PR #30964 完整报告

sgl-project/sglang

[AMD] Support DeepSeek V4 DSpark on AMD HIP platform

合并时间：2026-08-09 06:22

原文链接：http://prhub.com.cn/sgl-project/sglang/pull/30964

执行摘要

- 一句话：DeepSeek V4 DSpark 支持 AMD HIP，打通 unified KV ring 注入
- 推荐动作：值得精读。核心看点：_unified_inject_loc 的 ring 寻址与双重 skip 设计、HIP 后端 ragged verify 图 key 改造（token-tier graph key）、以及围绕 nightly-only 覆盖缺口的 CI 流程讨论。若要在 AMD 上部署 DSV4 DSpark，需同时跟进 #34147 保证夜间覆盖，并在后续改动中保持 target_verify_num_draft_tokens 与 server args 的 gamma 约定一致。

功能与动机

PR body 开篇明确说明动机："Follow https://github.com/sgl-project/sglang/pull/30261, support dspark for deepseek v4 on AMD platform"。此前 HIP 后端（DeepseekV4HipRadixBackend）在 ragged verify 的 CUDA graph 路径直接抛 NotImplementedError，DSpark 在 ROCm 上完全不可用；同时 DSpark draft 的 KV 存储与 target-hidden 注入仅支持非 unified 的 fp8 swa_kv_pool，无法利用 unified_kv_triton 的共享 bf16 ring，吞吐与显存都受限制。本 PR 需要同时补齐 HIP ragged verify 图支持与 unified KV 写入通道，并为 ROCm 构建未注册的 sgl_kernel 算子提供 torch 回退。

实现拆解

1. HIP 后端打通 ragged verify CUDA graph

涉及 python/sglang/srt/layers/attention/deepseek_v4_backend_hip_radix.py：

- 新增类属性 supports_ragged_verify_graph = True，移除 TARGET_VERIFY bucket 中针对 ragged verify 的 NotImplementedError 守卫。
- __init__ 新增 is_dspark_draft 标志与 target_verify_num_draft_tokens = speculative_num_draft_tokens - 1：DSpark draft worker 只验证 gamma 行，服务端参数沿用 CUDA 侧 "gamma + 1" 约定，避免原地改属性导致下游 5 处用法隐式消费 gamma（对应 review 意见）。
- init_forward_metadata_target_verify_old 接受 ragged_layout：通过 ExpandPrefillCausally 在 GPU 上做 prefill 展开，新增 compress_gpu_plan / extend_start_loc 支持 GPU 侧压缩计划；eager 路径回退到 sum(verify_lens)。
- init_forward_metadata_out_graph 的 TARGET_VERIFY bucket 用 ragged_layout.padded_to_bucket() 后的 graph_num_tokens 作为 token-tier 图 key（

graph_key = num_tokens_v）；make_forward_metadata_from_raw_verify 同步改用 target_verify_num_draft_tokens。

2. unified KV ring 写入通道

涉及 python/sglang/kernels/ops/attention/dsv4/unified_kv_kernels/runtime.py 与 python/sglang/srt/mem_cache/deepseek_v4_memory_pool.py：

- 新增 Triton _scatter_loc_kernel 与包装函数 scatter_bf16_into_unified：逐行把已 norm + RoPE 的 bf16 K scatter 进 unified_kv[loc]，loc < 0 的行跳过（即按行 commit 掩码）。
- DeepSeekV4TokenToKVPool 新增 set_unified_key_buffer_radix_fused_norm_rope：先 fused_norm_rope_inplace（与主模型 _compute_kv_bf16 同一 freqs_cis 路径），再 scatter 进 get_unified_kv(layer_id)，签名与非 unified 版本完全一致。

