PR #50912 完整报告

vllm-project/vllm

[Kimi K3 Perf] option to shard the shared expert for non mega case, 16.98 GiB memory/GPU saved

合并时间：2026-08-05 02:00

原文链接：http://prhub.com.cn/vllm-project/vllm/pull/50912

执行摘要

- 一句话：新增非 MegaMoE 共享专家分片选项，单卡省 16.98 GiB 显存
- 推荐动作：值得精读。核心看点： 1) 用删除参数的方式把 " 场景是否可分片 " 从模型代码中移除，分片决策收敛到 env + 维度整除性，接口更简洁； 2) _disable_shared_experts_overlap 通过 getattr 隐性契约耦合模型层与 FusedMoE runner，是一个值得记录的设计约定； 3) PR body 中的 benchmark 脚本（CUDA graph + max-rank 同步 + 多 token 规模扫描）可作为模型级性能评估模板。使用时需明确边界：仅建议 PD 分离的 decode 节点 + 小 batch 场景开启。

功能与动机

PR body 明确声明这是 https://github.com/vllm-project/vllm/pull/50656 的 follow-up：base 代码中 shared experts 构造传入 can_shard_sequence_parallel=self.use_mega_moe，分片仅限 MegaMoE 路径，非 MegaMoE 的 FusedMoE 路径下 shared expert 权重在每张卡上整份复制。本 PR 的目标是让非 MegaMoE 场景也能按 VLLM_KIMI_K3_SHARD_SP_SHARED_EXPERT 选择分片，以显著降低显存占用（body 数据：TP=4 下单卡省 16.98 GiB）。body 同时说明安全性来源与使用边界："The update is safe as we make the shared experts overlap off, this might be a future optimization point"、"Recommend to use with PD disaggregation (decode node)"。

实现拆解

1. 精简分片门控（vllm/models/kimi_k3/nvidia/model.py）：shard_sequence_parallel_mlp() 删除 eligible 参数，KimiMLP.__init__ 删除 can_shard_sequence_parallel 参数。分片判定简化为 "env 开关 + use_sequence_parallel + 维度整除 TP" 三要素；shared experts 构造不再以 use_mega_moe 作为可分片前置条件，非 MoE dense MLP 也沿用同一门控。这属于接口级 " 数据契约 " 变更，调用方（模型构造、测试）需同步适配。
2. FusedMoE runner 安全兜底（vllm/model_executor/layers/fused_moe/runner/shared_experts.py）：_disable_shared_experts_overlap 属性新增 getattr(self._layer, " shard_sequence_parallel", False) 检查，命中即返回 True，令 _determine_shared_experts_order() 走 NO_OVERLAP 分支。原因：分片后的 shared expert 前向需要 all-gather/reduce-scatter 集合通信，不能像复制版那样放到独立 aux 流

上与 routed expert 重叠；作者留 TODO 标记未来优化点。

3. 测试配套（tests/models/kimi_k3/test_sequence_parallel.py）：test_shard_sequence_parallel_mlp_gating 参数化从 5 元组改为 4 元组，去掉 eligible 维度，保留 " 默认关闭（opt-in only）"" 非 SP 不复制 ""TP=1 无分片 ""6144 % 5 不可整除 " 等关键边界；test_sharded_sequence_parallel_mlp_matches_replicated 继续验证分片实现（all-gather → 部分和 → reduce-scatter）与复制实现的数值等价性。
4. 演进过程：3 个 commit 依次为功能实现（"non mega shared experts"）、补 TODO、删除冗余参数（"delete can_shard_sequence_parallel"，回应 reviewer 意见）。改动集中、无返工，benchmark 脚本以 AI-generated script 形式附在 body 中供复现。

