СИСТЕМА АКТИВНОГО ХРАНЕНИЯ ДАННЫХ НА БАЗЕ БИБЛИОТЕКИ ДИНАМИЧЕСКОГО РАСПАРАЛЛЕЛИВАНИЯ TSIM

А.А. Московский, Е.О. Тютляева , Е.В. Шевчук

Введение

Идея "умного" хранилища данных была разработана несколькими авторами в конце 90-х годов XX столетия на основе идей, которые высказывались уже в начале 80-х годов в мире баз данных. Тогда была предложена концепция "активных дисков"[1], базирующаяся на идее использования вычислительной мощности дисков-хранилищ для запуска обрабатывающих (оптимизирующих) приложений. Однако с течением времени, по ряду причин большинство производителей дисков-хранилищ перестали предоставлять требуемую программную и аппаратную поддержку, что не позволило концепции "активных дисков" стать повсеместно используемой. Более успешное воплощение данная концепция получила применительно к параллельным файловым системам на вычислительных установках кластерного типа. Одним из примеров является проект "Активное хранение и обработка в параллельных файловых системах" 2006 года [2], основанный на добавлении дополнительного уровня к файловой системе Lustre, который позволяет создавать вместе с файлом данных обрабатывающую компоненту(Active Storage Processing Component), с которой пользователь может работать, используя специальную среду, предоставляемую разработчиками. Следует отметить, что данная разработка, также как и концепция "активных дисков", была нацелена на оптимизацию процесса чтения/записи данных.

Идея создания активных хранилищ, нацеленных на хранение больших объемов данных на кластере (либо в вычислительной сети) и эффективную их обработку при помощи задействованных вычислительных узлов, которая используется в данной работе, появилась сравнительно недавно. Среди наиболее популярных проектов, реализующих данную концепцию, следует назвать Active Storage[3], Pig[4], Cascading[5].

Как показывает практика, использование распределенных активных хранилищ позволяет создавать масштабируемые, высокоскоростные, с высокой пропускной способностью, управляемые распределенные системы. Преимущества активного распределенного хранения данных достигаются за счет организации обработки этих данных непосредственно на узлах хранения и сокращения непроизводительных расходов на передачу данных по сети.

В данной работе предлагается один из возможных подходов к организации активного хранилища. В реализованной системе объединены возможности широко известной кластерной файловой системы Lustre и библиотеки шаблонных классов C++ TSim[6], реализующей автоматическое динамическое распараллеливание программ. Именно библиотека TSim реализует в описываемой системе функцию активного хранения, обеспечивая направление задания по обработке данных на узел хранения в ФС Lustre. Для двух различных способов хранения и обработки данных разработаны подходы по организации активного хранилища (своего рода "шаблоны задач", которыми может воспользоваться пользователь, если способ хранения и обработки данных для его задачи совпадает со способом, предложенным в шаблоне). Причем, сам алгоритм обработки данных может быть любым, лишь бы он удовлетворял условиям шаблона.

Отличительной особенностью данного проекта является нацеленность на данные дистанционного зондирования Земли; система позволяет повысить эффективность обработки космических снимков, а также, применима для других задач, связанных с обработкой больших массивов графических данных.

Структура активного хранилища

Основными компонентами разработанной программной системы являются:

1. Кластерная файловая система Lustre, которая отвечает за распределенное хранение данных на узлах.
2. Библиотека шаблонных классов C++ TSim, базирующаяся на концепции автоматического динамического распараллеливания.

Пользователю предоставляются два шаблона задач - два подхода по организации активного хранилища, которые позволяют эффективно работать с двумя классами задач по обработке данных дистанционного зондирования Земли.

В данной системе в качестве интерфейса между пользователем и хранилищем выступает библиотека TSim. Благодаря этому, пользователь получает многие преимущества автоматического динамического распараллеливания, обеспечиваемые библиотекой:

- использование бесконфликтной модели вычислений на основе чистых (не имеющих побочных эффектов) функций и "неготовых значений" как средства синхронизации доступа к результатам вычислений;
- использование уже реализованных стратегий выравнивая нагрузки на вычислительных узлах и конструирование новых стратегий при помощи средств библиотеки TSim (в рамках создания данной

системы особенное внимание уделялось стратегии выравнивания нагрузки, которая направляет вычисления на узел хранения обрабатываемых данных);

– использование в своих программах высокоуровневых шаблонов (скелетонов) параллельного программирования (например, шаблона Map[7]), которые используют TSim в качестве более низкоуровневого средства.

