Вычислительные технологии, 2021, том 26, № 2, с. 98–108. © ФИЦ ИВТ, 2021
Computational Technologies, 2021, vol. 26, no. 2, pp. 98–108. © FRC ICT, 2021
ISSN 1560-7534
eISSN 2313-691X

ИНФОРМАЦИОННЫЕ ТЕХНОЛОГИИ

DOI:10.25743/ICT.2021.26.2.008

Применение стека технологий Apache Big Data в задачах климатического мониторинга

С. Ю. Золотов[1,2,*], И. Ю. Турчановский[2]

[1]Институт мониторинга климатических и экологических систем СО РАН, 634055, Томск, Россия

[2]Томский филиал Федерального исследовательского центра информационных и вычислительных технологий, 645055, Томск, Россия

[*]Контактный автор: Золотов Сергей Юрьевич, e-mail: sergey-zo@yandex.ru

Поступила 4 декабря 2020 г., доработана 5 марта 2021 г., принята в печать 12 марта 2021 г.

Описан эксперимент по использованию технологий Apache Big Data в исследованиях климатических систем. В ходе эксперимента реализовано четыре варианта решения тестовой задачи. Ускорение расчетов с помощью технологий Apache Big Data вполне достижимо, и наиболее эффективный способ для этого найден в четвертом варианте решения тестовой задачи. Суть найденного решения сводится к преобразованию исходных наборов данных к формату, подходящему для хранения в распределенной файловой системе и применения технологи Spark SQL из стека Apache Big Data для параллельной обработки данных на вычислительных кластерах.

Ключевые слова: климатический мониторинг, технология Apache Big Data, масштабирование вычислений.

Цитирование: Золотов С.Ю., Турчановский И.Ю. Применение стека технологий Apache Big Data в задачах климатического мониторинга. Вычислительные технологии. 2021; 26(2):98–108. DOI:10.25743/ICT.2021.26.2.008.

Введение

Исследования климатических систем, как правило, сопряжены с обработкой больших массивов данных, например данных реанализов NCEP, JRA, ERA, а также данных, регулярно получаемых сетью метеорологических станций, данных космического мониторинга и т. д. В основные задачи этих исследований входят: мониторинг состояния климатической системы; изучение и анализ явлений и процессов в атмосфере, океане и на суше; мониторинг возможных физических и экологических изменений в окружающей среде в результате климатических изменений [1, 2]. Для решения этих задач ученые часто используют архивы баз данных реанализов — это динамически согласованные поля климатических величин, характеризующих состояние атмосферы, суши и океана по всему земному шару. Основным достоинством глобальных баз данных реанализов является пространственно-временна́я непрерывность выходных данных.

Можно выделить две проблемы, связанные с обработкой больших массивов данных, лежащие в технологической области: большое время считывания многомерных данных и ограничение объема обрабатываемых данных в оперативной памяти ЭВМ.

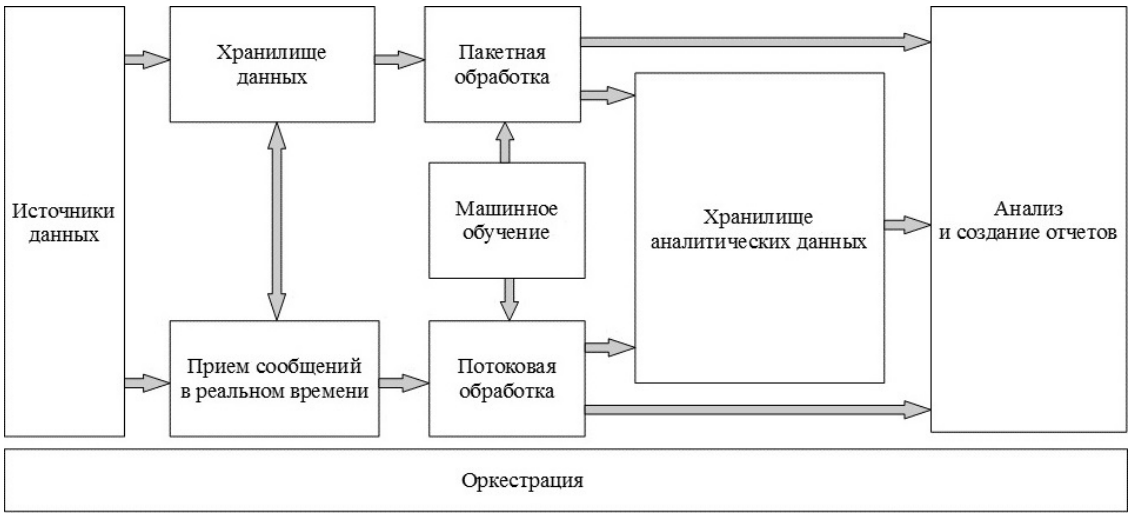
С появлением технологий Big Data, развиваемых под эгидой фонда свободного программного обеспечения Apache, возникла возможность существенно ослабить имеющиеся технологические ограничения. Ядром стека Apache Big Data являются две технологии: Apache Hadoop [3] — для организации распределенных файловых хранилищ неограниченной емкости и Apache Spark [4] — для организации параллельных вычислений на многомашинных кластерах. Эти технологии взаимно дополняют друг друга.

