PR #2181 完整报告

THUDM/slime

[3/n] Disaggregated rollout: engine-side /pull_weights

合并时间：2026-07-07 10:22

原文链接：http://prhub.com.cn/THUDM/slime/pull/2181

执行摘要

- 一句话：引擎侧新增 /pull_weights 端点，解耦权重同步 fan-out
- 推荐动作：建议精读 PR 的设计文档（body 和 commit message），特别是非 POSIX 文件系统的可见性处理策略。架构师应关注如何将执行流从 slime 下沉到引擎，这种模式可复用于其他需要跨 host 同步的场景。

功能与动机

"#2089 fanned the per-host delta apply out from slime via Ray (all_engine_actors) because only node 0 of a multi-node engine has an HTTP server — leaking engine topology into slime and breaking for external engines. This PR moves the pull into the engine: POST /pull_weights syncs the host-local checkpoint on every host the engine spans. slime talks to one endpoint per engine."

实现拆解

1. 引擎侧新增 /pull_weights 端点：在 sglang 的 HTTP 服务器上添加 POST /pull_weights 路由，接收 PullWeightsReqInput（包含 local_checkpoint_dir、source_dir、target_version）。通过 scheduler 广播到所有 host，由 SchedulerWeightUpdaterManager.pull_weights 调用 local_checkpoint.pull 执行实际 delta 应用，并通过 all-gather 确认全部成功。
2. slime 侧移除重复逻辑：删除 slime/ray/rollout.py 中的 all_engine_actors 传播和 sync_local_checkpoint，替换为对每个引擎调用 pull_weights。移除 slime/utils/disk_delta.py 中接收侧的函数（如 _apply_lock、init_local_checkpoint、apply_deltas），保留发布侧的函数。
3. 非 POSIX 文件系统后写钩子：新增 --custom-update-weight-post-write-path 参数，训练器在写入权重后执行此钩子（例如上传到对象存储），确保引擎可见。引擎侧新增 --custom-pull-weights-pre-read-hook，在读取前刷新文件系统缓存。
4. 修复全量 disk 模式的非 POSIX 兼容性问题：在 hf_checkpoint_saver.py 中拆分 _finalize_shard_files 为 _finalize_local_shards 和 _plan_shard_finalization，每个 rank 只重命名自己的 shard 文件，避免跨 rank 重命名在非 POSIX 文件系统上失败。同时修复了 update_weight_from_disk.py 中每个 rank 独立创建目录的问题。

关键文件：

- slime/backends/sglang_utils/sglang_engine.py（模块 引擎通信；类别 source；类型 core-logic；符号 set_weight_version, sync_local_checkpoint, pull_weights）：核心变更文件，新增 pull_weights 方法替代原来的 sync_local_checkpoint，移除 init_local_checkpoint 线程启动，删除 set_weight_version。展示了 slime 端如何调用引擎的 /pull_weights。
- slime/utils/disk_delta.py（模块 Delta 发布；类别 source；类型 dependency-wiring；符号 _apply_lock, _read_applied_version, _write_applied_version, drop_page_cache）：删除接收侧全部函数（_apply_lock、_read_applied_version、_write_applied_version、drop_page_cache、init_local_checkpoint、_apply_version、apply_xor、apply_overwrite），只保留发布侧的函数（overwrite_encode、checksum 等）。文件改为纯发布者（trainer-side），职责更清晰。
- docker/patch/latest/sglang-pull_weights.patch（模块 引擎补丁；类别 test；类型 test-coverage；符号 update_weight_version, SlowDownReqInput, check_weights）：引擎侧实现 /pull_weights 端点的完整 diff，新增路由、请求 / 响应结构、调度器注册、weight_updater 处理逻辑。是 PR 的另一半核心逻辑所在。
- slime/backends/megatron_utils/hf_checkpoint_saver.py（模块 检查点保存；类别 source；类型 core-logic；符号 _finalize_shard_files, _finalize_local_shards, _plan_shard_finalization）：修复非 POSIX 文件系统上全量 checkpoints shard 重命名的问题，每个 rank 只操作自己的 shard 文件，避免跨 rank 重命名失败。

关键符号：pull_weights (sglang_engine.py), pull_weights (weight_updater.py, in patch), _finalize_local_shards, _plan_shard_finalization, init_local_checkpoint (removed), sync_local_checkpoint (removed)

关键源码片段

slime/utils/disk_delta.py

删除接收侧全部函数（_apply_lock、_read_applied_version、_write_applied_version、drop_page_cache、init_local_checkpoint、_apply_version、apply_xor、apply_overwrite），只保留发布侧的函数（overwrite_encode、checksum 等）。文件改为纯发布者（trainer-side），职责更清晰。

```
# slime/utils/disk_delta.py (head)

# Trainer-side (publish) helpers for disk-level delta weight sync. The receive side —
# materializing the host-local checkpoint and applying published deltas in place — lives in
# the engine behind its /pull_weights endpoint (sglang.srt.weight_sync.disk_delta), so it
# runs on every host of a multi-node engine while slime only talks to one endpoint.

def overwrite_encode(new: np.ndarray, changed_mask: np.ndarray) -> np.ndarray:
    """The 'overwrite' delta: changed-position count (u4), positions (u4 each), then new values.
    Idempotent to apply, unlike xor (an involution); the trainer picks the encoding per the docs.
    """
    pos = np.flatnonzero(changed_mask).astype("<u4")
    return np.concatenate([np.array([pos.size], "<u4").view(np.uint8), pos.view(np.uint8),
    new[changed_mask]])
```

