PR #48798 完整报告

vllm-project/vllm

Add tiering offloading metrics

合并时间：2026-08-10 10:23

原文链接：http://prhub.com.cn/vllm-project/vllm/pull/48798

执行摘要

- 一句话：为 KV 多级卸载新增分层指标，覆盖传输、查询、失败与活跃度
- 推荐动作：值得精读。重点学习：可观测性数据如何跨异步线程传递（JobResult.transfer_time）、追踪器与 orchestrator 的职责划分、per-block 异步延迟的语义定义，以及 NIXL telemetry 的接入方式。后续关注 #44008 的剩余指标项与 p2p telemetry follow-up。

功能与动机

PR body 明确该变更 'Covers the scope of TieringOffloadingSpec-Level Metrics in #44008'，目标是为多级 KV 卸载提供规范级可观测性：一是新增带 per-tier 标签的指标定义，二是让 JobResult 携带传输大小 / 时间，三是从 TieringOffloadingManager 发出读写、失败与 block 查询 / 命中计数。此前该子系统只有 LOOKUP_SYNC_DELAY 与 LOOKUP_ASYNC_DELAY 两个直方图，缺少传输量、失败率与 tier 占用情况，运维无法判断各 tier 的健康度与瓶颈。

实现拆解

TieringOffloadingSpec.build_metric_definitions() 新增 12 个指标，除 PROMOTION_ALLOCATION_FAILURES 外全部带 ("tier",) 标签；同时把既有 LOOKUP_SYNC_DELAY/LOOKUP_ASYNC_DELAY直方图的语义从请求级累计改为per-block、per-tier 的阻塞 / 异步等待，并同步更新 documentation。

JobMetadata 更名为 TransferJob，JobResult 新增 transfer_time 字段；fs 的 DualQueueThreadPool._worker 在 task() 前后计时，get_finished 返回累计时长；obj 改用 agent.get_xfer_telemetry(handle) 获取真实传输时长；example tier 填 0；p2p 暂用本地时间差并留待后续接入 telemetry。

TieringMetricsTracker 集中维护 _RequestMetricsState（lookup 去重与异步起点）和 _TierState（活跃 job/ 读写块数增量计数），提供 on_lookup、on_job_registered、on_job_finished、take_stats、assert_idle 等接口；lookup 计数按 (request, block, tier) 去重，请求分配后停止上报。

引入 JobMetadata(transfer_job, tier_idx) NamedTuple 统一 job 跟踪，_register_job/_pop_job 成为唯一入口并同步 metrics；_SecondaryTierFacingParent、_pending_load_submissions、request_level_tiers 全部从 tier 对象改为 tier_idx，消除重复

映射；get_stats() 聚合 tracker 与各 secondary tier 的统计。

新增 tests/v1/kv_offload/tiering/test_metrics.py（6 个用例覆盖 lookup、去重、job 完成、部分成功、gauge、分配失败）；改造 tests/v1/kv_offload/tiering/test_tiering_offloading.py 与 tests/v1/kv_connector/unit/offloading_connector/test_metrics.py 验证 spec 注册与 manager 聚合；各 tier 测试同步适配 TransferJob 接口。

关键文件：

- vllm/v1/kv_offload/tiering/metrics.py（模块 指标层；类别 source；类型 core-logic；符号 TieringMetricsTracker, _RequestMetricsState, _TierState, on_lookup）：新增的指标追踪器核心模块，集中实现 lookup 去重、job 生命周期计数与 stats 聚合，是本 PR 的主体。
- vllm/v1/kv_offload/tiering/manager.py（模块 多级卸载；类别 source；类型 core-logic；符号 JobMetadata, _register_job, _pop_job, _SecondaryTierFacingParent）：指标集成与核心重构发生地：统一 job 跟踪、接入 tracker、parent 包装改用 tier_idx，直接影响整个卸载协调流程。
- vllm/v1/kv_offload/tiering/base.py（模块 数据契约；类别 source；类型 data-contract；符号 TransferJob, JobResult, TieringOffloadingMetrics, submit_store）：定义传输数据契约：JobMetadata 更名为 TransferJob，JobResult 增加 transfer_time，新增全部指标常量，是各 tier 实现的公共接口。
- vllm/v1/kv_offload/tiering/spec.py（模块 指标注册；类别 source；类型 configuration；符号 TieringOffloadingSpec.build_metric_definitions）：指标对外注册入口，定义指标名、文档与标签，直接决定 Prometheus 暴露的指标形态。
- vllm/v1/kv_offload/tiering/fs/thread_pool.py（模块 线程池；类别 source；类型 core-logic；符号 JobState.task_done, DualQueueThreadPool.get_finished, DualQueueThreadPool._worker）：实现纯 I/O 计时：在 worker 任务执行前后记录耗时并随完成队列返回，是 transfer_time 数据链路的源头。
- tests/v1/kv_offload/tiering/test_metrics.py（模块 指标测试；类别 test；类型 test-coverage；符号 test_tiering_metrics_tracker_records_lookup_metrics, test_tiering_metrics_tracker_stops_lookup_metrics_after_allocation, test_tiering_metrics_tracker_records_finished_job_metrics, test_tiering_metrics_tracker_records_partial_promotion_success_bytes）：新增指标追踪器的专项测试，覆盖 lookup 去重、分配后停止、job 完成 / 部分成功、gauge 与分配失败计数。

