PR #7362 完整报告

verl-project/verl

[ci, trainer] feat: add fsdpturbo backend engine support.

合并时间：2026-08-17 20:31

原文链接：http://prhub.com.cn/verl-project/verl/pull/7362

执行摘要

- 一句话：新增 fsdp_turbo 后端引擎，支持 FSDP+EP+CP 混合并行
- 推荐动作：值得精读，重点看三处设计：一是 _build_module 中临时把 ulysses_sequence_parallel_size 置 1 以规避 verl monkey patch 与 Turbo CP split 冲突的写法；二是 offload 职责划分（禁用 verl 自有 offload 交由 fsdp_turbo 统一处理，避免双重 offload）；三是 review 推动的 " 新引擎独立成文件 " 组织约定，对 transformer_impl.py 这类大文件的演进有借鉴意义。

功能与动机

PR body 仅一句 "add fsdpturbo backend engine support."，结合新增文档 docs/advance/fsdp_turbo_backend.rst 可推断：fsdp-turbo 是 Ascend 开源的高性能 FSDP 训练库，能在单一 DeviceMesh 上同时支持 fully-sharded DP、expert parallel (EP) 与 context parallel (CP)，并同时覆盖 Ascend NPU 与 NVIDIA GPU。verl 此前只有 fsdp/fsdp2 策略，NPU 侧缺少原生 CP/EP 的 FSDP 训练路径（Megatron/VeOmni 重量较重），本 PR 旨在补齐该后端并形成配置、测试、CI、示例、文档的完整接入闭环。

实现拆解

1. 新引擎类与注册：新增 verl/workers/engine/fsdp/fsdp_turbo_impl.py，定义 FSDPTurboEngineWithLMHead（继承 FSDPEngineWithLMHead），通过 @EngineRegistry.register(model_type="language_model", backend="fsdp_turbo", device=["cuda", "npu"]) 注册；随后在 verl/workers/engine/fsdp/__init__.py 与 verl/workers/engine/__init__.py 中导出，纳入引擎家族。
2. 并行状态与 CP 适配：_init_parallel_state 将 engine_config.turbo_config（dict）翻译为 fsdp_turbo 的 FSDPTurboConfig 并初始化其 parallel_state；_build_module 临时把 ulysses_sequence_parallel_size 置 1，规避 verl 的 Qwen VLM monkey patch 提前对文本模型做 CP 切分；_process_ulysses_config 在 Turbo CP 与 verl Ulysses SP 同时开启时抛 ValueError，并把 Turbo 的 ulysses 分组回填给 verl 的 SP 字段。
3. FSDP 封装、权重加载与 offload 职责划分：_build_fsdp_module 用 FSDPTurbo 包装模块，在 offload_policy/forward_only 下禁用 verl 自有 param/optimizer offload，统一交给 fsdp_turbo 的 CPUOffloadPolicy，再经 fsdp2_load_full_state_dict 恢复权重；数据并行 rank/size/group 与 is_mp_src_rank_with_outputs 在 _parallel_state 存在时优先委托给 fsdp_turbo。

