

PR #56893 完整报告

vllm-project/vllm

[Model][DSv4.1] Store the whole KV in MXFP8 (FlashMLA V4.1 record)

合并时间: 2026-09-15 11:12

原文链接: <http://prhub.com.cn/vllm-project/vllm/pull/56893>

执行摘要

- 一句话: DSv4.1 在 SM100 改用 FlashMLA 528 B MXFP8 KV 记录, KV 缩小 9.6%
- 推荐动作: 值得精读, 尤其是维护 KV cache 布局、attention 后端与跨内核一致性的人。
- 1. 重点看 vllm/models/deepseek_v41/attention.py 里 `_use_v41_mxfp8_kv_record()` 与 `kv_bytes_per_token / kv_page_alignment` 的绑定关系: 把「记录宽度」和「页对齐」变成一组由单一 gate 派生的字段, 是这次能在多处复用而不漂移的关键设计。
- 2. 看 `k_cache.shape[-1]` 作为记录标识的做法: 用已经发布的 KV cache spec 宽度做隐式契约, 好处是 V4 路径零改动地跑在同一条代码路径上, 代价是魔法数字散落在分派点与测试里——这正是 reviewer 要重构的点。
- 3. 看索引器对齐这个「每个 spec 单独看都正确、合起来 block stride 不整除」的排障故事: 它解释了为什么 `alignment=576 if ... and not _use_v41_mxfp8_kv_record() else 512` 必须存在, 也提醒后续给同一 block 组加页类型的改动要先算总和。
- 4. 若要跟进: 优先按 reviewer 建议把 576/512/528/584 收敛成记录描述对象; 在合入 #56344 时确认索引器对齐修复被带走; 并为 SM100 上非 FlashMLA sparse 后端的记录选择补一条显式断言或测试。

功能与动机

PR body 指出: DeepSeek-V4.1 一直在复用 DeepSeek-V4 的分页 `fp8_ds_mla` 记录, 而 FlashMLA 有一个 V4.1 专用记录, 会把 RoPE 维一起量化。切换后 data row 从「448 B fp8 NoPE + 128 B bf16 RoPE」变成「512 B fp8 e4m3 全量化」, scale row 从「7 个 64 维 UE8M0 + 1 pad」变成「16 个 32 维 UE8M0 (MXFP8)」, 每 token 从 584 B 降到 528 B (-9.6%), 页 TMA stride 从 576 B 降到 512 B; 由于 block_size 恒为 32 的倍数, V4.1 每页正好是 512 的整数倍, V4 需要的 padding 也一并消失。作者强调「No new cache dtype and no new flag」, `--kv-cache-dtype fp8` 仍解析为 `fp8_ds_mla`, 记录随架构自动选择。

实现拆解

1. 记录门控与页几何 (`vllm/models/deepseek_v41/attention.py`): 新增 `_use_v41_mxfp8_kv_record()`, 用 `current_platform.is_device_capability_family(100)` 判断是否启用 FlashMLA 的 V4.1 记录; `DeepseekV4Attention.__init__` 由此派生 `self.kv_mxfp8`、`self.kv_bytes_per_token` (528/584)、`self.kv_page_alignment` (

512/576)，并把后两者注入 `DeepseekV4SWACache`。同一个 spec 里 `state_content_bytes` 和 `alignment` 改为从这两个字段取值，`_fused_qnorm_rope_kv_insert` 调 op 时多传一个 `self.kv_mxfp8`。效果是「记录宽度」和「页对齐」被绑定成一组，不再在 spec 与内核里各写一份常量。

