PR #44050 完整报告

vllm-project/vllm

[MRV2] Support breakable CUDA graph

合并时间：2026-05-31 00:40

原文链接：http://prhub.com.cn/vllm-project/vllm/pull/44050

执行摘要

- 一句话：MRV2 支持可中断 CUDA 图
- 推荐动作：本 PR 是 MRV2 CUDA graph 功能的重要扩展，值得关注其设计：通过 run_pw_graph 统一两种分段 CUDA graph 实现，降低调用方复杂度。建议 review 者重点关注 cudagraph_mode.has_piecewise_cudagraphs() 的语义是否正确覆盖了启用条件。

功能与动机

实现关联 issue #42304 中提出的可中断 CUDA 图支持，为 MRV2 提供不依赖 torch.compile 的分段 CUDA 图方案。

实现拆解

1. CUDAGraphManager 扩展：在 CUDAGraphManager.__init__ 中新增 use_breakable_cg 字段，通过 is_breakable_cudagraph_enabled() 和 cudagraph_mode.has_piecewise_cudagraphs() 共同决定是否启用，同时预留 breakable_cg_runner 属性。
2. 初始化与运行桥接：新增 init_breakable_cg_runner 方法（延迟初始化 BreakableCUDAGraphWrapper 并绑定图池），以及 run_pw_graph 方法（根据 use_breakable_cg 选择执行可中断图或模型直接调用）。
3. 模型运行器适配：在 model_runner.py 的 execute_model 中，当 batch_desc.cg_mode == CUDAGraphMode.PIECEWISE 时，改用 cudagraph_manager.run_pw_graph 替代直接模型调用。
4. Eagle 投机解码器适配：在 speculator.py 的 run_model 中，当 cudagraph_runtime_mode == CUDAGraphMode.PIECEWISE 时，同样使用 prefill_cudagraph_manager.run_pw_graph；在 capture 中新增 init_breakable_cg_runner 的调用。
5. 捕获流程统一：在 ModelCudaGraphManager.capture 中也调用了 init_breakable_cg_runner，确保捕获时 runner 已初始化。

关键文件：

- vllm/v1/worker/gpu/cudagraph_utils.py（模块 CUDA 图管理；类别 source；类型 core-logic；符号 init_breakable_cg_runner, run_pw_graph）：核心变更文件，新增 breakable CUDA graph 的初始化和运行逻辑。

- vllm/v1/worker/gpu/model_runner.py（模块 模型运行器；类别 source；类型 data-contract）：修改 execute_model 中的前向路径，将 PIECEWISE 模式下的模型调用委托给 run_pw_graph。
- vllm/v1/worker/gpu/spec_decode/eagle/speculator.py（模块 投机解码；类别 source；类型 core-logic）：适配 Eagle 投机解码器的 prefill 路径，支持 breakable CUDA graph。

关键符号：init_breakable_cg_runner, run_pw_graph

关键源码片段

vllm/v1/worker/gpu/cudagraph_utils.py

核心变更文件，新增 breakable CUDA graph 的初始化和运行逻辑。

```
# vllm/v1/worker/gpu/cudagraph_utils.py

from vllm.compilation.breakable_cudagraph import (
    BreakableCUDAGraphWrapper,
    is_breakable_cudagraph_enabled,
)

class CudaGraphManager:
    def __init__(self, ...):
        # ...
        # 判断是否启用 breakable CUDA graph
        self.use_breakable_cg = (
            is_breakable_cudagraph_enabled()
            and self.cudagraph_mode.has_piecewise_cudagraphs()
        )
        self.breakable_cg_runner: BreakableCUDAGraphWrapper | None = None

    def init_breakable_cg_runner(self, model: nn.Module) -> None:
        # 延迟初始化 runner，并绑定图池
        if self.breakable_cg_runner is None:
            self.breakable_cg_runner = BreakableCUDAGraphWrapper(
                model, self.vllm_config
            )
            self.breakable_cg_runner.graph_pool = self.pool

    def run_pw_graph(self, model: nn.Module, model_inputs: dict[str, Any]) -> Any:
        # 根据 use_breakable_cg 选择执行路径
        if not self.use_breakable_cg:
            # 默认走 torch.compile 的分段 CUDA graph
            return model(**model_inputs)
        assert self.breakable_cg_runner is not None
        return self.breakable_cg_runner(**model_inputs)
```

