Многоуровневый дизайн параллельной реализации сеточных методов решения прямых и обратных задач *

С.С. Титаренко[1], И.М. Куликов[2,3], И.Г. Черных[2], М.А. Шишленин[2,3,4], О.И. Криворотько[2,3], Д.А. Воронов[2,3], М. Хилдъярд[1]

School of Earth and Environment, University of Leeds[1],
Институт вычислительной математики и математической геофизики Сибирского отделения РАН[2],
Новосибирский государственный университет[3],
Институт математики им. С.Л. Соболева Сибирского отделения РАН[4]

В работе представлены результаты по разработке сеточных методов для решения прямых и обратных задач на различных стадиях эффективной реализации. Основное внимание уделено решению гиперболических уравнений конечно-разностными и конечно-объемными методами на многоядерных вычислительных архитектурах. Рассмотрены различные стадии параллельной реализации: геометрическое разбиение расчетной области, распределение задач по потокам средствами OpenMP, векторизация вычислений и вопросы эффективности. Представлены результаты исследования производительности астрофизического кода AstroPhi на гибридном кластере Политехник RSC PetaStream, оснащенном ускорителями Intel Xeon Phi, векторизация геофизических вычислений на процессорах Intel Core i7-3930K и исследования энергоэффективности. Показаны результаты вычислительных экспериментов, полученных с помощью разработанных реализаций.

Ключевые слова: Высокопроизводительные вычисления, ускорители Intel Xeon Phi, Intel Core i7, сеточные методы

Введение

В основе математического моделирования различных природных процессов и явлений лежит решение гиперболических уравнений с помощью конечно-разностных или конечно-объемных схем. Современное состояние математического моделирования накладывает высокие требования к разрешениям численных моделей, сложности подсеточной физики, а также к точности численных методов. Все это приводит к необходимости эффективного использования высокопроизводительных вычислений. Если еще десять лет назад стандартом параллельной реализации было геометрическая декомпозиция расчетной области средствами MPI (Message Passing Interface, интерфейс передачи сообщений), то сегодня, в связи с развитием и повсеместным внедрением ускорителей различного типа (GPU и Intel Xeon Phi), эффективная параллельная реализация это комбинация геометрической декомпозиции области, распределение подзадач по потокам и векторизация вычислений при ограничениях, связанных с энергоэффективностью.

При решение прямых и обратных задач геофизики, ввиду их сверхсложности, часто невозможно найти точное решение и его заменяют приближенным численным. Важным примером прямого численного моделирования является распространение волны в упругих средах. Впервые задача была решена конечно-разностных методом в работе [1] и применена к генерации синтетических сейсмограм в [2]. Позднее, похожий подход был применен для

*Работа поддержана грантами Российского Фонда Фундаментальных Исследований 14-01-00208, 15-31-20150 мол-а-вед, 15-01-00508, 16-01-00755, 16-31-00382 и 16-31-00189, грантом Президента РФ МК — 6648.2015.9. и Министерством образования и науки Российской Федерации. Грантами RCUK grant NE/L000423/1 и NERC, the Environment Agency and Radioactive Waste Management Ltd.

генерации звукового поля в задачах акустики (см. [3, 4]). Решение прямой задачи также используется в задачах обратного волнового преобразования. Все эти задачи требуют больших компьютерных ресурсов и времени, поэтому большинство научных групп в настоящее время используют оптимизацию посредством распределенной памяти, распараллеливание на графических процессорах и кластерах с помощью MPI. Примеры решения задач геофизики на суперкомпьютерах представлены в [5–10].

Решение задач конечно-разностными методами подразумевает вычисление производной на структуре типа stencil. К сожалению применение высокуровневой векторизации к циклам такого типа является невозможным в силу его структуры. В данной работе мы представляем метод по реорганизации памяти, который позволяет применить векторизацию, не применяя низкоуровневых команд. Он является универсальным для любых вычислений на структурах типа stencil. Применение выбранной векторизации повышает скорость вычислений, причем время вычислений практически не зависит от порядка производной. Последнее позволяет существенно повысить точность схемы, при этом не увеличивая времени вычислений. Мы обсуждаем особенности одновременного использования векторизации и OpenMP директив. Полученные результаты продемонстрированы на задаче распространения волны в упругих средах.

Магнитное поле играет ключевую роль в формировании и динамике астрофизических объектов. Важную роль учет влияния магнитного поля оказывает на развитие межзвездных турбулентных течений, где магнитное поле достаточно сильное [11]. В задачах развития МГД (магнитная гидродинамика) турбулентности были исследованы субальфвеновские течения течения [12], скорость звездообразования [13] и проведено сравнение различных кодов на задаче сверхзвуковой турбулентности [14]. Для моделирования астрофизических МГД течений за последнюю декаду создано большое число оригинальных новых численных методов и кодов, помимо более ранних классических, AMR (Adaptive Mesh Refinement, метод динамической адаптации разностных сеток) и SPH (Smoothed-Particle Hydrodynamics, гидродинамика сглаженных частиц), подходов (см. обзор методов и кодов в [15–17]).

Коэффициентная обратная задача для двумерной системы гиперболических уравнений исследована в работе [18]. В этой работе задача акустической томографии сформулирована в виде коэффициентной обратной задачи для системы гиперболических уравнений первого порядка (системы уравнений акустики). Для решения обратной задачи применен градиентный метод оптимизации целевого функционала, широко используемый в теории обратных и некорректных задач [19–23]. На каждом шаге итерационного процесса оптимизации решается прямая и сопряженная задачи, в связи с чем возникает необходимость использования эффективных численных методов решения прямой задачи предлагаемый в данной статье.

