УДК 004.4, 004.9, 004.42, 004.67, 004.915, 004.427

А.Н. ЛАВРЕНЮК, С.И. ЛАВРЕНЮК, Е.В. НАЗАРЕНКО

ПОДХОД К ОПТИМИЗАЦИИ ПРОГРАММЫ ОБРАБОТКИ БОЛЬШИХ ОБЪЕМОВ СЕЙСМИЧЕСКИХ ДАННЫХ НА ПРИМЕРЕ 3D МИГРАЦИИ ДУПЛЕКСНЫХ ВОЛН

Оптимизация вычислений

Рассмотрен подход к оптимизации обработки больших объемов данных, на процессорах, состоящих из большого количества ядер. На примере обработки сейсмических данных программой 3D миграции дуплексных волн показано устранение узких мест, которые сдерживают производительность через управление модулями программы режимами использования вычислительных ресурсов и оперативной памяти. Этот подход дает возможность улучшить качество обработки данных и уменьшить время работы программы.

Введение. Для качественной и быстрой обработки сейсмических данных в режиме реального времени требуются производительные вычислительные ресурсы, которые имеют большие объемы оперативной памяти. При этом данные ресурсы должны быть настроены на выполнение интенсивных файловых операций ввода/вывода. Несмотря на все большую доступность больших вычислительных кластеров, например, как суперкомпьютер Института кибернетики имени В.М. Глушкова НАН Украины (СКИТ) [1] или интеллектуальных рабочих станций, например, как Инпарком [2], для обработки сейсмических данных, возникают проблемы связанные с ограничениями по объему оперативной памяти для одного параллельного процесса и по пропускной способности дисковой системы хранения данных. На первый взгляд, при запуске задачи на кластере, можно уменьшить число процессов на один вычислительный узел, тем самым увеличится объем доступной оперативной памяти для одного параллельного процесса и снизится нагрузка на дисковую систему. Но программа миграции дуплексных волн [3, 4] состоит из трех основных модулей:

- модуль расчета кубов времен от источников и от приемников, при этом полученные результаты сохраняются в виде файлов;

• модуль выполнения миграции на основе сохраненных файлов времен и сохранение промежуточного результата;

• модуль объединения промежуточного результата всех потоков программы в один результирующий файл.

При работе первого модуля происходит считывание данных из входных файлов, выполняется расчет данных и проводится интенсивная запись в файлы кубов времен и данных промежуточных расчетов. При увеличении числа ядер в одном процессоре и больших объемах входных данных происходит задержка доступа к файловой системе для процессов, в случае если на вычислительном узле кластера запускается k-процессов программы, когда k – число ядер на узле.

Сложность в том, что для работы первого модуля программы необходимо оперативной памяти намного больше, чем для работы двух других модулей. Компенсируется это записью промежуточных результатов расчетов во временные файлы. Чем меньше доступно процессу оперативной памяти, тем чаще происходит обращение к временным файлам. При этом приходится уменьшать качество работы программы иначе она не сможет дать результат за необходимое реальное время. Если же иметь достаточный объем памяти для расчета куба времени без промежуточной записи данных во временные файлы, то программа обработки сейсмических данных будет работать быстрее и самое главное с лучшим качеством. А другие модули не столь требовательны к объему оперативной памяти и менее интенсивно производят запись файлов. Соответственно если при запуске задачи указывать, что на процессоре с k ядрами запустить m процессов, где $m < k$, то получим больше оперативной памяти для одного процесса, но потеряем в производительности модуля расчета времен и, особенно, в модуле миграции, так как $k - m$ ядер будут не задействованы для вичислений.

Этапы устранения мест, которые сдерживают производительность. При возрастании числа ядер на один процессор возникает задержка при работе с операциями записи в файл, когда при запуске задачи на вычислительном узле запускается k-процессов, где k – число ядер на узле.

На первом этапе оптимизации, в программу внедрены алгоритмы сжатия данных файлов времен без потери качества результата работы программы. Есть два варианта работы программы: упаковка файлов времен с помощью наилучшей Чебышевской аппроксимации сжатия числовых данных [5] (режим 3) и упаковка полученного результата в режиме 3 с помощью алгоритма архивирования LZW (zip) (режим 4) [6]. Например, при обработке одного входного файла на конкретной реальной задаче файлы времен без сжатия занимают 49 Гб. При сжатии только в режиме 3 объем файлов времен уменьшился до 25 Гб. А при работе в режиме 4 объем файлов стал 12 Гб. Таких входных файлов на реальных задачах бывает от 10 до 30 и больше.

