ISSN 2079-3316 ПРОГРАММНЫЕ СИСТЕМЫ: ТЕОРИЯ И ПРИЛОЖЕНИЯ №3(26), 2015, с. 73–92

УДК 004.75:004.021

В. П. Фраленко, А. Ю. Агроник

Средства, методы и алгоритмы эффективного распараллеливания вычислительной нагрузки в гетерогенных средах

Аннотация. Работа посвящена анализу современного состояния исследований в области алгоритмического, математического и программного обеспечения распределения задач по вычислительным узлам гетерогенной вычислительной среды. Предложена классификация стратегий распределения нагрузки: по принципу учета динамики, по принципу управления, по признаку универсальности, с прогнозированием/без прогнозирования состояния системы и пр. Рассмотрен ряд методов, систем и комплексов распределения нагрузки, в том числе следующие: метод с представлением задачи в виде направленного ациклического графа, модель планировщика задач на основе метаданных, системы «DIET», «ProActive», «Moab», «Maui», система поддержки «пластичных» заданий, комплекс потоковой обработки в терминах теории массового обслуживания, сервис-ориентированный подход. Использование указанного обеспечения позволяет минимизировать время простоя вычислительных устройств, сократить объемы и время передачи данных от одних исполнительных устройств другим, повысить общую масштабируемость, минимизировать время доступа к данным и пр. Выявлены достоинства и недостатки, даны предложения по применению.

Ключевые слова и фразы: распределение вычислительной нагрузки, вычислитель, планировщик, модель, рекомендации, обеспечение, алгоритм.

Введение

Гетерогенной вычислительной системе, потенциально включающей кластеры, персональные компьютеры, микрокомпьютеры и пр. устройства, близки отличительные свойства, характерные для ГРИД-систем [1]: распределенность компонентов, динамическое изменение

Работа выполнена при частичной финансовой поддержке Российского фонда фундаментальных исследований (проект № 15–29–06945–офи_м «Развитие моделей, методов и программных средств обработки мультиспектральных снимков, видеопотоков и данных телеметрии для задач космического мониторинга арктической зоны»).

конфигурации, неоднородность (использование узлов с разнородными программно-аппаратными ресурсами), зачастую неконтролируемое множество задач, решаемых на конкретном вычислителе. Следует отметить, что данные могут распределяться на вычислительные узлы не всегда с целью осуществления каких-то неотложных вычислительных действий для решения той или иной задачи. Предварительное сохранение данных на вычислителях (возможно, с дублированием информации — для обеспечения отказоустойчивого доступа) может быть как частью подготовки к началу распределенного решения сложной задачи в будущем, так и независимой процедурой в рамках функционирования пассивного или активного распределенного хранилища данных [2–5] (в таких случаях данные становятся ресурсом). При этом и сохранение, и чтение, в свою очередь, могут в полной мере считаться операциями с определенными оценками, так как их выполнение требует использования не только физических устройств хранения данных, но и процессорных ресурсов, и некоторого объема оперативной памяти.

Наиболее известными исследованиями в этой области можно считать работы Бершадского А.М., Голубева И.А., Кальпеевой Ж.Б., Селиверстова Е.Ю., Покусина Н.В., Хачкинаева Г.М., Телеснина Б.А., Бычкова И.В., Agrawal D., Zhao H., Sakellariou R., Amar A., Bolze R., Boix E., Amedro B., Bodnartchouk V. и Baduel L. Выбор способа распределения задач (данных) по вычислительным узлам определяет общую эффективность вычислительной среды. Корректный выбор может минимизировать время простоя вычислительных устройств, сократить объемы и время передачи данных от одних исполнительных устройств другим, повысить общую масштабируемость, минимизировать время доступа к данным и пр. Настоящая работа представляет собой аналитический обзор научно-технической литературы, дающий представление о методах эффективного распараллеливания вычислительной нагрузки.

1. Классификация стратегий распределения нагрузки

На основе работ [6–9] можно выделить следующую классификацию.

(1) Статические, полудинамические и динамические стратегии (по способу учета динамики). Первая стратегия предполагает фиксированный и определенный заранее план распределения нагрузки. План полудинамической стратегии задается при инициализации,

до старта основных расчетов. Динамическая стратегия распределения вычислений/данных, как очевидно, периодически изменяется — под воздействием стимулов и условий окружающей среды или по заранее определенному графику. Адаптивные стратегии позволяют выполнять балансировку нагрузки и необходимую рекомбинацию ресурсов при переконфигурировании распределенной среды. Например, при появлении новых вычислительных узлов или при отказах уже работающих. В случае зависимого распределения задач отслеживаются события изменения особенностей загрузки, запланированные во времени события и пр. При обнаружении таких событий требуется построить обновленный план распределения нагрузки.

(2) Стратегии с децентрализованным, централизованным и иерархическим управлением (по принципу управления). Централизованные стратегии предполагают наличие центрального элемента — планировщика, который для принятия решений опирается на информацию от каждого узла распределенной системы. В децентрализованных алгоритмах распределение данных планируется каждым вычислительным узлом независимо, либо с учетом данных от самых ближайших узлов. Иерархический подход сочетает свойства двух других: главный планировщик, называемый метапланировщиком, распределяет поступающие задания в соответствии с некоторым набором правил, глобальная единая очередь заданий разбивается на несколько очередей — по количеству элементов на каждом из уровней иерархии.

(3) Универсальные и специализированные стратегии (по признаку универсальности). Специализированные стратегии могут ориентироваться на специфичность архитектуры распределенной системы, частный случай топологии сетевой среды и пр.

