PR #32109 完整报告

sgl-project/sglang

[Perf] Skip blocks past per-request live length in full-width Triton kernels

合并时间：2026-07-23 13:00

原文链接：http://prhub.com.cn/sgl-project/sglang/pull/32109

执行摘要

- 一句话：扩展 early-exit 模式至多个全宽 Triton 内核
- 推荐动作：值得精读：该 PR 展现了如何通过 per-request 早退显著优化 Triton 内核性能，以及如何严格定义和测试部分写入契约。特别关注 kv_len 与 req_idx 的流水线加载技巧，以及测试中 live_mask 的构造和应用。对于正在开发 Triton 内核的工程师有直接参考价值。

功能与动机

延续 #31981 的 per-request early-exit 模式，消除全宽 grid 中大量不必要的计算。因为这些内核的 grid 或内循环跨越静态最大上下文长度的全部范围，但实际请求的 kv 长度通常远小于该范围，导致每个解码步骤执行大量无效的加载与写入。

实现拆解

1. 在 python/sglang/kernels/ops/attention/metadata.py 的 _fused_metadata_kernel_general 和 _fused_metadata_kernel_ps1_no_swa 中：在列块循环前加载 seq_len，计算 num_live_pages = cdiv(seq_len + seq_len_delta, page_size)，若当前块起始列 col_start >= num_live_pages 则直接返回；同时将 mask 从 max_seq_pages 改为 num_live_pages。更新 normal_decode_set_metadata 的 docstring 阐明契约。
2. 在 python/sglang/kernels/ops/attention/dsa_metadata.py 的 _fused_dsa_decode_metadata_kernel 和 _fused_dsa_target_verify_metadata_kernel 中：在加载 req_idx 之后立即加载 kv_len（流水线化两个标量加载），若 col_block * BLOCK_N >= kv_len 则返回。更新 fused_dsa_decode_metadata 和 fused_dsa_target_verify_metadata 的 docstring。
3. 在 python/sglang/kernels/ops/kvcache/trtllm_mha_graph_metadata.py 的 update_trtllm_mha_graph_metadata_kernel 中：将内循环上限从 max_seq_pages 改为 num_live_pages = min(cdiv(seqlen, PAGE_SIZE), max_seq_pages)，并更新 update_trtllm_mha_graph_metadata 的 docstring。
4. 在 python/sglang/kernels/ops/attention/extend_attention.py 的 extend_attention_fwd 和 extend_attention_fwd_unified 中添加类似守卫，针对 attention 计算块添加早退。
5. 在 python/sglang/kernels/ops/attention/dsa/index_buf_accessor.py 的 _get_k_and_s_triton 和 python/sglang/kernels/ops/attention/dsa/transform_index.py

的 transform_index_page_table_prefill 中添加块级早退。

6. 测试配套：在 test_normal_decode_set_metadata.py 中新增 page_table_live_mask 辅助函数，将断言改为仅比较 page_table[live_mask] 和 swa_page_table[live_mask]。在 test_dsa_metadata.py 中构造 live_mask 用于 page_table_1 和 real_page_table 检查。在 test_trtllm_mha_graph_metadata.py 中将 page_table 和 swa_page_table 断言改为比较 live_mask。明确测试契约：只有活动前缀需正确，尾部保持未定义。