При этом получено ускорение в работе программы для режимов 3 – 1.64 и 4 – 1.78 раза.

Но этот этап оптимизации все же не позволяет уменьшать объем временных файлов, так как качество результата миграции очень чувствительно к качеству промежуточных результатов вычислений, которые при недостаточном количестве оперативной памяти приходится вынужденно сохранять во временных файлах, но без сжатия.

Также все дальше происходит развитие процессоров и увеличение количества вычислительных ядер, которые содержит один процессор. Если на вычислительном узле стоят несколько процессоров с большим количеством ядер первый модуль программы при превышении определенных объемов входных сейсмических данных не может работать на максимальной производительности из-за ожидания записи данных в файлы. Учитывая данное проведен второй этап оптимизации программы.

На втором этапе оптимизации предложено в программе выделить виртуальные рабочие группы Wg_n, состоящие из tr ядер, где tr – кратно общему числу ядер в процессоре и подбирается экспериментально, n – число виртуальных рабочих групп, при этом $n = k/tr$. В каждой такой группе выделяется главное ядро. Запуск программы выполняем точно также: k-процессов, где k – число ядер на узле. Но модуль расчета времен, требующий большого объема оперативной памяти и работающий с файловыми операциями, активно работает только на главном ядре.

На рис. 1 и 2 показано разделение многоядерного *CPU* Wg_n и объединение ядер в виртуальные рабочие группы.

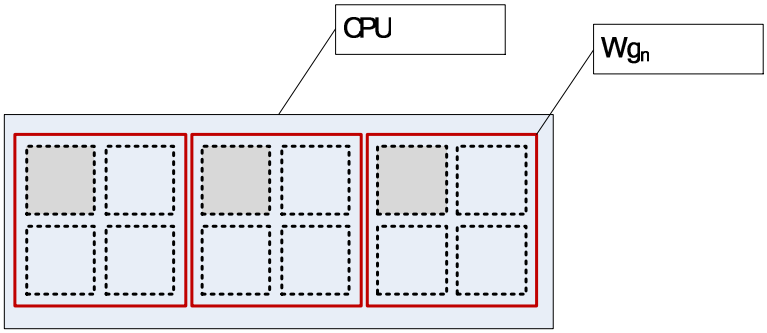

РИС. 1. Виртуальные рабочие группы процессора

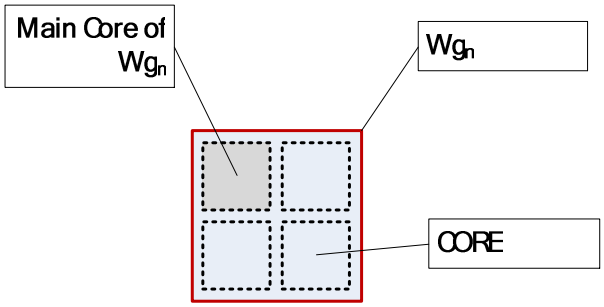

РИС. 2. Объединение ядер процессора в виртуальную рабочую группу

Для того чтобы другие ядра, кроме главного из Wg_n были задействованы в вычислительном процессе, но использовали меньше памяти и не работали с дисковой системой выделили вычислительные операции, которые выполняли с помощью нитей, порождаемых процессом на главном ядре. Для создания и управления такими нитями используется мощный механизм OpenMP [7]. Механизм распараллеливания OpenMP универсальный для разных платформ. Но при кажущейся простоте он требует внимательности при более глубоком распараллеливании уже имеющейся программы. Механизм OpenMP удалось использовать для распараллеливания многих циклов благодаря тому, что в 3D миграции дуплексных волн, есть циклические участки, в которых результат вычислений в следующей итерации не зависит от результата вычислений на текущей или предыдущей итерации.

Так как были выделены процессы, запущенные не на главных ядрах Wg_n не требующие большого объема оперативной памяти, то ее использует процесс главного ядра.

При этом процесс на главном ядре использует под свои нужды объем памяти:

$$RAM_1 = RAM_{all} / n, \quad (1)$$

где RAM_{all} вся оперативная вычислительного память узла, n – количество виртуальных групп.

После того, как отработал первый модуль, оперативная память перераспределяется, так как при работе второго и третьего модулей идет только чтение данных из файлов и не частое записывание результата. При этом каждый процесс работает на своем ядре без порождения нитей.

$$RAM_2 = RAM_{all} / k. \quad (2)$$

Соответственно $RAM_2 <= RAM_1$. Так мы получаем дополнительную оперативную память.

