PR #35796 完整报告

sgl-project/sglang

[diffusion] fall back to a component's default attention backend

合并时间：2026-08-21 22:56

原文链接：http://prhub.com.cn/sgl-project/sglang/pull/35796

执行摘要

- 一句话：全局 attention-backend 对辅助组件回退默认后端
- 推荐动作：值得精读。核心价值在于 get_attn_backend() 对“全局 CLI 选择 vs 组件级 override”的语义区分，以及 is_cross_attention 角色在多层注意力封装间的显式传播——这是多组件扩散管线中后端选择与安全回退的典型设计模式。建议重点阅读 selector.py 的回退分支和 component_loader.py 的上下文传递，以及新增的交叉注意力角色测试，理解 fail-closed 稀疏拒绝的取舍。

功能与动机

Issue #35743 报告：传入全局 --attention-backend sage_attn 时，即使 MiniMax-H3 音频 VAE 的注意力层声明了 supported_attention_backends={FA, TORCH_SDPA} 与 default_attention_backend=TORCH_SDPA，selector 仍因 selection_is_explicit 为真而拒绝回退，抛出 ValueError: Attention backend 'sage_attn' is not supported by this attention layer，最终导致 Rank 0 scheduler is dead。Issue 原文指出：“Not honouring an explicit request silently is reasonable; the problem is that a global flag is treated as an explicit request for every component, including ones that cannot possibly satisfy it and that ship a declared default for that situation.”

实现拆解

1. 选择器契约扩展（python/sglang/multimodal_gen/runtime/layers/attention/selector.py）：ComponentAttnBackendContext 新增 allow_global_backend_fallback 字段，并新增 _component_allows_global_backend_fallback() 帮助函数；get_attn_backend() 引入 selected_from_global_cli 标志，在来自全局 CLI 的后端不受支持时新增 global backend fallback 回退分支；候选序列改为 default_attention_backend -> None -> 组件支持集合；同时在候选评估中对 is_cross_attention=True 且后端为 sparse 的候选直接拒绝（fail-closed）。这是整个回退机制的核心，影响所有 diffusion 组件的注意力后端解析。
2. 组件加载器链路（runtime/loader/component_loaders/component_loader.py）：ComponentLoader 新增类属性 allow_global_attention_backend_fallback = True，而 GenericComponentLoader 显式置为 False，即未知 out-of-tree 组件默认不允许回退；_load_customized_with_context、_load_native_with_context 与 load_component 均增加对应参数并传入 component_attn_backend_context_manager，让组件构造期间的所有 get_attn_backend() 调用都能感知该开关。

3. 管线装配解耦（runtime/pipelines_core/composed_pipeline_base.py）：load_modules 移除手动 component_attn_backend_context_manager 包装，改为把 component_attn_backend 与 component_attn_name 透传给 PipelineComponentLoader.load_component，由加载器统一创建上下文，职责更清晰。
4. 交叉注意力角色传播：layer.py 中 LocalAttention/USPAttention 把 is_cross_attention 传入 get_attn_backend()；mova_dual_tower.py、ltx_2.py、SANA WM、Cosmos3、Helios 等模型的注意力构造处标记条件 / 上下文注意力为交叉注意力；ltx_2_connector.py 删除未使用的 USPAttention 实例。
5. VAE 后端约束与测试 + 文档：MiniMax-H3 的 video VAE 注意力层限制到已验证的 FA/Torch SDPA 后端；新增 test_cross_attention_backend_roles.py，扩展 test_attention_backend_selector.py（覆盖显式全局回退、辅助组件回退、组件级严格、稀疏交叉注意力拒绝四类场景）与 test_qwen3vl_text.py（辅助组件从全局后端回退），并在文档中说明选择契约。

