# PR #28085 完整报告

sgl-project/sglang

[PD] Optimize SWA allocation

合并时间：2026-06-16 02:01

原文链接：http://prhub.com.cn/sgl-project/sglang/pull/28085

## 执行摘要

- 一句话：优化 PD decode 场景 SWA 分配路径与内存
- 推荐动作：本 PR 属于中等重要度的重构 + 小优化，代码结构清晰，改动量小，推荐相关维护者精读 swa.py 中的分配重构逻辑，可作为同类代码清理的参考模式。

## 功能与动机

在 PD（Prefill/Decode）分离部署的解码路径中，SWA 混合分配器存在两处可优化点：一是容量检查与索引映射代码重复，二是解码预分配时未跟踪窗口外 KV 长度，导致无法释放窗口外内存。PR body 明确指出优化目标为“Allocation-path cleanup + a memory optimization for the SWA hybrid KV allocator”。

## 实现拆解

1. 提取公用方法 new_pages_available：在 python/sglang/srt/mem_cache/allocator/swa.py 中新增 new_pages_available(num_full_pages, num_swa_pages) 方法，统一封装 full 与 SWA 子分配器的可用容量检查，替代 alloc_extend 和 alloc_extend_swa_tail 中原有的两段重复判断。
2. 统 一映射写入入口：将 alloc、alloc_extend、alloc_extend_swa_tail 中分散的平台分支（NPU 与非 NPU）都替换为调用已有的 set_full_to_swa_mapping 方法，减少代码重复，也与 alloc 方法保持一致。
3. 传递 num_new_pages 给 SWA 子分配器：alloc_extend_swa_tail 原本未向子 alloc_extend 传递 num_new_pages，本 PR 将 num_full_pages 和 num_swa_pages 分别传入 full/SWA 子分配器，使行为与 alloc_extend 一致，避免子分配器重复计算。
4. 跟踪 swa_evicted_seqlen：在 python/sglang/srt/disaggregation/decode.py 的预分配路径中，当使用 alloc_extend_swa_tail 分配后，设置 req.swa_evicted_seqlen = fill_len - swa_tail_len，记录窗口外被丢弃的序列长度，使得后续解码可释放这部分 KV 内存。
5. 更新测试桩：在 test/registered/unit/mem_cache/test_swa_alloc_extend_page_estimation.py 的 _make_self 桩中添加了 new_pages_available 和 set_full_to_swa_mapping 方法，使测试能正确构造 allocator，保持现有回归测试通过。

关键文件：

- python/sglang/srt/mem_cache/allocator/swa.py（模块 内存分配器；类别 source；类型 core-logic；符号 new_pages_available, set_full_to_swa_mapping）：核心重构文件，提

取 new_pages_available 方法，统一映射写入入口，传递 num_new_pages 参数，是行为保持的清理和优化基础。

- test/registered/unit/mem_cache/test_swa_alloc_extend_page_estimation.py（模块 测试；类别 test；类型 test-coverage；符号 new_pages_available, set_full_to_swa_mapping）：测试配套更新，为桩对象添加 new_pages_available 和 set_full_to_swa_mapping 方法，确保重构后回归测试通过。
- python/sglang/srt/disaggregation/decode.py（模块 PD 解码；类别 source；类型 core-logic）：实际内存优化落地处，添加一条记录 swa_evicted_seqlen 的赋值语句，使解码路径可知窗口外可释放的 KV 长度。

关键符号：new_pages_available, set_full_to_swa_mapping, alloc_extend, alloc_extend_swa_tail, _pre_alloc

