PR #32115 完整报告

sgl-project/sglang

[MLX] Size request capacity by attention DP

合并时间：2026-07-30 09:18

原文链接：http://prhub.com.cn/sgl-project/sglang/pull/32115

执行摘要

- 一句话：MLX 请求容量按 attention DP 大小分配，修复纯 DP 副本容量不足
- 推荐动作：建议：MLX 后端开发者和 CI 维护者审查此 PR，确保后续 #32101、#32102 可以无缝集成。测试覆盖较全，可放心合并。注意 @noob-se7en 正在重构 model_runner_stub，未来可能再次调整此逻辑。

功能与动机

作为 #32101 和 #32102（Gemma 4 Apple Silicon MVP）的前置条件。MLX stub 之前将 max_running_requests 除以系统 dp_size，在启用 DP attention 时正确（attn_dp_size == dp_size），但在纯 DP 场景下会导致每个 replica 分配到的容量不足，因为 attn_dp_size == 1 时每个 replica 应保留全量配置限制。需要与 KVCacheConfigurator.resolve_max_num_reqs() 的行为一致，即按 ps.attn_dp_size 划分请求容量。

实现拆解

1. 修改 _resolve_max_running_requests 方法：将请求容量划分依据从 self.dp_size 改为 self.ps.attn_dp_size。纯 DP（attn_dp_size=1）时每个 replica 保留完整配置限制；DP attention（attn_dp_size>1）时按 attention-DP 所有者数量分割。
2. 新增 _explicit_aux_state_size_per_worker 方法：将全局 max_mamba_cache_size 按 ps.attn_dp_size 分割为 per-worker 值，并在 _resolve_max_running_requests 中替代直接使用全局值。
3. 更新 initialize() 方法：在构造 MlxAuxiliaryStateReqToTokenPool 时使用 per-worker 辅助状态容量。
4. 改进错误诊断：保留全局 max_mamba_cache_size 用于错误消息，推荐最小全局值为 ratio * ps.attn_dp_size，并显示 per-worker 辅助状态 cap。
5. 新增测试文件 test_attn_dp_request_capacity.py，注册 MLX 和 CPU CI，包含 4 个测试用例：纯 DP 保留全量、DP attention 分割、混合 attention-DP 分割辅助状态、辅助状态错误报告使用全局单位。

关键文件：

- python/sglang/srt/hardware_backend/mlx/model_runner_stub.py（模块 MLX 后端；类别 source；类型 core-logic；符号 _explicit_aux_state_size_per_worker, _resolve_max_running_requests, initialize）：核心逻辑变更：修改请求容量和辅助状态容

量的分割方式，按 attention DP 大小而非全局 DP 大小划分

- test/registered/unit/hardware_backend/mlx/test_attn_dp_request_capacity.py（模块 MLX 测试；类别 test；类型 test-coverage；符号 _arch, _stub_for_initialize, _initialize_stub, TestAttentionDpRequestCapacity）：新增全覆盖测试：验证纯 DP、DP attention、混合 auxiliary-state 分片、全局单位诊断

关键符号：_explicit_aux_state_size_per_worker, _resolve_max_running_requests, initialize

