Proyecto de Sistemas Distribuidos: BitTorrent

Jesús Santos Capote, Kenny Villalobos Morales, Diamis Alfonso Pérez

Facultad de Matemática y Computación, Universidad de La Habana, La Habana, Cuba

Keywords: BitTorrent · Pyro4 · Chord · Python · Docker

1. Funcionamiento General del Proyecto

Se propone una implementación del protocolo BitTorrent sobre una red peer to peer, donde cada peer es un Cliente BitTorrent o un Tracker, utiizando la biblioteca de python Pyro4. Con el fin de evitar que los Trackers constituyan puntos de falla en la red, se hizo uso del protocolo Chord entre los Trackers de la red, donde cada Tracker es un nodo Chord y se conectan entre ellos formando un anillo.

1.1. Tracker

Servidor encargado de guardar información sobre que peer contiene que archivo, o parte del archivo. Los clientes BitTorrent hacen peticiones a estos servidores para saber que peer contiene, tentativamente, partes del archivo. El Tracker responderá a estas peticiones con una lista de tuplas (IP, Puerto) de peers a los cuales el cliente se puede conectar para lograr su objetivo, entre ellos, el peer que tiene el archivo completo. La lógica de esta entidad es implementada en la clase Tracker del archivo tracker.py.

1.2. Cliente BitTorrent

Un Cliente BitTorrent es un peer que sirve y descarga archivos. Una vez que un cliente tiene una parte de un archivo inmediatamente este la comparte en la red para que otros clientes puedan descargar de él dicha pieza. Cuando un cliente comienza una descarga, crea una instancia de la clase PieceManager, clase que se encarga de manejar las piezas que el cliente descarga y sirve, de un archivo determinado.

1.3. Archivo Torrent

Un torrent es un archivo de texto con extensión .torrent que contiene información sobre el archivo que se quiere descargar. Dicha información esta dispuesta en una serie de campos:

1. announce: IP y Puerto del Tracker que coordina la transferencia de archivos.

2. announce-list: IPs y Puertos de Trackers que tambien pueden coordinar la transferencia.
3. info: un diccionario que describe los archivos que se van a descargar, incluyendo su nombre, tamaño, número de piezas, etc.
4. piece length: el tamaño de cada pieza en bytes.
5. pieces: una cadena que contiene el hash SHA-1 de cada pieza de los archivos que se van a descargar.
6. name: el nombre de la carpeta o archivo que se va a descargar.
7. length: el tamaño del archivo que se va a descargar.
8. private: un valor opcional que indica si la descarga es privada o no.

1.4. Transferencia de Archivos

Este proceso empieza con un cliente BitTorrent que quiere compartir un archivo, para esto debe crear un archivo .torrent con el formato requerido. Luego debe notificar a un Tracker, o a varios de ellos, que desea compartir el archivo, para esto manda al Tracker el campo Pieces del archivo .torrent que creó. El Tracker guarda en su Base de Datos el par "Pieces": [(cliente-ip, cliente-puerto)], como el cliente es el primero que subio el archivo, al principio, estará solo en esa lista.

Para que otros clientes puedan descargar el archivo compartido por el primer cliente, primero deben tener el .torrent que este creó. Luego hacerle una solicitud, enviando el campo Pieces, a algún Tracker del announce-list del .torrent. El Tracker buscará en su Base de Datos el campo Pieces que le llega en la solicitud y responderá con la lista de peers que están descargando el archivo. Luego el cliente interesado solicita el el Bitfield de los peers proporcionados por el Tracker, con esta información calcula cual es la pieza del archivo más rara y establece conexción con su dueño para comenzar a descargarla. Este proceso se repite hasta que el cliente interesado obtiene el archivo entero. El Bitfield no es más que un array booleano que indica que piezas tiene un cliente y cuál no.

La rareza de una pieza está dada por el número de peers en la red BitTorrent que la tengan. Mientras menos peers tengan una pieza más rara es. Por esta razón la prioridad de descarga de las piezas es por rareza, para contribuir a que la pieza más rara se disemine primero por la red.

Cuando un peer comienza a descargar una pieza informa al Tracker, mandando el campo Pieces del .torrent y el Tracker lo añade a la lista de Peers que le corresponde a ese campo Pieces en la Base Datos. De esta forma cuando otro cliente quiera descargar el mismo archivo y haga la solicitud al Tracker, la lista peers devueltos estará actualizada con peers que poseen, potencialmente, partes del archivo.

La transferencia de una pieza se realiza mediante bloques. Una pieza está compuesta por varios bloques y un archivo está compuesto por varias piezas.

2. Chord entre los Trackers

Con la implementación del protocolo Chord sobre los Trackers se evita que la información sobre que peer tiene que archivo esté centralizada en un solo servidor Tracker, pues estará diseminada por el anillo Chord.

Cada Tracker posee un ID, que es el entero resultante de aplicar el hash sha256 a la concatenación de su IP y Puerto. Además guardan el IP y el Puerto de su sucesor y predecesor en el anillo. La información que distribuyen es la almacenada en la Base de Datos de cada Tracker.

A un Tracker le pertenece todos los pares "Pieces":Lista-de-Peers cuyo hash sha256 de "Pieces", casteado a entero, sea menor o igual que el ID de dicho Tracker. Al Tracker con ID más bajo le corresponde los pares con hash menor que su ID y los pares con hash mayor que el ID del Tracker con ID más grande.

Se implementó una función find-successor, la cual a partir de un ID, ya sea de un Tracker o de un par, encuentra el sucesor de ese ID en el anillo Chord, es decir, el Tracker con ID mas cercano por abajo al ID pasado como argumento. Cuando un nuevo Tracker se va a unir al anillo Chord, llama a la función find-successor del Tracker que le sirve como entrada, pasándole su ID como parámetro. Así encuentra su sucesor en el anillo y se coloca en el lugar que le corresponde. Luego le pide a su sucesor los pares que le pertenece y finalmente el anillo queda en un estado consistente.

Para los clientes BitTorrent todo este proceso es transparente. Cuando un cliente hace una solicitud de peers o notifica que está descargando una pieza, envía el campo Pieces de su .torrent al Tracker que le sirve como entrada al anillo. Este lo hashea y llama a find-successor con ese valor para encontrar el Tracker que posee la información. Finalmente el sucesor encontrado responde al cliente o actualiza su Base de Datos.

Cuando un Tracker deja el anillo le cede todos sus pares a su sucesor.