Массивно-параллельный расчет неустойчивости Рэлея — Тейлора с помощью аналитического выражения функции Грина соответствующей краевой задачи

Т. В. Абрамов

Институт нефтегазовой геологии и геофизики СО РАН, Новосибирск, Россия
Новосибирский государственный университет, Россия
Контактный e-mail: AbramovTV@ipgg.sbras.ru

Представлены модификация для трехмерного случая и программная реализация алгоритма расчета неустойчивости Рэлея — Тейлора в высоковязкой несжимаемой ньютоновской жидкости. Алгоритм основан на использовании аналитического выражения функции Грина соответствующей краевой задачи, что значительно ускоряет расчет, особенно при использовании параллельных вычислений. Разработанное программное обеспечение может работать как на ресурсах центральных процессоров (CPU) с использованием инструкций SSE и протокола MPI, так и на графических ускорителях (GPU) NVIDIA в случае доступности узлов гибридного кластера. Взаимодействие с GPU реализовано с помощью технологий CUDA и MPI.

Работа ориентирована на моделирование геологического процесса — соляного диапиризма как частного случая неустойчивости Рэлея — Тейлора, выражающегося во всплывании твердой каменной соли через более плотные перекрывающие осадки в геологическом масштабе времени.

Ключевые слова: NVIDIA CUDA, параллельное программирование, ползущее течение, функция Грина, неустойчивость Рэлея — Тейлора, соляной диапиризм.

Введение

Численное решение сложных гидродинамических задач с помощью разностных методов требует значительных вычислительных ресурсов. Основная сложность при этом состоит в решении больших систем линейных алгебраических уравнений (СЛАУ). Для этого существует множество программных библиотек, тем не менее эта задача до сих пор остается очень ресурсоемкой и неудобной для распараллеливания — скорость расчета на параллельных вычислительных машинах оказывается ограниченной пропускной способностью памяти, а не пиковой производительностью системы [1]. Зависимость производительности от количества вычислительных узлов существенно нелинейна и довольно быстро выходит на практический предел прироста скорости вычислений [2].

Этих трудностей можно избежать в тех редких случаях, когда известно точное решение соответствующей краевой задачи. Одним из таких случаев является задача о действии силы тяжести на плотностные неоднородности вязкого полупространства со свободной поверхностью. Для используемой в настоящей работе формулировки этой задачи в [3] найдена аналитически функция Грина, что позволяет получать решение задачи как свертку этой функции с правой частью уравнения, в данном случае с возмущением плотности в полупространстве.

Прямое вычисление свертки значительно быстрее и проще в реализации, чем решение СЛАУ. Этот алгоритм эффективно распараллеливается практически на любое количество вычислительных узлов и любую архитектуру, при этом лимитирующим фактором будет именно пиковая производительность системы, а не быстродействие памяти. Его легко реализовать на центральных процессорах и оптимизировать с помощью векторных инструкций MMX, SSE или AVX, но более целесообразно использовать возможности графических ускорителей (GPU), которые благодаря массивному параллелизму имеют значительно бóльшую пиковую производительность, чем CPU.

Потребность в высокой скорости вычислений при решении этой задачи обусловлена ее практическим применением. В работе моделируются процессы соляного диапиризма как частного случая неустойчивости Рэлея — Тейлора, выражающегося во всплывании твердой каменной соли в более плотных осадках в геологическом масштабе времени (обычно десятки миллионов лет). Чтобы достоверно восстановить эволюцию реальных структур по их известному конечному состоянию, необходимо решать множество прямых задач моделирования с различными исходными параметрами. Актуальность исследований с практической точки зрения определяется тем, что этот процесс в значительной мере контролирует размещение залежей углеводородов практически во всех крупных нефтегазоносных провинциях [4].

1. Постановка задачи

Формально решается задача о происходящем под действием силы тяжести ползущем течении множества несмешивающихся несжимаемых ньютоновских жидкостей с различной плотностью и однородной, но очень высокой вязкостью ($\sim 10^{20}$ Па $\cdot$ с), занимающих ограниченное свободной поверхностью полупространство. Такая постановка задачи для описания процесса соляного диапиризма подробно рассмотрена в [5, 6], там же обоснована возможность использования однородной вязкости.

На рис. 1 схематично показано полупространство в декартовой системе координат x_1, x_2, x_3, заполненное жидкостями $W_1, \ldots, W_4$, со своими плотностями $\rho_1, \ldots, \rho_4$ и единой вязкостью μ. Они разделены границами $S_1, \ldots, S_3$, F — свободная поверхность.

Распределение плотности в полупространстве в момент времени t однозначно задается актуальной конфигурацией границ S_i: $\rho_{(\mathbf{x},t)} = \rho_i$ для $\mathbf{x} \in W_i$. Плотность ρ, тензор напряжений $\mathbf{T}$ и давление P представляются в виде

$$\begin{aligned} \rho_{(\mathbf{x},t)} &= \rho^0_{(x_3,t)} + \sigma_{(\mathbf{x},t)}, \\ \mathbf{T}_{(\mathbf{x},t)} &= \mathbf{T}^0_{(x_3,t)} + \boldsymbol{\tau}_{(\mathbf{x},t)}, \\ P_{(\mathbf{x},t)} &= P^0_{(x_3,t)} + p_{(\mathbf{x},t)}, \end{aligned} \tag{1}$$

