SGLang性能分析文档

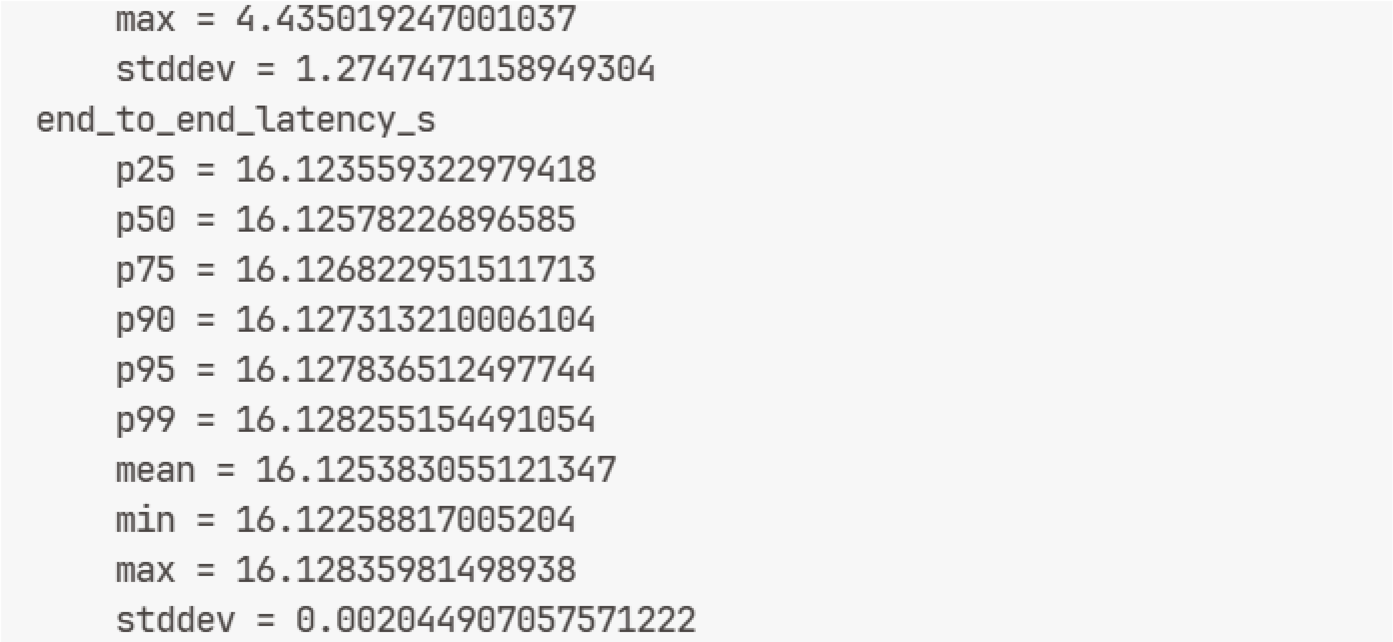
一、问题现象：LLMPerf 测试结果存在明显波动

在使用 **UCM 自带的 llmperf 工具**对 **SGLang 推理服务**进行性能测试时，发现即便在**相同序列、相同并发命中条件**下，多次测试结果仍然存在明显波动。
为便于对比分析，本文选取了性能表现明显不同的两组结果，分别称为 **快组** 与 **慢组**。

