УДК 004.75

ИССЛЕДОВАНИЕ ВОПРОСОВ БАЛАНСИРОВКИ НАГРУЗКИ В ОБЛАЧНЫХ ВЫЧИСЛЕНИЯХ

Петренко Т.В., магистрант; Червинский В.В., доц., к.т.н., доц.
(ГОУ ВПО «Донецкий национальный технический университет», г. Донецк, ДНР)

В настоящее время облачные вычисления называют новым трендом в развитии информационных технологий. Переход от уникальности к масштабности позволяет воспринимать вычислительные мощности уже не как отдельный компьютер-сервер, стоящий в здании организации, а как услугу, которая предоставляется неким далеким датацентром. На рынке облачных вычислений присутствуют не только общеизвестные решения, такие как VMware ESX, Xen, Distributed Resource Scheduler и другие, но и хорошо документированные комплексы с открытым исходным кодом, такие как OpenStack

Особенности облачных вычислений:

- нагрузка на ключевые ресурсы носит периодический и неравномерный характер;
- одновременно происходят обращения к нескольким типам ресурсов;
- интенсивность обращения к каждому ресурсу может изменяться в зависимости от внешних условий;
- ввиду отсутствия распределения нагрузки между ресурсами при пиковой нагрузке оборудование не всегда позволяет обслужить все запросы;
- до 90% нагрузки предопределены, поскольку для доступа к ресурсам используется предварительная регистрация.

Кроме того, стоит отметить, что 80% ресурсов востребованы лишь в 20% времени работы сервисов.

В настоящее время существующие решения, построенные на базе облачных сервисов, используют универсальный подход для организации доступа к размещаемым в них ресурсам.

При проектировании и создании облачных решений важное место занимает обеспечение надежности и оптимальности использования ресурсов – мониторинг**,** или получение информации о доступности и загрузке аппаратных ресурсов платформ с копиями распределенного приложения, и балансировка**,** то еть распределение поступающих пользовательских запросов между имеющимися аппаратными платформами.

Можно выделить следующие классы решений, используемые при построении многоузловых систем (рис. 1): балансировка с пропуском трафика через одно устройство балансировки; балансировка средствами кластера; балансировка без пропуска трафика через одно устройство балансировки.

Рассмотрим новый подход к обнаружению перегруженности или недогруженности хоста, основанный на долгосрочных прогнозах использования ресурсов, которые учитывают неопределенность в прогнозировании и накладные расходы живой миграции.

Балансировка нагрузки в облачных вычислениях.

Виртуальные машины дают возможность распределять ресурсы динамически в соответствии с требованиями, оптимизируя производительность приложений и потребление электроэнергии.

Для динамического перераспределения ресурсов существует ряд возможностей решения, одной из основных является живая миграция виртуальных машин. Она позволяет облачным провайдерам перемещать виртуальные машины с перегруженных хостов, поддерживая их производительность при заданном SLA и динамически консолидировать виртуальные машины на наименьшем числе хостов, чтобы экономить электроэнергию при низкой загрузке. Используя «живую» миграцию и применяя онлайн-алгоритмы, которые позволяют принимать решения о миграции в реальном времени, можно эффективно управлять облачными ресурсами, адаптируя распределение ресурсов к нагрузкам VM

(виртуальных машин), поддерживая уровни производительности VM в соответствии с SLA и снижая энергопотребление инфраструктуры.

Рисунок 1 – Принципы балансировки нагрузки: а) балансировка с пропуском трафика через одно устройство балансировки; б) балансировка средствами кластера; в) балансировкабез пропуска трафика через одно устройство балансировки

Важной проблемой в контексте живой миграции является обнаружение состояния перегрузки или недогрузки хоста. Большинство современных подходов основаны на мониторинге использования ресурсов, и, если фактическое или прогнозируемое следующее значение превышает заданный порог, узел объявляется перегруженным. Тем не менее, живая миграция имеет свою цену, обоснованную нарушением производительности VM в процессе миграции. Проблема с существующими подходами заключается в том, что обнаружение перегрузки хоста по одному замеру использования ресурсов или нескольким будущим значениям может привести к поспешным решениям, излишним накладным расходам на живую миграцию и проблемам со стабильностью VM.

