УДК 681.3

В.Г. ТУЛЬЧИНСКИЙ, П.Г. ТУЛЬЧИНСКИЙ

ПРИМЕНЕНИЕ СУПЕРКОМПЬЮТЕРОВ СКИТ ДЛЯ РАЗРАБОТКИ И ВЫПОЛНЕНИЯ ПАРАЛЛЕЛЬНЫХ ГЕОФИЗИЧЕСКИХ ПРОГРАММ

Ключевые слова: *параллельное программирование, обработка сейсмических данных, оптимизация кластерных вычислений, большие наборы данных.*

ВВЕДЕНИЕ

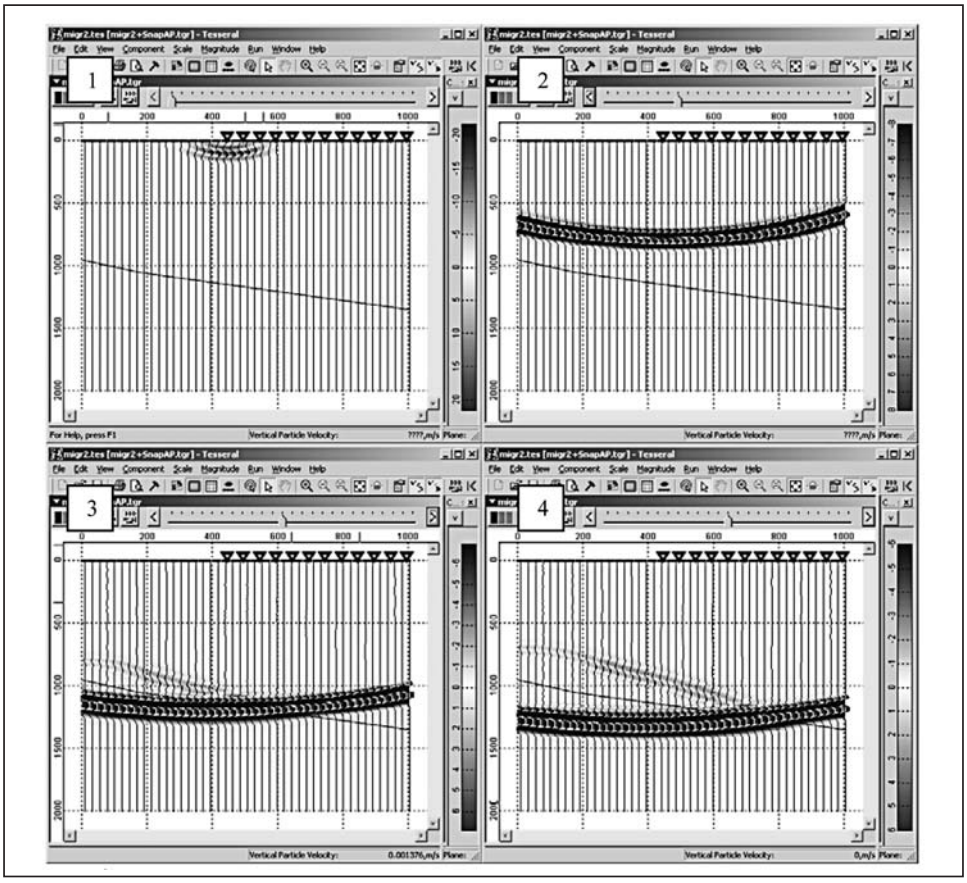
Проблематика параллельного программирования достаточно хорошо разработана еще в 1970–1980 гг. Настоящий взрыв параллельных вычислений начался в последние годы в связи с непрерывным накоплением данных и увеличением размеров задач в условиях технологических ограничений дальнейшего роста тактовой частоты процессоров. На аппаратном уровне эта тенденция проявилась в переходе на многоядерные компьютеры, распространении многомашинных кластеров, появлении параллельных вычислителей на основе видеокарт. Косвенно она привела к скачку производительности локальных сетей и дисковых хранилищ. На общесистемном уровне концепции кластерных вычислителей (high performance computing, HPC), распределенных в Интернете вычислителей переменного состава (grid), и оперативной аренды удаленных вычислительных ресурсов (cloud computing) косвенно способствуют программированию в открытых кодах и продвижению Linux, несмотря на высокое качество ПО для Windows-кластеров.

Украина не осталась в стороне от этих тенденций благодаря сочетанию инициативы ученых и появившихся в период экономического роста финансовых возможностей государства. Украинский кластер СКИТ на протяжении последних лет постоянно занимал передовые позиции в рейтинге производительности [1], интенсивно использовался удаленными пользователями и несколькими grid-системами. Большое число кластеров меньшего размера закуплено у коммерческих поставщиков или построено силами украинских предприятий для собственных потребностей. В связи с экономической рецессией в ближайшее время маловероятны значительные инвестиции в вычислительную технику. Поэтому повышается актуальность увеличения производительности и эффективного использования существующих вычислителей.

Одной из областей применения суперкомпьютеров является обработка данных сейсморазведки.

1. ОБРАБОТКА ДАННЫХ СЕЙСМОРАЗВЕДКИ

Изложим типичную схему сейсмического исследования земной коры [2]. На поверхности вдоль планового сейсмического профиля размещают приемники (геофоны), способные регистрировать свои перемещения в пространстве. Вдоль того же профиля помещают источник сейсмического сигнала: закопанный взрывпакет или группу вибраторов — машин, синхронно ударяющих в землю тяжелыми молотами. В результате воздействия источника в земной коре возбуждается акустическая волна, которая со скоростью звука в среде распространяется вниз и в стороны от источника. На сейсмической границе, т.е. границе перепада скорости звука в породе, происходит преломление и отражение волн. Отраженная волна распространяется в обратном направлении, достигает поверхности и регистрируется геофонами (рис. 1). В каждом геофоне записывается сейсмическая трасса, представляющая собой зависимость зарегистрированной скорости вертикального смещения прибора от времени. Совокупность сейсмических трасс образует сейсмограмму. После регистрации сейсмоисточник и приемники перемещают вдоль профиля и повторяют операцию. Для получения модели района сейсмические исследования проводят вдоль сети профилей. Широко используется 3D-сейсморазведка, в которой волну регистрируют приемники, расположенные



Šuń. 1. Cõåĝą šąńļšīńņšąķåķč˙ āīėķū č åå īņšąęåķč˙ īņ ńåéńĝč÷åńźīé ćšąķčöū

вдоль нескольких параллельных линий приема, как правило, в крест с профилями возбуждения. Сетку наблюдений стараются сохранять регулярной. Распространяется многокомпонентная сейсморазведка 3С, при которой геофон отдельно регистрирует скорости смещения вдоль трех пространственных осей, и сейсморазведки 9С, при которой трехкомпонентный прием сочетается с поочередным возбуждением трех направленных волн в каждой точке.