Вычислительные кластеры строятся давно. Появляются новые, более мощные кластера, и также совершенствуются уже существующие. На практике доступны для работы кластеры с разнородной структурой вычислительных узлов. При запуске задача обработки сейсмических данных может распределиться на узлы с разным количеством процессоров и разным количеством

ядер в одном процессоре. Предложенный подход по объединению ядер в виртуальные рабочие группы самой программой дает гибкий механизм для оптимизации использования ресурсов в зависимости от потребностей вычислительных модулей одной программы. Заметим, что *tr* может быть разным для разных процессов одной задачи.

Далее на рис. 3 показан результат полученного ускорения в зависимости от количества ядер на процессор и виртуальных групп, объединяющиеся в ядра.

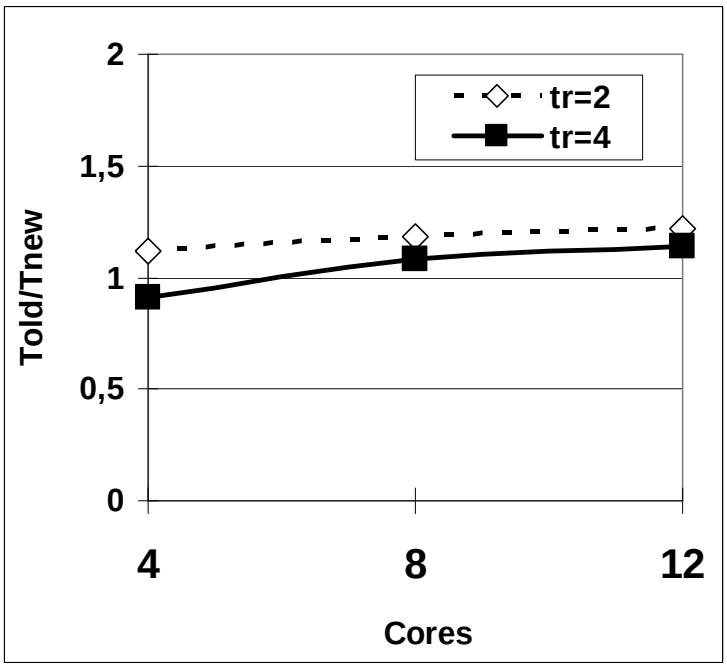

РИС. 3. Зависимость времени вычислений от величины виртуальной группы ядер

Следует заметить, что увеличение числа ядер в рабочей группе не дает большого прироста по скорости обработки данных, а иногда может вызвать и замедление. Но при увеличении числа ядер в одной группе можно задействовать больший объем оперативной памяти для главного ядра группы, а это позволит увеличить качество обработки данных, что зачастую необходимо, но сложно реализовать из-за ограниченности ресурсов. Затем, перед началом работы модуля миграции программа выполняет перераспределение ядер по виртуальным группам с $tr = 1$, так как для данного модуля нет необходимости в очень большом объеме оперативной памяти.

На рис. 4 показана схема работы программы после второго этапа оптимизации.

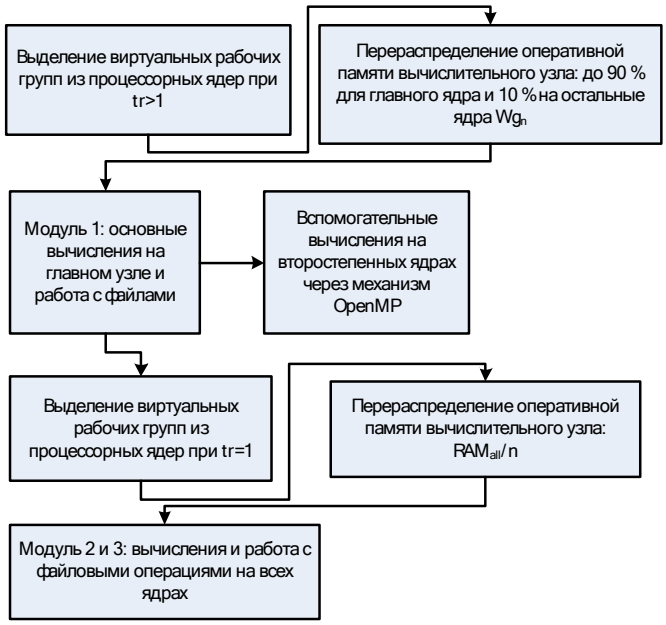

РИС. 4. Структурная схема работы программы после оптимизации

Выводы. 1. В работе показан подход по управлению вычислительными ресурсами и оперативной памятью отдельными модулями одной программы для оптимизации работы программы. Такой подход дает улучшение качества обработки данных и увеличения скорости работы программы.

