PR #27455 完整报告

sgl-project/sglang

[SM120] Add FlashInfer sparse MLA decode for DSv4-Flash

合并时间：2026-06-30 07:27

原文链接：http://prhub.com.cn/sgl-project/sglang/pull/27455

执行摘要

- 一句话：集成 FlashInfer SM120 稀疏 MLA 解码，decode 提升 2.2-3.7x
- 推荐动作：值得精读，尤其关注 _page_split_kernel 的 fused layout 转换设计和 FlashInfer 集成的惰性缓冲区策略。设计决策中，索引重映射恒等移除和直接导入 FlashInfer（而非自动检测）是重要权衡。

功能与动机

The existing Triton FlashMLA decode kernel (merged in #24692) works but leaves performance on the table. FlashInfer's native SM120 decode_dsv4 kernel uses CUTLASS with block-scaled MXFP8 MMA, achieving 2.2-3.7x decode speedup.

实现拆解

1. Page-split 融合内核：在 flash_mla_sm120.py 中新增 _page_split_kernel Triton JIT kernel，将 SGLang 256-token page（footer 布局）单次 launch 转换为 4 个 64-token 页面，消除原先 8 次独立拷贝（节省 344 launches/step）。配套 _split_kv_pages_to_64() 驱动函数管理惰性 per-device 缓冲区。
2. FlashInfer 解码分支：新增 _flash_mla_flashinfer() 函数，直接调用 FlashInfer 的 sparse_mla_sm120_decode_dsv4，传入分割后的 KV 页面；额外压缩 cache（C4/C128）保持原样传递。预分配 mid_out/mid_lse/output/out_lse 临时缓冲区。
3. 入口调度调整：修改 flash_mla_with_kvcache_sm120()，新增 FlashInfer 分支作为默认后端（当 _sm120_default_backend == "flashinfer" 时优先执行），Triton 和 PyTorch 分支保留为环境变量回退。
4. 环境变量注册：在 environ.py 的 Envs 类中添加 SGLANG_SM120_FLASHMLA_BACKEND = EnvStr("flashinfer")，统一使用 SGLang 的环境管理模块替代原有的 os.environ.get。
5. 测试配套：重命名测试文件为 test_flash_mla_backends.py，新增 test_flashinfer_backend_matches_triton 用例（mock backend 对比输出，atol=5e-2）；所有测试类添加 @unittest.skipUnless(_IS_SM120, ...) 守卫，确保非 SM120 设备跳过。
6. 清理：回退 deepseek_v4_backend.py 的不必要修改（该文件只读 swa_page_size，与 page-split 无直接依赖）。

关键文件：

- python/sglang/srt/layers/attention/flash_mla_sm120.py（模块 MLA 解码；类别 source；类型 core-logic；符号 _page_split_kernel, _split_kv_pages_to_64, _flash_mla_flashinfer）：核心实现文件，新增 page-split Triton kernel 和 FlashInfer 解码集成，修改入口调度逻辑。
- python/sglang/srt/environ.py（模块 配置层；类别 source；类型 configuration）：注册新环境变量 SGLANG_SM120_FLASHMLA_BACKEND，统一环境管理。
- test/registered/kernels/test_flash_mla_backends.py（模块 MLA 测试；类别 test；类型 test-coverage；符号 test_flashinfer_backend_matches_triton）：测试文件重命名并新增 FlashInfer vs Triton 对比测试，所有测试类添加 SM120 守卫。

关键符号：flash_mla_with_kvcache_sm120, _page_split_kernel, _split_kv_pages_to_64, _flash_mla_flashinfer, test_flashinfer_backend_matches_triton

