PR #25372 完整报告

sgl-project/sglang

[PDD] Add true request retraction for PDD

合并时间：2026-07-09 15:33

原文链接：http://prhub.com.cn/sgl-project/sglang/pull/25372

执行摘要

- 一句话：在 PDD 模式下实现真正的请求撤回与 KV 重算
- 推荐动作：该 PR 设计稳健，尤其是通过 bootstrap 注册 prefill_http_port 避免修改前端接口的决策值得借鉴。建议阅读 conn.py 的线程池和会话管理，以及 decode.py 的重新引导流程，以深入理解 PDD 模式下的状态管理。

功能与动机

引用 PR 描述：Currently we can't do true request retraction in PDD mode. This PR adds it: on retract, the decode worker drops the KV cache and, on continue_generation, asks the original prefill worker to recompute the prefix KV under the current weights (replaying the last emitted token). This makes retract → update_weights → continue correct in PDD mode.

实现拆解

1. 新增重新引导 payload 构建：在 Req 类 (schedule_batch.py) 中添加 build_rebootstrap_payload 方法，将完整的 token 序列（origin_input_ids + output_ids）转换为 Python int 列表，并携带采样参数（强制 max_new_tokens=1），确保 JSON 可序列化。
2. decode 侧暂存与重新入队：在 DecodePreallocQueue (decode.py) 中添加 hold_rebootstrap 和 enqueue_held_rebootstrap 方法。撤回时，弹出最后一个已发出 token 并标记为重新引导，将请求暂存到 held_rebootstrap_reqs 列表中；恢复生成时重新入队，并设置 is_rebootstrap=True 以绕过 radix 缓存，进行全新 KV 预分配。
3. 通信层线程池管理：在 CommonKVManager (conn.py) 中为 DECODE 模式新增共享 ThreadPoolExecutor（默认 16 线程）和每线程 requests.Session，通过 submit_prefill_recompute 方法向预填充实例提交异步 /generate 请求。失败时通过 KVPoll.Failed 路径回退。
4. 调度器撤回 / 恢复逻辑：修改 Scheduler.pause_generation 和 continue_generation (scheduler.py)：撤回时跳过 KV 卸载（仅在 OOM 撤回时卸载），转而调用 hold_rebootstrap；恢复时调用 enqueue_held_rebootstrap，使预分配队列在暂停期间保持为空以支持缓存刷新。

5. 预填充端口注册：在 PrefillServerInfo (conn.py) 中添加 prefill_http_port 字段，预填充实例在 bootstrap 注册时上报 HTTP 端口。*decode* 侧在需要重新引导时从已缓存的 server info 中取出端口，结合已知 host 构造 prefill_url，因此无需修改路由器（移除了 mini_lb.py 和 pd_router.rs 的改动）。

测试配套：新增单元测试（test_priority_scheduling_disaggregation.py、test_scheduler_pause_generation.py）验证 payload 构建、线程池分发、撤回 - 恢复流程；新增 E2E 测试（test_disaggregation_basic.py）覆盖 retract pause/resume 和 retract + weight_update 场景。

关键文件：

- python/sglang/srt/disaggregation/common/conn.py（模块 解聚通信；类别 source；类型 *core-logic*；符号 *_ensure_prefill_recompute_executor*, *_get_prefill_recompute_session*, *_resolve_rebootstrap_prefill_url*, *submit_prefill_recompute*）：核心通信层添加了预填充重新计算的线程池和 HTTP 会话管理，新增 PrefillServerInfo.prefill_http_port 字段，使得 *decode* 侧可自行构造 prefill URL，移除对路由器的依赖。
- python/sglang/srt/disaggregation/*decode*.py（模块 解码队列；类别 source；类型 *core-logic*；符号 *add*, *_create_receiver_and_enqueue*, *hold_rebootstrap*, *enqueue_held_rebootstrap*）：核心调度逻辑，添加了 is_rebootstrap 字段、*hold_rebootstrap*/*enqueue_held_rebootstrap* 方法，修改 *add* 和 *_create_receiver_and_enqueue* 支持重新引导请求。
- python/sglang/srt/managers/schedule_batch.py（模块 调度批处理；类别 source；类型 *core-logic*；符号 build_rebootstrap_payload, retract_all）：在 Req 类中新增 build_rebootstrap_payload 方法和 pd_rebootstrap_forced_output_id 字段，用于构建重新引导请求的 payload。
- test/registered/unit/managers/test_priority_scheduling_disaggregation.py（模块 优先级调度测试；类别 test；类型 *test-coverage*；符号 TestDecodePreallocQueueRebootstrapPayload, *_sampling_params*, *_new_req*, *test_build_rebootstrap_payload_converts_numpy_ids_to_json_lists*）：新增了两个测试类，覆盖重新引导 payload 构建和 CommonKVManager 的预填充重新计算分发。
- test/registered/disaggregation/test_disaggregation_basic.py（模块 解聚 E2E 测试；类别 test；类型 *test-coverage*；符号 TestDisaggregationPauseResumeDecodeRetract, setUpClass, *test_retract_pause_decode_running_batch*, *test_retract_weight_update_decode_running_batch*）：新增 E2E 测试类 TestDisaggregationPauseResumeDecodeRetract，验证撤回 - 暂停 - 恢复和权重更新场景。
- test/registered/unit/managers/test_scheduler_pause_generation.py（模块 暂停生成测试；类别 test；类型 *test-coverage*；符号 test_pd_decode_retract_requeues_for_rebootstrap, test_pd_decode_continue_releases_held_rebootstrap）：新增单元测试验证 *decode* 撤回时的重新引导入队和 *continue* 释放逻辑。

