PR #25264 完整报告

sgl-project/sglang

move runs_on + rdma into runner_configs.yml

合并时间：2026-05-15 19:19

原文链接：http://prhub.com.cn/sgl-project/sglang/pull/25264

执行摘要

- 一句话：将 runs_on 和 rdma 移入 runner_configs.yml
- 推荐动作：该 PR 值得 CI 相关开发者精读，展示了如何通过 CLI 脚本 + YAML 配置统一管理 GitHub Actions 的多维度参数（安装、标签、RDMA）。设计上采用双模式分离职责，并利用 $GITHUB_OUTPUT 传递结构化数据，是 CI 配置中心化的良好范例。

功能与动机

PR body 描述：将 per-runner 字段（runs_on、rdma_devices）移入 scripts/ci/runner_configs.yml，使其与已有的 install、artifact_version、install_timeout 配置相邻。删除 pr-test.yml 中 13 行 runs_on: 定义，并替换 _pr-test-stage.yml 中的内联 RDMA 三元表达式。

实现拆解

1. 扩展 runner_configs.yml 配置结构：为每个 runner_config 增加 runs_on 和可选的 rdma_devices 字段，其中 B200 类型使用 $b200_runner 占位符，便于动态替换。
2. 重写 runner_configs.py 为双模式 CLI：单配置模式（<runner_config>）仅输出安装相关字段，跳过 runs_on（由 map 模式专门处理）；新增 --map 模式（--map <b200_runner_label>）遍历所有配置，将 $b200_runner 占位符替换为实际标签，输出完整 runs_on_map JSON 供调用方使用。
3. 调整 _pr-test-check-changes.yml：增加 Build runs_on_map 步骤，调用 runner_configs.py --map 替换 B200 sentinel，并将结果设为输出 runs_on_map。
4. 修改 _pr-test-stage.yml：移除内联 runs_on 输入和 SGLANG_CI_RDMA_ALL_DEVICES 环境变量硬编码，改为从 runs_on_map 动态解析 runs-on，并在 setup 步骤中通过 rc.outputs.rdma_devices 导出 RDMA 环境变量。
5. 更新各子工作流：pr-test.yml、pr-test-extra.yml、pr-test-jit-kernel.yml、pr-test-multimodal-gen.yml、pr-test-sgl-kernel.yml 删除冗余的 runs_on 输入，新增 runner_config 和 runs_on_map 传递。

关键文件：

- scripts/ci/runner_configs.py（模块 基础设施；类别 infra；类型 infrastructure；符号 _emit_single, _emit_map）：核心脚本，实现双模式 CLI，负责从 YAML 加载配置并输出给 GitHub Actions。新增 _emit_map 模式解析 B200 sentinel 并生成 JSON 映射。

- scripts/ci/runner_configs.yml（模块 基础设施；类别 infra；类型 infrastructure）：新增 runs_on 和 rdma_devices 字段，是配置集中化的目的地，定义了所有 runner 的标签和设备映射。
- .github/workflows/_pr-test-stage.yml（模块 工作流；类别 infra；类型 infrastructure）：核心阶段模版，修改 runs-on 解析方式为动态从 runs_on_map 获取，并移除内联 RDMA 环境变量，改为从 setup 步骤导出。
- .github/workflows/_pr-test-check-changes.yml（模块 工作流；类别 infra；类型 infrastructure）：新增 Build runs_on_map 步骤，调用 runner_configs.py --map 替换 sentinel，产出 runs_on_map 输出供下游使用。

