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

Павлишин Кирилл Юрьевич
608 группа
15 вариант

Численное интегрирование многомерных функций методом Монте-Карло

Москва
2022

1 Введение

В качестве модельной задачи предлагается задача вычисления многомерного интеграла методом Монте-Карло.

Программная реализация должна быть выполнена на языке C или C++ с использованием библиотеки параллельного программирования MPI.

Требуется исследовать масштабируемость параллельной MPI-программы на параллельной вычислительной системе ВМК МГУ IBM Polus.

2 Математическая постановка задачи

Функция $f(x,\, y,\, z)$ — непрерывна в ограниченой замкнутой области $G \subset \mathbb{R}^3$. Требуется вычислить определённый интеграл:

$$I = \iiint_G f(x,\, y,\, z)\, dx\, dy\, dz \ ,$$

$$\text{где функция } f(x,\, y,\, z) = \sqrt{y^2 + z^2} \ ,$$

$$\text{область } G = \{(x,\, y,\, z) \,:\, 0 \leqslant x \leqslant 2 \,,\; y^2 + z^2 \leqslant 1\}$$

3 Численный метод решения задачи

Пусть область G ограниченна параллелепипедом

$$\Pi : \begin{cases} a_1 \leqslant x \leqslant b_1, \\ a_2 \leqslant y \leqslant b_2, \\ a_3 \leqslant z \leqslant b_3 \end{cases}$$

Рассмотрим функцию:

$$F(x,y,z) = \begin{cases} f(x,y,z)\,, & (x,y,z) \in G \\ 0\,, & (x,y,z) \notin G \end{cases}$$

Преобразуем искомый интеграл:

$$I = \iiint_G f(x,\, y,\, z)\, dx\, dy\, dz \;=\; \iiint_\Pi F(x,\, y,\, z)\, dx\, dy\, dz$$

Пусть $p_1(x_1,\, y_1,\, z_1), p_2(x_2,\, y_2,\, z_2), \ldots$ — случайные точки, равномерно распределённые в Π. Возьмём n таких случайных точек. В качестве приближённого значения интеграла предлагается использовать выражение:

$$I \;\approx\; |\Pi| \cdot \tfrac{1}{n} \sum_{i=1}^{n} F(p_i),$$

где $|\Pi|$ - объём параллелепипеда Π. $|\Pi| = (b_1 - a_1)(b_2 - a_2)(b_3 - a_3)$

4 Нахождение точного значения интеграла аналитически

$$I = \iiint_G \sqrt{y^2 + z^2}\, dx\, dy\, dz \;=\; \{x = x,\, y = r\cos\phi,\, z = r\sin\phi\} \;=\; \int_0^2 dx \int_0^{2\pi} d\phi \int_0^1 r^2 dr \;=\; \tfrac{4\pi}{3}$$

5 Описание программной реализации

Параллельная MPI-программа принимает на вход требуемую точность и генерирует случайные точки до тех пор, пока требуемая точность не будет достигнута. Программа вычисляет точность как модуль разности между приближённым значением, полученным методом Монте-Карло, и точным значением, вычислинным аналитически.

Программа считывает в качестве аргумента командной строки требуемую точность ϵ и выводит четыре числа:

- Посчитанное приближённое значение интеграла
- Ошибка посчитанного значения: модуль разности между приближённым и точным значениями интеграла
- Количество сгенерированных случайных точек
- Время работы программы в секундах

Время работы программы измеряется следующим образом. Каждый MPI-процесс, кроме процесса-мастера, измеряет своё время выполнения, затем среди полученных значений берётся максимум.

В моём варианте реализована парадигма «мастер–рабочие»: один из процессов («мастер») генерирует по очереди для каждого из остальных процессов («рабочих») n / (количество процессов$-$ 1) случайных точек и передаёт их. Все процессы–рабочие получают свой набор точек и вычисляют свою часть суммы. Затем в процессе-мастере вычисляется общая сумма с помощью операции редукции. После чего в «мастере» вычисляется ошибка (разность между посчитанным значением и точным значением, вычисленным аналитически). В случае если ошибка выше требуемой точности, которую подали на вход программе, то генерируются дополнительные n / (количество процессов $-$ 1) точек для каждого процесса-рабочего и расчёт продолжается.

6 Исследование мастшабируемости программы на системе Polus

Для корректного исследования мастшабируемости программы было выбрано $n = 16740$ и зерно для генератора случайных чисел, равное 14251, одинаковые для всевозможных комбинаций ϵ и числа MPI-процессов.

