

PR #39155 完整报告

sgl-project/sglang

[AMD] GLM-5.2 NextN: cast draft fused MoE to per-channel FP8

合并时间: 2026-09-16 09:56

原文链接: <http://prhub.com.cn/sgl-project/sglang/pull/39155>

执行摘要

- 一句话: GLM-5.2 草稿专家层转逐通道 FP8, 修复 Quark AITER 路径不可达
- 推荐动作: 值得精读, 尤其是量化 / 推理性能方向的工程师。建议重点看三处设计决策: (1) `should_apply_glm_nextn_moe_ptpc` 这一“同一判定服务权重转换与模块构建”的模式, 以及提交 64c44144 为何要专门对齐两侧门控——这是静默数值错误的高发点; (2) `_resolve_nextn_quant_config` 从直接改动共享 `QuarkConfig` 改为 `copy + deepcopy` 的过程, 以及配套的 `test_flag_on_does_not_mutate_caller_config`; (3) `QuarkW8A8FP8MoE` 的 runner 选择从死标志 `apply_router_weight_on_input` 改为 `kernel` 前置条件, 并用 `print_info_once` 让回退可观测。若你负责 ROCm 上其它 quark FP8 MoE 模型的线上服务, 建议在升级后对比 AITER 与 `--moe-runner-backend triton` 的表现与数值一致性和显存形态。此外, PR body 与最终实现存在文件位置不一致 (body 仍写 `deepseek_weight_loader.py`), 阅读时应以 `glm4_moe.py` 的 `GlmMoeDsaForCausalLMNextN` 为准。

功能与动机

GLM-5.2 的 MXFP4 checkpoint 把 MTP 草稿层的 `routed/shared experts` 保留为 bf16 (71.7 MB/ 专家, 对比 MXFP4 解码层 19.0 MB), 而 MTP 每个 draft step 都要完整读一遍该层; 在带宽受限的 decode 场景下, 它的实际开销远超参数占比。原文表述为: "Layer 78 of GLM-5.2 is the MTP draft layer... In bandwidth-bound decode it therefore costs more than its share of the parameter count suggests."。另一动机是接线时发现的既有缺陷: "QuarkW8A8FP8MoE could never reach its AITER runner, so every quark per-channel FP8 MoE on ROCm was silently running Triton."。此外 NVIDIA GB200/GB300 的 GLM-5.2 FP4 配方早已通过 `SGLANG_NVFP4_CKPT_FP8_NEXTN_MOE=1` 开启等价路径, 而 AMD MI355X 的 MTP 配方仍在用 bf16 草稿专家, 存在平台能力落差。

实现拆解

1. 新增开关与统一的判定谓词 (`python/sglang/srt/envron.py`、`python/sglang/srt/models/glm4_moe.py`): 注册 `SGLANG_GLM_NEXTN_MOE_PTPC = EnvBool(False)`, 注释明确与 NVFP4 的 `block-FP8 NextN` 路径无关。在 `glm4_moe.py` 中新增三个纯函数: `enable_glm_nextn_moe_ptpc` (开关打开且 `quant_config` 为 `quark`)、`glm_nextn_mtp_fused_experts_excluded` (MTP 层的 `.mlp.experts` 出现在 `exclude_layers`, 等价于 checkpoint 里本来就是 bf16)、

`should_apply_glm_nextn_moe_ptpc`（二者与）。把“是否该量化”的前提下沉为可单测的纯逻辑，避免权重转换与模块 `scheme` 选择两处判定漂移。

