PR #35929 完整报告

sgl-project/sglang

Report the whole server's world size in the scheduler's internal state

合并时间：2026-08-24 20:21

原文链接：http://prhub.com.cn/sgl-project/sglang/pull/35929

执行摘要

- 一句话：调度器内部状态新增整机 world_size 字段
- 推荐动作：值得快速浏览：它展示了如何在不破坏现有契约的前提下，为调度器内部状态叠加一个派生字段，并用四象限测试矩阵固定 DP / DP attention 的并行语义边界。关注 observability 或调度状态契约的读者可以精读 scheduler.py 的 get_internal_state() 与新增测试文件。

功能与动机

PR body 与提交信息指出：/server_info 携带每个 DP scheduler 的 get_internal_state() 负载，但仅凭声明的并行尺寸无法得知服务器实际占用多少 GPU。没有 --enable-dp-attention 时，每个 data-parallel 副本都会拉起自己的一套 TP 进程组、独占各自 GPU；开启 DP attention 时 DP rank 则共享 TP GPU。因此需要有一个统一的推导函数，把整台服务器的 world size 直接暴露给外部消费者（如定位外部集群规模的对端）。

实现拆解

1. 在 python/sglang/srt/server_args.py 中新增纯函数 compute_world_size(server_args)。- 公式为 (1 if enable_dp_attention else dp_size) * tp_size * pp_size，把「DP attention 复用 TP GPU、普通 DP 各自占 GPU」的语义集中在一个地方，避免各调用点重复推导。
2. 在 python/sglang/srt/managers/scheduler.py 中接线：
   - 新增 get_server_args 与 compute_world_size 的导入；
   - 在 get_internal_state() 构造返回 dict 时写入 ret["world_size"] = compute_world_size(get_server_args())，使 /server_info 与 /set_internal_state 读回均能拿到该字段。
3. 新增 test/registered/unit/managers/test_scheduler_internal_state_world_size.py：
   - TestComputeWorldSize 用四象限用例固定公式语义（单卡 =1、tp*pp 相乘、普通 DP 相乘、DP attention 不乘 dp_size）；
   - TestSchedulerInternalStateWorldSize 通过 Scheduler.__new__构造实例并 patch runtime context 依赖，验证 get_internal_state 输出中的 world_size 为整机值而非单副本值。
4. 修改 test/registered/unit/managers/test_scheduler_internal_state_env_vars.py：

- 在既有 _get_internal_state 的 patch 栈中补充 get_server_args 与 compute_world_size 的 mock，避免新增调用导致旧用例在未发布 server_args 的测试环境中抛异常。

关键文件：

- python/sglang/srt/server_args.py（模块 服务参数；类别 source；类型 core-logic；符号 compute_world_size）：新增 compute_world_size 纯函数，集中定义整机 GPU 数的推导公式，是本次变更的核心逻辑。
- python/sglang/srt/managers/scheduler.py（模块 调度器；类别 source；类型 dependency-wiring；符号 get_internal_state）：在 get_internal_state() 中写入 world_size 字段，并新增 get_server_args / compute_world_size 导入，是本次变更的接线点。
- test/registered/unit/managers/test_scheduler_internal_state_world_size.py（模块 调度状态；类别 test；类型 test-coverage；符号 _make_server_args, TestComputeWorldSize, test_a_single_gpu_server_holds_one_gpu, test_tensor_and_pipeline_stages_multiply）：新增测试文件，用四象限单测固定 compute_world_size 公式，并用集成式用例验证 get_internal_state 输出整机 world_size。
- test/registered/unit/managers/test_scheduler_internal_state_env_vars.py（模块 调度状态；类别 test；类型 test-coverage；符号 _get_internal_state）：在既有测试的 patch 栈中补充 get_server_args 与 compute_world_size 的 mock，保证新增调用不破坏旧用例。

