МОСКОВСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
имени М.В. Ломоносова
Факультет вычислительной математики и кибернетики

Суперкомпьютерное моделирование и технологии

Отчет по заданию №4 «Задача для трёхмерного гиперболического уравнения в прямоугольном параллелепипеде»

Вариант №1

студент 628 группы
Гугучкин Егор Павлович

2022 год

1. Математическая постановка задачи

В трехмерной замкнутой области

$$\Omega = [0 \le x \le L_x] \times [0 \le y \le L_y] \times [0 \le z \le L_z]$$

для $0 \le t \le T$ требуется найти решение $u(x, y, z, t)$ уравнения в частных производных $\frac{\partial^2 u}{\partial t^2} = \Delta u$ с начальными условиями

$$u\,(t = 0) = \phi(x, y, z)$$
$$\frac{\partial u}{\partial t}\,(t = 0) = 0$$
$$u(0, y, z, t) = 0$$
$$u(L_x, y, z, t) = 0$$
$$u(x, 0, z, t) = 0$$
$$u(x, L_y, z, t) = 0$$
$$u(x, y, 0, t) = u(x, y, L_z, t)$$
$$u_z(x, y, 0, t) = u_z(x, y, L_z, t)$$

2. Численный метод решения задачи

Введем на Ω сетку $\omega_{h\tau} = \overline{\omega_h} \times \omega_\tau$, где $T = T_0$,

$$L_x = L_{x0}, L_y = L_{y0}, L_z = L_{z0},$$

$$\overline{\omega_h} = \{(x_i = ih_x, y_j = jh_y, z_k = kh_z), i, j, k = \overline{0, N}, h_x N = L_x, h_y N = L_y, h_z N = L_z\},$$

$$\omega_\tau = \{t_n = n\tau, n = \overline{0, K}, \tau K = T\}$$

Через ω_h обозначим множество внутренних, а через γ_h – множество граничных узлов сетки $\overline{\omega_h}$.

Для аппроксимации исходного уравнения воспользуемся следующей системой уравнений:

$$\frac{u_{i,j,k}^{n+1} - 2u_{i,j,k}^{n} + u_{i,j,k}^{n-1}}{\tau^2} = \Delta_h u^n, (x_i, y_i, z_i) \in \omega_h, n = \overline{1, K-1}$$

Здесь Δ_h – семиточечный разностный аналог оператора Лапласа:

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

Приведенная выше разностная схема является явной – значения $u^{n+1}_{i,j,k}$ на $(n+1)$-ом шаге можно явным образом выразить через значения на предыдущих слоях.

Для начала счета должны быть заданы значения:

$$u^0_{i,j,k}, u^1_{i,j,k}, (x_i, y_i, z_i) \in \omega_h:$$

$$u^0_{i,j,k} = \phi(x_i, y_i, z_i), (x_i, y_i, z_i) \in \omega_h$$

$$u^1_{i,j,k} = u^0_{i,j,k} + \frac{\tau^2}{2}\Delta_h \phi(x_i, y_i, z_i)$$

$$u^{n+1}_{i,j,0} = u^{n+1}_{i,j,N}$$

$$u^{n+1}_{i,j,1} = u^{n+1}_{i,j,N+1}$$

$$i, j, k = \overline{0, N}$$

3. Программная реализация

Реализовано две версии программы: последовательная и параллельная с использованием MPI +OpenMP. В качестве входных аргументов задаются следующие переменные: N – количество точек сетки вдоль одной оси, L – длина сетки вдоль одной оси, *filename* – имя выходного файла. На выходе программа выводит N, число MPI-процессов и погрешность полученного решения.

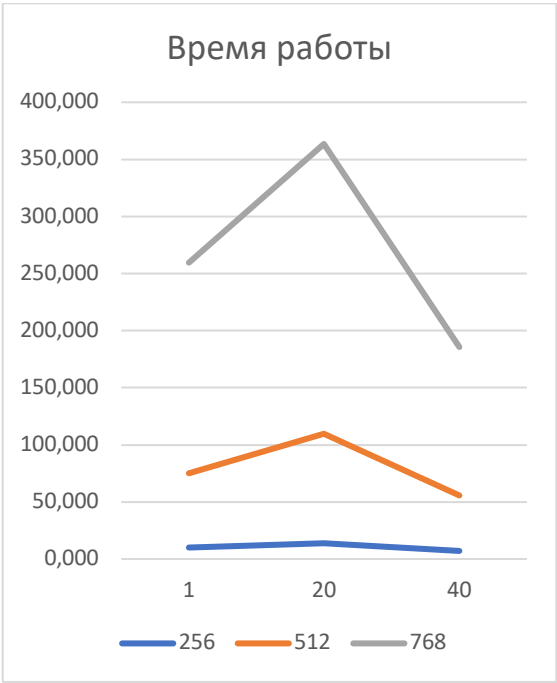
Параллельная версия программы выполнена следующим образом:

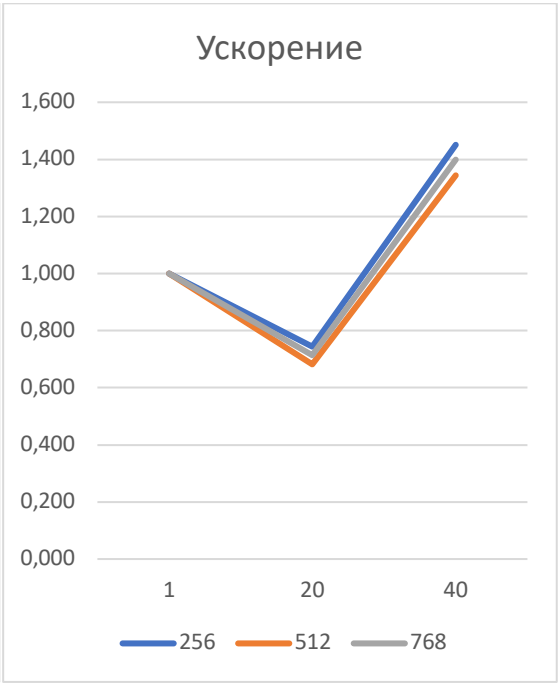
1. Сетка разделяется на *size* блоков, где *size* – число MPI-процессов. Каждому процессу выделяется свой блок.
2. Процессы находят ранги процессов-соседей и вычисляют координаты границ блока.
3. Процессы вычисляют u_0 и u_1 для своего блока.
4. Процессы вычисляют u_i и обмениваются граничными значениями.
5. Итоговая погрешность редуцируется с помощью оператора MPI_Reduce.

4. Результаты расчетов

MPI программа; L = 1; сравнение с 1-процессной последовательной программой

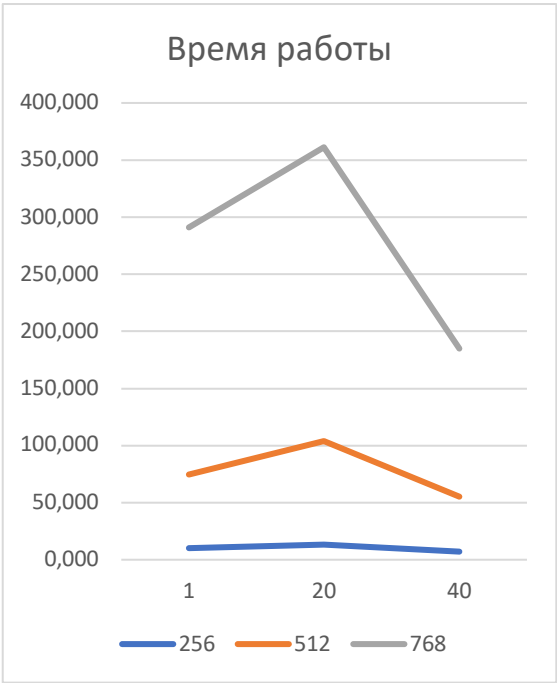
Число MPI-процессов N_p	Число точек сетки N^3	Время решения T	Ускорение S	Погрешность σ
1	256^3	10,281	1,000	5,96E-08
20	256^3	13,822	0,744	5,96E-08
40	256^3	7,086	1,451	5,96E-08
1	512^3	74,846	1,000	3,96E-09
20	512^3	109,696	0,682	3,96E-09
40	512^3	55,687	1,344	3,96E-09
1	768^3	259,663	1,000	4,53E-10
20	768^3	363.508	0,714	4,53E-10
40	768^3	185,656	1,399	4,53E-10

