PR #34923 完整报告

sgl-project/sglang

Apply latest DeepEP branch

合并时间：2026-08-19 07:24

原文链接：http://prhub.com.cn/sgl-project/sglang/pull/34923

执行摘要

- 一句话：升级 DeepEP 至 0.1.1，动态抬高 NVSHMEM QP 深度并强制安装 NCCL 2.30.7
- 推荐动作：值得精读。这是一个小而完整的"依赖升级 → 运行时适配 → 部署配套"闭环样例，值得关注的设计点是 _set_nvshmem_qp_depth 的 max() 保护语义：既满足新版本下限要求，又尊重用户显式配置，避免"魔法值覆盖用户配置"的常见错误。团队应跟踪两件事：上游 DeepEP PR#10（恢复 SBO 测试）；考虑为 QP 深度逻辑补一个低成本单测（如 mock os.environ 验证 max() 计算），为移除的单测做回归补偿。

功能与动机

PR body 明确给出 DeepEP v2 的新约束："DeepEP v2 requires NCCL Gin from NCCL 2.30.7 and an NVSHMEM QP depth of at least twice the low-latency dispatch capacity plus two"，并指出 "For the DeepSeek V4 configuration with 1024 dispatch tokens per rank, the required QP depth is 2050, which is larger than NVSHMEM default of 1024"。也就是说，不升级依赖就拿不到 DeepEP v2；不装 NCCL 2.30.7 就无法使用 NCCL Gin 特性；不抬高 NVSHMEM_QP_DEPTH，低延迟模式下 NVSHMEM 队列深度不足，DeepSeek V4 这类大 dispatch 配置会面临通信初始化或运行期阻塞。

实现拆解

1. 源码适配：动态抬高 NVSHMEM QP 深度（python/sglang/srt/layers/moe/token_dispatcher/deepep.py，+15 行）。新增模块级常量 _NVSHMEM_QP_DEPTH_DEFAULT = 1024 与函数 _set_nvshmem_qp_depth()，算法为 max(current_qp_depth, 1024, 2 * (num_max_dispatch_tokens_per_rank + 1))。调用点位于 get_deepep_buffer() 的 low-latency 分支，且在 Buffer.get_low_latency_rdma_size_hint() 之前执行，并用 if not _is_npu 包裹，确保 zbal（NPU）路径与普通 DeepEP 路径完全不受影响。这样既满足 DeepEP v2 的下限要求，又不会覆盖用户显式设置的更大值。
2. 依赖升级（python/pyproject.toml，+1/-1）。sgl-deep-ep 从 0.1.0 升到 0.1.1，这是 DeepEP v2 的落地载体，也是整个 PR 的源头；PyPI 验证 0.1.1 为最新发布版。
3. CI 配套（scripts/ci/cuda/ci_install_dependency.sh，+12 行）。新增 install_nccl() 函数：CUDA major 为 13 时执行 nvidia-nccl-cu13==2.30.7 的 --force-reinstall --no-deps 安装，其他 CUDA 版本打印跳过日志；在 main() 中排在 install_cuda12_deepep_wheel 与 install_sglang 之后。原因是 CI 基础镜像自带的 NCCL 版本不满足 DeepEP v2 的 NCCL Gin 要求。

4. Docker 配套（docker/Dockerfile，+6 行）。新增 NCCL_VERSION=2.30.7 构建参数，在 CUDA 13 主镜像安装 sglang 的 RUN 步骤中追加条件安装：CUDA_VERSION 主版本为 13 时强制重装 nvidia-nccl-cu13==${NCCL_VERSION}；CUDA 12 分支保持原有卸载 cu13 wheel 的逻辑不变。这保证发布镜像与 CI 环境行为一致。
5. 测试调整（test/registered/4-gpu-models/test_deepseek_v3_cutedsl_4gpu.py，+4 行）。TestDummyWithSBO 整体加上 @unittest.skipIf(True, ...)，注释明确说明当前 DeepEP 不接受 --enable-single-batch-overlap 参数，待上游 sgl-project/DeepEP#10 修复后恢复。此外，提交历史中有一笔 chore: remove DeepEP NVSHMEM QP depth test，说明作者曾为 QP 深度逻辑写单测，最终移除，改由 4-gpu 集成测试兜底。