- 索引器页对齐同步（同文件 `DeepseekV4IndexerCache.get_kv_cache_spec`）：
`alignment` 从固定 576 if `uses_fp8_ds_mla_layout` else 512 改为 576 if `uses_fp8_ds_mla_layout` and not `_use_v41_mxfp8_kv_record()` else 512。原因是 `_get_bytes_per_block` 把组内每个页的 stride 相加得到块 stride，只有所有页 padding 一致时整除才成立；PR body 记录了保持 576 会产生 212160 的块 stride，在干净加载权重后的启动 profile 阶段触发 FlashMLA 的 `block_stride_bytes % TMA_K_STRIDE == 0` 断言，而「每个 spec 单独看都正确」。
- 写入端两套内核：SWA 路径给 `torch.ops._C.fused_deepseek_v4_qnorm_rope_kv_rope_quant_insert` 增加 `kv_mxfp8` 参数，`csrc/libtorch_stable/fused_deepseek_v4_qnorm_rope_kv_insert_kernel.cu` 与 `ops.h / torch_bindings.cpp` 同步签名，MXFP8 分支改成每 lane pair 一个 UE8M0 scale、RoPE lane 参与量化；compressed-KV 路径在 `vllm/models/deepseek_v41/common/ops/fused_compress_quant_cache.py` 新增 `_rope_quant_insert_mxfp8_kernel`，`rope_quant_insert` 按 `kv_cache.shape[-1]` in (584, 528) 选内核，MXFP8 版本先做 RoPE 再选 scale（RoPE 维这次也要量化）、只在压缩组边界写入。`vllm/models/deepseek_v41/{amd,nvidia}/dspark.py` 的 `_insert_context_kv` 把 `attn.kv_mxfp8` 透传给同一个 op。
- 读取端参数化记录：`vllm/models/deepseek_v41/common/ops/cache_utils.py` 新增 `_quantize_and_insert_k_mxfp8_kernel` 与 `_dequantize_and_gather_k_mxfp8_kernel`，入口用 `bytes_per_token` 参数与 `k_cache.shape[-1]` 分派，常量集中在模块头部（`V4_BYTES_PER_TOKEN=584`、`V41_BYTES_PER_TOKEN=528`、`V41_QUANT_BLOCK=32`、`V41_NUM_SCALES=16`）；
`vllm/models/deepseek_v4/nvidia/ops/dequant_gather_k_cutedsl.py` 把 `fp8_dim / group_size` 从类常量改成 `CompileKey` 字段，`scale_bytes = head_dim // group_size`、bf16 tail 用 `bf16_dim > 0` 守卫，使同一份 CuteDSL 代码同时支持 448/64 与 512/32；
`vllm/v1/attention/backends/mla/flashmla_sparse.py` 补写 V4.1 记录格式与「页内先 data 区、后 scale 区」的文档。
- 构建与测试配套：`cmake/external_projects/flashmla.cmake` 把 pin 换到 `vllm-project/FlashMLA@b6ef01c`，跟随 fork 目录重组（`csrc/sm* → csrc/kernels/sm*`）并补齐 `api.cpp` 注册的 `fused_norm_rope_attn_rope_cast_*` 与两个 permute kernel 实例化（镜像 fork 的 `setup.py`，去掉训练用 dense backward）；commit `f5f74909` 再 pin 一次以修 nvcc 13.0.88 在 SM100 pass 上实例化被丢弃 else 分支里 `static_assert` 的构建失败（FlashMLA#24 改为 `else if constexpr`，SASS 不变）。测试侧在 `tests/kernels/test_compressor_kv_cache.py` 新增 `_mxfp8_record_reference` 参考量化器、`test_v41_rope_insert_mxfp8_record`（byte-exact，覆盖负 slot 与页 padding）、`test_v41_mxfp8_cache_round_trip`（insert→gather 往返并按 tile 尺度给误差上界），在 `tests/kernels/test_fused_deepseek_v4_qnorm_rope_kv_insert.py` 新增 `test_kv_path_mxfp8_record_matches_reference` 拿 CUDA 融合 op 对比 Triton 编码器并断言 NoPE 维 byte-identical。