关键符号：*pause_generation*, *continue_generation*, *hold_rebootstrap*, *enqueue_held_rebootstrap*, *build_rebootstrap_payload*, *submit_prefill_recompute*,

_ensure_prefill_recompute_executor, dispatch_prefill_recompute, add,_create_receiver_and_enqueue

关键源码片段

python/sglang/srt/disaggregation/common/conn.py

核心通信层添加了预填充重新计算的线程池和 HTTP 会话管理，新增 PrefillServerInfo.prefill_http_port 字段，使得 decode 侧可自行构造 prefill URL，移除对路由器的依赖。

```
# python/sglang/srt/disaggregation/common/conn.py

# CommonKVManager.__init__ 的 DECODE 分支末尾添加了线程池和会话管理
self._prefill_recompute_executor: Optional[concurrent.futures.ThreadPoolExecutor] = None
self._prefill_recompute_executor_lock = threading.Lock()
self._prefill_recompute_sessions = threading.local()

def _ensure_prefill_recompute_executor(self) -> concurrent.futures.ThreadPoolExecutor:
    """懒创建共享线程池，用于向预填充实例发起 /generate 重新计算请求。
    池大小取自环境变量 SGLANG_DISAGGREGATION_THREAD_POOL_SIZE，默认 16。
    """
    executor = self._prefill_recompute_executor
    if executor is not None:
        return executor
    with self._prefill_recompute_executor_lock:
        if self._prefill_recompute_executor is None:
            workers = envs.SGLANG_DISAGGREGATION_THREAD_POOL_SIZE.get()
            if workers is None:
                workers = 16
            self._prefill_recompute_executor = concurrent.futures.ThreadPoolExecutor(
                max_workers=max(1, workers),
                thread_name_prefix="pd-rebootstrap-prefill",
            )
        return self._prefill_recompute_executor

def submit_prefill_recompute(self, prefill_url: str, payload: dict) -> concurrent.futures.Future:
    """将重新计算任务提交到共享线程池，返回 Future。
    失败时通过 _fail_prefill_recompute 路由到 KVPoll.Failed 路径。
    """
    executor = self._ensure_prefill_recompute_executor()
    return executor.submit(self._run_prefill_recompute, prefill_url, payload)

def _run_prefill_recompute(self, prefill_url: str, payload: dict) -> None:
    """实际执行 POST 请求，并在失败时调用 _fail_prefill_recompute。
    """
    session = self._get_prefill_recompute_session()
    try:
        resp = session.post(prefill_url, json=payload, timeout=30)
```

```
            resp.raise_for_status()
        except Exception as e:
            logger.error("Prefill recompute failed for %s: %s", payload.get("rid"), e)
            self._fail_prefill_recompute(payload.get("bootstrap_room"), str(e))
```

python/sglang/srt/disaggregation/decode.py

核心调度逻辑，添加了 is_rebootstrap 字段、hold_rebootstrap/enqueue_held_rebootstrap 方法，修改 add 和 _create_receiver_and_enqueue 支持重新引导请求。

```
# python/sglang/srt/disaggregation/decode.py ( 部分 )

# DecodePreallocQueue 初始化中添加暂存队列
self.held_rebootstrap_reqs: List[Req] = []

def hold_rebootstrap(self, req: Req) -> None:
    """将撤回的请求暂存，从 output_ids 中弹出最后一个 token 并记录为边界 token，
    后续在 decode 侧覆写预填充采样的结果。
    """
    # 弹出最后已发出的 token，保存用于覆写
    req.pd_rebootstrap_forced_output_id = int(req.output_ids.pop())
    req.pd_rebootstrap_in_progress = True
    self.held_rebootstrap_reqs.append(req)

def enqueue_held_rebootstrap(self) -> None:
    """在 continue_generation 时将暂存的重新引导请求入队到 prealloc 队列。
    重新引导请求需要全新预分配 KV（绕过 radix 缓存），设置 is_rebootstrap=True。
    """
    while self.held_rebootstrap_reqs:
        req = self.held_rebootstrap_reqs.pop(0)
        self.add(req, is_retracted=False, is_rebootstrap=True)

def add(self, req: Req, is_retracted: bool = False, is_rebootstrap: bool = False) -> None:
    """添加请求到 pending 队列。
    is_rebootstrap 标记 PD 真撤回请求，其前缀 KV 需由原预填充工作器重新计算。
    """
    if self._check_if_req_exceed_kv_capacity(req):
        return
    if is_retracted:
        req.retraction_mb_id = None
        self.retracted_queue.append(req)
    else:
        decode_req = self._create_receiver_and_enqueue(req, is_rebootstrap=is_rebootstrap)
        # ... 后续处理
```