2. 模块构建侧：注入 PTPC `scheme` 并隔离共享配置（`glm4_moe.py` 的 `GlmMoeDsaForCausalLMNextN._resolve_nextn_quant_config`）：先把 `caller` 复用于目标模型和 `lm_head` 的 `QuarkConfig` 做 `copy.copy` + 对内部 `quant_config` dict 做 `copy.deepcopy`，消除草案层专用改写对共享对象的副作用；开关打开时给 `model.decoder.mlp.experts` 注入静态 per-channel 权重 + 动态 per-channel 输入的 `fp8_e4m3` 方案（即 `QuarkW8A8FP8MoE` 的 per-token 输入），否则沿用原来的粗粒度 `bf16 exclude` 行为。
3. 加载侧：惰性、流式的权重重量化（`glm4_moe.py` 的 `_maybe_quant_glm_nextn_moe_to_ptpc / load_weights`）：用 `_GLM_NEXTN_EXPERT_PROJ_RE` 只命中 `mlp.experts.N.{gate_proj,up_proj,down_proj}.weight`，逐张量 `fp32` 化后按输出通道做 `amax` 得到 `scale`（clamp `1e-12` 防除零），产出 `fp8_e4m3` 权重与配套 `weight_scale`；非 2D 张量直接 `ValueError`。转换以 generator 形式插在 `super().load_weights()` 之前，加载保持流式，峰值额外内存约等于单份 expert 权重。
4. 量化后端：修正 runner 选择与权重布局（`python/sglang/srt/layers/quantization/quark/schemes/quark_w8a8_fp8_moe.py`）：`create_moe_runner` 改用 `_use_aiter` and `is_weight_per_channel` and `a2a_supports_aiter` and `(auto or aiter)` 决定 AITER，使 `auto` 与其同目录兄弟方案（`quark_w4a4_mxfp4_moe.py`、`quark_w4a8_mxfp4_moe.py`）一致；`process_weights_after_loading` 的预重排改按 `runner_backend.is_aiter()` 判定并给权重打 `is_shuffled` tag（Triton 保持 canonical layout）；`apply_weights` 的 AITER 分支改走 `AiterMoeQuantInfo(AiterQuantType.PER_TOKEN, expert_mask=layer.dispatcher.expert_mask_gpu)`，删除不再使用的 `rocm_fused_experts_tkw1` 导入；新增一次 `print_info_once` 暴露最终 runner，使 Triton 回退可观测。
5. 测试配套（`test/registered/unit/models/test_glm_nextn_moe_ptpc.py`）：新增 160 行 CPU 单测，`register_cpu_ci`（`est_time=5, suite="base-a-test-cpu"`），用 `unittest.mock.patch` 覆盖 `env` 与 `SimpleNamespace` 伪造 `QuarkConfig`，覆盖默认关闭、开启必须为 `Quark`、必须 MTP fused experts 被 `exclude`、`scheme` 注入字段、不污染 `caller` 配置、`cast` 只命中被 `exclude` 的 `bf16` 专家且确实产出 `weight_scale`。该测试不加载 `MXFP4` 权重，真正的 `serve+generate` 场景仍需 `ROCm + AITER + checkpoint`，未纳入本 PR。

关键文件：

- `python/sglang/srt/models/glm4_moe.py`（模块 模型加载；类别 `source`；类型 `data-contract`；符号 `_GLM_NEXTN_EXPERT_PROJ_RE`, `enable_glm_nextn_moe_ptpc`, `glm_nextn_mtp_fused_experts_excluded`, `should_apply_glm_nextn_moe_ptpc`）：本 PR 的核心：新增 PTPC 门控谓词、加载期 `fp8_e4m3` 逐通道权重重量化、`load_weights` 覆写，以及 `_resolve_nextn_quant_config` 的配置隔离与 `scheme` 注入；也是响应 HaiShaw "conversion 保持 model-local" 重构后的最终落点。
- `python/sglang/srt/layers/quantization/quark/schemes/quark_w8a8_fp8_moe.py`（模块 量化方案；类别 `source`；类型 `dependency-wiring`；符号

QuarkW8A8FP8MoE.create_moe_runner, QuarkW8A8FP8MoE.process_weights_after_loading, QuarkW8A8FP8MoE.apply_weights)：修复 AITER runner 因 apply_router_weight_on_input 死标志而永不可达的缺陷，改变了 ROCm 上所有 quarkper-channel FP8 MoE 的默认 kernel 选择、权重布局与 apply_weights 路径，是本 PR 影响面最广的一块。

