# PR #26814 完整报告

sgl-project/sglang

Add rids/bootstrap-room int-hash plumbing for deterministic per-request identification

合并时间：2026-05-31 09:57

原文链接：http://prhub.com.cn/sgl-project/sglang/pull/26814

## 执行摘要

- 一句话：添加请求 ID 哈希与 bootstrap room int 张量管道
- 推荐动作：建议阅读 forward_batch_info.py 中的哈希函数设计和 CUDA Graph 的集成方式。该设计展示了如何在现有框架中安全添加自定义张量并贯穿捕获 / 回放流程，值得作为模板。建议合并后立即补充测试，尤其是 CUDA Graph 的 replay 一致性测试。

## 功能与动机

PR body 说明：'Add the per-request identification plumbing the deterministic token oracle builds on'。需要在 GPU 上确定性地识别每个请求，以便 token oracle 可以在 cuda 图执行中准确验证 token 输出。通过哈希字符串形式的 rid 和 bootstrap room 为 int64 张量，可在采样阶段高效比较。

## 实现拆解

1. 环境开关：在 python/sglang/srt/environ.py 新增 SGLANG_KV_CANARY_ENABLE_TOKEN_ORACLE 环境变量，默认 False，控制是否启用 token oracle 相关逻辑。
2. 数据结构扩展：在 ForwardBatch（forward_batch_info.py）和 SamplingBatchInfo（sampling_batch_info.py）中添加两个可选 int64 张量字段 rids_int 和 bootstrap_room_ids_int。同时新增三个顶层函数：_stable_hash_str_to_i64（基于 Blake2b 8 字节哈希）、_hash_rids_to_tensor（将请求 ID 列表转换为张量）、_bootstrap_rooms_to_tensor（将 bootstrap room 列表转换为张量，None→ -1）。
3. 初始化集成：在 ForwardBatch.init_new 中，当环境变量开启时，遍历 batch.reqs 的 rid 和 bootstrap_room，调用上述函数生成张量，并同时设置到 ForwardBatch 实例和 batch.sampling_info 中。
4. CUDA Graph 管道：在 CudaGraphRunner（cuda_graph_runner.py）的 DecodeInputBuffers 和 EAGLEDraftCudaGraphRunner（eagle_draft_cuda_graph_runner.py）的 EagleDraftInputBuffers 中添加对应字段。在 create 中按环境变量分配初始张量（零或 -1），在 capture_one_batch_size 中切片传递，在 replay/populate_from_forward_batch 中从 ForwardBatch 拷贝数据到缓冲区。
5. Padding 处理：在 ForwardBatch._pad_inputs_to_size 中添加对 rids_int 和 bootstrap_room_ids_int 的 padding 逻辑（后者用 -1 填充），并同步回 sampling_info。

关键文件：

- python/sglang/srt/model_executor/forward_batch_info.py（模块 批处理元数据；类别 source；类型 data-contract；符号 _hash_rids_to_tensor, _bootstrap_rooms_to_tensor, _stable_hash_str_to_i64）：核心变更文件，定义 rids_int / bootstrap_room_ids_int 字段，实现哈希函数，是 per-request 标识管道的基石。
- python/sglang/srt/speculative/eagle_draft_cuda_graph_runner.py（模块 推测解码；类别 source；类型 dependency-wiring）：在 EAGLE 推测解码的 CUDA Graph 管道中添加标识张量支持，包括输入缓冲区定义、初始化和回放拷贝。
- python/sglang/srt/model_executor/cuda_graph_runner.py（模块 CUDA 图；类别 source；类型 data-contract）：在主模型 CUDA Graph 管道的 DecodeInputBuffers 中添加标识张量，实现 create、populate_from_forward_batch、capture_one_batch_size 中的集成。
- python/sglang/srt/sampling/sampling_batch_info.py（模块 采样信息；类别 source；类型 core-logic）：在 SamplingBatchInfo 中添加 rids_int 和 bootstrap_room_ids_int 字段，确保采样阶段也能访问请求标识。
- python/sglang/srt/environ.py（模块 环境配置；类别 source；类型 core-logic）：新增 SGLANG_KV_CANARY_ENABLE_TOKEN_ORACLE 环境变量，作为整个 token oracle 功能的开关。

