УДК 004.021:550.34

А. А. Никитин, **А. С. Сердюков**, **А. А. Дучков**

Институт нефтегазовой геологии и геофизики им. А. А. Трофимука СО РАН
пр. Акад. Коптюга, 3, Новосибирск, 630090, Россия

Новосибирский государственный университет
ул. Пирогова, 2, Новосибирск, 630090, Россия

NikitinAA@ipgg.sbras.ru, AleksanderSerdyukov@yandex.ru
DuchkovAA@ipgg.sbras.ru

ПАРАЛЛЕЛЬНЫЙ АЛГОРИТМ РЕШЕНИЯ УРАВНЕНИЯ ЭЙКОНАЛА ДЛЯ ТРЕХМЕРНЫХ ЗАДАЧ СЕЙСМОРАЗВЕДКИ [*]

Решение уравнения эйконала применяется при обработке больших объемов сейсмических данных в задачах сейсморазведки для расчета времен первых вступлений сейсмических волн. В данной работе представлен новый параллельный алгоритм решения уравнения эйконала на основе численного метода алгоритма Fast Sweeping Method (FSM). Данный алгоритм показал более высокую эффективность по сравнению с существующими параллельными реализациями FSM за счет оптимизации работы с кэш-памятью процессора.

Ключевые слова: уравнение эйконала, параллельный алгоритм, fast sweeping method, сейсморазведка, прямая кинематическая задача.

Введение

Решение прямой кинематической задачи расчета времен пробега сейсмических волн является важнейшим элементом во многих алгоритмах обработки сейсмических данных, таких как сейсмическая томография, миграция и др. Математически кинематика сейсмических волн описывается так называемым уравнением эйконала [1], которое является нелинейным уравнением в частных производных первого порядка. Заметим, что кроме сейсмики аналогичное уравнение возникает во многих других областях науки, таких как моделирование волн цунами, теория оптимального управления, компьютерное зрение, обработка цифровых изображений, задачи планирования маршрута, вычислительная биология и др. [2].

Для численного решения уравнения эйконала разработано много методов [3], которые можно отнести к двум основным категориям: использование метода характеристик (лучевое трассирование) и конечно-разностные подходы. Использование лучевого трассирования позволяет быстро решать прямую задачу расчета времен пробега для относительно простых слоистых моделей сред. Конечно-разностные подходы позволяют проводить моделирование для произвольно-неоднородных моделей. Однако нелинейность уравнения эйко-

[*] Исследование выполнено при финансовой поддержке стипендии Президента Российской Федерации для молодых ученых и аспирантов, осуществляющих перспективные научные исследования и разработки по приоритетным направлениям модернизации российской экономики № СП-2899.2015.5 и гранта РФФИ № 15-35-20932 мол_а_вед.

Никитин А. А., *Сердюков А. С.*, *Дучков А. А.* Параллельный алгоритм решения уравнения эйконала для трехмерных задач сейсморазведки // Вестн. Новосиб. гос. ун-та. Серия: Информационные технологии. 2015. Т. 13, вып. 3. С. 19–28.

нала приводит к возникновению неустойчивостей в ходе расчетов, поэтому требуется регуляризация численной схемы, т. е. использование так называемых «вязких» решений [4]. Эти решения позволяют найти только времена первых вступлений волн, что оказывается достаточным для многих задач по обработке сейсмических данных.

Следует отметить, что для сейсмических приложений характерны большие объемы данных полевых измерений. Для обработки таких данных необходимо проводить расчеты времен пробега для больших трехмерных скоростных моделей, причем проводить массовые повторные вычисления для плотной сети источников (например, для томографии) или сетки построения изображений (например, для миграции), см. [5]. Таким образом, актуальной задачей является не только разработка численных методов решения уравнения эйконала, но и их эффективная программная реализация, включая параллельные алгоритмы, оптимизированные для современных высокопроизводительных вычислительных платформ.

Начиная с первых работ по численному конечно-разностному решению уравнения эйконала [6; 7] было разработано несколько вариантов численных методов. В настоящий момент существуют два наиболее популярных и быстрых подхода. Первый из них основан на использовании алгоритма Дейкстры поиска минимальных путей в графах. Алгоритмом данного типа является схема Fast Marching Method (FMM) [8] и ее различные модификации [9; 10]. Второй подход, представленный схемой Fast Sweeping Method (FSM) [11] и ее модификациями (см., например, [12]), основан на итерациях с чередованием направления обхода сетки типа Гаусса-Зейделя (с фиксацией новых значений после каждой итерации). Рассмотрим каждый из двух подходов более подробно на примере FMM и FSM.

