Implementación de un Sistema de Archivos Distribuido Basado en Etiquetas

Miguel Alejandro Yáñez Martínez Adrián Navarro Foya

Introducción

Un sistema de archivos basado en etiquetas es un tipo de almacenamiento de archivos que permite clasificar archivos y directorios de forma más flexible y dinámica que un sistema tradicional basado en un árbol de directorios. Cada archivo o directorio puede tener una o varias etiquetas asociadas, que permita clasificar y encontrar los archivos de forma más eficiente. Las etiquetas pueden ser asignadas o modificadas de forma dinámica por el usuario, sin necesidad de mover o copiar los archivos, lo que permite una mayor flexibilidad y organización. En ellos es mucho mas facil la busqueda y recuperacion de los archivos ya que solo es necesario saber algunas de las etiquetas asociadas a ese archivo en lugar de la ubicacion exacta de este asi como tambien permite una organización más personalizada y adaptable. El presente informe tiene como objetivo mostrar la arquitectura utilizada en nuestro sistema de archivos distribuido basado en etiquetas.

Arquitectura del sistema

Para la arquitectura distribuida del sistema, se utilizará el protocolo **Chord**. Este es un protocolo de comunicación y un algoritmo de enrutamiento descentralizado para la implementación de tablas de hash distribuidas. Es un sistema sencillo, escalable y está diseñado para funcionar en redes descentralizadas peer-to-peer (P2P). Su objetivo es permitir que cualquier nodo dentro de la red pueda buscar y acceder a un objeto almacenado en cualquier otro nodo, incluso si no se tiene información sobre la ubicación física del objeto o el nodo que lo almacena.

Es un protocolo de tipo anillo, en el que cada nodo almacena información sobre los nodos que están a su derecha en el anillo. Cada nodo recibe una identificación única, que se utiliza para asignar el nodo a una posición en el anillo. Las identificaciones se asignan utilizando una función hash, lo que permite que los nodos se distribuyan uniformemente en el anillo.

El algoritmo de enrutamiento utiliza una tabla de enrutamiento (Finger Table) en cada nodo para determinar el siguiente salto en el anillo que se acerca más al nodo que almacena el objeto.

El protocolo es escalable y tolerante a fallos, debido a que los nodos pueden unirse y abandonar la red sin afectar significativamente el rendimiento o la disponibilidad del sistema. El tiempo de búsqueda de datos es $O(\log N)$, donde N es el número de nodos en la red. La asignación de datos a nodos se realiza de manera uniforme utilizando funciones hash.

Conceptos Fundamentales de Chord

El protocolo Chord es una estructura clave en el sistema distribuido propuesto. Su principal objetivo es organizar los nodos en un anillo lógico que permita localizar eficientemente cualquier archivo o etiqueta almacenada en el sistema. A continuación, se describe cómo se implementa Chord en este proyecto, detallando sus principales componentes y operaciones.

Chord organiza los nodos en un espacio de identificadores generado por una función hash, como SHA-1. Cada nodo y archivo tienen un identificador único dentro de este espacio, lo que define su posición en el anillo. Los datos se asignan al nodo cuyo identificador es el menor mayor o igual (sucesor) al identificador del archivo.

Los principales conceptos de Chord son:

- **Sucesor:** Es el nodo inmediatamente mayor al nodo actual en el anillo.
- **Predecesor:** Es el nodo inmediatamente menor al nodo actual en el anillo.
- **Finger Table:** Es una tabla de referencias mantenida por cada nodo, que apunta a otros nodos en el anillo a distancias exponenciales. Esta tabla permite realizar búsquedas en $O(\log N)$, donde N es el número de nodos.

Principales Operaciones de Chord

1. Estabilización

La estabilización es un proceso periódico que asegura que las referencias de sucesores sean consistentes a medida que nodos se unen o abandonan el anillo. Este proceso incluye:

- Actualización de los sucesores.
- Corrección de referencias en las Finger Tables.
- Redistribución de datos entre los nodos afectados por cambios en la topología.

