Student: Irakli Kelbakiani

Université de Neuchâtel

Concurrency: Multi-core Programming and Data Processing

Homework 1

Student: Irakli Kelbakiani

Email: Irakli.Kelbakiani@students.unibe.ch

Student: Irakli Kelbakiani

Assignment 1:

March 17,2022

Github repo: https://github.com/KelbakianiIrakli/Multi-core-Programming/tree/concurrency-hw1/src/hw1

Exercise 1.1

Write a Java program that takes as arguments two integers, T and N, and forks T threads that will together search for and print all primes between 1 and N. The program should evenly split the range [1..N] into T sub-ranges assigned to each thread.
Execute the program with N={10'000'000,100'000'000} and T={1,2,4,8,16}. Report execution times.

Please, see Exercise1.java in my repo. I first calculate the results for same N, for each number of threads in T.

```
99999959
99999971
99999989
[11.108, 6.855, 3.812, 2.618, 2.425, 276.038, 175.535, 69.884, 48.819, 42.733]
```

Exercise 1.2

Modify the program of 1.1 so that it uses a shared counter to assign the next number to test to the threads. The shared counter should be protected from concurrent accesses by a monitor (e.g., using synchronized method). As before, execute the program and print the results.
Please, see Exercise2.java in my repo. I first calculate the results for same N, for each number of threads in T.

```
99999959
99999971
99999989
[15.879, 8.027, 4.539, 3.273, 2.998, 439.071, 226.495, 124.277, 81.929, 66.26]
```

Exercise 1.3

Consider the classical producer-consumer problem: a group of P producer threads and a group of C consumer threads share a bounded circular buffer of size N. If the buffer is not full, producers are allowed to add elements; if the buffer is not empty, consumers can consume elements. Write a program that can correctly coordinate the producers and consumers and their depositing and retrieving activities. For simplicity we assume that we

have the same number T of producers and consumer threads. T and N are program arguments.
Please, see Exercise3.javaand LockBasedQueue.java in my repo. I first calculate the results for same N, for each number of threads in T.

Exercise 1.4

You have a choice between buying one uniprocessor that executes five zillion instructions per second, or a ten-processor multiprocessor where each processor executes one zillion instructions per second. Using Amdahl's Law, explain how you would decide which to buy for a particular application.

Please, see Exercise 1.4.pdf handwritten solution in pdf in my repo.