PR #29664 完整报告

sgl-project/sglang

[Diffusion] Reuse shared AlignedVector and tidy jit_kernel/diffusion

合并时间：2026-06-30 11:38

原文链接：http://prhub.com.cn/sgl-project/sglang/pull/29664

执行摘要

- 一句话：统一扩散 CUDA 内核向量化实现并清理冗余代码
- 推荐动作：值得深度阅读，特别是对 SGLang 扩散内核或 CUDA JIT 内核感兴趣的开发者。PR 展示了： 1) 如何通过提取公共组件消除重复代码； 2) 用 SASS 不变量验证重构无回退； 3) 清理死代码和参数简化。这些实践可在同类重构中复用。

功能与动机

KDA-Pilot 扩散原生 CUDA 快速路径都需要 128 位向量化加载 / 存储。第一个落地的 norm_scale_shift（PR#27392）已使用 sgl_kernel/vec.cuh 中的共享 device::AlignedVector 组件，但两个更新的内核（PR#29281, PR#29361）仍然手写自己的本地 union。本 PR 使所有扩散 CUDA 内核复用共享 AlignedVector，并合并了在审查其余 jit_kernel/diffusion/ 时发现的一些小的、已验证的清理。无行为变更。

实现拆解

1. 提取公共工具函数：将 to_cute_arg 和 to_fake_cute_args 从 scale_residual_norm_scale_shift.py、norm_tanh_mul_add_norm_scale.py 中删除，统一添加到 utils.py，后续所有 CuTe-DSL 内核从同一处导入。
2. 替换 Native CUDA 向量化结构：在 residual_gate_add.cuh、causal_conv3d_cat_pad.cuh、timestep_embedding.cuh 中，将手写的 union Vec16<T>、union Pack、float4 分别替换为 device::AlignedVector<T, kVec>，并通过 load()/store()/operator[] 保持接口一致。
3. 修复 copy_if 类型检查 bug：norm_tanh_mul_add_norm_scale.py 中 @cute.jit 装饰的 copy_if 函数的条件 isinstance(src, Tensor) and isinstance(src, Tensor) 第二个应为 dst，本 PR 修正为 isinstance(dst, cute.Tensor)。
4. 删除死代码：sana_wm_gdn.py 中 _precompute_inv_rms 函数已被后续的 fused_qk_inv_rms Triton 融合内核取代，无调用者，直接删除；同时修复 docstring 中错误的模块名引用。
5. 简化参数签名：scale_shift.py 的 _fused_scale_shift_4d_kernel 删除未使用的参数 rows 及其调用处；validate_weight_bias 删除未使用的 B/S 参数，仅保留必需的 D。
6. 添加命名空间：timestep_embedding.cuh 添加 #pragma once 和命名空间 sglang_timestep_embedding，对应的 Python wrapper 中加上了命名空间前缀，使其与兄

弟内核一致。

关键文件：

- python/sglang/jit_kernel/diffusion/cutedsl/utils.py（模块 公共工具；类别 source；类型 core-logic；符号 to_cute_arg, to_fake_cute_args）：新增公共工具函数 to_cute_arg 和 to_fake_cute_args，后续所有 CuTe-DSL 内核从 此处导入，消除 ~36 行重复代码。
- python/sglang/jit_kernel/diffusion/cutedsl/norm_tanh_mul_add_norm_scale.py（模块 扩散内核；类别 source；类型 bugfix；符号 to_cute_arg, to_fake_cute_args, copy_if）：移除本地函数定义并改为从 utils.py 导入；修复 copy_if 中第二个 isinstance 错写为 src 的 bug（应为 dst）。
- python/sglang/jit_kernel/diffusion/cutedsl/scale_residual_norm_scale_shift.py（模块 扩散内核；类别 source；类型 core-logic；符号 to_cute_arg, to_fake_cute_args, validate_weight_bias）：移除本地复制的 to_cute_arg/to_fake_cute_args，改为从 utils.py 导入；简化 validate_weight_bias 签名，删除未使用的 B/S 参数。
- python/sglang/jit_kernel/diffusion/triton/sana_wm_gdn.py（模块 扩散内核；类别 source；类型 core-logic；符号 _precompute_inv_rms, fused_qk_inv_rms, fused_bigdn_func）：删除已无调用者的 _precompute_inv_rms 函数，并修复 docstring 中的模块名引用错误。
- python/sglang/jit_kernel/timestep_embedding.py（模块 扩散内核；类别 source；类型 core-logic）：为对应的 CUDA wrapper 添加命名空间前缀，与兄弟内核保持一致。
- python/sglang/jit_kernel/diffusion/triton/scale_shift.py（模块 扩散内核；类别 source；类型 core-logic）：删除 _fused_scale_shift_4d_kernel 中未使用的参数 rows 及其调用处。

关键符号：to_cute_arg, to_fake_cute_args, validate_weight_bias, copy_if, _precompute_inv_rms, fused_norm_scale_shift, fused_scale_residual_norm_scale_shift