关键文件：

- `vllm/models/deepseek_v41/attention.py`（模块 注意力层；类别 source；类型 data-contract；符号 `_use_v41_mxfp8_kv_record`, `DeepseekV4Attention.init`, `DeepseekV4Attention._fused_qnorm_rope_kv_insert`, `DeepseekV4Attention.get_kv_cache_spec`）：本次记录切换的唯一门控与页几何来源：新增 `_use_v41_mxfp8_kv_record()` 与 `kv_mxfp8 / kv_bytes_per_token / kv_page_alignment`，注入 SWA 缓存、透传给融合 insert、改写压缩 KV spec，并携带索引器页对齐修复。
- `vllm/models/deepseek_v41/common/ops/cache_utils.py`（模块 缓存算子；类别 source；类型 core-logic；符号 `V4_BYTES_PER_TOKEN`, `V41_BYTES_PER_TOKEN`, `V41_QUANT_BLOCK`, `V41_NUM_SCALES`）：读取 / 写入记录的核心分派点：新增 `_quantize_and_insert_k_mxfp8_kernel` 与 `_dequantize_and_gather_k_mxfp8_kernel` 两个 Triton 内核，并用 `k_cache.shape[-1] / bytes_per_token` 在 V4 与 V4.1 记录之间选路，是本 PR 最大的源码改动（+186/-8）。
- `vllm/models/deepseek_v41/common/ops/fused_compress_quant_cache.py`（模块 融合算子；类别 source；类型 core-logic；符号 `_rope_quant_insert_mxfp8_kernel`, `rope_quant_insert`）：压缩 KV 缓存的写入端：新增 `_rope_quant_insert_mxfp8_kernel`, `rope_quant_insert` 按 cache 宽度 584/528 选内核；MXFP8 内核必须「先旋转再定 scale」，是本次量化语义变化最直接的体现。
- `csrc/libtorch_stable/fused_deepseek_v4_qnorm_rope_kv_insert_kernel.cu`（模块 算子内核；类别 source；类型 core-logic；符号 `fused_deepseek_v4_qnorm_rope_kv_rope_quant_insert`）：SWA 融合插入 CUDA 内核新增 `kv_mxfp8` 分支：每 lane pair 一个 UE8M0 scale，RoPE lane 不再存 bf16 而参与量化；配套 `ops.h / torch_bindings.cpp` 同步 op 签名。
- `vllm/v1/attention/backends/mla/sparse_swa.py`（模块 滑窗缓存；类别 source；类型 data-contract；符号 `DeepseekV4SWACache.init`, `DeepseekV4SWACache.get_kv_cache_spec`）：`DeepseekV4SWACache` 新增 `packed_bytes_per_token / packed_page_alignment` 参数，spec 的 `state_content_bytes` 与 `alignment` 改由参数驱动；reviewer 的行内评论正是落在这个文件的 `alignment` 行。
- `tests/kernels/test_compressor_kv_cache.py`（模块 缓存测试；类别 test；类型 test-coverage；符号 `_mxfp8_record_reference`, `test_v41_rope_insert_mxfp8_record`, `test_v41_mxfp8_cache_round_trip`）：新增 `_mxfp8_record_reference` 参考量化器（对齐 FlashMLA 的 `KVCacheLayout.V41_FP8Sparse`）、byte-exact 的 `test_v41_rope_insert_mxfp8_record` 与 `insert→gather` 的 `test_v41_mxfp8_cache_round_trip`，把新记录锁在字节级。
- `tests/kernels/test_fused_deepseek_v4_qnorm_rope_kv_insert.py`（模块 算子测试；类别 test；类型 test-coverage；符号 `test_kv_path_mxfp8_record_matches_reference`, `_dequant`, `V41_HEAD_BYTES`）：新增 `test_kv_path_mxfp8_record_matches_reference`，让 CUDA 融合 op 的 `kv_mxfp8=True` 分支与 Triton 编码器对齐，并断言 NoPE 维 byte-identical、整行误差不超过一个 tile 尺度步长。