1. Описание используемых технологий распределенных вычислений

При определении архитектуры для систем обработки больших данных на передний план выходят алгоритмы, способные принимать, обрабатывать и анализировать данные, которые являются слишком объемными или слишком сложными для традиционных аналитических систем. Предполагается, что часть данных поступает в непрерывном временно́м ритме, которые постоянно нужно собирать и проводить с ними первичную обработку. Другая часть данных поступает в обычном режиме "по запросу", но в очень больших блоках. В этом случае архитектуру системы для обработки больших данных возможно применять для следующих сценариев [5]: 1) хранение и обработка данных в объемах, слишком больших для традиционных баз данных; 2) преобразование неструктурированных данных для анализа и создания отчетов; 3) запись, обработка и анализ непривязанных потоков данных в режиме реального времени или с низкой задержкой.

На рисунке показаны логические компоненты, которые входят в архитектуру системы для обработки больших данных [5]. Отдельные решения могут не содержать все компоненты в этой схеме. Большинство архитектур систем для обработки больших данных включают следующие некоторые (или все сразу) компоненты.

1. Источники данных. Все решения для обработки больших данных начинаются с одного или нескольких источников данных: хранилища данных приложений (например, реляционные базы данных); статические файлы, которые создаются приложениями; источники данных с передачей в режиме реального времени.



Компоненты архитектуры системы для обработки больших данных
Components of system architecture for big data processing

2. Хранилище данных. Данные для пакетной обработки обычно хранятся в распределенном хранилище файлов, где могут содержаться значительные объемы больших файлов в различных форматах.

3. Пакетная обработка. Так как наборы данных очень велики, для них часто необходима обработка пакетными заданиями, включающая в себя фильтрацию, статистическую обработку и другие процессы подготовки данных к анализу. Обычно в эти задания входит чтение исходных файлов, их обработка и запись выходных данных в новые файлы.

4. Прием сообщений в реальном времени. Если решение содержит источники в режиме реального времени, то в системе должен быть предусмотрен способ сбора и сохранения сообщений в режиме реального времени для потоковой обработки.

5. Потоковая обработка. Сохранив сообщения, поступающие в режиме реального времени, система выполняет для них фильтрацию, статистическую обработку и другие процессы подготовки данных к анализу.

6. Хранилище аналитических данных. Во многих системах данные необходимо подготовить для дальнейшего анализа. Затем обработанные данные структурируются в соответствии с форматом запросов для средств аналитики.

7. Анализ и создание отчетов. Большинство решений по обработке больших данных предназначены для анализа и составления отчетов, что позволяет получить важную информацию.

8. Оркестрация. Большинство решений для обработки больших данных состоят из повторяющихся рабочих процессов: преобразование исходных данных, их перемещение между несколькими источниками и приемниками, а также загрузка результатов обработки данных в хранилища аналитических данных, формирование отчетов или вывод этих результатов на панель мониторинга системы. Эти рабочие процессы выполняются во всевозможных приложениях, объединенных в контейнеры, осуществляющие разработку, развертывание и управление приложениями в рамках изолированной среды. Оркестрация позволяет управлять всеми контейнерами в инфраструктуре разнообразных сред их исполнения.