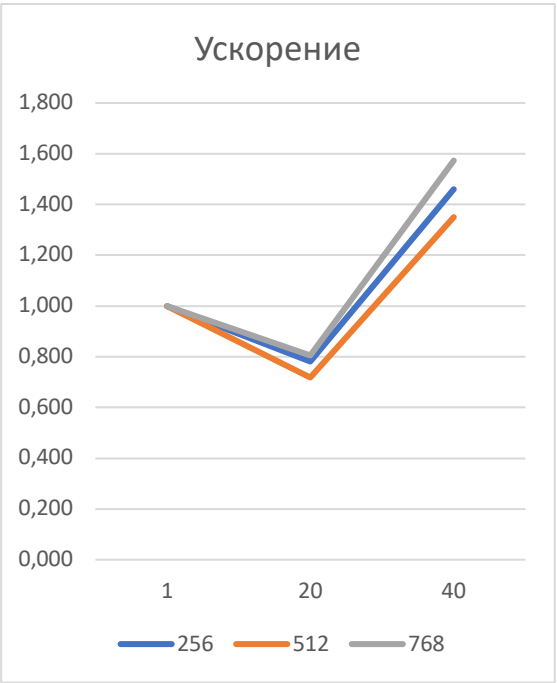




MPI программа; L = π; сравнение с 1-процессной последовательной программой

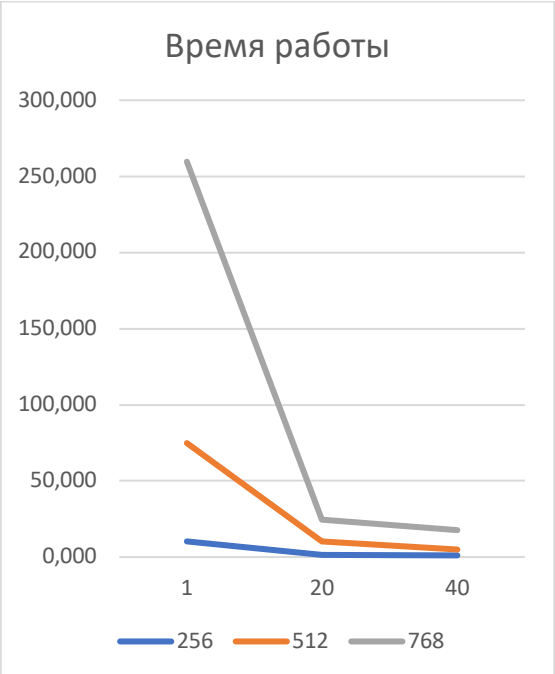
Число MPI-процессов N_p	Число точек сетки N^3	Время решения T	Ускорение S	Погрешность σ
1	256^3	10,332	1,000	7,38E-09
20	256^3	13,229	0,781	7,38E-09
40	256^3	7,078	1,460	7,38E-09
1	512^3	74,543	1,000	1,73E-09
20	512^3	103,890	0,718	1,73E-09
40	512^3	55,214	1,350	1,73E-09
1	768^3	290,853	1,000	6,87E-10
20	768^3	361,198	0,805	6,87E-10
40	768^3	184,890	1,573	6,87E-10

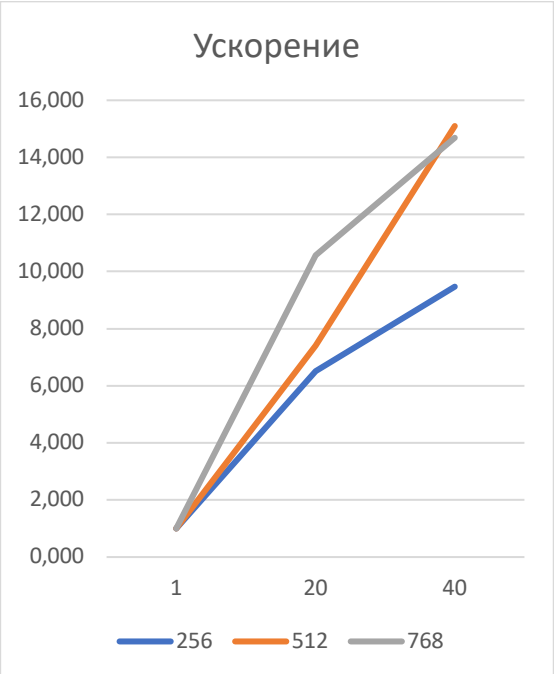




MPI+OpenMP (128 нитей) программа; L = 1; сравнение с 1-процессной последовательной программой

