Problem set 5, MPI & OpenMP Intro

TDT4200, Fall 2015

Deadline: 22.10.2015 at 23:59. Contact course staff if you cannot meet the deadline.

Evaluation:

6/10 points for functioning OpenMP implementation,
10/10 for functioning hybrid OpenMP/MPI implementation.

Delivery: Use It's Learning. Deliver exactly one file:

- *yourNTNUusername_code_ps5.{zip |tar.gz |tar}* containing your modified versions of the file(s):
  - `pythagoreanTriplets.c`

General notes: Code must compile and run on the following systems:

1. `its-015-XX.idi.ntnu.no`
2. `Problem_set_5` in the *TDT4200_H2015* group on `climb.idi.ntnu.no`.

Part 1, Code

This time, you are supposed to create `pythagoreanTriplets.c`, which counts all of the Pythagorean triplets (defined below), within a given range.

Primitive Pythagorean triplets

Any non-negative integers a, b, and c make a Pythagorean triplet if they satisfy condition (1).

$$a^2 + b^2 = c^2 \tag{1}$$

A Pythagorean triplet is primitive if: (1) holds true for the non-negative integers $\{a, b, c\}$, (1) holds true for a different set of non-negative integers $\{x, y, z\}$, and there exist no integer $k > 1$ which satisfies condition (2).

$$\begin{aligned} k \cdot x &= a \\ k \cdot y &= b \\ k \cdot z &= c \end{aligned} \tag{2}$$

The program

The range is given as two values at execution, $start$ and $stop$. Valid input values for $start$ and $stop$ are all non-negative integers. $start$ should always be smaller than $stop$.

Your program ***must*** implement at least two for-loops, iterating over non-negative integers a primitive Pythagorean triplet may consist of. You may, of course, effectivize the for-loops by skipping select values that you're sure won't give you a result, but your program must be based on the premise of having at least two loops iterating over non-negative integers[1].

If your executable is intended to run OpenMP (OMP), then you should compile it with the flag `-DHAVE_OPENMP`. Likewise, if you compile your program into an MPI executable, you should compile it with the flag `-DHAVE_MPI`. If you are compiling a hybrid solution, you need to add both flags. These are the flags used on CMB, and in our scripts, to test your OMP/MPI implementations. If neither flag is present, your program should be compilable, and run serially, as described below.

Output is *only* "%d\n", where d is an integer giving the number of primitive Pythagorean triplets found in the range given.

[1] There exists a much faster algorithm for finding primitive Pythagorean triplets, which does *not* need for-loops iterating through a, b, and c. However, this is an exercise in learning to use MPI and OpenMP, and thus, this more effective algorithm is not suitable and ***should not*** be used for this PS.

Input, output, and execution

Input: $start$, $stop$

Output: `printf("%d\n", amountOfPrimitiveTripletsFound);`

1. Your program should work when executed serially like this;
   - `./executable < input.txt`
2. and with OpenMP like this (when compiled with `-DHAVE_OPENMP`);
   - `./executable < input.txt`
3. and with MPI like this (when compiled with `-DHAVE_MPI`);
   - `mpirun -n <numberOfProcesses> ./executable < input.txt`
4. and with OpenMP and MPI combined (when compiled with `-DHAVE_OPENMP -DHAVE_MPI`), like this:
   - `mpirun -n <numberOfProcesses> ./executable < input.txt`

`input.txt` will be structured as follows:

```
<N>
<start1> <stop1> <*t1>
...
<startN> <stopN> <*tN>
```

*The third parameter per line in the input, `<tX>`, may not always be there. Your program should be able to handle this, and by default set the number of threads equal to 1.

Additional requirements and things to keep in mind

With the above requirements in mind, the following should also hold true for your program:

a) $start$ should always be assigned to one of the three numbers making a Pythagorean triplet. If we call this value of the triplet a, and the other two b and c, the following should always hold true: $c > b > a$. Thus, your program should never have to check a Pythagorean triplet where $a == b$, $b == c$, or $a == c$.

1) If $c > b > a$, then c should be checked for each value in the range $[start, stop)$.
2) b should be checked for each value in the range $[4, c)$.
3) And a should be checked for each value in the range $[3, b)$.

Your program will be tested for the following things:

a) OpenMP version with the number of threads ranging from $[1, 10)$.

b) MPI version with the number of processes being $[1, 10)$.

c) Any possible combination of OpenMP threads and MPI processes, with the above two ranges of threads/processes.

d) Valid and invalid input.

- Invalid input ranges should just print 0 to terminal as the sum of primitive Pythagorean triplets found.
- Valid values for $start$ and $stop$ will be no higher than $20\ 000$.

Additional details can be found in the recitation slides for this Problem set.