Параллельные реализации метода конечных элементов для краевой задачи для уравнений мелкой воды *

Е.Д. Карепова, В.В. Шайдуров, М.С. Вдовенко

В настоящей работе проведено исследование эффективности двух параллельных реализаций алгоритма численного решения краевой задачи для уравнений мелкой воды, выполненных с помощью библиотеки MPI для языка Си. Представлены результаты численных экспериментов на модельной сетке и неструктурированной сетке для акватории Охотского моря. Приведены сравнительные результаты ускорения вычислений в зависимости от количества процессов, способа реализации коммуникаций, способа декомпозиции вычислительной области.

1. Введение

Модели мелкой воды хорошо описывают большой круг природных явлений, таких как крупномасштабные поверхностные волны, возникающие в морях и океанах, цунами, приливные течения, поверхностный и русловой сток, гравитационные колебания поверхности океанов [1,2]. В работах [2–4] рассмотрено численное моделирование поверхностных волн в больших акваториях с учетом сферичности Земли и ускорения Кориолиса на основе уравнений мелкой воды. В работе [2] для дифференциальной постановки задачи выведены полезные априорные оценки, обеспечивающие устойчивость решения и однозначную разрешимость задачи. В [3,4] для этой же задачи построен метод конечных элементов, для которого получены необходимые априорные оценки. Там же приведены результаты численных экспериментов на модельных сетках, для акваторий Охотского моря и Мирового океана.

Разработка программных комплексов для проведения крупномасштабных вычислительных экспериментов на параллельных вычислительных системах представляет собой сложную в теоретическом и практическом плане задачу - это многоэтапный технологический процесс, неизбежно включающий в себя исследование дифференциальной задачи и численного метода решения, выбор модели программы, ее декомпозицию на параллельные процессы, анализ производительности и организацию вычислительного эксперимента.

В настоящей работе проведено исследование эффективности двух параллельных реализаций алгоритма численного решения краевой задачи для уравнений мелкой воды, выполненных с помощью библиотеки MPI для языка Си. Первая из них основана на декомпозиция области с перекрытием, ширина которого диктуется шаблоном дискретного аналога.Вторая параллельная реализация явно использует проведение вычислений в методе конечных элементов по треугольникам с декомпозицией без использования теневых граней.

Численные эксперименты проводились на модельной сетке и сетке для акватории Охотского моря. Приведены сравнительные результаты ускорения вычислений в зависимости от количества процессов, способа реализации коммуникаций (блокирующие, неблокирующие передачи), способа декомпозиции вычислительной области.

2. Моделирование поверхностных волн в водоемах методом конечных элементов

Для стандартной сферической системы координат (r, λ, θ) с началом в центре земного шара будем использовать вместо угла θ географическую широту $\varphi = \pi - \theta$, так что $0 \le \varphi < \pi$. Через λ обозначена географическая долгота $0 \le \lambda \le 2\pi$. Полагаем всюду

*Работа выполнена при финансовой поддержке Российского фонда фундаментальных исследований (грант 08-01-00621-а) и Президентской программы «Ведущие научные школы РФ» (грант НШ-3431.2008.9)

$r = R_E$, где R_E — радиус Земли, который считается постоянным.

Рассмотрим задачу в следующей постановке. Пусть Ω — заданная область на сфере с границей $\Gamma{=}\Gamma_1 \cup \Gamma_2$, где Γ_1 – часть границы, проходящая вдоль берега, а $\Gamma_2 = \Gamma \setminus \Gamma_1$ — часть границы, проходящая по морю. Обозначим через m_1 и m_2 — характеристические функции соответствующих участков границы. Для простоты считается, что точки $\varphi = 0$ и $\varphi = \pi$ (полюса) не входят в Ω. Относительно неизвестных функций $u = u(t, \lambda, \varphi)$, $v = v(t, \lambda, \varphi)$ и $\xi = \xi(t, \lambda, \varphi)$ запишем в $\Omega \times (0, T)$ уравнения баланса импульсов и уравнение неразрывности [2]:

$$\begin{aligned} \frac{\partial u}{\partial t} &= lv + mg\,\frac{\partial \xi}{\partial \lambda} - R_f u + f_1, \\ \frac{\partial v}{\partial t} &= -lu + ng\,\frac{\partial \xi}{\partial \varphi} - R_f v + f_2, \\ \frac{\partial \xi}{\partial t} &= m\left(\frac{\partial}{\partial \lambda}(Hu) + \frac{\partial}{\partial \varphi}\left(\frac{n}{m} Hv\right)\right) + f_3, \end{aligned} \tag{1}$$

где u, v — компоненты вектора скорости $\mathbf{U}$ по осям λ и φ соответственно; ξ — отклонение свободной поверхности от невозмущенного уровня; $H(\lambda, \varphi) > 0$ — глубина водоема в точке (λ, φ); функция $R_f = r_*|\mathbf{U}|/H$ учитывает силу трения о дно, r_* — коэффициент трения; $l = -2\omega \cos\varphi$ — параметр Кориолиса; $m = 1/(R_E \sin\varphi)$; $n = 1/R_E$; g — ускорение силы тяжести; $f_1 = f_1(t, \lambda, \varphi)$, $f_2 = f_2(t, \lambda, \varphi)$ и $f_3 = f_3(t, \lambda, \varphi)$ — заданные функции внешних воздействий.

Граничные условия рассмотрим в следующем виде [2]:

$$HU_n + \beta m_2 \sqrt{gH}\,\xi = m_2 \sqrt{gH}\, d \quad \text{на} \quad \Gamma \times (0, T), \tag{2}$$

где $U_n = \mathbf{U} \cdot \mathbf{n}$, $\mathbf{n} = (n_1, \dfrac{n}{m} n_2)$ — вектор внешней нормали к границе; $\beta \in [0, 1]$ — заданный параметр, $d = d(t, \lambda, \varphi)$ — заданная граничная функция, определенная на границе Γ_2.