Более перспективным является подход принятия решений о живой миграции на основе прогнозов использования ресурсов на несколько шагов вперед. Это не только повышает стабильность, так как миграционные действия начинаются только, когда нагрузка сохраняется в течение нескольких временных интервалов, но также позволяет облачным провайдерам прогнозировать состояние перегрузки до того, как это произойдёт. С другой стороны, прогнозирование более отдалённого будущего увеличивает ошибку прогнозирования и неопределенность, уменьшая при этом преимущества долгосрочного прогнозирования. Ещё одна важная проблема заключается в том, что живая миграция должна выполняться только в том случае, если штраф за возможные нарушения SLA превышает накладные расходы на миграцию.

Рассмотрим управление облаком IaaS, в котором несколько виртуальных машин работают на нескольких физических узлах. Общая архитектура диспетчера ресурсов и его основных компонентов показана на рис. 2. Существует агент VM для каждой виртуальной машины, который определяет распределение ресурсов на своей виртуальной машине в каждом временном интервале. Для каждого хоста есть хост-агент, который получает решения о распределении ресурсов всех агентов VM и определяет окончательные распределения, разрешая любые возможные конфликты. Он также обнаруживает, когда узел перегружен или недогружен, и передает эту информацию глобальному агенту. Глобальный агент инициирует решения о миграции виртуальной машины, перемещая виртуальные машины от перегруженных или недогруженных хостов на консолидирующие хосты для уменьшения потерь за нарушение SLA и сокращения количества физических узлов.

Агент виртуальной машины (VM agent) отвечает за локальные решения о распределении ресурсов путем динамического определения общих ресурсов, которые должны быть распределены на его собственной виртуальной машине. Решения о

распределении выполняются в дискретных временных интервалах, где в каждом интервале определяется доля ресурсов в следующем временном интервале. В этой работе временной интервал устанавливается в 10 секунд, чтобы быстро адаптироваться к изменяющейся нагрузке. Интервал не установлен на менее чем 10 секунд, поскольку при долгосрочном прогнозировании это увеличило бы количество шагов времени для прогнозирования в будущем, снижая точность прогноза. Установка большего интервала времени может привести к неэффективности и нарушениям SLA из-за отсутствия быстрой адаптации к изменению нагрузки. Наша работа сосредоточена на распределении CPU, но в принципе подход может быть распространен и на другие ресурсы. Для распределения ресурсов процессора используется настройка CPU CAP, которую предлагает большинство современных технологий виртуализации. CAP — это максимальная мощность CPU, которую может использовать VM, в процентах от общей ёмкости, что обеспечивает хорошую изоляцию производительности между виртуальными машинами.

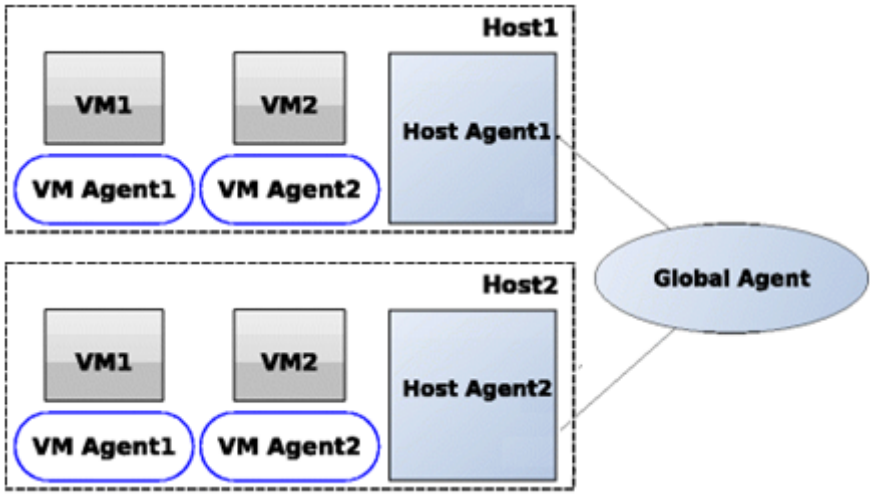

Рисунок 2 – Архитектура диспетчера ресурсов

Чтобы оценить долю CPU, выделенную каждой VM, сначала прогнозируется значение использования CPU для следующего временного интервала. Затем общий ресурс CPU рассчитывается как прогнозируемое использование CPU плюс 10% от мощности CPU. Установив CPU CAP на 10% больше, чем требуемое использование CPU, мы можем учесть ошибки прогнозирования и уменьшить вероятность нарушений SLA, связанных с производительностью. Для прогнозирования следующего значения использования CPU используется техника прогнозирования временных рядов, основанная на истории предыдущих значений использования CPU. В частности, используется машинное обучение, основанное на гауссовских процессах. Хотя для локального распределения ресурсов необходим только один шаг вперед, наш VM-агент прогнозирует на несколько шагов вперёд с целью обнаружения перегрузки.