FMM является не итерационным алгоритмом, в котором используется динамическое определение порядка вычисления решения в узлах сетки (решение находится за один проход всей сетки). Значения времен обновляются в так называемой «узкой полосе» узлов сетки, вокруг перемещающегося фронта волны. На каждом шаге алгоритма точка с минимальным временем в «узкой полосе» фиксируется, а соседи соответствующего принятого узла добавляются к «узкой полосе», и в них производится локальное решение уравнения эйконала. Для быстрого поиска узла с минимальным временем пробега в «узкой полосе» применяется быстрый алгоритм сортировки на графах (алгоритм пирамидальной сортировки). Поэтому вычислительная сложность FMM составляет $O(N \log N)$, где N – количество узлов сетки. Нерегулярный обход узлов расчетной сетки, который зависит от сортировки времен для узлов в «узкой полосе», затрудняет эффективную параллельную реализацию данного алгоритма. Отметим, что даже в последовательной версии данного алгоритма, возникает проблема оптимального использования кэш-памяти.

FSM является итерационным алгоритмом, в котором на каждой итерации происходит смена чередующихся направлений обхода сетки (4 направления обхода в двухмерном случае, 8 – в трехмерном). Вычислительная сложность для одного обхода FSM составляет $O(N)$, где N – количество узлов сетки. Алгоритм сходится за конечное число итераций M, которое зависит от сложности рассматриваемой модели [11] (для однородной среды для сходимости достаточно одного обхода в каждом направлении). Таким образом, общая вычислительная сложность алгоритма составляет $O(M\,N)$ и зависит от сложности модели. FSM больше подходит для параллельной реализации.

В данной работе был проведен анализ и сравнение существующих подходов к распараллеливанию FSM [13; 14], что позволило предложить новый подход, оптимизированный для вычислительных систем с общей памятью за счет более эффективного использования кэш-памяти. В статье приведено описание нового параллельного алгоритма и результаты его тестирования.

Численный метод решения уравнения эйконала

Рассмотрим уравнение эйконала:

$$\left|\nabla t(x)\right| = f(x), \;\; x \in \backslash\Omega\Gamma \subset R^n, \tag{1}$$

с заданными краевыми условиями:

$$t(x) = g(x), \;\; x \in \partial\Gamma, \tag{2}$$

где t – неизвестная функция, описывающая время пробега волны в точку x, f – заданная положительная функция медленности (величина, обратная к скорости распространения волны) в точке x, Ω – расчетная область пространства R^n, Γ – подобласть Ω (точка или область вокруг сейсмического источника) с границей $\partial\Gamma$.

Наиболее распространенным подходом к дискретизации уравнения эйконала, применяемым в различных схемах первого порядка, является использование противопоточной конечно-разностной аппроксимации [15]:

$$\left[\left(t_{i,j}^{\text{new}} - t_{x_{\min}}\right)^+\right]^2 + \left[\left(t_{i,j}^{\text{new}} - t_{y_{\min}}\right)^+\right]^2 = f_{i,j}^2 h^2, \tag{3}$$

где h – шаг дискретизации сетки $i = 1..I$, $j = 1..J$,

$$\left(x\right)^+ = \begin{cases} x, x > 0, \\ 0, x \le 0, \end{cases}$$

$$t_{x_{\min}} = \begin{cases} t_{i,j-1}, & j = 1, \\ t_{i,j+1}, & j = J, \\ \min\left(t_{i,j-1}, t_{i,j+1}\right), & j \in [2, J-1], \end{cases}$$

$$t_{y_{\min}} = \begin{cases} t_{i+1,j}, & i = 1, \\ t_{i-1,j}, & i = I, \\ \min\left(t_{i-1,j}, t_{i+1,j}\right), & i \in [2, I-1]. \end{cases}$$

В целях простоты изложения приведены соотношения для двумерного случая. В трехмерном случае противопоточная аппроксимация имеет аналогичный вид.

Выражение (3) представляет собой квадратное алгебраическое уравнение относительно $t_{i,j}^{\text{new}}$ и используется для обновления значения времени пробега в соответствующем узле декартовой сетки. В начале вычислительного процесса в области источника Г решение задается аналитическим способом, а в области $\Omega\backslash\Gamma$ полагается равным бесконечности (на практике большим, чем максимально возможное значение t). Так как при дискретизации используется противопоточная аппроксимация, для вычисления нового значения в узле сетки требуются только соседние узлы с меньшими значениями. Это согласуется с принципом причинно-следственной связи, вытекающей из физического смысла уравнения эйконала (постепенное распространение волны, т.е. узлы сетки с большими значениями времен зависят от узлов сетки с меньшими значениями поля времен).