Шаблоны задач для активного хранилища

В качестве примера использования библиотеки шаблонных классов C++ TSim для эффективной работы с данными, расположенными на кластере при помощи ФС Lustre, было реализовано два шаблона обработки данных дистанционного зондирования Земли. Под "шаблоном" в данном случае понимается некая схема организации активного хранилища, характеризующаяся определенным способом размещения данных в ФС Lustre и способом их обработки. Эту схему можно использовать для решения некоторых других задач, а не только тех, для которых она разрабатывалась, что и позволяет говорить о ней как о шаблоне.

Данные шаблоны реализуют следующую схему обработки, которую рекомендуется использовать и для реализации иных, не попадающих под решение при помощи шаблонов, задач:

– Задается разбиение для данных, расположенных на кластере, при помощи ФС Lustre, наиболее отвечающее особенностям решаемой задачи.

– В обрабатывающей программе при помощи функций интерфейса ФС Lustre, определяется схема расположения файлов в хранилище. Ориентируясь на данную схему можно однозначно определить, на каком узле хранилища будет расположена обрабатываемая часть файла.

– По номеру узла определяется его IP-адрес, и, при помощи планировщика TSim, задача по обработке данного фрагмента файла отправляется на найденный узел.

В настоящий момент, в системе предоставляются готовые решения (шаблоны) для следующих типов задач:

- Для задач обработки TIFF-файлов достаточно большого размера, обработку которых можно распараллелить (исходный файл с данными разбить на порции и осуществить их обработку в независимых потоках).
- Для задач, которые требуют многократного запуска последовательного кода для различных наборов данных.

Шаблоны реализованы на основании реальных задач обработки данных ДЗЗ, отвечающих указанным требованиям. Ниже дано краткое описание задач и методов их решения, оформленных в виде шаблонов для повторного использования.

Задача классификации космических снимков по метрике Махаланобиса

Метрика Махаланобиса- это особый вид расстояния, который активно применяется в статистике. Расстояние Махаланобиса основано на корреляции, благодаря которой можно анализировать различные сложные структуры данных. Эта метрика отличается от Евклидового расстояния тем, что учитывает корреляции внутри анализируемого множества. Параллельный классификатор использует расстояние Махаланобиса для определения принадлежности текущего обрабатываемого элемента множества к одному из выбранных экспертом подмножеств из всего множества входных данных. Такой выбор осуществляется с помощью определения минимального из расстояний Махаланобиса от обрабатываемого элемента до каждого из выбранных экспертом подмножеств. В случае если найденное минимальное расстояние оказывается больше выбранного экспертом порога, текущий элемент считается неклассифицированным. Такой метод позволяет поставить в соответствие изображение на космических снимках реальным географическим объектам - озерам, рекам, горам и т.п.

Технически, в основе данной задачи лежит независимая обработка множества пикселей, поэтому для решения был использован шаблон параллельного программирования Map, который для каждой порции исходных данных (в нашем случае для каждой полоски из исходных TIFF-файлов), нуждающихся в обработке, будет при помощи библиотеки libtiff определять, на каких из узлах хранилища она расположена и стремиться максимально приблизить вычисления по обработке данной полоски к узлу ее расположения.

Данный подход оформлен в виде независимого файла-шаблона, который можно использовать для решения технически аналогичных задач. Функцию обработки данных (в нашем случае это алгоритм обработки по метрике Махалонобиса) пользователь может определить самостоятельно.

Задача перепроецирования и склейки данных дистанционного зондирования Земли

Результаты обработки спутниковых данных, как правило, представляют собой набор чисел, заданных на определенной сетке, имеющей географическую привязку, т. е., для каждого из узлов известны соответствующие ему географические координаты. При построении изображения нередко требуется перенести данные с исходной сетки на другую, более удобную для представления данных в рамках решаемой задачи. В силу кривизны земной поверхности, данные смежных сканов широкоугольных сенсоров оказываются

географически-накладывающимися друг на друга; сетка является нерегулярной и неудобной для восприятия. Также, нередко возникает задача наложения на результирующем изображении нескольких слоев данных и аннотирования данных, т. е., наложение на географическую сетку контуров территорий и названий населенных пунктов. Для этого данные, также, должны быть приведены к одной проекции. Географическая привязка координатной сетки, на которой представлены данные, может быть осуществлена двумя способами. Во-первых, может быть использован отдельный массив данных, сопоставляющий каждой точке данных ее географические координаты. Во-вторых, могут быть заданы параметры проекции, так что координаты точки в проекции являются одновременно и индексами массива данных. Первый способ используется на предварительных этапах обработки данных и привязан к витку спутника. На более поздних этапах обработки, данные нередко (из соображений удобства совмещения данных разных витков) приводятся к единой координатной сетке (к синусоидальной сетке с определенным масштабом).

