ПРИМЕНЕНИЕ КОНЦЕПЦИИ АКТИВНЫХ ХРАНИЛИЩ В ЗАДАЧАХ ОБРАБОТКИ ДАННЫХ СЕЙСМИЧЕСКИХ НАБЛЮДЕИЙ

Е.О. Тютляева, А.А. Московский, С.С. Конюхов, Е.А. Курин

Введение

Решение задач обработки данных, полученных при сейсмической разведке, предъявляет высокие требования к вычислительным ресурсам. С одной стороны это связано с большими объемами данных сейсмических наблюдений и их первичной обработки, которые достигают сотен терабайт. С другой стороны сложность обусловлена свойствами применяемых на практике математических методов, требующих произвести большой объем вычислений в достаточно сжатые сроки.

Например, в 2009 г. на месторождении Карачаганак (Казахстан) проводились трехмерные наблюдения на площади около 900 кв. км. При этом были зарегистрированы около трех миллиардов записей (трасс). Общий объем данных превысил 100 терабайт [1]. Другим примером может служить работа [2], где приводится сопоставление различных методов построения глубинного изображения среды по сейсмическим данным и необходимых вычислительных ресурсов.

Приведенные в работах [1,2] примеры показывают, что в ряде случаев применение суперкомпьютеров может значительно улучшить качество результатов обработки сейсмических данных, а также обеспечить решение задач в экономически оправданные сроки.

Многообразие решаемых задач в сейсморазведке порождает многообразие применяемых вычислительных методов и программ обработки сейсмических данных. Одним из популярных пакетов по обработке сейсмических данных является свободно распространяемый пакет Seismic Un*x [3], создание которого началось более 24-х лет назад в Center for Wave Phenomena Colorado School of Mines. В настоящее время данный пакет содержит большинство необходимых операций над сейсмическими данными и активно используется и обновляется специалистами по сейсмологии со всего мира. Seismic Un*x позволяет строить комплексные задания обработки данных из набора простых процедур при помощи конвейеров Unix.

При обработке сейсмических данных при помощи стандартного набора процедур из пакета Seismic Un*x возможна обработка отдельных порций данных независимо друг от друга. Для этого класса задач узким местом является подсистема ввода-вывода.

Одним из возможных подходов к преодолению данного узкого места является концепция систем активного хранения данных. Основная идея системы активного хранения данных в контексте параллельных файловых систем заключается в использовании вычислительных ресурсов узлов хранения для обработки данных, расположенных на данном узле. Предложенный подход позволяет использовать вычислительный кластер как для хранения, так и обработки данных.

Обработка данных непосредственно на узлах хранения позволяет продемонстрировать значительную эффективность за счет минимизации количества дорогостоящих операций передачи данных по сети. Дополнительным достоинством при обработке значительных объемов данных может являться своевременная доставка данных для обработки к вычислительному процессору, и связанное с этим снижение вероятности простоя вычислительных ресурсов в ожидании доставки данных.

Описанная концепция реализована в системе активного хранения данных, базирующейся на библиотеке шаблонных классов C++ TSim для распараллеливания вычислений и кластерной файловой системе Lustre для обеспечения высокопроизводительного масштабируемого ввода-вывода.

Разработанный прототип системы и опыт ее использования для хранения и обработки данных дистанционного зондирования уже были представлена на конференции «Научный сервис в сети интернет» [4]. В развитие предшествующих результатов, в данной статье будет представлен ряд модификаций данной системы и интеграция данного подхода со свободно распространяемым пакетом Seismic Un*x, предназначенным для обработки сейсмических данных.

Структура данных и вычислительных модулей в Seismic Un*x

Базовой структурной единицей хранения сейсмических данных в формате Seismic Un*x (SU) является трасса. Каждая трасса содержит заголовок фиксированного размера и бинарные данные, отражающие результаты измерения с описанными в заголовке параметрами. Файл SU содержит набор трасс с результатами различных измерений, проведенных в ходе эксперимента. Большинство программ из пакета Seismic Un*x обрабатывают трассы (все, или отвечающие определенным критериям) последовательно, независимо друг от друга. Типовая схема работы с сейсмическими данными приведена на листинге 1.

```
/* Считывание первой трассы; инициализация основных параметров */
if (!gettr(&tr)) err("can’t read first trace"); [9]
f (!tr.dt) err("dt header field must be set"); [10]
/* Loop over traces */
do {
```

```
 /* Обработка трассы, подготовка модифицированного результата на запись*/
}
puttr(&tr);  //запись обработанной трассы в файл результата
} while (gettr(&tr)); //считывание следующей трассы для обработки
```

Листинг 1. Типовая схема обработки сейсмических данных

Поскольку, как видно из типовой схемы, трассы в большинстве случаев обрабатываются независимо, возможно эффективно организовать распределенные вычисления в системе активного хранения данных.

