PR #2261 完整报告

THUDM/slime

fix(rollout): restore partial continuation token budget

合并时间：2026-08-12 13:26

原文链接：http://prhub.com.cn/THUDM/slime/pull/2261

执行摘要

- 一句话：修复 partial rollout 续跑时重复使用完整响应预算
- 推荐动作：该 PR 值得精读，虽然改动小，但涉及正确性修复，且回归源于 1eed2493 重构。建议关注其根因与测试缺失，未来应补充单元测试覆盖 partial rollout 续跑预算计算。

功能与动机

Issue #2260 报告：启用 --partial-rollout 时，中断样本恢复后，SGLang 请求使用完整 max_new_tokens，导致请求 token 数超过上下文限制（如 19202 输入 + 24576 生成 > 40960），SGLang 返回 400，且通用 HTTP 重试机制对确定性错误无效重试。

实现拆解

1. 在 slime/rollout/sglang_rollout.py 的 *generate*() 函数中，在调用 _prepare_prompt_ids() 之后、构造 payload 之前，增加 sampling_params["max_new_tokens"] -= sample.response_length。
2. 在 slime/rollout/sglang_streaming_rollout.py 的 *generate_streaming*() 函数中做同样的修改，保证流式路径行为一致。
3. 因为响应预算在每次请求前独立扣除，不影响初始请求（此时 response_length 为 0）；当预算耗尽时，现有的零预算分支将样本标记为 TRUNCATED，不发送请求。
4. 未修改重试策略或上下文长度裁剪逻辑，超出历史行为范围，以最小改动恢复原有语义。

关键文件：

- slime/rollout/sglang_rollout.py（模块 训练引擎；类别 source；类型 *core-logic*）：主变更文件，在 *generate*() 中恢复响应预算减法，修复续跑超上下文问题。
- slime/rollout/sglang_streaming_rollout.py（模块 训练引擎；类别 source；类型 *core-logic*）：流式路径的对应修复，保证行为与普通路径一致。

关键符号：*generate*, *generate_streaming*

关键源码片段

slime/rollout/sglang_rollout.py

主变更文件，在 *generate*() 中恢复响应预算减法，修复续跑超上下文问题。

```
async def generate(args: Namespace, sample: Sample, sampling_params: dict[str, Any]) ->
Sample:
    """Generate using traditional SGLang router with token-based workflow"""
    # ... 状态初始化、URL 构造等 ...

    # 准备 prompt，包含已生成 token 的累积（partial rollout 场景）
    prompt_ids = _prepare_prompt_ids(sample, state.tokenizer, state.processor)

    # 关键修复：扣除已生成的 response token，避免续跑超预算
    # 初始请求时 response_length 为 0，行为不变；续跑时只请求剩余预算
    sampling_params["max_new_tokens"] -= sample.response_length

    # 预算为 0 时标记 TRUNCATED 并返回，不发送请求
    assert sampling_params["max_new_tokens"] >= 0, \
        f"max_new_tokens: {sampling_params['max_new_tokens']} should not be less than 0"
    if sampling_params["max_new_tokens"] == 0:
        sample.status = Sample.Status.TRUNCATED
        return sample

    # 构造请求 payload（略）...
```

slime/rollout/sglang_streaming_rollout.py

流式路径的对应修复，保证行为与普通路径一致。

```
async def generate_streaming(args: Namespace, sample: Sample, sampling_params: dict[str,
Any]) -> Sample:
    """Streaming counterpart to generate，基于 SSE 流式输出"""
    # ... 状态初始化、URL 构造等 ...

    # 准备 prompt，与普通路径相同
    prompt_ids = _prepare_prompt_ids(sample, state.tokenizer, state.processor)

    # 同样扣除已生成 token，保证流式续跑遵守总响应预算
    sampling_params["max_new_tokens"] -= sample.response_length

    # 预算为 0 时标记 TRUNCATED 并返回
    assert sampling_params["max_new_tokens"] >= 0, \
        f"max_new_tokens: {sampling_params['max_new_tokens']} should not be less than 0"
    if sampling_params["max_new_tokens"] == 0:
        sample.status = Sample.Status.TRUNCATED
        return sample

    # 构造流式 payload（略）...
```

评论区精华

无 Review 评论或讨论。

- 暂无高价值评论线程

风险与影响

- 风险：该改动简单且局部，风险较低。但缺少单元测试，可能未覆盖以下场景：sample.response_length 大于 max_new_tokens 时，断言会失败（负值预算未处理，仍是原有断言行为）；流式路径中 SSE 中断后样本的 response_length 计数的准确性依赖上游状态维护，若在中断前未正确更新样本，可能导致预算扣除不足。
- 影响：影响范围限于 partial rollout 功能，特别是 oversampling 产生的中断样本续跑。正面影响：避免超出上下文限制的请求及无效重试，提升训练稳定性。负面影响：若中断样本的 response_length 统计不准确，可能导致续跑预算错误（过多或过少），但现有断言会暴露负值情况。团队影响小，无需配置或部署变更。
- 风险标记：缺少测试覆盖，核心路径变更，回归风险（源于重构）

关联脉络

- PR #2262 feat(glm5): align Megatron DeepEP training with SGLang rollout: 涉及 SGLang rollout 路径，属于相关功能区域。