Parallel Programming

Laboratory 5

~ 2022 ~

Balázs Benedek

Group 30444/1

Problem 1:

Value = 8500

M = 6 * 1.3 = 7.8

PROCESSES	Execution time [seconds]	[Relative] Speedup S(n) = T(1)/T(n)	[Relative] Efficiency E(n) = S(n) / M
1	158284	1	0.1282051282
6	31739	4.987050632	0.6393654656
12	27048	5.851966874	0.7502521633
24	26913	5.881321295	0.7540155506

TABLE 1. Performance parameters for Problem 1

```
import java.io.FileWriter;
import java.io.IOException;
import java.math.BigDecimal;
import java.math.BigInteger;
import java.math.MathContext;
import java.util.ArrayList;


public class FibonacciNumbers extends Thread {
    int N;
    int maxThreads;
    int threadIndex;
    BigInteger[] results;

    FibonacciNumbers(int N, int maxThreads, int threadIndex, BigInteger[]
results)
    {
        this.N = N;
        this.maxThreads = maxThreads;
        this.threadIndex = threadIndex;
        this.results = results;
    }

    public void run() {
        MathContext MC = new MathContext(this.N);
        BigDecimal FIVE = BigDecimal.valueOf(5.0).sqrt(MC);
        BigDecimal TWO = BigDecimal.valueOf(2.0);

        BigDecimal PHI = BigDecimal.ONE.add(FIVE, MC).divide(TWO, MC);

        int initialIndex = this.threadIndex;
        while (initialIndex <= this.N) {
            this.results[initialIndex] = (PHI.pow(initialIndex, MC))
                    .divide((FIVE), MC)
                    .add(BigDecimal.valueOf(0.5), MC).toBigInteger();
            initialIndex += this.maxThreads;
        }
    }

    public static void main(String[] args) throws InterruptedException,
IOException {
        long sp = System.currentTimeMillis();
        int N = 8500;

        int maxThreads = 24;
        BigInteger[] results = new BigInteger[N + 1];

        ArrayList<FibonacciNumbers> fibs = new ArrayList<>();

        for (int i = 1; i <= maxThreads; i++)
        {
            FibonacciNumbers t = new FibonacciNumbers(N, maxThreads, i, results);
            fibs.add(t);
        }

        for (Thread f: fibs)
        {
            f.start();
        }

        for (Thread f: fibs)
        {
            f.join();
        }
        System.out.println("spent (ms): " + (System.currentTimeMillis() - sp));
        FileWriter myWriter = new FileWriter("Fibonacci.txt");
        for (int i = 1; i <= N; i++) {
            myWriter.write("F(" + i + "): " + results[i] + "\n");
        }
        myWriter.close();
    }
}
```