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

Суперкомпьютерное моделирование и технологии

Отчет по заданию №2 «Численное интегрирование многомерных функций методом Монте-Карло»

Вариант №6

Богатенкова Анастасия Олеговна
628 группа

14 октября 2022 г.

Содержание

1 Математическая постановка задачи

Функция $f(x, y, z)$ – непрерывна в ограниченной замкнутой области $G \subset R^3$. Требуется вычислить определённый интеграл:

$$\iiint\limits_G f(x, y, z)\, dxdydz$$

В полученном варианте используется следующие функция и область:

$$f(x, y, z) = \sin(x^2 + z^2) \cdot y$$

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

2 Численный метод решения задачи

Пусть область G ограниченна параллелепипедом Π : $\begin{cases} a_1 \leqslant x \leqslant b_1 \\ a_2 \leqslant y \leqslant b_2 \\ a_3 \leqslant z \leqslant b_3 \end{cases}$

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:

$$\iiint\limits_G f(x, y, z)\, dxdydz = \iiint\limits_\Pi F(x, y, z)\, dxdydz$$

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:

$$I \approx |\Pi| \cdot \frac{1}{n} \sum_{i=1}^{n} F(p_i) \tag{1}$$

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

3 Нахождение точного значения интеграла аналитически

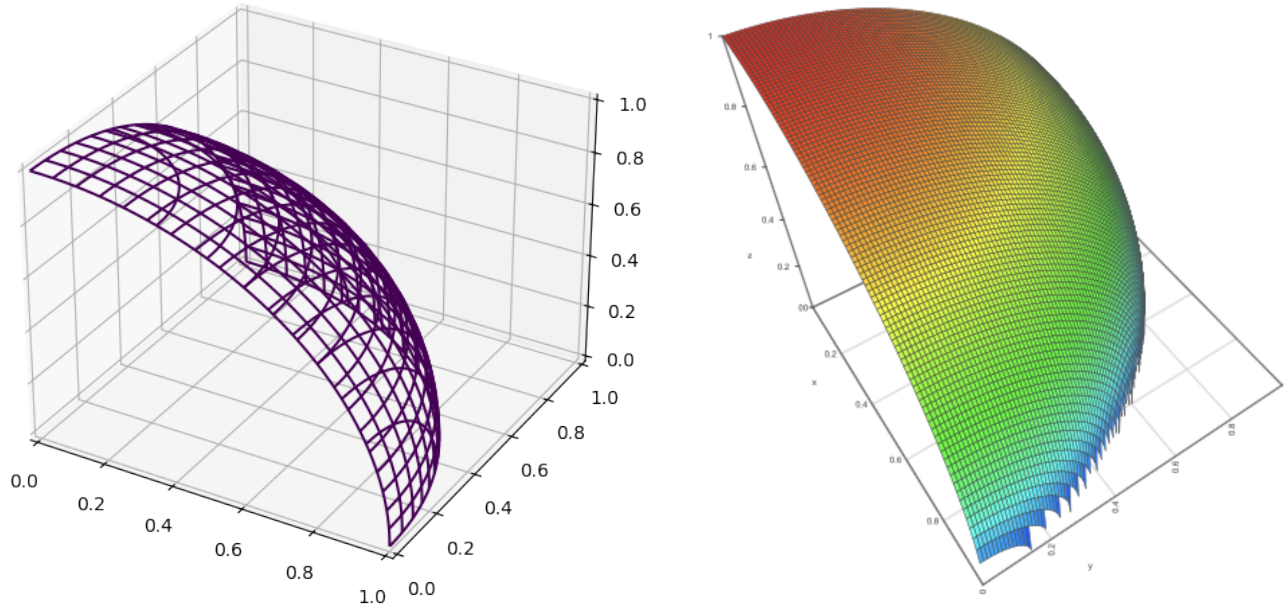

Рис. 1: Область интегрирования

Ограниченная область представляет собой сектор шара (см. рисунок 1). Для вычисления интеграла по этой области, перейдём в цилиндрическую систему координат.