Связка Apache Spark и Apache Hadoop полностью подходит для создания систем обработки больших данных с использованием вышеописанной архитектуры. Основная идея, реализуемая Spark, — разделение данных на отдельные части (партиции) и обработка этих частей в памяти множества ЭВМ, объединенных сетью. Пересылка данных выполняется только при необходимости, и Spark автоматически определяет, когда будет произведен обмен. Автоматическое распределение данных существенно упрощает модель программирования для исследователя по сравнению с такой традиционной технологией, как MPI. При использовании Spark в сопряжении с Hadoop вычисления производятся с учетом локальности данных. Распределенная файловая система Hadoop (HDFS) хранит файлы в виде блоков, распределенных между дисковыми подсистемами ЭВМ, объединенных в кластер.

Принцип вычислений с учетом локальности предполагает, что программа (или фрагмент параллельной программы) передается и вычисляется на ЭВМ, на которой находятся необходимые блоки данных. Этот принцип позволяет минимизировать обмен данными в процессе счета. Результат расчета также может записываться в файлы HDFS. Можно представить процесс обработки данных в виде конвейера, когда данные поэтапно трансформируются и выходные данные одного этапа являются входными для другого. Сохраняя промежуточные результаты, исследователь имеет возможность вернуться

к ним без повторения всей предшествующей цепочки трансформаций и продолжить расчет иным методом или с другими параметрами.

При работе с большими данными вычисления на кластерах ЭВМ выполняются в режиме совместного доступа множества пользователей. Результаты работы необходимо представлять исследователям своей группы, а также осуществлять обмен с другими группами в рамках научного сотрудничества. Наиболее удобным способом такого представления являются веб-технологии, поэтому еще одним компонентом систем обработки больших данных являются приложения типа "научный журнал". Одним из известных "журналов" в этом классе является Zeppelin [6], который также разрабатывается и развивается как часть сообщества Apache.

2. Постановка задачи

В качестве тестовой выбрана задача подсчета среднемесячных, среднегодовых и сезонных трендов температуры атмосферы нашей планеты за период с 1960 по 2010 г. по данным реанализов NCEP/NCAR [7] и JRA-55 [8]. Основной вопрос эксперимента: возникнет ли ускорение вычислений за счет распараллеливания на многопроцессорной ЭВМ или многомашинном кластере средствами стека Apache Big Data? Основным критерием являлось общее время расчета, в том числе считывание данных и вычисление трендов. При возможности отдельно оценивался вклад операций ввода-вывода и непосредственно вычислений.

Выбор тестовой задачи и стека технологий обусловлен требованиями, выдвинутыми в ФИЦ ИВТ [9, 10], где в качестве центрального требования выступает возможность обработки сверхбольших массивов данных (Big Data), а также требованиями, сформулированными на основе предыдущего опыта разработки информационных систем научных исследований атмосферы и океана Земли [11–14], в которых задача считывания и агрегация моделей реанализа являются базовыми составляющими элементами.

В данной задаче взяты пять уровней по высоте (1000, 850, 500, 200 и 10 гПа) и пять географических точек (60°с. ш., 0°в. д.; 30°с. ш., 0°в. д.; 0°с. ш., 0°в. д.; 30°ю. ш., 0°в. д.; 60°ю. ш., 0°в. д.). Таким образом, результат представляет собой таблицу из 25 строк (5 уровней $\times$ 5 точек) и 17 столбцов (для всех месяцев, четырех сезонов и итоговый за год). В каждую ячейку таблицы заносится значение линейного тренда, рассчитанное за 51 год с 1960 по 2010 г. Итоговое время выполнения задачи является общим временем полного заполнения всех ячеек таблицы.

Данные реанализа NCEP/NCAR представляют собой совокупность файлов по каждой метеорологической величине отдельно. Один файл содержит значения величины за один календарный год с временны́м шагом 6 ч (1460 или 1464 отсчета) по всем уровням давления (17 уровней от 1000 до 10 гПа) на сетке 2.5×2.5° (73 значений широты и 144 значений долготы). Объем одного файла для температуры воздуха равен порядка 310 Мбайт. Данные в файлах реанализа NCEP/NCAR находятся в формате HDF5 [15]. Одной отличительной особенностью этого формата является индексация данных в виде Б-дерева, что позволяет получать к ним параллельный доступ.

В архиве JRA-55 в одном файле хранятся значения одной метеорологической величины за один временной срок. В файле имеются значения величины по всем уровням давления (37 уровней от 1000 до 1 гПа) на сетке 1.25×1.25° (145 значений широты и 288 значений долготы). Объем одного файла для температуры воздуха равен порядка 2.2 Мбайт, что составляет 3.2 Гбайт за один календарный год (1460–1464 файлов).