Принципы интеграции пакета Seismic Un*x и системы активного хранения

Большое значение при интеграции имела возможность организации распределенных вычислений без модификации исходных кодов пакета Seismic Un*x. Система активного хранения на базе TSim предоставляет такую возможность при обработке разных файлов, распределенных по узлам вычислительного кластера.

Поскольку данные сейсмических наблюдений обычно представлены в виде одного файла в формате SU, для удобства распределенной обработки в системе активного хранения файл был разбит на фрагменты, количество которых было выбрано кратным количеству узлов вычислительного кластера. Разбиение выполняется при помощи утилиты suwind пакета Seismic Un*x.

```
suwind skip=0 count=18400 < File.su > /mnt/lustre/seismo/out0.su
```

Полученные фрагменты исходного файла были размещены в параллельной файловой системе Lustre для последующего хранения и обработки. Перед размещением файлов была задана схема разбиения таким образом, чтобы каждый полученный фрагмент лежал целиком на одном узле, при этом чтобы все фрагменты были распределены равномерно по всем узлам вычислительного кластера.

Предложенный подход имеет ряд преимуществ:

1. допускается постепенный перенос данных и обработки в кластер: часть данных и обработчиков могут находиться в «традиционном» серверном Seismic Un*x, а часть быть реализована в активном хранилище;
2. размещение данных прозрачно для ОС и для пользователя: файлы доступны средствами ФС Lustre;
3. за счет возможностей ФС Lustre возможна интеграция в высокопроизводительные вычислительные комплексы (например, за счет интерфейса MPI-IO).

Для дальнейшей обработки файлов достаточно вызывать планировщик системы активного хранения на базе библиотеки TSim. При вызове планировщика ему необходимо передать полный путь к обрабатываемым файлам данных, расположенным в ФС Lustre, конвейер обработки и список файлов, к которым необходимо применить данный конвейер. Далее применение заданного конвейера обработки к заданным файлам будет выполнено автоматически, в соответствии с логикой активного хранения данных. Упрощенная схема распределения данных по узлам вычислительного кластера и последующей обработки изображена на рисунке 1.