关键文件：

- python/sglang/srt/layers/moe/token_dispatcher/deepep.py（模块 MoE 调度；类别 source；类型 core-logic；符号 _set_nvshmem_qp_depth）：核心源码变更：新增 _set_nvshmem_qp_depth()，在 DeepEP 低延迟 buffer 初始化前动态抬高 NVSHMEM_QP_DEPTH，是 DeepEP v2 能否在 DeepSeek V4 配置下正常工作的关键。
- scripts/ci/cuda/ci_install_dependency.sh（模块 CI 脚本；类别 infra；类型 infrastructure）：CI 基础设施关键配套：新增 install_nccl()，在 CUDA 13 下强制重装 NCCL 2.30.7，否则 DeepEP v2 的 NCCL Gin 特性在 CI 上不可用。
- docker/Dockerfile（模块 部署镜像；类别 infra；类型 infrastructure）：发布镜像配套：CUDA 13 主镜像在安装 sglang 后强制安装 NCCL 2.30.7，保证用户拿到与 CI 一致的环境。
- python/pyproject.toml（模块 依赖配置；类别 config；类型 configuration）：依赖升级源头：sgl-deep-ep 从 0.1.0 升到 0.1.1，整个 PR 的出发点，确保 DeepEP v2 被正确打包分发。
- test/registered/4-gpu-models/test_deepseek_v3_cutedsl_4gpu.py（模块 集成测试；类别 test；类型 test-coverage）：测试配套：TestDummyWithSBO 因新版 DeepEP 不支持 --enable-single-batch-overlap 而临时跳过，待上游 PR#10 修复后恢复。

关键符号：_set_nvshmem_qp_depth, install_nccl

关键源码片段

python/sglang/srt/layers/moe/token_dispatcher/deepep.py

核心源码变更：新增 _set_nvshmem_qp_depth()，在 DeepEP 低延迟 buffer 初始化前动态抬高 NVSHMEM_QP_DEPTH，是 DeepEP v2 能否在 DeepSeek V4 配置下正常工作的关键。

```
# python/sglang/srt/layers/moe/token_dispatcher/deepep.py
# NVSHMEM 默认 QP 深度为 1024，但 DeepEP v2 要求至少为 low-latency dispatch
容量的两倍再加 2
_NVSHMEM_QP_DEPTH_DEFAULT = 1024


def _set_nvshmem_qp_depth(num_max_dispatch_tokens_per_rank: int) -> None:
    """在 DeepEP 低延迟 buffer 初始化前，把 NVSHMEM_QP_DEPTH 抬到足够深度。

    最关键的是 max() 语义：用户已显式设置的更大值会被保留，
```

```
    默认 1024 兜底，2 * (num_max_dispatch_tokens_per_rank + 1) 保证满足
    DeepEP v2 的下限要求，不会出现“魔法值覆盖用户配置”的问题。
    """
    min_qp_depth = 2 * (num_max_dispatch_tokens_per_rank + 1)
    current_qp_depth = int(
        os.environ.get("NVSHMEM_QP_DEPTH", _NVSHMEM_QP_DEPTH_DEFAULT)
    )
    os.environ["NVSHMEM_QP_DEPTH"] = str(
        max(current_qp_depth, _NVSHMEM_QP_DEPTH_DEFAULT, min_qp_depth)
    )


# get_deepep_buffer() 内，仅在 low-latency 模式开启且非 NPU（zbal）时执行，
# 必须在 Buffer.get_low_latency_rdma_size_hint() 之前完成设置，
# 否则 NVSHMEM 初始化时读到的是默认 1024。
if deepep_mode.enable_low_latency():
    assert num_max_dispatch_tokens_per_rank != -1
    assert num_experts != -1 and num_experts % group.size() == 0
    if not _is_npu:
        _set_nvshmem_qp_depth(num_max_dispatch_tokens_per_rank)
    num_rdma_bytes = max(
        Buffer.get_low_latency_rdma_size_hint(
            num_max_dispatch_tokens_per_rank,
            hidden_size,
            group.size(),
            num_experts,
        ),
        num_rdma_bytes,
    )
```

scripts/ci/cuda/ci_install_dependency.sh

CI 基础设施关键配套：新增 install_nccl()，在 CUDA 13 下强制重装 NCCL 2.30.7，否则 DeepEP v2 的 NCCL Gin 特性在 CI 上不可用。

```
# scripts/ci/cuda/ci_install_dependency.sh
# DeepEP v2 的 NCCL Gin 特性依赖 NCCL 2.30.7，而基础镜像可能自带其他版本，
# 因此 CUDA 13 环境下必须强制重装该 wheel；CUDA 12 不受影响
install_nccl() {
    if [ "$CU_MAJOR" = "13" ]; then
        $PIP_CMD install "nvidia-nccl-cu13==2.30.7" \
            --force-reinstall --no-deps $PIP_INSTALL_SUFFIX
    else
        echo "CUDA ${CU_MAJOR} does not require the NCCL Gin wheel"
    fi

    mark_step_done "${FUNCNAME[0]}"
}
```