关键源码片段

python/sglang/srt/hardware_backend/mlx/model_runner_stub.py

核心逻辑变更：修改请求容量和辅助状态容量的分割方式，按 attention DP 大小而非全局 DP 大小划分

```
def _explicit_aux_state_size_per_worker(self) -> int | None:
    '''Return the explicit auxiliary-state cap for this attention-DP owner.'''
    aux_state_size = self.server_args.max_mamba_cache_size
    if aux_state_size is None:
        return None
    # 按 attention-DP 大小分割全局容量，使每个所有者获得相等份额
    return aux_state_size // self.ps.attn_dp_size

def _resolve_max_running_requests(self) -> int:
    '''Concurrency cap handed to the scheduler.

    核心变更：将 `requested` 除以 `self.ps.attn_dp_size` 而非 `self.dp_size`，
    使得纯 DP（attn_dp_size=1）时保留完整限制，DP attention 时按注意力 DP 所有者分割。
    '''
    capacity_cap = self.max_total_num_tokens // 2
    requested = self.server_args.max_running_requests
    if requested is None:
        requested_per_worker = None
        resolved = min(capacity_cap, 4096)
    else:
        requested_per_worker = requested // self.ps.attn_dp_size
        resolved = min(requested_per_worker, capacity_cap)

    aux_state_size = self._explicit_aux_state_size_per_worker()
    if (
        mambaish_config(self.model_config) is not None
        and aux_state_size is not None
    ):
        ratio = self._aux_state_slots_per_request()
        resolved = min(resolved, aux_state_size // ratio)
        if resolved <= 0:
            global_aux_state_size = self.server_args.max_mamba_cache_size
            min_global = ratio * self.ps.attn_dp_size
```

```
            raise RuntimeError(
                'MLX auxiliary-state cache is too small to serve any '
                'requests: max_mamba_cache_size={} '
                'backs only {} concurrent requests '
                'per attention-DP worker (per-worker cap={}, '
                '{} slots per request). Increase --max-mamba-cache-size '
                'to at least {}.'.format(
                    global_aux_state_size,
                    aux_state_size // ratio,
                    aux_state_size,
                    ratio,
                    min_global,
                )
            )
        if requested_per_worker is not None and resolved < requested_per_worker:
            logger.warning(
                'max_running_requests reduced from %d to %d '
                '(per attention-DP worker) due to KV cache or auxiliary-state capacity.',
                requested_per_worker, resolved,
            )
        return resolved
```

评论区精华

Review 中 yeahdongcn 提出了几项关键改进：要求同步分割 max_mamba_cache_size（已实现）；建议错误消息保持全局 CLI 单位（已实现）；要求测试通过 initialize() 初始化且使用 CustomTestCase（已实现）。jlee5814 也指出了 fixture 设置方式可能隐藏错误。整体上讨论聚焦于保持与基类解析器行为一致和测试充分性。

- 同步分割辅助状态容量 (correctness): 作者在 9c7e71c6e 中实现分割，使用 _explicit_aux_state_size_per_worker 方法。
- 错误消息使用全局 CLI 单位 (correctness): 作者在 ebb4e9cb6 中实现，错误消息使用全局 max_mamba_cache_size，显示 per-worker cap，推荐 ratio * ps.attn_dp_size。
- 测试 fixture 应通过 initialize() 初始化 (testing): 作者重写测试 fixture 使用 ParallelState.trivial 并调用 initialize()，添加了 hybrid 测试用例。
- 测试类应继承 CustomTestCase (style): 作者将继承从 unittest.TestCase 改为 sglang.test.test_utils.CustomTestCase。

风险与影响

- 风险：风险：仅影响 MLX 后端；容量计算变更可能改变现有部署中请求限制的实际值，需确保与基类解析器行为一致；错误消息变更可能影响用户排查；对非 MLX 后端无影响。测试覆盖了主要场景，但实际部署验证仍有必要。
- 影响：影响：MLX 后端用户（Apple Silicon）；纯 DP 部署现在能正确分配请求容量，DP attention 部署行为不变；hybrid 模型（如 Mamba）的辅助状态容量正确分割；错误消息提供更准确的全局建议值，方便用户调整参数。

- 风险标记：MLX 后端容量分配变更，测试覆盖新行为

关联脉络

- PR #32101 Prerequisite for Gemma 4 MLX: 本 PR 是 #32101 的前置条件，修复请求容量分配问题。
- PR #32102 Gemma 4 Apple Silicon MVP: 本 PR 修复的问题是 Gemma 4 Apple Silicon MVP 的前置条件。
- PR #30547 Previous MLX test suite regression: Review 中引用此 PR 的测试缺口，提示注意 fixture 设置方式。