УДК 004.6-044.62

ЛУЧШИЕ ПРАКТИКИ РАЗРАБОТКИ BIG DATA ПРИЛОЖЕНИЙ НА БАЗЕ HADOOP

М.П. Хоронеко
Студент БГУИР

Н.В. Харитонов
Студент БГУИР

М.А. Медунецкий
Студент БГУИР

М.В. Стержанов
Доцент кафедры информатики БГУИР

Белорусский государственный университет информатики и радиоэлектроники, Республика Беларусь
E-mail: maxim.horoneko@gmail.com., mmed.log@gmail.com, nikita.kharitonov99@gmail.com. sterjanov@bsuir.by.

М.П. Хоронеко
В 2016 году закончил светлогорскую гимназию. Принимал активное участие в олимпиадном движении. В школьное время неоднократно становился победителем олимпиад по математике, программированию и астрономии.

Н.В. Харитонов
В июне 2016 года закончил светлогорскую гимназию, где принимал активное участие в олимпиадном движении. В школьное время неоднократно становился победителем областной и призером республиканской олимпиады по физике.

М.А. Медунецкий
Победитель множества олимпиад по программированию, в том числе полуфинала международной студенческой олимпиады (ACM ICPC). В 2016 году поступил на факультет КСиС Белорусского Государственного Университета Информатики и Радиоэлектроники.

М.В. Стержанов
Родился в 1984 году в Минске. В 2001 поступил в БГУИР на специальность Информатика.

Аннотация. Целью данной работы является анализ инфраструктуры Apache Hadoop. Нами представлены наиболее эффективные и незатратные с точки зрения реализации практики программирования приложений на базе Hadoop. В работе также описаны основные методы обнаружения дефектов в приложении, способы оптимизации системы Hadoop и процесса разработки. Указаны инструменты, позволяющие облегчить процессы поиска дефектов и совершенствования эффективности программы.

Ключевые слова: Большие данные, MapReduce, Hadoop, распределенные системы, лучшие практики.

Hadoop – это бесплатная платформа для разработки и выполнения распределенных программ в рамках парадигмы MapReduce, находящаяся под управлением ASF (Apache Software Foundation)[1]. Помимо распределенных вычислений, Hadoop может использоваться как хранилище огромных массивов данных и использует HDFS (Hadoop Distributed File System) в качестве основной файловой системы. Ее особенность заключается в том, что она многократно реплицирует блоки данных, распределяя их по серверам. Поэтому данные не будут утеряны даже при выходе из строя одного или даже нескольких вычислительных

узлов. Hadoop приложения могут работать с файлами в HDFS через программный интерфейс Java.

Hadoop MapReduce

Hadoop MapReduce является свободно доступной реализацией модели программирования MapReduce, разработанной в компании Google, и основана на использовании HDFS[2]. Именно реализация MapReduce в проекте Hadoop принесла этой парадигме такую популярность, благодаря своей открытости и доступности, а широкое использование в коммерческих и исследовательских проектах стимулирует разработчиков к постоянному совершенствованию системы.

Для простоты дальнейшего описания особенностей работы с MapReduce кратко рассмотрим его организацию в терминах Hadoop.

Hadoop MapReduce опирается на использование HDFS. Файлы этой системы имеют блочную структуру и блоки одного файла распределяются по узлам данных (DataNode). Вся файловая система работает под управлением выделенного узла имен (NameNode), в котором хранятся метаданные файлов (в том числе размеры, размещение блоков и их реплик) (рисунок 1).

