Міністерствоосвіти і науки України

Національний університет „Львівська політехніка”

Кафедра ЕОМ



**Лабораторна робота №1**

З дисципліни:”Паралельні та розподіленіобчислення”

На тему:” ВИКОРИСТАННЯ ФУНКЦІОНАЛЬНОЇ ДЕКОМПОЗИЦІЇ ДЛЯ РОЗВ’ЯЗКУ ОБЧИСЛЮВАЛЬНИХ ЗАДАЧ”

Виконав: ст.гр.КІ-34

Алексєєнко Д. С.

Прийняв:

Козак Н. Б.

Львів 2020

**Мета:**

Вивчити методи декомпозицій задач. Набути навиків розв’язування задач з використанням функціональної декомпозиції.

**Варіант 1**

|  |  |  |  |
| --- | --- | --- | --- |
| 1 | стовпець | | |
| bi=1/(i2+2) для парних і  bi=1/i для непарних і | A1(b1+c1) | A2(B2-C2)  Cij=1/(i+j+2) |

*Правила знаходження елементів виразу.*

1).Задати\* квадратну матрицю А порядку n. Отримати вектор(стовпець) , де b – вектор-стовпець, елементи якого обраховуються за формулою, згідно варіанту.

2).Задати квадратну матрицю А1 порядку n та вектори-стовпці b1 та c1з nелементами кожен. Отримати вектор  згідно формули, що задається варіантом.

3).Задати квадратні матриці А2 та B2порядку n. Отримати матрицю , яка залежить від А2, B2 та додатково визначеної матриці С2, елементи якої знаходяться за формулою, вказаною варіантом.

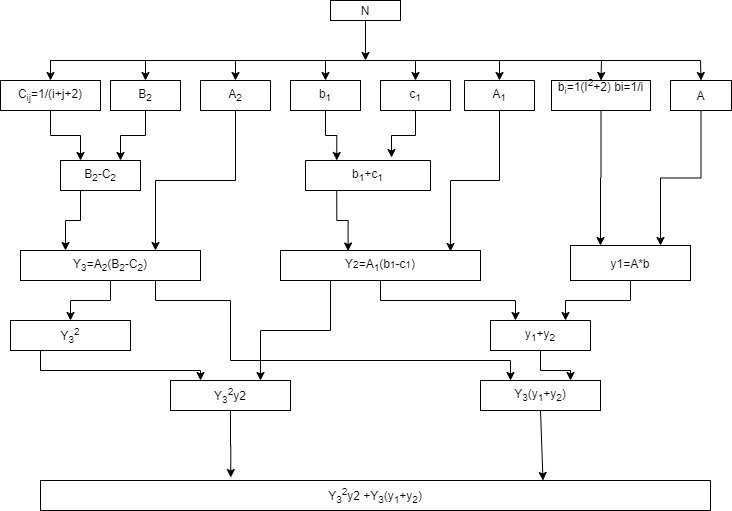


Рис.1. Схема декомпозиції поставленої задачі

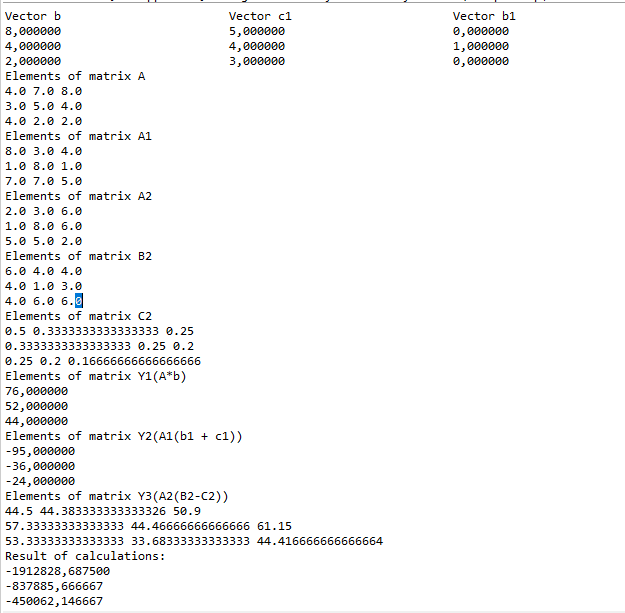


Рис.2. Результат обчислень виразу

**Висновок:**

Виконуючи дану лабораторну роботувивчив метод декомпозицій задач. Набув навиків розв’язування задач з використанням функціональної декомпозиції.