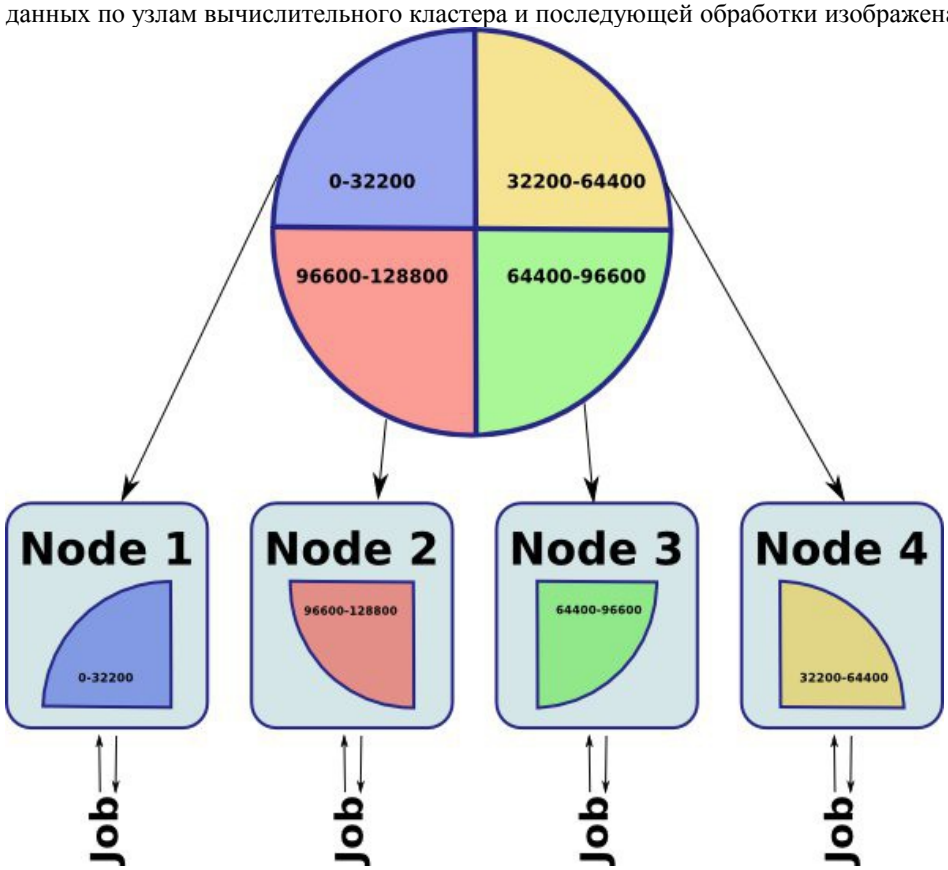

Рис. 1

Алгоритм распределенной обработки состоит из следующих этапов:

1. На первом шаге для обработки каждого файла порождается отдельная параллельная задача
2. Для каждой задачи планировщик системы активного хранения определяет схему расположения обрабатываемого файла в ФС Lustre, в частности, на каком устройстве хранения расположен данный файл.
3. По конфигурационным настройкам Lustre планировщик определяет, к какому вычислительному узлу относится определенное устройство хранения и перенаправляет вычисления на заданный узел.

Предварительная оценка производительности

Описанная схема распределенной обработки была применена для обработки файла сейсмических данных размером в 767 Мегабайт на вычислительном кластере, обладающем следующими характеристиками:

Расположение: ИПС им. А.К. Айламазяна РАН

Количество узлов: 7

Тип вычислительных процессоров: Intel Xeon E5410 (12M L2 Cache, 2.33 GHz,1333 MHz FSB, 4 ядра) - 10 шт., на 5 узлах

Intel Xeon X5570 (8M Cache, 2.93 GHz, 6.40 GT/s Intel QPI, 4 ядра) – 4 шт., на 2 узлах

Количество вычислительных ядер: 56

Объем ФС Lustre: 6.3 TB

Интерконнект: Gigabit Ethernet

Форм-фактор: Full ATX

Файл данных был разбит на 7 частей по 110 Мегабайт в соответствии с доступным количеством вычислительных узлов кластера. Полученные фрагменты были распределены по одному на каждый узел средствами ФС Lustre.

Конвейер обработки содержал только те команды, которые можно применять к каждой трассе независимо и имел вид:

sufilter f=3,7,23,30 amps=0.5,1,1,0 < filei.su| supef maxlag=MAXLAG_SPIKING | sunmo cdp=600,1200 tnmo=0,1.2,1.8,2.4,2.9,3.6,4.0,4.3 vnmo=1500,150 0,1630,1770,1925,2080,2350,2280 tnmo=0,0.7,1.4,2.06,2.46,3.37,3.88,4.07 vnmo=1500,1500,1670,1780,1900,2130,2250,2220 | sugain tpow=2 agc=1 | sustack |sufilter f=3,7,23,30 amps=0.5,1,1,0

Данный конвейер применялся параллельно ко всем частям исходного файла данных.

Последовательная обработка целого файла при помощи приведенного конвейера занимает 21.784 секунд.

Параллельная обработка по указанной схеме в системе активного хранения занимает 4.400 секунд.

Параллельная обработка при помощи планировщика “барабанного” типа (не учитывающего расположения файлов) занимает 11.009 секунд. Важно отметить, что файловая система Lustre позволяет клиенту выполнять операции с данными вне зависимости от того, где они расположены, что позволило обеспечить корректную обработку данных в случае «барабанного» планировщика без дополнительных операций с данными

Приведенные значения измерения времени являются усредненными результатами по 10 экспериментам.

