ANALISI PELACAKAN MOBIL DENGAN METODE BFS (*Breadth First Search*)

Ibnu Mas'ud (13506034)

Program Studi Teknik Informatika
Sekolah Teknik Elektro dan Informatika Institut Teknologi Bandung
e-mail: ibnumasud_88@yahoo.com

ABSTRAK

Pada zaman sekarang ini, proses pelacakan sesuatu barang sangat berkembang dengan pesat. Akan tetapi tidak semuat teknologi tersebut memiliki solusi yang optimum. Oleh karena, banyak praktisi di bidang *science* melakukan penelitian-penelitian untuk menemukan solusi untuk proses pelacakan. Salah satu solusinya yaitu dengan teknik komputasi. Di dalam teknik komputasi terdapat berbagai macam algoritma untuk menyelesaikannya. Tujuan pelacakan ini adalah menemukan lintasan mobil yang hilang dalam suatu kota, bermulai dari terakhir kali mobil tersebut di parkirkan sebelum hilang. Dalam metoda analisi yang akan kita pakai kita membutuhkan sebuah algoritma yang dapat menelusuri kota tempat mobil itu hilang sampai mobil itu diketemukan. Untuk melakukannya banyak algoritma yang dapat memenuhinya. Akan tetapi, penulis akan mencoba memaparkan metoda pelacakan dengan algoritma BFS (*Breadth First Search*). Dengan metoda BFS ini, mobil yang hilang harus memiliki semacan alat yang memancarkan informasi.

Kata kunci: *Breadth First Search*, menemukan, lintasan, mobil.

1. PENDAHULUAN

Pada zaman sekarang ini, pencarian mobil yang hilang dapat dilakukan dengan berbagai macan cara. Dengan alat GPS kita dapat melacak langsung dimana posisi mobil hilang tersebut. Penulis mencoba memberikan solusinya dengan metoda BFS (*Breadth First Search*).

2. ANALISI PENCARIAN MOBIL

2.1 Pencarian Mobil

Suatu ketika terjadi pencurian mobil di dalam sebuah kota. Lalu pencuri mobil itu mengendarainya ke suatu tempat di kota itu juga. Di dalam mobil tersebut telah terpasang alat pengaman yang selalu memancarkan informasi tertentu mengenai mobil tersebut saat mobil itu bergerak.

Alat tersebut bukan sejenis alat modern yang dapat memancarkan informasi koodinat keberadaannya, melainkan suatu alat yang hanya mendeteksi ke arah mana mobil itu menghadap menurut empat arah mata angin: utara, selatan, barat, dan timur, dan memancarkannya. Alat hanya memancarkan satu kali saja sebelum terjadi terjadi perubahan berikutnya.

Jalan-jalan di kota itu vertikal atau horisontal dengan simpangan-simpangan pada posisi bilangan bulat. Data yang terekam terakhir dari mobil itu adalah posisi mobil diparkir terakhir sebelum hilang (kita sebut pada posisis awal), kemudian sejak itu terekam sederetan sinyal-sinyal arah hadap mobil tersebut hingga kemudian berhenti entah di mana.

2.2 Batasan Pencarian Mobil

Dalam analisi penyelesaian masalah pencarian mobil dengan algoritma BFS (*Breadth First Search*) terdapat batasan-batasan yang harus dipenuhi sebelumnya. Yang pertama mobil tersebut harus memiliki alat yang dapat mendeteksi kea rah mana mobil itu menghadat menurut empat arah mata angin: utara, selatan, barat, dan timur, dan memancarkannya.

Yang kedua jalan-jalan di dalam kota tersebut dibatasi, yaitu hanya vertical atau horizontal dengan simpangan-simpangan pada posisi bilangan bulat.

Alat yang terpasang di dalam mobil memberikan data yang di asumsikan menjadi file teks bernama lacak.in yang berisi data dalam format sebagai berikut

- Baris pertama dari file masukan berisi dua bilangan integer R dan C, $1 \le R \le 50$, $1 \le C \le 50$, yang dipisahkan oleh suatu karakter spasi, masing-masing menyatakan baris dan kolom dari peta kota tersebut.
- Pada setiap dari R baris berikutnya terdapat deretan C buah karakter: 'X', '.' atau '*' yang

menjelaskan bagian-bagian dari peta seperti pada penjelasan di atas.

