ПРОГРАММИРОВАНИЕ

УДК 004.932

СИСТЕМА ОБРАБОТКИ ИЗОБРАЖЕНИЙ С АВТОМАТИЧЕСКИМ РАСПАРАЛЛЕЛИВАНИЕМ НА ОСНОВЕ MAPREDUCE

А.В. Созыкин, М.Л. Гольдштейн

Целью работы является создание системы обработки изображений в параллельном режиме под управлением Apache Hadoop на основе технологии MapReduce, которая скрывает от прикладного программиста детали внутреннего устройства Hadoop и предоставляет простой программный интерфейс для работы с изображением, уже загруженным в память.
Основными результатами являются архитектура системы обработки изображений с автоматическим распараллеливанием на основе Hadoop и ее практическая реализация в виде первой очереди комплекса программ. Созданный комплекс программ применен для обработки изображений от системы Particle Image Velocimetry (источник данных – проект PIV Challenge). Тестирование комплекса программ на кластере Hadoop из четырех узлов показало почти линейную масштабируемость.
Практическое применение возможно в научной сфере (обработка изображений от физических экспериментальных установок, астрономических наблюдений, спутниковых снимков земной поверхности и т.д.), медицине (обработка изображений, получаемых в результате применения высокотехнологичной медтехники) и коммерческих компаниях (анализ данных с камер видеонаблюдения в системах безопасности, в геоинформационных системах и т.п.).
Предложенный подход позволяет повысить производительность обработки изображений за счет применения параллельных вычислительных систем и повышает эффективность работы прикладных программистов, позволяя им концентрироваться на алгоритмах обработки изображений, а не на деталях параллельной реализации.

Ключевые слова: обработка изображений, MapReduce, Hadoop, распределенная файловая система, автоматизация распараллеливания.

Введение

Задачи обработки изображений часто встречаются как в науке, так и в промышленности. Примерами могут служить анализ данных медицинских исследований, астрономических наблюдений, снимков земной поверхности с космических спутников, видео с камер наблюдения систем безопасности и многое другое. При этом особенностью современных задач является требование обработки больших объемов данных, измеряемых в терабайтах и приближающихся к петабайтам [1]. Такие объемы эффективно могут быть обработаны только на параллельных вычислительных системах.

Обработка изображений на параллельных вычислительных системах сопряжена с рядом проблем. Разработка параллельных программ является сложной задачей, т.к. от программиста требуется вручную разбивать задачу на отдельные части, которые можно выполнить

параллельно, распределять нагрузку между узлами вычислительной системы, обрабатывать отказы оборудования и т.д. В результате повышаются требования к квалификации разработчиков, растет число программных ошибок и увеличивает срок создания системы обработки изображений.

Хранение больших объемов изображений также представляет собой проблему. Системы хранения большого объема отличаются высокой стоимостью и часто не обладают достаточной производительностью передачи данных для работы в составе крупного вычислительного кластера.

Актуальной является задача создания системы обработки изображений с автоматическим распараллеливанием (СОИСАР), предоставляющей возможность разработчику написать последовательную программу, обрабатывающую одно изображение (или группу связанных изображений), и затем автоматически запустить эту программу в параллельном режиме на кластере для обработки большого числа изображений. Также должна быть обеспечена возможность хранения изображений на внутренних дисках серверов кластера, организованных в логически единую распределенную файловую систему.

Наиболее перспективной технологией с автоматическим распараллеливанием сегодня можно считать технологию Apache Hadoop [2] в рамках модели программирования MapReduce [3].

1. MapReduce и Apache Hadoop

MapReduce – это модель параллельного программирования, разработанная компанией Google [3], для решения задач информационного поиска. Основными возможностями данной технологии являются:

- автоматическое распараллеливание задачи на кластере из серверов стандартной архитектуры;
- балансировка нагрузки между узлами кластера;
- защита от сбоев оборудования путем перезапуска задачи на другом узле кластера;
- распределенная файловая система для хранения данных на внутренних дисков серверов кластера [4].

