

PR #39227 完整报告

sgl-project/sglang

Force reasoning mode for GLM-5.3 chat templates

合并时间: 2026-09-15 13:16

原文链接: <http://prhub.com.cn/sgl-project/sglang/pull/39227>

执行摘要

- 一句话: 新增 GLM-5.3 模板强制推理规则, 修复推理内容泄漏
- 推荐动作: 值得精读。这是一个小改动但信息密度高的 PR: 适合学习 sglang 模板检测系统的规则设计 (special_case 语义、规则顺序敏感性)、跨 PR 冲突的处理方式, 以及如何写 "必红" 回归测试。特别建议关注 #37977 合并后规则整合是否落实, 否则 GLM-5.3 的 effort 映射会失效。

功能与动机

GLM-5.3 的 chat 模板在生成 prompt 中直接打开 `thinking` (模型只输出关闭标签), 而现有规则没有一条命中, `detect_reasoning_pattern` 返回 (False, None)。Glm45Detector 的拆分完全依赖 `force_reasoning`: 普通请求靠 fallback (`reasoning_config` 为 None 时取 `reasoning_default`) 正常工作, 但 `reasoning_effort: "none"` 和 `chat_template_kwargs: {enable_thinking: false}` 两种请求会解析出 `enable_thinking=False`——该开关已被 GLM-5.3 移除, prompt 仍会打开 `thinking`, 导致推理内容泄漏到 content (实测输出 `17*23 = 391 response391`), 且 `sgl-eval` 会把推理 token 记入 `num_answer_tokens`。该问题在 #38297 中被观察到、被推迟到 #37977, 而 #37977 的规则未设置 `special_case`, `has_toggle` 为 false, 同样走回退路径。PR body 原话: "No rule recognizes these templates, so `detect_reasoning_pattern` returns (False, None)"。

实现拆解

1. 新增检测规则 (核心变更): 在 `python/sglang/srt/parser/template_detection.py` 的 `REASONING_MODE_RULES` 元组中、`force_reasoning_pattern` 之后插入 `glm53_always_think` 规则, 值为 `ReasoningToggleConfig(special_case="always")`, `predicate` 为 `lambda ctx: _is_glm53(ctx)`。`special_case="always"` 使 `_get_reasoning_from_request` 在请求级开关判断之前短路, 从而忽略 `reasoning_effort: "none"` 与 `chat_template_kwargs: {enable_thinking: false}`, 统一走强制推理分支。
2. 判定边界: `predicate` 复用 #38297 引入的 `_is_glm53`, 要求模板包含 `[gMASK]` 且不含 `enable_thinking`, 因此 GLM-4.5/4.6/4.7 等仍保留 `toggle` 规则的模板不会命中, 只有移除开关的 GLM-5.3 模板进入新规则。
3. 回归测试: 在 `test/registered/unit/parser/test_template_manager.py` 新增 `test_glm53_effort_template_forces_reasoning`, 对 `_glm53_template("+)` 断言 `force=True` 且 `config == ReasoningToggleConfig(special_case="always")`; 回退规则后

该用例必红，保证检测状态与真实触发条件绑定。

4. 测试裁剪（提交 1865e9d6）：JustinTong0323 移除了与兄弟测试重复的断言（推理 / 内容拆分已由 TestGlm45Detector.test_forced_reasoning_mode 覆盖，special_case="always" 由 test_get_reasoning_from_request_special_cases 覆盖），并删去 concat 矩阵（+ / ~），因为该运算符不是规则读取的对象。
5. 冲突预案（未实施）：PR body 给出与 #37977 的合并方案——单条规则同时携带 special_case="always"、effort_levels 与 effort_aliases，使 has_toggle=True 且 _get_reasoning_from_request 命中 special_case == "always" 分支，同时 toggle_param / effort_kwarg 保持 None，不改变 _apply_jinja_template、_get_reasoning_toggle_param 与 apply_reasoning_enabled 的行为。

关键文件：

- python/sclang/srt/parser/template_detection.py（模块 模板检测；类别 source；类型 core-logic；符号 glm53_always_think）：核心变更所在：在 REASONING_MODE_RULES 元组中新增 glm53_always_think 规则，以 ReasoningToggleConfig(special_case="always") 强制 GLM-5.3 推理模式，predicate 复用 _is_glm53。
- test/registered/unit/parser/test_template_manager.py（模块 测试用例；类别 test；类型 test-coverage；符号 test_glm53_effort_template_forces_reasoning）：新增回归测试 test_glm53_effort_template_forces_reasoning，验证 force=True 与 special_case="always" 配置，且无规则时用例必红。

