УДК621.39

ПАРАЛЛЕЛЬНЫЕ И РАСПРЕДЕЛЁННЫЕ СИСТЕМЫ В ОБЛАЧНЫХ ТЕХНОЛОГИЯХ

Карпенёв А.С., магистрант; Молоковский И.А., доц., к.т.н., доц.
(ГОУ ВПО «Донецкий национальный технический университет», г. Донецк, ДНР)

Облачные вычисления тесно связаны с параллельными и распределенными вычислениями. Облачные приложения основаны на парадигме клиент-сервер с относительно простым программным обеспечением, тонким клиентом, работающим на машине пользователя, в то время как вычисления выполняются в облаке. Многие облачные приложения интенсивно работают с данными и используют несколько экземпляров, которые запускаются одновременно. Системы обработки транзакций, такие как веб-сервисы, представляют собой большой класс приложений, размещенных при вычислении облаков; такие приложения запускают несколько экземпляров службы и требуют надежной и упорядоченной доставки сообщений.

Коммуникационные протоколы поддерживают координацию распределенных процессов и транспортной информации через «шумные» и ненадежные каналы связи, которые могут потерять сообщения или предоставить дубликаты, искаженные или неуправляемые сообщения. Для обеспечения надежной и упорядоченной доставки сообщений протоколы маркируют каждое сообщение порядковым номером; в свою очередь, приемник отправляет подтверждение с его собственным порядковым номером, чтобы подтвердить получение сообщения. Часы отправителя и получателя могут не синхронизироваться, поэтому эти порядковые номера действуют как логические часы. Таймауты используются для запроса повторной передачи потерянных или задержанных сообщений.

Понятие непротиворечивых сокращений и распределенных снимков лежит в основе процедур перезапуска с контрольной точки для длительных вычислений. Действительно, многие вычисления облаков являются интенсивными данными и выполняются в течение продолжительных периодов времени на нескольких компьютерах в облаке. Контрольные точки берутся периодически в ожидании необходимости перезапуска программного процесса, когда одна или несколько систем выходят из строя; когда происходит сбой, вычисление перезапускается с последней контрольной точки, а не с начала. Многие функции компьютерного облака требуют информации, предоставляемой средствами мониторинга, системными компонентами, которые собирают информацию о состоянии из отдельных систем.

Параллельные вычисления позволяют решать большие проблемы, разбивая их на более мелкие и одновременно разрешая их. Параллельные аппаратные и программные системы позволяют нам решать проблемы, требующие больше ресурсов, чем те, которые предоставляются единой системой, и в то же время сократить время, необходимое для получения решения. Ускорение измеряет эффективность параллелизации; в общем случае ускорение параллельного вычисления определяется как:

$$S(N) = \frac{T(1)}{T(N)}, \qquad (1)$$

где $T(1)$ - время выполнения последовательного вычисления;

$T(N)$ - время выполнения при N параллельных вычислений.

Потенциальное ускорение параллельных вычислений определяется законом Амдала. Он утверждает, что часть вычисления, которая не может быть распараллелена, определяет общее ускорение. Если α - это доля времени выполнения, которое последовательная программа расходует на непараллельные сегменты вычисления, то:

$$S = \frac{1}{\alpha}. \quad (2)$$

Закон Амдала применяется к фиксированному размеру проблемы. В этом случае объем работы, назначенной каждому из параллельных процессов, уменьшается, когда число процессов увеличивается, и это влияет на эффективность параллельного выполнения. Когда размер проблемы разрешен для изменения, масштабированное ускорение с N параллельными процессами представлено законом Густафсона:

$$S(N) = N - \alpha(N - 1). \quad (3)$$

Закон Амдала, выраженный формулой 1, и масштабированное ускорение, данное уравнением 3, предполагают, что всем процессам присваивается одинаковый объем работы. Масштабированное ускорение предполагает, что объем работы, назначенный каждому процессу, будет одинаковым, независимо от размера задачи. Затем, чтобы поддерживать одинаковое время выполнения, количество параллельных процессов должно увеличиваться с масштабированием задачи. Масштабированное ускорение отражает сущность эффективности, а именно то, что ограничения последовательной части кода можно сбалансировать, увеличив размер задачи.

Координация параллельных вычислений может быть довольно сложной и требует существенных расходов, что в конечном итоге уменьшает ускорение этих вычислений. Часто параллельное вычисление включает многоступенчатые этапы, и все параллельные действия должны заканчиваться на один этап раньше до начала выполнения следующего; эта барьерная синхронизация дополнительно уменьшает ускорение.

