Суперкомпьютерное моделирование и технологии

Отчет

Численное интегрирование многомерных функций методом   
Монте-Карло

Лазарев Владимир Александрович

8 вариант

07.11.2021

**Оглавление**

[Математическая постановка задачи 3](#_Toc87202157)

[Численный метод решения задачи 3](#_Toc87202158)

[Нахождение точного значения интеграла аналитически 4](#_Toc87202159)

[Программная реализация 5](#_Toc87202160)

[Результаты запусков программ на различных кластерах 6](#_Toc87202161)

[Время запусков на различных кластерах 8](#_Toc87202162)

[Ускорение на различных кластерах 9](#_Toc87202163)

[Выводы 9](#_Toc87202164)

**Математическая постановка задачи**

Функция *f(x, y, z)* – непрерывна в ограниченной замкнутой области *G ⸦ R3*. Требуется вычислить определенный интеграл:

*I =*

В полученном варианте используется следующая функция и область: *f(x, y, z) = x2y2z2*, где область *G = {(x, y, z) : |x| + |y| ≤ 1,   
-2 ≤ z ≤ 2}*.

**Численный метод решения задачи**

Пусть область G ограничена параллелепипедом :

Рассмотрим функцию и перепишем интеграл:

*I =*

Пусть *p1(x1, y1, z1)*, *p2(x2, y2, z2)*, … – случайные точки, равномерно распределенные в *П*. Возьмем *n* таких точек. В качестве приближенного значения интеграла предлагается использовать выражение:

Где |П| – объем параллелепипеда П, рассчитанного по формуле:

**Нахождение точного значения интеграла аналитически**

Проанализировав ограниченную область, можно установить границы параллелепипеда: и область для функции:

Посчитаем аналитически тройной интеграл:

*I =*

# **Программная реализация**

Реализована параллельная MPI-программа, принимающая аргумент *-eps=<value>* – необходимая точность решения (если не передавать аргумент, то по умолчанию точность установится в 10-4).

При запуске программы идет проверка введенного аргумента точности, и в случае некорректного ввода, программа выведет сообщение, никаких вычислений произведено не будет).

В качестве параллельной реализации используется классическая парадигма, т.е. независимая генерация точек MPI-процессами. Все процессы высчитывают свою часть суммы, затем вычисляется общая сумма при помощи MPI\_Reduce и итоговый интеграл.

Полученное значение интеграла сравнивается с ранее аналитически рассчитанным эталоном. В случае достижения требуемой точности процессам передается флаг об окончании выполнения вычислений. Иначе, наступает следующая итерация цикла.

# **Результаты запусков программ на различных кластерах**

**Таблица 1. Результаты расчетов на Blue Gene/P**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Точность ε** | **Число MPI-процессов** | **Время работы программы (с)** | **Ускорение** | **Ошибка** |
| 1.0 \* 10-4 | 1 | 0.0285908 | 1 | 4.19977e-05 |
| 4 | 0.033704 | 0.848291 | 4.90635e-05 |
| 16 | 0.0105698 | 2.70495184 | 6.99499e-05 |
| 64 | 0.0054323 | 5.26311139 | 4.65182e-05 |
| 2.0 \* 10-5 | 1 | 2.51958 | 1 | 1.44311e-05 |
| 4 | 0.0436065 | 57.7799181 | 1.10327e-05 |
| 16 | 0.0366541 | 68.739377 | 1.26013e-05 |
| 64 | 0.0197201 | 127.767101 | 1.50121e-05 |
| 0.8 \* 10-5 | 1 | 3.07022 | 1 | 6.68338e-06 |
| 4 | 3.46266 | 0.88666516 | 5.30369e-06 |
| 16 | 0.196495 | 15.6249268 | 5.8681e-06 |
| 64 | 0.0467732 | 65.6405805 | 3.81295e-06 |

**Таблица 2. Результаты расчетов на Polus**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Точность ε** | **Число MPI-процессов** | **Время работы программы (с)** | **Ускорение** | **Ошибка** |
| 3.0 \* 10-5 | 1 | 0.004416 | 1 | 1.63389e-05 |
| 4 | 0.004561 | 0.96820873 | 1.30654e-05 |
| 16 | 0.004793 | 0.92134363 | 1.78315e-05 |
| 64 | - | - | - |
| 5.0 \* 10-6 | 1 | 1.42757 | 1 | 4.40235e-06 |
| 4 | 1.3841 | 1.03140669 | 4.1574e-06 |
| 16 | 0.0352944 | 40.4474931 | 3.5356e-06 |
| 64 | - | - | - |
| 1.5 \* 10-6 | 1 | 1.69038 | 1 | 1.22898e-06 |
| 4 | 1.9209 | 0.87999375 | 1.38187e-06 |
| 16 | 0.0549338 | 30.7712192 | 9.04449e-07 |
| 64 | - | - | - |

Запуски производились при *N* = 5000 (кол-во генерируемых точек каждым из процессов). Время работы и ошибка были усреднены по 50-ти запускам.

# **Время запусков на различных кластерах**

График 1. Время работы на Blue Gene/P от количества процессов.

График 2. Время работы на Polus от количества процессов.

# **Ускорение на различных кластерах**

График 3. Ускорение на Blue Gene/P от количества процессов.

График 4. Ускорение на Polus от количества процессов.

# **Выводы**

Как следует из приведенных выше таблица, парадигма с независимой генерацией точек отлично подходит для решения задачи численного программирования. Однако, при небольшой требуемой точности эффект от ускорения получился не таким явным. Также в среднем при увеличении количество MPI-процессов уменьшается ошибка в расчетах. При замерах на Polus использование 64 процессов было невозможно в силу технических проблем на кластере.