CMPS 202

Win 2018 Final Project

Github Repo: https://github.com/ZiyeHan/CMPS-202/

Throughput: Jackon-Core
Latency: Vanilla Music Ap

Ziye Han
zhan12@ucsc.edu

Throughput

Test Target: Jackson-Core
Github: https://github.com/FasterXML/jackson-core

Description: It is a jar plugin for JSON String serialization and deserialization

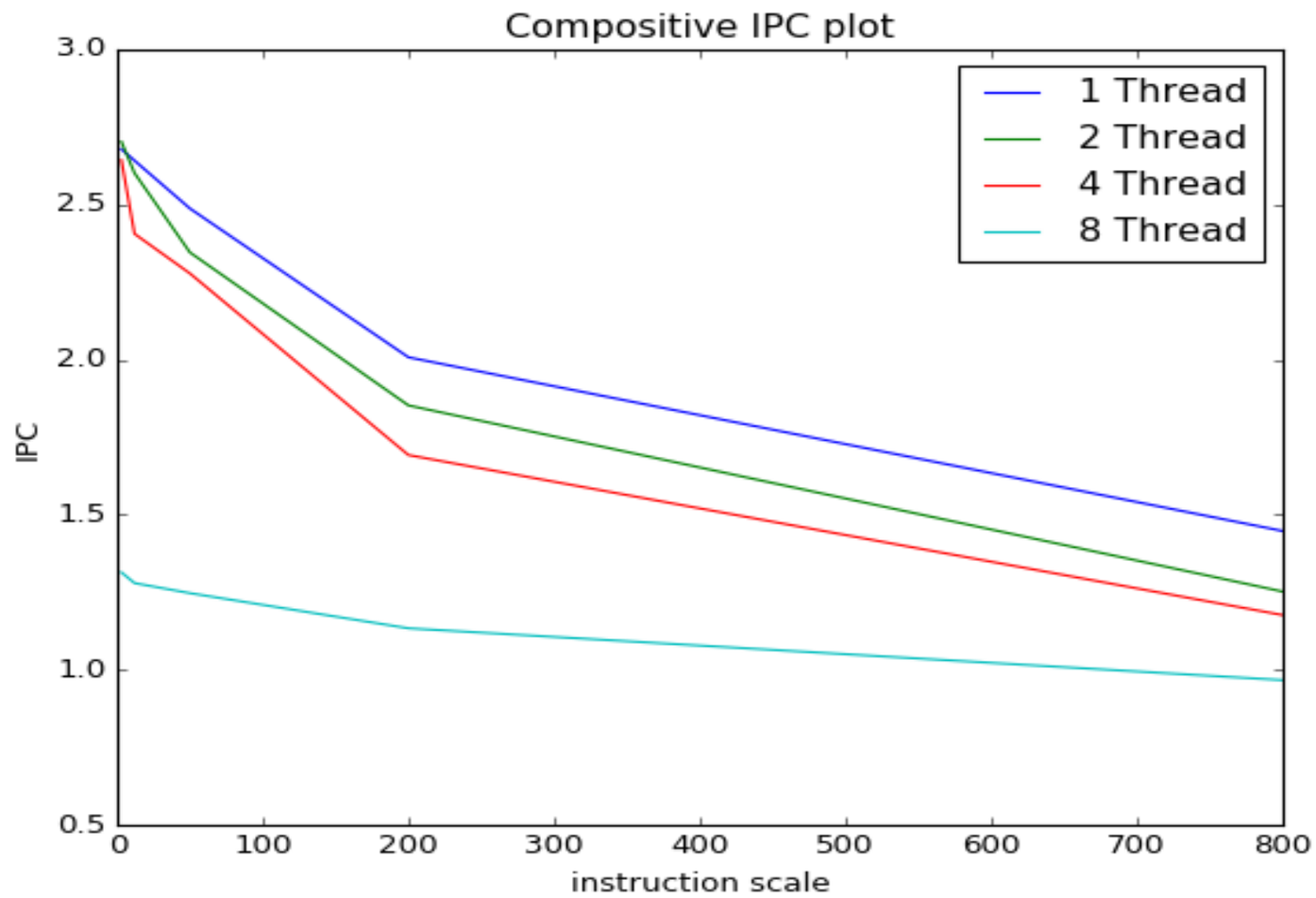
Latency

Test Target: Vanilla Music App
Github: https://github.com/vanilla-music/vanilla