- test/registered/unit/models/test_glm_nextn_moe_ptpc.py (模块 单元测试；类别 test；类型 test-coverage；符号 _quark_cfg, TestEnableGlmNextnMoePtpc, test_off_by_default, test_on_requires_quark)：新增的 CPU 单测，专门覆盖默认关闭路径之外的 ON 接线 (Quark scheme 注入与权重 cast)，弥补原 PR 因 off-by-default 导致 CI 从不执行开启态的空白；同时把“不得污染 caller 配置”的约束固化下来。
- python/sglang/srt/envron.py (模块 环境配置；类别 source；类型 configuration；符号 Envs.SGLANG_GLM_NEXTN_MOE_PTPC)：注册 SGLANG_GLM_NEXTN_MOE_PTPC 开关并显式说明它与 NVFP4 block-FP8 NextN 路径无关，是控制这项改动默认关闭、按需启用的入口。

关键符号：enable_glm_nextn_moe_ptpc, glm_nextn_mtp_fused_experts_excluded, should_apply_glm_nextn_moe_ptpc, GlmMoeDsaForCausalLMNextN._maybe_quant_glm_nextn_moe_to_ptpc, GlmMoeDsaForCausalLMNextN._cast, GlmMoeDsaForCausalLMNextN.load_weights, GlmMoeDsaForCausalLMNextN._resolve_nextn_quant_config, QuarkW8A8FP8MoE.create_moe_runner, QuarkW8A8FP8MoE.process_weights_after_loading, QuarkW8A8FP8MoE.apply_weights

关键源码片段

test/registered/unit/models/test_glm_nextn_moe_ptpc.py

新增的 CPU 单测，专门覆盖默认关闭路径之外的 ON 接线 (Quark scheme 注入与权重 cast)，弥补原 PR 因 off-by-default 导致 CI 从不执行开启态的空白；同时把“不得污染 caller 配置”的约束固化下来。

```
# test/registered/unit/models/test_glm_nextn_moe_ptpc.py (节选)
```

```
# 只注册 CPU CI: 单元测试无法注册 AMD suite;  
# 完整 serve+generate 需要 runner cache 里的 MXFP4 权重，另作 nightly  
register_cpu_ci(est_time=5, suite="base-a-test-cpu")
```

```
LAYER = 78 # GLM-5.2 的 MTP 草稿层编号  
_PTPC_ENV = "sglang.srt.models.glm4_moe.envs.SGLANG_GLM_NEXTN_MOE_PTPC.get"
```

```
class TestResolveNextnQuantConfigPtpcOn(CustomTestCase):  
    def _resolve(self, cfg, flag: bool):  
        # 不构造完整模型: __new__ 绕过 __init__, 只提供 _resolve_nextn_quant_config 需要的属性  
        model = GlmMoeDsaForCausalLMNextN.__new__(GlmMoeDsaForCausalLMNextN)  
        hf = SimpleNamespace(num_hidden_layers=LAYER)  
        with patch(_PTPC_ENV, return_value=flag):
```

```

return model._resolve_nextn_quant_config(hf, cfg)

def test_flag_on_assigns_ptpc_scheme_instead_of_bf16_exclude(self):
    out = self._resolve(_quark_cfg(), flag=True)
    # 开启时必须用 per-channel FP8 scheme 取代粗粒度 bf16 exclude
    self.assertNotIn("model.decoder.mlp.experts", out.exclude_layers)
    scheme = out.quant_config["layer_quant_config"]["model.decoder.mlp.experts"]
    self.assertEqual(scheme["weight"]["qscheme"], "per_channel")
    self.assertFalse(scheme["weight"]["is_dynamic"])
    self.assertTrue(scheme["input_tensors"]["is_dynamic"])

def test_flag_on_does_not_mutate_caller_config(self):
    src = _quark_cfg()
    # caller 还会拿这份配置去建目标模型和 lm_head, 绝不能就地改写
    self._resolve(src, flag=True)
    self.assertEqual(src.exclude_layers, list(src.exclude_layers))