Первые два раздела настоящей работы будут посвящены реализации кода AstroPhi [16] с использованием геометрической декомпозиции расчетной области (первый раздел) и распределением задач средствами OpenMP в native и offload режимах (второй раздел). Третий раздел будет посвящен исследованиям векторизации сеточных вычислений с помощью конечно-разностных схем. Четвертый раздел посвящен вопросам энергоэффективности вычислений. В пятом разделе будут кратко приведены результаты вычислительных экспериментов.

1. Геометрическая декомпозиция расчетной области

Использование равномерной сетки в декартовых координатах для решения уравнений гидродинамики позволяет использовать произвольную декартову топологию для декомпозиции расчетной области. Такая организация вычислений имеет потенциально бесконечную масштабируемость. В коде AstroPhi используется многоуровневая одномерная геометрическая декомпозиция расчетной области [16]. По одной координате внешнее одномерное разрезание происходит средствами технологии MPI, внутри каждой подобласти разрезание

Таблица 1: Исследование масштабируемости кода на различном числе ускорителей суперкомпьютера Политехник RSC PetaStream СПбПУ.

MIC	Total (SPb)	Scalability (SPb)
1	55,5742	1,0000
8	56,3752	0,9857
64	64,1803	0,8659
128	68,6065	0,8101
256	76,1687	0,7296

происходит средствами OpenMP, адаптированного для MIC-архитектур. Такая декомпозиция связана с топологией и архитектурой гибридного суперЭВМ RSC PetaStream, который был использован для вычислительных экспериментов.

Проводилось исследование масштабируемости кода AstroPhi на ускорителях Intel Xeon Phi 5120 D. Для чего была использована сетка $512p \times 256 \times 256$ и четыре логических ядра на каждый ускоритель, где p — число используемых ускорителей. Таким образом, на каждый ускоритель приходится одинаковый размер подобласти при любом числе исследуемых ускорителей. Для исследования масштабируемости замерялось время каждого этапа численного метода, в секундах, а затем вычислялась их сумма (Total) при различном числе используемых ускорителей Intel Xeon Phi (MIC). Масштабируемость T вычислялась по формуле

$$T = \frac{\text{Total}_1}{\text{Total}_p} \tag{1}$$

где Total_1 — время вычислений на одном ускорителе при использовании одного ускорителя, Total_p — время вычислений на одном ускорителей при использовании p ускорителей. Результаты исследований масштабируемости приведены в таблице 1. Таким образом, была получена 73-процентная эффективность на 256 ускорителях Intel Xeon Phi 5120 D.

2. Распределение задач в рамках ускорителя Intel Xeon Phi

При реализации кода AstroPhi на ускорителях Intel Xeon Phi был использован регулярный шаблон вычислений, который следует из схемы декомпозиции расчетной области и состоит в распределении работ по нитям.

Исследование ускорения проводилось на сетке 512×256^2. Для измерения ускорения замерялось время каждого этапа численного метода, в секундах, а затем вычислялась их сумма при различном числе используемых логических ядер (Threads). Ускорение P вычислялось по формуле:

$$P = \frac{\text{Total}_1}{\text{Total}_K} \tag{2}$$

где Total_1 — время вычислений на одном логическом ядре, Total_K — время вычислений при использовании K логических ядер. Результаты исследований ускорения для суперкомпьютера Политехник RSC PetaStream СПбПУ (SPb) приведены в таблице 2. Таким образом, было получено 84-кратное ускорение в рамках одного ускорителя Intel Xeon Phi 5120 D.

3. Векторизация вычислений

Векториазция является эффективным способом ускорить вычисления на современных компьютерах. Практически все современные архитектуры на сегодняшний день поддержи-

Таблица 2: Исследование ускорения кода на различном числе логических ядер суперкомпьютера Политехник RSC PetaStream СПбПУ.

Threads	Total (SPb)	Speed-Up (SPb)
1	219,7956	1,0000
8	27,7089	7,9323
32	7,9673	27,5872
128	2,6271	83,6647
240	2,5905	84,8467

вают SIMD операции (Single Instruction Multiple Data, одиночный поток команд, множественный поток данных). Лидерами на рынке являются процессоры Intel с SSE (Streaming SIMD Extensions) и AVX (Advanced Vector Extensions) инструкциями, Apple и IBM с AltiVec инструкциями и ARM архитектуры с NEON. В данной работе задачи вычисления производились на процессоре Intel iCore-7 с использованием SSE и AVX инструкций.

Если при подготовке программы явно не указать на использование векторизации, то компилятор будет загружать значения в регистры и применять последующие арифметические и логические операции к *по одному* значению. Однако, современные процессоры имеют набор регистров размера 128 (XMM тип), 256 (YMM тип) и 512 (MIC тип) бит. Это означает что в один регистр можно загрузить четыре (восемь, шестнадцать) 32-разрядных числа одинарной точности или два (четыре, восемь) 64-разрядных числа двойной точности. В этом случае при использовании соответствующего флага компилятор может применить арифметическую/логическую операцию к *вектору*, что приведет к ускорению вычислений.

Подобный подход осуществим при включении определенного флага оптимизации (`/O2` и выше) или выбора определенных свойств процессора с помощью разных флагов (к примеру, `/QxSSE4.1`, `/QxAVX`) и применим к циклам с выровненной памятью, длиной вектора кратной как минимум 4. При этом не должны происходить конфликты при чтении и записи в память. К примеру, цикл типа

```
// loop with a unit stride
for(int i = 0; i < N; i++){
  a[i] = b[i] + c[i];
}
```

будет автоматически векторизован (предполагая что `N` кратно четырем, а память выравнена). Однако приведенный ниже цикл не может быть векторизован высокоуровневыми инструкциями.

```
// loop with a unit stride
__assumed_aligned(a,32);
__assumed_aligned(b,32);
#pragma simd
for(int i=0; i<N; i++){
  a[i] = b[i] + b[i+2];
}
```

Вычисление производной на структуре типа stenil является типичным примером такого цикла. Ситуация однако меняется, если мы перегруппируем память так, как это показано на рис. 1. В этом случае мы избегаем конфликта памяти, что позволяет компилятору загружать и применять арифметические операции четверками.