Зададим также начальные условия

$$u(0, \lambda, \varphi) = u_0(\lambda, \varphi),\ v(0, \lambda, \varphi) = v_0(\lambda, \varphi),\ \xi(0, \lambda, \varphi) = \xi_0(\lambda, \varphi). \tag{3}$$

Для дискретизации по времени разобьем временной отрезок $[0, T]$ на K интервалов: $0 = t_0 < t_1 < \ldots < t_K = T$ с шагом $\tau = T/K$. Аппроксимируем производные по времени левыми разностями и рассмотрим систему (1)–(2) на временном интервале (t_k, t_{k+1}):

$$\begin{aligned} &\left(\frac{1}{\tau} + R_f\right) u - lv - mg\,\frac{\partial \xi}{\partial \lambda} = f_1 + \frac{1}{\tau} u^k \qquad \text{в} \quad \Omega, \\ &\left(\frac{1}{\tau} + R_f\right) v + lu - ng\,\frac{\partial \xi}{\partial \varphi} = f_2 + \frac{1}{\tau} v^k \qquad \text{в} \quad \Omega, \\ &\frac{1}{\tau}\xi - m\left(\frac{\partial}{\partial \lambda}(Hu) + \frac{\partial}{\partial \varphi}\left(\frac{n}{m} Hv\right)\right) = f_3 + \frac{1}{\tau}\xi^k \qquad \text{в} \quad \Omega, \end{aligned} \tag{4}$$

$$HU_n + \beta m_2 \sqrt{gH}\,\xi = m_2 \sqrt{gH}\, d \quad \text{на} \quad \Gamma, \quad k = 0, 1, \ldots, K-1, \tag{5}$$

где $f(t_k, \lambda, \phi) = f^k$, $f(t_{k+1}, \lambda, \phi) = f^{k+1} = f$. Индекс $k+1$ в разностных выражениях далее будем опускать там, где это не вызывает двойного толкования. Донное трение задается в виде $R_f = r_*|\mathbf{U}^k|/H$. Заметим, что граничные условия (5) являются естественными для задачи (4).

Для построения метода Бубнова-Галеркина рассмотрим некоторую согласованную триангуляцию $\mathcal{T} = \{\omega_i\}|_{i=1}^{N_{el}}$ области Ω, состоящую из невырожденных треугольников с прямолинейными сторонами в координатах λ и φ и содержащую область Ω. Согласованность означает, что каждое ребро треугольника либо является граничным и принадлежит лишь одному треугольнику, либо является общим для двух соседних треугольников, внутренность

которых попарно не пересекается. Сетка в общем случае может быть неструктурированной (рис. 1). Пусть $\bar{\Omega}_h$ — множество узлов (т.е. вершин треугольных элементов) общим числом N_{nd}, Ω_h — множество внутренних узлов.

Для каждого узла $z_j \in \bar{\Omega}_h$ введем базисную функцию $\Psi_j(\lambda, \varphi)$, которая равна единице в z_j, равна нулю во всех остальных узлах $\bar{\Omega}_h$ и линейна в каждом треугольнике. Обозначим линейную оболочку этих функций через $H_h(\Omega_h) = \mathrm{span}\{\Psi_j\}_{j=1}^{N_{nd}}$.

Для действительных вектор-функций $\mathbf{\Phi}_h = (u^h, v^h, \xi^h)$, $\hat{\mathbf{\Phi}}_h = (\hat{u}^h, \hat{v}^h, \hat{\xi}^h) \in \mathbf{H}_h(\Omega_h) \equiv (H_h(\Omega_h))^3$ рассмотрим [2–4] дискретный аналог скалярного произведения

$$(\mathbf{\Phi}_h, \hat{\mathbf{\Phi}}_h)_h = \sum_{i=1}^{N_{el}} \frac{1}{3} S_i \sum_{j=0}^{2} R_E^2 \sin(\varphi_{ij}) \left(H_{ij}(u_{ij}\hat{u}_{ij} + v_{ij}\hat{v}_{ij}) + g\xi_{ij}\hat{\xi}_{ij} \right). \tag{6}$$

Здесь через S_i обозначена площадь i-го треугольного элемента, вершины которого занумерованы через 0, 1, 2, следовательно $f_{ij} = f(\lambda_{ij}, \varphi_{ij})$ – значение функции в j-й вершине i-го элемента.

В терминах скалярного произведения (6) сформулируем метод Бубнова-Галеркина [3,4] следующим образом: *для фиксированного момента времени найти вектор-функцию* $\mathbf{\Phi}_h = (u^h(\lambda, \varphi), v^h(\lambda, \varphi), \xi^h(\lambda, \varphi))$, *где*

$$u^h(\lambda, \varphi) = \sum_{j=1}^{N_{nd}} \alpha_j^u \Psi_j(\lambda, \varphi), \quad v^h(\lambda, \varphi) = \sum_{j=1}^{N_{nd}} \alpha_j^v \Psi_j(\lambda, \varphi), \quad \xi^h(\lambda, \varphi) = \sum_{i=1}^{N_{nd}} \alpha_j^\xi \Psi_j(\lambda, \varphi)$$

такую, что тождество

$$a^h(\mathbf{\Phi}_h, \mathbf{W}_h) = f^h(\mathbf{W}_h) + b^h(d^h, \mathbf{W}_h) \tag{7}$$

справедливо $\forall\ \mathbf{W}_h = (w_u^h, w_v^h, w_\xi^h) \in \mathbf{H}_h$. Для аппроксимации интегралов при получении билинейной $a^h(\cdot, \cdot)$ и линейных $f^h(\cdot)$, $b^h(\cdot)$ форм использовалась формула трапеций и ее двумерный аналог.