Одна из обязанностей хост-агента — играть роль арбитра. Он получает требования к процессору от всех агентов виртуальных машин, и, разрешая любые конфликты между ними, он решает об окончательных распределениях CPU для всех виртуальных машин. Конфликты могут возникать, когда требования CPU всех VM превышают общую мощность CPU. Если конфликты отсутствуют, окончательное распределение CPU совпадает с распределением, запрашиваемым агентами виртуальной машины. Если есть конфликт, главный агент вычисляет окончательные распределения CPU в соответствии со следующей формулой:

Финальная квота VM = (Запрашиваемая доля VM/Сумма запрашиваемых ресурсов всеми VM) * Общая мощность CPU

Другая обязанность хост-агента, которая является основным объектом этой работы, заключается в определении факта перегруженности или недогруженности хоста. Эта информация передается глобальному агенту, который затем инициирует живую миграцию

для перемещения VM от перегруженных или недогруженных хостов в соответствии с алгоритмом глобального распределения.

Обнаружение перегрузки.

Для обнаружения перегрузки используется долгосрочное прогнозирование временных рядов. В контексте этой работы это означает предсказание значений для 7 временных интервалов в будущем. Хост объявляется перегруженным, если фактическое и прогнозируемое общее использование CPU для 7 временных интервалов в будущем превышают порог перегрузки. Прогнозируемое общее использование CPU для будущего временного интервала оценивается путем суммирования прогнозируемых значений использования CPU всеми VM в соответствующем временном интервале. Значение 7 временных интервалов прогнозирования в будущем выбирается таким образом, чтобы оно было больше расчетного среднего времени живой миграции (около 4 временных интервалов). Среднее время живой миграции считается известным, и его значение, равное четырем временным интервалам, оценивается путем усреднения по всем временам живой миграции VM в течение нескольких экспериментов. В реальных сценариях это значение заранее неизвестно, но его можно оценить на основе опыта. Другим более тонким подходом было бы применение метода прогнозирования времени живой миграции на основе текущих параметров VM.

Выполнение действий живой миграции для перегрузок, которые длятся меньше времени живой миграции, бесполезно, поскольку в этом случае живая миграция не сокращает время перегрузки. Использовать значение большее, чем 7 временных интервалов, тоже не очень полезно, поскольку это может привести к пропуску некоторых состояний перегрузки, которые не будут длиться долго, но которые могут быть устранены с помощью живой миграции. Некоторые предварительные эксперименты показали, что увеличение числа временных интервалов предсказания в будущем не повышает стабильность и эффективность подхода. Значение порога перегрузки определяется динамически на основе количества виртуальных машин и зависит от системы штрафов за нарушения SLA, что подробнее описано далее.

Обнаружение недостаточной загруженности.

Хост-агент также определяет недостаточную загруженность хоста для применения динамической консолидации путём живой миграции всех своих виртуальных машин на другие узлы и отключения хоста для экономии электроэнергии. Здесь также используются долгосрочные прогнозы временных рядов использования CPU. Хост объявляется недогруженным, если фактическое и прогнозируемое общее использование CPU в течение 7 временных интервалов в будущем ниже порога недогрузки. Опять же, значение 7 временных интервалов достаточно, чтобы пропускать кратковременные состояния недогрузки, но не слишком велико, чтобы упустить любую возможность для консолидации. Пороговое значение является постоянным значением, и в этой работе оно установлено на 10% от мощности CPU, но оно может быть настроено администратором в соответствии с его предпочтениями в отношении агрессивности консолидации.

Обнаружение консолидирующих хостов.

