16-124-836
Marcel ZAUDER

Question 01: Prime Numbers

In the following section, I will recap the results of the *primeFinder*-algorithm. I also added the cases of 4^{11}, 4^{12}, 4^{13}, and 4^14 because the difference at 4^{10} between the range and ticketing approach was nearly negligible:

Case 1: 4^{10}

```
Number of Primes: 82025
Time with RangeFinder: 173 ms
Thread 0 finished at time: 68 ms
Thread 1 finished at time: 130 ms
Thread 2 finished at time: 120 ms
Thread 3 finished at time: 125 ms

Number of Primes: 82025
Time with TicketManager: 184 ms
Thread 0 finished at time: 180 ms
Thread 1 finished at time: 180 ms
Thread 2 finished at time: 181 ms
Thread 3 finished at time: 181 ms

Number of Primes: 82025
Reference-Time with 1 Thread: 347 ms
```

RANGEFINDER

The RANGEFINDER-Threads takes 173ms to finish searching for primes. But from the output we can see that the thread which will finish first, only is computing half of the time and the rest of it he is waiting for the others to finish.

TICKETING-SYSTEM

The TICKETING-SYSTEM needs surprisingly longer than the RANGEFINDER-Threads. The cause for this could be the large overhead which is produced when each thread is fetching (synchronized) their next number to compute, so I added several more cases with more numbers to check this assumption. In contrast to the RANGEFINDER approach, here all the threads will finish at nearly the same time, so no thread will be idle for too long.

ONE-THREAD-REFERENCE + SPEEDUP

For one thread the computation takes significantly longer, so we compute the speed-up:

RANGEFINDER-SPEEDUP:
$$SpeedUp = \frac{OldExecutionTime}{NewExecutionTime} = \frac{347}{173} = 2.006$$
TICKETING-SYSTEM-SPEEDUP:
$$SpeedUp = \frac{OldExecutionTime}{NewExecutionTime} = \frac{347}{184} = 1.886$$

For both approaches the speedup is approximately 2, which is what we would

expect (faster but not 4 times faster because we have synchronized code snippets).

Case 2: 4^{11}

```
Number of Primes: 295947
Time with RangeFinder: 1184 ms
Thread 0 finished at time: 649 ms
Thread 1 finished at time: 950 ms
Thread 2 finished at time: 1052 ms
Thread 3 finished at time: 1141 ms

Number of Primes: 295947
Time with TicketManager: 1081 ms
Thread 0 finished at time: 1067 ms
Thread 1 finished at time: 1066 ms
Thread 2 finished at time: 1069 ms
Thread 3 finished at time: 1067 ms

Number of Primes: 295947
Reference-Time with 1 Thread: 1790 ms
```

RANGEFINDER

The RANGEFINDER-Threads takes 1184ms to finish searching for primes. The first one to finish needs 649ms for its range part (again approximately half the computing time of the last one).

TICKETING-SYSTEM

Now the TICKETING-SYSTEM has overtook the RANGEFINDER and terminates faster, because every thread is working all the time and there is no idle time

ONE-THREAD-REFERENCE + SPEEDUP

The ONE-THREAD-REFERENCE needs 1790ms, the speedup is therefore:

RANGEFINDER-SPEEDUP:
$$SpeedUp = \frac{OldExecutionTime}{NewExecutionTime} = \frac{1790}{1184} = 1.512$$
TICKETING-SYSTEM-SPEEDUP:
$$SpeedUp = \frac{OldExecutionTime}{NewExecutionTime} = \frac{1790}{1081} = 1.656$$

In this case it is still faster but the speedup is not as huge as before which could be because of the very small time differences in the first case.

Case 3: 4^{12}

```
Number of Primes: 1077871
Time with RangeFinder: 7266 ms
Thread 0 finished at time: 3768 ms
Thread 1 finished at time: 5454 ms
Thread 2 finished at time: 6291 ms
Thread 3 finished at time: 7190 ms

Number of Primes: 1077871
Time with TicketManager: 4656 ms
Thread 0 finished at time: 4640 ms
Thread 1 finished at time: 4640 ms
Thread 2 finished at time: 4640 ms
Thread 3 finished at time: 4640 ms

Number of Primes: 1077871
Reference-Time with 1 Thread: 14103 ms
```

RANGEFINDER

The RANGEFINDER-Threads takes 7266ms to finish searching for primes. The first one to finish needs 3768ms for its range part (again approximately half the computing time of the last one).