В архиве JRA-55 используется формат данных GRIB-1 [16], который не предназначен для организации параллельного доступа к данным. Дополнительно в файле находятся массивы сжатых значений величины вместо их оригинальных значений.

Для проведения испытания использовалась вычислительная система на базе четырехъядерного Intel Xeon с частотой 2.4 ГГц, ОЗУ — 64 Гбайт. Система оснащена RAID-массивом и работает под управлением CentOS Linux 7.

3. Программная реализация и результаты решения задачи

В ходе эксперимента реализовано четыре варианта решения тестовой задачи (см. таблицу). Варианты отражают эволюцию программы расчета от полностью последовательной реализации с размещением оригинальных файлов моделей на локальной файловой системе (ФС) Linux ext3 до полностью параллельной, с переходом на специальный формат хранения и размещением на распределенной файловой системе.

Извлечение и трансформация данных реализованы на языке Scala, а непосредственно алгоритм расчета тренда реализован на языке Java. В стеке Apache Big Data языки Java и Scala обеспечивают наибольшую производительность, а реализация расчета тренда на другом языке позволяет оценить временны́е задержки, появляющиеся при вызовах внешних подключаемых библиотек.

Варианты решения задачи и результаты вычислений
Options for solving the problem and calculation results

Вариант	Описание варианта	Параллелизм	Вид файловой системы; формат хранения	Применен кэш	Время расчета для моделей, с	
					NCEP	JRA-55
1	Последовательное считывание файлов модели в обычной ФС Linux ext3 в оригинальном формате и последовательный расчет	Полностью последовательно 1 узел, 1 ядро	Локальная ext3; файлы HDF5, GRIB	Да	883	2003
				Нет	2764	22096
2	Применен API Spark RDD, файлы модели хранятся в обычной ФС Linux ext3 в оригинальном формате	Параллельно 1 узел, 4 ядра	Локальная ext3; файлы HDF5, GRIB	Да	224	981
				Нет	1604	9117
3	Применен API Spark RDD, файлы модели загружены в HDFS в оригинальном формате	Параллельно 2 узла, 8 ядер	Распределенная HDFS; файлы HDF5, GRIB	Да	256	1121
				Нет	867	4927
4	Применен API Spark SQL, файлы модели преобразованы в формат Parquet, загружены в HDFS	Параллельно 2 узла, 8 ядер	Распределенная HDFS; файлы Parquet	Да	214	294
				Нет	492	641

Опишем далее результаты, полученные в ходе эксперимента для каждого варианта реализации.

Первый вариант представляет наиболее простую реализацию без организации параллелизма. В ходе повторных запусков программы ярко проявился эффект кэширования файлов в ОС Linux. Так как вычисления часто проводятся повторно на тех же входных данных при изменении каких-либо настроек программы, было полезно оценить, какое ускорение можно получить за счет кэширования исходных данных.

Модель NCEP предоставляется в виде одного файла формата HDF5 ежегодно, в то время как JRA-55 — в виде 1460 файлов в формате GRIB за каждый год. Таким образом, время считывания данных модели JRA-55 существенно выше, а ускорение за счет кэширования проявляется сильнее (примерно в десять раз для JRA-55 и в три раза для NCEP). Для получения времени расчета без предварительного кэширования проводился сброс всех файловых кэшей узла с помощью специальных команд ОС Linux.

Второй вариант реализации предполагает параллельное считывание данных, агрегирование и последующий расчет трендов. Данные считываются из локальной файловой системы так же, как и в первом варианте. Параллельные варианты реализации для NCEP и JRA-55 дают ускорение приблизительно в 2–4 раза, что соответствует линейному ускорению пропорционально числу процессоров в системе. При этом время, затрачиваемое непосредственно на вычисления, остается стабильным и пренебрежимо малым по сравнению со временем считывания на 3–4 порядка и составляет примерно 0.05 с.

Таким образом, мы видим задачу, в которой время ввода-вывода преобладает над временем расчета. В этом варианте массивы обрабатываемых данных организованы в ОЗУ в виде Resilient Distributed Datasets (RDD) — это один из первых способов представления наборов данных для параллельной обработки, появившийся в составе Apache Spark. Так как входные файлы хранились на локальной файловой системе, число ядер ограничивалось максимальным числом ядер одного узла.