Description: It is a music player on Android

What Did I Run

Throughput

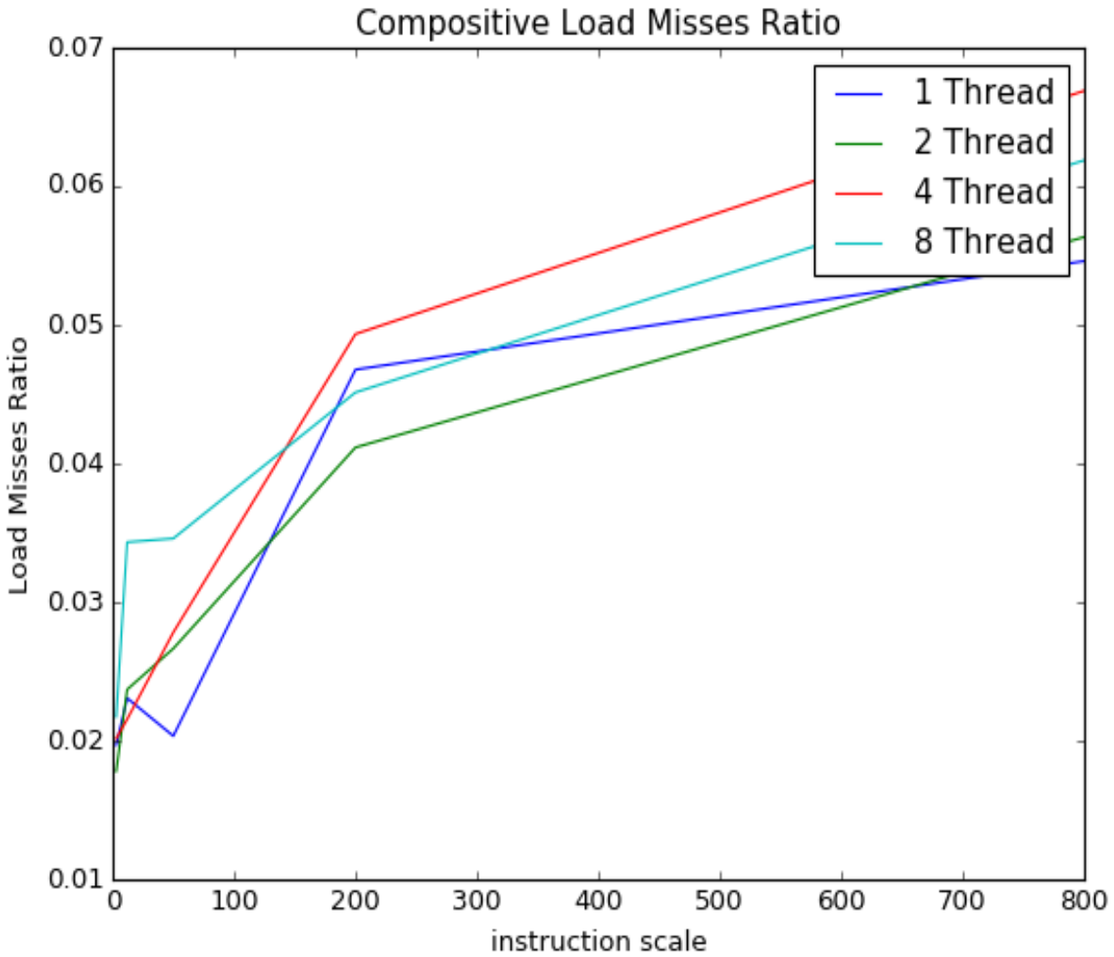


IPC

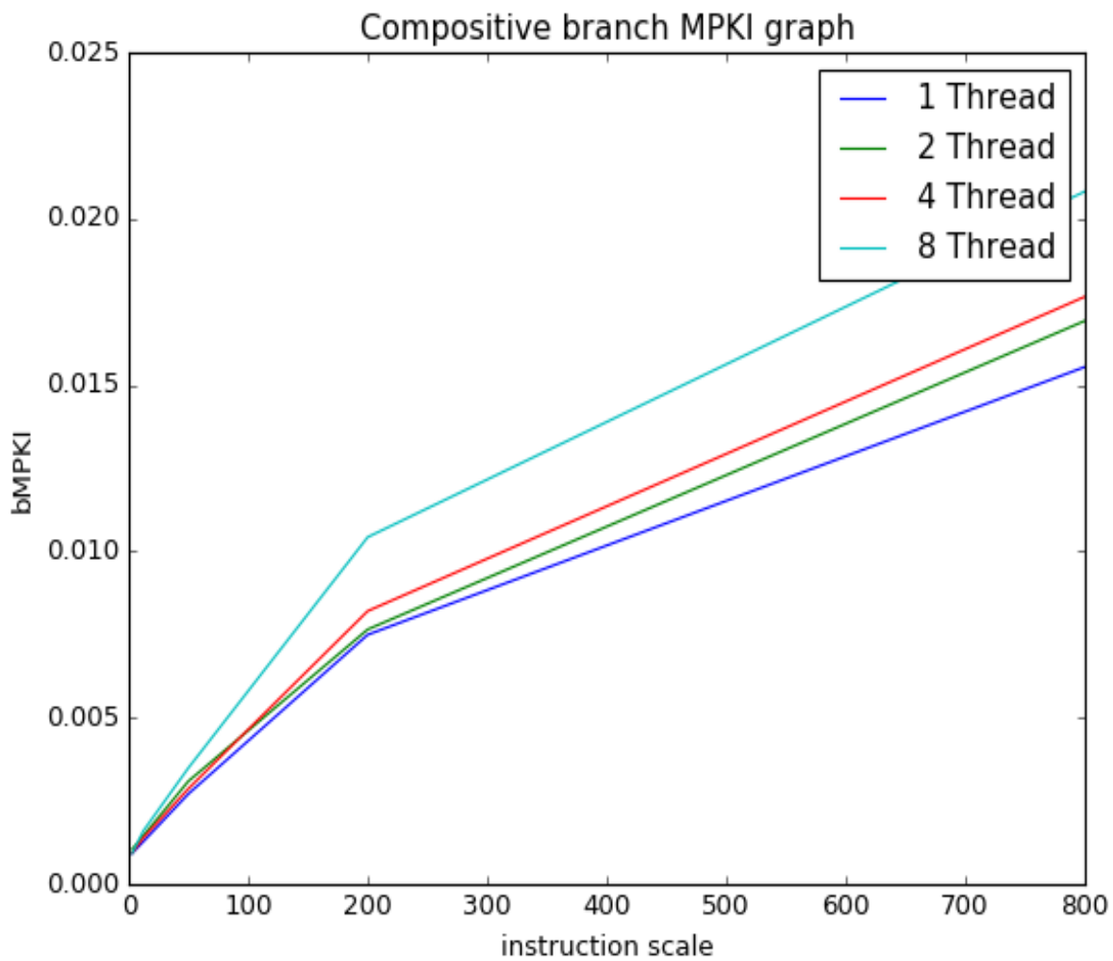
For more plots please see:
https://github.com/ZiyeHan/CMPS-202/tree/master/throughput/results

