PR #2142 完整报告

radixark/miles

Drop TE cached quantized weights before offloading the training actor

合并时间：2026-08-30 03:02

原文链接：http://prhub.com.cn/radixark/miles/pull/2142

执行摘要

- 一句话：清空 TE 量化缓存，offload 省 45% 主机内存流量
- 推荐动作：本 PR 值得精读。diff 极小（+32/-1），但有两层学习价值：一是 PR body 展示了如何用“每元素字节数 × 模块数”做理论测算并与实测对照（91%/96% 兑现率），并清晰量化了 MXFP8 与 NVFP4 两种布局的成本差异；二是评审者的收敛过程说明——默认开启的开关配 assert 会误伤无关配置（bf16 + CUDA graphs），以及跨后端共享工具中 Megatron 特定逻辑的放置取舍。可作为 offload 前释放派生缓冲的性能优化参考模板。

功能与动机

PR body 明确指出：低精度配方（MXFP8 / NVFP4 / blockwise FP8）下，TE 把每个权重现场量化并缓存在 module._fp8_workspaces，唯一的 .clear() 受 recipe 类型变化守护，缓存从第一次 forward 活到进程退出。colocate RL 中 actor 每 rollout step 被 offload，torch_memory_saver.pause() 把每个 tracked allocation 复制到 pinned host 内存；workspace 没有自己的 region，落入 default tag 被一并复制，每次 offload 一次 D2H + 恢复时一次 H2D，纯属浪费。而 workspace 是高精度权重的纯函数，TE 在下次 forward 会走 cache-miss 路径重建，备份它毫无意义。同时 CUDA graphs 下 captured graph 回放时 workspace 地址已烘焙进图，释放后可能被后续分配复用，因此该优化必须在该场景关闭。

实现拆解

1. 新增 CLI 开关（miles/utils/arguments.py）：在 add_cluster_arguments 中注册 --clear-quantized-weight-workspaces-on-offload，使用 argparse.BooleanOptionalAction，默认 True，支持 --no- 前缀显式关闭，保证默认收益且可回退。
2. 新增清理方法（miles/backends/megatron_utils/actor.py）：_clear_quantized_weight_workspaces 用三重条件收口——flag 开启、transformer_impl == "transformer_engine"、cuda_graph_impl == "none"；延迟导入 TransformerEngineBaseModule 后遍历 self.model 各 chunk 的所有子模块，对 TE 实例调用 module._fp8_workspaces.clear()。延迟导入保证非 TE 构建不触达 TE 模块类型。
3. 挂入 offload 路径：sleep() 在 clear_memory(clear_host_memory=True) 之前插入调用，使 torch_memory_saver.pause() 拷贝 tracked allocation 时 workspace 已归还分配器，从而不被复制。

4. 测试配套（tests/fast/backends/megatron_utils/test_shared_ppo_lifecycle.py）：_lifecycle_worker fixture 补上 clear_quantized_weight_workspaces_on_offload=False，因为 sleep() 会无条件读取该属性，否则 test_sleep_is_idempotent 抛 AttributeError。

设计演进上，原始实现曾放在共享的 memory_utils 模块并对 CUDA graphs 做 assert，评审者 yueming-yuan 将其收敛为 actor 内方法、断言改条件跳过，详见评论区精华。

关键文件：

- miles/backends/megatron_utils/actor.py（模块 训练端；类别 source；类型 core-logic；符号 _clear_quantized_weight_workspaces, sleep）：核心变更文件：新增 _clear_quantized_weight_workspaces 方法并在 sleep() offload 路径中调用，是本次优化的主逻辑所在。
- miles/utils/arguments.py（模块 参数配置；类别 source；类型 configuration；符号 add_cluster_arguments）：新增 --clear-quantized-weight-workspaces-on-offload 参数（默认 True），为该行为提供显式 gate 和回退手段。
- tests/fast/backends/megatron_utils/test_shared_ppo_lifecycle.py（模块 生命周期；类别 test；类型 test-coverage；符号 _lifecycle_worker）：配套测试修正：sleep() 无条件读取新 flag，fixture 需补字段否则 test_sleep_is_idempotent 抛 AttributeError。