关键符号：glm53_always_think, test_glm53_effort_template_forces_reasoning

关键源码片段

python/sclang/srt/parser/template_detection.py

核心变更所在：在 REASONING_MODE_RULES 元组中新增 glm53_always_think 规则，以 ReasoningToggleConfig(special_case="always") 强制 GLM-5.3 推理模式，predicate 复用 _is_glm53。

```
# REASONING_MODE_RULES 元组内、force_reasoning_pattern 之后的新增规则。
# 规则按声明顺序匹配、取第一条命中，因此位置敏感。
DetectionRule(
    name="glm53_always_think",
    # special_case="always" 让 _get_reasoning_from_request 在请求级开关
    # 判断之前短路，reasoning_effort="none"、enable_thinking=false 全部失效；
    # toggle_param 与 effort_kwarg 保持 None，不改变其余模板渲染路径。
    value=ReasoningToggleConfig(special_case="always"),
    # GLM-5.3 的生成 prompt 直接打开 thinking 标签，输出里没有开启标签，
    # parser 无法靠文本特征判断推理边界；
    # 用 lambda 延迟求值，因为 _is_glm53 定义在元组下方。
    predicate=lambda ctx: _is_glm53(ctx),
)
```

test/registered/unit/parser/test_template_manager.py

新增回归测试 `test_glm53_effort_template_forces_reasoning`，验证 `force=True` 与 `special_case="always"` 配置，且无规则时用例必红。

```
# 回归测试：去掉 glm53_always_think 规则后，本用例会变红。
def test_glm53_effort_template_forces_reasoning(self):
    # 无规则命中时 detect_reasoning_pattern 返回 (False, None),
    # Glm45Detector 无法拆分 reasoning_content，推理内容泄漏进 content。
    vocab = ["<tool_call>", "<arg_key>", "<arg_value>", "<luserl>", "<lendoftextl>"]
    force, config, _ = self._detect(_glm53_template("+"), vocab)

    # 规则生效后：force_reasoning=True，配置为无条件强制推理。
    self.assertTrue(force)
    self.assertEqual(config, ReasoningToggleConfig(special_case="always"))
```

评论区精华

核心讨论来自 Issue 评论区。zRzRzRzRzRzRzR 认为 GLM-5.3 只支持 thinking 模式，非 thinking 模式性能会显著下降，建议使用 glm47 parser，并主张把映射规则移出 SGLang 实现、让用户通过环境变量配置；该意见未被采纳，PR 坚持服务端规则方案。b8zhong 则质疑 reasoning_parser 默认值仍为 None 而非 auto，认为其他模型也不应有副作用，且 GLM-5/5.1/5.2 用户会默认沿用旧设置——材料中未见直接回复。提交 1865e9d6 体现了测试裁剪共识：只保留无规则时必红的用例，避免与兄弟测试重复。

- GLM-5.3 强制推理与 parser 选择 (design): 未采纳环境变量方案；本 PR 以 `special_case="always"` 在服务端强制推理，parser 沿用 #38297 解析出的 glm45 / glm47。
- reasoning_parser 默认值 None vs auto (question): 材料中未见直接答复，PR 合并时未改动默认值。
- 测试裁剪：只保留必红用例 (testing): 已落实：仅保留 `test_glm53_effort_template_forces_reasoning` 作为回归锚点。
- 与 #37977 的规则冲突 (design): 本 PR 未实施合并，作为后续动作留给 #37977 的合并处理。

风险与影响

- 风险：
 1. 规则顺序耦合：REASONING_MODE_RULES 按声明顺序取第一条匹配，`glm53_always_think` 插在 `force_reasoning_pattern` 与 `mistral_reasoning_effort` 之间；若未来模板同时命中更早的规则，GLM-5.3 行为会随顺序改变。
 2. 与 #37977 冲突：两条规则都匹配 zai-org/GLM-5.3-Flash 与 RadixArk/GLM-5.3-NVFP4，后合并者对 GLM-5.3 是死代码——要么泄漏重开（`has_toggle=False` 回退），要么 `effort_aliases` 永不生效；PR 给方案但未实施。
 3. 强制覆盖请求语义：`special_case="always"` 使 `_get_reasoning_from_request` 短路，`reasoning_effort="none"`、`enable_thinking=false` 全部失效；Anthropic 端点显式 thinking: {"type": "disabled"} 返回 400。

4. 判定条件脆弱: `_is_glm53` 依赖 " 无 `enable_thinking`" 这一负向特征, 若官方模板恢复该开关, 规则会静默失效并回退旧路径。 - 影响: 覆盖 `zai-org/GLM-5.3-Flash` 与 `RadixArk/GLM-5.3-NVFP4` 两个已发布模型的模板, 统一推理拆分行为; 修复 `reasoning_tokens=0` 导致的计费与评测错记 (`sgl-eval` 把推理 token 记入 `num_answer_tokens`)。对 GLM-4.5/4.6/4.7 无行为变化, `_is_glm53` 已排除含 `enable_thinking` 的模板。改动范围小 (2 个文件、14 行), 但影响 GLM-5.3 全部推理请求的拆分正确性。 - 风险标记: 规则顺序敏感, 与 #37977 规则冲突, 强制覆盖请求级关闭, 判定依赖负向特征

关联脉络

- PR #38297 (标题未提供): PR body 引用: `_is_glm53` 来自 #38297, 本 PR 的 `predicate` 直接复用它; GLM-5.3 推理拆分问题最初在 #38297 中被观察到。
- PR #37977 (标题未提供): PR body 详细分析了与 #37977 的规则冲突: 两条规则均匹配 GLM-5.3 模板, 后合并者对 GLM-5.3 成为死代码, 并给出单规则合并方案。
- PR #38890 Support non-strict GLM47 tool calls with EBNF constraints: 同属 GLM 推理 / 工具调用解析链路, 涉及 `glm47 parser` 与 `reasoning` 相关配置, 与 GLM-5.3 解析链路相互影响。