PR #37049 完整报告

sgl-project/sglang

[Diffusion] Make component execution options fail closed

合并时间：2026-08-30 21:06

原文链接：http://prhub.com.cn/sgl-project/sglang/pull/37049

执行摘要

- 一句话：组件执行选项 fail-closed 化，显式配置未兑现即报错
- 推荐动作：值得精读。最有价值的设计是 selector.py 中在 context manager 的 finally 之后抛异常的写法——先保证上下文重置与日志汇总，再向调用方报告未兑现的显式选项；以及 " 显式 vs 自动 " 的语义区分（用户传入必须兑现，框架推导可以豁免）。对设计运行时准入校验的团队，这个 fail-closed 模式可以直接借鉴。

功能与动机

PR body 明确给出了共享契约："an explicit component option must either be honored by the actual runtime path or fail before fallback/use"。此前 diffusion 运行时存在三类静默失效：native fallback 无法兑现 TP/SP/Ulysses/Ring/KV-gather/FSDP 等分布式布局、组件级 attention backend 覆盖落在不消费 SGLang attention 层的组件上、显式组件 /layerwise offload 没有请求期 ComponentUse 却从不搬运设备。这些配置此前会被接受但实际不生效，用户难以察觉，因此需要 fail-closed 的准入校验。

实现拆解

变更围绕 " 显式选项要么被兑现、要么提前失败 " 这一契约，按四条边界落地：

1. 组件 attention backend fail-closed（selector.py + minimax_h3.py + server_args.py）：component_attn_backend_context_manager 新增 require_component_backend_selection 参数（默认 True），在上下文退出时检查 selected_backends 是否为空且存在显式 backend；若组件从未构建 SGLang attention 层，则在上下文重置后抛出 ValueError。为避免误伤，新增 record_component_attn_backend 供非层构造路径（如 minimax_h3.py 中 deferred model-specific resolution）登记选择；同时 server_args.py 用 _automatic_component_attention_backend_keys 记录 LTX2 / MiniMax H3 自动推导的 text_encoder: torch_sdpa，通过 is_component_attention_backend_automatic 豁免自动默认。
2. native fallback 分布式布局拒绝（transformer_loader.py + pipeline_configs/base.py）：validate_native_fallback 先调用 super() 走基类的 native_only_components 契约检查，再逐项拒绝 tp_size、sp_degree、ulysses_degree、ring_degree、kv_gather_degree 大于 1 或 FSDP 请求，在真正执行 native load 之前抛 RuntimeError。native_only_components 从 component_loader.should_raise_customized_load_error 里

的 getattr 反射探测改为 PipelineConfigBase 上的类型化字段直接访问。

3. 显式 offload 与请求期声明绑定（component_manager.py）：begin_request 末尾新增 _validate_explicit_nonresident_components，枚举 pipeline.modules 中 explicit_residency_mode 为 COMPONENT_OFFLOAD / LAYERWISE_OFFLOAD 的模块，若其名称未出现在各 stage 声明的 ComponentUse 中，则抛 ComponentResidencyError，避免 " 配置了 offload 但前向前永不被搬运 " 的静默失效。
4. 测试配套：五个单元测试文件覆盖三条真实边界——test_transformer_loader_fallback.py 用 mock 验证分布式执行在 native load 前被拒绝、replicated 配置保留 fallback；test_component_residency.py 验证 offload 必须有 ComponentUse 声明、有声明则放行；test_attention_backend_selector.py 验证显式 backend 未消费 attention 层时报错、加载异常保留且上下文重置、自动 backend 可跳过检查；test_server_args.py 验证 LTX2 自动 backend 不算显式覆盖；test_vae_loader.py 验证 native_only_components 默认空元组。文档两处（cli.mdx、attention_backends.mdx）同步更新错误信息措辞。

