PR #27610 完整报告

sgl-project/sglang

Avoid scattered assignment of extend_input_len and fill_len by merging them into Req.extend_range

合并时间：2026-06-25 08:33

原文链接：http://prhub.com.cn/sgl-project/sglang/pull/27610

执行摘要

- 一句话：合并 extend_input_len/fill_len 为 Req.extend_range，消除分散赋值
- 推荐动作：值得精读。该 PR 展示了如何通过强类型消除隐式语义二义性，是 Req 数据结构走向更不可变、更能防御错误的良好模式。推荐跟进后续 PR #27611（内联 property）和 #27616（先决条件）了解完整演进链。

功能与动机

原实现中 extend 范围被拆分为两个独立赋值的字段，且 start 从未显式存储，通过 len(prefix_indices) + extend_input_len 重建绝对位置，在 Disagg decode + HiCache（committed prefix 不等于 len(prefix_indices)）以及 Mixed-batch running reqs（decode 步骤的 KV 未折叠进 prefix_indices）两种场景下会得出错误值。

实现拆解

1. 定义 Range 类型：在 python/sglang/srt/utils/common.py 新增 Range(NamedTuple)，包含 start、end 字段和只读 length 属性。
2. 改造 Req 核心字段：在 python/sglang/srt/managers/schedule_batch.py 中将 self.fill_len: int 和 self.extend_input_len: int 替换为 self.extend_range: Optional[Range] = None；添加 dllm_initialized: bool = False 用于 DLLM 首次轮标志。将旧字段改为 @property 只读属性，派生自 extend_range；新增 set_extend_range(start, end) 统一设置 range 并自动重算 extend_logprob_start_len。
3. 替换所有写入点：在 schedule_policy.py、scheduler_pp_mixin.py、disaggregation/decode.py、scheduler.py、dllm/mixin/req.py 等 6 个源文件中，将原来分散的 set_extend_input_len + fill_len = 配对改为一条 set_extend_range(start, end)。DLLM 部分的 _init_fill_ids_for_dllm 不再写 fill_len 而是设置 dllm_initialized = True。
4. 同步测试适配：修改了 test_hisparse_unit.py（新增 _FakeReq 用 extend_range 模拟 property）、test_decode_radix_lock_ref.py、test_prefill_adder.py、test_unified_radix_cache_unittest.py、test_scheduler_chunked_req_gate.py 共 5 个测试文件，使其 mock 或直接适配新接口。
5. 修复后续发现的 crash：在 scheduler.py 的 cache_unfinished_req 断言中添加 extend_range is None 守卫，避免 decode 阶段 range 为 None 时触发属性访问异常。

关键文件：

- python/sglang/srt/managers/schedule_batch.py（模块 调度器；类别 source；类型 core-logic；符号 fill_len, extend_input_len, set_extend_range, set_extend_input_len）：核心数据结构 Req 的改动所在，定义新字段、property、set_extend_range 和日志逻辑调整。
- python/sglang/srt/utils/common.py（模块 通用工具；类别 source；类型 core-logic；符号 Range, length）：定义 Range 新类型，供全局使用。
- test/registered/unit/managers/test_hisparse_unit.py（模块 测试；类别 test；类型 test-coverage；符号 _FakeReq, fill_len, extend_input_len）：测试文件中的 mock Req 改为基于 extend_range 的 property，验证新接口在 HiSparse 路径下的正确性。

关键符号：Range.length, Req.fill_len, Req.extend_input_len, Req.set_extend_range, Req._recompute_extend_logprob_start_len

