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

Отчет о практической работе по курсу

Суперкомпьютерное моделирование и технологии

Студент:
Шевляков И.А. 619/2

Москва, 2024

Содержание

1 Постановка задачи

Область $\mathcal{D}$ - трапеция с вершинами в точках (0, 0), (0, 3), (2, 3), (3, 0). В области $\mathcal{D}$ решается уравнение Пуассона:

$-\Delta u = f(x, y)$

С оператором Лапласа

$-\Delta u = \frac{d^2u}{dx^2} + \frac{d^2u}{dy^2}$

$f(x, y) = 1$ в $\mathcal{D}$

$u(x, y) = 0$ на границе $\mathcal{D}$

Цель: найти u, являющееся численным решением уравнения Пуассона.

2 Численные методы

Для решения задачи будет использован метод фиктивных областей. В качестве фиктивной области будет прямоугольник П с вершинами в точках (0, 0), (0, 3), (3, 3), (3, 0), который содержит в себе область $\mathcal{D}$.
Фиктивная область: $\hat{D} = \prod \setminus \overline{D}$

Задача Дирихле сводится к:

$$-\frac{\partial}{\partial x}(k(x,y)\frac{\partial v}{\partial x}) - \frac{\partial}{\partial y}(k(x,y)\frac{\partial v}{\partial y}) = F(x,y) \tag{1}$$

где $\begin{cases} k(x,y) = 1 & (x,y) \in D \\ k(x,y) = \dfrac{1}{\varepsilon} & (x,y) \in \hat{D} \end{cases}$
И

$\begin{cases} F(x,y) = 1 & (x,y) \in D \\ F(x,y) = 0 & (x,y) \in \hat{D} \end{cases}$
v(x,y) равномерно приближает решение u(x,y) для некоторого $C > 0$:

$$\max_{(x,y)\in\overline{D}} |v(x,y) - u(x,y)| < C\varepsilon \tag{2}$$

3 Разностная схема

Данную задачу можно решить методом конечных разностей. В замыкании П определяется сетка $\omega_h = \omega_1 \times \omega_2$

$$\omega_1 = \{x_i = A_i + ih_1, i = \overline{0, M}\}$$
$$\omega_2 = \{y_i = A_2 + jh_2, i = \overline{0, N}\}$$

Где

$$h_1 = \frac{B_1 - A_1}{M} \quad h_2 = \frac{B_2 - A_2}{N} \tag{3}$$

Уравнение (1) аппроксимируется:

$$-\frac{1}{h_1}(a_{i+1j}\frac{\omega_{i+1j} - \omega_{ij}}{h_1} - a_{ij}\frac{\omega_{ij} - \omega_{i-1j}}{h_1}) - \frac{1}{h_2}(b_{ij+1}\frac{\omega ij + 1 - \omega_{ij}}{h_2} - b_{ij}\frac{\omega_{ij} - \omega_{ij-1}}{h_2}) = F_{ij} \tag{4}$$

$$i = \overline{1, M-1} \quad j = \overline{1, N-1} \tag{5}$$

Левая часть:

$$a_{ij} = \frac{1}{h_2}\int_{j-\frac{1}{2}}^{j+\frac{1}{2}} k(x_{i-\frac{1}{2}}, t)dt \tag{6}$$

$$b_{ij} = \frac{1}{h_1}\int_{i-\frac{1}{2}}^{i+\frac{1}{2}} k(t, y_{j-\frac{1}{2}}, t)dt \tag{7}$$

Правая часть:

$$F_{ij} = \frac{1}{h_1 * h_2}\iint_{\Pi_{ij}} F(x, y)dxdy; \quad \prod_{ij} = \{(x, y) : x_{i-\frac{1}{2}} \leq x \leq x_{i+\frac{1}{2}}, y_{j-\frac{1}{2}} \leq y \leq y_{j+\frac{1}{2}}\} \tag{8}$$

Для решения исходной задачи нужно решить полученное СЛАУ итерационным методом, например, при помощи метода скорейшего спуска.