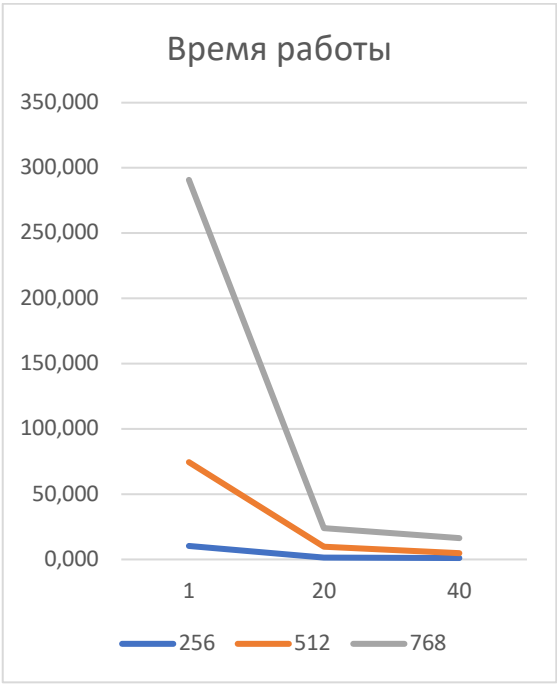
Число MPI-процессов N_p	Число точек сетки N^3	Время решения T	Ускорение S	Погрешность σ
1	256^3	10,281	1,000	5,96E-08
20	256^3	1,576	6,522	5,96E-08
40	256^3	1,085	9,472	5,96E-08
1	512^3	74,846	1,000	3,96E-09
20	512^3	10,109	7,404	3,96E-09
40	512^3	4,957	15,099	3,96E-09
1	768^3	259,663	1,000	4,53E-10
20	768^3	24,566	10,570	4,53E-10
40	768^3	17,679	14,688	4,53E-10

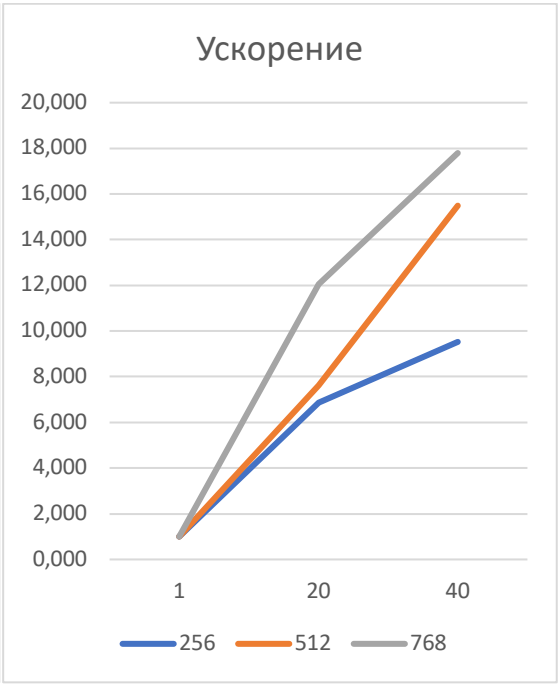




MPI+OpenMP (128 нитей) программа; L = π; сравнение с 1-процессной последовательной программой

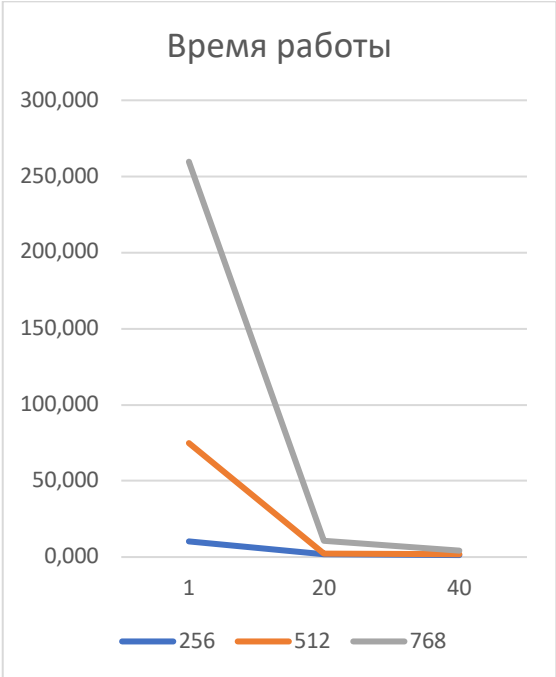
Число MPI-процессов N_p	Число точек сетки N^3	Время решения T	Ускорение S	Погрешность σ
1	256^3	10,332	1,000	7,38E-09
20	256^3	1,503	6,873	7,38E-09
40	256^3	1,085	9,526	7,38E-09
1	512^3	74,543	1,000	1,73E-09
20	512^3	9,773	7,627	1,73E-09
40	512^3	4,812	15,491	1,73E-09
1	768^3	290,853	1,000	6,87E-10
20	768^3	24,148	12,045	6,87E-10
40	768^3	16,344	17,795	6,87E-10