- Setelah itu, pada baris ke (*R*+2) tedapat bilangan integer *N*, 1 ≤ *N* ≤ 1000, yang menyatakan banyaknya sinyal arah mobil yang terekam.
- Masing-masing dari *N* baris berikutnya berisi string-string **NORTH**, **SOUTH**, **WEST** dan **EAST**, yang menyatakan arah gerakan mobil.
- Tidak ada dua sinyal berurutan yang sama.

2.3 Dasar Teori

Dalam metode analisis, kita akan menggunakan algoritma BFS (Breadth First Search).

BFS merupakan salah satu pencarian solusi masalah yang direpresentasikan dengan graf. Algoritma BFS adalah sebagai berikut : Kunjungi simpul x, kemudian semua simpul yang bertetangga dengan simpul z dikunjungi terlebih dahulu. Selanjutnya, simpul yang belum dikunjungi dan bertetangga dengan simpul-simpul tadi dikunjungi, demikian seterusnya. Jika graf berbentuk pohon berakar, maka semua simpul paad aras y dikunjungi terlebih dahulu sebelum simpul-simpul pada aras d+1[1].

Algoritma BFS memerlukan sebuah antrian q untuk menyimpan simpul yang telah dikunjungi. Simpul-simpul yang telah dikunjungi suatu saat diperlukan sebagai acuan untuk mengunjungi simpul-simpul yang bertetangga dengannya. Tiap simpul yang telah dikunjungi masuk ke dalam antrian hanya satu kali.

2.4 Analisis

Pencarian mobil yang hilang ini dapat diselesaikan dengan algoritma BFS karena persoalan ini dapat diumpamakan dengan graph traversal BFS (Breadth Firsh Search). Khusus untuk masalah ini BFS dapat diimplementasikan tanpa menggunakan queue untuk menyimpan pencabangannya (breadthnya).

Karena, dari suatu titik dan arah, kita mendapatkan sejumlah titik berikutnya sebagai breadth dalam BFS. Karena ukuran kota yang diumpamakan dengan matrix 50x50 maka untuk kasus terburuk akan ada 49x49=2401 titik dalam breadth. Karena searching space sangat terbatas maka breadth tidak akan mengalami "peledakan".

Jadi kita menggunakan array breadth yang setiap elemennya menyimpan koordinat titik pencabangan. Dalam inisialisasi array hanya berisi satu titik yaitu posisi awal mobil.

Dalam setiap sinyal arah yang dibaca maka dari setiap titik dalam array breadth ini kita temukan kembali semua kemungkinan titik berikutnya di dalam peta dan menandainya. Setelah hal tersebut dilakukan untuk semua titik dalam breadth maka titik-titik pada peta yang telah ditandai dicatat ke dalam array breadth.

Hal tersebut dilakukan untuk setiap masukah yang dibaca dan setelah masukan arah terakhir maka tanda-tanda pada peta menyatakan semupa kemungkinan posisi mobil terakhir.

Dalam implementasinya kita perlu definisikan type posisi berikut.

```
type posisi=record r, c: integer end;
```

Breadth adalah array dari elemen bertype posisi ini. Selanjutnya kita perlu prosedur untuk memindahkan posisi bertanda "*" ke dalam array breadth.

```
nb := 0;
for i := 1 to nrow do
   for j := 1 to ncol do
    if peta[i,j] = '*' then
      begin
           inc(nb);
           breadth[nb].r := i;
           breadth[nb].c := j;
           peta[i,j] := '.';
{ clearkan posisi tsb}
            end;
```

Lalu untuk setiap titik breadth[b] dilakukan penandaan kembali posisi selanjutnya sesuai dengan arah yang diberikan. Misalnya untuk arah EAST maka

```
posNext := breadth[b];
inc(posNext.c);
while Jalan(posNext) do
begin
   peta[posNext.r,posNext.c] := '*';
    inc(posNext.c);
end;
```

Untuk arah NORTH maka lakukan dengan

```
posNext := breadth[b];
dec(posNext.r);
while Jalan(posNext) do
begin
   peta[posNext.r,posNext.c] := '*';
    dec(posNext.r);
end;
```

Begitu pula untuk arah-arah yang lainnya.