Такие широкие возможности достигаются за счет использования единственного алгоритма – MapReduce, который ориентирован на обработку списков и состоит из двух шагов: Map и Reduce. Map получает на вход список пар <ключ, значение> и выдает преобразованный список пар <ключ, значение>. Reduce получает на вход список пар <ключ, значение> с одинаковым ключом, и выдает только одно значение. При этом обработка элементов списков может выполняться независимо друг от друга, в том числе параллельно. Алгоритм содержит только одну фазу коммуникации. После завершения шага Map значения с одинаковым ключом пересылаются по сети на один узел кластера, где они обрабатываются на шаге Reduce. Пользователь пишет функции Map и Reduce, обрабатывающие одно значение, а параллельное выполнение над большим объемом данных обеспечивает программная система MapReduce.

В MapReduce каждый узел кластера используется одновременно как для хранения данных, так и для вычислений. При этом планировщик MapReduce старается распределять задания на те узлы кластера, где хранятся данные, подлежащие обработке. Такое распределение позволяет существенно повысить производительность. Компания Google реализовала MapReduce на C, но не распространяет свою реализацию.

Apache Hadoop [2] – это наиболее популярная из доступных к использованию технологий, реализующих MapReduce, которая включает распределенную файловую систему HDFS [5] и ряд готовых к использованию приложений, таких как распределенная база данных HBase, хранилище данных Hive, система машинного обучения Mahout.

Hadoop, рассчитанный на параллельное выполнение независимых задач, хорошо подходит для обработки изображений, т.к. изображение или группа изображений могут быть обработаны независимо. В отличие от MPI, Hadoop обеспечивает автоматическое распараллеливание последовательной программы и перезапуск задач в случае отказа оборудования. Распределенная файловая система предоставляет возможность хранить данные больших объемов прямо на внутренних дисках серверов кластера без необходимости приобретения дорогостоящей системы хранения данных.

Все описанные преимущества, а также наличие реализаций с открытыми исходными кодами, в первую очередь Hadoop [2], позволяет использовать MapReduce в качестве основы для СОИСАР.

2. Обзор аналогов систем обработки изображений с автоматическим распараллеливанием

MapReduce и Hadoop широко используются для обработки изображений на параллельных вычислительных системах. В проекте IM2GPS [6] Hadoop используется для определения позиции снимка местности путем сравнения его с шестью миллионами снимков, для которых заданы координаты GPS. В работах [7, 8] описывается применение Hadoop для обработки данных астрономических наблюдений: объединения нескольких снимков в один для повышения качества. Работа [9] посвящена применению Hadoop для выделения контуров в изображениях. В работе [10] рассматриваются вопросы использования Hadoop для анализа изображений дистанционного зондирования, а в работе [11] – для анализа пространственных данных. Применение MapReduce и Hadoop во всех работах позволило повысить производительность обработки изображений.

Анализ работ [6 – 11] позволил выявить ряд особенностей применения Hadoop для обработки изображений. Во-первых, Hadoop рассчитан на обработку текстовых файлов, обработка изображений требует расширения его возможностей. Это приводит к необходимости разработчикам самостоятельно реализовывать функции загрузки изображений в Hadoop, разбиения изображений на части и чтения изображений в разных форматах, что, безусловно, можно трактовать как серьезный недостаток. Эти функции являются общими для разных систем и требуют знания внутреннего устройства Hadoop. Желательно скрыть от прикладного программиста, занимающегося обработкой изображений, технические детали Hadoop по организации параллельной обработки и загрузки изображений из распределенной файловой системы, и предоставить простой программный интерфейс для работы с изображением, уже загруженным в память. Подобный подход используется в системе HIPI [12], автоматизирующей процесс загрузки изображений в Hadoop и предоставляющий прикладному программисту класс FloatImage, содержащий описание пикселей изображения.

