

PR #37134 完整报告

sgl-project/sglang

[ROCm] Fix EAGLE spec-decode verify silently sampling greedy on HIP

合并时间: 2026-09-16 09:49

原文链接: <http://prhub.com.cn/sgl-project/sglang/pull/37134>

执行摘要

- 一句话: 修复 ROCm 上 EAGLE verify 静默贪心采样, HIP 接入 Triton 采样链
- 推荐动作: 值得精读。核心价值不只是那个 `_verify_uses_greedy` 门控, 而是整套“默认行为变更 + 数据链打通 + 底层 kernel 守卫”的联动修复。建议重点关注: `_verify_uses_greedy` 如何做到非 HIP 平台字节等价、`speculative_hook.py` 中 `cfg` 与 `resolved_view` 的读写契约差异、以及 `reject_sampling.py` 中 `q` 守卫的两次迭代 (`residual` 与 `accept test` 分别加固)——后三者都是一般 review 容易漏掉的隐蔽点。

功能与动机

PR body 明确指出: `eagle_sample` 中 `_is_hip` 与 `is_all_greedy` 并列, 导致 HIP 无条件走 `torch.argmax`, `temperature` 和 `top_p` 被完全忽略, 日志中也无任何提示, `temp > 0` 时退化为重复循环。根因是采样分支依赖 `tree_speculative_sampling_target_only`、`top_k_renorm_prob`、`top_p_renorm_prob`, 这些在 `sgl_kernel` 中仅支持 CUDA/MUSA。评论中 xiaobochoen-amd 还补充: 该路径此前在 HIP 上从未被进入, 所以相关代码是死代码, 隐藏 bug 只有在本次激活采样后才会触发。

实现拆解

实现分为四步:

1. 改造 verify 采样门控 (`python/sglang/srt/speculative/eagle_utils.py`): 把内联的 `if sampling_info.is_all_greedy or _is_cpu or _is_hip or _is_xpu` 抽成纯函数 `_verify_uses_greedy`, 其语义为 `is_all_greedy or is_cpu or is_xpu or (is_hip and not use_rejection_sampling)`。非 HIP 平台完全等价于原谓词; HIP 仅当 rejection sampling 开启且 batch 非全贪心时进入采样分支。采样分支内部把 `sampling_fn` 的“三元表达式 + eager import”改为 `if/else` 分支内 import, 避免 HIP 解析 CUDA-only 的 `tree_speculative_sampling_target_only`; `top_k/top_p_renorm_prob` 在 HIP 上通过 `dflash_utils.py` 同款的别名方式复用 `renorm_triton.py` 的 Triton 实现。
2. HIP 自动开启 rejection sampling (`python/sglang/srt/arg_groups/speculative_hook.py`): 新增 `_should_auto_enable_hip_rejection_sampling` 纯函数, 在 `_handle_eagle_family` 中调用; 条件为 HIP、未显式开启、算法为 EAGLE、无 token map、`eagle_topk == 1`、`accept` 阈值均为 1.0、非确定性推理。满足时通过 `declare_resolution` 写入 `speculative_use_rejection_sampling=True` 并打 `logger.info`。随后校验块从 `cfg` 读取改为 `resolved_view(server_args)`, 因为 `declare_resolution` 写入的是 stash 而非 dataclass,

旧读法会跳过所有自动开启的配置。EAGLE3 因 hot-token 小词表 (32000 vs 128256) 尚无法喂给 rejection kernel, 被显式排除。

3. 修通 draft 侧 proposal 数据链 (python/sglang/srt/speculative/spec_utils.py、eagle_draft_cuda_graph_runner.py、eagle_worker_v2.py) : sample_draft_proposal 新增 top_ks 参数, 对 top_k <= 1 的贪心行强制 argmax 作为 proposal, 避免把 temperature=0 (被重写为 T=1, top_k=1) 的请求误当作 T=1 采样; CUDA graph runner 新增设备端 top_ks 缓冲并在 replay 前拷贝; eagle_worker_v2.py 在 graph 执行后对 draft_probs 做 clone(), 防止其指向的 graph 私有内存池在 eagle_sample 读取前被回收。
4. 加固 Triton 采样器的概率守卫 (python/sglang/kernels/ops/speculative/reject_sampling.py) : 把 accept 测试从 coin * q < p 改为 q_is_prob & (coin * q < p), 其中 q_is_prob = (q > 0.0) & (q <= 1.0), 确保非法 q (NaN、±inf、零、负值) 不会无条件接受; 两处 residual 计算的 NaN 守卫也改为对概率区间 [0, 1] 的判定, 因为 q_val == q_val 无法排除 inf。