Untuk mencek keadaan sedang jalan atau tidak maka kita membutuhkan method Jalan yang memeriksanya.

```
function   Jalan(posNext:   posisi):
boolean;
begin
    ....
    { kalau keluar peta maka false }
    { kalau tembok maka false }
    { kalau tidak maka true }
    ....
end;
```

Setelah semua titik pada breadth diproses maka breadh dikosongkan kembali dan diisi oleh titik-titik pada peta yang bertanda "*" kembali (dengan memanggil procedure di paling atas).

Setelah arah masukan terakhir maka peta langsung dituliskan

Contohnya , peta masukan berikut ini

```
..........
.X.X..XXX.
.X..X.....
.X..X.....
....X.....
*......XX.
..........
```

Breadth hanya berisi 1 titik, yaitu (6.1). Jika kemudian masukan arah adalah EAST maka breadth kemudian akan berisi titik-titik di samping kanan tanda bintang yaitu posisi-posisi kosong sampai ketemu tembok. Sekarang peta menjadi sbb.

```
..........
.X.X..XXX.
.X..X.....
.X..X.....
....X.....
.******XX.
..........
```

Breadth hanya berisi 6 titik, yaitu (6.2), (6,3),...,(6,7). Jika kemudian masukan adalah NORTH maka hal seperti di atas dilakukan ke arah atas untuk masing-masing tanda bintang. Dan peta menjadi sbb.

```
..*..*....
.X*X.*XXX.
.X**X**...
.X**X**...
.***X**...
.......XX.
..........
```

Jika kemudian masukan adalah WEST maka hal seperti di atas dilakukan ke arah atas untuk masing-masing tanda bintang. Dan peta menjadi sbb.

```
*****.....
.X.X*.XXX.
.X*.X*....
.X*.X*....
***.X*....
.......XX.
..........
```

Jika kemudian masukan adalah NORTH maka hal seperti di atas dilakukan ke arah atas untuk masing-masing tanda bintang. Dan peta menjadi sbb.

```
*.*.**....
*X*X.*XXX.
*X*.X*....
*X*.X*....
....X.....
.......XX.
..........
```

Jika kemudian masukan adalah WEST maka hal seperti di atas dilakukan ke arah atas untuk masing-masing tanda bintang. Dan peta menjadi sbb.

```
*****.....
.X.X*.XXX.
.X..X.....
.X..X.....
....X.....
.......XX.
..........
```

Jika kemudian masukan adalah NORTH maka hal seperti di atas dilakukan ke arah atas untuk masing-masing tanda bintang. Dan peta menjadi sbb.

```
....*.....
.X.X..XXX.
.X..X.....
.X..X.....
....X.....
.......XX.
..........
```

Jika sinyal masukan sudah habis maka tanda-tanda bintang menyatakan semua kemungkinan mobil itu berada saat ini.

IV. KESIMPULAN

Kesimpulan yang dapat kita dapat adalah bahwa dengan algoritma *Breadth First Search* kita dapat menemukan mobil yang hilang. Karena dibatasi kotanya memiliki n buah simpul graf maka cara diatas dapat dipakai. Akan tetapi, apabila simpulnya banyak sehingga menimbulkan peledakan apabila di turunkan semua cabangnya maka penyelesaian dapat ditambahkan dengan queue yang menampung simpul yang telah dikunjungi sehingga kita tidak harus menurunkan semua cabang dari simpul graf tersebut.

REFERENSI

[1] Munir,Rinaldi,*Strategi Algoritmik,2008*
[2] Soal seleksi TOKI tahun 2007