Сейсмограммы несут важную информацию о структурном строении земной коры. В частности, нефть и газ находятся в пористых породах-коллекторах (песчаниках, трещиноватых известняках), ограниченных плотными покрышками. Значительная разница плотности и соответственно скорости звука между коллектором и плотной покрышкой создает резкие границы, которые отображаются на сейсмограммах высокими амплитудами сигнала (яркими полосками на рис. 2).

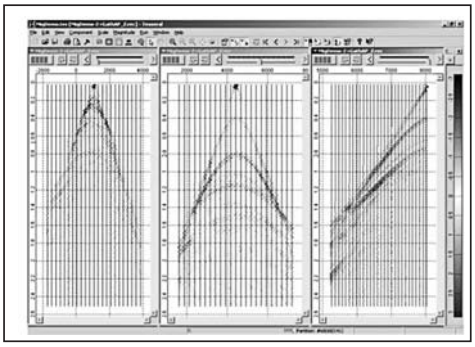
Если источник и приемник расположены на плоской поверхности земли, волна распространяется в однородной среде со скоростью звука V и отражается от горизонтальной границы, расположенной на глубине z, а расстояние между источником и приемником — x, отраженный сигнал достигнет приемника через время

$$t = \frac{2}{V}\sqrt{\left(\frac{x}{2}\right)^2 + z^2}. \qquad (1)$$

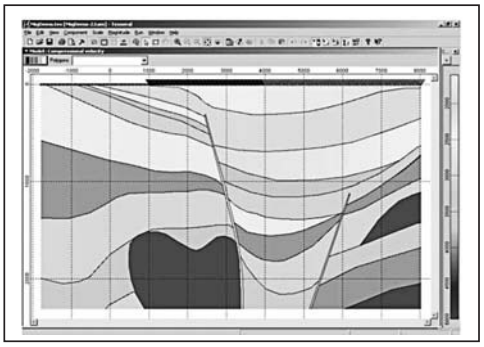
Формула (1) получена согласно закону оптики о равенстве углов падения и отражения. Ее преобразование позволяет оценить глубину z горизонтальной границы по времени регистрации:

$$z = \frac{\sqrt{V^2 t^2 - x^2}}{2}. \qquad (2)$$

Согласно формуле (2) сигнал, зарегистрированный геофонами, расположенными вдоль профиля на горизонтальной поверхности, будет в такой среде иметь форму верхнего купола эллипса (в случае 3D-наблюдений — эллипсоида вращения). От двух или большего числа горизонтальных границ получим концентрические эллипсы. Усложнение модели (наклон границ, анизотропия, ограниченные по размеру вкрапления) приводит к искажению эллиптической формы сигнала (рис. 2, 3).



Šćń. 2. Ņšč ńåéńġīćšąġġū tāķīćī įšīōčė˙ (2D)



Šćń. 3. Ńźīšīńņķą˙ ġīääåėü įšīōčė˙ äė˙ šćń. 2

Важное следствие формулы (2) — возможность одновременного подбора V и z при наличии замеров времени прихода сигнала t для разных значений расстояния x в случае горизонтальной границы и однородной среды над границей. А именно, достаточно перебрать возможные значения V и для каждого x вычислить z. Если скорость подобрана верно, все глубины совпадут. Если она завышена, z будет увеличиваться с ростом x, если занижена — то уменьшаться с ростом x. На этом принципе построены алгоритмы восстановления скоростной модели среды по сейсмическим данным:

– исходим из какой-то модели, варьируем ее параметры и получаем ряд моделей;

– рассчитываем для каждого варианта модели каждой трассы положение границ;

– выбираем тот вариант, в котором наблюдается наилучшее совпадение положения границ по разным трассам с границами, заданными в модели.

Обычно модель строят сверху вниз, поочередно подбирая границы и скорости над ними. После первоначального определения скоростной модели в некоторых точках поверхности на основе описанного грубого горизонтально-слоистого приближения границы и скорости между ними интерполируют, формируя таким образом глубинную скоростную модель. Затем для каждой трассы вычисляют времена распространения волны от источника и от приемника до каждой точки среды. Сумма этих времен задает позицию на трассе, откуда переносят (добавляют) значение в соответствующую позицию выходного изображения. Подобные процедуры называются пре-стек миграциями. Яркие, контрастные изображения границ мигрированного разреза или куба, совпадающие с границами исходной модели, означают, что модель построена хорошо. Для достижения такого результата границы модели приходится итеративно корректировать в сторону изображений границ на мигрированном разрезе или кубе. В процессе уточнения модели миграции выполняются неоднократно. Алгоритмы миграций различаются главным образом по способу расчета времен.

Задачи для распараллеливания. В качестве примера распараллеливания рассмотрим три характерные задачи, значительно различающиеся по соотношению потребностей в процессорном ресурсе, оперативной памяти и нагрузке на систему хранения файлов.

Первая задача — 3D пре-стек миграция дважды отраженных (дуплексных) волн (МДВ) [2]. По свойствам алгоритма это типичная миграционная процедура. Но в отличие от обычной миграции МДВ регистрирует не горизонтальные, а вертикальные границы (разломы, зоны трещиноватости, границы соляных штоков и т.п.).

На вход МДВ помимо скоростной модели среды и сейсмограммы подается карта отражающего горизонта. Различают два типа дуплексных волн в зависимости от порядка отражений: сначала от заданной горизонтальной границы, затем — от вертикальной (МДВ-П), и в обратном порядке (МДВ-И). Алгоритм МДВ для одной трассы состоит из трех шагов:

1) расчет алгоритмом эйконала времени распространения упругой волны от источника (для МДВ-И) или до приемника (МДВ-П) для каждой точки среды;

2) расчет алгоритмом эйконала времени распространения упругой волны, отраженной от заданного отражающего горизонта, от приемника (МДВ-И) или до источника (МДВ-П);

3) вычисление для каждой точки среды суммарного времени и прибавление в соответствующую точку мигрированного куба значения трассы, записанного в это суммарное время (при условии выполнения апертурных ограничений, ограничений по значениям времен и их производных).

Вторая задача — синтез 3D-сейсмограмм по геологической модели среды методом конечно-разностного эластического анизотропного моделирования 2,5D (M2,5D) [3]. В отличие от альтернативных методов получения синтетических 3D-сейсмограмм — лучевого моделирования и конечно-разностного акустического моделирования — M2,5D позволяет учитывать тонкослоистость, пространственную анизотропию свойств (TTI) и сочетание нескольких пространственных систем трещиноватости. В то же время M2,5D накладывает жесткое ограничение на геометрию геологической модели среды, а именно все свойства должны быть постоянны вдоль оси Y. Неудобство указанного ограничения компенсируется возможностью синтезировать реалистические сейсмограммы, отражающие натуральную сложность геологического строения, но точно соответствующие известной модели среды. Даже суперкомпьютеры не могут выполнить эластическое анизотропное моделирование 3D в тонкослоистой среде за приемлемое время. Таким образом, сейсмограммы M2,5D обладают уникальными свойствами для задачи планирования сейсморазведки (принятия решения о размахе и плотности сетки наблюдений, о выполнении многокомпонентных исследований), тестирования последовательностей обработки, оценки предельной точности геофизической интерпретации. Алгоритм M2,5D состоит из трех шагов:

1) конечно-разностное моделирование распространения упругих волн для отдельных частот пространственного Фурье-разложения среды 2,5D вдоль оси постоянных свойств Y;

2) обратное преобразование Фурье набора 2D-сейсмограмм, соответствующих отдельным частотам, для получения 3D-сейсмограмм в обычном пространстве;

3) репликация (размножение) 3D-сейсмограмм от одной линии источников для получения полноценной 3D-сети наблюдений.

