PR #33576 完整报告

sgl-project/sglang

[AMD] Add Work-Centric (Lean) Attention: a persistent-CTA decode kernel for long-context serving

合并时间：2026-08-29 08:00

原文链接：http://prhub.com.cn/sgl-project/sglang/pull/33576

执行摘要

- 一句话：新增 AMD 持久 CTA 工作窃取解码注意力内核
- 推荐动作：值得精读：WCA 的 persistent-CTA + work-stealing + capture-time 决策三个设计点对注意力 kernel 工程很有借鉴价值；建议重点看自动门控在 eager 与 CUDA graph 下的两套决策逻辑，以及 1xCU 网格在 MI300X/MI355X 上的 A/B 验证过程。若团队后续要支持更长上下文或引入更多 AMD 内核，本 PR 的分层（kernel / dispatch / gate / bench / test）是一个很好的模板。

功能与动机

PR 指出标准 flash-decoding（SplitK）在长上下文低 batch 时按 batch × head-tiles × num_splits 启动固定网格，长序列难以填满所有 CU，导致大量 CU 空闲而 decode 延迟随上下文近乎线性增长；在 ragged batch 下也无法把长请求的工作量再平衡到短请求空出的 CU 上。为此引入 Work-Centric Attention（WCA / Lean）：固定 CU 数量的持久网格 + 设备端工作窃取，使延迟几乎平坦并吸收 raggedness（ITL 最高降低 3.62 倍），同时保持与标准内核数值一致。

实现拆解

1. 内核与启动辅助（python/sglang/kernels/ops/attention/decode_attention.py）：新增 _lean_attention_decode_kernel、_decode_lean_attention_fwd、_lean_decode_launch_params、_lean_head_dim_ok、_lean_num_cus 等。decode_attention_fwd 在入口处增加 enable_lean、lean_Mp 等参数，并在 _is_hip、head_dim 合法、gate 通过时提前分派到 Lean 路径。持久网格按设备 CU 数确定（SGLANG_FORCE_LEAN_GRID_CU_MULT 默认 1.0），内核从 kv_indptr 推导全局 tile 计划并在线做跨 CTA 归约。
2. 后端接线（python/sglang/srt/layers/attention/triton_backend.py）：ForwardMetadata 新增 lean_Mp/lean_Lp/lean_Op/lean_locks 字段；__init__ 从 model_runner.server_args.enable_lean_attention 读取开关并预计算 lean_total_programs；init_forward_metadata 在 decode 路径按持久网格大小分配四块缓冲。决策链为 explicit override → 自动门控 → kill-switch（SGLANG_DISABLE_LEAN_ATTENTION）。

3. 配置入口（python/sglang/srt/server_args.py、environ.py）：--enable-lean-attention 是 Optional[bool]（默认 None 自动），归入 exec.kernel 命名空间；SGLANG_DISABLE_LEAN_ATTENTION 作为紧急隔离开关，SGLANG_FORCE_LEAN_GRID_CU_MULT 作为持久网格大小 A/B 调参旋钮，均无需 rebuild。
4. 自动门控：eager 模式用 lean_decode_seqlen_gate（基于 query-head 并行度 tiles = ceil(num_q_heads / min(16, kv_group)) 与 batch 放宽阈值）；CUDA graph 捕获期用 lean_capture_policy（基于 batch、tiles、is_mla 预决：GQA/MHA batch ≥ 16、MLA batch ≥ 8、tiles < 4 不启用），保证捕获图在 replay 时按真实 seq_len 工作窃取，无 host sync。
5. 测试与基准：test/registered/kernels/test_lean_attention.py 通过 _run_pair 系列在 bf16、fp8、paged KV 上对比 SplitK 与 Lean，验证余弦接近 1（分别为 0.999、0.99、0.999）并锁定 gate/policy 行为；benchmark/lean_kernel_sweep.py 对 Qwen2.5-7B 与 Llama-3.1-8B，在 batch {1..32} × context {8K..128K} 上输出延迟、加速比、余弦与 gate 决策 CSV。

关键文件：

- python/sglang/kernels/ops/attention/decode_attention.py（模块 解码内核；类别 source；类型 core-logic；符号 _lean_head_dim_ok, _lean_num_cus, _lean_decode_block_n, remap_xcd）：Lean 内核本体所在文件（+832 行）：新增 _lean_attention_decode_kernel、_decode_lean_attention_fwd、_lean_decode_launch_params 以及 lean_decode_seqlen_gate / lean_capture_policy 自动门控；decode_attention_fwd 增加 enable_lean 分派。
- test/registered/kernels/test_lean_attention.py（模块 CI 测试；类别 test；类型 test-coverage；符号 _run_pair, _lean_scratch, _run_pair_fp8, _call）：CI 门控测试（stage-b-test-1-gpu-small-amd-mi35x）：校验 Lean 与标准 SplitK 在 bf16/fp8/paged KV 上的余弦一致性，并锁定 eager gate 与 capture policy 的行为，防止内核回归。
- python/sglang/srt/layers/attention/triton_backend.py（模块 注意力后端；类别 source；类型 dependency-wiring）：注意力后端接线：ForwardMetadata 新增 lean_Mp/Lp/Op/locks 字段；__init__解析 enable_lean_attention 并缓存 _lean_decode_launch_params 结果；init_forward_metadata 按持久网格分配解码缓冲，forward_decode 按 explicit → auto-gate → kill-switch 顺序裁决。
- benchmark/lean_kernel_sweep.py（模块 性能基准；类别 source；类型 dependency-wiring；符号 bench, run, main）：Kernel 级 sweep 基准：对 Qwen2.5-7B（28Q/4KV）与 Llama-3.1-8B（32Q/8KV），在 batch {1..32} × context {8K..128K} 上对比 SplitK 与 Lean 的延迟、加速比、余弦一致性与 eager gate 决策，输出 CSV 供 PR 表格复现。
- python/sglang/srt/environ.py（模块 环境配置；类别 source；类型 core-logic）：新增 SGLANG_DISABLE_LEAN_ATTENTION 紧急开关与 SGLANG_FORCE_LEAN_GRID_CU_MULT 持久网格扩展倍率，为线上隔离与网格 A/B 提供无需 rebuild 的旋钮。

