PR #48630 完整报告

vllm-project/vllm

[MRV2][Spec Decode] Avoid rejection sampler OOM by chunking

合并时间：2026-07-23 18:13

原文链接：http://prhub.com.cn/vllm-project/vllm/pull/48630

执行摘要

- 一句话：分块拒绝采样避免 OOM
- 推荐动作：建议精读 rejection_sampler.py 中的分块实现，特别是 _iter_request_chunks 和 _verify_in_chunks 的设计，该方法通过固定缓冲区避免 OOM 且保持无同步循环，对未来类似问题有借鉴意义。

功能与动机

在推测解码中，大 batch 拒绝采样需要为所有候选 token 物化 FP32 logits 缓冲区，可能高达数 GB 导致 OOM。本 PR 通过分块复用 1GB 固定缓冲区解决该问题，同时保持数值行为一致。

实现拆解

1. 在 rejection_sampler.py 中定义 MAX_CHUNK_BYTES = 1GB 上限，基于 FP32 字节数计算每块最大 logits 数 max_chunk_logits，并新增 _iter_request_chunks 工具函数用于生成不分割请求的 chunk 区间。
2. 将原 __call__ 拆分为 _verify（处理单块采样逻辑）和 _verify_in_chunks（遍历 chunk 区间、循环调用 _verify 并拼接 sampled、num_sampled 和 logprobs 结果）。_verify 从 input_batch 参数改为直接接收所需张量，使其可独立于 InputBatch 对象调用。
3. 在 outputs.py 的 LogprobsTensors 类中新增 cat 静态方法，支持拼接分块产生的 logprob 张量（logprob_token_ids、logprobs、selected_token_ranks），并在 prompt_logprob.py 中复用此方法。
4. 在 vllm/config/model.py 中添加 PROCESSED_LOGPROBS_MODES 常量，统一多处对 processed_logits / processed_logprobs 字面量的判断，同时修改 sampler.py、prompt_logprob.py 等文件以使用该常量。
5. 新增单元测试文件 test_gpu_rejection_sampler_chunking.py，覆盖 chunk 边界不分割请求、分块分数与全 batch 一致性；在 test_rejection_sampler_utils.py 中增加 test_chunked_requests_match_full_batch 验证底层 rejection_sample 的分块等效性。

关键文件：

- vllm/v1/worker/gpu/spec_decode/rejection_sampler.py（模块 拒绝采样；类别 source；类型 core-logic；符号 _iter_request_chunks, call, _verify, _verify_in_chunks）：核心实现文件，重构了拒绝采样器，新增 chunk 迭代和分块调度逻辑。

- vllm/v1/outputs.py（模块 输出层；类别 source；类型 core-logic；符号 cat）：新增 LogprobsTensors.cat 静态方法，支持分块 logprob 拼接。
- tests/v1/worker/test_gpu_rejection_sampler_chunking.py（模块 拒绝采样测试；类别 test；类型 test-coverage；符号 test_iter_request_chunks_preserves_request_boundaries, test_chunked_scores_match_full_batch, fake_verify）：新增测试文件，验证 chunk 边界和分数一致性。
- tests/v1/spec_decode/test_rejection_sampler_utils.py（模块 采样工具测试；类别 test；类型 test-coverage；符号 test_chunked_requests_match_full_batch）：扩展测试，验证分块 rejection_sample 等效性。

关键符号：_iter_request_chunks, _verify, _verify_in_chunks, RejectionSampler.call, LogprobsTensors.cat

关键源码片段

vllm/v1/worker/gpu/spec_decode/rejection_sampler.py

核心实现文件，重构了拒绝采样器，新增 chunk 迭代和分块调度逻辑。

```
# 分块拒绝采样的核心迭代器函数
def _iter_request_chunks(cu_num_logits: np.ndarray, max_chunk_logits: int) -> Iterator[tuple[int, int]]:
    """根据 cu_num_logits 数组生成不分割请求的 chunk 区间 (start, end)"""
    assert max_chunk_logits > 0
    num_reqs = cu_num_logits.size - 1
    start = 0
    while start < num_reqs:
        max_logit = int(cu_num_logits[start]) + max_chunk_logits
        # 找到第一个大于 max_logit 的索引，减 1 得到最后请求的 end
        end = int(np.searchsorted(cu_num_logits, max_logit, side='right') - 1)
        # 至少包含一个请求
        end = min(num_reqs, max(start + 1, end))
        yield start, end
        start = end
```

vllm/v1/outputs.py

新增 LogprobsTensors.cat 静态方法，支持分块 logprob 拼接。

```
@staticmethod
def cat(
    tensors: Sequence['LogprobsTensors'],
    cu_num_generated_tokens: list[int] | None = None,
) -> 'LogprobsTensors':
    """拼接扁平化的 logprob 张量"""
    assert tensors
    assert cu_num_generated_tokens is not None or all(
        tensor.cu_num_generated_tokens is None for tensor in tensors
    )
    # 单元素时直接返回（若需更新 cu_num_generated_tokens 则替换）
```

```
    if len(tensors) == 1:
        tensor = tensors[0]
        if cu_num_generated_tokens is None:
            return tensor
        return tensor._replace(cu_num_generated_tokens=cu_num_generated_tokens)
    # 多元素时逐字段 torch.cat
    return LogprobsTensors(
        logprob_token_ids=torch.cat([t.logprob_token_ids for t in tensors]),
        logprobs=torch.cat([t.logprobs for t in tensors]),
        selected_token_ranks=torch.cat([t.selected_token_ranks for t in tensors]),
        cu_num_generated_tokens=cu_num_generated_tokens,
    )
```

评论区精华

- njhill建议将 logprob 拼接逻辑抽象为 LogprobsTensors.cat 静态方法，并复用至 prompt_logprob；mgoin采纳并实现。
- TheEpicDolphin询问单 chunk 时能否避免 torch.cat；mgoin回应现有单 chunk 返回已跳过 cat，无需额外优化。
- njhill提出若干代码风格微调（冗余括号、返回语句简化），mgoin全部吸纳。
- 将 logprob 拼接逻辑重构到 LogprobsTensors.cat (design): 已采纳，移入 LogprobsTensors.cat。
- 单 chunk 时避免不必要 cat (performance): 已有逻辑处理，无需额外修改。
- 代码风格微调 (style): 采纳建议并修改。

风险与影响

- 风险：核心路径变更（拒绝采样是推测解码关键环节），分块引入额外 cat 操作可能带来微小性能开销，但测试显示平均接受长度未下降甚至略有提升，且单 chunk 时跳过 cat。固定缓冲区大小（1GB）为硬编码值，若极端场景需更多内存可能仍不足，但已比之前无限制安全。无同步循环依赖 np.searchsorted 和 cu_num_logits 的正确性。
- 影响：影响所有使用推测解码（spec_decode）的用户，尤其是大 batch 场景。不再因 rejection sampler 临时缓冲区过大而 OOM，KV cache 可用量提升（测试中从 47.94 GiB 恢复至 52.02 GiB）。数值行为一致，gsm8k 准确率不变，平均接受长度从 3.06 提升至 3.18。开发者可从分块设计模式受益。
- 风险标记：核心路径变更，固定缓冲区大小，无同步循环

关联脉络

- 暂无明显关联 PR