

PR #47972 完整报告

vllm-project/vllm

Support DeepSeek-V4 AMD Quark NVFP4 with emulation kernel

合并时间: 2026-08-07 16:19

原文链接: <http://prhub.com.cn/vllm-project/vllm/pull/47972>

执行摘要

- 一句话: 支持 DeepSeek-V4 Quark 混合量化 checkpoint 加载
- 推荐动作: 值得精读。重点关注三个设计决策: per-block FP8 拆分为独立 scheme 类的模式 (对齐 modelopt 写法)、通用 QuarkConfig 与模型专属映射的边界划分 (含 packed_modules_mapping 类级复制细节)、process_weights_after_loading 只做委托不写后端特判的约定。评审中关于 NVFP4 MoE w1/w3 global scale 策略 (raise vs max vs w1) 的交锋也值得回味, 最终选择了与 ModelOpt / compressed-tensors 一致的既有约定。

功能与动机

PR body 明确目标: "Add support for DeepSeek-V4 AMD Quark mixed-quantized checkpoints, where different parts of the model may use different quantization layouts, including per-block FP8 linear layers and NVFP4 MoE experts." 需要解决三点: 既有 QuarkW8A8Fp8 只覆盖 per-tensor/per-channel, 遇到 per-block 配置会静默按默认分块处理; 若把 gate_up_proj / fused_wqa_wkv 等 DeepSeek-V4 专属 fused 映射注入通用 QuarkConfig, 会让通用量化配置类带上架构知识; Quark NVFP4 MoE 的 w1/w3 global scale 处理与既有 ModelOpt / compressed-tensors NVFP4 MoE 约定不一致。同时 PR 明确 Out of Scope: 不引入新的 NVFP4 scale 选择策略、不做通用 NVFP4 MoE scale 处理重构、per-block FP8 路径在 Quark 导出 checkpoint 可用前不做完整 E2E 声明。

实现拆解

变更入口是量化配置的 scheme 路由, 随后是 DeepSeek-V4 模型侧适配与 CI 评估配套, 共 5 步:

1. 新增 per-block FP8 量化 scheme: 在 quark_w8a8_fp8.py 新增 QuarkW8A8Fp8PerBlock, 构造时强制 qscheme == per_block 且必须提供 block_size; 激活键固定 kFp8Dynamic128Sym、权重键固定 kFp8Static128BlockSym; create_weights 用 create_fp8_weight_parameter / create_fp8_scale_parameter (BlockQuantScaleParameter) 创建参数, scale 注册为 weight_scale_inv; process_weights_after_loading 仅一行委托给 fp8_linear。原 QuarkW8A8Fp8 对 per_block 配置直接抛错并提示使用新类, schemes/init.py 同步导出。
2. 扩展 QuarkConfig 识别与路由: quark.py 的 _is_fp8_w8a8 新增 per-block 分支 (权重静态对称 + block_size [128, 128] + 激活动态 per-group, 且 group_size 取 block_size[1]); _get_scheme_from_config 按 per_block 路由到新类; 同时把

packed_modules_mapping 改为从类级复制 (dict(self.packed_modules_mapping))，修复基类实例级空 dict 遮蔽子类类级覆盖的问题。

3. DeepSeek-V4 模型侧适配: deepseek_v4/amd/model.py 的 DeepseekV4ForCausalLM 增加 packed_modules_mapping 类属性; load_weights 新增嵌套 _resolve_param_name 对 *_inv 后缀做兜底解析; _make_deepseek_v4_weights_mapper 增加 .input_scale -> .input_scale_2 后缀映射。
4. NVFP4 MoE global scale 策略对齐: quark_moe.py 把 w1/w3 global scale 不一致从 raise 改为「告警 + 取 w1 ([:, 0])」, 对齐 ModelOpt 与 compressed-tensors 的既有约定; fused_moe/oracle/nvfp4.py 的允许后端集合加入 NvFp4MoeBackend.EMULATION, 使 ROCm 仿真 kernel 可被选中。
5. 测试与 CI 配套: test_quark.py 新增 7 个轻量用例 (per-block 检测、静态输入拒绝、group_size 不匹配拒绝、block_size 缺失报错、QuarkConfig 不含模型专属映射、类级映射保留); 新增 DeepSeek-V4-Pro-NVFP4 与 Flash 两份 GSM8K 评估配置, Flash 注册进 models-gfx950-large.txt, .buildkite/test-amd.yaml 增加 TP=8 的 LM Eval Large Models (8xB200-8xMI355) optional 评估组。