- `vllm/models/deepseek_v4/nvidia/ops/dequant_gather_k_cutedsl.py` (模块 解量化内核; 类别 source; 类型 infrastructure; 符号 `DequantGatherKCacheKernel.CompileKey`, `DequantGatherKCacheKernel.host_entrypoint`, `DequantGatherKCacheKernel.dispatch`, `DequantGatherKCacheKernel.call`) : CuteDSL gather 从「硬编码 448/64」改为 `CompileKey` 携带 `fp8_dim / group_size`, `scale_bytes` 与 `bf16 tail` 分支都由记录参数推导, 是读取端同时支持 584 B 与 528 B 的关键。
- `vllm/v1/attention/backends/mla/flashmla_sparse.py` (模块 解码后端; 类别 source; 类型 documentation; 符号 `QUANTIZED_DS_MLA_CACHE_FORMATS`) : 模块顶部格式说明补齐 V4.1 528 B 记录 (512 B 全 fp8 + 16 个 UE8M0 scale) 以及「页内 data 区在前, scale 区在后, 页大小按 TMA stride 上取整」的关键说明, 是理解本次布局变更的最佳文档入口。
- `cmake/external_projects/flashmla.cmake` (模块 构建配置; 类别 infra; 类型 configuration; 符号 `FlashMLA_SOURCE_DIR`, `FLASH_MLA_SOURCES`) : FlashMLA pin 换到 `vllm-project/FlashMLA@b6ef01c`, 跟随 fork 目录重组并补齐 `api.cpp` 新增内核的实例化; 后续又为 `nvcc 13.0` 的 `static_assert` 构建失败再 pin 一次, 是本次改动的构建前提。
- `vllm/models/deepseek_v41/nvidia/dspark.py` (模块 上下文插入; 类别 source; 类型 data-contract; 符号 `_insert_context_kv`) : `_insert_context_kv` 把 `attn.kv_mxfp8` 与 `apply_q_norm=True` 透传给融合 op, 保证 DSpark 的纯 KV context 插入与主 attention 写入同一条记录; `amd/dspark.py` 有对称改动。
- `csrc/libtorch_stable/ops.h` (模块 算子注册; 类别 source; 类型 data-contract; 符号 `fused_deepseek_v4_qnorm_rope_kv_rope_quant_insert`) : 融合 q/kv insert 的 C++ 签名新增 `kv_mxfp8` 布尔参数 (`torch_bindings.cpp` 同步注册), 是 CUDA 内核与 Python 调用方的接口契约。

关键符号: `_use_v41_mxfp8_kv_record`, `DeepseekV4Attention.init`, `DeepseekV4Attention._fused_qnorm_rope_kv_insert`, `DeepseekV4Attention.get_kv_cache_spec`, `DeepseekV4IndexerCache.get_kv_cache_spec`, `DeepseekV4SWACache.init`, `DeepseekV4SWACache.get_kv_cache_spec`, `_quantize_and_insert_k_mxfp8_kernel`, `_dequantize_and_gather_k_mxfp8_kernel`, `quantize_and_insert_k_cache`, `dequantize_and_gather_k_cache_triton`, `_rope_quant_insert_mxfp8_kernel`, `rope_quant_insert`, `DequantGatherKCacheKernel.dispatch`, `DequantGatherKCacheKernel.host_entrypoint`, `_mxfp8_record_reference`, `test_v41_rope_insert_mxfp8_record`, `test_v41_mxfp8_cache_round_trip`, `test_kv_path_mxfp8_record_matches_reference`