4. 优化器步骤适配：optimizer_step 改用 fsdp_turbo.training.clip_grads.clip_grad_norm 处理 EP 混合 mesh 场景的梯度裁剪，保留 scaler 与 QAT 逻辑；_gradient_sync_context 置空以避免 OOM。
5. 配置契约：FSDPEngineConfig 新增 turbo_config: dict[str, Any] 字段，__post_init__ 的 strategy 断言扩展为 ["fsdp", "fsdp2", "fsdp_turbo"]；verl/trainer/config/engine/fsdp.yaml 与 verl/trainer/config/_generated_ppo_trainer.yaml 补充 turbo_config 全量 schema；verl/utils/tokenizer/continuous_token_wiring.py 有一行 tokenizer 接线调整（评审材料未展开具体细节，推测与 Qwen3.5 图像输入相关）。
6. 测试与 CI：新增 GPU e2e smoke（tests/special_e2e/run_ppo_trainer_fsdp_turbo.sh，Qwen3.5-0.8B，8×L20，单步）与 NPU nightly smoke（tests/special_npu/nightly_ci_ascend/run_grpo_qwen3_5_2b_fsdp_turbo_npu.sh，Qwen3.5-2B，5 步）；新增 .github/workflows/e2e_ppo_trainer_fsdp_turbo_vllm.yml，workflow 内 clone FSDPTurbo 仓库并 export PYTHONPATH=/FSDPTurbo；nightly_ascend.yml 增加 turbo 任务。
7. 示例与文档：新增 examples/grpo_trainer/run_qwen3_5_35b_fsdp_turbo.sh（MoE，EP=8，NPU 下 ep_dispatcher=fused）与 run_qwen3_5_27b_fsdp_turbo.sh（dense），脚本自动探测 GPU/NPU 并设置 HCCL 环境变量；docs/advance/fsdp_turbo_backend.rst 覆盖配置 schema、CP 约束、offload 策略与混合 mesh 梯度裁剪 patch（引用 verl/workers/engine/fsdp/utils.py:apply_clip_grad_norm_patch）。

关键文件：

- verl/workers/engine/fsdp/fsdp_turbo_impl.py（模块 引擎实现；类别 source；类型 core-logic；符号 FSDPTurboEngineWithLMHead, _init_device_mesh, _init_parallel_state, _build_module）：新增引擎核心实现，定义 FSDPTurboEngineWithLMHead 并注册 backend="fsdp_turbo"，包含并行状态初始化、CP/Ulysses 冲突规避、FSDPTurbo 封装、offload 职责划分与梯度裁剪适配，是本 PR 的源码主路径。
- verl/workers/config/engine.py（模块 配置层；类别 source；类型 configuration；符号 FSDPEngineConfig, turbo_config）：FSDPEngineConfig 新增 turbo_config 字段并将 strategy 断言放开到 fsdp_turbo，是本 PR 的配置契约变更点，所有 fsdp_turbo 启动脚本都依赖该字段。
- tests/special_e2e/run_ppo_trainer_fsdp_turbo.sh（模块 GPU 测试；类别 test；类型 test-coverage）：GPU 侧 e2e smoke 脚本，覆盖 Qwen3.5-0.8B dense 模型在 8×L20 上的 fsdp_turbo 训练路径，是 CI 工作流的实际执行体。
- tests/special_npu/nightly_ci_ascend/run_grpo_qwen3_5_2b_fsdp_turbo_npu.sh（模块 NPU 测试；类别 test；类型 test-coverage）：NPU 侧 nightly smoke 脚本，覆盖 Ascend NPU 上的 Qwen3.5-2B GRPO（SP_SIZE=2，启用 CP patch），是 NPU 路径能否持续可用的关键回归。
- .github/workflows/e2e_ppo_trainer_fsdp_turbo_vllm.yml（模块 CI 流水线；类别 infra；类型 infrastructure）：新增 GPU e2e CI 工作流，在 8×L20 上 clone FSDPTurbo 仓库并运行 smoke 脚本，是后端可被持续验证的基础设施。

- examples/grpo_trainer/run_qwen3_5_35b_fsdp_turbo.sh（模块 示例脚本；类别 other；类型 examples）：MoE 示例（Qwen3.5-35B-A3B），展示 EP=8 与 NPU 下 ep_dispatcher=fused 的配置，是 turbo_config 表达能力最强的参考脚本。
- docs/advance/fsdp_turbo_backend.rst（模块 文档；类别 docs；类型 documentation）：独立文档完整说明 fsdp_turbo 后端的配置 schema、CP 约束、offload 语义与混合 mesh 梯度裁剪 patch，是用户上手与后续维护的权威依据。