(4) Прогностические стратегии и стратегии без использования механизмов прогноза состояний системы (например, загруженности узлов). Стратегии, применяющие механизмы краткосрочного и долгосрочного прогноза развития вычислительных процессов, существенно превосходят те, что не имеют таких механизмов. Прогностические стратегии делятся на стратегии с учетом и без учета причин разбалансировки.

(5) Стратегии, учитывающие только производительность вычислительных узлов распределенной системы (в том числе производительность имеющихся локальных хранилищ данных), и стратегии,

также учитывающие производительность коммуникационной подсистемы. В последнем случае осуществляется анализ характера информационно-управляющего трафика сети.

(6) Стратегии, не использующие и использующие знания о необходимых для решения задач свободных ресурсах (в том числе — с привлечением приближенных и реалистичных методов оценки их количества). В последнем случае могут учитываться такие характеристики узлов, как количество активных потоков, количество и скорость имеющихся вычислительных устройств (процессоров, ускорителей вычислений), объем доступной оперативной памяти, наличие свободного места для выполнения файловых операций и пр. В рамках следующих стратегий не используются знания о необходимых для выполнения подзадач ресурсов: First in First Out (FIFO, выполнение задач в порядке появления); Last in First Out (LIFO, обратный порядок выполнения); Random Selection for Service (случайный порядок выполнения); Time Sharing (равномерное выделение ресурсов всем задачам); Least Attained Service (для выполнения выбирается та задача, что получила наименьшее время обслуживания). Использование указанных подходов опирается на предположение о том, что решаемые задачи имеют близкую вычислительную и пространственную трудоемкость, кроме того предполагается использование схожих узлов-вычислителей при одновременном решении на каждом узле не более одной задачи. Однако, при наличии знаний о ресурсах могут использоваться дополнительные стратегии, например, Shortest Job First (для выполнения выбирается та подзадача, которая имеет меньшие требования к ресурсам); Shortest Remaining processing time (выполняется задача с минимальной оценкой времени обслуживания).

(7) Стратегии с разными инициаторами распределения задач. Инициирующей стороной может выступать как приемник нагрузки (недогруженные узлы), так и источник нагрузки (центральные узлы или даже перегруженные узлы).

Представленная классификация позволяет лучше понять внутренние принципы работы программ-планировщиков, дает основания полагать, что универсальной стратегии под все случаи жизни не существует. В одном и том же программном продукте, как очевидно, может использоваться несколько разнородных стратегий распределения нагрузки. Далее предлагается перейти к рассмотрению пригодных для

использования в гетерогенных вычислительных условиях частных решений для распределения и балансировки задач.

2. Актуальные решения для распределения и балансировки задач

В работе [10] описан подход с представлением решаемой задачи в виде ациклического графа (Directed Acyclic Graph, DAG) [11]. Каждой его вершине сопоставляется специфическая задача, выполняемая в ходе работы над общей заявкой, дуги же определяют последовательность работ. Граф обладает одной вершиной-истоком, сопоставляемой с функцией формирования исходных данных. Считается, что имеется одна вершина-сток, сопоставляемая с функцией формирования результата обработки. Вес каждой вершины графа соответствует стоимости конкретной работы, вес ребер графа — цене передачи данных с одного обработчика на другой. Стоимость (сложность) выполнения отдельных работ выражается в абстрактных условных единицах. Аналогично выражается временная цена передачи данных между независимыми обработчиками (см. рис. 1). Эксперт, работающий с гетерогенной распределенной системой вычислений, состоящей из разнородных вычислителей, объединенных в единую вычислительную сеть, определяет понятие элементарной задачи и временную сложность ее выполнения на каждом счетном устройстве, эксперт задает и время передачи такой задачи от одного вычислителю другому. Зная условную стоимость каждой из работ общей задачи очень просто выполнить оценку выполнения всей задачи на конкретном вычислителе. Алгоритм упорядочивания вершин DAG для выделения групп работ с возможностью независимого планирования выполнения описан в работе [12]. Для планирования работ в группе предлагается использование метода сбалансированного минимального времени завершения (Balanced Minimum Completion Time, BMCT).

В работе [7] предложена математическая модель планировщика задач, отличающаяся от существующих моделей совместным учетом удовлетворяющих критерию близости метаданных предыстории выполнения и метрик загруженности ресурсных объектов при отображении решаемых задач на имеющиеся вычислительные ресурсы. Предложен новый алгоритм поиска ближайших соседей, использующий локализованное хеширование. Его отличительной особенностью является учет типов атрибутов и их значимости для ресурсопотребления. В модели планирования учитывается ряд множеств: задач, ат-

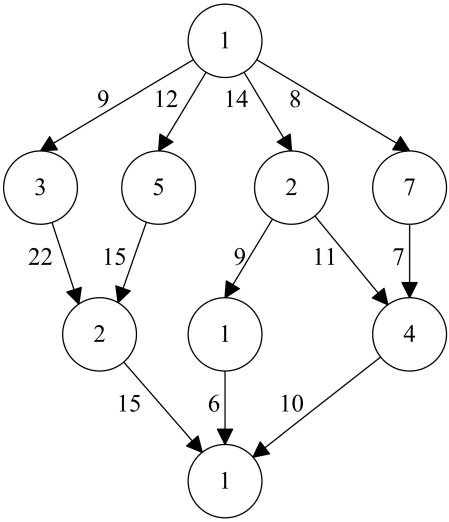

Рис. 1. Пример ацикличного графа вычислений

рибутов данных (метаданных), узлов-обработчиков, метрик загрузки ресурсов узлов-вычислителей. Задается отображение множества задач на множество узлов-обработчиков, учитываются экспертные оценки вычислительных затрат при выполнении задач. К примеру, вычислительный узел может характеризоваться количеством процессоров и объемом свободной оперативной памяти или памяти длительного хранения. Предлагаемый метод основывается на метаданных и ресурсных метриках и представляет собой последовательность применения ряда следующих специальных функций: получения метрик загрузки вычислительных ресурсов; оценки вычислительных затрат для задач предыстории; оценки затрат ресурсов на запуск и решение новых задач на имеющихся вычислительных узлах; вычисления коэффициентов назначения на основе полученных оценок и пр. Для каждой пары (узел, задача) на определенную глубину анализируется накопленная счетная предыстория и вычисляется расстояние до всех ранее решенных задач; при вычислении расстояния учитывается различная цена атрибутов задач (метаданных, относящихся к ресурсопотреблению). Для сравнения с методом использовался подход с разнесением заданий по узлам-вычислителям на основе очереди FIFO и подход с выделением наименее занятого вычислительного узла-вычислителя под новые

