PR #36726 完整报告

sgl-project/sglang

[Diffusion] Fix the five unit tests failing on main

合并时间：2026-08-28 12:18

原文链接：http://prhub.com.cn/sgl-project/sglang/pull/36726

执行摘要

- 一句话：修复 diffusion 五个单元测试，含两个 realtime 相机运行时 bug
- 推荐动作：值得精读。重点看 ControlStateQueue 的“无信号 vs 保持电平”状态建模（_received_any + mark_received），以及 CI glob 遗漏导致测试静默失效的根因修复方式；对 diffusion/realtime 模块的开发者是很好的参考。insight 主要来自语义设计与测试工程实践，而非算法创新。

功能与动机

PR body 明确指出：Five tests under python/sglang/multimodal_gen/test/unit fail on current main. They never ran in CUDA CI — the nested unit directories (realtime/, sana_wm/) sit outside the CI file glob, and one expectation depended on the host GPU — so the failures accumulated silently. 其中两个失败是真实运行时 bug：ControlStateQueue.sample_chunk 在从未收到控制时仍合成保持态转换，导致新会话把默认相机状态误报为活动信号；缓存 streaming 相机分支跳过了 dense 分支的 RMS downscale，破坏单 chunk 归约到 dense 计算的契约。

实现拆解

按提交顺序拆解为 4 步：

1. 修复控制信号语义（提交 1）：runtime/realtime/control_signals.py 的 ControlStateQueue 新增 _received_any 标志——push() 首次调用即置位，sample_chunk() 在无 pending 转换且从未收到控制时返回 None（不再合成默认状态），clear() 重置标志；新增 mark_received() 用于“无 transition 但需保持电平”的场景。配套 states/camera_control.py 的 receive_camera_action_script() 在 clear() 后调用 mark_received()，保证相机动作脚本本身算作信号，脚本排空后保持结束状态而非回到无信号。test/unit/realtime/test_realtime_runtime.py 更新两处过期期望：fresh 会话 sample_camera_actions(3) 应为 None；latent-prep stub 移除多余的 arch_config 嵌套。
2. 恢复缓存相机分支的 RMS downscale（提交 2）：runtime/models/dits/sana_wm_components.py 的 _cam_branch_cached 与 _cam_branch_softmax_cached 在 UCPE 投影后，对 q/k/v 调用 _downscale_to_reference_rms 缩回 pre-UCPE per-token RMS，与 dense 分支 _cam_branch/_cam_branch_softmax 数值对齐；单 chunk 无携带状态时精确退化为 dense 计算，由 test_cam_gdn_cached_single_chunk_reduces_to_dense 校验。dense 路径完全不动。

3. 修复过期测试 pin（提交 3）：test/unit/realtime/test_realtime_webui.py 将资源版本从 realtime-sr-v38/rgb-worker-v6 更新为 realtime-record-v49/rgb-worker-v10，同步 UI 字符串断言（如 setStatus("Receiving", "live")、decodeQueue.push()）并新增读取 playback_controller.js；test/unit/test_consistency_metrics.py 通过 monkeypatch 固定 SGLANG_DIFFUSION_CONSISTENCY_PLATFORM=h100，消除对宿主 GPU 的依赖。
4. 扩大 CI 发现范围（提交 4）：test/server/gpu_cases.py 的 _discover_unit_tests() 在非 AMD 平台直接返回 ["../unit"]，让 pytest 递归发现 realtime/、sana_wm/、progressive_resolution/ 等嵌套目录全部测试；AMD/ROCm 保留 vetted 列表。这从根因上杜绝"测试静默漏跑"，新嵌套测试无需注册即可被 CUDA CI 覆盖。