- **快组**：整体响应时间较短，prefill 与 prefetch 阶段耗时较低

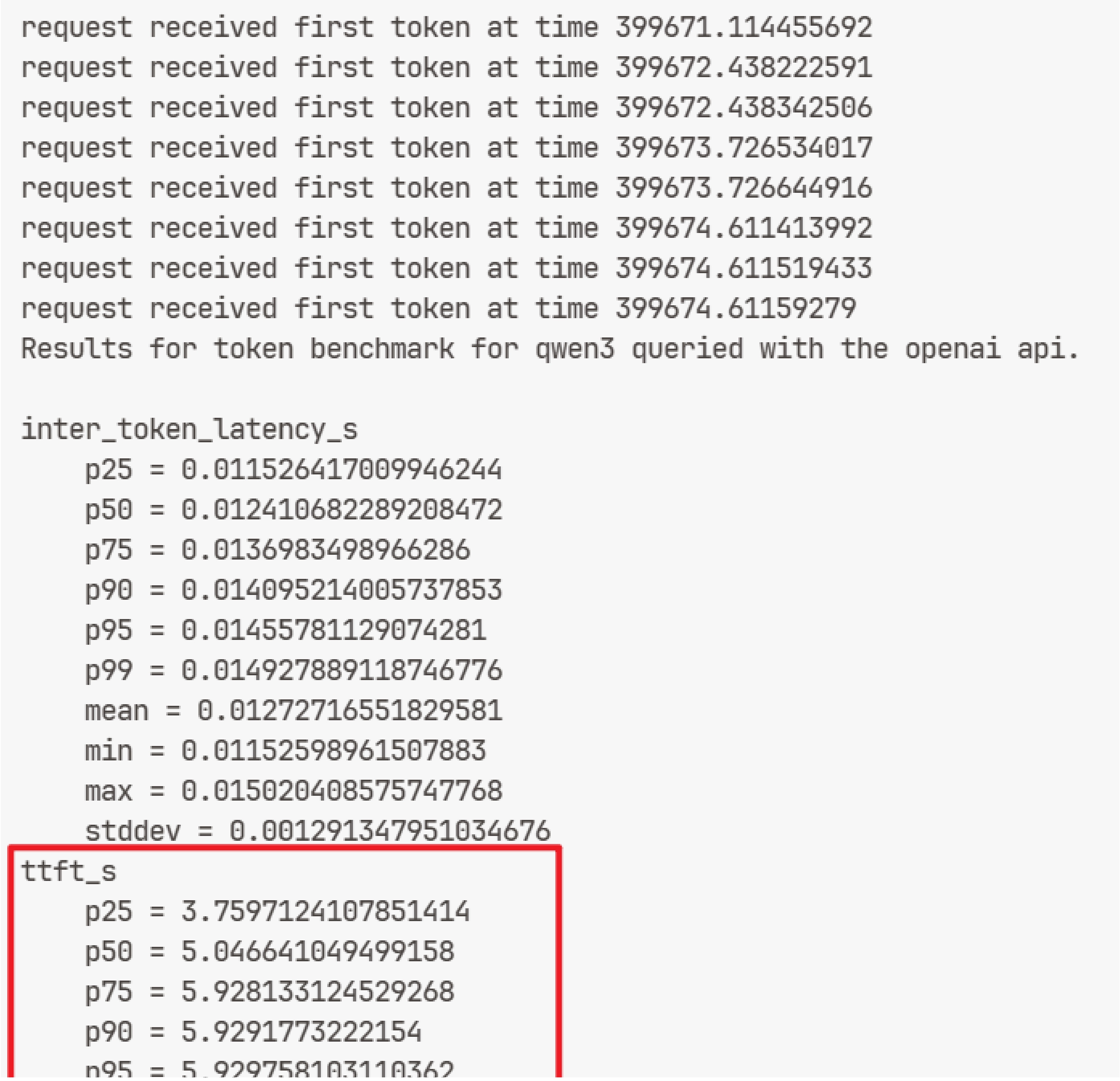
```
request received first token at time 399624.0861841
request received first token at time 399625.394210617
request received first token at time 399625.394389888
request received first token at time 399626.658094349
request received first token at time 399626.658303304
request received first token at time 399627.542406879
request received first token at time 399627.542502741
request received first token at time 399627.542554018
Results for token benchmark for qwen3 queried with the openai api.

inter_token_latency_s
    p25 = 0.011703218928929168
    p50 = 0.012560524107797861
    p75 = 0.013823129636403355
    p90 = 0.014215097148951793
    p95 = 0.014672239405221305
    p99 = 0.015037953210236913
    mean = 0.012870382974701122
    min = 0.011677066035862719
    max = 0.015129381661490816
    stddev = 0.0012705566639244325
ttft_s
    p25 = 2.29076557373628
    p50 = 3.5524465949856676
    p75 = 4.433487335045356
    p90 = 4.43444941972265 9
    p95 = 4.434734333361848
    p99 = 4.434962264273199
    mean = 3.246264018009242
    min = 0.982097816013265 4
```