Можно использовать параллелизм на разных уровнях:

– билинейный параллелизм. Количество бит, обработанных за такт, часто называемое размером слова, постепенно увеличивалось с 4-разрядных процессоров до 8-разрядных, 16-разрядных, 32-разрядных и с 2004 года 64-разрядных. Это уменьшило количество инструкций, необходимых для обработки больших операндов, и позволило значительно повысить производительность. Во время этого эволюционного процесса число битов адреса также увеличилось, что позволяет инструкциям ссылаться на большее адресное пространство;

– параллельность на уровне инструкций. Сегодняшние компьютеры используют многоступенчатые конвейеры для ускорения выполнения. Как только n-этапный конвейер будет заполнен, инструкция будет завершена в каждом такте. «Классический» конвейер архитектуры Reduced Instruction Set Computing (RISC) состоит из пяти этапов: выбор команды, декодирование команд, выполнение команд, доступ к памяти и запись;

– параллелизм данных или параллельность цикла. Циклы программы могут обрабатываться параллельно;

– заданный параллелизм. Проблема может быть разложена на задачи, которые могут выполняться одновременно. Широко распространенный тип параллелизма задач - это парадигма Same Program Multiple Data (SPMD). Как следует из названия, отдельные процессоры выполняют одну и ту же программу, но на разных сегментах входных данных. Зависимости данных вызывают разные потоки управления в отдельных задачах.

Распределенная система представляет собой набор автономных компьютеров, которые подключаются через сетевое и программное обеспечение распространения, называемое промежуточным программным обеспечением, которое позволяет компьютерам координировать свои действия и совместно использовать ресурсы системы. Пользователи распределенной системы воспринимают систему как единый интегрированный вычислительный объект.

Распределенная система имеет несколько характеристик: ее компоненты автономны, планирование и другие политики управления ресурсами и безопасности реализуются каждой системой, есть несколько точек управления и несколько точек отказа, и ресурсы могут быть

недоступны в любое время. Распределенные системы могут масштабироваться путем добавления дополнительных ресурсов и могут быть разработаны для обеспечения доступности даже при низком уровне надежности аппаратного / программного обеспечения / сети.

Среднее ПО должно поддерживать набор желательных свойств распределенной системы:

– доступность прозрачности. Доступ к локальным и удаленным информационным объектам осуществляется с использованием идентичных операций;

– прозрачность местоположения. Доступ к информационным объектам осуществляется без знания их местонахождения;

– прозрачность параллелизма. Несколько процессов запускаются одновременно с использованием общих информационных объектов без помех между ними;

– прозрачность репликации. Несколько экземпляров информационных объектов используются для повышения надежности без знания пользователей или приложений;

– прозрачность прозрачности. Сокрытие недостатков;

– прозрачность миграции. Информационные объекты в системе перемещаются, не затрагивая выполняемую им операцию;

– прозрачность производительности. Система может быть переконфигурирована на основе нагрузки и качества обслуживания;

– масштабирование прозрачности. Система и приложения могут масштабироваться без изменения структуры системы и не влиять на приложения.

Чтобы понять важные свойства распределенных систем, используют модель, абстракцию, основанную на двух критических компонентах: процессах и каналах связи. Процесс - это программа в исполнении, а поток - легкий процесс. Поток выполнения - это наименьшая единица обработки, которая может быть запланирована операционной системой.

Процесс характеризуется своим состоянием. Такое состояние представляет собой некоторое количество информации, необходимое для перезапуска процесса после его приостановки. Событие – это изменение состояния процесса. События, влияющие на состояние процесса P_1, последовательно нумеруются как $e_i^1, e_i^2, e_i^3, \ldots$, как показано на пространственно – временных диаграммах (рис. 1, рис. 2, рис. 3). Процесс P_1 находится в состоянии σ_i^j сразу после появления события e_i^j и остается в этом состоянии до появления следующего события e_i^{j+1}.

Временные диаграммы отображают локальные и коммуникационные события в течение жизненного цикла процесса. Местные события – маленькие черные круги. Коммуникационные события в разных процессах связаны линиями, происходящими в событии отправки и заканчивающимися стрелкой в событии приема:

1. Все события в случае одного процесса P_1 являются локальными; процесс находится в состоянии σ_1 сразу после появления события e_1^1 и остается в этом состоянии до появления события e_1^2:

Рисунок 1 – Пространственно-временная диаграмма (1 процесс)

2. Два процесса P_1 и P_2; событие e_1^2 является событием связи, P_1 отправляет сообщение P_2; событие e_2^3 является событием связи, процесс P_2 принимает сообщение, отправленное P_1.