задачи LLF (Least Loaded First). Экспериментальные исследования показали следующее:

- общее время обработки заявок сокращается на 20% при очень интенсивном поступлении задач в систему, однако увеличивается при средней и слабой интенсивностях на 4,5% и 10,7% соответственно;
- усредненное время ожидания обработки сокращается на 7,6% при высокой интенсивности появления задач, но растет на 10,6% и 70,6% в случае средней и, соответственно, слабой интенсивностей (из-за значительной доли накладных расходов);
- увеличение интенсивности поступления заявок расширяет использование предыстории, что приводит к повышению качества прогнозирования.

В этом исследовании упомянуты системы «DIET», «ProActive», «Moab» и «Maui». Это наиболее популярные западные системы для планирования вычислений, рассмотрим их далее.

Система «DIET» [**13**] предназначена для организации вычислений на основе разнородных типов распределенных ресурсных объектов. Архитектура системы (см. рис. 2) состоит из следующих типов компонентов:

- клиентское приложение для выполнения задач (Client);
- управляющий агент (Master Agent, MA) — работает с запросами на выполнение вычислений от клиента, получает данные о свободных ресурсах с вычислителей, выбирает подходящий исполняющий узел (вычислитель) и отправляет его адрес клиенту;
- местный агент (Local Agent, LA) — обеспечивает передачу запросов и информации между управляющими агентами и вычислителями, осуществляет локальное планирование для вычислителей;
- серверная служба (Server Daemon, SeD) — служба, функционирующая на конечных узлах-вычислителях, необходима для получения информации о разнородных типах решаемых задач, информации о количестве ресурсных объектов и их загруженности.

При поступлении новой задачи серверные службы «DIET» осуществляют оценку свойств ресурсов распределенного вычислителя. Такие оценки передаются на родительский агент, где вычислители упорядочиваются по некоторому оптимизационному критерию: по умолчанию — в порядке времени последнего применения. Если возможности хранения временных отметок нет, то вычислитель выбирается случайным

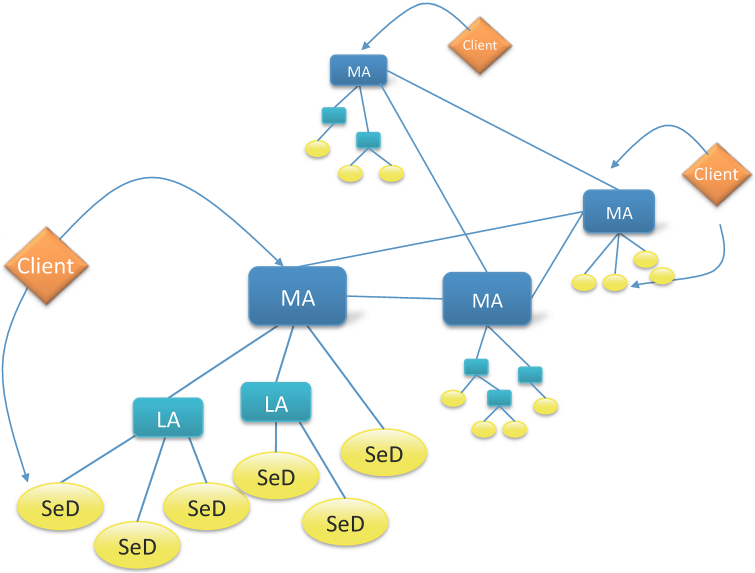

Рис. 2. Иерархия компонентов системы «DIET»

способом. «DIET» реализует простую схему автоматизированного выбора ресурсов (данных, вычислительных устройств, оперативной памяти и пр.) на базе мониторинга состояния узлов системы и не использует информацию о ресурсных ограничениях. «DIET» позволяет модифицировать свой планировщик за счет подключаемых модулей. С их помощью можно дополнить множество метрик оценки производительности новыми прикладными метриками и задать критерии, необходимые для упорядочивания узлов-претендентов на выполнение новой задачи. При разработке подключаемых модулей проблема планирования решается разработчиком. Из недостатков системы можно выделить требования к программной реализации подпрограммы планирования для новых прикладных приложений.

Кроссплатформенный программный комплекс «ProActive» [14] состоит из системы управления ресурсами «ProActive Grids and Clouds», распределенного планировщика «ProActive Orchestration and Scheduling», программного стека «ProActive Programming» для ГРИД-систем и ряда графических приложений для работы пользователя. Подпрограмма-планировщик получает задачи из очереди в соответствии с

политикой FIFO, однако, доступны интерфейсы и рекомендации для создания новых методов планирования. Планировщик не обеспечивает выбор узла-исполнителя для задачи, он реализует способ поиска ресурсов под конкретные запросы клиентов, содержащие критерии отбора узлов. Задача выполняется, если имеется необходимое количество ресурсов, в противном случае она ожидает в очереди. После выделения ресурсов клиенту (например, на том или ином вычислительном узле появились необходимые для дальнейшей работы данные), клиенту делегируется исключительный прямой доступ к узлу-обработчику до момента завершения требуемых операций. У «ProActive» есть два основных недостатка: 1) даже для самой простой задачи требуется реализовывать методы выбора узлов-вычислителей; 2) отсутствуют необходимые на практике механизмы конфигурирования политик извлечения задач из очереди.