3. 模型与注入 / 验证路径路由

- python/sglang/srt/models/deepseek_v4_dspark.py：_store_block_kv 在 is_unified_kv_triton() 时改用 get_unified_swa_loc（逐位置实时重算，支持多步 draft decode）+ unified 写入；write_target_hidden_kv 选择 unified store 函数，swa_loc/positions 契约不变。
- python/sglang/srt/speculative/dspark_components/dspark_kv_inject.py：新增 _unified_inject_loc，地址公式 loc = state_slot * ring + pos % ring，叠加两层 skip（SWA 窗口裁剪旧 token 防 ring 槽位竞争、commit 门控丢未验收 token）；inject_ragged 分流到 build_unified_commit_inject_layout。
- python/sglang/srt/speculative/dspark_components/dspark_verify.py：commit_hidden 为 unified 模式组装逐 token state_slot（由 batch.req_pool_indices 展开）；_commit_inject 分流到 unified 布局构建。
- python/sglang/srt/speculative/dspark_components/dspark_worker_v2.py：_forward_prefill 用 repeat_interleave 为每个 prefill token 计算 state_slot 与 final_pos，使注入器能裁剪窗口外 token——长 prefill chunk 下这些 token 会共享 ring 槽位造成覆盖。

4. ROCm 采样回退与测试配套

- python/sglang/srt/speculative/dflash_utils.py：为 top-k / top-p 概率重归一化增加 HIP torch 回退，CUDA/MUSA 行为不变。
- 新增 test/registered/amd/test_deepseek_v4_pro_fp4_dspark.py：两个 kernel 单测用参考实现逐元素比对（test_build_unified_commit_inject_layout、test_scatter_bf16_into_unified）；8 卡 MI35x 全量 GSM8K（1319 题，5-shot）e2e 注册为 nightly 套件，门禁 accuracy > 0.92 且 avg_spec_accept_length > 3.0。

关键文件：

- test/registered/amd/test_deepseek_v4_pro_fp4_dspark.py（模块 端到端测试；类别 test；类型 test-coverage；符号 TestDSparkUnifiedKVKernelsAMD, test_build_unified_commit_inject_layout, test_scatter_bf16_into_unified, TestDeepseekV4DSparkUnifiedKVGSM8K）：本 PR 唯一的专用验证：两个 unified KV

kernel 单测（参考实现逐元素比对）+ 8 卡 MI35x 全量 GSM8K e2e，注册为 nightly 套件；也是 PR CI 覆盖缺口讨论的中心文件。

- python/sglang/srt/speculative/dspark_components/dspark_kv_inject.py（模块 投机注入；类别 source；类型 dependency-wiring；符号 _unified_inject_loc, inject_target_hidden, inject_ragged, _inject_mla）：DSpark target-hidden 注入的核心入口：新增 _unified_inject_loc 实现 ring 寻址与 SWA/commit 双重 skip，inject_ragged 分流到 unified 布局构建，是 unified KV 数据契约落地的关键。
- python/sglang/srt/layers/attention/deepseek_v4_backend_hip_radix.py（模块 注意力后端；类别 source；类型 dependency-wiring；符号 DeepseekV4HipRadixBackend, init_forward_metadata_prefill, init_forward_metadata_target_verify_old, init_forward_metadata_out_graph）：HIP 后端 DSpark 支持的核心：supports_ragged_verify_graph 开启、显式 gamma 语义（target_verify_num_draft_tokens）、GPU prefill 展开与压缩计划、token-tier 图 key，是本 PR 改动量最大的源码文件。
- python/sglang/srt/mem_cache/deepseek_v4_memory_pool.py（模块 KV 缓存；类别 source；类型 core-logic；符号 set_unified_key_buffer_radix_fused_norm_rope）：unified KV 写入通道的核心：新增 set_unified_key_buffer_radix_fused_norm_rope，把 DSpark draft 的 norm+RoPE 后 bf16 K 写入 bf16 ring，替代非 unified 的 fp8 swa_kv_pool 路径。
- python/sglang/srt/models/deepseek_v4_dspark.py（模块 模型层；类别 source；类型 data-contract；符号 _store_block_kv, write_target_hidden_kv）：DSpark 模型层的数据契约改造：_store_block_kv 与 write_target_hidden_kv 在 unified 模式下改写 ring，保持 swa_loc/positions 外部契约不变。
- python/sglang/srt/speculative/dspark_components/dspark_verify.py（模块 投机验证；类别 source；类型 dependency-wiring；符号 commit_hidden, _commit_inject）：验证提交注入的适配点：commit_hidden 为 unified 模式传递逐 token state_slot, _commit_inject 选择 unified commit-inject 布局。
- python/sglang/srt/speculative/dspark_components/dspark_worker_v2.py（模块 投机工作器；类别 source；类型 dependency-wiring；符号 _forward_prefill）：prefill 注入的数据准备：为每个 token 计算 state_slot 与 final_pos，保证注入器能裁剪 SWA 窗口外的旧 token，避免 ring 槽位竞争。
- python/sglang/kernels/ops/attention/dsv4/unified_kv_kernels/runtime.py（模块 内核层；类别 infra；类型 infrastructure；符号 _scatter_loc_kernel, scatter_bf16_into_unified）：新增 Triton scatter 内核，是 unified ring 写入的底层算子，被 memory pool 与注入路径共同依赖。
- python/sglang/kernels/ops/speculative/dspark/dspark_verify_window.py（模块 内核层；类别 infra；类型 infrastructure；符号 build_unified_commit_inject_layout）：新增 build_unified_commit_inject_layout：unified 模式下 commit-inject 布局的静态 shape 构建器，CUDA graph 安全。
- python/sglang/srt/speculative/dflash_utils.py（模块 投机采样；类别 source；类型 compatibility）：ROCm 兼容性回退：top-k / top-p 概率重归一化在 HIP 上无法使用