Следует отметить, что даже при таком незначительном времени обработки файла удалось получить повышение эффективности при использовании системы активного хранения данных, несмотря на необходимые накладные расходы, связанные с планированием и пересылкой задач на вычислительные узлы. Мы надеемся, что при проведении экспериментов с большими объемами данных, удастся получить ускорение, близкое к линейному.

Полученные предварительные результаты продемонстрировали целесообразность использования системы активного хранения для обработки сейсмических данных.

Сортировка данных в активном хранилище

Программы из набора Seismic Un*x, которые обладают более сложными зависимостями по данным чем описано в предыдущем разделе, очевидно, более сложны для распараллеливания. Характерным примером является операция сортировки трасс в файле (модуль susort), которая используется в большинстве конвейеров обработки.

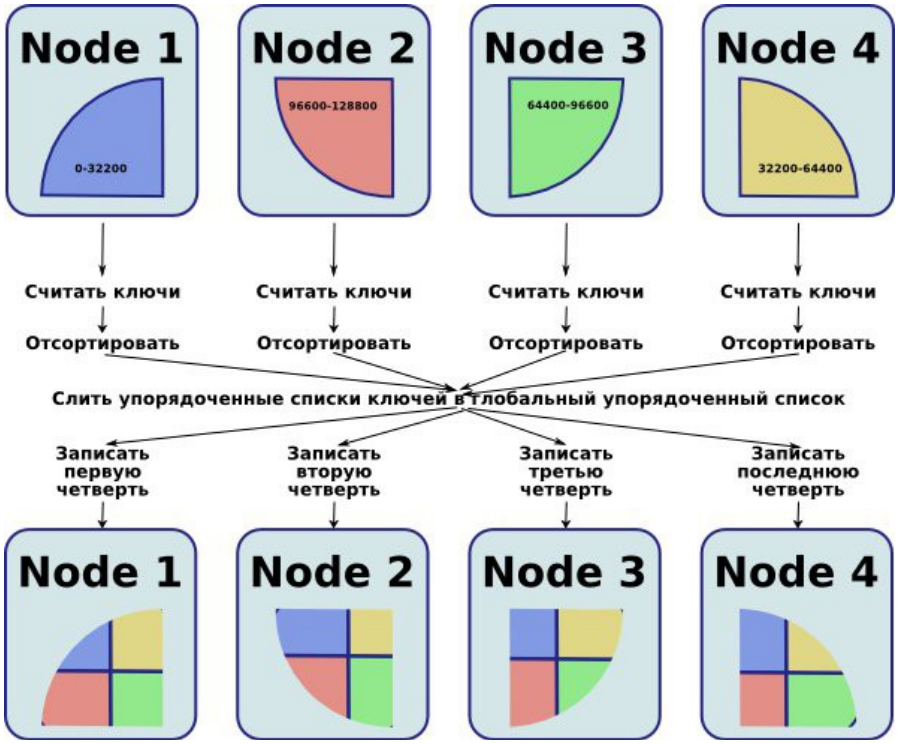

Рис. 2

При сортировке файла сейсмических данных при помощи операции susort сначала считываются значения тех ключей, по которым будет проводиться сортировка. Затем множество ключей сортируется в соответствии с заданным порядком, при этом с каждым ключом связан номер трассы, к которой данный ключ относится. Наконец, трассы исходного файла считываются в соответствии с выстроенным после сортировки порядком и записываются в результирующий файл.Для распараллеливания данной операции в системе активного хранения была реализована следующая схема (см. рисунок 2):

1) На первом шаге производится считывание значений выбранных ключей для каждого из фрагментов исходного файла на том узле, где он расположен. На этом шаге следует ввести сквозную нумерацию по всем обрабатываемым фрагментам и связать с каждым ключом уникальный номер.
2) Затем выполняется локальная сортировка считанных подмножества на соответствующих узлах хранения.
3) Затем производится слияние упорядоченных подмножеств в единое отсортированное множество. (В связи с линейной сложностью операции слияния упорядоченных списков данную операцию не следует распределять по всем доступным узлам. Данная операция может быть выполнена на одном или нескольких узлах в зависимости от размера исходного файла)
4) На заключительном шаге выполняется запись фрагментов упорядоченного множества на узлах хранения. Для определения, к какому исходному файлу относилась рассматриваемая трасса, был использован уникальный номер заданный на этапе сквозной нумерации. (Отсортированное множество трасс записывается таким образом, чтобы на i-ом узле кластера были сохранены трассы с (i-1)*size/N по i*size/N, где N – количество доступных узлов, а size – общее количество трасс в исходном файле с данными.)