关键符号：compute_world_size, Scheduler.get_internal_state

关键源码片段

python/sglang/srt/server_args.py

新增compute_world_size纯函数，集中定义整机GPU数的推导公式，是本次变更的核心逻辑。

```
# server_args.py

def compute_world_size(server_args: ServerArgs) -> int:
    """返回整个服务器占用的 GPU 总数（跨所有数据并行副本）。"""
    # DP attention 开启时，各 DP rank 复用同一套 TP GPU，dp_size 不再作为乘数；
    # 否则每个 DP 副本都会拉起一套完整的 TP 进程组，需要乘上 dp_size。
    return (
        (1 if server_args.enable_dp_attention else server_args.dp_size)
        * server_args.tp_size # 张量并行规模
        * server_args.pp_size # 流水线并行规模
    )
```

python/sglang/srt/managers/scheduler.py

在 get_internal_state() 中写入 world_size 字段，并新增 get_server_args / compute_world_size 导入，是本次变更的接线点。

```
# scheduler.py 的 get_internal_state() 关键片段

def get_internal_state(self, recv_req: GetInternalStateReq):
```

```
# Resolved config（pristine server_args + post-publish overrides），
# 让读回值反映 /set_internal_state 之后的配置，而非启动快照。
ret = get_context().resolved_server_args_dict()

# 本次新增：让 /server_info 消费者直接拿到整台服务器占用的 GPU 数。
# 在 DP 场景下每个 scheduler 汇报的是整个 server 的 world_size，
# 而不是本 replica 的 tp_size * pp_size。
ret["world_size"] = compute_world_size(get_server_args())

ret["last_gen_throughput"] = self.metrics_reporter.last_gen_throughput
# 后续字段（draft_graph_memory_usage 等）省略
```

评论区精华

该 PR 没有代码层的 review 评论（review_comments 为 0），主要讨论集中在 CI 失败归因，作者多次用 Claude Code / Codex 代理自动排查并给出证据链：

- CPU shard 7 的 15 分钟固定超时被归因为上游 commit 64aa859（PR #35907）扩展 test_resolution_is_reproducible.py 后耗时 358-365 秒，而非本 PR 回归；
- rebased head 的 frozen-base 不兼容问题（test_full_builder_sizes_sidecar_for_anchor_logical_space 需要 publish memory config bag）也被确认与 op22-2 无关；
- 最终通过 /rerun-test 手动验证本 PR 新增与修改的测试文件全部通过，CI gate 判定为 PASS。
- CPU shard 7 超时归因 (testing): 确认为上游引入的既有失败，与本 PR 无关。
- frozen-base 不兼容导致 shard 1 失败 (testing): 确认为 frozen-base 不兼容，非 op22-2 回归。
- 手动验证本 PR 测试文件 (testing): PASS，最终 CI gate 判定为 PASS。

风险与影响

- 风险：
  1. scheduler.py 的 get_internal_state() 现在无条件调用 get_server_args()：若 runtime context 尚未发布 server_args（例如初始化早期或某些嵌入场景）将抛异常，使 /server_info 返回 500。测试中都必须 patch 该依赖，说明这是一个需要上下文保证的新依赖。
  2. world_size 语义依赖 enable_dp_attention：DP attention 开启时忽略 dp_size 是设计意图，但若未来引入新的并行维度（如 context parallel、expert parallel 独立扩展），compute_world_size 需要同步扩展，否则统计会偏离实际占用。
  3. 对内部状态契约是向后兼容的新增字段，旧消费者不受影响；但新消费者需理解「DP 场景下每个 scheduler 都汇报整机 world_size」这一语义。 - 影响：影响面集中在调度器状态报告链路：/server_info 与所有读取 get_internal_state() 的消费者（外部 fleet 调度、监控系统）将获得整机 GPU 数；对 DP/PP 大集群部署可避免各消费方自行推导且口径不一。运行时开销可忽略（每次调用仅做一次乘法）。对团队而言，compute_world_size 成为并行规模推导的唯一入口，利于后续维护。 - 风险标记：内部

状态契约新增字段，依赖运行时上下文 server_args，测试需 patch 运行时依赖，并行语义需随新维度扩展

关联脉络

- PR #35907 未知标题（讨论中提及的 commit 64aa859）：讨论中提及该 PR 扩展 test_resolution_is_reproducible.py 导致 CPU shard 7 超时，被确认为与本 PR 无关的既有问题。