2. Búsqueda de Sucesores

Cuando un nodo recibe una solicitud para almacenar un archivo o etiqueta, utiliza su Finger Table para encontrar el nodo responsable del identificador asociado. Esto se realiza iterativamente o mediante un proceso recursivo hasta encontrar el nodo correcto.

3. Entrada de Nuevos Nodos

Cuando un nuevo nodo se une a la red:

1. Utiliza el primer nodo existente como punto de entrada para unirse al anillo.
2. Determina su posición en el anillo consultando los sucesores.
3. Solicita al nodo responsable de sus datos transferir los archivos y etiquetas que ahora están bajo su responsabilidad.
4. Actualiza las Finger Tables de los nodos cercanos para reflejar su entrada.

4. Salida de Nodos

Si un nodo abandona la red, transfiere sus datos al nodo sucesor antes de desconectarse. En caso de un fallo inesperado, los nodos vecinos redistribuyen los datos utilizando las réplicas mantenidas.

Finger Tables y Eficiencia

Cada nodo mantiene una Finger Table que contiene referencias a otros nodos en el anillo. Las posiciones en la Finger Table se calculan como:

$$\text{Finger}[i] = \text{nodo}\big((\text{ID} + 2^{i-1}) \mod 2^m\big)$$

donde i es la posición en la tabla, ID es el identificador del nodo actual, y m es el tamaño del espacio de identificadores (generalmente 160 bits para SHA-1).

Estas tablas permiten que una búsqueda en el anillo, que de otro modo requeriría $O(N)$ pasos en una búsqueda lineal, se reduzca a $O(\log N)$ saltos.

Ventajas de Utilizar Chord en el Sistema

La implementación de Chord ofrece varias ventajas para el sistema distribuido:

- **Eficiencia Escalable:** La búsqueda y localización de archivos es altamente eficiente incluso en redes con miles de nodos.
- **Resiliencia a Fallos:** La replicación y el manejo dinámico de fallos aseguran la continuidad del servicio.
- **Distribución Equitativa:** La asignación de identificadores basada en hash garantiza que la carga de almacenamiento y procesamiento esté distribuida uniformemente entre los nodos.

Implementación de Chord en el Sistema de Archivos

El sistema de archivos distribuido utilizará Chord para localizar y gestionar de forma eficiente los datos. Los archivos y etiquetas serán asigandos al nodo correspondiente a partir un esquema de particionamiento consistente utilizando la función hash (SHA-1). A continuación, se detalla cómo se implementará Chord en el sistema:

1. Identificación de Nodos y Datos

Cada nodo en el sistema tendrá un identificador único calculado mediante la función hash aplicada a su dirección IP. De manera similar, cada archivo será asignado al nodo cuyo hash sea sucesor del identificador hash del archivo obtenido a partir de su nombre.

2. Búsqueda de Datos

La búsqueda se realiza por etiquetas. Se comprueba en cada nodo cuales archivos tiene como etiquetas un superconjunto de las pasadas como consulta.Para navegar por el anillo, al llegar a un nodo, se le envía la solicitud de busqueda al sucesor del nodo actual.

3. Adaptación Dinámica

Chord ajusta automáticamente la topología del anillo lógico cuando los nodos se unen o salen del sistema. Esto es clave para garantizar la robustez y disponibilidad, ya que los datos serán redistribuidos entre los nodos restantes para mantener la consistencia.

4. Replicación y Tolerancia a Fallos

Además de la localización de datos, Chord será utilizado para implementar un esquema de replicación. Cada archivo será replicado en nodos especializados en archivos asignados a cada nodo del chord. Esto permitirá recuperar datos en caso de fallos de nodos y mejorar la disponibilidad general del sistema.

