Булочникова Н.М., Горицкая В.Ю., Сальников А.Н.

СИСТЕМА ПОДДЕРЖКИ СБОРА И АНАЛИЗА СТАТИСТИКИ О РАБОТЕ ВЫЧИСЛИТЕЛЬНОЙ СИСТЕМЫ

Факультет ВМиК, Московский Государственный Университет им. М.В. Ломоносова, г. Москва, bulya@angel.cs.msu.su, vera@angel.cs.msu.su, salnikov@angel.cs.msu.su

При решении задач на мощных вычислителях остро стоит проблема правильного распределения ресурсов этих систем – это касается как времени, так и нагрузки на процессоры и среду передачи данных. Кроме того, важно, чтобы параллельное приложение, выполняющееся на вычислительной системе, выполнялось наиболее эффективно. Для решения этих задач предназначены системы профилирования программ (например, VTUNE, PABLO, Xprofiler, Parallel Performance Project, ParaGraph, Vampir) и системы планирования вычислений (LoadLeveler, OpenPBS, Sun Grid Engine, Condor и т.д.). Проблема использования этих систем состоит в отсутствии унифицированного подхода к организации работы планировщиков, системы рекомендаций пользователю по запуску своих заданий, а также в отсутствии единого подхода к оценкам эффективности работы программы на разных вычислителях. До текущего момента задача эффективного распределения ресурсов вычислителя решалась для систем с определенной операционной системой (как правило, из семейства UNIX). Появление новых аппаратных платформ, среди которых распределенные системы, элементами которых могут быть мультипроцессоры, кластеры и т.д. требуют разработки новых методов анализа и управления потоком задач для планирования вычислений в этих системах.

Данная работа посвящена разработке и реализации методов сбора статистики о работе вычислительной системы и алгоритмов обработки собранных данных для выяснения обстоятельств, влияющих на поведение вычислительной системы и эффективность программ пользователей. Обработка собранных данных включает поддержку прогнозирования некоторых характеристик прохождения потока задач на вычислителе, исследования эффективности выполнения пользовательских приложений, а также средств визуального анализа процессов, происходящий в вычислительной системе. Разработанная система обладает независимым от программно-аппаратной платформы пользовательским интерфейсом, поэтому может быть использована на любой многопроцессорной системе. Кроме того, в системе поддерживается возможность удаленного доступа к средствам анализа.

Архитектура системы построена таким образом, что все решаемые ею задачи распределены по двум большим приложениям. Одно из приложений (условно названное сервером системы) отвечает за все работы, связанные с вычислителями: тестирование, сбор статистики, получение прогнозов. Второе приложение реализует всевозможные графические интерфейсы для работы с системой, оно включает в себя пользовательский интерфейс для доступа к средствам сервера системы, средства визуального анализа статистических данных и результатов тестирования. Одним из основных компонентов системы является база данных (см. 1).

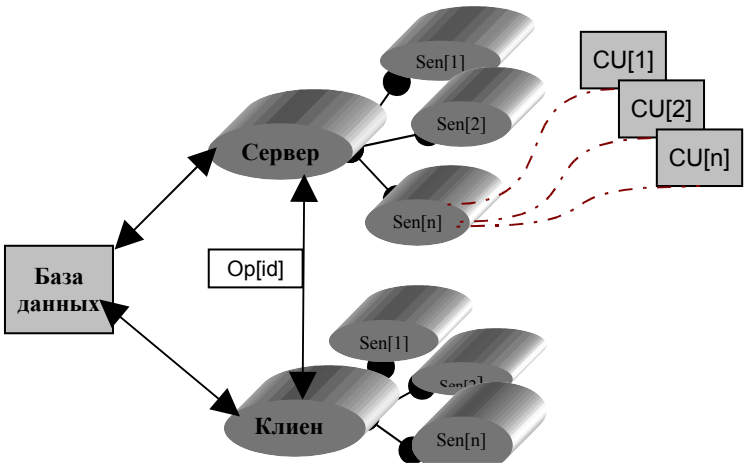

Рис. 1 Архитектура системы

В базе данных хранится вся информация о характеристиках вычислительных систем и выполняющихся задачах, которые необходимы системе для выполнения своих функций.

Клиентское приложение предоставляет пользователю различные интерфейсы для анализа статистики, тестов и работы своих программ. Оно может связываться с БД для получения необходимой информации, а также для записи туда информации о заданиях пользователя. Клиентское приложение может напрямую общаться с сервером через сокеты по протоколу TCP/IP. Это необходимо, например, в случае, когда пользователь хочет получить прогноз времени выполнения для своей задачи. Информация о задаче идет в базу данных, а серверу посылается сообщение вида <прогноз для задачи с id=id задачи>.