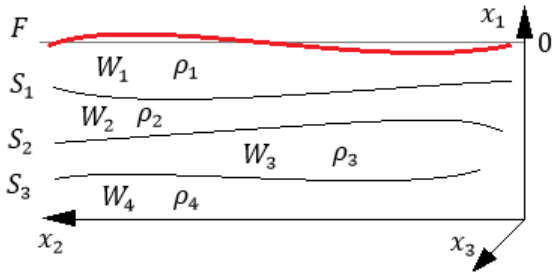

Рис. 1. Полупространство, занятое жидкостью

где $\rho^0_{(x_3,t)}$, $\mathbf{T}^0_{(x_3,t)}$, $P^0_{(x_3,t)}$ характеризуют гидростатическое состояние ($\mathbf{T}^0_{(x_3,t)} = -\delta_{ij} P^0_{(x_3,t)} = -\delta_{ij} g \int\limits_0^{x_3} \rho^0_{(x_3,t)} dx_3$, δ_{ij} — дельта Кронкера), а $\sigma_{(\mathbf{x},t)}$, $\boldsymbol{\tau}_{(\mathbf{x},t)}$, $p_{(\mathbf{x},t)}$ — отклонения от него.

Для гидростатического состояния скорость течения $\mathbf{v}^0 \equiv 0$, так что течение $\mathbf{v}$ связано только с возмущениями. Как показано в [3, 5, 6], задача вычисления ползущего течения в весомом полупространстве со свободной границей может быть сформулирована относительно возмущений с разделением на квазистационарную и эволюционную части в виде, линеаризованном по граничным условиям.

Квазистационарная часть

$$\begin{gathered} \mu \boldsymbol{\nabla}^2 \mathbf{v} - \boldsymbol{\nabla} p = -\sigma g, \\ \boldsymbol{\nabla} \cdot \mathbf{v} = 0, \\ (v_3 = \tau_{31} = \tau_{32} = 0) \mid_{x_3=0} \end{gathered} \tag{2}$$

с дополнительным условием для определения $\zeta(x_1, x_2)$ — величины возмущения свободной поверхности $F = x_3 - \zeta(x_1, x_2) = 0$:

$$(\tau_{33})_{x_3=0} = -\rho^0 g \zeta. \tag{3}$$

Эволюционная часть

$$\frac{\partial S_i}{\partial t} + \mathbf{v} \cdot \boldsymbol{\nabla} S_i = 0. \tag{4}$$

Расчет ползущего течения в каждый момент времени $t^{(k)}$ происходит в два этапа: сначала отыскивается поле течения $\mathbf{v}^{(k)}$, которое задается распределением $\rho_{(\boldsymbol{x},t_k)}$, т. е. конфигурацией границ $S_i^{(k)}$. На втором этапе решается эволюционное уравнение, в результате чего границы перемещаются полем $\mathbf{v}^{(k)}$ в новое положение $S_i^{(k+1)}$. Новая конфигурация границ задает новое распределение плотности $\rho_{(\mathbf{x},t_{k+1})}$, из которого рассчитывается новое поле $\mathbf{v}^{(k+1)}$. Таким образом, эволюция описывается набором квазистационарных состояний, связанных эволюционным уравнением.

Для задачи (1) в [3] найдено аналитическое выражение функции Грина, что позволяет отыскивать поле течения как ее свертку с распределением плотности:

$$\mathbf{v}(\mathbf{x}) = \iiint \mathbf{V}(\mathbf{x}, \boldsymbol{\xi}) \sigma(\boldsymbol{\xi}) d\xi_1 d\xi_2 d\xi_3, \tag{5}$$

где $\mathbf{V}$ — функция Грина, а $\mathbf{x}$, $\boldsymbol{\xi}$ — декартовы координаты точек пространства.

Такой подход имеет множество преимуществ по сравнению с использованием разностных схем. Во-первых, поскольку это точное решение задачи, повышается точность расчета. Во-вторых, по формуле (5) можно рассчитывать $\mathbf{v}$ в любой точке полупространства, независимо от других. Это понижает размерность задачи, так как для решения эволюционного уравнения (4) необходимо вычислять скорость только на границах S_i. Но основным преимуществом является скорость расчета из-за меньшей алгоритмической сложности ($O(N^2)$ вместо $O(N^3)$ [2]) и чрезвычайной эффективности распараллеливания по аналогии с задачей N-тел. Параллельная реализация позволяет достигать производительности, близкой к пиковой, в том числе на GPU [8].

Этот метод был реализован в двухмерной постановке [5] применительно к моделированию процессов соляного диапиризма, где он показал огромные преимущества в скорости работы даже при однопоточной реализации на CPU. Это позволило авторам решать обратные задачи — восстановление истории развития геологических структур (в 2D) по конечному состоянию, известному в настоящее время, путем перебора решений прямых задач с различными начальными условиями.

Реальные геологические структуры, как правило, существенно трехмерные объекты, поэтому в настоящей работе представлена трехмерная реализация метода. Это увеличивает вычислительную сложность и требует алгоритмов и структур данных, более сложных, чем для двумерной задачи. В частности, для вычисления свертки целесообразно использовать параллельные вычисления. Выделим подзадачи, которые требуется решить:

1) создание структур данных и алгоритмов для работы с границами, которые в трехмерной постановке представляют собой поверхности (это, возможно, наиболее сложная часть работы с алгоритмической точки зрения);
2) получение дискретного распределения плотности $\rho_{(\mathbf{x})}$ из границ S_i;
3) вычисление свертки (5), что обычно является самой затратной операцией по времени.

Далее в статье кратко рассматриваются эти подзадачи, их реализация и перспективы дальнейшей оптимизации.

2. Границы разделов сред

В трехмерной постановке границы S_i представляют собой поверхности. Они однозначно задают состояние модели, т. е. распределение плотности в выбранном участке полупространства. В процессе расчета границы перемещаются полем $\mathbf{v}$ и деформируются, каждый раз задавая новое распределение плотности. На рис. 2 показано, как слой легкой жидкости всплывает через более тяжелую, образуя грибообразную структуру.

