Лабораторна робота №1

з дисципліни

Програмування комп’ютерних мереж

на тему:

**«Блокуючі передачі в MPI»**

**Варіант №18**

|  |  |
| --- | --- |
| Виконав студент  групи КВ-64М  Подольський С. В.  залікова книжка № КВ6415 | Перевірив:  СтіренкоС. Г.  \_\_\_\_\_\_\_\_\_ |

# Мета роботи

Ознайомитись з бібліотекою розробки паралельних програм для систем з локальною пам'яттю – MPI; навчитись створювати програми, що використовують MPI; вивчити блокуючі функції MPI для передачі повідомлень в режимі один-до-одного; навчитись аналізувати алгоритми щодо необхідності передач даних.

# Завдання

Розробити паралельну програму для обчислення значення заданої по варіанту функції в будь-якій точці з заданою точністю  за допомогою обчислення суми ряду.

|  |  |  |
| --- | --- | --- |
| **№** | **Функція** | **Точністьобчислення** |
| 18 | Арксинус |  |

# Математичний апарат

Арксинус: 

# Повний текст програми

/\* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \*

\* Parallel integration using MPI \*

\* Programming of computer networks Lab №2 (1st semester 2010) \*

\* \*

\* Author: \*

\* Podolsky Sergey \*

\* Group: KV-64M \*

\* \*

\* written: 17.11.2010 \*

\* remarks: debugged with Visual Studio 2010. Tested on a cluster. \*

\* \*

\* Project definition: \*

\* integrate.c The entry point for the console application \*

\* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \*/

#include <stdio.h>

#include <math.h>

#include <stdbool.h>

#include <mpi.h>

#define input\_filename "input.txt"

#define output\_filename "output.txt"

/\* Обчислення функції \*/

double function(double x)

{

return pow(x, 3) \* cos(x);

}

/\* Інтегрування методом трапецій \*/

double integrate\_composite\_trapezium\_rule(double start, double finish, double epsilon)

{

int iteration\_count = (int)(1 / sqrt(epsilon)); /\* Початкова кількість ітерацій \*/

double previous\_result, next\_result = 0;

do

{

previous\_result = next\_result;

next\_result = (function(start) + function(finish)) / 2;

double delta\_x = (finish - start) / iteration\_count;

for (int i = 1; i < iteration\_count; i++)

next\_result += function(start + i \* delta\_x);

next\_result \*= delta\_x;

iteration\_count \*= 2;

}

while (fabs(next\_result - previous\_result) / 3 > epsilon);

return next\_result;

}

int main(int argc, char\* argv[])

{

int np;

int rank;

/\* Ініціалізація MPI \*/

MPI\_Init(&argc, &argv);

/\* Отримати кількість процесів \*/

MPI\_Comm\_size(MPI\_COMM\_WORLD, &np);

/\* Отримати індивідуальний ідентифікатор процесу \*/

MPI\_Comm\_rank(MPI\_COMM\_WORLD, &rank);

/\* Введення даних з файлу в масив з трьох змінних.

\* Відбувається у процесі з рангом 0.

\* 1 - нижня межа інтегрування

\* 2 - верхня межа інтегрування

\* 3 - допустима похибка \*/

double input[3];

if (rank == 0)

{

/\* Відкриття вхідного файлу у режимі лише для читання \*/

FILE \*file = fopen(input\_filename, "r");

/\* Перевірка правильності відкриття файлу \*/

if (file == NULL)

{

printf("Failed to open the file");

/\* Завершення усіх процесів, що включені в комунікатор MPI\_COMM\_WORLD \*/

MPI\_Abort(MPI\_COMM\_WORLD, 1);

}

/\* Зчитування 3 чисел типу double \*/

for (int i = 0; i < 3; i++)

fscanf(file, "%lg", &input[i]);

/\* Закриття файлу \*/

fclose(file);

}

/\* Передача введених даних від процесу з рангом 0 до всіх інших процесів,

\* що включені у комунікатор MPI\_COMM\_WORLD \*/

MPI\_Bcast(input, 3, MPI\_DOUBLE, 0, MPI\_COMM\_WORLD);

double start = input[0];

double finish = input[1];

double epsilon = input[2];

double step = (finish - start) / np;

double result\_part = integrate\_composite\_trapezium\_rule(start + rank \* step, start + (rank + 1) \* step, epsilon / np);

double total\_result;

/\* Передача проміжного результату інтегрування від усіх процесів до процесу з рангом 0

\* та збереження суми проміжних результатів у змінній result \*/

MPI\_Reduce(&result\_part, &total\_result, 1, MPI\_DOUBLE, MPI\_SUM, 0, MPI\_COMM\_WORLD);

/\* Виведення процесом 0 результату роботи програми у вихідний файл \*/

if (rank == 0)

{

/\* Відкриття файлу на запис \*/

FILE \*file = fopen(output\_filename, "w");

/\* Перевірка правильності відкриття файлу \*/

if (file == NULL)

{

printf("Failed to open the file");

/\* Завершення усіх процесів, що включені в комунікатор MPI\_COMM\_WORLD \*/

MPI\_Abort(MPI\_COMM\_WORLD, 1);

}

/\* Запис 1 числа типу double в файл \*/

fprintf(file, "%f\n", total\_result);

/\* Закриття файлу \*/

fclose(file);

}

/\* Закінчити роботу з MPI \*/

MPI\_Finalize();

return 0;

}

# Вхідні дані (in.txt)

0.479425539

# Вихідні дані (out.txt)

0.500000000450262



Рис. . Відлагодження програми