Assignments

Assignment 3 - In progress

Honor Pledge Accepted
Draft - In progress
Submitted
Returned

Assignment Details

Title
Assignment 3
Due
Mar 7, 2023 11:55 PM
Number of resubmissions allowed
Unlimited
Accept Resubmission Until
Mar 9, 2023 11:55 PM
Status
Honor Pledge Accepted
Grade Scale
Points (max 100.00)

Instructions

Assignment overview

The mathematics behind prime numbers is a hot topic in Computer Science.

- The number of primes less than N is ~ N/log(N)
- If there are X primes less than N, then the sum of all primes less than or equal to N is ~ $(X^2 \ln(X))/2$

For example, the number of primes less than 25 is 9 (2, 3, 5, 7, 11, 13, 17, 19, 23) and the sum of those primes is 100. The number of primes less than 100 is 25 and the sum is 1060.

For this assignment, write a C program which will accept two command-line parameters which indicate the number of threads to create and the maximum number to calculate prime numbers up to (denoted N). The program will then count the number of primes up to N and the sum of those primes. This task will be accomplished by dividing the task across multiple threads.

Purpose

The goals of this assignment are the following:

- Learn how to use multi-threading and data parallelism to accomplish tasks faster
- Get experience with the `pthread_create()` and `pthread_join()` system functions
- Gain more experience with the C programming language from an OS perspective

Computing platform

You are welcome to develop your program on your own workstation if you wish, but you are responsible for ensuring that your program compiles and runs without error on the Gaul computing platform. Marks will be deducted if your program fails to compile, or your program runs into errors on Gaul.

- https://wiki.sci.uwo.ca/sts/computer-science/gaul

Instructions

Attached to this assignment is a tarball with the following files in it. **None of these files should be modified**:

```
Makefile          <--- A pre-packaged Makefile. This tells you how your program should be structured
run-assignment.sh <--- A shell script that will automatically run your program
```

Download this tarball and upload it to Gaul. Extract the tarball (`tar -xvf Assignment-3.tar`). Change to the `Assignment-3` directory.

You will write a program called `assignment-3.c`. This program will:

- Accept two command-line parameters. (For simplicity, you can assume both parameters are integers)
  1. The first command-line parameter indicates how many threads your program should create
  2. The second command-line parameter indicates the maximum number your prime counting/summing program should go up to.
- Your program will then employ data parallelism to divide the task of counting and summing all primes across all threads equally. For example, if 1 thread is requested and the maximum number is 100, your program should create 1 thread to count all the primes and keep a sum of all primes from 0 - 100. However, if 4 threads are requested and the maximum number is 100, thread 1 should count all the primes and keep a sum from 0 - 25, thread 2 for 25-50, thread 3 for 50-75, and thread 4 for 75-100.
- Your program will then gather all the counts and sums from all the threads and sum everything together to produce a grand sum showing the number of primes less than the maximum number provided and the sum of all those primes. You can pass data back to the parent process via pipe, shared global variable, or by passing data to the thread by reference.

Your program should be able to handle any number up to 1,000,000.

Output

Executing `./assignment-3 1 25` should produce the following output:

```
Thread # 0 is finding primes from low = 0 to high = 25
Sum of thread 0 is 100, Count is 9

        GRAND SUM IS 100, COUNT IS 9
```

Executing `./assignment-3 4 25` should produce the following output:

```
Thread # 0 is finding primes from low = 0 to high = 7
Sum of thread 0 is 10, Count is 3
Thread # 3 is finding primes from low = 21 to high = 25
Thread # 1 is finding primes from low = 7 to high = 14
Sum of thread 1 is 31, Count is 3
Thread # 2 is finding primes from low = 14 to high = 21
Sum of thread 2 is 36, Count is 2
Sum of thread 3 is 23, Count is 1

        GRAND SUM IS 100, COUNT IS 9
```

