PR #34242 完整报告

sgl-project/sglang

[diffusion] Warn when BCG disables Cache-DiT

合并时间：2026-08-14 16:45

原文链接：http://prhub.com.cn/sgl-project/sglang/pull/34242

执行摘要

- 一句话：BCG 下 Cache-DiT 静默禁用，现补一次性警告
- 推荐动作：值得快速浏览而非精读：改动虽小，但“一次性告警 + 请求级路径覆盖 + 测试对全局去重函数做 cache_clear 隔离”的设计可作为同类静默失效问题的修复模板。若后续要扩展 Cache-DiT 与 BCG 的兼容性，本 PR 的告警位置是天然入口。

功能与动机

Issue #34177 明确指出：同时启用 --enable-breakable-cuda-graph 和 SGLANG_CACHE_DIT_ENABLED=1 时，Cache-DiT 既不生效、也不报错、也不警告，“唯一症状是 1.00x 加速”，容易被用户误读为“Cache-DiT 对该模型无效”。issue 还强调仅启动时检查不够，因为 MiniMax-H3 会在 quality="high" 批次动态挂载 Cache-DiT，告警应覆盖请求级路径，并且“每进程一次即可，不应逐请求记录”。

实现拆解

1. 定位早退分支：python/sglang/multimodal_gen/runtime/pipelines_core/stages/denoising.py 中 DenoisingStage._maybe_enable_cache_dit 开头即检查 server_args.enable_breakable_cuda_graph，命中则直接 return，这是 Cache-DiT 被静默禁用的根因位置。
2. 注入告警：在 return 之前新增 if self._cache_dit_requested(): logger.warning_once(...)。_cache_dit_requested() 统一表达“环境变量或请求级显式请求”，因此启动路径和 MiniMax-H3 动态挂载路径共享同一判定；logger.warning_once 基于 logging_utils._print_warning_once 做进程级去重，未请求时保持原有静默行为。
3. 测试配套（两个文件、三个用例）：test_diffusion_bcg_padding.py 新增 test_bcg_warns_when_cache_dit_is_requested（连续两次调用断言 logger.warning 只调用一次且 stacklevel=2，测试前后 _print_warning_once.cache_clear() 重置全局去重缓存）和 test_bcg_does_not_warn_when_cache_dit_is_not_requested（patch 返回 False，断言 warning_once 不被调用）；test_minimax_h3_admission.py 新增 test_high_quality_request_warns_when_bcg_suppresses_cache_dit，构造 quality="high" 且 _explicit_fields={"quality"} 的批次验证请求级路径。
4. 配置与部署配套：无配置文件、schema 或文档改动，符合 PR body 中 diagnostic-only change 的定位。CI Extra 存在一次失败（Run #31369399616），材料未给出失败原因，需以 CI 日志为准。

关键文件：

- python/sglang/multimodal_gen/runtime/pipelines_core/stages/denoising.py（模块 去噪管线；类别 source；类型 core-logic；符号 _maybe_enable_cache_dit, _cache_dit_requested）：核心逻辑变更处：在 _maybe_enable_cache_dit 的 BCG 早退分支中新增 _cache_dit_requested() 判断并发出进程级一次性警告，是本次修复的关键入口。
- python/sglang/multimodal_gen/test/unit/test_diffusion_bcg_padding.py（模块 图形捕获；类别 test；类型 test-coverage；符号 test_bcg_warns_when_cache_dit_is_requested, test_bcg_does_not_warn_when_cache_dit_is_not_requested）：新增两个单元测试，验证请求了 Cache-DiT 且 BCG 启用时只告警一次、未请求时不告警，并验证 stacklevel=2 与进程级去重语义。
- python/sglang/multimodal_gen/test/unit/test_minimax_h3_admission.py（模块 请求准入；类别 test；类型 test-coverage；符号 test_high_quality_request_warns_when_bcg_suppresses_cache_dit）：覆盖 MiniMax-H3 请求级动态挂载路径：quality="high" 批次在 BCG 抑制下同样触发告警，补全 issue 要求的请求级场景。

