PR #33170 完整报告

sgl-project/sglang

config: route parallel config-leaf reads through get_parallel()

合并时间：2026-08-01 23:58

原文链接：http://prhub.com.cn/sgl-project/sglang/pull/33170

执行摘要

- 一句话：并行配置读取全量迁至 get_parallel()，弹性 EP 写路径改走 override
- 推荐动作：值得精读。三个设计决策值得学习：(1) 访问器慢路径保持 dynamo 可追踪——graph-break 是编译路径上的隐性炸弹，用 _fields 字典判断和槽位直读替代 getattribute；(2) live-shadowed 拓扑尺寸原则——活属性优先于 bag，避免 " 同一名字两个值 " 的歧义；(3) ratchet 机制让跨 80+ 文件的迁移可以被机械验证。对要扩展 SGLang 配置体系、或编写 torch.compile 兼容代码的工程师都有参考价值。

功能与动机

PR body 明确指出："The parallel namespace was the last reader family left on get_server_args(): 106 config-leaf reads ... still bypassed the published config bags"。在已发布的配置 bag 模型下，直读 server_args 会绕过被 override 过的已解析配置，造成读写两侧分离的 desync——这正是 #33168 暴露的 chunked-prefix gate 问题类型（写入落在无人读取的实例上）。此外，原先的写路径 server_args.override 只改 pristine 实例，而读取方已在迁移到 bag，弹性 EP 扩容后的新值无法被读取方看到。

实现拆解

1. 访问器核心改造（python/sglang/srt/runtime_context.py）

重构 ParallelContext.getattr：原实现用 object.getattribute(" _config ") 防御式取槽位，这在 torch.compile 编译路径下会 graph-break；而 enable_moe_dense_fully_dp() 等 gate helper 运行在编译后的模型 forward 里。新实现直接访问 slots__槽位 self._config，对下划线属性直接抛 AttributeError（同时打断 pickle/copy 协议在 __init__前探测槽位导致的递归），并用nameinconfig._fields（普通__dict__条目）替代_ConfigBag.__contains（不可追踪）。配套 fullgraph 回归测试 test_parallel_config_leaves_trace_under_torch_compile 钉住该模式。

2. 106 处读取翻转（跨 81 文件）

按家族分布：

- PP 调度循环（managers/scheduler_pp_mixin.py）：pp_async_batch_depth、enable_dsa_prefill_context_parallel 等 9 处切换；

- 弹性 EP 读取（model_executor/model_runner.py）：ep_join_rank_offset、elastic_ep_initial_size、enable_dp_attention、dwdp_size、dcp_replicate_q_proj 等；
- gather 模式判定（utils/common.py）：require_mlp_tp_gather / require_attn_tp_gather / require_mlp_sync 的 dp_size、enable_dp_attention、moe_dense_tp_size、enable_dp_lm_head、disable_attn_tp_gather、elastic_ep_backend 全部改读 bag；
- 专家放置（eplb/expert_location.py）：ep_join_mode、moe_a2a_backend、ep_num_redundant_experts、ep_dispatch_algorithm 等从 get_exec().moe 读取；
- PD 连接（disaggregation/common/conn.py）：dp_size、load_balance_method、enable_dsa_cache_layer_split、nnodes，含 bootstrap_room % dp_size 的 prefill DP 路由校验；
- 投机解码（speculative/dspark_components/dspark_worker_v2.py）：enable_dp_attention、speculative_eagle_topk、mamba_track_interval；
- 一批 MoE 模型文件（deepseek_v4、glm4_moe、qwen3_moe、sdar、bailing_moe 等，各 +2/-8）：enable_dp_lm_head 从 get_server_args() 改为 get_parallel()；
- 模型加载（model_loader/loader.py）：weight-info 字典的 moe_dense_tp_size、enable_dp_lm_head 改读 parallel 访问器。 刻意保留 5 个 live-shadowed 拓扑尺寸（tp/pp/dcp/attn_cp/moe_dp_size）继续读 server_args：live @property 在访问器上优先，条件初始化 group 在无条件调用点会 fail-loud。

