Penerapan Decrease and Conquer dan Pattern Matching dalam Pencarian Data dengan Lokasi Terdekat

Ihsan Imaduddin Azhar - 13517043
Program Studi Teknik Informatika
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung, Jalan Ganesha 10 Bandung 40132, Indonesia
ihsaniazhar@students.ib.ac.id

***Abstrak*—Geohash adalah smetode mengkode koordinat lokasi menjadi string. String yang dihasilkan dapat digunakan untuk menjadi atribut suatu record dalam basis data dan cukup baik untuk melakukan indexing dengan attribute tersebut.**

Kata kunci-decrease and conquer; Pattern Matching; Geohash,Basis Data

I. PENDAHULUA

Dewasa ini banyak *start up* yang menyediakan berbagai jasa kepada oenggunanya. Mulai dari ojek online, layanan taksi online, layanan rekomendasi tempat makan. Kebanyakan startup yang ada, menggunakan basis data NoSQL sebagai solusi basis data yang mereka gunakan. Basis data NoSQL adalah basis data yang popular saat ini karena performnya dalam mengquery data.

Bagi start up yang sering mengquery data berdasarkan lokasi, performa query lokasi adalah menjadi sesuatu yang fatal dalam pemodelan basis data.

Salah satu cara menyimpan lokasi dalam basis data adalah dengan menyimpan data lokasi sebagai attribute dari record. Data lokasi disimpan dalam bentuk string hasil dari geohas. Geohash sendiri adalah metode pengkodean titik koordinat menjadi suatu string dengan metode decrease and conquer.

Dengan menyimpan lokasi sebagai string, pencarian data berdasarkan atribut lokasi yang sudah di *index* akan lebih trivial dibanding mmbandingkan dua attribute float.

II. LANDASAN TEORI

A. Decrease and Conquer

Decrease and Conquer adalah metode perancangan algoritma yang bekerja dengan mereduksi persoalan menjadi sub-persoalan yang lebih kecil lalu memproses satu sub persoalan yang menuju ke solusi. Tahapan dari algoritma Decrease and conquer dibagi menjadi dua bagian. Yang pertama adalah tahapan decrease. Pada tahapan decrease, persoalan akan direduksi menjadi bagian yang lebih kecil. Yang kedua adalah tahap Conquer. Pada tahap conquer, persoalan yang sudah direduksi menyelesaikan persoalan yang sudah direduksi. Algoritma Decrease and Conquer dapat dibagi menjadi tiga variasi, yakni:

1. Decrease by Constant

   Pada variasi ini, persoalan direduksi dengan konstanta yang tetap pada setiap iterasi atau decrease yang dilakukannya. Contoh algoritma yang menggunakan Decrease by Constant adalah insertion sort.

2. Decrease by a constant factor

   Pada variasi ini, persoalan direduksi dengan factor tetap pada tiap iterasi atau decrease yang dilakukannya. Contoh algoritma yang menggunakan Decrease by a constant factor adalah binary search.

3. Decrease by a variable size

   Pada variasi ini, persoalan direduksi berdasarkan nilai vaiabel pada setiap iterasi atau decrease yang dilakukannya.

B. Geohash

Geohash adalah metode pengkodean berdasarkan longitude dan longitude yang ditemukan oleh Gustao Niemeyer. Geohash adalah struktur data hierarki dalam merepresentasikan lokasi.

Pada dasarnya, geohash mengkode kordinat geografi dua dimensi menjadi string yang terdiri dari huruf dan angka. Setiap huruf dan angka dari hasil geohash suatu kordinat geografi merepresentasikan salah satu dari sekian banyak kotak yang membagi suatu area. Dengan kata lain, semakin panjang jumlah karakter dalam string geohash, semakin akurat titik yang direpresentasikan.