关键文件：

- python/sglang/multimodal_gen/runtime/layers/attention/selector.py（模块 注意力选择器；类别 source；类型 core-logic；符号 get_attn_backend, _component_allows_global_backend_fallback, ComponentAttnBackendContext）：核心变更文件：区分全局 CLI 选择与组件级 override，新增 global backend fallback 分支、默认后端候选与稀疏交叉注意力拒绝逻辑。
- python/sglang/multimodal_gen/runtime/loader/component_loaders/component_loader.py（模块 组件加载器；类别 source；类型 core-logic；符号 ComponentLoader, GenericComponentLoader, load_component, _load_customized_with_context）：确定回退作用域：注册组件默认允许回退，未知 out-of-tree 组件默认严格，并把开关传递给注意力后端上下文的入口。
- python/sglang/multimodal_gen/test/unit/test_cross_attention_backend_roles.py（模块 注意力测试；类别 test；类型 test-coverage；符号 test_local_attention_forwards_cross_attention_role, test_ltx2_derives_cross_attention_role_from_context, test_mova_bridge_marks_conditional_attention_as_cross_attention）：新增测试文件，锁定 LocalAttention/USPAttention 在 LTX-2 与 MOVA 桥接中的交叉注意力角色传递行为，防止后续回归。
- python/sglang/multimodal_gen/test/unit/test_attention_backend_selector.py（模块 选择器测试；类别 test；类型 test-coverage；符号 _FakeSparseBackend, test_explicit_global_backend_uses_component_default, test_explicit_global_backend_falls_back_for_auxiliary_component, test_explicit_component_backend_remains_strict）：扩展选择器测试，覆盖全局显式后端回退组件默认、辅助组件回退、组件级严格、稀疏交叉注意力拒绝四类核心场景。
- python/sglang/multimodal_gen/test/unit/test_qwen3vl_text.py（模块 文本编码测试；类别 test；类型 test-coverage；符号 _ExplicitServerArgs, _FakeFABackend, test_qwen3vl_auxiliary_component_falls_back_from_global_backend）：覆盖真实 MiniMax-H3 Qwen3-VL 文本编码器路径，验证辅助组件在全局后端不支持时回退到 FA，防止 issue 场景回归。

- python/sglang/multimodal_gen/runtime/pipelines_core/composed_pipeline_base.py（模块 管线装配；类别 source；类型 dependency-wiring；符号 load_modules）：把组件注意力上下文管理从管线装配层移到加载器内部，统一传入 component_attn_backend 与名称，布线简化且职责清晰。
- python/sglang/multimodal_gen/runtime/models/adapter/ltx_2_connector.py（模块 模型适配器；类别 source；类型 data-contract）：删除未使用的 USPAttention 实例，避免残留的旧式注意力初始化干扰新的角色传播契约。
- python/sglang/multimodal_gen/runtime/models/vaes/minimax_h3_video_vae/attention.py（模块 VAE 注意力；类别 source；类型 data-contract）：将 MiniMax-H3 Video VAE 注意力层限制到已验证的 FA/Torch SDPA 后端，与 audio VAE 行为对齐，并配合选择器回退逻辑。

关键符号：get_attn_backend, _component_allows_global_backend_fallback, ComponentLoader._load_customized_with_context, ComponentLoader._load_native_with_context, ComponentLoader.load_component, composed_pipeline_base.load_modules, test_local_attention_forwards_cross_attention_role, test_ltx2_derives_cross_attention_role_from_context, test_mova_bridge_marks_conditional_attention_as_cross_attention, test_explicit_global_backend_uses_component_default, test_explicit_global_backend_falls_back_for_auxiliary_component, test_explicit_component_backend_remains_strict, test_sparse_backend_falls_back_for_unconstrained_cross_attention, test_qwen3vl_auxiliary_component_falls_back_from_global_backend

关键源码片段

python/sglang/multimodal_gen/runtime/loader/component_loaders/component_loader.py

