PR #39912 完整报告

vllm-project/vllm

[Kernel] Batch invariant NVFP4 linear using cutlass

合并时间：2026-05-23 21:41

原文链接：http://prhub.com.cn/vllm-project/vllm/pull/39912

执行摘要

- 一句话：为 NVFP4 CUTLASS 添加 batch invariant 模式
- 推荐动作：此 PR 展示了在已有 GPU kernel 上安全添加确定性模式的优秀实践：通过条件编译、静态断言和显式的 TileScheduler 指定来防止未来意外 break。值得精读以了解 vLLM 的 batch invariance 设计和 CUDA kernel 重构方法。

功能与动机

当前 NVFP4 CUTLASS 线性内核虽已有 batch invariant 特性，但属于意外实现（by accident），未来可能被调度器变更破坏。此 PR 通过显式 flag 和静态断言保护该属性。同时，之前的 VLLM_BATCH_INVARIANT 强制使用 emulation 后端，此 PR 优先使用 CUTLASS 后端以便获得更好性能。PR 标题提到 'alternative to #39727'。

实现拆解

1. C++ 内核端：在 nvfp4_scaled_mm_sm120_kernels.cu 和 nvfp4_scaled_mm_kernels.cu 中，为 sm100_fp4_config_default 和 sm120_fp4_config_default 结构体显式指定 TileScheduler 为 cutlass::gemm::PersistentScheduler（batch-invariant 所需的固定调度），并在 dispatch 函数中通过 vllm::vllm_is_batch_invariant() 检查环境变量，若启用则使用默认配置（固定调度），且添加静态断言确保 TileScheduler 为 PersistentScheduler。
2. Python 选择逻辑：在 vllm/model_executor/kernels/linear/__init__.py 的 init_nvfp4_linear_kernel() 中，修改 VLLM_BATCH_INVARIANT 分支：优先检查 CutlassNvFp4LinearKernel.is_supported()，若支持则强制使用 CUTLASS 后端；否则回退到 Emulation 后端。
3. 测试：新增 tests/v1/determinism/test_nvfp4_batch_invariant_scaled_mm.py，包含测试 test_nvfp4_gemm_batch_invariance，验证每行输出在 M=1 和 M=full 下的一致性。
4. CI 配置：在 .buildkite/test_areas/misc.yaml 的 Batch Invariance (B200) 部分添加新测试的运行命令。

关键文件：

- vllm/model_executor/kernels/linear/__init__.py（模块 内核选择器；类别 source；类型 core-logic；符号 init_nvfp4_linear_kernel, CutlassNvFp4LinearKernel, EmulationNvFp4LinearKernel）：核心 Python 选择逻辑，控制

VLLM_BATCH_INVARIANT 下使用 CUTLASS 还是 emulation 后端

- csrc/libtorch_stable/quantization/fp4/nvfp4_scaled_mm_sm120_kernels.cu（模块 SM120 内核；类别 other；类型 core-logic；符号 sm120_fp4_config_default, Fp4GemmSm120, runGemm, cutlass_fp4_f16_gemm_dispatch）：SM120 架构 kernel 配置和 dispatch 逻辑，添加 batch-invariant 分支和静态断言
- csrc/libtorch_stable/quantization/fp4/nvfp4_scaled_mm_kernels.cu（模块 SM100 内核；类别 other；类型 core-logic；符号 sm100_fp4_config_default, sm100_fp4_config_M256, sm100_fp4_config_M16, Fp4GemmSm100）：SM100 架构 kernel 配置和 dispatch 逻辑，与 SM120 变化对称
- tests/v1/determinism/test_nvfp4_batch_invariant_scaled_mm.py（模块 确定性测试；类别 test；类型 test-coverage；符号 test_nvfp4_gemm_batch_invariance）：新增单元测试，验证 NVFP4 CUTLASS GEMM 的 batch invariance 属性
- .buildkite/test_areas/misc.yaml（模块 CI 配置；类别 config；类型 configuration）：CI 配置新增 batch-invariant 测试步骤

关键符号：init_nvfp4_linear_kernel, test_nvfp4_gemm_batch_invariance, cutlass_fp4_f16_gemm_dispatch, cutlass_fp4_bf16_gemm_dispatch

