PR #30394 完整报告

sgl-project/sglang

fix: make automatic NUMA binding configurable

合并时间：2026-08-13 08:40

原文链接：http://prhub.com.cn/sgl-project/sglang/pull/30394

执行摘要

- 一句话：恢复自动 NUMA 绑定开关，支持通过环境变量禁用
- 推荐动作：值得快速精读，改动小而目标明确，是“配置项声明与实现失同步”的经典修复案例。重点关注三点：一是默认值 false -> true 背后的行为保持推理（关联 PR#19452）；二是门控位置放在显式 --numa-node 判断之后、NUMA 探测之前的优先级设计；三是测试对 SGLANG_NUMA_BIND_V2 正交性的 subTest 验证方式。

功能与动机

PR body 明确指出 SGLANG_AUTO_NUMA_BIND 变量 "remained declared and documented ... but its gate was accidentally dropped during the automatic NUMA configuration refactor, leaving the variable ineffective"，即环境变量和文档都存在，但控制逻辑在重构中丢失，用户即使设置 =0 也无法真正禁用自动 NUMA 绑定。作者在 issue 评论中进一步说明，PR#19452 起默认行为已变为自动绑定，本 PR 需要把默认值改为 true 才能忠实保持该行为。

实现拆解

变更入口在启动路径的 NUMA 解析函数 get_numa_node_if_available（python/sglang/srt/utils/numa_utils.py），配套环境变量定义、文档与测试同步更新，具体拆解如下：

1. 校准环境变量默认值（python/sglang/srt/environ.py）：将 SGLANG_AUTO_NUMA_BIND 默认值由 EnvBool(False) 改为 EnvBool(True)，与重构后实际的“未指定参数时自动 NUMA 绑定”行为对齐，保证显式未配置时行为不变。
2. 恢复环境变量门控（python/sglang/srt/utils/numa_utils.py）：在 get_numa_node_if_available 中、检查 server_args.numa_node 之后、NUMA 可用性探测之前插入 if not envs.SGLANG_AUTO_NUMA_BIND.get(): return None。这样 SGLANG_AUTO_NUMA_BIND=0 时直接跳过 _is_numa_available 与 _query_numa_node_for_gpu 的探测路径，worker 保持不绑定。显式 --numa-node 的优先级不受影响。
3. 补齐单元测试（test/registered/utils/test_numa_utils.py）：新增 test_auto_numa_bind_enabled_by_default（验证默认开启）、test_returns_explicit_numa_node_when_auto_bind_disabled（验证禁用时显式节点仍生

效）、test_auto_bind_disabled_skips_numa_detection（验证禁用时跳过 NUMA 探测，并用 subTest 覆盖 SGLANG_NUMA_BIND_V2 两种取值证明两个变量正交）；既有自动探测用例补充 SGLANG_AUTO_NUMA_BIND=1 环境设定。测试注册到 CPU CI base-a-test-cpu 和 CUDA CI 4-gpu-gb300 / 4-gpu-b200。

4. 更新文档（docs/docs/references/environment_variables.mdx）：重写 SGLANG_AUTO_NUMA_BIND 说明，明确"未指定 --numa-node 时自动检测 GPU 所在 NUMA 节点并绑定，设为 false 则不绑定，显式 --numa-node 优先"；同时澄清 SGLANG_NUMA_BIND_V2 只选择实现（pre-launch numactl wrapper 或 worker 内绑定）、不控制开关，避免两个变量职责混淆。

关键文件：

- python/sglang/srt/utils/numa_utils.py（模块 NUMA 配置；类别 source；类型 core-logic；符号 get_numa_node_if_available）：核心控制流变更：在 get_numa_node_if_available 中恢复 envs.SGLANG_AUTO_NUMA_BIND 门控，决定启动路径是否自动探测并绑定 NUMA 节点。
- python/sglang/srt/environ.py（模块 环境配置；类别 source；类型 configuration；符号 SGLANG_AUTO_NUMA_BIND）：环境变量默认值修复的源头：SGLANG_AUTO_NUMA_BIND 默认值从 false 改为 true，与 PR#19452 之后的实际自动绑定行为对齐。
- test/registered/utils/test_numa_utils.py（模块 NUMA 测试；类别 test；类型 test-coverage；符号 test_auto_numa_bind_enabled_by_default, test_returns_explicit_numa_node_when_auto_bind_disabled, test_auto_bind_disabled_skips_numa_detection）：新增 3 个测试用例覆盖默认启用、禁用时显式节点优先、禁用时跳过 NUMA 探测三态，是本次修复正确性的主要验证载体。
- docs/docs/references/environment_variables.mdx（模块 文档；类别 other；类型 documentation）：同步更新 NUMA 相关环境变量的语义与默认值表格，澄清 SGLANG_AUTO_NUMA_BIND 与 SGLANG_NUMA_BIND_V2 的职责边界。

