PR #33587 完整报告

sgl-project/sglang

[Scheduler] Align WAR fences with CUDA graph metadata reads

合并时间：2026-08-06 04:52

原文链接：http://prhub.com.cn/sgl-project/sglang/pull/33587

执行摘要

- 一句话：对齐 WAR 栅栏与图元数据读取，覆盖 prefill 与 spec decode
- 推荐动作：值得精读。该 PR 展示了如何用『图内事件节点 + 策略枚举 + 算法能力接口』把 CUDA graph 的同步边界精确对齐到读取发生点，是理解 SGLang overlap scheduler 与 WAR 屏障机制的关键案例。可重点关注 _war_read_done_policy 的优先级判断和 make_war_read_done_event 的外部事件用法。

功能与动机

overlap scheduler 会在前一 forward 仍在另一 CUDA stream 上运行时准备下一批请求；在重写共享 request/attention 元数据之前，WAR 屏障必须等待 forward 对这些状态的最后一次读取。原有 decode 路径只能在整张 CUDA graph replay 之前或之后发布 read-done，无法表达图内部的读取边界；prefill 完全没有细粒度发布，只能退化为整个forward 的粗粒度屏障。TRTLLM MHA 的 SWA 缓存写入也会在提前解除栅栏后重新读取实时的 full-to-SWA 映射，而不是使用为 forward 准备的元数据快照。

实现拆解

1. 在 runner_utils/war_event.py 中定义 WarReadDonePolicy 枚举（NONE / PRE_REPLAY / IN_GRAPH / POST_REPLAY），并实现 make_war_read_done_event 与 maybe_publish_prefill_war_read_done；前者创建 CUDA 外部事件供图捕获使用，后者集中承载 prefill 发布的全部门控（forward 模式、非 spec 算法、backend compliance、环境变量）。
2. 在 decode_cuda_graph_runner.py 中新增三个方法：_plant_war_read_done_node 在捕获期把事件记录成图节点并置位 _war_read_done_node_planted；_war_read_done_policy 按 forward 模式与 backend 特征计算发布策略；_publish_war_read_done 区分图内事件复用与即时记录两种情况。execute 中的旧发布逻辑被策略调用替换。
3. 在 spec_info.py 与 spec_registry.py 中新增 supports_target_verify_war_read_done capability 接口，默认仅 DFlash 家族返回 True，从而让 target verify 可以继承 decode 的细粒度栅栏，其他算法保持回退。
4. prefill_cuda_graph_runner.py 在 replay 准备（含 chunked-prefix gather）结束后调用 maybe_publish_prefill_war_read_done；该路径默认关闭，需设置 SGLANG_ENABLE_PREFILL_WAR_READ_DONE=1 启用。

5. trtllm_mha_backend.py 将 SWA 层的 out_cache_loc 翻译结果在 metadata 初始化时一次性写入 forward_metadata.swa_out_cache_loc，_get_layer_cache_loc 只消费该快照，并新增 assert 防止缺失或长度不足。
6. 测试配套：新增 test_decode_cuda_graph_war_fence.py（策略选择与发布时序）、test_prefill_war_read_done.py（开关与门控），并为 test_trtllm_mha_graph_metadata.py 增加竞态与快照用例；environ.py 注册新环境变量。

关键文件：

- python/sglang/srt/model_executor/runner_utils/war_event.py（模块 WAR 栅栏；类别 source；类型 data-contract；符号 WarReadDonePolicy, make_war_read_done_event, maybe_publish_prefill_war_read_done）：新增 WAR 栅栏核心契约文件：定义 WarReadDonePolicy 枚举、外部事件创建与 prefill 门控发布逻辑，是全 PR 的语义中心。
- python/sglang/srt/model_executor/runner/decode_cuda_graph_runner.py（模块 解码执行器；类别 source；类型 data-contract；符号 _plant_war_read_done_node, _war_read_done_policy, _publish_war_read_done）：WAR 栅栏在 decode 路径上的主要改造点：捕获期植入事件节点，replay 期按策略发布，替换原有粗粒度发布逻辑。
- python/sglang/srt/layers/attention/trtllm_mha_backend.py（模块 注意力后端；类别 source；类型 core-logic；符号 _get_layer_cache_loc）：修复 SWA 缓存写入在 WAR 栅栏提前后读取实时映射的竞态，并声明 prefill metadata-init compliance，是正确性关键改动。
- python/sglang/srt/model_executor/runner/prefill_cuda_graph_runner.py（模块 预填执行器；类别 source；类型 data-contract；符号 maybe_publish_prefill_war_read_done）：prefill 路径的 WAR 发布接入点：在 replay 准备（含 chunked-prefix gather）结束后调用工具函数，实现本 PR 的核心性能目标。
- python/sglang/srt/model_executor/model_runner.py（模块 模型执行器；类别 source；类型 data-contract；符号 make_war_read_done_event）：创建持久化的 war_read_done_event 外部事件并挂到 model_runner，供 decode capture 与发布共用。
- python/sglang/srt/speculative/spec_info.py（模块 推测解码；类别 source；类型 core-logic；符号 supports_target_verify_war_read_done）：新增 target verify 是否支持 WAR read-done 的 capability 接口，默认仅 DFlash 家族放行，是 spec 路径安全接入的关键。
- test/registered/unit/model_executor/runner/test_decode_cuda_graph_war_fence.py（模块 单元测试；类别 test；类型 test-coverage；符号 test_war_read_done_policy, test_publish_war_read_done, test_execute_publishes_the_planted_graph_event, test_execute_records_pre_replay_for_snapshot_backends）：对 decode runner 的 WAR 策略与发布时序做了完整的纯 CPU 单元覆盖，是保证重构正确性的主要测试。