sgl_kernel 算子时退回 torch 实现，保证采样正确性。

关键符号：scatter_bf16_into_unified, _scatter_loc_kernel, set_unified_key_buffer_radix_fused_norm_rope, _unified_inject_loc, inject_target_hidden, inject_ragged, build_unified_commit_inject_layout, init_forward_metadata_target_verify_old, init_forward_metadata_out_graph, make_forward_metadata_from_raw_verify, commit_hidden, _commit_inject, _store_block_kv, write_target_hidden_kv

关键源码片段

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

DSpark target-hidden 注入的核心入口：新增 _unified_inject_loc 实现 ring 寻址与 SWA/commit 双重 skip，inject_ragged 分流到 unified 布局构建，是 unified KV 数据契约落地的关键。

```
# dspark_kv_inject.py —— unified_kv 下 target-hidden 注入的 ring 行地址计算

def _unified_inject_loc(
    self,
    *,
    pool,
    positions: torch.Tensor,
    cache_loc_2d: Optional[torch.Tensor],
    commit_lens: Optional[torch.Tensor],
    state_slot: Optional[torch.Tensor],
    final_pos: Optional[torch.Tensor],
) -> torch.Tensor:
    """unified_kv 模式下 target-hidden 注入的 ring 行地址计算。

    地址公式为 loc = state_slot * ring + pos % ring，并带两层 skip（-1）规则：
      * SWA 窗口裁剪：每个请求只有最后 win 个 token 落在 ring 内；更早的 token
        会与窗口内 token 共享 ring 槽位（pos % ring 相同）造成互相覆盖，所以
        直接丢弃——长 prefill chunk 场景必须依赖此规则。
      * commit 门控：未通过验收的 verify token（col >= commit_len）丢弃。
    """
    if state_slot is None:
        raise RuntimeError(
            "unified_kv target-hidden injection requires state_slot "
            "(per-token draft req_pool_indices)."
        )
    ring = pool.unified_swa_ring_size
    win = pool.unified_swa_window
    pos = positions.to(torch.int64)
    loc = state_slot.to(torch.int64) * ring + pos % ring
    if final_pos is not None:
        # 只保留窗口内的最新 token，其余置为 -1 由 scatter 跳过
        keep = pos > (final_pos.to(torch.int64) - win)
```

```
        loc = torch.where(keep, loc, torch.full_like(loc, -1))
    if commit_lens is not None and cache_loc_2d is not None:
        # 按请求展开列索引，未 commit 的列整体置 -1
        bs, verify_len = cache_loc_2d.shape
        col = torch.arange(verify_len, device=positions.device).view(1, -1)
        committed = (col < commit_lens.to(torch.long).view(-1, 1)).reshape(-1)
        loc = torch.where(committed, loc, torch.full_like(loc, -1))
    return loc.to(torch.int32)
```

