PR #35687 完整报告

vllm-project/vllm

[Bugfix] Treat as implicit reasoning end in Qwen3 parser

合并时间：2026-04-24 09:10

原文链接：http://prhub.com.cn/vllm-project/vllm/pull/35687

执行摘要

- 一句话：修复 Qwen3.5 推理中工具调用被隐式丢弃的问题
- 推荐动作：该 PR 修复了真实用户场景中的工具调用丢失问题，代码实现清晰、测试充分，建议合并。设计上值得关注的是：通过反向遍历序列并排除配对标记，巧妙区分了模型输出与提示模板中的相同标记。

功能与动机

Qwen3.5 models sometimes emit inside the block without closing first. When this happens, Qwen3ReasoningParser.extract_reasoning() classifies the entire output as reasoning, the tool parser receives empty content, and the tool call is silently dropped. This is the same class of issue fixed for Kimi K2 in #33646.

实现拆解

1. 初始化工具调用标记：在 __init__ 中，从词汇表获取 <tool_call> 和 </tool_call> 的 token ID，存储为 _tool_call_token_id 和 _tool_call_end_token_id，用于后续判断。
2. 新增 is_reasoning_end 方法：重写父类方法，反向遍历 input_ids 序列，检测 </think> 或 <tool_call> 作为推理结束标记。如果发现配对出现的 <tool_call>...</tool_call>（来自提示模板示例），则跳过；只有孤立的 <tool_call> 才视为隐式结束。代码片段如下：

```python\ndef is_reasoning_end(self, input_ids: Sequence[int]) -> bool:\n start_token_id = self.start_token_id # \n end_token_id = self.end_token_id # \n tool_call_token_id = self._tool_call_token_id\n tool_call_end_token_id = self._tool_call_end_token_id

# 从后向前遍历，优先匹配最近的结束标记\n for i in range(len(input_ids) - 1, -1, -1):\n token_id = input_ids[i]\n if token_id == start_token_id:
# 出现<think>且后面没有</think>或<tool_call>，认为推理未结束\n return False\n if token_id == end_token_id:\n return True\n if tool_call_token_id is not None and token_id == tool_call_token_id:
# 如果当前<tool_call>后面还有</tool_call>，则是配对模板内容，跳过\n if tool_call_end_token_id is not None and any(\n input_ids[j] == tool_call_end_token_id\n for j in range(i + 1, len(input_ids))\n ):\n continue\n return True\n return False
```

`` 3. **新增is_reasoning_end_streaming方法**：在父类检测的基础上，额外检查delta_ids中是否包含token，确保流式场景也能正确识别。 4. **新增extract_content_ids方法**：当父类提取不到内容时，回退到从第一个位置截取后续 token 作为内容，保证工具调用能被正确解析。 5. **修复extract_reasoning中的排序问题**：第二个 commit 将thinking_enabled检查提前到tool_call检查之前，避免thinking_disabled模式下因存在而错误拆分为推理 + 内容。 6. **测试配套**：在测试文件中新增两组测试用例（非流式 / 流式），分别覆盖无前缀和有前缀但缺少的场景，以及多 token delta 中的隐式结束检测。 关键文件：
-vllm/reasoning/qwen3_reasoning_parser.py（模块 推理解析器；类别 source；类型 core-logic；符号 is_reasoning_end, is_reasoning_end_streaming, extract_content_ids, extract_reasoning）核心源码变更，新增is_reasoning_end、is_reasoning_end_streaming、extract_content_ids方法，并修复extract_reasoning中的排序问题。
-tests/reasoning/test_qwen3_reasoning_parser.py`（模块 测试；类别 test；类型 test-coverage）：新增测试用例覆盖 作为隐式推理结束的各种场景，包括流式和非流式。 关键符号：is_reasoning_end, is_reasoning_end_streaming, extract_content_ids, extract_reasoning

关键源码片段

vllm/reasoning/qwen3_reasoning_parser.py

核心源码变更，新增 is_reasoning_end、is_reasoning_end_streaming、extract_content_ids 方法，并修复 extract_reasoning 中的排序问题。 python def is_reasoning_end(self, input_ids: Sequence[int]) -> bool: """判断推理是否结束：检测 </think>或孤立的<tool_call>（隐式结束）""" start_token_id = self.start_token_id # <think> end_token_id = self.end_token_id # </think> tool_call_token_id = self._tool_call_token_id tool_call_end_token_id = self._tool_call_end_token_id # 反向遍历，优先找到最近的结束标记 for i in range(len(input_ids) - 1, -1, -1): token_id = input_ids[i] if token_id == start_token_id: # 找到 <think> 且后面没有 </think> 或 <tool_call>，推理未结束 return False if token_id == end_token_id: return True if tool_call_token_id is not None and token_id == tool_call_token_id: # 如果后面有配对的 </tool_call>，则是模板示例，跳过 if tool_call_end_token_id is not None and any(input_ids[j] == tool_call_end_token_id for j in range(i + 1, len(input_ids))): continue return True return False

评论区精华

1. 性能优化建议（由 gemini-code-assist[bot] 提出）：建议在 is_reasoning_end 中避免两次遍历，将 super() 调用和 <tool_call> 检查合并为一次循环。最终实现采用自定义反向遍历，未调用 super()，已避免性能问题。
2. 排序问题（由 chaunceyjiang 指出）：extract_reasoning 中 thinking_enabled 检查应位于 tool_call 检查之前，否则 thinking_disabled 模式下会错误截断。第二个 commit 已修复。

- is_reasoning_end 性能优化 (performance): 最终实现采用自定义反向遍历，未调用 super()，已避免两次遍历。

- extract_reasoning 中 thinking_disabled 顺序 (correctness): 第二个 commit 修复了顺序问题，确保 thinking_disabled 模式下正确返回。

风险与影响

- 风险:
  1. 回归风险：新逻辑可能影响现有的 Qwen3 推理解析行为，但测试覆盖了主要场景（正常、截断、工具调用），风险较低。
  2. 性能风险：is_reasoning_end 实现为 O(n) 反向遍历，且被频繁调用，但相比原实现仅增加了 <tool_call>token 判断，开销可接受。
  3. 兼容性风险：对于不使用工具调用的 Qwen3 版本，新代码仅在检测到 <tool_call>token 时才会触发，不改变已有行为。
- 影响:
  1. 用户影响：使用 Qwen3/Qwen3.5 模型并启用 --reasoning-parser qwen3 --tool-call-parser qwen3_xml 的用户，在长多轮工具调用中偶发的工具调用缺失问题得到修复。
  2. 系统影响：仅修改推理解析器模块，不影响其余组件。
  3. 团队影响：修复与 Kimi K2 的修复模式一致，便于后续类似问题的统一处理。 - 风险标记：核心路径变更 ，流式非流式双路径

关联脉络

- PR #33646 [Bugfix] Handle case when kimi ends reasoning with a tool call: 同类修复 ：Kimi K2 推理解析器中也存在工具调用未关闭 的问题，采用了相同的处理模式。