2. Предложенный подход позволил увеличить качество обработки сейсмических данных, при ограниченных ресурсах, при этом не только не потеряв в скорости работы программы, а во многих случаях и выигрывая. При этом второй этап оптимизации устраняет те узкие места в работе программы, которые невозможно устранить только сжатием файлов времен.

3. В дальнейших исследованиях планируется выявление и устранение узких мест в работе третьего модуля, когда происходит в основном интенсивное чтение данных из файлов и не очень интенсивная запись результата. Так же планируется, для интенсивных вычислительных операций использовать графические ускорители (GPU) и язык программирования графических процессоров OpenCl [8], позволяющий более универсально работать на устройствах разных производителей [9]. Предварительная оценка времени работы программы [10] показывает положительный результат в выигрыше времени работы программы.

А.М. Лавренюк, С.І. Лавренюк, Є.В. Назаренко

ПІДХІД ДО ОПТИМІЗАЦІЇ ПРОГРАМИ ОБРОБКИ ВЕЛИКИХ ОБСЯГІВ СЕЙСМІЧНИХ ДАНИХ НА ПРИКЛАДІ 3D МІГРАЦІЇ ДУПЛЕКСНИХ ХВИЛЬ

Розглянуто підхід до оптимізації обробки великих обсягів даних, на процесорах, що складаються з великої кількості ядер. На прикладі обробки сейсмічних даних програмою 3D міграції дуплексних хвиль показано усунення вузьких місць, які стримують продуктивність

обчислень через керування модулями програми режимами використання обчислювальних ресурсів і оперативної пам'яті. Цей підхід дає можливість для поліпшення якості обробки даних і зменшення часу роботи програми.

A.M. Lavreniuk, S.I. Lavreniuk, E.V. Nazarenko

AN APPROACH TO A PROGRAM FOR OPTIMIZATION OF THE LARGE VOLUMES OF SEISMIC DATA PROCESSING ON THE EXAMPLE OF DUPLEX WAVE 3D MIGRATION

An approach to optimizing the processing of large volumes of data on the processors with a large number of cores is considered. On the example of seismic data processing using 3D migration of duplex wave program, the removing bottlenecks is shown that hinder the productivity through the program module management of modes of computing resources and RAM. This approach makes it possible to improve the quality of data and reduce the operation time of the program.

1. *Суперкомпьютер* ИК НАН Украины // [Электронный ресурс] – http://icybcluster.org.ua/
2. *Интеллектуальная* рабочая станция // [Электронный ресурс] – http://inparcom.kiev.ua/
3. *Мармалевський Н.Я., Роганов Ю.В., Горняк З.В.* та ін. Міграція дуплексних хвиль – новий засіб формування сейсмічних зображень субвертикальних границь // Зб. наук. праць. УкрДГРІ, 2007. – № 2. – С. 173 – 190.
4. *Marmalyevskyy N., Gornyak Z., Kostyukevych A., Mershchiy V., Roganov. Y.* Method, system and apparatus for interpreting seismic data using duplex waves: Patent US / – 7. – 110. – 323 B2. – 2006.
5. *Каленчук-Порханова А.А., Вакал Л.П.* Наилучшая Чебышевская аппроксимация для сжатия численной информации // Компьютерная математика. – 2009. – № 1. – С. 99–107.
6. *Назаренко Е.В., Тульчинский В.Г., Тульчинский П.Г.* Оптимизация обработки больших массивов данных в кластерных системах // Проблеми програмування. – 2010. – № 2–3. – С. 149 – 154.
7. *OpenMP* // [Электронный ресурс] – http://openmp.org/.
8. *OpenCL* – The open standard for parallel programming of heterogeneous systems // [Электронный ресурс] – http://www.khronos.org/opencl/
9. *Лавренюк А.М., Лавренюк С.І.* Один підхід до вирішення проблеми універсального використання мови програмування OpenCl на різних GPU // Проблеми програмування. – 2012. – № 2–3. – С. 77 – 84.
10. *Тульчинский В.Г., Чарута А.К.* Оценка времени обработки данных в кластерных системах // Там само. – 2006. – № 2–3. – С. 118 – 123.

Получено 21.10.2013

Об авторах:

Лавренюк Алла Николаевна,
кандидат технических наук,
доцент Физико-технического института НТУ Украины «КПИ»,

Лавренюк Сергей Иванович,
кандидат физико-математических наук, научный сотрудник
Института кибернетики имени В.М. Глушкова НАН Украины,

Назаренко Евгений Владимирович,
младший научный сотрудник
Института кибернетики имени В.М. Глушкова НАН Украины.