Перед разработчиками стояла задача организовать эффективное хранение данных и распараллеливание задачи перепроецирования и склейки итогового изображения.

Для того, чтобы выполнить склейку изображения, требуется провести перепроецирование большого количества независимых изображений, что создает предпосылки для применения распараллеливания для данной задачи.

Для реализации параллельной версии этой задачи при помощи библиотеки TSim был реализован планировщик заданий, который, используя функции интерфейса ФС Lustre, позволяет определить, на каком из узлов хранилища расположен обрабатываемый файл и направить на данный узел вычисления по обработке данного файла. Важно подчеркнуть, что в связи с небольшим размером файлов, при использовании данного шаблона файл располагается на узле целиком. Весь архив обрабатываемых файлов располагается на кластере при помощи алгоритма round-robin, что способствует выравниванию нагрузки.

Для эффективного использования распределенных ресурсов кластера планировщик заданий должен получить информацию о расположении каждого файла с данными. Планировщик был реализован нами в виде интерпретатора, который получает на вход последовательность командных строк, запускающих код обработки на различных наборах данных. В командной строке содержится, кроме всего прочего, информация о полном пути к обрабатываемым данным, которая извлекается синтаксическим анализатором планировщика. Таким образом, для использования данного шаблона для решения других подобных задач (многократный запуск кода на различных наборах данных) достаточно лишь изменить функцию синтаксического анализатора командной строки, если вид строки будет отличаться от вида, предложенного в шаблоне. Важной особенностью данного шаблона является то, что последовательный код обработки вообще не нуждается ни в каких изменениях, что позволяет очень легко организовать параллельную обработку большого набора данных, состоящего из отдельных файлов небольшого размера. Функция обработки может поставляться даже в виде исполняемого файла, совместимого по формату с операционной средой кластера.

Результаты тестирования производительности системы

Для того, чтобы проверить теоретические выкладки по повышению эффективности обработки данных 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 кластерах, технические характеристики которых представлены в Таблице 1.

Таблица 1. Характеристики тестовых установок

Место расположения	ИПС РАН	ИПС РАН
Число вычислительных узлов	2	8
Тип процессора	Intel(R) Xeon(TM) 2.80GHz	Intel(R) Xeon(R) 3.00GHz
Количество ядер	2	4
Количество процесоров в узле	2	2
Оперативная память узла	1 GB	16 GB
Дисковая память установки	250+80 GB	Рейд-массив на 3 TB
Тип системной сети	Gigabit Ethernet	Gigabit Ethernet
Конструктив (форм-фактор	2U	5U

Тестирование классификатора изображений по метрике Махаланобиса.

Данные ДЗЗ при тестировании классификатора 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 распределялись по этим узлам.

Было проведено многократное тестирование системы; усредненные результаты обработки шести снимков по 151 Mb на кластере demo показаны в Таблице 2 и на кластере blade - в Таблице 3.

Таблица 2: Результаты тестирования на кластере demo.botik.ru

Количество узлов	1 узел	2 узла
Время, сек	1650.234	924.166
Процентное соотношение	100,00%	56,00%

Таблица 3: Результаты тестирования на кластере blade.botik.ru

Количество узлов	1 узел	2 узла	4 узла
Время, сек	203.93	136.76	84.57
Процентное соотношение	100,00%	67,00%	41,00%

Тестирование приложения по перепроецированию и склейке изображений.

Для тестирование приложения были предоставлены: 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: каждый файл располагался целиком на узле; файлы были равномерно распределены по узлам с использованием алгоритма round-robin.

Было проведено многократное тестирование системы; усредненные результаты обработки 80 запросов (320 перепроецирований) на кластерах demo и blade показаны в Таблицах 4 и 5, соответственно.

Таблица 4: Результаты тестирования на кластере demo.botik.ru

Количество узлов	1 узел	2 узла
Время, сек	102.996	35.602
Процентное соотношение	100,00%	34,00%

Таблица 5: Результаты тестирования на кластере blade.botik.ru

Количество узлов	1 узел	2 узла	4 узла
Время, сек	33.68	26.95	18.76
Процентное соотношение	100,00%	80,00%	55,00%

Как видно из приведенных таблиц, данный метод позволяет существенно снизить расходы на передачу больших массивов данных, что позволяет повысить эффективность работы системы и получить значительно меньшее время выполнения задачи на нескольких узлах по сравнению с последовательной версией.

