PR #36198 完整报告

sgl-project/sglang

[Weight Cache] Enhance test and support EPLB

合并时间：2026-08-27 14:11

原文链接：http://prhub.com.cn/sgl-project/sglang/pull/36198

执行摘要

- 一句话：weight cache 放开 EPLB 限制并补齐 DP/EP 测试
- 推荐动作：值得精读。核心看点是 daemon 与引擎通过 compute_initial_expert_location_metadata 共享同一 EPLB 布局初始化入口，以及测试基类用类属性参数化 daemon/server 参数的模式。关注 weight cache 功能线的读者建议结合 #33684 和 #34053 一起阅读，理解从 TP-only 到 DP/EP/EPLB 全覆盖的演进路径。

功能与动机

PR body 说明这是 #33684 的 follow-up：此前 weight cache daemon 没有覆盖 DP 和 EP 并行布局的测试，且 server_args 直接拒绝 --weight-cache-mode 与 --enable-eplb 的组合。本 PR 一方面补齐 DP/EP layout 的测试，另一方面解除 EPLB 封锁，让 MoE 模型在 EP + EPLB 场景下也能复用跨进程权重缓存。

实现拆解

1. 解除参数互斥：server_args.py::_handle_load_format 删除对 weight_cache_mode != "off" && enable_eplb 的 ValueError 拒绝分支，使 daemon/client 模式与 EPLB 组合成为合法配置。
2. daemon 初始化 EPLB 元数据：weight_cache/daemon.py::load 在 _init_distributed（并行组就绪）之后调用新增的 _initialize_eplb_expert_location_metadata(model_config)；该方法在 enable_eplb 开启时，通过 compute_initial_expert_location_metadata 与 set_global_expert_location_metadata 构建与引擎相同的初始专家物理布局，并传入 get_parallel().moe_ep_rank。
3. 测试基类参数化：TestWeightCacheDaemonTP2 增加 model_override、daemon_args、server_args 类属性，setUpClass 在启动 daemon 与 client server 时透传这些参数，复用现有 IPC 加载夹具。
4. 新增 E2E 测试类：TestWeightCacheDaemonQwen3MoeDP 覆盖 --dp 2 --enable-dp-attention --enable-dp-lm-head；TestWeightCacheDaemonQwen3MoeEP 改用 DEFAULT_TARGET_MODEL_EAGLE_DP_ATTN 覆盖 --ep-size 2 --enable-eplb --ep-num-redundant-experts 2。
5. 协议与 CI 配套：test_weight_cache_protocol.py 的 spawn 转发断言补上 enable_eplb 与 ep_num_redundant_experts 字段；test_weight_cache_daemon.py 的 est_time 依实

测数据从 100 上调至 280 秒。

关键文件：

- python/sglang/srt/weight_cache/daemon.py（模块 权重缓存；类别 source；类型 core-logic；符号 _initialize_eplb_expert_location_metadata）：核心逻辑变更：daemon 启动时新增 EPLB 专家布局元数据初始化，确保与引擎侧布局一致，这是放开 EPLB 组合的关键支撑。
- test/registered/model_loading/test_weight_cache_daemon.py（模块 权重缓存；类别 test；类型 test-coverage；符号 TestWeightCacheDaemonQwen3MoeDP, TestWeightCacheDaemonQwen3MoeEP）：测试主文件：基类参数化并新增 DP/EP 两组端到端测试，覆盖本 PR 放开的所有新组合，同时更新 CI 时间估算。
- python/sglang/srt/server_args.py（模块 服务参数；类别 source；类型 core-logic）：删除 weight_cache 与 EPLB 的互斥校验，是本次功能放开的入口变更。
- test/registered/unit/server_args/test_server_args.py（模块 服务参数；类别 test；类型 test-coverage；符号 test_weight_cache_daemon_allows_static_eplb）：新增 CPU 单元测试，验证 weight_cache_mode='daemon' 与 enable_eplb=True 能通过参数校验，防止未来回归到互斥拒绝。
- test/registered/unit/model_loader/test_weight_cache_protocol.py（模块 权重缓存；类别 test；类型 test-coverage；符号 test_spawn_forwards_complete_server_args_without_projection）：补充 daemon spawn 参数转发测试中的 EPLB 字段，确保新增配置能透传到子进程。

关键符号：_initialize_eplb_expert_location_metadata, TestWeightCacheDaemonQwen3MoeDP, TestWeightCacheDaemonQwen3MoeEP, test_weight_cache_daemon_allows_static_eplb