Необходимо отметить, что память можно перегруппировать так, что компилятор будет работать с векторами размером 8/16 (для переменных одинарной точности) и 4/8 (для переменных двойной точности). В таком случае значения, ответственные за расчет производной в точке i, будут располагаться друг от друга с шагом 8.

Важной особенностью данного подхода является не только то, что он позволяет векторизовать цикл высокоуровневыми инструкциями, но и то, что порядок производной практически не влияет на скорость вычисления. Последнее обстоятельство является очень важным в задачах, где получение требуемой точности решения сталкивается с проблемой стоимости машинного времени.

В качестве примера была рассмотрена задача вычисления производной второго, четвертого, шестого, восьмого и десятого порядков на сетке размером 8192×8192 (500 циклов). Таблица 3 показывает ускорение задачи для разного порядка производных без оптимизации (флаг `/Od`), с применением автоматической агрессивной оптимизации без перераспределения памяти и агрессивной векторизации с перераспределением памяти (флаг `/O3`).

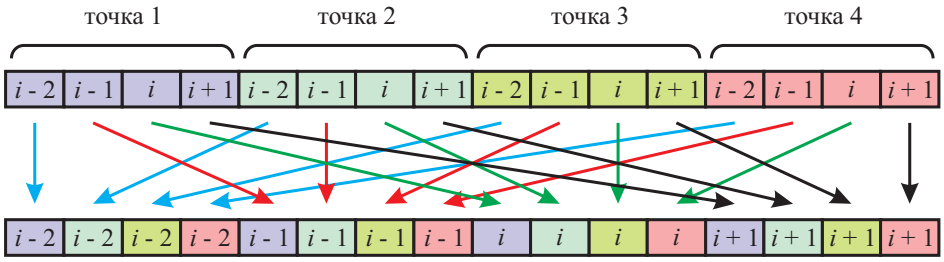

Рис. 1: Реогранизация памяти для выполнения высокоуровневой векторизации.

Таблица 3: Применение различных уровней оптимизации для производных высокого порядка. Время вычисления в секундах.

порядок	`/Od`	`/O3`, $[1 \times 1]$	`/O3`, $[4 \times 1]$
2	311,601	63,978	40,987
4	378,303	96,804	41,502
6	433,277	122,147	42,687
8	485,380	158,753	41,350
10	543,615	192,288	46,996

4. Исследование энергоэффективности

На текущий момент проблема энергоэффективности алгоритмов и программ наиболее актуальна для коммерческих центров обработки данных. Однако, с приближением эры суперкомпьютеров экзафлопсной производительности, роль энергоэффективности для HPC будет только расти. Неэффективное использование десяток мегаватт электроэнергии для экзафлопсных систем может свести на нет всю идею построения машин такого класса. В целом, под энергоэффективностью понимают порядка 20 показателей, большая часть которых сводится к вычислительной эффективности. Мы исследовали энергоэффективность по следующим параметрам: наиболее эффективное использование каждого ядра процессора или ускорителя вычислений, минимизацию обменов данными между вычислительными узлами, хорошую балансировку программ. Минимизация обменов данными позволяет минимизировать время простоя ядер процессоров или ускорителей. Хорошая балансировка программы позволяет равномерно нагрузить вычислительную систему. Для кода AstroPhi нам удалось уменьшить время MPI операций до 7–8% от общего времени выполнения программы и добиться уровня разбалансировки процессов не более 2–3% между всеми нитями процессов. Такие показатели позволили получить 72% эффективность (масштабируемость в “слабом” смысле) распараллеливания на 256 ускорителях Intel Xeon Phi (более 50K ядер).

Современные ускорители вычислений позволяют достичь максимальной вычислительной эффективности за счет многопоточности и векторизации. Благодаря векторизации кода AstroPhi с использованием команд SIMD-512 удалось ускорить код в 6,5 раз и приблизиться по вычислительной эффективности к таким библиотекам, как MAGMA MIC.

5. Результаты вычислительных экспериментов

5.1. Моделирование распространения волн

При решении волнового уравнения методом конечных разностей обычно переходят к системе уравнений первого порядка.

$$\begin{cases} \rho\dfrac{\partial u}{\partial t} = \dfrac{\partial \sigma_{11}}{\partial x_1} + \dfrac{\partial \sigma_{12}}{\partial x_2}, & \rho\dfrac{\partial v}{\partial t} = \dfrac{\partial \sigma_{22}}{\partial x_2} + \dfrac{\partial \sigma_{12}}{\partial x_1}, \\ \dfrac{\partial \sigma_{11}}{\partial t} = \Big(\lambda + 2\mu\Big)\dfrac{\partial u}{\partial x_1} + \lambda\dfrac{\partial v}{\partial x_2}, & \dfrac{\partial \sigma_{12}}{\partial t} = \mu\dfrac{\partial u}{\partial x_2} + \mu\dfrac{\partial v}{\partial x_1}, \\ \dfrac{\partial \sigma_{22}}{\partial t} = \lambda\dfrac{\partial u}{\partial x_1} + \Big(\lambda + 2\mu\Big)\dfrac{\partial v}{\partial x_2}, \end{cases} \tag{3}$$

где (u, v) — вектор скорости, σ_{ij} — компоненты тензора напряжений, а λ и μ — параметры Ламэ. При этом осуществляют переход к так называемой сдвинутой сетке. Метод был впервые применен в [24] для решения задач распространения волн в упругих средах и, как показано в [25], является предпочтительным для схем четвертого порядка в силу его лучшей устойчивости.

