

PR #54809 完整报告

vllm-project/vllm

[Quant][Kernel] Remove GPTQ Group/Dynamic Activation Ordering

合并时间: 2026-09-09 03:45

原文链接: <http://prhub.com.cn/vllm-project/vllm/pull/54809>

执行摘要

- 一句话: 移除 GPTQ Group/Dynamic Activation Ordering 支持, 简化量化内核与数据契约。
- 推荐动作: 该 PR 值得精读, 因为它代表了一个重要的架构简化决策。关键设计决策包括:
 1. 选择完全移除而非保持向后兼容, 以换取长期的性能优势和代码简洁性;
 2. 在内核模板中使用 `static_assert` 确保编译时安全;
 3. 协调了压缩库和推理引擎的移除步骤。对于维护量化模块或关注推理性能的工程师, 该变更具有高参考价值。

功能与动机

根据 Issue #48148 (RFC), 移除运行时组激活排序 (actorder) 的主要动机是:

1. 现代高性能内核 (如 Humming、TensorRT-LLM、FlashInfer) 不支持基于 `g_idx` 的运行时重排序;
2. 组 actorder 在运行时产生约 2.5% 的性能开销;
3. 静态激活排序 (在量化时重排序权重) 提供类似的精度但零运行时开销;
4. 支持 actorder 在 vLLM (W4A16、W8A16、MoE 变体)、compressed-tensors 和 llm-compressor 中增加了显著的代码复杂度。该变更已获得社区批准, 旨在简化代码库并提升性能。

实现拆解

1. 数据契约简化: 从 `MPLinearLayerConfig` 数据类中移除 `has_g_idx` 字段, 从 `MPLinearKernel.__init__()` 中移除 `w_gidx_param_name` 参数。在 `marlin_utils.py` 中简化 `marlin_is_k_full` 和 `marlin_repeat_scales_on_all_ranks` 函数签名, 移除 `act_order` 参数, 并删除 `marlin_make_empty_g_idx` 和 `marlin_sort_g_idx` 函数。
2. 内核与算子清理: 在 `_custom_ops.py` 中移除 `gptq_gemm`、`gptq_shuffle`、`gptq_gemm_rdna3`、`gptq_marlin_repack`、`marlin_gemm`、`moe_wna16_marlin_gemm` 和 `cpu_gemm_wna16` 等函数的 `g_idx/perm/is_k_full` 参数。删除 `permute_cols` 函数。在 `csrc/libtorch_stable/quantization/marlin/marlin_template.h` 和 `csrc/libtorch_stable/moe/marlin_moe_wna16/marlin_template.h` 中移除 `has_act_order` 模板参数、`g_idx` 参数及相关设备端逻辑 (如 `scale4`、`fetch_act_order_scales_to_shared`)。

3. 量化配置调整：在 `compressed_tensors_moe_wna16.py` 中移除 `ActivationOrdering` 导入、`self.actorder` 属性、`is_actorder` 变量、`_w2_scale_sharding` 方法，以及创建 `w13_weight_g_idx`、`w2_weight_g_idx`、`w13_g_idx_sort_indices`、`w2_g_idx_sort_indices` 参数的代码。简化 `create_weights` 和 `process_weights_after_loading` 方法。
4. MoE 与权重加载逻辑更新：在 `routed_experts.py` 中移除 `_needs_intermediate_size_param` 方法、`_load_g_idx` 方法以及 `weight_loader` 中处理 `g_idx` 的代码块。简化 `_load_model_weight_or_group_weight_scale` 和 `_load_w2` 方法，移除 `load_full_w2` 参数。
5. 测试配套：更新或移除相关测试，包括 `tests/lora/test_quant_model.py`（更换测试模型、简化测试逻辑）、`tests/model_executor/model_loader/test_reload.py`（移除 `act-order` 相关测试）、`tests/quantization/test_compressed_tensors.py`（更新测试以验证拒绝运行时激活排序）等。

