Informe 2da Revisión del Proyecto
Sistemas Distribuidos

Tema: Scraper Distribuido
Autor: Lidier Robaina Caraballo
Grupo: C-411
Enero de 2025

1 Arquitectura

El sistema está compuesto por dos redes, una de clientes y otra de servidores distribuidos, interconectadas a través de un router, simulando el funcionamiento real de la comunicación a través de la red. La red de servidores consta de un conjunto de nodos dispuestos en un anillo Chord.

Roles del sistema

- Cliente: Solicita el scraping de sitios web al enviar sus URL al servidor, y almacena localmente los archivos de los sitios ya scrapeados para mitigar el efecto de peticiones repetidas.
- Router: Enruta las peticiones del cliente hacia el servidor y las datos del scraping y las consultas del servidor hacia el cliente.
- Servidor: Es un servidor distribuido con tres roles fundamentales:
  - Enrutador: Recibe y transmite las mensajes internos entre los nodos del anillo. Todos los nodo tienen constancia del estado de los demás, por lo que el nodo que recibe la petición del cliente designa al responsable del scraping de forma tal que se distribuya equitativamente la carga del sistema.
  - Scraper: Encargado de realizar el proceso de scraping.
  - Almacenamiento: Funciona como memoria caché del sistema, permite la persistencia de los datos de los sitios previamente scrapeados incrementando significativamente la rapidez de respuesta del sistema a largo plazo.

  Todos los nodos son idénticos en cuanto a responsabilidades. Se tomó esta decisión de diseño para evitar distribuir en exceso, permitiendo una mayor simplicidad y manejabilidad del sistema.

2 Procesos

- Multiprocesos: Cada nodo ejecuta cada rol en un proceso independiente. Se adopta este enfoque en lugar de multihilos para evitar la limitación del Global Interpreter Lock (GIL) de Python y lograr un paralelismo real. También mejora la resiliencia, ya que un fallo en un proceso (rol) no afecta a otros procesos o al resto de la aplicación.
- Programación asíncrona: Dentro de cada proceso, se utiliza async/await para gestionar la concurrencia de forma eficiente. Esto permite que cada proceso maneje simultáneamente múltiples operaciones, especialmente aquellas de entrada/salida (I/O) que normalmente bloquearían el hilo de ejecución principal. Esta asincronía maximiza el uso de recursos sin recurrir al uso intensivo de hilos.

3 Comunicación

El sistema contará con dos protocolos de comunicación:

- HTTP REST: Se implementa mediante un servidor FastAPI. Este protocolo se utiliza para la comunicación cliente-servidor y para tareas generales de comunicación entre nodos, como consultas y la replicación de datos. HTTP REST ofrece interoperabilidad y facilidad de uso para estas operaciones.
- RPC: Se implementa mediante un servidor gRPC. Este protocolo se destina a la comunicación interna entre procesos para tareas críticas como la gestión, estabilización y el monitoreo del anillo Chord. gRPC se selecciona debido a su baja latencia y alta eficiencia, que son esenciales para estas operaciones.

Esta estrategia híbrida combina la sencillez de HTTP REST con el alto rendimiento de gRPC, optimizando así el sistema para diferentes necesidades de comunicación. Para simplificar la configuración de red, se empleará multiplexación en capa de aplicación. Cada nodo expondrá un único puerto y, según el tipo de solicitud, esta se redirigirá al servidor HTTP REST o al servidor gRPC correspondiente.

4 Nombrado y Localización

Usa un sistema DNS basado en Chord, lo que combina un sistema de nombres de dominio (el dominio de las URLs a scrapear) con el nombrado DHT (basado en una función Hash):

- Cada conjunto de archivos correspondientes a una URL scrapeada está asociado de forma única con una clave, mediante una función Hash de la URL.
- A cada nodo se le asigna un identificador del mismo conjunto de todos los valores hash posibles y se hace responsable de almacenar datos asociados a un subconjunto específico de claves (las que estén en el rango desde la clave propia hasta la clave del sucesor).
- Los nodos pueden resolver nombres de dominio a direcciones IP utilizando el mecanismo de búsqueda eficiente del protocolo Chord.

5 Consistencia y Replicación

- Cada nodo replicará los datos de los que es responsable en su antecesor y en su sucesor en el anillo.
- No se necesitan protocolos de consistencia, no hay escritura sobre los datos una vez almacenados.

6 Respuesta a errores

- Redundancia: Cada dato posee tres copias en el sistema, garantizando una tolerancia a fallas de grado 2.
- Reorganización Dinámica: Cuando un nodo se une o sale de la red, Chord reconfigura automáticamente las tablas y redistribuye las responsabilidades, minimizando el impacto de fallas.

7 Seguridad

FastAPI es un framework que facilita la implementación de mecanismos de seguridad. El sistema empleará autenticación y autorización basadas en el estándar OAuth2.