Project 1: Problem 2

소프트웨어학부 20204898 박소은

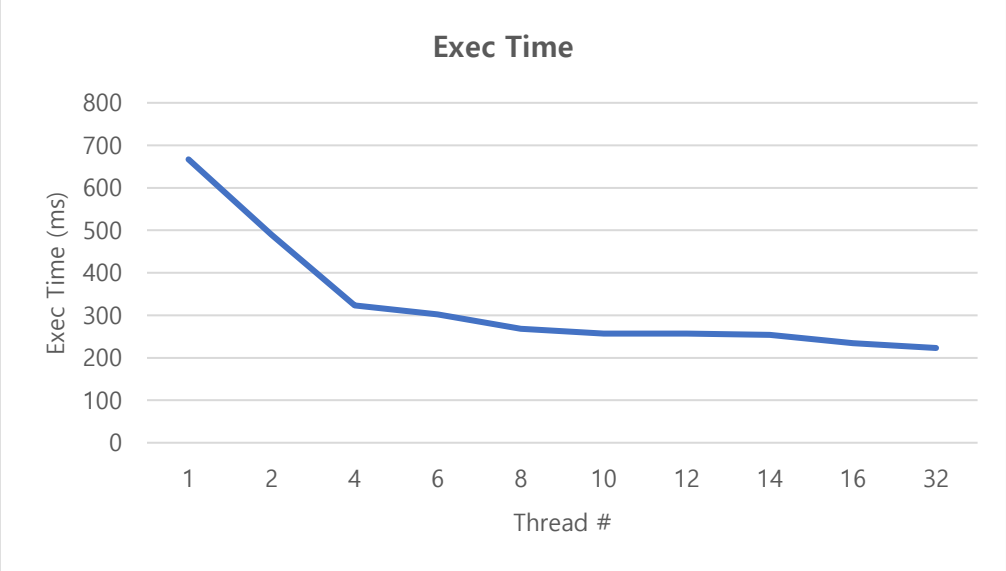
Environment

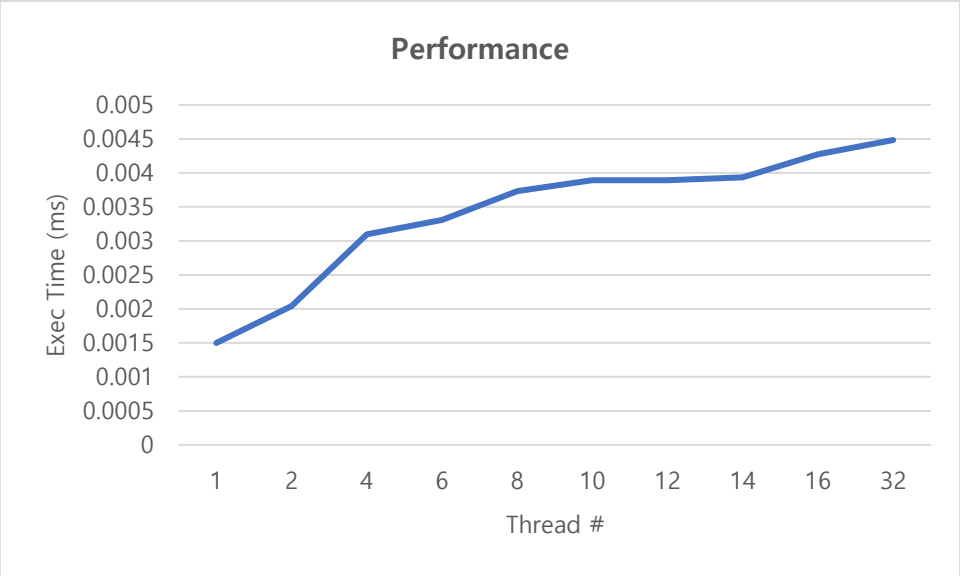
- **Processor**: Intel(R) Core(TM) i7-1065G7 CPU @ 1.30GHz 1.50 GHz
- **Number of cores**: 4
- **RAM**: 16.0GB(15.8GB available)
- **OS**: Windows 11 (64 bit)

Tables and graphs

Thread #	1	2	4	6	8	10	12	14	16	32
Exec Time(ms)	667	490	323	302	268	257	257	254	234	223

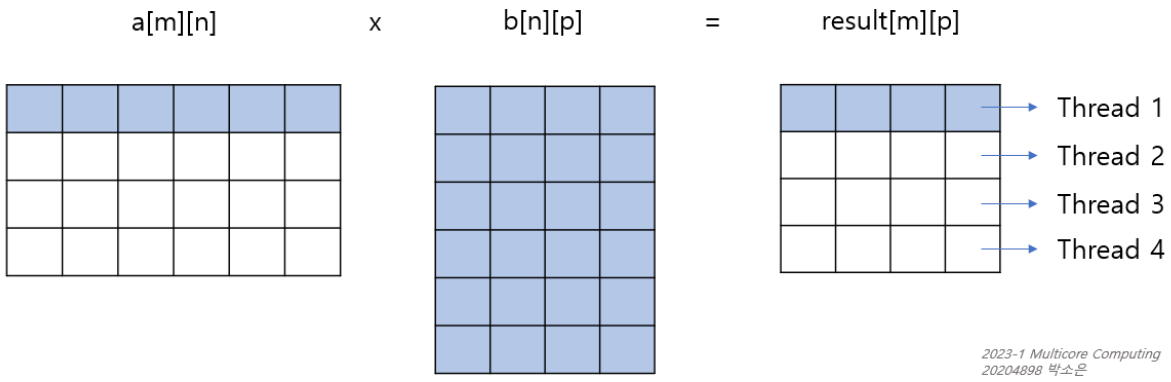
Thread #	1	2	4	6	8	10	12	14	16	32
Performance(1/ms)	0.0015	0.00204	0.0031	0.003311	0.003731	0.003891	0.003891	0.003937	0.004274	0.004484





Explanation / Analysis

Using dynamic load balancing, each thread calculates a 'row'.