Использование аппроксимации (3), обеспечивает правильный порядок пересчета только на локальном уровне – в рамках конечно-разностного шаблона. Для глобального (во всей расчетной области) обеспечения причинной зависимости в рамках FSM используется итерационный алгоритм [11], в котором на каждой итерации происходит смена чередующихся направлений обхода сетки, соответствующих координатным осям с положительными или отрицательными направлениями вдоль каждой из координат. В двумерном случае получаются следующие четыре цикла: 1) $i = 1:I$, $j = 1:J$, 2) $i = I:1$, $j = 1:J$, 3) $i = I:1$, $j = J:1$, 4) $i = 1:I$, $j = J:1$. При обходе в каждом направлении обновляются те узлы сетки, в которых новое значение решения, вычисленное согласно (3), меньше предыдущего. Чередование направлений обхода позволяет итерационно уточнять времена прихода для областей, где направление движения волн меняется. Так, для однородной модели и источника в центре каждый обход позволяет правильно рассчитать поле для одного квадранта, а расчет сходится к точному решению за 4 обхода в двумерном случае и 8 обходов – в трехмерном. Соответствующие направления обхода показаны на рис. 1. Красными стрелками показаны различные направления обхода сетки (см. направление осей i, j на самом левом рисунке и подписи), зеленым цветом выделен источник, а серым цветом показан квадрант, в узлах которого обновляются значения времен пробега в соответствующую итерацию.

Рассмотрим первое направление обхода в двумерном случае (рис. 2, слева). Исходя из данного направления обхода и схемы (3), вычисления в узле с индексами (i, j) для заданного направления обхода могут быть проведены после вычислений в узлах с меньшими индексами (i, j). Таким образом, алгоритм FSM является существенно последовательным. В то же время узлы сетки в данном алгоритме можно легко разбить на множества «независимых» узлов, значения внутри которых могут быть вычислены независимо друг от друга, и распределить эти вычисления по потокам исполнения [14]. Примером такого разбиения являются диагонали для двумерного случая (разные множества «независимых» узлов показаны разными оттенками серого на рис. 2, в центре) и диагональные плоскости в трехмерном случае (см. рис. 2, справа). Заметим, что это разбиение не является единственным – возможны и другие множества «независимых» улов.

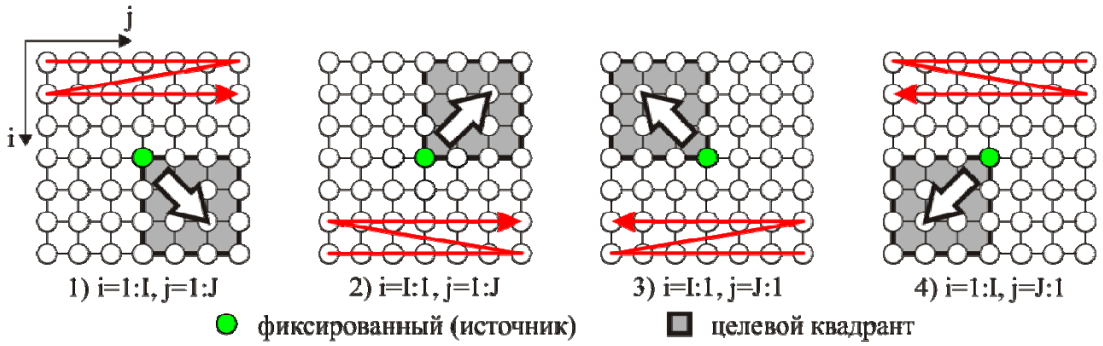

Рис. 1. Пример работы алгоритма Fast Sweeping Method для задачи с однородной скоростной моделью в двумерном случае

