PR #35191 完整报告

sgl-project/sglang

[Scheduler] Cap prefill-delayer queue target by admission capacity

合并时间：2026-08-18 21:40

原文链接：http://prhub.com.cn/sgl-project/sglang/pull/35191

执行摘要

- 一句话：prefill 延迟队列目标改用 admission capacity 封顶
- 推荐动作：值得调度器方向的读者精读。重点学习两点：一是用配置容量替代运行时观测高水位做目标封顶的取舍；二是保留 fallback 保证向后兼容的写法。单测构造的 max_prefill_bs=1 场景清晰复现问题，可作为同类调度修复的测试范式。若后续要推广，建议补充真实流量下 TTFT 与 prefill 批大小的基准数据。

功能与动机

PR body 明确指出：prefill-delayer queue trigger caps its accumulation target with max_prefill_bs, an observed high-watermark that can remain near one under trickle arrivals. This allows each request to immediately satisfy the trigger and keeps prefills fragmented. 即真实请求逐个到达时，max_prefill_bs 被压到 1，queue_min = min(running * ratio, 1) 也变成 1，等待队列几乎总是满足条件，延迟器形同虚设，prefill 无法聚合成大 batch。

实现拆解

1. 变更入口：读取 admission capacity

- 文件：python/sglang/srt/managers/prefill_delayer.py
- 在 PrefillDelayer.__init__中新增 self._prefill_max_requests = server_args.prefill_max_requests。该值来自全局 ServerArgs，表示 admission 层的请求数上限，是比运行时观测高水位更稳定的封顶依据。

2. 核心逻辑：queue target 封顶与回退

- 在 _negotiate_should_allow_prefill 的 queue 触发分支中，先计算 queue_capacity：配置了 prefill_max_requests 时使用它，否则回退到 global_max_prefill_bs_max。
- 随后 queue_min_effective = min(int(running_batch * ratio), queue_capacity)。这样即使观测到的 max_prefill_bs 为 1，排队目标也不会被压到 1，延迟器能真正等到队列积累到足够数量。
- slot_condition 分支仍使用 global_max_prefill_bs_max，不受本次改动影响。

3. 配置文档同步

- 文件：python/sglang/srt/server_args.py
- 更新 prefill_delayer_queue_min_ratio 的 help 文本，把 min(running_req * ratio, max_prefill_bs) 改为 min(running_req * ratio, prefill_max_requests)，并注明未配置请求上限时回退观测 max_prefill_bs 的行为。

4. 回归测试

- 文件：test/registered/scheduler/test_prefill_delayer.py
- 为 NegotiateTestCase 增加 prefill_max_requests 字段，并在 _run_negotiate_test 构造 SimpleNamespace 时传入。
- 新增 queue_trigger_uses_admission_capacity 用例：running=500、max_prefill_bs=1、queue_len=1、ratio=0.02、prefill_max_requests=128，断言两次调用后仍应 delay；该用例在旧逻辑下因为 queue_min_effective=1 会直接放行，能精确复现 bug。

5. 验证与 CI

- PR 内报告 pytest -q test/registered/scheduler/test_prefill_delayer.py -k test_negotiate 通过，OSS pre-commit hooks 通过。
- ispobock 触发 /rerun-test 后，github-actions 在 8-gpu-h200 上重新执行该测试文件并通过。

关键文件：

- python/sglang/srt/managers/prefill_delayer.py（模块 调度器；类别 source；类型 core-logic；符号 PrefillDelayer.init, PrefillDelayer._negotiate_should_allow_prefill）：调度器核心逻辑所在：读取 prefill_max_requests 并在 queue trigger 中用它封顶排队目标，同时保留未配置时的观测值回退，是本次修复的主路径。
- test/registered/scheduler/test_prefill_delayer.py（模块 调度测试；类别 test；类型 test-coverage；符号 NegotiateTestCase, _run_negotiate_test, queue_trigger_uses_admission_capacity）：新增 prefill_max_requests 测试字段和 queue_trigger_uses_admission_capacity 回归用例，用 max_prefill_bs=1 的低观测水位场景精确复现 bug，并覆盖配置封顶与 skip_first_delayer 交互。
- python/sglang/srt/server_args.py（模块 启动参数；类别 source；类型 configuration；符号 ServerArgs.prefill_delayer_queue_min_ratio）：同步更新 prefill_delayer_queue_min_ratio 的 help 文档，明确 queue target 以 prefill_max_requests 封顶并回退观测值，避免用户误解配置语义。

