Sieve of Eratosthenes (Finding Prime Number) by using MPI

ITCS443_Parallel and Distributed Systems

CONTENTS

ALGORITMS

We use Algorithm to find prime numbers

Then algorithms will remove non primes, leaving only primes by repeating each remaining number.

Invented by Greek mathematician named Eratosthenes (276-194 BC)

2 is the first number of prime number and it kept. All multiples of this number are deleted as they cannot be prime.

Implementation

Sequential

1) Create list of unmarked natural numbers 2, 3, ..., n.

2) k ← 2

3) Repeat

(a) Mark all multiples of k between k*k and n

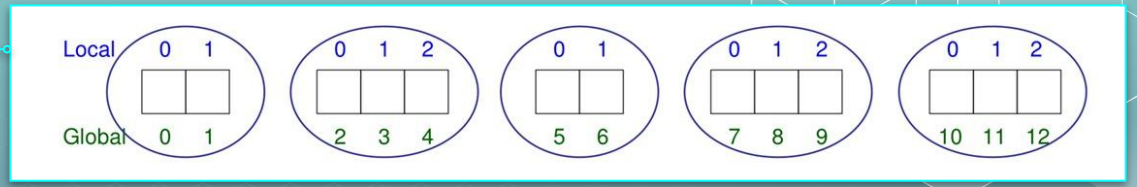
(b) k ← smallest unmarked number > k

until k * k > n

4) The unmarked numbers are primes

Implementation

1. Create list of unmarked natural numbers 2, 3, . . . , n each process creates its share of the list
2. k ← 2 all processes perform this
3. Repeat
   (a) Mark all multiples of k between k*k and n
   [each process marks its share of the list]
   (b) k ← smallest unmarked number > k
   [process 0 only]
   (c) Process 0 broadcasts k to rest of processes until k*k > n
4. The unmarked numbers are primes
5. Reduction to compute number of primes.
6. After the program finish its work it will displays the output and the total number of prime number.



Parallel Program

When n =100 and we want to use 4 processes. The block will have 4 blocks and each block contain 25 index or size will equal 25. So, it use the local indices.

```
size = BLOCK_SIZE (id,p,n);

for (i =0 ; i< size; i++) {

i = index on the each process}
```

Sequential Program

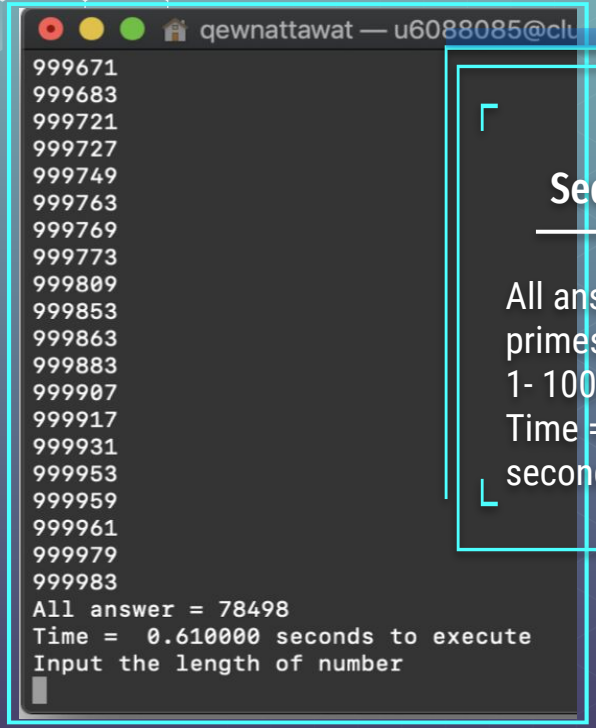
If n = 100 the sequential program will start at index 0 until 99. So, sequential program use global index.

```
for (i=0 ; i<n :i++) {

…

}
```

Evaluation

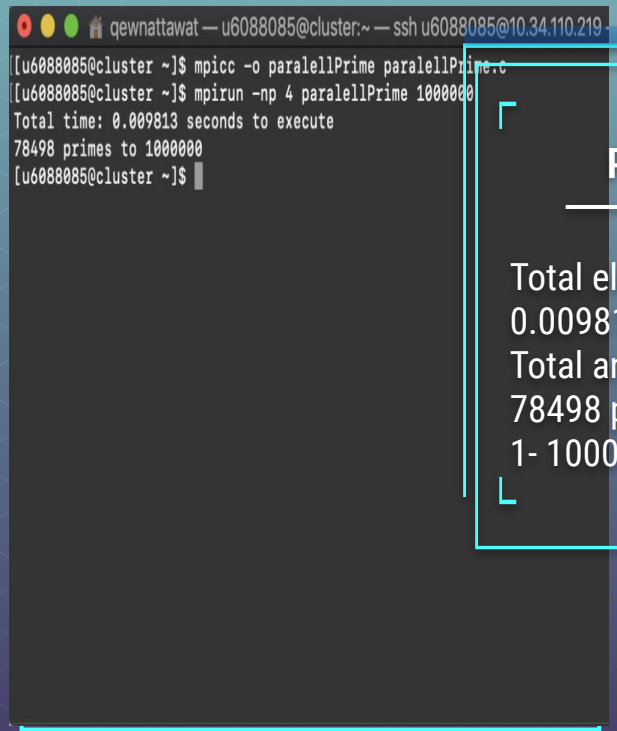
LOADING

Performance Results



Sequential

All answer = 78498 primes from 1- 1000000
Time = 0.610000 seconds to execute



Parallel

Total elapsed time: 0.009813 seconds
Total answer = 78498 primes from 1- 1000000

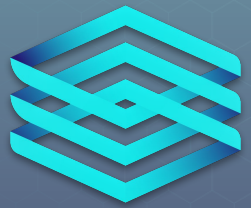
Discussion about the results

The difficult part is writing a code from sequential process to make it operate together or how to make each process independent since we split into blocks. Other outside blocks can not know how far that process is running. Due to that, we need to use broadcast command to find k value to other blocks that they can access same value. Then, we need to MPI_reduce to sum up the results from all the blocks. Moreover, if the size of data is too large the runtime will be very slow.

Analysis of the algorithm

The algorithm that we used will balance the process and speed up the runtime. Moreover, it is very complicated to define which block belong to what process.

Further optimization



Remove the even numbers that is not number 2 or not add to the block at all so it will be easier to search and limited the communication so runtime process will take shorter.

MEMBERS

- Sasirat Harnwatthanasiri 6088024
- Pantita Wang 6088219
- Phummarat Yosamornsuntorn 6088233

- Nattawat Lumtansawan 6088085
- Waris Vorathumdusadee 6088128
- Sirichoke Yooyen 6088232