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

Отчёт по заданию № 2 в рамках курса
Суперкомпьютерное моделирование и технологии

Численное интегрирование многомерных функций методом Монте-Карло

Петров Олег Максимович, 628 группа
Вариант 13 (интеграл 5, парадигма “мастер — рабочие”)

Москва, осень 2022

Содержание

Постановка задачи

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 параллельных вычислительных системах ВМК МГУ IBM Blue Gene/P [2] и IBM Polus [3].

Функция $f(x, y, z) = x^3 y^2 z$ непрерывна в ограниченном замкнутом параллелепипеде $G = [-1; 0]^3 \subset R^3$. Требуется вычислить определённый интеграл $I = \iiint_G f(x, y, z)\, dx\, dy\, dz = \int_{-1}^{0}\int_{-1}^{0}\int_{-1}^{0} x^3 y^2 z\, dx\, dy\, dz$.

Согласно методу Монте-Карло предлагается в качестве приближённого значения интеграла использовать $\frac{|G|}{n}\sum_{i=1}^{n} F(p_i)$, где $|G| = (0 - (-1))^3 = 1$ — объём параллелепипеда, $p_1, \ldots, p_n$ — случайные точки, равномерно распределённые в Π.

Интегралы, предлагаемые в рамках данного модельного задания, вычисляются аналитически. Требуется реализовать параллельную MPI-программу, которая 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, вычисленным 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 измеряет своё время выполнения, затем среди полученных значений берётся максимум.

В данном варианте задания предлагается использовать парадигму “мастер — рабочие”: один из процессов (“мастер”) генерирует случайные точки и передаёт каждому из остальных процессов (“рабочих”) отдельный, предназначенный для него, набор сгенерированных случайных точек. Все процессы-рабочие вычисляют свою часть суммы в формуле приближённого значения. Затем вычисляется общая сумма с помощью операции редукции.

После этого вычисляется ошибка (разность между посчитанным значением и точным значением, вычисленным аналитически). Если ошибка выше требуемой точности, которую подали на вход программе, то генерируются дополнительные точки и расчёт продолжается.

Необходимо провести запуски программы на системах Blue Gene/P и Polus для различного числа MPI-процессов и различных значений входного параметра ε в соответствии с таблицами 1 и 2, которые нужно заполнить и включить в отчёт. Требуется построить графики зависимости ускорения программы от числа используемых MPI-процессов для каждого значения ε.

Под ускорением программы, запущенной на p MPI-процессах, понимается величина $S_p = T_1/T_p$, где T_1 — время работы программы с одним “рабочим” , а T_p — время работы программы на p MPI-процессах.

Аналитическое решение

$$I = \iiint_G f(x,y,z)\,dx\,dy\,dz = \int_{-1}^{0}\int_{-1}^{0}\int_{-1}^{0} x^3y^2z\,dx\,dy\,dz =$$

$$= \frac{z^2}{2}\Big|_{-1}^{0} \cdot \int_{-1}^{0}\int_{-1}^{0} x^3y^2\,dx\,dy = \frac{-1}{2}\cdot\frac{y^3}{3}\Big|_{-1}^{0}\cdot\frac{x^4}{4}\Big|_{-1}^{0} = \frac{-1}{2}\cdot\frac{1}{3}\cdot\frac{-1}{4} = \frac{1}{24}$$

Описание программной реализации

Мастер генерирует точки в заданном параллелепипеде трёхмерного пространства, а затем рассылает сгенерированные точки рабочим. Рабочие считают свою часть интеграла, а затем с помощью редукции находится общая сумма у мастера. Мастер проверяет, достаточно ли точное полученное значение интеграла. Если да, то с помощью редукции находится максимум по времени среди запущенных процессов, мастер выводит результат, и на этом программа завершается.

Три координаты точек, равномерно распределённых в параллелепипеде $[-1;0]^3$, генерируются с помощью выражения `-drand48()`. Мастер генерирует по 32 точки для каждого работника. Генератор случайных чисел инициализируется константным значением 2.

Рассылка точек от мастера рабочим производится с помощью функции `MPI_Scatter`. К недостаткам такого решения можно отнести необходимость выделить лишнюю память под точки, которые должны были бы пересылаться к самому мастеру. Однако преимущество заключается в более простой, компактной и, вероятно, менее подверженной ошибкам реализации алгоритма.

Все рабочие считают сумму значений на своих точках и пересылают её мастеру с помощью редукции, а мастер добавляет её к суммам, полученным ранее. Значение интеграла на данном шаге получается делением этой суммы на количество сгенерированных точек.

Когда значение интеграла оказывается достаточно точным (отличается от аналитически вычисленного значения не более чем на заданную в аргументе командной строки величину), мастер рассылает в качестве набора точек набор данных, в которых вместо первого вещественного числа находится `NaN`. Таким образом рабочие получают сигнал о завершении работы без дополнительной синхронизации через рассылку флага.