Планировщик «Moab» [**15**] может работать в режиме опроса системы контроля за свободными ресурсами и в режиме регистрации событий. Следующие события могут активировать очередной цикл планирования: появление новой задачи; окончание работы над задачей; появление нового узла-вычислителя; корректировка политик (настроек). Каждая итерация планирования предполагает запрос к системе контроля за ресурсами об актуальной информации об их состоянии, уровне рабочей нагрузки, текущих настройках планировщика. Затем «Moab» получает список задач с выполненными условиями запуска и упорядочивает эти задачи в соответствии с их относительным приоритетом. Относительный приоритет вычисляется на основании знаний о владельце задачи, ее размере, длительности ожидания в очереди и пр. Планировщик «Moab» основывается на подходе с назначением весов для некоторого количества независимых целей (максимизация загрузки ресурсов, повышенные приоритеты отдельных пользователей, исключение длительного ожидания выполнения), с помощью этих весов вычисляется приоритет каждой из задач. Основные целями «Moab» при выборе вычислительного узла является рост производительности при исполнении текущей задачи и обеспечение гибкости планирования новых задач. Гибкость модульного подхода к планированию гарантируется возможностью настройки политики выбора узлов-исполнителей не только локально (для текущей задачи), так и на системном уровне. Недостатком системы следует считать отсутствие автоматизации определения ресурсных требований решаемых задач. Кроме того, последовательный процесс планирования выполняется

лишь для самых приоритетных подзадач, это приводит к локальному оптимуму, то есть не самому оптимальному решению по планированию выполнения всей задачи.

Информация о задачах обработки для «Maui» [16] содержит ряд атрибутов: учетная запись, от имени которой была запущена задача, статус задачи, наличие предварительно полученных данных и пр.; требования к ресурсам: общее количество необходимых ресурсов и временной интервал, на котором требуется их зарезервировать; условия выбора узла-обработчика. Узел-обработчик для «Maui» – набор ресурсных сущностей с множеством связанных атрибутов (процессоров, памяти, дисковых накопителей, сетевых карт, интерфейсов, лицензий на программное обеспечение и т.д.). Информация об узлах, включая их состояние и количество доступных ресурсов, поступает к планирующей подпрограмме от системы управления ресурсами. «Maui» поддерживает политики планирования, при их применении используется понятие «класса» задачи, ассоциируемого с одним или большим числом атрибутов (ограничений). Например, это могут быть атрибуты задачи, описывающие ее длительность или требования к ресурсам; атрибуты-ограничения на минимальное/максимальное число узлов-обработчиков; атрибуты, задающие ожидаемое время поступления задачи, минимальный объем дискового пространства и т.п. Отдельные классы предусматривают независимую очередь обработки. В то же время, классы отслеживаются «Maui» как отдельный вид потребляемых ресурсов. Подпрограмма-планировщик «Maui» на каждой итерации планирования выполняет те же действия, что и «Moab», однако «Maui» поддерживает резервирование в отношении ресурсов (с заданием временных интервалов и списков доступа). Недостатки системы в том, что она требует задания точных условий выбора узлов-обработчиков; упреждающего разбиения решаемых задач на классы; задания ограничений доступа; определения политики работы с очередью.

В источниках [17, 18] представлена система, ориентированная на работу с пластичными заданиями (moldable jobs), то есть с такими заданиями, которые могут параллельно обсчитываться на разном количестве процессорных устройств (количество устройств определяет размерность задания). Эксперт задает список возможных размерностей, выбор необходимого значения размерности осуществляется программной-планировщиком в соответствии с условиями задачи и уровнем загрузки вычислителей. Все вычислительные устройства

делятся на непересекающиеся классы. Понятие «класс» объединяет вычислители с одинаковой архитектурой и схожими процессорными устройствами (ПУ) близкой производительности. Узлы одного класса могут иметь разное количество оперативной и постоянной памяти, могут отличаться количеством ПУ, набором имеющегося специализированного аппаратно-программного обеспечения. Администратор системы берет на себя функцию описания имеющихся ресурсов. Ресурс — свойство, отличающее объекты разных уровней. При постановке нового задания формируется список необходимых ресурсов. Планировщик должен удовлетворить требования к ним при запуске задания. Ресурсы могут быть как постоянными, так и расходуемыми. Ресурс не привязывается к одному и тому же уровню объектов. Например, для SMP-вычислителей оперативная память является ресурсом уровня узла, в случае разделяемой памяти последняя является ресурсом ПУ. Очередь заданий возникает лишь в случае, если сумма размерностей имеющихся заданий превышает имеющееся количество ПУ. В иных случаях конкуренция разных заданий исключена. Для своих заданий владелец определяет список классов вычислителей, на которых они могут быть исполнены, определяет диапазон размерностей и список необходимых ресурсов (в первую очередь — необходимое процессорное время). Реализация планировщика основывается на централизованном клиент-серверном подходе с сервером, модулями-агентами и клиентскими программами. Модули-агенты, как и единственный сервер, представлены в виде фоновых процессов. Агенты собирают различную информацию о своих узлах и отсылают ее на сервер планировщика. Они же запускают задания. Поддерживается система приоритетов заданий, учитывающая общее время простоя заданий в очереди, их сложность, данные о владельцах и их активности в прошлом.