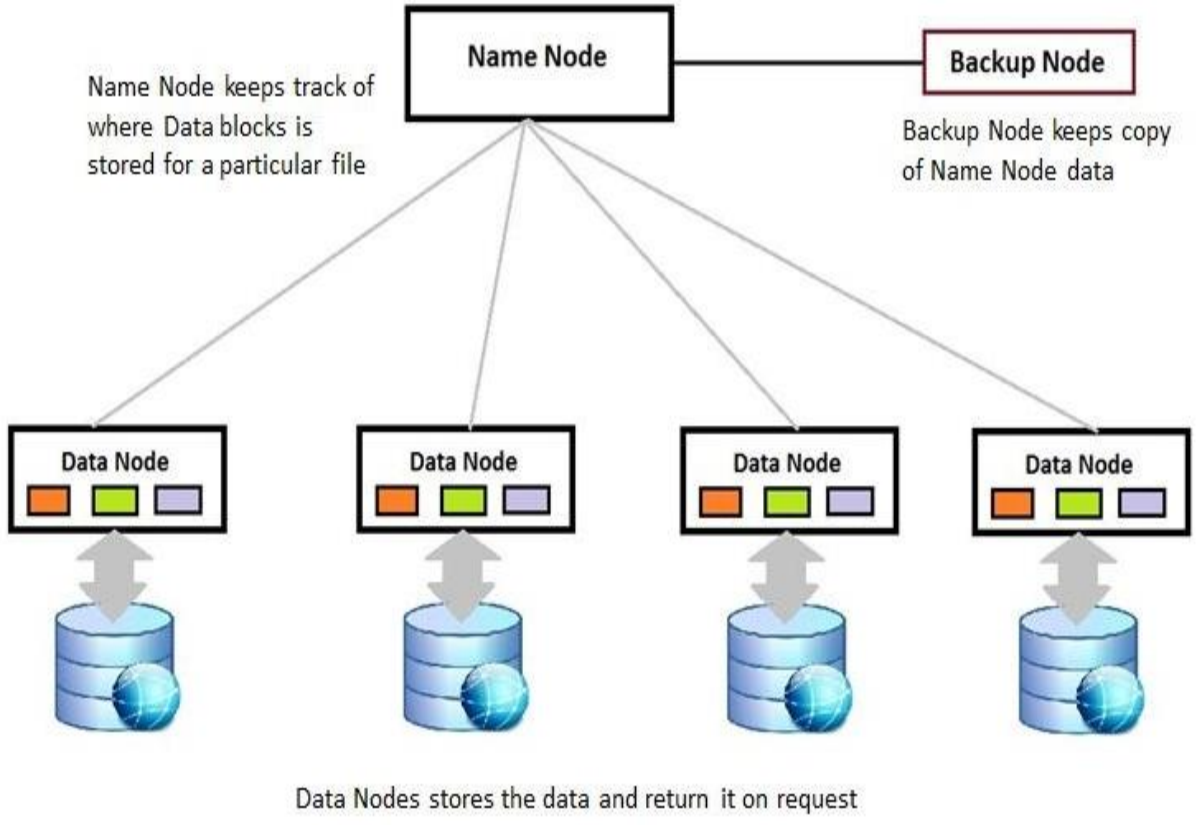

Рисунок 1. Общая структура HDFS

В самом Hadoop MapReduce в соответствии с HDFS поддерживаются множество узлов-исполнителей (TaskTracker) и один узел-поручитель (JobTracker). В узле JobTracker планируются MapReduce-задания, а также отслеживаются работа узлов TaskTracker и свободные ресурсы[3].

Работа Hadoop MapReduce разделена на Map и Reduce задачи. Во время выполнения задачи, Hadoop отправляет Map и Reduce задачи на подходящие серверы в кластере[4].

Опционально между Map и Reduce этапами могут выполняться так называемые Combiners. Их основная задача - сгруппировать вывод Mapper'ов по ключу, причем делают они это локально. По завершении, их вывод, уменьшенный по сравнению с выводом Map задачи отправляется на Reducer, тем самым снижая нагрузку на сеть.

Фреймворк контролирует все детали прохождения данных по кластеру: создание задач, проверка выполнения задач, копирование данных между узлами.

Большинство вычислений проходит на узлах с необходимыми данными на локальных дисках, что позволяет сократить нагрузку на сеть и увеличить скорость вычислений.

