Recuperatorio Hashing y LSH

Lo contratan de Copate Cupido, una startup argentina con el objetivo de reactivar el romance post pandemia. En la app le aparecen personas que están cerca suyo para charlar. El radio de búsqueda es de 2km. Dado que se espera tener más de cien millones de usuarios registrados y con varios millones usando la aplicación en cualquier momento dado, es necesario que esta búsqueda de personas cercanas sea eficiente. Habiendo cursado la materia, usted propone usar LSH.

a) Elija una métrica a utilizar justificando adecuadamente. Desarrolle cómo es una función de minhash para esa métrica indicando los valores de los parámetros.(25pts)

b) Realice un diagrama del proceso de LSH (utilizando r=3, b=2) detallando todos los pasos con una única tabla final. (30pts)

c) Definiendo los valores necesarios de los parámetros de las funciones a utilizar, haga una consulta del punto [-34.6,-58.4], detallando los cálculos e indicando qué espera hallar en la estructura final. Utilice r=3 b=2.(30pts)

d) Enumere todas las causas que pueden generar falsos positivos en el diagrama del b). (15pts)

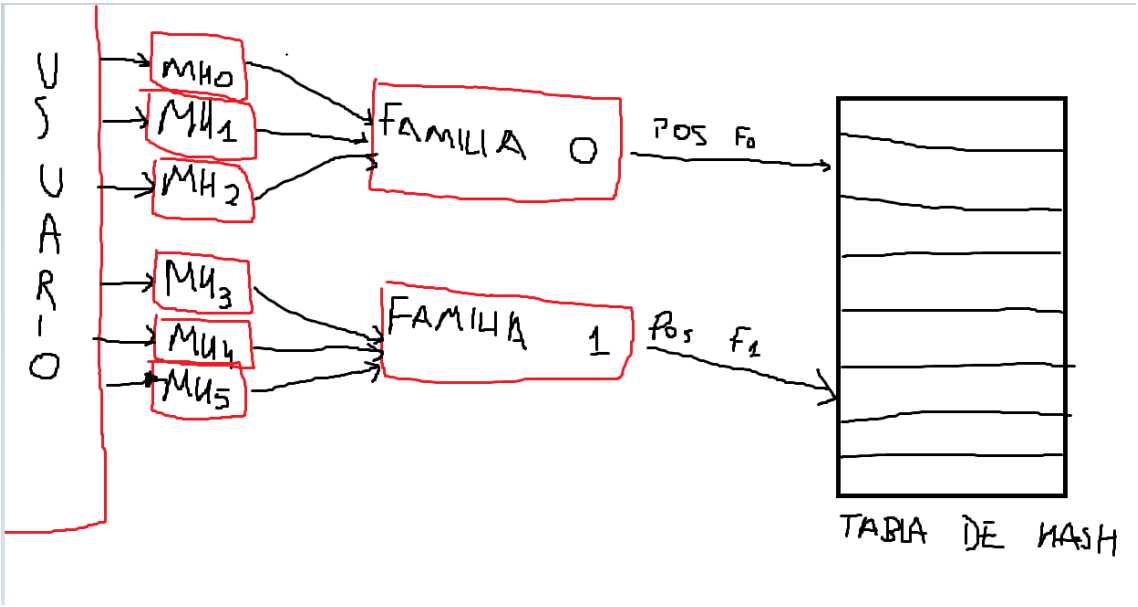
RESOLUCIÓN

a) Una buena métrica a utilizar sería la de distancia euclídea. Esto se debería a que dado que las posiciones de las personas pueden verse cartografiadas como un vector de R2 (Posición (X,Y) relativa respecto del centro del planisferio, con componentes pertenecientes a los reales dentro del rango del ancho y alto en km del mundo plasmado en un planisferio), entonces aquellos que tengan posiciones similares serán vectores similares.
   Utilizar el minhash correspondiente a la distancia euclídea significaría multiplicar estos vectores de posición por un hiperplano aleatorio, sumarle una constante a, y dividir por un w constante, que definiría la cantidad de buckets y el ancho de los mismos.
   Para aplicar distintos minhashes correspondientes a euclídea bastaría con utilizar distintos hiperplanos aleatorios en R2, y una constante a que puede variar, pero estando siempre en el rango entre 0 y w-1.
   Entonces, el minhash correspondiente a la distancia euclídea queda definido como:

$$\frac{Vector\ Pos\ .\ Hiperplano + a}{w} = MH(Vector\ Pos)$$

   Donde como se dijo anteriormente, tanto el vector de posición del usuario, y el hiperplano serán vectores pertenecientes a R2.
   El vector posición puede ser considerado una posición en kilómetros respecto del centro de un planisferio mundial.

b) El utilizar r=3 y b=2 significa que habrá dos grupos de familias de hashing, que relacionan cada uno de ellos a 3 minhashes euclídeos como el que se nombró en el inciso a.
El proceso se vería como el siguiente gráfico:



En donde cada familia de hashing multiplica a cada uno de los minhashes obtenidos por un escalar, y luego se le aplica mod p mod m, siendo p un numero primo mayor a la cantidad de buckets de la tabla de hash, y m la cantidad de buckets que posee la tabla de hashing.
En resumen, la ecuación que aplican las familias es la siguiente:

$$Fn(Mh0, Mh1, Mh2) = (C0 \,.\, MH0 + C1 \,.\, MH1 + C2 \,.\, MH2) \,\%\, p \,\%\, m$$

Donde C0, C1, y C2 son constantes que establece cada familia de hashing, MH0, MH1, y MH2 son los minhashes obtenidos para dicha familia, p es un número primo mayor que m, y m es la cantidad de buckets de la tabla de hashing.