python/sglang/kernels/ops/attention/dsv4/unified_kv_kernels/runtime.py

新增 Triton scatter 内核，是 unified ring 写入的底层算子，被 memory pool 与注入路径共同依赖。

```
# unified_kv_kernels/runtime.py —— bf16 K 的 ring scatter 内核与包装函数

@triton.jit
def _scatter_loc_kernel(
    kv_ptr, # [T, D] bf16，已 norm + rope 的 draft K
    loc_ptr, # [T] int，unified ring 行号；< 0 表示跳过
    unified_ptr, # [pages, D] bf16，目标 unified KV buffer
    n_rows,
    D: tl.constexpr,
    BLOCK_D: tl.constexpr,
):
    row = tl.program_id(0)
    if row >= n_rows:
        return
    loc = tl.load(loc_ptr + row).to(tl.int64)
    if loc < 0:
        return # 未 commit 或被窗口裁剪的 token，跳过写入
    offs = tl.arange(0, BLOCK_D)
    mask = offs < D
    vals = tl.load(kv_ptr + row * D + offs, mask=mask, other=0.0)
    tl.store(unified_ptr + loc * D + offs, vals, mask=mask)


def scatter_bf16_into_unified(
    *,
    kv: torch.Tensor, # [T, head_dim] bf16，已做 norm + RoPE
    loc: torch.Tensor, # [T] int32/int64，ring 行号；< 0 跳过
    unified_kv: torch.Tensor, # [pages, head_dim] bf16
) -> None:
    """把已 norm + rope 的 bf16 K scatter 进 unified_kv[loc]，loc < 0 跳过。

    与 store_swa_into_unified 配套：本函数面向已持有预计算 ring 行号的调用方
    （DSpark draft 的 get_unified_swa_loc、或 commit-inject 布局），
    用 loc == -1 表达按行的 commit 掩码，避免单独传 mask。
    """
    n_rows, D = kv.shape
```

```
    if n_rows == 0:
        return
    assert kv.is_contiguous() and kv.dtype == unified_kv.dtype
    assert loc.is_contiguous()
    assert unified_kv.is_contiguous()
    _scatter_loc_kernel[(n_rows,)](
        kv,
        loc,
        unified_kv,
        n_rows,
        D=D,
        BLOCK_D=triton.next_power_of_2(D),
        num_warps=8,
    )
```

评论区精华

核心 review 交锋集中在三处：

1. kkHuang-amd 指出 __init__ 中原地修改 speculative_num_draft_tokens -= 1 是 action-at-a-distance，建议镜像 CUDA 后端的局部变量模式；head 版本已改为显式 is_dspark_draft + target_verify_num_draft_tokens，并在注释中说明 CUDA 约定 gamma+1，同时提醒核对 graph buffer sizing。
2. 1am9trash 提醒新测试应复用 PR#28920 清理后的默认环境变量，避免硬编码导致未来默认值变更时逐文件同步；作者已按建议更新并附上 CI 结果。
3. amd-bot 多次给出同一结论：本 PR 唯一专用测试是 nightly-only 且路径被 SGLANG_HACK_FLASHMLA_BACKEND=unified_kv_triton 门控，PR CI 完全不验证变更代码，合入前必须由作者在 MI35x 上跑通 nightly 套件；michaelzhang-ai 随后手动调度该套件通过（GSM8K accuracy 0.9515 > 0.92，avg_spec_accept_length 3.730 > 3.0），并指出套件未挂入任何 nightly workflow、已提交 #34147 补排程。