Обозначим $\delta u \equiv u^{i-2} - 27u^{i-1} + 27u^i - u^{i+1}$, $\delta v \equiv v^{j-2} - 27v^{j-1} + 27v^j - v^{j+1}$, $\delta\sigma_{11} \equiv \sigma_{11}^{i-2} - 27\sigma_{11}^{i-1} + 27\sigma_{11}^i - \sigma_{11}^{i+1}$, $\delta\sigma_{22} \equiv \sigma_{11}^{j-2} - 27\sigma_{11}^{j-1} + 27\sigma_{11}^j - \sigma_{11}^{j+1}$. Тогда при замене производных на конечные разности мы получаем следующие выражения для напряжений σ_{11}, σ_{22}, σ_{12} и скоростей u, v:

$$\sigma_{11}^t = \sigma_{11}^{t-1} + \frac{(\lambda + 2\mu)\Delta t}{24\Delta x_1}\delta u + \frac{\mu\Delta t}{24\Delta x_2}\delta v, \tag{4}$$

$$\sigma_{22}^t = \sigma_{22}^{t-1} + \frac{\mu\Delta t}{24\Delta x_1}\delta u + \frac{(\lambda + 2\mu)\Delta t}{24\Delta x_2}\delta v, \tag{5}$$

$$\sigma_{12}^t = \sigma_{12}^{t-1} + \frac{\Delta t\mu}{24\Delta x_2}\delta u + \frac{\Delta t\mu}{24\Delta x_1}\delta v, \tag{6}$$

$$u^{t+1/2} = u^{t-1/2} + \frac{\Delta t}{24\rho\Delta x_1}\delta\sigma_{11} + \frac{\Delta t}{24\rho\Delta x_2}\left(\sigma_{12}^{j-2} - 27\sigma_{12}^{j-1} + 27\sigma_{12}^j - \sigma_{12}^{j+1}\right), \tag{7}$$

$$v^{t+1/2} = v^{t-1/2}\frac{\Delta t}{24\rho\Delta x_1}\left(\sigma_{12}^{i-2} - 27\sigma_{12}^{i-1} + 27\sigma_{12}^i - \sigma_{12}^{i+1}\right) + \frac{\Delta t}{24\rho\Delta x_2}\delta\sigma_{22}. \tag{8}$$

На рис. 2 показана общая схема блоков векторизации. Применение граничных условий приведет к тому, что они будут применены к внутренним блокам памяти и решение, таким образом, будет расходиться. *Виртуальные блоки* помогают решить эту проблему. Нужно помнить, однако, что на каждом шаге по времени происходит копирование данных из *реальных* в *виртуальные* блоки.

Таблица 4 показывает ускорение для задач разных размеров. При флаге `/O1` компилятор Intel применяет автоматическую оптимизацию, но не векторизацию.

Задача легко распараллеливается посредством OpenMP (здесь мы подразумеваем, что вся область памяти имеет блочную структуру). Распараллеливание по ядрам происходит стандартным способом:

- Каждый блок имеет “буферные” зоны на границах. Для схемы четвертого порядка минимальное число “буферных” точек с каждой границы блока 2.

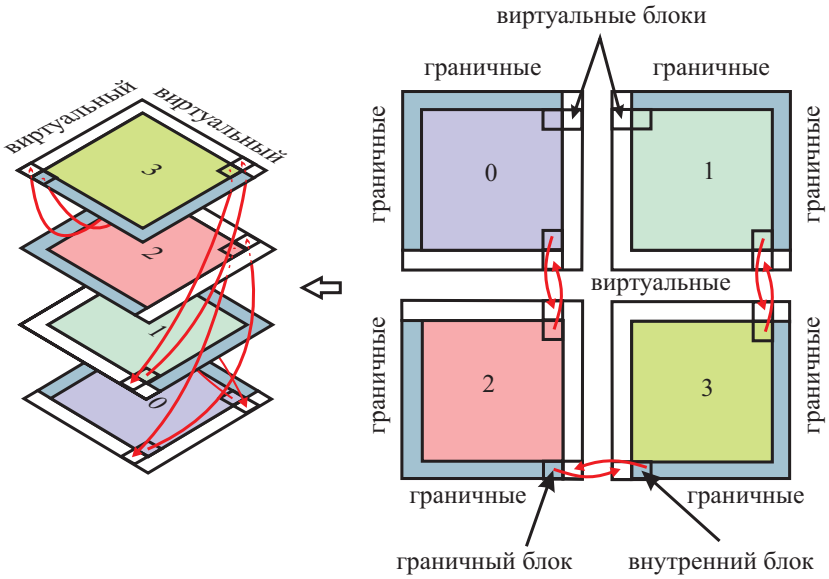

Рис. 2: Копирование данных из реальных блоков данных в виртуальные.

Таблица 4: Сравнение времени вычислений для задач разных размеров без оптимизации, с флагами `/O1` и `/O3`. Задачи посчитаны на одном ядре. Размер задачи равен $N \times N$. Время подсчета в секундах.

N	`/Od`	`/O1`	`/O3`	ускорение
$32 \times 32 \times 2$	216,761	54,143	26,133	8,295
$64 \times 32 \times 2$	974,981	186,679	81,356	11,984
$128 \times 32 \times 2$	3789,820	705,704	309,508	12,245

- На каждом шаге по времени происходит копирование внутренних и “буферных” точек блока.