Entonces, para almacenar un usuario en la tabla lo que se debería hacer es:

- Se le aplican las r*b funciones de minhashing al usuario para obtener los valores de sus minhashes.
- Se aplican las familias con sus minhashes correspondientes.
- Los resultados de las familias son las posiciones de la tabla en las cuáles se almacenará nuestro usuario.

Y finalmente, para resolver un Query se realizaría lo mismo que se hace en el pre-procesamiento para almacenar datos, pero en vez de almacenarlo en la tabla, nos quedamos con los elementos con los cuales colisiona en la tabla.

Esto se debe a que LSH es una solución al problema de los vecinos más cercanos, y entonces aquellos con los que se colisiona, son semejantes al elemento Query.

Además, en nuestro ejemplo, nos gustaría corroborar que los elementos con los que colisionó se encuentren en un radio de 2km, por lo que se debería realizar una comparación con los elementos obtenidos de la tabla de hashing, para descartar posibles falsos positivos.

c) Considerando que la tierra tiene un ancho en planisferio de 40.000 km, y un alto de 20.000 km, y considerando los millones de usuarios que tengo, la cantidad de buckets debería ser alta para evitar una alta cantidad de falsos positivos. Consideraré por ejemplo 10.000 buckets.
Para facilitar las cuentas de euclídea, y no tener que encontrar el parámetro "w" que es difícil hallarle un valor óptimo, puedo efectuar el siguiente cálculo:

$$(Vector\ Pos\ .Hiperplano + a)\%p\%m = Minhash(Vector\ Pos)$$

Que en términos del producto escalar puede verse como:

$$(Vx\ .Hx + Vy\ .Hy + a)\%p\%m = Minhash(V)$$

Un número primo mayor que m sería 10.007, entonces me queda definido p=10007, m=10000; restaría definir 6 hiperplanos en R2, que pueden ser los siguientes:

H1 = (1,1) ; H2 = (-1, -1) ; H3 = (0, 1) ; H4 = (1, 0) ; H5 = (0, -1) ; H6 = (-1, 0)

Y por comodidad utilizaré en todos los minhashes el valor a=0.

Entonces, si pasamos al elemento Q = [-34.6,-58.4] por los minhash, obtendríamos los siguientes resultados:

MH1(Q) = (-34,6-58,4)%10007%10000 = 9914
MH2(Q) = (34,6+58,4)%10007%10000 = 93
MH3(Q) = (0-58,4)%10007%10000 = 9948
MH4(Q) = (-34,6+0)%10007%10000 = 9972
MH5(Q) = (58,4)%10007%10000 = 58
MH6(Q) = (34,6)%10007%10000 = 34

Teniendo los minhash aplico ambas familias, a las cuales le debo definir las constantes C0, C1, C2.

Para la familia 1 defino: C0 = 2, C1 = 1, C3=0
Y para la familia 2 defino: C0 = 1, C1 = -1, C3 = 2

Entonces:

F1(Mh1,Mh2,Mh3) = (2*9914+1*93+0*9948)%10007%10000 = 9914

F2(Mh4,Mh5,Mh6) = (1*9972-1*58+34*2)%10007%10000 = 9982

Entonces los elementos que se considerarían similares al Query serían aquellos que estén en las posiciones 9914 o 9982 de la tabla de LSH (Un merge de ambos conjuntos).
Finalmente, como quiero quitar los falsos positivos obtenidos por el método, entonces compararé las posiciones de los elementos que se consideran similares, con la posición del Query. De esta forma, solo me quedaré con aquellos elementos que se encuentren dentro de un radio de 2km respecto de la posición del Query.
Realizar la comparación es factible ahora debido a que me saqué de encima gran cantidad de usuarios al aplicar LSH; si no fuese por esto, se tendría que comparar por fuerza bruta con millones de usuarios, lo cuál no sería óptimo; mientras que al haber aplicado LSH, los usuarios con los que comparo, son millones de veces menos que los que serían sin utilizarlo.

d) Algunas causas que pueden generar falsos positivos son las siguientes:
    - Que un elemento no similar se establezca luego del pre-procesamiento en el mismo lugar en la tabla que el elemento Query.
    - Que no se esté siendo lo suficientemente estricto con la cantidad de minhashes. Una buena opción para remediar este problema es utilizar un r mayor, ya que comparará con más minhashes, lo que significa que se debe coincidir en más datos entre aquellos que colisionen (Es como utilizar AND minhashes, y de esta manera, pedís que más datos tengan que coincidir entre los elementos).
    - Que la cantidad de buckets sea muy baja para la cantidad de elementos que se poseen, ya que no habrá muchas opciones en las cuales almacenar al elemento.