PR #31992 完整报告

sgl-project/sglang

fix(hisparse): correct DSA KV memory budget

合并时间：2026-07-27 11:19

原文链接：http://prhub.com.cn/sgl-project/sglang/pull/31992

执行摘要

- 一句话：修复 HiSparse DSA KV 内存预算与实际分配不匹配
- 推荐动作：建议关注 _compute_cell_size 与 DSATokenToKVPool 以及 calculate_mla_kv_cache_dim 的契约关系。本 PR 展示了如何利用已有分配函数来保障预算准确性，是内存生命周期管理的最佳实践。值得精读以避免类似内存偏差问题。

功能与动机

PR body 指出 HiSparseDSATokenToKVPool 分配 indexer buffer 时使用 size * host_to_device_ratio，但 _compute_cell_size 只预算了 size。此外 FP8 MLA 实际分配包含 FP32 scales 和 BF16 RoPE，预算仅使用 kv_lora_rank + qk_rope_head_dim 导致低估。关联 Issue #28752 原已修复 indexer ratio 部分但不完整，本 PR 完整修复。

实现拆解

1. 在 _compute_cell_size 中导入 calculate_mla_kv_cache_dim，替换硬编码的 kv_lora_rank + qk_rope_head_dim，使 FP8 MLA 布局计算与 DSATokenToKVPool 一致。
2. 在 DSA indexer 预算项中，当 enable_hisparse 时解析 host_to_device_ratio 并缩放 indexer 大小。
3. 新增测试文件，模拟 GLM DSA 配置，通过 _compute_cell_size 验证预期字节数：无 HiSparse 时 BF16 2568 B/token、FP8 1576 B/token；HiSparse 启用 ratio=2 时 1840 B/token、ratio=4 时 2368 B/token。这些值对应实际分配布局。

关键文件：

- python/sglang/srt/model_executor/pool_configurator.py（模块 内存预算；类别 source；类型 data-contract；符号 _compute_cell_size）：核心修改：确保 MLA KV 缓存维度与分配器一致，并正确缩放 HiSparse indexer 预算。
- test/registered/unit/model_executor/test_hisparse_pool_configurator.py（模块 HiSparse 测试；类别 test；类型 test-coverage；符号 TestHiSparsePoolConfigurator, test_mla_layout_without_hisparse, test_hisparse_indexer_scales_with_ratio）：新增 CPU 回归测试，覆盖 BF16/FP8 MLA 布局及 HiSparse indexer 缩放，确保预算计算正确。

关键符号：_compute_cell_size, calculate_mla_kv_cache_dim, parse_hisparse_config

关键源码片段

python/sglang/srt/model_executor/pool_configurator.py

核心修改：确保 MLA KV 缓存维度与分配器一致，并正确缩放 HiSparse indexer 预算。

```
# pool_configurator.py: _compute_cell_size 中 if kvc.use_mla_backend 分支（关键修改部分）
if kvc.use_mla_backend:
    from sglang.srt.mem_cache.kv_cache_configurator import (
        calculate_mla_kv_cache_dim, # 复用分配器实际使用的维度计算
    )
    # MLA 主缓存：使用 calculate_mla_kv_cache_dim 确保与 DSATokenToKVPool 一致
    cell_size = (
        calculate_mla_kv_cache_dim(
            model_config=model_config,
            kv_cache_dtype=kv_cache_dtype,
            server_args=kvc.server_args,
        )
        * effective_num_layers
        * kv_size
    )
    # ... (kv_scale_buffer 处理略 )

    # DSA indexer 缓存预算
    if is_deepseek_dsa(model_config.hf_config):
        index_head_dim = get_dsa_index_head_dim(model_config.hf_config)
        indexer_size_per_token = (
            index_head_dim
            + index_head_dim // DSATokenToKVPool.quant_block_size * 4
        )
        element_size = torch._utils._element_size(
            DSATokenToKVPool.index_k_with_scale_buffer_dtype
        )
        # 默认 ratio=1（等同无 HiSparse）
        indexer_ratio = 1
        if kvc.server_args.enable_hisparse:
            from sglang.srt.mem_cache.sparsity import parse_hisparse_config
            indexer_ratio = parse_hisparse_config(
                kvc.server_args
            ).host_to_device_ratio # 解析用户配置
        cell_size += int(
            indexer_size_per_token
            * effective_num_layers
            * element_size
            * indexer_ratio # 此处缩放
        )
```

