PR #47495 完整报告

vllm-project/vllm

[Bugfix][KV-transfer] MoRIIO: retry RDMA send-queue-full backpressure instead of failing the read

合并时间：2026-07-17 04:02

原文链接：http://prhub.com.cn/vllm-project/vllm/pull/47495

执行摘要

- 一句话：修复 MoRIIO RDMA 发送队列满时重试读请求
- 推荐动作：值得精读，尤其关注瞬态错误重试策略和 ACK 缓冲机制的设计权衡。_is_sq_full_status 的实现依赖字符串匹配，建议未来考虑更健壮的检测方式（如 mori 库的错误码）。

功能与动机

在高并发下，每个 QP 的 RDMA 发送队列是硬件限制的（如 bnxt max_qp_wr=4351），read_remote_data 同步提交 READ 请求时如果 SQ 满则立即返回 Failed 状态，此前该状态被当作传输失败，导致预填充释放 block、丢弃请求，最终因超时中止。实际上 CQ 轮询线程会在几毫秒内释放 SQ 深度，请求本可成功。

实现拆解

1. 缓冲未映射的 ACK：在 MoRIIOConnectorWorker.__init__ 中新增 _pending_unmapped_acks 列表，get_finished 先从该列表取 ACK，再合并 pop_finished_req_ids 的新 ACK，对于 transfer_id 不在 transfer_id_to_request_id 中的 ACK 不再直接警告丢弃，而是追加到缓冲列表等待下次轮询重试，解决通知先于 start_load_kv 到达导致的竞争。
2. SQ full 重试逻辑：在 _read_blocks 中，当 status.Failed() 且消息包含 SQ full（通过 _is_sq_full_status 静态方法检测）时，不直接标记失败，而是退避重试 READ，重试总时间受 transfer_timeout 限制。若超时仍未成功，则将失败状态存储，由 get_finished 按非致命方式处理。非 SQ full 的失败行为不变。
3. 放宽 block ID 校验：moriio_layout.py 中 compute_block_transfer_offsets 原要求 local_block_ids 长度严格等于 remote_block_ids，现改为只拒绝 local_block_ids 长于 remote_block_ids，允许短 / 空列表（对应 READ 模式的纯释放路径）。
4. 测试适配：更新 test_moriio_kv_layout.py 中的测试函数名为 test_local_block_ids_longer_than_remote_raises_value_error，新增 test_empty_local_block_ids_is_free_only_noop；在 test_moriio_tp_ack.py 的 test_worker_get_finished_counts_structured_release_fan_in 中为手工构造的 worker 设置 _pending_unmapped_acks = []，避免属性不存在错误。

关键文件：

- vllm/distributed/kv_transfer/kv_connector/v1/moriio/moriio_connector.py（模块 KV 传输；类别 source；类型 core-logic；符号 _is_sq_full_status）：核心变更文件：实现 SQ full 重试逻辑（_is_sq_full_status + _read_blocks 修改）和未映射 ACK 缓冲（_pending_unmapped_acks + get_finished 修改），+70/-9。
- vllm/distributed/kv_transfer/kv_connector/v1/moriio/moriio_layout.py（模块 KV 传输；类别 source；类型 core-logic）：放宽 compute_block_transfer_offsets 的校验，允许 local_block_ids 短于 remote_block_ids，支持 READ 模式纯释放路径。
- tests/v1/kv_connector/unit/test_moriio_kv_layout.py（模块 测试；类别 test；类型 test-coverage；符号 test_block_id_length_mismatch_raises_value_error, test_local_block_ids_longer_than_remote_raises_value_error, test_empty_local_block_ids_is_free_only_noop）：测试更新：更新原有测试函数名与匹配字符串，新增空 local_block_ids 的 no-op 测试。
- tests/v1/kv_connector/unit/test_moriio_tp_ack.py（模块 测试；类别 test；类型 test-coverage）：修复手工构造 worker 时未设置 _pending_unmapped_acks 导致的 AttributeError，仅一行变更。

关键符号：_is_sq_full_status, get_finished, compute_block_transfer_offsets, _read_blocks