关键符号：_clear_quantized_weight_workspaces, sleep

关键源码片段

miles/backends/megatron_utils/actor.py

核心变更文件：新增 _clear_quantized_weight_workspaces 方法并在 sleep() offload 路径中调用，是本次优化的主逻辑所在。

```
# miles/backends/megatron_utils/actor.py（节选重构）
    def _clear_quantized_weight_workspaces(self) -> None:
        """offload 前清空 TE 量化缓存，让 pause() 不再复制纯派生数据。"""
        if not (
            # 开关默认开启，可通过 --no-clear-quantized-weight-workspaces-on-offload 显式关闭
            self.args.clear_quantized_weight_workspaces_on_offload
            # 仅对 TransformerEngine 后端生效
            and self.args.transformer_impl == "transformer_engine"
            # 已捕获的 CUDA graph 回放时地址已烘焙进图，释放后会让后续分配复用该内存
            and self.args.cuda_graph_impl == "none"
        ):
            return
        # 延迟导入：非 TE 构建不该触达 TE 模块类型
        from transformer_engine.pytorch.module.base import TransformerEngineBaseModule

        for model_chunk in self.model: # Megatron 的 model 可能是多 chunk 列表
            for module in model_chunk.modules():
                if isinstance(module, TransformerEngineBaseModule):
                    # workspace 是高精度权重的纯函数，下次 forward 走 cache-miss 路径重建
                    module._fp8_workspaces.clear()
```

```
    @with_logs
    @timer
    def sleep(self) -> None:
        assert self.args.offload_train
        if self._asleep:
            logger.info("sleep() called while already offloaded; skipping")
            return

        # 在 torch_memory_saver.pause() 之前清空 workspace，避免 D2H/H2D 纯开销
        self._clear_quantized_weight_workspaces()
        clear_memory(clear_host_memory=True)
        print_memory("before offload model")
        should_log_cpu_memory = is_first_replica_megatron_main_rank() and hasattr(self, "_last_
        rollout_id")

        destroy_process_groups()

        if self.args.rematerialize_param_from_master_weight and self.role == "actor":
            # 参数保持驻留供 update_weights 使用，之后由 pause 处理
            torch_memory_saver.pause(tag="grad_buffer")
            torch_memory_saver.pause(tag="default")
        else:
            tag = "default" if lora_rollout_enabled(self.args) else None
            torch_memory_saver.pause(tag=tag)
```

miles/utils/arguments.py

新增 --clear-quantized-weight-workspaces-on-offload 参数（默认 True），为该行为提供显式 gate 和回退手段。

```
# miles/utils/arguments.py（节选重构）
parser.add_argument(
    "--clear-quantized-weight-workspaces-on-offload",
    action=argparse.BooleanOptionalAction, # 支持 --no-... 显式关闭
    default=True, # 默认开启，绝大多数场景收益为正
    help=(
        "Drop TransformerEngine's cached quantized weights before offloading the "
        "training actor. They are rebuilt on the next forward, so backing them up "
        "to pinned host memory is pure overhead. Ignored when TransformerEngine "
        "is not in use or CUDA graphs are enabled."
    ),
)
```

评论区精华

仓库 review 评论为空，实质评审意见体现在 yueming-yuan 的后续提交记录中，按 commit 消息提炼：

- CUDA graphs 下 assert 改静默跳过（commit d4d47b3）：原实现“Asserted off under CUDA graphs”，但 flag 默认开，导致所有 cuda_graph_impl != none 且 offload 的跑批