Блоки запускаются на нитях в любом порядке. Для нашей задачи мы нашли, что следующая структура дает оптимальное ускорение.

```
#pragma omp parallel
 {
    for (int j = 0; j < numberOfSteps; j++){
      int i;
#pragma omp for private (i) schedule(auto)
      //velocity calculations
#pragma omp for private (i) schedule(auto)
      //copy velocities
#pragma omp for private (i) schedule(auto)
      //stress calculations
#pragma omp for private (i) schedule(auto)
      //copy stress
#pragma omp for private (i) schedule(auto)
      //calculate boundaries
#pragma omp for private (i) schedule(auto)
      //copy virtual blocks
    }
 }
```

Необходимо отметить, что блоки векторизации могут быть расположены разным образом и их расположение оказывает небольшое влияние на скорость вычислений. В данной

задаче исследовались структуры типа $[4 \times 1]$, $[1 \times 4]$, $[2 \times 2]$. Все они представлены на рис. 3.

Эксперименты и результаты профилирования с помощью VTune Intel Amplifier показывают, что наиболее удачной композицией является $[4 \times 1]$. Именно её мы и используем в данной работе. Таблица 5 показывает ускорение задачи на многоядерном процессоре.

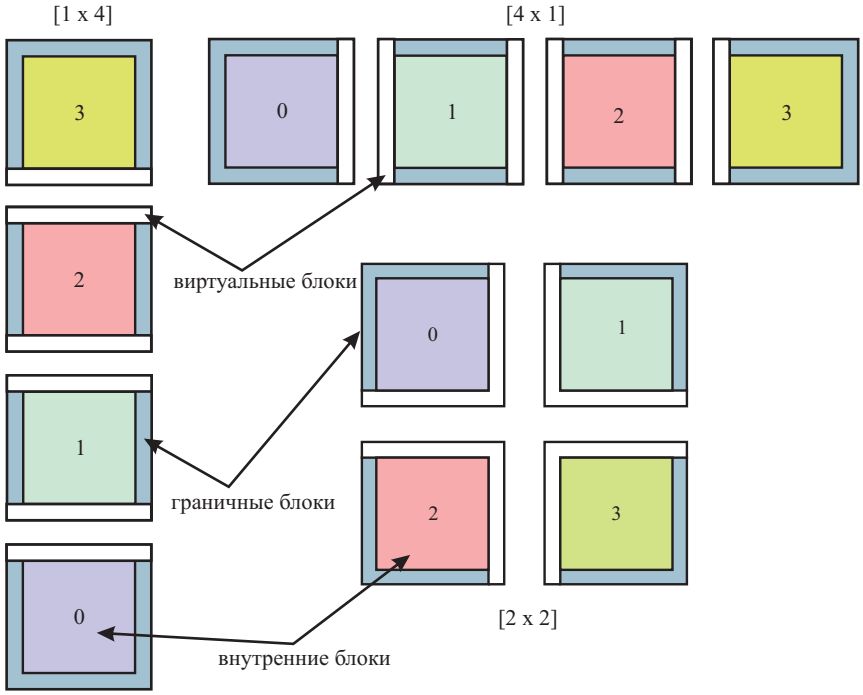

Рис. 3: Возможные структуры блоков векторизации.

Таблица 5: Распараллеливание задачи посредством OpenMP API. Размер задачи $[4096 \times 4096]$, структура памяти типа $[4 \times 1]$. Везде применена наиболее агрессивная высокоуровневая оптимизация. Время подсчета в секундах.

N нитей	время вычислений	ускорение
1	82,3	1,000
4	28,157	2,923
6	25,286	3,255

5.2. Моделирование развития МГД турбулентности

Численная модель основана на совместном решении уравнений многокомпонентной магнитной газовой динамики, обыкновенного дифференциального уравнения для эволюции концентрации ионизированного водорода, а также специальной формы для внешней силы. Внешняя сила является следствием закона сохранения массы и уравнения Пуассона, эволюция по времени которой записывается в виде уравнения типа Коши–Ковалевской. Использование такой модели позволяет сформулировать единый параллельный вычислительный метод [17], основанный на комбинации метода разделения операторов, метода Годунова и кусочно-параболического метода на локальном шаблоне. В результате вычислительного эксперимента была получена плотная область в виде “пальмовой ветки” (см. рис. 4, сверху), отдаленно напоминающая туманность NGC 6188. Видна (см. рис. 4, снизу слева) корреляция $M \sim n^2$ (белая линия) и большая часть облака $n > 10$ см$^{-3}$ попадают в сверхальф-

