PR #25596 完整报告

sgl-project/sglang

[diffusion] fix LTX2 resident defaults and stage profiling

合并时间：2026-05-19 10:41

原文链接：http://prhub.com.cn/sgl-project/sglang/pull/25596

执行摘要

- 一句话：修复 LTX2 resident 默认卸载与 stage profiling 名称冲突
- 推荐动作：建议熟读 base.py 中注册名称与 profile 名称的设计，以及 composed_pipeline_base 中的去重逻辑，这是 pipeline stage 命名的重要设计决策。同时建议后续修复 H200 兼容性缺失。

功能与动机

PR body 指出：高内存 resident 模式期望两个 DiT 驻留在 GPU 上，但之前的自动默认仍会对辅助组件应用 layerwise offload，导致模式不纯。profiler 仅使用 Python 类名，导致重复 stage 类如两个 LTX2LoRASwitchStage 指标合并为一个。

实现拆解

1. PipelineStage 基类扩展：在 base.py 中添加 _registered_stage_name 和 _profile_stage_name 字段及 set 方法，修改 _component_stage_name 和 _active_component_stage_name 优先使用注册名称，新增 _active_profile_stage_name 方法，并在 __call__ 中替换 self.__class__.__name__ 为 self._active_profile_stage_name()。
2. 组合管线集成：在 composed_pipeline_base.py 的 add_stage 中调用 stage.set_registered_stage_name(stage_name) 和 stage.set_profile_stage_name(self._profile_stage_name(stage, stage_name))，其中 _profile_stage_name 方法检测类名重复时返回注册名称，否则保持类名。
3. resident 模式辅助组件驻留：在 server_args_auto_tune.py 新增 _maybe_keep_ltx23_resident_aux_components_resident，在 maybe_adjust_auto_component_residency_after_offload 末尾调用，将未显式设置的辅助组件卸载标志强制设为 False，并将 layerwise_offload_components 置 None。
4. 服务参数判断：在 server_args.py 添加 _uses_ltx23_high_memory_resident_two_stage_mode 方法，基于设备内存和模式判断是否处于高内存 resident 状态。
5. 测试覆盖：新增 test_pipeline_stage_profiling.py 验证 profile 名称使用和注册名称不影响 profile；在 test_server_args.py 添加三个测试验证 resident 模式辅助组件驻留、original 模式保留 layerwise、以及显式覆盖不被覆盖。

关键文件：

- python/sglang/multimodal_gen/runtime/pipelines_core/stages/base.py（模块 管线基类；类别 source；类型 core-logic；符号 set_registered_stage_name, set_profile_stage_name, _active_profile_stage_name）：PipelineStage 基类，添加了注册名称和 profile 名称机制，核心变更。
- python/sglang/multimodal_gen/runtime/server_args_auto_tune.py（模块 自动调参；类别 source；类型 core-logic；符号 _maybe_keep_ltx23_resident_aux_components_resident）：自动调参逻辑，添加了 resident 模式辅助组件驻留强制逻辑。
- python/sglang/multimodal_gen/runtime/pipelines_core/composed_pipeline_base.py（模块 管线编排；类别 source；类型 core-logic；符号 _profile_stage_name）：组合管线基类，新增 _profile_stage_name 方法并在 add_stage 中设置注册和 profile 名称。
- python/sglang/multimodal_gen/runtime/server_args.py（模块 服务参数；类别 source；类型 core-logic；符号 _uses_ltx23_high_memory_resident_two_stage_mode）：服务参数类，新增 _uses_ltx23_high_memory_resident_two_stage_mode 方法。
- python/sglang/multimodal_gen/test/unit/test_pipeline_stage_profiling.py（模块 profiling 测试；类别 test；类型 test-coverage；符号 NamedNoOpStage, TestPipelineStageProfiling, test_profiler_uses_profile_stage_name, test_registered_stage_name_does_not_change_profile_name）：新增测试文件，验证 profiler 使用 profile stage name 以及注册名称不影响 profile。
- python/sglang/multimodal_gen/test/unit/test_server_args.py（模块 参数测试；类别 test；类型 test-coverage；符号 test_auto_high_memory_ltx23_resident_keeps_aux_components_resident, test_auto_high_memory_ltx23_original_keeps_default_layerwise_components, test_explicit_layerwise_components_preserved_in_ltx23_resident）：测试文件，新增三个测试验证 resident 模式辅助组件驻留行为。
- python/sglang/multimodal_gen/runtime/pipelines/diffusers_pipeline.py（模块 扩散管线；类别 source；类型 core-logic）：扩散管线文件，微量调整（+2 行），可能与注册名称设置相关。

关键符号：set_registered_stage_name, set_profile_stage_name, _active_profile_stage_name, _maybe_keep_ltx23_resident_aux_components_resident, _profile_stage_name, _uses_ltx23_high_memory_resident_two_stage_mode