测试配套: 新增 test/registered/unit/spec/test_eagle_gate_routing.py (CPU 真值表, 覆盖 gate 收敛性与 HIP 自动开启条件, 4 个测试类)、test/registered/spec/eagle/test_rocm_eagle_renorm.py (renorm 与独立 torch 参考的数值等价及 top_p == 1.0 边界, 注册到 CUDA/AMD 双端)。测试文件在后续 commit 中按 registered-test 分类规范移到 test/registered/kernel/...

关键文件:

- test/registered/unit/spec/test_eagle_gate_routing.py (模块 门控测试; 类别 test; 类型 test-coverage; 符号 _pre_patch_gate, TestEagleGateRouting, _gate, test_non_hip_is_byte_identical_to_pre_patch) : Gate 真值表与 HIP 自动开启边界测试, 纯布尔逻辑可在 CPU CI 运行, 守住非 HIP 平台字节等价契约。
- python/sglang/srt/arg_groups/speculative_hook.py (模块 参数解析; 类别 source; 类型 core-logic; 符号 _should_auto_enable_hip_rejection_sampling, _handle_eagle_family) : 新增 HIP 自动开启 rejection sampling 的判定与写入逻辑, 并修正 cfg/resolved_view 读写契约, 直接影响所有 ROCm EAGLE 启动路径。
- python/sglang/srt/speculative/eagle_utils.py (模块 投机采样; 类别 source; 类型 core-logic; 符号 _verify_uses_greedy, eagle_sample) : 核心门控 _verify_uses_greedy 与平台相关采样 dispatch 所在, 决定 HIP 何时从 argmax 切换到 Triton 链式采样。
- python/sglang/srt/speculative/spec_utils.py (模块 草稿采样; 类别 source; 类型 core-logic; 符号 sample_draft_proposal) : sample_draft_proposal 携带真实 per-request top_k, 修掉贪心请求被当作 T=1 采样的隐蔽问题, 是数据链修复的关键一环。
- python/sglang/srt/speculative/eagle_draft_cuda_graph_runner.py (模块 图执行器; 类别 source; 类型 dependency-wiring; 符号 EagleDraftInputBuffers, capture_one_shape, execute) : CUDA graph runner 需要与 temperatures 一样携带设备端 top_ks 缓冲, 并在 replay 前拷贝, 否则贪心行修正到不了 draft proposal。
- python/sglang/srt/speculative/eagle_worker_v2.py (模块 草稿执行; 类别 source; 类型 core-logic; 符号 draft, draft_forward, _draft_extend_for_decode) : graph 模式下把 draft_probs 从 graph 私有内存池 clone 出来, 并给两个 sample_draft_proposal 调用点

传入真实 top_ks。

- python/sglang/kernels/ops/speculative/reject_sampling.py (模块 采样内核; 类别 infra; 类型 infrastructure; 符号 speculative_sampling_classic_kernel) : Triton 采样器收紧 q 有效性判断: accept 测试拒绝非概率 q, residual 守卫生效覆盖 inf/NaN/ 负数。

关键符号: _verify_uses_greedy, eagle_sample, _should_auto_enable_hip_rejection_sampling, _handle_eagle_family, sample_draft_proposal, draft, draft_forward, _draft_extend_for_decode, speculative_sampling_classic_kernel

关键源码片段

python/sglang/srt/arg_groups/speculative_hook.py

新增 HIP 自动开启 rejection sampling 的判定与写入逻辑, 并修正 cfg/resolved_view 读写契约, 直接影响所有 ROCm EAGLE 启动路径。

```
def _should_auto_enable_hip_rejection_sampling(
    *,
    is_hip: bool,
    use_rejection_sampling: bool,
    algorithm: Optional[str],
    token_map: Optional[str],
    eagle_topk: int,
    accept_threshold_single: float,
    accept_threshold_acc: float,
    enable_deterministic_inference: bool,
) -> bool:
    """判断 HIP 上是否可以把 rejection sampling 默认打开。
```

