PR #6410 完整报告

verl-project/verl

[fsdp] feat: Support zero2 optional feature for FSDP1 in engine worker.

合并时间：2026-05-20 17:51

原文链接：http://prhub.com.cn/verl-project/verl/pull/6410

执行摘要

- 一句话：FSDP1 engine worker 支持 Zero2 分片
- 推荐动作：该 PR 值得精读，尤其适合关注 FSDP 训练性能优化的工程师。设计决策上，复用 reshard_after_forward 配置而非新增参数是一个值得借鉴的简洁思路。但建议后续清理冗余赋值，并补充针对 Zero2 策略的回归测试。

功能与动机

根据 PR body，目的是将之前已合并 PR #4659 中 fsdp_workers 的特性迁移到 engine workers 中，允许用户在使用 FSDP1 时选择是否启用 Zero2 分片，从而在训练大模型时减少通信开销、降低显存压力，同时保持对 Zero3 的兼容。

实现拆解

1. 修改分片策略选择函数（verl/workers/engine/fsdp/utils.py）：在 get_sharding_strategy 函数签名中增加 zero3_enable=True 参数。当 zero3_enable=True 时，仍使用原来的 FULL_SHARD（1D mesh）或 HYBRID_SHARD（2D mesh）；当 zero3_enable=False 时，替换为 SHARD_GRAD_OP（Zero2）或 _HYBRID_SHARD_ZERO2（2D mesh）。通过条件分支避免了硬编码，使得 Zero2/Zero3 的切换逻辑清晰。
2. 调用端注入配置（verl/workers/engine/fsdp/transformer_impl.py）：在构建 FSDP 模块的 _build_fsdp_module 方法中，调用 get_sharding_strategy(fsdp_mesh, zero3_enable=self.engine_config.reshard_after_forward)，即利用现有配置字段 reshard_after_forward（通常为 True 表示 Zero3，False 表示 Zero2）来控制分片策略。此设计无需新增配置项，复用已有语义。
3. 代码清理：虽然 review 评论指出改动中存在冗余赋值（如先赋 FULL_SHARD 再被 fsdp_strategy 覆盖），但本 PR 未做进一步清理，推测为设计简化或未来优化预留。

关键文件：

- verl/workers/engine/fsdp/utils.py（模块 引擎模块；类别 source；类型 core-logic；符号 get_sharding_strategy）：核心逻辑变更：修改了 get_sharding_strategy 函数，新增 zero3_enable 参数，实现了 Zero2/Zero3 策略的动态选择。
- verl/workers/engine/fsdp/transformer_impl.py（模块 引擎模块；类别 source；类型 core-logic）：调用端适配：在 _build_fsdp_module 方法中传入

zero3_enable=self.engine_config.reshard_after_forward，实现配置驱动。

关键符号：get_sharding_strategy

关键源码片段

verl/workers/engine/fsdp/utils.py

核心逻辑变更：修改了 get_sharding_strategy 函数，新增 zero3_enable 参数，实现了 Zero2/Zero3 策略的动态选择。

```
# verl/workers/engine/fsdp/utils.py 中的核心分片策略选择函数
# 新增 zero3_enable 参数，允许用户选择 Zero2 或 Zero3

def get_sharding_strategy(device_mesh, zero3_enable=True):
    """
    Determine the appropriate sharding strategy based on the device mesh dimensions
    and zero3_enable flag.

    Args:
        device_mesh: torch.distributed.device_mesh.DeviceMesh
        zero3_enable: bool, 若为 True 使用 Full Shard (Zero3), 否则使用 SHARD_GRAD_OP (Zero2)

    Returns:
        ShardingStrategy
    """
    from torch.distributed.fsdp import ShardingStrategy

    # 根据 zero3_enable 选择基础策略
    if zero3_enable:
        fsdp_strategy = ShardingStrategy.FULL_SHARD
        hsdp_strategy = ShardingStrategy.HYBRID_SHARD
    else:
        fsdp_strategy = ShardingStrategy.SHARD_GRAD_OP
        hsdp_strategy = ShardingStrategy._HYBRID_SHARD_ZERO2

    # 根据 mesh 维度决定最终策略
    if device_mesh.ndim == 1:
        sharding_strategy = ShardingStrategy.FULL_SHARD # 冗余赋值，后续被覆盖
        sharding_strategy = fsdp_strategy
    elif device_mesh.ndim == 2:
        sharding_strategy = ShardingStrategy.HYBRID_SHARD # 冗余赋值
        sharding_strategy = hsdp_strategy
    else:
        raise NotImplementedError(f"Get device mesh ndim={device_mesh.ndim}, but only support
        1 or 2")

    return sharding_strategy
```