关键文件:

- vllm/model_executor/layers/quantization/quark/schemes/quark_w8a8_fp8.py (模块 量化方案; 类别 source; 类型 core-logic; 符号 QuarkW8A8Fp8PerBlock, QuarkW8A8Fp8.init, QuarkW8A8Fp8PerBlock.create_weights, QuarkW8A8Fp8PerBlock.process_weights_after_loading): 本 PR 核心: 新增 QuarkW8A8Fp8PerBlock scheme, 独立处理 per-block FP8 的参数创建、kernel 初始化与权重后处理, 并在 QuarkW8A8Fp8 中加入 per_block 引导性报错。
- vllm/model_executor/layers/quantization/quark/quark.py (模块 量化配置; 类别 source; 类型 core-logic; 符号 QuarkConfig.init, QuarkConfig._is_fp8_w8a8, QuarkConfig._get_scheme_from_config): 通用 Quark 配置的识别与路由改造: _is_fp8_w8a8 增加 per-block 分支并将 group_size 校验绑定到 block_size[1], _get_scheme_from_config 路由到新 scheme, packed_modules_mapping 改为类级复制以支持子类覆盖。
- vllm/models/deepseek_v4/amd/model.py (模块 模型适配; 类别 source; 类型 data-contract; 符号 DeepseekV4ForCausalLM.packed_modules_mapping, DeepseekV4ForCausalLM.load_weights, _resolve_param_name, _make_deepseek_v4_weights_mapper): DeepSeek-V4 模型侧适配: fused 模块映射移到模型类, load_weights 用 _resolve_param_name 兜底解析 _inv 后缀, weights mapper 增加 input_scale 命名映射, 支撑 Quark 混合量化 checkpoint 的权重加载。
- tests/quantization/test_quark.py (模块 量化测试; 类别 test; 类型 test-coverage; 符号 test_quark_config_has_no_model_specific_fused_mappings, test_quark_config_preserves_existing_packed_modules_mapping, CustomQuarkConfig, test_quark_fp8_w8a8_detects_per_block_config): 新增 7 个轻量测试, 锁定 per-block 配置检测、拒绝路径、block_size 必填、QuarkConfig 不含模型专属映射与类级映射保留, 是防止通用路由回归的关键防线。

- `vllm/model_executor/layers/quantization/quark/quark_moe.py` (模块 MoE 量化; 类别 source; 类型 core-logic; 符号 `QuarkW4A8Nvfp4MoEMethod.process_weights_after_loading`) : NVFP4 MoE 的 w1/w3 global scale 处理从硬报错改为告警并使用 w1 scale, 与 `ModelOpt / compressed-tensors` 既有约定对齐, 是评审中最富争议的改动点。
- `vllm/model_executor/layers/quantization/quark/schemes/__init__.py` (模块 量化方案; 类别 source; 类型 data-contract; 符号 `QuarkW8A8Fp8PerBlock`) : scheme 统一出口, 导出新类 `QuarkW8A8Fp8PerBlock`, 供 `QuarkConfig` 与外部引用。
- `tests/evals/gsm8k/configs/DeepSeek-V4-Flash-NVFP4.yaml` (模块 评估配置; 类别 test; 类型 test-coverage) : GSM8K 评估配置 (TP=8, AITER 仿真路径), Flash 模型作为 per-block FP8 + NVFP4 MoE 混合布局的更小、更快的 CI 信号源。
- `tests/evals/gsm8k/configs/DeepSeek-V4-Pro-NVFP4.yaml` (模块 评估配置; 类别 test; 类型 test-coverage) : GSM8K 评估配置 (TP=8, 启动超时 3600s), Pro 模型是唯一同时覆盖 per-block FP8 attention + NVFP4 experts 混合布局的原始验证目标, 后因 CI 超时暂停注册。
- `vllm/model_executor/layers/fused_moe/oracle/nvfp4.py` (模块 MoE 后端; 类别 source; 类型 configuration; 符号 `select_nvfp4_moe_backend`) : NVFP4 MoE 后端选择集合加入 EMULATION, 使 ROCm 仿真 kernel 路径可被正常选中。
- `.buildkite/test-amd.yaml` (模块 CI 配置; 类别 config; 类型 configuration) : 新增 TP=8 的 LM Eval Large Models (8xB200-8xMI355) optional 评估组, 是 DeepSeek-V4 NVFP4 的 CI 落地载体。
- `tests/evals/gsm8k/configs/models-gfx950-large.txt` (模块 评估配置; 类别 docs; 类型 documentation) : gfx950 大模型评估清单, 注册 DeepSeek-V4-Flash-NVFP4, Pro 暂被注释。

