PR #26662 完整报告

sgl-project/sglang

[AMD][CI] Update v4 CI setting and move the task to main branch

合并时间：2026-05-29 15:12

原文链接：http://prhub.com.cn/sgl-project/sglang/pull/26662

执行摘要

- 一句话：将 DeepSeek-V4 AMD CI 测试迁移到主分支标准镜像
- 推荐动作：该 PR 是基础设施改进，没有引入新功能，但对于保持 AMD CI 的可持续性很重要。建议 CI 维护者关注工作流变更后的实际运行时间变化，以及测试脚本中的环境变量是否与最新 run_dsv4.sh 保持同步。一般开发者可跳过阅读。

功能与动机

DeepSeek-V4 模型代码和 ROCm 优化已通过 PR #26383 合并到 main 分支，因此 V4 nightly CI 无需再从专用 DSv4 分支构建镜像，可直接使用标准 ROCm 7.2 镜像，以降低维护成本和镜像构建时间。

实现拆解

1. 更新四个 DSv4 测试脚本：将 COMMON_ENV_VARS 中的环境变量从老旧的 tilelang/AITER 混合方案更新为以 AITER indexer + triton 注意力为主的新配置，移除不必要的变量（如 SGLANG_OPT_USE_TILELANG_SWA_PREPARE、SGLANG_OPT_USE_OLD_COMPRESSOR），新增 SGLANG_DEFAULT_THINKING、SGLANG_OPT_USE_FUSED_COMPRESS_TRITON 等，并简化 FP4_ENV_VARS/FP8_ENV_VARS（移除 SGLANG_FORCE_TRITON_MOE_FP8）。同时调整服务器启动参数：将 --attention-backend 从 compressed 改为 dsv4，新增 --mem-fraction-static 0.90 和 --swa-full-tokens-ratio 0.1 以匹配 main 分支的配置。
2. 修改 GitHub Actions 工作流 nightly-test-amd-rocm720.yml：移除为 DSv4 专用镜像准备的 Resolve DSv4 image tag 步骤（包括 Docker Hub 认证和镜像标签解析），将 Setup docker 步骤改为使用标准 scripts/ci/amd/amd_ci_start_container.sh --rocm-version rocm720。取消 --skip-sglang-build 和 --skip-aiter-build 参数，改为完整安装依赖，使 V4 任务与其他 AMD nightly 任务使用相同的基础镜像和安装流程。
3. 保持测试结构不变：四个测试文件（FP4/FP8 × Flash/Pro）的类结构、GSM8K 评估入口和 register_amd_ci 注册保持不变，仅调整配置以对齐 main 分支的运行时需求。

关键文件：

- test/registered/amd/test_deepseek_v4_flash_fp4.py（模块 V4 测试；类别 test；类型 test-coverage）：DeepSeek-V4 Flash FP4 测试脚本，是四个测试中 VP 之一，变更了环境变量和启动参数以匹配 main 分支。

- test/registered/amd/test_deepseek_v4_flash_fp8.py（模块 V4 测试；类别 test；类型 test-coverage）：DeepSeek-V4 Flash FP8 测试脚本，与 FP4 类似变更，使用 FP8 专用专家设置。
- test/registered/amd/test_deepseek_v4_pro_fp4.py（模块 V4 测试；类别 test；类型 test-coverage）：DeepSeek-V4 Pro FP4 测试脚本，Pro 模型使用更长的超时和不同模型路径。
- test/registered/amd/test_deepseek_v4_pro_fp8.py（模块 V4 测试；类别 test；类型 test-coverage）：DeepSeek-V4 Pro FP8 测试脚本，与 Pro FP4 类似。
- .github/workflows/nightly-test-amd-rocm720.yml（模块 CI 配置；类别 infra；类型 infrastructure）：GitHub Actions 工作流，移除了 DSv4 镜像解析步骤，简化了 V4 任务的 Docker 设置和依赖安装。