Заметно, что эти деформации значительно усложняют форму поверхностей, увеличивают площадь, создают “опрокидывания” на поздних стадиях эволюции. Очевидно, что необходимо увеличивать детализацию описания поверхностей по мере увеличения степени деформации — это первое и основное требование. Вместе с тем стоит задача минимизации числа точек, описывающих поверхность, так как поле $\mathbf{v}$ рассчитывается именно в них.

В текущей программной реализации триангулированная поверхность хранится в виде линейных массивов вершин, ребер и треугольников. Вершины заданы тремя декартовыми координатами $\mathbf{p}_i = (x_1, x_2, x_3)$, ребра $\mathbf{rb}_i = (p_1, p_2, tr_1, tr_2)$ содержат индексы

двух точек из массива вершин — p_1 и p_2 и индексы двух треугольников tr_1 и tr_2, а треугольники ссылаются на три ребра из массива ребер — $\mathbf{tr}_i = (rb_1, rb_2, rb_3)$.

Как правило, каждая вершина используется несколькими смежными ребрами, а каждое ребро — несколькими треугольниками (одним или двумя). Поэтому была применена известная в компьютерной графике технология индекс-буферов, т. е. ребра ссылаются только на вершины, а треугольники — на ребра. Сами структуры при этом хранятся отдельно, в виде массивов, что позволяет избежать их дублирования.

Дополнительная информация в виде ребер помогает быстро находить смежные треугольники и соседние точки без необходимости поиска во всем массиве объектов. Это нужно для быстрого увеличения детализации поверхности и производится путем деления длинных ребер пополам с добавлением новой точки. На рис. 3 схематично показан этот процесс.

В массиве ребер ищется то, длина которого больше некоторого заданного значения $l_{\max}$ (на рис. 3 ребро BD). Это ребро делится на два добавлением новой точки. Точка лежит на ребре, поэтому форма поверхности при этом никак не меняется, но в дальнейшем, при движении полем $\mathbf{v}$, она будет иметь свой вектор скорости и свою траекторию.

Выбранные структуры данных позволяют быстро, со сложностью $O(1)$ получить два треугольника, примыкающих к выбранному ребру, и все ребра и вершины, участвующие в делении. В результате получаются два дополнительных треугольника, одно ребро и одна новая точка, переопределяются ссылки. Важно, что операция локальна и алгоритм распараллелен.

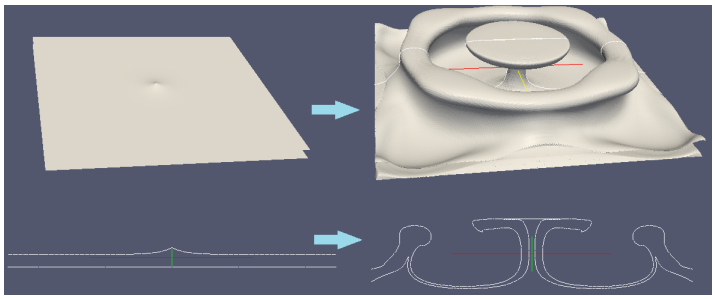

Рис. 2. Процесс всплывания, инициированный точечным возмущением: слева — начальное состояние и поперечный разрез через ось симметрии, справа — конечное состояние и его разрез

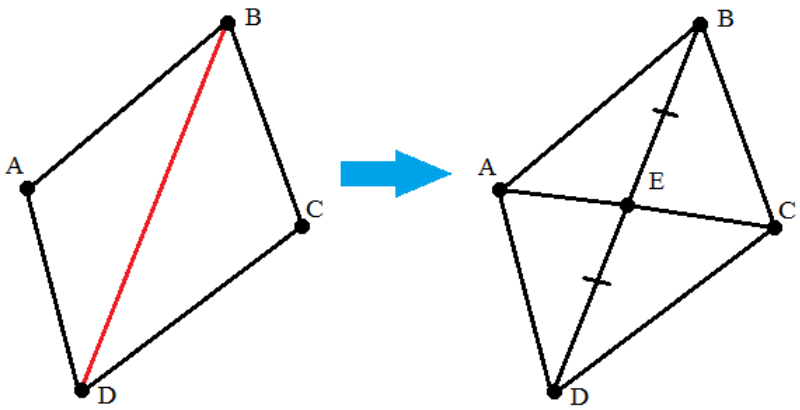

Рис. 3. Схема деления длинного ребра с добавлением новой точки

Этим достигается выполнение первого и основного требования, без которого нельзя было бы рассчитывать эволюцию до поздних стадий. Реализованный подход позволяет пока только увеличивать детальность. Было бы желательно и уменьшать ее там, где в процессе расчетов сетка получается излишне густой, но удовлетворительного решения этой задачи пока не найдено.

3. Дискретизация поля плотности

Поверхности однозначно задают распределение плотности в выбранном участке полупространства, но для численного расчета нужно иметь его в дискретном виде, желательно по регулярной сетке.

Важное требование к дискретизации — учет даже очень малых изменений положения границ, особенно в вертикальном направлении. Для задачи это имеет принципиальное значение, поскольку малые возмущения неустойчивого слоя жидкости вызывают ее движение, амплитуда возмущений увеличивается и возрастает скорость течения, т. е. имеет место положительная обратная связь.

Дискретизация происходит в два этапа. На первом ищется пересечение поверхностей S_i с множеством вертикальных линий, отстоящих друг от друга на шаг сетки h (рис. 4, *а*). В результате получаем набор отрезков $\{(0;h_1),(h_1;h_2),(h_2;h_3),...\}$, каждый из которых интерпретируется как вертикальный столбик какого-либо вещества W_i. На выходе первого этапа имеем поле плотности, дискретное по x_1, x_2 и непрерывное по x_3 (рис. 4, *б*).

