PR #46433 完整报告

vllm-project/vllm

[XPU] Optimize XPU worker shutdown logic to prevent resource leak

合并时间：2026-06-30 15:27

原文链接：http://prhub.com.cn/vllm-project/vllm/pull/46433

执行摘要

- 一句话：优化 XPU Worker 关闭逻辑，防止资源泄漏
- 推荐动作：此 PR 对于在 Intel XPU 上部署 vLLM 的用户是必须合入的修复，建议详细阅读 xpu_worker.py 和 gpu_model_runner.py 的 shutdown 实现，理解跨平台资源清理的设计模式。

功能与动机

XPU 上默认 shutdown_timeout=0 导致 worker 被 SIGKILL 强制终止，不执行任何清理代码，Level Zero 驱动资源和 oneCCL 通信组未释放，再次启动相同设备上的 vLLM 服务器时，oneCCL all_reduce 会永久挂起。PR body 明确描述了可复现步骤。

实现拆解

1. 平台配置调整：在 vllm/platforms/xpu.py 的 XpuPlatform.check_and_update_config 中，当 shutdown_timeout == 0 时将其改为 5，启用 drain 模式使 worker 有足够时间执行清理（附带 info 日志）。
2. Worker 关闭方法：在 vllm/v1/worker/xpu_worker.py 中新增 shutdown 方法，先调用 super().shutdown() 触发父类（GPUWorker）的资源释放，再通过 XpuMemAllocator.instance.release_pools() 释放 XPU 专用内存池。
3. 模型运行器清理扩展：在 vllm/v1/worker/gpu_model_runner.py 的 shutdown 方法中，将原本仅对 ROCm 执行的 gc.collect()、torch.accelerator.empty_cache()、torch.accelerator.synchronize() 的条件改为 is_rocm() or is_xpu()，确保 XPU 也能彻底回收 GPU 缓存。
4. 通用 allocator 保护：在 vllm/v1/worker/gpu_worker.py 的 shutdown 中，将 CuMemAllocator.release_pools 用 current_platform.is_cuda_alike() 包裹，避免在非 CUDA 平台（如 CPU）上导入调用失败。
5. 测试工具增强：在 tests/utils.py 的 _get_gpu_memory_used 中添加 is_xpu() 分支，使用 torch.xpu.mem_get_info 计算已用显存，使测试框架能正确汇报 XPU 显存用量。
6. CI 配置微调：在 .buildkite/intel_jobs/lora_intel.yaml 中重新排列 LoRA 测试顺序，增加 test_llama_tp.py 的显式调用；在 basic_correctness.yaml 中新增一行测试，避免因 worker 关闭问题导致 CI 挂起。

关键文件：

- vllm/v1/worker/xpu_worker.py（模块 XPU Worker；类别 source；类型 core-logic；符号 shutdown）：新增 XPU 专属 shutdown 方法，调用父类清理并释放 XpuMemAllocator 内存池，是本次修复的核心入口。
- vllm/platforms/xpu.py（模块 XPU 平台；类别 source；类型 core-logic）：平台配置中强制设置 shutdown_timeout=5，是 drain 模式能够生效的前提，防止 worker 被直接强杀。
- vllm/v1/worker/gpu_model_runner.py（模块 GPU 模型运行器；类别 source；类型 data-contract）：将 ROCm 专用的资源回收步骤扩展到 XPU，确保模型运行器关闭后 GPU 缓存和同步操作在 XPU 上也被执行。
- vllm/v1/worker/gpu_worker.py（模块 GPU Worker；类别 source；类型 dependency-wiring）：修改 shutdown 中 CuMemAllocator 的访问条件，用 is_cuda_alike 保护避免 CPU 平台报错。
- tests/utils.py（模块 测试工具；类别 test；类型 test-coverage）：为测试工具添加 XPU 显存查询分支，使集成测试能正确检测 XPU 内存泄漏。
- .buildkite/intel_jobs/lora_intel.yaml（模块 CI 配置；类别 config；类型 configuration）：调整 CI 中 LoRA 测试的执行顺序和范围，确保关闭逻辑改动后测试稳定性。
- .buildkite/intel_jobs/basic_correctness.yaml（模块 CI 配置；类别 config；类型 configuration）：增加 basic_correctness 测试条目以验证关闭流程正确性。

关键符号：XPUWorker.shutdown, GPUModelRunner.shutdown, XpuPlatform.check_and_update_config