关键符号：FSDPTurboEngineWithLMHead, _init_device_mesh, _init_parallel_state, _build_module, _build_fsdp_module, _is_ulysses_enabled, _process_ulysses_config, get_data_parallel_rank, get_data_parallel_size, get_data_parallel_group, is_mp_src_rank_with_outputs, _gradient_sync_context, optimizer_step

关键源码片段

tests/special_e2e/run_ppo_trainer_fsdp_turbo.sh

GPU 侧 e2e smoke 脚本，覆盖 Qwen3.5-0.8B dense 模型在 8×L20 上的 fsdp_turbo 训练路径，是 CI 工作流的实际执行体。

```
# tests/special_e2e/run_ppo_trainer_fsdp_turbo.sh（GPU smoke 用例节选）
# 依赖：GPU vllm==0.18.0, transformers@<cc7ab9be>, fsdp_turbo
set -xeuo pipefail

# 给 actor 与 ref 统一声明 turbo_config 前缀，避免两处重复拼写
ACTOR_TURBO="actor_rollout_ref.actor.fsdp_config.turbo_config"
REF_TURBO="actor_rollout_ref.ref.fsdp_config.turbo_config"

# FSDP 切分范围：视觉块、patch_embed、pos_embed、embed_tokens、全部 decoder 层与 lm_
head
FSDP_APPLY_MODULES='{model.visual.blocks.\{*\}:{},'\
'model.visual.patch_embed:{},'\
'model.visual.pos_embed:{},'\
'model.language_model.embed_tokens:{},'\
'model.language_model.layers.\{*\}:{},'\
'lm_head:{}}'

# hook 与重计算都作用在 decoder 层，重计算额外覆盖视觉块
HOOK_MODULES="['model.language_model.layers.{*}']"
RECOMPUTE_PLAN="['model.language_model.layers.{*}','model.visual.blocks.{*}']"

ACTOR=(
    actor_rollout_ref.actor.strategy=fsdp_turbo
    "${ACTOR_TURBO}.distributed.fully_shard_parallel_size=${FSDP_SIZE}"
    "${ACTOR_TURBO}.distributed.ulysses_parallel_size=${SP_SIZE}"
    "+${ACTOR_TURBO}.distributed.fsdp_plan.apply_modules=${FSDP_APPLY_MODULES}"
    "+${ACTOR_TURBO}.distributed.fsdp_plan.hook_modules=${HOOK_MODULES}"
    "+${ACTOR_TURBO}.memory.recompute=True"
    "+${ACTOR_TURBO}.memory.recompute_plan=${RECOMPUTE_PLAN}"
    actor_rollout_ref.actor.fsdp_config.offload_policy=True
```

```
    actor_rollout_ref.actor.fsdp_config.param_offload=True
)

REF=(
    actor_rollout_ref.ref.strategy=fsdp_turbo
    "${REF_TURBO}.distributed.fully_shard_parallel_size=${FSDP_SIZE}"
    "${REF_TURBO}.distributed.ulysses_parallel_size=${SP_SIZE}"
    "+${REF_TURBO}.distributed.fsdp_plan.apply_modules=${FSDP_APPLY_MODULES}"
    "+${REF_TURBO}.memory.recompute=True"
    actor_rollout_ref.ref.fsdp_config.offload_policy=True
)
```

评论区精华

3 条 review 评论全部围绕代码组织：wuxibin89 在 FSDPTurboEngineWithLMHead 最初被追加到 verl/workers/engine/fsdp/transformer_impl.py 末尾的 diff 上要求 "Move to separate file."；pengnuoheng 追问是否移到 verl/workers/engine/fsdp_turbo/transformer_impl.py，并最终确认已拆到 verl/workers/engine/fsdp/fsdp_turbo_impl.py。这条讨论不涉及算法本身，但体现了对 1600+ 行 transformer_impl.py 可维护性的约束，直接决定了最终文件布局：新增引擎独立成文件，仅在 __init__.py 中做导出接线。