关键符号：get_numa_node_if_available

关键源码片段

python/sglang/srt/utils/numa_utils.py

核心控制流变更：在 get_numa_node_if_available 中恢复 envs.SGLANG_AUTO_NUMA_BIND 门控，决定启动路径是否自动探测并绑定 NUMA 节点。

```
def get_numa_node_if_available(server_args: ServerArgs, gpu_id: int) -> Optional[int]:
    """
    返回给定 GPU id 对应的 NUMA 节点；若 server_args 未显式配置，
    则根据 SGLANG_AUTO_NUMA_BIND 决定是否自动探测并绑定。
    """
    # 优先级 1：显式 --numa-node 永远优先，不经过环境变量门控
    if server_args.numa_node is not None:
        return server_args.numa_node[gpu_id]

    # 优先级 2：用户显式禁用自动绑定（SGLANG_AUTO_NUMA_BIND=0）时，
```

```
    # 跳过 NUMA 可用性探测与 GPU -> NUMA 节点查询，保持 worker 不绑定
    if not envs.SGLANG_AUTO_NUMA_BIND.get():
        return None

    # 优先级 3：默认路径，自动探测 GPU 所在 NUMA 节点
    if _is_numa_available():
        queried_numa_node = _query_numa_node_for_gpu(gpu_id)
        if len(queried_numa_node) == 0:
            return None
        if len(queried_numa_node) > 1:
            # 单 GPU 可能对应多个 NUMA 节点，当前取第一个；
            # 尚无硬件配置出现多于一个节点，因此仅告警不报错
            logger.warning(
                f"Multiple NUMA nodes found for GPU {gpu_id}: {queried_numa_node}. Using the
                first one."
            )
        return queried_numa_node[0]
    return None
```

test/registered/utils/test_numa_utils.py

新增 3 个测试用例覆盖默认启用、禁用时显式节点优先、禁用时跳过 NUMA 探测三态，是本次修复正确性的主要验证载体。

```
class TestGetNumaNodeIfAvailable(unittest.TestCase):
    def _make_server_args(self, numa_node=None):
        args = MagicMock()
        args.numa_node = numa_node
        return args

    # 默认行为：未显式设置环境变量时，自动绑定处于开启状态
    def test_auto_numa_bind_enabled_by_default(self):
        with patch.dict(os.environ, {}, clear=True):
            self.assertTrue(envs.SGLANG_AUTO_NUMA_BIND.get())

    # 禁用自动绑定后，显式 --numa-node 仍然优先
    @patch.dict(os.environ, {"SGLANG_AUTO_NUMA_BIND": "0"})
    def test_returns_explicit_numa_node_when_auto_bind_disabled(self):
        args = self._make_server_args(numa_node=[2, 3, 0, 1])
        self.assertEqual(get_numa_node_if_available(args, 0), 2)
        self.assertEqual(get_numa_node_if_available(args, 3), 1)

    # 禁用自动绑定时，跳过 NUMA 可用性与 GPU 节点探测
    @patch("sglang.srt.utils.numa_utils._query_numa_node_for_gpu")
    @patch("sglang.srt.utils.numa_utils._is_numa_available")
    def test_auto_bind_disabled_skips_numa_detection(self, mock_avail, mock_query):
        args = self._make_server_args(numa_node=None)
        # 同时验证 SGLANG_NUMA_BIND_V2 两种取值下开关均生效
        for bind_v2 in ("0", "1"):
            with self.subTest(bind_v2=bind_v2), patch.dict(
```

```
                os.environ,
                {"SGLANG_AUTO_NUMA_BIND": "0", "SGLANG_NUMA_BIND_V2": bind_v2},
            ):
                self.assertIsNone(get_numa_node_if_available(args, 0))
        mock_avail.assert_not_called()
        mock_query.assert_not_called()
```

评论区精华

该 PR 没有实质性 review 争论，合并者 Fridge003 直接批准。关键澄清来自作者 lluki 在 issue 评论中的说明："Starting with 19452, the default behavior became to NUMA pin if no options are given. My PR keeps that behavior intact: Pin, if no option is given. This is why we have to change the default of SGLANG_AUTO_NUMA_BIND to true." 这解释了默认值看似翻转、实为修复"声明与实际行为不一致"的设计决策。合并前由 maintainer 触发了一次针对 test_numa_utils.py 的 CI 重跑，4-gpu-gb300 与 4-gpu-b200 均通过；gemini-code-assist 自动 review 无实质反馈。

- 暂无高价值评论线程

风险与影响

- 风险：
  - 行为变化风险：对从未设置过该变量的用户完全无感知；但凡是此前设置了 SGLANG_AUTO_NUMA_BIND=0 的用户，修复后会自动绑定真正失效，可能带来预期的性能变化或意外回归。
  - 启动路径变更：改动位于 get_numa_node_if_available（python/sglang/srt/utils/numa_utils.py），这是每个 GPU worker 启动时都会经过的 NUMA 决策路径，属于启动关键路径，但分值路径清晰且被新测试覆盖。
  - 兼容性：SGLANG_NUMA_BIND_V2 与 SGLANG_AUTO_NUMA_BIND 职责正交，前者的实现选择语义不受影响；测试专门覆盖了 SGLANG_NUMA_BIND_V2=0/1 两种组合。
  - 测试覆盖：新增测试覆盖了默认启用、显式禁用、显式 --numa-node 优先三态，但未覆盖"禁用 + 非 CUDA 环境"或"禁用 + NUMA 不可用"的组合，这两条路径逻辑上由短路返回保证，风险很低。
  - 影响：影响范围覆盖所有通过环境变量管理 NUMA 绑定的 SGLang 多 GPU 部署：恢复了已声明配置项的实际功能，默认行为保持不变，不需要用户修改现有启动命令；对文档读者而言，两个 NUMA 相关环境变量的职责边界（开关 vs 实现选择）更加清晰。对团队而言，这是一次低成本、低风险的配置一致性修复，且补齐了回归测试，防止后续重构再次丢失门控逻辑。
  - 风险标记：配置项功能修复，启动路径变更，默认值调整，有测试覆盖

关联脉络

- PR #19452 Automatic NUMA configuration refactor（作者在 issue 评论中提及）：该 PR 将默认行为改为未指定参数时自动 NUMA 绑定，但漏掉了 SGLANG_AUTO_NUMA_BIND 门控逻辑；本 PR 是其功能收尾修复，默认值 true 正是为

保持 19452 后的行为而设定。