PR #41110 完整报告

vllm-project/vllm

[Frontend]Responses API supports Tool/Function calling with streaming with named tool/function

合并时间：2026-04-29 17:11

原文链接：http://prhub.com.cn/vllm-project/vllm/pull/41110

执行摘要

- 一句话：Responses API 流式命名函数调用支持
- 推荐动作：此 PR 展示了如何为 Responses API 补齐流式命名工具调用的能力，并修复了因递增时机错误导致的计数问题。设计上值得关注的是：将函数名提取逻辑抽离为 _get_function_name，以及将流式 tool call 构建委托给专门的 extract_named_tool_call_streaming 函数，实现了关注点分离。建议流式 tool call 相关功能开发者精读。

功能与动机

此前 Responses API 的流式模式仅支持 auto 与 required 的 tool_choice，对 tool_choice: {"type": "function", "name": "..."} 这类指定具体函数的 named tool 调用无法正确解析和流式输出。此 PR 旨在补齐这一能力，使 Responses API 在流式场景下也能正确输出命名函数的调用增量。

实现拆解

1. 修整工具函数：在 vllm/tool_parsers/streaming.py 中，将 extract_named_tool_call_streaming 的返回值从三元组简化为二元组，移除 created_new_tool_call 标志，参数 tool_call_array_index 增加默认值 0，使函数更通用。
2. 统一 Parser 层：在 vllm/parser/abstract_parser.py 中，新增 _get_function_name 私有方法，从请求体中提取函数名称；修改 _parse_tool_calls 分支，将原本独立的 ToolChoiceFunction 和 ChatCompletionNamedToolChoiceParam 处理合并为一次调用 self._get_function_name；修改 _extract_tool_calls_streaming 方法，使其在遇到 ToolChoiceFunction 或 ChatCompletionNamedToolChoiceParam 时调用 extract_named_tool_call_streaming 进行流式解析，并返回 DeltaMessage 和 function_name_returned。
3. 改造 Serving 层：在 vllm/entrypoints/openai/chat_completion/serving.py 的 chat_completion_stream_generator 方法中，移除原有的内联命名 tool 调用构建逻辑，改为统一调用 extract_named_tool_call_streaming；同时修正了 history_tool_call_cnt 的递增条件，仅当 delta_message 包含有效 id（即新工具调用被创建）时才递增，避免每个 token 块都计数。

4. 补充测试覆盖：在 tests/entrypoints/openai/responses/test_function_call.py 中，为 test_function_calling_with_streaming_types 和 test_function_calling_with_streaming_forced_tool_choice 两个测试的参数化列表添加 {"type": "function", "name": "get_weather"} 这类 named tool choice 场景，确保新路径被集成测试覆盖。

关键文件：

- vllm/parser/abstract_parser.py（模块 解析器；类别 source；类型 dependency-wiring；符号 _get_function_name）：核心变更文件，添加了 _get_function_name 方法，重构了 _parse_tool_calls 和 _extract_tool_calls_streaming 以统一支持 ToolChoiceFunction 和 ChatCompletionNamedToolChoiceParam。
- vllm/entrypoints/openai/chat_completion/serving.py（模块 服务层；类别 source；类型 dependency-wiring）：聊天补全服务的核心流式生成器，替换内联命名 tool 逻辑为 extract_named_tool_call_streaming 并修正计数器递增条件。
- vllm/tool_parsers/streaming.py（模块 工具流式解析；类别 source；类型 core-logic）：流式工具解析工具函数，简化了 extract_named_tool_call_streaming 返回值并添加参数默认值。
- tests/entrypoints/openai/responses/test_function_call.py（模块 测试；类别 test；类型 test-coverage）：增加 named tool choice 参数化测试场景，确保新功能被集成测试覆盖。

关键符号：_get_function_name, _extract_tool_calls_streaming, _parse_tool_calls, extract_named_tool_call_streaming, chat_completion_stream_generator