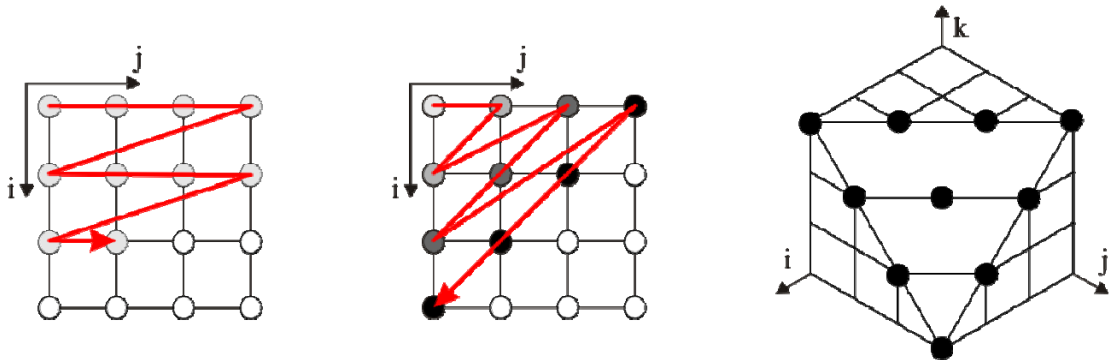

Рис. 2. Примеры обхода сетки для одного направления в алгоритме FSM (красными стрелками): слева – обход сетки из [11], в центре – обход сетки из [14] для двумерного случая. Множества узлов, допускающих независимое вычисление, показаны: в центре – оттенками серого для двумерного случая, справа – одна плоскость для трехмерного случая

Реализация FSM в системах с общей памятью

Нами был проведен анализ двух существующих параллельных реализаций алгоритма FSM, который показал их существенные ограничения. В работе [13] предлагается стратегия параллельной реализации алгоритма FSM, основанная на распределении разных направлений обхода сетки по различным потокам исполнения. После каждого полного обхода всех направлений производится синхронизация потоков с редукцией по минимальному времени для каждого узла сетки. При этой стратегии максимальное количество параллельных потоков соответствует количеству направлений обхода и составляет 4 для двухмерного случая и 8 для трехмерного. В этом случае также требуется хранить 4 или 8 копий массива модели, что затрудняет использование данного подхода при решении больших задач. Кроме того, примене-

ние WENO схем высокого порядка аппроксимации частных производных (см. [12]) не позволяет использовать минимизацию по старому/текущему значению, поэтому данная стратегия оказывается неподходящей при увеличении порядка точности схемы FSM. В связи с этими ограничениями, рассмотренная схема параллельной реализации не будет далее рассматриваться.

Вторая существующая параллельная реализация FSM (см. [14]) основана на параллельном вычислении новых значений для точек из множества «независимых» узлов, см. рис. 2. В этом случае масштабируемость задачи на количество потоков ограничена только лишь ее размерами (максимальным размером множества «независимых» узлов).

В рамках данной работы были реализованы последовательная и OpenMP версии параллельного алгоритма [14], обозначаемые далее как serial_Detrixhe и omp_Detrixhe, и проведено их сравнение с последовательной реализацией оригинального алгоритма FSM [11], обозначаемой как serial_Zhao. Результаты тестирования и сравнения этих реализаций приведены на рис. 3. Все тесты в работе выполнялись на кластере ИВЦ НГУ на вычислительных узлах с двумя 4-ядерными процессорами Intel Xeon E5540 с тактовой частотой 2 530 МГц на узел.

Как видно из результатов тестирования, представленных на рис. 3, реализация serial_Detrixhe оказывается значительно более медленной, чем реализация serial_Zhao, вследствие чего параллельная реализация omp_Detrixhe не позволяет достигнуть приемлемого ускорения по сравнению с serial_Zhao. Мы считаем, что в первую очередь это связано с менее эффективным использованием кэш-памяти процессора в реализациях serial_Detrixhe и omp_Detrixhe. Так, в реализации serial_Zhao (см. рис. 2, слева) последовательно обрабатываются элементы массива времен, лежащие последовательно в памяти, а в реализации serial_Detrixhe и omp_Detrixhe последовательно в потоке обрабатываются элементы, которые в памяти значительно (и не системно) сдвинуты друг относительно друга (см. рис. 2, в центре), что приводит к увеличению количества кэш-промахов.

Рис. 3. Результаты тестирования производительности FSM на однородной трехмерной модели в зависимости от размера сетки. Для OpenMP реализаций приведены результаты для 8 потоков

Для решения этой проблемы нами предлагается новая параллельная реализация FSM, основанная на декомпозиции сетки на подзадачи. Внутри подзадач вычисления производятся аналогично [11], что позволяет сохранять высокую пространственную и временную локальность обращений в память. При этом зависимости между подзадачами устанавливаются согласно информационным зависимостям в алгоритме [11], и порядок параллельного вычисления подзадач может быть реализован аналогично стратегии [14]. Далее мы будем предполагать, что при индексации (i, j, k) быстрым индексом является k, т. е. соседние по k элементы массива расположены последовательно в памяти.