What Did I Run

Throughput



LOAD MISS RATIO OF L1

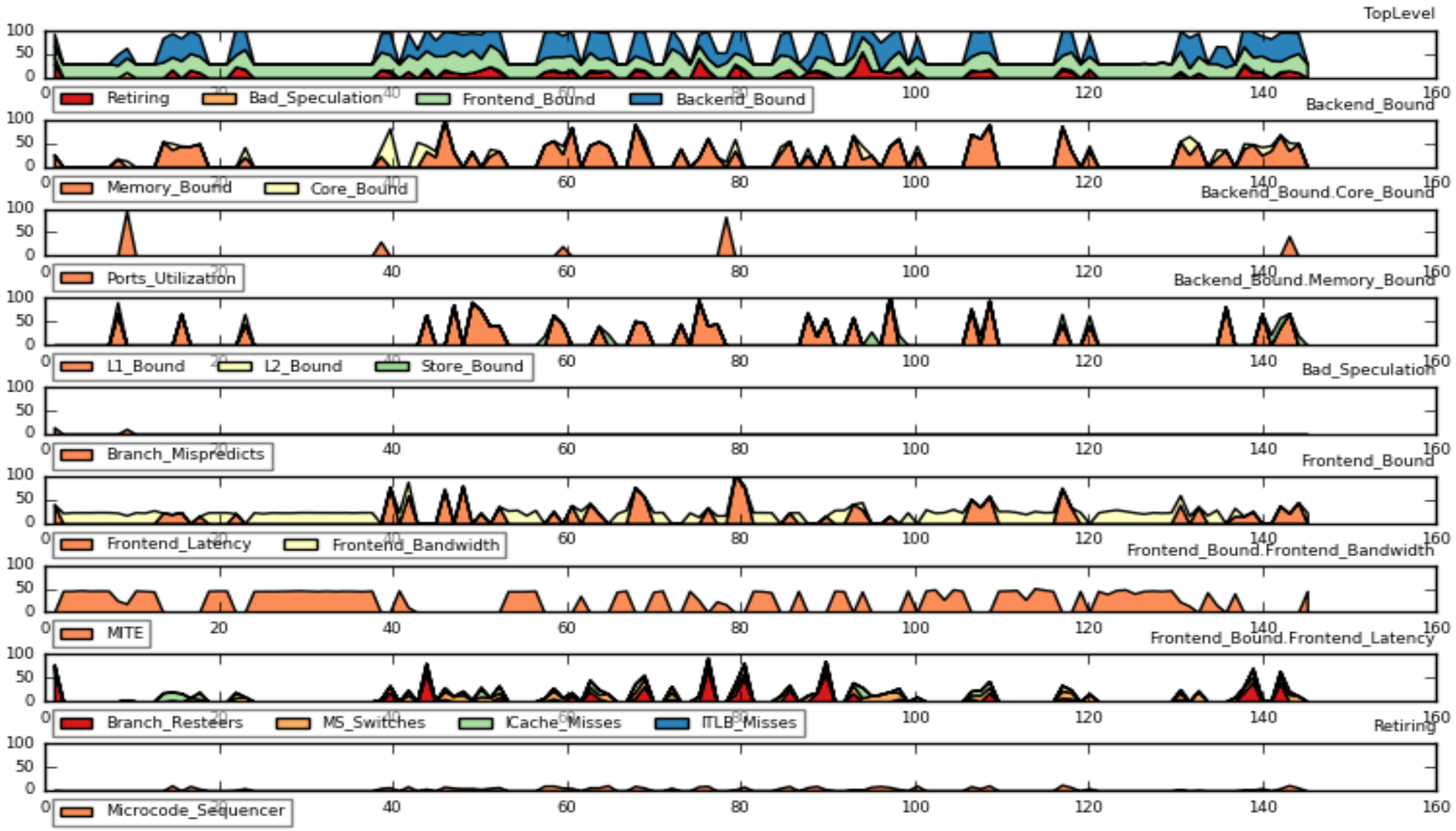


BRANCH MPKI

For more plots please see:
https://github.com/ZiyeHan/CMPS-202/tree/master/throughput/results