确定回退作用域：注册组件默认允许回退，未知 out-of-tree 组件默认严格，并把开关传递给注意力后端上下文的入口。

```
class ComponentLoader(ABC):
    # --attention-backend 主要面向 DiT 主模型；辅助组件在全局选择
    # 不受支持时允许回退到声明的默认后端或平台兼容后端。
    # 子类可以显式收紧该开关。
    allow_global_attention_backend_fallback = True

    def _load_customized_with_context(
        self,
        component_model_path,
        server_args,
        component_name: str,
        attn_backend: Any,
        component_attn_name: str | None,
        allow_global_backend_fallback: bool,
    ) -> AutoModel:
```

```
    with component_attn_backend_context_manager(
        attn_backend,
        component_name=component_attn_name,
        allow_global_backend_fallback=allow_global_backend_fallback,
    ):
        # 组件构造期间所有 get_attn_backend() 调用都会看到该上下文
        load_kwargs = self.customized_load_kwargs_for_component(
            server_args, component_name
        )
        return self.load_customized(
            component_model_path, server_args, component_name, load_kwargs
        )

def _load_native_with_context(
    self,
    transformers_or_diffusers: str,
    attn_backend: Any,
    component_attn_name: str | None,
    allow_global_backend_fallback: bool,
) -> AutoModel:
    with component_attn_backend_context_manager(
        attn_backend,
        component_name=component_attn_name,
        allow_global_backend_fallback=allow_global_backend_fallback,
    ):
        # 原生 transformers/diffusers 路径同样在上下文中完成加载
        return self.load_native(
            component_model_path, server_args, component_name, ...
        )

def load_component(self, ...):
    try:
        with component_attn_backend_context_manager(
            component_attn_backend,
            component_name=component_attn_name,
            allow_global_backend_fallback=(
                loader.allow_global_attention_backend_fallback
            ),
        ):
            return loader.load(
                component_model_path,
                server_args,
                component_name,
                transformers_or_diffusers,
            )
    except Exception as e:
        # 保留原有异常归一化逻辑
        ...
```

```
class GenericComponentLoader(ComponentLoader):
    # 未知的 out-of-tree 组件本身也可能是主 transformer，
    # 必须通过注册的 loader 显式 opt-in 才允许回退；
    # 默认保持与 DiT 相同的严格语义，避免静默掩盖配置错误。
    allow_global_attention_backend_fallback = False
```

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

新增测试文件，锁定 LocalAttention/USPAttention 在 LTX-2 与 MOVA 桥接中的交叉注意力角色传递行为，防止后续回归。

```
from unittest import mock

import pytest
import torch
from torch import nn

from sglang.multimodal_gen.runtime.layers.attention import layer as attention_layer
from sglang.multimodal_gen.runtime.models.bridges import mova_dual_tower
from sglang.multimodal_gen.runtime.models.dits import ltx_2
from sglang.multimodal_gen.runtime.platforms import AttentionBackendEnum


class _FakeAttentionImpl(nn.Module):
    # 仅用于隔离后端实现，避免真实内核参与单测
    def __init__(self, **_kwargs) -> None:
        super().__init__()


class _FakeAttentionBackend:
    @classmethod
    def get_enum(cls) -> AttentionBackendEnum:
        return AttentionBackendEnum.FA

    @classmethod
    def get_impl_cls(cls):
        return _FakeAttentionImpl


def test_local_attention_forwards_cross_attention_role():
    # LocalAttention 必须把 is_cross_attention 传给 get_attn_backend，
    # 否则稀疏回退与角色判定会失真
    with (
        mock.patch.object(attention_layer, "get_compute_dtype", return_value=torch.bfloat16),
        mock.patch.object(attention_layer, "get_attn_backend", return_value=_
        FakeAttentionBackend) as get_backend,
        mock.patch.object(attention_layer, "wrap_attention_impl_forward"),
    ):
        attention_layer.LocalAttention(
```

```
            num_heads=1,
            head_size=64,
            is_cross_attention=True,
        )

    assert get_backend.call_args.kwargs["is_cross_attention"] is True


@pytest.mark.parametrize("use_local_attention", [False, True])
def test_ltx2_derives_cross_attention_role_from_context(use_local_attention):
    # 无论走 LocalAttention 还是 USPAttention，LTX-2 都应从
    # context_dim 是否出现推导出 is_cross_attention 角色
    selected_layer = "LocalAttention" if use_local_attention else "USPAttention"
    with (
        mock.patch.object(ltx_2, "get_tp_world_size", return_value=1),
        mock.patch.object(ltx_2, "ColumnParallelLinear", return_value=nn.Identity()),
        mock.patch.object(ltx_2, "RowParallelLinear", return_value=nn.Identity()),
        mock.patch.object(ltx_2, selected_layer) as attention,
    ):
        # 带 context_dim 的构造：交叉注意力
        ltx_2.LTX2Attention(
            query_dim=8,
            context_dim=8,
            heads=1,
            dim_head=8,
            use_local_attention=use_local_attention,
        )
        cross_attention_kwargs = attention.call_args.kwargs
        attention.reset_mock()
        # 不带 context_dim 的构造：自注意力
        ltx_2.LTX2Attention(
            query_dim=8,
            heads=1,
            dim_head=8,
            use_local_attention=use_local_attention,
        )
        self_attention_kwargs = attention.call_args.kwargs

    assert cross_attention_kwargs["is_cross_attention"] is True
    assert self_attention_kwargs["is_cross_attention"] is False
```