3. 弹性 EP 写路径重路由（model_runner.py）

_initialize_elastic_ep_joiner、_expand_eplb_metadata_for_scale、_finalize_scale_up 中的 4 处 server_args.override("elastic_ep.scale_join"/"elastic_ep.scale", ...) 改为 get_context().override(...)，使扩容后的新值直接落进已发布 bag，与读取侧同源；配套把对应实例读取（ep_join_rank_offset、elastic_ep_initial_size、expert_location gpus-per-node 路径）一并切到 bag。ServerArgs.override 调用点 ratchet 从 38 降至 34。

4. 专家放置辅助函数去参数化 + 测试联动（python/sglang/srt/eplb/expert_location.py、test/registered/unit/eplb/test_compute_logical_to_rank_dispatch_physical_map.py）

compute_logical_to_rank_dispatch_physical_map / _compute_logical_to_all_physical_map / _prefer_same_node_experts 删除 server_args 参数，全部从 bag 读取；_init_raw 同步删除未用的 server_args 参数。单测从 types.SimpleNamespace stub 改为 get_context().override_server_args(...) 作用域发布（新增 _published 辅助函数），被测函数读到的就是测试发布的那份配置，避免 stub 与真实读取路径脱节。

5. 验证与配套

全量注册 CPU 套件在相同机器 / 环境双跑（本分支 vs main，16 并行分区）零不对称失败；GPU smoke：piecewise-prefill 编译（58 token tiers，无 recompile storm）与 tp2/dp2 DP-attention + enable_dp_lm_head 服务均通过。labels 含 amd 与 deepseek，牵动 AMD 与 DeepSeek 相关 CI 测试面。未包含多节点弹性 EP 的端到端测试。

关键文件：

- python/sglang/srt/runtime_context.py（模块 运行时上下文；类别 source；类型 core-logic；符号 ParallelContext.getattr, ParallelContext）：ParallelContext.__getattr__重构为本 PR 的基石：从 object.__getattribute__防御式取槽改为槽位直读 + _fields 字典判断，使配置叶子读取在 torch.compile 下保持可追踪，并打断 pickle/copy 协议探测槽位的递归。
- python/sglang/srt/eplb/expert_location.py（模块 专家路由；类别 source；类型 core-logic；符号 _prefer_same_node_experts, _compute_logical_to_all_physical_map, compute_logical_to_rank_dispatch_physical_map, ExpertLocationMetadata._init_common）：专家放置辅助函数族（_prefer_same_node_experts、_compute_logical_to_all_physical_map、compute_logical_to_rank_dispatch_physical_map、_init_common/_init_raw）彻底去除 server_args 参数，全部配置读改走 get_exec().moe / get_parallel()，是本 PR 中配置契约变化最深的模块。
- python/sglang/srt/model_executor/model_runner.py（模块 模型执行；类别 source；类型 data-contract；符号 _initialize_elastic_ep_joiner, _expand_eplb_metadata_for_scale, _elastic_global_rank, _finalize_scale_up）：弹性 EP 写路径重路由的主战场：_initialize_elastic_ep_joiner、_expand_eplb_metadata_for_scale、_finalize_scale_up 的 4 处 override 从 server_args 改为 get_context().override，使扩容值落进已发布 bag，同时批量切换同文件配置读取。
- python/sglang/srt/utils/common.py（模块 共享工具；类别 source；类型 dependency-wiring；符号 require_mlp_tp_gather, require_attn_tp_gather, require_mlp_sync）：gather 模式判定三件套（require_mlp_tp_gather / require_attn_tp_gather / require_mlp_sync）改读 bag，消除弹性扩容后从陈旧 dp_size 推导 gather 模式的风险——这正是 review 中 base_runner 评论暴露的问题。
- python/sglang/srt/managers/scheduler_pp_mixin.py（模块 PP 调度；类别 source；类型 dependency-wiring；符号 event_loop_pp, event_loop_pp_disagg_prefill, event_loop_pp_disagg_decode, init_pp_loop_state）：PP 调度热循环中 9 处 pp_async_batch_depth / enable_dsa_prefill_context_parallel 从 server_args 切到 get_parallel()，覆盖 event_loop_pp 三个变体与 init_pp_loop_state，是调度路径的代表性变更。
- python/sglang/srt/disaggregation/common/conn.py（模块 PD 连接；类别 source；类型 core-logic；符号 register_to_bootstrap, _sync_bootstrap_port_across_nodes, CommonKVManager.init）：prefill DP 路由（bootstrap_room % dp_size 校验）、load_balance_method、enable_dsa_cache_layer_split、nnodes 等改读 bag，解决 review 指出的弹性扩容后路由与 bag 不一致问题及防御性 getattr 残留。
- test/registered/unit/eplb/test_compute_logical_to_rank_dispatch_physical_map.py（模块 单元测试；类别 test；类型 test-coverage；符号 _published, TestComputeLogicalToRankDispatchPhysicalMap）：测试从 SimpleNamespace stub 改为 get_context().override_server_args 作用域发布（_published），确保被测函数读到的就是测试发布的配置，代表了迁移后单元测试的正确写法。