关键源码片段

python/sglang/multimodal_gen/runtime/pipelines_core/stages/base.py

PipelineStage 基类，添加了注册名称和 profile 名称机制，核心变更。

```
class PipelineStage(StageDedupMixin, ABC):
    def __init__(self):
        self.server_args = get_global_server_args()
        self._component_residency_manager = None
        # 注册的 stage 名称（由 pipeline 设置，用于组件命名）
        self._registered_stage_name: str | None = None
        # profiling 使用的 stage 名称（由 pipeline 设置，避免重复类名覆盖）
        self._profile_stage_name: str | None = None
```

```
    def set_registered_stage_name(self, stage_name: str) -> None:
        """设置注册的stage名称，用于组件落位和日志。"""
        self._registered_stage_name = stage_name

    def set_profile_stage_name(self, stage_name: str) -> None:
        """设置profiling使用的stage名称，覆盖默认类名。"""
        self._profile_stage_name = stage_name

    def _component_stage_name(self, stage_name: str | None = None) -> str:
        """返回组件命名用的stage名称，优先级：显式传入 > 注册名称 > 类名。"""
        return (
            stage_name
            or getattr(self, "_registered_stage_name", None)
            or self.__class__.__name__
        )

    def _active_component_stage_name(self) -> str:
        """返回当前活跃组件的stage名称，优先使用residency manager中的stage_name。"""
        manager = getattr(self, "_component_residency_manager", None)
        manager_state = getattr(manager, "state", None)
        manager_stage_name = getattr(manager_state, "stage_name", None)
        if manager_stage_name is not None:
            return manager_stage_name
        return self._component_stage_name()

    def _active_profile_stage_name(self) -> str:
        """返回profiling使用的stage名称，若设置了_profile_stage_name则使用，否则回退类名。"""
        return getattr(self, "_profile_stage_name", None) or self.__class__.__name__

    def __call__(self, batch: Req, server_args) -> Req:
        # 使用 profile stage name 记录指标，而非原始类名
        stage_name = self._active_profile_stage_name()
        # ... 后续逻辑
```

python/sglang/multimodal_gen/runtime/server_args_auto_tune.py

自动调参逻辑，添加了 resident 模式辅助组件驻留强制逻辑。

```
  def _maybe_keep_ltx23_resident_aux_components_resident(self) -> None:
    """在LTX2.3高内存两阶段resident模式下，确保未显式设置的辅助组件保持驻留。"""
    args = self.server_args
    # 仅当处于高内存 resident 模式时执行
    if not args._uses_ltx23_high_memory_resident_two_stage_mode():
        return

    changed: list[str] = []
    # 如果 layerwise_offload_components 未显式设置，则置为 None 以取消默认 layerwise
    if (
        args.layerwise_offload_components is not None
        and not args.is_arg_explicitly_set("layerwise_offload_components")
```

```
    ):
        args.layerwise_offload_components = None
        changed.append("layerwise_offload_components=None")

    # 强制将未显式设置的辅助组件卸载标志改为 False（保持驻留）
    for arg_name in (
        "text_encoder_cpu_offload",
        "image_encoder_cpu_offload",
        "vae_cpu_offload",
    ):
        if getattr(args, arg_name) and not args.is_arg_explicitly_set(arg_name):
            setattr(args, arg_name, False)
            changed.append(f"{arg_name}=False")

    if changed:
        logger.info(
            "Keeping LTX2 high-memory two-stage auxiliary components resident: %s",
            ", ".join(changed),
        )
```

python/sglang/multimodal_gen/runtime/pipelines_core/composed_pipeline_base.py

组合管线基类，新增 _profile_stage_name 方法并在 add_stage 中设置注册和 profile 名称。

```
  def _profile_stage_name(self, stage: PipelineStage, stage_name: str) -> str:
      """决定stage的profile名称：如果stage类名已在之前stage中出现过，
      则使用注册名称（stage_name）避免指标覆盖；否则保持类名。"""
      class_name = stage.__class__.__name__
      # 遍历已添加的 stage，检查是否有重复类名
      if any(
          existing.__class__.__name__ == class_name
          for existing in self._stages
      ):
          return stage_name # 注册名称（由 add_stage 传入）
      return class_name

  def add_stage(
      self,
      stage: PipelineStage,
      stage_name: str | None = None
  ) -> "ComposedPipelineBase":
      # ... 其他逻辑（如验证 stage_name 不重复）
      # 为 stage 设置注册名称和 profile 名称
      stage.set_registered_stage_name(stage_name)
      stage.set_profile_stage_name(self._profile_stage_name(stage, stage_name))
      self._stages.append(stage)
      self._stage_name_mapping[stage_name] = stage
      return self
```

评论区精华

review 中 gemini-code-assist 指出 _uses_ltx23_high_memory_resident_two_stage_mode 缺少 H200 设备检查，与 _resolve_default_ltx2_two_stage_device_mode 不一致，可能导致 H200 上未正确保持辅助组件驻留。该问题未在 PR 中解决。

- H200 兼容性检查缺失 (correctness): 未解决，PR 已合并。

风险与影响

- 风险:
  1. H200 兼容性缺失：新增的判断未包含 H200 特定检查，与现有逻辑不一致，可能漏掉对 H200 设备的正确驻留行为。
  2. profiling 兼容性：__call__ 中 stage 名称来源改变，但 profile 名称会回落类名，对已有指标仅可能新增注册名称指标，向后兼容。
  3. 配置隐式覆盖：layerwise_offload_components 被强制置 None 可能覆盖用户显式设置，但通过 is_arg_explicitly_set 保护。 - 影响：对 LTX2.3 两阶段 pipeline 用户，resident 模式现在正确保持辅助组件驻留，提升性能。profiling 指标避免重复类名覆盖，提高可观测性。影响范围限于 diffusion 模块。 - 风险标记：H200 兼容性缺失，核心路径变更（stage profiling），配置隐式覆盖

关联脉络

- 暂无明显关联 PR