5. Prevención de Corrupción de Datos

Para garantizar la integridad de los datos, cada archivo almacenado incluirá un hash de verificación(o numero obtenido a partir de CRC por ejemplo) que se validará antes de cualquier operación. Si un nodo detecta datos corruptos, podrá utilizar las copias redundantes almacenadas en los otros nodos para restaurar el contenido original. Este mecanismo de recuperación, combinado con verificaciones periódicas de integridad, garantizará la confiabilidad del sistema.

Arquitectura del Sistema

El sistema de archivos distribuido basado en etiquetas está diseñado para funcionar como una red descentralizada y tolerante a fallos, compuesta por múltiples nodos que interactúan entre sí. La arquitectura del sistema sigue principios de modularidad y escalabilidad, asegurando que los componentes puedan operar de manera independiente y crecer según las necesidades del sistema.

Componentes Principales

La arquitectura del sistema se organiza en varios componentes clave, cada uno con responsabilidades bien definidas:

- **Nodos de Almacenamiento:** Estos nodos son responsables de almacenar archivos y sus etiquetas asociadas. Sus principales tareas incluyen mantener la integridad de los datos almacenados mediante sumas de verificación, gestionar las réplicas de los datos y servir solicitudes de recuperación de datos realizadas por otros nodos o usuarios.

- **Nodo de Consulta:** Estos nodos procesan las solicitudes relacionadas con búsquedas basadas en etiquetas. Reciben consultas de usuarios y devuelven los resultados basados en las etiquetas especificadas.Actua como intermediarios entre los nodos responsables de una operación específica. Esto reduce la latencia y optimiza la eficiencia de la red.

- **Nodos Chord:** Utilizan el protocolo Chord para enrutar eficientemente las solicitudes entre los nodos. Este protocolo organiza los nodos en un anillo lógico, donde

cada nodo tiene referencias a su sucesor y predecesor, lo que permite localizar cualquier nodo en $O(\log N)$ saltos.

- **Cliente:** Aunque no forman parte del sistema distribuido propiamente dicho, los clientes interactúan con los nodos del sistema. Envían solicitudes para agregar, consultar o eliminar archivos y etiquetas.

Organización

El protocolo Chord organiza los nodos en un anillo lógico donde cada nodo tiene un identificador único derivado de una función hash (como SHA-1). Este identificador determina la posición del nodo en el anillo y la responsabilidad de los datos que almacena.

Los principales mecanismos de Chord son:

- **Búsqueda de Sucesores:** Cada nodo mantiene referencias a su sucesor y predecesor en el anillo, lo que le permite localizar otros nodos eficientemente.
- **Estabilización:** Los nodos ejecutan periódicamente un proceso de estabilización para corregir referencias rotas o inconsistentes debido a la entrada o salida de nodos.
- **Tablas de Referencia (Finger Tables):** Cada nodo mantiene una tabla de referencias a otros nodos en el anillo, lo que reduce el número de saltos necesarios para encontrar un nodo específico.

Tolerancia a Fallos y Escalabilidad

El sistema está diseñado para ser altamente tolerante a fallos. Esto se logra mediante:

- **Replicación de Datos:** Cada archivo y sus etiquetas se replican en varios nodos, garantizando la disponibilidad de los datos incluso si algunos nodos fallan.
- **Reconfiguración Dinámica:** Cuando un nodo se une o abandona la red, los datos y las referencias se redistribuyen automáticamente entre los nodos restantes.
- **Eficiencia Escalable:** A medida que el número de nodos aumenta, la eficiencia de Chord se mantiene, ya que las operaciones de búsqueda requieren $O(\log N)$ saltos, donde N es el número de nodos en la red.

Interacciones Entre Componentes

Cuando un usuario realiza una operación (como buscar un archivo por etiqueta), el sistema sigue los siguientes pasos:

1. El cliente envía una solicitud al nodo de consulta.
2. El nodo consulta se conecta al primer nodo del anillo chord y este a su vez envia la solicitud a su sucesor.
3. Cada nodo del anillo Chord envía la solicitud a su sucesor y luego busca los datos que corresponan a la consulta, si los encuentra los envía al nodo consulta.
4. Si se necesita replicación o redistribución, los nodos coordinan las operaciones adicionales de forma transparente.