关键符号：TieringMetricsTracker.init, TieringMetricsTracker.on_lookup, TieringMetricsTracker.on_job_registered, TieringMetricsTracker.on_job_finished, TieringMetricsTracker.take_stats, TieringOffloadingManager._register_job, TieringOffloadingManager._pop_job, TieringOffloadingManager.get_stats, DualQueueThreadPool._worker, JobState.task_done, TieringOffloadingSpec.build_metric_definitions

关键源码片段

vllm/v1/kv_offload/tiering/manager.py

指标集成与核心重构发生地：统一 job 跟踪、接入 tracker、parent 包装改用 tier_idx，直接影响整个卸载协调流程。

```
# vllm/v1/kv_offload/tiering/manager.py —— job 注册 / 弹出与 stats 聚合（整理后）

class JobMetadata(NamedTuple):
    # transfer_job 描述一次异步传输；tier_idx 记录该 job 归属的 secondary tier
    transfer_job: TransferJob
    tier_idx: int

class TieringOffloadingManager(OffloadingManager):
    def _register_job(self, transfer_job: TransferJob, tier_idx: int) -> None:
        # 统一入口：登记 job 元数据的同时更新 tracker 的活跃计数与 gauge
        job_metadata = JobMetadata(transfer_job, tier_idx)
        self._jobs[transfer_job.job_id] = job_metadata
        self._metrics.on_job_registered(job_metadata)

    def _pop_job(self, job_id: JobId) -> JobMetadata | None:
        # 统一出口：job 完成时移除登记并让 tracker 同步递减活跃状态
        return self._jobs.pop(job_id, None)

    def get_stats(self) -> OffloadingConnectorStats | None:
        # 聚合 tracker 的观察结果与各 secondary tier 自行上报的统计；
        # 返回后由上层收集为 Prometheus 指标
        stats = self._metrics.take_stats()
        for tier in self.secondary_tiers:
            tier_stats = tier.get_stats()
            if tier_stats is not None:
                if stats is None:
                    stats = tier_stats
                else:
                    stats.aggregate(tier_stats)
        return stats
```

vllm/v1/kv_offload/tiering/base.py

定义传输数据契约：JobMetadata 更名为 TransferJob，JobResult 增加 transfer_time，新增全部指标常量，是各 tier 实现的公共接口。

```
# vllm/v1/kv_offload/tiering/base.py —— 异步传输数据契约（整理后）

@dataclass
class TransferJob:
    # Metadata for an in-flight async transfer job.
    job_id: JobId
    keys: Collection[OffloadKey]
    block_ids: np.ndarray
    is_promotion: bool # True：secondary → primary（promotion）；False：primary →
    secondary（cascade）
```

```
    req_context: ReqContext


@dataclass
class JobResult:
    # Result of an async transfer job.
    job_id: JobId
    success: bool
    # 仅 promotion 部分失败时使用，标识成功加载的 keys；None 表示全部一致
    successful_keys: Collection[OffloadKey] | None = None
    # 纯 I/O 传输耗时（秒）：fs 来自线程池 task() 前后计时，obj 来自 NIXL telemetry
    transfer_time: float | None = None
```

