Е. О. Тютляева

Разработка и реализация распределенного архива изображений дистанционного зондирования Земли

Научный руководитель: к.х.н. А. А. Московский

Аннотация. Статья описывает разработку и реализацию технологии эффективного хранения и обработки данных дистанционного зондирования Земли с использованием кластерных установок. В статье представлены два подхода к параллельной обработке данных с использованием файловой системы Lustre. Кроме того, в статье представлен обзор аналогичных разработок.

1. Сокращения

В тексте будут использоваться следующие сокращения:

- ПО — программное обеспечение;
- ПС — программная система;
- ДЗЗ — дистанционное зондирование Земли;
- ФС — файловая система;
- API — интерфейс программирования приложений.

2. Введение

Технология Активных хранилищ, использованная при реализации системы «Архус», получила в настоящее время распространение для организации эффективного управления большими объемами данных. Эта технология позволяет решить проблему большой стоимости перемещения данных между обрабатывающими узлами и устройствами хранения путем проведения вычислений непосредственно в местах хранения данных. Передача некоторых вычислительных задач на узлы хранения данных, которые должны в этих вычислениях участвовать, существенно снижает объемы передачи данных по сети, и, следовательно, общесетевой трафик. Это также позволяет эффективно использовать вычислительные мощности узлов хранения. Активные хранилища нацелены на приложения с интенсивной стадией ввода/вывода и данными, которые возможно разбить на независимые наборы. Такая технология нуждается в новых программных

системах для управления каждым из устройств хранения данных, для определения месторасположения различных частей данных и для направления вычислений на узел, хранящий данные, которые должны подвергнуться обработке. В ПС «Архус» предлагается один из возможных подходов к организации такой системы. Использование распределенных Активных хранилищ позволяет создавать масштабируемые, высокоскоростные, с высокой пропускной способностью, управляемые распределенные системы.

3. Обзор аналогов

В рамках разработки ПС «Архус» был проведен обзор аналогичных разработок. Рассмотрим наиболее интересующие проекты (концептуально близкие к реализованной программной системе). При проведении обзора наибольший интерес представляли следующие данные:

- на чем базируется данная система,
- дата выхода последней версии,
- краткий обзор концепции системы.

3.1. Active Storage

Active Storage [1] — система, базирующаяся на ФС Lustre.
Дата выхода последней версии: октябрь 2007.
Краткий обзор: На одном из клиентов кластерной файловой системы Lustre (вычислительный узел, узел–хранилище или любой выделенный узел) будет запущена программа на питоне asmaster, которая получает правило, описывающее задание для Активного хранилища в виде xml-файла и выполняет действия, которые описаны в данном правиле. На остальных узлах запущен Active Storage Runtime Framework, которые помогают asmaster'у запускать задание. Правило (задание) в Active Storage Framework определяет программу, которая использует каждый узел, хранящий данные, как вычислительный узел для обработки.

3.2. Cascading

Cascading [2] — система, базирующаяся на Apache Hadoop.
Дата выхода последней версии: версия 0.9.0 была реализована 24 ноября 2008 г.

Краткий обзор: Данные хранятся на Hadoop кластере. Сама программная система Cascading является API для определения, распределения и запуска потоков обработки данных в вычислительной сети или на кластере. API системы позволяет разработчику быстро описывать требуемую распределенную обработку данных при помощи функциональных операций Map–Reduce (поэлементной обработки множества данных и свертки множества данных по определенной операции соответственно). Для эффективности планировщик использует информацию о зависимостях файлов и прочие мета–данные.

3.3. Pig

Pig [3] — система, базирующаяся на Hadoop.
Дата выхода последней версии: 2008 г.
Краткий обзор: Платформа для анализа больших множеств данных, состоит из языка программирования высокого уровня Pig, который синтаксически является расширением языка Java, соединенного с инфраструктурой для оценки программ. Базируется на данных, расположенных в Hadoop. Программа, написанная на языке параллельного программирования Pig, при помощи компилятора разбивается на последовательность операций Map–Reduce, которые проводятся параллельно.

3.4. Hadoop

Hadoop [4] — самостоятельная система.
Дата выхода последней версии: 2008 г.
Краткий обзор: Hadoop представляет собой платформу для организации распределенных вычислений с использованием парадигмы map/reduce, когда задача делится на множество более мелких обособленных фрагментов, каждый из которых может быть запущен на отдельном узле кластера. В состав Hadoop входит также реализация распределенной файловой системы Hadoop Distributed Filesystem (HDFS), автоматически обеспечивающей резервирование данных и оптимизированной для работы Map–Reduce приложений. Система создана как субпроект поискового механизма и апробировалась в кластере с 600 узлами.