Во-вторых, Hadoop реализован на языке Java, который не позволяет эффективно задействовать вычислительную мощность оборудования. Это мы относим к недостатку, который может быть преодолен с помощью технологии Hadoop Streaming, которая позволяет использовать в качестве функций Map и Reduce программы, написанные на любом языке программирования.

В-третьих, имеются различия в использовании функций Map и Reduce. Некоторые системы, в частности IM2GPS [6], вообще не используют Reduce, т.к. все необходимые действия совершаются в Map. Reduce используется в двух случаях:

- Система обрабатывает изображения большого объема, которые могут быть разбиты на части, обрабатываемые независимо. Примером является система выделения контуров изображений [9], в которой несколько процессов Map находят контуры на разных частях изображения, а процесс Reduce объединяет части, подготовленные Map, в единое изображение.

- Система строит одно изображение на основе нескольких. Например, система обработки астрономических данных [7, 8] пытается строить одно высококачественное изображение по нескольким низкокачественным изображениям того же участка неба. В этом случае в процессах Map выполняется обработка исходных изображений и приведение их к единому виду, а Reduce строит результирующее изображение.

3. Архитектура

В данном разделе представлена разработанная архитектура СОИСАР, призванная устранить недостатки обработки изображения в Hadoop, описанные выше. Схема предлагаемой логической архитектуры СОИСАР приведена на рис. 1. В верхней части схемы показаны компоненты Hadoop, используемые в системе, а в нижней части новые компоненты СОИСАР.

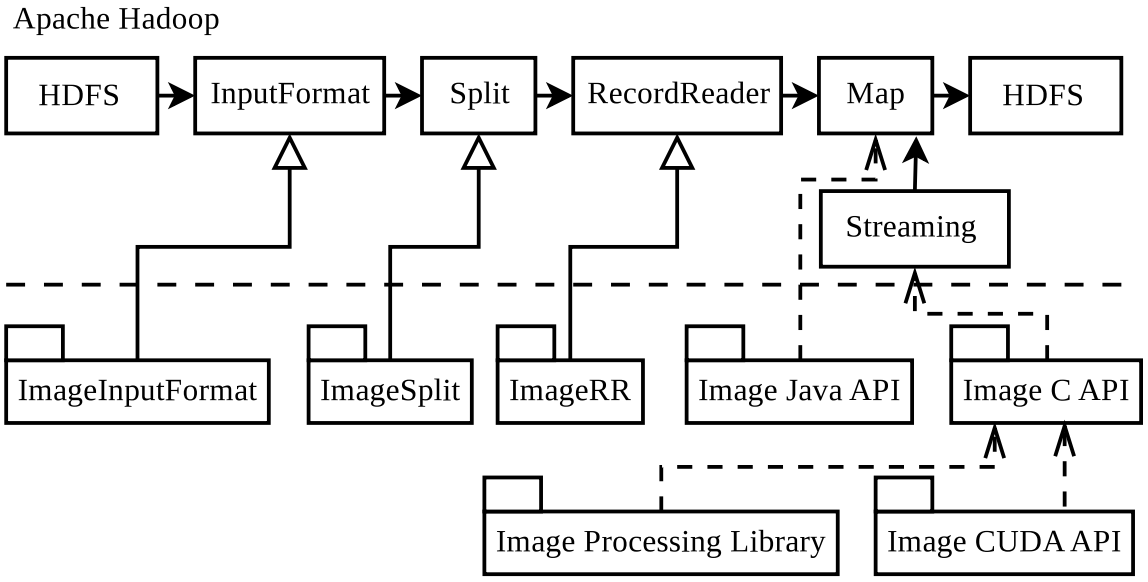

Рис. 1. Логическая архитектура системы обработки изображений с автоматическим распараллеливанием на основе Hadoop

Изображения, которые нужно обработать, записываются в распределенную файловую систему HDFS. СОИСАР содержит пакеты, читающие изображения в различных форматах: ImageInputFormat включает классы, задающие формат чтения изображений, ImageSplit – классы, разбивающее изображение на части, а ImageRR – классы, выполняющее чтение изображений в популярных форматах. Разбивать изображение на части рекомендуется только если его размер превышает размер блока HDFS (64 МБ по-умолчанию), в противном случае применение ImageSplit может привести к снижению производительности.