关键源码片段

`vllm/models/deepseek_v41/attention.py`

本次记录切换的唯一门控与页几何来源: 新增 `_use_v41_mxfp8_kv_record()` 与 `kv_mxfp8 / kv_bytes_per_token / kv_page_alignment`, 注入 SWA 缓存、透传给融合 insert、改写压缩 KV spec, 并携带索引器页对齐修复。

```

# vllm/models/deepseek_v41/attention.py
# V4.1 的 MXFP8 记录只在 SM100 的 FlashMLA 稀疏解码内核里可用：
# 记录宽度、页对齐、RoPE 是否量化，全部由这一个 gate 决定。
def _use_v41_mxfp8_kv_record() -> bool:
    return current_platform.is_device_capability_family(100)

class DeepseekV4Attention(...):
    def __init__(self, ...):
        ...
        # 三个量必须成组变化：528 B/512 B 页 (V4.1) 或 584 B/576 B 页 (V4) 。
        # 单独改任何一个都会让页 stride 与解码内核的 TMA stride 不再整除。
        self.kv_mxfp8 = _use_v41_mxfp8_kv_record()
        self.kv_bytes_per_token = 528 if self.kv_mxfp8 else 584
        self.kv_page_alignment = 512 if self.kv_mxfp8 else 576

        self.swa_cache_layer = DeepseekV4SWACache(
            head_dim=self.head_dim,
            window_size=self.window_size,
            dtype=self.kv_cache_torch_dtype,
            prefix=f"{prefix}.swa_cache",
            cache_config=cache_config,
            backend_cls=self.swa_backend_cls,
            block_size=32,
            packed_bytes_per_token=self.kv_bytes_per_token,
            packed_page_alignment=self.kv_page_alignment,
        )

    def _fused_qnorm_rope_kv_insert(self, q, kv, positions):
        # uint8 是 fp8_ds_mla 分页路径；最后两个参数告诉 CUDA 内核写哪条记录。
        if swa_kv_cache.dtype == torch.uint8:
            swa_kv_cache_2d = swa_kv_cache.view(swa_kv_cache.shape[0], -1)
            return torch.ops._C.fused_deepseek_v4_qnorm_rope_kv_rope_quant_insert(
                q,
                kv,
                swa_kv_cache_2d,
                swa_metadata.slot_mapping,
                positions,
                cos_sin_cache,
                self.padded_heads,
                self.eps,
                swa_metadata.block_size,
                False,
                self.kv_mxfp8, # True 时 RoPE lane 也量化，scale 密度翻倍
            )

    def get_kv_cache_spec(self, vllm_config):
        if not self.is_kv_source:
            return None

```

```

uses_fp8_ds_mla_layout = self.kv_cache_dtype == "fp8_ds_mla"
return MLAAttentionSpec(
    block_size=vllm_config.cache_config.block_size,
    num_kv_heads=1,
    head_size=self.head_dim,
    dtype=torch.uint8 if uses_fp8_ds_mla_layout else self.kv_cache_torch_dtype,
    tokens_per_state=self.compress_ratio,
    cache_dtype_str=self.kv_cache_dtype,
    # 页对齐随记录切换，索引器页必须与之同步（见下方 IndexerCache）。
    alignment=self.kv_page_alignment if uses_fp8_ds_mla_layout else 512,
    model_version="deepseek_v4",
    kv_quant_mode=get_kv_quant_mode(self.kv_cache_dtype),
    # 打包后每 token 的字节宽度；head_size 仍是语义上的 512。
    state_content_bytes=(
        self.kv_bytes_per_token if uses_fp8_ds_mla_layout else None
    ),
)

```

```

class DeepseekV4IndexerCache(torch.nn.Module, AttentionLayerBase):
    def get_kv_cache_spec(self, vllm_config):
        # 索引器页与主 KV 页共用同一个物理 block，block stride 是组内各页之和，
        # 所以索引器必须跟着主缓存一起从 576 B 迁到 512 B，否则启动 profile
        # 阶段会触发 FlashMLA 的 block_stride_bytes % TMA_K_STRIDE 断言。
        return MLAAttentionSpec(
            ...,
            alignment=(
                576
                if uses_fp8_ds_mla_layout and not _use_v41_mxfp8_kv_record()
                else 512
            ),
        )

```

vllm/models/deepseek_v41/common/ops/fused_compress_quant_cache.py

压缩 KV 缓存的写入端：新增 `_rope_quant_insert_mxfp8_kernel`，`rope_quant_insert` 按 cache 宽度 584/528 选内核；MXFP8 内核必须「先旋转再定 scale」，是本次量化语义变化最直接的体现。

```

# vllm/models/deepseek_v41/common/ops/fused_compress_quant_cache.py
# uint8 分页缓存有两种记录宽度，宽度即选择器：528 -> V4.1 MXFP8 内核。
if kv_cache.dtype == torch.uint8:
    assert kv_cache.shape[-1] in (584, 528), (
        f"unsupported paged KV record width {kv_cache.shape[-1]}"
    )
    kernel = (
        _rope_quant_insert_mxfp8_kernel
        if kv_cache.shape[-1] == 528
        else _rope_quant_insert_kernel
    )

