Project 2: Experimental Evaluation

CS6200 Graduate Intro to OS

Extra Credit assignment

Due Sunday Oct. 14, 11:59pm AOE

Imagine you are a researcher conducting experiments on the GETFILE client-server protocol that you worked with in Project 1. Now that you’ve developed a library for that protocol, we want you to think about it from a different perspective: how does it behave under certain conditions? What would be required to instrument that behavior and configure it for different experiments?

As a researcher in this project, your job is to now design an experiment that answers those types of questions. However, to do that you must know what constitutes a good experiment. In addition, you must be able to identify the typical trends you would expect to see in such experimental scenarios.

Why is this important? As the P2L5 lecture mentions, there is more to designing an experiment than just gathering a few metrics and showing some results. The experiment must be relevant to the solution that all stakeholders are interested in.

Note that we are not asking you to run any physical experiments in this project. *Consider this project as a thinking exercise that teaches you about proper experimental design. The Flash paper and P2L5 will be helpful in successfully completing the project.*

For full credit, you will need to submit a report responding to items in Sections I – IV below. The report should be a **maximum of 2 pages long**.

1. **Instrumentation:**

Regardless of the question you want to answer, you will need to instrument your program(s) in some way. You will likely be capturing metrics such as Throughput, Response Time, or another metric that you would like to experiment with. Therefore, you will need to describe the instrumentation you would use to capture start and end times for some operations (e.g. using calls to something like *clock\_gettime*) and to keep track of number of Bytes or number of requests (if not known from the command line).

**So, how will you instrument your code to gather these metrics?**

Answer the following:

1. Which part of the GETFILE system will you instrument?
   1. Client
   2. Server
   3. Both
2. Describe how will you instrument the code to compute throughput.

For example, imagine you are computing Average Throughput = #Requests/Time. Where will you place the operations for your chosen system call to compute the start and end time of your program? What system calls will you use?

1. Describe how will you instrument the code to compute your average response time and describe what the nominator and denominator are in the formula:
   1. Average Response Time = A / B?
   2. What are the units of A and B?
   3. Where would you insert the instrumentation to capture the information necessary to compute A and B?
2. Specify a metric that you will want to capture for the experimental evaluation you will describe below. If different than 2. or 3. above, how will you instrument the code and compute the value for this metric for each experimental run?

You will not lose points if you chose to answer same as in 2. or 3. But feel free to define other metric(s) you think are important for the question you would like to explore (e.g., throughput per thread, 99th-percentile latency, bytes/sec, … )

1. **Design of Experiment: 4 different configurations**

Next, you should design an experiment to explore how multithreading affects the behavior of your server.

1. Specify the **question** that you want to answer in the experiment:

Some example questions include: “How does multithreading affect the throughput of a GetFile Server hosting many very large files?” or “How does multithreading affect the average response time for a GetFile server hosting a few small files?”

You should design the question based on the instrumentation you specified in Section I above.

1. Create four (4) experimental configurations:

Using the parameters below, create 4 configurations for your experiment with the intent to answer the question you specified earlier. A configuration consists of at least three of the following parameters.

* **Number of worker threads**
  + This parameter represents the number of worker threads used by the client or server
  + This will be specified as an unsigned integer value
* **Workload request pattern**
  + The parameter represents the file pattern that is used during testing
  + Use one or more of the following patterns: FIXED\_FILE, FIXED\_SIZE, or MIXED\_FILES
    - * If FIXED\_FILE**\***, requests will be made for a single file of a fixed file size. This should be used with the Request Size parameter below.
      * If FIXED\_SIZE**\***, requests will be made for different files of a fixed file size. This should be used with the Request Size parameter below.
    - If MIXED\_FILES**\*\***, requests will be made for different files that fall within a range of file sizes. This should be used with the Request Range parameter below.
* **Workload request size (in bytes) \***
  + This parameter represents the size (in bytes) of a fixed file
  + This will be specified as a long long integer value
  + This value must be greater than 0
  + This will only be specified if a FIXED\_FILE or FIXED\_SIZE pattern is chosen
* **Workload request range (in bytes) \*\***
  + This range represents the minimum and maximum size (in bytes) for files used in testing
  + This will be specified as two long long integer values
  + This range cannot start at 0
  + This will only be specified if the MIXED\_FILES pattern is chosen

For example, a single run of the experiment could be configured in the following way:

#\_Server\_Threads, Request\_Pattern, (Request\_Parameters)

Once you have chosen the parameters you will test with, create 4 different experimental configurations with concrete values for each **(NOTE: you are only allowed to create 4 configurations)**.

For example, some sample values might look something like the following:

* + [1, FIXED\_FILE, 1000]
  + [13, MIXED\_SIZE, 100, 1000000]

Be careful about choosing values that are meaningful for the question you want to explore. You will use these values to justify the graph you will create in section IV below and draw conclusions based on these 4 configurations. For example, picking experiments where you vary the number of threads in the server to be very close to each other (e.g. 31, 32, 33, and 34) is unlikely to provide relevant insight into how your server’s performance is affected by multithreading.

Can you see why the following would not be a relevant set of configurations?

* + [31, FIXED\_FILE, 1000]
  + [32, FIXED\_FILE, 1000]
  + [33, FIXED\_FILE, 1000]
  + [34, FIXED\_FILE, 1000]

Also consider that each configuration will produce 1 point or 1 bar in a final graph that you would expect to see (*you can assume that you can run each of them multiple times to get statistical confidence in the results).* So, choose values that can be compared appropriately within the context of your question.

1. **Machine Specification**

As your results will depend on the machine you use to perform the experiments on, provide some data to describe the hardware you will use:

* How many cores?
* How much cache?
* How much memory?
* Any other hardware specifications you think is pertinent to the experiment.

1. **“Results” and Explanation**

Finally, sketch a hypothetical graph that you could expect to generate from your experiment’s results. Then, provide the rationale for the behavior you’re observing in that graph as a conclusion to your report.

Remember not to compare “apples and oranges”!

Good luck and have fun!