PR #29406 完整报告

sgl-project/sglang

Stop reading cur_batch in is_fully_idle and abort_request

合并时间：2026-07-10 08:54

原文链接：http://prhub.com.cn/sgl-project/sglang/pull/29406

执行摘要

- 一句话：消除调度器中 cur_batch 的非必要读取
- 推荐动作：建议快速合入。本 PR 变更量小（4 行新增，5 行删除），每一步都附带了等价性证明和有意的行为变更说明。审核机器人提出的正确性担忧已被 PR 设计解决。对于关注调度器架构和状态管理的工程师，本 PR 展示了一个很好的增量式重构案例——通过逐步建立不变式来安全地清理字段依赖。

功能与动机

该 PR 是调度器状态字段局部化重构系列的一部分。其根本动机是消除对 self.cur_batch 的外部读取，使得 cur_batch 可以成为仅调度器内部使用的局部变量，从而减少不依赖传递和错误访问。PR body 明确指出："This PR removes every load-bearing read of self.cur_batch outside the PP microbatch slot arrays"，并且 "lets a following PR localize the field cleanly"。

实现拆解

本 PR 通过三步完成对 self.cur_batch 读取的移除：

1. 语法拒绝请求直接标记完成（行为有意变更）：在 scheduler_components/batch_result_processor.py 中，_apply_prefill_grammar 和 _accept_grammar_tokens 两个方法在语法拒绝时不调用 self.abort_request(AbortReq(rid=req.rid))，而是直接设置 req.to_finish = FINISH_ABORT()。这是唯一在迭代中期（process_batch_result 内部）调用 abort_request 的地方，移除它建立了后续步骤所需的不变式。同时修复了旧代码中因 rid 前缀匹配导致意外终止其他请求的 bug。
2. abort_request 改用 last_batch：在 scheduler.py 的 abort_request 方法中，非 pipeline-parallel 模式下扫描运行中请求的列表从 [self.running_batch, self.cur_batch] 改为 [self.running_batch, self.last_batch]。根据第一步建立的不变式（abort_request 仅在事件循环顶部被调用），此时 cur_batch 和 last_batch 等价（或均为 None），因此行为等价。
3. is_fully_idle 移除冗余的 cur_batch 项：从 is_fully_idle 的 idle 条件中删除了 (self.cur_batch is None or self.cur_batch.is_empty())。该条件在 PP 模式下是 _pp_microbatches_drained() 的子集，在非 PP 模式下则与已存在的 last_batch 条件重复。

测试配套：本次变更未包含专门的测试文件修改。

关键文件：

- python/sglang/srt/managers/scheduler_components/batch_result_processor.py（模块 调度器；类别 source；类型 core-logic；符号 _apply_prefill_grammar, _accept_grammar_tokens）：核心变更文件：将语法拒绝时的 abort_request 调用替换为直接设置 req.to_finish，这是建立 invariant 的关键。同时修正了因 rid 前缀匹配导致误中止其他请求的 bug。
- python/sglang/srt/managers/scheduler.py（模块 调度器；类别 source；类型 core-logic；符号 is_fully_idle, abort_request）：次要变更文件：在 is_fully_idle 中删除 cur_batch 检查，在 abort_request 中将 cur_batch 替换为 last_batch。

关键符号：_apply_prefill_grammar, _accept_grammar_tokens, is_fully_idle, abort_request

关键源码片段

python/sglang/srt/managers/scheduler_components/batch_result_processor.py

核心变更文件：将语法拒绝时的 abort_request 调用替换为直接设置 req.to_finish，这是建立 invariant 的关键。同时修正了因 rid 前缀匹配导致误中止其他请求的 bug。

```
# File: python/sglang/srt/managers/scheduler_components/batch_result_processor.py

from sglang.srt.disaggregation.utils import DisaggregationMode
from sglang.srt.environ import envs
from sglang.srt.layers.logits_processor import LogitsProcessorOutput
# 移除了对 AbortReq 的导入
# from sglang.srt.managers.io_struct import AbortReq
from sglang.srt.managers.schedule_batch import (
    FINISH_ABORT, # 新增导入：用于直接标记请求终止
    Req,
    ScheduleBatch,
)
# ... 省略无关代码 ...

class BatchResultProcessor:
    # ...

    def _apply_prefill_grammar(self, *, req: Req, next_token_id: int) -> None:
        # FIXME: this try-except block is for handling unexpected xgrammar issue.
        try:
            req.grammar.accept_token(next_token_id)
        except ValueError as e:
            # Grammar accept_token can raise ValueError if the token is not in the grammar.
            # This can happen if the grammar is not set correctly or the token is invalid.
            logger.error(
                f"Grammar accept_token failed for req {req.rid} with token {next_token_id}: {e}"
            )
```

```
            # 旧代码：self.abort_request(AbortReq(rid=req.rid))
            # 新代码：直接标记请求为终止，避免通过 rid 扫描所有队列 / 批次
            req.to_finish = FINISH_ABORT()
        req.grammar.finished = req.finished()

    def _accept_grammar_tokens(
        self, req: Req, tokens: Union[int, List[int]]
    ) -> List[int]:
        """Advance the grammar over the accepted token(s), stopping at the token
        that terminates it.
        ...
        """
        if isinstance(tokens, int):
            tokens = [tokens]
        retained = []
        try:
            for token_id in tokens:
                req.grammar.accept_token(token_id)
                retained.append(token_id)
                if req.grammar.is_terminated():
                    break
        except ValueError as e:
            logger.error(
                f"Grammar accept_token failed for req {req.rid} with token "
                f"{tokens}: {e}"
            )
            # 同样的改动：直接标记终止，而非调用 abort_request
            req.to_finish = FINISH_ABORT()
        return retained
```

