PR #33298 完整报告

sgl-project/sglang

[Spec] Support sampling in the DSPARK graph-folded draft proposal

合并时间：2026-08-03 09:46

原文链接：http://prhub.com.cn/sgl-project/sglang/pull/33298

执行摘要

- 一句话：DSPARK 折叠草案支持采样，AUTO 内存门控回退 greedy
- 推荐动作：值得精读，尤其是 DsparkDraftSampler 的图内采样实现和 _resolve_folded_sampling 的 AUTO 门控逻辑。这类 " 图折叠 + 采样 " 的设计在推测解码场景中具有代表性，greedy 与采样行的混合处理、静态缓冲区与 CUDA graph 捕获的配合方式、以及 eager accept 的降级策略都有借鉴价值。建议关注后续是否会补上针对采样路径的专项测试。

功能与动机

此前 DSPARK 的图折叠草案只支持 greedy 路径，任何包含采样的批次都会被迫走 eager proposal 路径，无法利用 CUDA graph 的捕获加速。PR body 明确指出目标是将折叠提案从 greedy-only 扩展到混合 greedy + 采样批次，从而在保证采样正确性的同时保持图折叠的性能收益。此外，代码组织上希望把折叠采样头独立成模块，减小 dspark_draft.py 的 diff 面。

实现拆解

实现按以下步骤拆解：

1. 新增独立采样器模块python/sglang/srt/speculative/dspark_components/dspark_draft_sampler.py：将原 dspark_draft.py 中的 greedy_step_sampler、DsparkDraftSampler、maybe_build_draft_sampler 整体搬移，并在 DsparkDraftSampler 中新增 folded_sampling 模式。该模式在初始化时按 max_bs * vocab 预分配 temperatures、greedy_mask、exp_noise 和 corrected_out 静态缓冲区，这些缓冲区被 CUDA graph 捕获引用。
2. Host 侧采样参数刷新：新增 stage_sampling_params(bs, sampling_info) 方法，在每次图回放前将 temperature（clamp 到 1e-5 以上）和 greedy 掩码（top_ks <= 1）写入静态缓冲区；dspark_draft.py 的 _run_forward 中在确定能走图时调用它，保证图内读取到的是当前批次参数。
3. 图内混合采样：DsparkDraftSampler.__call__ 在 folded_sampling=True 时定义闭包 sampler，通过 self.exp_noise[:bs].exponential_() 生成图内 philox 噪声，再调用 SampleStepTokens.execute 对非 greedy 行做 Gumbel 采样；同时将 markov_head.sample_block 返回的 corrected_logits 写入 corrected_out，供 eager accept 路径使用。

4. 内存门控决策：新增 _resolve_folded_sampling，依据 SGLANG_DSPARK_FOLDED_SAMPLING 环境变量决定：OFF 直接关闭；FORCE 强制开启；AUTO 则估算所需噪声与 logits 缓冲区大小（max_bs * vocab 的 float32 噪声 + max_bs * gamma * vocab 的权重 dtype logits），与 get_available_gpu_memory 对比，剩余内存不足 1.0 GB 时回退到 greedy-only 并给出 warning。
5. 编排层接入：dspark_draft.py 的 DraftBlockProposer.propose 中，折叠条件从 all_greedy 放宽到 all_greedy or draft_sampler.folded_sampling；若折叠且开启了采样，直接从 sampler 取 greedy_mask、temperatures 和 corrected_logits 构造 DraftBlockResult；若仅 greedy 折叠则仍走旧逻辑。dspark_worker_v2.py 的 fold_eligible 条件新增 (sampling_info is None or sampling_info.is_all_greedy) 限制，因为折叠 epilogue 的图内 accept 是 greedy 的，采样批次必须走 eager accept 路径。
6. 配套配置：python/sglang/srt/environ.py 新增 DsparkFoldedSampling(IntEnum)（OFF/AUTO/FORCE）和 SGLANG_DSPARK_FOLDED_SAMPLING = EnvInt(DsparkFoldedSampling.AUTO) 环境变量。

测试方面：PR 没有新增或修改测试文件，仅通过 CI rerun 跑既有 DSPARK 用例（test_dspark_kernel_parity、test_dspark_draft_path_default、test_dspark_scheduler 等）验证回归。

