УВЕЛИЧЕНИЕ СКОРОСТИ ОБРАБОТКИ ДАННЫХ НА ОСНОВЕ БАЗЫ ДАННЫХ ORACLE И МОДЕЛИ MAPREDUCE

Тюменцев А. Д.

Белорусский государственный университет информатики и радиоэлектроники
г. Минск, Республика Беларусь

Бондарик В. М. - доцент

Аннотация. Описан один из возможных способов масштабирования базы данных с использованием модели MapReduce для увеличения скорости обработки данных, снижения нагрузки на сервера и повышения производительности системы в целом. Выявлены как положительные стороны данного подхода, так и отрицательные. Описано применение данной модели на практике в приложении-прототипе.

В современных условиях организации в процессе своего функционирования создают большое количество неструктурированных данных, таких как текстовые документы, изображения, видеозаписи, машинные коды, таблицы и т. д. Вся эта информация хранится во множестве репозиториев, порой даже за пределами организации. Компании могут иметь доступ к огромному массиву собственных данных и не иметь необходимых инструментов, которые могли бы установить взаимосвязи между этими данными и сделать на их основе значимые выводы. Традиционные методы анализа информации не могут угнаться за огромными объемами постоянно растущих и обновляемых данных, что в итоге и открывает дорогу технологиям Big Data.

Большие данные (Big Data) – обозначение структурированных и неструктурированных данных огромных объёмов и значительного многообразия, эффективно обрабатываемых горизонтально масштабируемыми программными инструментами.

Горизонтальное масштабирование предполагает под собой увеличение количества вычислительных ресурсов за счёт увеличения количества серверов. Вертикальное масштабирование идёт по принципу обновления оборудования для сервера: установка более производительных жёстких дисков, увеличение объёма оперативной памяти и т.д. Если при работе с приложением пользователь обратился к большому объёму данных и вычислительный сервер не справляется с нагрузкой, то время отклика может заметно увеличиться. Чтобы решить эту проблему используется модель MapReduce [1].

MapReduce – это модель распределённой обработки данных, предложенная компанией Google для обработки больших объёмов данных на компьютерных кластерах.

Предполагается, что данные организованы в виде некоторых записей. Поэтому обработка данных происходит в две стадии:

1. Стадия Map. На этой стадии данные предобрабатываются при помощи функции Map. Работа этой стадии заключается в предобработке и фильтрации данных. Функция Map, применённая к одной входной записи, выдаёт множество пар ключ-значение. Множество – т.е. может выдать только одну запись, может не выдать ничего, а может выдать несколько записей.
2. Стадия Reduce. На данной стадии данные, полученные из функции Map будут обработаны и возвращены в соответствии с изначально сформулированной задачей.

Данная модель была применена при разработке приложения, позволяющего автоматизировать процесс перевозки лекарственных препаратов и приборов медицинского назначения, а именно: определение класса опасности, номера ООН, упаковки, маркеров, контейнеров, заполнение сопроводительных документов. Эффективность использования предложенной модели обусловлена большим количеством данных, которые необходимо фильтровать и возвращать, что занимает довольно значительную часть времени при обработке запроса [2].

При реализации проекта на стороне WCF сервисов был создан класс, в котором реализованы две функции map и reduce. Функция map принимает в качестве параметров: искомое значение, т.е. ключ и коллекцию данных, которую необходимо обработать. Коллекция данных имеет динамический тип, поэтому может быть использована для любого набора данных. На выходе функция вернёт коллекцию ключ-значение. Далее данные будут переданы в функцию Reduce. Данная функция реализует в себе фильтрацию данных в зависимости от поставленной задачи. В нашем случае – это разбиение данных на части для корректного представления их в таблице, применение фильтров, выбранных в таблице, поиск заданного значения. Данные функции позволили снизить время обработки запроса практически вдвое. Далее необходимо было реализовать конвейер, которые

позволил использовать данные функции асинхронно, используя все ядра процессора. Для этого был реализован конвейер ASP.NET с помощью средств виртуальных серверов IIS.