- speculative_num_draft_tokens 原地减一（gamma off-by-one 风险） (design): head 版本已改为显式 is_dspark_draft 标志 + target_verify_num_draft_tokens 局部语义，与 CUDA 模式一致。
- commit_hidden 中的死变量 (style): 已清理，reviewer 最终 APPROVED（LGTM）。
- 测试环境变量应默认化而非硬编码 (design): 作者已按建议更新环境变量并附上 CI 结果。
- nightly-only 覆盖缺口与合入门禁 (testing): michaelzhang-ai 手动调度 nightly 套件（run 31291406552）通过：GSM8K accuracy 0.9515 > 0.92、avg_spec_accept_length 3.730 > 3.0，满足门禁。
- nightly 套件未挂入排程（coverage dormant） (other): 待 #34147 合入后闭环；本 PR 合入时该缺口仍存在。

风险与影响

- 风险：

  - 测试覆盖空白（高风险）：PR CI 不执行任何本 PR 的代码路径；合入时 nightly 排程也缺失（#34147 未合入），覆盖实际处于休眠状态，回归只能靠夜间任务兜底。
  - 门控依赖环境变量：is_unified_kv_triton() 依赖 SGLANG_HACK_FLASHMLA_BACKEND，unified / 非 unified 双路径增加组合数，新布局代码只在 unified 模式激活，普通 CI 无法触达。
  - gamma 语义耦合：target_verify_num_draft_tokens 与 graph runner 读取的 server_args.speculative_num_draft_tokens（未减一）必须始终对齐，kkHuang 也提醒检查 graph buffer sizing。
  - 分支卫生：64 个 commit、约 20+ 次 main merge、多次 revert/re-apply（如 CUTLASS FP8 相关 revert），存在冲突解决引入无关变更的风险。
  - 性能：scatter 内核每行一个 program、num_warps = 8，极端 shape 下可能成为瓶颈；但实测吞吐提升明显，风险可控。
- 影响：
  - 对 AMD 用户：MI35x 等 ROCm 平台首次获得 DSV4 Pro DSpark 投机解码支持，TP8 单并发 TTT 最高 2.93x、TP8DP8 256 并发 unified 后端 1.37x 且 TTFT 改善 2.4x 以上；实现为 HIP-only，不影响 CUDA/MUSA 路径。
  - 对系统：改动横跨 attention backend、mem cache、模型层、speculative 三组件与两个 kernel 模块，"state_slot * ring + pos % ring" 成为 DSV4 DSpark 注入的新数据契约。
  - 对团队流程：暴露并推动了 nightly 套件排程补齐（#34147），也示范了 env-gated HIP 特性的合入门禁做法。
  - 风险标记：仅 nightly 覆盖，PR CI 不验证，nightly 排程缺口待 #34147 补齐，HIP 专属 + 环境变量门控路径，投机解码核心路径变更，64 commits 长分支，多次 main 合并

关联脉络

- PR #30261 DeepSeek V4 DSpark support (CUDA baseline): 本 PR 的移植基准：PR body 明确写 Follow #30261，将 CUDA 侧 DSpark 逻辑镜像到 HIP，并沿用其 gamma+1 参数约定。
- PR #34147 Add nightly-8-gpu-mi35x-deepseek-v4-pro-dspark-rocm720 job: michaelzhang-ai 在评论中提交的补排程 PR：把本 PR 的 nightly 套件真正挂入 nightly-test-amd 工作流，否则覆盖处于休眠状态。
- PR #31705 Fix DeepSeek-V4 CUDA sparse-prefill crash: amd-bot CI 报告指出 PR CI 唯一真实失败是既存 DSV4 CUDA sparse-prefill 崩溃，修复在 #31705 进行，与本 PR 无代码交集但影响合入判定。
- PR #31531 [Refactor] Separate ROCm-specific DeepSeek MHA and MLA forward paths: 同属 AMD + DeepSeek 功能线，为 ROCm 侧 DeepSeek 注意力路径的独立性打下基础，本 PR 的 HIP 后端改造与其同一演进脉络。
- PR #33892 [Fix] Speculative decoding crashes with DP-Attention: 投机解码与 DP-Attention 组合的稳定性修复；本 PR 的 TP8DP8 压测数据同样验证了 DSpark 与 DP-Attention 在 AMD 上的组合可用性。