PR #33564 完整报告

sgl-project/sglang

Fix Nightly NV CI

合并时间：2026-08-06 09:06

原文链接：http://prhub.com.cn/sgl-project/sglang/pull/33564

执行摘要

- 一句话：修复 NgramEmbedding 传 0 维 tensor 致 Triton 编译崩溃
- 推荐动作：值得精读：核心修复仅一行 int()，但揭示了「CUDA tensor 传入需 tl.constexpr 的 Triton 参数」这一通用崩溃模式；更重要的是 overrides.py 中 LongcatFlash 排除背后的架构语义——LongCat 的 top-k 选择跨越零专家 logits，与 trtllm-gen fused routing 不兼容，是理解 LongCat 与 DeepSeek 家族 MoE 差异的关键材料。

功能与动机

PR body 明确指出：NgramEmbedding passed exclusive_oe_embedder_size_sums[-1]（一个 0 维 CUDA tensor）作为 num_embeddings，fused Triton embedding kernel 在该位置需要 tl.constexpr int，拿到 tensor 对象后编译失败，报 TypeError("cannot convert 61341792 of type <class 'torch.Tensor'> to tensor")，直接击穿 nightly test_longcat_flash_lite_fp8.py。作者用 2-rank 复现验证了修复前后行为，并确认输出与非 Triton masked 路径 bit-identical。后续提交进一步说明：B200 nightly 命中 assert TopKOutputChecker.format_is_bypassed(topk_output)，因为 LongCat 被 _deepseek_moe_quant_resolution 错误强制到 trtllm-gen MoE runner，其 fused routing 无法表达覆盖零专家 logits 的 top-k 选择。

实现拆解

1. 核心修复（python/sglang/srt/layers/n_gram_embedding.py）：NgramEmbedding.__init__ 在 CUDA 上构造 exclusive_oe_embedder_size_sums（int32 tensor）后，把 int(self.exclusive_oe_embedder_size_sums[-1]) 传给 VocabParallelEmbedding 作为 num_embeddings。原代码直接传 0 维 CUDA tensor，fused Triton embedding kernel 需要 tl.constexpr int 才能展开，编译期报 TypeError。该文件被 LongCat 家族（Flash-Lite / Flash-Chat / 2.0）共用，一处修复同时覆盖三个模型。
2. MoE runner 排除（python/sglang/srt/arg_groups/overrides.py）：_deepseek_moe_quant_resolution 的 moe_runner_backend = flashinfer_trtllm 强制分支新增 and not model_arch.startswith("LongcatFlash")。背景是 LongcatFlashConfig 会把架构归一化为 LongcatFlashForCausalLM，落入 _DEEPSEEK_FAMILY_ARCHS 被强制到 trtllm-gen；但 LongCat 的 top-12 从 256 路由专家 + 128 零专家的 logits 中选取，

trtllm-gen 的 fused routing 只能看到 256 路由 logits，触发 TopKOutputChecker 断言。函数末尾对 fp8_gemm_runner_backend 的 LongCat 覆盖保持不变。

3. 测试重构（test/registered/8-gpu-models/test_longcat_flash_lite_fp8.py）：抽出 _run helper 统一执行 run_combined_tests；test_longcat_flash_lite_fp8 只保留 EP8+none 基线；test_longcat_flash_lite_fp8_deepep 独立成测试并 @unittest.skip，原因是 DeepEP low-latency dispatch 在 internode_ll.cu 中断言 num_topk <= kNumMaxTopK（值为 11），LongCat moe_topk 为 12。同时 register_cuda_ci 的 suite 从 nightly-8-gpu-common 改为 nightly-8-gpu-h200，LongCat 不再跑 B200。
4. GLM-5.2 配套（test/registered/8-gpu-models/test_glm52_fp8.py）：--mem-fraction-static 从 0.85 降至 0.8，为 DP8 CUDA graph capture 预留显存；提交信息明确该目的，rerun 在 H200 与 B200 均通过。
5. 配套验证：issue 中通过 /rerun-test 迭代验证：GLM-5.2 双端通过；LongCat 修复后 H200 通过、B200 仍有失败，最终通过收窄 suite 与跳过 deepep 变体收敛，PR 合并。