Встроенный веб-сервер IIS очень легко масштабируется. Но эти технологии не ограничиваются просто обслуживанием веб-страниц и размещением веб-сайтов. Нет никакой технической причины, по которой его нельзя использовать в качестве механизма конвейера общего назначения, доступного через HTTP. Шаги конвейера ASP.NET выполняются последовательно (не переходя к следующему шагу, пока не завершится предыдущий), но каждый шаг может выполняться асинхронно. Веб-сервер IIS можно настроить для запуска нескольких конвейеров ASP.NET (несколько w3wp.exe), обслуживающих HTTP-запросы. Исходя из этого наши MapReduce запросы могут выполняться асинхронно, тем самым снижая время на обработку данных [3].

Для реализации асинхронной обработки данных созданный нами класс был перемещён в библиотеку, которую можно использовать во всех проектах, а наш WCF сервис был помещён на несколько пулов (процессов w3wp.exe). Библиотека была подключена как модуль, вызываемый с помощью атрибута при обработке определённых REST-запросов, которые мы сами указываем при определении сигнатур функций для сервисов. Этого достаточно, чтобы запустить конвейер ASP.NET и обрабатывать запросы асинхронно, разделяя данные между несколькими процессами и агрегируя их уже на уровне бизнес-логики. Чтобы добиться полноценной работы модели MapReduce необходимо реализовать так называемую «сетку». Для этого было добавлено ещё несколько виртуальных серверов. Чем больше «сетка», тем серьёзней проблему можно решить при обработке данных, разбивая её на более мелкие составляющие, которые нужно решить, и тем выше уровень параллелизма, который потенциально может быть достигнут. Асинхронный конвейер ASP.NET, объединённый с несколькими конвейерами на сервер, обеспечивает параллелизм в ядрах одного сервера. Поскольку серверы IIS - это просто автономные серверы, дополнительная настройка не требуется. Ещё одно преимущество архитектурной серверной «сетки» заключается в том, что она не полагается на главный узел для работы. В таких продуктах, как Hadoop, главный узел управляет кластером серверов и расположением данных в этом кластере. И это главный узел, который был бы источником сбоя. В предложенной нами структуре в «сетке» серверов нет главного узла. Любой серверный узел может инициировать запрос MapReduce [4].

После настройки конвейера, время на обработку запроса было снижено ещё вдвое.

Проблема данной модели заключается в том, что функция подразумевает под собой всегда полное сканирование данных без использования индексов. Это означает, что данная модель плохо применима, если ответить требуется очень быстро.

Ключевым преимуществом MapReduce является то, что модель позволяет «масштабировать» вместо «масштабирования». Другими словами, вы просто добавляете больше обычных серверных узлов, в нашем случае виртуальных серверов, а не приобретаете лучшее оборудование для одного основного узла. Соответственно, при сбое любого из серверов, обработка данных может быть передана на любой другой сервер при условии, что входные данные для проводимой операции доступны. Данный случай недоступен, если всей обработкой данных занимается только один серверный узел.

Одним из достоинств проекта является масштабируемость, полученная с использованием конвейера ASP.NET в качестве конвейера MapReduce. Поскольку конвейер ASP.NET работает последовательно, он подходит для выполнения шагов «Map» и «Reduce». Положительным моментом является то, что, хотя конвейер последователен и не будет двигаться к следующему шагу, пока не завершится предыдущий шаг, каждый шаг может выполняться асинхронно. Это позволяет конвейеру продолжать получать и обрабатывать новые запросы MapReduce, даже когда конвейер заблокирован, ожидая, что вызовы Map возвратятся с других распределенных серверных узлов.

Функция MapReduce позволила увеличить скорость обработки данных в нашем приложении по автоматизации процесса перевозки лекарственных средств и приборов медицинского назначения в несколько раз, при этом удалось сделать это абсолютно без каких-либо серьёзных изменений в архитектуре проекта и без серьёзных вложений в серверное оборудование.

Список используемых источников:

1. Горизонтальное масштабирование базы данных реального проекта с помощью SQL Azure Federations [Электронный ресурс] – Режим доступа: https://msdn.microsoft.com/ru-ru/dn458578.aspx

2. Принцип работы с большими данными, парадигма MapReduce [Электронный ресурс] – Режим доступа: https://msdn.microsoft.com/en-us/magazine/mt147240.aspx

3. Большие данные – MapReduce без Hadoop. Использование конвейера ASP.NET [Электронный ресурс] – Режим доступа: https://msdn.microsoft.com/en-us/magazine/mt147240.aspx

4. Введение в MapReduce для разработчиков .NET [Электронный ресурс] – Режим доступа: https://developerzen.com/introduction-to-mapreduce-for-net-developers-1030e070698a