Из двух распространенных подходов к организации обработки изображений в Hadoop мы выбрали подход без использования Reduce, при котором вся обработка выполняется в функции Map. Это существенным образом упрощает реализацию системы и делает ее более

удобной для пользователя. Недостатком подобного подхода является невозможность разбиения большого изображения на части средствами Hadoop и объединение частей в единое изображение в Reduce, что могло бы повысить производительность обработки изображений. Но такой подход существенно усложнил бы структуру СОИСАР и не позволил бы скрыть от пользователя внутреннее устройство Hadoop.

Предоставляется два интерфейса работы с изображениями: Java API и C API. Java API использует стандартные возможности Hadoop, а C API основан на Hadoop Streaming. Java API предоставляет возможность быстрого создания обработчиков изображений, особенно программистам, знакомым с Hadoop, а C API обеспечивает высокую производительность. Как C API, так и Java API предоставляют возможность разрабатывать последовательную программу, обрабатывающую одно изображение (или группу связанных изображений), а применение этой программы к большому числу изображений в параллельном режиме обеспечивается СОИСАР с помощью средств Hadoop.

На основе C API предлагается создать набор готовых обработчиков изображений, реализующих, например, преобразование Фурье, масочную фильтрацию, изменение размеров и т.п. Готовые обработчики включаются в пакет ImageProcessingLibrary и могут применяться без дополнительного программирования в параллельном режиме. Высокопроизводительная реализация обработчиков возможна с использованием библиотек Intel MKL, Intel Integrated Performance Primitives и AMD Core Math Library.

Пакет ImageCudaAPI позволяет разрабатывать программы для обработки изображений, работающих на графических процессорах производства компании NVIDIA.

Схема архитектуры развертывания СОИСАР представлена на рис. 2.

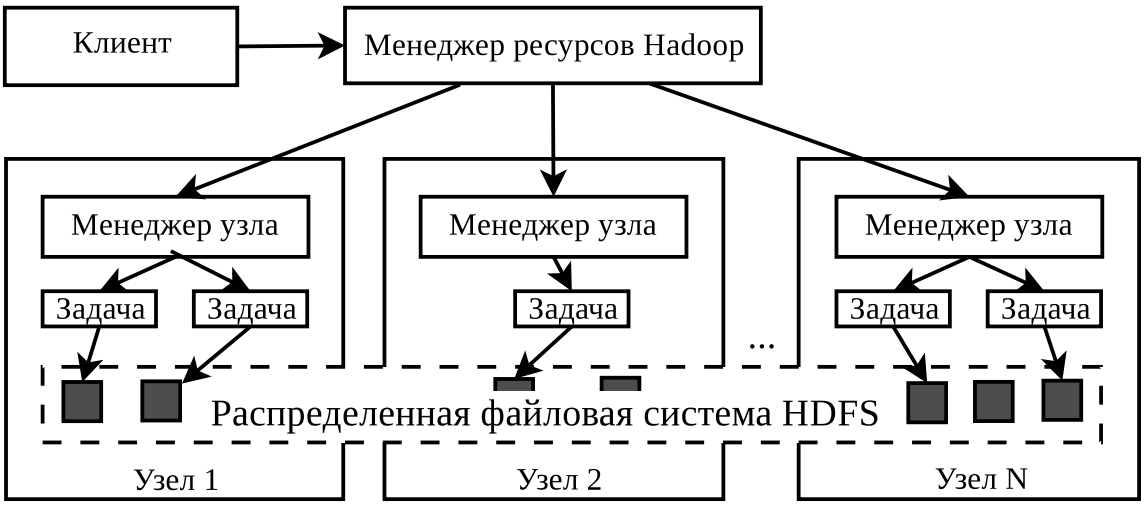

Рис. 2. Архитектура развертывания системы обработки изображений с автоматическим распараллеливанием на основе Hadoop