关键文件：

- vllm/models/kimi_k3/nvidia/model.py（模块 模型层；类别 source；类型 core-logic；符号 shard_sequence_parallel_mlp, KimiMLP.init）：核心变更文件：删除 eligible/can_shard_sequence_parallel 门控，让非 MegaMoE 的 FusedMoE 路径也能按 VLLM_KIMI_K3_SHARD_SP_SHARED_EXPERT 对 shared expert 做 TP 分片，92 层共省 16.98 GiB/GPU。
- vllm/model_executor/layers/fused_moe/runner/shared_experts.py（模块 共享专家；类别 source；类型 core-logic；符号 _disable_shared_experts_overlap）：FusedMoE runner 侧的安全兜底：分片场景下强制关闭 shared experts 的独立 CUDA 流重叠，避免集合通信与 routed expert 竞争导致正确性问题。
- tests/models/kimi_k3/test_sequence_parallel.py（模块 序列并行；类别 test；类型 test-coverage；符号 test_shard_sequence_parallel_mlp_gating, test_sharded_sequence_parallel_mlp_matches_replicated）：测试配套：同步去掉 eligible 参数维度，保留 opt-in、TP=1、不可整除等门控边界用例，并继续维护分片与复制的数值等价性验证。

关键符号：shard_sequence_parallel_mlp, KimiMLP.init, _disable_shared_experts_overlap, test_shard_sequence_parallel_mlp_gating

关键源码片段

vllm/models/kimi_k3/nvidia/model.py

核心变更文件：删除 eligible/can_shard_sequence_parallel 门控，让非 MegaMoE 的 FusedMoE 路径也能按 VLLM_KIMI_K3_SHARD_SP_SHARED_EXPERT 对 shared expert 做 TP 分片，92 层共省 16.98 GiB/GPU。

```
# 是否对序列并行 MLP 做 TP 分片而非整权复制。 # 分片是 opt-in 行为：必须显式设置 VLLM_KIMI_K3_SHARD_SP_SHARED_EXPERT = 1， # 默认关闭时保持原有复制（replicated）布局，行为与旧版一致。 defshard_sequence_parallel_mlp(     hidden_size: int,     intermediate_size: int,     use_sequence_parallel: bool, ) -> bool: """Whether to TP-shard a sequence-parallel MLP instead of replicating it. Opt-in via ``VLLM_KIMI_K3_SHARD_SP_SHARED_EXPERT``; see :class:`KimiMLP` for the trade-off. """     enabled = envs.VLLM_KIMI_K3_SHARD_SP_SHARED_EXPERT     if not (use_sequence_parallel and enabled):         return False     tp_size =
```

```
get_tensor_model_parallel_world_size()     # 维度必须能被 TP 整除，否则后续的 divide() 会直接报错；      # 这正是 PR body 表格中 TP=5 一类场景不生效的原因。      return (  tp_size > 1         and intermediate_size % tp_size == 0         and hidden_size % tp_size == 0     ) classKimiMLP(nn.Module): """Dense / shared-expert MLP, optionally TP-sharded under sequence parallel."""     def__init__(         self,         hidden_size: int,         intermediate_size: int,         hidden_act: str,         quant_config: QuantizationConfig | None = None,         reduce_results: bool = True,         use_sequence_parallel: bool = False,         prefix: str = "",         activation_situ_beta: float | None = None,         activation_situ_linear_beta: float | None = None,     ) -> None:         super().__init__()         # 本 PR 的核心变化：不再由调用方通过 can_shard_sequence_parallel 声明         # 是否可分片（旧代码 shared experts 只允许 MegaMoE 路径分片），         # 现在只需 env 开关 + SP + 维度整除性即可判定，非 MegaMoE 同样生效。         self.shard_sequence_parallel = shard_sequence_parallel_mlp(             hidden_size,             intermediate_size,             use_sequence_parallel,         )         replicate = use_sequence_parallel and not self.shard_sequence_parallel         self.gate_up_proj = MergedColumnParallelLinear(             hidden_size,             [intermediate_size] * 2,             bias=False,             quant_config=quant_config,             disable_tp=replicate,             prefix=f"{prefix}.gate_up_proj",         )         self.down_proj = RowParallelLinear(             intermediate_size,             hidden_size,             bias=False,             quant_config=quant_config,             # 分片场景下 reduce 由 forward() 末尾的 reduce-scatter 完成，             # 该集合通信会同时把序列分片还原，因此这里必须关闭 reduce_results。             reduce_results=False if self.shard_sequence_parallel else reduce_results,             disable_tp=replicate,             prefix=f"{prefix}.down_proj",         )         # shared experts：MegaMoE 与 FusedMoE 路径统一走这里，         # 不再区分 use_mega_moe，分片与否完全交由上面的门控决定。         # FusedMoE runner 侧由 _disable_shared_experts_overlap 负责关闭         # aux 流重叠，避免分片后的集合通信与 routed expert 竞争。         if self.num_shared_experts is not None:             shared_intermediate_size = moe_intermediate_size * self.num_shared_experts             self.shared_experts = KimiMLP(                 hidden_size=config.hidden_size,                 intermediate_size=shared_intermediate_size,                 hidden_act=config.hidden_act,                 quant_config=quant_config,                 reduce_results=False,                 use_sequence_parallel=use_sequence_parallel,                 prefix=f"{prefix}.shared_experts",                 activation_situ_beta=activation_situ_beta,                 activation_situ_linear_beta=activation_situ_linear_beta,             )         else:             self.shared_experts = None
```