Esta arquitectura modular y distribuida asegura que el sistema pueda operar eficientemente, manejar fallos y escalar según las demandas de almacenamiento y consulta.

Implementación del Descubrimiento de Nodos en el Sistema Distribuido

En el proyecto, uno de los desafíos fundamentales es permitir que los nodos descubran dinámicamente a otros nodos de la red sin depender de direcciones IP fijas. Esto es crucial para garantizar la flexibilidad, la escalabilidad y la robustez de la red en entornos dinámicos donde los nodos pueden unirse o abandonar la red en cualquier momento.

Enfoque Basado en Mensajes de Difusión (Broadcast)

Para resolver el problema del descubrimiento dinámico de nodos, se empleará un mecanismo basado en mensajes de difusión (broadcast). Este enfoque permite a un nodo recién iniciado anunciar su presencia a toda la red y recibir información sobre los nodos existentes. El proceso funciona de la siguiente manera:

1. **Anuncio de Presencia:** Cuando un nodo se inicializa, envía un mensaje de difusión en la red local a través de un puerto específico predefinido (por ejemplo, el puerto 8255). Este mensaje incluye la dirección IP y el puerto del nodo emisor, junto con una solicitud para unirse a la red.

2. **Respuesta de los Nodos Existentes:** Los nodos que ya están en la red escuchan constantemente en el puerto de difusión. Al recibir el mensaje de anuncio, responden al nodo solicitante con su propia dirección IP y puerto. Esto le permite al nodo nuevo construir una lista inicial de nodos vecinos.

3. **Construcción de la Red:** Una vez que el nodo nuevo recibe respuestas de los nodos existentes, utiliza esta información para iniciar el proceso de conexión al sistema Chord. El nodo elige un nodo vecino como punto de entrada para unirse al anillo de Chord y comienza el proceso de estabilización para integrarse completamente a la red.

Ventajas del Enfoque de Difusión

El uso de mensajes de difusión ofrece varias ventajas:

- **No Dependencia de Direcciones IP Fijas:** Los nodos pueden descubrirse dinámicamente sin necesidad de conocer previamente sus direcciones IP.

- **Simplicidad:** El enfoque de difusión es simple de implementar y funciona eficientemente en redes locales.

- **Soporte para Redes Dinámicas:** Permite que los nodos se unan o abandonen la red de manera dinámica sin necesidad de reconfiguración manual.

Consideraciones Técnicas

- **Compatibilidad con Subredes:** El uso de difusión está limitado a la subred local. En redes más grandes o distribuidas geográficamente, se podría considerar el uso de mecanismos de descubrimiento basados en directorios distribuidos o servicios como mDNS (Multicast DNS).
- **Seguridad:** Para prevenir la incorporación de nodos no autorizados, los mensajes de difusión deben incluir un token de autenticación. Este token será validado por los nodos existentes antes de enviar sus respuestas.
- **Manejo de Colisiones:** Si múltiples nodos intentan unirse simultáneamente, se implementará un sistema de espera con tiempos aleatorios (backoff exponencial) para evitar colisiones de mensajes.

Integración con Chord

Una vez descubierto un nodo vecino, el nodo nuevo lo utilizará como punto de entrada para el protocolo Chord. Esto se realiza enviando una solicitud de incorporación al anillo de Chord, tras lo cual el nodo vecino lo guía para encontrar su posición en la estructura distribuida.

El nodo nuevo comienza inmediatamente el proceso de estabilización en Chord:

- Identifica su sucesor y predecesor en el anillo.
- Actualiza las réplicas de los datos necesarios desde sus nodos vecinos.
- Notifica a los nodos afectados de su incorporación para ajustar sus referencias.