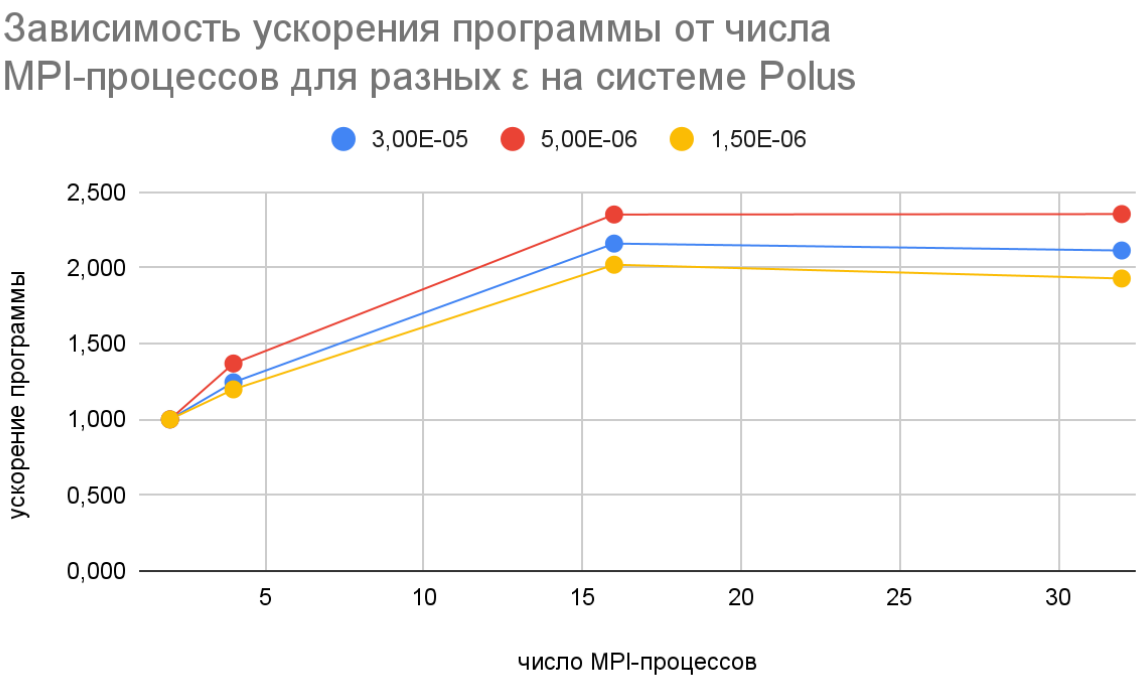
После этого мастер с помощью редукции получает максимальное время работы среди всех процессов. Время работы каждого процесса считается как разница между вызовами `MPI_Wtime` в начале и в конце работы. Затем мастер выводит результаты работы программы.

Исследование масштабируемости

Были проведены запуски программы для значений точности $3,0 \cdot 10^{-5}$, $5,0 \cdot 10^{-6}$ и $1,5 \cdot 10^{-6}$ и 2, 4, 16 и 32 выделенных MPI-процессах. Для уменьшения ошибки измерения времени проводилось по 7 запусков с одними и теми же параметрами.

Таблица с результатами расчётов для системы Polus

	Число MPI-процессов	Время работы программы (с)	Ускорение	Ошибка
	2	0,3640	1,000	3,00E-05
	4	0,2924	1,245	2,96E-05
	16	0,1685	2,160	2,95E-05
3,00E-05	32	0,1721	2,115	2,96E-05
	2	0,4104	1,000	4,77E-06
	4	0,2999	1,369	4,47E-06
	16	0,1745	2,352	4,71E-06
5,00E-06	32	0,1743	2,355	4,55E-06
	2	0,3647	1,000	1,27E-06
	4	0,3044	1,198	1,32E-06
	16	0,1805	2,021	1,45E-06
1,50E-06	32	0,1890	1,930	1,16E-08



Видно, что программа достигает ускорения около 2,2 раз при 16 MPI-процессах, что даёт эффективность менее 0,3 по числу используемых процессов. С увеличением числа процессов до 32 ускорение не увеличивается.

Заключение

Реализована параллельная программа подсчёта интеграла методом Монте-Карло с использованием стандарта MPI. Проведено несколько серий экспериментальных запусков с разной требуемой точностью и с разным числом MPI-процессов на одной суперкомпьютерной системе. Программа достигает ускорения около 2,2 раза при 16 MPI-процессах, что даёт эффективность менее 0,3 по числу используемых процессов. С увеличением числа процессов роста ускорения нет.

Можно сделать предположение, что причина малой эффективности связана с тем, что линейная сложность самого вычисления интеграла (5 умножений и как правило одно сложение на точку) сравнима со сложностью генерации случайных точек (три числа типа `double` на точку). При этом генерацией занят только мастер, и даже если использовать асинхронную рассылку точек, процессам придётся синхронизироваться при редукции суммы, так что “спрятать” генерацию точек и их рассылку за вычислениями работников не представляется возможным.

Второй причиной плохой масштабируемости можно назвать низкую вычислительную интенсивность алгоритма: и количество операций, и объём рассылаемых от мастера данных растут линейно с ростом числа точек (не считая операции редукции).