Серверное приложение пользователю недоступно, по существу, оно работает автономно. Сервер может соединяться с БД для чтения и записи необходимой информации. Каждая из утилит (функций) сервера имеет возможность соединяться с вычислительным узлом.

В связи с тем, что разрабатываемая система должна быть независимой от платформы, на которой работает, сервер баз данных выбирался из тех же соображений. В качестве сервера баз данных была принята СУБД MySQL.

В базе данных хранится следующая информация:

- описание вычислителей (включает имя вычислителя, технические характеристики, параметры соединения и проч.);
- описание методов тестирования вычислителей (методы тестирования, параметры, результаты);
- статистика по загруженности вычислителя;
- статистика по прохождению задач пользователя;
- параметры настройки и результаты задач прогнозирования.

В процессе разработки системы были выделены универсальные характеристики функционирования вычислителя и прохождения потока задач, общие для разных архитектур и программного обеспечения вычислительных систем.

По потоку задач собирается следующая статистика:

- имя задачи, присвоенное планировщиком,
- время постановки задачи в очередь,
- время отправки задачи на счет,
- время окончания счета,
- ресурсы, требующиеся задаче (максимальное время выполнения, количество процессоров).

Также собирается следующая информация о работе вычислительной системы:

- использование процессоров,
- состояние сети,
- производительность операций ввода/вывода.

Предполагаемая устойчивость вычислителя к большим нагрузкам выясняется при помощи тестирования производительности процессоров и коммуникационной среды многопроцессорной системы. *Тестирование производительности процессоров* заключается в произведении известного числа операций с плавающей точкой и замера времени этих операций. *Тесты производительности сетевых обменов* измеряют время передачи сообщений между процессорами в зависимости от способа передачи данных (синхронно или асинхронно) и интенсивности “фоновых” передач. Более подробное описание тестов см. в [1-2].

Прогнозирование прохождения потока задач заключается в предсказании времени ожидания в очереди и времени нахождения задачи на счете в вычислительной системе (не является временем выполнения, так как в зависимости от политики распределения ресурсов задача может приостанавливаться) по времени постановки задачи в очередь и ресурсов, требующихся задаче для выполнения. Поддерживается прогнозирование статистическими методами (расчет математических ожиданий) и прогнозирование при помощи нейронной сети (используется сигмоидальная нейросеть с одним скрытым слоем).

В рамках системы реализован также мониторинг эффективности использования ресурсов пользовательской программой. Для каждой пользовательской задачи, запускаемой на вычислителе, собрана статистика по ее выполнению за определенный промежуток времени. С помощью системных утилит мониторинга загруженности многопроцессорной системы исследуется соотношение требуемых и реально используемых задачей ресурсов.

Система имеет гибкие и переносимые между платформами средства для поддержания визуального анализа данных о прохождении потока задач на вычислителе. Эти средства позволяют провести следующие виды анализа:

- анализ загруженности процессоров,
- анализ загруженности сетевых интерфейсов,
- анализ результатов тестирования вычислительной системы,
- анализ эффективности ресурсных запросов пользователя,
- анализ эффективности использования вычислительных ресурсов.

Проведение такого рода анализа не только дает полную картину о прохождении потока задач, но и позволяет найти недостатки в алгоритмах рассматриваемых задач.

При помощи описанных методов и алгоритмов визуального анализа прохождения потока заданий были проведены исследования эффективности использования ресурсов пользователями вычислительного комплекса IBM eServer pSeries 690 (Regatta) на основании выборочной статистики.

На 2 показан пример анализа эффективности ресурсных запросов пользователей. По горизонтальной оси – время выполнения задачи пользователя, по вертикальной оси – пользователи. Каждый кружок соответствует одной задаче. Цвет кружка варьируется в зависимости от величины коэффициента эффективности задачи. Чем краснее кружок, тем менее эффективно выполнялась задача (на черно-белом рисунке красные кружки самые темные).