关键文件：

- python/sglang/multimodal_gen/runtime/realtime/control_signals.py（模块 控制信号；类别 source；类型 core-logic；符号 ControlStateQueue, push, sample_chunk, mark_received）：核心运行时 bug 修复：ControlStateQueue 新增 _received_any 标志与 mark_received()，规定"从未收到控制时采样返回 None、收到后保持电平"的语义，消除 fresh 会话误报默认相机状态的问题。
- python/sglang/multimodal_gen/runtime/models/dits/sana_wm_components.py（模块 模型组件；类别 source；类型 data-contract；符号 _cam_branch_cached, _cam_branch_softmax_cached, _downscale_to_reference_rms）：模型 forward 数值契约修复：缓存 streaming 相机分支恢复与 dense 分支相同的 per-token RMS downscale，保证单 chunk 无携带状态时精确退化为 dense 计算。
- python/sglang/multimodal_gen/test/server/gpu_cases.py（模块 测试编排；类别 test；类型 test-coverage；符号 _discover_unit_tests, _AMD_READY_NESTED_UNIT_TESTS）：CI 根因修复：_discover_unit_tests 在 CUDA lane 改为递归运行整棵 unit 树，让 realtime/、sana_wm/ 等嵌套测试进入 CI，杜绝再次静默漏跑。
- python/sglang/multimodal_gen/runtime/realtime/states/camera_control.py（模块 控制信号；类别 source；类型 core-logic；符号 receive_camera_action_script）：receive_camera_action_script 在 clear() 后调用 mark_received()，把脚本本身视为信号，脚本排空后保持结束状态而非回到无信号。
- python/sglang/multimodal_gen/test/unit/realtime/test_realtime_webui.py（模块 界面测试；类别 test；类型 test-coverage；符号 test_realtime_webui_presets_do_not_emit_camera_scripts）：webui 测试 pin 更新：资源版本与 UI 字符串从 realtime-sr-v38 更新到 realtime-record-v49，并新增 playback_controller.js 的断言。
- python/sglang/multimodal_gen/test/unit/realtime/test_realtime_runtime.py（模块 运行时测试；类别 test；类型 test-coverage；符号 test_lingbot_realtime_state_uses_control_script_and_prompt_queues, test_realtime_chunk_latent_preparation_uses_chunk_spec）：修复两处过期期望：fresh 会话采样应为 None 而非空列表；latent-prep stub 移除多余的 arch_config 嵌套。
- python/sglang/multimodal_gen/test/unit/test_consistency_metrics.py（模块 一致性测试；类别 test；类型 test-coverage）：通过 monkeypatch 固定 SGLANG_DIFFUSION_CONSISTENCY_PLATFORM=h100，消除测试对宿主 GPU 的依赖，使候选路径解析逻辑在任何机器上可复现。

关键符号：ControlStateQueue.push, ControlStateQueue.sample_chunk, ControlStateQueue.mark_received, ControlStateQueue.clear, receive_camera_action_script, _cam_branch_cached, _cam_branch_softmax_cached, _discover_unit_tests