В работе [**19**] гетерогенный комплекс потоковой обработки информации представлен как разомкнутая сеть массового обслуживания, состоящая из ряда перенумерованных одноканальных узлов. Каждый узел-вычислитель — аппарат с известной интенсивностью обслуживания. Время обслуживания — случайная величина, распределенная по экспоненциальному закону. Норма обслуживания определяется следующим образом: если требование, полученное одним из аппаратов, застает его за вычислениями, то оно ставится в очередь ожидания этого устройства. Окончание обработки приводит к удалению требования из сети. Авторами работы проведены исследования модели на определение стационарного распределения вероятностей, определение

среднего пребывания требований в системе, минимизацию среднего пребывания требования в системе за счет управления делением входного потока. Анализ результатов показывает, что зависимость вероятности обслуживания заданий от интенсивности потока требований такова, что при повышении интенсивности минимальные вероятности передачи заданий узлу-обработчику увеличиваются, а максимальные падают. Для рассматриваемой модели были исследованы метод с асинхронными обработчиками, метод минимизации времени пребывания заданий в очереди (задание направляется вычислителю, «обещающему» минимальное время) и метод с поддержанием фиксированного распределения загрузки (считаем, что статистически оцениваемые вероятности отправки заданий на вычислители близки к известным оптимальным значениям). Экспериментально подтверждается, что метод с асинхронными обработчиками наилучшим образом подходит для входного потока с высокой интенсивностью поступления заявок. Второй — для малой интенсивности входного потока, его не рекомендуется применять для случаев с большой нагрузкой, так как на наиболее быстром обработчике имеем разрастание очереди, слабые же простаивают. Третий метод наиболее эффективен как для крайне низкой, так и для крайне высокой интенсивности входного потока заявок, однако, при этом он превосходит первые методы в широкой области средних значений.

В работе [20] описан мультиагентный подход с сервисно-ориентированным управлением распределенной задачей. В мультиагентной системе (см. рис. 3) определен ряд правил группового поведения, на их основе и выполняется координация действий агентов. У агентов есть заданные роли, для каждой имеется набор правил поведения в сообществе агентов. Сообщество включает агентов, обеспечивающих постановку задачи, планирование вычислений, классификацию, конкретизацию и выполнение заданий, мониторинг и разделение ресурсов. Агенты могут объединяться друг с другом и даже соперничать. Аналогичный подход был ранее представлен и в работах других исследователей, например, в материалах [21]. Алгоритм управления потоками детально рассмотрен авторами в работе [22]. В его основе экономическая техника управления спросом и предложениями в отношении ресурсов, это позволяет использовать информацию о политике администрирования узлов-вычислителей распределенной сети и обеспечивает необходимый уровень справедливости при управлении ресурсами. Сформированное агентами задание пересылается агенту-

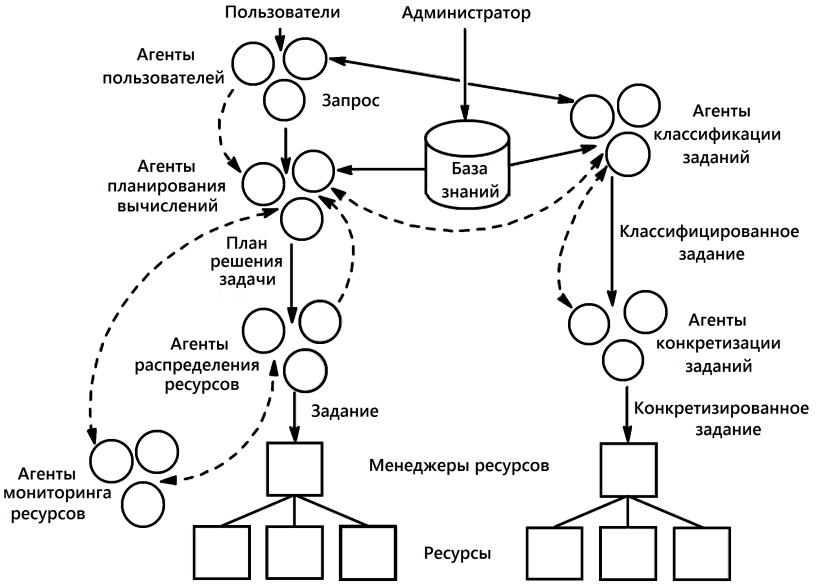

Рис. 3. Структура мультиагентной системы

менеджеру, взаимодействующему с агентом мониторинга ресурсов. Такое взаимодействие приводит к распределению задач на локальные агенты вычислителей. Задачи распределяются с помощью тендерной модели, где лоты — вычислительные работы, а участники — претендующие на них вычислители. Описанное управление может приводить к увеличению времени выполнения отдельных задач, так как не всегда возможно учесть особенности структуры подзадач и пользовательские предпочтения в отношении ресурсов.

Заключение

Методы, используемые в системах планирования/распределения вычислительной нагрузки, в большей части случаев опираются на применение информации о ресурсных запросах задачи, что перекладывает проблему настройки планирования ресурсов на конечного пользователя. Учет характера загрузки вычислителей заданиями и краткосрочное прогнозирование высвобождения вычислительных ресурсов (в том числе — локальных физических хранилищ данных), опирающееся на составленные планы выполнения работ, позволяют

качественно манипулировать вычислительными ресурсами для решения вновь поступающих задач, что дает возможность использовать планирование для уравновешивания нагрузки на распределенный вычислитель.

Методы, использующие метаданные и ресурсные метрики, показывает хорошую эффективность лишь при высокой частоте поступления заявок. Однако, сочетание метода с известными, например, с FIFO и LLF, позволяет сократить временные затраты. Использование инструментов с адаптивным ограничением размерности заданий и циклическим выравниванием нагрузки позволяет снизить время прохождения заданий.

