые процессы, которые не были сняты СУПЗ после завершения обработки заданий; локальное дисковое пространство узлов, использованное завершенными заданиями для ускорения работы приложений с временными файлами и неочищенное СУПЗ. Наличие таких ресурсов негативно влияет на обслуживание других заданий и обоснованно требует их своевременного выявления и освобождения. Кроме того, перед запуском вычислительных процессов на выделенных узлах требуется подтверждение корректной работы системного ПО, в частности, проверки работы подсистем СУПЗ, коммуникационных интерфейсов, сетевых директорий и системных служб.

В докладе представлено средство тестирования выделяемых узлов кластера Центра коллективного пользования (ЦКП) «Иркутский суперкомпьютерный центр СО РАН» (ИСКЦ) [3] и освобождения их ресурсов в процессе обработки потока заданий. Кластер включает два сегмента, узлы которых имеют различные вычислительные характеристики. Сегменты кластера объединены в ГРВС. Средство тестирования узлов кластера разработано в рамках системы мониторинга ресурсов [4], а также используется в системе непрерывной интеграции прикладного ПО, которая входит в состав инструментального комплекса Orlando Tools, предназначенного для разработки масштабируемых приложений [5]. Применение этого средства в процессе тестирования модулей приложений позволяет заранее выявлять возможные проблемы с освобождением ресурсов. В настоящее время средство тестирования узлов кластера реализовано с использованием программной платформы node.js [6] и интегрировано с программным обеспечением СУПЗ PBS Torque [7]. После очередной итерации планирования СУПЗ и выделения узлов для выполнения задания запускается процесс тестирования этих узлов, который включает в себя шесть основных этапов.

1. Проверка наличия ненужных процессов на узле по заданному фильтру. Считается, что узел не имеет ненужных процессов, если на нем присутствуют процессы только тех пользователей, которые определены в конфигурации узла. Найденные ненужные процессы снимаются системной командой kill операционной системы с сигналом немедленного завершения

* Работа выполнена в рамках научного проекта IV.38.1.1 программы фундаментальных исследований СО РАН, а также при поддержке РФФИ, проект № 19-07-00097-а.

SIGKILL с ожиданием результата в течение 2 секунд. Если после данной операции ненужные процессы в системе продолжают работать, то осуществляется повторная попытка снятия процессов с ожиданием результата в течение 4 секунд. Когда процессы не удается снять (в некоторых случаях такая ситуация бывает обусловлена ожиданием завершения операций чтения/записи в сетевую директорию, соединение с которой было утеряно), осуществляется помещение данного узла в пул сбойных узлов или перезагрузка узла с использованием интерфейса Integrated Performance Monitoring [8] в зависимости от заданных настроек узла.

2. Очистка дискового пространства в директориях локальных дисков узлов, предназначенных для хранения временных файлов вычислительных процессов (на кластерах – это зачастую директория /store, а также системная директория /tmp для временных файлов).
3. Проверка доступности всех сетевых директорий на узле и корректной работы операций чтения/записи данных в эти сетевые директории.
4. Просмотр буфера сообщений ядра операционной системы на предмет наличия критических системных ошибок (вывод команды dmesg операционной системы).
5. Анализ коммуникационных сетевых интерфейсов на предмет наличия критических ошибок с помощью команд ifconfig и ethtool операционной системы.
6. Запуск MPI-программы или распределенной программы на группе узлов, выделенных для тестирования корректного взаимодействия процессов.

При обнаружении критических состояний ресурсов узлов или их неправильного функционирования они помещаются в пул сбойных узлов для дальнейшего анализа и устранения выявленных проблем администратором ГРВС. Узлы, на которых не было выявлено проблем, возвращаются в пул доступных узлов. Затем запускается очередная итерация планирования СУПЗ.

Экспериментальный анализ результатов обработки синтетического потока заданий разных классов в ЦКП ИСКЦ с применением разработанного средства и без него показал, что в первом случае коэффициент полезного выделения узлов 1-го и 2-го сегментов в среднем выше на 2.03% и 3.21% соответственно, а общее время выполнения потока заданий сокращено на 0.87%.

Литература

1. Никитенко Д.А., Воеводин В.В., Жуматий С.А. Octoshell: система для администрирования больших суперкомпьютерных комплексов // Вестник Южно-Уральского государственного университета. Серия: Вычислительная математика и информатика. 2016. Т. 5. № 3. С. 76–95.
2. Сидоров И.А., Кузьмин В.Р. Обзор современных средств для комплексного мониторинга высокопроизводительных вычислительных систем // Фундаментальные исследования. 2016. № 9-1. С. 62–67.
3. Иркутский суперкомпьютерный центр СО РАН. URL: http://hpc.icc.ru (дата обращения: 13.04.2019).
4. Bychkov I.V., Oparin G.A., Novopashin A.P., Sidorov I.A. Agent-Based Approach to Monitoring and Control of Distributed Computing Environment // Lecture Notes in Computer Science. 2015. Vol. 9251. P. 253–257.
5. Феоктистов А.Г., Горский С.А., Сидоров И.А., Костромин Р.О., Фереферов Е.С. Непрерывная интеграция функционального наполнения распределенных пакетов прикладных программ // Параллельные вычислительные технологии: Короткие статьи и описания плакатов XIII междунар. конф. Челябинск: Издательский центр ЮУрГУ, 2019. С. 464.
6. Wilson J. Node.js 8 the Right Way: Practical, Server-Side JavaScript That Scales. Pragmatic Bookshel. 2018. 336 p.
7. TORQUE Resource Manager. URL: http://adaptivecomputing.com/products/open-source/torque/ (дата обращения: 13.04.2019).
8. IPM – Overview. URL: http://ipm-hpc.sourceforge.net/ (дата обращения: 13.04.2019).