关键符号：未识别

关键源码片段

test/registered/amd/test_deepseek_v4_flash_fp4.py

DeepSeek-V4 Flash FP4 测试脚本，是四个测试中 VP 之一，变更了环境变量和启动参数以匹配 main 分支。

```
# test/registered/amd/test_deepseek_v4_flash_fp4.py ( 关键变更部分 )

# 更新后的公共环境变量，使用 AITER indexer + triton 注意力 + ROCm700A
COMMON_ENV_VARS = {
    "SGLANG_DEFAULT_THINKING": "1",
    "SGLANG_DSV4_REASONING_EFFORT": "max",
    "SGLANG_OPT_DEEPGEMM_HC_PRENORM": "false",
    "SGLANG_USE_AITER": "1",
    "SGLANG_USE_ROCM700A": "1",
    "SGLANG_OPT_USE_FUSED_COMPRESS": "true",
    "SGLANG_OPT_USE_FUSED_COMPRESS_TRITON": "true",
    "SGLANG_HACK_FLASHMLA_BACKEND": "triton",
    "SGLANG_OPT_FP8_WO_A_GEMM": "false",
    "SGLANG_OPT_USE_JIT_INDEXER_METADATA": "false",
    "SGLANG_OPT_USE_TOPK_V2": "false",
    "SGLANG_OPT_USE_AITER_INDEXER": "true",
    "SGLANG_OPT_USE_TILELANG_INDEXER": "false",
    "SGLANG_OPT_USE_TILELANG_MHC_PRE": "false",
    "SGLANG_OPT_USE_TILELANG_MHC_POST": "false",
    "SGLANG_FP8_PAGED_MQA_LOGITS_TORCH": "1",
    "SGLANG_OPT_USE_MULTI_STREAM_OVERLAP": "false",
    "SGLANG_ROCM_USE_MULTI_STREAM": "false",
    "AITER_BF16_FP8_MOE_BOUND": "0",
}

# FP4 专用环境变量
FP4_ENV_VARS = {
```

```
    "SGLANG_DSV4_FP4_EXPERTS": "true",
}

# 服务器启动参数，现在使用 dsv4 注意力后端并优化内存比例
other_args = [
    "--trust-remote-code",
    "--tp", "8",
    "--disable-radix-cache",
    "--attention-backend", "dsv4",
    "--max-running-requests", "256",
    "--page-size", "256",
    "--mem-fraction-static", "0.90",
    "--swa-full-tokens-ratio", "0.1",
    "--chunked-prefill-size", "8192",
    "--disable-shared-experts-fusion",
    "--tool-call-parser", "deepseekv4",
    "--reasoning-parser", "deepseek-v4",
]
```

test/registered/amd/test_deepseek_v4_flash_fp8.py

DeepSeek-V4 Flash FP8 测试脚本，与 FP4 类似变更，使用 FP8 专用专家设置。

```
# test/registered/amd/test_deepseek_v4_flash_fp8.py (FP8 专用部分 )

COMMON_ENV_VARS = { ... } # 与 FP4 相同

# FP8 专用环境变量：保留 SGLANG_DSV4_FP4_EXPERTS=false
FP8_ENV_VARS = {
    "SGLANG_DSV4_FP4_EXPERTS": "false",
}

# 服务器启动参数与 FP4 一致
other_args = [
    "--trust-remote-code",
    "--tp", "8",
    "--disable-radix-cache",
    "--attention-backend", "dsv4",
    "--max-running-requests", "256",
    "--page-size", "256",
    "--mem-fraction-static", "0.90",
    "--swa-full-tokens-ratio", "0.1",
    "--chunked-prefill-size", "8192",
    "--disable-shared-experts-fusion",
    "--tool-call-parser", "deepseekv4",
    "--reasoning-parser", "deepseek-v4",
]
```

评论区精华

无 review 评论，仅获得 HaiShaw 的批准。PR 描述中作者提供了两个模型（V4-Pro FP4 和 V4-Flash FP8）的加速比测试数据，验证了迁移后精度和吞吐量满足要求。

- 暂无高价值评论线程

风险与影响

- 风险：主要风险在于环境变量和启动参数的变更可能导致测试在特定硬件或软件组合下失败。例如，--attention-backend dsv4 依赖于 DSv4 特定的注意力实现，若 ROCm 环境缺少相关依赖可能出错。但作者已烟雾测试两个配置并通过。此外，工作流中去掉了 --skip-sglang-build，每次运行会重新编译 sglang，可能增加 CI 耗时，但确保与 main 分支代码一致，降低了因镜像过时而导致的隐藏问题。其他 AMD nightly 任务不受影响，因为 V4 任务独立运行。
- 影响：影响范围限于 AMD GPU 上的 DeepSeek-V4 nightly CI 测试。减少了维护专用镜像的负担，简化了 CI 配置。对用户无直接影响，但对 AMD 平台测试团队有正面维护收益。系统层面，工作流文件减少 64 行，变得简洁。测试脚本的环境变量调整使配置更贴近生产运行脚本（python/run_dsv4.sh）。
- 风险标记：环境变量变更可能导致兼容性问题，CI 步骤简化可能隐藏镜像依赖

关联脉络

- PR #26383 [AMD] Merge DeepSeek-V4 model and ROCm optimizations into main: 本 PR 的基础：DSv4 模型和 ROCm 优化合并到 main 后，才使 V4 nightly 可以使用标准镜像。