Третья задача — фильтрация поверхностных волн рэлеевского типа в трехкомпонентных сейсмических наблюдениях (Ф3С) — представляет собой типичную процедуру локальной обработки данных. На вход подаются сейсмограммы с записью двух горизонтальных (продольной и поперечной) и одной вертикальной компонентов. Подавление волны Рэлея обеспечивается выполнением в скользящем окне на каждой многокомпонентной записи (наборе из трех соответствующих трасс сейсмограмм компонентов) следующей процедуры [4].

1. Сингулярное разложение матрицы размера $N \times 3$, три вектор-столбца которой содержат соответствующие по времени фрагменты трасс трех компонентов записи. По трем собственным значениям $\sigma_1 \geq \sigma_2 \geq \sigma_3$ вычисляется параметр $e(t) = \sqrt{(\sigma_1^2 - \sigma_3^2)(\sigma_2^2 - \sigma_3^2)}$. Значение $e(t)$, превышающее порог, является признаком искажения рэлеевскими волнами.

2. Обработка искаженных участков низкочастотным фильтром.

Каждая из перечисленных задач обладает свойствами, характерными для широкого класса алгоритмов, интенсивно используемых в сейсморазведке, и естественным параллелизмом.

Обоснование распараллеливания и выбор платформы. Решение о разработке параллельной программы или о распараллеливании последовательной программы часто принимается интуитивно. Между тем этот вопрос достаточно сложен и требует глубокого анализа.

Во-первых, инженеры склонны переоценивать коммерческую привлекательность быстрых вычислений. Во многих случаях приобретение, установка, содержание и обслуживание специальной вычислительной техники для ускорения лишь нескольких программ оказываются нецелесообразными с точки зрения лиц, принимающих решения.

Во-вторых, интуитивная оценка потенциального выигрыша в производительности часто бывает завышена. Закон Амдала (1967) выражает этот феномен математически. При идеальном распараллеливании доли вычисления η на N процессорных элементов (ПЭ) ускорение составит

$$S_N = \frac{1}{(1-\eta)+\eta/N} \le \frac{1}{1-\eta}. \tag{3}$$

Значение этого закона проясняет подстановка чисел: ускорение вычисления в результате идеального распараллеливания 95% расчетов на 20 ПЭ составляет всего 1/(0,05+0,95/20)=10,26. На практике эффективность параллельных вычислений оказывается еще ниже. Простой эксперимент (запуск четырех независимых экземпляров последовательной программы M2,5D на современном ПК, оснащенном двумя двухядерными процессорами) показал ускорение всего в полтора раза по сравнению с их последовательным выполнением. Для двух экземпляров программы выигрыш времени составил 25%. Наконец, распараллеливание обычно связано с накладными расходами на коммуникацию и синхронизацию, дополнительно замедляющими параллельное решение одной задачи по сравнению с решением нескольких независимых задач на том же вычислителе.

Однако в законе Амдала имеется «лазейка», обнаруженная в 1988 г. Густафсоном и Барсисом. Часто время непараллельных вычислений программы мало зависит от объема ее параллельной части. В частности, это происходит, если непараллельную часть программы составляют только подготовка вычислений и обработка результатов. В этом случае в качестве параметра целесообразно взять долю этой непараллельной части не в исходном, а в распараллеленном расчете:

$$\sigma_N = \frac{1-\eta}{\eta/N+1-\eta}. \tag{4}$$

Преобразовав формулу (3) с учетом (4), Густафсон и Барсис получили закон, названный их именами:

$$S_N \le N + (1-N)\sigma_N \Rightarrow \sigma_N \le \frac{N-S_N}{N-1}. \tag{5}$$

По формуле (5) можно оценить предельную долю последовательной части для достижения заданного ускорения. Например, для 19-кратного ускорения на 21 ПЭ доля последовательных расчетов на каждом ПЭ не должна превышать $(21-19)/(21-1)$, что составляет 10%. Главный вывод из закона Густафсона–Барсиса: распараллеливание эффективно при большом времени параллельных вычислений, т.е. для больших задач. Распараллеливание на большое число процессоров эффективно только для очень больших задач.

Для задач обработки данных сейсморазведки решающим фактором являются чрезвычайно большие объемы и высокая коммерческая стоимость решения. Типичный объем однокомпонентных сейсмограмм 3D составляет 50–500 ГБ на месторождение. Затраты на сейсморазведку несопоставимы со стоимостью бурения даже одной скважины: 1–5 млн долл. — на суше, 10–50 млн долл. — на шельфе. Объемы сейсмограмм постоянно растут за счет расширения зоны размещения приемников, увеличения плотности сетки наблюдений и регистрации 3C/9C. Перспективными для нефтегазодобычи являются купольные структуры, вертикальные зоны трещиноватости, приштоковые зоны, которые идентифицируются по скоростной модели. Для повышения точности построения скоростной модели необходимо увеличивать число прогонов обрабатывающих процедур за ограниченное время подготовки к тендеру или выполнения сервисных работ. Этим объясняется готовность сервисных компаний на значительные инвестиции в вычислительную технику и программное обеспечение. Таким образом, для рассматриваемых задач ни коммерческая

привлекательность распараллеливания, ни потенциальный выигрыш в производительности не вызывают сомнений. Кластерные вычисления уже повсеместно применяются в сейсморазведке.

Помимо целесообразности распараллеливания важным вопросом является выбор параллельной платформы для реализации. В большинстве случаев выбор делается между:

– многоядерными ПК,

– ПК с вычислителями на видеокартах (графические процессорные устройства (ГПУ)),

– вычислениями в локальных сетях ПК,

– вычислениями в Интернет (грид- и клауд-архитектуры).

Для МДВ необходимы два временных куба при миграции каждой трассы. На один источник часто приходится 1000–4000 координат приемников при общем числе разных координат приемников до 10000. Число точек возбуждения также измеряется тысячами. Неоднократный расчет временных кубов существенно снизил бы производительность. В связи с этим для эффективной реализации МДВ необходимо обеспечить доступ всех процессов к общему (возможно, распределенному) хранилищу временных файлов. Типичный объем файлов временных кубов измеряется терабайтами. Таким образом, единственный практически приемлемый выбор вычислителя для задачи МДВ — кластер с высокоскоростным подключением к высокопроизводительному дисковому хранилищу (возможно, распределенному).