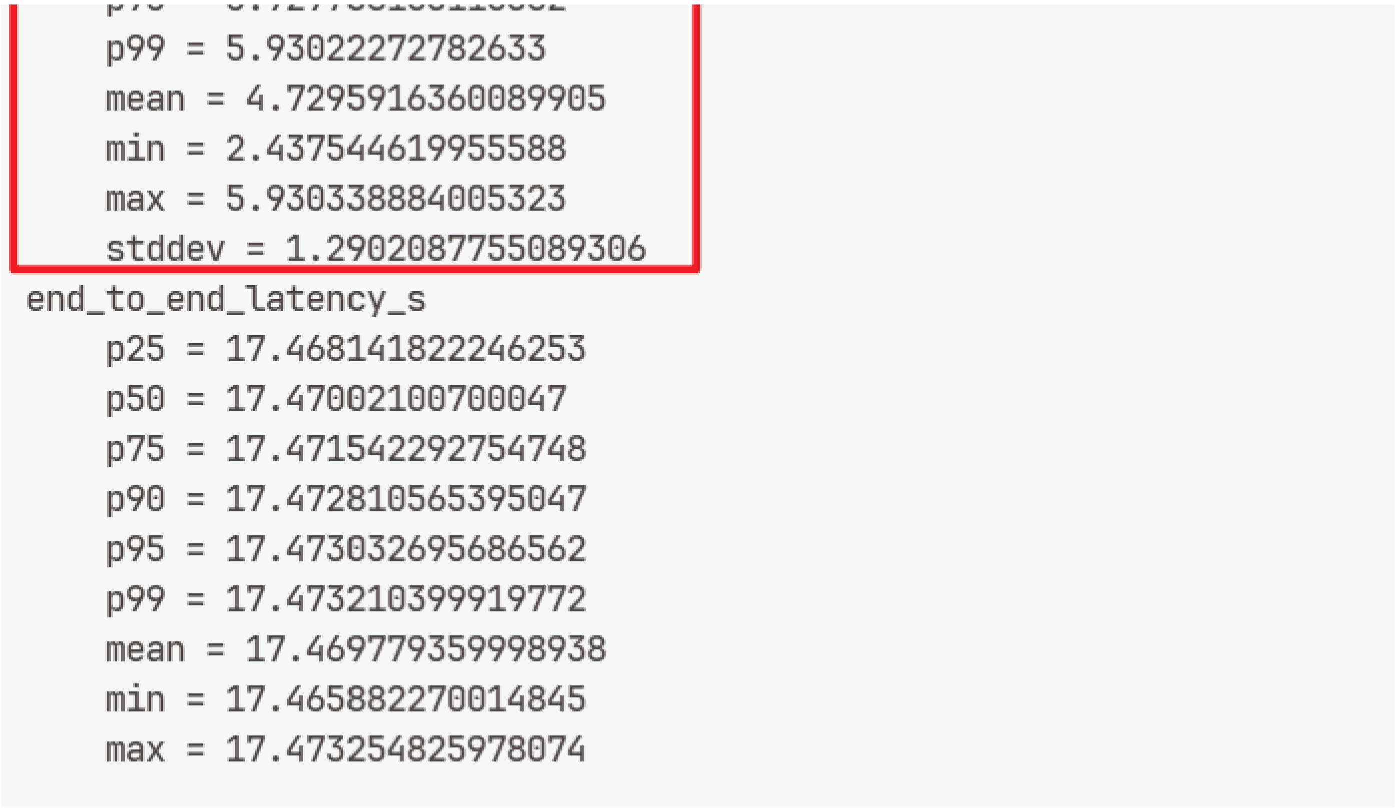
```
    max = 4.435019247001037
    stddev = 1.2747471158949304
end_to_end_latency_s
    p25 = 16.123559322979418
    p50 = 16.12578226896585
    p75 = 16.126822951511713
    p90 = 16.127313210006104
    p95 = 16.127836512497744
    p99 = 16.128255154491054
    mean = 16.125383055121347
    min = 16.12258817005204
    max = 16.12835981498938
    stddev = 0.0020449070575712221
```

