PR #35947 完整报告

sgl-project/sglang

Publish gated DSV4 DFLASH-family target-prefill read completion

合并时间：2026-08-27 12:33

原文链接：http://prhub.com.cn/sgl-project/sglang/pull/35947

执行摘要

- 一句话：DSV4 目标预填充提前发布共享读完成事件以提升重叠调度
- 推荐动作：值得精读。该 PR 是投机解码调度重叠的重要一环，展示了如何通过“提前快照共享读 + 声明 PRE_REPLAY 边界”来安全地放宽 WAR 屏障，对理解 DSV4 的 sparse prefill 数据流和 CUDA graph 元数据生命周期有直接帮助。建议结合 #34816 和 #34515 一起阅读，把握完整的调度重叠演进路径。但需关注合并后 CI 回归的修复情况，确认 token_to_kv_pool 问题已解决后再合入主线。

功能与动机

DSV4 sparse prefill 此前在第一个 attention 层才懒构建 request-to-token 与 Full-to-SWA 快照，PR body 指出“The overlap scheduler therefore could not safely release or mutate those shared inputs at the metadata boundary and had to retain them through a coarser forward boundary.” 由于 DFLASH/DSPARK 没有单独的 prefill draft-extend 读取器，对非 CP 目标 prefill，在发布 PRE_REPLAY 边界前物化快照可以避免与 sparse-prefill 读取器竞争，同时让后续调度工作与目标模型计算重叠。本 PR 是 #34816 的 target-prefill 对应物，从 #34515 的集成性能工作中提取，最终为移除所有阻塞后的 40% E2E 吞吐提升做出贡献。

实现拆解

按以下 4 步拆解实现：

1. 后端扩展（deepseek_v4_backend.py）：在 DSV4Metadata 中新增 prefill_shared_reads_snapshotted 布尔标志，并在 copy_ / refresh_for_breakable_cuda_graph_replay_ 中重置，避免 replay 残留污染；新增 prepare_prefill_shared_read_snapshot 方法，在满足环境变量开关、EXTEND 模式、spec_algorithm.is_dflash_family() 且非 CP-v2 时，若属 sparse prefill（num_qo_tokens > _LARGE_INDEXER_QUERY_THRESHOLD 或显式开启 SGLANG_OPT_FLASHMLA_SPARSE_PREFILL）则调用新增的 _build_sparse_prefill_chunk_cache 构建 chunk cache，并将 prefill_shared_reads_snapshotted 置为 True；shared_read_ends 在 EXTEND 且已快照时返回 SharedReadEnds.PRE_REPLAY。
2. Runner 接入（eager_runner.py、prefill_cuda_graph_runner.py）：在 init_forward_metadata 之后立即调用 prepare_prefill_shared_read_snapshot，并随后调

用 maybe_publish_prefill_shared_read_done；prefill CUDA graph runner 使用 padding 后的 token 数作为 num_qo_tokens，确保与 graph replay 的几何尺寸一致。

3. 事件门控（shared_read_event.py、base_attn_backend.py）：maybe_publish_prefill_shared_read_done 的投机算法判断从“仅无投机”放宽为“无投机或 DFLASH 家族”，其余投机算法（如 EAGLE/MTP）因存在后续 draft-extend 读取器而保持保守粗屏障；base_attn_backend.py 新增默认 no-op 的 prepare_prefill_shared_read_snapshot 钩子，保证未实现该语义的后端不会误声明 PRE_REPLAY。
4. 测试配套：在 test_deepseek_v4.py 新增 test_prefill_snapshot_declares_pre_replay_boundary、test_snapshot_builds_cache_only_for_sparse_prefill、test_sparse_prefill_snapshot_marks_success_only_after_build 三个单元测试，覆盖快照声明、dense/sparse 分支、构建成功后才标记；在 test_prefill_cuda_graph_padding.py 新增 test_replay_snapshot_uses_padded_token_count，验证 replay 使用 padding 后的 token 数调用快照；在 test_prefill_shared_read_done.py 新增 test_dflash_family_target_prefill_publishes，验证 DFLASH 家族事件发布路径。