- FSDPTurboEngineWithLMHead 应独立成文件 (design): 新类从 1600+ 行的 transformer_impl.py 拆出，独立为 fsdp_turbo_impl.py，并在 fsdp/init.py 与 engine/init.py 中导出。

风险与影响

- 风险：
  1. 运行时强依赖 fsdp_turbo 包：fsdp_turbo 的 import 全部放在函数体内，未安装该包时不会在导入 verl 时报错，但一旦构造 fsdp_turbo 引擎就会 ImportError；与 verl/workers/engine/__init__.py 中 torchtitan/veomni 采用 try/except ImportError 兜底警告的模式不一致，建议后续补上延迟加载或警告。
  2. CP 冲突是显式破坏：Turbo CP 与 verl Ulysses SP 同时开启会直接 ValueError，对沿用 ulysses_sequence_parallel_size>1 旧配置的用户属于显式行为变更，但文档已要求改为 1，属可接受的防护性设计。
  3. offload 语义变化：turbo 引擎下 offload_policy=True 会禁用 verl 自有的 param/optimizer offload 路径，依赖 verl 侧 offload 统计或自定义 checkpoint 逻辑的用户需注意行为差异。
  4. 版本敏感：示例脚本头部 pin 了 vllm==0.18.0、transformers@<cc7ab9be>，NPU smoke 还依赖 vllm-ascend@<54879467>；CP patch 的 target_functions（如 transformers.models.qwen3_5.eager_attention_forward、Qwen3_5GatedDeltaNet.forward）强依赖 transformers 源码结构，升级可能破坏 patch。
  5. 测试深度有限：CI 内只有 Qwen3.5-0.8B 单步 GPU smoke 与 NPU 5 步 smoke；35B MoE 示例未纳入 CI，EP + fused dispatcher 等组合缺少持续回归保障。 - 影响：

用户侧：新增 strategy=fsdp_turbo 入口，GPU 与 Ascend NPU 用户均可一键启用，对 NPU 用户是首个原生支持 EP/CP 的 FSDP 训练路径，利好 MoE（如 35B-A3B）与长序列 CP 训练。系统侧：EngineRegistry 增加一个 backend 分支，FSDPEngineConfig契约新增 turbo_config 子树；CI 新增 1 个 GPU e2e 任务与 1 个 NPU nightly 任务，长期占用 L20×8 资源。团队侧：给出了完整的新后端接入范本（引擎 + 配置 + 测试 +CI + 示例 + 文档），降低后续接入其他训练后端的复制成本。

- 风险标记：新引擎运行时强依赖 fsdp_turbo 包，可选引擎未沿用 try/except 兜底导入，双 CP 冲突显式抛错影响旧配置用户，offload 语义与 verl 自有路径不同，依赖特定 transformers/vllm 版本，仅 smoke 级测试覆盖，MoE 未纳入 CI

关联脉络

- PR #7374 [ci, trainer] feat: remove mindspeedllm backend engine support.: 同期对 Ascend/NPU 引擎家族的减法清理（移除 Mindspeed-LLM），与本次新增 fsdp_turbo 同属 NPU 后端演进脉络，一减一增共同收敛引擎矩阵。
- PR #6804 [BREAKING][rollout] feat: Add Multimodal Continuous Token: 本 PR 修改了 verl/utils/tokenizer/continuous_token_wiring.py，与 Continuous Token 接线的功能线相关，Qwen3.5 多模态示例依赖该接线。
- PR #7407 [megatron,veomni] feat: use torch.int16 for routed_experts: MoE/EP 基础设施演进（routed_experts 与 router replay 重构），与 fsdp_turbo 的 expert_parallel 支持共同服务于超大 MoE 模型训练。
- PR #7291 [ckpt] feat: Node-local multi-sender broadcast in NCCL checkpoint engine: 权重同步性能优化，fsdp_turbo 示例脚本中配置了 update_weights_bucket_megabytes=6144，与快速权重广播能力联动。