关键符号：_maybe_enable_cache_dit, _cache_dit_requested, _print_warning_once

关键源码片段

python/sglang/multimodal_gen/runtime/pipelines_core/stages/denoising.py

核心逻辑变更处：在 _maybe_enable_cache_dit 的 BCG 早退分支中新增 _cache_dit_requested() 判断并发出进程级一次性警告，是本次修复的关键入口。

```
# python/sglang/multimodal_gen/runtime/pipelines_core/stages/denoising.py
# 每个请求进入去噪流程时都会调用本方法，负责挂载或刷新 Cache-DiT。
# 当启用 breakable CUDA graph 时，Cache-DiT 的 step-skipping 控制流
# 不能被打进被捕获的 CUDA 图，因此原本在这里直接提前返回（静默禁用）。
# 本 PR 在返回前补上“仅当 Cache-DiT 被请求时”的一次性告警。
def _maybe_enable_cache_dit(
    self, num_inference_steps: int | tuple[int, int], batch: Req
) -> None:
    if self.server_args.enable_breakable_cuda_graph:
        # Cache-DiT wraps transformer.forward with step-skipping control
        # flow that must not be baked into a captured CUDA graph.
        if self._cache_dit_requested():
            # warning_once 基于 _print_warning_once 实现进程级去重，
            # 从环境变量或请求级（如 MiniMax-H3 的 quality="high"）触发的
            # 首次抑制才会输出，避免逐请求刷日志。
            logger.warning_once(
                "Cache-DiT was requested but is disabled because breakable "
                "CUDA graphs are enabled."
            )
        return

    # 新请求到达时需要刷新 cache-dit 上下文；以下为原有逻辑，本 PR 未改动。
    if self._cache_dit_enabled:
        primary_num_steps, secondary_num_steps = self._cache_dit_step_counts(
```

```
            num_inference_steps
        )
        # ... refresh_context_on_transformer / refresh_context_on_dual_transformer ...
        return
```

python/sglang/multimodal_gen/test/unit/test_diffusion_bcg_padding.py

新增两个单元测试，验证请求了 Cache-DiT 且 BCG 启用时只告警一次、未请求时不告警，并验证 stacklevel=2 与进程级去重语义。

```
# 验证"请求了 Cache-DiT 但被 BCG 抑制"时只告警一次。
# _print_warning_once 是进程级去重的底层函数，测试前后清理其缓存，
# 防止其他用例或本用例重复执行时影响断言。
def test_bcg_warns_when_cache_dit_is_requested(self):
    self.stage.server_args = SimpleNamespace(enable_breakable_cuda_graph=True)
    _print_warning_once.cache_clear()
    self.addCleanup(_print_warning_once.cache_clear)

    with (
        # 模拟环境变量或请求级显式请求 Cache-DiT。
        patch.object(self.stage, "_cache_dit_requested", return_value=True),
        # warning_once 内部最终调用 logger.warning(..., stacklevel=2),
        # 这里 patch 住 warning，断言进程内只输出一次。
        patch.object(denoising_module.logger, "warning") as warning,
    ):
        # 连续两次调用（warmup 与非 warmup）都应被去重为一次。
        self.stage._maybe_enable_cache_dit(1, SimpleNamespace(is_warmup=True))
        self.stage._maybe_enable_cache_dit(1, SimpleNamespace(is_warmup=False))

    warning.assert_called_once_with(
        "Cache-DiT was requested but is disabled because breakable CUDA "
        "graphs are enabled.",
        stacklevel=2,
    )
```

评论区精华

本 PR 没有 review 评论（review_comments_count = 0），唯一评论是维护者 mickqian 的 /tag-and-rerun-ci，用于重跑 CI。技术决策全部沉淀在关联 Issue #34177 的讨论中：

> "A startup-time check alone is not sufficient. Cache-DiT can also be requested per request … the warning should also fire the first time a request asks for Cache-DiT while BCG suppresses it. Once per process is enough; this should not log per request." 实现完整遵循了这三条约束：启动与请求级双覆盖、首次触发、进程级一次性。

- 告警触发时机与频率 (design): PR 在 _maybe_enable_cache_dit 的 BCG 早退分支内调用 logger.warning_once，同时覆盖环境变量启动路径与请求级路径，并依赖 _print_warning_once 实现进程级去重；新增 3 个单测分别覆盖两类路径。

风险与影响

- 风险：回归风险极低：模型执行路径零改动（仅新增日志分支），test_diffusion_bcg_padding.py 与 test_minimax_h3_admission.py 共 34 个用例在 Python 3.12 CPU 下通过。性能上，每个请求多一次 _cache_dit_requested() 判定，开销可忽略；告警输出被 warning_once 去重，热路径不会反复打日志，但需留意 _cache_dit_requested() 的实现本身不应过重。日志语义依赖 logger.warning_once 的进程级去重（_print_warning_once），若未来替换日志工具需保持去重语义，否则可能退化为逐请求刷日志。测试隔离方面，全局去重缓存通过 cache_clear() 清理，若其他测试并发使用同一全局函数可能产生交叉影响，当前单测顺序执行风险低。
- 影响：影响范围为开启 BCG 且请求 Cache-DiT 的 diffusion 用户（如 MiniMax-H3 高画质批次、Qwen-Image 等模型）：从“静默 1.00x”变为明确告警，显著降低配置冲突的排查成本。对团队而言，改动集中在去噪管线的一处早退分支，维护面小，且为后续 Cache-DiT 与 BCG 兼容性改进提供了明确的告警入口。
- 风险标记：热路径新增判定分支，告警去重依赖日志工具，全局日志缓存跨用例污染

关联脉络

- PR #34121 [Diffusion] Fix cache-first fast path accepting a metadata-only snapshot: 同属 diffusion 运行时缓存行为修复线，强调 cache 静默失效的用户可观测性。
- PR #34650 feat(diffusion): rebuild MiniMax-H3 AdaLN outputs on demand: 改动了 MiniMax-H3 模型管线与 diffusion 运行时，与本 PR 的请求级 quality="high" 挂载路径直接相关。