$$I = \iiint\limits_G \sin(x^2+z^2)\cdot y\; dxdydz = \{x = r\cos\varphi,\ y = y,\ z = r\sin\varphi\} =$$

$$= \int\limits_0^1 \int\limits_0^{\frac{\pi}{2}} \int\limits_0^{\sqrt{1-r^2}} \sin(r^2\cos^2\varphi + r^2\sin^2\varphi) yr\; drd\varphi dy = \int\limits_0^1 \int\limits_0^{\frac{\pi}{2}} \int\limits_0^{\sqrt{1-r^2}} \sin(r^2) yr\; drd\varphi dy =$$

$$= \int\limits_0^{\frac{\pi}{2}} d\varphi \int\limits_0^1 r\sin(r^2)\; dr \int\limits_0^{\sqrt{1-r^2}} y\; dy = \frac{\pi}{2}\int\limits_0^1 r\sin(r^2)\; dr\cdot\frac{1-r^2}{2} =$$

$$= \frac{\pi}{8}\int\limits_0^1 \sin(r^2)(1-r^2)\; dr^2 = \frac{\pi}{8}\int\limits_0^1 \sin(r^2)\; dr^2 - \frac{\pi}{8}\int\limits_0^1 \sin(r^2) r^2\; dr^2 =$$

$$= \{\int x\sin x = -x\cos x + \sin x\} = -\frac{\pi}{8}\cos(r^2)\Big|_0^1 + \frac{\pi}{8}(r^2\cos(r^2) - \sin(r^2))\Big|_0^1 = \frac{\pi}{8}(1-\sin 1)$$

4 Описание программной реализации

Реализована параллельная MPI-программа, которая принимает на вход требуемую точность ε и имя выходного файла для вывода результатов. Программа генерирует случайные точки до тех пор, пока требуемая точность не будет достигнута. Точность вычисляется как модуль разности между приближённым значением, полученным методом Монте-Карло, и точным значением, вычисленным аналитически в предыдущей главе.

Программа считывает в качестве аргумента командной строки требуемую точность ε и имя выходного файла 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 измеряет своё время выполнения, затем среди полученных значений берётся максимум с помощью операции редукции *MPI_Reduce*.

В рамках полученного варианта, распараллеливание в программе проводилось независимым генерированием случайных точек каждым из процессов. Каждый процесс генерировал 1000 точек с помощью функции *rand()*, предварительно вызвав $srand(2^{process\_num})$, где $process\_num$ – номер текущего процесса. Все процессы вычисляют свою часть суммы в формуле. Затем вычисляется общая сумма с помощью операции редукции *MPI_Reduce* на одном из процессов. Далее процесс, который получил значение суммы, вычисляет приближённое значение интеграла по формуле (1), а также ошибку посчитанного значения и рассылает с помощью *MPI_Bcast* значение флага, означающего индикатор продолжения работы процесса (нужно ли сгенерировать больше точек). Если нужная точность достигнута, процессы завершают свою работу, и один из них выводит полученные результаты в файл. Иначе, генерация точек продолжается и пересылки происходят аналогично описанным выше.

В рамках полученного варианта, все вычисления проводились с функцией $f(x, y, z) = \sin(x^2 + z^2) \cdot y$ в области $G = \{(x, y, z) : \ x^2 + y^2 + z^2 \leqslant 1, \ x \geqslant 0, \ y \geqslant 0, \ z \geqslant 0)\}$, которая ограничивается прямоугольником $\Pi = [0, 1] \times [0, 1] \times [0, 1]$, $|\Pi| = 1$, приближённое значение интеграла сравнивалось с вычисленным аналитически $\frac{\pi}{8}(1 - \sin 1)$.