关键文件：

- python/sglang/srt/layers/attention/deepseek_v4_backend.py（模块 注意力后端；类别 source；类型 core-logic；符号 prepare_prefill_shared_read_snapshot, _build_sparse_prefill_chunk_cache, shared_read_ends, DSV4Metadata）：核心实现文件：新增 prepare_prefill_shared_read_snapshot 与 _build_sparse_prefill_chunk_cache，扩展 DSV4Metadata 状态并修改 shared_read_ends 以声明 PRE_REPLAY 边界，是本次优化最关键的行为变更点。
- python/sglang/srt/model_executor/runner/eager_runner.py（模块 运行器；类别 source；类型 data-contract；符号 _execute_extend）：在 eager 执行的元数据初始化之后接入快照与事件发布，是快速路径在 eager 模式下的入口。
- python/sglang/srt/model_executor/runner_utils/shared_read_event.py（模块 事件发布；类别 source；类型 data-contract；符号 maybe_publish_prefill_shared_read_done）：事件发布门控逻辑的核心文件：将投机算法条件放宽到 DFLASH 家族，保留 EAGLE/MTP 的粗屏障，决定 PRE_REPLAY 事件是否最终发布。
- python/sglang/srt/layers/attention/base_attn_backend.py（模块 注意基类；类别 source；类型 core-logic；符号 prepare_prefill_shared_read_snapshot）：为所有 attention 后端定义默认 no-op 的 prepare_prefill_shared_read_snapshot 钩子，保证未实现该语义的后端不会误声明 PRE_REPLAY。
- python/sglang/srt/model_executor/runner/prefill_cuda_graph_runner.py（模块 图谱运行器；类别 source；类型 data-contract）：CUDA graph 模式下同样接入快照与事件发布，确保 replay 路径保持一致的 PRE_REPLAY 语义。
- test/registered/attention/unittests/dsv4/test_deepseek_v4.py（模块 DSV4 测试；类别 test；类型 test-coverage；符号 test_prefill_snapshot_declares_pre_replay_boundary, test_snapshot_builds_cache_only_for_sparse_prefill, test_sparse_prefill_snapshot_marks_success_only_after_build）：新增三个单元测试，覆盖快照声明、dense/sparse 分支、构建成功后才标记，是本次核心语义的回归防线。

- test/registered/unit/model_executor/runner/test_prefill_cuda_graph_padding.py（模块 图谱填充测试；类别 test；类型 test-coverage；符号 test_replay_snapshot_uses_padded_token_count）：验证 CUDA graph replay 使用 padding 后的 token 数调用快照，保证 replay 几何与捕获一致。
- test/registered/unit/model_executor/runner/test_prefill_shared_read_done.py（模块 共享读测试；类别 test；类型 test-coverage；符号 test_dflash_family_target_prefill_publishes）：验证 DFLASH 家族在 target prefill 场景能正确发布共享读完成事件，补充事件门控的测试覆盖。

关键符号：prepare_prefill_shared_read_snapshot, _build_sparse_prefill_chunk_cache, maybe_publish_prefill_shared_read_done, shared_read_ends, test_prefill_snapshot_declares_pre_replay_boundary, test_snapshot_builds_cache_only_for_sparse_prefill, test_sparse_prefill_snapshot_marks_success_only_after_build, test_replay_snapshot_uses_padded_token_count, test_dflash_family_target_prefill_publishes