python/sglang/srt/managers/schedule_batch.py

在 Req 类中新增 build_rebootstrap_payload 方法和 pd_rebootstrap_forced_output_id 字段，用于构建重新引导请求的 payload。

```
# python/sglang/srt/managers/schedule_batch.py (Req 类新增方法 )
```

```
def build_rebootstrap_payload(self) -> dict:
    """构建向预填充实例发起 /generate 请求的 payload。
    包含完整 token 序列（origin_input_ids + output_ids），转换为原生 Python int
    确保 JSON 序列化成功。sampling_params 限制 max_new_tokens=1，只采样一个边界
    token，该 token 后续在 decode 侧被 pd_rebootstrap_forced_output_id 覆写。
    """
    # TODO: 多模态请求暂不支持，缺少 image/audio/video 输入
    sp = self.sampling_params
    return {
        "input_ids": [int(x) for x in self.origin_input_ids] +
                     [int(x) for x in self.output_ids],
        "sampling_params": {
            "max_new_tokens": 1,
            "temperature": sp.temperature,
            "top_p": sp.top_p,
            "top_k": sp.top_k,
            "min_p": sp.min_p,
            "frequency_penalty": sp.frequency_penalty,
            "presence_penalty": sp.presence_penalty,
            "repetition_penalty": sp.repetition_penalty,
            "ignore_eos": sp.ignore_eos,
            "skip_special_tokens": sp.skip_special_tokens,
            "spaces_between_special_tokens": sp.spaces_between_special_tokens,
            "no_stop_trim": sp.no_stop_trim,
        },
        "return_logprob": False,
        "stream": False,
        "rid": self.rid,
        "bootstrap_host": self.bootstrap_host,
        "bootstrap_port": self.bootstrap_port,
        "bootstrap_room": self.bootstrap_room,
        "priority": self.priority,
        "extra_key": self.extra_key,
        "routing_key": self.routing_key,
        "disagg_prefill_dp_rank": self.disagg_prefill_dp_rank,
    }
```

评论区精华

- 重构实现方式避免修改路由器：ShangmingCai 强调不应修改前端接口，MrAta 改为通过 bootstrap 注册 prefill_http_port，由 *decode* 侧自行构造 URL，彻底移除路由器改动。
- 撤回时 KV 卸载优化：ShangmingCai 指出不应先卸载再立即删除，MrAta 修改为仅在 OOM 撤回时卸载，避免无用操作。
- 暂存请求不可中止风险：ShangmingCai 指出 held_rebootstrap_reqs 中的请求无法被 /abort_request 中止，MrAta 添加注释明确该限制，并说明在 RL 场景不会发生。
- 避免修改路由器前端接口 (design): 采用 bootstrap 注册端口，移除路由器的所有变更。

- 撤回时 KV 卸载优化 (correctness): 修改 pause_generation，添加条件判断，仅 OOM 撤回卸载。
- 暂存请求不可中止风险 (correctness): 添加注释记录限制，未修改实现。

风险与影响

- 风险:
  - 多模态请求未支持：build_rebootstrap_payload 方法注释明确指出多模态请求尚不支持，若启用可能导致 KV 重算错误。
  - 暂存队列不可中止：held_rebootstrap_reqs 中的请求在暂停期间无法被 /abort_request 中止，可能造成资源泄漏（但 RL 场景通常不中止）。
  - 线程池资源消耗：每个 *decode* 节点默认创建 16 线程的池，高频撤回时可能成为性能瓶颈。
  - 错误传播风险：后台线程中 /generate 失败通过 KVPoll.Failed 路由，但异常的流式输出未验证，可能导致客户端挂起。
  - 核心路径变更：修改了调度暂停 / 恢复核心逻辑，可能影响 OOM 撤回等其他撤回模式。
- 影响:
  - 用户影响：PDD 模式下启用 true retraction 后，可在不丢失生成进度的前提下更新模型权重，对在线 RL 训练等场景至关重要。
  - 系统影响：增加了 *decode* 端到 prefill 端的 HTTP 调用，可能增加网络延迟和 prefill 负载；新增线程池和会话池占用额外资源。
  - 团队影响：重构了解聚通信层，为未来支持多模态和更多撤回策略奠定基础。
  - 风险标记：核心路径变更，多模态未支持，线程池资源，中止请求遗漏，错误流式传播

关联脉络

- 暂无明显关联 PR