关键符号：_emit_single, _emit_map

关键源码片段

scripts/ci/runner_configs.py

核心脚本，实现双模式 CLI，负责从 YAML 加载配置并输出给 GitHub Actions。新增 _emit_map 模式解析 B200 sentinel 并生成 JSON 映射。

```
"""Emit runner_config setup for GitHub Actions $GITHUB_OUTPUT.

Two modes:
  1. runner_configs.py <runner_config>
     → outputs key=value lines (install, artifact_version, install_timeout, rdma_devices)
       but skips runs_on (resolved via --map to avoid leaking the sentinel).
  2. runner_configs.py --map <b200_runner_label>
     → outputs runs_on_map={json} with $b200_runner substituted.
"""
import json
import os
import sys
import yaml

_YAML_PATH = os.path.join(os.path.dirname(__file__), "runner_configs.yml")
_B200_SENTINEL = "$b200_runner"

def load() -> dict:
    with open(_YAML_PATH) as f:
        return yaml.safe_load(f)["runner_configs"]

def _emit_single(rc: str) -> None:
    # Output all fields EXCEPT runs_on;
    # runs_on is handled exclusively via --map to avoid exposing the sentinel.
    cfg = load().get(rc)
    if cfg is None:
        sys.exit(f"unknown runner_config: {rc!r}")
    for key, value in cfg.items():
        if key == "runs_on":
            continue
```

```
        print(f"{key}={value}")

def _emit_map(b200_runner: str) -> None:
    # Build a flat dict {runner_config: runs_on} and print as JSON.
    runs_on = {
        name: (b200_runner if cfg.get("runs_on") == _B200_SENTINEL else cfg["runs_on"])
        for name, cfg in load().items()
    }
    print(f"runs_on_map={json.dumps(runs_on, separators=(',', ':'))}")

if __name__ == "__main__":
    args = sys.argv[1:]
    if len(args) == 1:
        _emit_single(args[0])
    elif len(args) == 2 and args[0] == "--map":
        _emit_map(args[1])
    else:
        sys.exit(
            "usage:\n"
            "  runner_configs.py <runner_config>\n"
            "  runner_configs.py --map <b200_runner_label>"
        )
```

评论区精华

1. Sentinel 未在单配置模式解析：gemini-code-assist[bot] 指出 _print_single（后重命名为 _emit_single）输出中 $b200_runner 不会被解析，若该输出被用于设置 runs-on 会导致工作流失败，建议在单配置模式中也支持标签解析。但作者最终选择将 runs_on 完全由 --map 模式处理，单模式跳过该字段以规避风险。
2. CLI 参数解析建议：Review 建议改用 argparse 增强健壮性，但作者未采纳，保持手动解析以保持简单。

- Sentinel 未在单配置模式解析 (correctness): 作者选择将 runs_on 完全由 --map 模式处理，单模式跳过该字段，从设计上避免 sentinel 泄露。
- CLI 参数解析健壮性 (style): 作者未采纳，保持手动解析以保持简单；实际双模式已满足需求。

风险与影响

- 风险：
  - 配置一致性风险：若 runner_configs.yml 中 runs_on 或 rdma_devices 填写错误，可能导致 CI 阶段跑到错误的 runner 或缺少 RDMA 设备，影响测试结果。
  - 动态替换失败风险：$b200_runner sentinel 的替换依赖 _pr-test-check-changes.yml 中 Build runs_on_map 步骤的正确执行，若该步骤出错，B200 阶段将无法匹配 runner。
  - 回退兼容性：由于不再传递内联 runs_on，旧分支合并后若未同步此配置，CI 会因缺失输入而失败。但考虑到这是已合入 main 的修改，向前兼容性由工作流定义保证。

- 影响：
  - 对用户：无直接影响，仅涉及 CI 基础设施。
  - 对系统：CI 配置更集中，新增或修改 runner 配置时只需编辑 runner_configs.yml，无需改动多个工作流文件。
  - 对团队：降低了维护 CI 配置的心智负担，新成员能更快定位 runner 标签定义。影响程度中等，属于正向重构。
  - 风险标记：配置中心化，sentinel 替换依赖，runner 标签错误风险

关联脉络

- PR #25322 Deprecate /rerun-stage; scrub CUDA target_stage infra: 同为 CI 基础设施重构，废弃了 /rerun-stage 和清理 target_stage 机制，与本次配置中心化形成持续优化趋势。
- PR #25320 ci: dispatch pr-test-extra.yml from pr-test.yml on the scheduled cron: 同样涉及 pr-test.yml 和 _pr-test-check-changes.yml 等核心工作流文件，修改了 CI 调度逻辑。