关键源码片段

python/sglang/multimodal_gen/runtime/realtime/control_signals.py

核心运行时 bug 修复：ControlStateQueue 新增 _received_any 标志与 mark_received()，规定“从未收到控制时采样返回 None、收到后保持电平”的语义，消除 fresh 会话误报默认相机状态的问题。

```
class ControlStateQueue:
    """State-mode sampler for level-triggered realtime controls."""

    def __init__(self, *, default_item, min_pulse_items=1, max_transitions=512) -> None:
        self.default_item = default_item
        self.min_pulse_items = min_pulse_items
        self._pending: deque[ControlStateTransition] = deque(maxlen=max_transitions)
        self._current_item = default_item
        self._current_seq_id: int | None = None
        self._latest_sampled_seq_id: int | None = None
        # 从未收到 transition 的队列没有可保持的状态：
        # 采样时返回 None，调用方整体省略该控制；
        # 收到第一个 transition 之后电平保持，default 值也视为保持态。
        self._received_any = False

    def clear(self) -> None:
        self._pending.clear()
        self._current_item = self.default_item
        self._current_seq_id = None
        self._latest_sampled_seq_id = None
        self._received_any = False

    def push(self, transition: ControlStateTransition) -> None:
        self._pending.append(transition)
        self._received_any = True

    def mark_received(self) -> None:
        """无 transition 也武装保持：替换脚本本身就是信号，
        因此脚本排空后继续采样其结束状态作为保持电平。"""
        self._received_any = True

    def sample_chunk(self, chunk_size: int) -> list[Any] | None:
        if chunk_size <= 0:
            return None

        transitions = self._drain_pending()
```

```
        if not transitions:
            # 关键修复：从未收到控制时返回 None，而不是把默认状态当作活动信号
            if not self._received_any:
                return None
            self._latest_sampled_seq_id = self._current_seq_id
            return [self._copy_item(self._current_item) for _ in range(chunk_size)]

        pulse = self._latest_non_default_transition(transitions)
        final = transitions[-1]
        self._current_item = final.payload
        self._current_seq_id = final.seq_id

        if pulse is not None and pulse.payload != final.payload:
            # 短暂脉冲：先填充 min_pulse_items 个脉冲值再切换为最终电平
            pulse_items = min(self.min_pulse_items, chunk_size)
            chunk = [self._copy_item(pulse.payload) for _ in range(pulse_items)]
            chunk.extend(
                self._copy_item(final.payload) for _ in range(chunk_size - pulse_items)
            )
            self._latest_sampled_seq_id = (
                final.seq_id if len(chunk) > pulse_items else pulse.seq_id
            )
            return chunk

        self._latest_sampled_seq_id = final.seq_id
        return [self._copy_item(final.payload) for _ in range(chunk_size)]
```

python/sglang/multimodal_gen/runtime/models/dits/sana_wm_components.py

模型 forward 数值契约修复：缓存 streaming 相机分支恢复与 dense 分支相同的 per-token RMS downscale，保证单 chunk 无携带状态时精确退化为 dense 计算。

```
  # _cam_branch_cached 内部：缓存 streaming 分支与 dense 分支保持同一数值契约
  q_pre_dn = q_bhnd.permute(0, 1, 3, 2)
  q_dn = q_proj.permute(0, 1, 3, 2)
  k_pre_dn = k_bhnd.permute(0, 1, 3, 2)
  k_dn = k_proj.permute(0, 1, 3, 2)
  v_pre_dn = v_bhnd.permute(0, 1, 3, 2)
  v_dn = v_proj.permute(0, 1, 3, 2)

  # 与 _cam_branch 相同的 per-token RMS downscale：
  # 单 chunk 无携带状态时精确退化为 dense scan 的计算路径
  q_dn = _downscale_to_reference_rms(q_pre_dn, q_dn)
  k_dn = _downscale_to_reference_rms(k_pre_dn, k_dn)
  v_dn = _downscale_to_reference_rms(v_pre_dn, v_dn)

  # 膨胀系数基于缩放前后的 K 范数比值计算，修正 beta
  pre_ucpe_k_norm = torch.linalg.vector_norm(
      k_pre_dn.float(), dim=2, keepdim=True
```

```
).clamp_min(1e-6)
post_ucpe_k_norm = torch.linalg.vector_norm(
    k_dn.float(), dim=2, keepdim=True
).clamp_min(1e-6)
inflation_sq = (post_ucpe_k_norm / pre_ucpe_k_norm) ** 2
frame_inflation_sq = inflation_sq.view(B, self.heads, T, S).mean(dim=-1)
if beta.ndim == 3:
    beta = beta / frame_inflation_sq.clamp_min(1.0)
elif beta.ndim == 4:
    beta = beta / frame_inflation_sq.unsqueeze(-1).clamp_min(1.0)

# 无状态 Triton scan 的缓存调用，携带 camera K 状态
out, cam_state = _single_path_delta_scan_cached(
    q_dn.float(),
    k_dn.float(),
    v_dn.float(),
    beta.float(),
    decay.float(),
    init_state_kv=kv_cache[_SLOT_CAM_K],
)
```