关键文件：

- test/registered/8-gpu-models/test_longcat_flash_lite_fp8.py（模块 模型测试；类别 test；类型 test-coverage；符号 test_longcat_flash_lite_fp8, _run, test_longcat_flash_lite_fp8_deepep）：该 PR 的直接验证目标：重构出 _run 统一执行，把 deepep 变体拆成独立测试并跳过（DeepEP top-k 上限 11 与 LongCat moe_topk=12 冲突），并把 suite 从 nightly-8-gpu-common 改为 nightly-8-gpu-h200，使 H200 夜间 CI 转绿。
- python/sglang/srt/layers/n_gram_embedding.py（模块 嵌入层；类别 source；类型 core-logic；符号 NgramEmbedding.init）：核心修复位置：oe_embeder 的 num_embeddings 之前是 0 维 CUDA tensor，Triton 内核无法把它展开为 tl.constexpr，加 int() 后编译通过；该文件被 LongCat 家族共用。
- python/sglang/srt/arg_groups/overrides.py（模块 参数覆盖；类别 source；类型 core-logic；符号 _deepseek_moe_quant_resolution）：第二个关键修复：把 LongcatFlash 从 flashinfer_trtllm MoE runner 强制覆盖中排除，避免 B200 上 trtllm-gen fused routing 触发 TopKOutputChecker 断言。
- test/registered/8-gpu-models/test_glm52_fp8.py（模块 模型评测；类别 test；类型 test-coverage；符号 test_glm52_fp8）：配套调整：mem-fraction-static 0.85→0.8，为 DP8 CUDA graph capture 腾出显存；commit 信息明确该目的，rerun 双端通过。

关键符号：NgramEmbedding.init, _deepseek_moe_quant_resolution, test_longcat_flash_lite_fp8, test_longcat_flash_lite_fp8_deepep, _run

关键源码片段

test/registered/8-gpu-models/test_longcat_flash_lite_fp8.py

该 PR 的直接验证目标：重构出 _run 统一执行，把 deepep 变体拆成独立测试并跳过（DeepEP top-k 上限 11 与 LongCat moe_topk=12 冲突），并把 suite 从 nightly-8-gpu-common 改为 nightly-8-gpu-h200，使 H200 夜间 CI 转绿。

```
# 拆分后的测试入口：_run 统一执行，deepep 变体独立成测试并跳过。
```

```
def _run(self, variant: ModelLaunchSettings):
    run_combined_tests(
        models=[variant],
        test_name="LongCat-Flash-Lite-FP8",
        # gsm8k 200q 5-shot 实测基线约 0.82-0.84，门限 0.78 吸收采样噪声
        accuracy_params=AccuracyTestParams(
            dataset="gsm8k",
            baseline_accuracy=0.78,
            num_examples=200,
        ),
        performance_params=PerformanceTestParams(
            profile_dir="performance_profiles_longcat_flash_lite_fp8",
        ),
    )


@unittest.skip(
    "Blocked: DeepEP low-latency dispatch asserts num_topk <= kNumMaxTopK "
    "(11 in internode_ll.cu), LongCat moe_topk is 12."
)
def test_longcat_flash_lite_fp8_deepep(self):
    self._run(
        ModelLaunchSettings(
            LONGCAT_FLASH_LITE_FP8_MODEL_PATH,
            tp_size=8,
            extra_args=COMMON_ARGS + ["--ep=8", "--moe-a2a-backend=deepep"],
            variant="TP8+EP8+deepep",
        )
    )
```

python/sglang/srt/layers/n_gram_embedding.py

核心修复位置：oe_embeder 的 num_embeddings 之前是 0 维 CUDA tensor，Triton 内核无法把它展开为 tl.constexpr，加 int() 后编译通过；该文件被 LongCat 家族共用。

```
# 构造 over-embedding 相关缓冲区：exclusive_oe_embedder_size_sums 是
# CUDA 上的 int32 tensor，逐项累加每个 over-embedding 子表的大小。
self.exclusive_oe_embedder_size_sums = torch.zeros(
    [over_embedding_k * (over_embedding_n - 1) + 1],
    dtype=torch.int32,
    device="cuda",
)
for i in range(over_embedding_k * (over_embedding_n - 1)):
    self.exclusive_oe_embedder_size_sums[i + 1] = (
        self.exclusive_oe_embedder_size_sums[i] + int(over_embedding_m + i * 2 + 1)
    )

# 关键修复：不能把 0 维 CUDA tensor 直接当作 num_embeddings 传给
# VocabParallelEmbedding，否则 fused Triton embedding kernel 在把
# tl.constexpr 参数展开为常量时会拿到 tensor 对象并编译失败，
```

```
# 报错为 TypeError: cannot convert ... of type torch.Tensor to tensor。
# 用 int() 取标量后，Triton 路径与非 Triton masked 路径输出一致。
self.oe_embeder = VocabParallelEmbedding(
    num_embeddings=int(self.exclusive_oe_embedder_size_sums[-1]),
    embedding_dim=oe_hidden_dim,
    use_attn_tp_group=use_attn_tp_group,
)
```