В ФЗС основные затраты времени приходятся на вычисление сингулярных разложений. Более простые однотрассные процедуры, например частотная фильтрация, могут выполняться и на ПК, несмотря на объемы данных, измеряемые сотнями ГБ. Применение Интернет-вычислений для распараллеливания ФЗС нецелесообразно ввиду недостаточной для таких объемов полосы пропускания. Остальные платформы пригодны для реализации и использования.

M2,5D — наиболее вычислительно сложная задача из отобранных. Она же работает с наименьшими объемами данных — до нескольких сотен мегабайт на самом длинном первом шаге алгоритма; объем одного передаваемого файла — 1–2 МБ. Возможно применение Интернет-вычислений для реализации этого шага. Однако второй и особенно третий шаги M2,5D сводятся к интенсивной обработке данных, что препятствует их выполнению в грид-системе. Применение ГПУ также возможно, но отсутствие необходимых для этой задачи комплексных чисел в CUDA 1.0 и недостаточный объем памяти на старых картах ограничивают рынок такого решения.

В целом кросс-платформная реализация в MPI обеспечивает оптимальное на сегодняшний день соотношение стоимость/эффективность для всех перечисленных задач.

2. ТЕХНОЛОГИЯ ПРОЕКТИРОВАНИЯ ПАРАЛЛЕЛЬНЫХ ПРОГРАММ

В настоящее время наиболее полную и детализированную технологию разработки параллельных программ предлагает Язык шаблонов параллельного программирования (ЯШПП) Мэттсона, Сэндерса и Мэссингилл (1999). Концепцию «шаблонов проектирования» в 1977 г. предложил К. Александер для строительства и архитектуры. Шаблон проектирования — это описание хорошего решения повторяющейся проблемы в определенном контексте. Шаблоны позволяют использовать опыт экспертов более конструктивным и неформальным образом, чем экспертные системы, в то же время обеспечивая единообразный терминологический словарь для предметной области проектирования. В программной инженерии шаблоны проектирования впервые использовали Бек и Куннингам (1987). В 1994 г. Гамма, Хелм, Джонсон и Влиссидес разработали шаблоны для объектно-ориентированного программирования. Применение шаблонов проектирования для параллельного программирования начали МакДоналд, Шефэр и Шафрон еще в 1997 г. [5], однако их подход был существенно уже, чем ЯШПП [6]. Идея ЯШПП — формулирование универсальной технологии разработки параллельных программ, способной включить все возможные подходы (известные, а также те, которые могут появиться в будущем) и все этапы проектирования. Для этого сформирован расширяемый набор взаимосвязанных документов (шаблонов), каждый из которых описывает один из подходов к решению

проблемы проектирования программы. Все шаблоны устроены единообразно и состоят из разделов «Проблема» (цели этого шаблона), «Контекст» (связи с другими шаблонами), «Симптомы» (признаки целесообразности использования шаблона), «Усилия» (цели оптимизации, ограничения и факторы, которые нужно учитывать), «Реализация» (общая схема и пошаговая процедура принятия проектного решения по шаблону) и «Примеры». Но ЯШПП — это не просто набор шаблонов, а комплексная методология разработки параллельной программы, начиная с поиска параллельности и заканчивая компиляцией. Согласно этой методологии шаблоны проектирования разделены на четыре группы (в терминологии ЯШПП четыре «пространства проектирования»), соответствующие четырем последовательным этапам разработки параллельной программы. Опишем их.

- «Поиск параллельности» — включает шаблоны декомпозиции задачи и данных, анализа зависимостей для синхронизации, группировки и упорядочения задач, разделения данных (основные факторы — сложность и эффективность распараллеливания, балансировка загрузки, минимизация простоя, синхронизации и коммуникаций, масштабирование по числу процессоров);
- «Структуры алгоритма» — включает дерево шаблонов способа распараллеливания, обеспечивающее выбор принципа организации параллельной программы для найденной параллельности (факторы — балансировка загрузки, минимизация простоя и простота реализации, однако каждый шаблон включает свои критерии соответствия найденной параллельности);
- «Вспомогательные структуры» — включает шаблоны структур программы и данных согласно выбранным способам распараллеливания (факторы — эффективное использование памяти и сетевых ресурсов, минимизация накладных расходов на синхронизацию и коммуникацию);
- «Механизмы реализации» — включает шаблоны отображения разработанного параллельного алгоритма на конкретную программно-аппаратную систему с использованием языков программирования, сред параллельного исполнения программ и т.п. либо подбор подходящего аппаратно-программного решения (основные факторы — простота реализации, минимизация накладных расходов, оптимизация использования потенциала аппаратуры).

В настоящее время из всех пространств ЯШПП наиболее детализированы «Поиск параллельности» и «Структуры алгоритма».

Проектирование с помощью ЯШПП. Опишем применение ЯШПП на примере проектирования параллельных программ для трех выбранных задач: МДВ, М2,5D и Ф3С.

Проектирование параллельных программ начинается с поиска параллелизма (рис. 4) с помощью шаблона «Декомпозиция». Все три задачи имеют простые алгоритмы (2–3 шага) и работают с большими объемами однородных данных. Параллельная обработка данных является естественным следствием такой природы задач, что приводит к приоритетному рассмотрению шаблона «Декомпозиция данных».

В задаче Ф3С данные обрабатываются независимо. Можно выбирать из нескольких уровней детализации такой обработки по окнам в рамках одной трассы, по трассам, по блокам трасс. Теоретически допустимо разделять задачи поиска и фильтрации поверхностных волн, однако отличное масштабирование задачи при реально возможном количестве вычислительных элементов делает такую декомпозицию нецелесообразной.

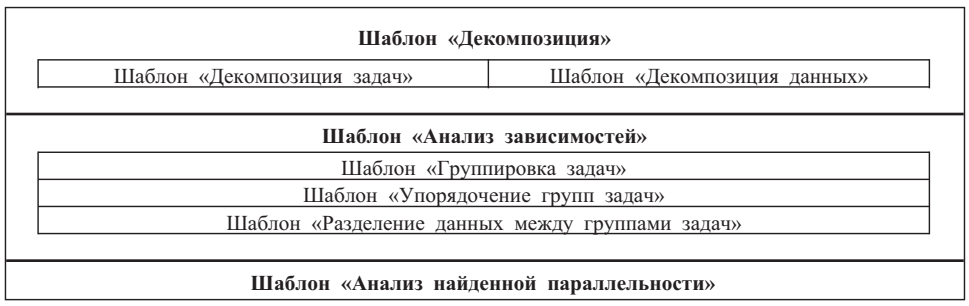


Рис. 4. Пространство проектирования «Поиск параллельности»

В задаче M2,5D моделирование распространения волны от каждого источника на каждой пространственной частоте не зависит от других. Типичное число источников в задаче — 100, число частот — от 10 до 100. В большинстве случаев декомпозиции по частотам и источникам достаточно. Однако иногда, например для горизонтально-слоистой среды, где достаточно одного источника, число ПЭ может оказаться избыточным. Тогда можно увеличить степень декомпозиции: каждый шаг конечно-разностных вычислений включает однородный независимый расчет большого массива переменных (скоростей смещения по напряжениям, напряжений — по скоростям смещения). При этом конечно-разностные расчеты необходимо группировать по источникам/частотам. Второй и третий шаги алгоритма напоминают по возможностям декомпозиции ФЗС.