```

```

kernel[(num_tokens,)](
    latent, positions, cos_sin_cache, kv_cache, cache_slots, ...
)

@triton.jit
def _rope_quant_insert_mxfp8_kernel(
    latent, positions, cos_sin, cache, cache_slots,
    COS_STRIDE: tl.constexpr,
    CACHE_STRIDE: tl.constexpr,
    CACHE_BLOCK: tl.constexpr,
    COMPRESS_RATIO: tl.constexpr,
    SANITIZE_CACHE_NANS: tl.constexpr,
):
    """V4.1 记录：先旋转，再对全部 512 维做 MXFP8 量化。

    RoPE 维在这里也被量化，所以必须「先旋转、后定 scale」；
    V4 内核则是先量化 NoPE，再把 RoPE 以 bf16 存下来。
    """
    t = tl.program_id(0)
    slot = tl.load(cache_slots + t)
    if slot < 0:
        return
    position = tl.load(positions + t)
    if (position + 1) % COMPRESS_RATIO != 0: # 只在压缩组边界落盘
        return

    d = tl.arange(0, 512)
    normed = tl.load(latent + t.to(tl.int64) * 512 + d).to(tl.float32)

    # NoPE 的 pair 取 (cos, sin) = (1, 0)，旋转即恒等；
    # 只有 pair >= 0 的尾部 64 维才真正参与旋转。
    even, odd = tl.split(tl.reshape(normed, (256, 2)))
    pair = tl.arange(0, 256) - 224
    cs = cos_sin + (position // COMPRESS_RATIO * COMPRESS_RATIO) * COS_STRIDE
    c = tl.load(cs + tl.maximum(pair, 0), pair >= 0, other=1.0).to(tl.float32)
    s = tl.load(cs + 32 + tl.maximum(pair, 0), pair >= 0, other=0.0).to(tl.float32)
    rotated = tl.interleave(even * c - odd * s, odd * c + even * s)
    if SANITIZE_CACHE_NANS:
        rotated = tl.where(rotated == rotated, rotated, 0.0)

    page = cache + (slot // CACHE_BLOCK).to(tl.int64) * CACHE_STRIDE
    values = page + (slot % CACHE_BLOCK) * 512
    scales = page + CACHE_BLOCK * 512 + (slot % CACHE_BLOCK) * 16

    quant = tl.reshape(rotated, (16, 32))
    amax = tl.maximum(tl.max(tl.abs(quant), 1), 1e-4)
    exponent = tl.ceil(tl.log2(amax * (1.0 / 448.0)))
    scaled = quant * tl.reshape(tl.exp2(-exponent), (16, 1))

```

```
fp8 = tl.clamp(scaled, -448.0, 448.0).to(tl.float8e4nv)
tl.store(values + d, tl.reshape(fp8.to(tl.uint8, bitcast=True), (512,)))

# NaN 清理模式下把指数码点压到 254, 避免 scale 值被解释成 NaN。
max_encoded: tl.constexpr = 254.0 if SANITIZE_CACHE_NANS else 255.0
encoded = tl.minimum(tl.maximum(exponent + 127.0, 0.0), max_encoded)
tl.store(scales + tl.arange(0, 16), encoded.to(tl.uint8))
```

