PR #31041 完整报告

sgl-project/sglang

[Spec] Add LFM2 and LFM2-MoE DSpark speculative decoding support

合并时间：2026-08-31 11:04

原文链接：http://prhub.com.cn/sgl-project/sglang/pull/31041

执行摘要

- 一句话：为 LFM2 与 LFM2-MoE 新增 DSpark 投机解码支持
- 推荐动作：值得精读。两个设计决策很有参考价值：① 面对新旋转风格，没有新增第二套 KV 写入管线，而是把既有内核泛化为 ' 两种配对 '，并用 IS_NEOX=True 与旧路径逐位一致作为回归护栏；② radix 策略（extra_buffer）的 eligibility 必须尊重 hybrid 架构的 prefix-cache 恢复语义，宁可撤功能也不牺牲确定性输出。建议阅读时对照 dspark.py 的 _build_fused_kv_write_bundle 校验逻辑与 lfm2.py 的 shortconv_target_verify。另请留意：作者承诺的 test/registered/spec/dspark/ 门禁测试尚未落地，合并后应补上，并确认 LFM2 + DSpark 的 speculative tape 在配置分辨率下的保障路径。

功能与动机

PR body 说明这是 #30261（DSpark 落地 main）之后的功能补全：LFM2 系列是 ShortConv + attention 的 hybrid 模型，dense（LFM2）与 MoE（LFM2-MoE）都需要 DSpark 投机解码加速。LFM2 草稿是从 DSpark checkpoint 导出、使用 interleaved RoPE 的 Qwen3 风格 GQA 模型，而 fused KV 写入内核此前只支持 neox 旋转，导致这类草稿只能退回逐层 eager KV 写入的慢路径。作者强调 interleaved RoPE 对草稿不是可选项：' Serving them neox collapses accept length to ~1.9 vs ~5.1 interleaved'。目标侧还缺少 TARGET_VERIFY 阶段 ShortConv 状态的记录 / 回滚能力，以及 DFlash 所需的 aux hidden 捕获钩子。

实现拆解

1. 内核泛化：interleaved RoPE 写入支持。python/sglang/kernels/ops/speculative/dspark/fused_kv_write.py 的 _fused_kv_norm_rope_write_kernel 新增 IS_NEOX: tl.constexpr 编译期分支，neox 配对 (i, i + D/2) 与 interleaved 配对 (2i, 2i + 1) 只影响 RMSNorm 权重下标、旋转公式与写回位置；cos/sin 缓存布局两种风格一致，无需额外处理。宿函数 fused_kv_norm_rope_write 新增 is_neox_style=True 默认参数，保持既有 neox 草稿行为不变。对应地，python/sglang/srt/models/dspark.py 的 _build_fused_kv_write_bundle 去掉 ' 必须 neox 否则 bail' 的检查，改为逐层比对 is_neox_style 一致性，write_target_hidden_kv 把首层风格传给内核。这样 interleaved 草稿也能走 ' 一次线性投影 + 一次融合写内核直写 KV 池 ' 的快速路径。
2. LFM2 目标侧 TARGET_VERIFY：ShortConv 验证与回滚。python/sglang/srt/models/lfm2.py 新增 register_shortconv_verify_buffers（预分配

256 槽 _intermediate_state_indices 磁带槽位缓冲，non-persistent）与 shortconv_target_verify（把 [bs * block, hidden] 的 Bx reshape 为 [bs, hidden, block] 后调用 Triton causal_conv1d_update，并把每一步卷积窗口写入 MambaPool.SpeculativeState.intermediate_conv_window，支撑验证后按接受边界回滚）。 Lfm2ShortConv.forward 增加 is_target_verify() 分支，构造时注册缓冲。 python/sglang/srt/models/lfm2_moe.py 直接复用 lfm2.py 导出的这两个函数， Lfm2MoeShortConv 走同一分支，arch 参数传 LFM2-MoE 以获得准确报错。