На втором этапе отрезки $\{(0;h_1),(h_1;h_2),(h_2;h_3),...\}$ преобразуются в узлы сетки с шагом h (рис. 4, *в*). Эти узлы “захватывают” интервалы $\{(0;h),(h;2h),(2h;3h),...\}$. Если в какой-либо интервал $(nh;(n+1)h)$ попадает более одного вещества, то соответствующему узлу приписывается взвешенное среднее значение плотности. Это позволяет учитывать даже самые малые отклонения границы в вертикальном направлении без дополнительного сгущения сетки.

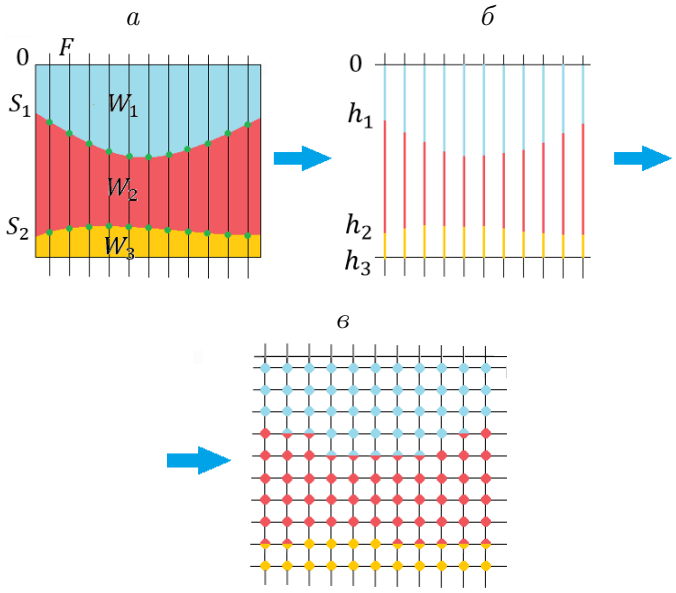

Рис. 4. Двухэтапный процесс дискретизации поля плотности

4. Вычисление свертки

В дискретной постановке интеграл (5) преобразуется в сумму вида

$$\mathbf{v}(\mathbf{x}) = \sum \mathbf{V}(\mathbf{x}, \boldsymbol{\xi})\sigma(\boldsymbol{\xi})\Delta\xi_1\Delta\xi_2\Delta\xi_3 = \sum \mathbf{V}(\mathbf{x}, \boldsymbol{\xi})m^a(\boldsymbol{\xi}), \tag{6}$$

где $m^a(\boldsymbol{\xi})$ — аномальная масса, заключенная в объеме $\Delta\xi_1\Delta\xi_2\Delta\xi_3$ с центром в точке $\boldsymbol{\xi}$. Имея регулярную сетку аномальной плотности с шагом h, из нее легко получить аномальные массы домножением каждого элемента на h^3.

Для решения эволюционного уравнения (4) нужно вычислять поле $\mathbf{v}$ только в точках границ, которые будем называть "приемниками". Формально приемник — это вектор $\mathbf{r} = (\mathbf{v}, \mathbf{x})$, содержащий точку поверхности S_i и значение вектора скорости в ней. Все множество приемников обозначим $\mathbf{R} = \{\mathbf{r}_j\},\ j = 1..M$.

Согласно формуле (6) мы считаем поле течения как сумму полей от каждого элементарного объема, обладающего аномальной плотностью (массой). Их будем называть "источниками" и обозначать $\mathbf{s} = (m_a, \boldsymbol{\xi})$, а все их множество — $\mathbf{S} = \{\mathbf{s}_i\},\ i = 1..N$.

Задача состоит в том, чтобы рассчитать действия всех источников на все приемники аналогично вычислению силового поля в задаче N-тел [8 – 10], что позволяет использовать существующие алгоритмы:

1) прямые методы (рассчитываются все пары взаимодействий источник — приемник, сложность оценивается как $O(N^2)$, где N — количество тел [8]);
2) методы, использующие теорему о свертке [11];
3) методы, использующие древовидную организацию источников [9, 10];
4) быстрый метод мультиполей [12, 13].

Четвертый метод имеет наименьшую оценку сложности — $O(N)$. Согласно ему сумму вида (6) можно легко вычислить за $O(N)$ операций, если $\mathbf{V}(\mathbf{x}, \boldsymbol{\xi})m^a(\boldsymbol{\xi})$ представима в виде $C(\mathbf{x})F(\boldsymbol{\xi})$ или в общем случае как $\sum\limits_{i=1}^{K} C_i(\mathbf{x})F_i(\boldsymbol{\xi})$. Как правило, это означает разложение $\mathbf{V}(\mathbf{x}, \boldsymbol{\xi})$ в ряд и выбор нескольких первых членов. Основную трудность здесь составляет неравномерная сходимость рядов, в связи с чем вместо одного глобального разложения используется множество локальных [13].

В нашем случае основная сложность здесь состоит в разложении обратного расстояния $1/R$, например, с помощью полиномов Лежандра, как в [14]. Численные эксперименты показали, что для обеспечения требуемой точности необходимо разбивать пространство на большое число подобластей, в которых обеспечивается хорошая сходимость. При такой реализации быстрый метод мультиполей является приближенным и разделение на подобласти повышает вычислительную сложность. Поэтому далее он не использовался.

Третий метод основан на том, что формой и размерами источника можно пренебречь, когда он достаточно далеко от точки приемника, т. е. можно группировать источники, отстоящие далеко от приемника. В алгоритме Барнса — Хатта (Barnes — Hut) в задаче N-тел [9] для этого используется представление пространства в виде иерархического октодерева. Сложность метода оценивается как $O(N \log(N))$.

