

Московский государственный университет имени М. В. Ломоносова  
Факультет вычислительной математики и кибернетики

Отчет

По курсу: «Суперкомпьютерное моделирование и технологии»  
«MPI»

Выполнила:  
Казанцева Варвара Денисовна  
Группа: 608

26 ноября 2025 г.

# Содержание

<b>1</b>	<b>Введение</b>	<b>2</b>
<b>2</b>	<b>Математическая постановка задачи</b>	<b>2</b>
<b>3</b>	<b>Метод фиктивных областей</b>	<b>2</b>
<b>4</b>	<b>Разностная схема решения задачи</b>	<b>2</b>
4.1	Определение коэффициентов разностной схемы . . . . .	3
4.2	Определение коэффициентов разностной схемы аналитически . . . . .	3
<b>5</b>	<b>Метод решения системы линейных алгебраических уравнений</b>	<b>3</b>
<b>6</b>	<b>Программная реализация</b>	<b>4</b>
6.1	Ключевые функции программы . . . . .	4
6.2	Используемые средств MPI . . . . .	5
6.3	Запуск программы . . . . .	6
<b>7</b>	<b>Результаты расчетов</b>	<b>6</b>
7.1	Параллельная программа с MPI . . . . .	6
7.2	Результаты расчетов на ПВС IBM Polus . . . . .	6
7.3	Графики . . . . .	6
7.4	График приближенного решения . . . . .	6
7.5	Графики ускорения . . . . .	7
7.6	Графики сходимости . . . . .	8
<b>8</b>	<b>Заключение</b>	<b>8</b>

# 1 Введение

В данной работе рассматривается приближенное решение двумерной задачи Дирихле для уравнения Пуассона в криволинейной области. В качестве расчетной области выбран **остроугольный треугольник с вершинами в точках C(3,0), A(3,0), B(0,4)** (вариант 2). Целью работы является распараллеливание последовательного решения данной задачи с использованием средств MPI.

## 2 Математическая постановка задачи

В области  $D \subset \mathbb{R}^2$ , ограниченной кусочно-гладким контуром  $\Gamma$ , рассматривается дифференциальное уравнение Пуассона:

$$-\Delta u = f(x, y), \quad (x, y) \in D \quad (1)$$

где оператор Лапласа:

$$\Delta u = \frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} \quad (2)$$

функция  $f(x, y) = 1$  задана. Для выделения единственного решения уравнение дополняется граничным условием Дирихле:

$$u(x, y) = 0, \quad (x, y) \in \Gamma \quad (3)$$

## 3 Метод фиктивных областей

Для приближенного решения задачи (1)-(3) используется метод фиктивных областей. Область  $D$  принадлежит прямоугольнику  $\Pi = \{(x, y) : A_1 < x < B_1, A_2 < y < B_2\}$ . В прямоугольнике  $\Pi$  рассматривается задача Дирихле с кусочно-постоянным коэффициентом:

$$k(x, y) = \begin{cases} 1, & (x, y) \in D \\ \frac{1}{\varepsilon}, & (x, y) \in \hat{D} \end{cases} \quad (4)$$

и правой частью:

$$F(x, y) = \begin{cases} f(x, y), & (x, y) \in D \\ 0, & (x, y) \in \hat{D} \end{cases} \quad (5)$$

где  $\hat{D} = \Pi \setminus D$  - фиктивная область,  $\varepsilon > 0$  - малый параметр.

## 4 Разностная схема решения задачи

Краевая задача решается численно методом конечных разностей. В замыкании прямоугольника  $\Pi$  определяется равномерная прямоугольная сетка  $\bar{\omega}_h = \bar{\omega}_1 \times \bar{\omega}_2$ , где:

$$\begin{aligned} \bar{\omega}_1 &= \{x_i = A_1 + ih_1, i = 0, 1, \dots, M\}, \\ \bar{\omega}_2 &= \{y_j = A_2 + jh_2, j = 0, 1, \dots, N\} \end{aligned}$$

с шагами  $h_1 = (B_1 - A_1)/M$ ,  $h_2 = (B_2 - A_2)/N$ .

Дифференциальное уравнение аппроксимируется разностным уравнением:

$$-\frac{1}{h_1}(a_{i+1,j}w_{x,ij} - a_{i,j}w_{\bar{x},ij}) - \frac{1}{h_2}(b_{i,j+1}w_{y,ij} - b_{i,j}w_{\bar{y},ij}) = F_{ij} \quad (6)$$

где коэффициенты  $a_{ij}$ ,  $b_{ij}$  вычисляются через интегралы от функции  $k(x, y)$ .