3. DFlash aux hidden 捕获钩子。两个模型文件的 DecoderLayer 增加 _is_layer_to_capture 标记与 captured_last_layer_outputs 参数；Lfm2Model / Lfm2MoeModel 维护 layers_to_capture 列表，forward 循环按标记把层输入追加进 aux_hidden_states，循环结束后若请求捕获末层输出则补录；捕获开启且有内容时返回 (hidden_states, aux_hidden_states) 元组，否则返回裸张量（调用方用 isinstance 收窄）。 Lfm2ForCausalLM / Lfm2MoeForCausalLM 暴露 capture_aux_hidden_states 属性与 set_dflash_layers_to_capture（非最后 PP rank 直接返回；层号 +1 偏移，因为捕获第 L 层输入等价于第 L-1 层输出），并把 aux_hidden_states 转发给 logits_processor。
4. 草稿架构注册。新文件 python/sglang/srt/models/lfm2_dspark.py 定义 Lfm2DSparkDraftModel(DSparkDraftModel) 并通过 EntryClass 注册。当前 LFM2 草稿是纯 attention 的 Qwen3 风格 GQA，薄子类即可；独立 arch 为未来草稿侧加入 ShortConv 等 LFM2 专属层预留扩展点。
5. 测试、配置与部署配套。本 PR 未新增测试文件；作者在 body 中提出可加 test/registered/spec/dspark/ 门禁测试，最终未包含。CI 通过 rerun 既有 DSpark 测试（test_dspark_stacked_ctx_kv_parity.py、test_dspark_kernel_parity.py、test_basic_sanity_dspark.py）验证内核与验证路径，性能与精度在 1xH100 手工验证。演化过程中曾包含 overrides.py extra_buffer 列表与 dspark_worker_v2.py 改动，均在 review / rebase 中撤下。

关键文件：

- python/sglang/srt/models/lfm2.py（模块 模型实现；类别 source；类型 data-contract；符号 register_shortconv_verify_buffers, shortconv_target_verify, set_dflash_layers_to_capture, capture_aux_hidden_states）：目标侧核心改造：TARGET_VERIFY 的 ShortConv 验证 / 回滚、DFlash aux hidden 捕获，改动量与信号最高。
- python/sglang/srt/models/lfm2_moe.py（模块 模型实现；类别 source；类型 data-contract；符号 set_dflash_layers_to_capture, get_input_embeddings, capture_aux_hidden_states）：MoE 目标复刻同一套 TARGET_VERIFY 与捕获钩子，通过复用 lfm2.py 导出的工具函数保持两实现一致。
- python/sglang/kernels/ops/speculative/dspark/fused_kv_write.py（模块 内核层；类别 infra；类型 infrastructure；符号 _fused_kv_norm_rope_write_kernel, fused_kv_norm_rope_write）：Triton 内核新增 IS_NEOX 编译期分支，interleaved 草稿得以走融合写快速路径；neox 路径要求 bit 级兼容。
- python/sglang/srt/models/dspark.py（模块 草稿模型；类别 source；类型 core-logic；符号 _build_fused_kv_write_bundle, write_target_hidden_kv）：放开 neox bail 的 bundle

校验，改为逐层风格一致性检查，并把 is_neox_style 传入内核，是内核改动与草稿框架的接缝。

- python/sglang/srt/models/lfm2_dspark.py（模块 草稿模型；类别 source；类型 entrypoint；符号 Lfm2DSparkDraftModel）：注册 Lfm2DSparkDraftModel 独立架构，为 LFM2 草稿后续演化预留扩展点。

关键符号：shortconv_target_verify, register_shortconv_verify_buffers, set_dflash_layers_to_capture, capture_aux_hidden_states, fused_kv_norm_rope_write, _build_fused_kv_write_bundle, write_target_hidden_kv, Lfm2DSparkDraftModel

关键源码片段

python/sglang/srt/models/lfm2.py

目标侧核心改造：TARGET_VERIFY 的 ShortConv 验证 / 回滚、DFlash aux hidden 捕获，改动量与信号最高。

```
# python/sglang/srt/models/lfm2.py
def shortconv_target_verify(
    conv: nn.Module,
    Bx: torch.Tensor,
    meta: 'ShortConvMetadata',
    draft_token_num: int,
    arch: str,
) -> torch.Tensor:
    # 在 TARGET_VERIFY 块上执行 depthwise short conv，并把每一步卷积窗口
    # 写入 speculative tape，使验证后能按接受边界回滚状态
    # 硬性要求池携带 speculative 磁带；缺少时直接报错，避免静默丢状态
    assert isinstance(meta.layer_cache, MambaPool.SpeculativeState), (
        f'{arch} TARGET_VERIFY needs the speculative conv tape; the mamba pool '
        'was built without speculative_num_draft_tokens.'
    )
    bs = meta.cache_indices.shape[0]
    # [bs * block, hidden] -> [bs, block, hidden] -> [bs, hidden, block],
    # 转置后才是 Triton update 内核期望的 [bs, channels, block] 布局
    Bx_reshaped = Bx.view(bs, draft_token_num, -1).transpose(1, 2)
    # 预分配的磁带槽位下标缓冲按需扩容
    if conv._intermediate_state_indices.shape[0] < bs:
        conv._intermediate_state_indices = torch.arange(
            bs, dtype=torch.int32, device=Bx.device
        )
    conv_out = causal_conv1d_update_triton(
        Bx_reshaped,
        meta.layer_cache.conv[0],
        conv.conv_weight,
        conv.conv_bias,
        activation=None,
        conv_state_indices=meta.cache_indices,
        intermediate_conv_window=meta.layer_cache.intermediate_conv_window[0],
```

```
            intermediate_state_indices=conv._intermediate_state_indices[:bs],
        )
        # 还原为 [bs * block, hidden]，交给上层 out_proj 完成 C 门控
        return conv_out.transpose(1, 2).reshape(bs * draft_token_num, -1)
```

