PR #45852 完整报告

vllm-project/vllm

[Bugfix][Gemma4] Pre-initialise streaming reasoning state when prompt ends inside an open `<|channel>` (fixes #45834)

合并时间：2026-06-17 18:16

原文链接：http://prhub.com.cn/vllm-project/vllm/pull/45852

执行摘要

- 一句话：修复 Gemma4 post-tool 推理泄露到 content
- 推荐动作：值得精读，尤其关注：
  - 如何在不破坏现有流程的前提下引入状态调整钩子（通过标志 _prompt_streaming_prepared 保证只执行一次）。
  - 设计决策：复用 is_reasoning_end 避免重复扫描逻辑，no-op 过渡处理边界情况。
  - 命名哲学：adjust_initial_state_from_prompt 准确表达了意图。
  - 测试设计：同时覆盖回归用例和鲁棒性用例，以及确保新回合不受影响。

功能与动机

问题源于 issue #45834 报告：Gemma4 工具调用开启 thinking 后，最后工具响应后模板会插入 <|channel>thought\n，导致 prompt 结束在开放推理 channel 中。is_reasoning_end(prompt_token_ids) 已正确返回 False，但该信号未被传播到引擎初始状态。PR body 指出需要在解析器层修复，不修改 chat template。

实现拆解

1. 基类添加钩子：在 vllm/reasoning/abs_reasoning_parsers.py 的 ReasoningParser 中添加空方法 adjust_initial_state_from_prompt(prompt_token_ids)，默认无操作。在 vllm/parser/engine/parser_engine.py 中 ParserEngine 添加同名空方法和 _prompt_streaming_prepared 标志。vllm/parser/engine/adapters.py 中 ParserEngineReasoningAdapter 转发调用。
2. 触发时机：修改 ParserEngine.parse_delta（parser_engine.py），在首次 delta 处理且 prompt_token_ids 非空时调用钩子（注意 flag 顺序：调用前不设置标志，因为钩子可能调用 reset 清标志）。同样在 AbstractParser.parse_delta（abstract_parser.py）中，当推理未结束时调用推理解析器的钩子。
3. Gemma4 实现：在 vllm/parser/gemma4.py 中 Gemma4Parser 重写钩子：调用 is_reasoning_end(prompt_token_ids) 检测 prompt 是否已闭合；若未闭合则 self._engine.reset(initial_state=ParserState.REASONING) 并标记 _streaming_initialized=True。同时添加 (REASONING, THINK_START) -> REASONING no-op 过渡，防止模型自身发出的 <|channel> 被当作 TEXT_CHUNK 泄露。

4. 测试：新增 tests/parser/engine/test_gemma4_streaming_reasoning.py 中的 TestGemma4PromptOpenReasoning（回归）和 TestGemma4PreInitReasoningRobustness（鲁棒性），覆盖开放推理的检测、post-reasoning 内容完整性、以及普通新回合场景不变。
5. 配套：仅修改 6 个文件，全增量无删除，pre-commit 通过，回归测试 1724 passed（11 个预失败无关）。

关键文件：

- tests/parser/engine/test_gemma4_streaming_reasoning.py（模块 解析器测试；类别 test；类型 test-coverage；符号 TestGemma4PromptOpenReasoning, open_reasoning_tokenizer, open_reasoning_parser, _prompt_ids_open_channel）：新增 208 行测试，覆盖 open reasoning prompt 场景和普通场景，确保修复的正确性和回归预防。
- vllm/parser/gemma4.py（模块 解析器；类别 source；类型 core-logic；符号 adjust_initial_state_from_prompt）：核心逻辑变更：新增 adjust_initial_state_from_prompt 方法预初始化引擎状态，以及 no-op 过渡处理边界情况。
- vllm/parser/engine/parser_engine.py（模块 解析器引擎；类别 source；类型 core-logic；符号 adjust_initial_state_from_prompt, _prompt_streaming_prepared）：框架层添加钩子声明和调用点，并维护 _prompt_streaming_prepared 标志确保只调用一次。
- vllm/reasoning/abs_reasoning_parsers.py（模块 推理解析器；类别 source；类型 core-logic；符号 adjust_initial_state_from_prompt）：基类添加钩子定义，为所有推理解析器提供扩展点。
- vllm/parser/engine/adapters.py（模块 适配器；类别 source；类型 core-logic；符号 adjust_initial_state_from_prompt）：适配器转发钩子调用到底层 engine。
- vllm/parser/abstract_parser.py（模块 解析器；类别 source；类型 core-logic）：在 AbstractParser.parse_delta 中调用 reasoning parser 的钩子。

关键符号：adjust_initial_state_from_prompt, is_reasoning_end, TestGemma4PromptOpenReasoning, TestGemma4PreInitReasoningRobustness

