PR #32339 完整报告

sgl-project/sglang

[comm] Enable multi-node custom-AR v2 on a single NVLink clique

合并时间：2026-07-26 08:25

原文链接：http://prhub.com.cn/sgl-project/sglang/pull/32339

执行摘要

- 一句话：允许 custom all-reduce v2 在单 NVLink clique 跨节点使用
- 推荐动作：该 PR 展示了如何基于硬件拓扑（NVLink fabric clique）做运行时决策，对通信优化场景有参考价值。值得精读其中的 is_one_nvlink_clique 设计模式和模板化 AllReduceParams 的性能考量。对于使用 NVL72/MNNVL 集群的团队，此 PR 意义重大。

功能与动机

在 NVL72/MNNVL 架构中，同一 fabric clique 内的 GPU 共享地址空间，custom-AR v2 的 symmetric-memory 缓冲跨节点有效。原有 nnodes > 1 的硬编码限制不合理，需要替换为更精确的连通性检测。

实现拆解

1. NVLink fabric clique 检测 (custom_all_reduce_utils.py)：新增 _gpu_fabric_clique() 通过 NVML 获取本地 GPU 的 clusterUuid 和 cliqueId；is_one_nvlink_clique() 对所有 rank all-gather 这些标识，统一判定是否属于同一 clique。对 AMD HIP 直接返回 False，异常时安全回退。
2. v2 准入逻辑改造 (custom_all_reduce_v2.py)：修改 can_use_custom_all_reduce_v2()，支持 world-size 范围扩展至 2-16。跨节点时要求 is_one_nvlink_clique() 且分配器为 VMM-backed；同节点仍走原 NVLink 检测。引入 _is_vmm_backed_allocator() 辅助函数。
3. 移除全局禁用 (server_args.py)：删除 nnodes > 1 时强制关闭 v2 的代码段，将决策权下放给运行时 clique 检测。
4. 配置与 world-size 扩展：在 configs/custom_all_reduce_v2.py 添加 world_size=16 的 heuristic 和 multicast block 配置；kernel header 中 kMaxWorldSize 从 8 升至 16，并通过模板化 AllReduceParams 避免参数结构体膨胀。
5. 测试与基准配套：默认测试 world size 从 (2,4,8) 扩展到 (2,4,8,16)，仅在有足够 GPU 的环境下实际运行；benchmark 同步调整。

关键文件：

- python/sglang/srt/distributed/device_communicators/custom_all_reduce_utils.py（模块 分布式通信；类别 source；类型 core-logic；符号 _gpu_fabric_clique, is_one_nvlink_clique）：核心新增：fabric clique 检测函数 _gpu_fabric_clique 和 is_one_nvlink_clique，是多节点准入的判断基础。

- python/sglang/srt/distributed/device_communicators/custom_all_reduce_v2.py（模块 分布式通信；类别 source；类型 core-logic；符号 _is_vmm_backed_allocator）：准入函数 can_use_custom_all_reduce_v2 重写为支持多节点，新增 VMM-backed allocator 检查。
- python/sglang/srt/server_args.py（模块 服务配置；类别 source；类型 core-logic）：移除 nnodes > 1 时强制禁用 v2 的代码，将决策下放给运行时。
- python/sglang/srt/distributed/device_communicators/configs/custom_all_reduce_v2.py（模块 调优配置；类别 source；类型 core-logic）：为 world_size=16 添加 SM100 的 graph/eager heuristic 和 multicast block 配置。
- python/sglang/srt/distributed/device_communicators/custom_all_reduce.py（模块 分布式通信；类别 source；类型 core-logic）：更新注释说明多节点 v2 准入逻辑已迁移到 can_use_custom_all_reduce_v2。
- test/registered/kernels/ops/communication/test_custom_all_reduce.py（模块 通信测试；类别 test；类型 test-coverage）：默认测试 world size 扩展至 16，确保多节点场景覆盖。
- test/registered/kernels/benchmark/communication/bench_custom_all_reduce.py（模块 性能基准；类别 test；类型 test-coverage）：基准测试同步扩展至 world-size 16。
- python/sglang/kernels/jit/csrc/distributed/custom_all_reduce.cuh（模块 CUDA 内核；类别 other；类型 core-logic）：将 kMaxWorldSize 从 8 提升至 16，并可能模板化 AllReduceParams 以隔离参数结构体大小。
- python/sglang/kernels/jit/include/sgl_kernel/distributed/communicator.cuh（模块 CUDA 内核；类别 other；类型 core-logic）：与 custom_all_reduce.cuh 同步调整 kMaxWorldSize。

关键符号：_gpu_fabric_clique, is_one_nvlink_clique, can_use_custom_all_reduce_v2, _is_vmm_backed_allocator