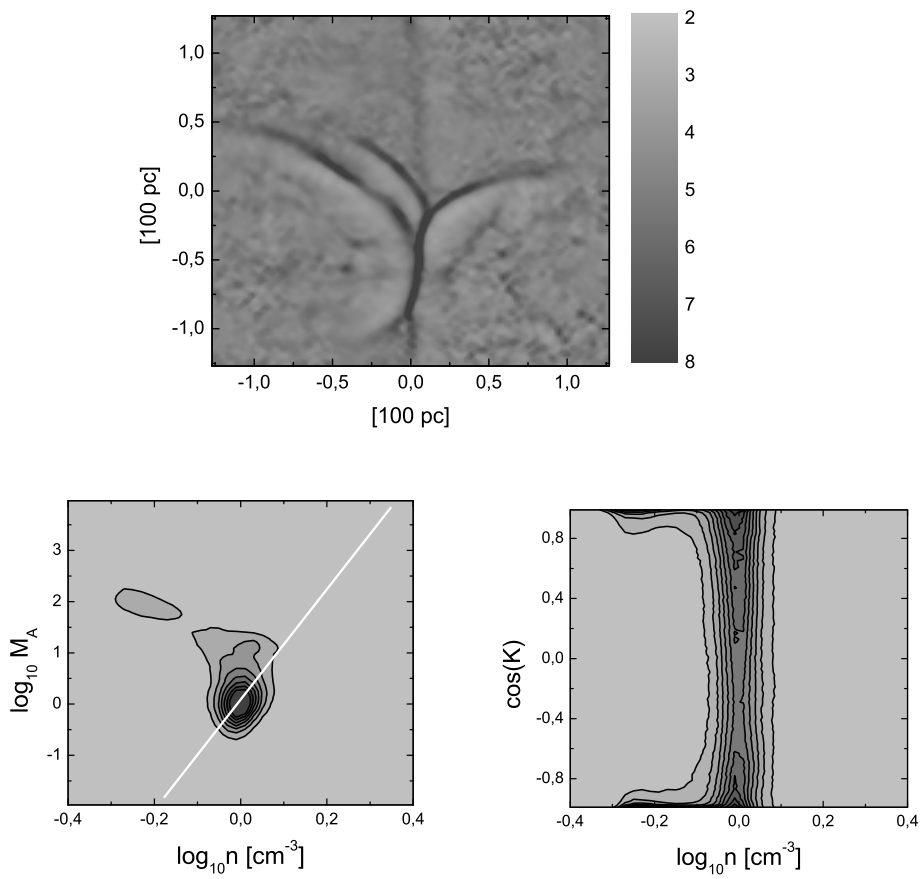

Рис. 4: Задача развития МГД турбулентности межзвездной среды. На рисунке приведена концентрация газа в см$^{-3}$ в момент времени $t = 15$ млн. лет (сверху), зависимость альфвеновской скорости от плотности газа (снизу слева) и косинуса угла колинеарности между векторами скорости и магнитного поля от плотности газа (снизу справа). После процесса ионизации водорода происходит процесс образования облачных структур. Для вычислительного эксперимента использовалось сетка 512^3 ячеек.

веновскую область. Контуры косинуса угла колинеарности между векторами скорости и магнитного поля образуют седловидную структуру (см. рис. 4, снизу справа), что говорит о том, что сжатие происходит вдоль силовых линий магнитного поля.

Заключение

В работе приведены результаты по дизайну сеточных методов для решения прямых и обратных задач на различных стадиях эффективной реализации. Основной фокус сделан на решении конечно-разностными и конечно-объемными методами гиперболических уравнений на многоядерных архитектурах. Рассмотрены различные стадии параллельной реализации: геометрическая декомпозиция расчетной области, распределение задач по потокам средствами OpenMP, векторизация вычислений и вопросы эффективности. Представлены результаты исследования производительности астрофизического кода AstroPhi на гибридном кластере Политехник RSC PetaStream, оснащенном ускорителями Intel Xeon Phi, векторизация геофизических вычислений на процессорах Intel Core i7-3930K и исследования энергоэффективности. Представлены результаты вычислительных экспериментов, полученных с помощью разработанных реализаций.

Авторы выражают благодарность коллегам из организаций Intel (Николаю Местеру и Дмитрию Петунину) и RSC Group (Александру Московскому, Павлу Лавренко и Борису Гагаринову), за предоставление доступа к кластеру RSC PetaStream и подробных консультаций по его использованию.

Литература

1. Alterman Z., Karal F. C. Propagation of elastic waves in layered media by finite-difference methods // Bulletin of the Seismological Society of America. 1968. Vol. 58. P. 367–398.

2. Synthetic seismograms: a finite-difference approach / K. R. Kelly, R. W. Ward, S. Tritel et al. // Geophysics. 1976. Vol. 41. P. 2–27.

3. Sakamoto S., Seimiya T., Tachibana H. Visualization of sound reflection and diffraction using finite difference time domain method // Acoustical Science and Technology. 2002. Vol. 23, no. 1. P. 34–39.

4. Acoustic visualizations using surface mapping / S. Siltanen, P. W. Robinson, J. Saarelma et al. // The Journal of the Acoustical Society of America. 2014. Vol. 135, no. 6. P. EL344–EL349.

5. Michéa D., Komatitsch D. Accelerating a three-dimensional finite-difference wave propagation code using GPU graphics cards // Geophysical Journal International. 2010. Vol. 182, no. 1. P. 389–402.

6. Zhang Y., Gao J. A 3D staggered-grid finite difference scheme for poroelastic wave equation // Journal of Applied Geophysics. 2014. Vol. 109. P. 281–291.

7. Evaluation of the 3-D finite difference implementation of the acoustic diffusion equation model on massively parallel architectures / M. Hernández, B. Imbernón, J. M. Navarro et al. // Computers & Electrical Engineering. 2015. Vol. 46. P. 190–201.

8. Parallel impelemntation of a velocity-stress staggered-grid finite-difference method for 2-D poroelastic wave propagation / D.-H. Sheen, Kagan-Tuncay, C.-E. Baag et al. // Computer & Geoscience. 2006. Vol. 32. P. 1182–1191.

9. Aochi H., Dupros F. MPI-OpenMP hybrid simulations using boundary integral equation and finite difference methods for earthquake dynamics and wave propagation: Application to the 2007 Niigata Chuetsu-Oki earthquake (Mw6.6) // Procedia Computer Science. 2011. Vol. 4. P. 1496–1505.