- **慢组**：整体响应时间显著增加，存在阶段性耗时拉长现象



```
request received first token at time 399671.114455692
request received first token at time 399672.438222591
request received first token at time 399672.438342506
request received first token at time 399673.726534017
request received first token at time 399673.726644916
request received first token at time 399674.611413992
request received first token at time 399674.611519433
request received first token at time 399674.61159279
Results for token benchmark for qwen3 queried with the openai api.

inter_token_latency_s
    p25 = 0.011526417009946244
    p50 = 0.012410682289208472
    p75 = 0.0136983498966286
    p90 = 0.014095214005737853
    p95 = 0.01455781129074281
    p99 = 0.014927889118746776
    mean = 0.01272716551829581
    min = 0.01152598961507883
    max = 0.015020408575747768
    stddev = 0.001291347951034676
ttft_s
    p25 = 3.7597124107851414
    p50 = 5.046641049499158
    p75 = 5.928133124529268
    p90 = 5.9291773222154
    p95 = 5.929758103110362
```

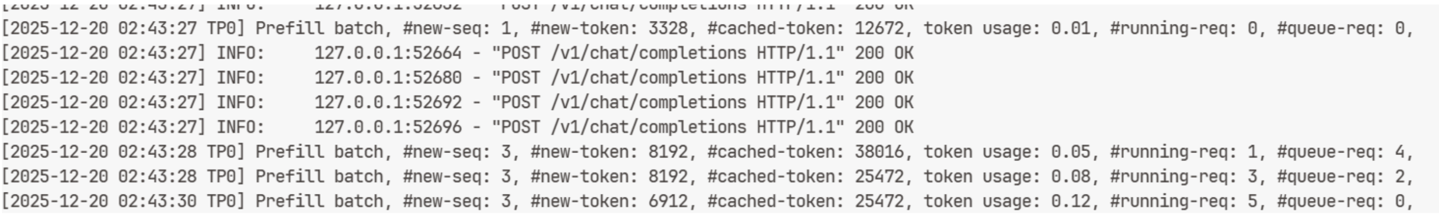


```
    p99 = 5.93022272782633
    mean = 4.7295916360089905
    min = 2.437544619955588
    max = 5.930338884005323
    stddev = 1.2902087755089306
end_to_end_latency_s
    p25 = 17.468141822246253
    p50 = 17.47002100700047
    p75 = 17.471542292754748
    p90 = 17.472810565395047
    p95 = 17.473032695686562
    p99 = 17.473210399919772
    mean = 17.469779359998938
    min = 17.465882270014845
    max = 17.473254825978074
```

二、请求调度层面分析：Prefill 调度行为一致

从推理引擎的请求调度日志来看，**快组与慢组在 Prefill 阶段的调度行为完全一致**，并未观察到调度策略差异。

调度示例（1 / 3 / 3 / 3）



```
[2025-12-20 02:43:27 TP0] Prefill batch, #new-seq: 1, #new-token: 3328, #cached-token: 12672, token usage: 0.01, #running-req: 0, #queue-req: 0,
[2025-12-20 02:43:27] INFO:     127.0.0.1:52664 - "POST /v1/chat/completions HTTP/1.1" 200 OK
[2025-12-20 02:43:27] INFO:     127.0.0.1:52680 - "POST /v1/chat/completions HTTP/1.1" 200 OK
[2025-12-20 02:43:27] INFO:     127.0.0.1:52692 - "POST /v1/chat/completions HTTP/1.1" 200 OK
[2025-12-20 02:43:27] INFO:     127.0.0.1:52696 - "POST /v1/chat/completions HTTP/1.1" 200 OK
[2025-12-20 02:43:28 TP0] Prefill batch, #new-seq: 3, #new-token: 8192, #cached-token: 38016, token usage: 0.05, #running-req: 1, #queue-req: 4,
[2025-12-20 02:43:28 TP0] Prefill batch, #new-seq: 3, #new-token: 8192, #cached-token: 25472, token usage: 0.08, #running-req: 3, #queue-req: 2,
[2025-12-20 02:43:30 TP0] Prefill batch, #new-seq: 3, #new-token: 6912, #cached-token: 25472, token usage: 0.12, #running-req: 5, #queue-req: 0,
```