python/sglang/multimodal_gen/test/server/gpu_cases.py

CI 根因修复：_discover_unit_tests 在 CUDA lane 改为递归运行整棵 unit 树，让 realtime/、sana_wm/ 等嵌套测试进入 CI，杜绝再次静默漏跑。

```
def _discover_unit_tests() -> list[str]:
    unit_dir = Path(__file__).resolve().parent.parent / "unit"
    if not unit_dir.is_dir():
        return []
    if not current_platform.is_hip():
        # pytest 递归进入目录：全部单元测试都会运行（含嵌套子目录），
        # 新文件无需注册即可被 CI 发现，防止再次出现测试静默漏跑
        return ["../unit"]
    # AMD/ROCm 保留 vetted 集合：仅跑已验证通过的扁平文件与嵌套测试
    flat = [f"../unit/{f.name}" for f in unit_dir.glob("test_*.py") if f.is_file()]
    nested = [
        f"../unit/{rel}"
        for rel in _AMD_READY_NESTED_UNIT_TESTS
        if (unit_dir / rel).is_file()
    ]
    return sorted(flat + nested)
```

评论区精华

PR 级评论：审核者 AgainstEntropy 在查看修复过程后提出 We can expand the searching paths to include these missing unit tests，这正是提交 4 的实现方式——CUDA lane 改为递归运行整棵 unit 树，AMD lane 维持 vetted 集合；该建议落地后审核者 APPROVED。行内评论：control_signals.py 第 418 行处 duplication here，指 push() 与新增的

mark_received() 都执行 self._received_any = True，属轻微重复；无后续回复，PR 按现状合并，单行标志赋值的重复可接受。

- CI 搜索路径应包含嵌套单元测试目录 (design): 提交 4 实现该建议：CUDA lane 的 _discover_unit_tests 改为返回 ["../unit"] 由 pytest 递归发现，AMD/ROCm 保留 vetted 集合；随后审核者 APPROVED。
- push 与 mark_received 重复设置 _received_any (style): 无后续回复，PR 按现状合并；重复仅为单行标志赋值，可接受的轻微冗余。

风险与影响

- 风险：
  1. 控制语义变更：sample_chunk() 现在对从未收到控制的队列返回 None 而非默认项，所有调用方必须能处理 None；若存在测试未覆盖的调用点（如脚本与状态队列组合路径），可能引入遗漏。
  2. 模型数值路径变更：sana_wm_components.py 的缓存 streaming 分支数值发生变化，虽然是对齐 dense 路径，但 streaming 输出、缓存状态与 beta 修正的数值均有差异，依赖旧输出的场景需回归验证；PR 用单 chunk 归约测试收紧约束，但多 chunk 携带状态场景仍依赖既有测试。
  3. CI 覆盖扩大：CUDA lane 从扁平 glob 扩展为递归整树，运行时间增长，且可能首次暴露其他此前静默失败的嵌套测试（这是有意的，但短期可能引入 CI 不稳定）。
  4. 测试与前端强耦合：test_realtime_webui.py 大量断言 webui 资源版本与 UI 字符串，前端迭代必须同步更新测试，存在维护摩擦。 - 影响：用户侧：realtime 视频生成 fresh 会话不再把默认相机状态误报为活动信号；流式缓存相机路径与 dense 数值对齐，输出一致性提升。系统侧：CUDA CI 对 diffusion 单元测试的覆盖面从扁平 glob 扩大为整棵 unit 树，质量门禁更真实。团队侧：multimodal_gen 测试基线从“偶然全绿”变为“真正全绿”，降低后续改动的回归误判；AMD/ROCm lane 不受影响，仍走 vetted 列表。 - 风险标记：控制语义变更，模型数值路径变更，CI 覆盖范围扩大，测试与前端强耦合

关联脉络

- PR #36658 [multimodal_gen] fix: make tail_attn_meta CUDA-graph capturable: 同属 multimodal_gen/diffusion 模块的测试与运行时修复线，且都涉及嵌套单元测试目录（realtime/）在 CI 中的覆盖问题。