

PR #1839 完整报告

radixark/miles

Clear the new-engine flag from the driver

合并时间: 2026-09-04 12:05

原文链接: <http://prhub.com.cn/radixark/miles/pull/1839>

执行摘要

- 一句话: 删除 RDT 权重同步协议及配套代码, 为架构重构清理道路。
- 推荐动作: 该 PR 是一次有计划的、大规模的功能移除和代码清理, 属于重构链的中间步骤。它本身不包含新的功能或复杂的逻辑创新, 主要是机械性删除。因此, 不需要精读具体实现细节。但关注以下几点: 1) 理解 RDT 模式为何被临时移除 (为重构让路); 2) 关注 rdt.py 文件删除的彻底性, 确保没有残留引用; 3) 检查对现有参数和验证逻辑的改动是否干净利落。建议结合整个 Issue #1837 链条的后续 PR 一起审视, 以理解完整的重构意图。

功能与动机

根据 commit 消息和关联的 Issue #1837 (一个追踪重构和增强的跟踪议题), 移除 RDT 是为了在重构链上可以重新构建架构。原 RDT 实现的依赖 (如 `server_group`、`actor_factory` 中的管道) 与重构目标冲突, 因此需要先进行“暂时回退”。commit 明确指出, RDT 之后会基于重构后的新结构重新引入。

实现拆解

1. 移除 RDT 核心协议: 完全删除了 `miles/backends/training_utils/weight_update/protocols/rdt.py` 文件 (378 行), 该文件实现了基于 NIXL/RDMA 的 P2P 权重传输逻辑, 是 RDT 模式的唯一入口。同时更新了 `miles/backends/training_utils/weight_update/protocol.py` 中的导入关系。
2. 清理驱动侧 RDT 特定逻辑:
 - 在 `miles/ray/rollout/sglang_engine.py` 中, 移除了 `_launch_sglang_server` 函数、`SGLangServerActor` 导入、以及 `SGLangEngine.__init__` 中的 `pg_bundles` 和 `_scheduler_actors` 等 RDT 状态字段。
 - 简化了 `SGLangEngine._init_normal` 方法, 删除了所有 RDT 相关的启动逻辑 (如设置 `use_ray=True`、调用 `_launch_sglang_server`), 现在仅保留标准的 `launch_server_process` 路径。
 - 移除了 `SGLangEngine.get_scheduler_actors` 方法及其在 `shutdown` 中的清理逻辑。
 - 删除了 `miles/ray/rollout/sglang_server_actor.py` 文件 (38 行), 该文件定义了 RDT 模式下承载 SGLang HTTP 服务器的 Ray Actor。
3. 简化配置与验证:

- 在 `miles/utils/arguments.py` 中，从 `--update-weight-transfer-mode` 参数的 `choices` 列表中移除了 "rdt" 选项。
- 删除了 `miles_validate_args` 函数中专门针对 RDT 模式的验证逻辑（检查 Megatron 后端、不兼容 Shared PPO）。
- 在 `miles/ray/placement_group.py` 中，移除了 `_create_placement_group` 函数的 `is_rdt` 参数和相关的放置组排序逻辑，并将 `allocate_train_group` 中的 `num_gpus_per_actor` 固定为 0.4（RDT 模式下曾为 1）。

4. 调整训练与服务器组逻辑：

- 在 `miles/ray/train/actor_factory.py` 中，移除了 RDT 模式下设置的特定环境变量（如 `RAY_EXPERIMENTAL_NOSET_CUDA_VISIBLE_DEVICES`、`PYTORCH_CUDA_ALLOC_CONF`）和 `max_concurrency` 调整。
- 在 `miles/ray/rollout/server_group.py` 中，移除了创建 `RolloutRayActor` 时传递 `rdt_pg_kwargs`（包含 `pg_bundles`）的逻辑，并将引擎协调 Actor 的 GPU 保留量固定为 0.2。

5. 测试配套：

- 删除了端到端测试文件 `tests/e2e/megatron/test_qwen3_30B_A3B_rdt.py`，该文件专门测试 RDT 模式下的 Qwen3-30B-A3B 训练。
- 删除了单元测试文件 `tests/fast/utils/test_arguments.py` 中的 `TestRdtValidation` 测试类，该类用于验证 RDT 模式下的参数约束。
- 更新了 `tests/fast/dashboard/test_hooks.py`，添加了针对 `register_router` 函数在不同 dashboard 设置下的单元测试，这可能与清理后的代码路径相关。