直接 abort，包括没有 workspace 的 bf16 跑批。结论：改为条件判断静默跳过，避免误伤无关配置。

- 实现位置收敛为 actor 方法（commit 61e03d8、9d2e797）：memory_utils 被 fsdp backend 与通用 ray 层共享，而该逻辑依赖 Megatron 的 TransformerConfig 与 TE 模块类型，放共享层不合适。结论：inline 为 actor 内方法，三个 gate 在调用点一处收口，非 TE 构建不触达 TE import。
- 测试 fixture 缺字段（commit 2dc8023）：sleep() 无条件读 flag，fixture 的 Namespace 缺该字段导致测试抛 AttributeError。结论：补 clear_quantized_weight_workspaces_on_offload=False。

三个问题均已通过后续 commit 解决，评审最终 APPROVED，无未解决疑虑。

- CUDA graphs 下由 assert 改为静默跳过 (correctness): 改为条件判断，CUDA graphs 下静默保留 workspace 不清理，行为在 flag help 文本中说明。
- 实现位置从共享 memory_utils 收敛为 actor 方法 (design): inline 为 actor 内方法，flag、transformer_impl、cuda_graph_impl 三个 gate 在调用点一处收口，延迟导入保证非 TE 构建不触达 TE import。
- sleep 生命周期测试 fixture 缺字段 (testing): fixture 补 clear_quantized_weight_workspaces_on_offload=False，使既有生命周期测试继续通过。

风险与影响

- 风险：
  - 默认开启的行为变更：flag 默认 True，所有 TE + colocate offload 的低精度跑批行为立即变化；虽有三重 gate，但仍需关注 bf16 + TE + 非 CUDA graphs 场景下清空操作的额外开销（重建缓存）。
  - 依赖 TE 私有 API：module._fp8_workspaces 不是公开接口，TE 版本升级可能改名或改变生命周期语义；仓库近期有 bump sglang 与 flash-linear-attention 引发适配的先例（PR#2714、PR#2790），升级 TE 时需回归验证。
  - CUDA graphs 兼容：该场景被显式排除，workspace 仍会被拷贝，属于有意取舍；但需保证 gate 判断与实际 graph 捕获路径一致，否则可能出现静默错误。
  - 测试覆盖偏弱：测试只补齐 fixture 让旧用例通过，没有 mock 模块断言 _clear_quantized_weight_workspaces 的清理行为；理论值与实测兑现率（91%/96%）只存在于 PR body，未固化为 CI 断言。
  - 重建开销未量化：清空后下次 forward 走 cache-miss 重建，PR 未量化重建的 GPU 计算成本，但从 forward 必然量化来看影响有限。
- 影响：
  - 用户 / 系统：低精度 colocate RL 训练的主机内存带宽显著下降——DeepSeek-V4-Flash MXFP8 每 offload 减少约 52.3 GB（4 ranks，-45%），GLM-5.2 NVFP4 减少约 4.63 GB（4 ranks），offload/ 恢复更快，rollout 步间隔缩短。
  - 范围：只影响 Megatron 后端 + TransformerEngine + 非 CUDA graphs 的 actor offload 路径；FSDP backend 与通用 ray 层因实现收敛在 actor 内而不受影响。

- 团队：提供了一个默认开启、可回退、有精确数字背书的性能开关，可作为同类“offload 前释放派生缓冲”优化的模板。
- 风险标记：核心路径变更，默认开启行为变更，依赖 TE 私有 API, CUDA graphs 兼容需验证，测试未覆盖清理逻辑断言

关联脉络

- PR #2764 perf(megatron): keep policy logits in model precision: 同属低精度训练优化线，共同修改 actor.py 与 test_shared_ppo_lifecycle.py，分别从计算精度与 offload 内存拷贝角度优化同一条 PPO 训练管线。
- PR #2739 feat: dist_muon offloading in megatron: 同为训练 actor offload 主题，在 arguments.py 中扩展 offload 参数族；本 PR 的 workspace 清理开关与其正交，可与 --offload-train-target disk 组合使用。
- PR #2818 fix(megatron): keep SFT logits in model precision: 低精度 / 精度保持方向的后续修复，涉及同一套 Megatron 训练工具链与共享生命周期测试，说明该仓库正在系统性地打磨 Megatron 低精度路径。