PR #25636 完整报告

sgl-project/sglang

Carve out SchedulerBatchResultProcessor for batch-result state

合并时间：2026-05-18 18:44

原文链接：http://prhub.com.cn/sgl-project/sglang/pull/25636

执行摘要

- 一句话：抽离 SchedulerBatchResultProcessor 准备批量结果状态管理
- 推荐动作：对于希望理解 SGLang 调度器架构演进的技术成员，建议精读此 PR，特别是新增的 SchedulerBatchResultProcessor 数据类设计以及静态方法的转换模式。日常使用者只需知晓其是代码清理工作即可。

功能与动机

PR 描述明确指出这是 'Inplace prep for the introduce-batch-result-processor mech move (the last extract from SchedulerOutputProcessorMixin)'。目的是将批量结果处理的状态和方法从庞大的调度器类中分离到独立组件，以改善可维护性和可测试性。

实现拆解

1. 创建数据类：在 scheduler_components/batch_result_processor.py 中新增 SchedulerBatchResultProcessor 数据类，其字段涵盖调度器运行所需的所有协作者（如 server_args、token_to_kv_pool_allocator）以及回调函数（如 abort_request、increment_generated_tokens）。
2. 方法改造：在 scheduler_output_processor_mixin.py 中将原本以 Scheduler 为第一参数的方法改为 @staticmethod，第一个参数类型变为 SchedulerBatchResultProcessor。例如 process_batch_result_prebuilt(self: Scheduler, ...) 变为 @staticmethod process_batch_result_prebuilt(self: "SchedulerBatchResultProcessor", ...)。同时将 maybe_collect_* 方法重命名为带前导下划线的内部方法（如 _maybe_collect_routed_experts）。
3. 实例化组件：在 scheduler.py 的 __init__ 中新增 self.batch_result_processor = SchedulerBatchResultProcessor(...) 实例化，传入对应参数。
4. 调用点更新：将所有调用 process_batch_result_* 的地方改为通过 self.batch_result_processor 传递，例如 self.process_batch_result_prebuilt(self.batch_result_processor, batch)。涉及文件包括 scheduler.py、scheduler_pp_mixin.py（未在变更文件中列出但属内部组件）、disaggregation/decode.py、disaggregation/prefill.py。
5. 调整数据流：在 disaggregation/prefill.py 中将直接引用 self.logprob_result_processor 改为通过 self.batch_result_processor.logprob_result_processor 访问，确保方法内部访问到正确的处理器实例。

6. 配套决策：PR 中存在 pragmmatic deviation，将本应留在后续 PR 的回调重构（如 increment_generated_tokens 等 Callable 替换）提前放入此 PR，以保证构建链连续性和后续移动时函数体的字节级一致性。

关键文件：

- python/sglang/srt/managers/scheduler_output_processor_mixin.py（模块 输出处理；类别 source；类型 core-logic；符号 process_batch_result_prebuilt, _maybe_collect_routed_experts, _maybe_collect_indexer_topk, _maybe_collect_customized_info）：核心修改文件：将批量结果处理方法转为静态方法，为后续迁移到独立组件做准备。
- python/sglang/srt/managers/scheduler_components/batch_result_processor.py（模块 批量处理器；类别 source；类型 core-logic；符号 SchedulerBatchResultProcessor）：新增文件：定义了承载批量结果处理状态的 SchedulerBatchResultProcessor 数据类。
- python/sglang/srt/managers/scheduler.py（模块 调度器；类别 source；类型 dependency-wiring）：主要调用方：导入并实例化 SchedulerBatchResultProcessor，将所有 process_batch_result_* 调用的接受者重定向到 batch_result_processor。
- python/sglang/srt/disaggregation/decode.py（模块 解码组件；类别 source；类型 core-logic）：解码分离组件中 process_batch_result_prebuilt 调用点更新。
- python/sglang/srt/disaggregation/prefill.py（模块 预填充组件；类别 source；类型 core-logic）：预填充分离组件中 logprob_result_processor 调用路径更新。

关键符号：SchedulerBatchResultProcessor, process_batch_result_prebuilt, _maybe_collect_routed_experts, _maybe_collect_indexer_topk, _maybe_collect_customized_info, process_batch_result_prefill, process_batch_result_decode, process_batch_result_idle