В задаче N-тел поле силы обратно пропорционально квадрату расстояния во всех направлениях, а в нашем случае поле течения в направлении x_3 затухает медленней — обратно пропорционально первой степени от расстояния. Поэтому второй подход применим для нашей задачи с меньшей эффективностью, так как группировать "источники"

по x_3 приходится со значительно меньшим шагом. Вычислительные эксперименты показали, что на используемых в оперативных расчетах сетках (порядка $100 \times 100 \times 100$) этот подход не дает существенного выигрыша на GPU по сравнению с прямым расчетом, поэтому он также не развивался в этой работе.

Метод, основанный на теореме о свертке, выглядит наиболее перспективным по производительности и, вероятно, будет реализован в дальнейшем.

В настоящее время реализован и отлажен прямой расчет свертки как на CPU с использованием SSE и MPI, так и на GPU (на ПК с видеокартой, одном узле гибридного кластера и на нескольких узлах кластера с использованием CUDA и MPI). Перечислим основные преимущества метода.

1. Простота реализации. Прямое вычисление суммы (6) не представляет большой алгоритмической сложности даже на параллельных архитектурах. Алгоритм может быть реализован без условных операторов, работа с памятью также легко оптимизируется как на GPU, так и на CPU.
2. Масштабируемость. Алгоритм может быть эффективно распараллелен практически на любое количество вычислительных узлов без явных ограничений. Рост производительности по мере наращивания вычислительных мощностей близок линейному.
3. Существование аналога. В задаче N-тел поле ускорения рассчитывается аналогично (6). Различия состоят только в расчетной формуле, но в обоих случаях приходится вычислять величину $1/R$ и обходить особенности, когда $R \approx 0$.

Кроме того, важно, что в нашем случае приемники — точки поверхностей, т. е. рассчитывается влияние трехмерного объекта (распределения плотности) на двухмерные (поверхности). Поэтому даже при использовании прямого метода сложность значительно меньше, чем для классической задачи N-тел — $O(N^{1.67})$ вместо $O(N^2)$. В любом случае это значительно меньше, чем $O(N^3)$ для решения СЛАУ, которые возникают при решении задачи разностными методами [2].

Расчет свертки (6) реализован на GPU по прямой аналогии с описанным в [8] алгоритмом для N-тел с использованием разделяемой памяти для кэширования “источников”. Реализация для CPU сделана аналогично с [15] с векторными инструкциями SSE. Разделение работы по вычислительным узлам с помощью MPI не представило больших затруднений.

Созданная программа расчета ползущего течения тестировалась на различных архитектурах — ядрах CPU с помощью технологии MPI, графических ускорителях Tesla M2090 гибридного кластера НГУ, персональном компьютере с Tesla C1060. На рис. 5 показан график зависимости скорости расчета от количества вычислительных узлов кластера. Скорость расчета на графике — величина, обратная времени, размерности с^{-1}.

Вид этих зависимостей очень близок к линейному даже с учетом того, что для передачи данных использовалась технология MPI. Это показывает, что метод расчета нечувствителен к пропускной способности памяти и скорость расчета в этом случае может быть увеличена добавлением новых вычислительных узлов. Явных ограничений на их количество нет, что позволяет эффективно загружать вычислительный кластер. В расчетах использовалась одинарная точность, что особенно критично для GPU, так как для них производительность с одинарной точностью значительно выше, чем для двойной.

Сравнение производительности CPU и GPU показывает, что одна Tesla M2090 работает приблизительно с той же скоростью, что и три центральных процессора Intel

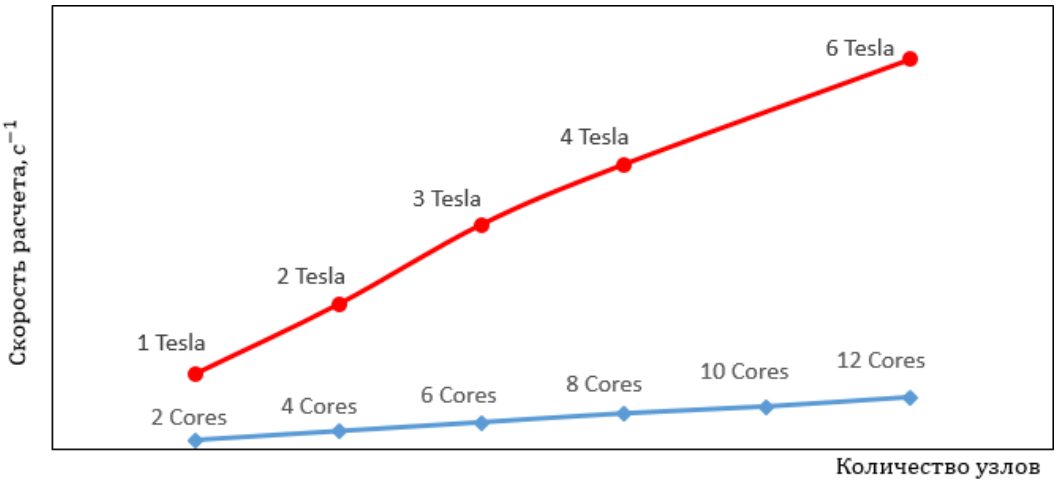

Рис. 5. Зависимость скорости расчета от количества вычислительных узлов

Xeon X5670 (18 ядер). С одной стороны, это подтверждает целесообразность использования GPU по соотношению цена — производительность. С другой, эта разница не так велика, а использование новых технологий AVX или AVX-512 вместо SSE позволило бы увеличить скорость расчета на CPU в два или четыре раза. Эффективная реализация алгоритмов позволила добиться максимума производительности от обеих архитектур.