5 Результаты численных экспериментов

Точность ε	**Число MPI-процессов**	**Время работы программы**	**Ускорение**	**Ошибка**
$1.0 \cdot 10^{-4}$	1	0.755	1.000	$9.9 \cdot 10^{-5}$
$1.0 \cdot 10^{-4}$	4	0.025	30.735	$7.4 \cdot 10^{-5}$
$1.0 \cdot 10^{-4}$	16	0.002	360.163	$7.9 \cdot 10^{-5}$
$1.0 \cdot 10^{-4}$	64	0.574	1.316	$9.9 \cdot 10^{-5}$
$2.0 \cdot 10^{-5}$	1	2.104	1.000	$1.9 \cdot 10^{-5}$
$2.0 \cdot 10^{-5}$	4	0.771	2.729	$1.8 \cdot 10^{-5}$
$2.0 \cdot 10^{-5}$	16	0.031	68.942	$1.2 \cdot 10^{-5}$
$2.0 \cdot 10^{-5}$	64	0.907	2.320	$1.9 \cdot 10^{-5}$
$8.0 \cdot 10^{-6}$	1	2.184	1.000	$7.3 \cdot 10^{-6}$
$8.0 \cdot 10^{-6}$	4	0.818	2.670	$7.8 \cdot 10^{-6}$
$8.0 \cdot 10^{-6}$	16	0.032	69.267	$5.6 \cdot 10^{-6}$
$8.0 \cdot 10^{-6}$	64	2.140	1.021	$7.6 \cdot 10^{-6}$

Таблица 1: Результаты для Bluegene

Точность ε	**Число MPI-процессов**	**Время работы программы**	**Ускорение**	**Ошибка**
$3.0 \cdot 10^{-5}$	1	0.237	1.000	$2.7 \cdot 10^{-5}$
$3.0 \cdot 10^{-5}$	4	0.015	15.695	$2.6 \cdot 10^{-5}$
$3.0 \cdot 10^{-5}$	16	0.017	13.822	$2.6 \cdot 10^{-5}$
$3.0 \cdot 10^{-5}$	32	0.007	33.707	$1.3 \cdot 10^{-5}$
$5.0 \cdot 10^{-6}$	1	0.278	1.000	$3.0 \cdot 10^{-6}$
$5.0 \cdot 10^{-6}$	4	0.156	1.776	$3.1 \cdot 10^{-6}$
$5.0 \cdot 10^{-6}$	16	0.071	3.889	$3.2 \cdot 10^{-6}$
$5.0 \cdot 10^{-6}$	32	0.033	8.521	$3.4 \cdot 10^{-6}$
$1.5 \cdot 10^{-6}$	1	0.276	1.000	$1.3 \cdot 10^{-6}$
$1.5 \cdot 10^{-6}$	4	0.144	1.912	$1.2 \cdot 10^{-6}$
$1.5 \cdot 10^{-6}$	16	0.068	4.044	$6.7 \cdot 10^{-8}$
$1.5 \cdot 10^{-6}$	32	0.037	7.499	$4.8 \cdot 10^{-7}$

Таблица 2: Результаты для Polus

В таблицах 1 и 2 представлены результаты запусков программы на суперкомпьютерах Bluegene и Polus. Для каждого набора параметров (точность, число

процессов) программа запускалась 10 раз и в таблицах представлены усреднённые результаты. Результаты работы также представлены на графиках времени работы и ускорения (см. рисунки 2 и 3).

