PR #5631 完整报告

verl-project/verl

[rollout] feat: enable Async RL for trtllm rollout

合并时间：2026-05-07 15:31

原文链接：http://prhub.com.cn/verl-project/verl/pull/5631

执行摘要

- 一句话：TRTLLM 异步 RL 完整流程实现
- 推荐动作：建议精读 trtllm_async_server.py 中 abort/resume 实现及 _resolve_chat_stop_tokens 函数，理解聊天模型生成结束控制的细节；关注 trtllm_rollout.py 中 standalone 设备网格初始化策略；跟踪后续性能优化 PR 以形成完整评估。

功能与动机

PR body 明确指出 "Requires verl's trtllm version to be updated to include https://github.com/NVIDIA/TensorRT-LLM/pull/12272 [Merged]"，并说明 "this MR only enables e2e async RL functionalities for trtllm rollout and tested convergence"。目的是在 TRTLLM 后端实现与 vllm 对等的异步强化学习训练流程，解决之前只能同步 rollout 的限制，并经过收敛性验证。

实现拆解

1. Abort/Resume 机制：在 verl/workers/rollout/trtllm_rollout/trtllm_async_server.py 中引入 asyncio.Event 控制生成暂停与恢复，映射 TRTLLM 的 pause_generation/resume_generation；新增 clear_kv_cache 实现、start_profile/stop_profile 集成 Nsys Profiler。
2. TorchSampler 停止 token 修复：新增独立函数 _resolve_chat_stop_tokens，自动识别聊天模型附加停止 token（如 <|im_end|>），避免生成进入第二回合导致长度膨胀。
3. Standalone 独立部署模式：在 trtllm_rollout.py 中，当 device_mesh 为 None 时通过 gloo 进程组构建 CPU 设备网格；获取 gpu_id 用于权重同步的 CUDA IPC 句柄创建。
4. 权重更新与前缀缓存：在 trtllm_worker_extension.py 中新增 reset_prefix_cache 方法；添加临时类 RlhfWorkerExtension 替代上游尚未包含的 WorkerExtension（TODO 待 TRTLLM 版本升级后移除）。
5. 环境兼容适配：在 verl/trainer/constants_ppo.py 中根据 GPU 计算代数（SM >= 10）自动禁用 NCCL_NVLS_ENABLE 和 NCCL_MNNVL_ENABLE，解决 GB200 集群 Megatron all_gather 崩溃。
6. 测试与 CI：新增 tests/workers/rollout/rollout_trtllm/test_trtllm_abort.py 端到端测试；新增 .github/workflows/e2e_fully_async_policy_trtllm.yml 专用 CI 工作流；更新

tests/special_e2e/run_fully_async_policy.sh 脚本；补充 docs/workers/trtllm_worker.rst 文档说明。

关键文件：

- verl/workers/rollout/trtllm_rollout/trtllm_async_server.py（模块 TRTLLM 服务；类别 source；类型 core-logic；符号 _resolve_chat_stop_tokens, clear_kv_cache, start_profile, stop_profile）：核心变更文件：新增 _resolve_chat_stop_tokens 函数、clear_kv_cache、start_profile/stop_profile 方法；添加 abort/resume 逻辑通过 _generation_allowed 事件控制生成暂停与恢复；引入 DistProfiler 支持。
- tests/workers/rollout/rollout_trtllm/test_trtllm_abort.py（模块 测试；类别 test；类型 test-coverage；符号 test_trtllm_abort）：新增端到端测试，覆盖 TRTLLM abort/resume 功能启动、中断请求、恢复生成的完整流程，使用 Qwen2.5-1.5B-Instruct 模型。
- verl/workers/rollout/trtllm_rollout/trtllm_rollout.py（模块 Rollout 主类；类别 source；类型 dependency-wiring）：关键文件：添加 standalone 模式设备网格初始化（通过 gloo group）；恢复 gpu_id 获取用于 CUDA IPC 句柄创建；在 resume/release/update_weights 中增加 device_mesh 为空时的 fallback。
- verl/workers/rollout/trtllm_rollout/trtllm_worker_extension.py（模块 权重更新；类别 source；类型 core-logic；符号 reset_prefix_cache, RlhfWorkerExtension, wait_for_engine_idle）：新增 reset_prefix_cache 方法确保权重更新后清除前缀缓存；添加临时类 RlhfWorkerExtension 替代上游未包含的 WorkerExtension，提供 wait_for_engine_idle 接口。
- .github/workflows/e2e_fully_async_policy_trtllm.yml（模块 CI 配置；类别 infra；类型 infrastructure）：新增独立 CI 工作流，为 TRTLLM fully_async 策略提供自动化测试，包含多 replica 与单 replica 变体，监控 trtllm_rollout 和 fully_async_policy 路径变更。
- verl/trainer/constants_ppo.py（模块 环境配置；类别 source；类型 dependency-wiring）：根据 GPU 计算代数自动禁用 NCCL NVLS/MNNVL，解决 GB200 集群上 Megatron all_gather 的 ncclUnhandledCudaError 崩溃。
- verl/workers/rollout/replica.py（模块 副本管理；类别 source；类型 entrypoint）：修改 standalone 模式初始化资源池的方式，允许独立控制 use_gpu 和 max_colocate_count 参数，满足 TRTLLM 特殊进程结构需求。