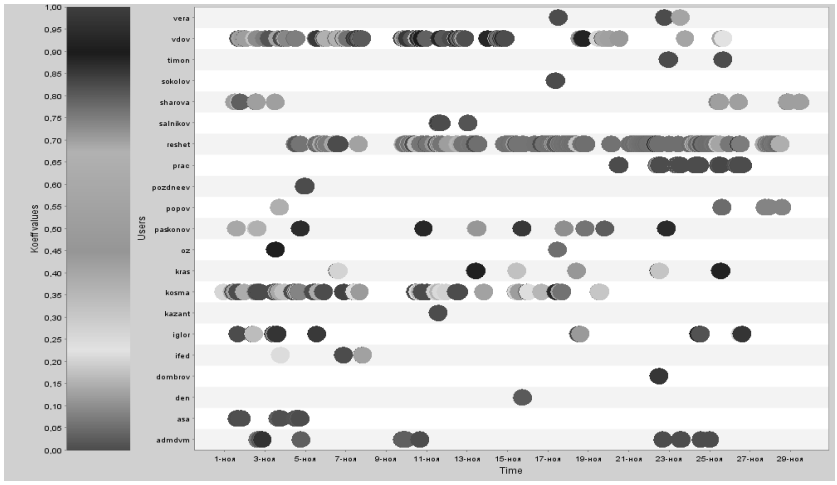

Рис. 2 Эффективность ресурсных запросов пользователей

Для анализа эффективности конкретной программы необходимо сначала определить, на каких процессорах выполнялась задача (на машине Regatta сбор такой статистики не поддерживался). Алгоритм выявления этих процессоров показан на 3. По горизонтальной оси отложены процессоры. Каждому процессору соответствует свой столбец, показывающий скачок загруженности на время начала задачи (синий, на черно-белом рисунке более светлый цвет) и окончания задачи (красный, на серно-белом рисунке более темный цвет). Для данного примера четко можно установить 6 из 8ми запрашиваемых пользователем процессоров (на рис. процессоры № 3, 7, 9, 11, 13, 16). После того, как процессоры, на которых выполнялась задача, установлены, можно посмотреть их загруженность (рис. 4). На данном рисунке отчетливо видны места простоя некоторых процессоров, что указывает на наличие в алгоритме задачи некоторых погрешностей.

Система будет полезна пользователям многопроцессорных вычислительных систем (поддерживается анализ эффективности приложений), администраторам (мониторинг загруженности вычислителя, анализ результатов тестирования и пользовательских запросов) и разработчикам систем управления заданиями (анализ процессов, происходящих в вычислительно системе, для внедрения новых алгоритмов распределения ресурсов).

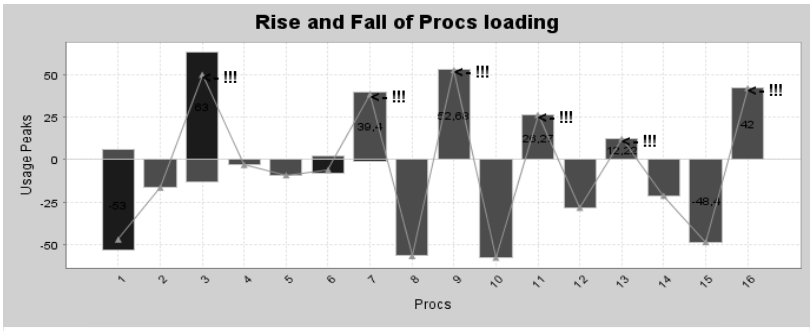

Рис. 3 Определение процессоров, на которых выполнялась задача

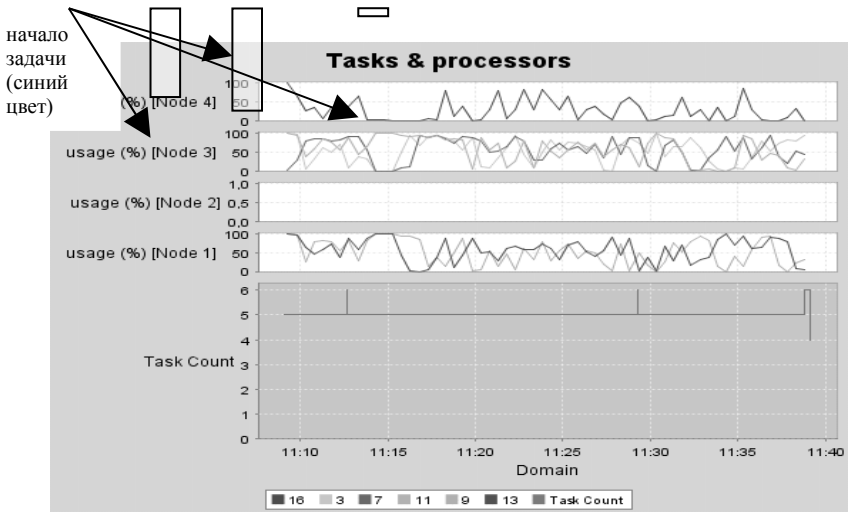

Рис. 4 Загруженность процессоров конкретной задачей пользователя

Литература

1. Булочникова Н.М., Горицкая В.Ю., Сальников А.Н. Некоторые аспекты тестирования многопроцессорных систем. //Тематический сборник №5 ф-та ВМиК МГУ, 2005
2. Булочникова Н.М., Горицкая В.Ю., Сальников А.Н. "Методы тестирования производительности сети с точки зрения организации вычислений" //Труды Всероссийской научной конференции "Научный сервис в сети Интернет 2004", Издательство Московского университета 2004 г. ISBN 5-211-05007-X , стр. 221-223.