rejection sampling 目前还不能消费 hot-token 小词表 (eagle_worker_v2 的 FIXME) , 自动开启会让之前能跑的贪心配置崩溃, 包括 EAGLE3 stage-a 测试 (draft 32000 vs target 128256) 。因此 EAGLE3 和已有 token map 的 EAGLE 都不开。

```
"""
return (
    is_hip
    and not use_rejection_sampling
    and algorithm == "EAGLE"
    and token_map is None
    and eagle_topk == 1
    and accept_threshold_single == 1.0
    and accept_threshold_acc == 1.0
    and not enable_deterministic_inference
)
```

```
def _handle_eagle_family(server_args: ServerArgs) -> None:
    cfg = resolving_view(server_args)
    # ... 中间逻辑省略 ...
```

ROCm 没有 CUDA/MUSA 采样 verify 内核, 不开 rejection sampling 的话

```

# EAGLE verify 会静默回退到 argmax, 忽略 temperature/top_p。
if _should_auto_enable_hip_rejection_sampling(
    is_hip=get_platform().is_hip,
    use_rejection_sampling=cfg.speculative_use_rejection_sampling,
    algorithm=cfg.speculative_algorithm,
    token_map=cfg.speculative_token_map,
    eagle_topk=cfg.speculative_eagle_topk,
    accept_threshold_single=cfg.speculative_accept_threshold_single,
    accept_threshold_acc=cfg.speculative_accept_threshold_acc,
    enable_deterministic_inference=cfg.enable_deterministic_inference,
):
    declare_resolution(
        server_args,
        "_handle_eagle_family",
        speculative_use_rejection_sampling=True,
    )
    logger.info(
        "ROCm needs rejection sampling for EAGLE spec-decode to sample at all; "
        "enabling speculative_use_rejection_sampling by default."
    )

# 用 resolved_view 而不是 cfg: 上面的 declare_resolution 把值写进 stash,
# 后续的 cfg 读取看不到, 否则自动开启的配置连校验都会跳过。
if resolved_view(server_args).speculative_use_rejection_sampling:
    if cfg.speculative_algorithm not in ("EAGLE", "EAGLE3"):
        raise NotImplementedError(...)

```

python/slang/srt/speculative/eagle_utils.py

核心门控 `_verify_uses_greedy` 与平台相关采样 dispatch 所在, 决定 HIP 何时从 argmax 切换到 Triton 链式采样。

```

def _verify_uses_greedy(
    *,
    is_all_greedy: bool,
    is_cpu: bool,
    is_hip: bool,
    is_xpu: bool,
    use_rejection_sampling: bool,
) -> bool:
    """判断 EAGLE verify 是否必须提交 argmax 而不是走采样路径。

    HIP 没有 CUDA/MUSA 的采样 verify 内核, 所以原来被无条件列在门控里。
    rejection sampling 会把 verify 路由到纯 Triton 链式采样器,
    因此只有“HIP 且未开 rejection sampling”时才需要强制走 greedy。
    其余平台该表达式与原谓词完全一致。
    """

    return is_all_greedy or is_cpu or is_xpu or (is_hip and not use_rejection_sampling)

def eagle_sample(