关键源码片段

python/sglang/srt/weight_cache/daemon.py

核心逻辑变更：daemon 启动时新增 EPLB 专家布局元数据初始化，确保与引擎侧布局一致，这是放开 EPLB 组合的关键支撑。

```
# load() 中位于 self._init_distributed(server_args, model_config) 之后调用，
# 此时 moe_ep_rank 等并行组信息已就绪，可安全构建 EPLB 布局
def _initialize_eplb_expert_location_metadata(self, model_config) -> None:
    """让守护进程与引擎构建出相同的初始专家物理布局。"""

    # 未开启 EPLB 时无需初始化，保持原有行为
    if not self.server_args.enable_eplb:
        return

    # 延迟导入，避免增加守护进程冷启动的 import 开销
    from sglang.srt.eplb.expert_location import (
        compute_initial_expert_location_metadata,
        set_global_expert_location_metadata,
    )
```

```
    # 与引擎侧（model_runner）走同一入口，确保 daemon 导出的权重
    # 在 EPLB 模式下与 client 端加载时看到的专家分片一致
    set_global_expert_location_metadata(
        compute_initial_expert_location_metadata(
            model_config=model_config,
            moe_ep_rank=get_parallel().moe_ep_rank,
        )
    )
```

test/registered/model_loading/test_weight_cache_daemon.py

测试主文件：基类参数化并新增 DP/EP 两组端到端测试，覆盖本 PR 放开的所有新组合，同时更新 CI 时间估算。

```
class TestWeightCacheDaemonQwen3MoeDP(TestWeightCacheDaemonTP2):
    """Qwen3 开启静态 attention DP，验证 daemon 与 client 的 DP 布局一致。"""

    daemon_args = [
        "--dp", "2",
        "--ep-size", "1",
        "--enable-dp-attention",
        "--enable-dp-lm-head",
        "--random-seed", "42",
    ]
    # client 侧 server 需要与 daemon 使用相同的并行参数，
    # 否则 IPC 导出的权重分片与 engine 布局不匹配
    server_args = daemon_args[:]


class TestWeightCacheDaemonQwen3MoeEP(TestWeightCacheDaemonTP2):
    """Qwen3 MoE 开启静态 EP 与 EPLB，覆盖本 PR 放开的新组合。"""

    model_override = DEFAULT_TARGET_MODEL_EAGLE_DP_ATTN
    daemon_args = [
        "--dp", "1",
        "--ep-size", "2",
        "--enable-eplb",
        "--ep-num-redundant-experts", "2",
        "--random-seed", "42",
    ]
    server_args = daemon_args[:]
```

评论区精华

review 讨论集中在 CI 时间估算上：liusy58 询问 est_time 如何计算并建议持续监控，UNIDY2002 给出了三个实测数据点（280s、277s、292s），最终把 est_time 定为 280 并承诺后续再校准。此外 liusy58 提出应维护一张汇总表，追踪 weight cache daemon 测试已覆盖的参数组合，该建议尚未落地。

- est_time 估算依据 (testing): 以实测数据为准，est_time 定为 280，后续继续监控并按需校准。
- 测试覆盖矩阵跟踪 (testing): 建议已提出，本 PR 未落地，作为后续跟踪事项。

风险与影响

- 风险：
  1. 新增组合缺乏运行时校验：移除 server_args 互斥校验后，weight_cache_mode 与 enable_eplb 同时开启变为合法；若 daemon 与 engine 的 EPLB 初始化参数（如 moe_ep_rank、ep_num_redundant_experts）不一致，可能导致专家权重分片错位，出现加载错误或推理结果错误。目前测试仅覆盖固定 seed 与单一模型，组合空间较大。
  2. daemon 启动路径变更：daemon.py 的 load() 在每次启动时新增 EPLB 元数据计算与全局状态设置，且函数内延迟导入 sglang.srt.eplb.expert_location，若该模块在 daemon 环境有隐式副作用会增加启动成本。
  3. CI 时长显著增加：est_time 从 100 上调到 280 秒，2-gpu-large runner 上实测约 280-292 秒，接近估算上限，可能影响队列周转。
  4. 测试依赖外部模型下载：EP 测试类使用 DEFAULT_TARGET_MODEL_EAGLE_DP_ATTN，网络波动会导致 CI 不稳定（历史上曾出现 2-gpu-h100 失败后重跑才通过）。 - 影响：功能上，--weight-cache-mode daemon/client 与 --enable-eplb 的组合从 " 拒绝启动 " 变为可用，MoE 模型在专家并行 + 负载均衡场景下可复用跨进程权重缓存；静态 DP（attention DP + lm-head DP）场景也获得了端到端验证。性能上，推理路径无变化（PR 声明 N/A），仅 daemon 启动时多一次元数据计算。团队侧，测试基类参数化降低了后续新增并行布局组合的测试成本，但 2-gpu CI 资源占用时间增加近 3 倍。 - 风险标记：核心路径变更（daemon 启动流程），新增参数组合缺运行时校验，CI 时长显著增加，测试依赖外部模型下载

关联脉络

- PR #34053 [Fix] Account resident weight memory in KV sizing: 同属 weight cache 功能线，改动了 weight_cache/daemon.py 与协议相关文件，本 PR 在其基础上扩展 DP/EP/EPLB 支持。
- PR #36586 [Core] Refactor server argument choices: 重构了 server_args.py 参数解析，本 PR 合入时解决了 self.enable_eplb 到 cfg.enable_eplb 的冲突。