Trabajo Práctico 3: Servidor HTTP Concurrente con Thread Pool en Rust

Programación Concurrente - Primer cuatrimestre 2025

Profesores: Emilio López Gabeiras y Rodrigo Pazos

25 de marzo de 2025

1. Introducción

Este trabajo práctico actualiza el TP2 para implementar un servidor HTTP concurrente en Rust, utilizando *thread pools* para evitar problemas por *context switching*. El objetivo es analizar el comportamiento del servidor bajo carga concurrente, comparando los resultados con la implementación del TP2.

2. Objetivos

- Actualizar el servidor HTTP para procesar *requests* de forma concurrente usando una *thread pool*.
- Demostrar el manejo eficiente de múltiples *requests* concurrentes.
- Utilizar exclusivamente librerías estándar de Rust (`std`).

3. Pruebas

Para evaluar el servidor concurrente, utilizar `ab` (Apache Benchmark) para generar carga.

```
$ ab -n 500 -c 50 http://localhost:3030/pi/1000000
```

Probar con diferentes valores de i y carga concurrente (`-c` en `ab`) para verificar robustez y rendimiento.

4. Preguntas Abiertas

Una vez operativo, explorar:

1. Bajo carga concurrente intensa (aumentando `-n` y `-c` en `ab`), ¿qué efectos se observan en el comportamiento del servidor? ¿Cómo se comparan con los resultados obtenidos en el TP2?

2. ¿Cómo se ve afectado el comportamiento ante carga concurrente intensa para diferentes tamaños de *thread pool*?