关键符号：_resolve_chat_stop_tokens, clear_kv_cache, start_profile, stop_profile, reset_prefix_cache, wait_for_engine_idle, test_trtllm_abort

关键源码片段

verl/workers/rollout/trtllm_rollout/trtllm_async_server.py

核心变更文件：新增 _resolve_chat_stop_tokens 函数、clear_kv_cache、start_profile/stop_profile 方法；添加 abort/resume 逻辑通过 _generation_allowed 事件控制生成暂停与恢复；引入 DistProfiler 支持。

```
# verl/workers/rollout/trtllm_rollout/trtllm_async_server.py

from verl.utils.profiler import DistProfiler
```

```
def _resolve_chat_stop_tokens(model_config) -> tuple[int, list[int]]:
    """Return (end_id, stop_token_ids) for TorchSampler.

    Both TRTLLM's samplers stops only on end_id.  For chat-format prompts the model
    naturally ends each assistant turn with a chat-end token (e.g. <|im_end|>
    for Qwen, <|eot_id|> for Llama-3) that is *different* from the base-model
    eos_token_id.  If end_id is set to the base eos the sampler ignores the
    chat-end token and the model loops into a second turn, inflating response
    lengths until max_tokens is hit.

    For models without a distinct chat-end token the return values are
    identical to the current default (end_id = hf_config.eos_token_id).
    """
    eos_token_id = model_config.hf_config.eos_token_id
    # 统一为列表形式
    all_stop_ids: list[int] = list(eos_token_id) if isinstance(eos_token_id, list) else [eos_token_id]

    # 合并 generation_config 中的额外 eos token
    if model_config.generation_config is not None:
        gen_eos = model_config.generation_config.eos_token_id
        if gen_eos is not None:
            for t in gen_eos if isinstance(gen_eos, list) else [gen_eos]:
                if t not in all_stop_ids:
                    all_stop_ids.append(t)

    chat_end_id = None
    # 检测聊天模板特有停止 token，如 <|im_end|>、<|eot_id|>
    if model_config.tokenizer is not None:
        _chat_stop_strings = ["<|im_end|>", "<|eot_id|>", "<|end_of_turn|>"]
        _added_vocab = model_config.tokenizer.get_added_vocab()
        for stop_str in _chat_stop_strings:
            if stop_str in _added_vocab:
                tid = _added_vocab[stop_str]
                if tid not in all_stop_ids:
                    all_stop_ids.append(tid)
                if chat_end_id is None:
                    chat_end_id = tid # 将首个 match 的 chat-end id 设为 end_id，防止二次生成

    primary_end_id = chat_end_id if chat_end_id is not None else eos_token_id
    logger.warning(f"TRT-LLM stop token IDs: {all_stop_ids}, end_id: {primary_end_id}")
    return primary_end_id, all_stop_ids
```

verl/workers/rollout/trtllm_rollout/trtllm_worker_extension.py

新增 reset_prefix_cache 方法确保权重更新后清除前缀缓存；添加临时类 RlhfWorkerExtension 替代上游未包含的 WorkerExtension，提供 wait_for_engine_idle 接口。

```
# verl/workers/rollout/trtllm_rollout/trtllm_worker_extension.py

    def reset_prefix_cache(self) -> None:
```

```
        """Invalidate the KV cache prefix reuse state after weight updates."""
        # 权重更新后必须清除前缀缓存，否则旧前缀失效导致错误复用
        self.engine.reset_prefix_cache()


# TODO: remove this class and revert the non-VLM path in trtllm_async_server.py
# to use "tensorrt_llm.llmapi.rlhf_utils.WorkerExtension" once verl's TRT-LLM version
# is bumped to include https://github.com/NVIDIA/TensorRT-LLM/pull/13784.
class RlhfWorkerExtension(TrtllmWorkerExtension):
    """Minimal extension of TRT-LLM's WorkerExtension for non-VLM RLHF models."""

    @control_action_decorator
    def wait_for_engine_idle(self) -> None:
        """Block until the engine has no active or queued requests."""
        # TRTLLM 引擎在无请求时自动空闲，此处无需额外等待
        pass
```