Двумерная декомпозиция (*omp_new1*). Двумерная декомпозиция вычислений производится по индексам (i, j), как показано на рис. 4, слева. В нашей реализации мы рассматривали

разбиение на единичные строки (*i*, *j*). При этом каждый поток полностью выполняет вычисления для этих строк по быстрому индексу *k*, что обеспечивает доступ к элементам массива, расположенным последовательно в памяти (как в [11], рис. 2, слева). Сами блоки могут выполняться параллельно в соответствии с «диагональной» структурой (как в [14], рис. 2, в центре), как показано цифрами на блоках на рис. 4, слева. Между потоками осуществляется барьерная синхронизация после обработки каждой j-строки массива для поддержания зависимостей по данным как в [11]. Стоит заметить, что с ростом количества потоков эффективность данного подхода будет снижаться из-за необходимости синхронизации потоков. Данный алгоритм будем далее обозначать omp_new1.

Трехмерная декомпозиция (*omp_new2*). Нами была разработана еще одна параллельная реализация алгоритма на основе трехмерной декомпозиции для того, чтобы осуществлять синхронизацию между потоками после обработки меньших блоков данных (по третьему быстрому индексу). Схема трехмерной декомпозиции показана на рис. 4, справа. Сами блоки вычисляются параллельно в соответствии с «диагональной» структурой (как в [14], рис. 2, справа), как показано цифрами на блоках на рис. 4, справа. Поток выполняет вычисления для одного блока; между потоками осуществляется барьерная синхронизация перед переходом к следующему уровню «диагональной» структуры (цифра на блоках на рис. 4, справа). В нашей реализации при декомпозиции в плоскости (i.j) опять рассматривались единичные строки (в направлении индекса k). Размер блока по быстрому индексу k выбирался кратным размеру кэш-строки для того, чтобы избегать ложного разделения (false sharing) кэш-строк между процессорами в SMP системах. Данный алгоритм будем далее обозначать omp_new2.

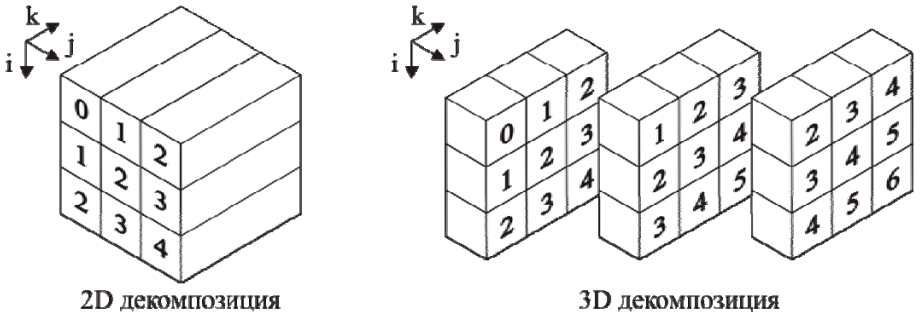

Рис. 4. Двумерная (слева) и трехмерная (справа) декомпозиция пространственной решетки на блоки алгоритма FSM. Цифры на блоках показывают порядок вычисления (блоки с одинаковыми номерами могут вычисляться параллельно)

Оптимизация и тестирование параллельного алгоритма

Было произведено тестирование производительности разработанных параллельных алгоритмов. Тестирование проводилось на кластере НГУ для однородной трехмерной модели с размером сетки 800^3. Ускорение параллельных реализаций вычислялось по отношению к последовательной реализации [11], которую мы обозначаем как serial_Zhao и которая идентична последовательной версии нашего алгоритма.

Вначале было проведено тестирование оптимального размера блока для алгоритма omp_new2 (трехмерная декомпозиция). Как было сказано выше, при трехмерной декомпозиции размер блока по быстрому индексу *k* выбирался кратным размеру кэш-строки. На рис. 5 показано ускорение, которое показывает алгоритм omp_new2 в зависимости пот размера блока (8 потоков для сетки 800^3). Наибольшее ускорение было получено для размера блока по быстрому индексу *k* 56 узлов; ускорение составило порядка 3.3, что соответствует эффективности 41 %. При малых размерах блоков (24 и менее) эффективность очень быстро падает. При увеличении размера блока эффективность начинает постепенно снижаться, что можно связать с увеличением времени синхронизации.