关键源码片段

vllm/parser/abstract_parser.py

核心变更文件，添加了 _get_function_name 方法，重构了 _parse_tool_calls 和 _extract_tool_calls_streaming 以统一支持 ToolChoiceFunction 和 ChatCompletionNamedToolChoiceParam。

```
# file: vllm/parser/abstract_parser.py ( 部分 )

def _get_function_name(
    self, request: ChatCompletionRequest | ResponsesRequest
) -> str:
    """从请求中提取强制调用的函数名称"""
    if request.tool_choice and isinstance(
        request.tool_choice, ToolChoiceFunction
    ):
        return request.tool_choice.name
    if request.tool_choice and isinstance(
        request.tool_choice, ChatCompletionNamedToolChoiceParam
    ):
        return request.tool_choice.function.name
    raise ValueError("Invalid tool_choice for function name extraction.")

def _extract_tool_calls_streaming(
```

```
    self,
    previous_text: str,
    current_text: str,
    delta_text: str,
    previous_token_ids: Sequence[int],
    current_token_ids: Sequence[int],
    delta_token_ids: Sequence[int],
    request: ChatCompletionRequest | ResponsesRequest,
    tool_call_idx: int | None = None,
    tool_call_id_type: str = "random",
    function_name_returned: bool = False,
) -> tuple[DeltaMessage | None, bool]:
    # 处理 named / forced tool choice
    if request.tool_choice and isinstance(
        request.tool_choice,
        (ToolChoiceFunction, ChatCompletionNamedToolChoiceParam),
    ):
        # 调用集中式的流式命名工具解析
        delta_message, function_name_returned = (
            extract_named_tool_call_streaming(
                delta_text=delta_text,
                function_name=self._get_function_name(request),
                function_name_returned=function_name_returned,
                tool_call_idx=tool_call_idx,
                tool_call_id_type=tool_call_id_type,
                tokenizer=self.model_tokenizer,
            )
        )
        return delta_message, function_name_returned
    # 后续处理 required / auto 模式（略）
```

vllm/entrypoints/openai/chat_completion/serving.py

聊天补全服务的核心流式生成器，替换内联命名 tool 逻辑为 extract_named_tool_call_streaming 并修正计数器递增条件。

```
# file: vllm/entrypoints/openai/chat_completion/serving.py（部分）

delta_message, function_name_returned[i] = (
    extract_named_tool_call_streaming(
        delta_text=delta_text,
        function_name=tool_choice_function_name,
        function_name_returned=function_name_returned[i],
        tool_call_idx=history_tool_call_cnt,
        tool_call_id_type=self.tool_call_id_type,
        tokenizer=tokenizer,
        tool_call_array_index=i,
    )
)
# 只有真正创建了新工具调用（tool_calls[0].id 非空）时才递增计数器
```

```
if (
    delta_message
    and delta_message.tool_calls
    and delta_message.tool_calls[0].id is not None
):
    history_tool_call_cnt += 1
    tools_streamed[i] = True
```

评论区精华

在 Review 中，gemini-code-assist[bot] 提出了两个高优先级问题：

- 问题 1（正确性）：history_tool_call_cnt 在每次流式 chunk 时都递增，导致 n>1 场景下工具调用 ID 索引膨胀。作者通过将递增条件改为 delta_message.tool_calls[0].id is not None 解决了该问题。
- 问题 2（统一性）：Parser 的 _extract_tool_calls_streaming 在遇到 named tool choice 时应同时处理 ToolChoiceFunction 和 ChatCompletionNamedToolChoiceParam，而不仅仅是前者。作者将判断条件扩展为 isinstance(..., (ToolChoiceFunction, ChatCompletionNamedToolChoiceParam))，并调用 self._get_function_name 统一提取名称。

两个建议均已被采纳并体现在最终提交中。

- history_tool_call_cnt 递增条件 (correctness): 作者将递增限制为仅当 delta_message.tool_calls[0].id is not None 时执行，从而只在真正创建新工具调用时递增计数器。
- Parser 中 named tool choice 的统一处理 (design): 作者将条件扩展为 isinstance(request.tool_choice, (ToolChoiceFunction, ChatCompletionNamedToolChoiceParam))，并通过 self._get_function_name 统一提取名称。

风险与影响

- 风险：
  - 回归风险：修改了 Parser 和 Serving 层的核心流式 tool calling 逻辑，可能影响 auto 和 required 模式的行为。现有测试（包括新增的 named 测试）通过，但覆盖可能不够全面。
  - 性能风险：无显著性能影响，新增函数调用路径与原逻辑一致。
  - 安全风险：无。
  - 兼容性：向后兼容，新增功能不会破坏现有 API 调用。
- 影响：
  - 用户角度：使用 Responses API 进行流式 tool calling 时，现在可以指定具体函数名称并获取正确的增量事件，tool call ID 不再因 chunk 重复而错乱。
  - 系统角度：改动集中于 4 个文件，影响范围可控，但关联到统一的 extract_named_tool_call_streaming 函数，因此 Chat Completion API 的流式工具调

用也会受益于相同的修复和简化。

- 团队角度：统一了 Parser 层对两种 API 风格（Responses 与 Chat Completion）中 named tool choice 的处理，降低后续维护成本。
- 风险标记：核心流式路径变更，tool_call 索引修复

关联脉络

- 暂无明显关联 PR