关键文件：

- python/sglang/kernels/ops/attention/metadata.py（模块 元数据内核；类别 infra；类型 core-logic；符号 _fused_metadata_kernel_general, _fused_metadata_kernel_ps1_no_swa, normal_decode_set_metadata）：核心内核文件，为 fused metadata kernel 添加了基于 seq_len 的早退守卫，跳过超过请求实时长度的页面列块，并更新了 docstring 定义 page table tail 契约。
- python/sglang/kernels/ops/attention/dsa_metadata.py（模块 DSA 元数据；类别 infra；类型 core-logic；符号 _fused_dsa_decode_metadata_kernel, _fused_dsa_target_verify_metadata_kernel, fused_dsa_decode_metadata, fused_dsa_target_verify_metadata）：在 DSA decode 和 target-verify 元数据内核中添加早退守卫，通过流水线化 kv_len 与 req_idx 加载来避免额外延迟。
- python/sglang/kernels/ops/kvcache/trtllm_mha_graph_metadata.py（模块 MHA 图元数据；类别 infra；类型 core-logic；符号 update_trtllm_mha_graph_metadata_kernel, update_trtllm_mha_graph_metadata）：在 TRTLLM MHA 图元数据内核的 for 循环中用 num_live_pages 替换 max_seq_pages 作为上限。
- python/sglang/kernels/ops/attention/extend_attention.py（模块 注意力扩展；类别 infra；类型 core-logic；符号 extend_attention_fwd, extend_attention_fwd_unified）：在 extend_attention_fwd 和 extend_attention_fwd_unified 中添加 attention 计算块的早退守卫。
- python/sglang/kernels/ops/attention/dsa/index_buf_accessor.py（模块 DSA 索引器；类别 infra；类型 core-logic；符号 _get_k_and_s_triton）：在 _get_k_and_s_triton 中添加早退守卫，跳过超过 kv_len 的列块。
- python/sglang/kernels/ops/attention/dsa/transform_index.py（模块 索引变换；类别 infra；类型 core-logic；符号 transform_index_page_table_prefill）：在 transform_index_page_table_prefill 中添加早退守卫。
- test/registered/attention/test_normal_decode_set_metadata.py（模块 测试覆盖；类别 test；类型 test-coverage；符号 page_table_live_mask, TestNormalDecodeSetMetadata）：新增 page_table_live_mask 辅助函数，并将断言改为仅比较 live mask 覆盖的活动前缀。
- test/registered/kernels/ops/attention/test_dsa_metadata.py（模块 测试覆盖；类别 test；类型 test-coverage；符号 _check_decode, _check_target_verify）：更新 _check_decode 和 _check_target_verify 中的断言，仅比较 live prefix，并添加注释解释尾部未定义。
- test/registered/attention/test_trtllm_mha_graph_metadata.py（模块 测试覆盖；类别 test；类型 test-coverage；符号 test_metadata_correctness）：将 page_table 和 swa_page_table 断言改为比较 live_mask，并更新注释说明内核自守卫行为。

关键符号：page_table_live_mask, _fused_metadata_kernel_general, _fused_metadata_kernel_ps1_no_swa, normal_decode_set_metadata, _fused_dsa_decode_metadata_kernel, _fused_dsa_target_verify_metadata_kernel, fused_dsa_decode_metadata, fused_dsa_target_verify_metadata, update_trtllm_mha_graph_metadata_kernel, update_trtllm_mha_graph_metadata, extend_attention_fwd, extend_attention_fwd_unified, _get_k_and_s_triton, transform_index_page_table_prefill

关键源码片段

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

核心内核文件，为 fused metadata kernel 添加了基于 seq_len 的早退守卫，跳过超过请求实时长度的页面列块，并更新了 docstring 定义 page table tail 契约。

```
@triton.jit
def _fused_metadata_kernel_general(
    # ... 参数略，读者可参看完整定义
):
    # 计算 pid_b, pid_c
    i = pid_b

    # 自守卫：基于设备端 seq_len 跳过超过请求实时长度的列块
    seq_len = tl.load(seq_lens + i * seq_lens_stride_0).to(tl.int32)
    if page_size == 1:
        num_live_pages = seq_len + seq_len_delta
    else:
        num_live_pages = (seq_len + seq_len_delta + (1 << SHIFT) - 1) >> SHIFT
    num_live_pages = tl.minimum(num_live_pages, max_seq_pages)
    col_start = pid_c * BLOCK_COLS
    if col_start >= num_live_pages:
        return # 此块完全超出活动区域，跳过

    # 加载行索引（同 block 内所有线程共享）
    row_idx = tl.load(req_pool_indices + i * req_pool_indices_stride_0)
    row_offset = row_idx * req_to_token_stride_0
    col_offsets = col_start + tl.arange(0, BLOCK_COLS)
    mask = col_offsets < num_live_pages # 仅限活页面范围
    # 后续计算 page_table / swa_page_table ...
```