关键符号：WarReadDonePolicy, make_war_read_done_event, maybe_publish_prefill_war_read_done, _plant_war_read_done_node, _war_read_done_policy, _publish_war_read_done, supports_target_verify_war_read_done, _get_layer_cache_loc

关键源码片段

python/sglang/srt/layers/attention/trtllm_mha_backend.py

修复 SWA 缓存写入在 WAR 栅栏提前后读取实时映射的竞态，并声明 prefill metadata-init compliance，是正确性关键改动。

```
# TRTLLM MHA：SWA 层的 cache 写入位置在 metadata 初始化时一次性快照，
# 避免每次 layer 实时读取 full-to-SWA 映射与调度器在栅栏释放后的写入竞争。

class TRTLLMHAAttnBackend(FlashInferAttnBackend):

    # prefill 的 metadata 初始化会快照全部调度器共享输入，可安全提前解除栅栏。
    prefill_shared_reads_end_at_metadata_init: bool = True

    def _get_layer_cache_loc(
        self, layer: RadixAttention, forward_batch: ForwardBatch
    ) -> torch.Tensor:
        """返回当前层应有的 cache 写入位置（full 或 SWA 索引空间）。"""
        if self._swa_kv_pool is not None:
            _, is_swa = self._swa_kv_pool.layers_mapping[layer.layer_id]
            if is_swa:
                swa_loc = self.forward_metadata.swa_out_cache_loc
                assert (
                    swa_loc is not None
                    and swa_loc.shape[0] >= forward_batch.out_cache_loc.shape[0]
                ), (
                    'SWA write locs missing or too short: init_forward_metadata '
                    'must translate out_cache_loc once; a per-layer gather of '
                    'the live mapping would race the scheduler after the WAR '
                    'fence releases'
                )
                # piecewise prefill 会按 attention 调用收窄 out_cache_loc，
                # 快照保留的是 padding 后的整批长度，直接截取即可。
                return swa_loc[: forward_batch.out_cache_loc.shape[0]]
        return forward_batch.out_cache_loc
```

评论区精华

本 PR 没有实质性的 review 评论，merrymercy 直接批准。从提交历史可看到两项主动收敛：一是提交 fix(prefill): keep WAR read-done non-speculative 将 prefill fast path 限定为非 speculative，避免未经验证的 spec prefill WAR 边界；二是提交 fix(attention): snapshot TRTLLM slots before WAR fence 将 SWA 缓存写入改为快照读取，防止栅栏提前后产生竞态。

- prefill 发布门控从宽到严 (design): maybe_publish_prefill_war_read_done 增加 spec_algorithm.is_none() 门控，target verify 则通过独立的 supports_target_verify_war_read_done 接口按算法声明能力决定是否参与。
- TRTLLM SWA 缓存写入读取实时映射的竞态 (correctness): 初始化 forward metadata 时一次性翻译 out_cache_loc 并保存到 forward_metadata.swa_out_cache_loc，cache 写入只读快照；新增 assert 保护缺失 / 长度不足路径。

风险与影响

- 风险：核心执行路径变更：*decode* 每次 *replay* 现在都会发布 *read-done* 事件，策略依赖 *use_captured_forward_metadata_for_breakable_cuda_graph* 与 *prefill_shared_reads_end_at_metadata_init* 两个 *backend* 属性；若 *backend* 声明与实际行为不符，调度器可能提前改写仍被 *forward* 读取的共享张量，造成静默错误。*TRTLLM MHA* 的 *_get_layer_cache_loc* 新增 *assert*，任何未填充 *swa_out_cache_loc* 的路径都会显式崩溃而非继续运行，*piecewise prefill* 的快照截取逻辑依赖 *batch* 长度约定。*prefill* 新功能默认关闭，风险可控；*target verify* 仅 *DFlash* 家族可提前放行，其他 *spec* 算法行为不变。非 *CUDA* 平台因 *make_war_read_done_event* 返回 *None*，自动回退旧行为。
- 影响：对调度器 / 执行器：*decode* 与 *dflash target verify* 的 *WAR* 屏障粒度更细，调度准备与 *forward* 内核可重叠，*PR* 内 *profiling* 显示 *scheduler ops* 从串行变为完全 *overlap*，多轮 *benchmark TTFT* 与 *forward* 占用率提升。对 *TRTLLM MHA SWA* 用户：修复了潜在同流竞态，行为更安全。对 *spec decode*：新 *capability* 接口为后续算法接入留出扩展点。对运维：新增环境变量控制 *prefill fast path*，默认关闭，便于逐步放量。
- 风险标记：核心路径变更，数据竞争风险，*CUDA* 图捕获，依赖后端声明

关联脉络

- PR #31686 Expand WAR fences to prefill support: PR body 明确说明本变更基于该 PR 并扩展 prefill 支持，是 WAR 栅栏细粒度量化的前序工作。