PR #33954 完整报告

sgl-project/sglang

[Diffusion] perf: build qwen's masked varlen metadata host-side

合并时间：2026-08-07 17:55

原文链接：http://prhub.com.cn/sgl-project/sglang/pull/33954

执行摘要

- 一句话：Qwen masked 注意力元数据改 host 侧构建，消除每步 GPU 同步
- 推荐动作：值得精读。改动仅 57 行，但展示了两个有价值的工程决策：一是复用已有信息 (txt_seq_lens) 消除不可避免的 GPU 同步，二是从被否定的较大方案 (#31852) 中剥离出独立成立的小改进并单独合入。实现层面建议关注 build_varlen_mask_meta_from_ranges 的契约与分支条件完备性，未来可补充基准数据验证收益量级。

功能与动机

PR body 明确指出 :qwen's masked branch rebuilds varlen metadata from the joint mask with a GPU nonzero(), which forces a device sync on every denoising step. 而 txt_seq_lens already carries each row's valid text prefix (the mask is derived from it), 因此元数据可以在 host 侧直接组装。该改动从 #31852 中拆分：整体前向图方案未通过验收，但这一小块无论是否使用 CUDA Graph 都成立且稳赚，值得单独合入。

实现拆解

1. 导入扩展: 在 python/sglang/multimodal_gen/runtime/models/dits/qwen_image.py 的 attention 导入中新增 build_varlen_mask_meta_from_ranges。
2. 核心分支调整: 在 forward 的 masked 分支中 ,is_in_breakable_cuda_graph() 判断之后插入一个新 elif: 当 txt_seq_lens 非空、长度等于 batch_size、且每个值都落在 [0, txt_len] 范围内时，用 [(0, int(n)), (txt_len, txt_len + image_seq_len)] 的区间列表调用 build_varlen_mask_meta_from_ranges, 得到与 GPU 路径等价的 cu_seqlens / indices / inv_indices。这里 int(n) 确保 tensor 元素可在 host 侧使用。
3. 兜底保留: 条件不满足时仍走原 build_varlen_mask_meta(joint_mask);BCG 的 DynamicVarlenMaskMeta() 动态构建路径完全不动。
4. 测试配套: 新增 test/unit/test_varlen_meta_host_build.py, 用包含全 padding 行的 3 行 batch(有效前缀长度 3、7、0) 同时跑两个 builder, 逐元素断言 cu_seqlens、indices、inv_indices、max_seqlen 完全一致。

关键文件：

- python/sglang/multimodal_gen/runtime/models/dits/qwen_image.py（模块 模型实现；类别 source；类型 core-logic）：核心源码变更文件：在 masked 分支新增 host 侧 ranges 构建 varlen 元数据的路径，消除每个 denoising step 的 GPU nonzero() 与设备同步，并

保留 mask 构建器作为兜底。

- python/sglang/multimodal_gen/test/unit/test_varlen_meta_host_build.py（模块 单元测试；类别 test；类型 test-coverage；符号 TestHostVarlenMetaEquivalence, test_prefix_text_plus_full_image_matches_nonzero_builder）：新增单元测试，验证 host 侧 ranges 构建器与 GPU nonzero 构建器的输出在 cu_seqlens、indices、inv_indices、max_seqlen 上逐元素一致，覆盖含全 padding 行的边界情况。

关键符号：build_varlen_mask_meta_from_ranges, build_varlen_mask_meta

关键源码片段

python/sglang/multimodal_gen/runtime/models/dits/qwen_image.py

核心源码变更文件：在 masked 分支新增 host 侧 ranges 构建 varlen 元数据的路径，消除每个 denoising step 的 GPU nonzero() 与设备同步，并保留 mask 构建器作为兜底。

```
# masked 分支：优先用 host 侧已知的 txt_seq_lens 组装 varlen 元数据，
# 避免每个 denoising step 在 GPU 上跑 nonzero() 并触发设备同步。
if encoder_hidden_states_mask is not None:
    encoder_hidden_states_mask = encoder_hidden_states_mask.to(
        device=hidden_states.device, dtype=torch.bool
    )
    batch_size, image_seq_len = hidden_states.shape[:2]
    image_mask = torch.ones(
        (batch_size, image_seq_len),
        dtype=torch.bool,
        device=hidden_states.device,
    )
    joint_mask = torch.cat([encoder_hidden_states_mask, image_mask], dim=1)
    block_attention_kwargs["attn_mask"] = joint_mask

    if is_in_breakable_cuda_graph():
        # BCG 会按 prompt 长度分桶 ,replay 时用当前静态 mask 现场构建元数据，
        # 不能闭包捕获旧的 cu_seqlens / indices。
        block_attention_kwargs["attn_mask_meta"] = DynamicVarlenMaskMeta()
    elif (
        txt_seq_lens is not None
        and len(txt_seq_lens) == batch_size
        and all(0 <= n <= encoder_hidden_states.shape[1] for n in txt_seq_lens)
    ):
        # txt_seq_lens 就是每行有效文本前缀长度 (mask 由它生成 ),
        # 直接在 host 侧组装 ranges: 文本段 [0, n) + 图像段
        # [txt_len, txt_len + image_seq_len), 产物 cu_seqlens / indices /
        # inv_indices 与 GPU 路径逐元素一致。
        txt_len = encoder_hidden_states.shape[1]
        block_attention_kwargs["attn_mask_meta"] = (
            build_varlen_mask_meta_from_ranges(
                [
                    [(0, int(n)), (txt_len, txt_len + image_seq_len)]
```

```
                for n in txt_seq_lens
            ],
            max_seqlen=txt_len + image_seq_len,
            device=hidden_states.device,
        )
    )
else:
    # 兜底 :txt_seq_lens 不可用或形状不符时，退回原 GPU mask 路径。
    block_attention_kwargs["attn_mask_meta"] = build_varlen_mask_meta(joint_mask)
```