关键文件：

- python/sglang/multimodal_gen/runtime/layers/attention/selector.py（模块 注意力选择；类别 source；类型 core-logic；符号 component_attn_backend_context_manager, record_component_attn_backend, _record_component_attn_backend）：fail-closed 核心机制所在：context manager 新增 require_component_backend_selection 校验，新增 record_component_attn_backend 供非层构造路径登记后端选择。
- python/sglang/multimodal_gen/runtime/managers/memory_managers/component_manager.py（模块 驻留管理；类别 source；类型 core-logic；符号 _validate_explicit_nonresident_components, begin_request）：新增 _validate_explicit_nonresident_components，把显式 offload 与请求期 ComponentUse 声明绑定，杜绝 " 配置了 offload 但前向前永不被搬运 " 的静默失效。
- python/sglang/multimodal_gen/runtime/loader/component_loaders/transformer_loader.py（模块 回退加载；类别 source；类型 core-logic；符号 validate_native_fallback）：native fallback 准入核心：先走基类 native_only_components 契约，再拒绝 TP/SP/Ulysses/Ring/KV-gather/FSDP 布局，新增 kv_gather_degree 拒绝项。
- python/sglang/multimodal_gen/runtime/loader/component_loaders/component_loader.py（模块 组件加载；类别 source；类型 core-logic；符号 should_raise_customized_load_error, _load_customized_with_context, _load_native_with_context, load_component）：native_only_components 从 getattr 反射探测改为类型化契约直接访问；三条加载路径统一计算 require_component_backend_selection。
- python/sglang/multimodal_gen/runtime/server_args/server_args.py（模块 服务参数；类别 source；类型 core-logic；符号 is_component_attention_backend_automatic, _adjust_attention_backend）：新增 _automatic_component_attention_backend_keys 与 is_component_attention_backend_automatic，区分框架自动推导与用户显式覆盖的组件 backend。
- python/sglang/multimodal_gen/configs/pipeline_configs/base.py（模块 流水线配置；类别 source；类型 data-contract；符号 native_only_components）：声明

native_only_components 类型化契约字段，替代 loader 中的反射探测，是本 PR " 契约化 " 的载体。

- python/sglang/multimodal_gen/runtime/models/dits/minimax_h3.py（模块 模型实现；类别 source；类型 data-contract；符号 record_component_attn_backend）：在模型构造入口登记 deferred model-specific 的 attention backend 选择，防止 fail-closed 检查误判为未兑现。
- python/sglang/multimodal_gen/test/unit/test_transformer_loader_fallback.py（模块 单元测试；类别 test；类型 test-coverage；符号 _mocked_load, test_distributed_execution_rejects_before_native_load, test_replicated_cfg_and_dp_keep_native_fallback）：用 mock 覆盖真实 load 边界：分布式执行在 native load 前被拒绝、replicated 配置保留 fallback，测试质量最高的配套文件。
- python/sglang/multimodal_gen/test/unit/test_component_residency.py（模块 单元测试；类别 test；类型 test-coverage；符号 _server_args_with_component_offload, test_explicit_component_offload_requires_a_declared_request_use, test_declared_component_use_admits_explicit_component_offload）：覆盖显式 offload 必须有请求期 ComponentUse 声明、有声明则放行的双向测试。
- python/sglang/multimodal_gen/test/unit/test_attention_backend_selector.py（模块 单元测试；类别 test；类型 test-coverage；符号 test_component_override_requires_an_sglang_attention_layer, test_component_override_preserves_load_error_and_resets_context, test_automatic_component_backend_may_skip_sglang_attention_layer）：覆盖 fail-closed 的三个关键语义：显式 backend 未消费 attention 层报错、加载异常保留且上下文重置、自动 backend 可跳过检查。

关键符号：record_component_attn_backend, component_attn_backend_context_manager, _validate_explicit_nonresident_components, is_component_attention_backend_automatic, validate_native_fallback, should_raise_customized_load_error, _load_customized_with_context

关键源码片段

python/sglang/multimodal_gen/runtime/layers/attention/selector.py

fail-closed 核心机制所在：context manager 新增 require_component_backend_selection 校验，新增 record_component_attn_backend 供非层构造路径登记后端选择。

```
@contextmanager
def component_attn_backend_context_manager(
    attn_backend: AttentionBackendEnum | None,
    component_name: str | None = None,
    allow_global_backend_fallback: bool = False,
    require_component_backend_selection: bool = True,
) -> Generator[None, None, None]:
    # 未指定 backend 且未指定组件名时直接透传，不进入 fail-closed 校验。
    if attn_backend is None and component_name is None:
        yield
        return
```

```
    token = component_attn_backend_context.set(
        ComponentAttnBackendContext(
            attn_backend,
            component_name,
            {},
            allow_global_backend_fallback,
        )
    )
    unused_component_name: str | None = None
    unused_backend_name: str | None = None
    completed = False
    try:
        yield
        completed = True
    finally:
        context = component_attn_backend_context.get()
        # 只有「显式指定 backend」且「组件从未记录任何后端选择」时才判定为未兑现。
        unused_component_override = (
            completed
            and require_component_backend_selection
            and (
                context is not None
                and context.backend is not None
                and context.component_name is not None
                and not context.selected_backends
            )
        )
        if unused_component_override:
            unused_component_name = context.component_name
            unused_backend_name = context.backend.name.lower()
        _log_component_attn_backend_summary(context)
        component_attn_backend_context.reset(token)
    # 把异常推迟到上下文重置之后抛出，保证 ContextVar 先被清理、日志先被输出。
    if unused_component_name is not None and unused_backend_name is not None:
        raise ValueError(
            f"Attention backend {unused_backend_name!r} was requested for component "
            f"{unused_component_name!r}, but that component "
            "did not construct an SGLang attention layer."
        )
```

python/sglang/multimodal_gen/runtime/managers/memory_managers/component_manager.py