test/registered/unit/model_executor/test_hisparse_pool_configurator.py

新增 CPU 回归测试，覆盖 BF16/FP8 MLA 布局及 HiSparse indexer 缩放，确保预算计算正确。

```
# test_hisparse_pool_configurator.py: 完整的测试类
```

```
import unittest
from types import SimpleNamespace
from unittest.mock import MagicMock

import torch

from sglang.srt.model_executor.pool_configurator import DefaultPoolConfigurator
from sglang.srt.runtime_context import get_parallel
from sglang.test.ci.ci_register import register_cpu_ci
from sglang.test.test_utils import CustomTestCase

register_cpu_ci(est_time=1, suite="base-a-test-cpu")


class TestHiSparsePoolConfigurator(CustomTestCase):
    def _compute_cell_size(
        self,
        kv_cache_dtype: torch.dtype,
        *,
        enable_hisparse: bool,
        host_to_device_ratio: int = 1,
    ) -> int:
        # 构造 GLM DSA 模型的模拟配置
        hf_config = SimpleNamespace(
            architectures=["GlmMoeDsaForCausalLM"],
            index_topk=2048,
            index_head_dim=128,
        )
        hf_config.get_text_config = lambda: hf_config
        kvc = MagicMock(
            use_mla_backend=True,
            kv_cache_dtype=kv_cache_dtype,
            model_config=SimpleNamespace(
                kv_lora_rank=512,
                qk_rope_head_dim=64,
                hf_config=hf_config,
            ),
            server_args=SimpleNamespace(
                enable_hisparse=enable_hisparse,
                hisparse_config=(f'{{"host_to_device_ratio": {host_to_device_ratio}}}'),
                dsa_prefill_backend="flashmla_sparse",
                dsa_decode_backend="flashmla_sparse",
            ),
        )
        with get_parallel().override(attn_tp_size=1):
            configurator = object.__new__(DefaultPoolConfigurator)
            return configurator._compute_cell_size(kvc, num_layers=2)

    def test_mla_layout_without_hisparse(self):
```

```
        # 验证无 HiSparse 时的 ML 缓存大小（BF16 和 FP8 两种 dtype）
        for kv_cache_dtype, expected_cell_size in (
            (torch.bfloat16, 2568),
            (torch.float8_e4m3fn, 1576),
        ):
            with self.subTest(kv_cache_dtype=kv_cache_dtype):
                cell_size = self._compute_cell_size(
                    kv_cache_dtype,
                    enable_hisparse=False,
                )
                self.assertEqual(cell_size, expected_cell_size)

    def test_hisparse_indexer_scales_with_ratio(self):
        # 验证启用 HiSparse 时 indexer 预算随 ratio 正确缩放（FP8 场景）
        for host_to_device_ratio, expected_cell_size in (
            (2, 1840),
            (4, 2368),
        ):
            with self.subTest(host_to_device_ratio=host_to_device_ratio):
                cell_size = self._compute_cell_size(
                    torch.float8_e4m3fn,
                    enable_hisparse=True,
                    host_to_device_ratio=host_to_device_ratio,
                )
                self.assertEqual(cell_size, expected_cell_size)


if __name__ == "__main__":
    unittest.main()
```

评论区精华

Reviewer @huangtingwei9988 建议将测试拆分为独立 case（MLA 布局测试与 indexer ratio 测试），便于发现回归。作者采纳并在 commit 67ddf3a 中分离了两个测试方法并添加 subTest。

- 测试用例拆分建议 (testing): 作者在 67ddf3a 中新增两个独立测试方法并添加 subTest，已满足要求。

风险与影响

- 风险：核心风险在于修改了 _compute_cell_size 方法，该方法决定 KV 缓存占用预算，直接影响内存分配合法性。若某布局下计算偏差仍存在，则可能导致 OOM 或资源浪费。但通过复用已有函数 calculate_mla_kv_cache_dim（负责实际分配维度）和新增端到端测试，风险被控制。需要确保回归覆盖所有相关模型（GLM-5.2 FP8/BF16、不同 ratio）及未来新增模型。
- 影响：影响用户群体：使用 DSA 模型（如 GLM-5.2）且开启 HiSparse 功能的用户。影响程度：高，因为此 bug 可能导致启动 OOM 或内存分配不足（如 PR body 所述 6,240

bytes/token 未被预算，在百万 token 时可达 5.8 GiB）。修复后系统可正确计算最大 token 数，确保稳定性，同时提升内存利用率。对团队维护者：需理解新的预算计算逻辑，并在修改池分配方式时保持同步。

- 风险标记：核心路径变更，测试覆盖有限，依赖外部函数一致性

关联脉络

- PR #28752 fix(hisparse): account for host_to_device_ratio in DSA indexer memory…: 原 PR 尝试修复同一问题但仅解决 indexer ratio 部分，本 PR 在此基础上补充了 MLA 布局修复和完整测试。