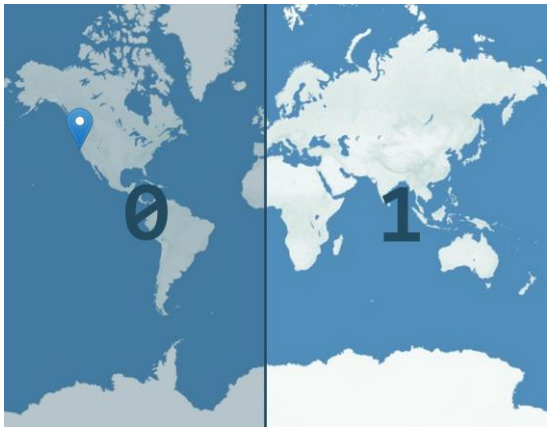
Kita mengetahui bahwa Garis Bujur (longitude) dihitung berdasarkan pengukuran sudut dari 0° di Meridian Utama ke +180° arah**timur** dan −180° arah barat, sedangkan Posisi lintang merupakan penghitungan sudut dari 0° di khatulistiwa sampai ke +90° di kutub utara dan -90° di kutub selatan. Saat

mengkode geohash dari suatu lokasi geografik, yang pertama dilakukan adalah menentukan apakah titik lokasi tersebut berada pada derajat lintang diatas 0° atau dibawah 0°. Dengan informasi kordinat dari suatu lokasi geografi, kita dapat menenrukan string hasil geohash dengan decrease and conquer.

Geohash menggunakan decrease and conquer:

- Bagi peta menjadi dua bagian, yakni bagian yang memiliki derajat bujur diatas 0 (timur)dan bagian yang memiliki derajat bujur dibawah 0 (barat). Jika titik yang sedang kita tinjau tersebut berada pada derajat lintang diatas 0°, karakter pertama dari string adalah 1 dan sebaliknya jika titik tersebut berada dibawah lintang 0°, karakter pertama dari string tersebut adalah 0.
- Setelah membagi peta menjadi dua bagian, abaikan bagian yang bukan bagian titik yang sedang kita tinjau berada.
- Bagi peta yang sedang kita tinjau menjadi dua bagian. Jika titik yang sedang kita tinjau berada pada bagian yang memiliki derajat lintang diatas 0°, karakter kedua dari string adalah 1, sedangkan jika titik yang sedang kita tinjau berada di bagian derajat lintang diatas 0°, maka karakter kedua dari string adalah 0.

Diatas adalah contoh algoritma geohashing sederhana. Dengan membagi peta dunia menjadi duabagian dan memberi nama tiap bagian menjadi 0 dan 1. Dan pada setiap pengulangan dari algoritma diatas, string yang dihasilkan akan semakin panjang. Semakin panjang string yang dihasilkan, string tersebut merepresentasikan titik yang semakin presisi. Pada contoh sebelumnya kita hanya membagi peta menjadi dua bagian atau disebut, *binary geohash encoding*.



Gambar 2.1 pembagian peta menjadi dua bagian

Gambar 2.2 pembagian bumi menjadi dua bagian berdasarkan lintang

Berikut adalah algoritma binary encoding geohashing

```
var geoHashString: String

val lat = arrayOf(-90.0, 90.0)
      val lon = arrayOf(-180.0, 180.0)
      val buffer = CharArray(10)
      for (i in 0 until 10) {
          var value = 0
              if (i % 2 == 0) {
                  val mid = (lon[0] + lon[1]) / 2
                  if (longitude > mid) {
                      value = 1
                      lon[0] = mid
                  } else
                      lon[1] = mid
              } else {
                  val mid = (lat[0] + lat[1]) / 2
                  if (latitude > mid) {
                      value = 1
                      lat[0] = mid
                  } else
                      lat[1] = mid
              }

          buffer[i] = valueToChar(value)
      }
      geoHashString = String(buffer)
  }
```

Gambar 2.3 algoritma binary geohash encoding

Pada kenyataannya geohash yang digunakan saat ini membagi peta menjadi 32 bagian pada setiap iterasinya, dan setiap karakter dari string yang dihasilkan berasal dari *base 32 character* yang terdiri dari huruf dang angka. Dengan membagi peta menjadi 32 bagian, kita mendapatkan tingkat presisi yang lebih tinggi dibanding geohash binary dengan panjang string yang sama.

Gambar 2.4 pembagian peta menjadi 32 bagian

Seperti *binary encoding geohash* langkah pengkodean dari geohash dengan 32 karakter dilakukan dengan iterasi berulang.

