PR #28567 完整报告

sgl-project/sglang

Add get_parallel(): a structured accessor for parallel-topology state

合并时间：2026-06-18 11:23

原文链接：http://prhub.com.cn/sgl-project/sglang/pull/28567

执行摘要

- 一句话：引入 get_parallel() 统一访问并行拓扑状态
- 推荐动作：值得精读。设计上惰性导入和 override 栈式恢复非常干净，可做为仓库基础架构层类似抽象的参考。关注其 _v() 方法如何统一覆盖与实时查找，以及 override 测试隔离的实现技巧。

功能与动机

Reading the parallel topology today means importing and calling a dozen free functions spread across parallel_state and dp_attention. This PR adds a single structured accessor, get_parallel(), so call-sites use one import and one consistent naming scheme, and so tests can force a topology without monkeypatching the individual getters.

实现拆解

1. 新增 runtime_context.py: 创建 ParallelContext 类，每个并行属性（tp_size, pp_rank, moe_ep_size, attn_dp_rank 等）定义为 @property，通过内部 _v() 方法委托给 parallel_state / dp_attention 的规范 getter。override() 上下文管理器使用栈式保存 / 恢复，支持嵌套和键校验。
2. 惰性导入: _ps() 和 _dp() 函数在属性第一次访问时才 import parallel_state 和 dp_attention，避免模块级循环依赖。
3. 模型文件迁移: 将 apertus.py、solar.py、gpt_oss.py、deepseek_v2.py 等 180+ 文件中的分散 getter 调用（如 get_tensor_model_parallel_world_size()）替换为 get_parallel().tp_size，并清理对应 import。
4. 测试文件适配: DSA / MLA attention 后端测试原通过 monkey-patch dp_attention.get_attention_tp_size，改为使用 get_parallel().override(attn_tp_size=1)，同时修复多个测试套件间的 override 隔离问题。
5. 单元测试: 新增加 test_runtime_context.py，覆盖 delegation 表格（每个属性映射到正确 getter）、singletons、override 优先级、嵌套和未知键报错。

关键文件：

- python/sglang/srt/runtime_context.py（模块 运行时；类别 source；类型 core-logic；符号 ParallelContext, get_parallel, get_context, _ps）：核心新文件，定义了

ParallelContext 类和 get_parallel() 函数，是本次变更的基石。

- python/sglang/srt/models/apertus.py（模块 模型层；类别 source；类型 data-contract；符号 ApertusAttention.init, ApertusDecoderLayer.init, import 列表）：模型文件代表，展示从分散 getter 到 get_parallel() 的典型迁移，包括 import 简化和调用替换。
- test/registered/unit/test_runtime_context.py（模块 测试；类别 test；类型 test-coverage；符号 TestParallelDelegation, TestRuntimeContextSingletons, TestParallelOverride, _IsolatedOverrides）：全面测试 delegation 映射、singletons、override 机制，确保核心逻辑正确。

关键符号：sglang.srt.runtime_context.get_parallel, sglang.srt.runtime_context.get_context, sglang.srt.runtime_context.ParallelContext._v, sglang.srt.runtime_context.ParallelContext.override, sglang.srt.runtime_context.RuntimeContext

关键源码片段

python/sglang/srt/runtime_context.py

核心新文件，定义了 ParallelContext 类和 get_parallel() 函数，是本次变更的基石。

```
# 并行字段白名单，用于 override() 的键校验
_PARALLEL_FIELDS = frozenset({
    "world_size", "world_rank", "tp_size", "tp_rank",
    "pp_size", "pp_rank",
    "moe_ep_size", "moe_ep_rank", "moe_dp_size", "moe_dp_rank",
    "moe_tp_size", "moe_tp_rank",
    "attn_tp_size", "attn_tp_rank", "attn_cp_size", "attn_cp_rank",
    "attn_dp_size", "attn_dp_rank",
    "world_group", "tp_group", "pp_group",
    "moe_ep_group", "moe_dp_group", "moe_tp_group",
    "attn_tp_group", "attn_cp_group",
})

class ParallelContext:
    __slots__ = ("_overrides",)

    def __init__(self):
        self._overrides = {} # 栈式 override 存储

    def _v(self, name, getter):
        # 如果 override 中有则返回覆盖值，否则实时调用 getter
        return self._overrides[name] if name in self._overrides else getter()

    @contextmanager
    def override(self, **kwargs):
        # 验证键名，保存当前状态，更新覆盖，退出时恢复
        unknown = set(kwargs) - _PARALLEL_FIELDS
        if unknown:
            raise ValueError(f"unknown parallel field(s): {sorted(unknown)}")
```

```
        saved = dict(self._overrides)
        self._overrides.update(kwargs)
        try:
            yield self
        finally:
            self._overrides = saved

    # 下面是部分属性的委托实现
    @property
    def world_size(self) -> int:
        return self._v("world_size", _ps().get_world_size)

    @property
    def tp_size(self) -> int:
        return self._v("tp_size", _ps().get_tensor_model_parallel_world_size)

    @property
    def attn_tp_size(self) -> int:
        return self._v("attn_tp_size", _ps().get_attn_tensor_model_parallel_world_size)

    @property
    def moe_ep_rank(self) -> int:
        return self._v("moe_ep_rank", _ps().get_moe_expert_parallel_rank)

    # ... ( 其余属性遵循相同模式 )
```