评论区精华

1. 对齐魔数是否该收敛 (ivanium, [vllm/v1/attention/backends/mla/sparse_swa.py:127](#))
: 行内评论指出「given that we now have self.packed_page_alignment, putting 512 here looks a bit hacky」——既然新增了 packed_page_alignment 参数, else 512 的兜底仍写死数字显得突兀。在整体 review 中进一步表态: 「KV cache part LGTM. I'd prefer we can later refactor the alignment part to avoid too much hardcoded numbers like 576 and 512」。结论: 本 PR 内未重构, reviewer 明确把常量收敛挂到后续工作, 并以 APPROVED 放行。
 2. 自动 review 被禁用: claude[bot] 说明这是 fork 提交的 PR, 自动 review 关闭, 需要 maintainer 手动 @claude review——本 PR 实际只有一名人类 reviewer 过目, 而改动横跨 CUDA/Triton/CuteDSL 三套内核。
 3. CI 往返与分支新鲜度: 作者连续四次 /ci run, 其中一次被拒绝: 「This PR is 9 commits behind upstream main. Your branch must contain every commit currently on upstream main.」需要先 rebase 才能重跑, 最终 38465d553f40 触发 Buildkite #88975。
- 对齐常量该不该收敛 (packed_page_alignment 旁边仍写死 512) (design): 作者未在本 PR 内重构, reviewer 以「later」措辞接受现状并给出 APPROVED, 把常量收敛记为后续工作。
 - fork PR 的自动 review 被禁用 (other): 没有人手动触发, PR 实际只有 ivanium 一名人类 reviewer 覆盖了横跨 CUDA/Triton/CuteDSL 的改动。
 - CI 分支新鲜度校验拦下一次 /ci run (other): 需要先 rebase 到最新 main 再重跑, 最终 commit 38465d553f40 成功触发 Buildkite #88975。
 - FlashMLA pin 在 nvcc 13.0 上的构建失败与二次 pin (other): 通过再 pin 解决, 记录与数值不变; 顺带纳入 FlashMLA#23 的可选 FP8 输出缓冲 (vLLM 暂未调用)。

风险与影响

- 风险:
 1. RoPE 量化精度: 64 维 RoPE 从 bf16 改为 fp8 e4m3 + UE8M0 scale, 是本次唯一涉及数值行为的改动。作者 gsm8k (温度 0, n=1319) 从 0.9272/0.9310 提升到 0.9333/0.9340, 但 GPQA 基线用的是 deepseekv41-flash-0909 镜像且 epoch_num: 1, 本 PR 是本地构建且 epoch_num: 4, 作者自己也标注「the arms differ in image and epoch count as well as in the record」, 因此长上下文、低温度下的 RoPE 量化影响缺少干净的专项评测。

2. 门控粒度只到架构: `_use_v41_mxfp8_kv_record()` 只看 `is_device_capability_family(100)`, 与 `backend_cls` 无关。PR body 只声明 SM90、ROCm 与 FlashInfer SM120 保留 V4; 若在 SM100 上落到非 FlashMLA 的 sparse 后端, 528 B 记录能否被对端正确解码没有在本 PR 中验证。
3. 记录宽度成为隐式契约: 分派依赖 `kv_cache.shape[-1]` in (584, 528) 与 `k_cache.shape[-1] == 528`, 测试里同样硬编码 584/528。一旦出现第三种记录宽度, 或 `block_size` 不再是 32 的倍数 (`math.ceil(block_size * 528 / 512) * 512` 的页几何前提被破坏), 会走错分支或直接触发断言。
4. 页对齐耦合且报错距离根因远: 索引器与主 KV 共享同一物理 block, alignment 分别在 `DeepseekV4Attention.get_kv_cache_spec` 和 `DeepseekV4IndexerCache.get_kv_cache_spec` 两处计算 (后者还重新调用 `gate`), 任何一页 padding 不一致只会在启动 profile 阶段以 FlashMLA 的 TMA 整除断言暴露, 属于难定位的启动期故障。
5. 外部依赖与构建: pin 到 fork 的 b6ef01c 已在 CI 的 CUDA 13.0.3 上暴露 nvcc 13.0.88 实例化被丢弃 else 分支 `static_assert` 的构建失败, 只能靠再 pin (f5f74909) 绕过; flashmla.cmake 的源码清单是手工镜像 fork 的 setup.py, fork 再调整目录结构时容易漏编内核导致链接错误。
6. 在飞 PR 冲突: #56344 改同一份 flashmla.cmake 与 V4.1 writer/reader (且其 pin c112cc1 与本 PR 只差一个提交), #56441 改同样三个 `csrc/libtorch_stable/` 文件并会在 op 签名与绑定上文本冲突; PR body 明确指出 #56344 没带索引器对齐修复, 需要人工带过去。

