PR #33774 完整报告

sgl-project/sglang

[AMD]Stage MI355X nightly by node count

合并时间：2026-08-06 09:04

原文链接：http://prhub.com.cn/sgl-project/sglang/pull/33774

执行摘要

- 一句话：按节点数拆分 MI355X nightly 测试为 2N/4N 两阶段
- 推荐动作：值得快速浏览。重点看两点：一是按硬件拓扑拆分 CI 矩阵的模式（关键字分组 + 空矩阵跳过 + 未匹配报错），对多节点 nightly 测试有通用参考价值；二是 YAML anchor 复用步骤的写法，可作为团队内部 CI job 模板化的范本。AMD/CI 相关的维护者可精读，其余成员了解即可。

功能与动机

PR body 未填写具体动机（仅为模板内容）。从变更本身看，原 workflow 仅支持 2 节点 1P1D 的 MI355X 配置，本次将 nightly 扩展为同时覆盖 2 节点与 4 节点（-2p1d-ep16）两类拓扑，按节点数分组便于分别调度与控制并行度，并在遇到未识别拓扑时直接失败，避免配置漂移导致漏测。

实现拆解

1. setup 阶段拆分矩阵：在 .github/workflows/nightly-amd-mi355x-disagg.yml 中，将原来只输出 matrix 的步骤改为输出 matrix-2n 与 matrix-4n 两个结果。内嵌 Python 脚本按配置名关键字 -1p1d / -2p1d-ep16 分组，未匹配的拓扑会触发 SystemExit，防止新增配置被静默漏测。
2. 2 节点 job 重命名：nightly-mi355x-benchmark 更名为 nightly-mi355x-2n，消耗 needs.setup.outputs.matrix-2n，并增加矩阵为空的跳过条件（needs.setup.outputs.matrix-2n != '{"include":[]}'），避免空矩阵空跑。
3. 新增 4 节点 job：新增 nightly-mi355x-4n，依赖 setup 与 nightly-mi355x-2n，消耗 matrix-4n，max-parallel 设为 1，避免与 2 节点阶段抢同一批 MI355X 自托管节点。
4. YAML anchor 复用：将 env 定义为 &mi355x-env、steps 定义为 &mi355x-steps，两个 job 通过 *mi355x-env / *mi355x-steps 引用，保证后续调整步骤只改一处。
5. 结果聚合调整：collect-results 改为同时依赖 nightly-mi355x-2n 与 nightly-mi355x-4n，并在 always() 下执行，两个阶段任一失败都会反映到汇总结果。

关键文件：

- .github/workflows/nightly-amd-mi355x-disagg.yml（模块 CI 工作流；类别 infra；类型 infrastructure）：这是该 PR 唯一修改的文件，承载全部 CI 逻辑：setup 阶段按节点数拆分矩阵、新增 4 节点 job、通过 YAML anchor 复用步骤并调整结果聚合依赖。

关键符号：未识别

关键源码片段

.github/workflows/nightly-amd-mi355x-disagg.yml

这是该 PR 唯一修改的文件，承载全部 CI 逻辑：setup 阶段按节点数拆分矩阵、新增 4 节点 job、通过 YAML anchor 复用步骤并调整结果聚合依赖。

```
  # 1) setup job 中，把整份配置矩阵按节点数拆成两个输出
  MATRIX="$MATRIX" python3 - <<'PY'
  import json
  import os

  matrix = json.loads(os.environ["MATRIX"])
  # "-1p1d" 命中 2 节点拓扑，"-2p1d-ep16" 命中 4 节点拓扑
  groups = {
      "matrix-2n": [entry for entry in matrix if "-1p1d" in entry["name"]],
      "matrix-4n": [entry for entry in matrix if "-2p1d-ep16" in entry["name"]],
  }
  matched = {entry["name"] for entries in groups.values() for entry in entries}
  unmatched = [entry["name"] for entry in matrix if entry["name"] not in matched]
  if unmatched:
      # 出现未识别拓扑直接失败，防止 nightly 悄悄漏测新配置
      raise SystemExit("Unrecognized MI355X topology for config(s): " + ", ".join(unmatched))

  with open(os.environ["GITHUB_OUTPUT"], "a") as output:
      for name, entries in groups.items():
          value = {"include": [{"config": entry} for entry in entries]}
          print(f"{name}={json.dumps(value, separators=(',', ':'))}", file=output)
  PY

  # 2) 2 节点与 4 节点 job 通过 YAML anchor 复用环境变量与步骤
  nightly-mi355x-2n:
    strategy:
      fail-fast: false
      max-parallel: 2
      matrix: ${{ fromJson(needs.setup.outputs.matrix-2n) }}
    env: &mi355x-env
      FRAMEWORK: sglang
      HW: mi355x
    steps: &mi355x-steps
      - name: Launch MI355X benchmark
        timeout-minutes: 180

  nightly-mi355x-4n:
    needs: [setup, nightly-mi355x-2n]
    strategy:
      fail-fast: false
      max-parallel: 1
```

```
    matrix: ${{ fromJson(needs.setup.outputs.matrix-4n) }}
  env: *mi355x-env
  steps: *mi355x-steps
```

评论区精华

该 PR 没有 review 评论线程，仅由 bingxche 直接批准。两个 commit（[CI] Stage MI355X nightly by node count 与 [CI] Reuse MI355X benchmark steps across stages）体现了先拆分矩阵、再抽取公共步骤的重构顺序，说明作者在实现过程中自行完成了可维护性取舍。

- 暂无高价值评论线程

风险与影响

- 风险：
  - YAML anchor 耦合：&mi355x-env / &mi355x-steps 被两个 job 共享，若未来 2N 与 4N 需要差异化运行参数或步骤，需先拆 anchor，否则会互相影响。
  - 矩阵分组依赖配置名：分组逻辑基于配置名中的 -1p1d / -2p1d-ep16 关键字，若 nightly-configs.yaml 新增其他拓扑而未同步更新分组脚本，setup 会直接 SystemExit，导致整个 nightly 失败；这是有意为之的防御，但也提高了后续维护门槛。
  - 4 节点 job 依赖链：nightly-mi355x-4n 依赖 2 节点阶段完成，虽然使用 !cancelled() 避免 2N 失败时 4N 被取消，但 2N 失败时 4N 仍会继续申请 MI355X 节点，资源开销可能翻倍。
  - 无配套测试：矩阵拆分逻辑只在真实 nightly 中验证，PR 未包含任何单测或语法校验步骤。
  - 空矩阵判断：matrix-4n 为空时的跳过条件写法为字符串比较 !='{"include":[]}'，与 setup 输出格式强耦合，格式变化会静默失效。
  - 影响：影响范围严格限于 AMD MI355X 的 nightly CI 流程：从原先只跑 2 节点 1P1D，扩展为同时跑 2 节点与 4 节点（2P1D EP16）两类拓扑，新增了一个持续约数小时的 4 节点测试路径。对端到端推理功能、模型输出和用户 API 无任何影响；对团队而言，AMD nightly 的维护者需要关注新 job 的资源调度与稳定性，自托管 runner 的并发占用会明显增加。
  - 风险标记：YAML anchor 复用，矩阵分组依赖配置名，新增 4 节点测试路径，collect-results 依赖变更，无配套测试

关联脉络

- PR #31483 [AMD] ci: run vetted nested multimodal_gen unit tests on AMD: 同为 AMD 平台的 CI 能力扩展，与本 PR 共同反映 AMD 测试覆盖持续演进的方向。