Занумеровав узлы $\bar{\Omega}_h$ от 1 до N_{nd}, задачу (7) можно записать в векторно-матричной форме: *для фиксированного момента времени* t^{k+1} *найти вектор* $\mathbf{V}^{k+1} = (u_1, \ldots, u_{N_{nd}}, v_1, \ldots, v_{N_{nd}}, \xi_1, \ldots, \xi_{N_{nd}})$, *удовлетворяющий системе линейных алгебраических уравнений*

$$A^{k+1}\mathbf{V}^{k+1} = \mathbf{F}^{k+1}. \tag{8}$$

В работе [3] показан второй порядок сходимости решений в норме, порождаемой скалярным произведением (6) на равномерной сетке.

Для решения системы (8) использовался итерационный метод Якоби, который обладает хорошим параллелизмом, а диагональное преобладание для его сходимости легко обеспечивается выбором шага по времени τ. В векторно-матричной форме метод Якоби запишется в следующем виде:

$$\mathbf{\Phi}^{(\nu+\mathbf{1})} = \mathbf{\Phi}^{(\nu)} - D^{-1}\left(A\mathbf{\Phi}^{(\nu)} - \mathbf{F} \right). \tag{9}$$

Здесь ν — номер итерации, индексы $(k+1)$ для шага по времени опущены, однако компоненты глобальной матрицы жесткости и вектора правой части зависят от времени и должны пересчитываться в начале каждого временного шага.

Отметим некоторые особенности реализации, диктуемые методом конечных элементов. Глобальная матрица жесткости A зависит от времени и должна пересчитываться на каждом временном шаге. Однако для реализации метода Якоби на конечных элементах не требуется явного хранения глобальной матрицы жесткости. В программе насчитываются только

элементы локальных матриц жесткости (причем только их диагональные элементы зависят от времени и перевычисляются на каждом временном шаге). Вычисление невязки $A\mathbf{\Phi}^{(\nu)} - \mathbf{F}$ в (9) производится по треугольникам с использованием элементов локальных матриц.

Сходимость метода определялась малостью разности $\mathbf{\Phi}^{(\nu)}$ для двух соседних итераций в дискоретном аналоге равномерной нормы:

$$\max_{1 \le i \le N_{nd}} |\Phi_i^{(\nu+1)} - \Phi_i^{(\nu)}| \le \varepsilon. \tag{10}$$

3. Параллельный алгоритм

3.1. Декомпозиция области

Метод конечных элементов для задачи (7) дает семиточечный шаблон, поэтому в методе Якоби для вычисления значения $\Phi_i^{(\nu+1)}$ на $(\nu+1)$-ой итерации требуются значения в семи точках $\Phi_i^{(\nu)}$ ν-ой итерации (рис.2).

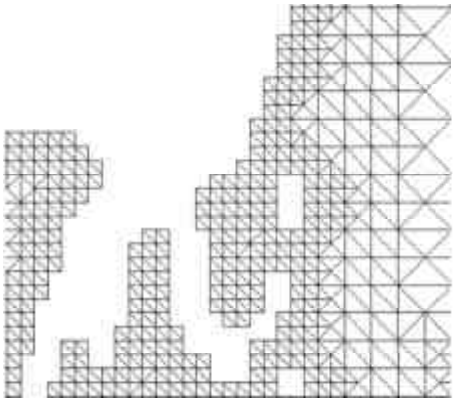

Рис. 1. Фрагмент триангуляции

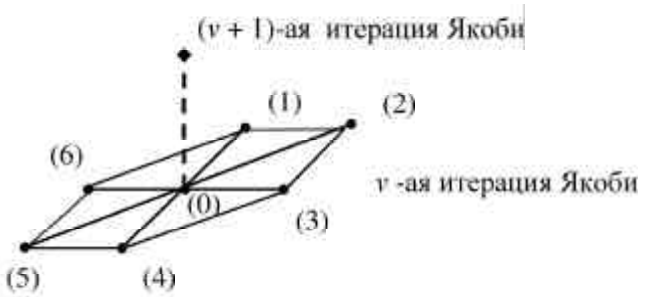

Рис. 2. Зависимость в итерациях Якоби

Используя явный параллелизм по данным, исходную расчетную область можно разбить на несколько пересекающихся только по границе разбиения подобластей. Расчеты в каждой подобласти можно проиводить независимо друг от друга в рамках итерации. Однако, после каждой итерации Якоби необходимо проводить согласование данных на границах подобластей. Имеет место, по крайней мере, два варианта разбиения.

1. *Декомпозиция без перекрытий.* Исходная область разрезается на взаимно неперекрывающиеся полосы (рис. 3). На границах каждой подобласти невязка насчитывается только по части треугольников, принадлежащих области вляния точки, лежащей в подобласти. При обмене данными после каждой итерации Якоби требуется дополнительное суммирование для значений невязки в граничных в подобласти точках.

2. *Декомпозиция с теневыми гранями.* Исходная область включает взаимно перекрывающиеся подобласти, определяемые шаблоном (рис.2). В этом случае невязка в граничных точках для подобласти i-го процесса насчитываются в подобласти соседнего процесса. С учетом семиточечного шаблона и согласованности триангуляции достаточно перекрытия областей в два слоя расчетных точек (рис. 4). При обмене данными после каждой итерации Якоби дополнительная обработка не требуется, однако для каждого процесса необходимо хранить больше данных (теневые грани шириной в две расчетные точки).