## 4.1 Определение коэффициентов разностной схемы

Коэффициенты  $a_{ij}$  вычисляются по формуле:

$$a_{ij} = \frac{1}{h_2} \int_{y_{j-1/2}}^{y_{j+1/2}} k(x_{i-1/2}, t) dt \quad (7)$$

где  $x_{i\pm 1/2} = x_i \pm 0.5h_1$ ,  $y_{j\pm 1/2} = y_j \pm 0.5h_2$ .

Коэффициенты  $b_{ij}$  вычисляются по формуле:

$$b_{ij} = \frac{1}{h_1} \int_{x_{i-1/2}}^{x_{i+1/2}} k(t, y_{j-1/2}) dt \quad (8)$$

Правая часть  $F_{ij}$  вычисляется как:

$$F_{ij} = \frac{1}{h_1 h_2} \iint_{\Pi_{ij}} F(x, y) dx dy \quad (9)$$

где  $\Pi_{ij} = \{(x, y) : x_{i-1/2} \leq x \leq x_{i+1/2}, y_{j-1/2} \leq y \leq y_{j+1/2}\}$ .

## 4.2 Определение коэффициентов разностной схемы аналитически

Для коэффициента  $a_{ij}$ :

$$a_{ij} = \begin{cases} 1, & \text{если отрезок } [P_{ij}, P_{ij+1}] \subset D \\ \frac{1}{\varepsilon}, & \text{если отрезок } [P_{ij}, P_{ij+1}] \subset \hat{D} \\ \frac{l_{ij}}{h_2} + \frac{1}{\varepsilon}(1 - \frac{l_{ij}}{h_2}), & \text{иначе} \end{cases} \quad (10)$$

где  $l_{ij}$  - длина части отрезка  $[P_{ij}, P_{ij+1}]$ , принадлежащей области  $D$ ,  $P_{ij} = (x_{i-1/2}, y_{j-1/2})$ .

Аналогично для коэффициента  $b_{ij}$ :

$$b_{ij} = \begin{cases} 1, & \text{если отрезок } [P_{ij}, P_{i+1j}] \subset D \\ \frac{1}{\varepsilon}, & \text{если отрезок } [P_{ij}, P_{i+1j}] \subset \hat{D} \\ \frac{l_{ij}}{h_1} + \frac{1}{\varepsilon}(1 - \frac{l_{ij}}{h_1}), & \text{иначе} \end{cases} \quad (11)$$

где  $l_{ij}$  - длина части отрезка  $[P_{ij}, P_{i+1j}]$ , принадлежащей области  $D$ .

Для правой части  $F_{ij}$ :

$$F_{ij} = \begin{cases} 0, & \text{если } \Pi_{ij} \subset \hat{D} \\ 1, & \text{если } \Pi_{ij} \subset D \\ \frac{S_{ij}}{h_1 h_2}, & \text{иначе} \end{cases} \quad (12)$$

где  $S_{ij}$  - площадь пересечения  $\Pi_{ij} \cap D$ .

$\varepsilon = h^2$  - малый параметр,  $h = \max(h_1, h_2)$ .

## 5 Метод решения системы линейных алгебраических уравнений

Приближенное решение разностной схемы находится итерационным методом сопряженных градиентов с диагональным предобуславливанием. Алгоритм метода:

1. Выбирается начальное приближение  $w^{(0)}$

2. Вычисляется невязка  $r^{(0)} = B - Aw^{(0)}$
3. Решается система  $Dz^{(0)} = r^{(0)}$
4. Задается направление  $p^{(1)} = z^{(0)}$
5. Для  $k = 0, 1, 2, \dots$  до сходимости:

$$\begin{aligned}
\alpha_{k+1} &= \frac{(z^{(k)}, r^{(k)})}{(Ap^{(k+1)}, p^{(k+1)})} \\
w^{(k+1)} &= w^{(k)} + \alpha_{k+1}p^{(k+1)} \\
r^{(k+1)} &= r^{(k)} - \alpha_{k+1}Ap^{(k+1)} \\
Dz^{(k+1)} &= r^{(k+1)} \\
\beta_{k+1} &= \frac{(z^{(k+1)}, r^{(k+1)})}{(z^{(k)}, r^{(k)})} \\
p^{(k+2)} &= z^{(k+1)} + \beta_{k+1}p^{(k+1)}
\end{aligned}$$

Условие остановки итерационного процесса:

$$\|w^{(k+1)} - w^{(k)}\|_E < \varepsilon \quad (13)$$

## 6 Программная реализация

Программа реализована с использованием технологии MPI для распределения вычислений на несколько вычислительных узлов. Основной подход – декомпозиция области на прямоугольные подобласти (домены) с организацией межпроцессного взаимодействия через виртуальную декартову топологию.