• 影响:

1. 用户与推理效果: SM100 (GB300/B200) 上的 DeepSeek-V4.1 每 token KV 从 584 B 降到 528 B, 且页不再需要 576 B padding, 等效可缓存 token 数增加; 作者实测启动路径分配 161.73 GiB、11,589,848 token, gsm8k 无退化。使用方式完全不变: `--kv-cache-dtype fp8` 仍解析成 `fp8_ds_mla`, 不需要新 flag 或新 dtype。非 SM100 平台 (SM90、ROCm、FlashInfer SM120) 行为与之前逐字节一致。
2. 系统与契约: KV cache 布局是跨 CUDA 融合 op、Triton 编码 / 解码、CuteDSL gather、KV cache spec 与索引器对齐的共同契约, 本 PR 让同一份代码同时理解 V4 (584 B / 576 B 页) 与 V4.1 (528 B / 512 B 页) 两种记录。KV cache 不持久化, 因此不存在 checkpoint 迁移问题; 但任何假设「V4.1 行宽 584 B」的外部工具或下游脚本会失效。
3. 团队维护成本: 三套内核都要为 MXFP8 记录单独维护一条路径, 576 / 512 / 584 / 528 分散在 `attention.py`、`sparse_swa.py`、`flashmla_sparse.py`、`cache_utils.py` 与两个测试文件中, reviewer 已明确要求后续收敛; 同时与 #56344、#56441 的 rebase 协调需要作者持续跟进。- 风险标记: KV 布局契约变更, RoPE 量化精度风险, 页对齐魔数分散, 三套内核需同步, 外部依赖 pin 到 fork, 门控只看架构不看后端

关联脉络

- PR #56344 [Perf] Deepseek V41 Attention MegaKernel: 同作者、同一条 V4.1 FlashMLA 记录线, 集成 FlashMLA PR#221 的融合 megakernel 与 `nvfp4_ds_mla`; 与本

PR 在 flashmla.cmake (pin c112cc1 vs b6ef01c) 和 V4.1 writer/reader 上重叠, PRbody 明确要求两者互相 rebase, 并指出 #56344 未带索引器对齐修复。

- PR #56441 [Perf][DSpark] Add KV-only context insertion across V4.1 cache formats: 改动同样三个 csrc/libtorch_stable/ 文件以增加 KV-only 包装 op, 会在 op 签名与 torch 绑定上与本 PR 文本冲突; 意图不重叠 (它去掉 dummy query, 本 PR 增加记录选择器)。
- PR #56169 [CI] Check target branch freshness before starting CI: 本 PR 的一次 /ci run 正是被该 PR 引入的分支新鲜度检查拒绝 (提示落后 upstream main 9 个 commit), 是这条 CI 策略上线后的真实命中案例。
- PR #56560 [ROCm][DSv4.1][Perf] Dequantize the MXFP8 weight once when dot_scaled cannot be used: 同为 DeepSeek-V4.1 模型线上的 MXFP8 优化, 但方向在权重侧 (ROCm、非对齐 K 时加载期一次解量化), 与本 PR 的 KV 侧记录切换正交, 可对照理解 DSv4.1 的量化策略分工。
- PR #56513 [ROCm][DSV4.1][Perf] Fold the mHC post step into the delayed pre projection: 同一 DeepSeek-V4.1 模型的性能优化序列, 涉及 vllm/models/deepseek_v41/ 与 _aiter_ops.py; 本 PR 的门控与页几何改动会影响该模型在 SM100 上的缓存配置, 属于同一演进脉络。