5. Результаты моделирования

В данной работе моделирование неустойчивости Рэлея — Тейлора используется для описания процессов соляного диапиризма. Он изучался ранее методами как математического [2, 7], так и физического [4] моделирования. На рис. 6 показаны в сравнении результаты работы созданной программы (*а*) и результаты физического моделирования (*б*) из [4]. Они имеют очевидное качественное сходство. Для точного воспроизве-

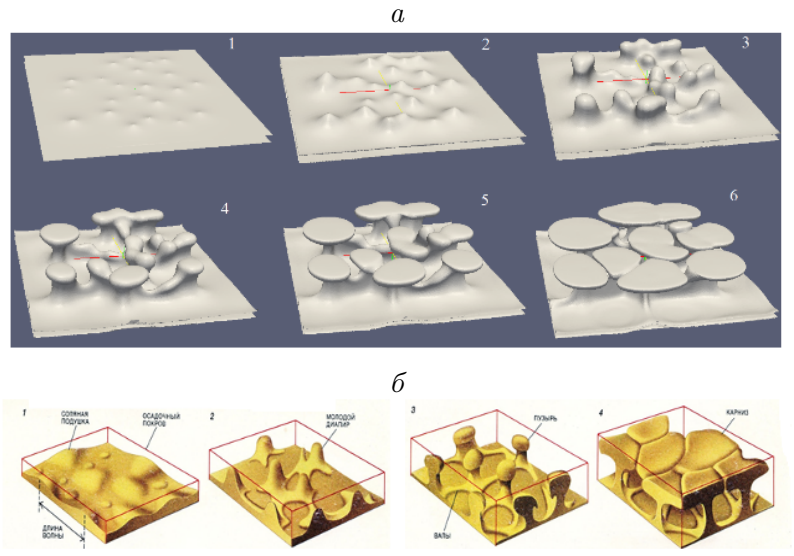

Рис. 6. Результаты моделирования: получены в данной работе (*а*), результаты из [4] (*б*)

дения эволюции (рис. 6, *б*) потребовалось бы задание идентичных начальных возмущений неустойчивого слоя и других параметров модели, но в данной работе этот эксперимент не проводился. Расчет (рис. 6, *а*) производился с помощью одного устройства Tesla C1060 в течение приблизительно одного часа.

Вычислительные эксперименты показали, что подробность описания поверхности и разрядность сетки плотности часто могут быть уменьшены без потери качественного и даже количественного сходства. На рис. 7 показаны два результата моделирования — с высокой детализацией и низкой. Очевидно сходство не только в целом, но даже в мелких деталях, что говорит об устойчивости метода и реализации. При этом низкая и высокая детализация использовалась на протяжении всего процесса эволюции. Время расчета на трех Tesla M2090 в случае (*а*) составило 20.09 мин, а в (*б*) — 4.97 мин.

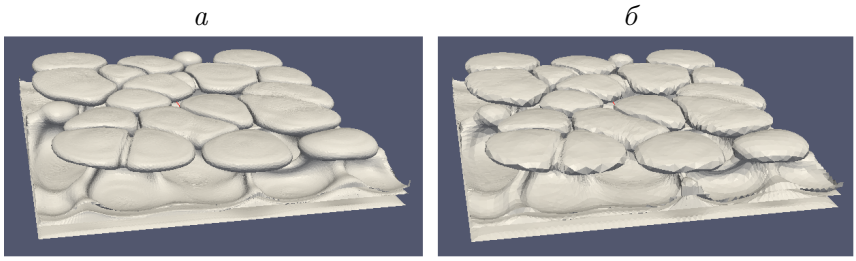

Рис. 7. Расчеты с различной детализацией поверхностей: *а* — высокая, *б* — низкая

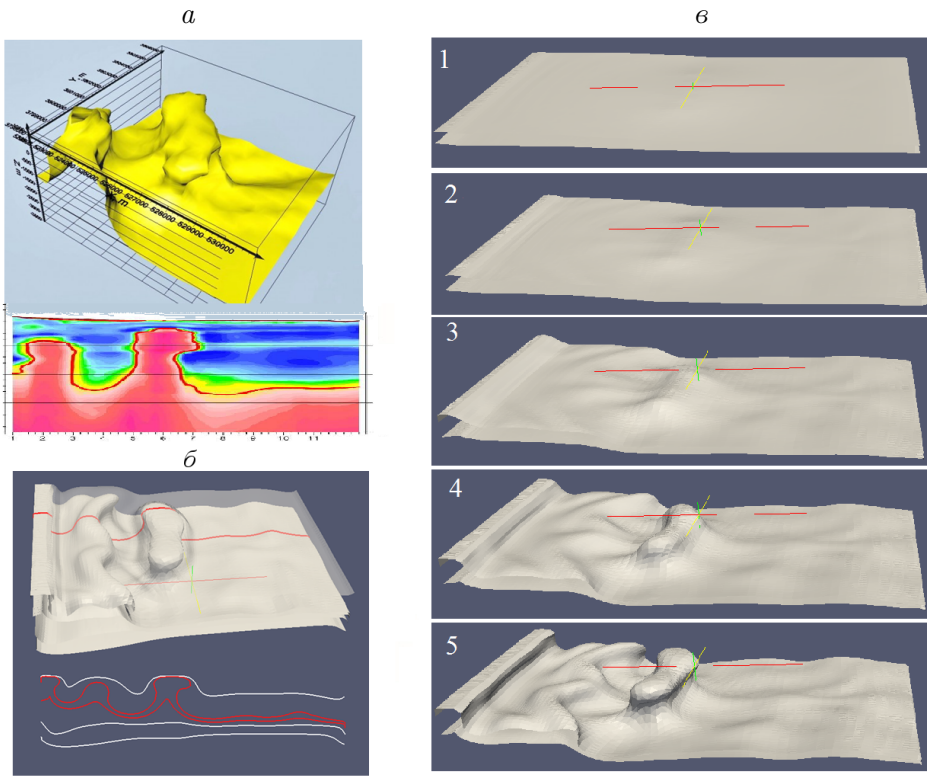

Рис. 8. Строение соляного тела и результаты моделирования: *а* — поверхность соли и разрез реальной структуры, полученные геофизическими методами [16]; *б* — модель и ее разрез, полученные в данной работе; *в* — рассчитанная эволюция соляного слоя с течением времени, показаны пять последовательных стадий