关键源码片段

vllm/distributed/kv_transfer/kv_connector/v1/moriio/moriio_connector.py

核心变更文件：实现 SQ full 重试逻辑（_is_sq_full_status + _read_blocks 修改）和未映射 ACK 缓冲（_pending_unmapped_acks + get_finished 修改），+70/-9。

```
# vllm/distributed/kv_transfer/kv_connector/v1/moriio/moriio_connector.py

# 在 __init__ 中新增：缓冲 READ 完成 ACK 的列表
# 当 decode 的 release ACK 在 transfer_id_to_request_id 建立之前到达时，
# 先缓冲在此，每次 get_finished 轮询重试，避免丢失导致 block 泄漏。
self._pending_unmapped_acks: list = []

# 静态方法：检测 RDMA 发送队列满的瞬态错误
@staticmethod
def _is_sq_full_status(status) -> bool:
    """True if a MoRIIO transfer status is a transient RDMA send-queue-full
    rejection (retryable backpressure), not a terminal failure."""
    try:
        # mori 将 SQ full 表示为 Failed() 且消息包含 "SQ full"
        return bool(status.Failed()) and "SQ full" in (status.Message() or "")
    except Exception:
        return False

def get_finished(self) -> tuple[set[str], set[str]]:
    # 合并缓冲区和新 ACK，然后清空缓冲区
```

```
        finished_acks = self._pending_unmapped_acks + list(
            self.moriio_wrapper.pop_finished_req_ids()
        )
        self._pending_unmapped_acks = []
        for ack in finished_acks:
            transfer_id = ack if isinstance(ack, str) else ack.transfer_id
            if transfer_id not in self.transfer_id_to_request_id:
                # 映射未就绪，缓冲并留待下次轮询（不丢弃）
                self._pending_unmapped_acks.append(ack)
                continue
            # ... 原有 resolve 逻辑

    def _read_blocks(self, ...):
        # ... 发送 READ 后
        if not status.Succeeded():
            if self._is_sq_full_status(status):
                # SQ full：退避重试，由 transfer_timeout 限制总时间
                if self._retry_read(...):
                    return
            # 其他失败或超时：走原有失败处理
        # ...
```

vllm/distributed/kv_transfer/kv_connector/v1/moriio/moriio_layout.py

放宽 compute_block_transfer_offsets 的校验，允许 local_block_ids 短于 remote_block_ids，支持 READ 模式纯释放路径。

```
    # vllm/distributed/kv_transfer/kv_connector/v1/moriio/moriio_layout.py

    def compute_block_transfer_offsets(...):
        # 之前要求长度严格相等，现在允许 local 更短
        # 因为 READ 模式下 decode 只需释放部分 block
        if len(local_block_ids) > len(remote_block_ids):
            raise ValueError(
                "local_block_ids longer than remote_block_ids: "
                f"{len(local_block_ids)} > {len(remote_block_ids)}"
            )
        # ... 后续逻辑基于 len(local) 循环，自动处理短列表
```

评论区精华

CI 中的预提交检查因 pip-compile-xpu 从 pytorch xpu wheel 镜像获取 pyzmq 时出现 503 Service Unavailable 而失败，与代码本身无关，作者请求 maintainer 重跑 CI。另外，V1 Core + KV + Metrics 单元测试失败的原因是手工构造的 worker 未初始化 _pending_unmapped_acks 属性，作者快速提交补丁修复（0ca33d79）。

- 单元测试失败：缺少 _pending_unmapped_acks 属性 (testing)：提交 0ca33d79 修复，在测试中初始化该属性。
- 预提交检查失败：pytorch xpu wheel 镜像 503 (other)：由 maintainer 重跑后通过。

风险与影响

- 风险：
  1. 重试循环可能增加延迟：如果持续 SQ full 直到 transfer_timeout 超限，会占用更多时间，但这是合理退避，不会无限阻塞。
  2. SQ full 判断依赖字符串匹配：_is_sq_full_status 通过检查 status.Message() 是否包含 SQ full 来判断，该字符串依赖 mori 库的异常消息格式，未来库升级可能变更，缺少防御性适配。
  3. 缓冲 ACK 带来内存风险：极端情况下如果映射持续不建立，_pending_unmapped_acks 可能无限增长，但 transfer_timeout 最终会失败并清理映射。
  4. 放宽 length 校验：允许 local_block_ids 短于 remote_block_ids 可能隐藏真正的问题（如广播 bug），但通过只允许短于（不允许长于）来保留对异常情况的报错。 - 影响：直接影响使用 MoRIIO KV 传输的离散预填充 / 解码（P/D）部署，尤其是高并发多解码 /MTP 场景。此前 SQ full 会导致请求超时、KV block 泄漏、预填充缓存死锁，本修复消除了这些故障，使系统在高并发下稳定运行。对不使用 MoRIIO 的用户无影响。 - 风险标记：SQ full 重试依赖字符串匹配，缓冲 ACK 潜在无限增长，放宽长度校验可能掩盖异常

关联脉络

- PR #48209 Vectorize prep xfer list creation: 同一 MoRIIO 模块的性能优化，与当前 PR 共同改进高并发 KV 传输的稳定性与效率。
- PR #48717 [Misc][Nixl] Unify _logical_to_remote_kernel_block_ids: 另一 KV 连接器（Nixl）的类似重构，反映团队对 KV 传输路径的持续优化。