以常见的一种调度情况为例：

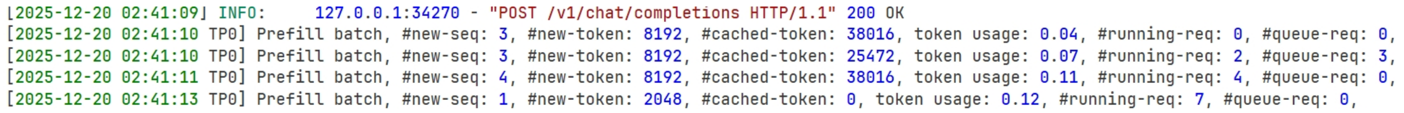
1. **第一轮调度**
    - 单轮 Prefill 最大长度为 **8192**
    - 第一个请求约 **80% token 命中 KV Cache**
    - 剩余部分可直接进入推理阶段，加入 running 队列
2. **第二轮调度**
    - 新进入 **3 个请求**
    - 由于未命中部分累计长度超过 8192

    - 剩余未完成的序列与第三个请求一起进入下一轮 Prefill

3. **第三轮调度**
    - 再次进入 **2 个请求**
    - 累计 Prefill token 数再次超过 8192
    - 部分请求延后至下一轮处理
4. **第四轮调度**
    - 新进入 **2 个请求**
    - 加上上一轮未完成的序列
    - 最终一次性完成 Prefill

该调度过程在快组和慢组中保持一致，**未发现请求进入、合并或拆分逻辑上的差异**。

其他调度情况



```
[2025-12-20 02:41:09] INFO:     127.0.0.1:34270 - "POST /v1/chat/completions HTTP/1.1" 200 OK
[2025-12-20 02:41:10 TP0] Prefill batch, #new-seq: 3, #new-token: 8192, #cached-token: 38016, token usage: 0.04, #running-req: 0, #queue-req: 0,
[2025-12-20 02:41:10 TP0] Prefill batch, #new-seq: 3, #new-token: 8192, #cached-token: 25472, token usage: 0.07, #running-req: 2, #queue-req: 3,
[2025-12-20 02:41:11 TP0] Prefill batch, #new-seq: 4, #new-token: 8192, #cached-token: 38016, token usage: 0.11, #running-req: 4, #queue-req: 0,
[2025-12-20 02:41:13 TP0] Prefill batch, #new-seq: 1, #new-token: 2048, #cached-token: 0, token usage: 0.12, #running-req: 7, #queue-req: 0,
```

此外，也观察到如下变体调度模式：

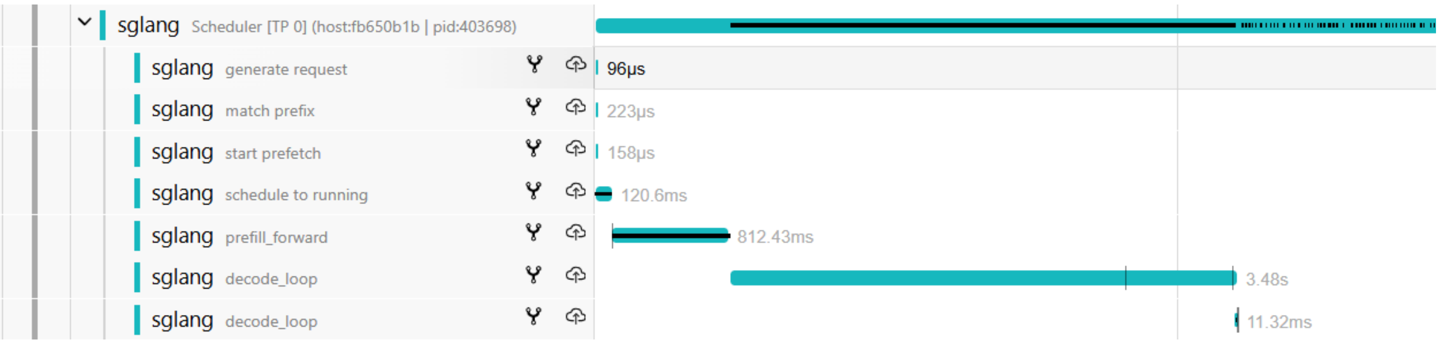
- 第一轮中同时调度 **3 个请求**
- 后续 Prefill 分批行为与上述流程基本一致