关键源码片段

python/sglang/srt/layers/attention/deepseek_v4_backend.py

核心实现文件：新增 prepare_prefill_shared_read_snapshot 与 _build_sparse_prefill_chunk_cache，扩展 DSV4Metadata 状态并修改 shared_read_ends 以声明 PRE_REPLAY 边界，是本次优化最关键的行为变更点。

```
def prepare_prefill_shared_read_snapshot(
    self, forward_batch: ForwardBatch, *, num_qo_tokens: int
) -> None:
    # 先重置快照标记，避免上次 replay 残留影响本次判定。
    metadata = self.forward_metadata
    if isinstance(metadata, DSV4Metadata):
        metadata.prefill_shared_reads_snapshotted = False

    # 仅当启用 WAR 优化、处于 EXTEND 模式、DFLASH 家族且非 CP-v2 时才快照。
    snapshot_shared_prefill_reads = (
        envs.SGLANG_ENABLE_PREFILL_WAR_READ_DONE.get()
        and forward_batch.forward_mode == ForwardMode.EXTEND
        and self.model_runner.spec_algorithm.is_dflash_family()
        and not is_cp_v2_active(forward_batch)
    )
    if not snapshot_shared_prefill_reads:
        return

    assert isinstance(metadata, DSV4Metadata)
    # 稀疏 prefill 需要额外构建 chunk cache，否则只声明已快照。
    use_sparse_prefill = not _is_sm120 and (
        num_qo_tokens > _LARGE_INDEXER_QUERY_THRESHOLD
        or envs.SGLANG_OPT_FLASHMLA_SPARSE_PREFILL.get()
```

```
        )
        if use_sparse_prefill:
            metadata.sparse_prefill_cache = self._build_sparse_prefill_chunk_cache(
                forward_batch, num_qo_tokens=num_qo_tokens
            )
        # 无论 dense 还是 sparse，都标记共享读已完成快照，从而允许
        # shared_read_ends 向调度器声明 PRE_REPLAY 边界。
        metadata.prefill_shared_reads_snapshotted = True
```

python/sglang/srt/model_executor/runner_utils/shared_read_event.py

事件发布门控逻辑的核心文件：将投机算法条件放宽到 DFLASH 家族，保留 EAGLE/MTP 的粗屏障，决定 PRE_REPLAY 事件是否最终发布。

```
def maybe_publish_prefill_shared_read_done(
    model_runner, forward_batch, device_module
) -> None:
    """在合规的元数据初始化后发布 prefill 读完成事件。"""
    if not envs.SGLANG_ENABLE_PREFILL_WAR_READ_DONE.get():
        return
    if forward_batch.forward_mode != ForwardMode.EXTEND:
        return
    # 除 DFLASH 家族外，其他投机算法可能还有后续 draft-extend 读取器，
    # 保持保守屏障，避免提前释放共享输入导致数据竞争。
    if (
        not model_runner.spec_algorithm.is_none()
        and not model_runner.spec_algorithm.is_dflash_family()
    ):
        return
    # 记录点位于 replay 准备之后，因此只支持 PRE_REPLAY 边界。
    declared = model_runner.attn_backend.shared_read_ends(
        forward_batch.forward_mode
    )
    if declared is not SharedReadEnds.PRE_REPLAY:
        return
    logger.info_once(
        "Prefill shared-read-done fastpath active (%s)",
        type(model_runner.attn_backend).__name__,
    )
    read_done = device_module.Event()
    read_done.record()
    model_runner.shared_read_done_event = read_done
```

评论区精华

审查中无 inline review 评论，但 Issue 评论中有三条关键讨论：

- nvpohanh 报告 CI 回归：本 PR 合并后导致 test_q8kv8_sparse_prefill_backend.py 中四个测试失败，报错 AttributeError: 'DeepseekV4AttnBackend' object has no attribute 'token_to_kv_pool'，并留言“This broke CI 😱 ... I am fixing it.”。这提示该 PR 或 main

