Отчёт

Мерзлякова Марина Андреевна, группа 622

Москва, 2025

1 Математическая постановка задачи

1.1 Задача

Рассматривается трёхмерная область

$$\Omega = \{(x, y, z) \mid 0 \le x \le L_x,\ 0 \le y \le L_y,\ 0 \le z \le L_z\},$$

и задача Коши для уравнения

$$\frac{\partial^2 u}{\partial t^2} = a^2 \Delta u, \quad (x, y, z) \in \Omega,\ 0 < t \le T,$$

с начальными условиями

$$u(x, y, z, 0) = \varphi(x, y, z), \qquad \frac{\partial u}{\partial t}(x, y, z, 0) = 0.$$

периодическими граничными условиями по x и по y:

$$u(0, y, z, t) = u(L_x, y, z, t), \qquad u_x(0, y, z, t) = u_x(L_x, y, z, t),$$

$$u(x, 0, z, t) = u(x, L_y, z, t), \qquad u_y(x, 0, z, t) = u_y(x, L_y, z, t).$$

и однородными граничными условиями 1 рода о направлению z:

$$u(x, y, 0, t) = 0, \qquad u(x, y, L_z, t) = 0.$$

1.2 Аналитическое решение

Аналитическое решение имеет вид:

$$u(x, y, z, t) = \sin\left(\frac{2\pi}{L_x}x + 3\pi\right) \sin\left(\frac{2\pi}{L_y}y + 2\pi\right) \sin\left(\frac{\pi}{L_z}z\right) \cos(a_t t + \pi)\,,$$

где

$$a_t = \pi$$

а параметр a^2 определяется выражением

$$\frac{1}{\frac{4}{L_x^2} + \frac{4}{L_y^2} + \frac{1}{L_z^2}}.$$

2 Цель работы

В рамках работы для заданного аналитического решения $u_{\text{analytical}}(x, y, z, t)$ и выбранных параметров сетки ($N_x = N_y = N_z = N$, шаг по времени τ, конечный момент T) требуется:

- реализовать численный алгоритм решения задачи;
- реализовать 4 варианта параллельной реализации — OpenMP, MPI, гибридной MPI + OpenMP, а также MPI + CUDA;
- на последнем временном слое вычислить максимальную по области погрешность

$$\delta = \max_{i,j,k} \left| u^n_{i,j,k} - u_{\text{analytical}}(x_i, y_j, z_k, t^n) \right|;$$

- измерить время работы каждой версии программы и на основе этих данных оценить ускорение и эффективность параллельных реализаций.

3 Численный метод решения

3.1 Построение сетки

Вводим равномерную пространственную сетку

$$x_i = ih_x, \quad y_j = jh_y, \quad z_k = kh_z, \qquad i, j, k = 0, \ldots, N,$$

где

$$h_x = \frac{L_x}{N_x}, \quad h_y = \frac{L_y}{N_y}, \quad h_z = \frac{L_z}{N_z - 1},$$

и временную сетку

$$t^n = n\tau, \quad n = 0, \ldots, K, \quad T = K\tau.$$

Обозначим через $u^n_{i,j,k}$ приближённое значение решения $u(x_i, y_j, z_k, t^n)$.

3.2 Разностная схема

Для аппроксимации уравнения $u_{tt} = a^2 \Delta u$ используем явную центрально-разностную схему второго порядка по времени и пространству:

$$\frac{u^{n+1}_{i,j,k} - 2u^n_{i,j,k} + u^{n-1}_{i,j,k}}{\tau^2} = a^2 \Delta_h u^n_{i,j,k}, \qquad n = 1, \ldots, K-1,$$

где Δ_h — семиточечный разностный аналог оператора Лапласа:

$$\Delta_h u^n_{i,j,k} = \frac{u^n_{i-1,j,k} - 2u^n_{i,j,k} + u^n_{i+1,j,k}}{h_x^2} + \frac{u^n_{i,j-1,k} - 2u^n_{i,j,k} + u^n_{i,j+1,k}}{h_y^2} + \\ + \frac{u^n_{i,j,k-1} - 2u^n_{i,j,k} + u^n_{i,j,k+1}}{h_z^2}.$$

Начальное условие на нулевом слое выписывается из $u(x, y, z, 0) = \varphi(x, y, z)$:

$$u^0_{i,j,k} = \varphi(x_i, y_j, z_k).$$

Второе начальное условие $u_t(x,y,z,t)\big|_{t=0} = 0$ мы аппроксимируем центральной разностью по времени:

$$\left.\frac{\partial u}{\partial t}\right|_{t=0} \approx \frac{u^1_{i,j,k} - u^0_{i,j,k}}{\tau} = 0.$$

Для более точной аппроксимации второго порядка раскладываем это решение в ряд Тейлора по времени в окрестности $t = 0$:

$$u(x_i, y_j, z_k, \tau) = u(x_i, y_j, z_k, 0) + \tau u_t(x_i, y_j, z_k, 0) + \frac{\tau^2}{2} u_{tt}(x_i, y_j, z_k, 0) + O(\tau^3).$$