Таблица 1: Таблица с результатами расчётов для системы Polus

Точность ϵ	**Число MPI-процессов**	**Время работы программы (с)**	**Ускорение**	**Ошибка**
$3.0 \cdot 10^{-5}$	2	0.0271082	1	8.40508e-06
	4	0.0109136	2.483891	8.40508e-06
	16	0.00542482	4.997069	8.40508e-06
	32	0.00197079	13.754991	8.40508e-06
$5.0 \cdot 10^{-6}$	2	0.0482266	1	4.01898e-06
	4	0.0166731	2.8924795	4.01898e-06
	16	0.00475759	10.13677	4.01898e-06
	32	0.00371255	12.990155	4.01898e-06
$1.5 \cdot 10^{-6}$	2	1.6383	1	1.15689e-06
	4	0.547051	2.99478476	1.15689e-06
	16	0.116177	14.1017585	1.15689e-06
	32	0.0591486	27.698035	1.15689e-06

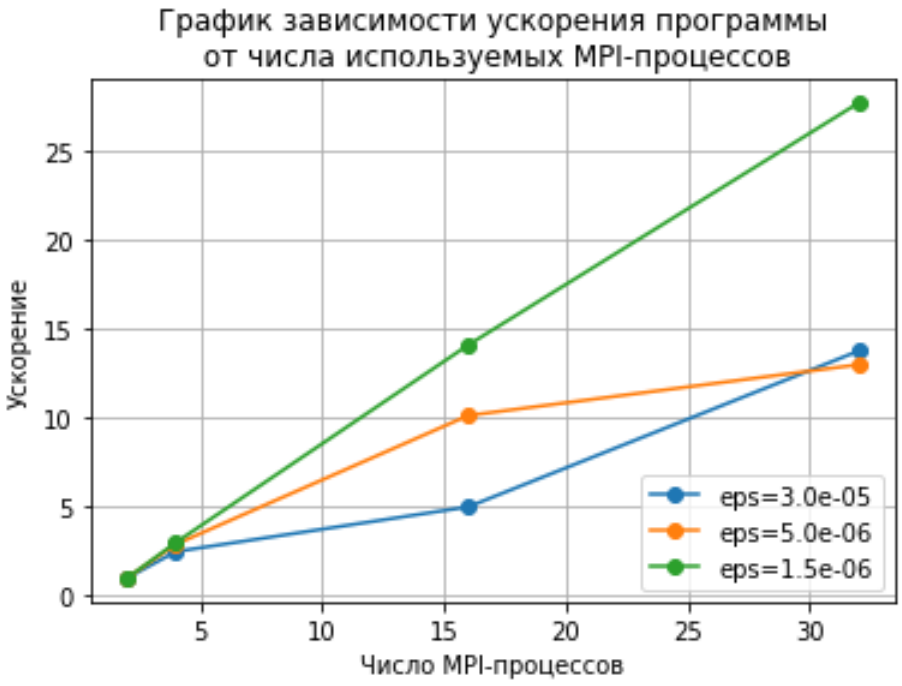

Рис. 1: График зависимости ускорения программы
от числа используемых MPI-процессов
для каждого значения ϵ

Таблица 2: Таблица с результатами расчётов для системы Polus
с усреднением по сидам 2, 22, 222, 2222;
a - ускорение, рассчитанное по среднему времени;
b - сначала считаются ускорения, а затем они усрдняются

Точность ϵ	**Число MPI-процессов**	**Время работы программы (c)**	**Ускорение (a/b)**	**Ошибка**	**Число точек**
$1.5 \cdot 10^{-6}$	2	0.2179797	1/1	9.2030575e-07	14132745
	4	0.0742319	2.936469/2.87525066	9.2030575e-07	14132745
	16	0.018988955	11.47929/10.0627162	9.2030575e-07	14132745

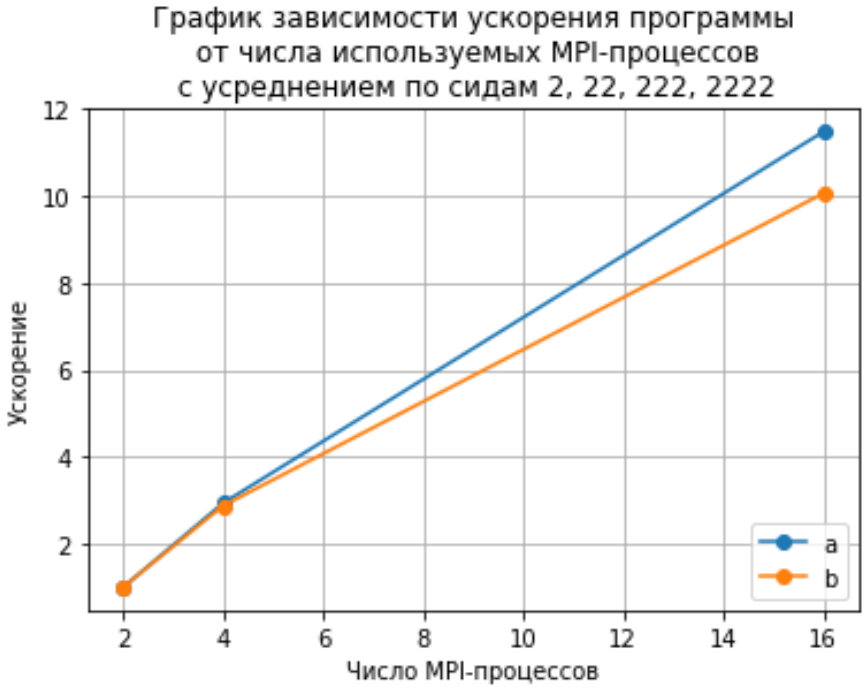

Рис. 2: График зависимости ускорения программы
от числа используемых MPI-процессов
с усреднением по сидам 2, 22, 222, 2222
для $\epsilon = 1.5 \cdot 10^{-6}$;
a - ускорение, рассчитанное по среднему времени;
b - сначала считаются ускорения, а затем они усрдняются