Таким образом, после операции сортировки на каждом узле расположен фрагмент содержащий size/N трасс, что позволит проводить дальнейшие операции обработки над каждым фрагментом по отдельности на узлах хранения. При этом конкатенация всех фрагментов позволит получить полностью отсортированный файл.

Для реализации первого, второго и последнего шага из приведенной схемы был использован шаблон параллельного программирования Map [5], входящий в состав библиотеки Tsim.

Предварительное тестирование полученной операции сортировки в системе активного хранения данных применительно к файлу размером 767M продемонстрировало следующие результаты:

Последовательное выполнение: 1 мин. 10 сек.

2 узла (данные разбиты на 2 фрагмента): 38 сек.

7 узлов (данные разбиты на 7 фрагментов): 14 сек.

Надежность и отказоустойчивость

Одним из ключевых требований при операциях с сейсмическими данными является надежность и отказоустойчивость системы. Так как обработка данных современных сейсмических наблюдений занимает

значительное время [1], то выход из строя элементов вычислительного оборудования приводит к значительным экономическим потерям из-за несоблюдения сроков выполнения работ.

Для системы активного хранения разработан специальный модифицированный планировщик с функцией распознавания отказа одного или нескольких узлов. Для того чтобы своевременно распознать отказ узла планировщик с заданным интервалом времени рассылает активные сообщения, сообщающие о статусе отправки. Узел считается отказавшим, если на него не удается успешно передать сообщение либо очередную задачу.

Для продолжения работы системы после отказа одного из вычислительных узлов в TSim существует стандартный тип “задача с сохранением". В случае отказа какого-либо вычислительного узла, все назначенные на него и не завершенные на момент отказа задачи распределяются между оставшимися узлами в соответствии с алгоритмом планирования.

Отличительной особенностью системы активного хранения является расположение устройств хранения на вычислительных узлах. Соответственно, при отказе узла данные, расположенные на этом узле, становятся недоступны для дальнейшей обработки. Следовательно, сохранение только задач, без входных данных, не имеет смысла. С целью поддержки механизмов отказоустойчивости на уровне задач, создан специализированный класс C++ TaskSaved.

При создании задачи типа TaskSaved планировщик не только определяет, на каком узле расположены входные данные для указанной задачи и отправляет задачу на выявленный узел, но и выполняет копирование входных данных на устройство хранения, расположенное на другом узле. В случае отказа одного из вычислительных узлов в системе незавершенные задачи перенаправляются на тот узел, где расположена сохраненная копия. Такой подход позволяет эффективно организовать отказоустойчивые вычисления в системе активного хранения данных, но он связан с неизбежно высокими накладными расходами на создание и поддержание копий. Копия может удаляться как по завершении вычислительной задачи, с которой она связана, так и после завершения вычисления в целом.

Мы проводили теоретические исследования, нацеленные на минимизацию накладных расходов при организации отказоустойчивого вычисления.[6] Мы ведем работы по расширению данных исследований в приложении к обработке сейсмических данных. Естественной точкой для повышения надежности системы представляется сортировка сейсмических данных, которая приводит к дублированию входных данных, только в отсортированном формате. Сложность заключается в том, что исходные и отсортированные фрагменты имеют разное расположение трасс относительно вычислительных узлов.

Обзор аналогов

Наиболее близким аналогом является проект Seismic Hadoop, предложенный в начале этого года [7]. В рамках указанного проекта была проведена интеграция пакета Seismic Un*x и библиотеки Java Crunch, нацеленной на организацию MapReduce конвейеров. Seismic Hadoop использует вычислительный кластер Hadoop для хранения и обработки данных. Hadoop обеспечивает автоматическое резервирование данных и оптимизирован для выполнения Map Reduce операций.