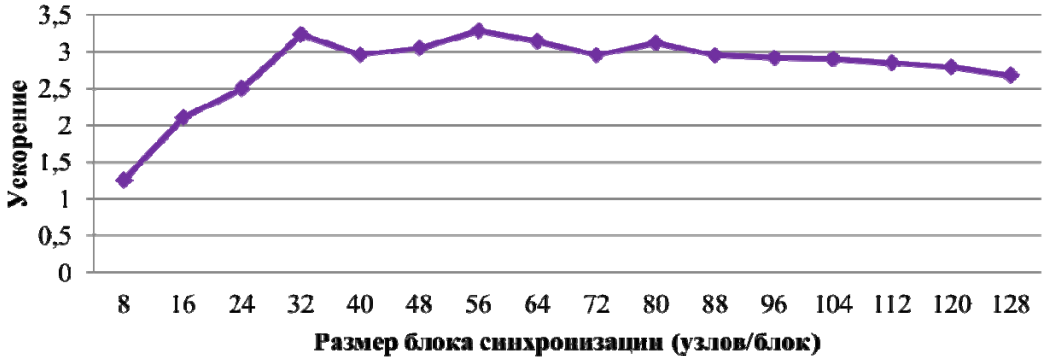

Рис. 5. Зависимость ускорения реализации omp_new2 от размера блока синхронизации на однородной скоростной модели с размером сетки 800^3 для 8 потоков

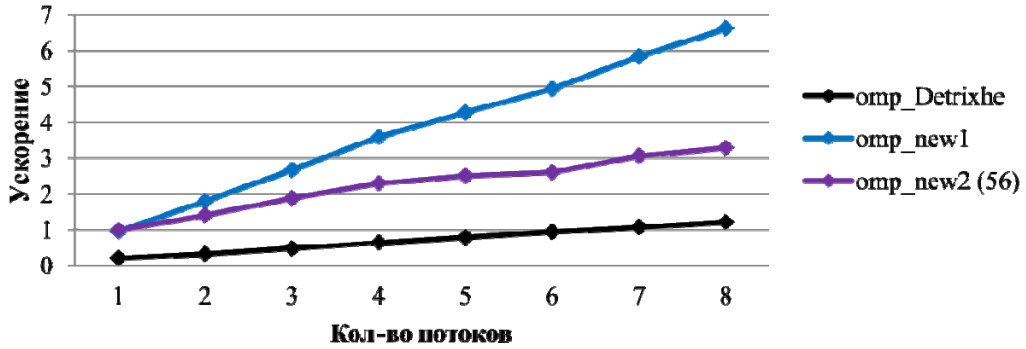

Рис. 6. Ускорение параллельных реализаций FSM по сравнению с последовательной версией (serial_Zhao) для сетки 800^3. Для реализации omp_new2 в скобках указан размер блока по быстрому индексу *k*

Ускорение для всех рассмотренных параллельных алгоритмов показано на рис. 6. На 8 потоках с размером сетки 800^3 для алгоритма omp_new1 (двумерная декомпозиция) было получено ускорение 6.6, что соответствует 82 % эффективности. Алгоритм omp_new2 (трехмерная декомпозиция) оказался значительно менее эффективным по сравнению с omp_new1. Видимо причиной этого является то, что увеличившиеся накладные расходы на более частую синхронизацию превысили выигрыш от меньшего времени ожидания. Самой медленной оказалась параллельная реализация omp_Detrixhe, которая показала ускорение только 1.2, что соответствует 15 % эффективности.

Таким образом, из рис. 3 и 6 видно, что из всех протестированных параллельных алгоритмов, наилучшую эффективность при использовании общей памяти показал алгоритм omp_new1 (двумерная декомпозиция сетки). Возможным направлением оптимизации данного алгоритма может быть использование SIMD расширений, доступных в большинстве современных процессоров. Для их эффективного использования необходимо разместить сетку в памяти в соответствии со структурой обращения алгоритма [14] (omp_Detrixhe). Однако это вызовет дополнительные накладные расходы, связанные с более сложным доступом к элементам массива и необходимостью реорганизации хранения сетки при смене направления обхода на противоположное.

На рис. 7 приведен пример расчета поля времен первых вступлений продольных волн для неоднородной трехмерной модели. Цветом показано распределение скорости продольных волн; черными контурами показаны линии уровня времен пробега волн (изохроны).