СОИСАР работает на кластере серверов стандартной архитектуры, управляемом Hadoop. Изображения хранятся на внутренних дисках серверов кластера, логически объединенных в единую распределенную файловую систему HDFS. Запуск задач обработки изображений выполняется с помощью менеджера ресурсов Hadoop, который распределяет задачи по серверам кластера. При этом каждый сервер кластера используется как для хранения изображений, так и для выполнения задач обработки изображений. Менеджер ресурсов Hadoop распределяет задачи таким образом, чтобы они выполнялись на узлах, которые содержат изображения, подлежащие обработке.

Задача представляет собой последовательную программу, обрабатывающую одно изображение (или группу связанных изображений), разработанную с использованием Image C API или Image Java API, или использующую готовый обработчик из пакета Image Processing Library (см. рис. 1). Параллельное применение программы к большому количеству изображений выполняется автоматически средствами Hadoop.

4. Реализация и практическое использование

Предложенная архитектура реализована в виде первой очереди комплекса программ для обработки изображений с автоматическим распараллеливанием, которая включает ImageInputFormat и ImageRR для формата BMP, а также базовый вариант Image Java API. Выполнено развертывание системы на кластере из четырех узлов Fujitsu-Siemens RX 220, в каждом по 2 процессора AMD Opteron 285, 8 ГБ памяти, жесткий диск 250 ГБ SATA, программное обеспечение Scientific Linux 6, Hadoop 1.0.

Комплекс программ практически применен для обработки изображений от системы Particle Image Velocimetry (PIV) [13], предназначенной для измерения скорости потока в жидкости или газе. В системе PIV поток визуализируется путем добавления в жидкость или газ частиц небольшого размера. Затем частицы освещаются пульсирующим лазером и фотографируются кросскорреляционной камерой, которая записывает по два изображения потока через небольшой промежуток времени. Поле скорости потока вычисляется путем анализа пар изображений. В качестве первого шага был реализован стандартный кросскорреляционный алгоритм определения поля скорости потока, и применен для анализа изображений от систем PIV проекта PIV challenge [14]. Пример работы алгоритма приведен на рис. 3. В дальнейшем планируется реализация более сложных алгоритмов на основе вейвлетов [15].

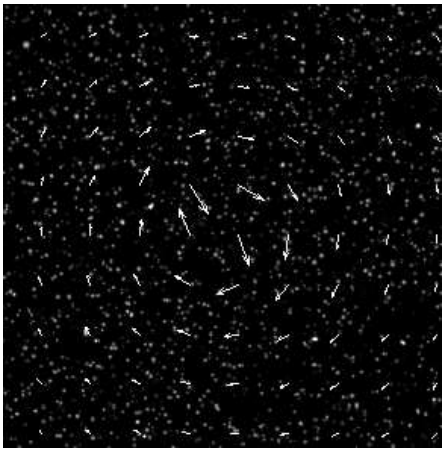

Рис. 3. Поле скорости потока в жидкости, определенное с помощью стандартного кросскорреляционного алгоритма

Для оценки масштабирования предложенного решения были проведены эксперименты по обработке изображений от системы PIV в кластере Hadoop из четырех серверов. Зависимость количества изображений, обрабатываемых за одну минуту, от количества серверов в кластере показана на рис. 4. Для каждого числа серверов эксперимент выполнялся 10 раз, представлены средние значения с доверительными интервалами.

Эксперименты показали почти линейную масштабируемость предлагаемой СОИСАР на задачах обработки изображений от системы PIV. Такой результат является ожидаемым, т.к. обработка пар изображений производится независимо друг от друга.