Чтобы принять решения о живой миграции, глобальному агенту необходимо знать хосты, которые не перегружены, чтобы использовать их в качестве хостов назначения для миграций. Хост объявляется как консолидирующий, если фактическое и прогнозируемое общее использование CPU в течение 7 временных интервалов в будущем меньше порога перегрузки. Фактическое и прогнозируемое общее использование CPU любого временного интервала оценивается путем суммирования фактического и прогнозируемого использования CPU всеми существующими виртуальными машинами, а также фактического и прогнозируемого использования CPU VM, которые планируется перенести на хост назначения. Цель состоит в том, чтобы проверить, не станет ли перегруженным косолидирующий узел после перенастройки виртуальных машин.

Неопределённость в долгосрочном прогнозировании.

Долгосрочные прогнозы несут в себе ошибки предсказания, которые могут приводить к ошибочным решениям. Чтобы учесть неопределенность в долгосрочных прогнозах, вышеупомянутые механизмы обнаружения дополняются вероятностной моделью распределения ошибки прогнозирования.

Сначала оценивается функция плотности вероятности для ошибки предсказания в каждом интервале прогнозирования. Поскольку распределение вероятности ошибки предсказания заранее неизвестно, а различные нагрузки могут иметь разные распределения, для построения функции плотности требуется непараметрический метод. В этой работе используется непараметрический метод оценки функции плотности вероятности, основанный на ядерной оценке плотности ядра. Он оценивает функцию плотности вероятности ошибки прогнозирования каждый интервал времени, основываясь на предыдущих ошибках прогнозирования. Например, функция плотности вероятности абсолютного значения ошибки предсказания.

Глобальный агент принимает решения о распределении ресурсов провайдера с помощью живых миграций виртуальных машин с перегруженных или недогруженных хостов на другие узлы для снижения нарушений SLA и потребления энергии. Он получает уведомления от хост-агента, если узел будет перегружен или недогружен в будущем, и выполнит перенос VM, если оно того стоит.

Глобальный агент применяет алгоритм Power Aware Best Fit Decreasing (PABFD) для размещения VM со следующими корректировками. Для обнаружения перегрузки или недогрузки используются наши подходы, представленные выше. Для выбора виртуальной машины используется политика минимального времени миграции (MMT), но с модификацией, что для миграции в каждом раунде принятия решений выбрана только одна виртуальная машина, даже если хост может оставаться перегруженным после миграции. Это делается для уменьшения количества одновременных виртуальных миграций виртуальных машин и связанных с ними затрат. Для процесса консолидации, рассматриваются только недогруженные хосты, которые обнаруживаются предлагаемыми подходами на основе долгосрочного прогнозирования. Из списка недогруженных хостов сначала рассматриваются те, которые имеют более низкую среднюю загрузку CPU.

Таким образом, проанализированы вопросы балансировки нагрузки в сфере облачных вычислений. Рассмотрен подход к обнаружению перегруженности или недогруженности хоста, основанный на долгосрочных прогнозах использования ресурсов, которые учитывают неопределенность в прогнозировании и накладные расходы живой миграции. Показано, что основными направлениями исследований в данной области являются: комбинирование методов обнаружения всплесков нагрузки с использованием методов прогнозирования нагрузки для разработки широкого спектра моделей нагрузки; дополнение используемой в настоящее время схемы предсказания следующего значения загрузки CPU для локального распределения ресурсов более сложными схемами, основанными на теории управления, нечеткой логике и т.д.; распределение ресурсов, при котором каждый хост-агент принимает решения о живой миграции в сотрудничестве с ближайшими хост-агентами.

Перечень ссылок

1. Петров, Д. Л. Оптимальный алгоритм миграции данных в масштабируемых облачных хранилищах / Д. Л. Петров // Управление большими системами. 2010. - №. 30 - С.180-197.

2. Клепиков, А. К. Модель распределения ресурсов при "облачных вычислениях" / А. К. Клепиков, А. Н. Привалов // Известия Тульского государственного университета. Технические науки. - 2012. - С. 151 - 157.

3. Копысов, С. П. Динамическая балансировка нагрузки для параллельного распределённого МДО / С. П. Копысов. – Москва : Изд-во МГУ, 2003. С. 222. – 228.

4. Петров, Д. Л. Динамическая модель масштабируемого облачного хранилища данных / Д. Л. Петров // Известия ЛЭТИ. – 2010. - №4 - С. 17-21.