关键文件：

- `miles/backends/training_utils/weight_update/protocols/rdt.py`（模块 权重更新协议；类别 `source`；类型 `deletion`；符号 `_staging_span`, `_EngineRankBucket`, `init`, `stage`）：完全删除了 RDT 权重同步协议的核心实现，是本次变更的核心动作。
- `miles/backends/sglang_utils/sglang_engine.py`（模块 SGLang 引擎；类别 `source`；类型 `core-logic`；符号 `_launch_sglang_server`, `get_scheduler_actors`）：移除了 `SGLangEngine` 中所有 RDT 特定的启动、状态管理和关闭逻辑，是驱动侧清理的关键文件。
- `miles/ray/placement_group.py`（模块 放置组管理；类别 `source`；类型 `core-logic`；符号 `_create_placement_group`）：移除了 RDT 模式下为避免死锁而调整的放置组排序逻辑，并将 GPU 保留量恢复为默认值，简化了资源分配。

关键符号： `UpdateWeightFromRDT.init`, `UpdateWeightFromRDT.connect`, `UpdateWeightFromRDT.create_gpu_replica`, `_EngineRankBucket.stage`, `SGLangEngine._init_normal`, `SGLangEngine.shutdown`, `SGLangEngine.get_scheduler_actors`, `_launch_sglang_server`, `SGLangServerActor.start`, `_create_placement_group`, `allocate_train_group`, `miles_validate_args`

关键源码片段

`miles/backends/training_utils/weight_update/protocols/rdt.py`

完全删除了 RDT 权重同步协议的核心实现，是本次变更的核心动作。

以下是被删除文件的核心类和函数的简化示例，用于说明其职责。

实际文件已删除，此代码片段为理解其作用的重建。

```
class _EngineRankBucket:
    """为一个引擎 rank 管理模型副本和 GPU 桶，用于 staging 权重。"""

    def __init__(self, model_replica, params_dict, param_specs, gpu_bucket, actors):
        self.model_replica = model_replica # 模型副本
        self.params_dict = params_dict # 参数字典
        self.param_specs = param_specs # 参数规格（形状、数据类型、字节数）
        self.gpu_bucket = gpu_bucket # 固定大小的 GPU 桶 (uint8)
        self.actors = actors # 负责拉取数据的 Ray Actor 列表

    def stage(self, names):
        """将参数名列表对应的参数 data 指向 GPU 桶内的连续视图。"""
        offset = 0
        views = []
        capacity = self.gpu_bucket.numel()
        for name in names:
            shape, dtype, nbytes = self.param_specs[name]
            # 计算对齐后的起止偏移量
            offset, end = _staging_span(offset, dtype, nbytes)
            assert end <= capacity # 桶溢出检查
            # 创建视图并重新指向参数 data
            view = self.gpu_bucket[offset:end].view(dtype).reshape(shape)
            self.params_dict[name].data = view
            views.append(view)
            offset = end
        return views

    def _staging_span(offset, dtype, nbytes):
        """计算在 uint8 桶中放置指定字节数的对齐起止偏移。"""
        start = (offset + dtype.itemsize - 1) // dtype.itemsize * dtype.itemsize
        return start, start + nbytes

class UpdateWeightFromRDT(WeightTransferProtocol):
    """实现基于 P2P 桶式 all-gather 和 HF 转换的 RDT 权重传输。"""

    def connect(self, rollout_engines, rollout_engine_lock, engine_gpu_counts, engine_gpu_
offsets, parallel_state, placement, selector):
        """规划传输，为每个引擎 rank 创建 GPU 副本和固定桶。"""
        # ... 核心连接逻辑，包括创建 _EngineRankBucket 实例
        pass

    def create_gpu_replica(self, parallelism_config, hf_checkpoint, server_args):
        """根据并行配置从 HF 检查点加载一个模型副本。"""
        # ... 从 SGLang 模型加载器获取模型并返回副本、参数字典和规格
```

pass

miles/backends/sglang_utils/sglang_engine.py

移除了 SGLangEngine 中所有 RDT 特定的启动、状态管理和关闭逻辑，是驱动侧清理的关键文件。

展示 SGLangEngine._init_normal 方法清理后的核心逻辑。

这是移除所有 RDT 分支后的简化版本。

```
def _init_normal(self, server_args_dict):
    """启动标准的 SGLang HTTP 服务器进程。"""
    logger.info(f"Launch HttpServerEngineAdapter at: {self.server_host}:{self.server_port}")
    # ServerArgs 一旦实例化就不允许修改，所以 IPv6 主机的方括号在此移除
    server_args = ServerArgs(**{**server_args_dict, "host": server_args_dict["host"].strip("[]")})
    # 现在总是通过 multiprocessing 启动服务器
    self.process = launch_server_process(server_args)

    # 向路由器注册 worker（如果配置了路由器）
    if self.node_rank == 0 and self.router_ip and self.router_port:
        if parse(sglang_router.__version__) <= parse("0.2.1") or self.args.use_miles_router:
            # ... 旧版路由器注册逻辑
        else:
            # ... 新版路由器注册逻辑
        response.raise_for_status()
```