docker/patch/latest/sglang-pull_weights.patch

引擎侧实现 /pull_weights 端点的完整 diff，新增路由、请求 / 响应结构、调度器注册、weight_updater 处理逻辑。是 PR 的另一半核心逻辑所在。

```
# docker/patch/latest/sglang-pull_weights.patch (head, 关键部分 )

# 在 http_server.py 中注册路由
@app.post("/pull_weights")
@auth_level(AuthLevel.ADMIN_OPTIONAL)
async def pull_weights(obj: PullWeightsReqInput, request: Request):
    """Have every host of this deployment pull published weight deltas into its
    local checkpoint (materialized from the model path on first use)."""
    success, message = await _global_state.tokenizer_manager.pull_weights(obj, request)
    content = {"success": success, "message": message}
    return ORJSONResponse(content, status_code=200 if success else HTTPStatus.BAD_
    REQUEST)

# 在 io_struct.py 中定义请求 / 响应结构
@dataclass
class PullWeightsReqInput(BaseReq):
    local_checkpoint_dir: str
    source_dir: str
    target_version: int

@dataclass
class PullWeightsReqOutput(BaseReq):
    success: bool
    message: str

# 在 scheduler.py 中注册处理器
(
    PullWeightsReqInput,
    self.weight_updater.pull_weights,
),

# 在 weight_updater.py 中实现 pull_weights 方法
def pull_weights(self, recv_req: PullWeightsReqInput):
    """Sync this host's local checkpoint up to recv_req.target_version."""
    from sglang.srt.weight_sync import local_checkpoint
    server_args = self.tp_worker.model_runner.server_args
    try:
        local_checkpoint.pull(
            local_checkpoint_dir=recv_req.local_checkpoint_dir,
            base_dir=server_args.model_path,
            source_dir=recv_req.source_dir,
            target_version=recv_req.target_version,
            pre_read_hook=server_args.custom_pull_weights_pre_read_hook,
        )
```

```
            return PullWeightsReqOutput(success=True, message="")
        except Exception as e:
            return PullWeightsReqOutput(success=False, message=str(e))
```

slime/backends/megatron_utils/hf_checkpoint_saver.py

修复非 POSIX 文件系统上全量 checkpoints shard 重命名的问题，每个 rank 只操作自己的 shard 文件，避免跨 rank 重命名失败。

```
# slime/backends/megatron_utils/hf_checkpoint_saver.py (head)

def _finalize_local_shards(
    path: Path,
    local_state: dict[str, Any],
    shard_states: list[dict[str, Any] | None],
    *,
    write_index: bool,
) -> None:
    """Rename this rank's shard files per the global plan; optionally write the index.

    The plan is deterministic from the gathered states, so each rank renames only
    its own files: on a non-POSIX shared filesystem another rank's unpublished
    writes are not visible, let alone renamable.
    """
    rename_map, index_data = _plan_shard_finalization(shard_states)
    for old_name in local_state.get("shard_files", []):
        os.replace(path / old_name, path / rename_map[old_name])
    if write_index:
        with open(path / "model.safetensors.index.json", "w", encoding="utf-8") as f:
            json.dump(index_data, f, indent=2)


def _plan_shard_finalization(
    shard_states: list[dict[str, Any] | None],
) -> tuple[dict[str, str], dict[str, Any]]:
    """Compute the shard rename map and index from every rank's gathered state."""
    # ... 收集所有 state，生成 rename_map 和 index_data ...
    return rename_map, index_data
```

评论区精华

无 review 评论。PR 作者在 body 中详细说明了设计动机和验证结果。

- 暂无高价值评论线程

风险与影响

- 风险：
  1. 回归风险：移除 sync_local_checkpoint 和 all_engine_actors 可能影响依赖这些接口的第三方代码或外部引擎集成。目前仅 slime 内部使用，需确保外部引擎正确实现

/pull_weights。

2. 性能风险：新端点在每个 host 上执行 delta 应用，仍然受限于磁盘 I/O 和 CPU，但相比之前的 Ray fan-out 减少了 slime 侧的网络开销，整体应无退化。
3. 兼容性风险：需要引擎侧打补丁（sglang-pull_weights.patch）。未打补丁的旧引擎无法使用 disk 模式同步。
4. 非 POSIX 文件系统：引入的后写钩子和 pre-read 钩子依赖用户正确配置，否则权重可能不一致。 - 影响：用户 / 系统：多节点引擎和外部引擎的用户受益于简化配置，不再需要暴露所有内部 host 端口。单节点引擎无感知。团队：降低了 slime 与引擎拓扑的耦合，便于后续增加新引擎后端。训练器的权重发布接口变化（custom_delta_pre_push_path -> custom_update_weight_post_write_path），需通知用户更新参数。 - 风险标记：跨模块耦合：引擎补丁与后端绑定，非 POSIX 文件系统兼容性依赖用户配置，删除旧接口可能影响外部引擎集成

关联脉络

- PR #2089 [2/n] Disaggregated rollout: disk-level delta weight sync: 本 PR 的前置系列 PR，实现了磁盘级 delta 权重同步的基础设施，本 PR 在此基础上将 fan-out 下移到引擎。
- PR #1806 [1/n] Disaggregated rollout: ...: 系列的第一部分，推测为分离式 rollout 的初始实现，本 PR 是其后续演进。