What Did I Run

Throughput

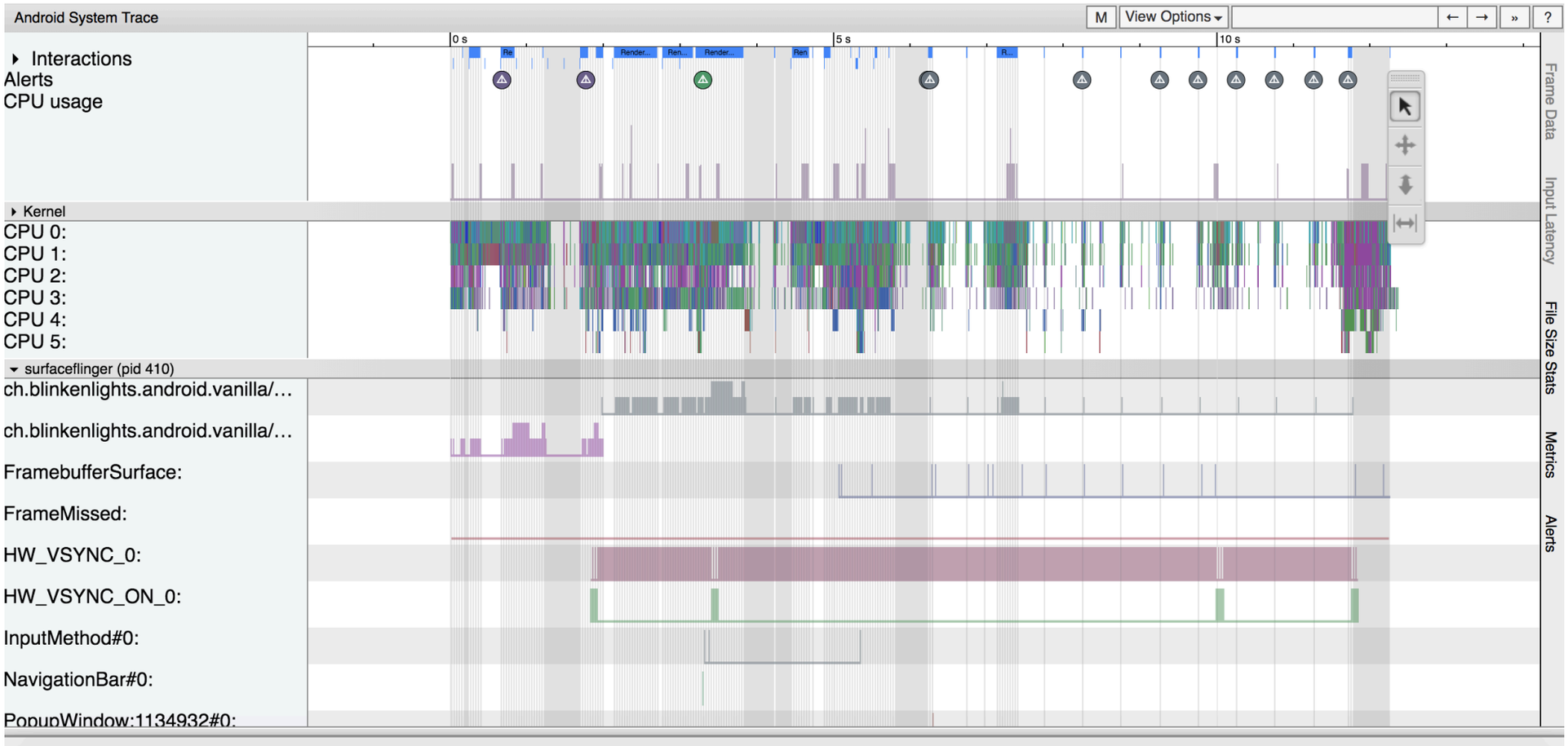


Miscellaneous

For more plots please see:
https://github.com/ZiyeHan/CMPS-202/tree/master/throughput/results

What Did I Run

Latency



systrace overall result (1 out of 10 times)

python systrace.py --time=100 -o latencyResult.html

Monitor 100s, then collect latency and generate report

adb shell monkey -p ch.blinkenlights.android.vanilla -v --throttle 200 500

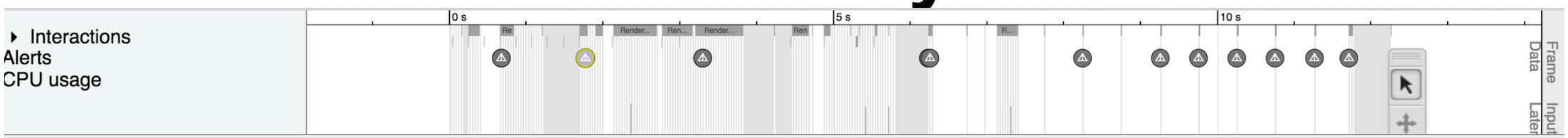
Send 500 random actions to phone, with interval 200ms

For more plots please see:

https://github.com/ZiyeHan/CMPS-202/tree/master/latency/results

What Did I Run

Latency



1 item selected. Alert (1)

Alert	Inefficient ListView recycling/rebinding
Time spent	11.073 ms
ListView items rebound	8.5
obtainView	took 0.21ms
obtainView	took 0.08ms
obtainView	took 0.05ms
obtainView	took 1.50ms
setupListItem	took 1.40ms
obtainView	took 0.61ms
setupListItem	took 0.66ms
obtainView	took 0.23ms
setupListItem	took 1.76ms
obtainView	took 0.43ms
setupListItem	took 0.63ms
obtainView	took 0.19ms
setupListItem	took 1.13ms
obtainView	took 0.18ms
setupListItem	took 1.00ms
obtainView	took 0.12ms
setupListItem	took 0.88ms
Frame	
Description	ListView recycling taking too much time per frame. Ensure your Adapter#getView() binds data efficiently.

systrace frame latency result (1 out of 10 times)

python systrace.py --time=100 -o latencyResult.html

Monitor 100s, then collect latency and generate report

adb shell monkey -p ch.blinkenlights.android.vanilla -v --throttle 200 500

Send 500 random actions to phone,

For more plots please see:

https://github.com/ZiyeHan/CMPS-202/tree/master/latency/results

Conclusion

Throughput

As instruction scale goes up, IPC goes down, Load Miss Ratio of L1 and branch MPKI goes up. As number of threads goes up, branch MPKI goes up but IPC goes down. This is wired and I think maybe that is because I was just monitoring one core?

Latency

The app I chose was quite robust at latency test. Despite I make the MonkeyRunner to send actions at a very high frequency (200ms interval) the latency remains really low as you can see from the plot. I think maybe that is because the developer used more Android officially recommended API calls to draw.

What Did I Learn

- How to use docker
- How to use Perf
- How to use pmu-tools to run throughput test
- How to control program multi-thread
- Some linux commands
- How to use Monkey Runner to run automatic tests on App
- How to use systrace to generate HTML report of latency