PR #7458 完整报告

verl-project/verl

[fsdp] fix: make deferred gradient sync configurable

合并时间：2026-08-28 10:31

原文链接：http://prhub.com.cn/verl-project/verl/pull/7458

执行摘要

- 一句话：FSDP 延迟梯度同步新增开关，实际默认改为 false
- 推荐动作：值得精读，尤其是 _gradient_sync_context 的 getattr 兼容写法与配置默认值在数据类 /YAML 两层的分歧。建议关注合入后默认值调整的决策背景（内存优先），并留意后续是否有统一默认值的跟进 PR。

功能与动机

PR #7095 在实现延迟梯度同步时未暴露文档中承诺的 use_no_sync_for_gradient_accumulation 配置，导致内存受限用户在遇到峰值内存问题时无法关闭该优化。关联 Issue #6010 报告 GRPO + FSDP2 在 Ascend 910B 上 update_actor 极慢，且延迟同步会保留未分片梯度（实测为 fp32），加剧显存压力，因此需要提供显式开关让用户在通信开销与显存占用之间自行权衡。

实现拆解

1. 新增配置字段：在 verl/workers/config/engine.py 的 FSDPEngineConfig 数据类中添加 use_no_sync_for_gradient_accumulation: bool = True 字段，并在 docstring 中说明语义与默认值。这是整个配置契约的数据入口。
2. 改造引擎同步逻辑：在 verl/workers/engine/fsdp/transformer_impl.py 的 _gradient_sync_context 方法中，通过 getattr(self.engine_config, "use_no_sync_for_gradient_accumulation", True) 读取开关，当开关为 False 时，即使非最终 micro-batch 也走常规同步路径，跳过 no_sync()（FSDP1）与 set_requires_gradient_sync(False)（FSDP2）的延迟逻辑。getattr 默认值 True 保证了旧版或子类化引擎配置对象（未定义该字段）仍保持原有的延迟同步行为。
3. 更新 Hydra 配置：在 verl/trainer/config/engine/fsdp.yaml 与生成后的 _generated_ppo_trainer.yaml 中新增配置项，且值均显式设为 false（由维护者 wuxibin89 在合入前通过 commit 调整），使实际运行时默认关闭延迟同步、每 micro-batch 同步一次。
4. 补充 CPU 单元测试：在 tests/workers/test_fsdp_gradient_accumulation_sync_on_cpu.py 中新增参数化测试，覆盖 FSDP1/FSDP2 在开关为 False 时保持同步、以及旧配置对象（无该字段）仍走延迟同步的兼容场景；在 tests/workers/config/test_engine_config_on_cpu.py 中验证 dataclass 默认值与显式构造 False 的行为。

5. 更新性能文档：在 docs/perf/perf_tuning.rst 中说明通信与显存的权衡关系，并给出 actor/critic 的覆盖路径示例。

关键文件：

- verl/workers/engine/fsdp/transformer_impl.py（模块 FSDP 引擎；类别 source；类型 core-logic；符号 _gradient_sync_context）：核心引擎逻辑，_gradient_sync_context 方法增加开关判断，直接控制 FSDP1/FSDP2 的梯度同步时机。
- verl/workers/config/engine.py（模块 配置层；类别 source；类型 configuration；符号 FSDPEngineConfig）：配置数据类新增字段，是整个配置契约的源头，决定用户可见的默认值。
- verl/trainer/config/engine/fsdp.yaml（模块 配置层；类别 config；类型 configuration）：Hydra 配置入口，决定实际运行时默认值，合入时被改为 false，与数据类默认值存在分歧。
- verl/trainer/config/_generated_ppo_trainer.yaml（模块 生成配置；类别 config；类型 configuration）：生成后的扁平配置，影响所有使用该配置模板的 PPO 训练任务。
- tests/workers/test_fsdp_gradient_accumulation_sync_on_cpu.py（模块 单元测试；类别 test；类型 test-coverage；符号 _make_engine, test_gradient_sync_context_keeps_sync_when_disabled, test_gradient_sync_context_defaults_to_deferred_sync_for_legacy_config）：核心测试文件，覆盖开关关闭、legacy 配置兼容、最终 micro-batch 同步等关键路径。
- tests/workers/config/test_engine_config_on_cpu.py（模块 配置测试；类别 test；类型 test-coverage；符号 test_gradient_accumulation_sync_can_be_restored_per_micro_batch）：验证数据类默认值和显式构造 False 的行为。
- docs/perf/perf_tuning.rst（模块 性能文档；类别 docs；类型 documentation）：更新性能调优文档，说明通信与显存权衡及配置路径。