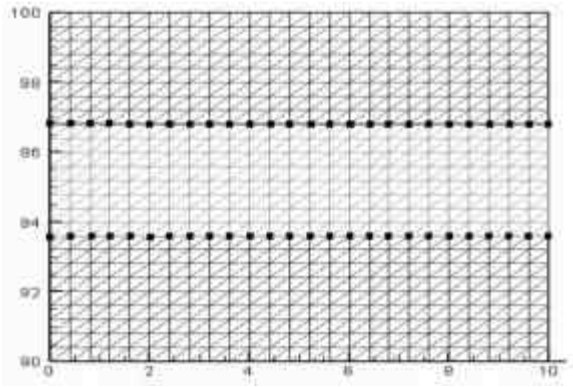

Рис. 3. Декомпозиция без перекрытий

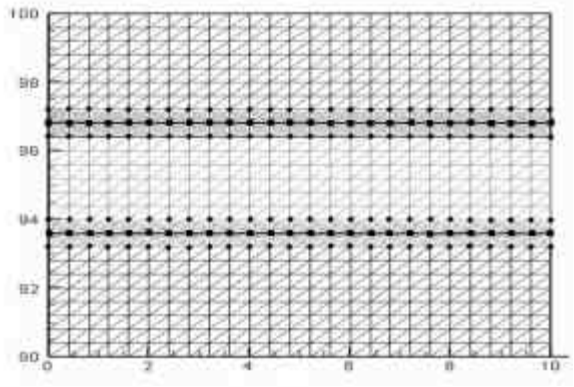

Рис. 4. Декомпозиция с теневыми гранями

Таким образом, первый способ декомпозиции области более экономичен по памяти, прост в программировании, однако предполагает дополнительное суммирование на каждой итерации Якоби. Можно предположить, что его преимущества проявятся только на больших сетках с протяженными границами подобластей. Кроме того, очевидно его достоинство для неструктурированных сеток, когда границы подобластей не являются последовательным множеством точек.

3.2. Ускорение и масштабируемость

В рамках выбранной схемы распределения данных все процессы осуществляют одни и те же вычисления, но только над разными подобластями. Структура обменов также является однородной, за исключением первого и последнего процессов.

На рис. 5 приведен псевдокод параллельной программы на одном шаге по времени для случая декомпозиции области без перекрытий. При использовании декомпозиции с теневыми гранями на Этапе 3 необходимо сначала вычислить новое приближение решения, а затем произвести обмены без дополнительного суммирования.

Реализация параллельной программы осуществлялась на языке программирования Си с применением функций библиотеки передачи сообщений MPI.

Выполним теоретические оценки возможного ускорения каждого из параллельных алгоритмов, следуя [7].

Предположим, что все операции, в том числе и обмены, имеют одинаковое время исполнения t. Пусть N_{nd} – общее количество точек сетки расчетной области, k – количество операций, выполняемых в одной расчетной точке, s – количество шагов по времени. Тогда общий объем вычислений V_{calc} в алгоритме определяется соотношением $V_{calc} = skN_{nd}$, а время выполнения алгоритма на одном процессоре соответственно можно оценить следующим образом: $T_1 \sim skN_{nd}t$. Тогда средняя степень параллелизма r данного алгоритма согласно [7] равна

$$r \sim \frac{skN_{nd}}{sk} = N_{nd}.$$

Потенциальное ускорение алгоритма оценивается как отношение времени вычисления на одном процессоре T_1 к времени вычислений на p процессорах T_p: $S_p = \dfrac{T_1}{T_p}$. Выполним теоретические оценки возможного ускорения каждого из параллельных алгоритмов, по возможности учитывая время, затрачиваемое при выполнении алгоритма на обмены и возможные накладные расходы на вычисления в перекрывающихся подобластях:

$$S_p = \frac{T_1}{T_1/p + T_{over} + T_{comm}}. \tag{11}$$

```
|{ начало цикла по времени
Этап 1: Подготовка данных для временного шага ~5% от объема вычислений
  Локальные вычисления в своей двумерной подобласти:
  | Расчет трения, внешних воздействий, коэффициентов локальных
  | матриц жесткости и правых частей, зависящих от времени;
Этап 2: Подготовка диагонали глобальной матрицы жесткости ~10% от объема вычислений
  | Вычисление диагонали глобальной матрицы жесткости;
  ************************************************************************
  Обмен с соседями граничными значениями для диагонали глобальной матрицы жесткости
  ************************************************************************
 Этап 3: Определение решения для текущего шага по времени ~85% от объема вычислений
  Локальные вычисления в своей двумерной подобласти:
  | Пока не достигнута точность метода Якоби выполнять (цикл по итерациям):
  | {
  |     { (начало цикла по треугольникам)
  |           Вычисление невязки АФ-F;
  |     } (конец цикла по треугольникам)
  ************************************************************************
  Обмен с соседями граничными значениями невязки.
  Для декомпозиции без перекрытия подобластей - суммирование по границам подобластей
  ************************************************************************
  Локальные вычисления в своей двумерной подобласти:
  |     Вычисление нового приближения решения,
  |     Вычисление локальной ошибки в норме C_h;
  ************************************************************************
  Глобальная операция MPI_Reduce(... MPI_MAX ...) для вычисления нормы ошибки
  ************************************************************************
  Локальные вычисления в своей двумерной подобласти:
  | } (конец цикла по итерациям метода Якоби)
|} (конец цикла по времени)
```

Рис. 5. Псевдокод фрагмента программы

Здесь T_1 – время вычислений на одном процессоре, T_{over} – время на дополнительные вычисления, связанные с декомпозицией области, T_{comm} – время, требуемое для обменов.

Как следует из принятой нами схемы распределения данных, на каждой итерации метода Якоби требуется обмен граничными элементами вектора решений и порождаемые распределенностью данных дополнительные вычисления. В обоих случаях декомпозиции области для вычисления невязки на каждом разрезе количество точек, в которых необходим обмен данными, соответствует количеству точек сетки на разрезе. Однако при декомпозиции области с теневыми гранями согласно шаблону метода Якоби (рис. 2) требуется хранить не только точки на разрезе, но и все точки, принадлежащие треугольникам, в которые входят точки разреза. Обозначим через N_{bnd} количество точек сетки, в которых необходим обмен данными, далее такие точки будем называть граничными в подобласти.