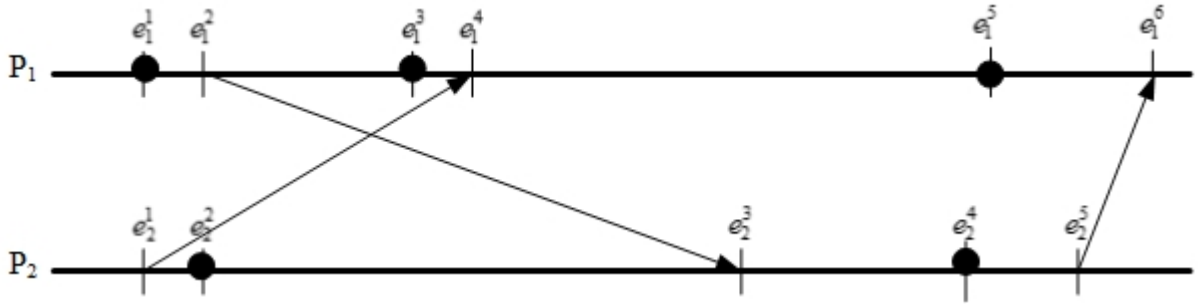

Рисунок 2 – Пространственно-временная диаграмма (2 процесса)

3. Три процесса взаимодействуют посредством событий связи.

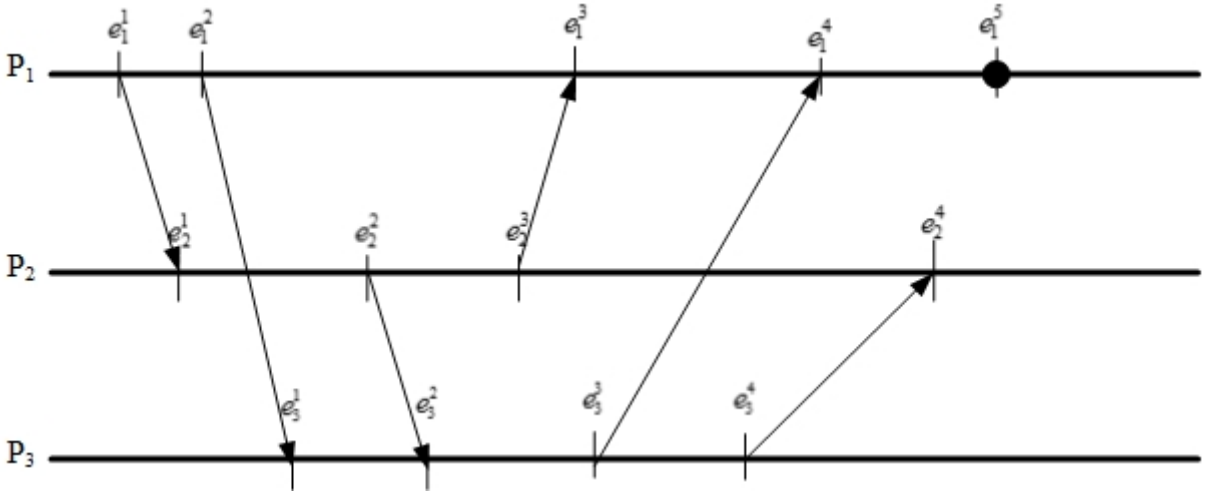

Рисунок 3 – Пространственно-временная диаграмма (3 процесса)

Группа процессов представляет собой совокупность взаимодействующих процессов. Данные процессы работают сообща и общаются друг с другом, чтобы достичь общей цели. Процессы в группе должны взаимодействовать друг с другом до тех пор, пока общие граничные значения, вычисленные одним процессом, не совпадут с общими граничными значениями, вычисленными другим. Многие проблемы в распределенных системах являются примерами глобальной задачи оценки предикатов (GPE), целью которой является оценка булевого выражения, элементы которого являются функциями глобального состояния системы.

Технологии, рассмотренные в этой статье, широко применяются в облачных технологиях, их применение позволяет достичь высоких результатов в скорости обработки данных и надёжности системы. Данные системы будут внедрены и будут использоваться в магистерской диссертации.

Перечень ссылок

1. Marinescu, D. C. Cloud Computing: Theory and Practice / Dan C. Marinescu. – Newnes, 2013.

2. Mell, P. The NIST Definition of Cloud Computing : Recommendations of the National Institute of Standards and Technology / P. Mell, T. Grance. – 2011, sp. 800-145.

3. Яремко, І. М. Імовірнісні характеристики центрів обробки даних і резервування / І. М. Яремко, В. В. Турупалов, І. О. Молоковский // Наукові праці інституту проблем модулювання в енергетиці ім. Г.Є. Пухова «Моделювання та інформаційні технології». – Київ, 2011 р. - Випуск 60. – С.141-146.