

PR #51794 完整报告

vllm-project/vllm

[ROCm][Perf] Enable CSA multi-stream overlap for DeepSeek-V4

合并时间: 2026-09-14 19:21

原文链接: <http://prhub.com.cn/vllm-project/vllm/pull/51794>

执行摘要

- 一句话: ROCm 上为 DeepSeek-V4 CSA 注意力引入多流重叠, 吞吐最高 +4.6%
- 推荐动作: 值得精读, 重点看 vllm/models/deepseek_v4/amd/rocm.py 中 `_enable_csa_multi_stream`、`_run_sequential_pipeline` 与 `_prepare_and_attn` 三者构成的“门控 + 递归回退”结构, 以及 `execute_in_parallel` 传入 `self.ln_events[0]` / `ln_events[1:3]` / `aux_streams[:2]` 时对事件与流的复用约定。设计上最值得注意的是: 用“是否处于 capture / metadata 是否 dict”来等价替代“是否存在 eager break”, 把多流的正确性押在 cudagraph 的静态图上, 这是一种以编译期结构换运行时同步可靠性的取舍, 值得借鉴但也最脆弱。建议在采纳前补齐两点: 为 capture 与非 capture 两条路径增加最小可重复的测试或 nightly e2e, 并保留一个 (哪怕是构建期) 的回退开关以应对 HIP event 挂死。

功能与动机

PR body 明确目标是 “Enable kernel-level multi-stream overlap for DeepSeek-V4 CSA (Compressed Sparse Attention, `compress_ratio=4`) attention layers on ROCm”, 动机是 CSA 层前向里主 compressor 与 indexer compressor 的 `wkv_gate` GEMM 挂在默认流的串行关键路径上, 而这两条链都从 `hidden_states` 出发、自成闭环 (body 原文: “each stream runs a self-contained chain straight from `hidden_states`, so no fine-grained cross-stream dependencies are required”), 因此天然适合拆到侧流。trace 对比显示 CSA 层耗时下降 12.99%, 是本次改动的直接收益来源; 此前 ROCm 因挂死问题在 `DeepseekV4Model` 里直接禁用了全部 `aux stream`, 本 PR 正是要把这部分算力重新拿回来。

实现拆解

- 打开 ROCm 的辅助流 (vllm/models/deepseek_v4/amd/model.py): 删除 `None if current_platform.is_rocm() else [torch.cuda.Stream() for _ in range(3)]` 的 ROCm 禁用分支, 改为所有平台统一创建 3 条 `torch.cuda.Stream()`, 同时移除“Disable them on ROCm because of hang issues”的注释; 是否真正使用改由注意力层按层判断。
- 收紧多流开关 (vllm/models/deepseek_v4/amd/rocm.py):
`DeepseekV4ROCMAttnMLAAttention.__init__` 中, `self.indexer is None` 的层把 `aux_stream_list` 置 `None` (目前只让 CSA 层重叠), CSA 层则把 `indexer.aux_stream` 置 `None` 禁用 indexer 内部嵌套重叠。新增 `_enable_csa_multi_stream()` 作为唯一判据: `aux_stream_list is not None and (torch.cuda.is_current_stream_capturing() or not`

`isinstance(attn_metadata, dict))`，用于排除 `piecwise cudagraph` 的 `eager break` 区（该处 `fork` 侧流依赖运行时 `HIP event` 同步，在 `ROCm` 上可能挂住）。

