King Saud University
College of Computer and Information Sciences
Department of Computer Science
CSC453 – Parallel Processing – Tutorial No 9 – Autumn 2022

Question

Let's consider the following parallel Java code that calculates, in parallel, the number of occurrences of the number 3 in an array.

```
public class Count3sParallel1 implements Runnable {
        int array[];
        int count, nbThread;
        Thread t;
        LinkedList<Integer> threadIds = new LinkedList<Integer>();

        public void count3s() {
                count =0;
                for (int i=0; i < nbThread; i++) {
                        t = new Thread(this);
                        threadIds.add(new Integer(i));
                        t.start();
                }
        }


        public void run() {
                int depth = (array.length / nbThread);
                int start = threadIds.poll().intValue() * depth;
                int end = start + depth;

                for (int i = start; i < end; i++ ) {
                        if (array[i] == 3)
                                count ++;
                }
        }
}
```

1. What is the main problem of this parallel code.
2. How can we fix this problem.
3. How can we enhance the performance of the solution which you have described in 2.
4. Explain the relationship between parallelism and throughput and latency.
5. According to Amandahl's Law, what is the maximum performance of a processing performed by p processors.

1.race condition, risk of overwriting each other.
2.by using locks or mutex objects. so only one thread can access the vairable.
3. we reduce the idle time by reducing the number of times threads requests locks or mutex.
we do that by creating a new local count variable that only request mutex or locks once after its done counting.
4.the main objective of parallelism is to reduce latency and increase throughput.
5. (1/s)*Ts+((1-(1/s))*(Ts/p)