Дальнейшее ускорение расчета возможно за счет масштабирования, т. е. увеличения числа подзадач, параллельно считывающих и вычисляющих составные части результата, подвергаемые затем агрегации. Если увеличение числа ядер будет осуществляться за счет объединения множества ЭВМ в виде кластера, то и хранение данных тоже должно быть организовано в распределенном виде. Поэтому третьим вариантом стал расчет тестовой задачи на двухузловом кластере с общим числом ядер 8.

Самым простым решением было разместить файлы архивов NCEP и JRA-55 в исходном формате в хранилище Hadoop, объединяющем дисковые подсистемы двух вычислительных машин. Однако на этапе адаптации программного кода выяснилось, что прикладная библиотека рассчитана на работу с файлами в режиме прямого доступа к файлам оригинального формата, тогда как Hadoop обеспечивает только последовательный доступ. Обходным решением стала загрузка всех файлов реанализов полностью в оперативную память рабочего процесса, тогда режим прямого доступа обеспечивался без обращения к файловой системе.

Отрицательной стороной этого решения стало увеличение времени считывания всего файла, а также увеличение требований к оперативной памяти рабочего процесса, тогда как для алгоритма расчета требуется лишь небольшой сегмент файла. Тем не менее применение распределенного хранилища и восьми ядер обеспечило ускорение расчета приблизительно в два раза по сравнению с четырехъядерным вариантом и хранением в локальной файловой системе ext3 без предварительной буферизации. Повторный расчет (когда данные кэшированы в ОЗУ) показывает, что в распределенной файло-

вой системе есть накладные расходы на ввод-вывод, а так как время вычислений на два порядка меньше времени ввода-вывода, увеличение числа ядер до восьми не приводит к выигрышу в общем времени расчета. В задачах с более вычислительно емкой составляющей увеличение числа задействованных ядер должно приводить к ускорению расчета.

Результаты третьего варианта поставили вопрос о том, как избежать полной загрузки входного файла в ОЗУ, так как для более сложных расчетов может потребоваться считывание дополнительных файлов модели, например с характеристиками ветра, и, соответственно, ограничение на объем ОЗУ станет критическим.

Решение, предложенное в четвертом варианте, заключается в том, чтобы предварительно преобразовать исходный формат файлов HDF5 или GRIB к виду, когда считывание из HDFS происходит выборочно, на основе заданных параметров. В стеке Apache Big Data разработан мощный инструментарий, подходящий для решения такого сорта задач. Во-первых, это формат файлов Parquet, ориентированный на табличное представление данных. Во-вторых, это представление наборов данных Spark DataSet, являющееся развитием Spark RDD и ориентированное на реляционное представление данных, и соответствующий прикладной интерфейс, напоминающий работу с традиционными СУБД: Spark SQL.

В четвертом варианте сначала была выполнена программа, трансформирующая исходный формат модели в табличное представление формата Parquet с записью в HDFS, а затем программа, осуществляющая выборку данных и последующий расчет. Алгоритм расчета был кардинально переработан в части агрегации данных, а именно в вычислении средних значений для месяца, сезона и года, так как Spark SQL предоставляет встроенные агрегатные функции, а также функции сортировки и группировки данных. Программа четвертого варианта стала существенно короче, так как большая часть операций была совмещена с операциями выборки данных.

Общее время расчета несколько сократилось по сравнению с третьим вариантом, как с кэшированием, так и без него. При этом ускорение за счет кэширования равнялось 200 % от общего времени расчета, в то время как во всех трех вариантах оно составляло от 300 до 1000 %. Это говорит о том, что механизм выборки Spark SQL не требует считывания большого объема данных в ОЗУ и кэширование обеспечивает меньшее ускорение. Также заметно, что время расчета для NCEP и JRA-55 различается незначительно, так как формат и способ выборки для этих моделей идентичны, а различие во времени обусловлено тем, что JRA-55 предоставляет более подробную пространственную сетку.

