PR #28451 完整报告

sgl-project/sglang

[GDN][KDA] ReplaySSM buffered output-only decode for Linear Attention

合并时间：2026-06-26 14:17

原文链接：http://prhub.com.cn/sgl-project/sglang/pull/28451

执行摘要

- 一句话：线性注意力加入 ReplaySSM 缓冲解码，HBM 流量减半
- 推荐动作：值得精读，特别是 ReplaySSM 数学推导、CUDA Graph 安全设计、Radix 协调机制。对于部署 GDN/KDA 模型的组织是重要优化方向。

功能与动机

源自 Issue #28511。GDN *decode* kernel 在批≥64 时 70-79% 受显存带宽限制，因为每步读写完整状态 [HV,V,K]。ReplaySSM 技术消除每步状态写入，HBM 流量约减半。该 PR 将 ReplaySSM *decode* 路径移植到 SGLang。

实现拆解

1. Triton Kernel(fused_recurrent_linear_replayssm.py): 新增缓冲输出专用解码 kernel，通过 IS_KDA 编译期参数统一 GDN 标量门和 KDA 逐 K 门。环缓冲大小由 --linear-replayssm-cache-len 控制（默认 16）。非刷新步仅从检查点 S0 + 环重建输出，完整状态 S 每 L 步刷新一次。
2. 后端集成(hybrid_linear_attn_backend.py): 在 MambaAttnBackendBase 中添加静态 CUDA Graph 安全缓冲区 (replayssm_write_pos_list, replayssm_force_flush_list)。_forward_metadata 中管理每步游标推进和强制刷新掩码。新增 _replayssm_enabled 和 _replayssm_track_flush_mask 方法。
3. 内存池扩展(memory_pool.py): MambaPool 新增 enable_linear_replayssm 和 linear_replayssm_cache_len 参数，按需分配 replayssm_d/k/g 环形张量。copy_from 增加环缓存复制和游标重置，并添加调试断言确保源槽已刷新。
4. 配置与开关(server_args.py, mamba_utils.py): 新增 --enable-linear-replayssm 和 --linear-replayssm-cache-len 参数。mamba_utils.py 添加 is_kda 属性和 num_k_heads_per_tp 以支持 KDA 环尺寸。gdn_triton.py/kda_triton.py 根据开关分发调用。
5. Radix 前缀缓存协调(mamba_radix_cache.py): 通过 force_flush 对齐 Radix 快照条件 (seq_lens_cpu % mamba_track_interval == 0)，确保缓存状态一致性。
6. 微基准与严格测试: 新增 bench_gdn_replayssm_decode.py 对比 packed vs replay 延迟和流量；test_linear_replayssm_decode.py 验证 L=1/4/8/16 下与 packed 基线数值一致，覆盖 GDN/KDA 的 fp32 和 bf16。

关键文件：

- python/sglang/srt/layers/attention/fla/fused_recurrent_linear_replayssm.py（模块 解码内核；类别 source；类型 dependency-wiring；符号 fused_recurrent_linear_replayssm_decode_kernel, fused_recurrent_linear_replayssm_decode）：核心 Triton kernel 实现，统一 GDN/KDA 缓冲解码。
- python/sglang/srt/layers/attention/hybrid_linear_attn_backend.py（模块 注意力后端；类别 source；类型 core-logic；符号 _replayssm_enabled, _replayssm_track_flush_mask）：后端集成层，管理 CUDA Graph 安全缓冲和游标推进、强制刷新掩码。
- python/sglang/srt/mem_cache/memory_pool.py（模块 内存池；类别 source；类型 core-logic）：MambaPool 扩展环形缓冲分配和 copy_from 环复制 / 断言。
- test/registered/attention/unittests/gdn/test_linear_replayssm_decode.py（模块 单元测试；类别 test；类型 test-coverage；符号 TestLinearReplaySSMDecode, _run_one, _tols, _build_inputs）：严格正确性测试，覆盖 GDN/KDA 在各种 L 和精度下的数值等价。
- python/sglang/srt/layers/attention/fla/bench_gdn_replayssm_decode.py（模块 基准测试；类别 source；类型 dependency-wiring；符号 _make_static, _state_bytes_per_step, _bench_cfg, main）：微基准测量 packed vs replay 延迟和状态流量。

关键符号：fused_recurrent_linear_replayssm_decode_kernel, fused_recurrent_linear_replayssm_decode, _replayssm_enabled, _replayssm_track_flush_mask, MambaPool.init, MambaPool.copy_from, BaseLinearStateParams.is_kda, Mamba2StateShape.create, KimiLinearStateShape.create

评论区精华

- copy_from 环状态一致性(@kaixih): 指出 copy_from 调用假设源槽已完全刷新，但 prefill 完成时无环条目，decode 复制未刷新槽可能丢失环中数据。@yuan-luo 确认并添加调试断言 (write_pos[src] == 0).all() 及文档说明。
- PD 分解模式 guard(@kaixih): 提醒 disagg decode 侧 HybridMambaDecodeReqToTokenPool 未接收参数。@yuan-luo 添加显式 guard 在 disagg 模式中禁用 ReplaySSM。
- KDA 性能差异(@yyq0210): 观察到 GDN 有 1.2-1.37× 提升，KDA 因每步环写入更大 (g_cache [slots, HV, L, K]) 在微基准中无正面收益。@yuan-luo 认为环写入开销较小但 baseline 更快，待进一步分析 L 选择。
- copy_from 环状态一致性 (correctness): @yuan-luo 确认并添加调试断言 (write_pos[src] == 0).all() 和文档说明。
- PD 分解模式 guard (design): @yuan-luo 添加显式 guard 在 disagg 模式中禁用 ReplaySSM。

- GDN vs KDA 性能差异 (performance): @yuan-luo 认为 KDA 环写入 [H,L,K] 对带宽影响不大，但 baseline 更快？待进一步分析。

风险与影响

- 风险：
  - 张量核心精度依赖: 重建点积必须在 Tensor Core 上运行以保持性能；若使用 IEEE fp32 精确模式，kernel 可能慢 10-20×。测试容差已设为 TF32 (~4e-4) / bf16 (~1e-3)，但用户若修改精度模式有回退风险。
  - copy_from 环状态丢失: 若从解码中间槽复制且尚未刷新，环中条目会丢失。已添加调试断言捕获，但生产环境可能未开启。
  - 增量内存开销: 每槽增加 d: HV*L*V, k: H*L*K, g 存储，相对全状态约额外 7-15% (L=16)。
  - Disagg 未完全支持: 当前 disagg 模式与 ReplaySSM 互斥，后续需扩展至预填充 / 解码池。
- 影响：
  - 用户: 默认不开启。启用后大 batch decode (>64) 典型加速 1.2-1.5×；KDA 模型可能无收益甚至略降，需按模型评估。
  - 系统: 新增环缓冲区，内存占用小幅增加；CUDA Graph 捕获后无额外 CPU 开销。
  - 团队: 引入统一 kernel 和 gate 特定路径，维护成本增加；但测试覆盖全，降低回归风险。
  - 风险标记：核心路径变更，张量核心精度要求，copy_from 环状态一致性，Disagg 未完全支持

关联脉络

- PR #28511 [RFC] Porting ReplaySSM to SGLang: faster decode and speculative decoding for hybrid (GDN/KDA) models: 该 PR 的动机来源和 Part A 实现源头。
- PR #28695 [GDN][KDA] ReplaySSM verify for speculative decoding: Part B 实现，共享环基础设施。
- PR #27658 Compact linear spec cache: 关联 spec-decode 内存优化，环缓冲思想类似。