总体来看，**调度差异并非导致性能波动的根本原因**。

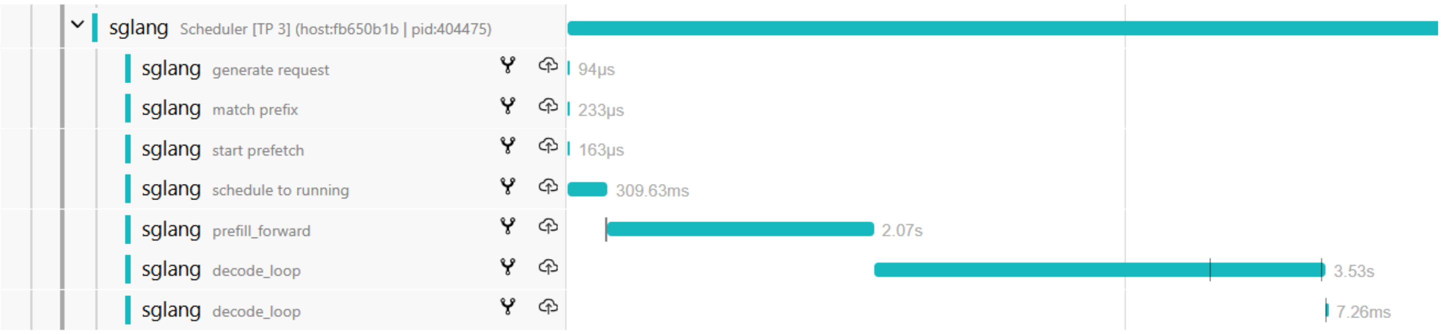
三、Prefetch 阶段分析：耗时存在显著波动

在进一步分析 Trace 数据后发现，**Prefetch 阶段的耗时存在明显差异**，快慢组之间的时间差约 **200ms 左右**。

- **快组**：Prefetch 执行时间较短



- **慢组**：Prefetch 执行时间明显拉长，存在抖动



```
sglang  Scheduler [TP 3] (host:fb650b1b | pid:404475)
sglang  generate request      94µs
sglang  match prefix          233µs
sglang  start prefetch        163µs
sglang  schedule to running   309.63ms
sglang  prefill_forward       2.07s
sglang  decode_loop           3.53s
sglang  decode_loop           7.26ms
```

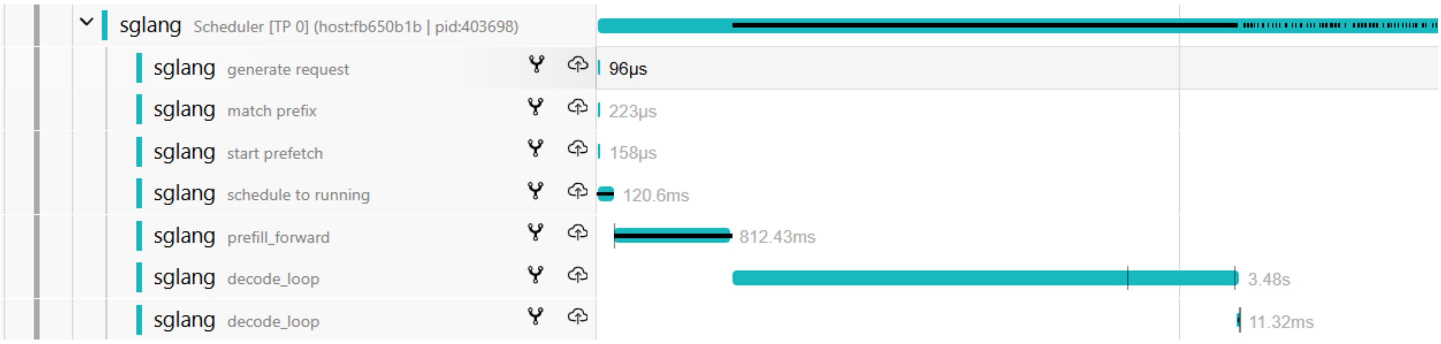
经进一步确认，该波动并非由 SGLang 本身引起，而是由于 **H20 机器的本地盘存在性能问题**。
因此：