vllm/model_executor/layers/fused_moe/runner/shared_experts.py

FusedMoE runner 侧的安全兜底：分片场景下强制关闭 shared experts 的独立 CUDA 流重叠，避免集合通信与 routed expert 竞争导致正确性问题。

```
    @property
    def _disable_shared_experts_overlap(self) -> bool:
```

```
        # 新增分支：layer 一旦被标记为 shard_sequence_parallel（如 Kimi K3
        # 分片后的 shared expert），其前向需要 all-gather / reduce-scatter
        # 集合通信，无法像复制版那样放到独立 aux 流上与 routed expert 重叠，
        # 因此直接返回 True 禁用重叠以保证正确性。
        # TODO: 后续可以进一步优化，把分片 shared expert 也纳入重叠调度。
        if getattr(self._layer, "shard_sequence_parallel", False):
            return True
        # 原有禁用条件保持不变：
        # - eplb 使用非默认 all2all 后端时存在正确性问题；
        # - flashinfer 双面 kernel 在 DP 下没有重叠收益。
        parallel_config = self._moe_config.moe_parallel_config
        return (
            parallel_config.enable_eplb
            and parallel_config.all2all_backend != "allgather_reducescatter"
        ) or parallel_config.use_fi_nvl_two_sided_kernels
```

tests/models/kimi_k3/test_sequence_parallel.py

测试配套：同步去掉 eligible 参数维度，保留 opt-in、TP=1、不可整除等门控边界用例，并继续维护分片与复制的数值等价性验证。

```
@pytest.mark.parametrize(
    ("enabled", "use_sequence_parallel", "tp_size", "expected"),
    [
        (True, True, 8, True),
        (False, True, 8, False), # opt-in only: 默认必须显式开启
        (True, False, 8, False), # 复制布局只存在于 SP 开启时
        (True, True, 1, False), # TP=1 时没有分片可言
        (True, True, 5, False), # 6144 % 5 != 0，分片会触发 divide() 失败
    ],
)
def test_shard_sequence_parallel_mlp_gating(
    monkeypatch,
    enabled: bool,
    use_sequence_parallel: bool,
    tp_size: int,
    expected: bool,
):
    # 通过 monkeypatch 注入 env 开关与 TP 大小，验证门控函数的每种组合；
    # 旧签名中的 eligible 参数已随 PR 删除，这里只保留实际生效的输入。
    monkeypatch.setattr(kimi_model.envs, "VLLM_KIMI_K3_SHARD_SP_SHARED_EXPERT",
    enabled)
    monkeypatch.setattr(
        kimi_model, "get_tensor_model_parallel_world_size", lambda: tp_size
    )
    assert (
        kimi_model.shard_sequence_parallel_mlp(
            hidden_size=7168,
            intermediate_size=6144,
            use_sequence_parallel=use_sequence_parallel,
```

```
        )
        is expected
    )
```