Executing `./assignment-3 25 100` should produce the following output:

```
Thread # 1 is finding primes from low = 4 to high = 8
Thread # 0 is finding primes from low = 0 to high = 4
Thread # 4 is finding primes from low = 16 to high = 20
...
Sum of thread 22 is 89, Count is 1
Sum of thread 23 is 0, Count is 0
Sum of thread 24 is 97, Count is 1

        GRAND SUM IS 1060, COUNT IS 25
```

and `./run-assignment.sh 3` should produce the following output:

```
ASSIGNMENT 3 STARTED - Dow Mon  ## ##:##:## AM/PM EST 2023

Cleaning environment
----------------------------
rm -f assignment-3

Checking environment
----------------------------
4b02c9d49c0df28c2b349926e435fb10  ./run-assignment.sh
**********UNIQUE ID**********  assignment-3.c
Makefile: OK
```

```
Building environment
----------------------------
make all
make[1]: Entering directory '/home/wbeldman/3305/Projects/Assignment 3/Assignment-3'
gcc -o assignment-3 assignment-3.c -Wall -Wpedantic -Wextra -std=gnu17
make[1]: Leaving directory '/home/wbeldman/3305/Projects/Assignment 3/Assignment-3'

Assignment 3
----------------------------
Proper usage is ./assignment-3 <threadCount> <highestInt>
Proper usage is ./assignment-3 <threadCount> <highestInt>
Proper usage is ./assignment-3 <threadCount> <highestInt>
Thread # 0 is finding primes from low = 0 to high = 100
Sum of thread 0 is 1060, Count is 25

        GRAND SUM IS 1060, COUNT IS 25

real    0m0.003s
user    0m0.000s
sys     0m0.002s
Thread # 0 is finding primes from low = 0 to high = 20
Thread # 2 is finding primes from low = 40 to high = 60
Thread # 3 is finding primes from low = 60 to high = 80
Sum of thread 0 is 77, Count is 8
Thread # 4 is finding primes from low = 80 to high = 100
Thread # 1 is finding primes from low = 20 to high = 40
Sum of thread 1 is 120, Count is 4
Sum of thread 2 is 243, Count is 5
Sum of thread 3 is 351, Count is 5
Sum of thread 4 is 269, Count is 3

        GRAND SUM IS 1060, COUNT IS 25

real    0m0.002s
user    0m0.000s
sys     0m0.002s
Thread # 0 is finding primes from low = 0 to high = 1000
Sum of thread 0 is 76127, Count is 168

        GRAND SUM IS 76127, COUNT IS 168

real    0m0.002s
user    0m0.001s
sys     0m0.001s
Thread # 0 is finding primes from low = 0 to high = 20
Thread # 4 is finding primes from low = 80 to high = 100
Thread # 7 is finding primes from low = 140 to high = 160
Thread # 1 is finding primes from low = 20 to high = 40
Thread # 10 is finding primes from low = 200 to high = 220
...

Cleaning environment
----------------------------
rm -f assignment-3

ASSIGNMENT 3 COMPLETED - Dow Mon ## ##:##:## AM/PM EST 2023
```

Helpful hints

- You can verify your results using Wolfram Alpha. Eg.
  - https://www.wolframalpha.com/input?i=integer+primes+less+than+25

- https://www.wolframalpha.com/input?i=sum+of+integer+primes+less+than+25
- The sum of all primes less than 1,000,000 is larger than what can be represented in a `signed int` on Gaul. Use a larger type instead to hold the sum.
- If the number of threads does not evenly divide your maximum number, the last thread will need to have its maximum number set manually. For example, 4 threads from 0-10 would mean: 0-3, 3-6, 6-9, and 9-10.
- You can find code to generate prime numbers here

Submitting

When you are finished your assignment, follow these steps

1. From inside the Assignment-3 directory, run the following command (make sure `run-assignment.sh` is executable): `script -c './run-assignment.sh 3' assignment-3.out`
   Your directory should now contain the following files:

```
assignment-3.c    <--- Your program
assignment-3.out  <--- The output produced by running the script command above
Makefile          <--- A pre-packaged Makefile. This tells you how your program should be structured
run-assignment.sh <--- A shell script that will automatically run your program and put the results in assignment-3.out
```

2. Assuming the command was successful, run the follow command to get out of the `Assignment-3` directory: `cd ..`
3. Package your assignment into a tarball: `tar -cvf Assignment-3.tar Assignment-3`
4. Verify the contents of your tarball (`tar -tvf Assignment-3.tar`) (`du -sh Assignment-3.tar`). **If your tarball is 10kb in size** you have an empty tarball and you made an error on this step. Make sure you are properly creating your tarball with the right files in it.
5. Use an SFTP program to download the tarball and then upload it to OWL.

Additional resources for assignment

- Assignment-3.tar (10 KB; Feb 8, 2023 9:25 pm)

Grading Rubric

Preview Rubric

Submission

Attachments

No attachments yet

Select a file from computer Choose File No file chosen

Proceed Preview Save Draft Cancel Don't forget to save or proceed!