В задаче МДВ миграцию (суммирование) трасс можно выполнять независимо. Более того, при миграции одной трассы расчет независимо выполняется во всех точках среды (в рамках апертур), что позволяет почти неограниченно масштабировать решение. Однако шаги расчета времен эйконалом ограничивают декомпозицию двумя уровнями, аналогичными M2,5D. Один поток может рассчитывать временное поле источника/приемника или часть временного поля на одной глубине с группировкой по источнику/приемнику.

Все рассматриваемые задачи можно разбить на значительное число подзадач. Оптимальную степень декомпозиции выбирают в зависимости от ожидаемого количества ПЭ и типичного размера задачи. Причем шаблон «Анализ зависимостей» не дает полезной информации для этих задач, поскольку обрабатываемые фрагменты данных независимы во всех рассмотренных вариантах декомпозиции. Шаги параллельной программы в целом совпадают с шагами последовательных алгоритмов. Шаблон «Анализ найденной параллельности» показывает пригодность рассмотренных вариантов декомпозиции для реализации.

Переходим в пространство структуры алгоритма (рис. 5). Выбранный принцип распараллеливания данных в сочетании с их регулярной структурой приводит к выбору шаблона «Геометрическая декомпозиция». Действительно, во всех трех задачах данные легко разделяются на большое число примерно равных по размеру блоков с естественным порядком следования по номеру трассы или источника/приемника.

Šč́ń. 5. Ļšīńņšąķńņāī ļšīåźņčšīāąķč˙ «Ńņšóźņóšū ąėćīščņģą»

Переходим в пространство вспомогательных структур (рис. 6). Рассмотрим основные шаблоны структуры программы. Она должна соответствовать ранее выбранной структуре алгоритма (см. рис. 5). На ее эффективность существенно влияет целевая платформа реализации (архитектура вычислителя и/или средства параллельного программирования).

В шаблоне «Одна программа/много данных» отсутствует непосредственное взаимодействие между потоками вычислений. Его особенность — равноправие ПЭ. Отсутствие управляющего потока увеличивает полезную мощность на один ПЭ, что существенно при их общем небольшом количестве. Но для балансировки нагрузки при этом шаблоне необходима специальная структура «Общая очередь». При отсутствии такой балансировки производительность полностью определяет «слабое звено». В целом этот шаблон оптимален для однородных вычислителей с

Шаблон «Структуры программы»

Соответствие выбранной структуре алгоритма	Шаблон «Одна программа / много данных»	Шаблон «Хозяин — работники»	Шаблон «Распараллеливание циклов»	Шаблон «Ветвление — объединение»
Параллельные задачи	****	****	****	**
Разделяй и властвуй	***	**	**	****
Геометрическая декомпозиция	****	*	***	**
Рекурсия по данным	**	*		
Конвейер	***	*		****
Координация по событиям	**	*		****

Шаблон «Структуры данных»

Шаблон «Общие данные»	Шаблон «Общая очередь»	Шаблон «Распределенный массив»

Šč ń. 6. Ļšīńņšąķńņāī ļšīåźņčšīāąķč˙ «Āńļīģīćąņåėüķūõ ńņšóźņóš»

небольшим числом ПЭ (например, многоядерных/многопроцессорных ПК) или плохо масштабируемых задач, в которых по одному ПЭ приходится на каждый поток. Иногда этот шаблон — единственный доступный выбор в силу особенностей целевой платформы (например, транспьютеры или ГПУ).

Шаблон «Хозяин — работники» обеспечивает динамическую балансировку загрузки, однако уменьшает полезную мощность на один ПЭ, выполняющий управляющий процесс. Это единственный шаблон структуры программы, устойчивый к отказам и безразличный к диспропорции в производительности ПЭ. В целом этот шаблон оптимален для вычислителей с большим числом ПЭ при условии хорошего масштабирования задачи (много подзадач на один ПЭ). В некоторых случаях этот шаблон — единственный доступный выбор в силу особенностей целевой платформы (например, грид-системы).

В шаблоне «Распараллеливание циклов», как и в шаблоне «Одна программа/много данных», отсутствует непосредственное взаимодействие между потоками вычислений. Этот шаблон идеален для распараллеливания готовых последовательных программ в однородных вычислителях с общей памятью. Здесь также имеется взаимосвязь с целевой платформой, например OpenMP.

Шаблон «Ветвление — объединение» применим, когда число потоков сложным образом меняется в процессе выполнения задачи, что затрудняет применение более простых шаблонов. Этот шаблон соответствует рекурсивным структурам алгоритма, а из средств реализации — PVM.

Шаблоны «Одна программа/много данных» и «Хозяин — работники» наиболее подходят для трех выбранных задач в силу возможной балансировки загрузки ПЭ. Окончательный выбор структуры программы неотделим от анализа целевой платформы. Для варианта вычислений в сети ПК под MPI реализация структуры данных «Общая очередь» несколько труднее, чем реализация структуры программы «Хозяин — работники», поскольку требуются специальные усилия на синхронизацию обращений к общему объекту (например, файлу) в потенциально гетерогенной среде. Однако все три задачи — вычислительно сложные и вероятность их выполнения на небольшом числе процессоров — ничтожна. Поэтому потери на выделение управляющего процесса пренебрежимо малы. Окончательный выбор сделан в пользу шаблона «Хозяин — работники» вопреки рекомендациям шаблона на рис. 6.

Структура данных взаимосвязана со структурой программы и особенностями платформы реализации. В силу выбора платформы MPI и поддержки кластеров необходимо ориентироваться на «Распределенный массив». Однако низкоуровневая декомпозиция в M2,5D и МДВ предполагает шаблон «Общие данные». На этом этапе недостаточно информации, чтобы выбрать для этих двух задач один готовый шаблон: целесообразны «Распределенный массив» (между узлами) и «Общие данные» (в рамках узла).

Пространство «Механизмы реализации» (рис. 7) содержит наименее формализованные шаблоны. Функции управления процессами или нитями (их созданием, подсчетом и идентификацией) во многих случаях выполняет платформа реализации. Она также предоставляет средства программирования для синхронизации и коммуникации потоков вычислений. Структура программы «Хозяин — работники» использует коммуникацию для посылки заданий ПЭ «работникам» и получения информации о результатах выполнения этих заданий. Низкоуровневая декомпозиция в М2,5D и МДВ усложняет коммуникацию и синхронизацию. Тогда главный процесс рассылает задания только главным процессам каждого узла, а уже эти процессы распределяют задания между остальными процессами узлов.

