PR #21593 完整报告

sgl-project/sglang

Fix tool call constrained decoding and parsing for models with native formats

合并时间：2026-04-11 11:37

原文链接：http://prhub.com.cn/sgl-project/sglang/pull/21593

执行摘要

本 PR 修复了当模型配置了原生工具调用格式时，tool_choice="required" 模式下约束解码与解析失败的问题。通过引入 at_least_one 标志强制至少一个工具调用，并调整解析逻辑以支持模型原生格式（如 DeepSeek、Kimi、Qwen 等），提升了 OpenAI 协议合规性和用户体验。该变更影响多个核心模块，经过充分测试和 review 讨论，是一个重要的 bugfix。

功能与动机

为什么做？当使用模型特定的 --tool-call-parser（例如 kimi_k2、deepseekv3、qwen25）时，tool_choice="required" 此前强制使用通用 JSON 模式约束，与模型的原生工具调用格式（如特殊标记 <|tool_calls_section_begin|>）冲突，导致解析失败并返回 tool_calls: null。这违反了 OpenAI 协议中 required 必须调用工具的要求。

关键引用：PR body 明确指出："tool_choice="required" previously forced a generic JSON schemaconstraint that conflicted with the model's native tool call format... resulting in tool_calls: null。"

实现拆解

按模块梳理关键改动点： | 模块 | 关键改动 | 影响 | |------|----------|------| | 协议层 (protocol.py) | 添加 at_least_one: bool = False 字段到 LegacyStructuralTagResponseFormat | 支持强制工具调用的语义传递 | | 文法后端 (xgrammar_backend.py) | 传递 at_least_one 字段到 xgrammar 的 StructuralTag API | 实现解码时强制至少一个工具调用 | | 功能调用解析器 (function_call_parser.py) | 修改 get_structure_constraint()，为 required/ 命名工具选择使用 structural_tag 并设置 at_least_one=True；添加 supports_structural_tag() 检查 | 核心约束逻辑调整，确保原生格式优先 | | 服务层 (serving_chat.py) | 更新 _process_tool_calls() 方法，镜像流式路径逻辑，使用 supports_structural_tag() 守卫 | 修复非流式解析失败，提升一致性 | | 检测器修复 (base_format_detector.py, deepseekv3_detector.py) | 修复 parse_streaming_increment() 中空参数检查；修复 structure_info() 格式前缀 | 确保流式解析正确性和模型识别 | | 测试更新 | 新增 TestProcessToolCallsWithRequiredToolChoice 和 TestGetStructureConstraint 等测试类 | 覆盖新逻辑，防止回归 |

关键代码示例（来自 function_call_parser.py）：

```
def get_structure_constraint(self, tool_choice, parallel_tool_calls=True):
    if self.detector.supports_structural_tag():
```

```
is_required = tool_choice == "required" or isinstance(tool_choice, ToolChoice)
if is_required or (tool_choice == "auto" and strict_condition):
    tag = self.get_structure_tag(at_least_one=is_required) # 设置 at_least_one
    return ("structural_tag", tag)
```

评论区精华

Review 讨论聚焦于逻辑一致性：

- AgainstEntropy提问："Should we also explicitly check supports_structural_tag() in _process_tool_calls just as here?"
- JustinTong0323回应："Good catch! Added supports_structural_tag() guard in the non-streaming _process_tool_calls path to mirror the streaming logic."

这确保了非流式与流式路径行为一致，避免了潜在解析错误。

风险与影响

具体风险：

1. 回归风险：serving_chat.py 的解析路径变更可能影响现有工具调用功能，但通过新增单元测试（如 TestProcessToolCallsWithRequiredToolChoice）已覆盖。
2. 兼容性风险：对于不支持 structural_tag 的模型（如 lfm2），PR 通过回退到 JSON 数组解析处理，但需确保所有检测器正确实现。
3. 性能影响：添加 supports_structural_tag() 检查可能增加轻微开销，但在工具调用场景中可忽略。

影响范围：

- 用户：修复了 deepseekv3、kimi_k2、qwen25 等模型在 required 模式下的工具调用失败，提升可靠性。
- 系统：改进 OpenAPI 协议合规性，正确分离 strict 和 required 语义。
- 团队：代码更清晰，测试增强，为后续工具调用功能扩展奠定基础。

关联脉络

与历史 PR 的关联：

- PR #20310（tokenizer 改进）：同样涉及解析逻辑优化，但专注于非流式请求处理；本 PR 在工具调用解析方面延续了类似的设计模式，强调性能与正确性权衡。

从近期 PR 趋势看，仓库在持续优化解析和约束解码机制（如 #22404 修复 CUDA Graph 捕获），本 PR 是这一方向的重要补充，专注于提升多模型工具调用支持。