По данным проведенного обзора мы можем заметить, что технология активных хранилищ приобрела особенную популярность именно в последние годы, что свидетельствует об актуальности данной тематики. Большинство рассмотренных разработок базируются на

какой-либо специализированной кластерной ФС, удобной для организации активного хранилища. ПС «Архус» не является исключением, данная разработка использует ФС Lustre. Отличительной особенностью нашей разработки является нацеленность на данные ДЗЗ, что позволяет повысить эффективность для решения задач этого типа. В частности, ПС «Архус» предоставляет пользователю два различных подхода к организации высокопроизводительных вычислений по обработке данных.

4. Структура ПС «Архус»

ПО «Архус» состоит из двух компонент, которые отвечают за выполнение основных функций активного хранилища:

(1) Кластерная ФС Lustre, которая отвечает за хранение данных и распределение их по узлам. При обращении к прикладному программному интерфейсу данной ФС можно получить информацию о расположении данных.
(2) Программный комплекс, позволяющий проводить эффективную обработку данных, расположенных в ФС Lustre, путем направления вычислений на узлы, хранящие обрабатываемые данные.

На рис. 1 изображена упрощенная схема взаимодействия основных компонент системы.

5. Шаблоны хранения и обработки данных

Информация, полученная от спутников ДЗЗ, может храниться в файлах различных форматов и размеров, что влияет на выбираемые способы обработки данных, а также разбиения для хранения в файловой системе Lustre. В связи с этим было разработано два шаблона задач, реализующих различные подходы к хранению и обработке изображений в системе «Архус». Также для каждого шаблона был представлен пример соответствующей задачи обработки данных ДЗЗ. Далее в тексте будет приведено описание подходов и условий, при которых уместно использовать тот или иной шаблон.

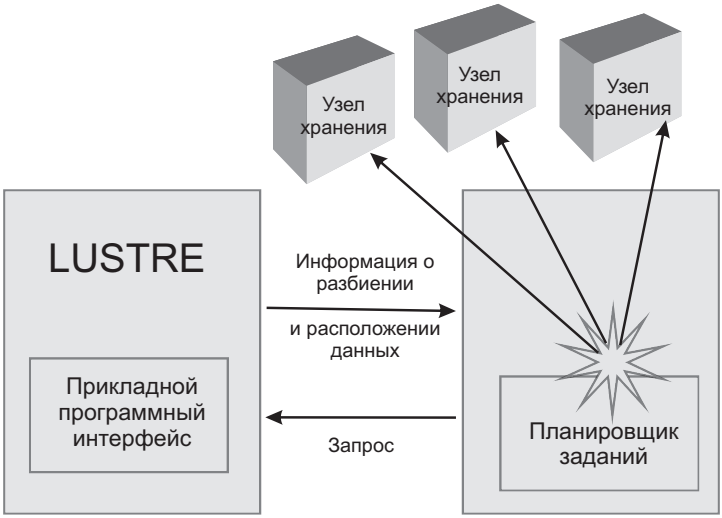

Рис. 1. Структура ПС «Архус»

5.1. Шаблон для обработки «больших» изображений

Данный шаблон предоставляет подход к решению задач по обработке изображений, которые обладают следующими свойствами:

- входные данные содержатся в файлах достаточно большого размера;
- формат описания данных — TIFF;
- вычислительная сложность задачи достаточно высока, общий алгоритм обработки можно представить в виде нескольких независимых потоков, так что затраты на распараллеливание вычислений несущественны по сравнению с затратами на обработку одной порции данных.

При распараллеливании вычислений при помощи этого шаблона используется пополосочная обработка TIFF–файла при помощи шаблона параллельного программирования Map. Более подробное описание этого метода, а также иллюстрирующего его использование примера по контролируемой классификации изображения ДЗЗ при помощи метрики Махаланобиса, содержится в моей курсовой работе за 4-й

курс. В этом году код был лишь слегка оптимизирован и поднят на более высокий уровень абстракции для удобства использования в качестве шаблона. В этом году был реализован принципиально новый шаблон, нацеленный на другой класс задач и более простой в использовании, описание которого дано в следующем полразделе.