Geohash 32 karakter menggunakan decrease and conquer:

- Bagi peta menjadi 32 bagian, yakni table 8 x 4. Tinjau hanya kotak yang dimana titik yang kita tinjau berada dan abaikan kotak sisanya.
- Bagi lagi peta menjadi 32 bagian. Tetapi kali ini dengan table 4x8. Lalu tinjau hanya kotak dimana titik yang kita tinjau berada.
- Ulangi langkah 1 dan 2 hingga mendapatkan panjang string yang kita inginkan

Pada geohash 32 karakter, tingkat presisi dari setiap bertambahnya panjang string dapat dilihat pada gambar berikut

Precision	Dimensions
12	~ 3.7cm x 1.8cm
11	~ 14.9cm x 14.9cm
10	~ 1.19m x 0.60m
9	~ 4.78m x 4.78m
8	~ 38.2m x 19.1m
7	~ 152.8m x 152.8m
6	~ 1.2km x 0.61km
5	~ 4.9km x 4.9km
4	~ 39km x 19.5km
3	~ 156km x 156km
2	~ 1,251km x 625km
1	~ 5,004km x 5,004km

Gambar 2.5 tingkat presisi berdasarkan panjang string geohash 32 karakter

Dalam pengkodean string lokasi, string dengan panjang 10 karekter sudah cukup presisi untuk menentukan posisi secara akurat.

Berikut adalah source code dalam pengkodean geohash string.

```
private const val 32_CHAR =
       "0123456789bcdefghjkmnpqrstuvwxyz"

fun valueToCharr(value: Int): Char {
           return 32_CHARS[value]
       }

val BITS = arrayOf(16, 8, 4, 2, 1)
var geoHashString: String

val lat = arrayOf(-90.0, 90.0)
       val lon = arrayOf(-180.0, 180.0)
       val buffer = CharArray(10)
       for (i in 0 until 10) {
           var value = 0
           for (j in 0 until 5) {
               val evenBit = (((i* 5) + j) % 2) == 0
               if (evenBit) {
                   val mid = (lon[0] + lon[1]) / 2
                   if (longitude > mid) {
                       value = value or BITS[j]
                       lon[0] = mid
                   } else
                       lon[1] = mid
               } else {
                   val mid = (lat[0] + lat[1]) / 2
                   if (latitude > mid) {
                       value = value or BITS[j]
                       lat[0] = mid
                   } else
                       lat[1] = mid
               }
           }
           buffer[i] = valueToChar(value)
       }
       geoHashString = String(buffer)
   }
```

Gambar 2.6 algoritma geohash 32 karakter

C. String Matching

String matching atau pencocokan string adalah algoritma yang digunakan untuk mencari kemunculan dari sebuah pattern string yang kita tinjau pada string lain yang lebih besar. String matching banyak diterapkan dalam autocomplete suatu kolom pertanyaan.

Contoh :

Input: text: “AABAACAADAABAABA”

text: “AABA”

Output: Pola ditemukan pada index 0, 9, 12.

Gambar 2.7 contoh ilustrasi string matching

D. Database

Basis data adalah kumpulan informasi yang disimpan didalam computer secara sistematik sehingga pengaksesan dan pembaruan data dapat dilakukan dengan mudah. Basis data sudah menjadi kebutuhan orgianisasi saat ini.

Basis data yang saat ini sedang popular bisa dibagi menjadi dua, yaitu basis data SQL dan basis data NoSQL

Basis data SQL

SQL adalah singkatan dari Structured Query Language. SQL adalah Bahasa standar yang digunakan untuk mengoperasikan database relasional.

SQL dikembangkan pertama kali pada awal 1970 oleh Raymond Boyce dan Donald Chamberlin yang bekerja pada IBM.

Basis data NoSQL

Basis data NoSQL adalah jenis database yang sedang popular saat ini. Consep dari basis data NoSQL tumbuh karena dipopulerkan oleh Google, Facebook, Amazon yang berurusan dengan jumlah data yang sangat besar.

NoSQL pertama kali dikembangkan karena kekurangan dari database SQL yaitu semakin jumlah datanya besar, waktu respon untuk querynya semakin lambat. Dengan database NoSQL.