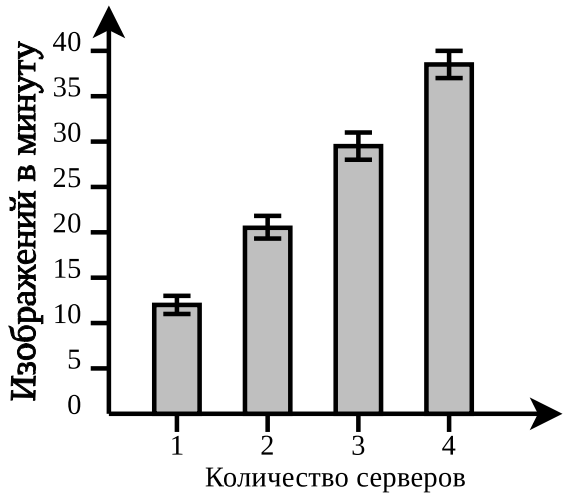

Рис. 4. Зависимость количества обрабатываемых изображений от числа серверов в кластере Hadoop

Следует отметить, что текущая реализация СОИСАР выполнена на Java и не обладает высокой производительностью. В дальнейшем планируется реализовать Image C API с использование Hadoop Streaming, что позволит увеличить производительность по сравнению с результатами, представленными на рис. 4.

Заключение

Использование Hadoop и MapReduce для построения системы обработки изображений позволило обеспечить автоматическое распараллеливание и хранение данных на внутренних дисках узлов кластера. Предложенная архитектура скрывает от прикладного программиста детали внутреннего устройства Hadoop и предоставляет простой программный интерфейс, который позволяет сконцентрироваться на алгоритмах обработки изображений, а не на организации параллельной обработки. Распределенная файловая система позволяет хранить данные большого объема на дисках серверов стандартной архитектуры без необходимости установки дорогостоящей системы хранения. Первая очередь реализации СОИСАР в виде комплекса программ и его практическое применение для обработки изображений от системы PIV позволили оценить масштабируемость предлагаемого решения, которая оказалась почти линейной.

В качестве направлений дальнейших работ можно отметит следующее:

- реализация Image C API для повышения производительности обработки изображений;
- реализация библиотеки готовых обработчиков изображений Image Processing Library;
- исследование возможностей повышения энергоэффективности СОИСАР [16], например, за счет использования аппаратных компонентов с низким энергопотреблением: процессоров архитектуры ARM и графических процессоров.

Работа проводилась при финансовой поддержке УрО РАН, грант 12-П-1-1029.