关键符号: `QuarkW8A8Fp8PerBlock.init`, `QuarkW8A8Fp8PerBlock.create_weights`, `QuarkW8A8Fp8PerBlock.process_weights_after_loading`, `QuarkW8A8Fp8PerBlock.apply_weights`, `QuarkConfig._is_fp8_w8a8`, `QuarkConfig._get_scheme_from_config`, `DeepseekV4ForCausalLM.load_weights`, `DeepseekV4ForCausalLM._resolve_param_name`, `QuarkW4A8Nvfp4MoEMethod.process_weights_after_loading`

关键源码片段

`vllm/model_executor/layers/quantization/quark/quark.py`

通用 Quark 配置的识别与路由改造: `_is_fp8_w8a8` 增加 per-block 分支并将 `group_size` 校验绑定到 `block_size[1]`, `_get_scheme_from_config` 路由到新 scheme, `packed_modules_mapping` 改为类级复制以支持子类覆盖。

```
def _is_fp8_w8a8(
    self,
    weight_quant: dict[str, Any] | None,
    input_quant: dict[str, Any] | None,
) -> bool:
    # 权重与输入都必须存在, 且权重为静态 FP8 E4M3
```

```

if weight_quant is None or input_quant is None:
    return False
is_fp8_dtype = (
    weight_quant.get("dtype") == "fp8_e4m3"
    and input_quant.get("dtype") == "fp8_e4m3"
)
is_static_weight = not weight_quant.get("is_dynamic")
if not is_fp8_dtype or not is_static_weight:
    return False

# per-block 分支: 仅接受 [128, 128] 对称权重量化, 且激活侧必须是
# 动态 per-group; group_size 直接取 block_size[1], 避免两处常量漂移
block_size = list(weight_quant.get("block_size") or [])
is_per_block_weight = (
    weight_quant.get("qscheme") == "per_block"
    and block_size == [128, 128]
    and weight_quant.get("symmetric") is True
)
if is_per_block_weight:
    if not input_quant.get("is_dynamic"):
        return False
    return (
        input_quant.get("qscheme") == "per_group"
        and input_quant.get("group_size") == block_size[1]
        and input_quant.get("symmetric") is True
    )

# 原有 per-tensor / per-channel 识别逻辑保持不变
is_per_tensor_or_channel_weight = weight_quant.get("qscheme") in [
    "per_tensor",
    "per_channel",
]
if not is_per_tensor_or_channel_weight:
    return False
if input_quant.get("is_dynamic"):
    return True
is_per_tensor_activation = input_quant.get("qscheme") == "per_tensor"
return is_per_tensor_activation