The result obtained by multiplying a[m][n] and b[n][p] is result[m][p]. Each thread calculates one row of the result matrix. For example, thread 1 calculates result[0], and thread 2 calculates result[1]. After one row of calculation is completed, the thread is assigned next row to calculate by 'ThreadController' object.

If the task size was one 'element' (for example, thread 1 calculates result[0][0] and thread 2 calculates result[1][1]), load balance would be better. But the overhead will increase if the task is divided into too small pieces, therefore I chose the task size to be one 'row'.

The 'ThreadController' object assigns row to threads and writes result of calculation to the 'shared variable'. Therefore, 'generateRowIndex()' and 'writeResult()' are protected by 'synchronized' keyword.

```
C:\Users\82104\OneDrive-CAU\4-1\멀티코어\Multicore_programming\Project1_Lab\src\Prob2>
javac MatmultD.java

C:\Users\82104\OneDrive-CAU\4-1\멀티코어\Multicore_programming\Project1_Lab\src\Prob2>
java MatmultD.java 4 < mat500.txt
[thread_no]: 4 , [Total Execution Time]: 275 ms
1 Thread: 273ms
2 Thread: 273ms
3 Thread: 273ms
4 Thread: 272ms
```

According to the results, all four threads have a good load balance, resulting about 273ms.

Java source code

```java
package Prob2;
import java.util.*;
import java.lang.*;
/*
    < command-line execution >
    Ex) 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
 */

public class MatmultD {
    private static int NUM_THREADS = 1;
    private static Scanner sc = new Scanner(System.in);

    public static void main(String[] args)
    {
        if (args.length == 1)
            NUM_THREADS = Integer.valueOf(args[0]);

        int a[][] = readMatrix();
        int b[][] = readMatrix();

        MulMatrixThread[] threads = new MulMatrixThread[NUM_THREADS];
        ThreadController controller = new ThreadController(a.length, b[0].length);

        long startTime = System.currentTimeMillis();    // program execution time starts

        // Start threads
        for (int i=0; i<NUM_THREADS; i++) {
            threads[i] = new MulMatrixThread(a, b, controller);
            threads[i].start();
        }

        // Thread join()
        try {
            for (int i=0; i<NUM_THREADS; i++) {
                threads[i].join();
            }
        } catch (InterruptedException e) {}
```

```
        long endTime = System.currentTimeMillis();  // program execution time ends

        // Print Result
        System.out.printf("[thread_no]:%2d , [Total Execution Time]:%4d ms\n", NUM_THREADS,
endTime-startTime);
        for (int i=0; i<NUM_THREADS; i++) {
            System.out.println(i+1 + " Thread: " + threads[i].diffTime + "ms");
        }
    }

    public static int[][] readMatrix() {
        int rows = sc.nextInt();
        int cols = sc.nextInt();
        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;
    }
}

class MulMatrixThread extends Thread {
    long diffTime;
    static int[][] a, b;
    int m, p, n;
    ThreadController controller;

    MulMatrixThread(int a[][], int b[][], ThreadController controller) {
        this.a = a;
        this.b = b;
        this.controller = controller;
        m = a.length;
        p = b[0].length;
        n = a[0].length;
    }

    @Override
    public void run() {
        long startTime = System.currentTimeMillis();
        int mIndex = controller.generateRowIndex();    // controller gives which row the
thread has to calculate

        while (mIndex < m) {
            controller.writeResult(mIndex, multMatrix(mIndex, n, p));
            mIndex = controller.generateRowIndex();
        }

        long endTime = System.currentTimeMillis();
        diffTime = endTime - startTime;
    }

    public static int[] multMatrix(int mIndex, int n, int p) {     // a[m][n], b[n][p]
        /*
            thread calculate one row at one time
         */
        int ans[] = new int[p];
```

```
        for (int i=0; i<p; i++) {
            for (int j=0; j<n; j++) {
                ans[i] += a[mIndex][j] * b[j][i];
            }
        }

        return ans;
    }
}

class ThreadController {
    // a[m][n], b[n][p] -> result[m][p]
    public static int mIndex = 0;
    public static int[][] result;
    int m, p;

    ThreadController(int m, int p) {
        result = new int[m][p];
        mIndex = -1;
        this.m = m;
        this.p = p;
    }

    public synchronized int generateRowIndex() {
        mIndex += 1;
        return mIndex;
    }

    public synchronized void writeResult(int index, int[] indexArray) {
        /*
            Thread can write the result with this method.
            indexArray: one row the thread calculated
            the lenght of indexArray should be 'p'
         */

        for (int i=0; i<p; i++) {
            result[index][i] = indexArray[i];
        }
    }
}
```

Screen capture image of program execution and output

```
C:\Users\82104\OneDrive-CAU\4-1\멀티코어\Multicore_programming\Project1_Lab\src\Prob2>
javac MatmultD.java

C:\Users\82104\OneDrive-CAU\4-1\멀티코어\Multicore_programming\Project1_Lab\src\Prob2>
java MatmultD.java 4 < mat500.txt
[thread_no]: 4 , [Total Execution Time]: 275 ms
1 Thread: 273ms
2 Thread: 273ms
3 Thread: 273ms
4 Thread: 272ms
```

How to compile and execute the source code

- `$ java MatmultD.java 4 < mat500.txt`