10. Guyau A., Cupillard P., Kutsenko A. Accelerating a finite-difference time-domain code for the acoustic wave propagation // 35th Gocad Meeting — 2015 RING Meeting. ASGA, 2015.

11. Extended scaling laws in numerical simulations of magnetohydrodynamic turbulence / J. Mason, J. C. Perez, F. Cattaneo et al. // The Astrophysical Journal Letters. 2011. Vol. 735, no. 2. p. L26.

12. McKee C. F., Li P. S., Klein R. I. Sub-Alfvénic non-ideal MHD turbulence simulations with ambipolar diffusion. II. Comparison with observation, clump properties, and scaling to physical units // The Astrophysical Journal. 2010. Vol. 720, no. 2. P. 1612–1634.

13. Federrath C., Klessen R. S. The star formation rate of turbulent magnetized clouds: comparing theory, simulations, and observations // The Astrophysical Journal. 2012. Vol. 761, no. 2. p. 156.

14. Comparing numerical methods for isothermal magnetized supersonic turbulence / A. G. Kritsuk, Å. Nordlund, D. Collins et al. // The Astrophysical Journal. 2011. Vol. 737, no. 1. p. 13.

15. Kulikov I. GPUPEGAS: a new GPU-accelerated hydrodynamic code for numerical simulations of interacting galaxies // The Astrophysical Journal Supplement Series. 2014. Vol. 214, no. 1. p. 12.

16. AstroPhi: a code for complex simulation of the dynamics of astrophysical objects using hybrid supercomputers / I. M. Kulikov, I. G. Chernykh, A. V. Snytnikov et al. // Computer Physics Communications. 2015. Vol. 186. P. 71–80.

17. Kulikov I., Vorobyov E. Using the PPML approach for constructing a low-dissipation, operator-splitting scheme for numerical simulations of hydrodynamic flows // Journal of Computational Physics. 2016. Vol. 317. P. 318–346.

18. Куликов И. М., Новиков Н. С., Шишленин М. А. Математическое моделирование распространения ультразвуковых волн в двумерной среде: прямая и обратная задача // Сибирские Электронные Математические Известия. 2015. Т. 12. С. 219–228.

19. Kabanikhin S. I., Shishlenin M. A. Quasi-solution in inverse coefficient problems // Journal of Inverse and Ill-Posed Problems. 2008. Vol. 16, no. 7. P. 705–713.

20. Kabanikhin S. I., Shishlenin M. A. About the usage of the a priori information in coefficient inverse problems for hyperbolic equations // Proceedings of IMM of UrB RAS. 2012. Vol. 18, no. 1. P. 147–164.

21. Inverse problems for the ground penetrating radar / S. I. Kabanikhin, D. B. Nurseitov, M. A. Shishlenin et al. // Journal of Inverse and Ill-Posed Problems. 2013. Vol. 21, no. 6. P. 885–892.

22. Regularization of the continuation problem for elliptic equations / S. I. Kabanikhin, Y. S. Gasimov, D. B. Nurseitov et al. // Journal of Inverse and Ill-Posed Problems. 2013. Vol. 21, no. 6. P. 871–884.

23. Comparative analysis of methods for regularizing an initial boundary value problem for the Helmholtz equation / S. I. Kabanikhin, M. A. Shishlenin, D. B. Nurseitov et al. // Journal of Applied Mathematics. 2014. Vol. 2014.

24. Virieux J. P-SV wave propagation in heterogeneous media; velocity-stress finite-difference method // Geophysics. 1986. Vol. 51, no. 4. P. 889–901.

25. Moczo P., Kristek J., Bystrický E. Stability and grid dispersion of the P-SV 4th-order staggered-grid finite-difference schemes // Studia Geophysica et Geodaetica. 2000. Vol. 44, no. 3. P. 381–402.

Multilevel parallelization: grid methods used for solving direct and inverse problems

S.S. Titarenko[1], I.M. Kulikov[2,3], I.G. Chernykh[2], M.A. Shishlenin[2,3,4], O.I. Krivorot'ko[2,3], D.A. Voronov[2,3], M. Hildyard[1]

School of Earth and Environment, University of Leeds[1],
Institute of Computational Mathematics and Mathematical Geophysics SB RAS[2],
Novosibirsk State University[3],
Sobolev Institute of Mathematics SB RAS[4]

Grid methods to solve direct and inverse problems with various levels of MPI runtime energy efficiency are presented in the paper. Finite difference and finite volume numerical methods are used to solve hyperbolic equations on multicore architectures. Several parallelization techniques (geometric decomposition of the calculative domain, workload distribution over threads within OpenMP directives, vectorization) are applied and their run-time efficiency is investigated. These developments have been tested on a hybrid Polytechnic RSC PetaStream cluster (consisting of Intel Xeon Phi accelerators) with AstroPhi code for an astrophysical problem and on an Intel Core i7-3930K multicore processor for a geophysical problem. The authors provide results of numerical simulations and their MPI runtime energy efficiency.

Keywords: High performance computing, Intel Xeon Phi accelerators, Grid-based numerical methods

References

1. Alterman Z., Karal F. C. Propagation of elastic waves in layered media by finite-difference methods // Bulletin of the Seismological Society of America. 1968. Vol. 58. P. 367–398.

2. Synthetic seismograms: a finite-difference approach / K. R. Kelly, R. W. Ward, S. Tritel et al. // Geophysics. 1976. Vol. 41. P. 2–27.

3. Sakamoto S., Seimiya T., Tachibana H. Visualization of sound reflection and diffraction using finite difference time domain method // Acoustical Science and Technology. 2002. Vol. 23, no. 1. P. 34–39.