关键源码片段

python/sglang/srt/layers/attention/flash_mla_sm120.py

核心实现文件，新增 page-split Triton kernel 和 FlashInfer 解码集成，修改入口调度逻辑。

```
# FlashInfer expects page_block_size=64 with footer per 64-token page.
# SGLang uses page_size=256. We fuse the conversion in one Triton kernel.

@triton.jit
def _page_split_kernel(
    src_ptr, dst_ptr,
    N_pages,
    src_stride0: tl.constexpr,
    dst_stride0: tl.constexpr,
    DATA_PER_SUB: tl.constexpr, # 64 * 576 = 36864
    SCALE_PER_SUB: tl.constexpr, # 64 * 8 = 512
    SRC_SCALE_OFF: tl.constexpr, # 256 * 576 = 147456
    DST_SCALE_OFF: tl.constexpr, # 64 * 576 = 36864
    RATIO: tl.constexpr, # 4
    BLOCK_SIZE: tl.constexpr,
):
    # One program per source page: 1 page -> four 64-token sub-pages.
    pid = tl.program_id(0)
    src_page_base = src_ptr + pid * src_stride0
    # Loop over the 4 sub-pages in this source page
    for i in tl.static_range(0, RATIO):
        # Compute offset into destination for this sub-page
        dst_sub_base = dst_ptr + (pid * RATIO + i) * dst_stride0
        # Compute start byte of data chunk inside source page
        data_start = i * DATA_PER_SUB
        # Copy data (nope+rope): 64 tokens * 576 bytes
        for off in tl.static_range(0, DATA_PER_SUB, BLOCK_SIZE):
            tl.store(dst_sub_base + off,
                     tl.load(src_page_base + data_start + off),
                     mask=None)
```

```
        # Copy scales: at offset SRC_SCALE_OFF (end of data) in source,
        # at offset DST_SCALE_OFF in destination sub-page
        scale_start = i * SCALE_PER_SUB
        for off in tl.static_range(0, SCALE_PER_SUB, BLOCK_SIZE):
            tl.store(dst_sub_base + DST_SCALE_OFF + off,
                     tl.load(src_page_base + SRC_SCALE_OFF + scale_start + off),
                     mask=None)

def _flash_mla_flashinfer(q, k_cache, indices, topk_length, attn_sink,
                          head_dim_v, softmax_scale,
                          extra_k_cache, extra_indices, extra_topk_length):
    """FlashInfer SM120 sparse MLA decode with page-split for SWA cache."""
    # Convert page_size=256 pages to page_size=64 sub-pages
    k_split = _split_kv_pages_to_64(k_cache)
    # Indices need no remapping (linear token order preserved)
    idx = indices.squeeze(1) if indices.dim() == 3 else indices
    # Pre-allocate scratch buffers for FlashInfer (same shape reuse)
    out_lse_buf = torch.empty(B, H, dtype=torch.float32, device=q.device)
    mid_out_buf = torch.empty(B, H, head_dim_v, dtype=torch.bfloat16, device=q.device)
    mid_lse_buf = torch.empty(B, H, dtype=torch.float32, device=q.device)
    from flashinfer.sparse_mla_sm120 import sparse_mla_sm120_decode_dsv4
    sparse_mla_sm120_decode_dsv4(
        q, k_split, idx, topk_length, attn_sink,
        out_lse=out_lse_buf, mid_out=mid_out_buf, mid_lse=mid_lse_buf,
        extra_k_cache=extra_k_cache, extra_indices=extra_indices,
    )
    return mid_out_buf, out_lse_buf
```

评论区精华

- 后端选择策略（b8zhong）：FlashInfer 已 pinned，应直接导入而非 try-import；AliceChenyy 采纳，后续 commit 移除自动检测逻辑（7bdc051）。
- 索引重映射恒等移除（gemini-code-assist）：_remap_indices_to_64 数学上恒等于原索引（ratio * PBS_DST = src_pbs），AliceChenyy 确认后移除，消减一次冗余 GPU kernel。
- 全局 scratch buffer 优化（gemini-code-assist）：建议预分配输出缓冲区避免分配开销；AliceChenyy 回应 CUDA graph replay 下 PyTorch 缓存分配器已复用，暂无需预分配。
- envs 模块使用（b8zhong）：要求使用 envs 类而非 os.environ；AliceChenyy 在 e9b4eef 中切换为 envs.SGLANG_SM120_FLASHMLA_BACKEND.get()。
- 测试守卫与重命名（Fridge003）：要求回退 deepseek_v4_backend.py、重命名测试文件、添加 SM120 设备守卫；AliceChenyy 逐一落实。
- 默认后端选择：直接导入 FlashInfer vs try-import 自动检测 (design): 默认直接导入 FlashInfer，不再 try-import；环境变量回退 Triton/Torch。
- 索引重映射恒等函数移除 (correctness): 移除 _remap_indices_to_64，直接使用原始索引。
- 全局 scratch buffer 预分配以减少分配开销 (performance): 暂不预分配，保持 per-call 一次性分配；未来若 profiling 显示分配开销再做优化。

- 使用 envs 模块替代 os.environ (style): 统一使用 SGLang 的 Envs 类管理环境变量。
- 测试文件重命名与 SM120 设备守卫 (testing): 文件重命名，所有测试类添加 @unittest.skipUnless(_IS_SM120, ...)，新增 FlashInfer 验证测试。

风险与影响

- 风险：
  1. 硬件绑定：代码路径仅在 SM120（compute capability 12.0）上激活，CI 缺少 SM120 设备，测试被跳过，回归风险需物理机验证。
  2. 外部依赖：需 FlashInfer 0.6.13+ 支持 sparse_mla_sm120 模块；PyPI 0.6.13rc2 不包含该模块，必须从 GitHub main 安装；API 变更会直接影响。
  3. 页面分割正确性：_page_split_kernel 是 fused Triton kernel，若布局偏移常数错误可能导致数据损坏，精度测试已覆盖但场景有限。
  4. CUDAgraph兼容：CUDAgraph捕获已测试BS=1-16，但非graph路径（如prefill）仍有临时分配，虽无性能敏感。
  5. 降级路径：环境变量 SGLANG_SM120_FLASHMLA_BACKEND=tritonltorch 提供回退，但需确保 Triton 后端不受影响（零回归已验证）。
- 影响：
  - 用户：RTX PRO 6000 用户显著受益，TTFT 最高提升 6.7x，TPOT 最高 3.7x（ISL=8K, TP=4）；接口不变，通过环境变量可调试回退。
  - 系统：改动集中在 3 个文件 +261/-7 行；未修改 KV cache 池、压缩器或其他模块，代码隔离良好。
  - 团队：需维护 FlashInfer 和 Triton 两套 MLA 后端，但 Triton 作为回退可逐步淘汰；测试文件重命名需更新 CI 引用。
  - 风险标记：SM120-only，依赖 FlashInfer 版本，无 CI 覆盖，CUDA graph 路径

关联脉络

- 暂无明显关联 PR