合并带入了对 DeepseekV4AttnBackend 接口的破坏，需要后续修复。

- YAMY1234 提供隔离验证：在 Qwen3.5 AgentX 工作负载上，Total TPS/GPU 为 116382.57 vs 117919.87 baseline（-1.304%），P90 TPS/User 为 108.631 vs 108.131（+0.462%），均在 3% 阈值内；但明确说明 Qwen3.5 使用 trtllm_mha，本 PR 修改的 DSV4 DFLASH 路径未被激活，因此该验证只确认隔离 cherry-pick 无回归，不验证 DSV4 主动路径性能。
- hnyls2002 触发 rerun：请求 rerun test_deepseek_v4.py、test_prefill_shared_read_done.py、test_prefill_cuda_graph_padding.py，相应 CI 在 4-gpu-b200 与 1-gpu-h100 上全部通过。
- 合并后 CI 回归：DeepseekV4AttnBackend 缺失 token_to_kv_pool (correctness): 本 PR 或 main 合并引入了对 DeepseekV4AttnBackend 接口的破坏，需要额外修复；截至 PR 合并时仍未解决。
- Qwen3.5 AgentX 无回归验证 (performance): 未观察到可测量的性能回归，但未验证 DSV4 主动路径性能。
- 测试 rerun 请求与结果 (testing): 相关针对性测试通过，未发现本 PR 引入的测试失败。

风险与影响

- 风险：主要风险如下：
- 合并后 CI 回归：nvpohanh 明确指出 test_q8kv8_sparse_prefill_backend.py 四个测试失败，DeepseekV4AttnBackend 缺失 token_to_kv_pool 属性。该回归可能源于本 PR 与 main 合并时的接口变化，也可能暴露了 DeepseekV4AttnBackend 初始化路径的隐式依赖，需要优先修复，否则影响主线稳定性。
- 行为开关默认关闭：快速路径由 SGLANG_ENABLE_PREFILL_WAR_READ_DONE 环境变量门控，默认不生效，因此默认部署风险可控；但一旦开启，所有依赖 PRE_REPLAY 边界安全性的路径都必须严格满足快照语义，否则可能出现共享输入被提前释放导致的数据竞争。
- 投机解码边界放宽：shared_read_event.py 将放行条件从“仅无投机”扩展至“DFLASH 家族”，若未来 DFLASH/DSPARK 引入新的按顺序读取器，可能破坏 WAR 保证；当前靠 is_dflash_family() 保守限定，但长期需补充更细粒度的边界声明。
- 快照重建开销：_build_sparse_prefill_chunk_cache 在元数据初始化阶段提前构建，会将该开销转移到 prefill 早期，若构建失败（如显存不足或异常），prefill_shared_reads_snapshotted 不会被置位，测试也覆盖了“成功后才标记”的语义，但实际失败恢复路径仍需观察。
- CUDA graph replay 一致性：prefill graph runner 使用 padding 后的 token 数调用快照，测试已验证 num_qo_tokens=16 的 padding 场景，但其他 padding 倍数及多 bucket 场景的覆盖仍有限。
- 影响：影响范围集中在 DSV4 模型 + DFLASH/DSPARK 投机解码 + 非 CP-v2 的 target prefill 路径，且默认关闭，因此对现有默认部署无行为变化。对启用该优化的用户，提前发布共享读完成事件可提升 prefill 与调度器的重叠度，为整体 40% E2E 吞吐改进贡献一部分；但对 Qwen3.5 等使用 trtllm_mha 的工作负载无影响（YAMY1234 验证）。对团队而言，新增的“快照钩子 + 事件发布”模式将成为后续其他后端实现同类优化的模板，同时需要维护 base_attn_backend.py 新增的 no-op 钩子和 shared_read_event.py 的投机算法门控，

避免后续后端误用。

- 风险标记：合并后 CI 回归，默认关闭（opt-in），核心路径变更，投机算法门控放宽，需验证 DSV4 主动路径性能

关联脉络

- PR #34816 Publish DSPARK *decode*-verify WAR boundary: PR body 明确指出本 PR 是其 target-prefill 对应物，二者共同完成 DSV4 投机解码的 prefill 与 *decode*-verify 共享读完成事件发布。
- PR #34515 DSV4 prefill overlap integration and performance work: 本 PR 从 #34515 中提取，该集成工作最终实现移除所有阻塞后约 40% 的 E2E 吞吐提升；本 PR 是其中 prefill 侧的关键拆分子项。