Время работы программы преобразования данных заняло порядка 2 ч для одного значения долготы в модели NCEP за период 1960–2010 гг. Соответственно, для больших периодов и всех значений долготы это может занять около 40 ч (в нашем случае это кластер с двумя узлами и восемью ядрами). С практической точки зрения эта операция должна быть выполнена единожды, а все последующие расчеты могут использовать модели в преобразованном формате. Может показаться, что представление многомерных данных в табличном виде приведет к избыточности из-за дублирования сочетаний “уровень, широта, долгота, время” для каждого значения функции, однако формат Parquet оснащен встроенными средствами сжатия и разбиения данных, и, как следствие, общий объем моделей в преобразованном виде незначительно превысил объем оригинальных файлов (примерно на 20–30 %), что вполне приемлемо, если учитывать существенный выигрыш в простоте реализации выборки и масштабировании вычислений.

Заключение

Результаты проведенного эксперимента позволяют ответить утвердительно на вопрос, поставленный в начале статьи. Ускорение с помощью стека Apache Big Data вполне достижимо, и наиболее эффективный способ для этого найден в четвертом варианте решения тестовой задачи. Суть найденного решения сводится к преобразованию исходных массивов данных к формату, подходящему для хранения в распределенной файловой системе HDFS и применения технологии Spark SQL из стека Apache Big Data для параллельной обработки данных на вычислительных кластерах.

Следует отметить, что мы ожидали более трудоемкого решения с точки зрения разработки исходного кода. Наши ожидания были сформированы результатами, полученными в НАСА при решении подобной задачи [17, 18]. Так, например, предполагалось, что потребуется использование специально модифицированного вида RDD для работы с многомерными данными. Но обнаружилось, что стек Apache Big Data в настоящее время предоставляет мощные инструменты, открывающие хорошие перспективы по эффективному анализу больших массивов данных в исследованиях атмосферы и океана нашей планеты без излишнего усложнения программного кода.

Список литературы

[1] Оценочный доклад об изменениях климата и их последствиях на территории Российской Федерации. Том I. Изменения климата. М.: Росгидромет; 2008: 228.

[2] **Stocker T.F., Qin D., Plattner G.-K., Tignor M., Allen S.K., Boschung J., Nauels A., Xia Y., Bex V., Midgley P.M.** Climate change 2013: The physical science basis. Contribution of Working Group I to the Fifth Assessment Report of the Intergovernmental Panel on Climate Change. Cambridge, United Kingdom: Cambridge University Press; 2013: 1535. DOI:10.1017/CBO9781107415324.

[3] Apache Hadoop. Available at: `https://hadoop.apache.org` (accessed 2.12.2020).

[4] Apache Spark. Available at: `https://spark.apache.org` (accessed 2.12.2020).

[5] Big data architectures. Available at: `https://docs.microsoft.com/en-us/azure/architecture/data-guide/big-data` (accessed 2.12.2020).

[6] Apache Zeppelin. Available at: `https://zeppelin.apache.org` (accessed 2.12.2020).

[7] **Kalnay E., Kanamitsu M., Kistler R., Collins W., Deaven D., Gandin L., Iredell M., Saha S., White G., Woollen J., Zhu Y., Chelliah M., Ebisuzaki W., Higgins W., Janowiak J., Mo K.C., Ropelewski C., Wang J., Leetmaa A., Reynolds R., Roy J., Dennis J.** The NCEP/NCAR 40-year reanalysis project. Bulletin of the American Meteorological Society. 1996; 77(3):437–471.

[8] **Kobayashi S., Ota Y., Harada Y., Ebita A., Moriya M., Onoda H., Onogi K., Kamahori H., Kobayashi C., Endo H., Miyaoka K., Takahashi K.** The JRA-55 reanalysis: General specifications and basic characteristics. Journal of the Meteorological Society of Japan. 2015; 93(1):5–48.

[9] **Юрченко А.В.** К концепции информационно-аналитической системы поддержки научных исследований, основанных на интенсивном использовании цифровых данных. Вычислительные технологии. 2017; 22(4):105–120.

[10] **Бойченко И.В., Турчановский И.Ю.** Построение сервиса данных в информационных системах научных исследований на основе парадигмы Big Data. Вестник Новосибирского государственного университета. Серия: Информационные технологии. 2015; 13(2):22–27.

[11] **Бойченко И.В., Катаев М.Ю., Маричев В.Н.** Информационная система для анализа данных лидарного зондирования озона. Метеорология и гидрология. 2001; (12):96–105.

[12] **Kataev M.Yu., Boichenko I.V., Maksyutov S.** Software 7S for simulating of the solar reflected radiance transfer in atmosphere. Proceedings Twelfth Joint International Symposium on Atmospheric and Ocean Optics/Atmospheric Physics. 2006; (6160):616007. DOI:10.1117/12.675203. (In Russ.)