5.2. Шаблон для обработки «небольших» изображений

Данный шаблон предоставляет подход к решению задач по обработке изображений, которые обладают следующими свойствами:

- имеется последовательный код обработки изображения, который нужно многократно запускать для различных наборов данных;
- данные хранятся в файлах небольших размеров, так что обработка файла целиком на одном узле более эффективна, чем разбиение его на порции;
- формат данных не имеет значения.

При реализации этого шаблона задачи в ПС «Архус» файл с данными целиком располагается на узле, его обработка происходит там же; последовательный код обработки отдельного изображения не подвергается никакой модификации, реализуется параллелизм по данным. Файлы равномерно распределены по узлам кластера при помощи средств ФС Lustre. При этом планировщик, реализованный с использованием библиотеки T-Sim, используется для направления вычислений на тот узел кластера, на котором расположен обрабатываемый файл. Отличительной особенностью данного шаблона является то, что последовательный код обработки изображения, как уже было указано, не подвергается никакой модификации. Для того, чтобы получить прирост производительности при обработке многократных запросов по работе с различными файлами, программисту достаточно лишь написать функцию для синтаксического разбора командной строки, которая подается на вход планировщику. Командная строка в данном случае — это строка для запуска последовательного кода программы. Основная задача парсера (синтаксического анализатора) — проанализировать командную строку и выделить полный путь к файлу с исходными данными в ФС Lustre. По этим данным планировщик сможет определить, на каком узле расположен файл, и на какой узел, соответственно, следует послать задачу по обработке этого файла. Конкретный вид строки зависит от того, как

реализован в том или ином случае запуск программы для последовательного исполнения. Командная строка может прямо содержать путь к файлу с данными, в этом случае задача парсера проста — необходимо выделить соответствующий аргумент командной строки и вернуть его. В другом случае, командная строка может содержать имя конфигурационного файла, который содержит имена всех файлов с исходными данными, как это требовалось по спецификации задачи перепроецирования данных. Для такого варианта парсер должен проанализировать содержание этого конфигурационного файла и из него извлечь информацию о полных путях к файлам с данными. Возможны и другие варианты. Таким образом, та работа, которую необходимо проделать парсеру, зависит от вида командной строки. Такой подход в шаблоне получается несколько трудоемким, но обладает большой гибкостью — важно то, что последовательный код программы и вид ее запуска не нуждаются ни в каких модификациях. Данный шаблон был реализован на основании задачи по перепроецированию данных ДЗЗ. Институтом Космических Исследований был предоставлен последовательный код, который перепроецировал указанные в конфигурационном файле изображения, а затем производил склейку всех полученных гранул. Предполагалось, что в дальнейшем будет развернут веб–сервис, к которому будут проводиться множественные запросы, то есть код по перепроецированию изображений нужно будет запускать многократно для различных гранул из архива. Таким образом, для повышения производительности данной задачи эффективно применять планировщик.

6. Результаты проверки повышения производительности системы

Для того, чтобы проверить теоретические выкладки по повышению эффективности, был проведен ряд измерений увеличения скорости обработки изображений ДЗЗ при увеличении числа задействованных узлов кластерного ВМВС на примере двух демонстрационных приложений. Тестирование проводилось на двух машинах — в начале, для работ были предоставлены только 2 узла кластера demo.botik.ru,

конфигурационные параметры которого показаны в таблице 1. Теоретически предполагалось, что при увеличении количества узлов будет получено повышение производительности. В феврале представилась возможность проверить данное утверждение в связи с получением доступа на 4 узла кластера blade.botik.ru, обладающего характеристиками, описанными в таблице 2.

Таблица 1. demo.botik.ru

Место расположения	ИПС РАН
Число вычислительных узлов	2
Тип процессора	Intel(R) Xeon(TM) 2.80GHz
Количество ядер	2
Количество процесоров в узле	2
Оперативная память узла	1 GB
Дисковая память установки	250+80 GB
Тип системной сети	Gigabit Ethernet
Конструктив узла (форм-фактор)	2U

Таблица 2. blade.botik.ru

Место расположения	ИПС РАН
Число вычислительных узлов	8
Тип процессора	Intel(R) Xeon(R) 3.00GHz
Количество ядер	4
Количество процесоров в узле	2
Оперативная память узла	16 GB
Дисковая память установки	Рейд-массив на 3 TB
Тип системной сети	Gigabit Ethernet
Конструктив узла (форм-фактор)	5U

6.1. Тестирование классификатора изображений по метрике Махаланобиса

При тестировании классификатора данные располагались в хранилище (в параллельной файловой системе Lustre). При тестовых запусках на одном узле все данные располагались на этом же узле; при тестировании на двух/четырех узлах, данные, соответственно,

располагались на этих двух/четырех узлах. Такая особенность расположения обрабатываемых данных связана с планировщиком алгоритма, который отправляет задачу, обрабатывающую данные на тот узел, на котором они расположены.

