**Міністерство освіти і науки України**

**Національний університет "Львівська політехніка"**

**Кафедра ЕОМ**

****

**Звіт з лабораторної роботи №2**

**з дисципліни “Паралельні та розподілені обчислення ”**

**Варіант 1**

**Виконав: студент .гр. КІ-33**

**Барсієнко В.О.**

**Прийняв: асистент**

**Козак Н.Б.**

**Львів 2020 р.**

**Мета:** Вивчити можливості паралельного представлення алгоритмів.

Набути навиків такого представлення.

**Завдання:**

Запропонувати та реалізувати локально-рекурсивний алгоритм обчислення виразу:

 ,

де А та В матриці з елементами  та , відповідно(), тобто:

 () .

**Індивідуальне завдання:**



**Код програми:**

#include <iostream>

#include <vector>

#include <ctime>

using namespace std;

void matrixOut(int n, vector<vector<int>> &A, vector<vector<int>> &B) {

int i, j;

cout << endl << "Matrix A" << endl;

for (i = 0; i < n; i++) {

for (j = 0; j < n; j++) {

cout << A[i][j] << "\t";

}

cout << endl;

}

cout << endl << "Matrix B:" << endl;

for (i = 0; i < n; i++) {

for (j = 0; j < n; j++) {

cout << B[i][j] << "\t";

}

cout << endl;

}

cout << endl;

}

void resOut(int n, vector<vector<int>> &Y) {

int i, j;

cout << endl << "Result matrix Y:" << endl;

for (i = 0; i < n; i++) {

for (j = 0; j < n; j++) {

cout << Y[i][j] << "\t";

}

cout << endl;

}

cout << endl;

}

vector<vector<int>> MMM(vector<vector<int>> a, vector<vector<int>> b, int size) {

vector<vector<int>> res;

res.resize(size);

for (int i = 0; i < size; ++i) {

res[i].resize(size);

}

for (int i = 0; i < size; i++) {

for (int j = 0; j < size; j++) {

res[i][j] = 0;

for (int k = 0; k < size; k++) {

res[i][j] += a[i][k] \* b[k][j];

}

}

}

return res;

}

int main() {

srand(time(NULL));

int i, j, n;

cout << "Enter matrix size(n): ";

cin >> n;

vector<vector<int>> A(n, vector<int>(n)), B(n, vector<int>(n)), Y(n, vector<int>(n));

for (i = 0; i < n; i++) { A[i][i] = n - i; }

for (i = 0; i < n; i++) { for (j = i; j < n; j++) { B[i][j] = rand() % 10; } }

matrixOut(n, A, B);

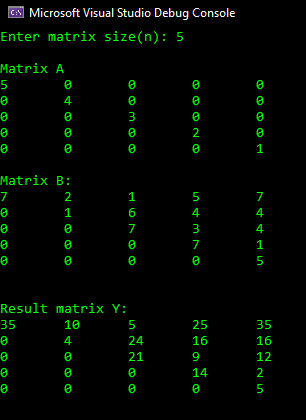
Y = MMM(A, B, n);

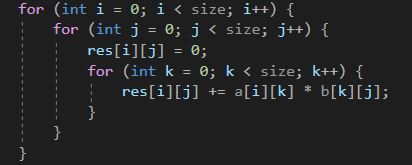
resOut(n, Y);

return 0;

}

**Результат виконання програми:**



**Рекурсивне рівняння:**

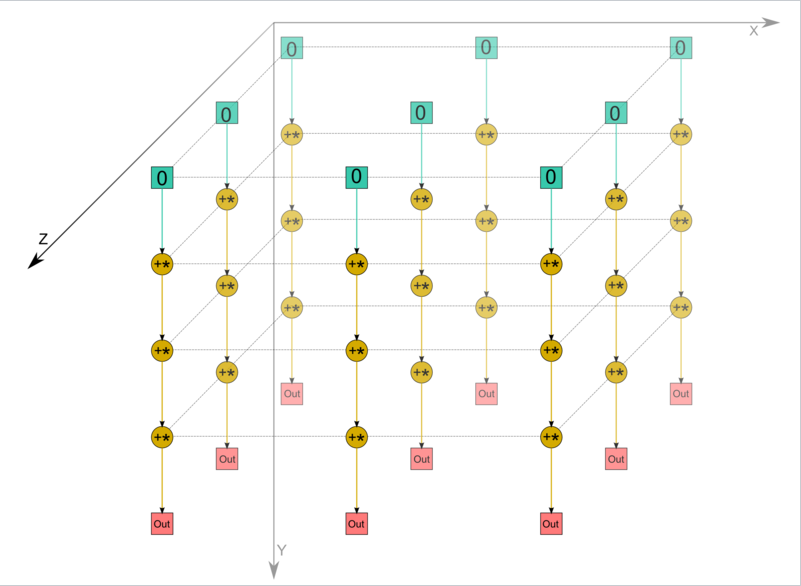
Yi(j+1) = Yi(j) + Ai(k) \* Bk(j)

Ai(k) = A[i][k]

Bk(j) = B[k][j]

J - індекс рекурсії

**Граф залежності:**



**Висновок:**

На цій лабораторній роботі я вивчив можливості паралельного представлення алгоритмів та набув навиків такого представлення.