评论区精华

1. 测试配置路径合理性（gemini-code-assist）：指出 config_dir 依赖当前工作目录，建议用 pathlib 确定项目根。hchings 回复为沿用现有测试惯例，未修改。
2. Ray 集群清理方式（gemini-code-assist）：建议不要使用 subprocess.run(['ray', 'stop']) 以免干扰并行测试。hchings 认为测试环境无现有 Ray 集群，两者并用无妨，保留。
3. Standalone 模式 use_gpu 设计（wuxibin89 ↔ hchings）：wuxibin89 对 replica.py 中新增 _standalone_use_gpu 表示困惑，建议由 rollout_mode 推导。hchings 解释 TRTLLM 的 Ray 进程结构与 vllm 不同，需独立控制。最终 wuxibin89 仍要求简化，但未完全达成一致。
4. GB200 NCCL 环境变量范围（wuxibin89）：担心 get_device_capability 在 Ascend NPU 上报错，后因 Ascend CI 通过而认为安全。
5. 环境变量传播（tongyuantongyu）：建议转发所有 TLLM_ 前缀环境变量给 Ray Actor。hchings 承诺后续 MR 处理，CI 已通过。
6. CI 任务精简（wuxibin89）：建议只保留 multi-replica 测试，移除 single-replica 和 fsdp2。hchings 在后续提交中调整。

- 测试配置路径的健壮性 (testing): hchings 回复为沿用现有测试惯例，未修改。
- Ray 集群清理方式 (testing): hchings 认为测试环境无已有 Ray 集群，两者并用可接受，保留。
- Standalone 模式 use_gpu 设计 (design): wuxibin89 仍要求简化，但未达成一致，最终 approved。
- GB200 NCCL 环境变量影响范围 (correctness): Ascend CI 通过，wuxibin89 认为安全，不再修改。
- TLLM_ 前缀环境变量传播 (design): hchings 承诺在后续 MR 中处理，当前 CI 已通过。

风险与影响

- 风险：
  1. 外部依赖版本：需 TRTLLM 包含 #12272 补丁，否则功能不可用；临时类 RlhfWorkerExtension 依赖未来版本 #13784。
  2. GB200 环境变量影响：NCCL_NVLS_ENABLE=0 和 NCCL_MNNVL_ENABLE=0 可能覆盖用户自定义环境，影响非 Blackwell GPU 的 NCCL 性能。
  3. Abort/Resume 状态管理：生成暂停与恢复的时序并发控制可能引入竞态，导致请求丢失或重复。
  4. Standalone 模式较新：未像 colocated 模式经过广泛验证，可能暴露资源调度或进程生命周期问题。
  5. Rollout 性能瓶颈：PR 自知存在 Python 侧 _update_requests 开销（尤其 TorchSampler），可能拖慢整体训练吞吐。
- 影响：
  1. 用户影响：TRTLLM 用户可启用异步 RL（设置 rollout.mode=async），但需更新 TRTLLM 镜像至包含所需 commit；现有同步 rollout 不受影响。
  2. 系统影响：新增 CI 工作流增加 GPU 资源消耗；constants_ppo.py 中环境变量生效范围需监控，避免 Blackwell 以外集群性能下降。
  3. 团队影响：后续需跟踪 TRTLLM 版本升级移除临时 hack（RlhfWorkerExtension、_standalone_use_gpu），并推进 rollout 性能优化。 - 风险标记：外部依赖性变更，GB200 专用 NCCL 配置，核心状态管理 (abort/resume)，Standalone 模式稳定度，Rollout 性能瓶颈未优化

关联脉络

- PR #6056 [fully_async, rollout] feat: enable online policy distillation in fully async training: 同属 fully_async 实验目录，本 PR 为其 TRTLLM 后端提供异步 rollout 基础，二者共同构成完整异步训练流程。
- PR #6230 [rollout] fix: trtllm rollout docker image and a few scripts: 同一仓库 trtllm rollout 系列修复，涉及 CI 脚本和 Docker 映像，与本 PR 新增的 CI 工作流相关。
- PR #12272 NVIDIA/TensorRT-LLM#12272 (external): 本 PR 依赖的 TRTLLM 上游补丁，启用 AsyncLLM 的 abort/resume 能力。