PR #25973 完整报告

sgl-project/sglang

Fix PD decode radix cache double-counting cached_tokens

合并时间：2026-05-28 11:50

原文链接：http://prhub.com.cn/sgl-project/sglang/pull/25973

执行摘要

- 一句话：修复 PD 分离模式下 cached_tokens 重复计数问题
- 推荐动作：建议所有使用 PD 分离模式且启用了 decode radix cache 的生产环境优先合并此 PR。值得关注的细节：通过播种 already_computed 来同步 prefill 和 decode 之间的状态，是一种简洁有效的状态传递模式，可参考用于类似的双阶段计数场景。

功能与动机

修复 Issue #25972：在 PD 分离模式下，当 decode 端启用 `--disaggregation-decode-enable-radix-cache` 时，`usage.prompt_tokens_details.cached_tokens` 被重复计数，可能超过 `prompt_tokens`（如 87 token 的 prompt 显示 128 个 cached_tokens），导致 OpenAI API 使用字段和 Prometheus 指标 `sglang_cached_tokens_total{engine_type="decode"}` 双重膨胀。根因是 decode 节点两次写入了 `req.cached_tokens` 且中间没有重置。

实现拆解

1. decode.py：在 `_commit_transfer_to_req` 中播种 `already_computed`：当 decode 端收到 prefill 传来的 metadata 时，除了设置 `cached_tokens`（prefill 端的缓存命中数），现在额外设置 `req.already_computed = req.cached_tokens`。这样后续 `prepare_for_prebuilt` 的 `cached_tokens += pre_len - already_computed` 只增加 decode 端新发现的缓存部分，而不会重复计算 prefill 已报告的部分。
2. decode_schedule_batch_mixin.py：在 `prepare_for_prebuilt` 中使用 `max(0, ...)` 钳位增量：将原来的 `req.cached_tokens += pre_len - req.already_computed` 改为 `delta = max(0, pre_len - req.already_computed); req.cached_tokens += delta; req.cached_tokens_device += delta`。这防止了当 decode 端的 prefix 比 prefill 报告的短时，出现负增量（即从 cached_tokens 中错误地减去）。同时也同步更新了 `cached_tokens_device`，确保指标一致性。
3. cache_hit_kit.py：增加上限断言：在现有的下限断言 `cached_tokens >= expected_cached` 之后，新增 `cached_tokens <= prompt_len` 的上限断言，从测试层面捕获该回归问题。之前的测试只检查下限，无法发现重复计数。
4. 测试验证：`test_disaggregation_decode_radix_cache.py` 是受影响的主要测试，加强后的断言在修复前会失败，修复后通过。另外，该逻辑只影响 PD 分离下的 decode 节点，colocated 模式不受影响。

关键文件：

- python/sglang/srt/disaggregation/decode.py（模块 分离调度；类别 source；类型 core-logic；符号 _commit_transfer_to_req）：修复核心：在 _commit_transfer_to_req 中添加 decode_req.req.already_computed = decode_req.req.cached_tokens，确保后续的 prepare_for_prebuilt 不会重复计数 prefill 端已经计数的 cached_tokens。
- python/sglang/srt/disaggregation/decode_schedule_batch_mixin.py（模块 分离调度；类别 source；类型 core-logic；符号 prepare_for_prebuilt）：修复核心：在 prepare_for_prebuilt 中，将直接相加减为 max(0, ...) 钳位，并同步更新 cached_tokens_device，确保计数正确且指标一致。
- python/sglang/test/kits/cache_hit_kit.py（模块 测试工具；类别 test；类型 test-coverage；符号 run_multiturn_cache_hit_test）：测试增强：增加上限断言 cached_tokens <= prompt_len，确保测试能捕获重复计数回归问题。

关键符号：_commit_transfer_to_req, prepare_for_prebuilt, run_multiturn_cache_hit_test

关键源码片段

python/sglang/srt/disaggregation/decode.py

修复核心：在 _commit_transfer_to_req 中添加 decode_req.req.already_computed = decode_req.req.cached_tokens，确保后续的 prepare_for_prebuilt 不会重复计数 prefill 端已经计数的 cached_tokens。

```
# decode.py 中 _commit_transfer_to_req 方法的修改片段
# ... 前面的代码略 ...
# Success - commit the transfer
decode_req.req.output_ids.append(output_id[0].item())
decode_req.req.cached_tokens = cached_tokens[0].item()
# The prefill node already reported its prefix-cache hit in
# cached_tokens[0]. Seed already_computed with it so that
# prepare_for_prebuilt's `cached_tokens += pre_len - already_computed`
# only adds decode-side reuse *beyond* what prefill counted, instead of
# double-counting the shared prompt prefix (which would make
# cached_tokens exceed prompt_tokens when decode radix cache is on).
decode_req.req.already_computed = decode_req.req.cached_tokens
decode_req.req.cached_tokens_device = cached_tokens[1].item()
decode_req.req.cached_tokens_host = cached_tokens[2].item()
decode_req.req.cached_tokens_storage = cached_tokens[3].item()
# ... 后续代码略 ...
```