3. 重排 `CSA` 前向（`rocm.py` 的 `forward / _run_sequential_pipeline`）：`forward` 先按 `FlashMLA` 对齐后的 `head` 数预分配 `o_padded`；命中开关时用全 `None` 哨兵调用 `_prepare_and_attn`，把 `fork/join` 提前到 `capture` 边界内；未命中则走 `_run_sequential_pipeline`，它临时把 `aux_stream_list` 置空后复用同一条调用链，保证 `eager` 区域与改动前逐算子等价（`_prepare_and_attn` 里 `aux_streams` is `None` 的分支再接回 `super()._prepare_and_attn`，形成受控递归）。
4. 三条链 `fork/join`（`rocm.py` 的 `_prepare_and_attn`）：`default_chain` 在默认流跑 `fused wqa+wkv GEMM` → `_split_qkv_and_norm` → `_wq_b_proj` → `_fused_qnorm_rope_kv_insert`（`SWA KV` 写入）；`main_compressor_chain` 与 `indexer_compressor_chain` 分别在 `aux stream 0/1` 跑 `torch.mm(hidden_states, fused_wkv_wgate.weight.T, out_dtype=torch.float32)` 加各自 `compressor` 写入；通过 `execute_in_parallel(default_chain, [...], self.ln_events[0], self.ln_events[1:3], aux_streams[:2], enable=True)` 汇合。`join` 之后串行执行 `indexer.weights_proj`、`indexer(..., skip_compressor=True)` 与 `_sparse_indexer_and_attn`。
5. 契约与配套改动：`vllm/models/deepseek_v4/attention.py` 的 `DeepseekV4Indexer.forward` 把 `compressed_kv_score` 改为 `torch.Tensor | None` 并新增 `skip_compressor: bool = False`，在 `dict metadata` 的短上下文分支和 `maybe_execute_in_parallel` 分支上分别以 `if not skip_compressor` 守卫，避免压缩器重复执行；`vllm/models/deepseek_v4/cpu/cpu_sparse.py` 的 `DeepseekV4CPUIndexer.forward` 同步签名以免子类不匹配。测试配套为空白：本次没有新增或修改任何测试文件，验证只依赖作者的手工基准（`SA InferenceX 8k1k`）与 `gsm8k` 精度测试。

关键文件：

- `vllm/models/deepseek_v4/amd/rocm.py`（模块 注意力层；类别 `source`；类型 `core-logic`；符号 `_enable_csa_multi_stream`, `_run_sequential_pipeline`, `forward`, `_prepare_and_attn`）：本次改动的全部实现落点：新增 `_enable_csa_multi_stream / _run_sequential_pipeline / forward / _prepare_and_attn`，把两条 `compressor` 链拆到 `aux stream 0/1` 并用 `execute_in_parallel` 统一 `join`，同时在 `__init__` 里按层决定是否启用多流。
- `vllm/models/deepseek_v4/attention.py`（模块 索引器；类别 `source`；类型 `data-contract`；符号 `DeepseekV4Indexer.forward`, `compressed_kv_score`, `skip_compressor`）：被 `ROCm` 路径依赖的契约变更：`DeepseekV4Indexer.forward` 新增 `skip_compressor` 开关、`compressed_kv_score` 变为 `Optional`，使外层可以接管压缩器；同时删除了“`ROCm` 上 `aux_stream_list` 为 `None`”的过时注释。
- `vllm/models/deepseek_v4/amd/model.py`（模块 模型初始化；类别 `source`；类型 `configuration`；符号 `DeepseekV4Model.init`, `aux_stream_list`）：取消 `ROCm` 上禁用 `aux stream` 的全局分支，改为所有平台统一创建 3 条 `torch.cuda.Stream()`，是多流优化能生效的前提；是否使用改由注意力层按层与执行区决定。

- `vllm/models/deepseek_v4/cpu/cpu_sparse.py` (模块 CPU 索引器; 类别 `source`; 类型 `data-contract`; 符号 `DeepseekV4CPUIndexer.forward`, `skip_compressor`) : CPU 覆写的 `DeepseekV4CPUIndexer.forward` 需要同步基类签名 (Optional 的 `compressed_kv_score` 与新增的 `skip_compressor`) , 否则子类与基类接口不一致; 但函数体并未消费 `skip_compressor`, 属于签名对齐型改动。

关键符号: `_enable_csa_multi_stream`, `_run_sequential_pipeline`,
`DeepseekV4ROCMAiterMLAAttention.forward`,
`DeepseekV4ROCMAiterMLAAttention._prepare_and_attn`, `default_chain`,
`main_compressor_chain`, `indexer_compressor_chain`, `DeepseekV4Indexer.forward`,
`DeepseekV4CPUIndexer.forward`, `DeepseekV4Model.init`