- python/sglang/srt/speculative/dspark_components/dspark_worker_v2.py（模块 投机解码；类别 source；类型 dependency-wiring；符号 _forward_prefill, _forward_decode, _commit_target_mamba_states_after_verify, _dp_verify_tier_num_tokens）：投机解码（DSPark）worker 的 enable_dp_attention、speculative_eagle_topk、mamba_track_interval 等 7 处读取切换访问器，展示 speculative 模块的迁移模式。
- python/sglang/srt/model_loader/loader.py（模块 模型加载；类别 source；类型 data-contract）：模型加载的 weight-info 字典中 moe_dense_tp_size、enable_dp_lm_head 改读 parallel 访问器，保证权重加载决策与已发布配置一致。
- python/sglang/srt/models/deepseek_v4.py（模块 模型定义；类别 source；类型 data-contract）：代表一批 MoE 模型文件（bailing_moe、glm4_moe、qwen3_moe、sdar、llada2 等，各 +2/-8）：lm_head 的 use_attn_tp_group 从 get_server_args() 改为 get_parallel()。

关键符号：ParallelContext.getattr, _prefer_same_node_experts, _compute_logical_to_all_physical_map, compute_logical_to_rank_dispatch_physical_map, ExpertLocationMetadata._init_common, ExpertLocationMetadata._init_raw, require_mlp_tp_gather, require_attn_tp_gather, require_mlp_sync, _initialize_elastic_ep_joiner, _expand_eplb_metadata_for_scale, _finalize_scale_up, init_pp_loop_state, register_to_bootstrap

