PR #25751 完整报告

sgl-project/sglang

[Kernel] Add SM90 Q8KV8 FP8 Sparse MLA Prefill JIT Kernel with Tests and Benchmark

合并时间：2026-06-30 09:00

原文链接：http://prhub.com.cn/sgl-project/sglang/pull/25751

执行摘要

- 一句话：新增 SM90 Q8KV8 FP8 稀疏 MLA 预填充 JIT 内核，性能提升 1.15-1.31x
- 推荐动作：此 PR 是高质量的独立内核提交，代码结构、测试覆盖和性能数据完善。推荐所有关注 Hopper FP8 优化和注意力机制的工程师精读，特别是 Python 封装和编译标志裁剪部分值得借鉴。

功能与动机

稀疏 MLA 预填充是长上下文服务中的性能关键路径，FlashMLA 已将 KV 量化为 FP8 但 Q 仍为 BF16。将 Q 升级为 FP8 可消除精度差距，释放完整的 FP8 张量核心吞吐并减少 Q 侧内存流量。此 PR 作为独立内核块，便于审查和测试，后续将在 PR2 中集成到运行时。

实现拆解

实现分为四步：

1. CUDA C++ 内核开发：在 python/sglang/jit_kernel/csrc/sparse_mla_q8kv8_prefill_sm90/ 下实现基于 CUTLASS/CuTe 的 SM90 FP8 GMMA 内核。kernel.cuh 包含核心注意力算法（QK、PV、online softmax），采用 producer-consumer pipeline 隐藏内存延迟。entry.cuh 提供两个调度入口（dispatch 和 dispatch_full，后者支持 attn_sink 和 topk_length）。头文件包含 helper、config 和 FP8 布局转换等工具。
2. Python JIT 封装：在 python/sglang/jit_kernel/sparse_mla_q8kv8_prefill_sm90.py 中，利用 load_jit 编译 CUDA 源码并暴露 sparse_mla_q8kv8_prefill_fwd 函数。该函数通过 register_custom_op 注册为 torch.library 自定义算子，便于 torch.compile 追踪。内部缓存已解析的 dispatch 入口点，避免重复查询。编译标志经逐项裁剪，仅保留必要的 --use_fast_math 等，减少编译开销。
3. 编译标志优化：从 FlashMLA 继承的多个标志被移除，因为它们在 tvm_ffi JIT 构建下无实际效果。保留 -O3、-DNDEBUG、CUTE_USE_PACKED_TUPLE=1、CUTLASS_ENABLE_TENSOR_CORE_MMA=1 和 --use_fast_math。--use_fast_math 对性能有 0.1%-4.3% 影响，保留。
4. 测试与基准：在 test/registered/jit/ 下添加两个文件。test_sparse_mla_q8kv8_prefill_sm90.py 包含参考对比测试（与 FP32 参考相比，相对误差约 0.7%）、边界用例（s_q=1, topk=256 等）和精度度量。bench_sparse_mla_q8kv8_prefill_sm90.py 使用 Triton 基准框架，在 CI 和长上下文形状

上比较 Q8KV8 与 Q16 FlashMLA。

关键文件：

- python/sglang/jit_kernel/sparse_mla_q8kv8_prefill_sm90.py（模块 JIT 内核；类别 source；类型 core-logic；符号 _q8kv8_cuda_flags, _jit_sparse_mla_q8kv8_prefill_module, _get_entries, _check_out_buffer）：内核的 Python 封装，提供公共 API sparse_mla_q8kv8_prefill_fwd，通过 JIT 编译 CUDA 代码并注册为自定义算子，是核心接口。
- test/registered/jit/test_sparse_mla_q8kv8_prefill_sm90.py（模块 测试；类别 test；类型 test-coverage；符号 _sm90_available, _make_fp8_tensor, _make_case, _torch_sparse_attention_ref）：包含精度测试、边界测试和参考对比测试，验证内核与 FP32 参考的一致性及量化误差。
- test/registered/jit/benchmark/bench_sparse_mla_q8kv8_prefill_sm90.py（模块 基准；类别 test；类型 test-coverage；符号 _sm90_available, _make_indices, _make_q16_inputs, _make_q8_inputs）：使用 Triton Benchmark 比较 Q8KV8 JIT 内核与 Q16 FlashMLA 的延迟，提供 CI 和长上下文用例的性能证据。
- python/sglang/jit_kernel/csrc/sparse_mla_q8kv8_prefill_sm90/kernel.cuh（模块 CUDA 内核；类别 source；类型 core-logic；符号 NamedBarriers）：核心 CUDA 内核实现（968 行），包含注意力计算的 producer-consumer pipeline、FP8 GMMA、online softmax 等关键算法。
- python/sglang/jit_kernel/csrc/sparse_mla_q8kv8_prefill_sm90/entry.cuh（模块 CUDA 入口；类别 source；类型 core-logic）：内核调度入口，封装 dispatch 和 dispatch_full 两个接口，处理参数解析和内核启动。

关键符号：sparse_mla_q8kv8_prefill_fwd, _sparse_mla_q8kv8_prefill_op, _sparse_mla_q8kv8_prefill_full_op, _get_entries, _check_out_buffer, _torch_sparse_attention_ref, test_sparse_mla_q8kv8_prefill_matches_reference, test_sparse_mla_q8kv8_prefill_corner_cases, test_sparse_mla_q8kv8_prefill_precision, bench_sparse_mla_q8kv8_prefill_sm90