Литература

1. Horowitz, M. The peta age / M. Horowitz. [Электронный ресурс]. URL: http://www.wired.com/images/press/pdf/petaage.pdf (дата обращения 29.05.2012).
2. Apache Hadoop [Электронный ресурс]. URL: http://hadoop.apache.org/ (дата обращения 29.05.2012).
3. Dean, J. MapReduce: simplified data processing on large clusters / J. Dean, S. Ghemawat // Commun. ACM. – 2008. – №51(1). – P. 107–113.
4. Ghemawat, S. The Google File System / S. Ghemawat, H. Gobioff, S.T. Leung // 19th ACM Symposium on Operating Systems Principles, Lake George, N.Y., 2003. – P. 29–43.
5. Hadoop Distributed File System [Электронный ресурс]. URL: http://hadoop.apache.org/hdfs (дата обращения 29.05.2012).
6. Hays, J. IM2GPS: estimating geographic information from a single image / J. Hays, A.A. Efros // Proceedings of the IEEE Conf. on Computer Vision and Pattern Recognition (CVPR). – 2008. – P. 1–8.
7. Astronomy in the Cloud: Using MapReduce for Image Co-Addition / K. Wiley, A. Connolly, J. Gardner, S. Krughoff, M. Balazinska, B. Howe, Y. Kwon, Y. Bu // Publications of the Astronomical Society of the Pacific. – 2011. – V. 123, №901. – P. 366–380.
8. Wiley, K. Astronomical Image Processing with Hadoop / K. Wiley, A. Connolly, S. Krughoff, J. Gardner, M. Balazinska, B. Howe, Y. Kwon, Y. Bu // Astronomical Data Analysis Software and Systems XX. ASP Conference Proceedings. – 2010. – V. 442. – P. 93–98.
9. Kumar, S. Distributed image processing using hadoop mapreduce framework [Электронный ресурс] / S. Kumar, B.A. Fernandez. URL: http://search.iiit.ac.in/cloud/presentations/26.pdf (дата обращения 29.05.2012).
10. Almeer, M.H. Cloud Hadoop Map Reduce For Remote Sensing Image Analysis / M.H. Almeer // J. of Emerging Trends in Computing and Information Sciences. – 2012. – V. 3, №4. – P. 637 – 644.
11. Experiences on Processing Spatial Data with MapReduce / A. Cary, Z. Sun, V. Hristidis, N. Rishe // Proceedings of the 21st International Conference on Scientific and Statistical Database Management. Springer-Verlag Berlin, Heidelberg. – 2009. – P. 302–319.
12. HIPI: A Hadoop Image Processing Interface for Image-based MapReduce Tasks / C. Sweeney, L. Liu, S. Arietta, J. Lawrence, B.S. Thesis. – University of Virginia. Department of Computer Science. – 2011. – P. 5.
13. Adrian, R.J. Particle Image Velocimetry / R.J. Adrian, J. Westerweel. – Cambridge University Press. – 2011. – 584 p.
14. PIV Challenge [Электронный ресурс]. URL: http://www.pivchallenge.org (дата обращения 29.05.2012).
15. Мизева, И.А. Вейвлетные корреляции двумерных полей / И.А. Мизева, Р.А. Степанов, П.Г. Фрик // Вычислительные методы и программирование. – 2006. – Т. 7. – С. 172–179.
16. Шестаков, А.Л. Южно-Уральский государственный университет как стартовая площадка энергосберегающих технологий и использования возобновляемых источников энергии / А.Л. Шестаков, И.М. Кирпичникова // Альтернативная энергетика и экология. – 2010.– №1. – С. 149–152.

Андрей Владимирович Созыкин, кандидат технических наук, отдел вычислительной техники, Институт математики и механики УрО РАН (г. Екатеринбург, Российская Федерация), avs@imm.uran.ru.

Михаил Людвигович Гольдштейн, кандидат технических наук, отдел вычислительной техники, Институт математики и механики УрО РАН (г. Екатеринбург, Российская Федерация), mlg@imm.uran.ru.

MSC 65Y05

MapReduce-based Image Processing System with Automated Parallelization

A.V. Sozykin, Institute of Mathematics and Mechanics,
Ural Branch of the Russian Academy of Sciences (Yekaterinburg, Russian Federation)**,**
M.L. Goldshtein, Institute of Mathematics and Mechanics,
Ural Branch of the Russian Academy of Sciences (Yekaterinburg, Russian Federation)

The article describes a parallel image processing framework based on the Apache Hadoop and the MapReduce programming model. The advantage of the framework is an isolation of the details of the parallel execution from the application software developer by providing simple API to work with the image, which is loaded into memory.
The main results of the work are the architecture of the Hadoop-based parallel image processing framework and the prototype implementation of this architecture. The prototype has been used to process the data from the Particle image velocimetry system (the data from the PIV challenge project have been used). Evaluation of the prototype on the four-node Hadoop cluster demonstrates near linear scalability.
The results can be used in science (processing images from the physics experimental facilities, astronomical observations, and satellite pictures of a terrestrial surface), in medical research (processing images from hi-tech medical equipment), and in enterprises (analysis of data from security cameras, geographic information systems, etc.).
The suggested approach provides the ability to increase the performance of image processing by using parallel computing systems, and helps to improve the work efficiency of the application developers by allowing them to concentrate on the image processing algorithms instead of the details of parallel implementation.

