PR #29403 完整报告

sgl-project/sglang

feat: sync npu nightly test improvements from Ascend testcases

合并时间：2026-07-06 22:41

原文链接：http://prhub.com.cn/sgl-project/sglang/pull/29403

执行摘要

- 一句话：NPU 夜间测试增强：重试、协调、参数调优
- 推荐动作：值得关注的设计决策：
  - 数据集感知的阈值计算方法：通过定义题目数量和波动容忍度，为不同评估集设置差异化的准确率门槛，减少误报。
  - 多次运行取平均的设计：在测试基类中提供 n_runs 和 run_accuracy_multiple，方便快速实现多次评估。
  - 环境变量注入和 CI 标志移除：为性能基准测试提供更精确的环境控制。

建议阅读 test_npu_accuracy_utils.py 和 test_npu_performance_utils.py 中的工具函数设计。注意 review 中未解决的配置问题（tp-size 不匹配、推理解析器缺少值等），建议在后续 PR 中修复。

功能与动机

根据 PR body，此 PR 的目的是从 Ascend 测试用例同步改进，包括更新 nightly-test-npu.yml 的 qwen3_6_27b 配置（2p→1p），在准确性工具中添加多次运行和重试支持，在多节点工具中添加 ConfigMap 操作函数，以及在性能工具中添加环境变量参数等。最终目标是提升 NPU 夜间测试的覆盖率和可靠性。

实现拆解

1. CI 工作流调整(nightly-test-npu.yml)：将 qwen3_6_27b_w8a8 测试从双卡（2p）改为单卡（1p），同时保持 a2 runner 标签。
2. 准确性测试增强(test_npu_accuracy_utils.py)：
   - 新增 DATASET_QUESTION_COUNTS 和 DATASET_FLUCTUATION 常量，用于基于题目数量的绝对波动阈值计算。
   - 提供 get_accuracy_threshold() 和 get_max_retries() 函数，使不同数据集使用不同的容忍度和重试次数。
   - 在 TestNpuAccuracyTestCaseBase 中添加 n_runs 类属性和 run_accuracy_multiple() 方法，支持多次运行并报告平均准确率。
   - 删除废弃的 MMMU_LOCAL_PATH 和 _get_dataset_args() 方法。
3. 多节点测试工具新增(test_npu_multi_node_utils.py)：

- 添加 ACTIVE_TEST_CLASS 和 SERVICE_EXIT_WAIT_SECONDS 常量。
- 新增 upsert_configmap_field_strict()：使用 Kubernetes API 更新 ConfigMap 字段，若 ConfigMap 不存在则抛出 RuntimeError。
- 新增 wait_for_prefill_decode_exit()：定期查询 ConfigMap，插入退出信号，并等待预填 / 解码节点退出（最多等待 ROUTER_CONFIGMAP_TIMEOUT 秒）。

4. 性能测试工具改进(test_npu_performance_utils.py)：
   - 为 run_bench_serving() 函数添加可选的 env 参数，允许调用方传入自定义环境变量字典（直接传递给 subprocess.Popen）。
   - 在三个测试基类中增加 pop_sglang_is_in_ci_for_gsp 标志，当为 True 且数据集为 generated-shared-prefix 时，在运行基准测试前移除 SGLANG_IS_IN_CI 环境变量。
   - 在 TestNpuPerfMultiNodePdSepTestCaseBase.tearDownClass 中增加轮询等待子进程退出的逻辑，并添加超时警告。
5. 测试用例同步调整：大量准确性 / 性能测试用例更新参数（如 --max-mamba-cache-size、--mem-fraction-static、--cuda-graph-bs、--prefill-delayer-* 等），添加 --reasoning-parser 和 --tool-call-parser，将类名从 TestAscend 改为 TestNpu，删除空置的 setUpClass/tearDownClass 等方法。新增 test_npu_qwen3_6_27b_w8a8_1p_in64k_out1k_50ms.py 单卡性能测试。