展示 SGLangEngine.shutdown 方法清理后的核心逻辑。

```
def shutdown(self):
    """关闭 SGLang 服务器引擎。"""
    if self.args.rollout_external:
        return
    logger.info(f"Shutdown engine {self.server_host}:{self.server_port}...")
    # 从路由器中移除 worker（如果配置了路由器）
    if self.node_rank == 0:
        # ... 移除逻辑
    # 终止服务器进程
    kill_process_tree(self.process.pid)
    # 注意：不再需要处理 self._sglang_server_actor，因为它已被移除
```

miles/ray/placement_group.py

移除了 RDT 模式下为避免死锁而调整的放置组排序逻辑，并将 GPU 保留量恢复为默认值，简化了资源分配。

展示 _create_placement_group 函数清理后的核心排序逻辑。

```
def _create_placement_group(num_gpus):
    """创建具有指定 GPU 数量的放置组。"""
    if num_gpus == 0:
        return None, [], []
```

```

bundles = [{"GPU": 1, "CPU": 1} for _ in range(num_gpus)]
pg = placement_group(bundles, strategy="PACK")
# ... 获取 GPU ID 并排序

bundle_infos = [(i, gpu_ids[i][0], gpu_ids[i][1]) for i in range(num_bundles)]
# 移除 is_rdt 分支后，统一使用节点 IP + GPU ID 的标准排序
sorted_bundle_infos = sorted(bundle_infos, key=sort_key)

# ... 生成重排序后的索引和 GPU ID 列表
return pg, pg_reordered_bundle_indices, pg_reordered_gpu_ids

# 在 create_placement_groups 中调用时，不再传递 is_rdt 参数
pg, actor_pg_reordered_bundle_indices, actor_pg_reordered_gpu_ids = _create_placement_group(num_gpus)

```

评论区精华

PR 的 review 评论仅有一条来自 [guapisolo](#) 的批准（APPROVED），没有技术讨论。这表明该变更是预先计划好的、机械性的代码清理，没有引发设计上的争议。

- 暂无高价值评论线程

风险与影响

- 风险：

1. 功能移除风险：完全移除了 rdt 权重同步模式。任何依赖此模式的工作流程或用户将会受到影响。但考虑到这是重构计划的一部分，且该模式存在使用限制（仅限 Megatron 后端，不兼容 Shared PPO），影响范围可能可控。
2. 回归风险：删除核心路径代码（如 `sglang_engine.py` 的启动逻辑、`placement_group.py` 的排序逻辑）可能影响非 RDT 模式的部署。虽然 commit 消息强调这是“暂时回退”，但删除操作本身是最终的，需要后续 PR 重新引入功能。
3. 配置兼容性风险：`--update-weight-transfer-mode` 参数的可选值发生变化，使用了无效值 "rdt" 的配置将立即在参数解析阶段失败。这属于明确的 breaking change。
4. 测试覆盖：虽然删除了 RDT 专项测试，但核心的源码删除（`rdt.py`）本身移除了大量逻辑，这可能导致相关测试也随之失效。其他模式（如 broadcast、p2p）的测试需要确保未受破坏。

- 影响：

1. 用户影响：使用 `--update-weight-transfer-mode rdt` 参数的用户将无法启动训练，需要更改参数。这是计划内的破坏性变更。
 2. 系统影响：简化了训练启动路径和运行时状态管理，移除了 `SGLangServerActor` 和复杂的放置组排序逻辑，可能轻微提升系统稳定性和代码可维护性。
 3. 团队影响：为后续的架构重构（如 PR #1842 Split RolloutManager）铺平了道路，减少了合并冲突和认知负担。是更大规模重构中的一个必要清理步骤。
- 风险标记：核心路径删除，功能移除（Breaking Change），测试覆盖减少

关联脉络

- PR #1837 Tracking issue for refactoring and enhancements: 本 PR 是 Issue #1837 所追踪的大型重构链的一部分，具体对应 op2-7 项。
- PR #1838 remove duplicated field in actor group, unify dashboard hook guard, and change init order: 与本 PR 同属重构链，且 commit 历史显示本 PR 基于此 PR 的更改之上。两者共同为后续的 RolloutManager 拆分做准备。
- PR #1840 Extract future inference engine hooks on generation and eval: 同属重构链（op2-9），在本 PR 清理 RDT 之后，进一步提取钩子以支持未来的异步场景。
- PR #1842 Split RolloutManager into InferenceController and RolloutExecutor: 同属重构链（op2-1），是本 PR 所做清理之后的重大架构重构步骤，将 RolloutManager 拆分为两个独立组件。