vllm/v1/worker/gpu/model_runner.py

修改execute_model 中的前向路径，将PIECEWISE 模式下的模型调用委托给run_pw_graph。

```
# vllm/v1/worker/gpu/model_runner.py (execute_model 方法内 )

if batch_desc.cg_mode == CUDAGraphMode.FULL:
    # ...
else:
    # ...
    with set_forward_context(...):
        self.kv_connector.pre_forward(scheduler_output)
        if batch_desc.cg_mode == CUDAGraphMode.PIECEWISE:
            # 统一入口，内部决定使用 compiled PW 还是 breakable cudagraph
            assert self.cudagraph_manager is not None
            model_output = self.cudagraph_manager.run_pw_graph(
                self.model, model_inputs
            )
        else:
            # Eager 模式：调用原始模型
            model_output = self.model(**model_inputs)
```

vllm/v1/worker/gpu/spec_decode/eagle/speculator.py

适配 Eagle 投机解码器的 prefill 路径，支持 breakable CUDA graph。

```
# vllm/v1/worker/gpu/spec_decode/eagle/speculator.py

class EagleSpeculator:
    def run_model(self, ..., cudagraph_runtime_mode=CUDAGraphMode.NONE, ...):
        # ...
        model_inputs = dict(
            input_ids=self.input_buffers.input_ids[:num_tokens],
            positions=self.input_buffers.positions[:num_tokens],
            hidden_states=self.hidden_states[:num_tokens],
            inputs_embeds=inputs_embeds,
        )
        if cudagraph_runtime_mode == CUDAGraphMode.PIECEWISE:
            assert self.prefill_cudagraph_manager is not None
            ret_hidden_states = self.prefill_cudagraph_manager.run_pw_graph(
                self.model, model_inputs
            )
        else:
            ret_hidden_states = self.model(**model_inputs)
        # ...

    def capture(self, attn_states):
        # ...
        if self.prefill_cudagraph_manager.use_breakable_cg:
            self.prefill_cudagraph_manager.init_breakable_cg_runner(self.model)
        self.prefill_cudagraph_manager.capture(...)
```

评论区精华

njhill 提出 use_breakable_cg 字段可能冗余，建议将检查逻辑内移到 init_breakable_cg_runner 中。作者 WoosukKwon 回应表示当前模式是为了清晰显式地选择执行器，有意保持原样。该讨论以 reviewer 的 nit 和作者的解释结束，未产生实际修改。

- use_breakable_cg 字段是否冗余 (design): 作者 WoosukKwon 表示偏好当前显式模式，便于清晰看出选择逻辑，不修改。

风险与影响

- 风险：回归风险：变更集中在前向路径的控制流，若 CUDAGraphMode.PIECEWISE 判定条件或 run_pw_graph 实现有误，可能导致模型推理结果错误。兼容性风险：新增的 use_breakable_cg 字段仅当 is_breakable_cudagraph_enabled() 为真时启用，默认行为不变，但需确保该函数在未集成 breakable cudagraph 的环境（如旧版本）中返回 False。
- 影响：影响范围：主要影响 MRV2 模型执行路径和 Eagle 投机解码的 prefill 阶段。对默认用户无感知，仅当显式启用 breakable CUDA graph 时行为变化。性能影响：引入 breakable CUDA graph 可能带来性能提升，但需实际 benchmark 验证。
- 风险标记：核心路径变更，缺少测试覆盖

关联脉络

- PR #42304 [Feature] Breakable CUDA graph: 关联 issue，本 PR 实现了该 issue 提出的 breakable CUDA graph 支持。