关键文件：

- python/sglang/test/ascend/e2e/test_npu_accuracy_utils.py（模块 准确率工具；类别 test；类型 test-coverage；符号 get_accuracy_threshold, get_max_retries, _get_dataset_args, run_accuracy_multiple）：准确性测试基础工具，新增数据集感知的阈值计算和重试逻辑，是本次变更的核心文件
- test/registered/ascend/performance/qwen3_6_27b/test_npu_qwen3_6_27b_w8a8_1p_in64k_out1k_50ms.py（模块 Qwen3.6 性能测试；类别 test；类型 test-coverage；符号 TestNPUQwen3_6_27B_2P_In64k_Out1k_50ms, test_npu_qwen3_6_27b_2p_in64k_out1k_50ms）：新增 Qwen3.6-27B-w8a8 单卡性能测试文件，配置了完整的启动参数和环境变量
- python/sglang/test/ascend/e2e/test_npu_multi_node_utils.py（模块 多节点工具；类别 test；类型 test-coverage；符号 upsert_configmap_field_strict, wait_for_prefill_decode_exit）：多节点测试工具，新增 upsert_configmap_field_strict 和 wait_for_prefill_decode_exit，用于 Kubernetes 环境下的测试协调
- test/registered/ascend/performance/qwen3_6_27b/test_npu_qwen3_6_27b_2p_in64k_out1k_prefix90_50ms.py（模块 前缀缓存测试；类别 test；类型 test-coverage；符号 TestNPUQwen3_6_27B_2P_In64k_Out1k_Prefix90_50ms, TestNPUQwen3_6_27B_1P_In64k_Out1k_Prefix90_50ms）：调整了前缀缓存性能测试类从 2p 改为 1p，同时更新了 server args 参数，并添加了推理工具调用解析器
- python/sglang/test/ascend/e2e/test_npu_performance_utils.py（模块 性能工具；类别 test；类型 test-coverage；符号 run_bench_serving, pop_sglang_is_in_ci_for_gsp）：性能测试工具，新增 env 参数、pop_sglang_is_in_ci_for_gsp 标志和进程退出等待逻辑

关键符号：get_accuracy_threshold, get_max_retries, run_accuracy_multiple, upsert_configmap_field_strict, wait_for_prefill_decode_exit, run_bench_serving, assert_metrics, TestNpuAccuracyTestCaseBase.run_accuracy, TestNpuPerformanceTestCaseBase.run_throughput, TestNpuPerfMultiNodePdMixTestCaseBase.run_throughput, TestNpuPerfMultiNodePdSepTestCaseBase.tearDownClass

关键源码片段

python/sglang/test/ascend/e2e/test_npu_accuracy_utils.py

准确性测试基础工具，新增数据集感知的阈值计算和重试逻辑，是本次变更的核心文件

```
# 数据集题目总数和允许的波动（绝对题数）
DATASET_QUESTION_COUNTS = {
    "aime25": 30,
    "aime26": 30,
    "gpqa_diamond": 198,
}
DATASET_FLUCTUATION = {
    "aime25": 2,
    "aime26": 2,
    "gpqa_diamond": 5,
}
MAX_RETRY_COUNT = 3 # 最大重试次数

def get_accuracy_threshold(datasets, baseline_accuracy):
    """根据数据集计算准确率阈值。

    对于有定义波动的数据集（如 aime, gpqa），使用绝对题数容忍度；
    其他数据集（如 mmmu）使用百分比容忍度。
    """
    dataset = datasets[0] if datasets else None
    if dataset in DATASET_FLUCTUATION and dataset in DATASET_QUESTION_COUNTS:
        # 波动比例 = 允许波动的题数 / 总题数
        fluctuation = DATASET_FLUCTUATION[dataset] / DATASET_QUESTION_COUNTS[dataset]
        return baseline_accuracy - fluctuation
    # 默认使用 99% 容忍度
    return baseline_accuracy * ACCURACY_TOLERANCE

def get_max_retries(datasets):
    """返回准确性测试的最大重试次数。

    gpqa 和 aime 数据集支持最多 MAX_RETRY_COUNT 次重试；
    mmmu 等其他数据集只执行 1 次（不重试）。
    """
    dataset = datasets[0] if datasets else None
    if dataset in DATASET_FLUCTUATION:
        return MAX_RETRY_COUNT
```

```
        return 1

    def assert_metrics(self, metrics):
        if not metrics:
            raise Exception("No metrics obtained from benchmark")
        if self.accuracy is not None:
            # 使用计算得到的阈值，而非固定的比例
            threshold = get_accuracy_threshold(self.datasets, self.accuracy)
            dump_metric("accuracy", float(metrics["accuracy"]),
                        labels={"test_case": self.__class__.__name__, "type": "accuracy"})
            dump_metric("accuracy_baseline", float(self.accuracy),
                        labels={"test_case": self.__class__.__name__, "type": "accuracy"})
            self.assertGreaterEqual(
                float(metrics["accuracy"]),
                threshold,
                f"Accuracy check failed. Expected >= {threshold}, Got: {metrics['accuracy']}",
            )
```

test/registered/ascend/performance/qwen3_6_27b/test_npu_qwen3_6_27b_w8a8_1p_in64k_out1k_50ms.py