4 Алгоритм решения

4.1 Последовательный

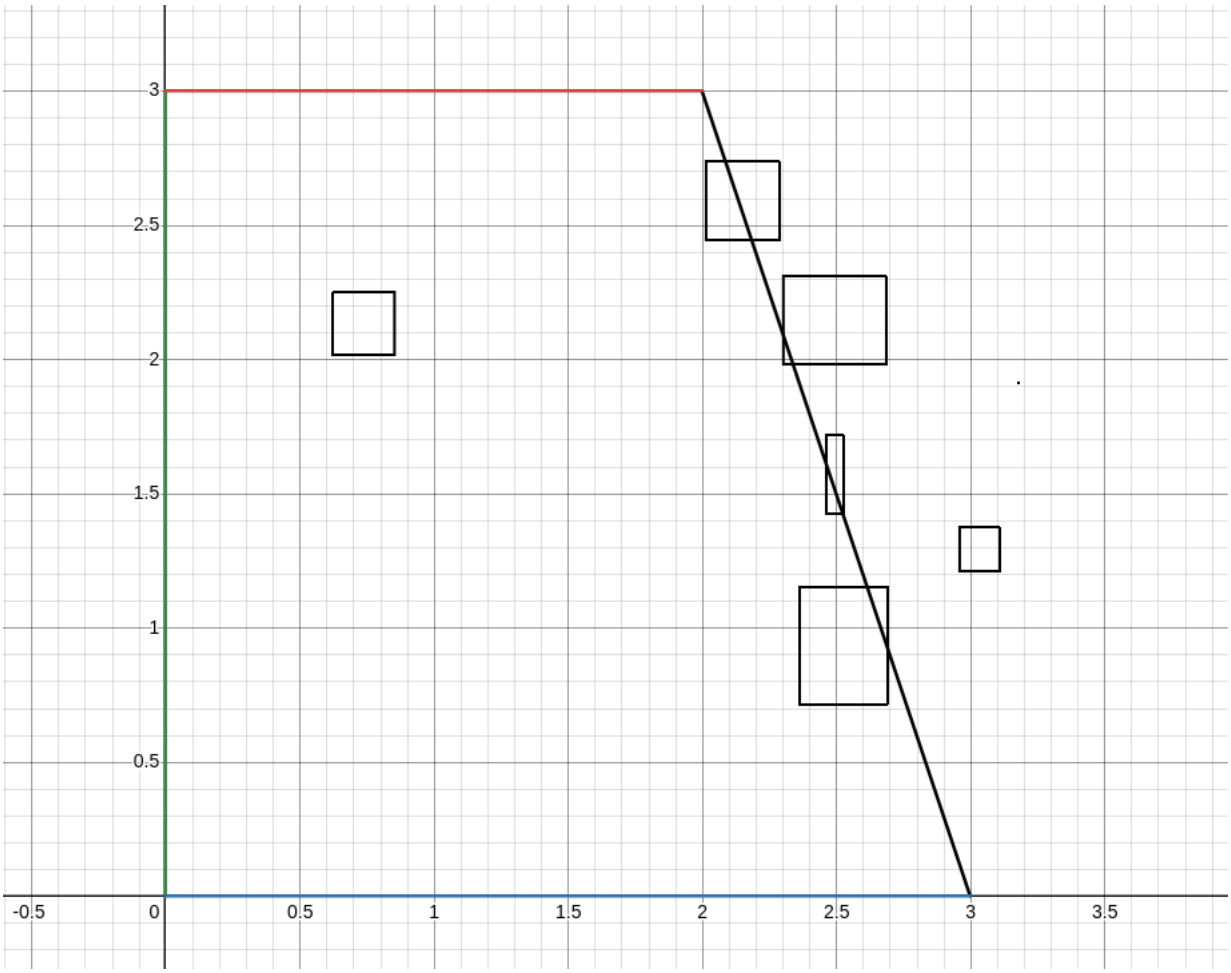
Для численного решения задачи необходимо вычислить значения коэффициентов a_{ij}, b_{ij}, F_{ij}

Значения a_{ij}, b_{ij} считаются аналитически, если соответствующие им отрезки $(x_{i-\frac{1}{2}}, y_{j-\frac{1}{2}})$-$(x_{i-\frac{1}{2}}, y_{j+\frac{1}{2}})$ и $(x_{i-\frac{1}{2}}, y_{j-\frac{1}{2}})$-$(x_{i+\frac{1}{2}}, y_{j-\frac{1}{2}})$ выходят за границу $\mathcal{D}$

F_{ij} считается как $h_1 * h_2 * S_{ij}$, где $S_{ij} = mes(\prod_{ij} \cap D)$

Прямоугольник в текущей итерации может занять 6 позиций относительно трапеции:

- все вершины внутри трапеции - 1 вариант
- внутри трапеции 3 вершины - 1 вариант
- внутри трапеции 2 вершины - 2 варианта
- внутри трапеции 1 вершина - 1 вариант
- никакая вершина не в трапеции - 1 вариант



4.2 Параллельный

Для параллельных вычислений используется OpenMP, а именно:
1) директива pragma omp parallel for для обычного цикла.
2) директива pragma omp parallel for collapse(2) для вложенного цикла

Помимо OpenMP для параллельности использовался механизм MPI: сетка разбивалась на подсетки (в зависимости от числа процессов), на которых параллельно решалась поставленная задача. Использовались директивы MPI_* библиотеки mpi.h.