## 关键源码片段

### python/sglang/srt/mem_cache/allocator/swa.py

核心重构文件，提取 new_pages_available 方法，统一映射写入入口，传递 num_new_pages 参数，是行为保持的清理和优化基础。

```
    # 新增：统一容量检查方法，避免重复代码
    def new_pages_available(self, num_full_pages: int, num_swa_pages: int) -> bool:
        # 同时校验 full 和 SWA 子分配器是否有足够页面
        return (
            num_full_pages
            <= self.full_attn_allocator.available_size() // self.page_size
            and num_swa_pages
            <= self.swa_attn_allocator.available_size() // self.page_size
        )

    def alloc_extend(self, ...):
        # ...
        num_new_pages = get_num_new_pages(...)
        # 替换原来的两段式检查
        if not self.new_pages_available(num_new_pages, num_new_pages):
            return None
        # ...
        # 统一使用 set_full_to_swa_mapping
        self.set_full_to_swa_mapping(alloc_full_indices, alloc_swa_indices)
        return alloc_full_indices

    def alloc_extend_swa_tail(self, ...):
        # ...
        # 传递 num_new_pages 参数，与 alloc_extend 对齐
        alloc_full_indices = self.full_attn_allocator.alloc_extend(
            ...,
            num_new_pages=num_full_pages,
        )
        # ...
```

```
        alloc_swa_indices = self.swa_attn_allocator.alloc_extend(
            ...,
            num_new_pages=num_swa_pages,
        )
        # 统一使用 set_full_to_swa_mapping
        self.set_full_to_swa_mapping(
            alloc_full_indices[-swa_tail_len:], alloc_swa_indices
        )
```

## test/registered/unit/mem_cache/test_swa_alloc_extend_page_estimation.py

测试配套更新，为桩对象添加 new_pages_available 和 set_full_to_swa_mapping 方法，确保重构后回归测试通过。

```
  def _make_self(*, page_size: int, full_available: int, swa_available: int):
      full_indices = torch.tensor([10, 11], dtype=torch.int64)
      swa_indices = torch.tensor([20, 21], dtype=torch.int64)
      full_to_swa_index_mapping = torch.zeros(64, dtype=torch.int64)

      # 新增：模拟 new_pages_available 方法，行为与生产一致
      def new_pages_available(num_full_pages: int, num_swa_pages: int) -> bool:
          return (
              num_full_pages <= full_available // page_size
              and num_swa_pages <= swa_available // page_size
          )

      # 新增：模拟 set_full_to_swa_mapping 方法
      def set_full_to_swa_mapping(
          full_indices: torch.Tensor, swa_indices: torch.Tensor
      ) -> None:
          full_to_swa_index_mapping[full_indices] = swa_indices

      return SimpleNamespace(
          # ...
          new_pages_available=new_pages_available,
          set_full_to_swa_mapping=set_full_to_swa_mapping,
          full_to_swa_index_mapping=full_to_swa_index_mapping,
      )
```

## python/sglang/srt/disaggregation/decode.py

实际内存优化落地处，添加一条记录 swa_evicted_seqlen 的赋值语句，使解码路径可知窗口外可释放的 KV 长度。

```
            # 在 alloc_extend_swa_tail 成功后记录窗口外被丢弃的序列长度
            if self._uses_swa_tail_prealloc() and prefix_len == 0:
                kv_loc = self.token_to_kv_pool_allocator.alloc_extend_swa_tail(...)
                # 新增：记录 evicted seqlen，供后续释放使用
                req.swa_evicted_seqlen = fill_len - self._swa_tail_len(fill_len)
            else:
                # ...
```

## 评论区精华

PR 仅有 0 条 review 评论和 2 条低信号 issue 评论（插件配额告警和 CI rerun 指令），无技术讨论。merrymercy 直接批准，表明变更逻辑清晰、风险低。

- 暂无高价值评论线程

## 风险与影响

- 风险：
  1. 回归风险低：重构部分（new_pages_available、set_full_to_swa_mapping）行为等价，且有单元测试覆盖；新增 swa_evicted_seqlen 字段仅增加跟踪，不影响现有分配逻辑。
  2. 性能风险：alloc_extend_swa_tail 新增 num_new_pages 参数传递，但子分配器已处理该参数，无额外开销。
  3. 兼容性：NPU 路径内部映射方式未变，_is_npu 分支被统一到 set_full_to_swa_mapping，该函数内部已处理平台差异，无影响。 - 影响：影响范围集中在 PD 分离部署的解码路径中 SWA KV 分配器，核心收益是滑动窗口外 KV 可被及时释放，从而提升内存利用率。对非 PD 场景无影响，NPU decode 路径不受影响。测试文件改动确保回归覆盖。 - 风险标记：重构行为等价，少量新增字段

## 关联脉络

- 暂无明显关联 PR