```

评论区精华

1. HaiShaw 要求把转换逻辑从 DeepSeek 基类里挪出来: "consider to refactor - make weight conversion and utils more or less in `glm5_next.py` vs. deepseek base." 这是本 PR 最重要的一条设计意见——原始实现挂在 DeepSeek 的 weight loader mixin 上, GLM 专属逻辑污染了共享基类。作者回应 "refactor done", 并以提交 `5de466565f47dd00b33f5d21a574e7c4d6c1b89f` "[ROCm] Keep GLM NextN PTPC conversion model-local" 落地: 转换与工具函数最终落在 `glm4_moe.py` 的 `GlmMoeDsaForCausalLMNextN` 内部 (未新增独立 `glm5_next.py`), 并用 `should_apply_glm_nextn_moe_ptpc` 做 loader 与模块构建的双重门控。
2. 量化门控必须与权重转换对齐 (提交 `64c44144`): "Align GLM NextN PTPC module-build and weight-cast gates. Only recast MTP fused experts that Quark excluded as bf16, so an already quantized draft MoE is not rewritten while the module stays on its original scheme." 这是评审中发现的关键正确性问题: 若只改权重不改 scheme (或反之), 会出现权重已转 FP8 但模块仍按 bf16/MXFP4 解释的静默错误。
3. 共享对象 mutation 的隐患: PR body 明确指出 `_resolve_nextn_quant_config` 的草案层改写会污染 caller 用于目标模型和 `lm_head` 的同一个 `QuarkConfig` ("the draft-only rewrite was mutating shared state"), 因此改为 wrapper 浅拷贝 + 内部 dict 深拷贝, 并用 `test_flag_on_does_not_mutate_caller_config` 固化。对 GLM-5.2-MXFP4 本身是 no-op (`layer_quant_config` 为空), 但对任何设置了该字段的 checkpoint 都会出错。
4. runner 是否能走 AITER 直接决定这一改动是优化还是回归: body 明确 "FP8 experts on the Triton path are slower than the bf16 they replace (TPOT p50 +7.5% to +11.6%)" , 所以若不修 AITER 分支, `SGLANG_GLM_NEXTN_MOE_PTPC=1` 反而会变慢; 并保留 `--moe-runner-backend triton` 作为回退开关。
5. CI 归因讨论: kzjeef 指出 AMD CI 失败在 `test/registered/attention/test_wave_attention_kernels.py` (exit code -6), 认为该用例在 main 上已被禁用、只是 rebase 要求全量跑; 作者补充 `test_basic_sanity.py::test_hellaswag_select` 因 `rowanz/hellaswag` 仓库下架导致 404, 已由 #39481 (切换到 Hugging Face mirror) 上游修复, 与本 PR 无关。