Создается двумерная декартова топология процессов с автоматическим определением оптимального разбиения. Далее исходная прямоугольная сетка размером  $M \times N$  равномерно распределяется между процессами. Каждый процесс получает свою подобласть с индексами:

start\_i, end\_i – границы (глобальные индексы узлов) по оси X  
start\_j, end\_j – границы (глобальные индексы узлов) по оси Y  
size\_M, size\_N – размер локальной подобласти

Каждый процесс хранит локальные массивы коэффициентов, правой части и решения для своей подобласти. Эти массивы чуть больше необходимого, так как дополнительно хранятся граничные зоны (это требуется для решения разностной схемы и удобства обмена данными).

### 6.1 Ключевые функции программы

**Функции предварительных вычислений:**

- `Domain::Domain()` – инициализация декартовой топологии и распределения данных
- `initializeCoefficients()` – вычисление коэффициентов  $a_{ij}$ ,  $b_{ij}$  для локальной подобласти
- `initializeF()` – вычисление правой части  $F_{ij}$  для локальной подобласти

### Функции численного метода:

- `conjugateGradient()` – реализация метода сопряженных градиентов
- `applyOperator()` – применение разностного оператора с обменом граничными значениями
- `solveDiagonal()` – решение системы с диагональным предобуславливанием
- `dot()` – вычисление скалярного произведения с MPI-редукцией
- `normDifference()` – вычисление нормы разности решений с MPI-редукцией

### Функция сохранения:

- `save()` – поочередное сохранение решения каждым процессом

## 6.2 Используемые средств MPI

Для организации параллельных вычислений и межпроцессного взаимодействия были использованы следующие функции библиотеки MPI:

### Инициализация и управление процессами:

- `MPI_Init(&argc, &argv)` – инициализация MPI-окружения
- `MPI_Finalize()` – завершение работы с MPI
- `MPI_Comm_rank(MPI_COMM_WORLD, &rank)` – получение номера текущего процесса
- `MPI_Comm_size(MPI_COMM_WORLD, &size)` – получение общего количества процессов

### Создание виртуальной топологии:

- `MPI_Dims_create(size, 2, dims)` – автоматическое определение оптимальной двумерной сетки процессов
- `MPI_Cart_create(MPI_COMM_WORLD, 2, dims, periods, 1, &comm_cart)` – создание декартовой топологии
- `MPI_Cart_coords(comm_cart, rank, 2, coords)` – получение координат процесса в декартовой сетке
- `MPI_Cart_shift(comm_cart, 0, 1, &left, &right)` – определение соседей по направлениям

### Коммуникационные операции:

- `MPI_Isend(buf, count, type, dest, tag, comm, &request)` – неблокирующая отправка данных
- `MPI_Irecv(buf, count, type, source, tag, comm, &request)` – неблокирующий прием данных
- `MPI_Waitall(count, requests, statuses)` – ожидание завершения всех неблокирующих операций

- `MPI_Allreduce(sendbuf, recvbuf, count, type, op, comm)` – глобальная редукция с распространением результата
- `MPI_Barrier(comm)` – синхронизация всех процессов

### 6.3 Запуск программы

Для параллельной версии с MPI используется расширенный Makefile. Команды для работы с параллельной версией:

```
make run_1      # 1 процесс
make run_2      # 2 процесса
make run_4      # 4 процесса
make run_polus_all # пакетный запуск на кластере Polus
```

## 7 Результаты расчетов

### 7.1 Параллельная программа с MPI

Проведены расчеты на сетке  $(M, N) = (40 \times 40)$  с различным количестве процессов. Последовательная программа обрабатывает на данной сетке за 0.00194 секунд.