Basis data NoSQL memiliki performa yang baik untuk query karena pada basis data noSQL, semua data disimpan dalam satuan data yang disebut document. Berbeda dengan basis data SQL yang memerlukan banyak join untuk mendapatkan informasi, basisdata NoSQL menyimpan semua informasi yang dibutuhkan dan sering digunakan bersama dalam satu dokumen tanpa mempedulikan redundansi dari data tersebut. Dengan mengorbankan hal tersebut, basis data NoSQL memiliki performa yang sangat baik dalam merespon query.

III. PEMBAHASAN

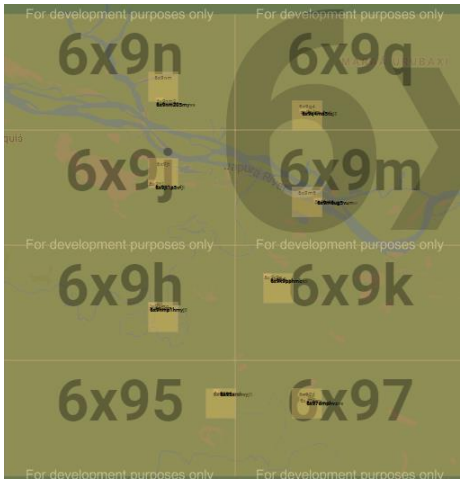
1. Menyimpan Data

   Untuk melakukan pencarian data menggunakan lokasi terdekat, atribut lokasi dari data tersebut juga harus disimpan ke database. Dan pada database, lokasi disimpan dalam bentuk string. String tersebut adalah string yang dihasilkan dari algoritma geohash

2. Pengaksesan data

   Dengan Geohash pencarian data yang memiliki lokasi terdekat bisa dilakukan dengan berbagai cara. Diantaranya dengan string matching.

2.2 Pengaksesan data dengan String matching



Gambar 3.1 contoh lokasi representasi string geohash

Dapat diperhatikan dari gambar diatas, lokasi yang bertetanggaan memiliki string yang terdiri dari karakter yang sama selain karakter terakhir.

Dengan informasi tersebut kita dapat mencari lokasi yang berdekatan dengan mencocokkan string hasil geohash yang kita miliki.

Langkah pertama untuk mencari data yang memiliki lokasi terdekat dengan lokasi yang kita inginkan, pertama kita harus meng-*hash* lokasi yang kita ingin tinjau dengan geohash.

Misalkan kita ingin mencari data yang memiliki lokasi dekat dengan -6.21462, 106.84513. Geohash string dari kordinate tersebut adalah qqguxkdjpyz7. Dari string yang kita dapat dari lokasi yang kita tinjau, kita dapat menghilangkan karakter paling terakhir dari string tersebut lalu menjalankan algoritma string matching untuk mencari data yang memiliki lokasi yang memiliki awalan sama dengan string geohash dari lokasi yang kita tinjau.

Menghilangkan satu karakter paling belakang dari string geohash 10 karakter akan memberikan data dengan ketelitian lokasi sekitar 4.9m. dengan menghilangkan 2 karakter terakhir, ketelitian turun menjadi 48m.

Selain dengan string matching, kita juga dapat mencari data yang memiliki lokasi nearby dengan metode bounding box

2.3. Metode Bounding Box

Ide dari metode bounding box adalah membuat kotak yang dari titik titik di sekeliling lokasi yang ingin kita tinjau.

Untuk metode bounding box, pertama kita harus menentukan berapa perbedaan jarak maksimum horizontal dan jarak maksimum vertikal. Setelah menentukan jarak vertikal dan horizontal, kita dapat menghitung derajat lintang dan derajat bujur dengan rumus

```
double R = 6371;

double  lat1 = lat
                - Math.toDegrees(radius/R);
double lng1 = lng
                - Math.toDegrees(radius/R/
                Math.cos(Math.toRadians(lat)));

double lng2 = lng
                + Math.toDegrees(radius/R/
                Math.cos(Math.toRadians(lat)));
double lat2 = lat
                + Math.toDegrees(radius/R);
```