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

在 DSA decode 和 target-verify 元数据内核中添加早退守卫，通过流水线化 kv_len 与 req_idx 加载来避免额外延迟。

```
    req_idx = tl.load(
        req_pool_indices + row * req_pool_indices_stride,
        mask=row < bs, other=0,
    )
    # 加载 kv_len，流水线化在 req_idx 之后（无额外延迟）
```

```
    kv_len = tl.load(
        seq_lens + row * seq_lens_stride,
        mask=row < bs, other=0,
    ).to(tl.int32)
    if col_block * BLOCK_N >= kv_len:
        return # 当前列块完全超出 kv 长度，跳过

    vals = tl.load(
        req_to_token + req_idx * req_to_token_stride_0 + offs_n * req_to_token_stride_1,
        mask=mask,
        other=0,
    )
    # 后续处理 ...
```

test/registered/attention/test_normal_decode_set_metadata.py

新增 page_table_live_mask 辅助函数，并将断言改为仅比较 live mask 覆盖的活动前缀。

```
def page_table_live_mask(
    seq_lens: torch.Tensor,
    seq_len_delta: int,
    page_size: int,
    max_seq_pages: int,
    width: int,
) -> torch.Tensor:
    # 每行 page table 的活动区域：fused kernel 自守卫于 seq_len，
    # 只写入 pages(seq_len + delta) 之前的列，其余保持旧值。
    live_pages = torch.clamp(
        (seq_lens + seq_len_delta + page_size - 1) // page_size, max=max_seq_pages
    )
    cols = torch.arange(width, device=seq_lens.device)
    return cols.view(1, -1) < live_pages.view(-1, 1)
```

评论区精华

BBuf 在 test_dsa_metadata.py 的 review 中评论：内核仅跳过完全超出 kv_len 的块，最后一个部分 live 块的末尾通道可能仍被写入（mask 仅限界到 max_len），因此只有完全不在范围内的块保证不变，部分块尾部是未定义的。作者通过更新测试仅比较 live_mask 的方式解决了该问题，并统一在多个测试模块中采用相同策略。决策：明确 page table tail 为未定义，消费者必须通过 cache_seqlens 限界读取。

- 定义 page table tail 契约 (correctness): 作者通过仅比较 live mask 的方式更新测试断言，明确尾部未定义，消费者不得读取超过 cache_seqlens 的位置。

风险与影响

- 风险：正确性风险：若 live_pages 计算错误（除零、溢出），可能错误跳过应写入的页面，导致 inference 错误。测试已覆盖多种 batch size 和 seq_len 组合，通过 live mask 对比确保活动前缀一致。尾部未定义写入：所有消费端注意力内核均通过 cache_seqlens 限界

读取，故安全；但未来新增内核若未遵守可能读取陈旧数据。性能风险：在 NOSKIP（每个块都 live）场景下守卫指令增加少量开销，benchmark 显示 <1% 退化且统计不显著。兼容性：无 schema 或 API 变更，仅内部优化。

- 影响：用户：*decode* 步骤延迟降低，尤其是大 batch (bs=64) 下 *metadata* 内核加速 30-34%，*extend attention* 不均匀批次加速 2.5-5.9x。系统：所有依赖这些内核的模型 (DSA、FA3、TRTLLM、标准 *attention*) 均受益。团队：需确保未来类似的全宽内核也遵循相同的 *early-exit* 契约模式，测试框架新增了 *live_mask* 比较模式可供复用。
- 风险标记：部分写入尾部未定义，*live pages* 计算正确性，NOSKIP 额外开销可忽略

关联脉络

- PR #31981 [Perf] Skip blocks past per-request live length in full-width Triton kernels (Draft *extend*): 此 PR 的基础：将 #31981 中 *draft-extend* 内核的 *early-exit* 模式扩展到其余全宽内核。