- python/sglang/srt/server_args.py（模块 启动参数；类别 config；类型 configuration；符号 enable_lean_attention）：新增 --enable-lean-attention 参数（Optional[bool]，默认 None 走自动门控），并归入 exec.kernel 命名空间，满足 server_args namespace CI 检查。

关键符号：_lean_attention_decode_kernel, _decode_lean_attention_fwd, _lean_decode_launch_params, _lean_head_dim_ok, lean_decode_seqlen_gate, lean_capture_policy, _should_use_lean_decode, decode_attention_fwd

关键源码片段

python/sglang/kernels/ops/attention/decode_attention.py

Lean 内核本体所在文件（+832 行）：新增 _lean_attention_decode_kernel、_decode_lean_attention_fwd、_lean_decode_launch_params 以及 lean_decode_seqlen_gate / lean_capture_policy 自动门控；decode_attention_fwd 增加 enable_lean 分派。

```
# decode_attention.py —— Lean 分派入口（位于 decode_attention_fwd 顶部）
# 仅在 ROCm/AMD 上开启（_is_hip），且 head_dim 必须能装进共享内存；
# logit_cap / sink / xAI 温度 / score_mod 等未实现特性则退回标准 SplitK。
if (
    _is_hip
    and _lean_head_dim_ok(k_buffer.shape[-1], v_buffer.shape[-1])
    and _should_use_lean_decode(
        enable_lean, logit_cap, sinks, xai_temperature_len, score_mod
    )
):
    # 持久网格大小只取决于设备 CU 数与 KV 头分组，与 batch 无关，
    # 因此可以放入 CUDA graph，并在每次 replay 时按 kv_indptr 推导 tile 计划。
    total_programs, XCD_REMAP, NUM_XCDS = _lean_decode_launch_params(
        v_buffer.shape[-2], kv_group_num
    )
    _decode_lean_attention_fwd(
        q, k_buffer, v_buffer, o,
        kv_indptr, kv_indices, total_programs,
        # 将 k_scale 折入 sm_scale，并单独传 v_scale：
        # 与标准 grouped kernel 的 fp8 反量化完全一致，保证数值等价。
        sm_scale * k_scale, v_scale,
        XCD_REMAP, NUM_XCDS,
        lean_Mp, lean_Lp, lean_Op, lean_locks,
        page_size=page_size,
    )
    return
```

评论区精华

PR 无独立 code review 评论，HaiShaw 直接 APPROVED 并无备注。Issue 侧只有 CI 命令与机器人提示（/tag-and-rerun-ci、@amd-bot ci-status、Gemini 停运通知），不涉及技术讨

论。技术决策全部沉淀在 23 个 commit 中：fp8 支持、paged KV 地址数学、CUDA graph capture 策略、seq_lens_sum = None 的 EAGLE crash 修复、tl.static_range 编译崩溃修复、1xCU 网格默认值。

- CI 与合并流程 (question): 合并者认可后直接合并，无未解决技术疑虑。

风险与影响

- 风险：风险集中在默认自动门控上：AMD 上长上下文（gate 命中）的 decode 路径会切换为新内核，虽然数值等价性由 CI 测试锁定，但对 gate 阈值和网格倍率的改动未覆盖所有硬件组合。CUDA graph 捕获期曾出现 tl.static_range 编译崩溃（commit 070d1f3 改为 while 循环），说明 Triton 编译器边界仍可能在其他组合复现。fp8 路径把 k_scale 折入 sm_scale 并 cast q → K.dtype，任何改动能轻易破坏一致性，依赖 test_fp8_kv_parity 守住。_lean_head_dim_ok 对 head_dim 256（如 Gemma-2/3）会走标准 kernel，属于防护性降级，但若 LDS 预算判断不准确仍可能触发 OutOfResources。EAGLE 场景 seq_lens_sum = None 曾引发 TypeError（commit 8693652 修复），说明 speculative decoding 与 gate 的组合仍需关注。
- 影响：对 AMD 用户：在 MI355X/MI300X 上，长上下文（64K–128K）低 batch 与 ragged 场景 decode 延迟显著下降（E2E 最高 1.52x，ITL 最高 3.62x），且默认保守 gate 保证短上下文与 NVIDIA 不受影响。对系统：新增 1 个 server arg、2 个环境变量和 1 个 CI 门控测试；标准 SplitK 路径在 Lean 未启用时完全不变。对团队：内核与门控集中沉淀在 decode_attention.py，后续调参只需动 gate 阈值或环境变量，无需 rebuild，便于在 AMD 硬件上做长上下文解码的持续优化。
- 风险标记：AMD-only 分派保护，默认自动门控影响 AMD decode 路径，CUDA graph 捕获期 Triton 编译兼容，fp8 反量化数值等价依赖，head_dim 256 共享内存溢出防护

关联脉络

- PR #36094 [AMD][DSV4] perf: retune decode split-K heuristic for MI355X: 同属 AMD 解码注意力路径的性能调优：该 PR 调优 DSv4 split-K 启发式并配套拆分启发式测试，与 Lean 共享 decode_attention 性能优化语境；两者都涉及 AMD decode 内核的 split/ 调度决策，可对照阅读。