6. 1am9trash 的批准说明: "This is a re-submission of #38476. All the change is same. Only fix lint issue... Only blocker: need to pass CI task.", 表明技术结论已在原 PR 完成评审, 本 PR 的增量主要是 lint、单测与 model-local 重构。
- 把 GLM 专属的权重转换从 DeepSeek 基类中拆出 (model-local 重构) (design): 转换与工具函数最终下沉到 python/sglang/srt/models/glm4_moe.py 的 GlmMoeDsaForCausalLMNextN 内部, 不再污染 DeepSeek 基类; 但未新增独立的 glm5_next.py, PR body 中对 deepseek_weight_loader.py 的描述已过期。
 - 权重转换门控与模块 scheme 门控必须一致 (correctness): 抽出 should_apply_glm_nextn_moe_ptpc 同时供 load_weights 侧与 _resolve_nextn_quant_config 侧调用, 判定条件统一为“开关打开 + Quark + MTP fused experts 在 checkpoint 中为 bf16”。
 - 共享 QuarkConfig 被 _resolve_nextn_quant_config 就地改写 (correctness): 改为 copy.copy(quant_config) + copy.deepcopy(quant_config.quant_config), 并新增 test_flag_on_does_not_mutate_caller_config 回归用例把约束固化。
 - QuarkW8A8FP8MoE 的 AITER runner 因死标志永不可达 (performance): runner 改为按 kernel 真实前置条件选择 (AITER 可用 + per-channel 权重 + a2a 支持 + auto/aiter), 预重排绑定到所选 runner (is_shuffled tag), apply_weights 走 AiterMoeQuantInfo(PER_TOKEN) 并传 expert_mask; 新增 print_info_once 让回退可见, 保留 --moe-runner-backend triton 回退开关。
 - 默认关闭特性缺少开启态 CI 覆盖 (testing): 新增 test/registered/unit/models/test_glm_nextn_moe_ptpc.py (register_cpu_ci, est_time=5), 用 mock env 覆盖 ON 接线; 真实 GPU + MXFP4 checkpoint 的 serve+generate 场景仍未覆盖, 测试注释建议后续按 nightly 注册。
 - AMD CI 失败归因 (wave attention 用例与 hellaswag 下载源) (testing): 两个失败均判定为与本 PR 无关; hellaswag 问题由上游 #39481 切换到 Hugging Face mirror 修复, 本 PR 不再额外改动。

风险与影响

- 风险:
 1. 跨模型的量化后端默认值变更 (最高风险): QuarkW8A8FP8MoE.create_moe_runner 现在让 auto 在 ROCm+AITER+ 支持 a2a 的 per-channel 场景下解析为 AITER, 这不再只影响 GLM-5.2 NextN, 而是影响所有使用该 scheme 的 quark per-channel FP8 MoE 模型 (历史上全部静默跑 Triton)。换 kernel 意味着不同的数值实现与不同的权重布局要求 (is_shuffled), 虽然与同目录 quark_w4a4_mxfp4_moe.py、quark_w4a8_mxfp4_moe.py 的行为对齐, 但缺少本 PR 内的端到端回归验证。
- 1. AITER 分支的运行时依赖: apply_weights 的 AITER 路径直接读取 layer.dispatcher.expert_mask_gpu 并传入 AiterMoeQuantInfo。PR body 提到 block FP8 的 FLAT kernel "drops expert_mask, so it faults under EP", 说明该字段在 EP 下是硬依赖; 若某些部署路径未初始化 dispatcher 或 expert_mask_gpu 为 None, 这条新可达的分支可能暴露新的失败面。

2. 加载期开销与非 2D 断言: `_maybe_quant_glm_nextn_moe_to_ptpc` 对每个 expert 张量做 `to(torch.float32)` 再量化, 加载时会有一次性 CPU/ 内存放大 (单张量级别, 流式), 并有 `tensor.ndim != 2` 的硬 `ValueError`——若未来 checkpoint 层格式变化 (如打包权重), 会导致启动直接失败而非降级。
3. 共享配置改写虽已加拷贝, 但仍是深拷贝任意嵌套结构: `copy.deepcopy(quant_config.quant_config)` 对包含 tensor 或大型嵌套的配置可能带来额外开销与不可序列化风险; 当前 Quark 配置是字符串 / 字典级别, 风险低但值得留意。
4. 测试覆盖不足: 新增单测只覆盖 CPU 上的门控与 scheme 注入, 不覆盖真实 GPU 上的 AITER kernel 数值、TP/EP 分片后 `weight_scale` 的正确性、以及 prefill/decode 图模式下的行为; PR checklist 中 "Add unit tests" 未勾选, 作者也说明需要 ROCm GPU + AITER + MXFP4 checkpoint 才能端到端验证。
5. 文档与实现不一致: PR body 仍描述权重转换位于 `deepseek_weight_loader.py`, 但最终提交已改为 `model-local (glm4_moe.py)`, 后续维护者按 body 检索会找不到对应实现。
 - 影响: 对用户: 在 MI355X (TP4/EP4) 运行 GLM-5.2-MXFP4 + MTP 的推理服务, 开启 `SGLANG_GLM_NEXTN_MOE_PTPC=1` 后可获得交互性 p90 +9.3%、TPOT p50 -4.7%/p90 -8.5%、TTFT p50 -11.5%, 每 GPU NextN 权重从 5.46 GB 降到 3.17 GB (-2.29 GB), 使得更大 KV pool 或更高并发成为可能; 准确率与草稿接受长度在噪声范围内 (GSM8K 0.9416→0.9371, accept len 2.810→2.829)。由于开关默认关闭, 未显式开启的用户行为不变。

