PR #24083 完整报告

sgl-project/sglang

Add benchmark/hicache/bench_warm_cache.py for exact warm-cache shared-prefix benchmarking

合并时间：2026-05-01 14:48

原文链接：http://prhub.com.cn/sgl-project/sglang/pull/24083

执行摘要

- 一句话：新增精确控制共享前缀比例的 warm-cache 基准测试
- 推荐动作：值得精读，尤其是设计精确控制变量的基准测试的方法。它展示了如何在不改动现有代码的前提下，为特定研究场景补充专用工具，其与现有基准对齐的指标设计也值得参考。

功能与动机

当前 hicache 共享前缀基准测试路径对于精确缓存研究控制不足：--enable-shared-prefix 绑定到分组数据集结构，仅对 loogle 和 nextqa 启用，不提供直接控制共享前缀长度、后缀长度或共享前缀百分比扫描的旋钮，也不支持所需的精确 token ID 构造。对于缓存聚焦的生产工作负载，需要一个能够回答诸如 '0% 到 99% 共享前缀对吞吐量影响如何 ' 等问题的基准测试。

实现拆解

1. 命令行参数解析：使用 argparse 定义所有 CLI 选项（--total-tokens, --pcts, --num-prompts, --output-len, --max-concurrency 等），初始化 tokenizer 和随机种子。
2. 共享前缀预热：对于每个指定的共享前缀百分比，计算共享前缀长度和唯一后缀长度，生成一个共享前缀 token 序列和多个唯一后缀序列，通过一次性 /generate 请求预热共享前缀。
3. 并发请求执行：使用 asyncio 和 aiohttp 创建客户端会话，以指定并发度发送流式请求，在 async_request_sglang_generate 中解析流式响应，记录 TTFT、TPOT、ITL 等时间戳。
4. 指标计算与输出：收集所有请求的结果后，计算 BenchmarkMetrics（吞吐量、延迟分布等），格式化为与 bench_serving.py 一致的表格，并可选择写入 JSONL。
5. 缓存刷新控制：每个百分比扫描点前调用 flush_cache 函数清空服务器 KV 缓存，确保预热状态可控。

关键文件：

- benchmark/hicache/bench_warm_cache.py（模块 基准测试；类别 source；类型 benchmark；符号 RequestFuncOutput, BenchmarkMetrics, _create_bench_client_session, async_request_sglang_generate）：新增的 warm-cache 基准测试核心文件，包含所有逻辑

关键符号：_create_bench_client_session, async_request_sglang_generate, run_batch, limited_request, flush_cache, gen_token_ids, main

关键源码片段

benchmark/hicache/bench_warm_cache.py

新增的 warm-cache 基准测试核心文件，包含所有逻辑

```
# benchmark/hicache/bench_warm_cache.py — 核心数据结构与会话工厂

from dataclasses import dataclass, field
from typing import List, Optional, Any
import aiohttp

AIOHTTP_TIMEOUT = aiohttp.ClientTimeout(total=20 * 60 * 60)
AIOHTTP_READ_BUFSIZE = 10 * 1024**2

# 单个请求的输出：包含成功状态、延迟、首 token 时间、token 间隔等
@dataclass
class RequestFuncOutput:
    generated_text: str = ""
    success: bool = False
    latency: float = 0.0
    ttft: float = 0.0 # Time to first token
    itl: List[float] = field(default_factory=list) # Inter-token latency 列表
    prompt_len: int = 0
    error: str = ""
    output_len: int = 0
    start_time: float = 0.0

# 一组请求的整体基准测试指标（部分字段，完整字段请见源文件）
@dataclass
class BenchmarkMetrics:
    completed: int
    total_input: int
    total_output: int
    total_output_retokenized: int
    request_throughput: float
    input_throughput: float
    output_throughput: float
    # ... 其他延迟分布字段
    concurrency: float

# 创建 aiohttp 会话的工厂函数，配置大超时和大缓冲区以支持长时间流式请求
def _create_bench_client_session() -> aiohttp.ClientSession:
    return aiohttp.ClientSession(
        timeout=AIOHTTP_TIMEOUT,
        read_bufsize=AIOHTTP_READ_BUFSIZE,
    )
```

评论区精华

Reviewer @ishandhanani 提出两点建议：

1) 在 hicache 文档中添加使用说明； 2) 支持 OAI 兼容的 chat/completions 端点以覆盖更多用户场景。维护者 @HaiShaw 批准了当前 PR，并指示在后续 PR 中实现这两项增强。

- 支持 OAI chat/completions 端点与文档补充 (feature): 同意在后续 PR 中添加，当前 PR 保持专注。

风险与影响

- 风险：该 PR 仅新增基准测试脚本，不修改已有代码，回归风险低。潜在风险包括： 1) 高并发请求可能压垮服务器，但可通过 --max-concurrency 和有限请求数控制； 2) 依赖 SGLang 原生 /generate 端点格式，未来端点变更需适配； 3) 无自动化测试覆盖基准脚本本身，需人工维护。
  - 影响：对用户：提供了一个精确可控的 warm-cache 基准测试工具，帮助进行缓存命中率对性能影响的定量分析。对系统：无运行时影响。对团队：维护成本较低，但需关注后续 OAI 接口支持请求。影响范围限定在 hicache 性能评估领域。
  - 风险标记：缺少测试覆盖 ，依赖特定端点 /generate

关联脉络

- PR #19746 [P/D disagg] - support decode side radix cache: 该 PR 引入了 decode 端 radix cache，本基准测试可用于测量其性能影响。