PR #40427 完整报告

vllm-project/vllm

[Platform] Fix RISC-V platform detection (lscpu parsing + non-NUMA meminfo)

合并时间：2026-04-24 12:33

原文链接：http://prhub.com.cn/vllm-project/vllm/pull/40427

执行摘要

- 一句话：修复 RISC-V 平台检测与 lscpu 解析问题
- 推荐动作：值得精读的部分：_get_cpu_list() 中 lscpu 输出的正则处理策略，以及如何优雅地处理不完全的拓扑信息。建议关注 _synthesize_cpu_list 的引入位置，它为后续可能出现的其他弱拓扑平台提供了复用基础。

功能与动机

RISC-V 架构在之前无法正确识别，平台解析为 UnspecifiedPlatform，且 lscpu 输出格式与 x86/ARM 不同，导致 CPU 资源初始化失败，阻碍 RISC-V 设备使用 vLLM。

实现拆解

1. 提取合成 CPU 列表函数(vllm/utils/cpu_resource_utils.py)：将原来 macOS 专用的合成逻辑抽取为 _synthesize_cpu_list()，返回每个逻辑 CPU 独占 core 且 NUMA node 为 0 的拓扑。该函数同时用于 lscpu 输出完全无效时的回退。
2. 增强 lscpu 正则处理：将裸 - 字段分两步处理：先映射 "node": - 为 "node": 0（保持非 NUMA 系统行为），再将 "cpu"/"core": - 引用为字符串，使 JSON 解析时转为 -1，随后被过滤器剔除，触发回退。同时恢复了 --online 标志（之前被误删）。
3. 非 NUMA meminfo 回退(get_memory_node_info)：当 /sys/devices/system/node/node0/meminfo 不存在时（常见于 RISC-V 开发板），改用 psutil.virtual_memory() 返回系统级内存信息，避免 RuntimeError。
4. 修正 setup.py 版本后缀：get_vllm_version() 中 CPU 分支改用本地变量 VLLM_TARGET_DEVICE（已被自动检测修改），而非直接读取环境变量，确保 CPU-only 主机（如 RISC-V）正确得到 +cpu 后缀。
5. 删除多余异常捕获(vllm/platforms/__init__.py)：移除了 cpu_platform_plugin() 中的 except Exception 子句（仅一行），简化代码。

关键文件：

- vllm/utils/cpu_resource_utils.py（模块 资源工具；类别 source；类型 core-logic；符号 _synthesize_cpu_list, _get_cpu_list, get_memory_node_info）：核心变更文件，包含 lscpu 解析增强、合成 CPU 函数提取、非 NUMA meminfo 回退，对 RISC-V 平台初始化至关重要。

- setup.py（模块 构建脚本；类别 source；类型 core-logic）：修正版本后缀逻辑，确保 CPU-only 主机（如 RISC-V）正确获得 +cpu 后缀，是平台检测的关键一环。
- vllm/platforms/__init__.py（模块 平台检测；类别 source；类型 core-logic）：删除了多余的异常捕获，简化代码（但在最终方案中 RISC-V 检测通过 setup.py 实现，此文件改动较小）。

关键符号：_synthesize_cpu_list, _get_cpu_list, get_memory_node_info, get_vllm_version

关键源码片段

vllm/utils/cpu_resource_utils.py

核心变更文件，包含 lscpu 解析增强、合成 CPU 函数提取、非 NUMA meminfo 回退，对 RISC-V 平台初始化至关重要。

```
# 从 _get_cpu_list 中提取出的合成函数，用于 macOS 和 lscpu 无效时回退
def _synthesize_cpu_list() -> list[LogicalCPUInfo]:
    """Synthesize a flat CPU list: each logical CPU is its own core on
    NUMA node 0.  Used when lscpu output is unavailable or unparsable
    (e.g. macOS, RISC-V)."""
    cpu_count = os.cpu_count()
    assert cpu_count
    # 每个逻辑 CPU 单独一个 core，NUMA 节点统一为 0（单节点）
    return [LogicalCPUInfo(i, i, 0) for i in range(cpu_count)]


def _get_cpu_list() -> list[LogicalCPUInfo]:
    if platform.system() == "Darwin":
        # macOS 直接合成
        return _synthesize_cpu_list()

    lscpu_output = subprocess.check_output(
        "lscpu --json --extended=CPU,CORE,NODE --online", shell=True, text=True
    )

    # 第一遍：将裸的 "node": - 替换为 "node": 0，保持非 NUMA 系统行为
    lscpu_output = re.sub(r'"node":\s*-\s*(,|\n|\})', r'"node": 0\1', lscpu_output)

    # 第二遍：将 "cpu" 或 "core" 字段的裸 - 引用为字符串 "-",
    # 这样 JSON 解析后会变成 -1，后续过滤掉，触发合成回退
    lscpu_output = re.sub(
        r'("(?:cpu|core)":\s*)-\s*(,|\n|\})',
        r'\1"-"\2',
        lscpu_output,
    )

    logical_cpu_list: list[LogicalCPUInfo] = json.loads(
        lscpu_output, object_hook=LogicalCPUInfo.json_decoder
    )["cpus"]
```

```
    # 过滤掉含有 -1 的无效条目（id, physical_core, numa_node 任一为 -1 则丢弃）
    logical_cpu_list = [
        x for x in logical_cpu_list if -1 not in (x.id, x.physical_core, x.numa_node)
    ]

    # 如果没有任何有效条目（例如 RISC-V），则回退到合成拓扑
    if not logical_cpu_list:
        logical_cpu_list = _synthesize_cpu_list()

    return logical_cpu_list
```