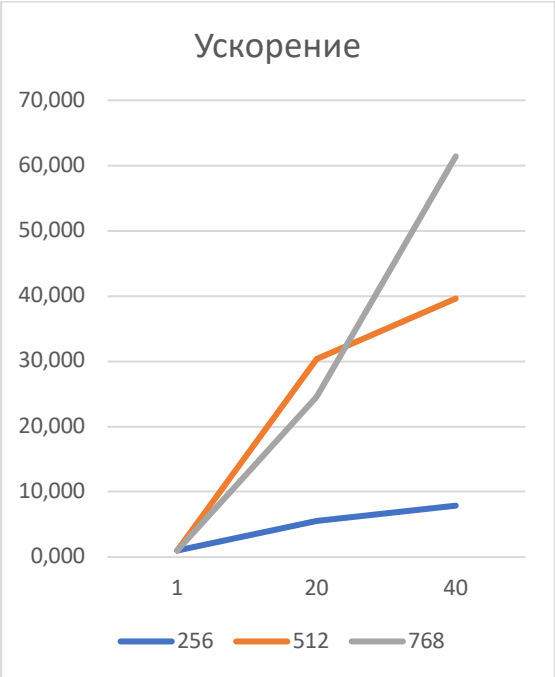




MPI+CUDA (1 GPU, 128 потоков) программа; L = 1; сравнение с 1-процессной последовательной программой

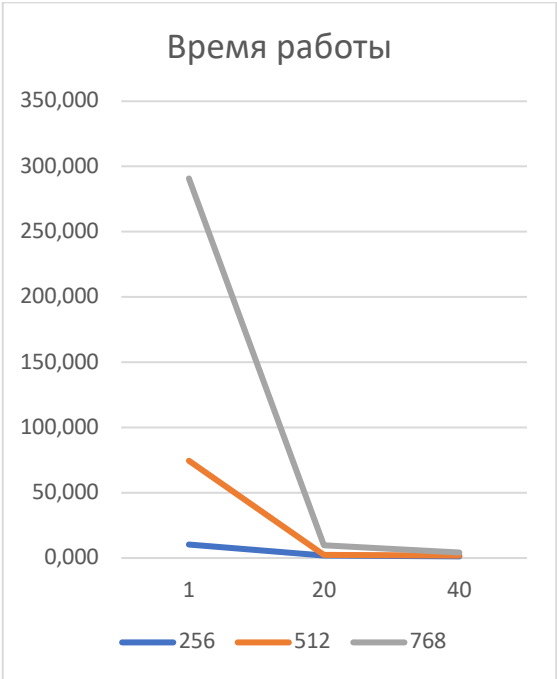
Число MPI-процессов N_p	Число точек сетки N^3	Время решения T	Ускорение S	Погрешность σ
1	256^3	10,281	1,000	5,96E-08
20	256^3	1,871	5,496	5,96E-08
40	256^3	1,305	7,877	5,96E-08
1	512^3	74,846	1,000	3,96E-09
20	512^3	2,469	30,308	3,96E-09
40	512^3	1,889	39,621	3,96E-09
1	768^3	259,663	1,000	4,53E-10
20	768^3	10,565	24,578	4,53E-10
40	768^3	4,228	61,412	4,53E-10

