Project 1 (CS739-Spring22)

1. General Feedback to All Groups

We first give feedback on things that many of you did and perhaps could use some feedback on. Specifically:

**Overall Presentation**

Make your presentation uniform! Make slides that look nice and make them all look consistent and clean. Spend time on this!

**Data Presentation**

Label all graph axes! You should never present a graph without clear labels on all axes – it just makes it harder to understand.

Use the right sized units! e.g., don’t put .0001 seconds when you can simply put .1 ms and don’t put 1000000 microseconds when you can put 1 second. Figure out what the best unit is for the granularity of measurement and use that.

Show all the data if you can. Not just averages – averaging hides information! Or, at least, look at all the data before you do the average, etc.

Graphs: generally, start at zero on the y-axis. Especially for bar graphs. Otherwise you could be misrepresenting the data.

Make graphs next to each other have same height/max value for y axes! This allows comparison across graphs.

Put the thing you’re comparing on the same graph near one another. Graphs facilitate comparison; thus, put the things you want the audience to compare near one another.

Use consistent colors across graphs, e.g., GRPC always blue, Thrift always green. That way, it’s easier to read graphs!

**Empirical Methods**

In general, you cannot meaningfully compare time across machines. Thus, when measuring, you have to think about what you can do on one machine.

A disk seek is not just an fseek! To get a disk to do a seek, you have to think about how to avoid caches (like the file system page cache, or the disk’s cache) and make sure the disk is read/writing blocks from far away spots on the disk.

Compare your measured numbers to spec sheets. What does the CPU/disk/SSD/DRAM expect to deliver? How do your numbers compare? Are there major discrepancies?

Testing should be done on the same setup, to facilitate comparison! Do not compare one system on one machine to another system on a completely different machine. This is not meaningful!

If the number looks weird, run it again. Ask why the number looks weird. Is it repeatable? What could be going on? What other numbers can you get to figure out what is going on? You can remove noise with careful experimentation!

Be curious! Figure out why things are the way they are. For example, a number of you used the C++ chronos::whatever timer. How does it work? You can find out!

**Drawing Conclusions**

Show data, and then try to draw conclusions. What does the data show? Make sure the reader/audience can see the same data, and draw the same conclusions from the data you have shown.
Don’t make guesses as to *why* - you’re probably wrong! Make a guess so as to then do subsequent measurements, and confirm/deny your guess. Having a guess is easy; showing that your reasoning is solid is hard and requires work.

Don’t draw strong conclusions when no strong conclusion is warranted! It’s better to say “I don’t know” than to put forward the wrong conclusion.

## 2. Feedback to Your Group

Good overall explanations and a coherent presentation.

### Part-1
- Detailed hardware spec reported, good!
- Using a variety of hosts for the experiments.
- Disk seek time was much lower than expected. Generally, to measure disk seek, you should use a large enough file, clear caches or make sure reads are not happening from RAM and use system calls for the IO.

### Part-2
- Running on vm is a nice touch (vm and not vm comparisons)
- Fraction of overhead in RTT - use consistent coloring scheme.
- Dip at 60000 not explained for bandwidth.
- 2 machine numbers are noisy. You can remove noise with careful experimentation!
- The inferences slide is too busy – too much stuff on it!

### Part-3
- The bar graphs don’t need lines in them. Use solid colors.
- Nice graphs showing the comparisons of all three systems - udp is fastest as expected.
- Marshalling/Unmarshalling:
  - You complex structure is not that “complex”. In the lecture of RPC paper, it was mentioned that “One thing the paper did not talk about if we really need it to work is”… POINTER. So a data structure include pointers? (e.g., map/linked list…).

### Handin
- Roster

<table>
<thead>
<tr>
<th>Member</th>
<th>Email</th>
</tr>
</thead>
<tbody>
<tr>
<td>Akshay Parashar</td>
<td><a href="mailto:aparashar@wisc.edu">aparashar@wisc.edu</a></td>
</tr>
<tr>
<td>Chahak Tharani</td>
<td><a href="mailto:ctharani@wisc.edu">ctharani@wisc.edu</a></td>
</tr>
<tr>
<td>Benita Kathleen Britto</td>
<td><a href="mailto:bbritto@wisc.edu">bbritto@wisc.edu</a></td>
</tr>
<tr>
<td>Deepak Ranganathan</td>
<td><a href="mailto:dranganathan@wisc.edu">dranganathan@wisc.edu</a></td>
</tr>
</tbody>
</table>

- Checked Handin: `bbritto`
- The slides were not included in `handin`