PR #52575 完整报告

vllm-project/vllm

[Rust Frontend] Simplify data-parallel size ownership

合并时间：2026-08-18 17:06

原文链接：http://prhub.com.cn/vllm-project/vllm/pull/52575

执行摘要

本 PR 是 Rust frontend 的一次所有权精简重构：把 deployment-wide 数据并行大小从 server 的 Config、AppState 中彻底移除，统一收敛到 engine-core-client 的 TransportMode，并将拓扑校验前移到 EngineCoreClient::connect() 打开 socket 之前。同时把引擎 Ready 上报字段重命名为 effective_data_parallel_size（通过 serde rename 保持 Python wire 兼容），新增 dense-DP 的 /get_world_size E2E 回归测试。整体消除重复状态漂移风险，并让直接使用 client 的调用方获得与 serve 一致的前置校验。

功能与动机

PR body 明确说明：

> Follow up on #51245 after the effective runtime fix landed in #51178. This PR focuses on simplifying deployment-wide data-parallel size ownership and adding a dense-DP regression test.

The refactor removes redundant state that could diverge from the client transport.

动机是：在 #51178 引入显式 DP rank 路由后，部署级 DP 大小仍被 Config 和 AppState 各持有一份且可通过 with_data_parallel_size 覆盖，容易与 client transport 的真实拓扑漂移。本次变更把唯一数据源收进 TransportMode，并让 client 在建立连接前统一校验拓扑。

实现拆解

1. 收敛 DP 大小到 TransportMode（rust/src/engine-core-client/src/client.rs）：
   - TransportMode::Bootstrapped 新增 data_parallel_size 字段，表示部署级 DP 大小，可能大于本 frontend 的 engine_count（supervisor 可把 rank 分区给多个 frontend）。
   - TransportMode::HandshakeOwner 不新增字段，data_parallel_size() 直接返回 engine_count，因为该模式下 Rust 进程负责全部 engine 的启动协商。
   - 新增 TransportMode::data_parallel_size() 统一取值入口。
2. 校验逻辑下沉到连接层：
   - 新增 TransportMode::validate()：校验 DP 大小非零、不超过 u16::MAX + 1（vLLM 两字节 engine 身份上限）；Bootstrapped 模式还校验 engine_count 非零、engine_start_index + engine_count 不溢出、引擎区间不超出部署级 DP 大小。
   - EngineCoreClientConfig::validate() 委托 transport_mode.validate()，EngineCoreClient::connect() 在建立任何 ZMQ socket 前先调用 config.validate()。这

样直接使用 engine-core-client 的调用方也能在开 socket 前得到同样的失败提示。

3. 删除服务端重复状态：
   - rust/src/server/src/config.rs：删除 Config.data_parallel_size 字段及整套 DP 校验分支，Config::validate() 改为 self.transport_mode.validate()?。
   - rust/src/server/src/state.rs：删除 AppState.data_parallel_size 字段、构造时的默认赋值、with_data_parallel_size() 与 data_parallel_size() 方法，路由层不再维护 override 副本。
4. 澄清 Ready 字段语义并保持 wire 兼容（rust/src/engine-core-client/src/protocol/handshake.rs）：
   - Rust 字段由 data_parallel_size 重命名为 effective_data_parallel_size，#[serde(rename = "data_parallel_size")] 保证 Python msgpack wire key 不变。
   - dense 独立 DP 引擎上报 1，部署级大小由 client transport 拥有；同步更新 mock_engine.rs、grpc/tests.rs、cli.rs、cli/tests.rs 与示例文件。
5. 测试与 CI 配套：
   - 新增 tests/v1/distributed/test_dense_dp_world_size.py：拉起 dense-DP 服务，断言 /get_world_size 默认返回 TP * DP，include_dp=false 返回 TP。
   - 新增 client_config_validates_bootstrapped_dp_range：验证前端可只拥有全局 DP rank 子集（配置合法），越界配置被拒绝。
   - rust/src/server/src/routes/tests.rs 删除依赖 override 路径的两个旧 world-size 测试，并简化 fixture；更新 3 个 Buildkite 配置接入新测试。