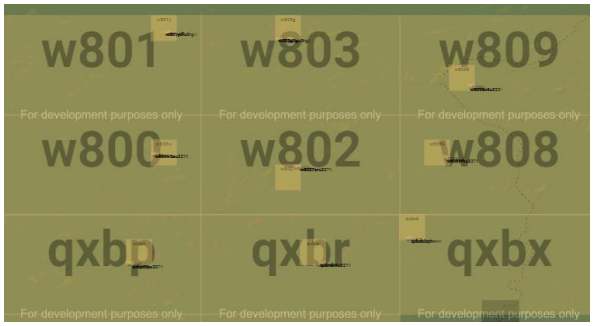
Gambar 3.2 rumus mencari kordinat bounding box

Langkah kedua adalah menentukan string geohash dari titik pojok kiri bawah dan pojok kanan atas dari bounding box yang kita dapat.

Langkah ketiga adalah mengquery data yang ada di database berdasarkan bounding point yang kita dapat.

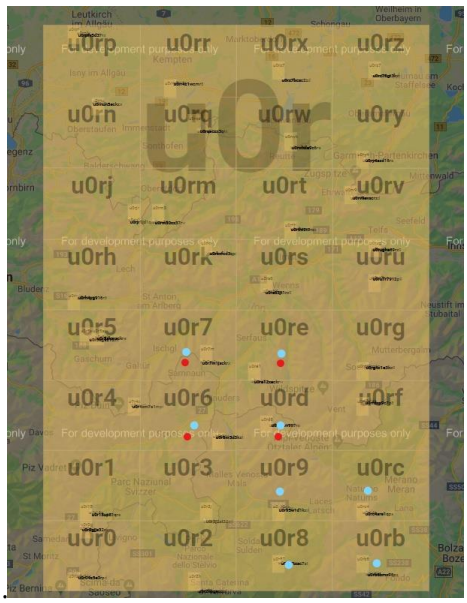
IV. KESIMPULAN

Pengaksesan data berdasarkan lokasi pada basis data NoSQL bisa dilakukan menggunakan geohash yang menggunakan decrease and counquer. Tapi masih banyak kekurangan dari query data dengan string geohash. Yang pertama adalah jika kita menggunakan metode string matching untuk query data dengan lokasi terdekat, sedangkan lokasi yang kita tinau adalah tepat kotak diatas garis khaltulistiwa, kotak tetangga yang ada dibawah tidak akan terhitung sebagai lokasi terdekat karena string geohash tidak memiliki awalan yang sama.



Gambar 4.1 representasi kotak pada dekat garis katulistiwa

Yang kedua adalah jika kita menggunakan metode bounding box, data yang didapat pada banyak kasus berada diluar kotak yang kita cari



Gambar 4.2 hasil query menggunakan metode bounding box

ACKNOWLEDGMENT *(Heading 5)*

Pertama tama, saya sebagai peulis ingin memanjatkan puji syukur kepada Tuhan yang Maha Esa atas berkat dan rahmat-nya sehingga penulis dapat menyelesaikan makalah ini. Penulis juga berterimakasih kepada dosen pengampu IF2120, mata kuliah Strategi Algoritma, untuk memberikan kesempatan kepada saya untuk menulis makalah ini. Penulis juga berterimakasih kepada orang tua penulis karena telah memberikan dukungan moral kepda saya selama masa

penulisan, penulis juga berterimakasih kepada penulis dari referensi dibawah untuk kontribusi mereka dalam ilmu computer.

REFERENCES

[1] Munir, Rinaldi.Diktat Kuliah IF2251 Strategi Algoritmik. Institut Teknologi Bandung. 2007.http://geohash.gofreerange.com/.

[2] https://www.geeksforgeeks.org/kmp-algorithm-for-pattern-searching/

[3] https://www.geeksforgeeks.org/wp-content/uploads/Pattern-Searching-2-1.png

PERNYATAAN

Dengan ini saya menyatakan bahwa makalah yang saya tulis ini adalah tulisan saya sendiri, bukan saduran, atau terjemahan dari makalah orang lain, dan bukan plagiasi.

Bandung, 29 April 2012

Ihsan Imaduddin Azhar dan 1351703