С учётом начального условия $u_t(\cdot, 0) = 0$ и уравнения $u_{tt} = a^2 \Delta u$ получаем

$$u(x_i, y_j, z_k, \tau) = \varphi(x_i, y_j, z_k) + \frac{a^2\tau^2}{2} \Delta\varphi(x_i, y_j, z_k) + O(\tau^3).$$

Заменяя непрерывный оператор Лапласа его разностным аналогом Δ_h, получаем формулу для первого слоя:

$$u^1_{i,j,k} = u^0_{i,j,k} + \frac{a^2\tau^2}{2} \Delta_h\varphi(x_i, y_j, z_k).$$

Таким образом, шаг по времени выглядит следующим образом:

- по формуле выше вычисляется первый и нулевой временной слой;
- для по явной схеме вычисляется u^{n+1} через u^n и u^{n-1}.

3.3 Аппроксимация граничных условий

Для периодических условий по x и y на сетке:

$$u^n_{0,j,k} = u^n_{N,j,k}, \qquad u^n_{1,j,k} = u^n_{N+1,j,k},$$

$$u^n_{i,0,k} = u^n_{i,N,k}, \qquad u^n_{i,1,k} = u^n_{i,N+1,k},$$

По z используются условия первого рода, поэтому на границах $z = 0$ и $z = L_z$ задаётся

$$u^n_{i,j,0} = 0, \qquad u^n_{i,j,N} = 0$$

3.4 Условие устойчивости

Для явной схемы вида

$$\frac{u^{n+1}_{i,j,k} - 2u^n_{i,j,k} + u^{n-1}_{i,j,k}}{\tau^2} = a^2 \Delta_h u^n_{i,j,k}$$

существует следующее условие устойчивости:

$$a^2\tau^2 \left(\frac{1}{h_x^2} + \frac{1}{h_y^2} + \frac{1}{h_z^2}\right) \le 1.$$

Отсюда получаем допустимый шаг по времени

$$\tau \le \frac{1}{a\sqrt{\dfrac{1}{h_x^2} + \dfrac{1}{h_y^2} + \dfrac{1}{h_z^2}}}.$$

4 Результаты численных экспериментов

Таблица 1: Результаты для OpenMP-реализации при $L = 1.0$

№	Число нитей N_p	Размер сетки N^3	Время T, с	Ускорение S	Погрешность δ
1	1	128^3	4.223	1	1.337575e-05
2	2	128^3	2.202	1.92	1.337575e-05
3	4	128^3	1.496	2.82	1.337575e-05
4	8	128^3	0.970	4.35	1.337575e-05
5	16	128^3	0.922	4.58	1.337575e-05
6	2	256^3	19.127	1	1.001029e-06
7	4	256^3	10.747	1.77	1.001029e-06
8	8	256^3	7.725	2.48	1.001029e-06
9	16	256^3	3.686	5.19	1.001029e-06
10	32	256^3	2.936	6.51	1.001029e-06

Таблица 2: Результаты для MPI-реализации при $L = 1.0$

№	Число процессов N_p	Размер сетки N^3	Время T, с	Ускорение S	Погрешность δ
1	1	128^3	7.16148	1	1.53425e-05
2	4	128^3	1.86578	3.84	1.53425e-05
3	8	128^3	1.10141	6.50	1.53425e-05
4	16	128^3	0.704326	10.17	1.53425e-05
5	32	128^3	0.618401	11.58	1.53425e-05
6	1	256^3	57.6828	1	1.15016e-06
7	4	256^3	15.4293	3.74	1.15016e-06
8	8	256^3	8.07737	7.14	1.15016e-06
9	16	256^3	4.19706	13.74	1.15016e-06
10	32	256^3	3.22447	17.89	1.15016e-06

Таблица 3: Результаты для MPI+OpenMP реализации при $L = 1.0$

№	MPI-процессы N_p	Нити N_t	Размер N^3	Время T, с	Ускорение S	Погрешность δ
1	4	1	128^3	2.30834	1	1.53425e-05
2	4	2	128^3	1.21325	1.90	1.53425e-05
3	4	4	128^3	0.709328	3.25	1.53425e-05
4	4	8	128^3	0.52708	4.38	1.53425e-05
5	8	1	256^3	9.79488	1	1.15016e-06
6	8	2	256^3	5.70424	1.72	1.15016e-06
7	8	4	256^3	3.44946	2.84	1.15016e-06
8	8	8	256^3	2.40149	4.08	1.15016e-06

Таблица 4: Результаты для OpenMP-реализации при $L = 3.14$

№	Число нитей N_p	Размер сетки N^3	Время T, с	Ускорение S	Погрешность δ
1	1	128^3	4.381	1	1.337575e-05
2	2	128^3	2.232	1.96	1.337575e-05
3	4	128^3	1.348	3.25	1.337575e-05
4	8	128^3	0.838	5.23	1.337575e-05
5	16	128^3	0.490	8.94	1.337575e-05
6	2	256^3	21.653	1	1.001029e-06
7	4	256^3	13.437	1.61	1.001029e-06
8	8	256^3	8.781	2.47	1.001029e-06
9	16	256^3	7.047	3.07	1.001029e-06
10	32	256^3	3.308	6.55	1.001029e-06