Среди отличительных особенностей нашего подхода можно выделить:

1. Для организации распределенных вычислений используется библиотека шаблонных классов C++ TSim, что предоставляет нам возможности:
   - a. Быстрой смены стратегии планирования, разработки собственной стратегии в виде шаблонного параметра планировщика.
   - b. В классических случаях возможно использовать шаблоны распараллеливания Map и Reduce. При необходимости реализации более сложных схем распараллеливания возможно использование параллелизма по задачам, неготовых переменных в качестве примитивов синхронизации, активных сообщений и т.п.
   - c. Использование всех возможностей шаблонов C++, легкая интеграция с исходными кодами библиотеки Seismic Un*x, написанной на C.
2. В качестве уровня хранения в нашей системе используется кластерная файловая система Lustre, которая установлена на ведущих суперкомпьютерах мира, таких как K computer (модифицированная версия ФС Lustre, называемая FEFS), Tianhe-1A, Jaguar (модифицированная версия – Spider), TSUBAME2.0 и др. Использование Lustre обеспечивает высокую производительность операций с данными и высокий потенциал масштабируемости.

Приведенные отличительные черты позволяют предположить, что интеграция системы активного хранения данных с пакетом Seismic Un*x может позволить исследовать эффективность различных комбинаций алгоритмов планирования и реплицирования данных и реализовать эффективные схемы распараллеливания для сложных подпрограмм обработки сейсмических данных.

Заключение

В работе предложен подход для организации распределенной обработки сейсмических данных на базе свободно распространяемого пакета Seismic Un*x и системы активного хранения данных с использованием TSim и ФС Lustre.

Несмотря на то, что работы по интеграции предложенных систем еще не завершены, на данном этапе рассмотрены ключевые проблемы обработки сейсмических данных, и для каждой предложено и обосновано использование соответствующего инструмента из арсенала системы активного хранения данных. Было проведено исследование производительности разработанного программного прототипа по обработке сейсмических данных в системе активного хранения для оценки перспектив интеграции. Результаты эксперимента демонстрируют повышение производительности при использовании архитектуры активного хранения данных как по сравнению с последовательной обработкой, так и с распределенной обработкой без привязки к местам расположения данных.

В самое ближайшее время планируется провести тестирование производительности на реальных данных размером от нескольких сотен гигабайт до нескольких терабайт. В дальнейшем планируется разработка и реализация эффективных схем обеспечения надежности и отказоустойчивости.

Работа проводилась при финансовой поддержке Министерства образования и науки Российской Федерации, Государственный контракт № 07.514.12.4007. Работа проводилась при финансовой поддержке Программы фундаментальных исследований Президиума РАН № 14

ЛИТЕРАТУРА:

1. Е.А. Курин: Сейсморазведка и суперкомпьютеры // Вычислительные методы и программирование. - 2011. - Том 12 № 1. - С. 38-43, УДК 519.6
2. Camp W.J, Thierry P. Trends for high-performance scientific computing // The Leading Edge. 2010. 29. 44–47
3. Seismic Un*x Home Page, http://www.cwp.mines.edu/cwpcodes/
4. А.А. Московский, Е.О. Тютляева , Е.В. Шевчук.: СИСТЕМА АКТИВНОГО ХРАНЕНИЯ ДАННЫХ НА БАЗЕ БИБЛИОТЕКИ ДИНАМИЧЕСКОГО РАСПАРАЛЛЕЛИВАНИЯ TSIM//Научный сервис в сети Интернет: масштабируемость, параллельность, эффективность: Труды Всероссийскойсуперкомпьютерной конференции (21-26 сентября 2009 г.,г. Новороссийск).– М.: Изд-во МГУ, 2009. – 226-230 с., ISBN 978-5-211-05697-8
5. Московский А.А., Первин А.Ю., Тютляева Е.О.: Параллельная реализация модели map-reduce с использованием шаблонных классов с++//Программные продукты и системы.-Тверь.- 2009.-№ 2(86)-С.53-57. Под редакцией С.В. Емельянова, ISSN 0236-235X
6. Ekaterina Tyutlyaeva and Alexander Moskovsky: An Initial Approximation to the Resource-Optimal Checkpoint Interval, Parallel Computing Technologies,Lecture Notes in Computer Science, 2011, Volume 6873/2011, 384-389, DOI: 10.1007/978-3-642-23178-0_33
7. Josh Wills: Seismic Data Science: Reflection Seismology and Hadoop // http://www.cloudera.com/blog/2012/01/seismic-data-science-hadoop-use-case/, January 25, 2012