关键源码片段

python/sglang/srt/managers/schedule_batch.py

核心数据结构 Req 的改动所在，定义新字段、property、set_extend_range 和日志逻辑调整。

```
# python/sglang/srt/managers/schedule_batch.py (partial)

# 旧字段被替换为单一 extend_range
self.extend_range: Optional[Range] = None # None 表示 decode/idle/retract 状态
self.dllm_initialized: bool = False # 替代旧的 fill_len is None 作为 DLLM 首次轮标志

# 保留原属性名作为只读 property，确保所有读端无需改动
@property
def fill_len(self) -> int:
    return self.extend_range.end

@property
def extend_input_len(self) -> int:
    return self.extend_range.length

def set_extend_range(self, start: int, end: int) -> None:
    """统一设置 extend 范围，自动更新 logprob 偏移。"""
    self.extend_range = Range(start, end)
    self._recompute_extend_logprob_start_len()

def _recompute_extend_logprob_start_len(self):
    """根据 extend_range 和 logprob_start_len 计算相对偏移。"""
    if self.logprob_start_len == -1:
        logprob_start_len = len(self.full_untruncated_fill_ids)
    else:
        logprob_start_len = self.logprob_start_len
    self.extend_logprob_start_len = logprob_start_len - self.extend_range.start
```

test/registered/unit/managers/test_hisparse_unit.py

测试文件中的 mock Req 改为基于 extend_range 的 property，验证新接口在 HiSparse 路径下的正确性。

```
# test/registered/unit/managers/test_hisparse_unit.py (partial)

from sglang.srt.utils.common import Range

class _FakeReq(SimpleNamespace):
    """模拟 Req 的 extend_range 属性，用于单元测试。"""
    @property
    def fill_len(self) -> int:
        return self.extend_range.end

    @property
    def extend_input_len(self) -> int:
        return self.extend_range.length

def _make_req(rid="test-req-0", origin_input_ids=None, output_ids=None):
    # ... 原本使用 SimpleNamespace + set_extend_input_len lambda
    req = _FakeReq(
        rid=rid,
        # ...
    )
    req.set_extend_range = lambda start, end: setattr(req, "extend_range", Range(start, end))
    return req
```

评论区精华

唯一有人参与的讨论是作者自己在 CI 下游 PR #27611 中发现 cache_unfinished_req 的断言在 extend_range is None 时崩溃，随后在该 PR 中修复并提及。未出现设计层面的争论。

- cache_unfinished_req 断言在 extend_range is None 时 crash (correctness): 已修复：decode 等非 extend 状态下不再因属性访问而异常。

风险与影响

- 风险：
  1. 行为保持假设风险：所有写入点都按 start = end - old_length 计算，但若存在未覆盖的写入路径（如分叉分支或后续新增代码），可能导致 range 不一致。
  2. DLLM 标志替换风险：dllm_initialized 完全替代了旧的 fill_len is None 检查，如果任一生命周期路径未设置该标志（如极端 retract 场景），可能导致阶段判断错误。
  3. 测试覆盖风险：作者注明未手动运行测试，仅依赖 CI；测试覆盖可能不充分（尤其 HiCache 和 mixed-batch 组合场景）。 - 影响：内部接口重构：影响所有涉及 extend 范围赋值的调度路径，但 property 保持读接口兼容，外部模块无感知。DLLM 模块：首次轮标志改变，可能与其他依赖 fill_len 的遗存代码冲突。测试：五处测试适配，需确保 mock 行为与真实 Req 一致。影响程度中等，因为改动范围集中但全局调度依赖该数据结

构。 - 风险标记：核心路径变更，DLLM 初始化标志替换，行为保持假设，测试覆盖可能不足

关联脉络

- PR #27611 Inline *extend_range* accessors and remove the *extend_input_len*/fill_len properties: 本 PR 的后继：将 *extend_range* property 内联为直接字段访问，完成字段合并的最终清理。
- PR #27616 Avoid dual semantics of *extend_input_len* by computing the candidate on the fly: 本 PR 的先决条件：将 init_next_round_input 中的候选长度改为动态计算，消除了 *extend_input_len* 的双重语义，使本 PR 中的字段合并得以进行。
- PR #27625 Remove Req.*extend_logprob_start_len* field and make it pure: 进一步重构：移除 *extend_logprob_start_len* 字段，改为纯自由函数，本 PR 中已开始依赖。