python/sglang/srt/models/apertus.py

模型文件代表，展示从分散 getter 到 get_parallel() 的典型迁移，包括 import 简化和调用替换。

```
# 变更前：import 多个 getter
from sglang.srt.distributed import (
    get_pp_group,
    get_tensor_model_parallel_rank,
    get_tensor_model_parallel_world_size,
)

# 变更后：只引入 get_parallel
from sglang.srt.runtime_context import get_parallel

# 使用对比
# 变更前 :
tp_size = get_tensor_model_parallel_world_size()
attn_tp_rank = get_attention_tp_rank()

# 变更后 :
tp_size = get_parallel().tp_size
attn_tp_rank = get_parallel().attn_tp_rank
```

test/registered/unit/test_runtime_context.py

全面测试 delegation 映射、singletons、override 机制，确保核心逻辑正确。

```
# 定义每个属性应该委托给哪个 getter
SIZE_RANK_DELEGATIONS = [
    ("world_size", f"{_PS}.get_world_size"),
    ("tp_size", f"{_PS}.get_tensor_model_parallel_world_size"),
    ("attn_dp_size", f"{_DP}.get_attention_dp_size"),
    # ... 完整表格见源码
]

class TestParallelDelegation(_IsolatedOverrides):
    def test_size_rank_delegate_to_canonical_getters(self):
        # 为每个 getter 打上不同桩值，确保属性正确委托
        for i, (attr, target) in enumerate(SIZE_RANK_DELEGATIONS):
            sentinel = 1000 + i
            with patch(target, return_value=sentinel):
                self.assertEqual(
                    getattr(get_parallel(), attr),
                    sentinel,
                    msg=f"{attr} must delegate to {target}",
                )

    def test_override_takes_precedence(self):
        p = get_parallel()
        with p.override(tp_size=99, tp_rank=3):
            self.assertEqual(p.tp_size, 99)
            self.assertEqual(p.tp_rank, 3)
        # 退出后恢复
        self.assertNotEqual(p.tp_size, 99)
```

评论区精华

review 中 chatgpt-codex-connector[bot] 提出两个 P2 问题：

1. DSA indexer 测试的 TP override：原测试通过 patch dp_attention.get_attention_tp_size 返回 1，但替换为 get_parallel().attn_tp_size 后需要 attn_tp_group 初始化而失败。作者在后续提交中改为使用 override() 并调整初始化顺序。
2. 测试隔离：test_runtime_context 的 teardown 清空整个 override map，可能误清其它测试文件设置的全局 override。作者增加 _IsolatedOverrides 基类，在 setUp/tearDown 中保存全局 override，只清除测试自身注入的部分。

- DSA indexer 测试的 TP 覆盖适配 (testing): 作者在后续提交中让测试在导入后端前通过 override(attn_tp_size=1) 设置覆盖，避免依赖实际初始化。
- override 全局状态隔离 (testing): 作者增加 _IsolatedOverrides 基类，在 setUp 保存全局 override，teardown 只清除测试自身注入的部分，避免污染。

风险与影响

- 风险：
  1. 回归风险（低）：模型文件替换了分散 getter 调用，但 get_parallel() 是纯 read-through，返回值与原始 getter 一致，不会改变计算语义。不过 184 个文件的 import 调整可能遗漏某些路径，需依赖 CI 测试验证。
  2. 测试隔离风险（已修复）：初始实现中 test_runtime_context 的 teardown 会清空所有 override，影响并行测试套件。后续提交通过 _IsolatedOverrides 基类解决了该问题。
  3. 导入周期风险（低）：使用惰性导入，不会在模块加载时触发 parallel_state 和 dp_attention，避免循环依赖。
- 影响：
  1. 开发者体验: 统一命名空间，新模型无需逐一 import 多个 getter，测试可直接使用 override() 而非 monkey-patch。
  2. 代码可维护性: 并行维度访问集中管理，未来增加维度（如 seq_parallel）只需在 ParallelContext 添加属性。
  3. 零运行时开销: 属性是实时委托，无缓存，不影响性能。
  4. 测试范围: 新增 delegation 和 override 测试，提高核心机制可靠性。 - 风险标记：大量文件变更，测试隔离问题（已修复）

关联脉络

- PR #28568 Use get_parallel() for parallel dimensions in the model forward path: 该 PR 是 #28567 的第二个 commit，完成模型文件的前向路径迁移，属于同一功能线的增量改动。