关键源码片段

tests/parser/engine/test_gemma4_streaming_reasoning.py

新增 208 行测试，覆盖 open reasoning prompt 场景和普通场景，确保修复的正确性和回归预防。

```
# tests/parser/engine/test_gemma4_streaming_reasoning.py
# —— 新增测试类：验证 prompt 结束在开放推理 channel 时的行为 ——

class TestGemma4PromptOpenReasoning:
    """当 add_generation_prompt=True 且 enable_thinking=True 时，
    模板留下 prompt 以 <|channel>thought\n 结尾，
    即位于开放推理 channel 内。生成的 token 在 <channel|> 之前
    必须被分类为 reasoning，而非可见 content。
    """
```

```
    @pytest.fixture
    def open_reasoning_tokenizer(self):
        # 模拟 <channel|> 后仍有 token 的场景
        return _make_tokenizer(_OPEN_REASONING_GEN_SEQUENCE)

    @staticmethod
    def _prompt_ids_open_channel() -> list[int]:
        # 模拟 prompt 以 <|channel> + 任意 token 结尾
        return [CHANNEL_START_ID, 3000, 3001]

    def test_reasoning_not_leaked_into_content(
        self, open_reasoning_parser, open_reasoning_tokenizer, request_obj
    ):
        results = _stream_tokens_batched(
            open_reasoning_parser, open_reasoning_tokenizer, request_obj,
            batch_size=1, prompt_token_ids=self._prompt_ids_open_channel(),
        )
        reasoning, content, _ = _collect_fields(results)
        assert "Sure, the answer is 42" in reasoning, \
            f"Expected pre-<channel|> tokens in reasoning, got reasoning={reasoning!r} content=
            {content!r}"
        # 验证推理内容没有泄露到 content 中
        for leaked in ("Sure", "answer", "42"):
            assert leaked not in content, \
                f"Reasoning text leaked into content: {content!r}"
```

vllm/parser/gemma4.py

核心逻辑变更：新增 adjust_initial_state_from_prompt 方法预初始化引擎状态，以及 no-op 过渡处理边界情况。

```
# vllm/parser/gemma4.py (Gemma4Parser.adjust_initial_state_from_prompt)
def adjust_initial_state_from_prompt(self, prompt_token_ids: Sequence[int]) -> None:
    """预初始化引擎状态为 REASONING，当 prompt 结束在开放推理 channel 时。

    原理：利用现有的 is_reasoning_end 方法向后扫描 prompt，
    若未发现 THINK_END (且前有 THINK_START)，则说明 channel 未关闭。
    覆盖 issue #45834 描述的场景。
    """
    # 如果 prompt 本身已经结束了推理（例如包含 <channel|>），无需调整
    if self.is_reasoning_end(list(prompt_token_ids)):
        return
    # 将底层引擎状态从默认 CONTENT 切换为 REASONING
    self._engine.reset(initial_state=ParserState.REASONING)
    # 阻止后续默认的 initialize_streaming() 用 CONTENT 覆盖此状态
    self._streaming_initialized = True

# 同时在 gemma4_config() 中新增了一个 no-op 过渡，处理模型自身
# 也可能输出 <|channel> 的情况：
(ParserState.REASONING, "THINK_START"): Transition(
```

```
        ParserState.REASONING, # 状态不变，不产生事件
        (), # 不触发任何语义事件，从而避免泄露 TEXT_CHUNK
    )
```

评论区精华

Reviewer bbrowning 提出数项改进建议，贡献者全部采纳：

- 命名：prepare_streaming_for_prompt 改名为 adjust_initial_state_from_prompt，准确反映语义（调整初始状态）。
- 实现简化：使用 self._engine.reset(initial_state=ParserState.REASONING) 代替 initialize_streaming，避免额外副作用。
- 复用现有方法：利用 is_reasoning_end 检测 prompt 末端，消除独立的 _prompt_ends_in_open_reasoning 和 token ID 扫描，隐式处理多轮对话和隐式推理开始 / 结束。
- Flag 顺序修复：纠正 parse_delta 中设置标志与调用 hook 的顺序，防止 hook 内调用 reset 清除标志后重复扫描。\nbbrowning 最终批准前手工验证了修复在一台具有多轮工具调用和 thinking 的服务器上有效。
- Flag 顺序修复：钩子调用前不应设置标志 (correctness): 修复为先调用钩子再设置标志，并在注释中说明原因。
- 用 _engine.reset 代替 initialize_streaming (design): 采纳，直接 reset 引擎状态并设置 _streaming_initialized = True 防止后续覆盖。
- 方法命名：prepare_streaming_for_prompt → adjust_initial_state_from_prompt (design): 重命名，更准确反映语义。
- 复用 is_reasoning_end 消除独立 token 扫描 (correctness): 完全移除独立的 _prompt_ends_in_open_reasoning，直接调用 is_reasoning_end。
- 添加 no-op 过渡 (REASONING, THINK_START) (design): 添加 (ParserState.REASONING, "THINK_START") -> ParserState.REASONING 不产生事件。

风险与影响

- 风险：风险较低：
  1. 新增 no-op 过渡 (REASONING, THINK_START) 目前无害，但若未来 Gemma4 模型行为变化（如需要双重 <|channel> 输出），可能需要调整。
  2. 对 prompt 全序列的 is_reasoning_end 线性扫描在长对话中可能产生性能影响，但扫描仅一次且 token 量有限。
  3. 钩子框架可能被其他解析器误用，但默认空实现保证安全。
  4. 测试覆盖了开放推理和常见场景，但未覆盖极端长多轮混合 tool_response 与 reasoning 的情况（但 is_reasoning_end 本身已覆盖）。 - 影响：影响范围：Gemma4 用户是直接受益者，post-tool 推理不再泄露。其他模型用户无影响。团队获得了可扩展的解析器初始状态调整机制，Qwen3、DeepSeek 等解析器未来可利用此钩子处理类似需求。兼容性：所有变更均为新增，无删除，向后兼容。性能无退化。 - 风险标记：新状态机过渡，线性扫描 prompt 性能，钩子框架误用风险

关联脉络

- 暂无明显关联 PR