```

vllm/models/deepseek_v4/amd/model.py

DeepSeek-V4 模型侧适配: fused 模块映射移到模型类, load_weights 用 _resolve_param_name 兜底解析 _inv 后缀, weights mapper 增加 input_scale 命名映射, 支撑 Quark 混合量化 checkpoint 的权重加载。

```

class DeepseekV4ForCausalLM(nn.Module, SupportsPP, SupportsEagle3):
    # 默认 mapper 假设原始 FP4 experts 的 checkpoint 布局,
    # 非 FP4 (如 NVFP4) 场景会在 __init__ 中按 expert_dtype 重建 mapper
    hf_to_vllm_mapper = _make_deepseek_v4_weights_mapper("fp4")
    # DeepSeek-V4 特有的 fused 模块映射放在模型侧而非通用 QuarkConfig,

```

```

# 避免把架构知识注入通用量化配置类
packed_modules_mapping = {
    "gate_up_proj": ["w1", "w3"],
    "fused_wqa_wkv": ["wq_a", "wkv"],
}

# load_weights 内的嵌套函数：做 scale 参数名的兜底解析
def _resolve_param_name(name: str) -> str:
    # Quark 混合量化 checkpoint 的 scale 命名与 vLLM 参数命名存在差异，
    # checkpoint 路由过来的名字可能不带 _inv 后缀，而 block-scale 参数
    # 实际注册为 *_weight_scale_inv，这里做一次兜底解析
    inv_name = f"{name}_inv"
    if name not in params_dict and inv_name in params_dict:
        return inv_name
    return name

# stacked/mapping 与默认路径统一使用解析后的参数名：
# param_name = _resolve_param_name(name)
# param = params_dict[param_name]
# weight_loader(param, loaded_weight, shard_id)
# loaded_params.add(param_name)

```

评论区精华

核心交锋围绕「通用配置类的边界」与「NVFP4 MoE scale 策略」展开：

- kewang-amd 指出 fused_wqa_wkv 是 AMD DeepSeek-V4 专属映射，注入通用 QuarkConfig 属于架构知识污染，作者最终把映射移到模型侧并新增回归测试。
- kewang-amd 指出 [128, 128] 与 group_size == 128 两常量独立硬编码会漂移，最终 group_size 改为从 block_size[1] 推导。
- fxmarty-amd 坚持 per-block 应参照 modelopt 拆成独立类，并要求 process_weights_after_loading 内不应存在 MarlinFP8ScaledMMLinearKernel 特判，最终 scheme 层只保留一行委托。
- fxmarty-amd 对 w1/w3 global scale 从报错放宽到告警的动机提出质疑，ColinZ22 补充 nvidia/GLM-5.1-NVFP4 存在不一致层，结论是采用「告警 + 取 w1」对齐既有约定，统一工具函数留作 follow-up。
- kewang-amd 追问 per-block FP8 的 Quark 导出 checkpoint E2E 验证，作者承认缺口并待 checkpoint 可用后补齐。
- AndreasKaratzas 坚持 GSM8K 评估组通过后才合入，期间历经 fixture 缺失、850GiB NFS 冷读启动超时（1200s -> 3600s）、HF 文件缺失等基础设施问题，最终改用 149B 的 Flash 模型注册。
- QuarkConfig 是否应注入 DeepSeek-V4 fused 映射 (design)：作者把 gate_up_proj / fused_wqa_wkv 映射移到 DeepseekV4ForCausalLM 模型侧，并新增 test_quark_config_has_no_model_specific_fused_mappings 锁定回归。

- block_size 与 group_size 硬编码漂移 (correctness): 激活 group_size 校验改为直接比较 `input group_size == block_size[1]`。
- per-block FP8 是否拆成独立 scheme 类 (design): 拆出独立类, QuarkW8A8Fp8 对 per_block 直接抛错引导; create/process/apply 路径各自独立。
- process_weights_after_loading 中的 Marlin 特判 (design): 删除特判, process_weights_after_loading 仅一行委托 `self.fp8_linear.process_weights_after_loading(layer)`。
- NVFP4 MoE w13 的 w1/w3 global scale 不一致策略 (correctness): 采用与 ModelOpt / compressed-tensors 一致的「告警 + 取 w1 (`[:, 0]`)」约定, 不引入新策略; 统一工具函数留作 follow-up PR。
- parameter.py 通用权重加载器改动 (style): 整段 revert, PR 最终不再修改通用权重加载逻辑。
- attention.py 3-D packed hidden state 的归属 (design): 拆出为独立 PR #48598, 本 PR 只保留量化相关改动。
- Quark per-block FP8 的 E2E 验证缺口 (question): 未解决: 待 Quark 导出 checkpoint 可用后补充; 作者用 Qwen3-8B-FP8 对比基线确认对上游无副作用。
- GSM8K TP=8 大模型评估组的 CI 可行性 (testing): Pro 模型暂停注册, 改用约 1/6 体积的 DeepSeek-V4-Flash-NVFP4 注册到 models-gfx950-large.txt; CI 组保留为 optional。

