PR #34159 完整报告

sgl-project/sglang

Fix deterministic inference all-reduce for tp>1

合并时间：2026-08-09 18:08

原文链接：http://prhub.com.cn/sgl-project/sglang/pull/34159

执行摘要

- 一句话：TP>1 确定性推理固定 NCCL channel 数，消除 prefill/decode 差异
- 推荐动作：该 PR 值得精读，尤其适合对 NCCL 确定性机制感兴趣的工程师：它揭示了“算法固定”不等于“消息大小无关”这一深层细节，以及用 NCCL_MIN/MAX_NCHANNELS 同时 pin 来消除 channel 数影响的做法。建议后续补充针对 channel pin 的专项回归测试，并在文档中给出 SGLANG_DETERMINISTIC_NCCL_NCHANNELS 的调参指引。

功能与动机

PR body 指出：*--enable-deterministic-inference* 仅将 NCCL 算法固定为 *allreduce:tree* 并禁用自定义 /symmetric-memory all-reduce，这只保证“对于给定消息大小可复现”，并不保证“不随消息大小变化”：NCCL 会根据消息大小挑选 channel 数，而每个 channel 携带形状不同的树，因此同一元素在不同 batch 大小下归约顺序不同。结果 prefill 与 decode 对同一 token 的 logprob 仍可能不一致。作者明确说明“Pinning the channel count closes that gap.”（固定 channel 数可消除该差异）。

实现拆解

变更分两步，全部集中在启动配置路径：

1. 新增环境变量（python/sglang/srt/environ.py）：在 Envs 类的 deterministic inference 分组下新增 SGLANG_DETERMINISTIC_NCCL_NCHANNELS = EnvInt(8)，默认 8，并添加注释说明其用途——固定 channel 数以保证同一 token 的归约顺序与 batch 无关，同时允许用户调大以换回更多带宽。
2. 启动时 pin channel（python/sglang/srt/server_args.py）：在 _handle_deterministic_inference 的 CUDA 分支（既有 NCCL_ALGO=allreduce:tree、disable_custom_all_reduce = True、enable_torch_symm_mem = False 之后）读取该环境变量，并同时写入 NCCL_MIN_NCHANNELS 与 NCCL_MAX_NCHANNELS，将 NCCL channel 数固定为同一值。AMD/HIP 分支不变，因为 1-stage all-reduce 本身已天然确定性。
3. 同步更新日志：将 warning 文案改为提示 channel 数已被 pin，便于用户在日志中确认生效状态。
4. 测试配套：未新增单测文件，依赖现有 registered/attention/test_deterministic.py 与 registered/8-gpu-models/test_inkling_nvfp4_nightly.py 做回归验证；PR 评论中针对这两

个测试发起了 /rerun-test。

关键文件：

- python/sglang/srt/server_args.py（模块 启动配置；类别 source；类型 core-logic；符号 _handle_deterministic_inference）：核心修复点：在 _handle_deterministic_inference 的 CUDA 分支中将 NCCL_MIN_NCHANNELS 与 NCCL_MAX_NCHANNELS 固定为同一值，消除 NCCL 因消息大小选择不同 channel 数带来的归约顺序差异。
- python/sglang/srt/environ.py（模块 环境变量；类别 source；类型 core-logic；符号 SGLANG_DETERMINISTIC_NCCL_NCHANNELS）：新增 SGLANG_DETERMINISTIC_NCCL_NCHANNELS 环境变量，默认 8，为 channel pin 提供可调入口，直接决定 server_args.py 中写入的值。

关键符号：_handle_deterministic_inference

关键源码片段

python/sglang/srt/server_args.py

核心修复点：在 _handle_deterministic_inference 的 CUDA 分支中将 NCCL_MIN_NCHANNELS 与 NCCL_MAX_NCHANNELS 固定为同一值，消除 NCCL 因消息大小选择不同 channel 数带来的归约顺序差异。

```
# python/sglang/srt/server_args.py（head 版本，重组展示核心分支）

# 该函数在 --enable-deterministic-inference 开启时被调用，负责配置 NCCL 相关环境。
def _handle_deterministic_inference(self):
    # ... 前面的注意力后端 / radix cache 检查省略 ...

    # 仅 TP 规模大于 1 时，all-reduce 才会成为精度不确定性来源
    if self.tp_size > 1:
        if is_hip():
            # AMD：使用 1-stage all-reduce kernel，各 GPU 固定顺序读取全量数据，
            # 天然确定性，无需额外 pin channel 数。
            logger.info("AMD/ROCm: Using 1-stage all-reduce kernel (deterministic)")
        else:
            # CUDA：固定 NCCL 树算法，并禁用自定义 / symmetric-memory all-reduce
            os.environ["NCCL_ALGO"] = "allreduce:tree"
            self.disable_custom_all_reduce = True
            # symmetric-memory 路径存在字节阈值，会随 token 数切换，破坏确定性
            self.enable_torch_symm_mem = False

            # 关键修复：NCCL 默认按消息大小选择 channel 数，而每个 channel 的树形状不同，
            # 同一元素在 prefill / decode 不同 batch 大小下归约顺序不同。
            # 这里将 channel 数上下限同时固定为环境变量指定值（默认 8），
            # 使归约顺序与 batch 中的 token 数无关，达到 bit 级一致。
            nchannels = str(envs.SGLANG_DETERMINISTIC_NCCL_NCHANNELS.get())
            os.environ["NCCL_MIN_NCHANNELS"] = nchannels
            os.environ["NCCL_MAX_NCHANNELS"] = nchannels
```

```
            logger.warning(
                "NCCL_ALGO is set to 'allreduce:tree', the NCCL channel count is "
                "pinned, and custom and symmetric-memory all reduce are disabled "
                "for deterministic inference when TP size > 1."
            )
```