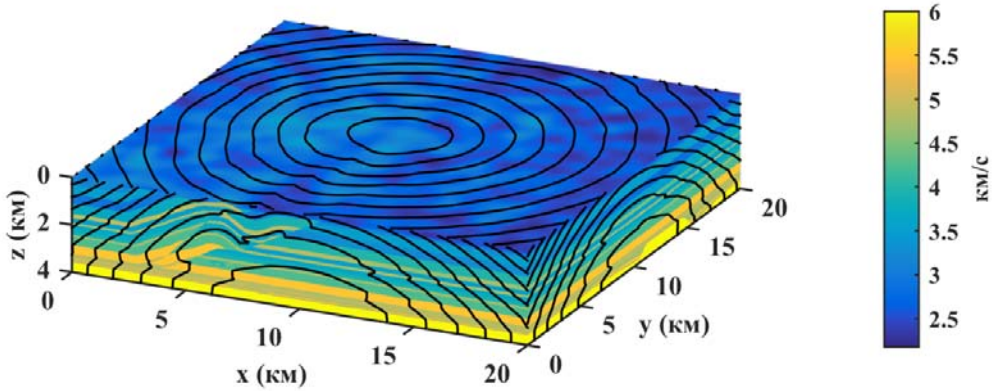

Рис. 7. Пример решения уравнения эйконала
Черными линиями показаны изохроны времен пробега продольных волн

Заключение

В работе был предложен новый параллельный алгоритм на основе метода FSM решения уравнения эйконала в трехмерном случае. При тестировании решения трехмерных задач в вычислительных системах с общей памятью новый алгоритм показал наиболее высокую эффективность работы по сравнению с существующими параллельными реализациями FSM. Наилучшие результаты были получены для модификации алгоритма с использованием двумерной декомпозиции расчетной области. Для нее было получено ускорение 6.6 при использовании 8 потоков, что соответствует 82 % эффективности. Основное преимущество предложенной параллельной реализации по сравнению с более ранними алгоритмами заключается в более эффективном использовании кэш-памяти процессора.

В дальнейшем планируется использование предложенной параллельной реализации для расчета времен пробега волн в сейсмических приложениях (задачи сейсмической томографии и построения миграционных изображений). Стоит заметить, что предложенная стратегия параллельной реализации FSM на основе декомпозиции может быть легко реализована и в вычислительных системах с распределенной памятью, и в дальнейшем мы планируем разработать параллельный алгоритм численного решения уравнения эйконала для гибридных кластеров, использующих GPU ускорители и сопроцессоры Intel Xeon Phi.

Список литературы

1. *Cerveny V.* Seismic ray theory. Cambridge University Press, 2005.
2. *Sethian J. A.* Level set methods and fast marching methods: evolving interfaces in computational geometry, fluid mechanics, computer vision, and materials science. Cambridge University Press, 1999. Vol. 3.
3. *Runborg O.* Mathematical models and numerical methods for high frequency waves // Communications in Computational Physics. 2007. Vol. 2, № 5. P. 827–880.
4. *Crandall M. G.*, *Lions P. L.* Viscosity solutions of Hamilton-Jacobi equations // Transactions of the American Mathematical Society. 1983. Vol. 277, № 1. P. 1–42.
5. *Yilmaz Ö.* Seismic data analysis. Society of Exploration Geophysicists, 2001.
6. *Vidale J.* Finite-difference calculation of travel times // Bulletin of the Seismological Society of America. 1988. Vol. 78, № 6. P. 2062–2076.
7. *Podvin P.*, *Lecomte I.* Finite difference computation of traveltimes in very contrasted velocity models: a massively parallel approach and its associated tools // Geophysical Journal International. 1991. Vol. 105, № 1. P. 271–284.
8. *Sethian J. A.* A fast marching level set method for monotonically advancing fronts // Proceedings of the National Academy of Sciences. 1996. Vol. 93, № 4. P. 1591–1595.

9. *Hassouna S. M.*, *Farag A. A.* Multistencils fast marching methods: A highly accurate solution to the eikonal equation on cartesian domains // IEEE Transactions on Pattern Analysis and Machine Intelligence. 2007. Vol. 29, № 9. P. 1563–1574.

10. *Kim S.* An *O*(*N*) Level Set Method for Eikonal Equations // SIAM Journal on Scientific Computing. 2001. Vol. 22, № 6. P. 2178–2193.

11. *Zhao H.* A fast sweeping method for eikonal equations // Mathematics of computation. 2005. Vol. 74, № 250. P. 603–627.

