텍스트이(가) 표시된 사진

자동 생성된 설명



**20185999 임태규**

1. **Execution environment**

* **CPU Type: Intel ® CORE™ i5 – 1135G7**
* **Number of cores: 4 cores**
* **Memory size: 8GB**
* **OS Type: Windows**
* **IDE: Visual Studio Code**

텍스트이(가) 표시된 사진

자동 생성된 설명

1. **Execution Result**
2. **Execution Time**

* **Graph/Table**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | 1 | 2 | 4 | 6 | 8 |
| exec time | 325 | 187 | 62 | 52 | 48 |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | 10 | 12 | 14 | 16 | 32 |
| exec time | 55 | 54 | 46 | 55 | 47 |

1. **Performance**

* **Graph/Table**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| exec time | 1 | 2 | 4 | 6 | 8 |
| static(block) | 0.003077 | 0.005348 | 0.016129 | 0.019231 | 0.020833 |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| exec time | 10 | 12 | 14 | 16 | 32 |
| static(block) | 0.018182 | 0.018519 | 0.021739 | 0.018182 | 0.021277 |

1. **Explanation / Analysis**

Look at the table and the graph above.

Looking at the correlation between thread count and execution time/performance, we found that performance increased rapidly as the number of threads initially increased from one to two and then from two to four...six...eight.

However, after a certain number, the rate of increase decreases or rarely exists, and sometimes the performance has decreases. This indicates that the task that needs to be performed when more than a certain number of threads are being used, that is, the overhead, is greater than the benefits derived from the number of threads.

1. **Source Code & Output**
2. **MatmultD.**
   1. **Code**

import java.util.\*;

import java.lang.\*;

import java.io.\*;

// command-line execution example) java MatmultD 6 mat500.txt

// 6 means the number of threads to use

// mat500.txt means the file that contains two matrices is given as standard input

//

// Original JAVA source code: http://stackoverflow.com/questions/21547462/how-to-multiply-2-dimensional-arrays-matrix-multiplication

public class MatmultD

{

    private static int MAX\_ROW = 500;

    private static int NUM\_THREADS = 1;

    private static int[] THREAD\_NUMS = { 1, 2, 4, 6, 8, 10, 12, 14, 16, 32 };

    private static String FILE\_NAME = "problem2\\mat500.txt";   
// 파일 입출력을 위한 변수 설정

    private static Scanner sc = null;

    static int a[][];

    static int b[][];

    static int ans[][];

    public static void main(String [] args) throws InterruptedException {

        if(args.length==2){

            THREAD\_NUMS = new int[1];

            THREAD\_NUMS[0] = Integer.valueOf(args[0]);  
//argument가 있을 경우 Thread 개수 설정

            FILE\_NAME = args[1];//argument가 있을 경우 파일명 설정

        }

        try{

            String path = System.getProperty("user.dir");//파일 현재경로(proj1)

            sc = new Scanner(new File(path+"\\"+FILE\_NAME));//파일 열기

        }catch(FileNotFoundException e){

            e.printStackTrace();//파일 없을 경우 Exception 처리

        }

        a = readMatrix();

        b = readMatrix();

        for (int i = 0; i < THREAD\_NUMS.length; i++) {

            NUM\_THREADS = THREAD\_NUMS[i];

            ans = new int[MAX\_ROW][MAX\_ROW];

            StaticThread[] thread = new StaticThread[NUM\_THREADS];

            for (int t = 0; t < NUM\_THREADS; t++) {

                thread[t] = new StaticThread(t);

            }

            long startTime = System.currentTimeMillis();

            for (int t = 0; t < NUM\_THREADS; t++) {

                thread[t].start();

            }

            for (int t = 0; t < NUM\_THREADS; t++) {

                thread[t].join();

            }

            long endTime = System.currentTimeMillis();

            long timeDiff = endTime - startTime;

            System.out.println("Thread " + NUM\_THREADS);

            for (int t = 0; t < NUM\_THREADS; t++) {

                System.out.println("Thread " + t + " Execution Time: " + thread[t].timeDiff + "ms");

            }

            System.out.println("\nProgram Execution Time: " + timeDiff + "ms");

            printMatrix(ans);

        }

    }

    public static int[][] readMatrix() {

        int rows = sc.nextInt();

        int cols = sc.nextInt();

        MAX\_ROW = rows;

        int[][] result = new int[rows][cols];

        for (int i = 0; i < rows; i++) {

            for (int j = 0; j < cols; j++) {

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

            }

        }

        return result;

    }

    public static void printMatrix(int[][] mat) {

        int rows = mat.length;

        int columns = mat[0].length;

        int sum = 0;

        for (int i = 0; i < rows; i++) {

            for (int j = 0; j < columns; j++) {

                sum+=mat[i][j];

            }

        }

        System.out.println("Matrix Sum = " + sum + "\n");

    }

    static class StaticThread extends Thread {

        int startRow;

        int endRow;

        static int counter;

        int temp = 0;

        long startTime;

        long endTime;

        long timeDiff;

        public StaticThread(int threadNum) {

            this.startRow = (MAX\_ROW / NUM\_THREADS) \* threadNum;

            if (threadNum != NUM\_THREADS - 1) {

                this.endRow = (MAX\_ROW / NUM\_THREADS) \* (threadNum + 1);

            } else {

                this.endRow = MAX\_ROW;

            }

        }

        @Override

        public void run() {

            startTime = System.currentTimeMillis();

            multMatrix();

            endTime = System.currentTimeMillis();

            timeDiff = endTime - startTime;

        }

        void multMatrix(){

            if(a.length == 0) return;

            if(a[0].length != b.length) return;

            int n = a[0].length;

            int p = b[0].length;

            for(int row = startRow; row < endRow; row++){

                for(int j = 0;j < p;j++){

                    for(int k = 0;k < n;k++){

                        ans[row][j] += a[row][k] \* b[k][j];

                    }

                }

            }

        }

    }

}

* 1. **Output**

텍스트이(가) 표시된 사진

자동 생성된 설명 텍스트이(가) 표시된 사진

자동 생성된 설명

텍스트이(가) 표시된 사진

자동 생성된 설명 텍스트이(가) 표시된 사진

자동 생성된 설명 텍스트이(가) 표시된 사진

자동 생성된 설명

1. **How to start**
2. **Please go to proj1 directory in your terminal.**
3. **enter the command “ java problem2/MatmultD.java “**
4. **Program’s default setting is file: mat500.txt, for loop 1 ~ 32 thread at once.  
   if you want to try just one kind of thread number, enter the command  
   “ java problem2/MatmultD.java ${num of thread} ${FilePath ex}problem2\\mat1000.txt} “**