Объем дополнительных вычислений, связанных с декомпозицией области зависит от количества процессоров p, на которых выполняется алгоритм и количества дополнительных вычислений g, которые необходимы для одной граничной в подобласти точки. Источники этих вычислений при различных способах декомпозиции области отличаются. При декомпозиции без теневых граней каждый процесс, вычисляя невязку в граничной в подобласти точке, "обрабатывает" свою часть треугольников, не дублируя вычисления. В этом случае после обмена требуется дополнительное суммирование частей невязок, насчитанных

в соседнем процессоре. При декомпозиции области с теневыми гранями соседние процессы дублируют друг друга при расчете невязки для точек *перед разрезом*. Таким образом, объем дополнительных вычислений оценивается следующим образом:

$$V_{over} \sim sgN_{bnd}(p-1).$$

При этом время, затрачиваемое на дополнительные вычисления в N_{bnd} граничных точках подобласти можно оценить так:

$$T_{over} \sim sgN_{bnd}t.$$

Пусть также для каждой граничной точки требуется пересылка m значений. Тогда объем пересылаемых данных соседним процессам можно оценить следующим образом:

$$V_{comm} \sim 2smN_{bnd}.$$

Для оценки времени, необходимого для обменов, рассмотрим их реализацию в MPI. При использовании библиотеки MPI возможно два принципиально разных способа организации обменов – с использованием блокирующих или неблокирующих функций приема и передачи данных. Оценим ускорение в обоих случаях.

Блокирующие передачи. На рис. 6 отображено влияние блокирующих передач на производительность на примере восьми процессов [8]. На первом этапе шестой процесс передает данные седьмому процессу, в то время как процессы с нулевого по пятый простаивают в ожидании приема данных своими правыми соседями. На втором этапе пятый процесс передает данные шестому, а процессы с нулевого по четвертый простаивают. Таким образом, все обмены завершатся за семь этапов.

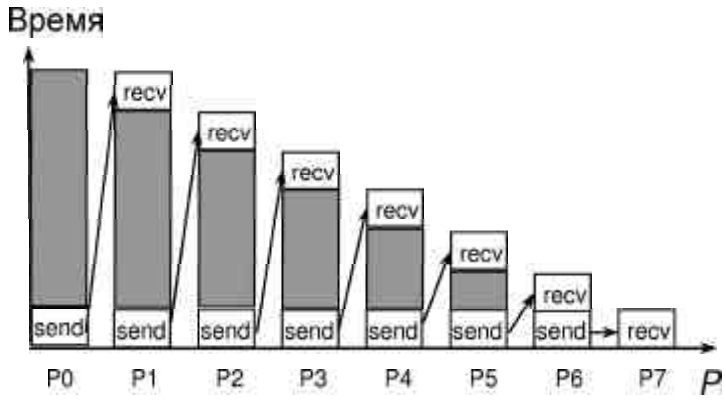

Рис. 6. Влияние блокирующих передач на ускорение.

В общем случае обмены с использованием блокирующих передач потребуют $T_{comm}(p-1)$ единиц времени, где $T_{comm} = 2smN_{bnd}t$ – время на обмен данными с одной границы, p – количество используемых процессоров.

С учетом (11) ускорение при использовании блокирующих передач с учетом всех оценок будет определяться следующим соотношением:

$$S_p^{bl} = \begin{cases} \dfrac{1}{\dfrac{1}{p} + \dfrac{g}{k}R + 2(p-1)\dfrac{m}{k}R}, & 1 \le p \le N_{nd}, \\ \dfrac{1}{\dfrac{1}{N_{nd}} + \dfrac{g}{k}R + 2(p-1)\dfrac{m}{k}R}, & p > N_{nd}. \end{cases} \tag{12}$$

Здесь через R обозначено отношение количества граничных точек в подобласти к общему числу точек расчетной области: $R = N_{bnd}/N_{nd}$

Неблокирующие передачи. В общем случае время, затрачиваемое на обмены с использованием неблокирующих передач, не зависят от количества участвующих в обменах процессов: $T_{comm} = Const$. Оценка (11) в случае неблокирующих обменов имеет вид:

$$S_p^{unbl} = \begin{cases} \dfrac{1}{\dfrac{1}{p} + \dfrac{g}{k}R + \dfrac{2m}{kN_{nd}}}, & 1 \le p \le N_{nd}, \\ \dfrac{1}{\dfrac{1}{N_{nd}} + \dfrac{g}{k}R + \dfrac{2m}{kN_{nd}}}, & p > N_{nd}. \end{cases} \tag{13}$$

Для получения численных оценок ускорения оценим количество операций k на одну расчетную точку. Каждый шаг по времени включает в себя: 1) вычисления трения, правой части, диагональных элементов локальных матриц жесткости $\sim 60N_{nd}$ операций; 2) вычисление диагонали глобальной матрицы жесткости $\sim 45N_{el}$ операций (N_{el} – количество треугольных элементов в триангуляции расчетной области); 3) решение системы (8) методом Якоби (9). Реализация метода Якоби со сборкой невязки по треугольникам требует 1) вычисления невязки $\sim 180N_{nd}$ операций; 2) $3N_{el}$ проверки на принадлежность точки границе; 3) $\sim 60N_b$ дополнительных операций для вычисления интегралов в N_b граничных точках расчетной области; 4) N_{nd} операций для умножения на D^{-1}; 5) $3N_{nd}$ проверок для вычисления глобального максимума.

В табл. 1 приведены результаты теоретических оценок ускорения параллельного алгоритма в зависимости от способа декомпозиции области, типа обменов и количества процессов для модельной прямоугольной области с сеткой 401×401 точек. Здесь $N_{nd} = 160801$, $N_{el} = 320000$.

Таблица 1. Значения оценки ускорения в зависимости количества процессов

Тип декомпозиции	p	1	2	4	8	12	16
Без перекрытия	S_p^{bl}	1.0000	1,9999	3,9986	7,9866	11,9526	15,8855
Без перекрытия	S_p^{unbl}	1.0000	1,9999	3,9995	7,9981	11,9957	15,9923
С теневыми гранями	S_p^{bl}	1.0000	1,9962	3,9830	7,9173	11,7814	15,5554
С теневыми гранями	S_p^{unbl}	1.0000	1,9962	3,9849	7,9399	11,8653	15,7615

Из таблицы 1 видно, что наиболее эффективной является реализация, основанная на разбиении области без перекрытия подобластей. Полученные результаты показывают, что алгоритм обладает значительным объемом потенциального параллелизма и хорошей с точки зрения распараллеливания структурой, что дает ускорение близкое к линейному в зависимости от количества используемых процессоров.