关键源码片段

python/sglang/srt/runtime_context.py

ParallelContext.__getattr__重构为本 PR 的基石：从 object.__getattribute__防御式取槽改为槽位直读 + _fields 字典判断，使配置叶子读取在 torch.compile 下保持可追踪，并打断 pickle/copy 协议探测槽位的递归。

```
class ParallelContext:
    """并行拓扑命名空间。

    实时拓扑（size / rank / group）通过 ``@property`` 读透（规范 getter），而
    并行**配置**叶子（``nccl_port``、``pp_max_micro_batch_size``、
    ``enable_dp_attention`` ...）通过 ``__getattr__`` 从已发布的 parallel
    配置 bag 供给。当配置叶子与实时 property 重名（如 ``tp_size``）时，
    property（实时事实）优先；dist 就绪后两者保持同值。
    """

    __slots__ = ("_overrides", "_config")

    def __init__(self):
        self._overrides = {}
        self._config = None # parallel 配置 bag，publish 时接线

    def __getattr__(self, name):
        # 只有在既非实时 @property 也非 slot 时才会走到这里：从已发布的 bag
        # 提供 parallel 配置叶子。函数体必须保持 dynamo 可追踪 ——
        # get_parallel().moe_dense_tp_size 这类读取可能跑在编译后的模型
```

```
        # forward 里，而 object.__getattribute__ 会 graph-break。
        if name.startswith("_"):
            # 配置叶子没有下划线前缀；这里直接抛错还能打断 pickle/copy
            # 协议在 __init__ 之前探测槽位导致的递归。
            raise AttributeError(name)
        config = self._config
        # _fields 是 bag 上的普通 __dict__ 条目；对 dict 做 in 判断，
        # 避开 _ConfigBag.__contains__（不可追踪）。
        if config is not None and name in config._fields:
            return getattr(config, name)
        detail = (
            "not a published parallel config leaf"
            if config is not None
            else "config not published"
        )
        raise AttributeError(f"ParallelContext has no {name!r} ({detail})")
```

python/sglang/srt/eplb/expert_location.py

专家放置辅助函数族（_prefer_same_node_experts、_compute_logical_to_all_physical_map、compute_logical_to_rank_dispatch_physical_map、_init_common/_init_raw）彻底去除 server_args 参数，全部配置读改走 get_exec().moe / get_parallel()，是本 PR 中配置契约变化最深的模块。

```
def _prefer_same_node_experts() -> bool:
    """同节点专家偏好判定；原签名携带 server_args，现从已发布配置读取。"""
    from sglang.srt.elastic_ep.elastic_ep import elastic_expanded_world_enabled
    from sglang.srt.runtime_context import get_exec

    return (
        get_exec().moe.ep_join_mode != "scale" and not elastic_expanded_world_enabled()
    )


@staticmethod
def _init_common(server_args: ServerArgs, model_config: ModelConfig):
    # 局部导入避免模块级循环依赖；读取统一走 bag 访问器。
    from sglang.srt.runtime_context import get_exec, get_parallel

    model_config_for_expert_location = (
        ModelConfigForExpertLocation.from_model_config(model_config)
    )
    if model_config_for_expert_location is None:
        return None

    base_num_physical_experts = (
        model_config_for_expert_location.num_logical_experts
        + get_exec().moe.ep_num_redundant_experts
    )
    # 弹性 EP 扩容会通过 get_context().override 改写已发布 bag 里的 ep_size，
```

```
    # 因此这里必须读 bag（唯一已解析来源），而不是 pristine 的 server_args。
    ep_size = get_parallel().ep_size
    num_physical_experts = base_num_physical_experts
    initial_ep_size = get_parallel().elastic_ep_initial_size
    if initial_ep_size is not None:
        if get_exec().moe.ep_join_mode == "scale":
            # tp_size 是刻意保留的 live-shadowed 拓扑尺寸（活属性优先）
            ep_size = max(
                ep_size,
                get_parallel().ep_join_rank_offset + server_args.tp_size,
            )
        num_physical_experts, num_local_physical_experts = (
            _compute_elastic_expert_layout(
                base_num_physical_experts,
                initial_ep_size,
                ep_size,
            )
        )
    else:
        assert num_physical_experts % ep_size == 0
        num_local_physical_experts = num_physical_experts // ep_size
    # ... 后续沿用原布局计算逻辑构建 common 字典
```