评论区精华

orozery 是主要审核者，围绕设计做了多轮交锋并最终 APPROVED。核心讨论：

- 抽取追踪器：orozery 认为指标代码 (~130 行) 有凝聚力，建议挪到 tiering/metrics.py，使 manager.py 保持纯编排、tracker 可独立测试，落地为 TieringMetricsTracker。
- lookup 延迟语义：orozery 指出 per-request 异步延迟会把 promotion 时间也算进去，建议按 per-block/per-tier 记录首次 unresolved 起点、解析时再观测，最终实现 observed_lookups 存 start time 并配直方图。
- 计数去重：orozery 要求 queries/hit 只计一次 per [req][block][tier]，且分配后停止，最终通过 on_request_allocated 置 observed_lookups=None 实现。
- 传输耗时度量：orozery 要求 read/write time 只算纯 I/O 时间，启发 fs 线程池在 task() 前后计时；obj/p2p 改用 agent.get_xfer_telemetry(handle)，p2p 因需要额外 plumbing 留作 @liranschour 的 follow-up。
- 数据契约简化：orozery 建议移除 JobResult.transfer_size，由 manager 按 key 数与 primary block size 计算。
- 将指标逻辑从 manager.py 抽离为 TieringMetricsTracker (design): 已新建 metrics.py 并移入全部指标状态与方法，落地为 TieringMetricsTracker。
- lookup 异步延迟度量改为 per-block per-tier (design): 已实现 observed_lookups 存 start time，sync/async 延迟均 per-block per-tier 上报。
- 每 [req][block][tier] 的查询 / 命中计数去重 (correctness): on_lookup 中 setdefault 标记已观测，on_request_allocated 置 observed_lookups 为 None；新增专项测试覆盖。
- fs tier 的 read/write time 只统计 IO 时间 (performance): 已改在任务线程内计时，get_finished 返回累计 transfer_time。
- obj/p2p tier 使用 NIXL telemetry 获取传输时长 (design): obj 已接入 telemetry；p2p 保留为 @liranschour 的 follow-up。
- JobResult.transfer_size 改由 manager 计算 (design): 已移除，transfer_size = completed_key_count * primary_block_size。
- create_store_job 的 tier_idx 可选参数 (style): tier_idx 改为必传。

风险与影响

- 风险：
  - 计数状态一致性：_TierState 的增减必须在 job 注册 / 完成时严格配对，assert_idle 与内部断言是主要防线；若未来 tier 实现绕过 _register_job/_pop_job 会静默破坏 gauge。
  - 接口兼容性：JobMetadata→TransferJob 更名与 JobResult.transfer_time 新增影响所有 SecondaryTierManager 子类（fs/obj/example/p2p），第三方自定义 tier 需要同步适配。
  - 热路径开销：每次 lookup 增加 time.monotonic() 与字典操作，但仅在请求分配前执行且每 block 只计入一次；量级可忽略，仍建议在大规模 prefix-cache 场景跑基准确认。
  - 精度边界：fs 的 transfer_time 是多任务累计值；p2p 仍是本地时间差而非真实传输时长，跨机场景偏差可能较大。
  - 测试盲区：指标注册与 tracker 行为覆盖充分，但未做 Prometheus scraping 的端到端断言。
- 影响：
  - 用户 / 运维：新增 vllm:kv_offload_tiering_* 系列指标，可直接观测各 tier 的读写量、耗时、失败率与 primary 占用，便于定位卸载瓶颈。
  - 系统：lookup 与 job 生命周期路径新增轻量计数 / 计时，对推理主路径无正确性影响。
  - 团队：5 个 tier 实现需同步接口，metrics.py 成为后续指标接入的范式（tracker 模式 + telemetry 取时长）。
  - 该变更限定在 v1 kv_offload 子系统内部，不触碰调度、注意力等核心路径。
  - 风险标记：接口重构波及 4 个 tier 实现，lookup 热路径新增计时与字典操作，计数状态依赖断言维护，P2P 时长暂用近似值，指标链路缺端到端验证

关联脉络

- PR #49328 [KV Offload] Fix failed-load livelock by marking the lookup verdict as a miss: 同处 vllm/v1/kv_offload/tiering 子系统（fs/manager.py、obj/manager.py、async_lookup），两者共同打磨异步 job 与 lookup 状态机的可靠性。
- PR #51161 [Bugfix][KV Offload] Handle chunked local attention in offloading scheduler: 同属 KV offload scheduler 生命周期与 job 完成收集路径，与本 PR 的 job 完成收集逻辑相邻，说明该领域正在密集成熟。
- PR #51243 [KV Offload] Emit self-describing events for partial recurrent blocks: 同属 KV offload 可观测性建设，事件机制与本 PR 的指标体系互补，后续可联合用于容量规划与故障定位。