新增 _validate_explicit_nonresident_components，把显式 offload 与请求期 ComponentUse 声明绑定，杜绝 " 配置了 offload 但前向前永不被搬运 " 的静默失效。

```
  def _validate_explicit_nonresident_components(self) -> None:
      """拒绝没有请求期 use 站点的显式 offload 选择器。

      Component placement 由请求时间线执行，而不仅是选择初始加载设备。
```

```
    如果没有声明 ComponentUse，显式 non-resident 模块会被接受，但
    在前向传播之前永远不会被移动到目标设备，从而静默失效。
    """
    # 单元测试常用 SimpleNamespace 替代 ServerArgs，此时跳过校验。
    if not isinstance(self.server_args, ServerArgs):
        return

    # 收集 pipeline 中所有 stage 声明的组件使用点。
    declared_components = {use.component_name for use in self._ordered_uses}
    unmanaged_components = sorted(
        component_name
        for component_name, module in self.pipeline.modules.items()
        if isinstance(module, nn.Module)
        and self.server_args.explicit_residency_mode(component_name)
        in (COMPONENT_OFFLOAD, LAYERWISE_OFFLOAD)
        and component_name not in declared_components
    )
    if unmanaged_components:
        names = ", ".join(repr(name) for name in unmanaged_components)
        raise ComponentResidencyError(
            "Explicit component residency requires "
            f"{names} to have a request-time ComponentUse declaration; "
            "none appears in this pipeline"
        )
```

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

native fallback 准入核心：先走基类 native_only_components 契约，再拒绝 TP/SP/Ulysses/Ring/KV-gather/FSDP 布局，新增 kv_gather_degree 拒绝项。

```
  def validate_native_fallback(self, server_args: ServerArgs, component_name: str) -> None:
    # 先执行基类契约检查：pipeline_config.native_only_components 中列出的
    # 组件一律走 native 加载，禁止自定义加载器回退。
    super().validate_native_fallback(server_args, component_name)

    # 收集所有请求的分布式执行布局；任一布局无法由 native fallback 兑现时，
    # 在真正执行 native load 之前直接失败，而不是静默忽略用户配置。
    requested_distributed_execution = []
    if server_args.tp_size is not None and server_args.tp_size > 1:
        requested_distributed_execution.append(f"tp_size={server_args.tp_size}")
    if server_args.sp_degree is not None and server_args.sp_degree > 1:
        requested_distributed_execution.append(f"sp_degree={server_args.sp_degree}")
    if server_args.ulysses_degree is not None and server_args.ulysses_degree > 1:
        requested_distributed_execution.append(
            f"ulysses_degree={server_args.ulysses_degree}"
        )
    if server_args.ring_degree is not None and server_args.ring_degree > 1:
        requested_distributed_execution.append(
            f"ring_degree={server_args.ring_degree}"
```

```
        )
    # kv_gather_degree 是本 PR 新增的拒绝项，与其余布局一并检查。
    if (
        server_args.kv_gather_degree is not None
        and server_args.kv_gather_degree > 1
    ):
        requested_distributed_execution.append(
            f"kv_gather_degree={server_args.kv_gather_degree}"
        )
    if server_args.should_use_fsdp_for_component(component_name):
        requested_distributed_execution.append("FSDP")
    if requested_distributed_execution:
        raise RuntimeError(
            f"Component {component_name!r} cannot honor requested distributed execution: "
            f"{', '.join(requested_distributed_execution)}. Use an SGLang-native "
            "transformer implementation or set tp_size, sp_degree, "
            "ulysses_degree, ring_degree, and kv_gather_degree to 1 without "
            "FSDP."
        )
```

评论区精华

该 PR 没有实质 review 评论（review 评论 0 条，唯一 issue 评论是 mintlify 文档预览 bot），核心权衡体现在 commit 迭代与 PR body 中：

