PR #24195 完整报告

sgl-project/sglang

Fix flashinfer autotune oom glm51

合并时间：2026-06-03 14:28

原文链接：http://prhub.com.cn/sgl-project/sglang/pull/24195

执行摘要

- 一句话：修复 FlashInfer 自动调优时因 lm_head 导致 OOM
- 推荐动作：此 PR 设计简洁，修复明确，值得阅读以了解如何通过上下文管理器在特定路径跳过计算，避免 OOM。

功能与动机

FlashInfer 自动调优时，_dummy_run 会触发完整的 LogitsProcessor.forward，其中的 lm_head + tensor-parallel all-gather 会分配大量显存，导致 OOM。PR body 指出这是从 #23796 搬运过来的修复，并引用了 OOM CI 日志。

实现拆解

1. 在 logits_processor.py 中新增上下文管理器 autotune_dummy_run_mode：设置全局标志 _in_autotune_dummy_run = True，并在 LogitsProcessor.forward 开头检查该标志，若为 True 则直接返回空输出（LogitsProcessorOutput(next_token_logits=None)），从而跳过所有 lm_head 计算和 TP all-gather。
2. 在 model_runner.py 的 _flashinfer_autotune 方法中启用该上下文管理器：导入 autotune_dummy_run_mode，并在 torch.inference_mode() 和 autotune() 上下文之上嵌套使用，确保 dummy run 期间自动跳过 lm_head。
3. 配套调整：新增 import from contextlib import contextmanager，在 _in_autotune_dummy_run 旁添加详细注释说明动机和 OOM 机制。

关键文件：

- python/sglang/srt/layers/logits_processor.py（模块 logits 处理器；类别 source；类型 core-logic；符号 get_in_autotune_dummy_run, autotune_dummy_run_mode）：核心变更：新增 _in_autotune_dummy_run 全局标志、get_in_autotune_dummy_run() 和 autotune_dummy_run_mode() 上下文管理器，并在 forward() 方法开头添加早期返回逻辑以跳过 lm_head 计算。
- python/sglang/srt/model_executor/model_runner.py（模块 模型运行器；类别 source；类型 data-contract）：变更点：在 _flashinfer_autotune 方法中导入并使用 autotune_dummy_run_mode() 上下文管理器，嵌套在现有 torch.inference_mode() 和 autotune() 上下文中，使得 dummy run 自动跳过 lm_head 计算。

关键符号：get_in_autotune_dummy_run, autotune_dummy_run_mode, LogitsProcessor.forward, ModelRunner._flashinfer_autotune

关键源码片段

python/sglang/srt/layers/logits_processor.py

核心变更：新增 _in_autotune_dummy_run 全局标志、get_in_autotune_dummy_run() 和 autotune_dummy_run_mode() 上下文管理器，并在 forward() 方法开头添加早期返回逻辑以跳过 lm_head 计算。

```
# logits_processor.py ( 片段 )

# 当该标志为 True 时，LogitsProcessor.forward 返回空输出并跳过
# LM head + tensor-parallel all-gather。FlashInfer autotune 只
# 需要 profile attention/MoE/GEMM 内核，LM-head all-gather 是
# 多余的计算，并且在 DP attention + 紧 mem_fraction_static 下
# 其 [batch * dp_size, vocab] 输出会 OOM。
_in_autotune_dummy_run = False


def get_in_autotune_dummy_run() -> bool:
    return _in_autotune_dummy_run


@contextmanager
def autotune_dummy_run_mode():
    """上下文管理器，在 FlashInfer autotune 期间启用 dummy run 模式。"""
    global _in_autotune_dummy_run
    _in_autotune_dummy_run = True
    try:
        yield
    finally:
        _in_autotune_dummy_run = False


class LogitsProcessor(nn.Module):
    # ...
    def forward(self, ...):
        # ...
        # 检查是否处于 autotune dummy run：若是则跳过所有 LM head 计算
        if _in_autotune_dummy_run:
            return LogitsProcessorOutput(next_token_logits=None)
        # 后续正常的 logits 处理逻辑 ...
```

python/sglang/srt/model_executor/model_runner.py

变更点：在 _flashinfer_autotune 方法中导入并使用 autotune_dummy_run_mode() 上下文管理器，嵌套在现有 torch.inference_mode() 和 autotune() 上下文中，使得 dummy run 自动跳过 lm_head 计算。

```
# model_runner.py（片段）

def _flashinfer_autotune(self):
    """Run flashinfer autotune."""
    from flashinfer.autotuner import autotune
    from sglang.srt.layers.logits_processor import autotune_dummy_run_mode

    cache_path = self._flashinfer_autotune_cache_path()
    # ... 缓存逻辑 ...

    self.forward_stream.wait_stream(torch.cuda.current_stream())
    with torch.get_device_module(self.device).stream(self.forward_stream):
        with (
            torch.inference_mode(),
            autotune(True, cache=str(autotune_cache)),
            autotune_dummy_run_mode(), # 新增：跳过 LM head
        ):
            self._dummy_run(batch_size=self.req_to_token_pool.size)
    torch.cuda.current_stream().wait_stream(self.forward_stream)
    logger.info("FlashInfer autotune completed.")
```

评论区精华

Fridge003 提出是否可以直接从 flashinfer 的 autotune 上下文中获取全局标志，避免自建 context manager。kpham-sgl 回复称 flashinfer 有一个相关的 PR（#3396）但尚未合并，目前没有替代方案，因此保留当前实现。

- 是否可使用 flashinfer 自带的全局标志替代自建 context manager? (design): 由于 flashinfer 尚未提供对应接口，保留当前自定义 context manager 实现。

风险与影响

- 风险：变更范围小（仅两个文件），且只在 autotune 路径下生效，不影响正常推理路径。风险较低。但需注意 LogitsProcessor.forward 的早期返回必须放在所有分支之前，当前实现已满足，但未来若增加新的早期分支需要留意顺序。
- 影响：仅影响 FlashInfer 自动调优的 dummy run 阶段，解决特定模型（GLM51）在 DP attention + 紧内存预算下的 OOM 问题。无其他影响。
- 风险标记：核心路径变更，缺少测试覆盖

关联脉络

- PR #23796 原 PR（本 PR 从此搬运）：本 PR 是 #23796 的搬运，用于触发 Nightly CI 验证修复效果。