关键文件：

- `vllm/model_executor/layers/quantization/utils/marlin_utils.py`（模块 量化工具；类别 `source`；类型 `data-contract`；符号 `marlin_is_k_full`，`marlin_repeat_scales_on_all_ranks`，`marlin_make_empty_g_idx`，`marlin_make_empty`）：核心量化工具模块，移除了 `actorder` 相关函数和参数，简化了 Marlin 内核的数据契约。
- `vllm/model_executor/layers/quantization/compressed_tensors/compressed_tensors_moe/compressed_tensors_moe_wna16.py`（模块 MoE 量化；类别 `source`；类型 `data-contract`；符号 `CompressedTensorsWNA16MoEMethod`，`create_weights`，`process_weights_after_loading`，`_w2_scale_sharding`）：`CompressedTensors WNA16 MoE` 量化方法的核心实现，大幅简化了权重创建和处理逻辑，移除了 `g_idx` 相关的参数和方法。
- `vllm/_custom_ops.py`（模块 自定义算子；类别 `source`；类型 `core-logic`；符号 `gptq_gemm`，`gptq_shuffle`，`gptq_gemm_rdna3`，`gptq_marlin_repack`）：定义了 PyTorch 自定义算子的 Python 包装器，移除了所有 `g_idx` 相关参数，简化了内核接口。
- `csrc/libtorch_stable/quantization/marlin/marlin_template.h`（模块 Marlin 内核；类别 `source`；类型 `core-logic`；符号 `Marlin`，`scale4`）：Marlin 内核的核心模板，移除了大量 `actorder` 相关的设备端代码，显著简化了内核逻辑。
- `vllm/model_executor/layers/fused_moe/routed_experts.py`（模块 融合 MoE；类别 `source`；类型 `data-contract`；符号 `RoutedExperts`，`_needs_intermediate_size_param`，`_load_g_idx`，`weight_loader`）：MoE 路由专家的核心实现，移除了 `actorder` 相关的辅助方法和权重加载逻辑。
- `tests/lora/test_quant_model.py`（模块 LoRA 测试；类别 `test`；类型 `test-coverage`；符号 `test_quant_model_lora`，`test_quant_model_tp_equality`，`do_sample`）：量化模型 LoRA 测试，更新了测试模型和测试逻辑，验证了简化后的量化路径。

关键符号：`marlin_is_k_full`，`marlin_repeat_scales_on_all_ranks`，`marlin_make_empty_g_idx`，`marlin_sort_g_idx`，`apply_gptq_marlin_linear`，`gptq_gemm`，`gptq_shuffle`，`gptq_marlin_repack`，`marlin_gemm`，`moe_wna16_marlin_gemm`，`permute_rows`，`sort_weights`，`gptq_quantize_weights`，`_needs_intermediate_size_param`，

```
_load_g_idx, CompressedTensorsWNA16MoEMethod,  
create_weights, process_weights_after_loading, _w2_scale_sharding,  
select_wna16_moe_backend, make_wna16_moe_kernel, _process_weights_marlin
```

关键源码片段

vllm/model_executor/layers/quantization/utils/marlin_utils.py

核心量化工具模块，移除了 actorder 相关函数和参数，简化了 Marlin 内核的数据契约。

```
# 变更后的 marlin_repeat_scales_on_all_ranks 函数  
# act_order 参数已移除，逻辑简化
```

```
def marlin_repeat_scales_on_all_ranks(group_size: int, is_row_parallel: bool) -> bool:  
    # 是否需要在所有 rank 上重复 scales  
    is_channelwise = group_size == -1  
    return is_channelwise and is_row_parallel
```

```
# 变更后的 apply_gptq_marlin_linear 函数签名  
# 移除了 g_idx, g_idx_sort_indices, is_k_full 参数
```

```
def apply_gptq_marlin_linear(  
    input: torch.Tensor,  
    weight: torch.Tensor,  
    weight_scale: torch.Tensor,  
    weight_zp: torch.Tensor,  
    workspace: torch.Tensor,  
    wtype: ScalarType,  
    output_size_per_partition: int,  
    input_size_per_partition: int,  
    input_global_scale: torch.Tensor | None = None,  
    bias: torch.Tensor | None = None,  
    use_fp32_reduce: bool = USE_FP32_REDUCE_DEFAULT,  
    input_dtype: torch.dtype | None = None,  
) -> torch.Tensor:  
    # 函数体简化，不再处理 g_idx 重排序  
    ...
```

vllm/model_executor/layers/quantization/compressed_tensors/compressed_tensors_moe/compressed_tensors_moe_wna16.py

CompressedTensors WNA16 MoE 量化方法的核心实现，大幅简化了权重创建和处理逻辑，移除了 g_idx 相关的参数和方法。

```
# 变更后的 __init__ 方法片段  
# actorder 属性和 is_actorder 变量已移除
```

```
class CompressedTensorsWNA16MoEMethod(CompressedTensorsMoEMethod):  
    def __init__(self, weight_quant, input_quant, moe, layer_name=None):  
        ...
```

```

self.group_size = (
    -1 if weight_quant.group_size is None else weight_quant.group_size
)
# actorder 相关属性已移除

# oracle 选择时, allow_tile_padding 始终为 True
self.wna16_backend, self.experts_cls = select_wna16_moe_backend(
    config=self.moe,
    weight_key=weight_key,
    quant_config=self.weight_quant,
    may_have_zp=not self.symmetric,
    may_have_bias=False,
    allow_tile_padding=True,
)
...
# create_weights 方法中不再创建 g_idx 相关参数
def create_weights(self, layer, num_experts, hidden_size, intermediate_size_per_partition,
    params_dtype, **extra_weight_attrs):
    ...
    # 不再有 intermediate_size_full 参数
    # 不再创建 w13_weight_g_idx, w2_weight_g_idx, w13_g_idx_sort_indices, w2_g_idx_sort_
    indices

