PR #24439 完整报告

sgl-project/sglang

fix(req_pool): bump pool.size to match actual tensor row count after #24243

合并时间：2026-05-06 07:58

原文链接：http://prhub.com.cn/sgl-project/sglang/pull/24439

执行摘要

- 一句话：修复请求池大小不匹配导致的越界访问与性能退化
- 推荐动作：值得精读。本 PR 展示了如何从性能回归测试逆向追踪到 off-by-one 数据大小偏差，以及如何通过内部 _alloc_size 解耦池逻辑与消费者，体现了防御性编程思路。修改量不大但定位精准，非常适合作为代码审查和回归根因分析的案例。

功能与动机

Bug #24243 reserved slot 0 of ReqToTokenPool / DecodeReqToTokenPool as a zero padding row, but self.size was left at the original size. Many attention backends size buffers as [pool.size], leading to out-of-bounds access. Effects are silent and config-dependent. Multi-layer EAGLE + DP-attention amplifies the issue, causing deterministic accuracy regression in MiMo-V2-Flash from 0.776 to ~0.72.

实现拆解

修复分为核心池类修正和下游消费者对齐两步完成：

1. ReqToTokenPool（memory_pool.py）：引入 self._alloc_size = size + 1 表示实际张量行数。self.free_slots 基于 _alloc_size 初始化（range(1, self._alloc_size)）；clear() 同理。self.size 保持原始值，避免破坏外部契约。
2. DecodeReqToTokenPool（decode.py）：类似修改，self._alloc_size = size + pre_alloc_size + 1；free_slots 和 clear() 都依据 _alloc_size。HybridMambaDecodeReqToTokenPool.clear() 也做相应调整。
3. 下游消费者适配：
   - model_runner.py：maybe_init_ngram_embedding 中，token_table 的大小从 self.req_to_token_pool.size 改为 self.req_to_token_pool.req_to_token.shape[0]，始终读取张量的实际第一维尺寸。
   - multi_layer_eagle_worker_v2.py：req_to_hidden_states_pool 的初始化类似，改用 self.req_to_token_pool.req_to_token.shape[0]。
4. （隐含）_check_req_pool 校验器：从泄漏检测的 req_total_size 中减去 1，以排除永久保留的填充槽，避免启动时误报泄漏。

所有修改不涉及注意力后端代码，仅修正池自身计数与消费者取数位置。

关键文件：

- python/sglang/srt/mem_cache/memory_pool.py（模块 内存池；类别 source；类型 core-logic）：核心修复文件。在 ReqToTokenPool 中引入 _alloc_size 并修正 free_slots 和 clear()，消除 row 0 padding 后自身体积未更新的漏洞。
- python/sglang/srt/disaggregation/decode.py（模块 解码请求池；类别 source；类型 core-logic）：DecodeReqToTokenPool 做类似修正，并连带修复 HybridMambaDecodeReqToTokenPool.clear()。
- python/sglang/srt/model_executor/model_runner.py（模块 模型转发器；类别 source；类型 data-contract）：ngram embedding 的 token_table 初始化改用张量实际 shape 而非 pool.size，确保与 _alloc_size 一致。
- python/sglang/srt/speculative/multi_layer_eagle_worker_v2.py（模块 推测解码；类别 source；类型 core-logic）：KV cache 回退缓冲区初始化类似修正，避免因 pool.size 未更新导致的分配不足。

关键符号：ReqToTokenPool.init, ReqToTokenPool.clear, DecodeReqToTokenPool.init, DecodeReqToTokenPool.clear, maybe_init_ngram_embedding

关键源码片段

python/sglang/srt/mem_cache/memory_pool.py

核心修复文件。在 ReqToTokenPool 中引入 _alloc_size 并修正 free_slots 和 clear()，消除 row 0 padding 后自身体积未更新的漏洞。

```
class ReqToTokenPool:
    def __init__(
        self,
        size: int,
        max_context_len: int,
        device: str,
        enable_memory_saver: bool,
    ):
        memory_saver_adapter = TorchMemorySaverAdapter.create(
            enable=enable_memory_saver
        )
        self.size = size
        # +1 padding row at index 0: cuda-graph padded batches 默认将
        # req_pool_indices 设为 0，因此通过 slot 0 将傀儡写入 / 读取导向此处无害。
        self._alloc_size = size + 1
        self.max_context_len = max_context_len
        self.device = device
        with memory_saver_adapter.region(GPU_MEMORY_TYPE_KV_CACHE):
            self.req_to_token = torch.zeros(
                (self._alloc_size, max_context_len),
                dtype=torch.int32,
                device=device,
            )
```

```
        # 可用 slot 从 1 到 _alloc_size-1（含），slot 0 永远空闲但被保留
        self.free_slots = list(range(1, self._alloc_size))

    def clear(self):
        self.free_slots = list(range(1, self._alloc_size))
```

评论区精华

PR body 本身提供了极其详尽的根因分析：通过 MiMo-V2-Flash 的 canary 测试（stable mean 0.776 → deterministic ~0.72）定位到 #24243 后的首次 nightly 出现退化，并论证了为什么只有该模型阈值敏感。作者明确指出"该 bug 不是 MiMo 专属——它影响所有 cuda-graph 用户"。Review 中无实质性反对意见，hnyls2002 直接批准。

- 根因分析与修复验证 (correctness): 修复正确，批准合并。

风险与影响

- 风险：修复本身风险较低，改动集中在池类内部和两个消费者。需要注意：
  - 任何仍直接引用 pool.size 的外部代码（如第三方模型、自注册后端）可能仍会分配不足，但 PR 保持了 self.size 的原始值以避免外部分支全部崩溃，然而那些代码仍需后续适配。
  - _check_req_pool 中减去 1 的逻辑可能在其他调度检查中漏报真正的泄漏（但概率很低，因为填充槽永不释放）。
  - 修改了张量 shape 的读取方式（从常规属性改为 .req_to_token.shape[0]），若未来 req_to_token 被重新绑定，可能引入新 bug。
  - 影响：对用户：修复了自 #24243 合并后部分模型（尤其是 MiMo-V2-Flash 等带多层 EAGLE + DP-attention 的场景）的确定性精度退化，恢复至历史基线。对其他模型：虽未出现明显退化，但 OOB 风险被彻底消除。对团队：明确了请求池大小管理规范，为后续数据结构变更提供了警惕教训。影响范围限于 SRT 运行时核心，涉及池类与两个消费者模块，不影响推理结果语义。
  - 风险标记：核心路径变更，依赖张量 shape 的消费者可能仍需同步

关联脉络

- PR #24243 Add zero padding row in ReqToTokenPool: 本 PR 修复了 #24243 引入的 pool.size 未同步更新的回归 bug。