python/sglang/srt/utils/common.py

gather 模式判定三件套（require_mlp_tp_gather / require_attn_tp_gather / require_mlp_sync）改读 bag，消除弹性扩容后从陈旧 dp_size 推导 gather 模式的风险——这正是 review 中 base_runner 评论暴露的问题。

```
def require_mlp_tp_gather(server_args: ServerArgs):
    """MLP 输入是否经 all-gather 而非 all-reduce 获得。

    仅当每个 MLP TP 组内含多个 attention DP 组时成立。elastic-EP 扩容会把
    dp_size 改写进已发布 bag，因此所有 DP 相关叶子必须读 bag，避免按陈旧
    实例值错误推导 gather 模式。
    """
    from sglang.srt.layers.moe.utils import get_moe_a2a_backend
    from sglang.srt.runtime_context import get_exec, get_parallel

    if get_parallel().enable_dp_attention:
        assert get_parallel().dp_size > 1, "dp_size must be greater than 1"
        if get_exec().moe.elastic_ep_backend is not None:
            from sglang.srt.elastic_ep.elastic_ep import (
                elastic_expanded_world_enabled,
            )

            if elastic_expanded_world_enabled():
                return True
        if get_parallel().moe_dense_tp_size is None:
            # TODO(ch-wan): 部分 MoE 模型没有 dense 层
```

```
            return True
        elif not get_parallel().enable_dp_lm_head:
            return True
        elif get_moe_a2a_backend().is_none():
            return True
        elif get_moe_a2a_backend().is_flashinfer():
            # FlashInfer MoE A2A 需要 rank 无关、DP 同步的每 rank token 数:
            # decode cuda-graph bucket 必须在各 EP rank 完全一致，否则重放
            # 不同尺寸的图会造成非法内存访问（issue #30242）。
            return True
        else:
            return (
                get_parallel().moe_dense_tp_size
                > server_args.tp_size // get_parallel().dp_size
            )
    else:
        return False
```

评论区精华

本 PR 由作者 ch-wan 自审，r2 有 4 项 open items，r3 全部关闭（"Clean — no open findings"）。核心交锋：

> "gather-mode detection still calls require_mlp_tp_gather(mr.server_args)... Those helpers still branch on server_args.dp_size... After bag-only scale, re-entrant _dummy_run / buffer alloc can recompute gather mode from stale dp_size."

base_runner.py 的评论指出弹性扩容后重新进入的 dummy buffer 分配可能基于陈旧 dp_size 推导 gather 模式，require_mlp_tp_gather 内部对 EP scale 的 short-circuit 会掩盖此问题但很脆弱。作者修复为三个 helper 全部改读 bag。

> "the function is half-migrated and will miss any future bag-only override of the siblings"

expert_location.py 的评论强调半迁移表面的隐患：ep_size 已切 bag 但同块 sibling 叶子（elastic_ep_initial_size、ep_join_rank_offset）还在 server_args 上；后续跟进评论又指出 ep_num_redundant_experts、ep_dispatch_algorithm 残留。均被修复，_init_raw 删除 server_args 参数，server_args.tp_size 作为刻意保留的 live-shadowed 读取。

conn.py 的评论指出防御性 getattr(self.server_args, "enable_dsa_cache_layer_split", False) 与迁移方向及项目 no-defensive-getattr 规范不一致，改为 get_parallel().enable_dsa_cache_layer_split。ratchet 数字的 nit（描述写 39→35，实际 38→34）也已修正描述。

- base_runner 的 gather 模式 helper 仍读 pristine server_args，弹性扩容后可能从陈旧 dp_size 重推导 gather mode (correctness): 三个 helper 全部改读 bag 叶子（enable_dp_attention、dp_size、moe_dense_tp_size、enable_dp_lm_head、disable_attn_tp_gather；elastic_ep_backend 走 get_exec().moe），tp_size 按 live-shadowed 规则保留参数。

- expert_location 半迁移表面：ep_size 已切 bag 但 sibling 叶子仍在 server_args (design): 全部改读 get_parallel() / get_exec().moe；_init_raw 删除未用 server_args 参数；server_args.tp_size 保留为刻意 live-shadowed 读取。
- conn.py bootstrap 注册保留防御性 getattr，与迁移方向不一致 (style): 改为 get_parallel().enable_dsa_cache_layer_split，去除防御性 getattr。
- r2 disposition 四项 open items（PD 路由、system_dp_size、dsa_cache_layer_split、expert_location 残留） (correctness): 四项全部在 head b38d5a7f98a4 修复；r3 复核 clean，无遗留 finding。
- ServerArgs.override ratchet 数字与 PR 描述 off-by-one (documentation): PR 描述已修正为 38 → 34。