风险与影响

- 风险: 风险点如下:
 - 通用 Quark 路由改动: `_is_fp8_w8a8` 被所有 Quark FP8 checkpoint 共享, 分支重构存在回归风险; 已保留原 per-tensor/per-channel 语义并用单测锁定。
 - 精度相关策略放宽: `quark_moe.py` 从抛错改为告警, 若 checkpoint 的 w1/w3 scale 真不一致会静默使用 w1 scale, GSM8K 之外的困难任务可能受影响 (ColinZ22 证实 GLM-5.1-NVFP4 上存在不一致层)。
 - 新 scheme 强绑定 [128, 128] 静态对称权重 + 动态 per-group 激活, 其他 per-block 形状会被直接拒绝, 限制性强但避免未验证路径。
 - `_resolve_param_name` 兜底解析若两套命名都不存在会继续用原名, 最终以 `KeyError` 暴露, 错误信息不够直接。
 - E2E 验证缺口: per-block FP8 路径目前只有 DeepSeek-V4 (非 Quark 导出) 与轻量单测覆盖。
 - CI 评估组依赖大 checkpoint 与 AITER 仿真路径, 存在超时与不稳定风险, 且 Pro 模型依赖 HF 公开发布节奏。
- 影响: 影响评估:
 - 用户侧: AMD MI350/MI355 用户可加载 DeepSeek-V4 Pro/Flash NVFP4 混合量化模型, Flash 模型 GSM8K 从 95.00 到 94.84, 精度恢复 99.83%。
 - 框架侧: Quark 量化家族获得首个 per-block FP8 linear scheme, 后续 Quark 导出的 per-block checkpoint 可直接受益。

- CI 侧：新增 optional 的 TP=8 大模型评估组，单次运行成本高（约 1 小时量级），依赖 amd 组织下的量化模型发布与共享存储性能。
- 团队协作：展示了 AMD 与上游围绕量化模型支持、CI 评估组、模型发布节奏的完整协作流程。
- 风险标记：共享量化路由变更，精度相关策略放宽，E2E 验证缺口，大模型 CI 评估组，依赖 amd 组织模型发布

关联脉络

- PR #48598 [DeepSeek-V4] Handle 3-D packed hidden states in attention forward: 本 PR 评审中按 fxmarty-amd 要求拆出的独立 PR，处理 HC/MTP 3-D packed hidden states，作者在讨论中明确引用 #48598。
- PR #49634 [评论引用，标题未知] gsm8k eval fixture 修复：AndreasKaratzas 与作者讨论 CI 评估组 fixture "config_filename not found" 问题时引用，作者认为该 PR 已修复对应问题。
- PR #50859 [评论引用，标题未知] Quantized Models Test hang 修复：fxmarty-amd 指出 AMD Quantized Models Test (mi355_1) 的 hang 由该 PR 修复，与本 PR 无关。
- PR #50879 [评论引用，标题未知] 合入 main 引发的回归（已被 revert）：作者在评论中指出本 PR 合并 main 后遇到的问题可能由 #50879 引起，并在本地验证 revert 该 PR 后无错。
- PR #47106 [Kernel] Support Nvfp4 Cutedsl Moe Swiglu-oai and Relu2(non-gated) Activation: 同属 NVFP4 MoE 量化支持线，涉及 fused_moe/oracle/nvfp4.py 的后端选择逻辑，与本 PR 纳入 EMULATION 后端属同一演进方向。