Limitaciones

Con este enfoque, el sistema de archivos distribuido será capaz de operar de manera flexible y eficiente en redes locales dinámicas, reduciendo la dependencia de configuraciones estáticas y aumentando la resiliencia ante cambios en la infraestructura. En sistemas distribuidos más amplios se deben considerar otras alternativas.

Integración con el Sistema de Archivos

En el sistema distribuido basado en etiquetas, Chord actúa como el núcleo que organiza los nodos y enruta las solicitudes. Cada operación sobre archivos y etiquetas (como agregar, buscar o eliminar) se traduce en una búsqueda de nodos responsables mediante el protocolo Chord. Esto permite al sistema ofrecer un rendimiento eficiente y escalable incluso con grandes volúmenes de datos y numerosos nodos.

La implementación de Chord, combinada con las capas adicionales de redundancia, descubrimiento de nodos y gestión de datos, asegura que el sistema sea robusto, flexible y adecuado para aplicaciones en entornos distribuidos dinámicos.

Gestión de Datos y Etiquetas

El sistema de archivos distribuido basado en etiquetas está diseñado para gestionar datos de manera flexible y eficiente. A diferencia de los sistemas de archivos tradicionales, que organizan los datos en jerarquías de directorios, este sistema utiliza etiquetas para clasificar, buscar y manipular archivos. Esta sección detalla cómo se implementan estas operaciones clave en un entorno distribuido.

Estructura de los Datos

Cada archivo en el sistema está asociado a un conjunto de metadatos que incluye:

- **Nombre del archivo:** Identificador único del archivo dentro del sistema.
- **Etiquetas:** Lista de etiquetas asociadas que describen el contenido o contexto del archivo.
- **Contenido:** Los datos binarios del archivo almacenados en el sistema.

Los datos se distribuyen entre los nodos utilizando una función hash (como SHA-1) aplicada al nombre del archivo. Esto asegura una distribución uniforme de la carga en el sistema.

Operaciones Principales

El sistema proporciona un conjunto de operaciones principales para la gestión de datos y etiquetas:

1. Agregar Archivos y Etiquetas

Cuando un archivo es agregado al sistema, se le asignan una o más etiquetas. El proceso incluye:

1. Generar un identificador único para el archivo mediante una función hash aplicada a su nombre.
2. Determinar el nodo responsable del archivo utilizando el protocolo Chord.
3. Almacenar el archivo en el nodo redundantes del nodo chort responsable, junto con las etiquetas asociadas.

2. Consultar Archivos por Etiquetas

El sistema permite buscar archivos que cumplan criterios basados en etiquetas a partir de Sets,facilitando la busqueda ya que solo hay que comprobar que la consulta sea un subconjunto de las etiquetas de un archivo.

3. Modificar o Eliminar Etiquetas

Se pueden agregar nuevas etiquetas a un archivo existente o eliminar etiquetas asociadas. Estas operaciones aseguran que:

- Las listas de archivos asociadas a las etiquetas sean actualizadas en todos los nodos relevantes.
- Los archivos no queden sin etiquetas, ya que cada archivo debe estar asociado al menos a una etiqueta.

4. Eliminar Archivos

Cuando un archivo es eliminado, el sistema realiza las siguientes acciones:

1. Elimina el archivo de su nodo responsable.
2. Actualiza las listas de etiquetas para eliminar referencias al archivo.
3. Replica los cambios en los nodos secundarios si la replicación está habilitada.

Consistencia y Redundancia

Para garantizar la consistencia y la disponibilidad de los datos, el sistema implementa las siguientes estrategias:

- **Replicación de Datos:** Cada archivo y sus etiquetas son replicados en varios nodos.Esto asegura que los datos estén disponibles incluso si algunos nodos fallan.
- **Sincronización de Etiquetas:** Cuando se actualizan las etiquetas de un archivo, los cambios se propagan automáticamente a los nodos responsables de esas etiquetas.
- **Manejo de Conflictos:** Si se detectan inconsistencias durante una operación (por ejemplo, debido a fallos en la red), el sistema utiliza un protocolo de consenso para resolver los conflictos y garantizar la integridad de los datos.