В рамках модели гетерогенного комплекса потоковой обработки данных, формализованного в виде сети массового обслуживания, хорошие результаты показывает метод на основе сохранения зафиксированного распределения загрузки узлов-вычислителей. Применение сервис-ориентированного агентного подхода позволяет выполнять контроль за вычислениями на уровне приложений и потоков заданий; распределять с помощью агентов и специальных менеджеров необходимые для операций ресурсы.

Полученные результаты проведенных аналитических исследований планируется применить при доработке и расширении ранее созданного в ИПС им. А.К. Айламазяна РАН математического, алгоритмического и программного обеспечения для организации высокопроизводительных вычислений в обычных и гетерогенных средах [**23**–**28**].

Список литературы

[1] В. В. Воеводин, Вл. В. Воеводин. *Параллельные вычисления*, БХВ-Петербург, СПб., 2002, 608 с. ↑ 73.

[2] J. Piernas, J. Nieplocha. *Active Storage User's Manual*, Pacific Northwest National Laboratory, 2007 (english), URL http://hpc.pnl.gov/active-storage/as_users_manual_october_2007.pdf ↑ 74.

[3] *Cascading | Application Platform for Enterprise Big Data* (english), URL http://www.cascading.org/ ↑ 74.

[4] *Welcome to Pig!* (english), URL http://hadoop.apache.org/pig/ ↑ 74.

[5] Е. О. Тютляева, «Разработка и реализация распределенного архива изображений дистанционного зондирования Земли», *Труды XIII научно-практической конференции Университета города Переславля* (Переславль-Залесский, 2009), с. 195–205, URL http://skif.pereslavl.ru/psi-info/rcms/rcms-publications/2009-rus/r-zond.pdf ↑ 74.

[6] А. М. Бершадский, Л. С. Курилов, А. Г. Финогеев. «Исследование стратегий балансировки нагрузки в системах распределенной обработки данных», *Известия высших учебных заведений. Поволжский регион*, 2009, №4, с. 38–48 ↑ 74.

[7] И. А. Голубев. *Планирование задач в распределенных вычислительных системах на основе метаданных*, Дис. ... к.т.н., СПб., 2014, 135 с. ↑ 74, 77.

[8] Ж. Б. Кальпеева. *Модели и методы организации вычислительных процессов в распределенной облачной среде*, Дис. ... докт. философии (PhD), Республика Казахстан, Алматы, 2014, 136 с. ↑ 74.

[9] Е. Ю. Селиверстов. «Обзор методов решения задачи планирования параллельных алгоритмов», *Инженерный вестник*, 2014, №12, с. 541–555 ↑ 74.

[10] Н. В. Покусин. «Балансировка нагрузки распределенной гетерогенной вычислительной системы в условиях априорной неопределенности о характере входного потока заявок», *Науковедение*, 2013, №3, URL http://naukovedenie.ru/PDF/82tvn313.pdf ↑ 77.

[11] D. Agrawal, L. H. Jaiswal, I. Singh, K. Chandrasekaran. «An Evolutionary Approach to Optimizing Cloud Services», *Computer Engineering and Intelligent System*, **3**:4 (2012), с. 47–54 (english) ↑ 77.

[12] H. Zhao, R. Sakellariou. «A Hybrid Heuristic for DAG Scheduling on Heterogeneous Systems», 13th Heterogeneous Computing Workshop (HCW 2004), 2004 (english), URL http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.121.3069&rep=rep1&type=pdf ↑ 77.

[13] A. Amar, R. Bolze, E. Boix. *DIET 2.8 user's manual*, 2011 (english), URL http://graal.ens-lyon.fr/~diet/download/doc/UsersManualDiet2.8.1.pdf ↑ 79.

[14] B. Amedro, V. Bodnartchouk, L. Baduel. *ProActive Scheduling v.3.3.2 user's manual*, 2013 (english), 152 с. ↑ 80.

[15] *Moab Workload Manager v.7.2.4 Administrator Guide. Adaptive Computing Enterprises*, 2013 (english), 1136 с. ↑ 81.

[16] *Maui v.3.2 Administrator's Guide. Adaptive Computing Enterprises*, 2011 (english), 287 с. ↑ 82.

[17] А. А. Букатов, Г. М. Хачкинаев, «Разработка системы управления параллельными заданиями в гетерогенной вычислительной среде», *Труды первой Всероссийской научной конференции «Методы и средства обработки информации»* (Москва, 2003), с. 197–202 ↑ 82.

[18] Г. М. Хачкинаев. *Система пакетной обработки заданий в гетерогенной вычислительной сети*, Дис. ... к.т.н., М., 2005, 162 с. ↑ 82.

[19] Б. А. Телеснин. *Методы и средства организации обработки потоковой информации на распределенных гетерогенных вычислительных комплексах*, Дис. ... к.т.н., Ростов-на-Дону, 2009, 134 с. ↑ 83.

[20] И. В. Бычков, Г. А. Опарин, А. Г. Феоктистов, В. Г. Богданова, А. А. Пашинин. «Сервис-ориентированное управление распределенными вычислениями на основе мультиагентных технологий», *Известия Южного федерального университета. Технические науки*, 2014, №12, с. 17–27 ↑ 84.

[21] А. И. Миков, Е. Б. Замятина, А. А. Козлов, «Оптимизация параллельных вычислений с применением мультиагентной балансировки», *Труды международной научной конференции «Параллельные Вычислительные Технологии»* (Нижний Новгород, 2009), с. 599–604 ↑ 84.

[22] И. В. Бычков, Г. А. Опарин, А. Г. Феоктистов, А. Н. Кантер. «Мультиагентный алгоритм распределения вычислительных ресурсов на основе экономического механизма регулирования их спроса и предложения», *Вестник компьютерных и информационных технологий*, 2014, №1, с. 39–45 ↑ 84.