[13] **Бойченко И.В., Катаев М.Ю.** Программная система моделирования отраженного от поверхности Земли солнечного излучения. Доклады Томского государственного университета систем управления и радиоэлектроники. Часть 1. 2009. 1(19):88–95.

[14] **Бойченко И.В., Маричев В.Н., Турчановский И.Ю.** Сервисно-ориентированный подход к построению систем обработки и анализа данных лидарного зондирования атмосферы. Вестник Новосибирского государственного университета. Серия: Информационные технологии. 2014; 12(4):5–12.

[15] The HDF5 library & file format. Available at: `https://www.hdfgroup.org/solutions/hdf5` (accessed 2.12.2020).

[16] A guide to the code form FM 92-IX Ext. GRIB. Available at: `https://www.wmo.int/pages/prog/www/WDM/Guides/Guide-binary-2.html` (accessed 2.12.2020).

[17] **Palamuttam R., Mogrovejo R.M., Mattmann C., Wilson B., Whitehall K., Verma R., McGibbney L., Ramirez P.** SciSpark: Applying in-memory distributed computing to weather event detection and tracking. IEEE International Conference on Big Data. Santa Clara, CA, USA: IEEE; 2015: 2020–2026.

[18] **Wilson B., Palamuttam R., Whitehall K., Mattmann C., Goodman A., Boustani M., Shah S., Zimdars P., Ramirez P.** SciSpark: Highly interactive in-memory science data analytics. IEEE International Conference on Big Data. Washington DC, USA: IEEE; 2016: 2964–2973.

Вычислительные технологии, 2021, том 26, № 2, с. 98–108. © ФИЦ ИВТ, 2021 ISSN 1560-7534
Computational Technologies, 2021, vol. 26, no. 2, pp. 98–108. © FRC ICT, 2021 eISSN 2313-691X

INFORMATION TECHNOLOGIES

DOI:10.25743/ICT.2021.26.2.008

Application of Apache Big Data technologies for the problems of climate monitoring

Zolotov Sergey Yu.[1,2,*], Turchanovskii Igor Yu.[2]

[1]Institute of Monitoring of Climatic and Ecological Systems SB RAS, 634055, Tomsk, Russia
[2]Tomsk Branch of the Federal Research Center for Information and Computational Technologies, 645055, Tomsk, Russia
[*]Corresponding author: Zolotov Sergey Yu., e-mail: `sergey-zo@yandex.ru`
Received December 4, 2020, revised March 5, 2021, accepted March 12, 2021

Abstract

The core of the Apache Big Data stack consists of two technologies: Apache Hadoop for organizing distributed file storages of unlimited capacity and Apache Spark for organizing parallel computing on computing clusters. The combination of Apache Spark and Apache Hadoop is fully applicable for creating big data processing systems.

The main idea implemented by Spark is dividing data into separate parts (partitions) and processing these parts in memory of many computers connected within a network. Data is sent only when needed, and Spark automatically detects when the exchange will take place.

For testing, we chose the problem of calculating the monthly, annual, and seasonal trends in the temperature of the atmosphere of our planet for the period from 1960 to 2010 according to the NCEP/NCAR and JRA-55 reanalysis data. During the experiment, four variants of solving the test problem were implemented.

The first variant represents the simplest implementation without parallelism. The second implementation variant assumes parallel reading of data from the local file system, aggregation, and calculation of trends.

The third variant was the calculation of a test problem on a two-node cluster. NCEP and JRA-55 reanalysis files were placed in their original format in the Hadoop storage (HDFS), which combines the disk subsystems of two computers. The disadvantage of this variant is loading all reanalysis files completely into the random access memory of the workflow.

The solution proposed in the fourth variant is to pre-convert the original file format to a form when reading from HDFS is selective, based on the specified parameters.

Keywords: climate monitoring, Apache Big Data technology, scaling computation.

Citation: Zolotov S.Yu., Turchanovskii I.Yu. Application of Apache Big Data technologies for the problems of climate monitoring. Computational Technologies. 2021; 26(2):98–108. DOI:10.25743/ICT.2021.26.2.008. (In Russ.)