关键符号：_gradient_sync_context, FSDPEngineConfig

关键源码片段

verl/workers/engine/fsdp/transformer_impl.py

核心引擎逻辑，_gradient_sync_context 方法增加开关判断，直接控制 FSDP1/FSDP2 的梯度同步时机。

```
@contextmanager
def _gradient_sync_context(self, *, is_last_micro_batch: bool):
    """控制 FSDP 梯度同步的时机。

    梯度累积期间优化器只在最后一个 micro-batch 后 step，因此理论上只需一次同步。
    延迟同步可将 reduce-scatter 从每 micro-batch 一次降为整个 mini-batch 一次，
    但代价是未分片梯度会被保留，峰值显存上升。

    新增的 `use_no_sync_for_gradient_accumulation` 开关允许内存受限场景关闭该优化：
    关闭后每个 micro-batch 都同步并重新分片，通信量增加但峰值显存降低。
    """
    # 兼容旧配置对象：未定义该字段时沿用延迟同步（默认 true）
```

```
    defer_sync = getattr(
        self.engine_config,
        "use_no_sync_for_gradient_accumulation",
        True,
    )
    if is_last_micro_batch or not defer_sync:
        # 最后一个 micro-batch 或显式关闭时，走常规同步 backward 路径
        yield
        return

    version = fsdp_version(self.module)
    if version == 1:
        # FSDP1 使用 no_sync() 上下文管理器跳过非最终 micro-batch 的同步
        with self.module.no_sync():
            yield
    elif version == 2:
        # FSDP2 通过 set_requires_gradient_sync(False) 关闭同步，并在 finally 中恢复
        self.module.set_requires_gradient_sync(False)
        try:
            yield
        finally:
            self.module.set_requires_gradient_sync(True)
    else:
        # 未知版本时保持同步路径
        yield
```

verl/workers/config/engine.py

配置数据类新增字段，是整个配置契约的源头，决定用户可见的默认值。

```
@dataclass
class FSDPEngineConfig(EngineConfig):
    """FSDP 引擎配置，继承 BaseConfig 提供 DictConfig 接口。

    Args:
        ...
        use_no_sync_for_gradient_accumulation (bool): 是否延迟 FSDP 梯度同步到最后一个
            micro-batch。关闭后每个 micro-batch 都会同步并重新分片，峰值显存更低，
            但通信量增加。默认 True，与 #7095 合入后的行为一致。
        ...
    """

    # ulysses_sequence_parallel_size 为向后兼容保留可变
    _mutable_fields = EngineConfig._mutable_fields | {"ulysses_sequence_parallel_size"}

    # fsdp 专用开关
    wrap_policy: dict[str, Any] = field(default_factory=dict)
    offload_policy: bool = False
    reshard_after_forward: bool = True
    fsdp_size: int = -1
```

```
    forward_prefetch: bool = False
    model_dtype: str = "fp32"
    use_orig_params: bool = False
    mixed_precision: Optional[dict[str, Any]] = None
    ulysses_sequence_parallel_size: int = 1
    entropy_from_logits_with_chunking: bool = False
    entropy_from_logits_chunk_size: int = 2048
    use_torch_compile: bool = True
    entropy_checkpointing: bool = False
    use_no_sync_for_gradient_accumulation: bool = True # 新增字段，默认与旧行为一致
    strategy: str = "fsdp"
    pad_to_length: bool = False
    pad_to_length_bucket: int = 1024
    qat: QATEngineConfig = field(default_factory=QATEngineConfig)
    turbo_config: dict[str, Any] = field(default_factory=dict)

    def __post_init__(self):
        super().__post_init__()
        assert self.strategy in ["fsdp", "fsdp2", "fsdp_turbo"], f"strategy {self.strategy} not
        supported"
```

评论区精华

PR 本身没有 review 评论，但关联 Issue #6010 的评论中，用户 alanhuangyoo 给出了关键量测：延迟同步保留的梯度缓冲是 fp32 而非 bf16，因为 MixedPrecisionPolicy 默认 reduce_dtype=fp32，导致 FSDP2 的 to_accumulated_grad_if_needed 提前返回分支永不触发，每个参数都会被 upcast 到 fp32 保存，显著增加显存。这一反馈很可能促使维护者在合入前将配置默认值从 true 调整为 false（见 commit set use_no_sync_for_gradient_accumulation=false），以默认关闭延迟同步、规避峰值内存风险。

- 延迟同步保留的梯度为 fp32 导致额外显存开销 (performance): 该量测佐证了提供关闭开关的必要性，并可能促使维护者在合入前将默认值调整为 false，优先规避显存风险。

风险与影响

- 风险：
  1. 默认行为变更：合入后实际默认值为 false（YAML 显式覆盖 dataclass 默认 true），与 PR body 声称的“保留当前延迟同步行为”不一致。这会使原本受益于延迟同步的大模型训练在默认配置下通信量增加，update_actor 可能变慢，特别是对通信敏感的场景。
  2. 配置契约分裂：FSDPEngineConfig 数据类默认 true 与 YAML 默认 false 并存，用户通过不同入口构造配置时可能得到不一致的默认行为，容易引发困惑。
  3. 内存 / 通信权衡不可预测：该开关直接作用于 _gradient_sync_context 核心路径，若用户开启延迟同步（true），在长序列、大 batch 下可能触发 OOM，需依赖文档提示。 - 影响：对用户：新增了一个 FSDP 引擎配置项，可通过 Hydra 在 actor、ref、critic 等路径独立控制；默认行为被调整为每 micro-batch 同步，可能改变既有训练脚本的运行表现（内存下降、通信上升）。对系统：FSDP1 与 FSDP2 两条梯度同步路径均被开关

门控，逻辑简单清晰，但默认值变化会传导到所有 FSDP 训练任务。对团队：维护者通过 commit 直接修改默认值，体现了对内存风险的优先取舍，但 PR 描述与最终实现不一致，需要后续文档对齐。 - 风险标记：默认行为变更，通信与内存权衡，配置默认值不一致

关联脉络

- PR #7095 [fsdp] feat: defer gradient sync until final micro-batch: 本 PR 是对 #7095 引入的延迟同步行为补全配置开关，实现同一功能的演进与契约收口。