PR #30096 完整报告

sgl-project/sglang

[DFLASH] Support grammar-constrained decoding in speculative verify

合并时间：2026-07-25 19:36

原文链接：http://prhub.com.cn/sgl-project/sglang/pull/30096

执行摘要

- 一句话：DFLASH 推测解码支持语法约束解码
- 推荐动作：值得精读。设计决策清晰：利用线性链退化树复用现有 EAGLE 机制，代码改动小但效果显著。同时暴露跨算法维护的挑战。

功能与动机

DFLASH 之前返回 HTTP 400 拒绝任意语法约束请求，导致无法用于 tool_choice, response_format 等结构化输出场景，PR 描述指出这是非常常见的服务路径。

实现拆解

1. 修改请求验证：validate_dflash_request 移除硬性拒绝逻辑，改为根据 spec_algorithm.supports_grammar_overlap() 条件判断，为 DFLASH 启用语法支持，DSPARK 仍被拒绝。
2. 新增线性链树构造：在 GrammarTree 类中添加 from_linear_chain 类方法，根据 draft_tokens 形状生成 retrieve_next_token 和 retrieve_next_sibling 张量，模拟 EAGLE 树退化形态。
3. 集成验证流程：在 DFlashWorkerV2.forward_batch_generation 中，当 batch.has_grammar 时调用 from_linear_chain 构建树，在目标模型前向后调用 build_grammar_vocab_mask 生成掩码，并在 apply_dflash_verify_logits_adjustments 之后应用 apply_vocab_mask 到 next_token_logits。
4. 测试配套：添加单元测试验证线性链遍历行为（正常顺序和语法拒绝停止）；添加集成测试覆盖 json_schema 和 regex 请求；调整测试类继承以复用 SpecGrammarKit。

关键文件：

- python/sglang/srt/speculative/dflash_worker_v2.py（模块 推测解码；类别 source；类型 core-logic）：核心验证路径，新增语法掩码构建和应用逻辑。
- python/sglang/srt/speculative/dflash_utils.py（模块 推测解码；类别 source；类型 core-logic；符号 validate_dflash_request）：修改请求验证函数，移除对语法约束的硬性拒绝。
- python/sglang/srt/speculative/spec_utils.py（模块 推测解码；类别 source；类型 core-logic；符号 from_linear_chain）：新增 GrammarTree.from_linear_chain 方法，为链式验证算法提供退化树。

- python/sglang/srt/managers/scheduler.py（模块 调度器；类别 source；类型 dependency-wiring）：调用 validate_dflash_request 时传入 spec_algorithm 参数。
- test/registered/unit/spec/test_spec_utils_traverse_tree.py（模块 测试；类别 test；类型 test-coverage；符号 _chain, test_linear_chain_visits_all_positions_in_order, test_linear_chain_stops_at_grammar_reject）：新增线性链单元测试，验证 from_linear_chain 和 traverse_tree 的正确性。
- test/registered/spec/dflash/test_dflash.py（模块 测试；类别 test；类型 test-coverage；符号 TestDFlashServerBase, test_grammar_logprob_count_matches_completion_tokens）：新增集成测试，验证 DFLASH 服务器对 json_schema 和 regex 请求的正确响应。
- python/sglang/srt/speculative/dflash_info.py（模块 推测解码；类别 source；类型 dependency-wiring）：导入调整，可能涉及类型注解。
- python/sglang/srt/speculative/spec_info.py（模块 推测解码；类别 source；类型 core-logic）：可能添加 supports_grammar_overlap 方法或修改枚举。

关键符号：validate_dflash_request, GrammarTree.from_linear_chain, DFlashWorkerV2.forward_batch_generation, build_grammar_vocab_mask, apply_vocab_mask

