Московский Государственный Университет имени М.В. Ломоносова

Факультет вычислительной математики и кибернетики

Отчёт по практическому заданию 1 в рамках курса

«Суперкомпьютерное моделирование и технологии»

Численное интегрирование многомерных функций методом Монте-Карло

Выполнил: Афанасьев Виталий Игоревич, 610 группа

Математическая постановка задачи и численный метод ее решения

В качестве задачи предлагает реализовать алгоритм решения 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 BlueGene/P и IBM Polus).

$$\iiint\limits_G \sin(x^2 + z^2) y dx dy dz$$

, где область $G = \{(x, y, z): x^2 + y^2 + z^2 \leq 1, x \geq 0, y \geq 0, z \geq 0\}$

Нахождение точного интеграла аналитически

$$\iiint\limits_G \sin(x^2 + z^2) y dx dy dz = (\text{зам.}\, x^2 + z^2 = r^2) = \iiint\limits_G \sin(r^2) r y dr dy d\varphi$$

$$= \int\limits_0^{\pi/2} d\varphi \int\limits_0^1 y dy \int\limits_0^{\sqrt{1-y^2}} r \sin r^2 dr = \frac{\pi(1 - \sin 1)}{8}$$

Краткое описание программной реализации.

Парадигма “мастер-рабочие”

Для корректного сравнения расчетов в функцию srand была подана одинаковая затравка, в противном случае нужно бы было проводить достаточно много расчетов для корректных результатов.

Реализация мастера процесса
Бесконечный цикл:

1. Генерация точек в количестве N=10000
2. Рассылка точек по процессам (рабочим)
3. Сбор результатов от процессов (рабочих)
4. Проверка точности
5. Рассылка флага продолжения/остановки по процессам (рабочим)
6. Если флаг остановки то выйти из цикла

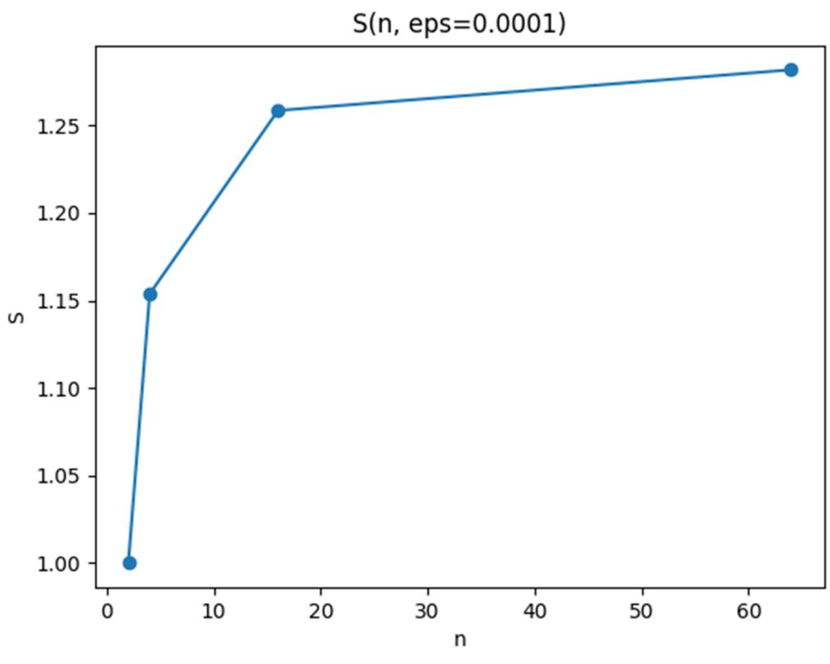
Реализация рабочих процессов
Бесконечный цикл:

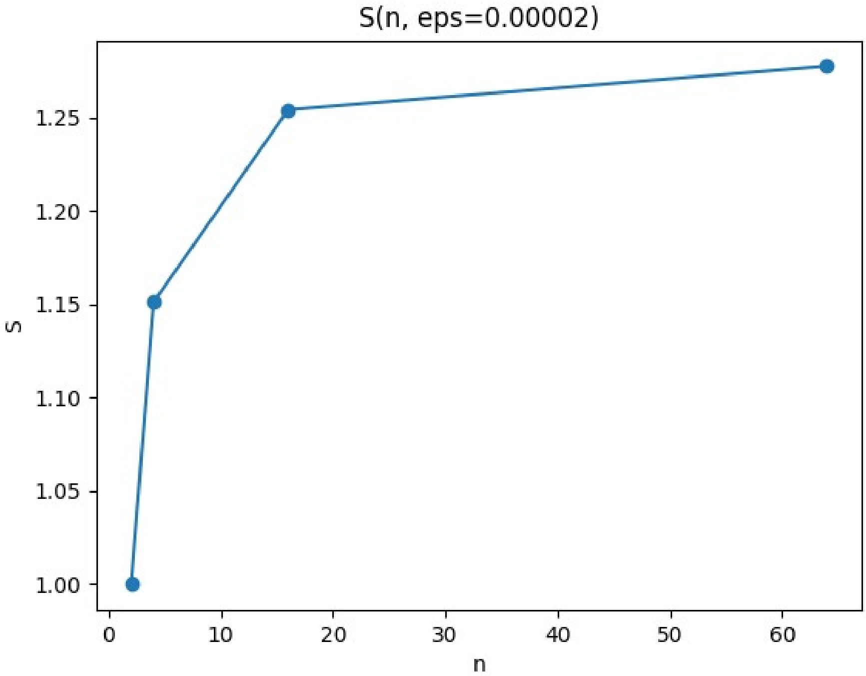
1. Ожидание точек от мастера
2. Расчет интеграла по методу Монте-Карло
3. Отправка результатов процессу-мастеру
4. Ожидание флага продолжения/остановки от мастера
5. Если флаг остановки то выйти из цикла

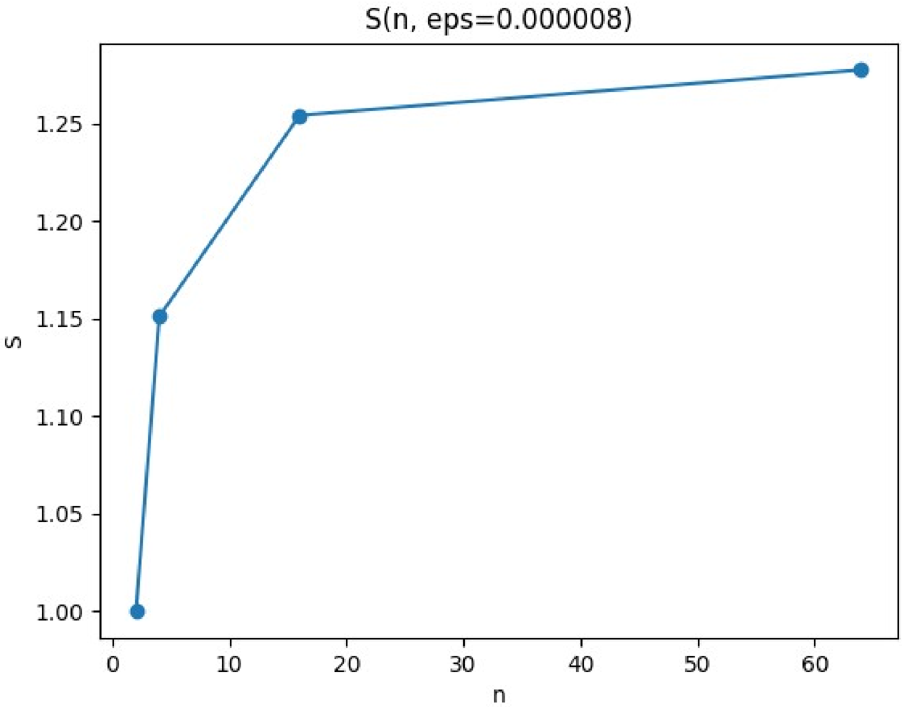
Исследование масштабируемости программы на системах BlueGene/P и Polus.