python/sglang/srt/disaggregation/decode_schedule_batch_mixin.py

修复核心：在 prepare_for_prebuilt 中，将直接相加减为 max(0, ...) 钳位，并同步更新 cached_tokens_device，确保计数正确且指标一致。

```
# decode_schedule_batch_mixin.py 中 prepare_for_prebuilt 方法的修改片段
            if not req.retracted_stain:
                # Clamp to avoid double-counting: already_computed is seeded from
                # the prefill-reported cached_tokens in _commit_transfer_to_req, so
```

```
                # a decode-side prefix shorter than the prefill report must not
                # subtract from cached_tokens.
                delta = max(0, pre_len - req.already_computed) # 钳位增量
                req.cached_tokens += delta
                req.cached_tokens_device += delta # 同步更新设备端指标
                req.already_computed = seq_len
            req.is_retracted = False
```

python/sglang/test/kits/cache_hit_kit.py

测试增强：增加上限断言 cached_tokens <= prompt_len，确保测试能捕获重复计数回归问题。

```
# cache_hit_kit.py run_multiturn_cache_hit_test 中的修改片段
            assert resp.cached_tokens >= expected_cached
            # Upper bound: cached tokens are a subset of the prompt, so they can
            # never exceed prompt_len. In PD disaggregation with decode radix
            # cache, the shared prefix was previously counted on both the prefill
            # and the decode node, making cached_tokens exceed prompt_len.
            assert resp.cached_tokens <= resp.prompt_len, (
                f"Round {round_num}, client {i}: cached_tokens="
                f"{resp.cached_tokens} exceeds prompt_len={resp.prompt_len} "
                f"(double-counted prefix across prefill/decode)"
            )
```

评论区精华

1. bot 建议同步更新 cached_tokens_device：gemini-code-assist 在 review 中指出，当 decode 端 radix cache 发现额外 prefix 匹配时，增量也应加到 req.cached_tokens_device，以确保导出的指标分项之和与总 cached_tokens 一致。该建议已被采纳（在最终代码中增加了 req.cached_tokens_device += delta）。
2. ShangmingCai 求证是否影响现有逻辑：他询问更新 cached_tokens_device 是否会改变当前行为，因为 decode 端目前还不支持 CPU mem/storage load-back。ishandhanani 确认这是正确的，不会影响现有逻辑。

- 同步更新 cached_tokens_device 以保持指标一致 (correctness): 作者采纳建议，在代码中增加了 req.cached_tokens_device += delta。
- 更新 cached_tokens_device 是否影响现有逻辑 (question): ishandhanani 确认这是正确的更新，不会影响现有逻辑。

风险与影响

- 风险：本次修复修改了 PD 分离模式下 decode 端的核心计数逻辑，范围限定在 decode.py 和 decode_schedule_batch_mixin.py 两个文件，共 14 行代码变更。风险较低，因为：
  - prepare_for_prebuilt 只有 decode.py 中一个调用者，修改不会影响 colocated 模式。
  - 当 decode radix cache 关闭时，pre_len == 0，delta = 0，行为不变。
  - 测试加强后能预防回归。

潜在风险：如果 already_computed 被其他地方意外修改（目前没有），可能导致计算错误；同步更新 cached_tokens_device 可能改变导出指标的值，但这是修正 bug 的必要部分。

- 影响：影响范围中等，专注于 PD 分离场景：
  - 用户可见：OpenAI API 的 usage.prompt_tokens_details.cached_tokens 现在正确，不再出现超过 prompt_tokens 的异常值。
  - 监控指标：Prometheus 的 sglang_cached_tokens_total{engine_type="decode"} 不再膨胀，缓存命中率 Dashboard 恢复真实。
  - 团队：消除了一个关键的数据一致性问题，避免误导性能分析。
  - 性能：无额外开销，仅增加一个简单的减法与 max 操作。
  - 风险标记：核心路径变更，缺少测试覆盖

关联脉络

- PR #23269 Support batch size > 1 when enable CP: 该 PR 也修改了分离调度相关代码，涉及 decode_schedule_batch_mixin.py，说明两者共享相近模块，功能上可能有互动风险。
- PR #26387 Support KV events for UnifiedRadixCache: 该 PR 增强了 radix cache 的事件支持，本 PR 修复的正是 decode radix cache 的计数 bug，两者在缓存语义上有关联。