关键源码片段

`vllm/models/deepseek_v4/attention.py`

被 ROCm 路径依赖的契约变更: `DeepseekV4Indexer.forward` 新增 `skip_compressor` 开关、`compressed_kv_score` 变为 Optional, 使外层可以接管压缩器; 同时删除了“ROCm 上 `aux_stream_list` 为 None”的过时注释。

```
def forward(
    self,
    hidden_states: torch.Tensor,
    qr: torch.Tensor,
    # 压缩分数改为可选: ROCm 多流路径下压缩器已由外层 aux stream 执行,
    # 这里只做 q 侧投影。
    compressed_kv_score: torch.Tensor | None,
    indexer_weights: torch.Tensor,
    positions: torch.Tensor,
    rotary_emb: nn.Module,
    qr_scale: torch.Tensor | None = None,
    # 新增: True 表示 compressor 已在别处 (aux stream 1) 跑过, 跳过它。
    skip_compressor: bool = False,
) -> tuple[torch.Tensor | None, torch.Tensor | None, torch.Tensor | None]:
    compressor = self.compressor

    attn_metadata = get_forward_context().attn_metadata
    if isinstance(attn_metadata, dict):
        indexer_metadata = cast(Any, attn_metadata[self.k_cache.prefix])
        if (
            indexer_metadata.max_seq_len // self.compress_ratio <= self.topk_tokens
            and not torch.cuda.is_current_stream_capturing()
        ):
            # 候选数少于 topk: 全部被选中, 但 K cache 仍需构建。
            # 注意 ROCm 多流路径只会出现在非 dict metadata (无 eager break) 下,
            # 因此这里通常不会命中; 守卫保留是为防御未来的调用方变化。
            if not skip_compressor:
                compressor(compressed_kv_score, positions, rotary_emb)
            assert self.topk_indices_buffer is not None
```

```

# 省略: 填充短上下文 topk indices 并按 num_tokens 触发 Triton kernel
return None, None, None

def wq_b_and_q_quant():
    q = self._wq_b_proj(qr, qr_scale)
    q = q.view(-1, self.n_head, self.head_dim)
    return fused_indexer_q_rope_quant(
        positions, q, rotary_emb.cos_sin_cache, indexer_weights,
        self.softmax_scale, self.n_head**-0.5, use_fp4=self.use_fp4_kv,
    )

if not skip_compressor:
    # compressor 返回 None 并把 K 写入 indexer KV cache; join 保证这次
    # 写入排在 indexer_op 之前 (skip_k_cache_insert=True)。
    (q_quant, weights), _ = maybe_execute_in_parallel(
        wq_b_and_q_quant,
        lambda: compressor(compressed_kv_score, positions, rotary_emb),
        self.ln_events[0],
        self.ln_events[1],
        self.aux_stream,
    )
else:
    # 压缩器已在 aux stream 1 上跑完, 这里直接走 q 侧投影与 RoPE/ 量化。
    q_quant, weights = wq_b_and_q_quant()

if isinstance(q_quant, tuple):
    q, q_scale = q_quant
else:
    q, q_scale = q_quant, None
return q, q_scale, weights

```

评论区精华

讨论集中在两处：一是 mawong-amd 质疑是否还需要为“eager 多流挂死”保留门控，作者解释该 PR 要同时落地 rocm-7.x 与 rocm-10.0、rocm-7.x 上 eager 多流仍会 hang，且尚未在 rocm-10.0 上验证 eager 路径，因此 `_enable_csa_multi_stream` 的 capture/metadata 判据必须保留；二是 tjtaana 依据 issue #17067 要求缓存

`envs.VLLM_ROCM_DSV4_CSA_MULTI_STREAM` 的读取，作者回应该优化已计划默认开启、环境变量门控是漏删的残留，并在 commit c850e5f1d5b1 中删除。整体上 review 由 AMD 侧维护者（Fangzhou-Ai、tjtaana）以 LGTM 通过，未曾对测试覆盖或缺省开启后的回退策略提出要求。

- ROCm 10 nightly 上是否还需要 eager 挂死门控 (correctness): 保留按 capture region 分流的门控（capture 中或 metadata 非 dict 才启用多流）；rocm-10.0 的 eager 多流仍属未验证区域。
- 环境变量读取是否要缓存 / 是否保留门控 (design): 环境变量门控整体移除，多流优化默认开启，`_enable_csa_multi_stream` 不再逐层读取 envs，也就无需缓存。

- 验证口径与测试覆盖 (testing): 以维护者人工验证 + 通用 CI 通过为合并依据, 未新增针对多流 /capture 边界的自动化测试。

风险与影响

- 风险:

1. GPU hang / 正确性 (最高风险): 多流的 fork/join 依赖 HIP event 同步, 在 piecewise cudagraph 的 eager break 区可能挂住或结果错乱; `_enable_csa_multi_stream()` 用 `not isinstance(attn_metadata, dict)` 作为“非 eager 区”的隐式代理, 一旦 cudagraph 元数据的表示方式、MRV1/MRV2 的 capture 切分逻辑 (`_prepare_and_attn_fn` 在 MRV1 下会切到 `_prepare_and_attn_eager`, 见 #51430) 发生变化, 这个判据就可能静默失效。rocm-10.0 上的 eager 多流也未被作者验证。
2. 缺少回退开关: 提交历史显示 `envs.VLLM_ROCM_DSV4_CSA_MULTI_STREAM` 门控已被删除 (commit fb2e6824、c850e5f1), 优化默认开启, 线上若出现 hang 或精度异常只能回滚版本。
3. 跨平台契约变更: `compressed_kv_score` 改为 Optional、`skip_compressor` 新增为公共参数, 但 `cpu_sparse.py` 的 CPU 覆写只加了签名、内部仍无条件调用 `compressor(compressed_kv_score, ...)`; 若未来 CPU 侧以 `skip_compressor=True + None` 调用, 会直接抛错。
4. 测试空白与回归信号弱: 无单测 /E2E 覆盖, 验证范围仅限 8k1k 负载与 gsm8k (1,319 例, 0.9439)。
5. 性能面: `conc=16` 时 Mean TTFT 反而 +0.50%, 说明额外 stream fork/join 与 event 记录存在固定开销, 短序列或小 batch 场景收益可能为负。
6. 可维护性: `_run_sequential_pipeline` 通过临时改写 `self.aux_stream_list` 再经 `_prepare_and_attn_fn` 回调自身, 靠 `aux_streams is None` 终止递归, 状态可变 + 隐式递归对后续改动不友好。- 影响: 对用户: ROCm (gfx950 系列) 上部署 DeepSeek-V4 的用户直接获益, 吞吐在 `conc=1/4/16/64` 分别 +4.59%/+3.82%/+2.30%/+1.33%, TPOT 全线下降 1.3%~4.0%, TTFT 三档改善 (`conc=1` 最多 -16.39%), 精度无回退; 但这些收益以默认开启的方式生效, 用户无法通过环境变量关闭。对系统: 注意力前向与 CUDA/HIP graph capture 的耦合度上升, DeepSeek-V4 在 ROCm 上开始拥有与 CUDA 不同的前向编排路径 (ROCm 专用 `_prepare_and_attn` 覆写), 后续任何针对 DeepseekV4Attention 基类前向的重构都必须同时兼顾这条 ROCm 分支。对团队: ROCm 与 NVIDIA 的维护成本出现分叉, 且本次缺少自动化测试, 长期需要补充 capture/eager 两种执行区下的回归覆盖。- 风险标记: GPU hang 风险, 缺少测试覆盖, 依赖 cudagraph 元数据形态, 默认开启无回退开关, 跨平台接口变更, 隐式递归控制流

关联脉络

- PR #51700 [2/2][Model Runner V2] FULL CUDA graph capture for microbatched steps (DBO): 同属多流 overlap 与 cudagraph capture 组合的演进线; 本 PR 的 `_enable_csa_multi_stream` 用 `is_current_stream_capturing()` 与 `metadata` 类型判断 capture 边界, 与 DBO 的 fig capture / 回放策略属于同一技术脉络, 双方对 capture 边

界的理解会互相影响。

- PR #56741 [Refactor] Normalize DeepSeek V4.1 model package naming: 同属 vllm/models/deepseek_v4 包的重构与注册线，模型包路径 / 命名变更会直接移动本 PR 改动的文件位置，是需要关注的落地顺序与 rebase 风险来源。
- PR #56773 [Bugfix][CPU] Fix DeepSeek-R1 (FP8 MLA + MoE) correctness on CPU backend: 同为 DeepSeek MLA 系列在多后端上的正确性 / 契约修正，本 PR 改动了 vllm/models/deepseek_v4/cpu/cpu_sparse.py 的 indexer 前向签名，与 CPU 侧 MLA 路径维护存在直接交集。
- PR #54965 [ROCm][Perf] W4A16: keep skinny GEMM zero-points packed 4-bit: 同属 ROCm 平台性能优化的连续投入（量化 kernel + 本 PR 的流级并行），反映 ROCm 侧正在系统性补齐推理性能，本 PR 是该方向在注意力编排层面的延伸。