关键源码片段

python/sglang/jit_kernel/diffusion/cutedsl/utils.py

新增公共工具函数 to_cute_arg 和 to_fake_cute_args，后续所有 CuTe-DSL 内核从 此处导入，消除 ~36 行重复代码。

```
# 文件：python/sglang/jit_kernel/diffusion/cutedsl/utils.py
from typing import Optional

import cutlass
import cutlass.cute as cute
import torch

WARP_SIZE = 32

# 将 PyTorch dtype 映射到 CuTeDSL 类型
TORCH_TO_CUTE_DTYPE = {
    torch.float16: cutlass.Float16,
    torch.bfloat16: cutlass.BFloat16,
    torch.float32: cutlass.Float32,
```

```
}

def to_cute_arg(
    t,
    *,
    assume_aligned: Optional[int] = 32,
    use_32bit_stride: bool = False,
    enable_tvm_ffi: bool = True,
):
    # 将 Python 值转换为 CuTeDSL 值
    if isinstance(t, torch.Tensor):
        return cute.runtime.from_dlpack(t, assumed_align=assume_aligned,
                                        use_32bit_stride=use_32bit_stride,
                                        enable_tvm_ffi=enable_tvm_ffi)
    if isinstance(t, int):
        return cutlass.Int32(t)
    if isinstance(t, float):
        return cutlass.Float32(t)
    return t

def to_fake_cute_args(t: torch.Tensor):
    # 将非最后维度替换为符号整数以最大化内核复用
    # 例：(1,2,1536):(3027,1536,1) -> (?,?,1536):(?,?,1)
    if isinstance(t, torch.Tensor):
        D = t.shape[-1]
        dtype = TORCH_TO_CUTE_DTYPE[t.dtype]
        # 前 n-1 维用符号，最后一维保留真实值
        shape = (*(cute.sym_int() for _ in range(t.ndim - 1)), D)
        stride = (*(cute.sym_int(divisibility=D) for _ in range(t.ndim - 1)), 1)
        fake_t = cute.runtime.make_fake_tensor(
            dtype, shape, stride, memspace=cute.AddressSpace.gmem, assumed_align=32
        )
        return fake_t
    return to_cute_arg(t)
```

python/sglang/jit_kernel/diffusion/cutedsl/norm_tanh_mul_add_norm_scale.py

移除本地函数定义并改为从 utils.py 导入；修复 copy_if 中第二个 isinstance 错写为 src 的 bug（应为 dst）。

```
# 文件：python/sglang/jit_kernel/diffusion/cutedsl/norm_tanh_mul_add_norm_scale.py
from typing import Optional, Tuple

import cuda.bindings.driver as cuda
import cutlass
import cutlass.cute as cute
import torch

# 从公共工具导入，而非本地定义
```

```
from sglang.jit_kernel.diffusion.cutedsl.utils import (
    WARP_SIZE,
    to_cute_arg,
    to_fake_cute_args,
)

# ... 其余导入和类定义 ...

class NormTanhMulAddNormScale:
    # ...
    @cute.jit
    def copy_if(src, dst):
        # 修复：第二个 isinstance 的参数从 src 改为 dst
        if cutlass.const_expr(
            isinstance(src, cute.Tensor) and isinstance(dst, cute.Tensor)
        ):
            cute.autovec_copy(src, dst)
```

python/sglang/jit_kernel/diffusion/cutedsl/scale_residual_norm_scale_shift.py

移除本地复制的 to_cute_arg/to_fake_cute_args，改为从 utils.py 导入；简化 validate_weight_bias 签名，删除未使用的 B/S 参数。

```
# 文件：python/sglang/jit_kernel/diffusion/cutedsl/scale_residual_norm_scale_shift.py
# 导入改为从公共工具获取 to_fake_cute_args，本地不再定义
from sglang.jit_kernel.diffusion.cutedsl.utils import (
    WARP_SIZE,
    to_fake_cute_args,
)

def validate_weight_bias(t: Optional[torch.Tensor], D: int):
    # 验证 weight 或 bias 张量：dtype, shape 和连续性
    if t is None:
        return
    if t.dtype not in (torch.float16, torch.bfloat16, torch.float32):
        raise ValueError(f'Validate failed: unsupported dtype: {t.dtype}')
    if t.shape != (D,):
        raise ValueError(f'Validate failed: unsupported tensor shape: {t.shape}.')
    if t.stride()[-1] != 1:
        raise ValueError(f'Validate failed: not contiguous on dim D.')
```

评论区精华

本 PR 无审查评论。作者在 PR body 中提供了完整的准确性测试结果（B200 上所有扩散 pytest 通过）和性能基准表，并通过 SASS 指令对比（ncu）证明生成的二进制码与旧版几乎完全一致（指令数相同，仅有编译器调度差异），确认无性能回归。

- 暂无高价值评论线程

风险与影响

- 风险：低风险。变更设计为行为保持（no behavior change），且通过多维度验证： 1) 所有相关 pytest 通过； 2) 基准测试与旧版持平； 3) SASS 指令级对比未引入新指令或局部内存操作。潜在风险点包括：替换 AlignedVector 后对齐假设是否一致（已验证无差异）；删除死函数可能影响未来复用（已确认无调用者）；参数简化可能漏掉隐式使用（通过测试覆盖）。整体风险可控。
  - 影响：对用户无行为改变，推理结果一致；性能无回归，甚至因代码统一可能为未来优化奠定基础。对维护者，代码减少约 160 行，重复率降低，可维护性提升；新增公共工具函数降低了后续添加类似内核的门槛。对团队，展示了如何通过 SASS 不变量和基准测试系统化地保障重构质量。
  - 风险标记：核心路径重构，无行为变化，SASS 对比验证

关联脉络

- PR #27392 norm_scale_shift kernel with AlignedVector: 本 PR 复用了该 PR 引入的共享 AlignedVector 组件。
- PR #29281 Diffusion kernel with hand-rolled union: 本 PR 替换了该 PR 中手写的 union Vec16/T。
- PR #29361 Diffusion kernel with hand-rolled union: 本 PR 替换了该 PR 中手写的 union Pack。