Волошин В. О.
Харьковский национальный
университет радиоэлектроники
vladyslav.voloshyn2@nure.ua

ОБЗОР ТЕХНОЛОГИЙ APACHE SPARK И HADOOP MAPREDUCE

Ежедневно количество данных возрастает и возникает необходимость в их быстрой обработке и надежном хранении, но для этих задач не подходят обычные базы данных. Поэтому для обработки больших объемов данных были разработаны особые программные платформы. На данном этапе развития, существует множество Big Data фреймворков, подходов и решений для выполнения операций над большими объемами данных. Но при наличии такого богатого выбора технологий, появляется проблема в выборе подходящего инструмента, который позволит рационально выполнить поставленную задачу.

Поскольку размеры данных превосходят возможности отдельных машин, пользователям необходимы новые системы для масштабирования вычислений на нескольких узлах. В результате произошел взрыв новых моделей кластерного программирования, ориентированных на разнообразные вычислительные нагрузки. Сначала эти модели были относительно специализированными, а затем были разработаны для новых рабочих нагрузок; например, поддержка пакетной обработки MapReduce, но Google также разработал Dremel для интерактивных запросов SQL и Pregel для итеративных графовых алгоритмов [1].

MapReduce обеспечивает надежную, масштабируемую и отказоустойчивую вычислительную среду для хранения и обработки массивных наборов данных. В то же время возникает необходимость в обосновании выбора технологии MapReduce среди других решений для обработки больших данных. Основными критериями выбора являются размерность данных, масштабируемость системы, тип вычислений, возможность выполнения задач автоматически с целью уменьшения профессиональных требований к специалисту.

Данную модель программирования поддерживают такие фреймворки как Hadoop и Spark. Оба инструмента отвечают за обработку данных, но ключевым является то, какой используется подход: Spark выполняет обработку в оперативной памяти и, следовательно, обладает некоторыми преимуществами в производительности, в том время как Hadoop выполняет чтение и запись на диск, что является трудоемким процессом для компьютера. Поэтому, в данном случае, преимуществом обладает Spark.

Hadoop MapReduce позволяет параллельно обрабатывать огромные объемы данных. Он разбивает входной набор данных на более мелкие, которые обрабатываются отдельно на разных узлах кластера и автоматически собирает результаты, для предфинальной обработки и получения одного результата. Если результирующий набор данных больше доступного ОЗУ, Hadoop MapReduce может превосходить Spark. Hadoop MapReduce это экономичное решение, если нет необходимости в получении немедленных результатов. MapReduce является хорошим решением, если скорость обработки не критична и выполнение обработки данных может быть отложено, например, если обработка данных для пользователей сайта может быть выполнена в ночные часы и это никак не повлияет на его производительность.

Первоначальный Apache Hadoop был сфокусирован на запуске массивных заданий MapReduce для обработки текста в Интернете. Широкое внедрение распределенных вычислительных ресурсов и их повсеместное использование расширило первоначально намеченные цели, выявив два ключевых недостатка: тесное взаимодействие конкретной модели программирования с инфраструктурой управления ресурсами, с обязательным использованием модели программирования MapReduce, и централизованная обработка потока управления заданиями, что привело к бесконечным проблемам масштабируемости.

Новая архитектура [2] отделяет модель программирования от инфраструктуры управления ресурсами и делегирует многие функции планирования (например, отказоустойчивость задачи) для компонентов каждого приложения.

Что касается Spark, одним из его преимуществ является быстрая обработка данных, которая выполняется в оперативной памяти, этот подход позволяет получить ускорение по сравнению с Hadoop MapReduce в несколько десятков раз для данных, которые находятся в ОЗУ. Также имеет возможность выполнять итеративную обработку. Если задача состоит в постоянной обработке данных – Spark выигрывает у Hadoop MapReduce. Устойчивые распределенные наборы данных Spark (Resilient Distributed Datasets) позволяют работать с несколькими картами в памяти, в то время как Hadoop MapReduce записывает промежуточные результаты на диск, но если Spark не может удерживать RDD в памяти между шагами, он будет записывать промежуточные результаты на диск, как это делает Hadoop.

Spark использует библиотеку MLlib – это встроенная библиотека для машинного обучения, в то время как Hadoop использует неофициальные разработки. MLlib имеет готовые алгоритмы, которые также выполняются в оперативной памяти, и позволяет выполнять их настройку и модификацию.

Из-за своей скорости Spark может создавать комбинации данных быстрее, хотя Hadoop может получить лучший результат по производительности, если требуется объединение очень больших наборов данных, требующих много перетасовки и сортировок.

Spark менее безопасен по сравнению с MapReduce, потому что он поддерживает аутентификацию с общим секретным паролем. Hadoop MapReduce использует сетевой протокол Kerberos, что позволяет защитить данные.

В работе [3] отмечено, что Spark может быть более предпочтительнее, чем Hadoop, поскольку:

• предлагает распределенную файловую систему с управлением отказами и репликацией данных;

• позволяет добавлять новые узлы во время выполнения;

• предоставляет набор инструментов для анализа и управления данными, который более простой в использовании, развертывании и обслуживании.

Интересным предметом исследований является интеграция между Hadoop и MPI / OpenMP, поскольку она может значительно повысить скорость работы аналитическую обработку данных в облаке

Таким образом, различия между Apache Spark и Hadoop MapReduce показывают, что Apache Spark – это многозадачная технология обработки данных для кластеров, чем MapReduce. Spark может обрабатывать как пакетные типы представления данных, так и потоковые, в то время как MapReduce имеет ограничения на обработку только пакетных данных.

Список используемых источников

1. Zaharia M. et al. Apache spark: a unified engine for big data processing //Communications of the ACM. – 2016. – Т. 59. – №. 11. – С. 56-65.

2. Vavilapalli V. K. et al. Apache hadoop yarn: Yet another resource negotiator //Proceedings of the 4th annual Symposium on Cloud Computing. – ACM, 2013. – С. 5.

3.Reyes-Ortiz J. L., Oneto L., Anguita D. Big data analytics in the cloud: Spark on hadoop vs mpi/openmp on beowulf //Procedia Computer Science. – 2015. – Т. 53. – С. 121-130.

Научный руководитель: Аксак Н. Г., к.т.н., профессор кафедры электронных вычислительных машин