rust/src/engine-core-client/src/client.rs

核心变更文件：Bootstrapped 新增 data_parallel_size 字段，新增 TransportMode::data_parallel_size()/validate()，并在 connect() 前统一执行拓扑校验，是整次重构的枢纽。

```
impl TransportMode {
    /// 返回 deployment-wide 的数据并行大小。
    ///
    /// HandshakeOwner 模式下 Rust 进程负责全部 engine 的启动协商，`engine_count`
    /// 即为全局 DP 大小；Bootstrapped 模式下 supervisor 可能把全局 DP rank 分割到
    /// 多个 frontend，因此必须显式记录 `data_parallel_size`。
    pub fn data_parallel_size(&self) -> usize {
        match self {
            Self::HandshakeOwner { engine_count, .. } => *engine_count,
            Self::Bootstrapped {
                data_parallel_size, ..
            } => *data_parallel_size,
        }
    }

    /// 在打开任何 socket 之前校验 transport 拓扑，避免配置错误延迟到引擎连接阶段
    /// 才暴露。该校验由 `EngineCoreClient::connect` 统一调用，因此直接使用 client
    /// 的调用方与 serve 路径体验一致。
```

```
    pub fn validate(&self) -> Result<()> {
        let data_parallel_size = self.data_parallel_size();
        if data_parallel_size == 0 {
            bail_invalid_client_config!("data parallel size must be at least 1");
        }
        if data_parallel_size > usize::from(u16::MAX) + 1 {
            bail_invalid_client_config!(
                "data parallel size ({data_parallel_size}) exceeds the two-byte engine identity limit"
            );
        }

        match self {
            Self::HandshakeOwner { .. } => {}
            Self::Bootstrapped {
                engine_start_index,
                engine_count,
                ..
            } => {
                // Bootstrapped 模式允许前端只拥有全局 DP rank 的子集（例如 supervisor
                // 把 4 个 rank 分给两个 frontend 各 2 个），但引擎区间必须落在部署级
                // DP 大小之内。
                if *engine_count == 0 {
                    bail_invalid_client_config!("engine count must be at least 1");
                }
                let engine_start_index = usize::try_from(*engine_start_index).map_err(|_| {
                    Error::InvalidClientConfig {
                        message: "engine start index does not fit usize".to_string(),
                    }
                })?;
                let engine_end_index =
                    engine_start_index.checked_add(*engine_count).ok_or_else(|| {
                        Error::InvalidClientConfig {
                            message: "engine start index + engine count overflows".to_string(),
                        }
                    })?;
                if engine_end_index > data_parallel_size {
                    bail_invalid_client_config!(
                        "connected engine range [{engine_start_index}, {engine_end_index}) exceeds
                        data parallel size ({data_parallel_size})"
                    );
                }
            }
        }

        Ok(())
    }
}
```

rust/src/engine-core-client/src/protocol/handshake.rs

Ready 响应字段重命名为 effective_data_parallel_size，并用 serde rename 保持 Python wire key 不变，是语义澄清与兼容性的核心。

```
pub struct EngineCoreReadyResponse {
    // ... 其他握手元数据字段省略 ...

    /// 本 EngineCore 有效并行配置中的 data-parallel 大小。
    ///
    /// dense 独立 DP 引擎会被重配置为上报 `1`，deployment-wide 数值由
    /// client transport 的 `TransportMode` 持有，避免引擎上报值与前端
    /// 路由拓扑不一致，也避免不同 frontend 各自持有可漂移的副本。
    #[serde(rename = "data_parallel_size")]
    pub effective_data_parallel_size: u64,
}
```

评论区精华

本 PR 的 review 没有实质技术交锋：claude[bot] 说明 fork 场景自动 review 被禁用，njhill 直接给出 APPROVE。

> Thanks @BugenZhao

设计讨论实际发生在 PR body 与提交信息中，例如第一条提交：

> move data_parallel_size from top-level server config down to bootstrapped transport mode

以及合并前的最后一条提交把 dense-DP world-size 测试隔离到独立进程，说明作者重视测试隔离，避免共享环境变量或设备状态造成 flaky。

风险与影响

- Rust 内部 API breaking：Config.data_parallel_size 与 AppState::with_data_parallel_size 被删除，外部直接构造 Config 的代码（如 external_engine_openai_qwen.rs）需要同步迁移；仓库内已全部更新。
- 校验时机前置：connect() 现在会在打开 socket 前拒绝非法拓扑，配置有误的调用方会得到更早、更明确的失败；但这会改变“先启动再报错”的旧行为，需要确认所有生产入口参数正确。
- 字段语义混淆：effective_data_parallel_size 对 dense 独立 DP 引擎固定为 1，若后续代码误读为部署级 DP 大小会出错；gRPC server info 断言已从 4 改为 2，表明不再依赖 frontend override。
- 测试依赖：新增 E2E 测试需要 GPU 分布式 CI，且依赖 Qwen/Qwen3-0.6B 模型下载，可能受资源波动影响。

整体影响集中在 Rust frontend 内部：系统层面消除重复状态漂移，调用方获得一致的前置校验，团队维护语义更清晰；对外部用户可见的 HTTP/gRPC API 行为保持 #51178 引入的语义不变。

关联脉络

本 PR 是 #51178（显式 DP rank 路由）与 #51245 的后续收尾。它把“部署级 DP 拓扑由 client transport 拥有”这一原则贯彻到底：不再在 server 配置里保留可覆盖副本，并把校验下沉到连接层。此前 #51178 已明确“Rely on EngineCore handshake metadata for deployment-wide DP topology; do not add frontend-owned DP-size or capability fields”，本次变更正是对该声明的最终落实。结合 #52593（Rust 版本信息注入）等近期 PR，可以看到 vLLM 正在持续强化 Rust 前端作为独立、自洽服务层的能力。