Также была был выполнен эксперимент с реализацией MPI+GPU. В этом случае поверх кода MPI использовались следующие директвы:

- pragma acc data copy - копирование на устройство.
- pragma acc parallel loop - при распараллеливании циклов. Если цикл двойной - использался суффикс collapse(2).
- pragma acc update self - обновление хоста.
- pragma acc update device - обновление на устройстве.

5 Численные эксперименты

Вычисления проводились на комплексе Polus.
В качестве погрешности для останова был взял параметр $\delta = 1e-7$. Для теста на сетке MxN = 160x180 - параметр $\delta = 5 * (1e-8)$
Параметр $\varepsilon = \max(h_1, h_2)^2$
Рассчеты были проведены на сгущающихся сетках (M,N) со значениями (10,10), (20,20), (40,40), (80, 90), (160, 180)

Число нитей	Размер сетки M x N	Число итераций	Время (сек)	Ускорение
1	10x10	1578	0.0025679	1
1	20x20	22336	0.156556	1
1	40x40	248324	11.3963	1
4	40x40	248324	6.34567	1.795917
16	40x40	248324	5.90423	1.930192

Дла следующих таблиц параметр $\delta = 9e - 8$

OMP:

Число нитей	Размер сетки M x N	Число итераций	Время (сек)	Ускорение
1	80x90	2196576	897.983	1
2	80x90	2196576	1380.891	0.650
4	80x90	2196576	495.589	1.812
8	80x90	2196576	273.397	3.284
16	80x90	2196576	164.081	5.473
1	160x180	377170	601.213	1
4	160x180	377170	324.259	3.684052
8	160x180	377170	168.863	6.575054
16	160x180	377170	90.447	6.647
32	160x180	377170	130.653	4.601

MPI:

Количество MPI процессов	Размер сетки M x N	Число итераций	Время (сек)	Ускорение
1	80x90	2196576	1236.829	1
2	80x90	2196576	637.877	1.938
4	80x90	2196576	329.42	3.754
1	160x180	377170	854.711	1
2	160x180	377170	324.259	2.635
4	160x180	377170	221.473	3.859

OMP+MPI:

Количество MPI процессов	Число нитей	Размер сетки M x N	Число итераций	Время (сек)	Ускорение
1	1	80x90	2196576	897.983	1
2	1	80x90	2196576	1528.171	0.587
2	2	80x90	2196576	533.754	1.682
2	4	80x90	2196576	306.260	2.932
2	8	80x90	2196576	265.169	3.386
1	1	160x180	377170	601.213	1
4	1	160x180	377170	347.509	1.730
4	2	160x180	377170	175.196	3.431
4	4	160x180	377170	221.473	2.714
4	8	160x180	-	-	-

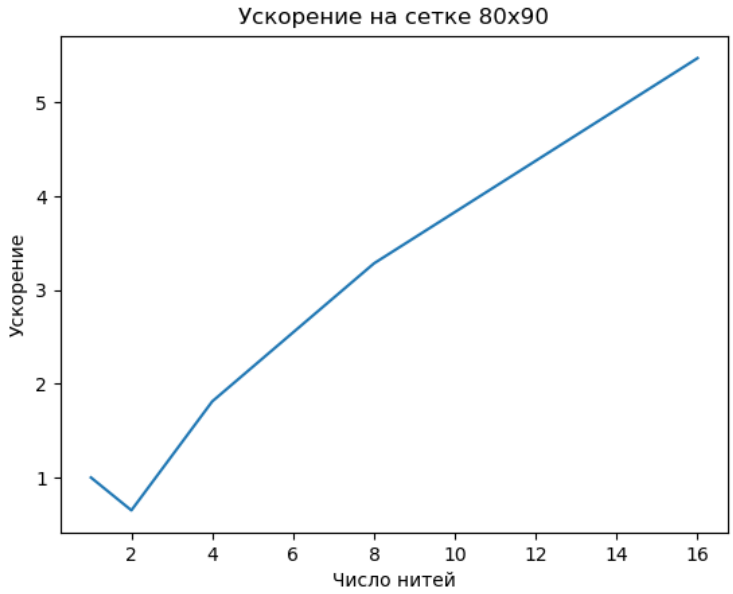
По неизвестным причинам на 4 процессах и 8 нитях происходит слишком сильное замедление работы программы, из-за чего выполнение на polus идет >3 часов. Было протестировано на компиляторах mpic++, mpiCC, mpicxx, mpixlC.