12. *Zhang Y. T.*, *Zhao H. K.*, *Qian J.* High order fast sweeping methods for static Hamilton – Jacobi equations // Journal of Scientific Computing. 2006. Vol. 29, № 1. P. 25–56.

13. *Zhao H.* Parallel implementations of the fast sweeping method // Journal of Computational Mathematics. 2007. Vol. 25, № 4. P. 421–429.

14. *Detrixhe M.*, *Gibou F.*, *Min C.* A parallel fast sweeping method for the Eikonal equation // Journal of Computational Physics. 2013. Vol. 237. P. 46–55.

15. *Rouy E.*, *Tourin A.* A viscosity solutions approach to shape-from-shading // SIAM Journal on Numerical Analysis. 1992. Vol. 29, № 3. P. 867–884.

Материал поступил в редколлегию 08.08.2015

A. A Nikitin, **A. S. Serdyukov**, **A. A. Duchkov**

Trofimuk Institute of Petroleum Geology and geophysics SB RAS
3 Koptug ave., *630090*, *Novosibirsk*, *Russia*

Novosibirsk State University
2 Pirogov Str., *630090*, *Novosibirsk*, *Russia*

NikitinAA@ipgg.sbras.ru, *AleksanderSerdyukov@yandex.ru*
DuchkovAA@ipgg.sbras.ru

PARALLEL ALGORITHM OF THREE-DIMENSIONAL EIKONAL SOLVER FOR SEISMIC APPLICATIONS

Solution to the eikonal equation is used in seismic problems to determine first arrival travel times of seismic waves. In this paper we present a new parallel algorithm of the eikonal equation solution based on the Fast Sweeping Method (FSM). The algorithm achieves higher efficiency compared to existing parallel implementations of FSM due to optimization of CPU cache use.

Keywords: eikonal equation, parallel algorithm, fast sweeping method, exploration seismology, forward traveltime problem.

References

1. Cerveny V. *Seismic ray theory*. Cambridge university press, 2005.

2. Sethian J. A. *Level set methods and fast marching methods: evolving interfaces in computational geometry*, *fluid mechanics*, *computer vision*, *and materials science*. Cambridge university press, 1999. vol. 3.

3. Runborg O. Mathematical models and numerical methods for high frequency waves. *Communications in Computational Physics*. 2007, vol. 2, iss. 5, p. 827–880.

4. Crandall M. G., Lions P. L. Viscosity solutions of Hamilton-Jacobi equations. *Transactions of the American Mathematical Society*. 1983, vol. 277, iss. 1, p. 1–42.

5. Yilmaz Ö. *Seismic data analysis*. Society of Exploration Geophysicists, 2001.

6. Vidale J. Finite-difference calculation of travel times. *Bulletin of the Seismological Society of America*. 1988, vol. 78, iss. 6, p. 2062–2076.

7. Podvin P., Lecomte I. Finite difference computation of traveltimes in very contrasted velocity models: a massively parallel approach and its associated tools. *Geophysical Journal International*. 1991, vol. 105, iss. 1, p. 271–284.

8. Sethian J. A. A fast marching level set method for monotonically advancing fronts. *Proceedings of the National Academy of Sciences*. 1996, vol. 93, iss. 4, p. 1591–1595.

9. Hassouna S. M., Farag A. A. Multistencils fast marching methods: A highly accurate solution to the eikonal equation on cartesian domains. *IEEE Transactions on Pattern Analysis and Machine Intelligence*. 2007, vol. 29, iss. 9, p. 1563–1574.

10. Kim S. An O(N) Level Set Method for Eikonal Equations. *SIAM Journal on Scientific Computing*. 2001, vol. 22, iss. 6, p. 2178–2193.

11. Zhao H. A fast sweeping method for eikonal equations. *Mathematics of computation*. 2005, vol. 74, p. 250. C. 603–627.

12. Zhang Y. T., Zhao H. K., Qian J. High order fast sweeping methods for static Hamilton – Jacobi equations. *Journal of Scientific Computing*. 2006, vol. 29, iss. 1, p. 25–56.

13. Zhao H. Parallel implementations of the fast sweeping method. *Journal of Computational Mathematics*. 2007, vol. 25, iss. 4, p. 421–429.

14. Detrixhe M., Gibou F., Min C. A parallel fast sweeping method for the Eikonal equation. *Journal of Computational Physics*. 2013, vol. 237, p. 46–55.

15. Rouy E., Tourin A. A viscosity solutions approach to shape-from-shading. *SIAM Journal on Numerical Analysis*. 1992, vol. 29, iss. 3, p. 867–884.