关键源码片段

python/sglang/srt/distributed/device_communicators/custom_all_reduce_utils.py

核心新增：fabric clique 检测函数 _gpu_fabric_clique 和 is_one_nvlink_clique，是多节点准入的判断基础。

```
# custom_all_reduce_utils.py — 新增 fabric clique 检测

# NVML_GPU_FABRIC_STATE_COMPLETED: GPU 已加入其 NVLink fabric clique
_NVML_GPU_FABRIC_STATE_COMPLETED = 3


def _gpu_fabric_clique(device: torch.device):
    """返回 (cluster_uuid, clique_id) 或 None（未完成 fabric 初始化）。"""
    cuda_visible_devices = os.environ.get("CUDA_VISIBLE_DEVICES", None)
    if cuda_visible_devices:
```

```
        device_ids = list(map(int, cuda_visible_devices.split(",")))
    else:
        device_ids = list(range(torch.cuda.device_count()))
    handle = pynvml.nvmlDeviceGetHandleByIndex(device_ids[device.index])
    fabric = pynvml.c_nvmlGpuFabricInfo_v3_t()
    fabric.version = pynvml.nvmlGpuFabricInfo_v3
    pynvml.nvmlDeviceGetGpuFabricInfoV(handle, ctypes.byref(fabric))
    if fabric.state != _NVML_GPU_FABRIC_STATE_COMPLETED:
        return None
    return (bytes(fabric.clusterUuid), int(fabric.cliqueId))


@with_nvml_context
def is_one_nvlink_clique(
    group: torch.distributed.ProcessGroup, device: torch.device
) -> bool:
    """仅当所有 rank 的 GPU 属于同一个 NVLink fabric clique 时返回 True。"""
    if _is_hip:
        return False
    try:
        clique = _gpu_fabric_clique(device)
    except Exception as e:
        logger.warning(
            "GPU fabric clique query failed (%r); custom-AR stays intra-node.", e
        )
        clique = None
    # all-gather 确保任何 rank 失败即全体返回 False，避免集体不匹配
    world_size = dist.get_world_size(group=group)
    gathered: List[object] = [None] * world_size
    dist.all_gather_object(gathered, clique, group=group)
    if any(c is None for c in gathered):
        return False
    return len(set(gathered)) == 1
```

python/sglang/srt/distributed/device_communicators/custom_all_reduce_v2.py

准入函数 can_use_custom_all_reduce_v2 重写为支持多节点，新增 VMM-backed allocator 检查。

```
# custom_all_reduce_v2.py — 修正后的准入函数

def _is_vmm_backed_allocator(device: torch.device) -> bool:
    """本地 GPU 的缓存分配器是否使用 VMM (expandable_segments)。"""
    probe = torch.empty(1, dtype=torch.uint8, device=device)
    return is_vmm_pointer(probe.data_ptr())


def can_use_custom_all_reduce_v2(
    group: ProcessGroup,
```

```
    device: torch.device,
) -> bool:
    supported = list(range(2, 17))
    if dist.get_world_size(group=group) not in supported:
        return False
    # 多节点场景：必须是同一 NVLink clique 且分配器为 VMM-backed
    if not all(in_the_same_node_as(group, source_rank=0)):
        return is_one_nvlink_clique(group, device) and _is_vmm_backed_allocator(device)
    # 单节点场景：仍使用原来的 NVLink 全互联检测
    full_nvlink = can_use_custom_all_reduce_with_nvlink(
        group=group,
        device=device,
        supported_world_size=supported,
        cls_name="CustomAllReduceV2",
    )
    return full_nvlink is True
```

评论区精华

1. 性能关注点：DarkSharpness 担心 kMaxWorldSize 增大导致内核参数结构体变大，影响小 world-size 性能。作者回应 "good point. updated." 并通过 commit Template AllReduceParams on world size 实现模板化编译，消除性能退化。
2. 配置调优与 CI 覆盖：DarkSharpness 询问 world-16 配置是否在实际 MNNVL 上调优以及 CI 覆盖情况。作者回复附上详细 benchmark 数据表格，说明 4MB 以下不同 num_block 选项几乎无差异，16MB/64MB 时 nb32 最优故选用；CI 覆盖限于有足够 GPU 的环境。

- kMaxWorldSize 提升导致内核参数结构体性能担忧 (performance): 作者采纳建议，通过 commit 'Template AllReduceParams on world size' 实现 world-size 模板化编译，消除了性能退化风险。
- world-16 调优配置和 CI 覆盖 (performance): 作者回复详细 benchmark 数据，显示 nb32 在大消息尺寸下最优，故选用 32；CI 覆盖限于有足够 GPU 的环境，不会影响常规 CI 耗时。

风险与影响

- 风险：
  1. 兼容性风险：多节点检测依赖 NVML GpuFabricInfo，较老驱动或不支持 NVLink 的 GPU 可能无法返回有效信息。代码通过 try-except 捕获异常并回退到 NCCL，无崩溃风险。
  2. 性能风险：kMaxWorldSize 从 8 提升至 16 可能使内核参数结构体变大，但已通过模板化 AllReduceParams 隔离不同 world-size 的编译实例，避免对小集群造成性能退化。
  3. 功能安全：多节点路径要求 VMM-backed 分配器（expandable_segments），若不满足则自动回退，不会因句柄类型不匹配而失败。 - 影响：用户层面：在 NVL72/MNNVL 集群上，跨节点 TP group 自动启用 custom all-reduce v2，获得比 NCCL 更低的 all-reduce 延迟；无需用户手动配置或调整环境变量。单节点行为完全不变。系统层面：

减少了 NCCL 依赖，可能缓解多节点通信带宽瓶颈。world-size 上限提升至 16 覆盖更大的 fabric 域。团队层面：代码改动集中在 device_communicators 下，模块化清晰，维护成本较低。新增的 clique 检测函数可复用给其他通信原语。

- 风险标记：依赖 NVML GpuFabricInfo，VMM-backed 分配器要求，多节点回退安全

关联脉络

- 暂无明显关联 PR