评论区精华

该 PR 没有正式的 review 评论线程，PR 评论与 Issue 评论区有 3 条有效讨论，集中在 CI 状态归因上：

- 作者 lgy1027 报告 NPU job 失败：若干既有 diffusion 用例超过性能基线，且日志显示 /usr/local/cuda/bin/nvcc 不可用导致 CUDA 快路径被禁用；模型仍正常选择 torch_sdpa，没有任何 attention-backend 错误。

- 作者补充确认 NPU runner 还出现容器执行失败（self-hosted runner 基础设施问题），请求图片生成接口返回 200，后端日志显示正常选型，因此判断为基础设施 / 基线问题而非本 PR 回归。
- 维护者mickqian触发/tag-and-rerun-ci，作者请求补上run-ci-extra标签以通过额外门禁。
- NPU job 失败是否属于本 PR 回归 (testing): 作者判断为基础设施与基线问题，非本 PR 引入的回归；单元测试与 GPU multimodal job 均通过，PR 最终合入。
- 额外 CI 门禁与 rerun (other): CI 重跑完成，额外门禁通过后 PR 合入。

风险与影响

- 风险：
  1. 核心选择器逻辑变更：get_attn_backend() 是 diffusion 所有注意力后端的统一入口，回退候选顺序与 selected_from_global_cli 分支的改变可能影响 DiT 主模型的后端解析；虽然用条件分支隔离了全局 CLI 与组件级 override，仍需关注混合加载管线。
  2. 跨 21 文件回归面大：改动波及 MOVA、Cosmos3、Helios、LTX-2、SANA WM 等模型与多个 VAE，任何一处的 is_cross_attention 标记遗漏都可能导致后端选择错误或稀疏回退误判。
  3. fail-closed 拒绝稀疏交叉注意力：即使 layer 未声明 supported 集合，sparse 后端也会被拒绝，若某模型实际依赖稀疏 kernel 做交叉注意力且未声明，会直接失败（这是有意设计，但属于行为变更）。
  4. NPU 性能基线失败未完全归因：作者判断为环境（nvcc 缺失、容器失败）与基线问题，但该环境变量组合下回退逻辑仍可能产生与 CUDA 设备不同的行为，需要后续持续观察。
  5. GenericComponentLoader 行为收紧：未知 out-of-tree 组件默认 allow_global_attention_backend_fallback=False，之前可用的全局后端现在可能直接报错，属于有意的契约收紧，但对外部自定义组件是潜在破坏性变更。 - 影响：用户影响：全局 --attention-backend 不再对辅助组件“一票否决”，MiniMax-H3 等混合组件管线的服务可用性显著提升；未知 out-of-tree 组件保持严格失败语义，契约更清晰。系统影响：注意力后端选择逻辑多了一级上下文开关与回退顺序，组件加载上下文从管线装配层移到加载器内部，后续新增 loader 需要理解并正确传递 allow_global_attention_backend_fallback。团队影响：选择契约被文档化，交叉注意力角色成为后端判定的正式输入，降低后续模型接入成本；21 个文件的改动需要维护者在合并后留意扩散类 issue 的回归报告。 - 风险标记：核心选择器逻辑变更，跨 21 文件回归面大，NPU 性能基线失败未归因，fail-closed 拒绝稀疏交叉注意力，GenericComponentLoader 行为收紧

关联脉络

- PR #35740 [multimodal] MiniMax-H3: fix quantized qkv scales and missing-param policy: 同为 MiniMax-H3 模型路径的修复，且本 PR 再次触碰 MiniMax-H3 的 VAE 注意力层，两者围绕同一模型族收敛后端与参数行为。
- PR #35728 [diffusion] Accelerate SANA-Video linear attention in quality=high: 同属 diffusion 注意力后端演进，SANA-Video 线性注意力快路径与本次注意力后端选择 / 角色传播共享 layer 与 selector 基础设施。

- PR #35724 [diffusion] Enable LongCat breakable CUDA graphs: 同为 diffusion 管线的后端与 server args 调整，涉及同一套 multimodal_gen 运行时的注意力与 CUDA graph 选择链路。
- PR #35774 [diffusion] Resolve LoRA weight sources deterministically: 同为 diffusion 侧组件加载器与管线核心的契约调整，与本 PR 共享 component loader 的加载上下文机制。