# PR #39870 完整报告

vllm-project/vllm

[BugFix] Support custom tool parsers when tool_choice is `required` and named function

合并时间：2026-04-18 00:38

原文链接：http://prhub.com.cn/vllm-project/vllm/pull/39870

## 执行摘要

- 一句话：修复 GLM 模型在 tool_choice 为 required 或命名函数时的工具调用解析问题，支持 XML 格式输出。
- 推荐动作：建议技术管理者关注 supports_required_and_named 标志的设计决策，它提供了优雅的回退机制。工程师应仔细阅读流式与非流式处理中的分支逻辑，以理解如何集成自定义解析器。此外，注意讨论中未解决的清理结构化输出问题，可能需要在后续 PR 中处理。

## 功能与动机

根据 PR body，此前 'required' 和命名函数路径使用硬编码的 JSON 解析（TypeAdapter(list[FunctionDefinition]).validate_json() 用于 required，原始内容传递用于 named），这适用于引导解码强制 JSON 输出的模型（如 Hermes），但对于像 GLM-5.1 这样使用 XML 工具调用语法的模型会失败，导致空或错误结果。

## 实现拆解

1. 添加类级标志：在 vllm/tool_parsers/abstract_tool_parser.py 的 ToolParser 类中，新增 supports_required_and_named: bool = True 标志，用于指示解析器是否支持处理 'required' 和命名函数 tool_choice。GLM 解析器设置此标志为 False。
2. 修改非流式解析逻辑：在 vllm/entrypoints/openai/engine/serving.py 的 _parse_tool_calls_from_content 函数中，添加条件检查 tool_parser_cls.supports_required_and_named，当为 False 时，将 'required' 和命名函数路径路由到工具解析器的 extract_tool_calls 方法。
3. 调整流式处理分支：在 vllm/entrypoints/openai/chat_completion/serving.py 的 chat_completion_stream_generator 函数中，引入 tool_choice_uses_parser 变量，并在控制流中添加相应分支，使用解析器的 extract_tool_calls_streaming 方法。
4. 更新 GLM 解析器：在 vllm/tool_parsers/glm4_moe_tool_parser.py 和 glm47_moe_tool_parser.py 中，设置 supports_required_and_named = False，并重写 adjust_request 方法，当 tool_choice 为 'required' 或命名函数时，跳过调用父类的 adjust_request 以避免设置结构化输出，从而允许 XML 自由输出。
5. 测试与兼容性：PR body 提到手动端到端测试，但无测试文件变更；此外，讨论中提及对 Responses API 的兼容性考虑。

关键文件：

- vllm/entrypoints/openai/engine/serving.py（模块 请求处理；类别 source；类型 core-logic；符号 _parse_tool_calls_from_content）：核心非流式工具调用解析逻辑，控制 required 和命名函数路径的路由，直接影响所有非流式请求的处理。
- vllm/entrypoints/openai/chat_completion/serving.py（模块 流式处理；类别 source；类型 core-logic；符号 chat_completion_stream_generator）：核心流式工具调用处理逻辑，引入 tool_choice_uses_parser 变量并调整控制流，影响所有流式请求。
- vllm/tool_parsers/abstract_tool_parser.py（模块 工具解析器；类别 source；类型 core-logic；符号 ToolParser, supports_required_and_named）：抽象工具解析器基类，添加 supports_required_and_named 类级标志，定义整个框架的行为。
- vllm/tool_parsers/glm4_moe_tool_parser.py（模块 GLM 解析器；类别 source；类型 core-logic；符号 Glm4MoeModelToolParser, supports_required_and_named, adjust_request）：GLM 工具解析器具体实现，设置 supports_required_and_named = False 并重写 adjust_request 以跳过结构化输出。
- vllm/tool_parsers/glm47_moe_tool_parser.py（模块 GLM 解析器；类别 source；类型 core-logic；符号 Glm47MoeModelToolParser, supports_required_and_named）：GLM-4.7 工具解析器，类似设置 supports_required_and_named = False，但变更较小。