关键源码片段

vllm/model_executor/kernels/linear/__init__.py

核心 Python 选择逻辑，控制 VLLM_BATCH_INVARIANT 下使用 CUTLASS 还是 emulation 后端

```
def init_nvfp4_linear_kernel() -> NvFp4LinearKernel:
    """Select and instantiate the best NVFP4 linear kernel for the current platform."""
    config = NvFp4LinearLayerConfig()

    # VLLM_BATCH_INVARIANT forces deterministic execution. Prefer the
    # batch-invariant CUTLASS implementation when available, otherwise fall
    # back to emulation.
    force_kernel: type[NvFp4LinearKernel] | None = None
    linear_backend = _get_linear_backend()
    if envs.VLLM_BATCH_INVARIANT:
        bi_supported, reason = CutlassNvFp4LinearKernel.is_supported()
        if bi_supported:
            # CUTLASS 后端可用，强制使用并获得性能优势
            if linear_backend not in ("auto", "cutlass"):
                logger.warning_once(
                    "VLLM_BATCH_INVARIANT overrides --linear-backend=%s; "
                    "using the CUTLASS backend for deterministic execution.",
                    linear_backend,
                )
            else:
                logger.info_once(
                    "VLLM_BATCH_INVARIANT forces NVFP4 linear to use the "
                    "CUTLASS backend for deterministic execution."
```

```
                )
            force_kernel = CutlassNvFp4LinearKernel
        else:
            # CUTLASS 不支持，回退到 emulation
            if linear_backend not in ("auto", "emulation"):
                logger.warning_once(
                    "VLLM_BATCH_INVARIANT overrides --linear-backend=%s; "
                    "using the emulation backend for deterministic execution.",
                    linear_backend,
                )
            logger.info_once(
                "VLLM_BATCH_INVARIANT is set but the batch-invariant NVFP4 "
                "kernel is not supported on this platform; falling back to "
                "emulation for deterministic execution. Reason: %s",
                reason,
            )
            force_kernel = EmulationNvFp4LinearKernel
    elif linear_backend == "auto":
        # Deprecated env-var overrides
        if envs.VLLM_USE_FBGEMM:
            force_kernel = FbgemmNvFp4LinearKernel
        elif envs.VLLM_USE_NVFP4_CT_EMULATIONS:
            force_kernel = EmulationNvFp4LinearKernel
        elif envs.VLLM_NVFP4_GEMM_BACKEND is not None:
            backend_name = envs.VLLM_NVFP4_GEMM_BACKEND
            force_kernel = _NVFP4_BACKEND_TO_KERNEL.get(backend_name)
            if force_kernel is None:
                raise ValueError(
                    f"Unknown VLLM_NVFP4_GEMM_BACKEND={backend_name!r}. "
                    f"Valid choices: {list(_NVFP4_BACKEND_TO_KERNEL.keys())}"
                )

    if force_kernel is not None:
        is_supported, reason = force_kernel.is_supported()
        if not is_supported:
            raise ValueError(
                f"Forced NVFP4 kernel {force_kernel.__name__} is not supported: {reason}"
            )
        logger.info_once("Using %s for NVFP4 GEMM", force_kernel.__name__)
        return force_kernel(config)

    # Auto-select from registry
    platform = current_platform._enum
    possible = list(_POSSIBLE_NVFP4_KERNELS.get(platform, []))
    ...
```