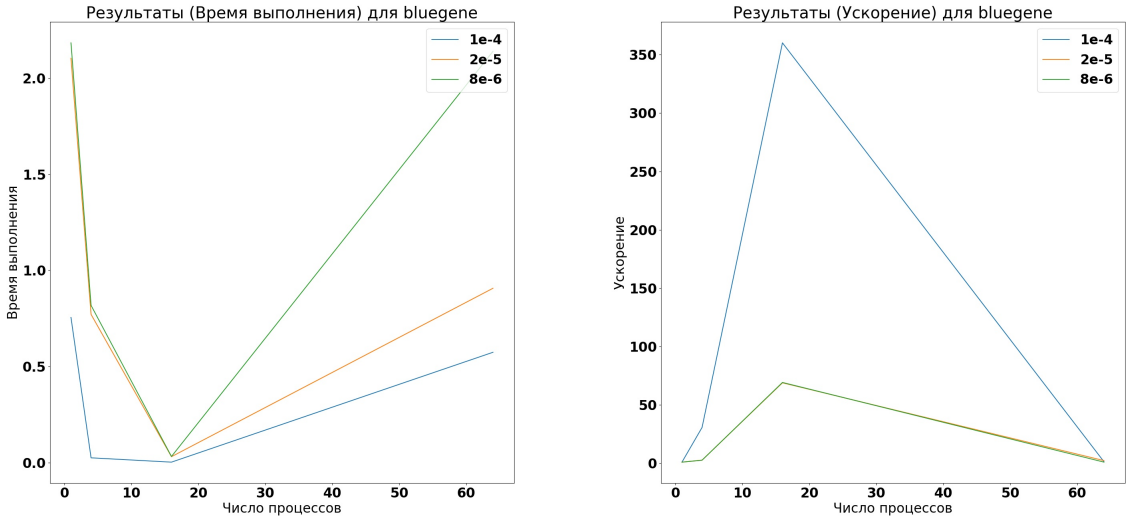

Рис. 2: Графики времени работы и ускорения для Bluegene

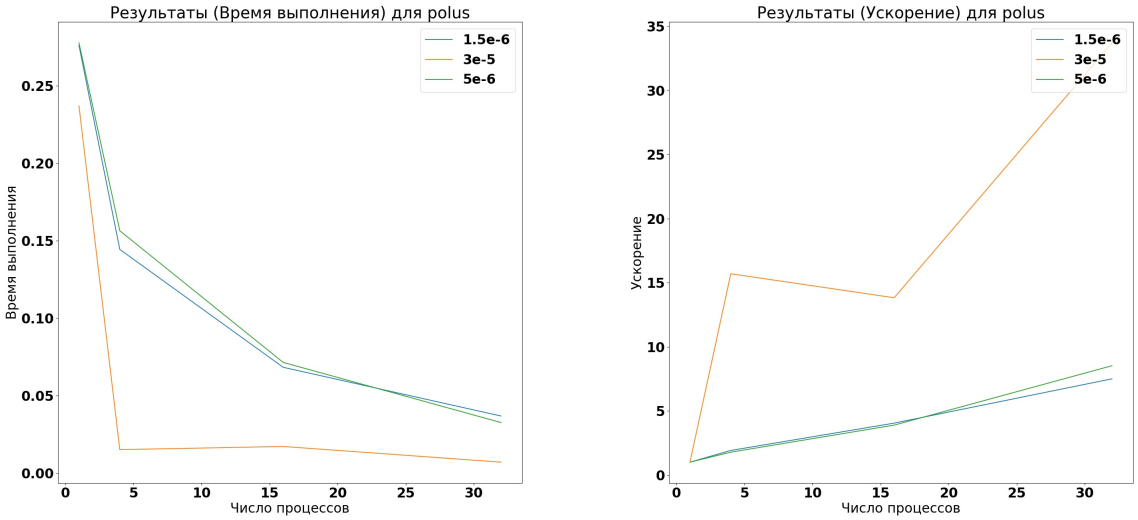

Рис. 3: Графики времени работы и ускорения для Polus

6 Выводы

Полученные результаты работы программы выглядят очень противоречиво, однако можно предположить, что это связано с неоднородностью генерации случайных точек в процессах. Количество генерируемых точек, необходимых для достижения определённой точности, сильно варьируется в зависимости от выбранного *seed* в *srand()*, а также от количества процессов, для каждого из которых генерируется своя последовательность чисел. Например, для $\varepsilon = 1.0 \cdot 10^{-4}$ при запуске на Bluegene на 1 процессе генерируется 753,000 точек, на 4 процессах – 96,000 точек, на 16 процессах – 32,000 точек, а на 64 процессах – 35,968,000 точек. Соответственно, для 1, 4 и 16 процессов ускорение повышается не только засчет распараллеливания, но и засчет «удачного» выбора точек, в случае с 64 процессами распараллеливание тоже есть, но оно незаметно, так как все ресурсы тратятся на генерацию большего количества точек и на накладные расходы в виде пересылок между процессами. Предположительно, эту проблему можно исправить выбором более «продвинутого» генератора случайных чисел, однако программу с таким генератором с большой вероятностью будет невозможно запустить на Bluegene, так как там устаревший компилятор.

По еще одному предположению, вычисление интеграла методом Монте-Карло – задача, не очень хорошо подходящая для распараллеливания, так как каждый из процессов выполняет достаточно простую задачу (генерация чисел, вычисление функции, сложение). Если бы у процессов были более «тяжеловесные» задачи, параллельный вариант был бы более эффективен.