评论区精华

核心讨论围绕中间版本遗留的死参数展开：reviewer tlrmchlsmth 在 shared experts 构造处（当时 can_shard_sequence_parallel 已从 self.use_mega_moe 改为 True）提问 "should we remove this argument now?"，作者 yewentao256 回复 "Nice catch, fixed!"，并在最终提交中删除该参数及相关注释，随后 tlrmchlsmth 给出 APPROVED。claude[bot] 仅提示仓库配置为手动 review，无实质技术内容。此外 body 中的设计取舍值得记录：为安全直接关闭 shared experts 重叠，并明确把它列为 future optimization point。

- can_shard_sequence_parallel 参数是否应删除 (design): 作者 yewentao256 回复 "Nice catch, fixed!"，在最终提交 "delete can_shard_sequence_parallel" 中删除参数及相关注释，随后 tlrmchlsmth 给出 APPROVED。

风险与影响

- 风险：
  - 开启时的性能回退：shard 模式在 global tokens ⩾ 256 后显著变慢（256/512/1024/2048 分别 +10.0%/+30.2%/+75.2%/+135.3%），长 prefill 或大 batch 场景不应开启；由于 env 默认关闭，回归面仅限显式 opt-in 的用户。
  - 正确性依赖隐性契约：_disable_shared_experts_overlap 通过 getattr(self._layer, "shard_sequence_parallel", False) 与模型层耦合，其他模型若意外使用同名属性会静默进入禁用重叠分支；当前仅 Kimi K3 使用，影响面受限但值得留意。
  - 接口兼容性：KimiMLP.__init__ 与 shard_sequence_parallel_mlp 的公开签名变更，外部调用（脚本、下游测试）需同步更新；KimiMLP 属 Kimi K3 内部实现，风险可控。
  - PP>1 约束未落代码：body 表格声明 PP>1 不启用，但 PR 代码中未见 PP 维度显式检查，该约束可能来自上游 #50656 的其他逻辑，建议后续确认。
  - 收益面：单卡 16.98 GiB 显存节省对长上下文 decode 节点价值明显，且小 batch decode 延迟同步下降 14%～39%。
  - 影响：影响范围仅作用于 Kimi K3 模型且为 env 显式 opt-in，默认配置下行为完全不变。对开启用户，TP=4 时单卡显存节省 16.98 GiB（92 层，BF16），decode 小 batch 延迟下降约 14%～39%，但大 batch 变慢；显存释放对 PD 分离 decode 节点部署尤其有价值。对团队而言，该改动示范了一条 " 模型层可分片 + runner 层关重叠 " 的组合模式，可复用到其他 MoE 模型；同时删除冗余参数降低了 Kimi K3 相关代码的维护成本。影响程度：中（单模型、opt-in、无默认行为变化）。
  - 风险标记：大 batch 性能回退，默认关闭的 opt-in 开关，runner 与模型层隐性契约，共享专家重叠被禁用，公开参数删除需同步调用方

关联脉络

- PR #50656 前序 PR：Kimi K3 MegaMoE shared expert 分片支持（PR body 引用）：本 PR 的直接前序，PR body 开头声明 "A following up PR for #50656"；base 代码中的 env 开关与 can_shard_sequence_parallel=self.use_mega_moe 即来自该 PR，本 PR 把分片机制推广到非 MegaMoE 路径。
- PR #50886 [Bugfix][Reasoning] kimi_k3: O(delta) reasoning-end check on the decode path: 同属 Kimi K3 decode 路径优化线，消除长上下文 GPU 空转，与本 PR " 配合 PD 分离 decode 节点使用 " 的建议互补，共同降低 decode 节点空转与显存压力。
- PR #50567 [Bugfix][Kimi-K3] Enforce packed rows and op availability in AttnRes dispatch: 同一 vllm/models/kimi_k3/nvidia/ 目录下的模型实现加固，共用 KimiK3 模型文件家族，后续改动需对其一并回归。