4. Acoustic visualizations using surface mapping / S. Siltanen, P. W. Robinson, J. Saarelma et al. // The Journal of the Acoustical Society of America. 2014. Vol. 135, no. 6. P. EL344–EL349.

5. Michéa D., Komatitsch D. Accelerating a three-dimensional finite-difference wave propagation code using GPU graphics cards // Geophysical Journal International. 2010. Vol. 182, no. 1. P. 389–402.

6. Zhang Y., Gao J. A 3D staggered-grid finite difference scheme for poroelastic wave equation // Journal of Applied Geophysics. 2014. Vol. 109. P. 281–291.

7. Evaluation of the 3-D finite difference implementation of the acoustic diffusion equation model on massively parallel architectures / M. Hernández, B. Imbernón, J. M. Navarro et al. // Computers & Electrical Engineering. 2015. Vol. 46. P. 190–201.

8. Parallel impelemntation of a velocity-stress staggered-grid finite-difference method for 2-D poroelastic wave propagation / D.-H. Sheen, Kagan-Tuncay, C.-E. Baag et al. // Computer & Geoscience. 2006. Vol. 32. P. 1182–1191.

9. Aochi H., Dupros F. MPI-OpenMP hybrid simulations using boundary integral equation and finite difference methods for earthquake dynamics and wave propagation: Application to the 2007 Niigata Chuetsu-Oki earthquake (Mw6.6) // Procedia Computer Science. 2011. Vol. 4. P. 1496–1505.

10. Guyau A., Cupillard P., Kutsenko A. Accelerating a finite-difference time-domain code for the acoustic wave propagation // 35th Gocad Meeting — 2015 RING Meeting. ASGA, 2015.

11. Extended scaling laws in numerical simulations of magnetohydrodynamic turbulence / J. Mason, J. C. Perez, F. Cattaneo et al. // The Astrophysical Journal Letters. 2011. Vol. 735, no. 2. p. L26.

12. McKee C. F., Li P. S., Klein R. I. Sub-Alfvénic non-ideal MHD turbulence simulations with ambipolar diffusion. II. Comparison with observation, clump properties, and scaling to physical units // The Astrophysical Journal. 2010. Vol. 720, no. 2. P. 1612–1634.

13. Federrath C., Klessen R. S. The star formation rate of turbulent magnetized clouds: comparing theory, simulations, and observations // The Astrophysical Journal. 2012. Vol. 761, no. 2. p. 156.

14. Comparing numerical methods for isothermal magnetized supersonic turbulence / A. G. Kritsuk, Å. Nordlund, D. Collins et al. // The Astrophysical Journal. 2011. Vol. 737, no. 1. p. 13.

15. Kulikov I. GPUPEGAS: a new GPU-accelerated hydrodynamic code for numerical simulations of interacting galaxies // The Astrophysical Journal Supplement Series. 2014. Vol. 214, no. 1. p. 12.

16. AstroPhi: a code for complex simulation of the dynamics of astrophysical objects using hybrid supercomputers / I. M. Kulikov, I. G. Chernykh, A. V. Snytnikov et al. // Computer Physics Communications. 2015. Vol. 186. P. 71–80.

17. Kulikov I., Vorobyov E. Using the PPML approach for constructing a low-dissipation, operator-splitting scheme for numerical simulations of hydrodynamic flows // Journal of Computational Physics. 2016. Vol. 317. P. 318–346.

18. Kulikov I. M., Novikov N. S., Shishlenin M. A. Matematicheskoe modelirovanie rasprostraneniya ul'trazvukovykh voln v dvumernoy srede: pryamaya i obratnaya zadacha [Mathematical modelling of ultrasound wave propagation in 2D medium: direct and inverse problems]. Sibirskie Elektronnye Matematicheskie Izvestiya [Siberian Electronic Mathematical Reports] 2015. Vol. 12. P. 219–228.
(Куликов И. М., Новиков Н. С., Шишленин М. А. Математическое моделирование распространения ультразвуковых волн в двумерной среде: прямая и обратная задача // Сибирские Электронные Математические Известия. 2015. Т. 12. С. 219–228.)

19. Kabanikhin S. I., Shishlenin M. A. Quasi-solution in inverse coefficient problems // Journal of Inverse and Ill-Posed Problems. 2008. Vol. 16, no. 7. P. 705–713.

20. Kabanikhin S. I., Shishlenin M. A. About the usage of the a priori information in coefficient inverse problems for hyperbolic equations // Proceedings of IMM of UrB RAS. 2012. Vol. 18, no. 1. P. 147–164.

21. Inverse problems for the ground penetrating radar / S. I. Kabanikhin, D. B. Nurseitov, M. A. Shishlenin et al. // Journal of Inverse and Ill-Posed Problems. 2013. Vol. 21, no. 6. P. 885–892.

22. Regularization of the continuation problem for elliptic equations / S. I. Kabanikhin, Y. S. Gasimov, D. B. Nurseitov et al. // Journal of Inverse and Ill-Posed Problems. 2013. Vol. 21, no. 6. P. 871–884.

23. Comparative analysis of methods for regularizing an initial boundary value problem for the Helmholtz equation / S. I. Kabanikhin, M. A. Shishlenin, D. B. Nurseitov et al. // Journal of Applied Mathematics. 2014. Vol. 2014.

24. Virieux J. P-SV wave propagation in heterogeneous media; velocity-stress finite-difference method // Geophysics. 1986. Vol. 51, no. 4. P. 889–901.

25. Moczo P., Kristek J., Bystrický E. Stability and grid dispersion of the P-SV 4th-order staggered-grid finite-difference schemes // Studia Geophysica et Geodaetica. 2000. Vol. 44, no. 3. P. 381–402.