# PR #22480 完整报告

sgl-project/sglang

[Observability] Add pending token count to prefill log and get_load

合并时间：2026-04-10 17:05

原文链接：http://prhub.com.cn/sgl-project/sglang/pull/22480

## 执行摘要

该 PR 为 SGLang 调度器新增了待处理令牌数（#pending-token）监控指标，用于增强长上下文请求分块预填充进度的可观测性。通过在预填充日志和 /get_load 接口中暴露该指标，运维人员可以更精确地监控预填充负载和实现跨引擎负载均衡。实现上通过新增辅助函数 _get_num_pending_tokens() 计算等待队列和当前分块请求的剩余令牌，并巧妙处理调度与查询的时间差。这是一个中等重要性的功能增强，为后续调度优化提供了数据基础。

## 功能与动机

为什么需要这个变更？根据 PR body 描述，当前引擎日志仅显示 #queue-req（等待请求数）和 #running-req（运行请求数），对于长上下文请求（单个请求可能包含数十万令牌，跨多个分块进行预填充）的负载理解不足。新增的 #pending-token 指标旨在解决两个具体问题：

1. 监控分块预填充进度：在日志中实时显示还有多少令牌等待预填充，便于调试和性能分析。
2. 跨引擎负载均衡：通过 /get_load 接口返回该指标，负载均衡器可以更智能地将长上下文请求分发到负载较轻的引擎。

## 实现拆解

实现涉及三个主要文件，按模块拆解如下：

| 模块 | 文件 | 关键改动 | 说明 |
| --- | --- | --- | --- |
| 调度器可观测性 | scheduler_metrics_mixin.py | 1. 在 PrefillStats 类添加 num_pending_tokens 字段 2. 新增 _get_num_pending_tokens() 辅助函数 3. 在 report_prefill_stats() 日志输出中添加 #pending-token 4. 在 get_load() 中调用辅助函数并填充返回字段 | 核心计算和集成逻辑，_get_num_pending_tokens() 是关键函数，通过 chunk_deduct 参数处理时间差。 |
| 调度器核心 | scheduler.py | 在 get_new_batch_prefill() 中调用 _get_num_pending_tokens()，并将结果快照到 PrefillStats | 确保在调度时刻捕获准确的待处理令牌数。 |

| 模块 | 文件 | 关键改动 | 说明 |
| --- | --- | --- | --- |
| IO 结构 | io_struct.py | 在 GetLoadReqOutput 类添加 num_pending_tokens 字段 | 扩展 /get_load 接口的返回数据结构。 |

关键函数 _get_num_pending_tokens() 的逻辑如下： def_get_num_pending_tokens(self: Scheduler, chunk_deduct: int = 0) -> int: num_pending_tokens = sum(req.seqlen for req in self.waiting_queue) if self.chunked_req is not None: req = self.chunked_req num_pending_tokens += req.seqlen - len(req.prefix_indices) - chunk_deduct return num_pending_tokens - 参数 chunk_deduct：用于处理调度时间（prefix_indices 尚未更新）和查询时间（prefix_indices 已更新）的时间差。调度时传入 extend_input_len，查询时使用默认值 0。 - 计算逻辑：等待队列令牌总数 + 当前分块请求剩余令牌数（减去已处理的 prefix_indices 和 chunk_deduct）。

## 评论区精华

由于该 PR 没有公开的 review 评论，讨论亮点主要从提交历史中推断：

- 作者在第二个提交（b02649f）中清理了 scheduler.py 中关于 prefix_indices 和 chunk_deduct 的过时注释，表明在实现过程中对时间差处理逻辑进行了微调和澄清。
- 从 PR body 的详细描述和示例日志可以看出，作者对调度器内部状态同步有深刻理解，并设计了优雅的参数传递机制来保证指标准确性。

## 风险与影响

技术风险：

1. 计算准确性风险：_get_num_pending_tokens() 依赖 chunk_deduct 参数的正确传递。若调用方传递错误的值（如调度时未传入 extend_input_len），将导致指标不准确。
2. 性能风险：函数需遍历等待队列求和，在队列极长时（如数千请求）可能有轻微 CPU 开销，但根据 PR body 说明为“轻量级计算”，且仅发生在日志输出和负载查询时，影响可控。
3. 兼容性风险：修改了 PrefillStats.from_adder() 方法签名（新增可选参数 num_pending_tokens），但所有调用方已适配，不会破坏现有功能。

影响评估：

- 用户影响：为运维人员提供更细粒度的预填充进度监控，尤其有助于长上下文请求的调试和负载均衡决策。
- 系统影响：纯观测性变更，不改变模型输出、调度逻辑或 API 行为（除扩展返回字段外）。
- 团队影响：为后续负载均衡算法优化和调度策略调整提供了关键数据指标。

## 关联脉络

从近期历史 PR 分析，该 PR 与以下 PR 存在关联：

1. PR #20801（“[Observability] Add Prometheus metrics endpoint for gRPC mode”）：同为可观测性增强，扩展了系统的监控能力，形成观测性功能线。
2. PR #22495（“Add page_size to admission token budget check”）和 PR #22470（“Fix SWA eviction boundary and page-align chunked prefill”）：涉及调度策略和分块预填充处理，与本 PR 监控的预填充进度和负载管理密切相关。

演进趋势：SGLang 仓库近期在 可观测性和 调度优化两个方向持续投入。本 PR 是观测性方向的又一重要补充，特别是针对长上下文场景的细粒度监控，反映了系统对生产环境可观测性需求的积极响应。结合历史 PR 中的分块预填充修复和负载检查优化，可以看出团队正在构建更健壮、更透明的调度系统。