关键源码片段

python/sglang/jit_kernel/sparse_mla_q8kv8_prefill_sm90.py

内核的 Python 封装，提供公共 API sparse_mla_q8kv8_prefill_fwd，通过 JIT 编译 CUDA 代码并注册为自定义算子，是核心接口。

```
# 编译标志函数：经过逐项裁剪，仅保留在 tvm_ffi JIT 构建下有实际效果的标志

def _q8kv8_cuda_flags() -> list[str]:
    # 最小标志集，在 SM90/H200 (CUDA 12.9) 上逐项确认
    # 保留 --use_fast_math，因为它将 softmax exp2f 映射到 MUFU ex2.approx.f32
    # 移除它会导致 0.1%-4.3% 的性能下降，且精度变化可忽略
    return [
        "-O3",
        "-DNDEBUG",
```

```
        "-DCUTE_USE_PACKED_TUPLE=1",
        "-DCUTLASS_ENABLE_TENSOR_CORE_MMA=1",
        "--use_fast_math",
    ]


# 公共前向函数：支持 with_sink 和 without_sink 两种 dispatch
@debug_kernel_api # 使调用参与内核 API 日志
@register_custom_op(op_name="sparse_mla_q8kv8_prefill", mutates_args=["out", "max_logits",
"lse"])
def sparse_mla_q8kv8_prefill_fwd(
    q: torch.Tensor,
    kv: torch.Tensor,
    indices: torch.Tensor,
    sm_scale: float,
    q_scale: torch.Tensor,
    kv_scale: torch.Tensor,
    d_v: int,
    attn_sink: Optional[torch.Tensor] = None,
    topk_length: Optional[torch.Tensor] = None,
) -> tuple[torch.Tensor, torch.Tensor, torch.Tensor]:
    if d_v != 512:
        raise ValueError(f"kernel 硬编码 D_V=512，但传入了 d_v={d_v}")
    # 解析 dispatch 入口
    dispatch, dispatch_full = _get_entries()
    # 准备输出缓冲区（caller-owned，调用者应保证不别名）
    out = torch.empty(
        (q.size(0), q.size(1), d_v),
        dtype=torch.bfloat16,
        device=q.device,
    )
    max_logits = torch.empty(
        (q.size(0), q.size(1)),
        dtype=torch.float32,
        device=q.device,
    )
    lse = torch.empty_like(max_logits)
    # 选择 dispatch 路径
    if attn_sink is not None:
        dispatch_full(...) # 完整路径
    else:
        dispatch(...) # 标准路径
    return out, max_logits, lse
```

评论区精华

Review 中 BBuf 提出四个关键问题：

- 输出缓冲区语义：原始代码在模块级缓存输出张量，可能导致不同调用共享同一存储，产生别名问题。JackChuang 随后改为 caller-owned 输出缓冲区，消除了风险。
- d_v 断言：内核硬编码 D_V=512，BBuf 建议添加断言。JackChuang 在 sparse_mla_q8kv8_prefill_fwd 中添加了 ValueError。
- 编译标志过多：BBuf 质疑是否需要诸多特殊编译标志。JackChuang 解释这些是从 FlashMLA 继承的无用标志，并删除了它们。
- --use_fast_math 影响：BBuf 询问如果移除该标志的性能损失。JackChuang 评估为 0.1%-4.3% 减速，建议保留。所有讨论均已解决并体现在最终代码中。
  - 输出缓冲区模块级缓存的别名问题 (correctness): JackChuang 接受了建议，将设计改为 caller-owned 输出缓冲区，移除了模块级缓存。
  - 添加 d_v=512 断言 (correctness): JackChuang 在 sparse_mla_q8kv8_prefill_fwd 中添加了 ValueError 检查。
  - 编译标志过多需要精简 (performance): 移除了除 --use_fast_math 外的所有多余标志，仅保留必要的最小集。
  - 评估移除 --use_fast_math 的性能影响 (performance): 保留 --use_fast_math，因为性能影响显著且精度损失可忽略。

风险与影响

- 风险：
  1. 精度损失：预期约 2.8% 的准确率下降，源于 BF16 到 FP8 的量化，在测试中得到确认，对精度敏感的应用需评估。
  2. 硬件限制：仅支持 SM90 (Hopper/H200)，在其他 GPU 上不可用，需在运行时检查 is_sm90_supported。
  3. 未集成运行时：当前仅作为独立内核，未接入模型推理路径，无法直接使用，依赖后续 PR2。
  4. 依赖 CUTLASS：JIT 编译依赖 extra_dependencies=["cutlass"]，若环境缺少 CUTLASS 头文件会失败。
  5. 输出缓冲区生命周期：虽然已改为 caller-owned，但若调用者错误地复用输出张量可能导致数据竞争。 - 影响：对用户：无法直接受益，需等待 PR2 集成到 DeepSeek-V3.2 等模型中。对系统：增加约 2500 行新代码，编译时间因 JIT 而略有增加。对团队：需维护一个新的硬件特定内核，但代码结构清晰且附有测试。 - 风险标记：仅 SM90，精度损失约 2.8%，未集成到运行时，依赖 CUTLASS，输出缓冲区需正确管理

关联脉络

- PR #25746 [Feature] SM90 Q8KV8 FP8 Sparse MLA Prefill Kernel and SGLang NSA Runtime Integration: 此 PR 是该 roadmap 的 PR1，关联 Issue 描述了完整的计划。