python/sglang/srt/managers/scheduler.py

次要变更文件：在 is_fully_idle 中删除 cur_batch 检查，在 abort_request 中将 cur_batch 替换为 last_batch。

```
# File: python/sglang/srt/managers/scheduler.py

class Scheduler:
    # ...

    def is_fully_idle(self, for_health_check=False) -> bool:
        # ...
        # Batch running status
        idle = (
            self.running_batch.is_empty()
            and self.chunked_req is None
            and not self.dllm_manager.any_staging_reqs()
            and (self.last_batch is None or self.last_batch.is_empty())
            # 旧代码：and (self.cur_batch is None or self.cur_batch.is_empty())
            # 新代码：移除了 cur_batch 项，因为 last_batch 已覆盖相同语义
```

```
            and (not self.enable_overlap or len(self.result_queue) == 0)
            and self._pp_microbatches_drained()
        )
        # ...

    def abort_request(self, recv_req: AbortReq):
        # ...
        # Delete requests in the running batch
        if self.ps.pp_size == 1:
            # 旧代码：inflight_batches = [self.running_batch, self.cur_batch]
            # 新代码：使用 last_batch 替代 cur_batch
            # 注：此时 abort_request 仅在事件循环顶部被调用，cur_batch == last_batch
            inflight_batches = [self.running_batch, self.last_batch]
        else:
            inflight_batches = [*self.running_mbs, *self.mbs]
        # ...
```

评论区精华

审核机器人 gemini-code-assist[bot] 在 abort_request 的改动上提出了严重正确性担忧，认为将 self.cur_batch 替换为 self.last_batch 会引入 bug。其论据是：当 abort_request 在 process_batch_result 期间同步调用时（如语法拒绝场景），请求只存在于 cur_batch 中，而此时 last_batch 尚未更新，因此扫描 last_batch 会找不到该请求。

然而该评论忽略了 PR 中第一步变更：语法拒绝场景已经不再调用 abort_request，而是直接设置 req.to_finish。因此 abort_request 不会再在 process_batch_result 内部被调用，建立了 PR body 中所述的不变式。该评论虽然正确指出了理论风险，但实际已被 PR 的第一步解决。作者在 PR body 中已明确阐述了这一步的意图和等价性证明。

- 将 cur_batch 替换为 last_batch 可能导致正确性 bug (correctness): 该担忧已被 PR 的第一步变更解决：语法拒绝场景不再调用 abort_request，而是直接设置 req.to_finish = FINISH_ABORT()。因此 abort_request 永远不会在 process_batch_result 内部被调用，cur_batch 和 last_batch 在 abort_request 调用时总是等价的。

风险与影响

- 风险：本 PR 存在以下技术风险：
  1. 正确性风险（低）：如果在未来的重构中意外引入新的 abort_request 在迭代中期调用的场景，而 cur_batch 已被移除，则可能无法正确中止请求。此风险已被 PR body 中明确的不变式声明和 Codex 审计结果缓解。
  2. 回归风险（中）：is_fully_idle 的变更虽然理论上行为等价，但 cur_batch 和 last_batch 在不同步时钟点（如 process_batch_result 刚执行完但尚未更新 last_batch 时）可能有短暂差异。然而 idle 检查通常只在调度循环开始时调用，实际影响很小。
  3. 测试覆盖缺失：PR 没有新增测试来验证新行为（尤其是第一步的有意变更——不再误杀前缀匹配的请求），回归风险更多依赖于代码审查而非自动化测试。 - 影响：影响范围：仅影响调度器（Scheduler）的两个方法：is_fully_idle 和 abort_request。对外部 API、

推理性能、用户可见行为无直接影响。影响程度：低。变更后的行为在大多数场景下与之前完全一致，仅在语法拒绝场景下行为有细微差异（不再误终止前缀匹配的其他请求），这实际上是一个 bugfix。团队影响：为后续将 cur_batch 局部化的重构扫清了障碍，有利于代码可维护性和状态清晰度。

- 风险标记：缺乏测试覆盖，核心路径变更

关联脉络

- PR #29407 Localize cur_batch field in Scheduler to avoid field-based state access: 这是本 PR 的后续工作，利用本 PR 建立的不变式将 cur_batch 字段局部化，仅看门狗使用。
- PR #29408 Avoid implicit field-based side channel in Scheduler planning: 与本 PR 同属调度器状态重构系列，消除隐式副作用通道，进一步减少不必要的字段依赖。
- PR #30573 Configurable decode retraction order: 修改了 schedule_batch.py，与本 PR 的调度器重构同属调度相关变更。