```

```

verify_input: EagleVerifyInput,
batch: ScheduleBatch,
logits_output: LogitsProcessorOutput,
grammar_mask: Optional[GrammarMask] = None,
uno_target_max_top_k: Optional[int] = None,
):
    # ... 前置 logits 加工省略 ...
    use_rejection_sampling = get_spec().speculative_use_rejection_sampling
    if _verify_uses_greedy(
        is_all_greedy=sampling_info.is_all_greedy,
        is_cpu=_is_cpu,
        is_hip=_is_hip,
        is_xpu=_is_xpu,
        use_rejection_sampling=use_rejection_sampling,
    ):
        # greedy 分支: argmax + 树形 verify, HIP 下还需 broadcast 保持 TP 一致
        target_predict = torch.argmax(next_token_logits, dim=-1)
        target_predict = target_predict.reshape(bs, verify_input.draft_token_num)
        predict, accept_index, num_correct_drafts = verify_tree_greedy_func(...)
    else:
        # 采样分支: 非 NPU 平台统一走 Triton 链式 sampler
        from sglang.kernels.ops.speculative.reject_sampling import (
            chain_speculative_sampling_triton,
        )

        # if/else 而不是三元: CUDA-only 的名字在未选中的分支里也必须能解析,
        # 而 HIP 只有 rejection sampling 开启时才会到达这里。
        if use_rejection_sampling:
            sampling_fn = chain_speculative_sampling_triton
        else:
            if not _is_npu:
                from sgl_kernel import tree_speculative_sampling_target_only
                sampling_fn = tree_speculative_sampling_target_only

            if _is_hip:
                # 与 dflash_utils.py 相同的别名方式: HIP 上复用 Triton renorm。
                from sglang.kernels.ops.sampling.renorm_triton import (
                    top_k_renorm_probs_triton as top_k_renorm_prob,
                )
                from sglang.kernels.ops.sampling.renorm_triton import (
                    top_p_renorm_probs_triton as top_p_renorm_prob,
                )
            elif not _is_npu:
                from sgl_kernel import top_k_renorm_prob, top_p_renorm_prob

        target_probs = F.softmax(next_token_logits / expanded_temperature, dim=-1)
        if sampling_info.need_top_k_sampling:
            target_probs = top_k_renorm_prob(target_probs, ...)
        if sampling_info.need_top_p_sampling:

```

```
target_probs = top_p_renorm_prob(target_probs, ...)
```

python/sglang/srt/speculative/spec_utils.py

sample_draft_proposal 携带真实 per-request top_k, 修掉贪心请求被当作 T=1 采样的隐蔽问题, 是数据链修复的关键一环。

```
def sample_draft_proposal(
    next_token_logits: torch.Tensor,
    temperatures: torch.Tensor,
    top_ks: Optional[torch.Tensor] = None,
):
    """Leviathan draft proposal:  $q = \text{softmax}(\text{logits} / T)$ ,  $X \sim q$ .
```

返回 (q, q(X), X)。verify 的 accept 测试 $\text{coin} * q(X) < p(X)$ 只有在 q 恰好是 X 的采样分布时才无偏, 所以调用方必须把这里返回的 q 传给 verify, 不能自己重算。

贪心行 (top_k == 1) 直接提议自己的 argmax。SamplingParams 会把 temperature 0 重写成 temperature=1.0 + top_k=1, 所以单看 T 无法区分贪心请求和普通 T=1 请求; 对尖锐但非退化的分布采样会频繁提出非 argmax token, 显著损失 accept 长度。

```
"""
probs = torch.softmax(next_token_logits / temperatures, dim=-1)
topk_p, topk_index = fast_sample(probs, num_samples=1)
if top_ks is not None:
    # 设备不匹配必须断言而不是静默跳过: host 端 top_ks 会让修正失效,
    # 且症状是 draft accept 长度悄悄降约 20%, 容易被误判为模型问题。
    assert top_ks.device == probs.device, (
        f"top_ks must be on {probs.device} to reach the draft proposal, "
        f"got {top_ks.device}; the caller has to carry the real per-request "
        "top_k, not a host placeholder"
    )
    greedy = (top_ks <= 1).view(-1, 1)
    topk_index = torch.where(greedy, probs.argmax(dim=-1, keepdim=True), topk_index)
    topk_p = probs.gather(1, topk_index)
return probs, topk_p, topk_index
```

评论区精华

1. 应复用现有 Triton renorm 而非新写内核: 1am9trash 在 CHANGES_REQUESTED 中指出 renorm_triton.py 已提供 HIP 版 top_k_renorm_prob / top_p_renorm_prob, 且 dflash_utils.py 已用平台条件 import 别名消费它们, 建议复用。xiaobo chen-amd 采纳后把私有 kernel 和对应测试删除, renorm 回到单一代码路径。
2. EAGLE3 自动开启会导致崩溃: jiejingzhangamd 跟进发现 HIP 上为 EAGLE3 自动开启 rejection sampling 会在 stage-a test_basic_sanity_eagle3 崩溃 (draft vocab 32000 vs target 128256), 随后提交 commit 将 EAGLE3 排除在 auto-enable 之外。