python/sglang/kernels/ops/speculative/dspark/fused_kv_write.py

Triton 内核新增 IS_NEOX 编译期分支，interleaved 草稿得以走融合写快速路径；neox 路径要求 bit 级兼容。

```
# python/sglang/kernels/ops/speculative/dspark/fused_kv_write.py
# IS_NEOX 是编译期常量，决定 RoPE 的旋转配对方式：
# neox（默认）把维度 i 与 i + D/2 配对；interleaved（GPT-J 风格）把 2i 与 2i + 1 配对。
# 两种风格的 cos/sin 缓存布局一致，因此内核只需要按配对切换维度下标。
if IS_NEOX:
    off1 = half_ar
    off2 = HALF + half_ar
else:
    off1 = 2 * half_ar
    off2 = 2 * half_ar + 1

cos = tl.load(cos_sin_ptr + pos * D + half_ar).to(tl.float32)
sin = tl.load(cos_sin_ptr + pos * D + HALF + half_ar).to(tl.float32)
# K 的 RMSNorm 权重也需按同一配对顺序读取，保证归一化与旋转作用于同一组维度
knw1 = tl.load(knw_ptr + l * D + off1).to(tl.float32)
knw2 = tl.load(knw_ptr + l * D + off2).to(tl.float32)

# 每个 attention head 各自执行：RMSNorm -> 按配对拆分 -> RoPE -> 写回 KV 池
k = tl.load(row + h * D + d_ar).to(tl.float32)
ms = tl.sum(k * k, 0) / D
inv = 1.0 / tl.sqrt(ms + EPS)
k1 = tl.load(row + h * D + off1).to(tl.float32) * inv * knw1
k2 = tl.load(row + h * D + off2).to(tl.float32) * inv * knw2
# 将 K 分量回落到 bf16 再参与旋转，保持与既有参考路径一致的数值行为
k1 = k1.to(tl.bfloat16).to(tl.float32)
k2 = k2.to(tl.bfloat16).to(tl.float32)
o1 = k1 * cos - k2 * sin
o2 = k2 * cos + k1 * sin
tl.store(k_buf + loc * ks0 + h * D + off1, o1.to(tl.bfloat16))
tl.store(k_buf + loc * ks0 + h * D + off2, o2.to(tl.bfloat16))
# V 不参与 norm/rope，按原维度顺序直接写回
v = tl.load(row + KV + h * D + d_ar)
tl.store(v_buf + loc * vs0 + h * D + d_ar, v)
```

python/sglang/srt/models/lfm2_dspark.py

注册 Lfm2DSparkDraftModel 独立架构，为 LFM2 草稿后续演化预留扩展点。

```
# python/sglang/srt/models/lfm2_dspark.py
# LFM2 系列的 DSpark 草稿模型。
# 独立成 arch 的目的是让 LFM2 草稿未来可以加入 ShortConv 等 LFM2 专属层，
```

```
# 而不必改动共享的 DSparkDraftModel；当前 checkpoint 是纯 attention 的
# Qwen3 风格 GQA + interleaved RoPE，因此这里只是一个薄子类。
from sglang.srt.models.dspark import DSparkDraftModel


class Lfm2DSparkDraftModel(DSparkDraftModel):
    pass


EntryClass = [Lfm2DSparkDraftModel]
```

评论区精华

核心交锋集中在两条。其一，kpham-sgl 建议跳过在 dspark_kv_inject.py 中新增的 fused-KV 辅助管线，直接让已有 DSpark 内核学会 interleaved RoPE；并指出新增 workspace 路径会让 Qwen3 草稿从 ' 一次 F.linear + 一次 fused_kv_norm_rope_write 直写 KV 池 ' 退化为 'torch.mm 写 workspace + 独立 norm/rope 内核 + 逐层 set_kv_buffer 回读 '，每次 verify 多出整份 K/V 的写读与 N 次 kernel launch，属于回归。tugot17 采纳并回退，改为教内核两种配对，且确认 IS_NEOX=True 与旧内核逐位一致。其二，kpham-sgl 质疑 LFM2 / LFM2-MoE 加入 extra_buffer radix 策略列表不安全（ShortConvAttn 如 Inkling 不支持），tugot17 实测确认强制开启会在 prefix-cache 命中时破坏确定性输出（temp=0 下 3 个 prompt 有 2 个文本变化），随即撤掉 overrides 与配套测试改动。

