PR #40812 完整报告

vllm-project/vllm

Auto-disable expandable_segments around cumem memory pool

合并时间：2026-04-27 09:37

原文链接：http://prhub.com.cn/vllm-project/vllm/pull/40812

执行摘要

- 一句话：自动禁用 expandable_segments 以兼容 cumem 内存池
- 推荐动作：建议精读此 PR，特别是 use_memory_pool 的 try/finally 重构和异常安全处理。设计上值得关注的是：在框架 API 不完善时，用环境变量加私有 API 实现临时开关；以及通过嵌套上下文管理器保持全局状态一致性的模式。

功能与动机

PyTorch issue 147851 指出 expandable segments 与 CUDAPluggableAllocator + MemPool 不兼容，导致 vLLM 的 sleep 模式无法与 expandable segments 同时使用。之前用 assert 硬性拒绝，用户只能二选一。此 PR 改为自动临时禁用，无需用户手动切换。

实现拆解

1. 移除 __init__ 中的硬性断言：删除了 CuMemAllocator.__init__ 中对 expandable_segments:True 的 assert，允许对象正常创建。
2. 在 use_memory_pool 入口检测并临时禁用 expandable_segments：进入上下文管理器时，解析 PYTORCH_CUDA_ALLOC_CONF 环境变量，若包含 expandable_segments:True 则调用 torch.cuda.memory._set_allocator_settings("expandable_segments:False") 临时关闭。
3. 在 finally 块中恢复 expandable_segments 和 current_tag：嵌套 try/finally 确保即使上下文内部抛出异常，也能恢复 self.current_tag 和 expandable segments 设置（若之前启用则恢复为 True）。
4. 异常安全与状态一致性：审查中建议将 self.current_tag 恢复从 try 块末尾移到 finally 块，防止 yield 阶段异常导致状态泄漏。

关键文件：

- vllm/device_allocator/cumem.py（模块 分配器；类别 source；类型 core-logic；符号 CuMemAllocator.init, CuMemAllocator.use_memory_pool）：核心变更文件，移除硬性断言并实现 expandable_segments 的临时禁用与恢复逻辑。

关键符号：CuMemAllocator.init, CuMemAllocator.use_memory_pool

关键源码片段

vllm/device_allocator/cumem.py

核心变更文件，移除硬性断言并实现 expandable_segments 的临时禁用与恢复逻辑。

```
# vllm/device_allocator/cumem.py

class CuMemAllocator:
    # ... 其他代码省略 ...

    def __init__(self):
        # 移除了原来对 expandable_segments:True 的 assert 检查
        self.pointer_to_data: dict[int, AllocationData] = {}
        self.current_tag: str = CuMemAllocator.default_tag
        self.allocator_and_pools: dict[str, Any] = {}
        self.python_malloc_callback = self._python_malloc_callback
        self.python_free_callback = self._python_free_callback

    @contextmanager
    def use_memory_pool(self, tag: str | None = None):
        if tag is None:
            tag = CuMemAllocator.default_tag
        assert isinstance(tag, str)

        # 进入内存池上下文前，先检查并临时关闭 expandable_segments
        conf = os.environ.get("PYTORCH_CUDA_ALLOC_CONF", "")
        expandable_was_enabled = "expandable_segments:True" in conf
        if expandable_was_enabled:
            # 调用 PyTorch 私有 API 临时关闭
            torch.cuda.memory._set_allocator_settings("expandable_segments:False")

        old_tag = self.current_tag
        self.current_tag = tag
        try:
            with use_memory_pool_with_allocator(
                self.python_malloc_callback,
                self.python_free_callback
            ) as data:
                # 保持对 data 的引用避免 PyTorch 2.6 gc 问题
                self.allocator_and_pools[tag] = data
                yield
                # 清理未使用分配 ...
                allocations = data[0].snapshot()
                for allocation in allocations:
                    if allocation["allocated_size"] == 0:
                        handle = self._python_free_callback(allocation["address"])
                        unmap_and_release(handle)
        finally:
            # 始终恢复 current_tag 和 expandable_segments 设置
            self.current_tag = old_tag
            if expandable_was_enabled:
```

torch.cuda.memory._set_allocator_settings("expandable_segments:True")

评论区精华

1. 检测手段的健壮性：gemini-code-assist[bot] 指出直接解析环境变量字符串（"expandable_segments:True" in conf）脆弱，无法覆盖空格、大小写、_set_allocator_settings 等变体，建议使用 torch.cuda.memory.get_allocator_settings()。但作者 youkaichao 实测发现该 API 在 PyTorch 2.11 中不存在，已回退到环境变量解析。
2. self.current_tag 的恢复位置：gemini-code-assist[bot] 建议将 self.current_tag = old_tag 放到 finally 块确保异常安全，作者采纳并已在最终提交中实现。

- expandable_segments 检测手段的健壮性 (correctness): 保留环境变量解析方案，但存在漏检测风险。
- self.current_tag 恢复应放入 finally 块 (correctness): 作者采纳，在最终提交中已将 self.current_tag 恢复和 expandable_segments 恢复都放入 finally 块。

风险与影响

- 风险：
  1. 环境变量解析不完整：当前仅检查 "expandable_segments:True" in conf，PyTorch 实际支持 true/True/1 及空格变体，可能出现漏判或误判。
  2. _set_allocator_settings 兼容性：该 API 是私有方法，未来 PyTorch 版本可能变更或删除。
  3. 单测覆盖缺失：PR body 提到需要手动测试，但未包含自动化测试，回归风险存在。 - 影响：影响范围：所有使用 cumem allocator sleep/wake 模式的 vLLM 部署。用户影响：用户现在可以同时启用 expandable_segments:True 和 sleep 模式，无需手动取舍，减少 OOM 风险。性能：临时禁用 expandable segments 仅在内存池上下文内生效，对整体性能影响极小。兼容性：向下兼容，未变更公共 API。
- 风险标记：缺少测试覆盖，依赖 PyTorch 私有 API

关联脉络

- PR #11743 [cumem allocator] Sleep mode for preemption-free scheduler: 引入 cumem allocator 和 sleep 模式，是此 PR 的上游依赖。
- PR #41268 [UX][Bugfix] Fix OOM by setting PyTorch max_split_size_mb during model loading: 同样是处理 PyTorch CUDA 内存分配配置和 OOM 的近期 PR，属于同一条优化线。