Московский государственный университет им. М.В. Ломоносова

Факультет вычислительной математики и кибернетики

Отчет по заданию №2

Разработка параллельной версии программы для вычисления определенного интеграла с использованием метода прямоугольников.

Савицкий Илья

321 группа

Москва – 2021

Постановка задачи

- Написать последовательный алгоритм подсчета интеграла методом прямоугольников
- Распараллелить его при помощи MPI
- Запустить на кластерах polus и bluegene с разными параметрами и сделать выводы о времени его выполнения

Текст параллельного алгоритма

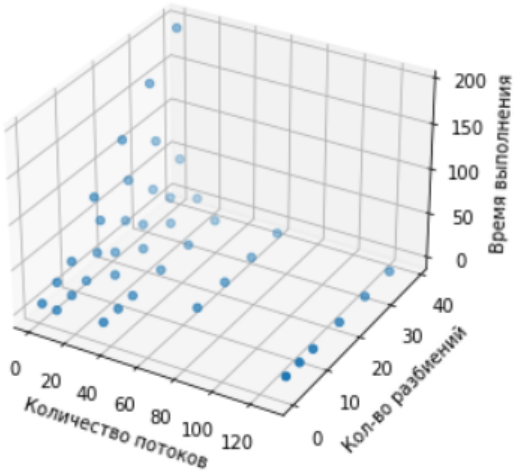
```
#include <stdio.h>
#include <math.h>
#include <stdlib.h>
#include "mpi.h"

int main(int argc, char *argv[]) {
    int n, myid, numprocs, i;
    double pi, h, sum, x, from = 0, to = 10000, start, stop;
    MPI_Init(&argc, &argv);
    MPI_Comm_size(MPI_COMM_WORLD, &numprocs);
    MPI_Comm_rank(MPI_COMM_WORLD, &myid);
    n = strtol(argv[1], NULL, 10);
    start = MPI_Wtime();
    h = (to - from) / (double)n;
    sum = 0.0;
    for (i = myid + 1; i <= n; i += numprocs) {
        x = h * ((double)i - 0.5);
        sum += sqrt(x);
    }
    sum = h * sum;
    MPI_Reduce(&sum, &pi, 1, MPI_DOUBLE, MPI_SUM, 0, MPI_COMM_WORLD);
    stop = MPI_Wtime();
    if (myid == 0) printf("%d,%lf,%d,%lf\n", numprocs, pi, n,
stop-start);
    MPI_Finalize();
    return 0;
}
```

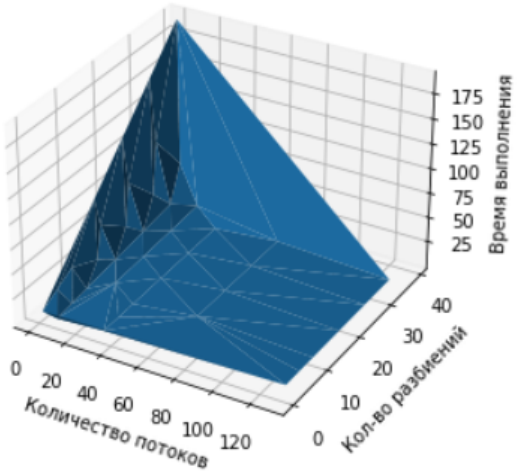
Сам алгоритм берет в себя интервал интегрирования, количество разбиение интервала и функцию, которую требуется проинтегрировать. После вычисления ширины одного прямоугольника, алгоритм проходится по интервалу этими прямоугольниками и складывает все их площади в переменную `sum`. Замер времени производится при помощи встроенной в `MPI` функции `MPI_Wtime()`, после чего основная информация о выполнении программы выводится в поток вывода.

Замер времени работы программы

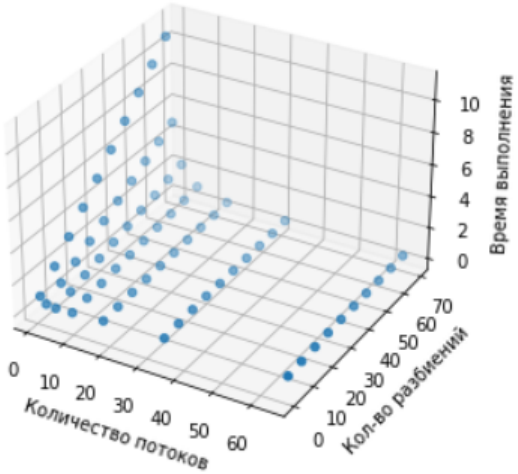
Для проверки работы алгоритма были выбраны следующие параметры:

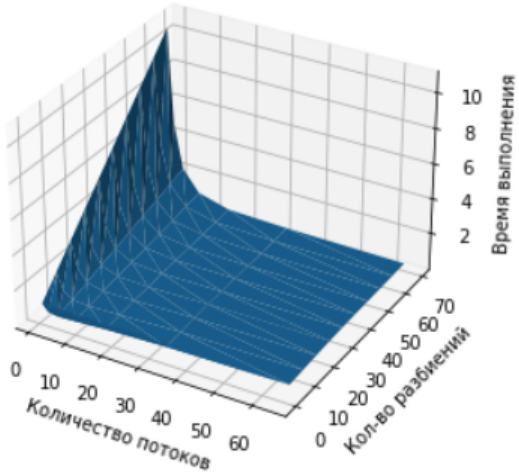
- Набор кол-ва тредов:
  - 1, 2, 4, 8, 16, 32, 64, 128, 160(для bluegene)
  - 1, 2, 4, 8, 16, 32, 64(для polus, поскольку на нем стоит ограничение на 80 вычислительных узлов)
- Так же был взят линейный набор разбиений: 100000000, 200000000, 500000000, 300000000, 400000000, 600000000, 700000000, 800000000, 900000000, 1000000000
- Функция для которой вычисляется интеграл - `sqrt(x)`
  Тогда для с/к bluegene выстраивается следующий график:





А для с/к polus:





Тогда, как видно из графиков, получаются выводы, похожие на выводы из отчета по OpenMP: на большом количестве процессоров и маленьком наборе данных время на выделение вычислительных ресурсов превосходит выигрыш во времени от параллелизма, однако такая программа хорошо масштабируется, при том сильно лучше, чем программа на OpenMP