Следует отметить, что тестирование показало также, что распараллеливание перепроецирования небольшого количества данных не дает выигрыша, т.к. затраты на распараллеливание сопоставимы с затратами на вычисления, выполняемые на одном узле. Поскольку общий объем данных небольшой, объем данных, обрабатываемых на одном узле, и соответственно, время, затраченное на их обработку, также невелико. Однако, при росте общего объема данных наблюдается значительный рост эффективности работы системы, что и продемонстрировано в приведенных таблицах. Следовательно, использование распределенного архива изображений с возможностью его эффективной обработки целесообразно при большом объеме хранимых данных.

Заключение

В данной статье рассматривается действующая программная система, разработанная и реализованная авторами, объединяющая кластерную файловую систему Lustre и ряд шаблонов и приложений, написанных с использованием библиотеки для автоматического распараллеливания TSim.

Предлагаемая программная система представляет собой распределенный архив данных ДЗЗ, организованный в соответствии с идеологией 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 по сети. Балансировка нагрузки на узлах осуществляется планировщиком библиотеки TSim на основе информации, предоставляемой интерфейсом кластерной файловой системы Lustre.

Для решения различных видов задач обработки космических снимков, хранящихся в распределенном архиве, реализованы две стратегии распараллеливания, оформленные в виде шаблонов задачи. Применение шаблонов задачи позволяет быстро и эффективно реализовать параллельные вычисления для задач обработки, удовлетворяющих условиям применения шаблонов. Кроме того, пользователь имеет возможность реализовать свой собственный способ организации активного хранилища используя в качестве инструменальных средств ФС Lustre, библиотеку TSim и высокоуровневые шаблоны параллельного программирования.

В качестве примеров использования предлагаемых шаблонов были реализованы задача контролируемой классификации изображений ДЗЗ при помощи метрики Махаланобиса и задача представления данных ДЗЗ в нужном масштабе и проекции. Тестирование задач, которое проводилось на двух различных установках, продемонстрировало повышение производительности приложения при использовании предлагаемых шаблонов задач.

Целесообразность использования описанной программной системы, как показало тестирование производительности, возрастает при работе с большими объемами исходных данных или при большой вычислительной сложности задач обработки изображения, что, впрочем, справедливо для большинства систем, использующих параллельные вычисления.

Благодарности

Работа выполнена при поддержке Программы "1 фундаментальных исследований Президиума РАН "Проблемы создания национальной научной распределенной информационно-вычислительной среды на основе развития GRID-технологий и современных телекоммуникационных сетей" и проекта РФФИ 07-07-12038-офи "Метапрограммирование на основе шаблонных классов C++ как средство создания высокопроизводительных распределённых приложений".

ЛИТЕРАТУРА:

1. E. Riedel, G. Gibson, and C. Faloutsos. Active storage for large-scale data mining and multimedia. In Proc. of the 24th Int. Conf. on Very Large Data Bases (VLDB), pages 62-73, 1998.
2. E.J. Felix, K. Fox, K. Regimbal, and J. Nieplocha. Active storage processing in a parallel file system. In Proc. of the 6th LCI International Conference on Linux Clusters: The HPC Revolution., 2006.
3. Piernas, J., Nieplocha, J., Active Storage User's Manual, Pacific Northwest National Laboratory. http://hpc.pnl.gov/projects/active-storage/as_users_manual_october_2007.pdf
4. Cascading. http://www.cascading.org/,
5. Welcome to Pig! http://hadoop.apache.org/pig
6. А.А. Московский. 2007. Реализация библиотеки для параллельных вычислений на основе шаблонных классов языка C++. Proc. Труды Международной научной конференции "Параллельные вычислительные технологии (ПаВТ'2007)", Челябинск, 29 января - 2 февраля 2007 г., Челябинск, изд. ЮУрГУ, Т 2, с. 256
7. А.А. Московский, А.Ю. Первин, Е.О. Сергеева. 2006. Первый опыт реализации шаблона параллельного программирования на основе Т-подхода. Proc. Международная конференция "Программные системы: теория и приложения", Переславль-Залесский, октябрь 2006, Наука,-Физматлит, М.. Т. 1, с. 245-255