References

1. Otsenochnyy doklad ob izmeneniyakh klimata i ikh posledstviyakh na territorii Rossiyskoy Federatsii. Tom I. Izmeneniya klimata [Assessment report on climate changes and their consequences on the territory of the Russian Federation. Volume I. Climate changes]. Moscow: Rosgidromet; 2008: 228. (In Russ.)
2. **Stocker T.F., Qin D., Plattner G.-K., Tignor M., Allen S.K., Boschung J., Nauels A., Xia Y., Bex V., Midgley P.M.** Climate change 2013: The physical science basis. Contribution of Working Group I to the Fifth Assessment Report of the Intergovernmental Panel on Climate Change. Cambridge, United Kingdom: Cambridge University Press; 2013: 1535. DOI:10.1017/CBO9781107415324.
3. Apache Hadoop. Available at: `https://hadoop.apache.org` (accessed 2.12.2020).
4. Apache Spark. Available at: `https://spark.apache.org` (accessed 2.12.2020).
5. Big data architectures. Available at: `https://docs.microsoft.com/en-us/azure/architecture/data-guide/big-data` (accessed 2.12.2020).
6. Apache Zeppelin. Available at: `https://zeppelin.apache.org` (accessed 2.12.2020).
7. **Kalnay E., Kanamitsu M., Kistler R., Collins W., Deaven D., Gandin L., Iredell M., Saha S., White G., Woollen J., Zhu Y., Chelliah M., Ebisuzaki W., Higgins W., Janowiak J., Mo K.C., Ropelewski C., Wang J., Leetmaa A., Reynolds R., Roy J., Dennis J.** The NCEP/NCAR 40-year reanalysis project. Bulletin of the American Meteorological Society. 1996; 77(3):437–471.
8. **Kobayashi S., Ota Y., Harada Y., Ebita A., Moriya M., Onoda H., Onogi K., Kamahori H., Kobayashi C., Endo H., Miyaoka K., Takahashi K.** The JRA-55 reanalysis: General specifications and basic characteristics. Journal of the Meteorological Society of Japan. 2015; 93(1):5–48.
9. **Yurchenko A.V.** On the concept of information-analytical system for supporting data intensive science. Computational Technologies. 2017; 22(4):105–120. (In Russ.)
10. **Boychenko I.V., Turchanovskiy I.Yu.** Designing of the data service in information systems for science research based on Big Data paradigm. Vestnik NSU. Series: Information Technologies. 2015; 13(2):22–27. (In Russ.)
11. **Boichenko I.V., Kataev M.Yu., Marichev V.N.** Information system to analyze lidar ozonesonde data. Meteorologiya i Gidrologiya. 2001; (12):96–105. (In Russ.)
12. **Kataev M.Yu., Boichenko I.V., Maksyutov S.** Software 7S for simulating of the solar reflected radiance transfer in atmosphere. Proceedings Twelfth Joint International Symposium on Atmospheric and Ocean Optics/Atmospheric Physics. 2006; (6160):616007. DOI:10.1117/12.675203. (In Russ.)

13. **Boychenko I.V., Kataev M.Yu.** Program system of modeling of the satellite monitoring an atmosphere and earth surface. Proceedings of TUSUR University. Part 1. 2009; 1(19):88–95. (In Russ.)
14. **Boychenko I.V., Marichev V.N., Turchanovskiy I.Yu.** Service-oriented approach for designing the software systems for the atmosphere lidar sounding data processing and analysis. Vestnik NSU. Series: Information Technologies. 2014; 12(4):5–12. (In Russ.)
15. The HDF5 library & file format. Available at: `https://www.hdfgroup.org/solutions/hdf5` (accessed 2.12.2020).
16. A guide to the code form FM 92-IX Ext. GRIB. Available at: `https://www.wmo.int/pages/prog/www/WDM/Guides/Guide-binary-2.html` (accessed 2.12.2020).
17. **Palamuttam R., Mogrovejo R.M., Mattmann C., Wilson B., Whitehall K., Verma R., McGibbney L., Ramirez P.** SciSpark: Applying in-memory distributed computing to weather event detection and tracking. IEEE International Conference on Big Data. Santa Clara, CA, USA: IEEE; 2015: 2020–2026.
18. **Wilson B., Palamuttam R., Whitehall K., Mattmann C., Goodman A., Boustani M., Shah S., Zimdars P., Ramirez P.** SciSpark: Highly interactive in-memory science data analytics. IEEE International Conference on Big Data. Washington DC, USA: IEEE; 2016: 2964–2973.