tests/v1/determinism/test_nvfp4_batch_invariant_scaled_mm.py

新增单元测试，验证 NVFP4 CUTLASS GEMM 的 batch invariance 属性

```
@pytest.mark.parametrize("dtype", DTYPES)
@pytest.mark.parametrize("shape", CONSISTENCY_SHAPES)
@torch.inference_mode()
def test_nvfp4_gemm_batch_invariance(
    dtype: torch.dtype,
    shape: tuple[int, int, int],
) -> None:
    """Batch invariance: each row of a full-M GEMM matches its M=1 counterpart.

    For row i, compares cutlass_scaled_fp4_mm run once over all M
    rows against a separate call with A sliced to a_dtype[i : i+1].
    Catches kernels whose reduction or scheduling depends on M or adjacent
    rows.
    """
    seed = int(os.getenv("VLLM_TEST_SEED", "12345"))
    set_random_seed(seed)
    m, n, packed_k = shape
    k = packed_k * 2 # real K (FP4 elements)

    a_dtype = torch.randn((m, k), dtype=dtype, device="cuda")
    b_dtype = torch.randn((n, k), dtype=dtype, device="cuda")

    a_global_scale = get_nvfp4_global_scale(a_dtype)
    b_global_scale = get_nvfp4_global_scale(b_dtype)
    alpha = 1.0 / (a_global_scale * b_global_scale)

    b_fp4, b_scale_interleaved = ops.scaled_fp4_quant(b_dtype, b_global_scale)

    a_fp4_full, a_sf_full = ops.scaled_fp4_quant(a_dtype, a_global_scale)
    out_full = ops.cutlass_scaled_fp4_mm(
        a_fp4_full, b_fp4, a_sf_full, b_scale_interleaved, alpha, dtype,
    )

    for i in range(m):
        a_row = a_dtype[i : i + 1]
        a_fp4_row, a_sf_row = ops.scaled_fp4_quant(a_row, a_global_scale)
        out_row = ops.cutlass_scaled_fp4_mm(
            a_fp4_row, b_fp4, a_sf_row, b_scale_interleaved, alpha, dtype,
        )

        # 逐行断言：在 batch-invariant 模式下，M=1 和 M=full 的第 i 行必须严格相等
        assert torch.equal(out_full[i], out_row[0]), (
            f"VLLM_BATCH_INVARIANT: row {i} differs between M={m} and M=1: "
            f"max_abs_diff={(out_full[i] - out_row[0]).abs().max().item()}"
        )
```

评论区精华

1. 测试必要性：@yewentao256 最初认为只需要 *e2e* 测试，@jzakrzew 认为单元测试可提供更直接的覆盖，最终保留单元测试。
2. 配置结构体重用：@yewentao256 建议避免复制代码，直接绑定到现有配置结构体，@jzakrzew 采纳并添加注释。
3. 测试位置：测试文件从 kernels.yaml 移至 misc.yaml 的 determinism 目录，与其他 batch invariant 测试保持一致。
4. seed 与环境变量：讨论测试中 seed 的使用以及 C++ 侧环境变量缓存问题，最终使用 VLLM_TEST_SEED 和 .conftest 中的 *enable_batch_invariant_mode* fixture。

- 是否需要 batch invariance 单元测试 (testing): 保留单元测试 test_nvfp4_gemm_batch_invariance，并移入 determinism 测试目录。
- 配置结构体重用 (design): 在默认配置中显式指定 TileScheduler，并添加 // Do not change... 注释，不新增独立配置结构体。
- 测试文件放置位置 (other): 移至 .buildkite/test_areas/misc.yaml 的 Batch Invariance (B200) 步骤。
- seed 与环境变量缓存 (correctness): 使用 VLLM_TEST_SEED 环境变量，并通过 conftest 的 *enable_batch_invariant_mode* fixture 设置环境变量。

风险与影响

- 风险：
  1. 环境变量缓存污染：C++ 端 vllm_is_batch_invariant() 在首次调用后缓存结果，如果同一进程中先运行了非 batch-invariant 测试，后续即使设置环境变量也不会生效。
  2. CUTLASS 路径一致性：不同 CUDA 架构或 CUTLASS 版本可能产生数值差异，当前测试覆盖的形状有限，长尾形状可能暴露问题。
  3. 静态断言覆盖不全：静态断言仅检查 TileScheduler 类型，但其他属性（如 EpilogueSchedule）改变仍可能破坏 invariance。
  4. 性能回退：batch-invariant 模式固定使用 persistent scheduler，在小 M 场景下可能比动态调度略慢。 - 影响：仅当用户显式设置 VLLM_BATCH_INVARIANT=1 时才会影响 NVFP4 GEMM 行为。默认行为不变。该功能主要用于对确定性有严格要求的场景，如测试、调试。团队需要维护两条代码路径，但代码注释和断言有助于防止退化。 - 风险标记：环境变量缓存污染，静态断言覆盖不全，测试形状有限，CUTLASS 路径分支增加

关联脉络

- PR #39727 [Kernel] Batch invariant NVFP4 linear using cutlass (alternative approach): 本 PR 是 #39727 的替代方案，采用不同实现方式（显式 flag vs 隐式 模式）。
- PR #35993 Add vllm_is_batch_invariant() helper and cache env var: 讨论中提到需要该 PR 的环境变量缓存机制，本 PR 依赖该 helper 函数。