关键符号：_parse_tool_calls_from_content, chat_completion_stream_generator, ToolParser.supports_required_and_named, Glm4MoeModelToolParser.adjust_request

## 关键源码片段

### vllm/entrypoints/openai/engine/serving.py

核心非流式工具调用解析逻辑，控制 required 和命名函数路径的路由，直接影响所有非流式请求的处理。

```
def _parse_tool_calls_from_content(
    request: ChatCompletionRequest | ResponsesRequest,
    tokenizer: TokenizerLike | None,
    enable_auto_tools: bool,
    tool_parser_cls: type[ToolParser] | None,
    content: str | None = None,
) -> tuple[list[FunctionCall] | None, str | None]:
    # ... 省略其他代码 ...
    # 当工具解析器不支持 required/named 时，使用标准 JSON 解析
    elif (
        not use_mistral_tool_parser
        and request.tool_choice == "required"
        and (tool_parser_cls is None or tool_parser_cls.supports_required_and_named)
    ):
        # "required" 使用标准 JSON 解析
        tool_calls = []
        with contextlib.suppress(ValidationError):
            content = content or ""
            tool_calls = TypeAdapter(list[FunctionDefinition]).validate_json(content)
        # ... 处理工具调用 ...
```

```
    # 当工具解析器支持或需要回退时，使用自动工具调用解析
    elif tool_parser_cls and (
        use_mistral_tool_parser
        or (
            enable_auto_tools
            and (
                request.tool_choice == "auto"
                or request.tool_choice is None
                or (
                    not tool_parser_cls.supports_required_and_named # 关键变更：当标志为 False
                    时，required/named 也走解析器路径
                    and request.tools
                    and (
                        request.tool_choice == "required"
                        or isinstance(request.tool_choice, ChatCompletionNamedToolChoiceParam)
                    )
                )
            )
        )
    ):
        # 自动工具调用解析（也用作 required/named 的回退）
        try:
            tool_parser = tool_parser_cls(tokenizer, request.tools)
        except RuntimeError as e:
            logger.exception("Error in tool parser creation.")
            raise e
        tool_call_info = tool_parser.extract_tool_calls(
            content if content is not None else "",
            request=request,
        )
        # … 处理提取的工具调用 …
```

## vllm/entrypoints/openai/chat_completion/serving.py

核心流式工具调用处理逻辑，引入 tool_choice_uses_parser 变量并调整控制流，影响所有流式请求。

```
async def chat_completion_stream_generator(
    self,
    request: ChatCompletionRequest,
    # … 省略参数 …
) -> AsyncIterator[ChatCompletionStreamResponse]:
    # … 省略其他代码 …
    # 确定 required/named tool_choice 是否应回退到工具解析器路径
    tool_choice_uses_parser = (
        self.tool_parser is not None
        and not self.tool_parser.supports_required_and_named #
        关键变更：当解析器不支持时，启用回退
        and request.tools
        and (
```

```
            request.tool_choice == "required"
            or isinstance(request.tool_choice, ChatCompletionNamedToolChoiceParam)
        )
    )
    # 更新控制流，将 tool_choice_uses_parser 加入条件
    if (
        is_mistral_grammar_path
        or tool_choice_auto
        or tool_choice_uses_parser # 新增条件
        or reasoning_parser
    ):
        # 这些仅在需要跟踪 token 时使用（如自动工具选择）
        all_previous_token_ids = [[] for _ in range(num_choices)]
        reasoning_end_arr = [False] * num_choices
        prompt_is_reasoning_end_arr: list[bool | None] = [None] * num_choices
    else:
        all_previous_token_ids = None
    # ... 后续流式处理中，类似地更新分支条件以包含 tool_choice_uses_parser ...
```