MPI+GPU:

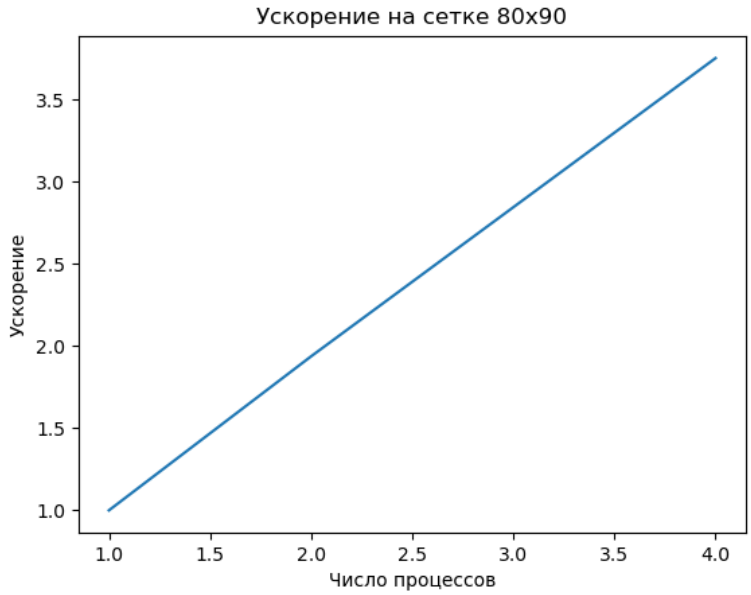
Для реализации MPI+GPU использовался пакет OpenACC. Тесты проводились на решетке 160x180.

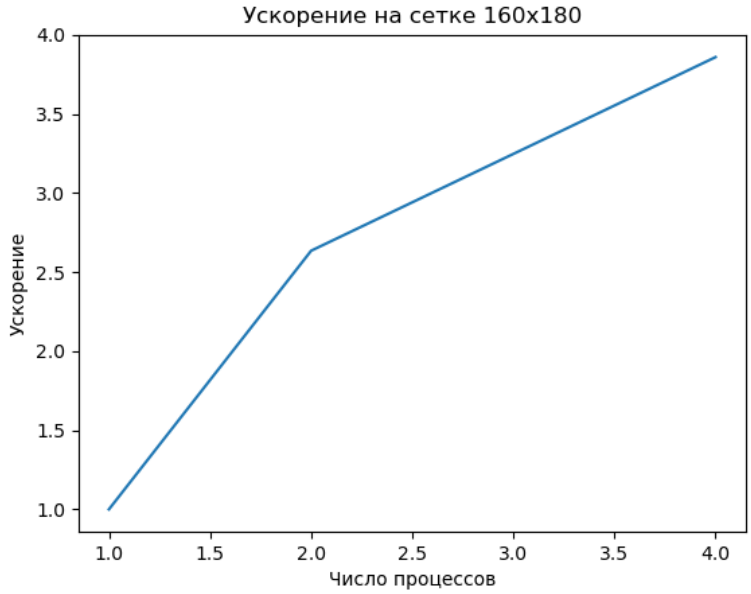
MPI процессы	GPU	Размер сетки M x N	Время (сек)
1	1	160x180	177.753
2	2	160x180	133.371

Графики, отображающие зависимость ускорения от числа нитей OMP:

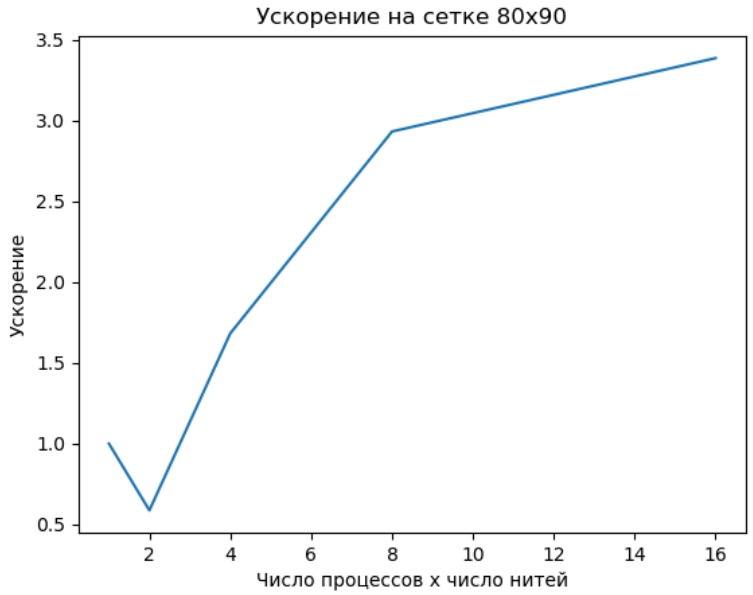


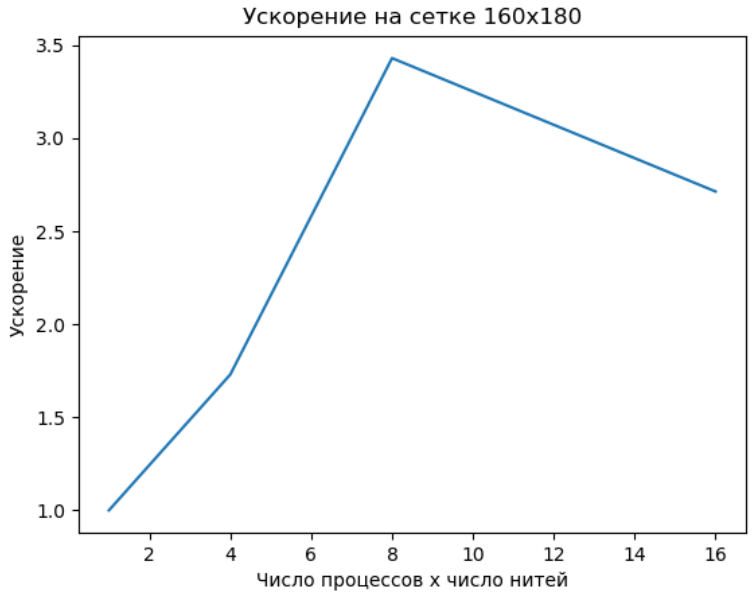
Графики, отображающие зависимость ускорения от числа процессов MPI:



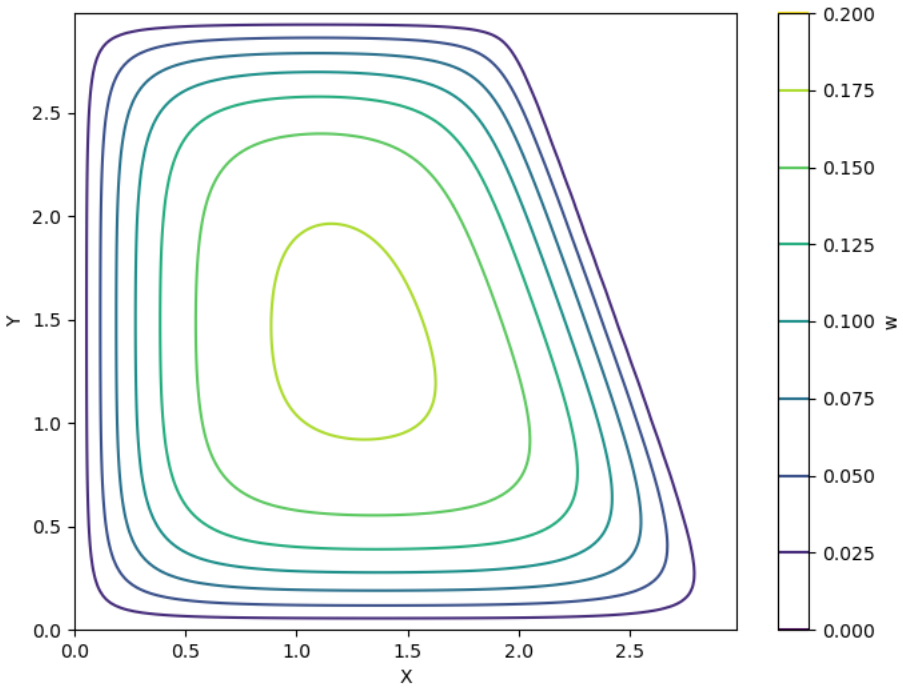


Графики, отображающие зависимость ускорения от числа процессов MPI x числа нитей OpenMP:





Визуализация полученного решения (в виде изолиний):



Из результатов видно, что реализация через MPI+GPU позволяет получить наименьшее время выполнения.