Шаблон «Выбор механизмов реализации»

Шаблон «Управление» (управление процессами или нитями)
Шаблон «Синхронизация» (порядок выполнения, разделяемые ресурсы)
Шаблон «Коммуникация» (обмен информацией между процессами или нитями)

Рис. 7. Пространство проектирования «Механизмы реализации»

Уточнение и выбор проекта. В идеале проходя последовательности шаблонов, разработчик получает комплексный проект параллельной программы. Кроме переходов сверху вниз, ряд шаблонов включает условия возврата на предыдущие уровни и переходы по горизонтали. Однако общая каскадная схема проектирования не гарантирует получения наилучшего проекта, а только пригодного для реализации локально оптимального варианта. Например, алгоритм распараллеливания, идеально подходящий к задаче, может оказаться неэффективным при реализации на конкретном компьютере. Но такие характеристики, как удельный объем оперативной памяти на процессор и скорость межпроцессорной коммуникации учитываются только на последнем этапе. Однако параллельная программа, спроектированная под конкретную машину, может утратить ценность в связи с поступлением новой техники. Фактически после прохождения всех шагов процесса проектирования ЯШПП мы упростили принятие решения, значительно сократив число альтернатив. Однако сложно выбрать единственный вариант на каждом шаге. Поскольку трудозатраты и время проектирования параллельной программы в рамках ЯШПП незначительны по сравнению с общими трудозатратами и временем разработки, целесообразно подготовить несколько проектов, сравнить их по комплексному критерию и только после этого выбрать проект для реализации. Для принятия решений в качестве метода экспертного анализа используем метод анализа иерархий (МАИ) Саати [7], основанный на декомпозиции цели в дерево факторов (рис 8).

Дерево факторов выбора проекта	Таблицы альтернатив
Цель: минимизировать издержки А. Потери производительности А1. Простой из-за синхронизации А2. Коммуникационные задержки А3. Доступ к долговременной памяти А3а. Чтение данных А3б. Запись данных А4. Восстановление прерванных расчетов Б. Ресурсоемкость Б1. Ограничения масштабирования Б1а. Оперативная память Б1б. Долговременная память Б2. Задержки из-за дефицита памяти В. Сложность реализации В1. Декомпозиция В2. Коммуникация и синхронизация В3. Восстановление прерванных расчетов	1. Проекты Ф3С: Ф1. Обрабатывать по окну трассы на ПЭ Ф2. Обрабатывать по трассе на ПЭ Ф3. Обрабатывать по блоку трасс на ПЭ 2. Проекты М2,5D: М1. Моделировать по одной частоте от одного источника на ПЭ М2. Моделировать по одной частоте от одного источника на узле (делить по ПЭ) М3. Моделирование на ГПУ 3. Проекты МДВ: Д1. Времена по блокам, миграция по блокам Д2. Времена по блокам, миграция по трассам Д3. Времена по частям, миграция по трассам

Šĉń. 8. Ääšâāī ōąźņīšīā č ąėüņåšķąņčāķūå ļšīåźņū šåąėččēąöčč Õ3Ń, Ģ2,5D č ĢÄĀ

Числовые/балльные/словесные оценки или матрицы парного сравнения вершин-братьев преобразуются МАИ в относительные веса терминальных вершин — элементарных факторов. Взвешивание альтернатив относительно каждого из элементарных факторов также происходит путем числовой/балльной/словесной оцен-

ки или с использованием матрицы попарного сравнения. Свертка вектора весов альтернатив относительно каждого из элементарных факторов с общим вектором весов этих факторов дает интегральную экспертную оценку этой альтернативы. В качестве комплексного критерия выбора принята минимальная интегральная экспертная оценка, т.е. минимум совокупных издержек разработки и выполнения параллельных программ (табл. 1–3).

Таблица 1. Экспертная оценка проектов реализации Ф3С

Взвешивание факторов				**Оценка альтернатив**			**Взвешивание альтернатив**		
Фактор	Балл	Вес в кусте	Вес	Ф1	Ф2	Ф3	Ф1	**Ф2**	Ф3
А	5	0,5							
Б	1	0,1							
В	4	0,4							
А1*	5	0,3125	0,15625	1	3	5	0,017361	0,052083	0,086806
А2*	4	0,25	0,125	5	4	1	0,0625	0,05	0,0125
А3	5	0,3125	0,15625						
А4*	2	0,125	0,0625	4	1	5	0,025	0,00625	0,03125
Б1	1	0,5	0,05						
Б2*	1	0,5	0,05	0	0	1	0	0	0,05
В1*	3	0,3	0,12	4	2	5	0,043636	0,021818	0,054545
В2*	5	0,5	0,2	5	3	5	0,076923	0,046154	0,076923
В3*	2	0,2	0,08	5	5	4	0,028571	0,028571	0,022857
А3а*	2	0,4	0,0625	5	4	1	0,03125	0,025	0,00625
А3б*	3	0,6	0,09375	5	4	1	0,046875	0,0375	0,009375
Б1а*	2	0,4	0,02	0	0	1	0	0	0,02
Б1б*	3	0,6	0,03	2	2	1	0,012	0,012	0,006
						Итого:	0,344117	**0,279377**	0,376506

Таблица 2. Экспертная оценка проектов реализации M2,5D

Взвешивание факторов				**Оценка альтернатив**			**Взвешивание альтернатив**		
Фактор	Балл	Вес в кусте	Вес	М1	М2	М3	М1	**М2**	М3
А	5	0,41666667							
Б	5	0,41666667							
В	2	0,16666667							
А1*	4	0,28571429	0,119048	4	5	3	0,039683	0,049603	0,029762
А2*	4	0,28571429	0,119048	3	4	5	0,029762	0,039683	0,049603
А3	1	0,07142857	0,029762						
А4*	5	0,35714286	0,14881	1	3	5	0,016534	0,049603	0,082672
Б1	5	0,625	0,260417						
Б2*	3	0,375	0,15625	3	1	5	0,052083	0,017361	0,086806
В1*	4	0,4	0,066667	1	2	5	0,008333	0,016667	0,041667
В2*	3	0,3	0,05	3	4	1	0,01875	0,025	0,00625
В3*	3	0,3	0,05	1	2	4	0,007143	0,014286	0,028571
А3а*	1	0,25	0,00744	3	4	5	0,00186	0,00248	0,0031
А3б*	3	0,75	0,022321	3	4	5	0,00558	0,00744	0,009301
Б1а*	5	0,83333333	0,217014	4	1	5	0,086806	0,021701	0,108507
Б1б*	1	0,16666667	0,043403	1	1	1	0,014468	0,014468	0,014468
						Итого:	0,281002	**0,258292**	0,460706