python/sglang/srt/arg_groups/overrides.py

第二个关键修复：把 LongcatFlash 从 flashinfer_trtllm MoE runner 强制覆盖中排除，避免 B200 上 trtllm-gen fused routing 触发 TopKOutputChecker 断言。

```
# _deepseek_moe_quant_resolution 中决定 MoE runner 的关键分支。
# 原逻辑对 DeepSeek 家族（含 LongcatFlash）一律强制 flashinfer_trtllm，
# 但 LongCat 的 top-12 是从 256 路由专家 + 128 零专家的 logits 里挑的，
# trtllm-gen 的 fused routing 只能看到 256 路由 logits，会触发
# TopKOutputChecker 断言，因此必须把 LongcatFlash 排除在外。
if (
    view.moe_a2a_backend == "none"
    and view.moe_runner_backend == "auto"
    # LongCat top-k 跨越零专家 logits，trtllm-gen 的 fused routing 看不到
    and not model_arch.startswith("LongcatFlash")
    and (
        quantization in ["fp8", "modelopt_fp8", "modelopt_fp4", "modelopt_mixed"]
        or is_kimi_k2_k25_thinking_int4
        or quantization is None
    )
):
    overrides["moe_runner_backend"] = "flashinfer_trtllm"
```

评论区精华

PR 没有 Review 评论，验证过程集中在 issue 的 /rerun-test 迭代（github-actions[bot] 输出）：

- 第一次 rerun LongCat：8-gpu-h200 与 8-gpu-b200 均 ❌。
- GLM-5.2 rerun：8-gpu-h200 与 8-gpu-b200 均 ❌。
- 修复后再次 rerun LongCat：8-gpu-h200 ❌、8-gpu-b200 ❌，B200 残留失败未在 PR 内定位。

作者通过把 suite 收窄到 nightly-8-gpu-h200 并跳过 DeepEP 变体收敛问题；Fridge003 approve 合并，说明 B200 残余失败被判定为不阻塞（可能是 DeepEP 上游 top-k 上限或 B200 环境问题）。

- LongCat-Lite 夜间测试在 B200 上仍有失败 (testing): 作者把 LongCat 测试 suite 从 nightly-8-gpu-common 收窄到 nightly-8-gpu-h200，并跳过 DeepEP 变体；Fridge003 approve 合并，B200 失败未在 PR 内根因定位。
- GLM-5.2 FP8 夜间测试显存占比调整 (testing): 变更合入，无争议。

风险与影响

- 风险：
  - 核心路径：n_gram_embedding.py 的 NgramEmbedding 被 LongCat 家族三个模型共用，int() 修复虽经验证 bit-identical，但该值如果再次被还原为 tensor 会立刻重现编译错误。
  - MoE runner：startswith("LongcatFlash") 是字符串前缀匹配，架构改名或出现相似前缀模型时该排除会失效；排除后 LongCat 在 Blackwell 上不再使用 trtllm-gen fused routing，性能路径或非最优。
  - 覆盖缩减：deepep 变体被跳过，moe_topk=12 与 DeepEP kNumMaxTopK=11 冲突期间无回归测试；LongCat 从 nightly-8-gpu-common 挪到 nightly-8-gpu-h200，B200 上不再运行该模型测试，sm_121 特有回归漏检。
  - GLM-5.2：mem-fraction-static=0.8 仅影响 CI 用例配置，不改变产品默认值，但更低静态显存占比理论上对长序列和吞吐有轻微影响。
- 影响：
  - 用户：LongCat 系列模型（Flash-Lite / Flash-Chat / 2.0）在 SM100 / Blackwell 上启动不再因 NgramEmbedding 或 MoE runner 选择而崩溃。
  - 系统：NV 夜间 CI 恢复稳定，LongCat 与 GLM-5.2 FP8 用例在 H200 通道通过；测试重构确立「变体拆分 + skip 阻塞项」的维护模式。
  - 团队：需要跟进 DeepEP 上游是否提高 low-latency top-k 上限，以及 B200 上 LongCat 失败的根本原因（目前被 suite 收窄掩盖）。
  - 风险标记：核心路径变更，测试覆盖缩减，架构名匹配脆弱，上游约束跳测

关联脉络

- PR #33748 [Fix] Vocab out of bounds in DSpark for Inkling-Small: 同属 vocab 大小边界类修复：一个在 speculative 侧用 padded vocab size 防越界，一个在 embedding 侧把 tensor 转 int 防编译崩溃。
- PR #33772 [CI] Temporarily disable prefill cuda graph for qwen3.5 nightly test: 同为 NV 夜间 CI 稳定性修复，体现小步修复 + 精准 skip/ 禁用 的近期 CI 维护策略。
- PR #33108 feat(dgx-spark): add inkling-small MoE support for sm_121: 同为 Blackwell / sm 系列上的 MoE runner 与内核适配，与 overrides.py 的 runner 选择逻辑处于同一适配线。