[23] А. А. Талалаев. «Организация конвейерно-параллельных вычислений для обработки потоков данных», *Информационные технологии и вычислительные системы*, 2011, №1, с. 8–13 ↑ 86.

[24] В. М. Хачумов, В. П. Фраленко. «Высокопроизводительная обработка изображений на кластерных устройствах», *Нейрокомпьютеры: разработка и применение*, 2012, №6, с. 38–46 ↑ 86.

[25] А. А. Талалаев, В. П. Фраленко. «Архитектура комплекса конвейерно-параллельной обработки данных в гетерогенной вычислительной среде», *Вестник РУДН. Серия Математика. Информатика. Физика*, 2013, №3, с. 113–117 ↑ 86.

[26] В. Ф. Заднепровский, А. А. Талалаев, И. П. Тищенко, В. П. Фраленко, В. М. Хачумов. «Программно-инструментальный комплекс высокопроизводительной обработки изображений медицинского и промышленного назначения», *Информационные технологии и вычислительные системы*, 2014, №1, с. 61–72 ↑ 86.

[27] В. М. Хачумов, И. П. Тищенко, А. А. Талалаев, К. А. Константинов, В. П. Фраленко, Ю. Г. Емельянова. *Нейросетевая система контроля телеметрической информации, диагностики подсистем космических аппаратов, обработки космических снимков (ПС НСКиД)*, Свидетельство о государственной регистрации программы для ЭВМ № 2012613261, дата приоритета: 18.11.2011, дата регистрации: 06.04.2012 ↑ 86.

[28] В. М. Хачумов, И. П. Тищенко, А. А. Талалаев, В. П. Фраленко, Д. Н. Степанов, А. А. Кондратьев, А. Е. Кирюшина. *Программно-инструментальный комплекс высокопроизводительных облачных конвейерно-параллельных вычислений (ПИК «Облако»)*, Свидетельство о государственной регистрации программы для ЭВМ № 2014610456, дата приоритета: 14.11.2013, дата регистрации: 10.01.2014 ↑ 86.

Рекомендовал к публикации *д.т.н., проф. В. М. Хачумов*

Об авторах:

Виталий Петрович Фраленко

К.т.н., старший научный сотрудник ИЦМС ИПС им. А.К. Айламазяна РАН. Область научных интересов: интеллектуальный анализ данных и распознавание образов, искусственный интеллект и принятие решений, параллельные алгоритмы, сетевая безопасность, диагностика сложных технических систем.

e-mail: alarmod@pereslavl.ru

Алексей Юрьевич Агроник

Аспирант МГТУ «СТАНКИН». Область научных интересов: информационно-телекоммуникационные системы, поисковые системы, полнотекстовый поиск, базы данных.

e-mail: aleksey@agronik.im

Пример ссылки на эту публикацию:

В. П. Фраленко, А. Ю. Агроник. «Средства, методы и алгоритмы эффективного распараллеливания вычислительной нагрузки в гетерогенных средах», *Программные системы: теория и приложения*, 2015, **6**:3(26), с. 73–92.

URL http://psta.psiras.ru/read/psta2015_3_73-92.pdf

Vitaly Fralenko, Alexey Agronik. *Tools, methods and algorithms for the efficient parallelization of computational loading in heterogeneous environments.*

Abstract. The article is devoted to the current state analysis of research in the field of algorithmic, mathematical and software support for distribution tasks on compute nodes in a heterogeneous environment. Proposed new classification of load balancing strategies: on the dynamics principle, on the management basis, on the universality basis, with forecasting / without predicting of system status and others. Investigated some load balancing methods, systems and complexes, including the following: method of the problem submission into a directed acyclic graph, scheduler model based on the metadata, "DIET", "ProActive", "Moab", "Maui", support system for the moldable jobs, complex stream processing in the terms of queuing theory and service-oriented approach. It allow to minimize devices downtime-computing, reduce volume and transmission time for data from one device to another, improve overall scalability, minimize data access time and so on. Identified the advantages and disadvantages, presented use proposals. *(In Russian).*

Key Words and Phrases: load balancing, computer, scheduler model, guidelines, support, algorithm.

References

[1] V. V. Voevodin, Vl. V. Voevodin. *Parallel Computing*, BHV-Peterburg, St. Petersburg, 2002 (in Russian), 608 p.

[2] J. Piernas, J. Nieplocha. "Active Storage User's Manual: Pacific Northwest National Laboratory", 2007, URL http://hpc.pnl.gov/active-storage/as_users_manual_october_2007.pdf.

[3] *Cascading — Application Platform for Enterprise Big Data*, URL http://www.cascading.org/.

[4] *Welcome to Pig!* URL http://hadoop.apache.org/pig/.

[5] E. O. Tyutliaeva, "Development and Implementation of Distributed Remote Sensing Data Storage", *Trudy XIII nauchno-prakticheskoj konferencii Universiteta goroda Pereslavlja* (Pereslavl-Zalessky, 2009), pp. 195–205 (in Russian), URL http://skif.pereslavl.ru/psi-info/rcms/rcms-publications/2009-rus/r-zond.pdf.

[6] A. M. Bershad, L. S. Kurilov, A. G. Finogeev. "Research of Load Balancing Strategies for Distributed Data Processing Systems", *University proceedings. Volga region*, 2009, no.4, pp. 38–48 (in Russian).

[7] I. A. Golubev. *Scheduling in Distributed Computing Systems Based on Metadata*, Diss. ... k.t.n., SPb., 2014 (in Russian), 135 p.

[8] Zh. B. Kalpeeva. *Models and Methods of Computing Processes Organization in a Distributed Cloud Environment*, Diss. ... dokt. filosofii (PhD), Respublika Kazahstan, Almaty, 2014 (in Russian), 136 p.