**Лістинги програми:**

**import** java.util.Scanner;

**import** java.util.SplittableRandom;

**public** **class** Main {

**private** **static** **void** printMatrix(**double**[][] arr)

{

**for** (**double**[] doubles : arr) {

**for** (**double** aDouble : doubles) {

System.***out***.print(aDouble + " ");

}

System.***out***.println();

}

}

**public** **static** **void** main(String[] args) {

Scanner sc = **new** Scanner(System.***in***);

SplittableRandom rand = **new** SplittableRandom();

**final** **int** minNum = 1;

**final** **int** maxNum = 9;

**int** N = 0;

**int** REZ;

**do** {

System.***out***.print("Quantity of elements(more than 2): ");

N = sc.nextInt();

} **while** (N < 3);

/////////////////////////

**double**[] tmp\_v = **new** **double**[N];

**double**[][] tmp\_m = **new** **double**[N][N];

**double**[] b1 = **new** **double**[N];

**double**[] c1 = **new** **double**[N];

**double**[] b = **new** **double**[N];

/////////////////////////

**double**[][] A = **new** **double**[N][N];

**double**[][] A1 = **new** **double**[N][N];

**double**[][] A2 = **new** **double**[N][N];

**double**[][] B2 = **new** **double**[N][N];

**double**[][] C2 = **new** **double**[N][N];

**double**[] Y1 = **new** **double**[N];

**double**[] Y2 = **new** **double**[N];

**double**[][] Y3 = **new** **double**[N][N];

/////////////////////////

**for** (**int** i = 0; i< N; i++)

{

A1[i] = **new** **double**[N];

A[i] = **new** **double**[N];

A2[i] = **new** **double**[N];

B2[i] = **new** **double**[N];

C2[i] = **new** **double**[N];

Y3[i] = **new** **double**[N];

tmp\_m[i] = **new** **double**[N];

b[i] = 8 / (i + 1);

}

System.***out***.println("Manually enter matrix elements - 1\nRandom filling - 2");

System.***out***.print("Choose: ");

REZ = sc.nextInt();

**for** (**int** i = 0; i< N; i++)

{

**for** (**int** j = 0; j < N; j++)

{

C2[i][j] = 1 / ((**double**)i + (**double**)j + 2);

}

}

**if** (REZ == 2)

{

**for** (**int** i = 0; i< N; i++)

{

**for** (**int** j = 0; j < N; j++)

{

A[i][j] = rand.nextInt(minNum, maxNum);

A1[i][j] = rand.nextInt(minNum, maxNum);

A2[i][j] = rand.nextInt(minNum, maxNum);

B2[i][j] = rand.nextInt(minNum, maxNum);

}

c1[i] = rand.nextInt(minNum, maxNum);

b1[i] = rand.nextInt(minNum, maxNum);

**if**((i%2==0)) {

b[i] = 1/(i\*i+2);

}

**else** {

b[i] = 1/i;

}

}

}

**else**

{

System.***out***.println("Matrix A: ");

**for** (**int** i = 0; i< N; i++)

{

**for** (**int** j = 0; j < N; j++)

{

A[i][j] = sc.nextInt();

}

}

System.***out***.println("Matrix A1: ");

**for** (**int** i = 0; i< N; i++)

{

**for** (**int** j = 0; j < N; j++)

{

A1[i][j] = sc.nextInt();

}

}

System.***out***.println("Matrix A2: ");

**for** (**int** i = 0; i< N; i++)

{

**for** (**int** j = 0; j < N; j++)

{

A2[i][j] = sc.nextInt();