关键源码片段

python/sglang/srt/managers/scheduler_output_processor_mixin.py

核心修改文件：将批量结果处理方法转为静态方法，为后续迁移到独立组件做准备。

```
@staticmethod
def process_batch_result_prebuilt(
    self: "SchedulerBatchResultProcessor", batch: ScheduleBatch
):
    # 注意：self 不再是 Scheduler 实例，而是 SchedulerBatchResultProcessor 实例
    # 但通过数据类组合，仍然可以访问所有需要的属性
    assert self.disaggregation_mode == DisaggregationMode.DECODE
    use_free_group = self.server_args.disaggregation_decode_enable_radix_cache
    if use_free_group:
        self.token_to_kv_pool_allocator.free_group_begin()
    for req in batch.reqs:
        req.time_stats.set_decode_prebuilt_finish_time()
        req.check_finished()
        if req.finished():
            req.time_stats.set_quick_finish_time()
```

```
            # 原来通过 self.enable_hisparse 访问，现在通过 self.server_args.enable_hisparse
            if self.server_args.enable_hisparse:
                self.hisparse_coordinator.request_finished(req)
            release_kv_cache(req, self.tree_cache)

        # Note: Logprobs should be handled on the prefill engine.
        self.output_streamer.stream_output(batch.reqs, batch.return_logprob)
        if use_free_group:
            self.token_to_kv_pool_allocator.free_group_end()
```

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

新增文件：定义了承载批量结果处理状态的 SchedulerBatchResultProcessor 数据类。

```
@dataclass(kw_only=True, slots=True, frozen=True)
class SchedulerBatchResultProcessor:
    # 从 Scheduler 中提取的所有协作者，均通过 __init__ 注入
    is_generation: bool
    disaggregation_mode: "DisaggregationMode"
    enable_overlap: bool
    enable_overlap_mlx: bool
    server_args: "ServerArgs"
    model_config: "ModelConfig"
    token_to_kv_pool_allocator: "BaseTokenToKVPoolAllocator"
    tree_cache: "BasePrefixCache"
    hisparse_coordinator: Optional["HiSparseCoordinator"]
    req_to_token_pool: "ReqToTokenPool"
    decode_offload_manager: Optional["DecodeKVCacheOffloadManager"]
    metrics_collector: "SchedulerMetricsCollector"
    metrics_reporter: "SchedulerMetricsReporter"
    draft_worker: "BaseTpWorker"
    model_worker: "BaseTpWorker"
    logprob_result_processor: "SchedulerLogprobResultProcessor"
    output_streamer: "SchedulerOutputStreamer"
    # 回调函数，用于向 Scheduler 报告信息
    abort_request: Callable
    report_prefill_stats: Callable
    report_decode_stats: Callable
    update_spec_metrics: Callable
    increment_generated_tokens: Callable
    advance_forward_ct_decode: Callable
```

评论区精华

此 PR 无 review 评论，但 PR body 中提到了一个设计权衡（PRAGMATIC DEVIATION）：为了保持构建链的连续性，将在后续 PR 中进行的某些回调重构捆绑至当前 PR。作者明确说明这些更改本应放在后续非机械变更的提交中，但合在一起保证了构建可通过且后续移动时函数体字节一致。

- 暂无高价值评论线程

风险与影响

- 风险：主要风险：如果 SchedulerBatchResultProcessor 构造器的入参与实际使用方法不匹配，或某些方法在子类中被重写，则可能导致运行时错误。但由于代码是机械式重构（将 self 类型从 Scheduler 改为 SchedulerBatchResultProcessor，且未改变业务逻辑），风险较低。另外，PR 缺少 CI 运行（缺少 run-ci 标签），但作者已完成合并，表明经过本地验证或后续 CI。无新增测试覆盖，存在回归隐患，但由于是重构且路径清晰，风险可控。
- 影响：对用户：无直接影响。对系统：不会改变运行逻辑，但为未来更清晰的组件化重构奠定了基础。对团队：调度器代码结构得到改善，后续可删除 SchedulerOutputProcessorMixin，减少大型类的复杂度。对开发：后续在批量结果处理逻辑上的维护可直接在 SchedulerBatchResultProcessor 内进行，无需影响 scheduler.py。
- 风险标记：核心路径变更，无测试覆盖，缺少 CI 验证

关联脉络

- PR #25637 Move batch-result processing to SchedulerBatchResultProcessor and retire output_processor mixin: 此 PR 的直系后继，将当前 PR 中转换为静态方法的方法物理移动到 SchedulerBatchResultProcessor 中，并删除 SchedulerOutputProcessorMixin。
- PR #25638 Move module-level helpers out of scheduler.py: 同一重构链中的另一项：将 scheduler.py 中的模块级辅助函数移到独立文件，进一步精简调度器。
- PR #25639 Delete the now-unused is_work_request from scheduler.py: 清理在前序重构中不再使用的函数，与当前 PR 同属调度器重构系列。