关键文件：

- python/sglang/srt/speculative/dspark_components/dspark_draft_sampler.py（模块 草案采样器；类别 source；类型 core-logic；符号 greedy_step_sampler, DsparkDraftSampler, init, stage_sampling_params）：新增核心模块，承载 DsparkDraftSampler 的折叠采样实现、内存门控决策和构建入口
- python/sglang/srt/speculative/dspark_components/dspark_draft.py（模块 草案调度；类别 source；类型 core-logic；符号 greedy_step_sampler, DsparkDraftSampler, init, call）：草案编排核心，propose 逻辑从仅 greedy 扩展到采样折叠，并新增 stage_sampling_params 调用
- python/sglang/srt/environ.py（模块 环境配置；类别 source；类型 configuration；符号 DsparkFoldedSampling）：新增 DsparkFoldedSampling 枚举与环境变量 SGLANG_DSPARK_FOLDED_SAMPLING，是功能开关的配置入口
- python/sglang/srt/speculative/dspark_components/dspark_worker_v2.py（模块 草案视图；类别 source；类型 dependency-wiring）：将采样器导入切换到新模块，并收紧 fold_eligible 条件（采样批次必须走 eager accept）

关键符号：greedy_step_sampler, DsparkDraftSampler.init, DsparkDraftSampler.stage_sampling_params, DsparkDraftSampler.call, _resolve_folded_sampling, maybe_build_draft_sampler, DraftBlockProposer.propose, DraftBlockProposer._run_forward

关键源码片段

python/sglang/srt/speculative/dspark_components/dspark_draft.py

草案编排核心，propose 逻辑从仅 greedy 扩展到采样折叠，并新增 stage_sampling_params 调用

```
# python/sglang/srt/speculative/dspark_components/dspark_draft.py
# DraftBlockProposer.propose 的核心分支：决定草案走图折叠还是 eager。
# 折叠条件从原来的 all_greedy 放宽为 all_greedy 或打开了 folded_sampling，
# 因为采样行现在也可以在图中生成。

    def propose(self, *, batch, draft_input, verify_window, bs, device,
                target_model, sampling_info) -> DraftProposal:
        embed_module = target_model.get_input_embeddings()
        draft_sampler = self._draft_sampler
        all_greedy = sampling_info is None or sampling_info.is_all_greedy

        # 将 sampler 和采样信息传入 _run_forward，以便在图上刷新参数。
        fwd = self._run_forward(
            batch=batch, draft_input=draft_input, verify_window=verify_window,
            bs=bs, device=device, embed_module=embed_module,
            draft_sampler=draft_sampler, sampling_info=sampling_info,
        )
        draft_block_ids = fwd.draft_block_ids

        folded_confidence = None
        confidence_tap = None
        folded = False
        if (
            draft_sampler is not None
            and fwd.can_run_graph
            and (all_greedy or draft_sampler.folded_sampling)
        ):
            folded = True
            if draft_sampler.folded_sampling:
                # 采样折叠：直接从静态缓冲取掩码和 temperature，
                # 并从 corrected_out 取 markov 修正后的 block logits。
                greedy_mask = draft_sampler.greedy_mask[:bs]
                temperatures = draft_sampler.temperatures[:bs]
                corrected_logits = (
                    None if all_greedy
                    else draft_sampler.corrected_out[: bs * self.gamma].view(bs, self.gamma, -1)
                )
            else:
                # 仅 greedy 折叠：hook 已对每行做 argmax，但没保留采样缓冲，
                # 所以掩码和温度需要现场推导。
                greedy_mask = resolve_greedy_mask(bs=bs, sampling_info=sampling_info, device=
                device)
                if sampling_info is None:
                    temperatures = torch.ones(bs, dtype=torch.float32, device=device)
                else:
                    temperatures = (
```

```
                sampling_info.temperatures.view(-1)
                .to(torch.float32).clamp_min(1e-5)
            )
        corrected_logits = None
    draft_block = DraftBlockResult(
        draft_tokens=draft_sampler.out[: bs * self.gamma].view(bs, self.gamma),
        corrected_logits=corrected_logits,
        greedy_mask=greedy_mask,
        temperatures=temperatures,
    )
    if draft_sampler.confidence_out is not None:
        folded_confidence = draft_sampler.confidence_out[:bs]
else:
    # eager 路径：完整计算 base_logits 并逐 token 采样。
    with self._base_logits_context():
        base_logits, confidence_tap = self.draft_model.compute_base_logits(fwd.raw_hidden)
        base_logits = base_logits.view(bs, self.gamma, -1)
    draft_block = sample_draft_block(
        base_logits=base_logits,
        anchor_tokens=draft_block_ids[:, 0],
        draft_hidden=fwd.draft_hidden_3d,
        sampling_info=sampling_info,
        markov_head=self.draft_model.markov_head,
        device=device,
    )
return DraftProposal(
    draft_block_ids=draft_block_ids,
    draft_block=draft_block,
    draft_hidden=fwd.draft_hidden_3d,
    confidence=folded_confidence,
    confidence_tap=confidence_tap,
    folded=folded,
)
```