关键符号：_hash_rids_to_tensor, _bootstrap_rooms_to_tensor, _stable_hash_str_to_i64

## 关键源码片段

### python/sglang/srt/model_executor/forward_batch_info.py

核心变更文件，定义 rids_int / bootstrap_room_ids_int 字段，实现哈希函数，是 per-request 标识管道的基石。

```
import hashlib
from typing import Optional, List
import torch

# 将字符串稳定哈希为 int64（使用 Blake2b 的 8 字节摘要）
def _stable_hash_str_to_i64(rid: str) -> int:
    digest = hashlib.blake2b(rid.encode('utf-8'), digest_size=8).digest()
    return int.from_bytes(digest, 'little', signed=True)

# 将请求 ID 列表转换为 int64 张量
def _hash_rids_to_tensor(*, rids: List[str], device: torch.device) -> torch.Tensor:
    values = [_stable_hash_str_to_i64(rid) for rid in rids]
    return torch.tensor(values, dtype=torch.int64, device=device)

# 将 bootstrap room 列表转换为 int64 张量，缺失时默认 -1
def _bootstrap_rooms_to_tensor(
    *, bootstrap_rooms: List[Optional[int]], device: torch.device
) -> torch.Tensor:
    values = [room if room is not None else -1 for room in bootstrap_rooms]
```

```
    return torch.tensor(values, dtype=torch.int64, device=device)

@dataclass
class ForwardBatch:
    # 新增：每个请求的 int64 哈希标识，用于确定性 token oracle
    rids_int: Optional[torch.Tensor] = None
    bootstrap_room_ids_int: Optional[torch.Tensor] = None

    @classmethod
    def init_new(cls, batch, model_runner):
        # … 其他字段初始化 …
        if envs.SGLANG_KV_CANARY_ENABLE_TOKEN_ORACLE.get():
            # 计算哈希张量并同时设置到 sampling_info 和自身
            hashed = _hash_rids_to_tensor(rids=[req.rid for req in batch.reqs], device=device)
            bootstrap_room_ids = _bootstrap_rooms_to_tensor(
                bootstrap_rooms=[req.bootstrap_room for req in batch.reqs], device=device
            )
            batch.sampling_info.rids_int = hashed
            batch.sampling_info.bootstrap_room_ids_int = bootstrap_room_ids
            ret.rids_int = hashed
            ret.bootstrap_room_ids_int = bootstrap_room_ids
```

## python/sglang/srt/speculative/eagle_draft_cuda_graph_runner.py

在 EAGLE 推测解码的 CUDA Graph 管道中添加标识张量支持，包括输入缓冲区定义、初始化和回放拷贝。

```
from sglang.srt.environ import envs

@dataclass
class EagleDraftInputBuffers(ForwardInputBuffers):
    # … 其他字段 …
    rids_int: Optional[torch.Tensor] # 新增：请求 ID 哈希张量
    bootstrap_room_ids_int: Optional[torch.Tensor] # 新增：bootstrap room 张量

class EAGLEDraftCudaGraphRunner:
    def __init__(self, eagle_worker, …):
        # … 其他初始化 …
        # 根据环境变量创建标识张量，默认 None 以节省显存
        rids_int = (
            torch.zeros((self.max_bs,), dtype=torch.int64)
            if envs.SGLANG_KV_CANARY_ENABLE_TOKEN_ORACLE.get()
            else None
        )
        bootstrap_room_ids_int = (
            torch.full((self.max_bs,), -1, dtype=torch.int64)
            if envs.SGLANG_KV_CANARY_ENABLE_TOKEN_ORACLE.get()
            else None
        )
        self.input_buffers = EagleDraftInputBuffers(
```

```
            # ... 其他参数 ...
            rids_int=rids_int,
            bootstrap_room_ids_int=bootstrap_room_ids_int,
        )

    def replay(self, forward_batch):
        # 回放时将 forward_batch 中的标识拷贝到缓冲区
        buffers = self.input_buffers
        if buffers.rids_int is not None and forward_batch.rids_int is not None:
            buffers.rids_int[:raw_bs].copy_(forward_batch.rids_int)
        if buffers.bootstrap_room_ids_int is not None and forward_batch.bootstrap_room_ids_int
        is not None:
            buffers.bootstrap_room_ids_int[:raw_bs].copy_(forward_batch.bootstrap_room_ids_int)
```

## 评论区精华

本 PR 未产生实质性技术讨论。CI 显示 PR Test (Extra) 失败，但原因未在 thread 中说明。唯一评论来自 gemini-code-assist[bot] 的每日配额提示，与变更无关。

- 暂无高价值评论线程

## 风险与影响

- 风险：
  1. 环境变量开关限制：SGLANG_KV_CANARY_ENABLE_TOKEN_ORACLE 仅在进程启动时读取，运行时更改无效，可能导致调试困惑。
  2. CUDA Graph replay 数据同步风险：若 rids_int 或 bootstrap_room_ids_int 在 replay 时未正确重置，会导致残留标识传递。当前在 replay 前显式清零 / 填充 -1，但未覆盖所有分支（如裁剪场景）。
  3. 缺少测试覆盖：无单元测试或集成测试验证新字段在 ForwardBatch、CUDA Graph 捕获与回放中的正确性，padding 路径也未测试。
  4. 下游依赖未就绪：token oracle sampler 尚未合并，当前管道仅作为基础设施，实际影响待后续 PR 评估。 - 影响：直接影响 KV-canary 子系统的 token oracle 功能开发。对主推理路径无影响，因为环境变量默认 False，相关代码默认不执行。新增字段均为 Optional，下游使用者需注意空值检查。团队协作上，此 PR 为后续 token oracle sampler 提供基础，后续 PR 将直接依赖此管道。 - 风险标记：环境变量开关控制，缺少单元测试，cuda graph 数据同步风险

## 关联脉络

- PR #26815 Add a deterministic token oracle and production write-input assertion: 本 PR 为 token oracle 提供 per-request 标识管道，token oracle PR 将消费此管道实现确定性采样。
- PR #26818 Add token-id verification to the KV-canary: 同样涉及 ForwardBatch 中 kv-canary 相关字段的添加，共享数据结构扩展模式。

- PR #26816 Add the KV-canary perturb framework for fault-injection self-tests: 同为 KV-canary 基础设施，共享同一环境变量系列和模块结构。