PR #34450 完整报告

sgl-project/sglang

Raise PD zmq per-context socket cap via SGLANG_DISAGGREGATION_ZMQ_MAX_SOCKETS

合并时间：2026-08-12 06:27

原文链接：http://prhub.com.cn/sgl-project/sglang/pull/34450

执行摘要

- 一句话：新增 ZMQ 套接字上限配置，修复 PD 大规模连接失败
- 推荐动作：建议运维和 disaggregation 相关开发阅读。该 PR 揭示了一个非直觉的规模瓶颈（libzmq per-context 上限而非 OS fd 上限），并以最小改动提供可调配置。值得关注的决策：默认值取 16384（约 16 倍默认值）并留在 Envs 中统一管理；同时作者明确承认 socket 缓存无界，后续 LRU eviction 是真正的根治方向，读者可跟踪该后续工作。

功能与动机

PR body 指出：Prefill ranks 可以耗尽 libzmq per-context socket cap（默认 1023），即使 OS 文件描述符限制足够。KV manager 为每个 decode endpoint 缓存一个 PUSH socket 和一个 PAIR monitor socket，因此 16 个 decode replicas × 32 DP ranks 需要 1025 个 sockets，transfer workers 在 connect 时开始失败。这是一个连接容量缓解措施。

实现拆解

1. 新增环境变量定义

在 python/sglang/srt/environ.py 的 Envs 类中，于 PD disaggregation 配置段新增 SGLANG_DISAGGREGATION_ZMQ_MAX_SOCKETS = EnvInt(16384)，默认值 16384（约 libzmq 默认上限 1023 的 16 倍），供所有组件通过 envs...get() 读取。

2. KVManager 初始化时提升 context 上限

在 python/sglang/srt/disaggregation/common/conn.py 的 KVManager 初始化方法中，创建 self._zmq_ctx = zmq.Context() 后立即调用 self._zmq_ctx.set(zmq.MAX_SOCKETS, envs.SGLANG_DISAGGREGATION_ZMQ_MAX_SOCKETS.get())，使该 context 内缓存的 PUSH/PAIR 套接字不受默认限制。

3. CommonKVReceiver 类级 context 同步设置

对共享的 CommonKVReceiver._ctx 类属性（zmq.Context()）同样调用 _ctx.set(zmq.MAX_SOCKETS, ...)，确保复用接收端套接字时也享有新上限。

4. 测试与文档

没有新增测试或文档（PR body 明确说明），作者仅验证了语法编译；注释在 4 个 commit 中逐步精简。

关键文件：

- python/sglang/srt/environ.py（模块 环境配置；类别 source；类型 configuration；符号 SGLANG_DISAGGREGATION_ZMQ_MAX_SOCKETS）：定义新环境变量 SGLANG_DISAGGREGATION_ZMQ_MAX_SOCKETS，作为配置入口，默认值 16384
- python/sglang/srt/disaggregation/common/conn.py（模块 KV 传输；类别 source；类型 core-logic；符号 KVManager.init, CommonKVReceiver）：核心变更，两处应用 zmq.MAX_SOCKETS，解决大规模 PD 部署连接失败

关键符号：KVManager.init, CommonKVReceiver

关键源码片段

python/sglang/srt/disaggregation/common/conn.py

核心变更，两处应用 zmq.MAX_SOCKETS，解决大规模 PD 部署连接失败

```
# python/sglang/srt/disaggregation/common/conn.py

class KVManager:
    def __init__(self, ...):
        ...
        # bind zmq socket
        self._zmq_ctx = zmq.Context()
        # 将 per-context 套接字上限从 libzmq 默认 1023 提升为可配置值，
        # 因为该 manager 为每个 decode endpoint 缓存 PUSH 与 PAIR monitor 两个 socket，
        # 大规模部署（如 16 decode × 32 DP）会超过默认上限，导致 transfer worker 连接失败。
        self._zmq_ctx.set(
            zmq.MAX_SOCKETS, envs.SGLANG_DISAGGREGATION_ZMQ_MAX_SOCKETS.get()
        )
        self.rank_port, self.server_socket = get_zmq_socket_on_host(
            self._zmq_ctx, zmq.PULL, host=self.local_ip
        )
        ...

class CommonKVReceiver(BaseKVReceiver):
    _ctx = zmq.Context()
    # 共享接收端 context 同样应用新上限，保证所有复用 socket 不受 libzmq 默认 1023 约束
    _ctx.set(zmq.MAX_SOCKETS, envs.SGLANG_DISAGGREGATION_ZMQ_MAX_SOCKETS.get())
    _socket_cache = {}
    ...
```

评论区精华

PR 没有 review 评论。作者在 4 次提交中反复调整注释（从移除冗余注释到澄清动机），最终注释简洁说明“每个 decode endpoint 缓存 2 个 socket，大规模部署会超过默认上限”。Issue 评论仅有 /tag-and-rerun-ci 指令，无技术讨论。

- 暂无高价值评论线程

风险与影响

- 风险：
  - 文件描述符瓶颈转移：提升 ZMQ 上限后，OS 层面 ulimit -n 可能成为新的约束（ZMQ socket 底层也占用 fd），大规模部署仍需同步调高 fd 限制。
  - 内存增长：socket 缓存仍然无界，作者在 commit 中承认这是 minimal mitigation，后续需 LRU 驱逐来回收不活跃 socket。
  - 类级设置时机：CommonKVReceiver._ctx.set(...) 在类定义时执行，若 pyzmq 导入失败会在导入阶段直接报错；不过 conn.py 本就强依赖 zmq，风险低。
  - 缺少自动化测试：没有针对新环境变量的单元测试，但改动仅调整 context 选项，不改变消息语义，回归面小。
- 影响：
  - 用户侧：PD 大规模部署（如 16 decode × 32 DP）不再因 libzmq 默认 socket 上限发生连接失败，可用性提升。
  - 系统侧：ZMQ context 容量上限默认提高至 16384，内存占用与活跃连接数正相关；常规规模无感知。
  - 团队侧：新增 SGLANG 前缀环境变量（专家级配置），建议在 PD 部署文档中补充说明；当前 PR 未同步文档。
  - 影响范围：仅限 PD disaggregation 的 socket 创建路径（conn.py），不影响其他模块和模型输出。
  - 风险标记：socket 缓存无界，默认值依赖 OS fd 限制，缺少测试覆盖，类级 context 导入期设置

关联脉络

- PR #33807 [PD] Support pipeline-parallel prefill with Mooncake staging buffer: 同属 PD disaggregation 稳定性与功能增强，涉及 conn.py 等公共连接组件，本 PR 是对其大规模部署场景的补充缓解。
- PR #34341 [npu] [bugfix] Fix HiCache MHA backup for NPU: 同为 disaggregation 路径的 bugfix，表明 PD 拆分场景持续有可靠性修复，与本 PR 目标一致。