Таблица 3. Экспертная оценка проектов реализации МДВ

Взвешивание факторов				Оценка альтернатив			Взвешивание альтернатив		
Фактор	Балл	Вес в кусте	Вес	Д1	Д2	Д3	**Д1**	Д2	Д3
А	5	0,41666667							
Б	5	0,41666667							
В	2	0,16666667							
А1*	5	0,3125	0,130208	4	5	3	0,043403	0,054253	0,032552
А2*	4	0,25	0,104167	1	4	5	0,010417	0,041667	0,052083
А3	5	0,3125	0,130208						
А4*	2	0,125	0,052083	2	1	4	0,014881	0,00744	0,029762
Б1	3	0,375	0,15625						
Б2*	5	0,625	0,260417	5	5	1	0,118371	0,118371	0,023674
В1*	5	0,5	00,83333	3	2	5	0,025	0,016667	0,041667
В2*	4	0,4	0,066667	4	3	5	0,022222	0,016667	0,027778
В3*	1	0,1	0,016667	1	2	4	0,002381	0,004762	0,009524
А3а*	2	0,4	0,052083	1	3	3	0,00744	0,022321	0,022321
А3б*	3	0,6	0,078125	1	2	2	0,015625	0,03125	0,03125
Б1а*	5	0,625	0,097656	4	5	2	0,035511	0,044389	0,017756
Б1б*	3	0,375	0,058594	2	3	3	0,014648	0,021973	0,021973
						Итого:	**0,3099**	0,37976	0,31034

Несмотря на различия в характере трех задач сейсморазведки, общая для них простая однородная структура данных решающим образом влияет на распараллеливание. Геометрическая декомпозиция со структурой «Одна программа / много данных» или «Хозяин — работники» оказывается естественным выбором для этих и других подобных задач. МАИ помогает уточнить некоторые детали реализации на основе собственных оценок разработчика. Опыт разработки и использования подобных задач на целевой платформе, понимание алгоритма и реальная оценка объемов данных, которые предстоит обработать программе, играют большую роль во взвешивании факторов, ранжировании альтернатив и уточнении деталей проекта. В нашем случае анализ показал преимущества проектов Ф2, М2 и Д1.

3. ОПТИМИЗАЦИЯ ПАРАЛЛЕЛЬНЫХ ПРОГРАММ

Пользователи параллельных компьютеров часто сталкиваются с тем, что производительность прикладной программы оказывается гораздо ниже, чем можно ожидать, исходя из оценки производительности компьютера по тесту Linpack. Это естественно, учитывая, сколько усилий и времени обычно затрачивают разработчики на подбор параметров Linpack для обеспечения наибольшей производительности кластера. Оптимизация параметров и среды выполнения каждой прикладной программы также требует проведения ряда экспериментов. Поскольку длительные эксперименты на больших компьютерах — дорогое удовольствие, важное направление теории высокопроизводительных вычислений — прогнозирование производительности программы в разных условиях ее выполнения по результатам небольшого числа тестовых запусков.

Ключ к ускорению программы — обнаружение «узкого места», с которым связаны основные потери производительности на определенном вычислителе. Если доступен исходный текст программы, то временные затраты на выполнение определенных операций можно измерить, вставив в него инструкции для печати времени выполнения определенных участков кода. Однако для пользователей коммерческих программ эта возможность отсутствует. Одним из первых методов идентификации «узкого места» по косвенным признакам стала метрика Карпа–Флатта (1990):

$$e = \frac{1/S_N - 1/N}{1 - 1/N}. \qquad (6)$$

Формула (6) описывает суммарные накладные расходы однородного вычисления в памяти гомогенного компьютера на синхронизацию и коммуникацию. Относительные затраты времени на синхронизацию почти не зависят от числа ПЭ (они возникают в результате различного времени вычисления подзадач разными потоками), а затраты времени на коммуникацию приблизительно линейно растут с увеличением числа ПЭ в силу ограниченной полосы пропускания коммуникационной среды, причем для $N=1$ затраты на коммуникацию — нулевые. Итак, линейная аппроксимация динамики изменения метрики Карпа–Флатта с ростом числа ПЭ позволяет разделить степень влияния синхронизации и коммуникации на потери производительности (табл. 4).

Таблица 4. Примеры определения «узкого места» по метрике Карпа–Флатта

Вариант 1								**Вариант 2**							
N	2	3	4	5	6	7	8	N	2	3	4	5	6	7	8
S_N	1,8	2,5	3,1	3,6	4,0	4,4	4,7	S_N	1,9	2,6	3,2	3,7	4,1	4,5	4,7
e	0,1	0,1	0,1	0,1	0,1	0,1	0,1	e	0,07	0,075	0,08	0,085	0,09	0,095	0,1
Вывод: синхронизация доминирует среди накладных расходов								**Вывод: коммуникация доминирует среди накладных расходов**							

Для обработки сейсмических данных решающим фактором является их объем. В большинстве случаев сейсмические данные слишком велики, чтобы держать их в оперативной памяти на протяжении всего времени выполнения программы. В результате увеличение числа процессоров приводит к пропорциональному росту интенсивности файловых операций. «Узким местом» оказывается файловая система или канал доступа к файловому хранилищу, обладающий ограниченной пропускной способностью.

Рассмотрим модель синхронной параллельной обработки большого объема данных, предложенную в [8]. Пусть N процессов выполняются параллельно. Работу каждого из них можно представить в виде чередования расчетов продолжительностью t_i^n и обращений к данным (чтения или записи файлов) продолжительностью τ_i^n с ожиданием продолжительностью ω_i^n. Другие S процессов или устройств обслуживают обращения к данным. Каждый из них ждет поступления запроса от одного из вычислительных процессов, затем выполняет его. Пока обслуживающий процесс выполняет запрос, на него могут поступить другие запросы. Все они ставятся в очередь неограниченной емкости и обслуживаются в порядке поступления. Коммуникационные расходы, не относящиеся к чтению/записи данных, предполагаем пренебрежимо малыми. Время завершения всех N процессов, состоящих из i шагов, обозначим $T_N = \max\limits_{n=1..N}(t_0^n + \omega_1^n + \tau_1^n + t_1^n + \omega_2^n + \tau_2^n + t_2^n + \ldots + \omega_i^n + \tau_2^n + t_2^n)$.

Пусть общие затраты на чтение/запись данных последовательной программы $D_1 = \sum \tau_i$, в том числе на операции чтения приходится R_1. Обозначим суммарное время выполнения обязательных для каждого процесса (не распараллеливаемых) операций чтения τ_{sec}^R, записи — τ_{sec}^W, максимальное время одной операции чтения/записи — $\tau_{\max}$, среднее время операции доступа к данным — $\mu(\tau)$, средний квадрат — $\mu(\tau^2)$. Все перечисленные параметры можно непосредственно измерить. Обозначим $0<\eta\le 1$ долю распараллеливаемой части расчетов. Значение η определяется по результатам небольшого числа экспериментов с параллельной программой. Доля расчетной части последовательной программы составляет $\theta=(T_1-D_1)/T_1$, доля чтения $\rho=R_1/T_1$.

Поведение такой модели зависит от соотношения между временами расчетов и доступа к данным. Рассмотрим два предельных случая: со слабой загрузкой, когда очередь запросов на доступ к данным одной операции успевает «рассосаться» до начала следующей операции доступа к данным ($\forall \sum\limits_n \tau_i^n << t_{i+1}^n$), и с полной загрузкой, когда эта очередь, возникнув на первом шаге, продолжает расти ($\forall i \sum\limits_n \tau_i^n \ge t_{i+1}^n$).