## vllm/tool_parsers/glm4_moe_tool_parser.py

GLM 工具解析器具体实现，设置 `supports_required_and_named = False` 并重写 `adjust_request` 以跳过结构化输出。

```
class Glm4MoeModelToolParser(ToolParser):
    """
    GLM-4 模型的工具解析器，支持增量字符串流式处理。
    """
    supports_required_and_named = False # 关键变更：设置为 False，表示不支持标准 JSON 解析

    def adjust_request(
        self, request: ChatCompletionRequest | ResponsesRequest
    ) -> ChatCompletionRequest | ResponsesRequest:
        """
        调整请求参数以处理工具调用 token。
        对于 required/named tool_choice，跳过设置 structured_outputs，因为 GLM 模型以 XML
        格式输出工具调用
        （根据聊天模板）。引导解码会强制 JSON 输出，与 XML 格式冲突导致解析失败。
        """
        if request.tools:
            tc = request.tool_choice
            if tc == "required" or isinstance(tc, ChatCompletionNamedToolChoiceParam):
                # 对于 required/named，不调用 super().adjust_request()，以避免设置 structured_
                outputs
                # 从而允许模型自由输出 XML，工具解析器将从 XML 中提取工具调用
                if request.tool_choice != "none":
                    request.skip_special_tokens = False
                return request
        request = super().adjust_request(request)
        if request.tools and request.tool_choice != "none":
```

```
            # 确保工具调用 token 不被跳过
            request.skip_special_tokens = False
        return request
```

## 评论区精华

review 中主要讨论了以下点：

- 标志默认值：sfeng33 建议 supports_required_and_named 默认应为 True，GLM 解析器设为 False，以确保向后兼容性；JaredforReal 已修复。
- 类型安全：gemini-code-assist 指出在流式处理中访问 tc.function 可能因类型不匹配导致 TypeError，需要谨慎处理。
- 清理结构化输出：Copilot 建议在 GLM 解析器的 adjust_request 中显式清理 response_format 和 structured_outputs 字段，以避免引导解码冲突；此问题在讨论中未明确解决。
- Responses API 兼容性：sfeng33 提醒需要支持 ResponsesRequest，JaredforReal 已修复。
  - supports_required_and_named 默认值设置 (design): 已修复，设置默认值为 True，GLM 解析器显式设为 False。
  - 流式处理中类型安全问题 (correctness): 讨论中未明确解决，可能需要额外处理以确保类型安全。
  - 清理 response_format 字段以避免引导解码冲突 (design): 未在讨论中明确解决，遗留潜在风险。

## 风险与影响

- 风险：技术风险包括：
- 回归风险：修改了核心解析路径 _parse_tool_calls_from_content 和 chat_completion_stream_generator，可能影响其他模型或 tool_choice 场景，特别是当自定义解析器未正确实现时。
- 性能风险：额外的解析步骤和条件检查可能略微增加延迟，但影响较小。
- 兼容性风险：新标志的引入需要所有自定义工具解析器显式设置，否则默认行为可能不适用于某些模型。
- 安全风险：无直接安全漏洞，但解析错误可能导致数据泄露或服务中断。 具体文件：vllm/entrypoints/openai/engine/serving.py 和 vllm/entrypoints/openai/chat_completion/serving.py 是关键风险点。
- 影响：影响范围：
- 用户影响：使用 GLM-4.7/5/5.1 等 XML 输出模型的用户现在能正确使用 'required' 和命名函数 tool_choice，提升功能完整性。
- 系统影响：扩展了工具解析器框架的灵活性，为未来支持更多输出格式奠定基础。
- 团队影响：开发人员需要了解新标志，并在实现自定义解析器时考虑是否支持 required/named 路径。 影响程度中等，主要针对特定模型和用例。

- 风险标记：核心路径变更，缺少测试覆盖，类型安全风险

## 关联脉络

- PR #40089 [Misc][UX] Map mimo reasoning and tooling parsers: 同样涉及工具解析器映射，扩展模型支持，技术领域相关。
- PR #40090 [Bugfix] Fix empty delta detection in Qwen3XMLToolParser streaming: 修复工具解析器在流式输出中的问题，涉及类似 XML 解析逻辑。