- 自动 backend 与显式 backend 的区分：第二个 commit fix: preserve automatic component attention defaults 表明 LTX2 / MiniMax H3 自动设置的 text_encoder: torch_sdpa 曾可能被误判为显式覆盖而触发 fail-closed。结论是显式覆盖必须失败、自动默认必须豁免，凭据是 server_args._automatic_component_attention_backend_keys。
- 无 attention 后端组件的兼容：第三个 commit fix: preserve backend-free component loader paths 表明 VAE 等不消费 SGLang attention 层的组件需要豁免路径。最终通过 require_component_backend_selection 参数显式控制，并在 attn_backend 为 None 时不触发检查。
- 四合一合并策略：body 明确说明本 PR 取代 #37041、#37044、#37045、#37046，作为唯一 NVIDIA CI 候选，减少 CI 噪音并集中审查。
  - 自动推导的组件 backend 不应触发 fail-closed (design): 显式覆盖必须 fail-closed，自动默认必须豁免；区分依据是 backend 是否由框架推导而非用户传入。
  - 无 attention 后端的组件加载路径兼容 (design): 该行为是刻意的 fail-closed；需要豁免的路径显式传入 require_component_backend_selection=False，且 attn_backend 为 None 时检查本身不触发。
  - 四个 PR 合并为单一 CI 候选 (other): 合并完成，本 PR 为最终审查单元，减少了 CI 噪音与重复审查。

风险与影响

- 风险：

- selector.py 行为 breaking change：require_component_backend_selection 默认 True，任何注册了组件 backend 上下文但未构建 SGLang attention 层的既有路径都会从静默变为 ValueError。未来新增不消费 attention 层的组件（如新的 VAE / 声码器）若显式指定 backend，必须显式传 False，否则启动失败。
- server_args.py 新增状态易遗漏：_automatic_component_attention_backend_keys 依赖 _adjust_attention_backend 中所有自动推导点同步登记，后续新增自动 backend 推导逻辑若忘记登记，会把自动默认误判为显式覆盖。
- component_manager.py 校验依赖枚举完整性：_validate_explicit_nonresident_components 只检查 isinstance(module, nn.Module) 的模块，且 isinstance(self.server_args, ServerArgs) 保护了测试用的 SimpleNamespace；若生产路径传入代理对象会跳过校验。既有 " 为预留显存而配置 offload 但模型实际不使用该组件 " 的配置在升级后会直接报错。
- transformer_loader.py 新增拒绝项：kv_gather_degree > 1 现在与 tp/sp/ulysses/ring 一起被拒绝，若用户环境中 KV-gather 并行与 diffusers 回退加载组合使用（此前可能静默异常），升级后将显式失败。
- CI 信号：PR Test (Extra) 与 AMD ROCm 7.2 运行显示失败（:x:），虽未说明原因，但属于需要留意的回归信号。
- 影响：对用户：三处显式配置（分布式布局、组件 attention backend、组件 offload）从 " 接受但静默失效 " 变为 " 带明确错误信息提前失败 "，错误信息直接指出修复方向（如回退到 SGLang-native transformer 或关闭并行选项）。对系统：变更集中在 diffusion 多模态生成运行时的校验路径，不触碰计算主路径，无新增生产模块，启动期 / 请求期开销可忽略。对团队：四个 PR 合并为单一 CI 候选，审查负担降低；但自动 backend 豁免集合成为新的维护状态，后续所有自动 backend 推导点都需同步更新。
- 风险标记：行为契约收紧可能影响既有配置，自动 backend 豁免依赖新增状态集合，跨模块校验逻辑分散在四条边界，CI Extra 与 AMD ROCm 运行失败待确认

关联脉络

- PR #37041 component-execution admission 前置 PR（被本 PR 整合）：PR body 明确说明本 PR 整合 #37041 的改动。
- PR #37044 component-execution admission 前置 PR（被本 PR 整合）：PR body 明确说明本 PR 整合 #37044 的改动。
- PR #37045 component-execution admission 前置 PR（被本 PR 整合）：PR body 明确说明本 PR 整合 #37045 的改动。
- PR #37046 component-execution admission 前置 PR（被本 PR 整合）：PR body 明确说明本 PR 整合 #37046 的改动。
- PR #36907 [Diffusion] Enforce component attention backend application: 同一 attention backend 选择子系统，改动 selector.py、minimax_h3.py、component_loader.py、server_args.py，与本 PR 的 fail-closed 校验同属一条功能演进线。
- PR #36991 [Diffusion] Add exact component precision overrides: 组件加载参数契约演进，涉及 component_loader.py / server_args.py，与本 PR 的 native_only_components 契约化方向一致。

- PR #36875 [diffusion] Preserve exact component identity during loading: 组件加载身份与一致性，涉及 component_loader.py / transformer_loader.py，与本 PR 的加载准入逻辑同属组件加载子系统。
- PR #36916 [Diffusion] Detect quantized transformer replacements: 涉及 transformer_loader.py 的加载探测逻辑，与本 PR 的 native fallback 准入改动同文件同区域。