PR #31620 完整报告

sgl-project/sglang

[spec decoding] replace torch.multinomial with several native torch op in rejection sampling

合并时间：2026-07-18 07:58

原文链接：http://prhub.com.cn/sgl-project/sglang/pull/31620

执行摘要

- 一句话：用原生算子替换 torch.multinomial
- 推荐动作：值得精读，尤其是对 CUDA graph 兼容性和 Gumbel-max 采样技巧感兴趣的工程师。此 PR 展示了一个简单而有效的优化模式：用原生算子组合替代黑盒算子，以支持 graph 捕获。

功能与动机

torch.multinomial 内部包含设备端合法性断言（validity assert），该断言在 CUDA graph 捕获后每次重放仍会执行，阻碍 graph 图回放。PR body 中的速度测试也表明这一替换显著降低了延迟。

实现拆解

1. 修改 fast_sample 函数（python/sglang/srt/speculative/spec_utils.py）：
   - 将 torch.multinomial(probs, num_samples=num_samples) 替换为 Gumbel-max trick：首先生成指数分布随机数 q，然后计算 scores = probs.float() / q，最后根据 num_samples 使用 argmax（单样本）或 topk（多样本）选取索引。
2. 保留相同的函数签名和返回值：仍然返回 (sample_p, sample_index)，调用方无需修改。
3. 仅修改一个文件，改动量小（10 行新增、1 行删除），且无新增依赖。

关键文件：

- python/sglang/srt/speculative/spec_utils.py（模块 推测解码；类别 source；类型 core-logic；符号 fast_sample）：核心变更文件，修改 fast_sample 函数，用 Gumbel-max 实现替代 torch.multinomial，是性能提升的直接来源。

关键符号：fast_sample

关键源码片段

python/sglang/srt/speculative/spec_utils.py

核心变更文件，修改 fast_sample 函数，用 Gumbel-max 实现替代 torch.multinomial，是性能提升的直接来源。

```
def fast_sample(probs: torch.Tensor, num_samples: int = 1):
    """Gumbel-max draw: argmax(probs / Exp(1)). Distributionally equivalent to
```

```
    torch.multinomial minus its device-side validity assert, which a capturing
    CUDA graph would replay every step."""
    # 生成指数分布随机数，用于 Gumbel-max trick
    q = torch.empty_like(probs, dtype=torch.float32).exponential_(1.0)
    # 防止除零，clamp 到 float32 最小正数
    q.clamp_min_(torch.finfo(torch.float32).tiny)
    # 计算 scores，argmax 等价于按概率采样
    scores = probs.float() / q
    if num_samples == 1:
        sample_index = scores.argmax(dim=-1, keepdim=True)
    else:
        sample_index = scores.topk(num_samples, dim=-1).indices
    sample_p = probs.gather(1, sample_index)
    return sample_p, sample_index
```

评论区精华

无显著的 review 讨论，唯一的审核来自 kpham-sgl 并直接批准。

- 暂无高价值评论线程

风险与影响

- 风险：风险较低：Gumbel-max trick 在数学上等价于 multinomial，且数值稳定性通过 clamp_min_(tiny) 保证。但需注意 exponential_ 是原地操作，若 probs 被后续复用可能产生副作用；当前实现已使用 torch.empty_like 分配新张量，无此风险。另外，probs.float() 会创建浮点副本，增加少量显存开销，但在 rejection sampling 路径中通常可接受。
- 影响：直接影响 speculation decoding 中的 rejection sampling 路径，特别是使用 EAGLE 系列算法的场景。性能提升显著（实测 ~7x），并且为后续 CUDA graph 融合扫清了障碍。不影响非 rejection sampling 路径。
- 风险标记：无测试配套变更，小幅数值精度风险

关联脉络

- PR #31614 [spec decoding] fix multi_layer_eagle rotate_input_ids kernel registration: 同为 speculative decoding 模块的 bugfix，且本 PR 将与此后 kernel 融合，属于同一功能线。