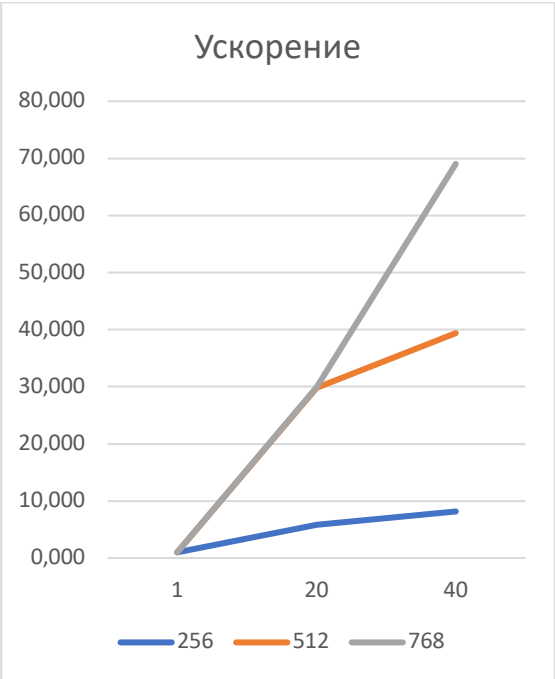




MPI+CUDA (1 GPU, 128 потоков) программа; L = π; сравнение с 1-процессной последовательной программой

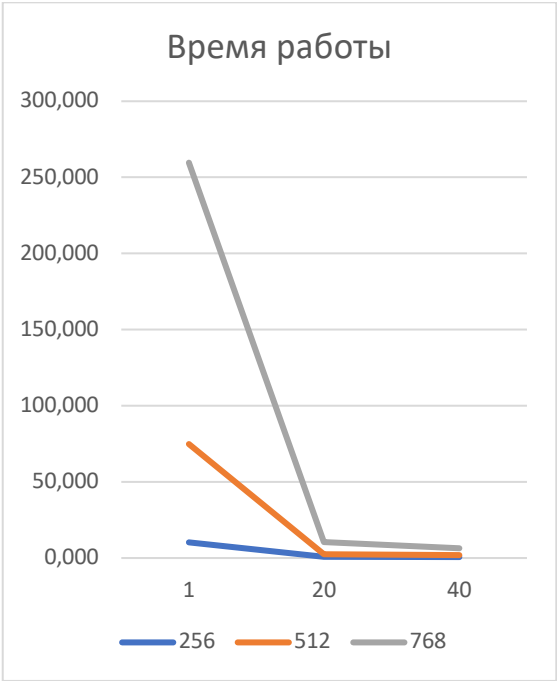
Число MPI-процессов N_p	Число точек сетки N^3	Время решения T	Ускорение S	Погрешность σ
1	256^3	10,332	1,000	10,332
20	256^3	1,786	5,785	1,786
40	256^3	1,267	8,153	1,267
1	512^3	74,543	1,000	74,543
20	512^3	2,504	29,773	2,504
40	512^3	1,893	39,388	1,893
1	768^3	290,853	1,000	290,853
20	768^3	9,712	29,948	9,712
40	768^3	4,213	69,030	4,213

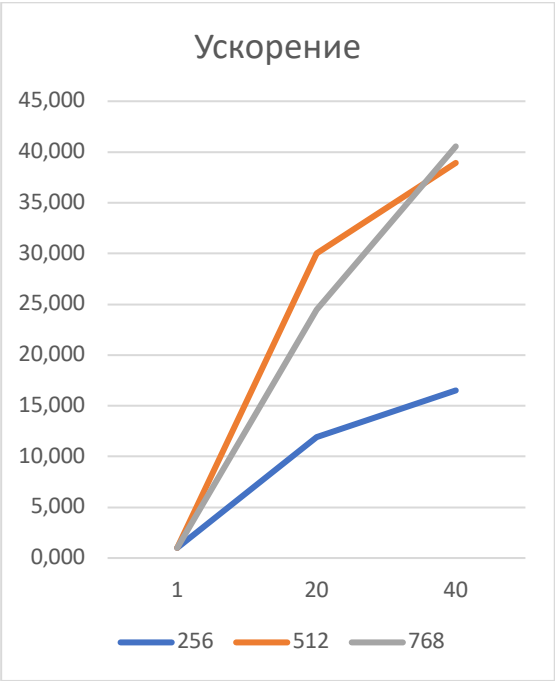




MPI+CUDA (2 GPU, 128 потоков) программа; L = 1; сравнение с 1-процессной последовательной программой

