PR #45600 完整报告

vllm-project/vllm

[Frontend] Skip structural tags for auto tool_choice without strict mode

合并时间：2026-06-16 03:55

原文链接：http://prhub.com.cn/vllm-project/vllm/pull/45600

执行摘要

- 一句话：auto tool_choice 跳过结构标签除非 strict=True
- 推荐动作：值得精读，重点关注：
  - 如何通过早期短路返回实现性能优化。
  - 兼容性处理：增加新字段时确保默认行为不变。
  - 测试设计：创建 sample_tools_strict fixture 分离严格与非严格场景。

功能与动机

防止结构标签在未明确要求严格函数调用时过度约束模型生成，与 OpenAI API 规范对齐。原始 PR #45003 引入的结构标签在 auto tool_choice 下会强制约束输出，即使客户端没有声明 strict。

实现拆解

1. 扩展协议模型：在 FunctionDefinition（vllm/entrypoints/openai/engine/protocol.py）和 AnthropicTool（vllm/entrypoints/anthropic/protocol.py）中添加 strict: bool | None = None 字段，序列化时若为 None 则忽略。
2. 添加 _any_tool_strict 检查：在 vllm/tool_parsers/structural_tag_registry.py 中新增辅助函数，遍历工具列表，检查是否存在任意工具的 strict == True。支持 ChatCompletionToolsParam 和 FunctionTool 两种类型。
3. 修改 get_model_structural_tag：在返回结构标签之前增加早期返回检查：若 tool_choice == "auto" 且 _any_tool_strict(tools) 返回 False，则直接返回 None，跳过后续的 schema 转储和标签构建。
4. Anthropic 服务传递 strict：在 vllm/entrypoints/anthropic/serving.py 中的工具转换逻辑中将 AnthropicTool.strict 传递给 FunctionTool.strict。
5. 调整测试：更新核心测试 (test_structural_tag_registry.py) 添加 sample_tools_strict fixture，并将需要结构标签的测试用例改为使用带 strict 的工具。为 Qwen3 Coder 和 DeepSeek V4 工具解析器测试添加 _with_strict 辅助函数。
6. 文档同步：更新 docs/features/tool_calling.md，说明 strict 字段在 tool_choice="auto" 下的作用。

关键文件：

- vllm/tool_parsers/structural_tag_registry.py（模块 结构标签；类别 source；类型 core-logic；符号 _any_tool_strict）：核心修改：新增 _any_tool_strict 函数，并在 get_model_structural_tag 中增加 tool_choice='auto' 时的跳过逻辑。
- tests/tool_parsers/test_structural_tag_registry.py（模块 标签测试；类别 test；类型 test-coverage；符号 sample_tools_strict, test_auto_tool_choice_skips_structural_tag_without_strict）：核心测试：新增 sample_tools_strict fixture，并更新多个测试用例使用 strict 工具，确保新逻辑覆盖。
- vllm/entrypoints/openai/engine/protocol.py（模块 协议模型；类别 source；类型 core-logic）：扩展 FunctionDefinition 模型，添加 strict 字段，并确保序列化时忽略 None 值。
- tests/tool_parsers/test_qwen3coder_tool_parser.py（模块 Qwen3 测试；类别 test；类型 test-coverage；符号 _with_strict）：为 Qwen3 Coder 测试添加 _with_strict 辅助函数，确保结构标签测试在 strict 场景下通过。
- tests/tool_parsers/test_deepseekv4_tool_parser.py（模块 DeepSeek 测试；类别 test；类型 test-coverage；符号 _with_strict）：为 DeepSeek V4 测试添加 _with_strict 辅助函数，结构标签相关测试适配 strict 要求。
- vllm/entrypoints/anthropic/protocol.py（模块 Anthropic 协议；类别 source；类型 core-logic）：AnthropicTool 模型添加 strict 字段，支持 Anthropic 协议的 strict 工具定义。
- vllm/entrypoints/anthropic/serving.py（模块 Anthropic 服务；类别 source；类型 core-logic）：将 AnthropicTool.strict 传递给 FunctionTool.strict，确保 Anthropic 请求中的 strict 设置被正确处理。
- docs/features/tool_calling.md（模块 文档；类别 docs；类型 documentation）：更新文档以反映新行为，指导用户如何使用 strict 字段。

