

Assignment 1: Threaded Servers

Course ISE 431: Distributed Information Systems

Due November 21, 2009

In class we discussed the role of threads and processes in servers and clients. In this assignment you will adapt a working threaded program to use a thread pool architecture as described in class. You may find the code for the multithreaded Java server on Telem.

The application you will be adapting is a simple web server which receives HTTP GET requests and returns the requested files. The application currently has the following behavior:

- The main thread listens on port 5000 (this can be changed in the code)
- When a request comes in on the server port, a new thread is created to handle the request.
- The new thread reads the request from the client.
- It prints out the client request and information on the server's output window.
- If the request is for a file, the server attempts to find it in the local directory. To help you find the local directory, the server process prints it out in its output window each time a thread answers a request.
- If the file is found, it is returned to the requestor. Otherwise, a custom error page is sent back.
- The thread then quits.

Since the server is multithreaded, it can handle several concurrent requests, launching a new thread for each as needed.

1 What to do

You may work on this task in groups of 2. Since there are an odd number of students, I will permit one group of 3 students.

Your first task is to change the architecture of the web server from being a spawn-on-demand system to one that uses a thread pool. A thread pool is a collection of worker threads which are created and wait for work to be assigned them. Create a thread pool of 10 worker threads. In case there are more than 10 concurrent requests, spawn a single use thread to handle each additional request.

Adapting the application to use a thread pool will require you to learn more about how thread pools are handled in Java. Read the online documentation for help.

You will also need to adjust the worker thread class to not quit after each request.

2 Runtime Measurements

Using the two implementations (spawn on demand and thread pool) perform some performance calculations. Perform the following checks on both versions of the application (at the very least):

- Response time for a single request for a new file

- Response time for repeated requests for the same file
- Response time for non-existing files
- Stress test your server by running `wget` on a multithreaded loop client program to get push up the number of simultaneous threads past 10.

Perform the experiments twice - (1) once when the client and server are on the same computer and (2) once when the client and server are on different computers connected by an Ethernet switch.

Consider which architecture is faster in each of the above cases based on the statistics you have gathered.

3 Future Work

The next phase of the assignment will be to adapt your web server to retrieve files that are stored using a Chord distributed hash table. You may use the Open Chord or Overlay Weaver Chord implementations. Keep this in mind when writing your code.

4 Grading

I will assign points for the assignment as follows:

- 80% code implementation. The code must be in Java and implement a thread pool architecture for handling concurrent web queries
- 20% report. The report must contain the elements above and include some conclusions about the relative performance of the two approaches.

I will test your assignment code using standard web browsers (Firefox, Microsoft IE), so you should test them as well.

Note: Due to its limited size, the assignment will be given a somewhat smaller weight in the calculation of assignments for the final course grade.

5 What to turn in

For this assignment, you must turn in:

- The names of the students in the submitting group
- The code for your thread pooled application
- A short report containing your runtime performance analysis

You may submit your work in one of the following ways on or before the end of November 21, 2009:

- In person via USB drive
- By uploading your code and report in a file to Telem
- Sending your code and report in a single zipped file to the course email address: `ise431@gmail.com`

Keep in mind that any submission received or time stamped after 11:59:59 on November 21, 2009 will be considered late.