- fused KV 写入路径：新增 workspace 辅助 vs 泛化内核 (performance): tugot17 回退该路径，改为教会既有写入内核两种旋转配对，并确认 IS_NEOX=True 与旧内核逐位一致；Qwen3 继续走 bundle 快速路径。
- LFM2 是否可进入 extra_buffer radix 策略列表 (correctness): overrides.py 两行与配套 test_model_overrides.py 改动全部撤下，LFM2 不进入 extra_buffer 列表，safe 路径保持 no_buffer + 关闭 overlap 调度。
- rebase 后与 #32828 重复的 dspark_worker_v2 混合状态提交 (design): 保留 main 上的通用实现，PR 只保留 LFM2 专属部分与内核改动，文件数从 7 降到 5。
- test_model_overrides.py 负例失效 (testing): tugot17 曾按建议修复；后因 overrides 行整体回退，测试文件一并还原，问题随之消失。
- lint 失败与 CI 标签协作 (other): 作者修复后 lint 转绿；run-ci 与 rerun 由维护者协作完成，最终既有 DSpark parity 测试通过。

风险与影响

- 风险： 1) 内核 bit 兼容性：IS_NEOX=True 分支必须与旧内核逐位一致，作者仅在随机输入上手工验证，本 PR 未固化该断言为自动化测试，后续内核改动可能静默破坏 Qwen3 等 neox 草稿。 2) 缺少新增测试：LFM2 DSpark 的验证 / 回滚路径依赖手工 H100 基准，作者在 body 中提出可加 test/registered/spec/dspark/ 门禁测试，最终未落地。 3) 状态磁带配置依赖：shortconv_target_verify 对 MambaPool.SpeculativeState 是硬断言，池未按 speculative_num_draft_tokens 构建会直接报错；而最终 PR 撤掉了 overrides.py 中 extra_buffer 列表改动，LFM2 + DSpark 下 speculative conv tape 如何保证配置，在可

用材料中没有直接证据，需合并后核查 radix 策略决议路径。 4) 模型返回契约变更：开启捕获后 Lfm2Model.forward 返回元组，调用方依赖 isinstance 收窄；任何假定裸张量的外部调用需要适配。

- 影响：用户侧：LFM2.5-1.2B 与 LFM2.5-8B-A1B 通过公开草稿路径即可一键启用 DSpark，长推理与代码场景获得约 2x 加速且精度持平；interleaved RoPE 内核支持也为未来非 neox DSpark 草稿铺平道路。系统侧：DSpark fused KV 写入从 ' 仅 neox' 泛化为 ' 两种配对 '，dspark.py 的 bundle 校验从一刀切 bail 改为逐层风格一致性检查；hybrid 模型第一次具备 TARGET_VERIFY 阶段可回滚的 ShortConv 状态记录（借助 speculative conv tape）。团队侧：shortconv_target_verify 与 register_shortconv_verify_buffers 被 dense / MoE 两个实现复用，形成标准接入范式；独立 Lfm2DSparkDraftModel arch 为草稿侧后续演化预留扩展点。影响面集中在 LFM2 系列 + DSpark 组合，既有 neox 草稿有 bit 级兼容护栏。
- 风险标记：投机解码核心内核变更，缺少新增自动化测试，依赖 speculative conv tape 配置，模型返回契约变更，extra_buffer 策略回退留待核实

关联脉络

- PR #30261 DSpark speculative decoding support: 本 PR 的基座，PR body 明确标注 follow-up；DSpark 落地 main 后才可能接入 LFM2。
- PR #30776 LFM2 DSpark support (superseded): 本 PR 的前身，原开在 DSpark PR 分支上，合入前被本 PR 取代并 rebase 到 main（body 中明确 Supersedes #30776）。
- PR #30780 LFM2-MoE serving defaults parity: 与 DSpark 无关但配套：为 Lfm2MoeForCausalLM 提供 attention-backend / radix-cache override 表，作者建议与本 PR 配对使用以实现 8B 目标的 serving 默认值。
- PR #32828 generic post-verify hybrid state commit: rebase 讨论中 kpham-sgl 指出已有部分改动合入 main；tugot17 发现 #32828 已通用实现 post-verify hybrid state commit 且覆盖更多，遂删除自己的 dspark_worker_v2.py 改动。