LSH - Locality Sensitive Hashing - 2C 2020 - Clase 2 Consulta

La nueva empresa estatal, *Música Justa*, quiere poder sugerirle a sus usuarios playlists similares a las que crearon. Una playlist es una lista de canciones, por ejemplo [“Oncle Jazz”, “Tailwhip”, “Lauren”]. Como hay millones de listas, y el servidor es un Pentium III, no es factible comparar contra todas.

Diseñar **integralmente** un esquema de LSH, detallando **qué métrica** se usa y por qué, **cómo se generan los minhashes** a partir de los datos, cómo se almacena la **estructura** y cómo se resuelve una **consulta**.

Si conocemos los títulos de las canciones, y estas están en una lista, es recomendable utilizar la distancia Jaccard y obtener singles de los títulos de las canciones, con una longitud predefinida como lo puede ser por ejemplo 6, o un shingle de longitud variable que nos permitiría utilizar cada canción como un shingle distinto.

Entonces, se recorre toda la lista en búsqueda de los singles; y para cada shingle se aplicará una función de minhash del estilo:

(Sh[0]\*C0 + … + Sh[5]\*C5 + C6)%P%M

Con P un número primo mayor que M, y M cantidad de buckets.

Y nos quedaremos con el mínimo de esos minhashes para terminar de definir el minhash de esos shingles para la playlist.

El esquema no define una probabilidad p1 o p2 para los cuales se tenga que definir un r y b determinados, pero se sabe que utilizar r=1, b=1 suele ser una mala resolución, por lo que por ejemplo se puede usar r=2, b=2.

En este contexto requeriré 4 minhashes (2 por cada familia), y pueden definirse modificando las constantes C0,…,C6.

Entonces, cada minhash nos dará un número para cada playlist; y debido a que hay varios minhash por familia los relacionamos de la siguiente manera:

Familia(Mh1, Mh2) = (K0.Mh1 + K1.Mh2 + K3)%P%M

Donde K0,…,K3 son constants definidas para la familia, P es un número primo mayor que M, y M es la cantidad de buckets a utilizar.

El resultado obtenido de la familia de hashing nos indica en qué posición de la tabla de hash debemos almacenar el elemento que está siendo hasheado.

La idea de LSH es que elementos similares colisionen en una misma posición de la tabla de hash, ya que es una solución al problema de los vecinos más cercanos.

Por cada grupo b el elemento se almacenará en una posición de la tabla; es decir, como máximo lo encontraremos en b posiciones de la tabla, aunque puede suceder que algunas de las familias de los b grupos indiquen que el elemento debe almacenarse en la misma posición (Por ejemplo que la familia 1 de 1 y la familia 2 también; entonces el elemento se almacenará una única vez en la posición 1 de la tabla de hash).

Finalmente, para resolver una consulta se realiza el mismo proceso de hashing para el elemento query. Una vez hasheado, este tendrá como resultados de las familias ciertas posiciones que le corresponden en base a sus elementos. Aquella posición que nos otorguen las familias serán las posiciones donde se encuentran los elementos semejantes a la playlist hasheada. Entonces, se realiza el OR (merge) de los conjuntos obtenidos como resultado de la búsqueda por LSH, y esos resultados serán los que se consideren semejantes. Para una mayor precisión se puede recorrer el listado de elementos obtenidos por LSH, y ver si cumplen con una semejanza determinada por el usuario.