

PR #50129 完整报告

vllm-project/vllm

[Rust Frontend] Extract shared tracing setup logic into `vllm-tracing`

合并时间: 2026-07-29 03:31

原文链接: <http://prhub.com.cn/vllm-project/vllm/pull/50129>

执行摘要

本 PR 将 `vllm-cmd` 和 `vllm-bench` 中重复的 tracing subscriber 初始化逻辑提取到新的 `vllm-tracing` workspace crate, 统一了 Rust 二进制文件的日志格式, 清理了 #48937 留下的技术债。变更纯属代码重组, 无功能变化, 经快速审核后合并。

功能与动机

PR #48937 为基准测试端点引入了 tracing 日志, 但两个二进制文件各自维护了独立的 subscriber 初始化代码。这导致重复代码和维护负担。本 PR 旨在消除重复, 确保所有 Rust 二进制文件输出一致的 vLLM 风格日志格式, 并方便未来新增二进制文件时复用。

实现拆解

1. 新建 `vllm-tracing` crate: 在 `rust/src/tracing/` 下创建 workspace crate, 将原来位于 `rust/src/cmd/src/logging.rs` 中的 `init_tracing`、`build_targets_filter` 和 `VllmEventFormatter` 搬迁至此, 并将入口函数可见性提升为 `pub`。
2. 更新 `rust/src/bench/src/main.rs`: 删除内联的 `init_tracing` 函数 (约 10 行) 和 TODO 注释, 改为一行调用 `vllm_tracing::init_tracing("Bench")`。
3. 更新 `rust/src/cmd/src/main.rs`: 删除 `mod logging` 声明, 改为调用 `vllm_tracing::init_tracing(process_label)`, 并相应地调整 `Cargo.toml` 依赖 (移除 `time`、`tracing-subscriber`, 添加 `vllm-tracing`)。
4. 调整 workspace 配置: 根 `Cargo.toml` 注册新成员, 其他子 crate 的 `Cargo.toml` 更新依赖关系。
5. `Cargo.lock` 自动更新。

`rust/src/tracing/src/lib.rs`

核心文件, 新 crate 的入口, 包含提取后的 `init_tracing` 函数。

```
// SPDX-License-Identifier: Apache-2.0
```

```
// SPDX-FileCopyrightText: Copyright contributors to the vLLM project
```

```
//! 共享的 vLLM 风格 tracing subscriber 和日志格式化, 用于所有 Rust 二进制文件。
```

```
use std::{env, fmt, process};
```

```
use time::UtcOffset;
```

```
use time::macros::format_description;
```

```

use tracing::level_filters::LevelFilter;
use tracing::{Event, Level, Subscriber};
use tracing_subscriber::Layer as _;
use tracing_subscriber::filter::Targets;
use tracing_subscriber::fmt::format::{FormatEvent, FormatFields, Writer};
use tracing_subscriber::fmt::time::FormatTime;
use tracing_subscriber::fmt::{FmtContext, FormattedFields};
use tracing_subscriber::layer::SubscriberExt as _;
use tracing_subscriber::registry::LookupSpan;
use tracing_subscriber::util::SubscriberInitExt as _;

const CYAN: &str = "\x1b[0;36m";
const GREY: &str = "\x1b[90m";
const GREEN: &str = "\x1b[32m";
const YELLOW: &str = "\x1b[33m";
const RED: &str = "\x1b[31m";
const WHITE: &str = "\x1b[37m";
const RESET: &str = "\x1b[0m";
const VLLM_TIME_FORMAT: &[time::format_description::FormatItem<'static>] =
    format_description!("[month]-[day] [hour]:[minute]:[second]");

/// 安装 process-wide 的 vLLM 风格 tracing subscriber。
/// `process_label` 会出现在日志前缀中，例如 `[Bench]`、`[RustFrontend]`。
pub fn init_tracing(process_label: &str) {
    let filter = build_targets_filter(
        env::var("VLLM_LOGGING_LEVEL").ok().as_deref(),
        env::var("RUST_LOG").ok().as_deref(),
    );
    let formatter = VllmEventFormatter::new(process_label);

    let _ = tracing_subscriber::registry()
        .with(
            tracing_subscriber::fmt::layer()
                .event_format(formatter)
                .with_writer(std::io::stderr)
                .with_filter(filter),
        )
        .try_init();
}

```

// build_targets_filter 和 VllmEventFormatter 定义保持原样，省略。

rust/src/bench/src/main.rs

使用新 crate 替换内联 tracing 初始化，是主要消费方之一。

```

// SPDX-License-Identifier: Apache-2.0
// SPDX-FileCopyrightText: Copyright contributors to the vLLM project

#[cfg(not(target_env = "msvc"))]

```

```

#[global_allocator]
static GLOBAL: mimalloc::MiMalloc = mimalloc::MiMalloc;

use anyhow::Context;
use clap::Parser;

#[derive(Parser)]
#[command(name = "vllm-bench", about = "Benchmark online serving throughput", version)]
struct Cli {
    #[command(flatten)]
    args: vllm_bench::BenchServeArgs,
}

fn main() -> anyhow::Result<> {
    // 使用共享的 vllm_tracing crate 初始化日志，进程标签为 "Bench"
    vllm_tracing::init_tracing("Bench");

    let cli = Cli::parse();
    vllm_bench::prepare_process();

    let runtime = tokio::runtime::Builder::new_multi_thread()
        .enable_all()
        .build()
        .context("Failed to build tokio runtime")?;

    runtime.block_on(vllm_bench::run(cli.args))
}

```

评论区精华

无实质讨论。Reviewer [esmeetu](#) 直接批准，表明变更简洁无争议。

风险与影响

- 风险：极低。纯代码提取，无行为变化。唯一需要注意 `init_tracing` 从 `pub(crate)` 变为 `pub`，但 `crate` 仅内部使用，不构成安全风险。
- 影响：所有 Rust 二进制文件现在输出统一格式的日志，便于调试和监控。对用户无直接影响，对开发者降低了维护成本。

关联脉络

直接关联 Issue/PR #48937，后者首次引入 Rust tracing 日志，本 PR 是其技术债的后续清理。此重构也为未来更多 Rust 二进制文件（如潜在的 [vllm-chat](#) 等）提供了共享日志基础设施。