[9] E. Yu. Seliverstov. "Review of Methods for Solving the Parallel Algorithms Planning Problem", *Engineering Bulletin*, 2014, no.12, pp. 541–555 (in Russian).

[10] N. V. Pokusin. “Load Balancing in Distributed Heterogeneous Computing System with a Priori Indefinite Input Stream Nature”, *Naukovedenie*, 2013, no.3 (in Russian), URL http://naukovedenie.ru/PDF/82tvn313.pdf.

[11] D. Agrawal, L. H. Jaiswal, I. Singh, K. Chandrasekaran. “An Evolutionary Approach to Optimizing Cloud Services”, *Computer Engineering and Intelligent System*, **3**:4 (2012), pp. 47–54.

[12] H. Zhao, R. Sakellariou. “A Hybrid Heuristic for DAG Scheduling on Heterogeneous Systems”, 13th Heterogeneous Computing Workshop (HCW 2004), 2004, URL http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.121.3069&rep=rep1&type=pdf.

[13] A. Amar, R. Bolze, E. Boix. *DIET 2.8 user's manual*, 2011, URL http://graal.ens-lyon.fr/~diet/download/doc/UsersManualDiet2.8.1.pdf.

[14] B. Amedro, V. Bodnartchouk, L. Baduel. *ProActive Scheduling v.3.3.2 user's manual*, 2013, 152 p.

[15] *Moab Workload Manager v.7.2.4 Administrator Guide. Adaptive Computing Enterprises*, 2013, 1136 p.

[16] *Maui v.3.2 Administrator's Guide. Adaptive Computing Enterprises*, 2011, 287 p.

[17] A. A. Bukatov, G. M. Hachkinaev, “Development of Control System Parallel Tasks in a Heterogeneous Computing Environment”, *Trudy pervoj Vserossijskoj nauchnoj konferencii “Metody i sredstva obrabotki informacii”* (Moscow, 2003), pp. 197–202 (in Russian).

[18] G. M. Hachkinaev. *Packet Processing Tasks in a Heterogeneous Computer Network*, Diss. ... k.t.n., M., 2005 (in Russian), 162 p.

[19] B. A. Telesnin. *Methods and Tools for Organization of Streaming Information Processing by Distributed Heterogeneous Computer Systems*, Diss. ... k.t.n., Rostov-na-Donu, 2009 (in Russian), 134 p.

[20] I. B. Bychkov, G. A. Oparin, A. G. Feoktistov, V. G. Bogdanov, A. A. Pashinin. “Service-oriented Control of Distributed Computing Based on Multiagent Approach”, *Izvestiya SFedU. Engineering sciences*, 2014, no.12, pp. 17–27 (in Russian).

[21] A. I. Mikov, E. B. Zamyatin, A. A. Kozlov, “Optimization of Parallel Computing Using a Multi-agent Balancing”, *Trudy mezhdunarodnoj nauchnoj konferencii ¡¡Parallelnye Vychislitelnye Tehnologii¿¿* (Nizhny Novgorod, 2009), pp. 599–604 (in Russian).

[22] I. B. Bychkov, G. A. Oparin, A. G. Feoktistov, A. N. Kanter. “Multiagent Algorithm for Resources Allocation Based on the Economic Mechanism of Regulating their Supply and Demand”, *Vestnik kompjuternyh i informacionnyh tehnologij*, 2014, no.1, pp. 39–45 (in Russian).

[23] A. A. Talalaev. “Organization of Pipeline-parallel Computing to Process Data Streams”, *Informacionnye tehnologii i vychislitelnye sistemy*, 2011, no.1, pp. 8–13 (in Russian).

[24] V. M. Khachumov, V. P. Fralenko. “High Performance Image Processing on a Cluster Computers”, *Nejrokompjutery: razrabotka, primenenie*, 2012, no.6, pp. 38–46 (in Russian).

[25] A. A. Talalaev, V. P. Fralenko. “The Architecture of a Parallel-pipeline Data Processing Complex for Heterogeneous Computing Environment”, *Vestnik*

RUDN. Serija Matematika. Informatika. Fizika, 2013, no.3, pp. 113–117 (in Russian).

[26] V. F. Zadneprovsky, A. A. Talalaev, I. P. Tishchenko, V. P. Fralenko, V. M. Khachumov. "Integrated Development Environment for High-performance Medical and Industrial Purpose Images Processing", *Informacionnye tehnologii i vychislitelnye sistemy*, 2014, no.1, pp. 61–72 (in Russian).

[27] V. M. Khachumov, I. P. Tishchenko, A. A. Talalaev, K. A. Konstantinov, V. P. Fralenko, Yu. G. Emelyanova. *Neural Network System for Control of Telemetry Data, Spacecraft Subsystems Diagnosing, Satellite Images Processing (PS NSKiD)*, Svidetelstvo o gosudarstvennoj registracii programmy dlja JeVM no. 2012613261, data prioriteta: 18.11.2011, data registracii: 06.04.2012 (in Russian).

[28] V. M. Khachumov, I. P. Tishchenko, A. A. Talalaev, V. P. Fralenko, D. N. Stepanov, A. A. Kondratiev, A. E. Kiryushina. *Software-tool Complex for High Cloud Pipeline-parallel Computing (PIC "Oblako")*, Svidetelstvo o gosudarstvennoj registracii programmy dlja JeVM no. 2014610456, data prioriteta: 14.11.2013, data registracii: 10.01.2014 (in Russian).

Sample citation of this publication:

Vitaly Fralenko, Alexey Agronik. "Tools, methods and algorithms for the efficient parallelization of computational loading in heterogeneous environments", *Program systems: theory and applications*, 2015, **6**:3(26), pp. 73–92. (*In Russian.*)

URL http://psta.psiras.ru/read/psta2015_3_73-92.pdf