4. Вычислительный эксперимент

4.1. Модельная область

Для численного исследования ускорения параллельного алгоритма рассмотрим следующую модельную задачу. Пусть Ω — «квадрат» на сфере: $\Omega = [0, \pi/10] \times [\pi/2, \pi/2 + \pi/10]$. Границы Ω считаются «твердыми». В Ω рассмотрена задача с известным точным решением [3]. В расчетной области построены две равномерные квадратные сетки 401×401 и 801×801 точек с соответствующей согласованной триангуляцией. В вычислительных экспериментах было сделано 1000 шагов по времени.

Вычисления выполнялись на 96-процессорном кластере ИВМ СО РАН. Кластер МВС-1000/ИВМ (собственная сборка ИВМ СО РАН, [11]) содержит 24 вычислительных узла AMD Athlon64/3500+/1Гb (однопроцессорные, одноядерные); 12 вычислительных узлов AMD Athlon64 X2 Dual Core/4800+/2Гб (однопроцессорные, двухъядерные); 12 вычислительных узлов 2 X Dual-Core AMD Opteron Processor 2216/4Гб (двухпроцессорные, двухъядерные). Управляющий узел, сервер доступа и файловый сервер Athlon64/3500+/1Gb с общей дисковой памятью 400 Гб под управлением ОС Gentoo Linux. Управляющая сеть FastEthernet (100 Мбит/сек), сеть передачи данных GigaEthernet (1000 Мбит/сек). Необходимо отметить, что поскольку кластер является гетерогенным, то временные характеристики выполнения программы осреднялись по результатам нескольких десятков расчетов.

На рис. 7 представлена зависимость ускорения вычислений от количества используемых процессов для модельных сеток. Чтобы не загромождать рисунок, не отображены графики ускорений, полученных для декомпозиции с теневыми гранями с блокирующими обменами, поскольку они практически совпадают с графиками ускорения для декомпозиции с теневыми гранями с неблокирующими обменами.

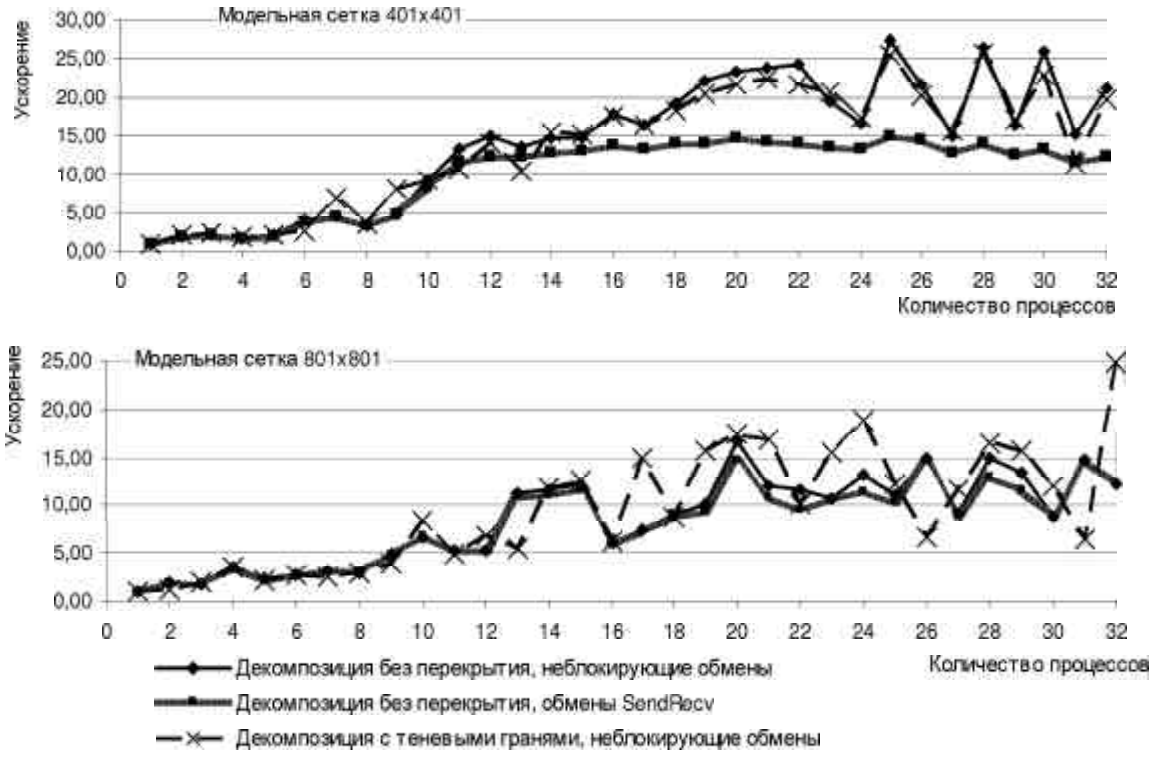

Рис. 7. Зависимость ускорения вычислений от количества доступных процессоров.

Анализ вычислительных экспериментов показывает, что при сравнительно небольшом количестве процессоров ($P \leq 12$) значение ускорения остается одинаковым для всех рассматриваемых вариантов, при этом можно отметить незначительное преимущество декомпозиции без теневых граней.

В алгоритме пока не используется возможность совмещения вычислений и обменов, поэтому влияние блокирующих и неблокирующих вариантов пересылок должно быть минимальным. В тоже время, на сетке размерностью 401×401 блокирующие обмены для декомпозиции без теневых граней не дают рост ускорения, начиная с вычислений на 12-ти процессах. В остальных случаях на сетке 401×401 рост ускорения наблюдается вплоть до использования 22-ух процессов. Наличие эффективности больше единицы в районе 18-ти – 22-ух процессов объясняется по-видимому попаданием в кэш. На сетке 801×801 при об-

щем увеличении ускорения вплоть до использования 20 процессов отмечается «провал» при 16-ти процессах. Неустойчивость при количестве процессов больше 22-ух говорит о преобладании в этих случаях времени обменов над вычислениями, архитектурными особенностями кластера и реализацией MPI.