```

csrc/libtorch_stable/quantization/marlin/marlin_template.h

Marlin 内核的核心模板，移除了大量 actorder 相关的设备端代码，显著简化了内核逻辑。

```

// 变更后的内核模板参数
// has_act_order 参数已移除
template <
    typename scalar_t,
    const int thread_m_blocks,
    const int thread_n_blocks,
    const int thread_k_blocks,
    const int stages,
    const int group_blocks,
    const bool is_zp_float
> __global__ void Marlin(
    const int4* __restrict__ A,
    int4* __restrict__ B,
    int4* __restrict__ C,
    int4* __restrict__ C_tmp,
    const int4* __restrict__ scales_ptr,
    // g_idx 参数已移除
    int prob_m, int prob_n, int prob_k,
    int* locks,
    bool use_fp32_reduce
) {
    // 添加了静态断言，禁止运行时激活排序
    static_assert(group_blocks != 0, "runtime activation ordering is not supported");

```

```
// ... 其余逻辑简化 ...  
}
```

评论区精华

Review 讨论主要集中在 CI 流程和合并冲突解决上。mgoin主导了代码清理和修复工作，并多次触发 CI。mergify[bot]多次报告合并冲突和 pre-commit 检查失败，要求作者 rebase 并运行 pre-commit。coderabbitai[bot]自动审查未生成可操作评论。claude[bot]指出这是来自 fork 的 PR，自动化审查已禁用。最终 mgoin以“It is so beautiful 🥰”批准了该 PR，表明对变更质量和影响的积极评价。

- CI 流程与合并冲突解决 (other): 通过多次 rebase、pre-commit 修复和 CI 重跑，最终解决了所有技术问题。
- 自动化审查与社区参与 (other): 由仓库维护者 mgoin 手动审查并批准，体现了社区协作模式。

风险与影响

- 风险：
 1. 回归风险：大规模删除（3918 行）可能遗漏依赖项，但测试更新（29 个测试文件）和 maintainer 审查降低了风险。移除 has_g_idx 字段和 g_idx 参数是破坏性变更，任何依赖旧检查点格式或内部 API 的外部代码都会失败。
 2. 性能风险：主要为正向影响（消除 2.5% 运行时开销）。内核模板简化（移除 has_act_order 分支）可能影响生成代码路径，但编译时静态断言（static_assert(group_blocks != 0)）确保了正确性。
 3. 兼容性风险：已知影响：使用 actorder=group 或 desc_act=True 的 GPTQ 量化模型（如部分旧版 TheBloke GPTQ 模型）将无法加载。RFC 中已说明影响范围有限，大多数现代模型使用静态重排序。压缩库（compressed-tensors、llm-compressor）已先行移除该特性。
 4. 测试风险：测试模型从 TinyLlama 更换为 Qwen3-0.6B，测试逻辑简化。移除了 GPTQ lora 测试不稳定性的 workaround，表明对量化后 lora 稳定性的信心。
 - 影响：对用户的影响：使用旧版 GPTQ 量化模型（actorder=group）的用户需要重新量化模型或使用兼容的旧版本。对系统的影响：代码库减少约 3900 行，内核参数简化，启用了更高效的量化内核路径。对团队的影响：显著降低量化模块的维护负担，简化跨平台（ROCm、Intel GPU、CPU）适配，为未来高性能内核的采用铺平道路。
 - 风险标记：核心路径变更，破坏性变更，大规模代码删除，内核接口简化，向后兼容性打破

关联脉络

- PR #48148 [RFC]: Remove Group Activation Ordering (actorder=group) Support from vLLM and Compression Libraries: 本 PR 的 RFC 提案，详细说明了移除 actorder 的动机、影响范围和实施计划。
- PR #8135 Original actorder PR: 最初引入 actorder 支持的 PR，本 PR 是对其的完全移除。
- PR #46390 Humming kernel PRs: Humming 内核 PR，本 PR 的动机之一是启用此类不支持 actorder 的高效内核。

- PR #42997 Humming kernel PRs: 另一个 Humming 内核 PR, 与本 PR 的性能目标相关。