python/sglang/multimodal_gen/test/unit/test_varlen_meta_host_build.py

新增单元测试，验证 host 侧 ranges 构建器与 GPU nonzero 构建器的输出在 cu_seqlens、indices、inv_indices、max_seqlen 上逐元素一致，覆盖含全 padding 行的边界情况。

```
"""host 构建的 varlen 元数据必须与 mask 版(nonzero)构建器逐元素一致。"""

import unittest

import torch

from sglang.multimodal_gen.runtime.layers.attention.layer import (
    build_varlen_mask_meta,
    build_varlen_mask_meta_from_ranges,
)


class TestHostVarlenMetaEquivalence(unittest.TestCase):
    def test_prefix_text_plus_full_image_matches_nonzero_builder(self):
        # 构造 3 行 batch: 有效文本前缀分别为 3、7、0( 最后一行是全 padding),
        # 图像段固定为全 True, 覆盖前缀文本 + 完整图像的标准形态。
        txt_len, img_len = 7, 5
        txt_seq_lens = [3, 7, 0]
        bs = len(txt_seq_lens)
        mask = torch.zeros(bs, txt_len + img_len, dtype=torch.bool)
        for row, n in enumerate(txt_seq_lens):
            mask[row, :n] = True
            mask[row, txt_len:] = True

        # GPU 掩码构建器作为参考实现
        ref = build_varlen_mask_meta(mask)
        host = build_varlen_mask_meta_from_ranges(
            [[(0, n), (txt_len, txt_len + img_len)] for n in txt_seq_lens],
            max_seqlen=txt_len + img_len,
            device=mask.device,
        )

        # 元数据契约 :cu_seqlens / indices / inv_indices 必须逐元素一致
        for key in ("cu_seqlens", "indices", "inv_indices"):
            torch.testing.assert_close(host[key], ref[key], rtol=0, atol=0)
```

```
        self.assertEqual(host["max_seqlen"], ref["max_seqlen"])


if __name__ == "__main__":
    unittest.main()
```

评论区精华

本 PR 没有 review 评论，唯一交互是作者在关联 issue 评论中的 /tag-and-rerun-ci(触发 CI 重跑 ,Extra 套件最终转绿)。值得注意的设计讨论记录在 PR body 中：整体前向图方案 (whole-forward graph) 未通过验收，但这一小块不依赖图也收益，故拆出单独合入——这是一种很好的变更拆分与验收策略。

- 从 #31852 拆分的范围控制 (design): 放弃整体前向图方案，保留并单独合并这段稳赚的性能优化。
- CI Extra 套件重跑 (other): 无技术分歧 ;Extra 套件最终转绿后合入。

风险与影响

- 风险:
  1. 正确性风险: 新分支依赖 txt_seq_lens 与 mask 结构的强一致性假设 (有效文本前缀 + 全图区间)。代码校验了长度与范围，但无法发现 mask 含中间 padding 等非前缀形态；若上游改变 mask 语义，该分支会静默产出错误元数据。
  2. 性能风险: 消除每个 denoising step 的 GPU 同步理论上必有收益，但 PR 未附带 benchmark 数据量化提升幅度。
  3. 兼容性风险:BCG 的 DynamicVarlenMaskMeta 路径未触碰 ;txt_seq_lens 不可用或形状不符时自动回退旧路径，无行为变化。
  4. 测试覆盖风险: 新增测试只覆盖标准形态 (前缀文本 + 全图), 未覆盖 txt_seq_lens 为 None 或长度不匹配的 fallback 分支。 - 影响: 影响范围限于 multimodal_gen 中 Qwen-Image 的 masked 分支 (传入 encoder_hidden_states_mask 的场景，典型为 SP 文本掩码), 每个 denoising step 少一次设备同步，预期对 eager 与非 BCG 路径的解码延迟和吞吐有正向影响，但未量化。对使用 BCG 的用户无行为变化。团队侧引入了一条新的 host/ranges 元数据构建路径，后续需要持续保持与 GPU builder 的等价性验证。 - 风险标记: 核心解码路径变更，mask 语义依赖上游约定，缺少性能基准，测试仅覆盖前缀文本形态

关联脉络

- PR #31852 Qwen 前向图整体方案 (未过验收): 本 PR 从中拆分出 host 侧 varlen 元数据构建这一独立收益部分 ,PR body 明确引用该验收结论。
- PR #33923 [Diffusion] Route zimage and hunyuanvideo attention through USPAttention: 同属 diffusion attention 重构线，共享 attention/layer.py 的 varlen 元数据契约。

- PR #32667 [Diffusion] Add K/V-gather sequence parallel attention: 同一 diffusion attention 演进方向，涉及 SP 文本掩码与 varlen 元数据的构建路径。
- PR #33953 [Diffusion] fix: scope the masked-path replicated guard to SP runs: 修复同一 masked 注意力路径的守卫逻辑，与本 PR 处于相邻代码区域。