В работах [5, 6] приведены некоторые результаты, характеризующие для этой же задачи эффективность распараллеливания кода с использованием языка Fortran-DVM при проведении численных экспериментов на равномерной модельной сетке и в акватории Мирового океана. Декомпозиция области в работе проводилась автоматически и использовала технологию теневых граней. Проводя сравнение с этими результатами заметим, что при использовании 8-ми и более процессов программирование параллельных алгоритмов при помощи функций MPI показывает несколько более высокие значения ускорения вычислений. Однако, данных для добросовестного сравнения возможностей MPI и DVM на данной задаче не достаточно.

4.2. Вычислительные эксперименты для акватории Охотского моря

Тестовые расчеты для акватории Охотского моря проводились на сетках, подготовленых С.Ф. Пятаевым и И.В. Киреевым на основе открытой батиметрической базы данных ETOPO2 [6, 10].

В рассмотренной сетке число узлов $N_{nd} = 43768$, а число треугольных элементов $N_{el} = 78929$ (рис. 8 а).

Численный эксперимент соответствует начальным данным с локальным подъемом уровня, описываемым гауссовской функцией

$$\xi(0, \lambda, \varphi) = A \exp\Big(-\Big(\frac{\lambda - \lambda_0}{2D}\Big)^2 - \Big(\frac{\varphi - \varphi_0}{2D}\Big)^2\Big). \tag{14}$$

В расчетах было принято $A = 10$, $\lambda_0 = 149.1^\circ$ восточной долготы, а $\varphi_0 = 53.1^\circ$ северной широты, $D = 0.005$, коэффициент трения $r_* = 0.0026$. Для скоростей предполагались нулевые начальные данные: $u(0, \lambda, \varphi) = v(0, \lambda, \varphi) = 0$.

В построенной сетке из общего числа граничных участков около 6.4% проходят по морю. На таких участках $m_2 = 1$ и в краевом условии (2) полагали $\beta = 1$ и $d(t, \lambda, \varphi) \equiv 0$.

Результаты численного моделирования показаны на рис. 8 б) - г), где представлена функция $\xi(t, \lambda, \varphi)$ для некоторых моментов времени. Линиями показаны границы, получившиеся при декомпозиции области на 8 частей. Поскольку сетка сильно сгущается в приграничных областях, то ширина полосы, которая отводится одному процессору, в прибрежной зоне значительно уже, чем в море.

В табл. 2 приведены результаты численных экспериментов на различном числе процессоров. Здесь верхние индексы «bl» и «$unbl$» соответствуют экспериментам с блокирующими и неблокирующими операциями обмена соответственно.

Из таблицы видно, что ускорение вычислений увеличивается пропорционально количеству доступных процессоров. При этом время вычислений при использовании неблокирующих передач незначительно меньше по сравнению со временем вычислений при наличии блокирующих передач.

Следует отметить, что декомпозиция области с теневыми гранями в этом случае является трудоемким процессом, поэтому не проводилась.

Сравнение с результатами, полученными при применении технологии DVM [6], дает преимущество наших алгоритмов при большом количестве процессоров.

5. Заключение

На примере численного решения краевой задачи для уравнений мелкой воды в работе рассмотрены технологические аспекты разработки масштабируемых параллельных алгоритмов для кластерных вычислительных систем с использованием библиотеки MPI.