关键符号：_any_tool_strict, get_model_structural_tag, _with_strict

关键源码片段

vllm/tool_parsers/structural_tag_registry.py

核心修改：新增 _any_tool_strict 函数，并在 get_model_structural_tag 中增加 tool_choice='auto' 时的跳过逻辑。

```
# vllm/tool_parsers/structural_tag_registry.py

def _any_tool_strict(
    tools: Sequence[ChatCompletionToolsParam | ResponsesTool],
) -> bool:
    """检查工具列表中是否有任意一个工具设置了 strict=True。"""
    for tool in tools:
        # 处理 Responses API 的 FunctionTool 类型
        if isinstance(tool, FunctionTool) and tool.strict is True:
            return True
        # 处理 Chat Completions API 的 ChatCompletionToolsParam 类型
        if isinstance(tool, ChatCompletionToolsParam) and tool.function.strict is True:
```

```
            return True
    return False

def get_model_structural_tag(
    model: str,
    tools: Sequence[ChatCompletionToolsParam | ResponsesTool] | None,
    tool_choice: ToolChoice,
    reasoning: bool,
) -> StructuralTag | None:
    """Build a structural tag with xgrammar's builtin model templates."""
    if not tools or tool_choice == "none":
        return None
    # 关键新增：当 tool_choice 为 "auto" 且没有任何工具声明 strict=True 时，
    # 跳过结构标签，避免过度约束模型生成。
    if tool_choice == "auto" and not _any_tool_strict(tools):
        return None
    dumped_tools = [_dump_tool_for_xgrammar(tool) for tool in tools]
    dumped_tool_choice = _dump_tool_choice_for_xgrammar(tool_choice)
    # 后续保持不变 ...
```

vllm/entrypoints/openai/engine/protocol.py

扩展 FunctionDefinition 模型，添加 strict 字段，并确保序列化时忽略 None 值。

```
# vllm/entrypoints/openai/engine/protocol.py

class FunctionDefinition(OpenAIBaseModel):
    name: str
    description: str | None = None
    parameters: dict[str, Any] | None = None
    strict: bool | None = None # 新增字段，匹配 OpenAI API 规范
    defer_loading: bool | None = None

    @model_serializer(mode="wrap")
    def _serialize(self, handler):
        data = handler(self)
        # 序列化时如果 strict 为 None 则移除，避免发送不必要的字段
        if self.strict is None:
            data.pop("strict", None)
        if self.defer_loading is None:
            data.pop("defer_loading", None)
        return data
```

评论区精华

- @bbrowning主张完全回退 #45003，认为结构标签应该仅通过环境变量 opt-in 启用。
- @chaunceyjiang提出对性能（遍历所有工具检查 strict）和兼容性（Claude Code 无 strict 字段）的担忧。

- @sfeng33回应：性能上是净收益（无 strict 时跳过 schema dump），兼容性上 strict 默认为 None，只有显式 True 才触发。
- 最终结论：同意合并，要求显式 opt-in 解决了兼容性问题。
  - 是否应该回退 #45003 (design): 不回退，接受本 PR 的 opt-in 方案作为改善。
  - 遍历工具检查 strict 的性能开销 (performance): 接受性能论证。
  - Anthropic 协议是否支持 strict 字段 (question): 需要添加 strict 字段以完全支持 Anthropic 协议。

风险与影响

- 风险：
  1. 兼容性风险：原依赖自动约束且未设置 strict 的客户端，在 auto 下不再获得结构标签约束。但作者声明 strict 默认 None，因此对未发送该字段的客户端无变化。
  2. 性能风险：每次请求需要遍历工具列表检查 strict，但 _any_tool_strict 短路返回且后续避免 schema dump，整体是优化。
  3. Anthropic 服务兼容性：新增字段已处理为 None 时忽略，不破坏现有请求。 - 影响：用户影响：使用 tool_choice='auto' 且未设置 strict=True 的请求不再获得结构标签约束。需严格格式的用户需在工具定义中设置 strict=True。系统影响：代码库增加了对 OpenAI strict 字段的支持，提升协议兼容性。团队影响：结构标签启用路径更清晰：auto 需要显式 opt-in，required 或 named 仍默认使用。
- 风险标记：默认行为变更，兼容性忧虑，遍历开销隐患

关联脉络

- PR #45003 Implement structural tag for tool calling (assumed): 本 PR 直接修改了 #45003 引入的 get_model_structural_tag 行为，增加了 strict 门控。
- PR #44965 [Bugfix] Reject out-of-range temperature values in SamplingParams: 虽然不直接相关，但同为近期前端修复，体现工具调用参数的规范化趋势。