Keywords: image processing, MapReduce, Hadoop, distributed file system, automated parallelization.

References

1. Horowitz M. *The Peta Age.* Available at: http://www.wired.com/images/press/pdf/petaage.pdf (accessed 29 May 2012).
2. *Apache Hadoop.* Available at: http://hadoop.apache.org/ (accessed 29 May 2012).
3. Dean J., Ghemawat S. MapReduce: simplified data processing on large clusters. // *Commun. ACM*, 2008, no. 51(1), pp. 107–113.
4. Ghemawat S., Gobioff H., Leung S.T. The Google File System. *19th ACM Symposium on Operating Systems Principles*, Lake George, NY, 2003, pp. 29–43.
5. *Hadoop Distributed File System.* Available at: http://hadoop.apache.org/hdfs (accessed 29 May 2012).
6. Hays J., Efros A.A. IM2GPS: Estimating Geographic Information from a Single Image. *Proceedings of the IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2008, pp. 1–8.
7. Wiley K., Connolly A., Gardner J., Krughoff S., Balazinska M., Howe B., Kwon Y., Bu Y. Astronomy in the Cloud: Using MapReduce for Image Co-Addition. *Publications of the Astronomical Society of the Pacific*, 2011, vol. 123, no. 901, pp. 366–380.

8. Wiley K., Connolly A., Krughoff S., Gardner J., Balazinska M., Howe B., Kwon Y., Bu Y. Astronomical Image Processing with Hadoop. *Astronomical Data Analysis Software and Systems XX. ASP Conference Proceedings* 2011, vol. 442, pp.93–98.
9. Kumar S., Fernandez B.A. *Distributed image processing using hadoop mapreduce framework.* Available at: http://search.iiit.ac.in/cloud/presentations/26.pdf (accessed 29 May 2012).
10. Almeer M.H. Cloud Hadoop Map Reduce For Remote Sensing Image Analysis. *J. of Emerging Trends in Computing and Information Sciences*, 2012, vol. 3, no. 4, pp. 637–644.
11. Cary A., Sun Z., Hristidis V., Rishe N. Experiences on Processing Spatial Data with MapReduce. *Proceedings of the 21st International Conference on Scientific and Statistical Database Management*, 2009, pp. 302–319.
12. Sweeney C., Liu L., Arietta S., Lawrence J., Thesis B.S. HIPI: A Hadoop Image Processing Interface for Image-based MapReduce Tasks. *University of Virginia. Department of Computer Science*, 2011, pp. 5.
13. Adrian R.J., Westerweel J. *Particle Image Velocimetry.* Cambridge University Press, 2011. 584 p.
14. *PIV Challenge.* Available at: http://www.pivchallenge.org (accessed 29 May 2012).
15. Mizeva I.A., Stepanov R.A., Frick P.G. Wavelet Crosscorrelations of Two-dimensional Signals [Vejvletnye Krosskorreljacii Dvumernyh Polej]. *Vychislitel'nye Metody i Programmirovanie [Numerical Methods and Programming]*, 2006, vol. 7, pp. 172–179.
16. Shestakov A.L., Kirpichnikova I.M. South Ural State University as Starting Platform of Power Saving Technologies and Use of Renewable Energy Sources [Yuzhno-Ural'skiy Gosudarstvennyy Universitet kak Startovaya Ploshchadka Energosberegayushchikh Tekhnologiy i Ispol'zovaniya Vozobnovlyaemykh Istochnikov Energii]. *Alternative Energy and Ecology [Al'ternativnaya Energetika i Ekologiya]*, 2010, no. 1, pp. 149–152.

Поступила в редакцию 8 июня 2012 г.