setup.py

修正版本后缀逻辑，确保 CPU-only 主机（如 RISC-V）正确获得 +cpu 后缀，是平台检测的关键一环。

```
    elif _is_cpu():
        # 使用本地变量 VLLM_TARGET_DEVICE（可能已被前面的 auto-detect 修改），
        # 而不是直接读环境变量 envs.VLLM_TARGET_DEVICE，
        # 这样 CPU-only 主机（包括 RISC-V）仍然能获取 +cpu 后缀
        if VLLM_TARGET_DEVICE == "cpu":
            version += f"{sep}cpu"
```

评论区精华

- gemini-code-assist[bot] 指出两个高优先级回归：一是去掉了 --online 标志，可能引入离线 CPU；二是正则修改导致非 NUMA 系统的 node 字段被错误引用为字符串，使所有 CPU 被过滤。作者在 commit cc682c316 中通过分两步处理和恢复 --online 解决了这两个问题。
- bigPYJ1151 建议通过 setup.py 添加 cpu 后缀：作者最初在 platforms/__init__.py 中添加了 RISC-V 检测，但 reviewer 指出更根本的解法是修正 setup.py 中版本后缀判断逻辑。作者采纳并改用了本地变量。
- 关于 lscpu 在 RISC-V 上完全无效的疑问：bigPYJ1151 质疑 lscpu 是否完全不可用。作者解释 RISC-V 内核未暴露拓扑 sysfs 节点，lscpu --json --extended 所有字段均为 -，因此合成回退是合理的。
  - 修复 --online 标志的回归 (correctness): 作者在提交 cc682c316 中恢复了 --online 标志。
  - 正则处理非 NUMA 系统 node 字段 (correctness): 作者将正则拆分为两遍：第一遍专门处理 "node": - 映射到 0，第二遍处理 "cpu"/"core": - 引用为字符串，确保非 NUMA 系统行为不变。
  - 使用 setup.py 而非运行时检测 RISC-V (design): 采用 reviewer 建议，修改 setup.py 中 get_vllm_version() 使用本地变量，并移除了 platforms/__init__.py 中最初的运行时检测代码。
  - RISC-V 上 lscpu 完全无效的合理性 (question): 作者解释 RISC-V 内核未暴露 sysfs 拓扑节点，lscpu 输出中所有字段均为 -，因此回退到合成拓扑是合理的。

风险与影响

- 风险：
  - 回归风险：正则修改可能影响其他架构的 lscpu 输出解析，但已通过分步处理和单元测试验证（作者提及对两个样本进行了测试）。
  - 性能影响：非 NUMA 系统的 meminfo 回退使用 psutil，增加一次系统调用，但仅在 meminfo 文件缺失时才触发，影响极小。
  - 兼容性：修改了 _get_cpu_list() 的返回路径，所有 Linux 平台均受影响。但合成回退只在无有效条目时触发，理论上不会改变现有行为。不过，如果某些平台的 lscpu 输出中包含部分有效条目，原有逻辑可能不再适用（但现有过滤逻辑未变）。
- 影响：
  - 用户影响：RISC-V 设备用户现在可以正常使用 CPU 后端，不再因平台检测失败或初始化错误而无法启动。非 NUMA 系统（如部分 ARM 开发板）也受益于 meminfo 回退。
  - 系统影响：代码变更集中在 CPU 资源工具函数，不影响 GPU 或其他加速器后端。
  - 团队影响：低。该 PR 由外部贡献者提交，审核过程简洁，没有引发大量讨论。
  - 风险标记：lscpu 解析正则回归风险，非 NUMA 系统 meminfo 回退路径，合成拓扑可能丢失核心信息

关联脉络

- PR #39781 [Platform] Fix non-NUMA node handling in lscpu parsing: 本 PR 中 " node": - 映射为 0 的逻辑继承自 #39781，并进一步增强。
- PR #40161 [Platform] Add --online flag to lscpu command: 本 PR 早期版本误删了 #40161 添加的 --online 标志，后在 review 中恢复。