verl/workers/engine/fsdp/transformer_impl.py

调用端适配：在 _build_fsdp_module 方法中传入 zero3_enable=self.engine_config.reshard_after_forward，实现配置驱动。

```
# verl/workers/engine/fsdp/transformer_impl.py 中构建 FSDP 模块的方法片段
# 展示如何将配置传递给 get_sharding_strategy

def _build_fsdp_module(self, module):
    # ... 前面的 mixed_precision 和 auto_wrap_policy 设置 ...

    fsdp_mesh = self.device_mesh
    # 关键改动：通过 reshard_after_forward 配置控制 Zero2/Zero3
    sharding_strategy = get_sharding_strategy(fsdp_mesh, zero3_enable=self.engine_config.
    reshard_after_forward)

    if self.engine_config.strategy == "fsdp":
        # ... CPU offload 设置 ...
        module = FSDP(
            module,
            # ... 其他参数 ...
            sharding_strategy=sharding_strategy, # 使用动态策略
            # ... 更多参数 ...
        )
```

评论区精华

仅有一条来自 gemini-code-assist[bot] 的 review 评论：指出在 get_sharding_strategy 函数中，当 device_mesh.ndim == 1 时，第 83 行先赋值为 FULL_SHARD 再被第 84 行的 fsdp_strategy 覆盖，存在冗余；同样在 2D mesh 分支（第 86-87 行）也存在类似问题。评论建议移除冗余赋值以提升代码清晰度。该评论状态为未解决（未显示回复），但 PR 最终仍被合并。

- 冗余赋值问题 (style)：未明确回应或修改，但 PR 已合并。

风险与影响

- 风险：
  1. 逻辑冗余风险：函数中存在未清理的冗余行，虽然不影响正确性，但可能在未来维护中造成理解混淆。
  2. 配置语义耦合：reshard_after_forward 原本可能用于控制是否在前向后重新分片，现在被复用为 Zero2/Zero3 开关，若其他逻辑也依赖该值，可能产生隐式依赖。
  3. 缺少测试：变更未包含单元测试或集成测试，无法确保 Zero2 策略在真实训练场景下的正确性（如梯度累积、混合精度等场景下的表现）。
  4. 兼容性：仅针对 FSDP1 引擎，未影响 FSDP2 或其他后端，风险范围有限。 - 影响：影响范围：仅限于使用 verl/workers/engine/fsdp 模块的 FSDP1 训练流程。用户可通过在配置中将 reshard_after_forward 设置为 False 来启用 Zero2 分片，从而减少跨设备通信量、降低显存占用，但可能导致参数更新后的内存碎片等副作用。影响程度：中等——提供了一种新的显存 / 通信权衡选择，但若用户明确配置了 reshard_after_forward 为

False，则行为会发生变化。 - 风险标记：核心逻辑变更，缺少测试覆盖，存在冗余代码

关联脉络

- PR #4659 [fsdp] feat: support zero2 for FSDP1（已合并）: 本 PR 正是将 #4659 中 fsdp_workers 的 Zero2 支持特性迁移到 engine workers 中。
- PR #6386 [fsdp] fix: emit distillation outputs in use_remove_padding=False path: 同为 fsdp 模块的 bugfix PR，涉及 transformer_impl.py，说明该文件是 FSDP 引擎的核心文件，频繁被修改。