> **当前环境下的 Prefetch 性能数据不具备参考价值，应避免将其作为 SGLang 性能评估依据。**

四、Prefill 阶段分析：同样存在耗时差异

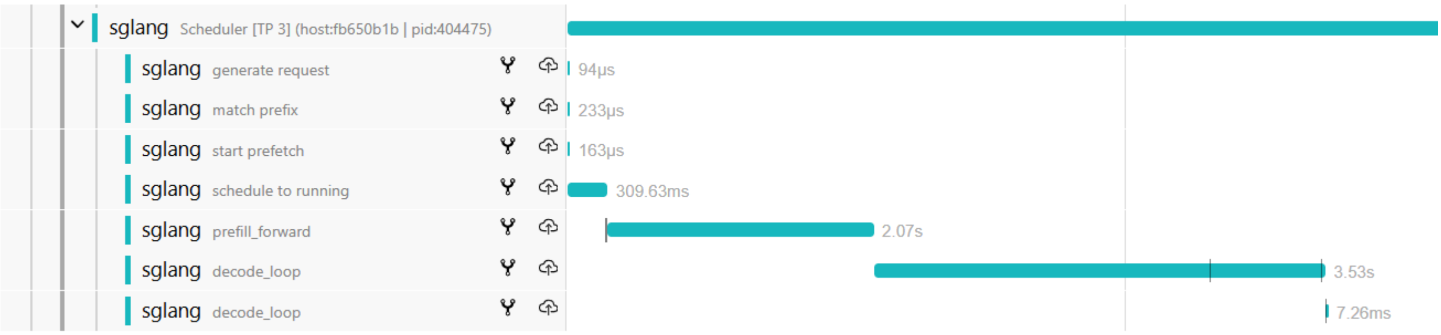
除 Prefetch 外，**Prefill 阶段的耗时在快慢组之间同样存在差距**：

- 快组



```
sglang  Scheduler [TP 0] (host:fb650b1b | pid:403698)
sglang  generate request      96µs
sglang  match prefix          223µs
sglang  start prefetch        158µs
sglang  schedule to running   120.6ms
sglang  prefill_forward       812.43ms
sglang  decode_loop           3.48s
sglang  decode_loop           11.32ms
```

- 慢组中 Prefill 耗时明显偏高



```
sglang  Scheduler [TP 3] (host:fb650b1b | pid:404475)
sglang  generate request      94µs
sglang  match prefix          233µs
sglang  start prefetch        163µs
sglang  schedule to running   309.63ms
sglang  prefill_forward       2.07s
sglang  decode_loop           3.53s
sglang  decode_loop           7.26ms
```

这说明在 **相同调度与相同输入条件下**，Prefill 耗时存在差异。

五、初步结论

1. **请求调度逻辑一致**
   - Prefill 调度轮次、请求合并策略在快慢组中无差异
   - 排除调度层面导致性能波动的可能性

2. **Prefetch 与 Prefill 阶段均存在耗时抖动**
    - Prefetch 波动已确认与 H20 本地盘性能问题相关
    - Prefill 波动可能同样受到硬件或系统状态影响
3. **当前测试环境不适合作为性能基准**
    - 建议在本地盘 IO 状态稳定或更换环境后重新评估性能