3. 该路径此前是死代码，隐藏了四个数据正确性 bug: JohnQinAMD 提交四个 commit 并评论说明——不修复则“ships the corruption in row 2”: draft_probs 指向 CUDA graph 私有内存池，被回收后 2-4% token 变成未训练 padding 行 (GPQA 掉到 0.4646) ; top_k 未传递导致贪心请求被当 T=1 采样; residual 守卫 $q == q$ 放行 inf; accept 测试 $\text{coin} * q < p$ 对非法 q 无条件接受。
4. 基准验证: xiaobochoen-amd 回应 At1a8 的 benchmark 请求时给出 GLM-5.2-FP8 数据 (GPQA 0.7071→0.8889、请求完成率 62.1%→94.4%、wall clock 60→24min) ; At1a8 随后确认 DeepSeek-V4 MTP 的 GSM8K/AA-LCR/GPQA 等也全部修复或持平。
- 是否复用现有 renorm_triton 而非新增私有 renorm kernel (design): xiaobochoen-amd 采纳: 删除私有 kernel、_is_hip 分支与其测试, 改为与 dflash_utils.py 相同的别名 import , 实现回到单一代码路径。
- HIP 上自动开启 EAGLE3 rejection sampling 会崩溃 (correctness): 新增 algorithm == "EAGLE" 限制, EAGLE3 与带 token map 的 EAGLE 不自动开启, 保持贪心路径; 后续再用 d2t scatter 支持。
- 路径激活后暴露的四个数据正确性 bug (correctness): 四个 commit 全部合入: draft_probs.clone()、top_ks 设备缓冲链路、q 范围守卫 [0,1]、accept 测试 q_is_prob 前置。
- 基准验证与跨模型收益确认 (question): 多个模型验证收益: GPQA 0.7071→0.8889, 请求完成率 62.1%→94.4%, GSM8K 92.6→96.44 (DSV4 MTP) 。

风险与影响

- 风险:
 1. 默认行为变更: HIP 上符合条件的 EAGLE 配置会自动开启 rejection sampling, 用户若有自定义采样参数 (如非 1.0 的 accept threshold) 不会被自动开启, 但显式开启仍受限于校验逻辑。EAGLE3 与带 token map 的 EAGLE 保持贪心, 不会因本 PR 改变行为。
 2. CUDA graph 内存池风险: draft_probs.clone() 修复了内存回收问题, 但增加了每次 graph 执行后的显存拷贝开销, 可能轻微影响吞吐; 该拷贝只在 rejection sampling 路径存在。
 3. 数值守卫收紧: reject_sampling.py 把 q 判定改为概率区间 [0,1], 理论上合法但极端的 q (如恰好为 0 的概率) 会从“接受”变为“拒绝”, 不过这不是合法概率分布, 方向是安全 (拒绝后 residual 走目标分布) 。
 4. 非 HIP 平台回归风险低: 门控抽取为纯函数后由真值表测试钉死与原谓词字节等价, 测试覆盖 CPU/CUDA/NPU/XPU 全组合。
 5. 测试缺口: EAGLE3 的 sampling 路径仍未被 HIP 覆盖 (被显式排除), 未来若支持 hot-token scattered proposal 需补测试。- 影响: 用户侧: AMD ROCm 上所有 EAGLE / DeepSeek-V4 MTP 等投机解码用户的输出质量和任务完成率大幅提升, temperature/top_p 从“被静默忽略”变为真正生效; 重复循环问题消失。系统侧: 改动横跨采样 kernel、spec 调度、CUDA graph runner、参数解析四个层次, 约 300 行代码 + 测试, 集中影响 speculation 子系统; resolved_view 的修正也修复了一个参数校验遗漏。团队侧: 该 PR 展示了“修复一个 dead-code gate 后必须连带激活同一数据链路上

所有潜在 bug”的排查模式，对后续 AMD spec-decode 维护有指导意义。 - 风险标记：核心路径变更，默认行为变更，CUDA graph 内存池风险，EAGLE3 未覆盖，新增数值守卫，性能影响 clone 拷贝

关联脉络

- PR #39253 DeepSeek-V4 MTP accuracy issue benchmark（评论中提及）：At1a8 指出本 PR 同时修复了 #39253 描述的 DeepSeek-V4 MTP 精度问题，实测 GSM8K 等指标从低于官方到超越 / 持平官方。
- PR #26358 ROCm argmax tie-break corrupts MTP draft selection on FP8 logits: eagle_worker_v2.py 中 self.topk == 1 and not _is_hip 的注释引用 #26358，与本 PR 的 HIP 贪心门控直接相关。