关键源码片段

vllm/v1/worker/xpu_worker.py

新增 XPU 专属 shutdown 方法，调用父类清理并释放 XpuMemAllocator 内存池，是本次修复的核心入口。

```
# vllm/v1/worker/xpu_worker.py
# 在 profile 方法后新增 shutdown 方法，用于在 drain 模式下安全释放 XPU 资源。
def shutdown(self) -> None:
    logger.info(
        "XPUWorker shutdown: cleaning up (rank=%d, local_rank=%d)",
        self.rank,
        self.local_rank,
    )
    # 调用父类 GPUWorker.shutdown()，释放分布式环境和模型运行器资源
    super().shutdown()
    # 延迟导入 XpuMemAllocator，避免初始化时加载依赖
    from vllm.device_allocator.xpumem import XpuMemAllocator

    # 显式释放 XPU 内存池，确保 Level Zero 驱动资源可被后续进程重用
    if XpuMemAllocator.instance is not None:
        XpuMemAllocator.instance.release_pools()
    logger.info(
        "XPUWorker shutdown: done (rank=%d, local_rank=%d)",
        self.rank,
```

```
            self.local_rank,
        )
```

vllm/platforms/xpu.py

平台配置中强制设置 shutdown_timeout=5，是 drain 模式能够生效的前提，防止 worker 被直接强杀。

```
# vllm/platforms/xpu.py
# 在 check_and_update_config 方法末尾，设置 spawn 模式之后
# XPU 需要优雅关闭以释放 oneCCL/Level Zero 资源，
# 否则后续在同一设备上启动服务会在 CCL 初始化时挂起。
if vllm_config.shutdown_timeout == 0:
    vllm_config.shutdown_timeout = 5
    logger.info(
        "XPU platform: set server shutdown_timeout=%d.",
        vllm_config.shutdown_timeout,
    )
```

vllm/v1/worker/gpu_model_runner.py

将 ROCm 专用的资源回收步骤扩展到 XPU，确保模型运行器关闭后 GPU 缓存和同步操作在 XPU 上也被执行。

```
# vllm/v1/worker/gpu_model_runner.py
# 在 shutdown 方法末尾的清理部分
# ... 释放模型、ROPE 字典等 ...
reset_workspace_manager()
# 对 ROCm 和 XPU 都需要显式执行 GC、清空缓存和设备同步，
# 否则后续进程可能遇到资源残留导致的错误。
if current_platform.is_rocm() or current_platform.is_xpu():
    gc.collect()
    torch.accelerator.empty_cache()
    torch.accelerator.synchronize()
```

评论区精华

核心讨论围绕 gpu_worker.py 中如何安全地释放内存池。

- yma11建议使用 get_mem_allocator_instance 来覆盖 XPU 路径，避免平台硬编码。
- zhenwei-intel指出直接调用 get_mem_allocator_instance 在 CPU 平台会抛出运行时错误，需要添加平台判断。
- jikunshang要求进一步确认，最终方案改为在 gpu_worker.py 的 shutdown 中用 is_cuda_alike() 条件保护 CuMemAllocator.release_pools，并在 xpu_worker.py 中手动调用 XpuMemAllocator.release_pools。
- CuMemAllocator.platform 安全保护 (design): 在 gpu_worker.py 的 shutdown 中用 if current_platform.is_cuda_alike() 保护 CuMemAllocator.release_pools，xpu_worker.py 中直接调用 XpuMemAllocator.release_pools。

风险与影响

- 风险：
  1. 平台兼容性：gpu_worker.py 中的 is_cuda_alike() 判断可能漏掉未来的非 CUDA 平台，需要持续维护。
  2. 配置覆盖：xpu.py 强制覆盖用户手动设置的 shutdown_timeout=0，若用户明确不想要 drain 模式则会受影响，但 PR 认为 XPU 必须 drain。
  3. 性能影响：增加的 gc.collect 和 synchronize 在 shutdown 时调用，不影响在线推理性能。
  4. 测试覆盖：测试工具增强仅支持显存查询，未新增 shutdown 流程的专用测试。 - 影响：用户影响：所有使用 Intel XPU 硬件的 vLLM 用户将受益，不再因重复启动服务而挂死，资源释放更彻底。系统影响：drain 模式使 worker 关闭耗时增加约 5 秒（可配置），但对常规重启流程无负面影响。团队影响：统一了 GPUWorker 和 ModelRunner 中平台相关的清理逻辑，后续其他平台可参考该模式扩展。
- 风险标记：配置覆盖可能性，平台兼容性维护负担

关联脉络

- 暂无明显关联 PR