Boost.Asio part2

making a thread pool using thread_group

Concurrent Programming

Scalable Computing Systems Laboratory
Hanyang University

Introduction

- What is the thread_group of Boost.Asio?
- Practice

What is the thread_group?

- thread_group provides for a collection of threads that are related in some fashion
- New threads can be added to the group with *add_thread()* and *create_thread()* member function
- You can use the *post()* function of the io_service to pass a job to the thread in the group

Simple thread pool with thread_group

```cpp
#include <iostream>
#include <boost/asio.hpp>
#include <boost/bind.hpp>
#include <boost/thread.hpp>

#define NUM_THREADS_IN_POOL  4

void Print() {
    std::cout << "Hi, i'm thread " << boost::this_thread::get_id() << std::endl;
}

int main(void) {
    boost::asio::io_service io;
    boost::thread_group threadpool;
    boost::asio::io_service::work* work = new boost::asio::io_service::work(io);

    for (int i = 0; i < NUM_THREADS_IN_POOL; i++) {
        threadpool.create_thread(boost::bind(
                    &boost::asio::io_service::run, &io));
    }

    for (int i = 0; i < 10; i++) {
        io.post(Print);
        sleep(1);
    }

    io.stop();
    threadpool.join_all();

    return 0;
}
```

Simple thread pool with thread_group

```
$ g++ -o prac_threadgroup prac_threadgroup.cpp –lboost_system –lboost_thread -lpthread
```

```
jongbin@multicore-96:~/TA/Multicore/lab07$ ./prac_threadgroup
Hi, i'm thread 7fe09fffe700
Hi, i'm thread 7fe0a07ff700
Hi, i'm thread 7fe09fffe700
Hi, i'm thread 7fe09f7fd700
Hi, i'm thread 7fe09effc700
Hi, i'm thread 7fe0a07ff700
Hi, i'm thread 7fe09fffe700
Hi, i'm thread 7fe09f7fd700
Hi, i'm thread 7fe09effc700
Hi, i'm thread 7fe0a07ff700
```

Simple thread pool with thread_group

```cpp
#include <iostream>
#include <boost/asio.hpp>
#include <boost/bind.hpp>
#include <boost/thread.hpp>

#define NUM_THREADS_IN_POOL  4

void Print() {
    std::cout << "Hi, i'm thread " << boost::this_thread
}

int main(void) {
    boost::asio::io_service io;
    boost::thread_group threadpool;
    boost::asio::io_service::work* work = new boost::asio::io_service::work(io);

    for (int i = 0; i < NUM_THREADS_IN_POOL; i++) {
        threadpool.create_thread(boost::bind(
                    &boost::asio::io_service::run, &io));
    }

    for (int i = 0; i < 10; i++) {
        io.post(Print);
        sleep(1);
    }

    io.stop();
    threadpool.join_all();

    return 0;
}
```

This ensures that the io_service object's run() function will not exit while work is underway, and that it does exit when there is no unfinished work remaining

Practice

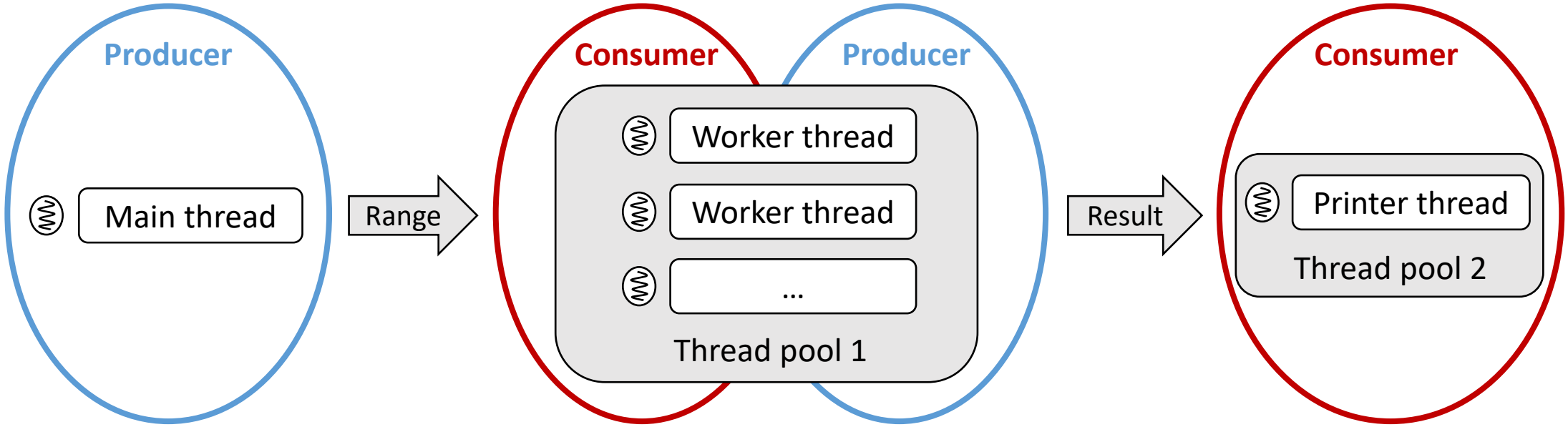
- Implement the program using thread_group
  - Two integer values *range_start* & *range_end* are given from the standard input
  - Pass this range and sequence number to a worker thread in the thread pool (thread pool size: 10)
  - The worker thread calculates the number of prime numbers in the given range, and sends the result with its sequence number to the single printer thread, which is a member of an another thread pool
  - Single printer thread prints each result in a line like below

    (*sequence_number*) number of primes in *range_start* ~ *range_end* is *result*
  - Increase the sequence number and repeat it until the given *range_start* is -1

```
jongbin@multicore-96:~/TA/Multicore/lab07$ ./prac_prime < workload.txt
(1)number of primes in 5000 ~ 8000 is 338
(3)number of primes in 2000 ~ 6000 is 480
(0)number of primes in 1 ~ 10000 is 1229
(4)number of primes in 10000 ~ 16000 is 633
(6)number of primes in 7000 ~ 10000 is 329
(5)number of primes in 350 ~ 12000 is 1368
(2)number of primes in 30000 ~ 40000 is 958
(9)number of primes in 5000 ~ 8000 is 338
```

The order of result's sequence number can be mixed, but single result should be printed in a single line

Practice


Producer
Main thread
Range
Consumer
Producer
Worker thread
Worker thread
...
Thread pool 1
Result
Consumer
Printer thread
Thread pool 2

Thank You