python/sglang/srt/environ.py

新增 SGLANG_DETERMINISTIC_NCCL_NCHANNELS 环境变量，默认 8，为 channel pin 提供可调入口，直接决定 server_args.py 中写入的值。

```
# python/sglang/srt/environ.py（head 版本，重组展示 deterministic 分组）

class Envs:
    # ... 其他环境变量省略 ...

    # Deterministic inference 相关开关
    SGLANG_ENABLE_DETERMINISTIC_INFERENCE = EnvBool(False)
    # AMD 平台上使用 1-stage all-reduce kernel（确定性、固定累加顺序）
    SGLANG_USE_1STAGE_ALLREDUCE = EnvBool(False)

    # CUDA 上固定 NCCL channel 数，使 all-reduce 对同一元素的归约顺序
    # 不随 batch 中的 token 数变化，prefill 与 decode 结果 bit 级一致。
    # 默认 8；在链路带宽允许时可以调大，以换回更高的 all-reduce 吞吐。
    SGLANG_DETERMINISTIC_NCCL_NCHANNELS = EnvInt(8)

    SGLANG_OPT_USE_CUSTOM_ALL_REDUCE_V2 = EnvBool(True)
    # ... 后续配置省略 ...
```

评论区精华

该 PR 没有进入人工 review 讨论（review 评论数为 0），评论区只有 CI bot 的重跑结果。作者两次发起 /rerun-test registered/attention/test_deterministic.py 与 registered/8-gpu-models/test_inkling_nvfp4_nightly.py：1-gpu-h100 与 8-gpu-h200 均通过，而 8-gpu-b200 连续两次失败。失败原因未在评论中解释，PR 最终在带 bypass-fastfail 标签的情况下被合并。

- CI 重跑验证与 B200 失败 (other): B200 失败原因未在评论区解释，PR 最终在带 bypass-fastfail 标签的情况下合并。

风险与影响

- 风险：主要风险如下：
- 缺少直接回归测试：没有新增针对 channel pin 的专项测试，现有 test_deterministic.py 和 nightly Inkling 测试是间接覆盖；如果未来有代码改动 NCCL 环境变量初始化顺序，可能回归且不易察觉。
- 性能风险：将 NCCL_MIN_NCHANNELS 与 NCCL_MAX_NCHANNELS 同时 pin 为 8，会禁用 NCCL 根据实际链路自动选择 channel 数；在支持更多通道的高带宽拓扑上可能损失 all-reduce 带宽。环境变量默认 8 是一个“够用”但未必最优的值。

- B200 CI 未通过：test_inkling_nvfp4_nightly.py 在 8-gpu-b200 上两次失败，未在评论区说明根因，可能是环境或 fp8 数值问题，需要关注后续 nightly 是否持续失败。
- 仅消除 all-reduce 源：PR body 明确 Qwen3-Next 在 tp=4 下 KL 散度只从 6.44e-4 降到 5.06e-4，说明 batch_invariant_ops 之外的 kernel 仍可能引入偏差，用户不应期望所有模型都达到 bit-identical。
- 环境变量全局生效：os.environ 的修改作用于整个 server 进程，会影响该进程内所有后续 NCCL 通信；但由于只在 deterministic 模式下设置，普通推理不受影响。
- 影响：影响范围集中在“确定性推理”这一实验性功能：
- 用户侧：启用 --enable-deterministic-inference 且 tp>1 的用户（如 Inkling 复现、KL 校验场景）获得 prefill 与 decode 一致的 logprob，这是核心收益。
- 性能侧：仅 deterministic 模式用户可能因 channel pin 而略降 all-reduce 带宽，可通过调大 SGLANG_DETERMINISTIC_NCCL_NCHANNELS 换回。普通推理路径完全不受影响。
- 团队侧：修复了确定性推理的一项正确性缺陷，使依赖 all-reduce 可复现性的测试与诊断更可靠；同时提供了一个可调性能开关，便于后续在“确定性”与“带宽”之间折中。
- 影响程度：低到中。改动仅 2 个文件、净增 11 行，且集中在启动配置初始化路径。
- 风险标记：缺少直接回归测试，默认 channel 数可能降低带宽，B200 CI 未通过，仅消除 all-reduce 不确定性源

关联脉络

- PR #32402 Switch inkling per-commit test to nvfp4: 该 PR 引入 Inkling-Small-NVFP4 的 CI 精度测试，而本次 PR 的验证数据（KL 散度 0.151 降至 0）正是基于该模型，二者共享相同的回归与 nightly 测试路径。