Таблица 5: Результаты для MPI-реализации при $L = 3.14$

№	Число процессов N_p	Размер сетки N^3	Время T, с	Ускорение S	Погрешность δ
1	1	128^3	7.08085	1	1.53425e-05
2	4	128^3	1.90763	3.71	1.53425e-05
3	8	128^3	1.07033	6.62	1.53425e-05
4	16	128^3	0.545937	12.97	1.53425e-05
5	32	128^3	0.516136	13.72	1.53425e-05
6	1	256^3	56.8984	1	1.15016e-06
7	4	256^3	14.6101	3.89	1.15016e-06
8	8	256^3	7.59559	7.49	1.15016e-06
9	16	256^3	4.18401	13.60	1.15016e-06
10	32	256^3	3.60228	15.80	1.15016e-06

Таблица 6: Результаты для MPI+OpenMP реализации при $L = 3.14$

№	MPI-процессы N_p	Нити N_t	Размер N^3	Время T, с	Ускорение S	Погрешность δ
1	4	1	128^3	2.35157	1	1.53425e-05
2	4	2	128^3	1.34403	1.74	1.53425e-05
3	4	4	128^3	0.86219	2.73	1.53425e-05
4	4	8	128^3	0.570748	4.12	1.53425e-05
5	8	1	256^3	10.6182	1	1.15016e-06
6	8	2	256^3	6.20969	1.71	1.15016e-06
7	8	4	256^3	3.45764	3.07	1.15016e-06
8	8	8	256^3	2.3317	4.55	1.15016e-06

Таблица 7: Сравнение времени работы методов для $L = 1.0$ и сетки 512^3

Метод	Время T, с	Ускорение S	Погрешность δ
seq	285.358	1	6.7166e-08
omp $t = 8$	37.741	7.56	6.7166e-08
mpi $np = 20$	27.2159	10.48	6.7166e-08
omp + mpi 8×20	13.5924	20.99	6.7166e-08
gpu 1, $np = 1$	2.4706	115.50	6.7166e-08
gpu 2, $np = 2$	1.353	210.91	6.7166e-08

Таблица 8: Сравнение основных временных характеристик для GPU

Метод	Total, s	Init, s	Final (- init), s	Compute K, s	Copy H2D, s	Copy D2H, s
gpu 1, $np = 1$	22.1341	19.6635	2.4706	0.205846	0.0123324	0.0173119
gpu 2, $np = 2$	11.3424	9.9894	1.3530	0.144885	0.0143254	0.0158661

Таблица 9: Сравнение временных характеристик коммуникации для GPU

Метод	Pack, s	Send, s	Wait (Block), s	Unpack, s	Total Comm, s
gpu 1, $np = 1$	0.412389	0.193572	0.677915	0.951123	2.23499
gpu 2, $np = 2$	0.213457	0.142389	0.356781	0.465296	1.17792

5 Анализ полученных результатов

По таблицам видно, что при переходе от последовательной реализации (в качестве последовательной реализации использовалась версия OpenMP с числом нитей = 1) к MPI+OpenMP, затем к MPI и гибридному MPI+OpenMP время расчёта последовательно уменьшается. Ускорение растёт с увеличением числа нитей и процессов, однако на некотором этапе рост становится более медленным: это связано с тем, что возрастает доля затрат на обмен данными, синхронизацию и обслуживание параллельных потоков, а также остаётся неизменной последовательная часть алгоритма. Таблицы для MPI+OpenMP полностью согласуются с таблицами для MPI и OpenMP, так как при одинаковых увеличениях числа потоков или процессов, ускорение, соответствующее этому увеличению, примерно одинаково растет во всех таблицах.

MPI+GPU–реализация демонстрирует наилучшее время выполнения по сравнению с CPU-вариантами. При этом детальная разбивка времён показывает, что основная часть затрат приходится на инициализацию задачи (подготовка сетки, распределение поддоменов, выделение и заполнение памяти на хосте и устройстве), тогда как собственно вычислительное ядро на GPU и операции копирования занимают существенно меньшую долю. Времена, связанные с коммуникациями между процессами и обменом граничных слоёв, остаются малы по сравнению с общим временем и практически не ограничивают производительность. Это означает, что сама численная схема на GPU правильно реализована и работает очень быстро, а основное проблемное место - именно последовательные этапы подготовки данных.

Также важным моментом является то, что для одинаковой сетки погрешность остается одинаковой независимо от метода реализации. То есть, распараллеливание, включая использование GPU, не повлияло на точность метода и изменило только время выполнения.

Таким образом, корректность всех параллельных версий, включая MPI+GPU, подтверждена взаимной проверкой результатов между собой и тем, что добавление параллельных механизмов влияет только на время выполнения, но не на точность вычислений.