PR #26182 完整报告

sgl-project/sglang

Fix Req array token-id concatenation

合并时间：2026-06-07 10:59

原文链接：http://prhub.com.cn/sgl-project/sglang/pull/26182

执行摘要

- 一句话：修复 Req token-id array 拼接错误
- 推荐动作：值得精读。特别是讨论中关于类型归一化策略的权衡（在 Req 内部转换 vs 调用者转换），以及 array 在序列操作中的性能优势。此外，custom_logit_processor 的优化展示了如何避免不必要的数据复制。

功能与动机

PR body 指出：PR #25098 迁移了 scheduler token-id 存储到 array.array("q")，但一些 scheduler/cache 路径仍然假定 list 拼接。当请求有现有生成 token 进入下一轮 prefill 时，通过 Req.init_next_round_input 会触发 TypeError: can only concatenate list (not "array.array") to list。

实现拆解

1. 移除 Req 内部的强制转换：在 Req.__init__ 中，将 self.origin_input_ids = array("q", origin_input_ids) 改为 self.origin_input_ids = origin_input_ids，同样移除对 origin_input_ids_unpadded 的强制转换，改为信任调用者已传入 array（见 schedule_batch.py）。
2. 更新所有直接构造 Req 的调用点：包括 bench_one_batch.py、test_forward_split_prefill.py、test_schedule_policy.py、test_custom_logit_processor.py 等，确保传入 array("q", ...) 而不是 list。
3. 优化 custom_logit_processor 中的 n-gram 比较：在 DeepseekOCRNoRepeatNGramLogitProcessor.__call__ 中，将 sequence 保持为 array，将 current_prefix 从 tuple 改为 array("q")，避免不必要的 PyList 切片和逐元素比较开销（Jialin 建议）。
4. 更新测试辅助函数：为 test_schedule_policy.py 新增 _make_req helper，为 test_custom_logit_processor.py 更新 mock Req，使其 origin_input_ids 和 output_ids 返回 array。
5. 最终提交：由 merrymercy 提交 "Convert direct Req list callers to arrays"，确保所有遗留调用点均完成转换。

关键文件：

- python/sglang/srt/managers/schedule_batch.py（模块 调度批处理；类别 source；类型 core-logic；符号 Req.init, Req.all_ids）：核心变更文件。移除了 Req.__init__中对 origin_input_ids 和 origin_input_ids_unpadded 的强制 array 转换，改为信任调用者已传入 array，统一了类型约定。
- python/sglang/srt/sampling/custom_logit_processor.py（模块 采样；类别 source；类型 performance；符号 DeepseekOCRNoRepeatNGramLogitProcessor.call）：优化 n-gram 比较逻辑，将 sequence 和 current_prefix 从 tuple 改为 array，避免 PyList 切片和逐元素比较，提升性能。
- python/sglang/bench_one_batch.py（模块 基准测试；类别 source；类型 dependency-wiring；符号 prepare_inputs_for_correctness_test, prepare_synthetic_inputs_for_latency_test）：修改 Req 构造调用点，传入 array("q", ...) 而非 list，是主要调用路径之一。
- test/manual/test_schedule_policy.py（模块 调度策略；类别 test；类型 test-coverage；符号 _make_req）：新增 _make_req helper，确保测试中 Req 的 origin_input_ids 为 array，验证调度策略的正确性。
- test/registered/unit/sampling/test_custom_logit_processor.py（模块 自定义逻辑处理器；类别 test；类型 test-coverage；符号 _make_req）：更新 mock Req 使其 origin_input_ids 和 output_ids 返回 array，覆盖 logit processor 路径。

关键符号：Req.init, DeepseekOCRNoRepeatNGramLogitProcessor.call, _make_req (test)

关键源码片段

test/manual/test_schedule_policy.py

新增 _make_req helper，确保测试中 Req 的 origin_input_ids 为 array，验证调度策略的正确性。

```
import unittest
from array import array

from sglang.srt.managers.schedule_batch import Req, ScheduleBatch
from sglang.srt.managers.schedule_policy import (
    CacheAgnosticPolicy,
    CacheAwarePolicy,
    SchedulePolicy,
)
from sglang.srt.mem_cache.radix_cache import RadixCache
from sglang.srt.sampling.sampling_params import SamplingParams
from sglang.test.test_utils import CustomTestCase


def _make_req(rid, origin_input_text, origin_input_ids, sampling_params=None, **kwargs):
    """Helper to create a Req with array token ids."""
    if sampling_params is None:
        sampling_params = SamplingParams()
    return Req(
```

```
        rid,
        origin_input_text,
        array("q", origin_input_ids), # 转换为 array 避免类型错误
        sampling_params,
        **kwargs,
    )


class TestSchedulePolicy(CustomTestCase):
    # ... 后续测试用例使用 _make_req 而非直接 Req(...)
```

评论区精华

1. all_ids() helper 的效率问题：merrymercy 指出 all_ids() 实现不够高效，mmangkad 随后回退了该方案，改为在调用层直接使用 origin_input_ids + output_ids。
2. 类型归一化策略：Jialin 建议将 PyList 到 PyArray 的转换推到调用者，而不是在 Req.__init__ 中做 isinstance 检查。mmangkad 采纳并移除了 Req 中的转换逻辑。
3. custom_logit_processor 性能优化：Jialin 建议将 current_prefix 从 tuple 改为 array，并避免每轮循环都做 tuple(ngram[:-1])。mmangkad 实现了该优化，并确认对于典型参数（ngram_size=30, window_size=90）array 索引访问比 KMP 更快。
4. 测试覆盖：Jialin 询问 logit processor 的修改是否有测试覆盖，mmangkad 确认已有单元测试。

- all_ids() helper 效率 (performance): 回退 all_ids() helper，改为直接拼接。
- 类型归一化策略 (design): 在调用者层面确保传入 array，Req 内部不再转换。
- n-gram 比较优化 (performance): 使用 array 而非 tuple，消除临时对象。
- 测试覆盖 (testing): 已有测试覆盖。

风险与影响

- 风险：主要风险是部分未改到的调用点可能仍传入 list，导致运行时错误。由于测试覆盖了调度策略、forward split prefill、custom logit processor 等关键路径，且 CI 运行了相关测试，风险较低。另外，custom_logit_processor 中的 array 切片语义与 list 一致，但需注意 array 不支持某些 list 方法（如 .extend 返回 None），但本次变更中未使用。
- 影响：
  - 用户：修复了特定场景下的崩溃，提高稳定性。
  - 系统：统一 token-id 类型为 array，减少 PyList 内存开销，可能轻微提升性能（尤其长序列的 + 操作）。
  - 团队：清理由 PR #25098 引入的技术债务，明确了调用者应传递 array 的约定。
  - 风险标记：核心路径变更，数据结构假设一致

关联脉络

- PR #25098 Migrate scheduler token-id storage to array('q'): 本 PR 直接修复 #25098 引入的类型不兼容问题。