关键源码片段

python/sglang/srt/speculative/dflash_worker_v2.py

核心验证路径，新增语法掩码构建和应用逻辑。

```
# 导入新增的语法工具
from sglang.srt.speculative.spec_utils import (
    GrammarTree,
    assign_req_to_token_pool_func,
    build_grammar_vocab_mask,
)

# forward_batch_generation 方法签名新增 grammar_barrier 参数
class DFlashWorkerV2:
    def forward_batch_generation(
        self,
        batch: ScheduleBatch,
        on_publish=None,
        grammar_barrier=None, # 新增：语法屏障回调
    ) -> GenerationBatchResult:
        # ... 省略预填充和草稿生成部分 ...

        # 在草稿生成后、目标模型前向之前，构造线性链树
        grammar_tree = (
            GrammarTree.from_linear_chain(draft_tokens) if batch.has_grammar else None
        )

        # 目标模型前向 ...
        # 在目标模型前向后、采样之前，构建并应用词汇掩码
```

```
        vocab_mask = None
        if batch.has_grammar:
            if grammar_barrier is not None:
                grammar_barrier() # 等待上一批次语法 FSM 推进完成
            vocab_mask = build_grammar_vocab_mask(
                reqs=batch.reqs,
                verify_input=verify_input,
                tree=grammar_tree,
                sampling_info=batch.sampling_info,
                device=logits_output.next_token_logits.device,
            )

        # 应用 DFLASH 验证 logits 调整（温度、top-k 等）
        if sampling_info is not None:
            apply_dflash_verify_logits_adjustments(...)

        # 在 argmax/ 采样之前应用语法掩码
        if vocab_mask is not None:
            verify_input.grammar.apply_vocab_mask(
                logits=logits_output.next_token_logits,
                vocab_mask=vocab_mask,
            )
        # 之后的 accept 步骤读取被掩码的 logits，确保语法合规
```

python/sglang/srt/speculative/dflash_utils.py

修改请求验证函数，移除对语法约束的硬性拒绝。

```
# 新增导入
from sglang.srt.speculative.spec_info import SpeculativeAlgorithm

# 修改后的验证函数
def validate_dflash_request(
    req: Req, enable_overlap: bool, spec_algorithm: SpeculativeAlgorithm
) -> Optional[str]:
    if req.return_logprob:
        return "DFLASH speculative decoding does not support return_logprob yet."
    if enable_overlap and req.return_hidden_states:
        return "DFLASH speculative decoding does not support return_hidden_states yet."

    # 语法支持由 verify-time 位掩码和语法屏障共同提供
    if not spec_algorithm.supports_grammar_overlap() and (
        req.sampling_params.json_schema is not None
        or req.sampling_params.regex is not None
        or req.sampling_params.ebnf is not None
        or req.sampling_params.structural_tag is not None
    ):
        return (
            f"{spec_algorithm.name} speculative decoding does not support "
            "grammar-constrained decoding yet."
```

```
        )
    return None
```

python/sglang/srt/speculative/spec_utils.py

新增 GrammarTree.from_linear_chain 方法，为链式验证算法提供退化树。

```
class GrammarTree:
    # ... 已有 from_device, from_host ...

    @classmethod
    def from_linear_chain(cls, verify_ids_2d: torch.Tensor) -> GrammarTree:
        """
        为链式验证算法构造退化树：节点 i 的唯一子节点是 i+1。

        verify_ids_2d 形状为 (bs, chain_len)，列 0 是已提交 token。
        链接关系由形状固定，只需复制 token ID 到主机。
        """
        bs, chain_len = verify_ids_2d.shape
        # 子节点索引：除最后一列外指向下一列，最后一列无子节点
        next_token = torch.full((bs, chain_len), -1, dtype=torch.int64)
        next_token[:, :-1] = torch.arange(1, chain_len, dtype=torch.int64)
        # 无兄弟节点
        next_sibling = torch.full((bs, chain_len), -1, dtype=torch.int64)
        return cls.from_device(next_token, next_sibling, verify_ids_2d)
```

评论区精华

shanemort1982 指出该 PR 同时移除了对 DSPARK 的保护（validate_dflash_request 共享），但掩码仅添加在 DFLASH 路径，可能导致 DSPARK 静默输出无约束结果。作者已创建 #31753 单独修复 DSPARK，本 PR 仅聚焦 DFLASH。

- DSPARK 覆盖不全 (design): 作者确认并在 #31753 中单独修复 DSPARK，本 PR 仅针对 DFLASH。

风险与影响

- 风险：
  1. DSPARK 覆盖缺失：共享验证函数被修改但 DSPARK 未获得掩码，需配合 #31753 使用。
  2. 性能开销：语法掩码构建和 GPU 复制很小（链长通常 ≤8），且与目标前向重叠，影响可控。
  3. 兼容性：需确保 SpeculativeAlgorithm 所有枚举值覆盖 supports_grammar_overlap() 方法。 - 影响：用户侧：之前被拒绝的语法约束请求现在可正常工作并保持推测加速。系统侧：验证路径新增条件分支，仅在有语法约束时执行。团队侧：需与 DSPARK 修复 PR 配合部署。 - 风险标记：DSPARK 覆盖缺失，核心路径变更，需配合后续 PR

关联脉络

- PR #31753 [DSPARK] Grammar-constrained decoding, incl. tool_choice=auto: 该 PR 修复了本 PR 遗留的 DSPARK 语法支持问题，是本 PR 的延伸。
- PR #32393 [Spec] Share the grammar mask build and verify-tree staging across spec workers: 后续优化，进一步整合 grammar mask 构建逻辑，与本 PR 的复用方向一致。