评论区精华

该 PR 没有 review 评论；issue 评论中主要是作者发起 /rerun-test 重跑 DSPARK 相关测试，以及 CI bot 回报全部通过的结果。团队没有留下设计讨论痕迹，核心权衡（AUTO 内存门控、accept 保持 eager）已在代码注释中说明。

- 暂无高价值评论线程

风险与影响

- 风险：风险集中在以下几个方面：
  - 显存占用风险：exp_noise 与 corrected_out 静态缓冲区大小为 max_bs * vocab 和 max_bs * gamma * vocab，在 vocab 较大（如 DeepSeek V4 的 129k）时会占用数 GB 显存。AUTO 门控在 target 初始化后探测剩余显存，但探测时机与 CUDA graph 捕

获之间的显存变化（如并发预留）可能造成误判；FORCE 模式若显存不足会直接 OOM。

  - 图内噪声语义风险：*exponential_()* 在 CUDA graph 回放时每次重放都会推进 RNG 状态，实现上依赖 SampleStepTokens.*execute* 对噪声的消费方式，若内核实现与该假设不一致会导致采样分布错误。
  - *accept* 路径行为变化：fold_eligible 新增 is_all_greedy 限制，采样批次即使草案已折叠，target 验证也必须走 *eager accept*，这与原先折叠后全图完成的性能预期有落差，但正确性优先。
  - 缺少新测试：没有为 folded sampling 新增专门测试，依赖既有 DSPARK 用例覆盖，采样路径的正确性（如 Gumbel 噪声分布、*corrected_logits* 与 *eager* 一致性）没有被显式断言。
- 影响：影响范围限定在 DSPARK 推测解码的草案生成路径：
  - 用户 / 模型：使用 DSPARK 且请求带采样的场景（temperature > 0 或 top_k > 1）将受益于图折叠加速，不再被迫走 *eager* 提案；默认 AUTO 保证显存不足时自动回退，行为安全。
  - 系统：每个 draft step 新增一次 host 侧 stage_sampling_params 写缓冲（小开销）和可能增加的静态显存占用；采样批次的 *accept* 仍为 *eager*，因此折叠图收益主要在 draft 侧。
  - 团队：新模块划分让 dspark_draft.py 保持编排职责，后续维护采样器逻辑更清晰；新增环境变量为线上 A/B 和强制开关提供了运维手段。
  - 风险标记：缺少专项测试覆盖，显存门控自动回退，采样 *accept* 保持 *eager*，新环境变量默认 AUTO

关联脉络

- PR #33276 Fix DSpark loading for hybrid DSV4 NVFP4: 同为 DSPARK/DeepSeek-V4 相关修复，改动 dspark_worker_v2.py 所在模块，本 PR 的导入调整可能与之存在冲突或协同。
- PR #33170 config: route parallel config-leaf reads through get_parallel(): 历史 PR 也修改了 dspark_worker_v2.py 与 model_loader，本 PR 在 dspark_worker_v2.py 中调整导入与 fold_eligible，两条线在 DSPARK worker 上重叠。
- PR #33294 test: stand up the config tiers two unit tests read from: 涉及 mm_utils 和配置层调整，与本 PR 的 environnement 变量读取方式间接相关，但无直接代码冲突。