При дублировании на S устройствах расположены копии данных. В частном случае централизации единственная копия данных управляется одним процессом на одном узле кластера или в дисковой стойке ($S = 1$). Для стратегии дублирования при слабой загрузке время работы параллельной программы составляет

$$\mu(T_N) = \left(\theta\left(1-\eta\frac{N-1}{N}\right)+\frac{1-\theta}{N}\left(1-\rho\frac{S-1}{S}\right)\right)T_1 + \\ + \frac{N-1}{N}\left(\tau^W_{\text{sec}} + \frac{\tau^R_{\text{sec}}}{S}\right) + \frac{N-1}{S}\tau_{\max}\,. \tag{7}$$

При большом числе процессов и интенсивном обращении к файлам обслуживание чтения и записи данных почти полностью определяет время работы

$$\mu(T_N) = (1-\theta)(1-\rho\frac{S-1}{S})T_1 + (N-1)(\tau^W_{\text{sec}} + \frac{\tau^R_{\text{sec}}}{S}) + t_{\text{const}}, \tag{8}$$

где $t_{\text{const}} = t_0^1 + t_i^N$.

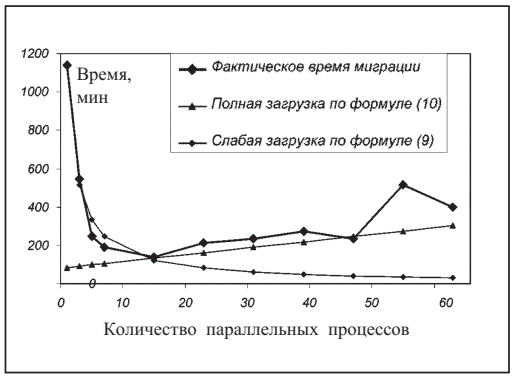
Расчленение означает, что непересекающиеся подмножества данных хранятся на разных устройствах. В случае расчленения время выполнения зависит от распределения данных между фрагментами, но дублирования записи не требуется. Для случайного распределения можно оценить среднее значение времени работы при слабой загрузке как

$$\mu(T_N) = T_1\left(\frac{1}{2}C_N + D_{NS} + \sqrt{\frac{1}{4}C_N^2 + D_{NS}^2\,\frac{\mu(\tau^2)}{2\mu(\tau)^2}}\right), \tag{9}$$

где $C_N = \theta\left(1-\eta\frac{N-1}{N}\right)$, $D_{NS} = \frac{1}{NS}\left(1-\theta+(N-1)\frac{\tau^W_{\text{sec}}+\tau^R_{\text{sec}}}{T_1}\right)$. При полной загрузке

$$\mu(T_N) = \frac{1-\theta}{S}T_1 + (N-1)\frac{\tau^W_{\text{sec}}+\tau^R_{\text{sec}}}{S} + t_{\text{const}}\,. \tag{10}$$

Экспериментальное опробование на десятке разных кластеров показало довольно высокую точность формул (7)–(10). Основной практический результат заключается в том, что точку пересечения теоретических графиков слабой и полной загрузки можно использовать в качестве первого приближения оптимального для задачи количества процессоров (рис. 9).



Šćń. 9. Ćšąōčź ļšīćķīēķīćī č ōąźņč÷åńźīćī āšåģåķč ĢÄĀ

Полученные оценки позволяют также взаимно настраивать программы и кластер, подбирая оптимальные конфигурации (способ размещения данных, значения S и N) для новых задач, исходя из ожидаемого времени операций доступа к данным и вычислений. Эти времена можно приблизительно оценить по аналогии с другими запусками той же программы, предполагая пропорциональность времени обработки объему исходных данных.

ЗАКЛЮЧЕНИЕ

Параллельные программы обработки данных сейсморазведки реализованы и эксплуатируются на кластерах СКИТ с 2004 г. Накоплен набор несложных приемов ускорения задач за счет уменьшения нагрузки на файловую систему:

– использование локальных папок для временных файлов отдельных процессов,

– дублирование интенсивно используемых данных (скоростной модели) на узлах,

– создание и применение индексных файлов для ускорения навигации в больших файлах данных,

– распределение временных разделяемых файлов по нескольким папкам для уменьшения скорости работы с директориями.

Администраторы СКИТ проделали большую работу для повышения производительности аппаратуры файлового хранилища и пропускной способности сети, обслуживающей доступ к дискам, оптимизации параллельной файловой системы и кеширования временных локальных папок. В результате получен значительный выигрыш в производительности программ обработки сейсмических данных. Если в 2005 г. МДВ площади 16 км^2/28 ГБ на 16 ПЭ выполнялась 12 суток, а на 48 ПЭ — даже 16 суток, то в 2009 г. МДВ площади 100 км^2/65 ГБ выполнялась 32 ч. на 56 ПЭ, а площадь 500 км^2/108 ГБ на 140 ПЭ была мигрирована за трое суток.

Дисковые операции не используются и соответственно не учитывается Linpack, однако они критически важны для многих реальных приложений, связанных с обработкой больших объемов данных, в том числе для рассмотренных задач сейсморазведки.

СПИСОК ЛИТЕРАТУРЫ

1. Top 50 СНГ (http://supercomputers.ru/?page=rating)
2. Marmalevskyi N., Roganov Y., Kostyukevych A., Roganov V. Duplex wave migration and interferometry for imaging onshore data without angle limitations / 70th EAGE Conf. & Exhibition — Rome, Italy, 9–12 June 2008, #P273 (http://www.tesseral-geo.com/publications/Marmalevskyi%20-%20Duplex%20Wave%20Migration%20and%20Interferometry.zip).
3. Kostyukevych A., Marmalevskyi N., Roganov Y., Tulchinsky V. Anisotropic 2.5D–3C finite-difference modeling / 70th EAGE Conf. & Exhibition. — Rome, Italy, 9–12 June 2008, #P043 (http://www.tesseral-geo.com/publications/Kostyukevych%20-%20Anisotropic%202.5D%20-%203C%20Finite-difference%20Modeling.zip).
4. Тяпкин Ю.К., Тяпкина Е.Ю. Поляризационная фильтрация поверхностных волн-помех на многокомпонентных сейсмических записях. Одноканальный метод // Геофизический журнал. — 2008. — **30**, № 3. — С. 14–26.
5. MacDonald S., Schaeffer J., Szafron D. Pattern-based object-oriented parallel programming // Proc. of ISCOPE'97. — LNCS, 1997. — **1343**. — P. 267–274.
6. Mattson T.G., Sanders B.A., Massingill B.L. Patterns for Parallel Programming. — Addison-Wesley, 2004. — 355 p.
7. Саати Т. Принятие решений. Метод анализа иерархий. — М.: Радио и связь, 1993. — 320 с.
8. Перевозчикова О.Л., Тульчинский В.Г., Ющенко Р.А. Построение и оптимизация параллельных компьютеров для обработки больших объемов данных // Кибернетика и системный анализ. — 2006. — № 4. — С. 117–129.

Поступила 08.07.2009