关键符号：PrefillDelayer.init, PrefillDelayer._negotiate_should_allow_prefill

关键源码片段

python/sglang/srt/managers/prefill_delayer.py

调度器核心逻辑所在：读取 prefill_max_requests 并在 queue trigger 中用它封顶排队目标，同时保留未配置时的观测值回退，是本次修复的主路径。

```
# PrefillDelayer.__init__ 中读取配置的 admission capacity。
self._prefill_max_requests = server_args.prefill_max_requests
```

```
# _negotiate_should_allow_prefill 的 queue 触发分支：
queue_condition = False
if self._queue_trigger_enabled and global_running_batch_max > 0:
    # 队列目标上限：优先用配置的 prefill_max_requests 封顶，
    # 未配置时回退到观测到的 max_prefill_bs 高水位，保持历史行为不变。
    queue_capacity = (
        self._prefill_max_requests
        if self._prefill_max_requests is not None
        else global_max_prefill_bs_max
    )
    # queue_min 表示期望攒够的排队请求数；max_prefill_bs 在“细水长流”式
    # 到达下会被压到 1，导致触发器被每个请求立刻满足，prefill 持续碎片化。
    queue_min_effective = min(
        int(global_running_batch_max * self._queue_min_ratio),
        queue_capacity,
    )
    queue_condition = (
        queue_min_effective > 0
        and global_waiting_queue_max < queue_min_effective
    )
    # 墙钟超时仍是兜底：即使排队始终达不到目标，也会按时放行避免 TTFT 失控。
    if queue_condition and prev_state is not None:
        elapsed_ms = (time.perf_counter() - prev_state.start_time) * 1000.0
        if elapsed_ms >= self._max_delay_ms:
            queue_condition = False
```

test/registered/scheduler/test_prefill_delayer.py

新增 prefill_max_requests 测试字段和 queue_trigger_uses_admission_capacity 回归用例，用 max_prefill_bs=1 的低观测水位场景精确复现 bug，并覆盖配置封顶与 skip_first_delayer 交互。

```
# 回归场景：running=500、观测 max_prefill_bs=1（trickle 到达导致高水位被压低）、
# queue_len=1、ratio=0.02。未修复时 queue_min_effective=min(10, 1)=1，
# queue_len 不小于 1，触发器不延迟；修复后按 prefill_max_requests=128 封顶，
# queue_min_effective=min(10, 128)=10，queue_len=1 < 10，应延迟。
NegotiateTestCase(
    name='queue_trigger_uses_admission_capacity',
    max_delay_passes=100,
    token_usage_low_watermark=0.8,
    queue_min_ratio=0.02,
    max_delay_ms=5000,
    prefill_max_requests=128,
    calls=[
        NegotiateCall(
            prefillable=[True, True, True, True],
            token_usage=[0.9, 0.9, 0.9, 0.9],
            running_batch=[500, 500, 500, 500],
            max_prefill_bs=[1, 1, 1, 1],
            waiting_queue_len=[1, 1, 1, 1],
```

```
            max_running_requests=1024,
        ),
        # skip_first_delayer 会吞掉第一次延迟，第二次相同调用必须真正延迟。
        NegotiateCall(
            prefillable=[True, True, True, True],
            token_usage=[0.9, 0.9, 0.9, 0.9],
            running_batch=[500, 500, 500, 500],
            max_prefill_bs=[1, 1, 1, 1],
            waiting_queue_len=[1, 1, 1, 1],
            max_running_requests=1024,
        ),
    ],
    expected_allow=False,
    expected_reason='delay',
)
```

评论区精华

该 PR 没有实质性的 review 代码讨论（review_comments_count=0），ispobock 直接 APPROVED。唯一的交互是 ispobock 发起 /rerun-test test/registered/scheduler/test_prefill_delayer.py，github-actions 在 8-gpu-h200 环境执行通过。值得记录的设计权衡由 PR body 和代码注释给出：未配置 prefill_max_requests 时回退到 max_prefill_bs，保证老用户行为不变；同时 max_delay_ms 仍是墙钟兜底，避免队列目标放大后最坏情况 TTFT 失控。

- /rerun-test 触发 prefill-delayer 专项测试 (testing): 单测在 8-GPU H200 环境通过，无额外 review 质疑。

风险与影响

- 风险：
  1. 调度器热路径变更：改动集中在 _negotiate_should_allow_prefill，每个 scheduler 迭代都会进入该分支，新增一次属性读取和 if 判断，开销可忽略。
  2. 行为兼容性：仅当 prefill_delayer_queue_min_ratio 配置且 prefill_max_requests 有值时才改变行为；未配置时 queue_capacity 取 global_max_prefill_bs_max，与旧逻辑完全一致。
  3. 配置不当风险：prefill_max_requests 设置过大会让 queue_min_effective 偏大，排队达不到阈值时依赖 max_delay_ms 放行，可能推高局部 TTFT；设置过小则聚合效果减弱，需要按实际流量调参。
  4. DP 一致性：queue_capacity 使用全局配置而非 per-rank 观测值，所有 rank 从同一 server_args 读取，不会出现 rank 间不一致；但测试覆盖偏向数值行为，未覆盖真实生产负载下的 batch 分布。
  5. 缺少性能基准：PR 没有给出真实流量下的 TTFT 或 prefill batch 大小基准，收益主要靠推理和用例论证。 - 影响：影响范围限于启用 DP attention 且开启 prefill-delayer queue 触发器的部署。收益是 trickle 到达时 prefill 能聚合成更大 batch，降低碎片化和调度开销，潜在改善 TTFT；对模型计算无影响，PR 明确 Accuracy Tests not

applicable。团队侧需要关注 server_args 文档更新，避免用户对 queue target 语义产生误解；测试基建新增的 prefill_max_requests 字段是后续调度测试的复用入口。 - 风险标记：调度器核心路径变更，缺少真实流量性能基准，配置不当可能影响 TTFT，DP 多卡行为依赖全局参数一致性

关联脉络

- 暂无明显关联 PR