Con estas funcionalidades, el sistema de archivos distribuido no solo ofrece una manera eficiente de almacenar y recuperar datos, sino que también asegura que los datos sean accesibles, consistentes y protegidos contra fallos o corrupción.

Prevención de Corrupción y Control de Redundancia

En un sistema de archivos distribuido, la prevención de la corrupción de datos y el control de la redundancia son aspectos críticos para garantizar la confiabilidad y disponibilidad de los datos almacenados. Esta sección detalla los mecanismos implementados para mitigar la corrupción, controlar la redundancia y garantizar la integridad de los datos en todo momento.

Prevención de la Corrupción de Datos

La corrupción de datos puede ocurrir debido a errores en la transmisión, fallos de hardware o problemas en la replicación. Para prevenirla, el sistema utiliza las siguientes técnicas:

1. Sumas de Verificación

Cada archivo almacenado en el sistema incluye una suma de verificación generada mediante un algoritmo criptográfico, como CRC. Estas sumas de verificación permiten:

- Detectar cambios accidentales en los datos durante la transmisión o almacenamiento.
- Comparar versiones de un archivo entre nodos para identificar discrepancias.

Antes de responder a cualquier solicitud de recuperación de datos, el nodo verifica la suma de verificación para asegurarse de que el archivo no haya sido modificado.

2. Validación de Réplicas

Los datos replicados en varios nodos se validan periódicamente comparando sus sumas de verificación. Si se detecta una discrepancia, el sistema utiliza un proceso de consenso para identificar la réplica válida y corregir las versiones corruptas.

Control de Redundancia

La redundancia es fundamental en sistemas distribuidos para garantizar la disponibilidad de los datos. Sin embargo, un control adecuado es necesario para evitar redundancias excesivas e inconsistencias. El sistema implementa las siguientes estrategias:

Replicación Configurable

Cada archivo y sus etiquetas se replican en K nodos, donde K es configurable según los requisitos del sistema. Esto asegura:

- Disponibilidad de los datos incluso si varios nodos fallan.
- Tolerancia a fallos en redes distribuidas geográficamente.

Gestión de Fallos y Recuperación

El sistema está diseñado para manejar fallos y garantizar la recuperación de datos de manera eficiente:

- **Monitoreo de Nodos:** Los nodos supervisan periódicamente el estado de sus sucesores. Si un nodo falla, sus datos son recuperados automáticamente de las réplicas almacenadas en otros nodos.
- **Reconstrucción de Réplicas:** Tras un fallo de nodo, las réplicas de los datos asignados a ese nodo se reconstruyen en nuevos nodos responsables.
- **Copia Automática en Casos de Corrupción:** Si una réplica se identifica como corrupta durante una validación, el sistema solicita una copia válida de otro nodo y reemplaza la versión dañada.

Tareas en Segundo Plano

El sistema realiza varias tareas en segundo plano para mantener la integridad y consistencia de los datos:

- **Validación Periódica:** Revisión de la integridad de archivos y etiquetas almacenados.
- **Manejo de Réplicas Antiguas:** Identificación y eliminación de réplicas obsoletas o inconsistentes.

Ventajas de la Estrategia Propuesta

La combinación de prevención de corrupción y control de redundancia ofrece múltiples beneficios:

- **Integridad Garantizada:** Los mecanismos de validación aseguran que los datos sean consistentes y válidos en todo momento.
- **Alta Disponibilidad:** La replicación configurable garantiza que los datos estén disponibles incluso en escenarios de múltiples fallos.
- **Recuperación Transparente:** Los procesos automáticos de reconstrucción y sincronización permiten al sistema recuperarse de fallos sin intervención manual.
- **Escalabilidad:** Las estrategias de replicación y validación están diseñadas para escalar eficientemente en redes grandes.

Con estas estrategias, el sistema es robusto ante fallos, protege los datos contra corrupción y garantiza la consistencia y disponibilidad en todo momento.