Число процессов	Число итераций	Время решения, с
1	103	0.00296
2	103	0.00205
4	103	0.00133

Таблица 1: Результаты расчетов MPI-программы на сетке  $40 \times 40$

### 7.2 Результаты расчетов на ПВС IBM Polus

Количество процессов	Число точек сетки $(M \times N)$	Число итераций	Время решения, с	Ускорение
0	$400 \times 600$	1077	6.11	1.0
2	$400 \times 600$	1077	2.558	2.39
4	$400 \times 600$	1077	1.400	4.36
8	$400 \times 600$	1077	0.873	7.00
16	$400 \times 600$	1077	0.427	14.31
0	$800 \times 1200$	2047	46.98	1.0
4	$800 \times 1200$	2047	9.893	4.75
8	$800 \times 1200$	2047	5.552	8.46
16	$800 \times 1200$	2047	3.267	14.38
32	$800 \times 1200$	2047	1.371	34.27

Таблица 2: Таблица с результатами расчетов на ПВС IBM Polus (MPI)

### 7.3 Графики

### 7.4 График приближенного решения

На рисунке 1 представлено приближенное решение задачи Дирихле для уравнения Пуассона в треугольной области, полученное на сетке  $800 \times 1200$ .

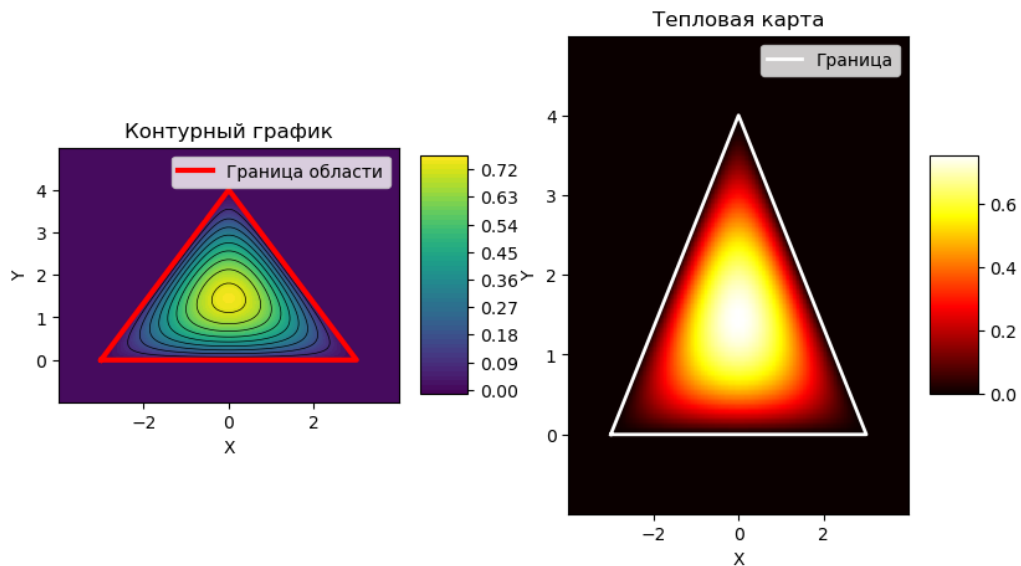


Рис. 1: Приближенное решение уравнения Пуассона в треугольной области на сетке  $800 \times 1200$

Решение демонстрирует ожидаемое поведение - максимальные значения в центре треугольной области с плавным уменьшением к границам, где задано нулевое граничное условие. Форма решения соответствует геометрии расчетной области.

## 7.5 Графики ускорения

На рисунках 2 и 4 приведены графики ускорения программы при различном числе MPI процессов для сеток  $400 \times 600$  и  $800 \times 1200$  соответственно.