Было проведено многократное тестирование системы, усредненные результаты обработки шести снимков 151 Mb на кластере demo показаны в таблице 3 и на кластере blade 4.

Таблица 3. Результаты тестирования на кластере demo.botik.ru

Количество узлов	1 узел	2 узла
Время, сек	1650.234	924.166
Процент	100	56

Таблица 4. Результаты тестирования на кластере blade.botik.ru

Количество узлов	1 узел	2 узла	4 узла
Время, сек	203.93	136.76	84.57
Процент	100	67	41

6.2. Тестирование планировщика перепроецирования и склейки изображений

При тестировании планировщика в наличии были следующие материалы: архив реальных данных ДЗЗ размеров 1,4 Гб, при том, что средний размер одного файла не превышал 400 КБайт; последовательный код, реализующий операции перепроецирования и склейки изображений. Данные ДЗЗ были размещены в распределенной файловой системе Lustre в соответствии с описанной выше схемой — один файл располагался целиком на узле, файлы были равномерно распределены по узлам с использованием алгоритма round–robin.

Аналогично с предыдущим шаблоном, данные располагались на тех же узлах, на которых проводилось тестирование.

Было проведено многократное тестирование системы, усредненные результаты обработки 80 запросов (320 перепроецирований) на кластере demo показаны в таблице 5 и на кластере blade 6.

Таблица 5. Результаты тестирования на кластере demo.botik.ru

Количество узлов	1 узел	2 узла
Время, сек	102.996	35.602
Процент	100%	34%

Таблица 6. Результаты тестирования на кластере blade.botik.ru

Количество узлов	1 узел	2 узла	4 узла
Время, сек	33.68	26.95	18.76
Процент	100%	80.02%	55.7 %

7. Выводы

В ходе работ был разработан механизм распределения заданий на основе информации, получаемой планировщиком от прикладного программного интерфейса Lustre. Данный механизм был реализован в двух различных шаблонах и соответствующих задачах, использующих эти шаблоны, что позволяет получать высокую эффективность при решении задач двух разных типов. Таким образом, была реализована и документирована ПС «Архус», которая является прототипом распределенного архива изображений, полученных от спутников дистанционного зондирования Земли. Данная программная система построена с использованием идеологии активных хранилищ. Эффективность работы системы обеспечивается обработкой данных на тех же узлах распределенного хранилища, на которых они хранятся, что позволяет существенно снизить расходы на передачу больших массивов данных. Балансировка нагрузки на узлах осуществляется планировщиком библиотеки T-Sim на основе информации, предоставляемой интерфейсом кластерной файловой системы Lustre. Для большего удобства использования и гибкости система включает в себя два шаблона, которые реализуют разные подходы к распараллеливанию вычислений. Тестирование, которое проводилось на двух различных машинах, показало повышение производительности. Целесообразность использования системы «Архус» возрастает при работе с большими объемами исходных данных или при большой вычислительной сложности задач обработки изображения.

8. Перспективы

Планируется продолжить работы над программной системой «Архус». В ближайших планах перенос и модифицирование web–сервиса для запуска программы перепроецирования–склейки изображений на кластер blade.botik.ru. Также планируется тестирование разработанных шаблонов в различных условиях — предполагается тестирование производительности с системной сетью Infiniband и на восьми узлах кластера, когда будет предоставлена такая возможность.

Список литературы

[1] Piernas J. (Nieplocha) Active Storage User's Manual: Pacific Northwest National Laboratory, http://hpc.pnl.gov/projects/active-storage/as_users_manual_october_2007.pdf. ↑3.1

[2] Cascading, http://www.cascading.org/. ↑3.2

[3] Welcome to Pig!, http://hadoop.apache.org/pig/. ↑3.3

[4] Welcome to Hadoop!, http://hadoop.apache.org/core/. ↑3.4

E. O. Tyutliaeva. *Development and implementation of distributed remote sensing data storage* // Proceedings of Junior research and development conference of Ailamazyan Pereslavl university. — Pereslavl, 2009. — p. 195–205. (*in Russian*).

Abstract. This paper describes development and implementation of technology for efficient distributed remote sensing data storage and processing using cluster system. Author specify two approaches to data parallel processing using the Lustre file system. Furthermore, the paper includes review of similar developments.