评论区精华

本 PR 没有正式的 review 评论（review_comments_count = 0），有价值的信息集中在 issue 评论与 CI 重跑记录中：

- Fridge003 在 issue 评论中直接给出硬性要求："Please reinstall nccl 2.30.7 in ci_install_dependency.sh"，随后由 install_nccl() 落地，说明 NCCL 2.30.7 是 DeepEP v2 在 CUDA 13 CI 上跑通的硬前提。
- 4-gpu-gb300 上的 test_deepseek_v3_cutedsl_4gpu.py 首次 rerun 失败（❓ Run #32112125697），作者再次 /rerun-test 后通过（❓ Run #32192985954）。失败原因未在评论中深究，存在环境波动嫌疑，最终以通过收尾。
- 作者记录了 GB300 四卡实测：TestDSV4FlashFP4B200Balanced 全部 9 项测试通过（165.585 秒），GSM8K 0.965，平均 speculative accept length 1.9488，说明 QP 深度抬高没有带来精度或吞吐回退。
- NCCL 2.30.7 安装要求 (infra): 新增 install_nccl() 函数，CUDA 13 下强制重装 nvidia-nccl-cu13==2.30.7。
- 4-gpu-gb300 测试重跑稳定性 (testing): 第二次通过，作者接受结果；失败根因未在评论中深究。
- SBO 测试临时跳过 (testing): 暂时跳过，待上游 PR 合并后恢复；当前 SBO 功能无集成测试覆盖。

风险与影响

- 风险：
  1. 环境变量生效时机依赖 DeepEP 初始化顺序：NVSHMEM_QP_DEPTH 是在 Python 运行时写入 os.environ 的，如果 deep_ep 库在更早阶段（如 import 期）就读取该值，设置将无效。PR 的实测数据表明当前时机正确，但 get_deepep_buffer() 带有单例缓存（state.buffer 非空即提前返回），未来若出现其他先触发 buffer 创建并绕过该分支的调用路径，QP 深度配置会被静默跳过。
  2. QP 深度配置缺少自动化测试：专门的单测在提交中被移除，目前只有 4-gpu 集成测试兜底，门槛高、成本大，后续调整计算规则时不易及时发现回归。
  3. CUDA 13 强制重装 NCCL 2.30.7：CI 脚本与 Dockerfile 都用了 --force-reinstall --no-deps，会覆盖环境中已有 NCCL；若 CUDA 13 环境下其他组件依赖不同 NCCL 版本，可能引入兼容性冲突。CUDA 12 完全不受影响。
  4. 功能临时回退：TestDummyWithSBO 被整体跳过，单 batch overlap（SBO）能力在本次发布中失去集成测试覆盖，且恢复依赖 DeepEP 上游 PR#10，时间不确定。
  5. 提交可追溯性一般：最后一个提交仅写 "upd"，期间有两次 Merge branch 'main'，对后续回溯 commit 动机不够友好。
- 影响：
  - 用户侧：使用 DeepEP 低延迟模式的部署（典型如 DeepSeek V4）会获得正确抬高的 NVSHMEM_QP_DEPTH，避免因队列深度不足导致的 NVSHMEM 通信问题；由于 max() 保护，用户自定义的更大值不受影响。CUDA 13 用户安装或升级 sglang 后会强制落下 NCCL 2.30.7。

- 系统侧：CUDA 13 的 CI 与 Docker 主镜像同步变更，NCCL 版本成为 DeepEP v2 的隐式硬依赖；NPU（zbal）与普通 DeepEP 路径完全不受影响。
- 团队侧：带 release-highlight 标签，说明该变更进入重点发布清单；SBO 功能的测试缺口需要跟踪 DeepEP 上游 PR#10 的进展，并在合并后移除 skipIf 恢复覆盖。
- 风险标记：DeepEP 低延迟初始化路径变更，NVSHMEM 配置缺少单测，CUDA 13 强制重装 NCCL, SBO 功能临时回退

关联脉络

- PR #35371 [Spec] DFlash2: local convolution + candidate selector: DFlash2 的 draft 与 verify 同样走 DeepEP（deep_ep）路径，sgl-deep-ep 升到 0.1.1 后共享新的 NCCL 2.30.7 与 NVSHMEM QP 深度要求，需要联动回归。
- PR #35265 [Spec] Page-align the DFLASH decode KV reservation: DFLASH 同样依赖 DeepEP buffer 与 low-latency 路径，与本次 QP 深度适配处于同一 DeepEP 依赖链，版本升级对其有连带影响。