После выполнения задачи, кластер собирает данные, формирует подходящий результат и отправляет на сервер Hadoop.

Лучшие практики MapReduce

Существует множество способов оптимизировать процесс разработки, а также саму систему. Среди них:

1. Использование подходящего языка программирования для создания MapReduce задач. Благодаря интерфейсу Hadoop Streaming, который предполагает работу со стандартным вводом-выводом, можно использовать практически любой удобный язык. Поэтому лучше заранее подумать, какой язык лучше всего удовлетворяет заданным требованиям и специфике предметной области. Например, Java больше всего подходит для больших повторящихся задач, Hive для анализа в стиле языка запросов, а Pig для узконаправленных задач.
2. Использование сжатия. Например LZO, bzip2, gzip. Так как Hadoop жестко разделяет файлы на разные блоки, то всегда лучше использовать алгоритмы сжатия, использующие разделение файлов.
3. Правильная настройка числа Mapper и Reducer: количество Mappers/Reducers = Количество узлов * Максимальное количество задач на узел. Максимальное количество задач на узел = количество процессоров на узел - 1.
4. Использование Combiner между Map и Reduce этапами (рисунок 2). Это может значительно увеличить скорость выполнения задач.
5. Отказ от использования Reducer, если существует такая возможность. Фильтрация и уменьшение шума в данных, например, не требует Reduce фазы. Если это так, то лучше убедиться что Reducer'ов нет, так как операции сортировки и перемешивания требуют много ресурсов.
6. Использование достаточно больших размеров блоков данных в HDFS. Hadoop разработан для обработки больших объемов данных. Более крупные блоки позволяют уменьшить время поиска по диску и увеличить скорость вычислений.
7. Разделение задач. Фундаментальной особенностью MapReduce является его модульность. Построение подзадач в цепочку в сложной задаче позволяет рассчитывать на то, что при ошибке в середине ее выполнения можно начать с последней успешно выполненной подзадачи.
8. Написание юнит-тестов и запуск их на небольших объемах данных. Hadoop имеет отличную поддержку юнит-тестов Mapper'ов и Reducer'ов.

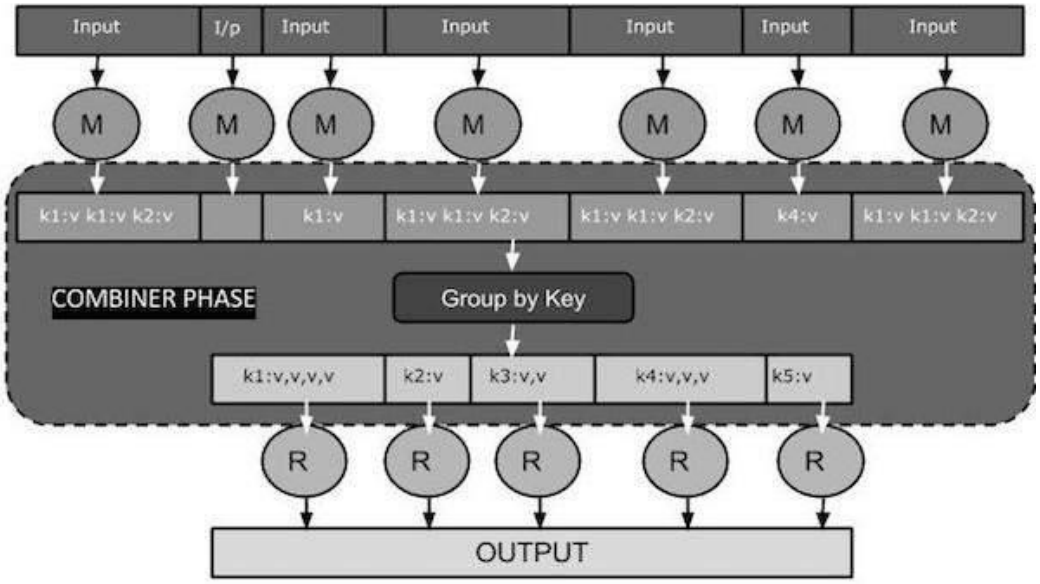

Рисунок 2. Фаза Combiner

Также приведем несколько готовых инструментов, использующихся для оптимизации высоконагруженных систем:

1. Manimal - автоматически анализирует и применяет подходящие оптимизации с учетом данных в программах MapReduce, не требуя дополнительной работы со стороны разработчика
2. SkewTune – применяется для оптимизации определенных пользователем MapReduce задач. Его преимущество заключается в том, что он расширяет Hadoop и сокращает время выполнения задач. Многие приложения становятся намного более эффективными при использовании SkewTune.
3. Starfish - при помощи нее разработчик может видеть поведение MapReduce задачи во время ее выполнения, а также то, как изменяется поведение программы при изменении параметров. Она также эффективно оптимизирует программу.

Обнаружение проблем

Важной частью жизненного цикла Hadoop приложения, как и любого другого, является обнаружение дефектов и проблем как во время разработки так и после введения продукта в эксплуатацию.

Системы MapReduce сталкиваются с огромными трудностями, связанными с возрастающими величиной, разнообразием и консолидацией данных, а также связанными с этим вычислениями. Поиск дефектов в крупномасштабных кластерах требует реалистичного, нагрузочно-ориентированного тестирования производительности.

Для решения этой задачи Калифорнийским университетом в Беркли в тесном сотрудничестве с Facebook был разработан SWIM (Statistical Workload Injector for MapReduce). Это тест, представляющий нагрузку большим объемом реальных данных интернет-сервисов. SWIM также обладает инструментами для синтеза показательных тестовых нагрузок со сложными шаблонами данных и вычислений. Это большое преимущество по сравнению с другими тестами MapReduce ограниченного охвата и разнообразия. SWIM обеспечивает точное измерение производительности систем MapReduce.

Выводы

Были проанализированы принципы работы Hadoop MapReduce - реализации модели распределенных вычислений, разработанной Apache, позволяющей производить параллельные вычисления над огромными, теоретически любыми, объемами данных на серверных кластерах. Были описаны базовые элементы инфраструктуры Hadoop: Mapper, Reducer, Combiner, файловая система HDFS. Рассмотрены современные инструменты оптимизации вычислений в распределенных системах - Manimal, SkewTime, Starfish. Приведены лучшие практики разработки приложений на Hadoop, способы оптимизации и поиска дефектов.

Литература

[1] Vance, Ashlee. Hadoop, a Free Software Program, Finds Uses Beyond Search, The New York Times, 2009, УДК 004.75

[2] Tom White. Hadoop: The Definitive Guide, 3rd Edition, O`Reilly Media, 2012, с. 9, УДК 004.75

[3] Vohra, Deepak. Practical Hadoop Ecosystem: A Definitive Guide to Hadoop-Related Frameworks and Tools, 2016, с. 165, УДК 004.75

[4] Venner, Jason. Pro Hadoop, Apress, 2009, с. 27, УДК 004.75

BEST PRACTICES OF BIG DATA APPLICATIONS DEVELOPMENT ON THE BASE OF HADOOP

M. HORONEKO
Student of BSUIR

N. HARITONOV
Student of BSUIR

M. MEDUNETSKI
Student of BSUIR

M. STERJANOV
Associate professor of the Informatics Department of the BSUIR

Abstract. The purpose of this work is analysis of Apache Hadoop infrastructure. We present the most effective and inexpensive, in terms of implementation, Hadoop applications development practices. The paper also describes the main methods for detecting defects in application, ways to optimize Hadoop based system and development process. Pointed tools that make it easier to search for defects and improve the efficiency of program.

Keywords: Big Data, MapReduce, Hadoop, distributed systems, best practices.