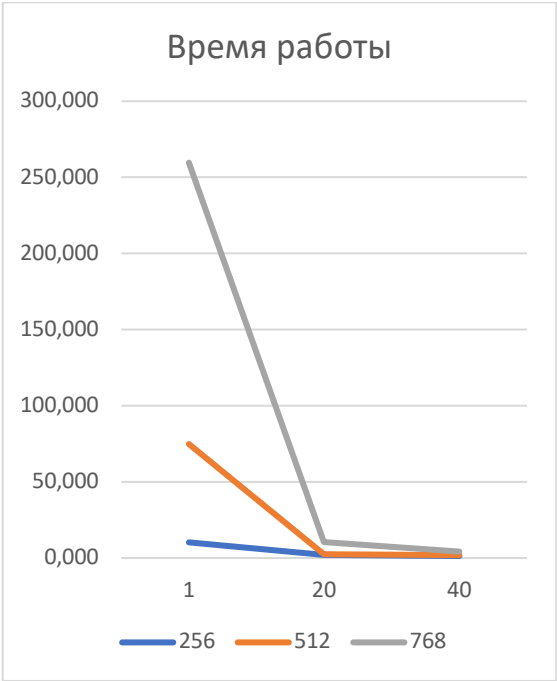
Число MPI-процессов N_p	Число точек сетки N^3	Время решения T	Ускорение S	Погрешность σ
1	256^3	10,281	1,000	5,96E-08
20	256^3	0,864	11,899	5,96E-08
40	256^3	0,623	16,510	5,96E-08
1	512^3	74,846	1,000	3,96E-09
20	512^3	2,492	30,037	3,96E-09
40	512^3	1,923	38,928	3,96E-09
1	768^3	259,663	1,000	4,53E-10
20	768^3	10,620	24,450	4,53E-10
40	768^3	6,403	40,553	4,53E-10

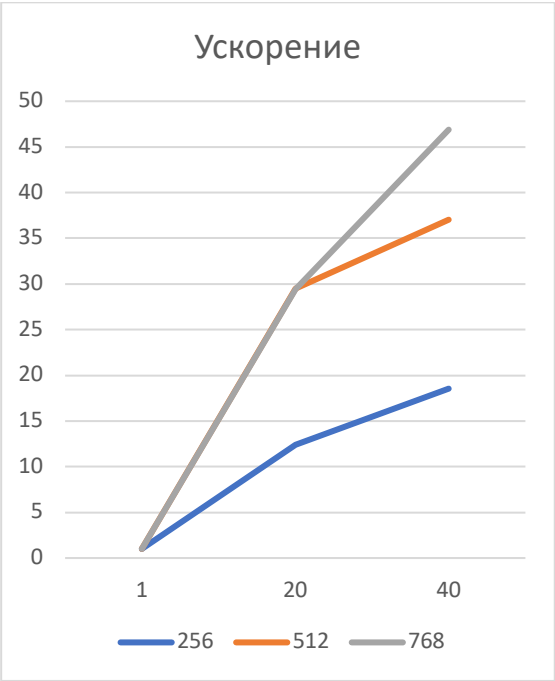




MPI+CUDA (2 GPU, 128 потоков) программа; L = π; сравнение с 1-процессной последовательной программой

Число MPI-процессов N_p	Число точек сетки N^3	Время решения T	Ускорение S	Погрешность σ
1	256^3	10,332	1,000	7,38E-09
20	256^3	0,835	12,368	7,38E-09
40	256^3	0,557	18,541	7,38E-09
1	512^3	74,543	1,000	1,73E-09
20	512^3	2,525	29,519	1,73E-09
40	512^3	2,013	37,036	1,73E-09
1	768^3	290,853	1,000	6,87E-10
20	768^3	9,890	29,408	6,87E-10
40	768^3	6,203	46,891	6,87E-10





5. Выводы

По полученным результатам, можно сделать вывод о том, что полученная реализация имеет высокий потенциал для распараллеливания.

Хочется отметить, что для MPI версии стоит использовать более 20-MPI процессов, из-за затрат на пересылку между процессами.

По результатам работы реализаций с помощью MPI-OpenMP (128 нитей) и MPI-CUDA (128 потоков) также можно сделать выводы о высоком потенциале распараллеливания. Тем не менее, MPI-CUDA имеет большее ускорение, по сравнению с MPI-OpenMP.

Также стоит отметить, что остается неясным каким образом стоит подбирать лучшее соотношение GPU/кол-во потоков для реализации MPI-CUDA.