TICKETING-SYSTEM

Now the TICKETING-SYSTEM is much faster than the RANGEFINDER with only 4656ms.

ONE-THREAD-REFERENCE + SPEEDUP

The ONE-THREAD-REFERENCE needs 14103ms, the speedup is therefore:

RANGEFINDER-SPEEDUP:

$$SpeedUp = \frac{OldExecutionTime}{NewExecutionTime} = \frac{14103}{7266} = 1.941$$

TICKETING-SYSTEM-SPEEDUP:

$$SpeedUp = \frac{OldExecutionTime}{NewExecutionTime} = \frac{14103}{4656} = 3.029$$

In this case we can see that the RANGEFINDER is approximately 2 times faster than the sequencial approach. The TICKETING-SYSTEM is even 3 times faster.

Case 4: 4^{13}

```
Number of Primes: 3957809
Time with RangeFinder: 38796 ms
Thread 0 finished at time: 19126 ms
Thread 1 finished at time: 29324 ms
Thread 2 finished at time: 34896 ms
Thread 3 finished at time: 38693 ms

Number of Primes: 3957809
Time with TicketManager: 33898 ms
Thread 0 finished at time: 33802 ms
Thread 1 finished at time: 33820 ms
Thread 2 finished at time: 33819 ms
Thread 3 finished at time: 33819 ms

Number of Primes: 3957809
Reference-Time with 1 Thread: 84101 ms
```

RANGEFINDER

The RANGEFINDER-Threads will take 38796ms to finish searching for primes. The first one to finish needs 19126ms for its range part (again approximately half the computing time of the last one).

TICKETING-SYSTEM

Now the TICKETING-SYSTEM is faster than the RANGEFINDER with only 33898ms but not as much as before. A reason for this could be that with higher numbers there are much less primes so the method isPrime will output false more often and frequently than before.

ONE-THREAD-REFERENCE + SPEEDUP

The ONE-THREAD-REFERENCE needs 84101ms, the speedup is therefore:

RANGEFINDER-SPEEDUP:

$$SpeedUp = \frac{OldExecutionTime}{NewExecutionTime} = \frac{84101}{38796} = 2.168$$

TICKETING-SYSTEM-SPEEDUP:

$$SpeedUp = \frac{OldExecutionTime}{NewExecutionTime} = \frac{84101}{33898} = 2.481$$

In this case we can see that the multi threaded approaches are more than 2 times faster than the one threaded variant. Additionally the TICKETING-SYSTEM is again more efficient but not as much as in the scenario before, also because of the higher computation time of really big primes.

Case 5: 4^{14}

```
Number of Primes: 14630843
Time with RangeFinder: 298359 ms
Number of Primes: 14630843
Time with TicketManager: 257184 ms
Number of Primes: 14630843
Reference-Time with 1 Thread: 664402 ms
```

Because I ran the test before I have implemented the "terminating times" of each thread there are only the "full-terminating times".

RANGEFINDER

The RANGEFINDER-Threads takes 298359ms to finish searching for primes.

TICKETING-SYSTEM

Now the TICKETING-SYSTEM is significantly faster than the RANGEFINDER with only 257184ms.

ONE-THREAD-REFERENCE + SPEEDUP

The ONE-THREAD-REFERENCE needs 664402ms, the speedup is therefore:

RANGEFINDER-SPEEDUP:
$$SpeedUp = \frac{OldExecutionTime}{NewExecutionTime} = \frac{664402}{298359} = 2.227$$
TICKETING-SYSTEM-SPEEDUP:
$$SpeedUp = \frac{OldExecutionTime}{NewExecutionTime} = \frac{664402}{257184} = 2.583$$

In this case we can see that the RANGEFINDER is approximately 2 times faster than the sequencial approach. The TICKETING-SYSTEM is even 2.5 times faster. This also adds to the previous case so we can say that these might be the general speedup-factors for the algorithms.

Conclusion

In the end we can say that the RANGFINDER approach is approximately more than 2 times faster and the TICKETING-SYSTEM approximately 2.5 times faster than the sequencial variant of the algorithm.