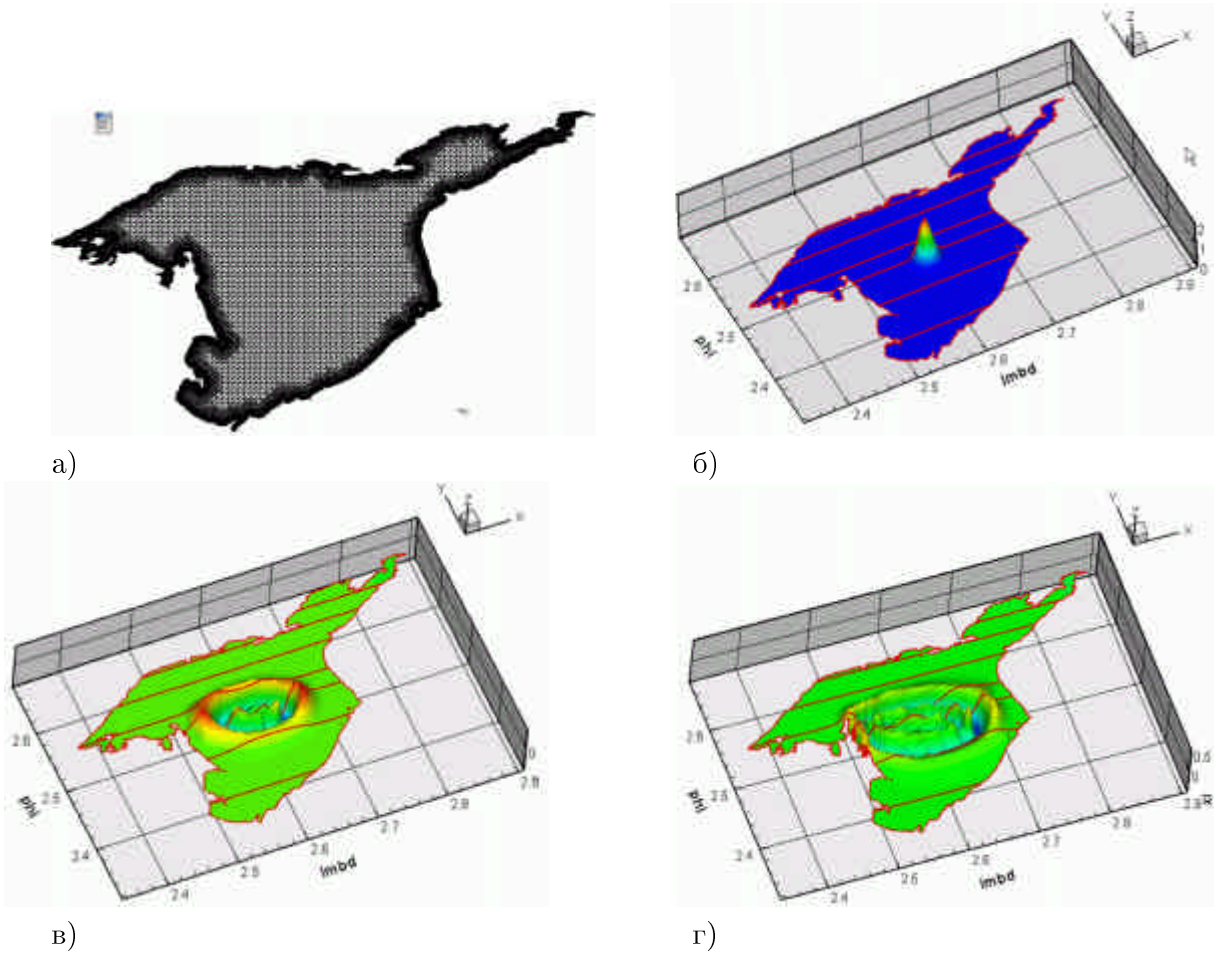

Рис. 8. Численные эксперименты в акватории Охотского моря на 8 процессорах: а) общий вид сетки; б) начальное возмущение; в) через 42 мин.; г) через 67 мин.

Поскольку при дискретизации задачи использовался метод конечных элементов с организацией вычислений по треугольным элементам, было рассмотрено два естественных подхода к декомпозиции области – без перекрытий и с теневыми гранями.

Теоретические оценки показали, что алгоритм обладает значительным объемом потенциального параллелизма и хорошей с точки зрения распараллеливания структурой, что дает ускорение в зависимости от количества используемых процессоров теоретически близкое к линейному.

Численные эксперименты показали, что использование неблокирующего режима обменов, которое допускается алгоритмом, является безусловно более эффективным. В дальнейшем планируется повысить эффективность алгоритма, в том числе и за счет совмещения вычислений с неблокирующими операциями обмена.

Следует отметить как явное преимущество простоту организации параллельного алгоритма при декомпозиции без перекрытия подобластей на неструктурированных триангуляциях реальных акваторий.

Список литературы

1. Марчук Г.И. Динамика океанских приливов. / Марчук Г.И., Каган Б.А. – Л.: Гидрометиздат, 1983.

Таблица 2. Результаты численных экспериментов для задачи в акватории Охотского моря

количество процессоров	1	2	4	8
Время t^{bl}, с	6,8729	4,0690	2,3030	1,6324
Ускорение S^{bl}	1.0000	1,6891	2,9843	4,2103
Эффективность E^{bl}	1.0000	0,8446	0,7461	0,5263
Время t^{unbl}, с	7,0290	4,1209	2,3389	1,5304
Ускорение S^{unbl}	1.0000	1,7057	3,0053	4,5928
Эффективность E^{unbl}	1.0000	0,8528	0,7513	0,5741

2. Agoshkov V.I. Inverse problems of the mathematical theory of tides: boundary-function problem // Russ. J. Numer. Anal. Math. Modelling. – 2005. – vol. 20, № 1. – P. 1–18.

3. Kamenshchikov L.P. Simulation of surface waves in basins by the finite element method / Kamenshchikov L.P., Karepova E.D., Shaidurov V.V. // Russian J. Numer. Anal. Math. Modelling - Vol. 21, №4 (2006). - Pp. 305 - 320.

4. Karepova E.D. Numerical Solution of the Boundary Problem foe Shallow Water Equations for Modelling Surface Waves in World Ocean by Finite Elements Methods / Kamenshchikov L.P., Karepova E.D., Shaidurov V.V. // In: Finite Difference Methods: Theory and Applications. Proceedings of Fourth International Conference FDM:T&A'06. - Bulgaria: Rousse, 2007. - P. 227 - 233

5. Каменщиков Л.П. Моделирование поверхностных волн в водоемах методом конечных элементов на вычислительном кластере / Каменщиков Л.П., Карепова Е.Д., Шайдуров В.В. // Избр. матералы Четвертой школы-семинара «Распределенные и кластерные вычисления», Красноярск: ИВМ СО РАН. - 2005. - С. 114-125.

6. Каменщиков Л.П. Моделирование гравитационных волн в Мировом океане методом конечных элементов с распараллеливанием / Каменщиков Л.П., Карепова Е.Д., Пятаев С.Ф., Шайдуров В.В. // Избр. матералы Шестой школы-семинара «Распределенные и кластерные вычисления», Красноярск: ИВМ СО РАН. - 2006. - С. 52-64.

7. Ортега, Дж. Введение в параллельные и векторные методы решения линейных систем: Пер. с англ. / Дж. Ортега. – М.: Мир, 1991.

8. Balay, S. Efficient Management of Parallelism in Object-Oriented Numerical Software Libraries, Modern Software Tools in Scientific Computing / S. Balay, W.D. Gropp, L.C. McInnes and others. – Birkhauser Press, 1997. – Pp. 163–202.

9. McBryan O.A. An overwiev of message passing environments / O.A. McBryan // Parallel Computing. – 1994. – V 20. – Pp.417-441.

10. National Geophysical Data Center. `http://www.ngdc.noaa.gov/ngdc.html`

11. Исаев С.В. Развитие Красноярского центра параллельных вычислений. /Исаев С.В., Малышев А.В., Шайдуров В.В. // Вычислительные технологии. - Т.11,спецвыпуск. - Новосибирск: Издательство Сибирского отделения Российской академии наук. - 2006. - С.28-33