}

}

System.***out***.println("Matrix B2: ");

**for** (**int** i = 0; i< N; i++)

{

**for** (**int** j = 0; j < N; j++)

{

B2[i][j] = sc.nextInt();

}

}

System.***out***.println("Culumn b1: ");

**for** (**int** i = 0; i< N; i++)

{

b1[i] = sc.nextInt();

}

System.***out***.println("Culumn c1: ");

**for** (**int** i = 0; i< N; i++)

{

c1[i] = sc.nextInt();

}

}

System.***out***.println("-----------------------------------------------------");

System.***out***.println("Vector b\t\t\tVector c1\t\t\tVector b1");

**for** (**int** i = 0; i< N; i++)

{

System.***out***.print(String.*format*("%f\t\t\t%2f\t\t\t%f\n", b[i], c1[i], b1[i]));

}

System.***out***.println("Elements of matrix A");

*printMatrix*(A);

System.***out***.println("Elements of matrix A1");

*printMatrix*(A1);

System.***out***.println("Elements of matrix A2");

*printMatrix*(A2);

System.***out***.println("Elements of matrix B2");

*printMatrix*(B2);

System.***out***.println("Elements of matrix C2");

*printMatrix*(C2);

**for** (**int** i = 0; i< N; i++) {

**for** (**int** j = 0; j < N; j++) {

tmp\_m[i][j] = (B2[i][j]) - C2[i][j]; // B2-C2

}

tmp\_v[i] = (b1[i]) - (c1[i]); // b1+c1

}

**for** (**int** i = 0; i< N; i++) {

Y2[i] = 0;

Y1[i] = 0;

**for** (**int** j = 0; j < N; j++) {

Y3[i][j] = 0;

**for** (**int** k = 0; k < N; k++) {

Y3[i][j] += A2[i][k] \* tmp\_m[k][j]; // A2(B2 - C2)

}

Y1[i] += A[i][j] \* b[j]; // A\*b так може нада пофікситии

Y2[i] += A[i][j] \* tmp\_v[i]; //A1(b1-c1)

}

}

System.***out***.println("Elements of matrix Y1(A\*b)");

**for** (**int** i = 0; i< N; i++)

{

System.***out***.print(String.*format*("%f\n", Y1[i]));

}

System.***out***.println("Elements of matrix Y2(A1(b1 + c1))");

**for** (**int** i = 0; i< N; i++)

{

System.***out***.print(String.*format*("%f\n", Y2[i]));

}

System.***out***.println("Elements of matrix Y3(A2(B2-C2))");

*printMatrix*(Y3);

**for** (**int** i = 0; i< N; i++) {

**for** (**int** j = 0; j < N; j++) {

tmp\_m[i][j] = 0;

**for** (**int** k = 0; k < N; k++) {

tmp\_m[i][j] += Y3[i][k] \* Y3[k][j]; // tmp\_m = Y3 \* Y3

}

}

}

**for** (**int** i = 0; i< N; i++) {

b1[i] = Y1[i] + Y2[i]; // b1=y1+y2

}

**for** (**int** i = 0; i< N; i++) {

**for** (**int** j = 0; j < N; j++) {

A1[i][j] = 0;

A2[i][j] = 0;

**for** (**int** k = 0; k < N; k++) {

A1[i][k] += Y3[i][k] \* b1[i]; // = Y3 (y1 \* y2) Y1[i] += A[i][j] \* b[j];

A2[i][k] += tmp\_m[i][j] \* Y2[i]; // A2 = y2\*Y(2)3

}

c1[i] += Y3[i][j] \* b[j]; // = Y3 (y1 \* y2)

Y1[i] += tmp\_m[i][j] \* Y2[i]; // Y(2)3 \* y2

}

}

**for** (**int** i = 0; i< N; i++) {

Y2[i] = c1[i] + Y1[i];

}

System.***out***.print("Result of calculations: \n");

**for** (**int** i = 0; i< N; i++) {

System.***out***.print(String.*format*("%f \n", Y2[i]));

}

}

}