新增 Qwen3.6-27B-w8a8 单卡性能测试文件，配置了完整的启动参数和环境变量

```
# 注意：此测试文件名及类名标记为 1p（单卡），但 --tp-size 设置为 2（双卡），
# 这是一个已知的配置冲突，可能在运行时导致错误或使用错误并行度。
QWEN3_6_27B_64K_1K_OTHER_ARGS = [
    "--tp-size",
    2, # 与单卡预期不符，应为 1
    "--nnodes",
    1,
    "--attention-backend",
    "ascend",
    "--device",
    "npu",
    "--chunked-prefill-size",
    -1,
    "--max-prefill-tokens",
    48000,
    "--disable-radix-cache",
    "--trust-remote-code",
    "--max-running-requests",
    6,
    "--max-mamba-cache-size",
    16,
    "--mem-fraction-static",
    0.6,
    "--cuda-graph-bs",
    1, 2, 4, 5, 6,
    "--quantization",
    "modelslim",
```

```
    "--dtype",
    "bfloat16",
    "--mamba-ssm-dtype",
    "bfloat16",
    "--speculative-algorithm",
    "NEXTN",
    "--speculative-num-steps",
    3,
    "--speculative-eagle-topk",
    1,
    "--speculative-num-draft-tokens",
    4,
    "--reasoning-parser",
    "qwen3",
    "--tool-call-parser",
    "qwen3_coder",
]

class TestNPUQwen3_6_27B_2P_In64k_Out1k_50ms(TestNpuPerformanceTestCaseBase):
    """Qwen3.6-27B-w8a8 单卡性能测试（但类名中的 2P 与意图 1P 矛盾）"""
    benchmark_tool = BENCHMARK_TOOL_DEFAULT
    dataset_type = AISBENCHMARK_DATASET_DEFAULT
    model = QWEN3_6_27B_W8A8_MODEL_PATH
    other_args = QWEN3_6_27B_64K_1K_OTHER_ARGS
    envs = QWEN3_6_27B_64K_1K_ENVS
    dataset_name = "random"
    max_concurrency = 6
    num_prompts = 12
    input_len = 64000
    output_len = 1000
    random_range_ratio = 1
    tpot = 50
    output_token_throughput = 57.85 # 期望的 output token throughput

    def test_npu_qwen3_6_27b_2p_in64k_out1k_50ms(self):
        self.run_throughput()
```

python/sglang/test/ascend/e2e/test_npu_multi_node_utils.py

多节点测试工具，新增 upsert_configmap_field_strict 和 wait_for_prefill_decode_exit，用于 Kubernetes 环境下的测试协调

```
# 以严格模式更新 ConfigMap 字段：若 ConfigMap 不存在则抛出异常
def upsert_configmap_field_strict(name: str, namespace: str, key: str, value: str):
    from kubernetes.client.rest import ApiException
    k8s_api = get_k8s_api()
    patch = {"data": {key: value}}
    try:
        k8s_api.patch_namespaced_config_map(name=name, namespace=namespace, body=patch)
        logger.info(f"Upserted ConfigMap {name}: {key}={value}")
```

```
        except ApiException as e:
            if e.status == 404:
                raise RuntimeError(f"ConfigMap {name} does not exist in namespace {namespace}")
            logger.error(f"Failed to upsert ConfigMap {name}: {e}")
            raise

    # 等待预填 / 解码节点退出：通过 ConfigMap 发送信号并等待节点响应
    def wait_for_prefill_decode_exit(
        key: str,
        value: str,
        timeout: int = ROUTER_CONFIGMAP_TIMEOUT,
        poll_interval: int = 15,
    ):
        start_time = time.time()
        while time.time() - start_time < timeout:
            configmap = query_configmap(CONFIGMAP_NAME, NAMESPACE)
            if not configmap or not configmap.data:
                logger.info(f"ConfigMap data is not available yet, waiting for {poll_interval}s...")
                time.sleep(poll_interval)
                continue
            existing_value = configmap.data.get(key)
            upsert_configmap_field_strict(CONFIGMAP_NAME, NAMESPACE, key, value)
            if existing_value is not None:
                logger.info("%s already set (%s), waiting %ds for prefill/decode to exit ...",
                            key, existing_value, SERVICE_EXIT_WAIT_SECONDS)
                time.sleep(SERVICE_EXIT_WAIT_SECONDS)
            else:
                logger.info("%s set for the first time (%s)", key, value)
            return
        # 注意：超时时函数静默返回 None，未抛出错误
```