对系统: `QuarkW8A8FP8MoE` 的 runner 选择改变是全局性影响——所有 ROCm + AITER + per-channel 权重 + 支持 aiter 的 a2a 后端的 quark FP8 MoE 都会从 Triton 切到 AITER per-token CK fused MoE, 涉及权重预重排布局 (`is_shuffled`)、数值路径与显存占用形态 (`shuffle` 期间有 `empty_cache` 与临时副本), 属于跨模型的默认行为变更, 虽然方向上与同目录兄弟 scheme 对齐且启动日志可观测。

对团队: 本 PR 是 #38476 的重提交, 保留了原作者归属与讨论, 确立了两个可复用的工程范式: 一是“加载期量化转换 + 模块 scheme 同步改写”必须由同一个判定函数门控; 二是“runner 选择应由 kernel 的真实前置条件决定, 而不是某个语义无关的标志位”, 后者对后续清理其他量化 scheme 中的死分支有直接借鉴价值。本次同步补上了 CPU 单测, 弥补了原 PR "off by default 导致 CI 从不执行 ON 路径" 的空白。

- 风险标记: 默认关闭的新路径, 跨模型量化 runner 默认值变更, 共享配置对象副作用, 缺少 ROCm 端到端覆盖, PR 描述与最终实现不一致

关联脉络

- PR #38476 GLM-5.2 NextN 逐通道 FP8 草稿 MoE (本 PR 的原始提交, 作者 @fanxingran): 本 PR 是它的重提交, 四个原始 commit 保留原作者归属, 性能测量与评审讨论都留在原 PR; 1am9trash 在批准时明确 "This is a re-submission of #38476. All the change is same. Only fix lint issue."。
- PR #39481 将 hellaswag 相关测试与 benchmark 切换到 Hugging Face mirror: 作者在评论中说明本 PR 遇到的 `test_basic_sanity.py::test_hellaswag_select 404` 失败由该 PR 修复, 是本次 CI 归因的直接依据。

- PR #39631 [AMD] Prefer HIP Top-K for GLM-5.x on ROCm: 同为 AMD ROCm 上 GLM-5.x 的默认 kernel/ 后端选择调整, 与本 PR 修改 Quark MoE 默认 runner 的思路一致: 把平台默认值对齐到实际更优的实现, 并保留回退开关。
- PR #39613 Accept MXFP8 dispatch in FlashInfer A2A TRT-LLM MoE: 同样是 MoE 入口守卫条件写得过严 / 不匹配, 导致本应支持的 dispatch 类型无法进入对应 runner, 与本 PR 修复 apply_router_weight_on_input 死标志属于同一类缺陷模式。
- PR #39547 [AMD][bugfix] Fix DSV4 MTP crash: 同为 AMD 平台 MTP (NextN) 路径的正确性修复 (用 is_dspark 门控 D2H 优化), 与本 PR 一样在 AMD 上用门控条件保护 MTP 专属优化路径。
- PR #38160 [Feature] Support BF16 and batch-invariant inference with DeepEP v2: 同样涉及 MoE runner/ 后端与 dtype、量化信息 (quant info) 组合的兼容性边界, 是理解本 PR 中 AiterMoeQuantInfo 与 runner 选择语义的上下文补充。