风险与影响

- 风险：
  1. 大面机械翻转（81 文件）：任何漏网读取会在运行时 fail-loud（AttributeError），这比静默读错值安全，但若类似 #33168 的写路径残留，会再次出现 " 写入无人读取的实例 " 类 bug——本 PR 正是为此专门重路由了 4 处弹性 EP 写入。
  2. dynamo 图编译约束：__getattr__必须保持可追踪，后续任何引入 object.__getattribute__或 _ConfigBag.__contains__的改动都会 graph-break，唯一的护栏是 test_parallel_config_leaves_trace_under_torch_compile fullgraph 测试。
  3. 弹性 EP 场景验证深度有限：review 发现并修复的 base_runner 重进入路径（_dummy_run / buffer alloc）仍依赖 elastic_expanded_world_enabled 的 short-circuit 掩盖；本 PR 未做多节点弹性扩容端到端测试。
  4. 行为语义变化：get_parallel() 对未发布字段抛 AttributeError 而非返回旧值，属于有意的 fail-fast，但对外部扩展代码是破坏性变化。
  5. 运行期开销：event_loop_pp 等热循环中 pp_async_batch_depth 由实例属性访问变为 getattr+ _fields 检查 + getattr，单次开销极小但处于每 micro-batch 每 step 的路径上，需留意后续性能回归。 - 影响：对用户：无任何 CLI/API 变化；行为修复体现在弹性 EP 扩容后配置一致性与 chunked-prefix 类 gate 的读写同源。对系统：完成 parallel 命名空间的访问器迁移，消除 106 处双源读取；override 调用点收敛至 34 处，配置只有 bag 一个已解析来源。对团队：确立了全仓配置读写规范——新代码必须通过 get_parallel()/get_exec()/get_spec() 等访问器读取配置，不得直读 server_args（除了刻意保留的 live-shadowed 拓扑尺寸）；同时 ratchet 机制让后续迁移可以机械验证、增量推进。 - 风险标记：跨 81 文件大范围读取路径变更，dynamo 图编译约束（__getattr__需保持可追踪），弹性 EP 扩容缺少多节点端到端测试，热循环新增 __getattr__间接层，fail-loud 行为变更（未发布字段抛 AttributeError）

关联脉络

- PR #33168 Fix the chunked-prefix-cache gate writing config the backends never read: 本 PR 基于它（PR body 要求先合并 #33168），且两者是同类问题：写入与读取两侧分离导致配置不一致。#33168 把 chunked-prefix gate 写入改走 get_context().override，本 PR 把弹性 EP 写入也改走 override；ratchet 数字在其上递减

（39→38→34）。

- PR #33013 config: read resolved config via namespace accessors: 配置访问器迁移的主 PR（160 文件 628 处），本 PR 是其 parallel 命名空间的收尾：parallel 是最后一批仍留在 get_server_args() 上的读取家族。
- PR #33011 config: preserve resolved config across nested publishes + mutation ratchets: 建立配置生命周期快照恢复与 mutation ratchet 基线机制，本 PR 的 ServerArgs.override 调用点 ratchet（38→34）依赖该机制验证。
- PR #33012 runtime_context: record the publishing process role: 让 runtime_context 记录进程角色并角色化 publish，get_context().override 写入 bag 的基础设施，本 PR 的写路径重路由建立其上。
- PR #33172 runtime_context: per-role namespace enforcement behind SGLANG_ROLE_NAMESPACES: 迁移栈的后续收尾：按角色强约束配置读取，本 PR 完成读取翻转后该强制机制才可全面生效。
- PR #33179 [CI] Fix runtime context setup in flat logprob tests: runtime context 迁移后的测试修复，与本 PR 属于同一迁移栈的防护网，防止访问器切换引入的测试环境问题。