BlueGene/P

eps	n	time	s	error
	2	13,299	1	1,48E-07
	4	11,552	1,151	1,48E-07
	16	10,603	1,254	1,48E-07
0,00002	64	10,409	1,278	1,48E-07
	2	0,799	1	1,92E-06
	4	0,693	1,154	1,92E-06
	16	0,635	1,258	1,92E-06
0,0001	64	0,624	1,282	1,92E-06
	2	13,299	1	1,48E-07
	4	11,552	1,151	1,48E-07
	16	10,601	1,254	1,48E-07
0,000008	64	10,408	1,278	1,48E-07

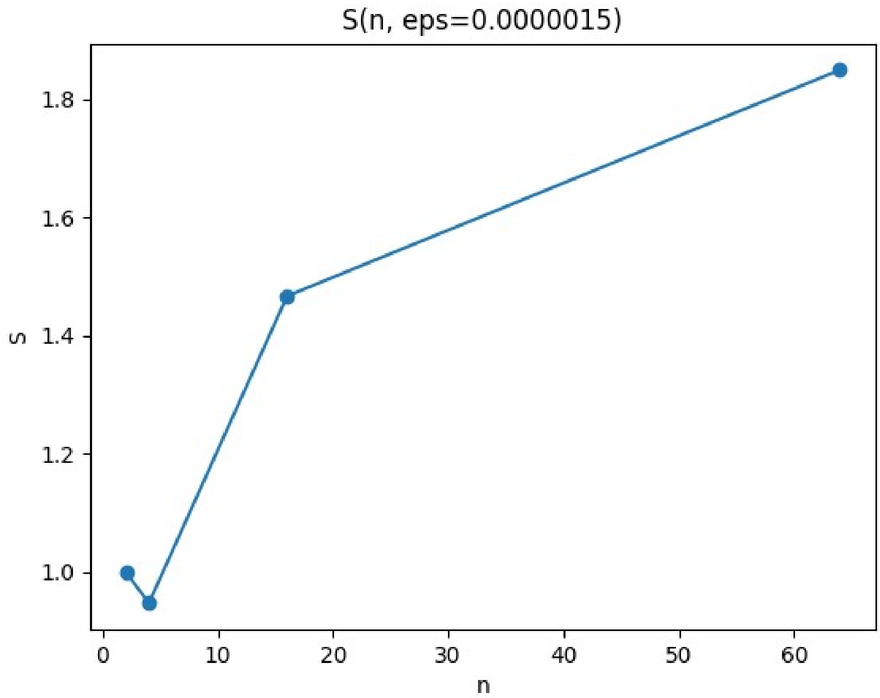


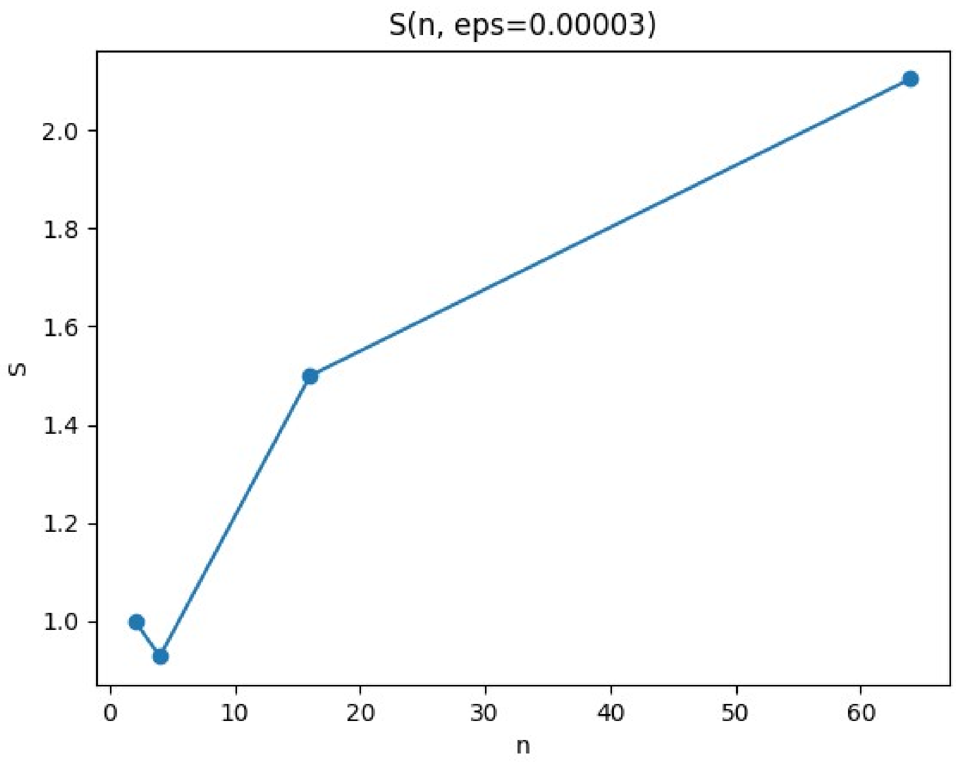

S(n, eps=0.00002)
1.25
1.20
1.15
1.10
1.05
1.00
S
0
10
20
30
40
50
60
n

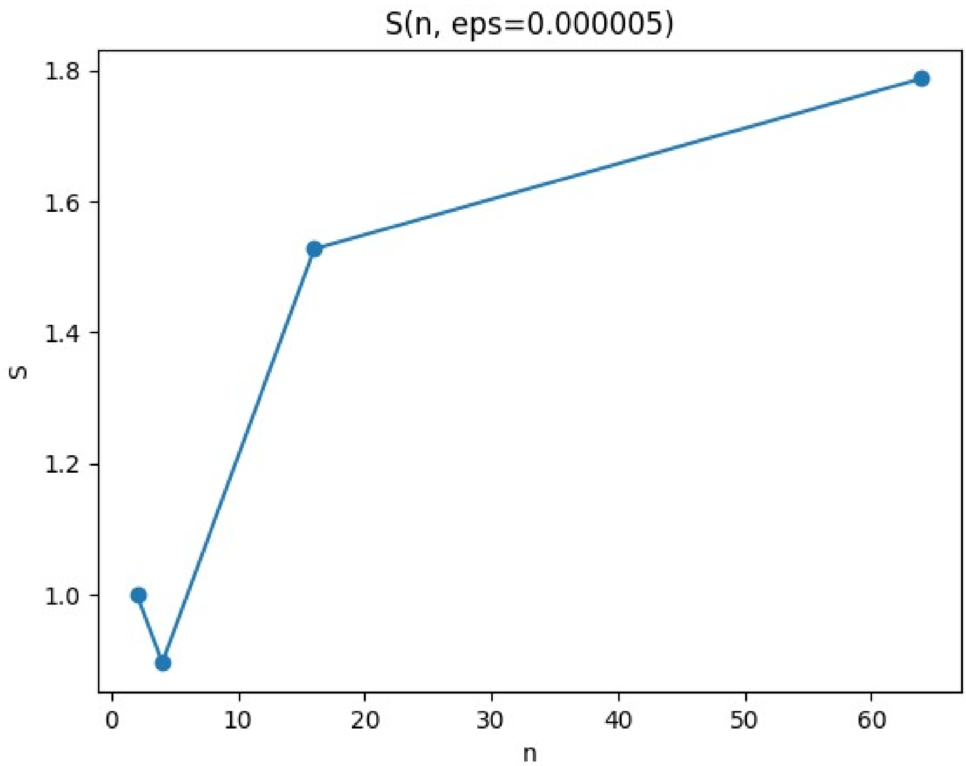

S(n, eps=0.000008)
1.25
1.20
1.15
1.10
1.05
1.00
S
0
10
20
30
40
50
60
n

Polus

eps	n	time	s	error
0,000005	2	2,264	1	1,48E-07
	4	2,525	0,897	1,48E-07
	16	1,482	1,527	1,48E-07
	64	1,267	1,787	1,48E-07
1,5E-06	2	2,743	1	6,3E-08
	4	2,895	0,948	6,3E-08
	16	1,869	1,467	6,3E-08
	64	1,483	1,85	6,3E-08
0,00003	2	2,637	1	1,43E-06
	4	2,841	0,928	1,43E-06
	16	1,759	1,499	1,43E-06
	64	1,254	2,104	1,43E-06







Вывод:

Расчеты сделаны с одинаковой затравкой функции srand и для каждого eps и n было сделано по 10 расчетов для уменьшение временной ошибки. Данная реализация имеет слабую эффективность распараллеливания. Это нетрудно заметить из графиков: при увеличении количества процессоров в 64 раза получаем ускорение не более чем в 2 раза. Такое слабое ускорение связано с медленной генерацией случайных точек и простой функцией. На генерацию случайных точек приходится 4 арифметических операции (используется линейный конгруэнтный метод) (тут я опускаю тот факт, что деление занимает больше тактов, чем сложение и умножение), а на расчет интеграла немного больше, но того же порядка, при этом мастер-процесс генерирует N –точек, а процесс рабочий считает только N / np (где np - количество процессов). Данный метод должен хорошо себя показать на расчётах более сложных интегралов.