PR #1899 完整报告

THUDM/slime

Patch Megatron TP grad coalesce to chunked all-reduce

合并时间：2026-05-11 16:26

原文链接：http://prhub.com.cn/THUDM/slime/pull/1899

执行摘要

- 一句话：将 Megatron TP 梯度归约从单一大 buffer 改为分块 all-reduce，避免大模型下 OOM
- 推荐动作：建议精读此 PR，尤其是 _split_into_chunks 贪心分块策略、_grad_attr 和 _fsdp_flag 的跨版本兼容写法，以及 monkey-patch 嵌入方式。这些设计模式可复用于其他 Megatron 内部函数修补场景。

功能与动机

在大型模型 TP 训练中，原版将梯度合并为一个连续 buffer 的 all_reduce 分配了巨量连续内存（GLM-4.6 750B 约 12.43 GiB），在 PYTORCH_ALLOC_CONF=expandable_segments:True 下因内存碎片稳定 OOM。通过分块避免一次大连续分配，解决可靠 OOM 问题。

实现拆解

1. 新增补丁模块：slime/backends/megatron_utils/megatron_patch/megatron_chunked_grad_coalesce_patch.py 定义了核心函数 _split_into_chunks（贪心分块）、_grad_attr（兼容两个版本的 grad 属性获取）、_fsdp_flag（兼容两个 FSDP 配置名），以及补丁版本的 _allreduce_non_tensor_model_parallel_grads。
2. 核心逻辑：遍历模型各 chunk 的所有参数，根据 average_gradients_across_tp_domain 和 sequence_parallel / qk_layernorm 分为 sum 和 avg 两类梯度列表；对每类调用 _split_into_chunks 按 SLIME_GRAD_COALESCE_CHUNK_BYTES（默认 1 GiB）分块，每块单独执行 _flatten_dense_tensors -> all_reduce -> _unflatten_dense_tensors 并写回原参数 grad。
3. 动态版本适配：通过 inspect.signature 判断 _get_main_grad_attr 是否接受 use_custom_fsdp 参数；通过函数签名判断 tp_group 是否传入；无需版本条件导入。
4. 补丁激活：megatron_patch/__init__.py 导入补丁模块，slime/backends/megatron_utils/__init__.py 在初始化末尾添加 from . import megatron_patch，沿用已有的 monkey-patch 模式。
5. 配置与异常处理：通过环境变量 SLIME_GRAD_COALESCE_CHUNK_BYTES 可调（默认 1073741824 字节）；try/except ImportError 捕获导入失败并发出警告，不会静默失败。

关键文件：

- slime/backends/megatron_utils/megatron_patch/megatron_chunked_grad_coalesce_patch.py（模块 梯度归约；类别 source；类型 core-logic；符号 _grad_attr, _fsdp_flag, _split_into_chunks, _allreduce_non_tensor_model_parallel_grads）：新核心补丁文件，实现可配置分块的 TP 梯度归约，包含跨版本兼容适配和贪心分块算法。
- slime/backends/megatron_utils/megatron_patch/__init__.py（模块 补丁入口；类别 source；类型 dependency-wiring）：使 megatron_chunked_grad_coalesce_patch 在导入 megatron_patch 时自动执行。
- slime/backends/megatron_utils/__init__.py（模块 初始化；类别 source；类型 dependency-wiring）：在 slime 初始化时导入 megatron_patch 子模块，从而自动应用梯度归约补丁。

关键符号：_split_into_chunks, _grad_attr, _fsdp_flag, _allreduce_non_tensor_model_parallel_grads

关键源码片段

slime/backends/megatron_utils/megatron_patch/megatron_chunked_grad_coalesce_patch.py

新核心补丁文件，实现可配置分块的 TP 梯度归约，包含跨版本兼容适配和贪心分块算法。

```
# 该文件是 slime/backends/megatron_utils/megatron_patch/megatron_chunked_grad_coalesce_patch.py
# 运行时动态检测 Megatron 版本，适配 API 差异。
import inspect
import os
import torch
from megatron.core.distributed.finalize_model_grads import (
    _flatten_dense_tensors, _get_main_grad_attr, _unflatten_dense_tensors,
)

# 检测 _get_main_grad_attr 是否接受 fsdp 参数 (post-core_v0.15.0rc7 版本 )
_gma_takes_fsdp_arg = len(inspect.signature(_get_main_grad_attr).parameters) >= 2

def _grad_attr(param, fsdp_on):
    """兼容两个版本：若支持 fsdp 标志则传入，否则仅传 param。"""
    if _gma_takes_fsdp_arg:
        return _get_main_grad_attr(param, fsdp_on)
    return _get_main_grad_attr(param)

def _fsdp_flag(ddp_config):
    """兼容两个版本的 FSDP 属性名 (use_megatron_fsdp / use_custom_fsdp)。"""
    return bool(getattr(ddp_config, "use_megatron_fsdp", False) or
                getattr(ddp_config, "use_custom_fsdp", False))

# 环境变量控制块大小，默认 1 GiB
_chunk_bytes = int(os.environ.get("SLIME_GRAD_COALESCE_CHUNK_BYTES") or (1 << 30))

def _split_into_chunks(params, grads, target_bytes):
```

```
    """贪婪分块：确保每个块总大小不超过 target_bytes，
    单个超大梯度单独成块。返回 (param_chunk, grad_chunk) 列表。"""
    chunks, cur_p, cur_g, cur_b = [], [], [], 0
    for p, g in zip(params, grads, strict=False):
        gb = g.numel() * g.element_size()
        if cur_g and cur_b + gb > target_bytes:
            chunks.append((cur_p, cur_g))
            cur_p, cur_g, cur_b = [], [], 0
        cur_p.append(p)
        cur_g.append(g)
        cur_b += gb
    if cur_g:
        chunks.append((cur_p, cur_g))
    return chunks
```

评论区精华

此 PR 无公开 review 讨论，直接由维护者审核并合并。PR body 详细说明了设计权衡和数学等价性。

- 暂无高价值评论线程

风险与影响

- 风险：1）性能风险：分块增加了 all_reduce 调用次数（约 12 vs 1），但 body 评估额外开销 <1ms，可忽略。2）兼容性风险：补丁依赖 Megatron 内部 _get_main_grad_attr、_flatten_dense_tensors 等私有函数，未来版本升级可能导致失效，但 try/except 会输出警告日志，避免静默失败。3）配置风险：若 SLIME_GRAD_COALESCE_CHUNK_BYTES 设置过小（如 <1 MB）会导致大量小 all_reduce，降低性能；默认值合理。4）未添加测试覆盖此补丁的集成行为，回归依赖运行时日志。
- 影响：对用户：使用 TP 并行且梯度总量大的大模型训练（如 GLM-4.6 750B）可直接避免 OOM，训练稳定性显著提升。对系统：无外部依赖变更，仅增强现有补丁机制。对团队：需维护跨版本兼容的补丁代码，但通过 inspect 动态适配减轻了负担。影响范围中等，但影响程度较高。
- 风险标记：核心训练路径变更，依赖 Megatron 内部 API，无测试覆盖

关联脉络

- 暂无明显关联 PR