Работа программы опробовалась не только на модельных примерах. Важным практическим результатом является воспроизведение эволюции соляного купола в Иране. Его современное состояние было получено компанией "Северо-Запад" совместно с иранским партнером ZAP Consulting Engeneers, которые проводили площадные электро- и гравиразведочные работы в Исламской Республике Иран. Требовалось изучить геометрию и свойства соляного тела для проектирования и строительства в нем первого на Ближнем Востоке газового хранилища в соляном куполе. В результате этих работ определены пространственные границы соляного купола, его размеры, строение, построена 3D-модель соляного тела [16], на рис. 8, *а* показана поверхность (кровля) соли вместе с разрезом. В настоящей работе воспроизведена эволюция этой структуры. На рис. 8, *б* показано конечное состояние модели, соответствующее ее реальному современному строению (рис. 8, *а*), а также эволюция соляного слоя (рис. 8, *в*). Воспроизведение эволюции потребовало решения нескольких десятков прямых задач с различными исходными параметрами модели. В данном случае в соответствии с геологическими соображениями варьировались в основном расположение и величина начальных возмущений слоя соли. Кроме того, потребовалось к основному полю течения $\mathbf{v}$ добавить некоторое наложенное течение $\mathbf{v}^*$, отвечающее за неравномерное погружение модели вследствие накопления осадков. В реальной структуре это выражается в заметном общем наклоне поверхности соли (рис. 8, *а*).

Заключение

Представлена модификация предложенных ранее алгоритмов расчета неустойчивости Рэлея — Тейлора в высоковязкой несжимаемой ньютоновской жидкости для трехмерного случая. Перенос задачи из двумерной в трехмерную постановку потребовал не только изменения расчетных формул, но и разработки существенно новых структур и алгоритмов для работы с трехмерным распределением плотности в полупространстве. Существенно возросшая вычислительная сложность повлекла также необходимость распараллеливания вычислений.

Программно алгоритмы реализованы на языке C++ и CUDA C для параллельного расчета свертки на GPU. Текущая реализация ориентирована на выполнение на кластере с распределенной памятью, разделение работы между узлами происходит по технологии MPI. Расчет может быть выполнен либо на CPU кластера, либо на GPU. Программные классы, отвечающие за представление трехмерного распределения плотности и работу с ним, могут быть использованы в других проектах, так как не имеют жесткой привязки к предметной области.

Созданное на основе описанного метода программное обеспечение позволяет решать прямую задачу моделирования неустойчивости Рэлея — Тейлора очень быстро благодаря гораздо меньшей, чем для разностных методов, вычислительной сложности алгоритма, эффективности распараллеливания и масштабируемости. Это делает его хорошим инструментом для изучения процессов соляного диапиризма, главным образом для восстановления эволюции геологической структуры по ее конечному состоянию путем перебора решений прямых задач с различными исходными параметрами.

Скорость работы программ такова, что для многих практических задач оказалось вполне достаточно ресурсов персонального компьютера с совместимым GPU. В частности, подбор структуры соляного купола в Центральном Иране выполнен именно на персональном компьютере с одним устройством NVIDIA Tesla C1060.

Благодарности. Работа выполнена в рамках программы VIII.73.2 фундаментальных научных исследований СО РАН.

Список литературы / References

[1] **Боресков А.В., Харламов А.А.** Основы работы с технологией CUDA. М.: ДМК Пресс, 2010. 234 с.
Boreskov, A.V., Harlamov, A.A. Fundamentals of CUDA Technology. Moscow: DMK Press, 2010. 234 p. (in Russ.)

[2] **Исмаил-заде А.Т., Цепелев И.А., Тэлбот К., Остер П.** Трехмерное моделирование соляного диапиризма: численный подход и алгоритм параллельных вычислений // Вычисл. сейсмология. 2000. Вып. 31. С. 62–76.
Ismail-Zadeh, A.T, Tsepelev, I.A., Talbot, C., Oster, P. Three-dimensional modeling of salt diapirism: A numerical approach and algorithm of parallel calculations // Comput. Seism. Geodyn. Amer. Geophys. Union, Washington D.C. 2004. No. 6. P. 33–41.

[3] **Лунёв Б.В.** Изостазия как динамическое равновесие вязкой жидкости // Докл. АН СССР. 1986. Т. 290, № 1. С. 72–76.
Lunev, B.V. Isostasy as the dynamic equilibrium of a viscous fluid // Doklady AN SSSR. 1986. Vol. 290, No. 6. P. 72–76. (in Russ.)

[4] **Талбот К.Дж., Джексон М.П.А.** Соляная тектоника // В мире науки. 1987. № 10. С. 40–50.
Talbot, C.J., Jackson, M.P.A. Salt Tectonics // Scientific American. August 1987. Vol. 257, No. 2. P. 70–79.