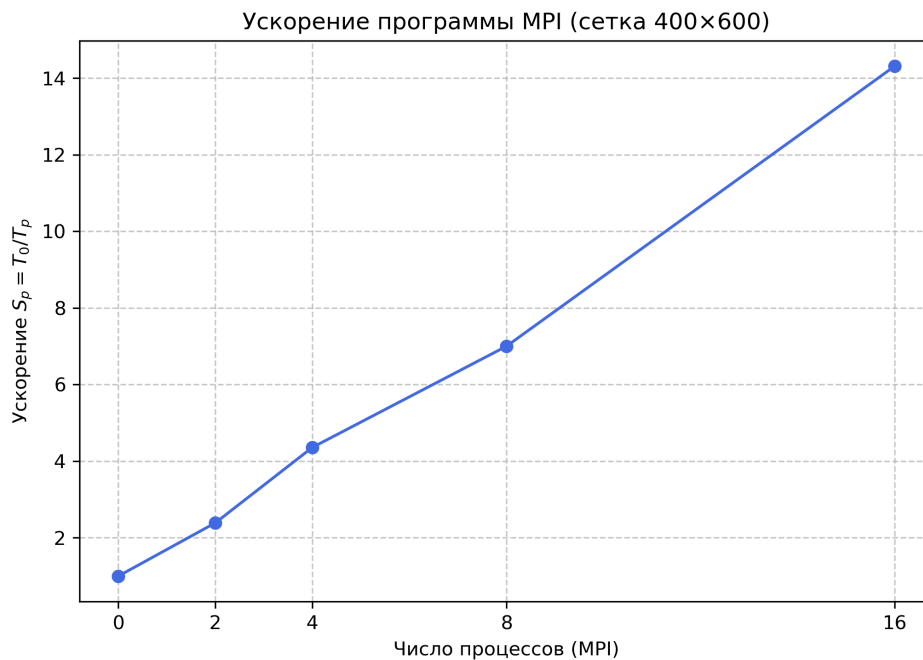


Рис. 2: Ускорение программы MPI для сетки  $400 \times 600$



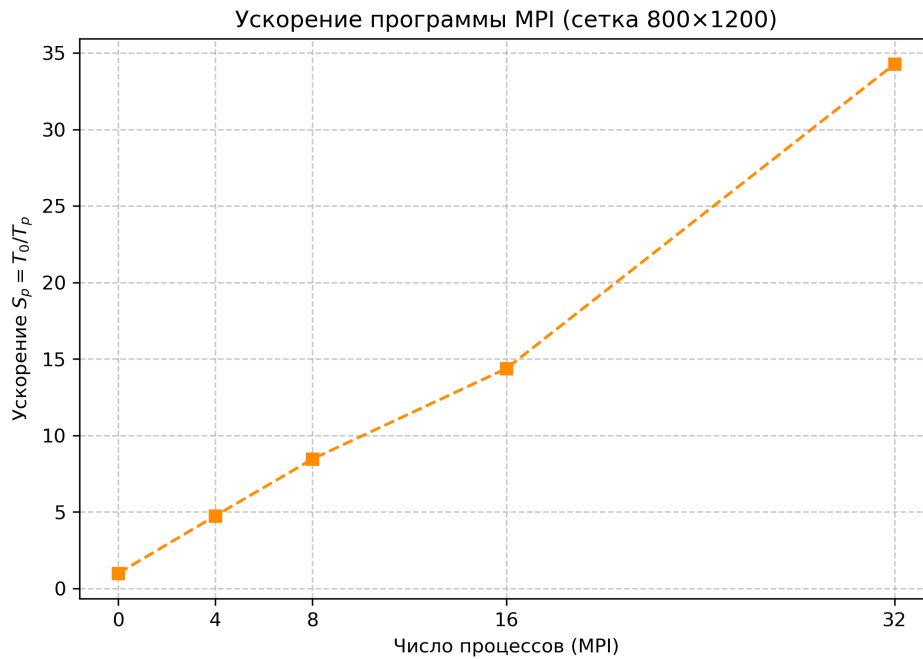


Рис. 3: Ускорение программы MPI для сетки  $800 \times 1200$

Из графиков видно, что с увеличением числа процессов ускорение значительно возрастает. Максимальное ускорение составило 34.27 при 32 процессах на сетке  $800 \times 1200$ .

## 7.6 Графики сходимости



Рис. 4: Сходимость решения для сетки  $800 \times 1200$

## 8 Заключение

В ходе работы была разработана и реализована параллельная версии программы для решения двумерной задачи Дирихле для уравнения Пуассона в криволинейной области (остроугольном треугольнике) методом фиктивных областей.

Основные результаты работы:

1. Разработана эффективная параллельная версия программы с использованием технологии MPI
2. Проведено тестирование программы на различных размерах сеток и количестве потоков

3. Получено ускорение вычислений до 34.27 раз на сетке  $800 \times 1200$  при использовании 32 MPI процессов
4. Продемонстрирована зависимость эффективности параллелизма от размера решаемой задачи

Полученные результаты подтверждают целесообразность использования MPI для ускорения итерационных методов решения сеточных уравнений.