评论区精华

Review 中 gemini-code-assist[bot] 指出了四个关键问题：

- (critical) 推理解析器参数缺少值：在 test_npu_deepseek_v3_2_8p_aime25.py 中 --reasoning-parser 后未提供值，会导致服务器启动失败。
- (high) 单卡测试配置冲突：新文件 test_npu_qwen3_6_27b_w8a8_1p_in64k_out1k_50ms.py 意图为单卡（1p）测试，但 --tp-size 设置为 2。
- (high) 环境变量覆盖风险：run_bench_serving 的 env 参数直接传给 subprocess.Popen，会完全替换子进程环境，导致 PATH 等关键变量丢失。
- (medium) 等待函数超时处理缺失：wait_for_prefill_decode_exit 在超时时仅返回 None，未抛出错误；同时日志信息硬编码 15s 而非使用 poll_interval。

作者对 tp-size 评论回复 "a regular change"，但最终代码中该值仍为 2，表明该问题未解决。其他评论未获得回复。PR 已合并，这些 risk 仍存在。

- 推理解析器参数缺少值 (correctness): 未修复，PR 已合并，问题仍存在。

- 单卡测试配置冲突（--tp-size）（correctness）：已讨论但未修改，PR 合并时保留冲突。
- 环境变量覆盖风险 (performance)：未回应，PR 合并时保留风险。当前调用场景已复制 os.environ 再修改，但函数接口设计仍有隐患。
- 等待函数超时处理缺失 (correctness)：未回应，PR 合并时保留问题。

风险与影响

- 风险：
  - 测试配置错误可能导致假失败 / 假通过：部分测试用例的参数（如 --tp-size 与单卡矛盾）未修正，可能在实际运行时报错或使用错误并行度，导致性能数据失真或测试失败。
  - 环境变量覆盖风险：run_bench_serving 直接传递 env 会替换整个环境，若调用方设置自定义环境但未包含系统关键变量（如 PATH），子进程可能无法运行。当前仅 pop_sglang_is_in_ci_for_gsp 场景使用 env，该场景已复制 os.environ 并移除单个变量，风险较低，但函数接口设计仍存在安全隐患。
  - 超时沉默失败：wait_for_prefill_decode_exit 超时时静默返回，可能导致多节点测试在等待阶段卡住或错误认为节点已退出，增加调试难度。
  - 大范围测试重构可能引入回归：涉及 37 个文件，大量类名和方法签名调整，缺少对应的回归测试覆盖。
- 影响：
  - 用户影响：仅影响 NPU（Ascend）平台的内部测试团队，生产环境用户无直接感知。
  - 系统影响：NPU 夜间测试的准确性和性能测试流程将更稳定、灵活，支持多次运行取平均和重试机制，降低假阳性。
  - 团队影响：NPU 测试维护者获得更强大的工具函数，减少了手动处理 ConfigMap 和轮询退出等操作，提升测试编写效率。但存在上述风险点，需要后续修复。
  - 风险标记：测试配置不一致，环境变量覆盖，超时静默失败，无回归测试

关联脉络

- PR #30047 Bugfix qwen prefix cache circumstances: 修复 NPU 前缀缓存相关 bug，本 PR 调整了前缀缓存测试的参数（如 --max-mamba-cache-size），两者关联。
- PR #30048 [XPU] Unbreak stage-b: re-add --disable-decode-cuda-graph, quarantine EAGLE3 parity: 同属 CI 稳定性改进，与本 PR 目标一致。
- PR #30186 Clean up ServerArgs post-init dispatch: 重构服务器参数调度，本 PR 大量测试用例调整了 server args（如 --reasoning-parser、--tool-call-parser），与参数清理相关。