[5] **Лунёв Б.В., Лапковский В.В.** Быстрое численное моделирование соляной тектоники: возможность оперативного использования в геологической практике // Физ. мезомеханика. 2009. Т. 12, № 1. С. 63–74.
Lunev, B.V., Lapkovskii, V.V. Fast numerical simulation of salt tectonics: possibility of the operational application in geological practice // Fizicheskaya Mezomekhanika. 2009. Vol. 12, No. 1. P. 63–74. (in Russ.)

[6] **Лунёв Б.В., Лапковский В.В.** Механизм развития инверсионной складчатости в подсолевом комплексе // Физика Земли. 2014. № 1. С. 59–65.
Lunev, B.V., Lapkovsky, V.V. Mechanism of development of inversion folding in the subsalt // Izvestiya, Physics of the Solid Earth. January 2014. Vol. 50, iss. 1. P. 57–63.

[7] **Исмаил-заде А.Т., Наймарк Б.М., Тэлбот К.** Реконструкция истории движения стратифицированной среды: обратная задача гравитационной неустойчивости // Вычисл. сейсмология. 2000. Вып. 31. С. 53–61.
Ismail-Zadeh, A., Naimark, B., Talbot, C. Reconstruction of the history of the movement of layered geostructures: Inverse problem of gravitational stability // Comput. Seism. Geodyn. Amer. Geophys. Union, Washington D.C. 2004. No. 6. P. 27–32.

[8] **Nyland, L., Harris, M., Prins, J.** GPU Gems 3. Chapter 31. Fast N-Body Simulation with CUDA. Available at: http://http.developer.nvidia.com/GPUGems3/gpugems3_ch31.html (accessed 10.07.2015).

[9] Parallel N-Body Simulations. Available at: http://www.cs.cmu.edu/~scandal/alg/nbody.html (accessed 10.07.2015).

[10] **Trenti, M., Hut, P.** N-body simulations (gravitational). Available at: http://www.scholarpedia.org/article/N-body_simulations_(gravitational) (accessed 10.07.2015).

[11] **Mattox, M.A.** Testing of a grid-based FFT n-body code for galactic simulations. Master's Theses. USA: San Jose State University, 1991. 203 p.

[12] **Greengard, L., Rokhlin, V.** A fast algorithm for particle simulations // Journal of Computational Physics. 1987. No. 73. P. 325–348.

[13] **Гумеров Н.А.** Быстрый метод мультиполей // Вест. АН РБ. 2013. Т. 18, № 4. С. 11–24.
Gumerov, N.A. Fast multipole method // Vestnik AN RB. 2013. Vol. 18, No 4. P. 11–24. (in Russ.)

[14] Разложение обратного расстояния с помощью полиномов Лежандра. Адрес доступа: http://vadimchazov.narod.ru/pertufun/funp05.htm (дата обращения 10.07.2015).
Decomposition of inverse distance using Legendre polynomials. Available at: http://vadimchazov.narod.ru/pertufun/funp05.htm (accessed 10.07.2015). (in Russ.)

[15] **Wilt, N.** The CUDA Handbook. A Comprehensive Guide to GPU Programming. Addison-Wesley Professional, 2013. 528 p.

[16] Изучение соляного купола в Иране. Адрес доступа: http://nw-geophysics.ru/geophysics/oil-and-gas/izuchenie-solianogo-kupola-v-irane/ (дата обращения 10.07.2015).
The study of the salt dome in Iran. Available at: http://nw-geophysics.ru/geophysics/oil-and-gas/izuchenie-solianogo-kupola-v-irane/ (accessed 10.07.2015). (in Russ.)

Поступила в редакцию 8 мая 2015 г.,
с доработки — 10 июля 2015 г.

Massively parallel Rayleigh — Taylor instability simulation using analytical expression of Green's function of the corresponding boundary value problem

Abramov, Timofey V.

Trofimuk Institute of Petroleum-Gas Geology and Geophysics SB RAS, Novosibirsk, 630090, Russia
Novosibirsk State University, Novosibirsk, 630090, Russia
Corresponding author: Abramov, Timofey V., e-mail: `AbramovTV@ipgg.sbras.ru`

A numerical algorithm for simulating Rayleigh — Taylor instability in high viscous incompressible Newtonian fluid was implemented. This algorithm uses analytical form of Green's function of the corresponding boundary value problem, so solution can be found as an integral of the product of two known functions without necessity to use finite difference schemes. In other words, the algorithm makes it possible to calculate the flow field at any point in space independently of other points, like in the problem of gravitational interaction of N-bodies. Because of the simplicity and the high degree of parallelism, the method is very well suited for the effective implementation, especially on massively parallel devices such as graphics cards (GPU) and hybrid clusters.

The developed software could employ an arbitrary number of GPU in hybrid cluster via MPI and CUDA technologies. The implementation for homogeneous clusters, which uses central processing units (CPU) with SSE instruction set via MPI, was proposed. In both cases the program shows a very high performance and more importantly the calculation productivity linearly depends on number of nodes. It happens due

to exceptional properties of the algorithm which requires much less memory accesses than the difference methods. Thus, the computing speed depends largely on the peak performance of the system than the memory bandwidth.

The program is used for fast numerical simulating of well known geological process, such as salt diapirism. It is a special case of Rayleigh — Taylor instability in solid rock, caused by lightness of solid rock salt, which is buried by more hard rocks. In geological time intervals (hundreds millions of years) this process is correctly described by fluid dynamics.

Keywords: NVIDIA CUDA, parallel programming, creeping flow, Green's function, Rayleigh — Taylor instability, salt diapirism.

Acknowledgements. The work was performed as part of the Program of Basic Scientific Research VIII.73.2 SB RAS

Received 8 May 2015
Received in revised form 10 July 2015