Implementasi Brute Force
pada Game Mahjong Titans

Yogi Adytia Marsal - 13508016
Program Studi Teknik Informatika
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung, Jl. Ganesha 10 Bandung 40132, Indonesia
IF18016@itb.ac.id

ABSTRAK

***Game Mahjong Titans* merupakan salah satu permainan standar yang terdapat pada windows yang bertujuan menyamakan semua gambar yang sama ataupun setipe. Gambar tersebut tertumpuk-tupuk sehingga tidak semua gambar bisa kerjakan dalam waktu bersamaan, Hanya ada beberapa gambar yang bisa dikerjakan. Selain itu tidak semua gambar bisa dilihat pada waktu awal permainan tapi seiring berjalannya permainan gambar tersebut akan terlihat. Jika satu gambar di samakan, maka gambar itu akan di keluarkan dari tumpukan permainan dan akan menambah ataupun mengurangi banyak gambar yang bisa diselesaikan. Hal ini dilakukan terus hingga semua gambar disamakan semua.**

Banyak algoritma yang bisa di gunakan untuk menyelesaikan permainan ini. Salah satunya adalah dengan menggunakan pohon dengan metoda pencariannya solusi menggunakan algoritma "Brute Force".

Algoritma ini menggunakan matriks yang salah satu ukuran 72x72. Hal ini karena jumlah semua mahjong yang tersusun adalah 144 buah dan tentu pastinya ada dua gambar yang sama. Matriks inilah yang akan berisi data-data pohon dimana baris dari matriks tersebut merupakan kedalaman dari pohon tersebut yaitu 72.

***Kata Kunci—Brute Force*, Matriks,pohon, mahjong titans.**

I. PENDAHULUAN

kita mendengar kata mahjong, tentu kita akan teringat dengan permainan judi asal cina yang sangat terkenal yang di mainkan oleh empat orang, tapi pada kesempatan ini penulis tidak bermaksud membahas Mahjong tersebut tapi Mahjong titans yang dimainkan oleh satu orang dan merupakan permainan komputer versi mahjong solitaire yang di kembangkan oleh Oberon Games yang terdapat dalam produk operating sistem *microsoft*. Permainan ini memiliki beberapa macam penentuan susunan gambar sesuai yang di inginkan user seperti gambar berikut ini.

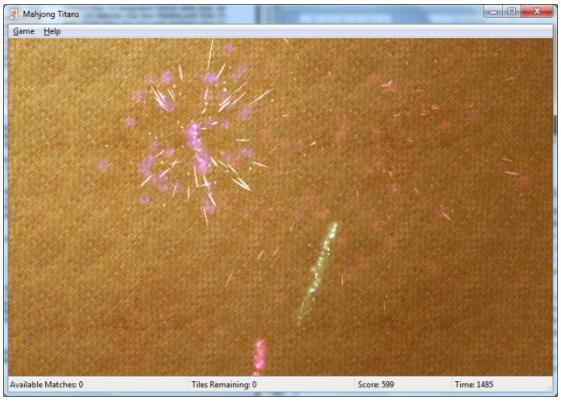
Gambar 1. Pemilihan Susunan Mahjong Titans

Ada 6 macam susunan yang bisa di pilih, setiap susunan memiliki tingkat kesulitan yang berbeda-beda. 6 susunan tersebut adalah, *Turtle*, *Dragon*, *Cat*,*Fortress*,*Crab* dan *Spider*.

Setelah memilih susunan tersebut barulah permainan dimulai. Mulailah pemain untuk menyamakan gambar tersebut satu persatu. Tidak semua gambar bisa disamakan karena masih tertutup oleh gambar lain. Gambar yang bisa di selesaikan adalah gambar yang teratas dan gambar yang terletak paling kiri atau kanan saja, walaupun gambar yang dibawahnya sudah bisa terlihat tapi karena bukan gambar yang terluar(terkiri atau terkanan) tetap tidak bisa di kerjakan.

Gambar 2. Contoh soal Mahjong Titans Turtle

Pada permainan ini ada 4 gambar yang memiliki kesamaan sehingga ada kemungkinan jika salah dalam memilih jalan yang digunakan akan membuat permainan ini tidak bisa diselesaikan dikarenakan gambar sama tinggal 2 buah dan gambar tersebut terdempet. Jika semua gambar bisa disamakan dengan kata lain game terselesaikan maka pemain akan menang dan akan memperoleh waktu pengerjaan dan poin, semakin cepat pemain menyelesaikan, semakin besar poin yang di peroleh.



Gambar 3. Soal Mahjong Titans yang sudah terselesaikan

Permasalahan pada permainan ini dapat diselesaikan dengan salah satu algoritma pencarian solusi yaitu *brute force*. Algoritma ini akan merepresentasikan permainan ini kedalam pohon, metode yang digunakan mirip dengan Breath First Search(BFS). Cuman karena semua kemungkinan dicoba menyebabkan algoritma ini lebih mengarah ke algoritma *brute force*, tapi tetap saja pencarian nya melebar seperti algorima BFS.

II. PENERAPAN ALGORITMA BRUTE FORCE

Penyelesaian “Mahjong Titans” dengan metode Brute Force membutuhkan pohon dengan kedalaman setegah dari jumlah mahjong yang disusun yaitu 72, karena dalam permainan ini pasti ada 2 gambar yang memiliki gambar yang sama atau setipe dan oleh karena itu total mahjong sebanyak 144 pasti akan menyamakan sebanyak 72 kali.

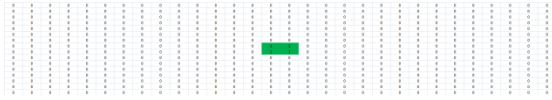
Dalam Permainan ini tidak semua gambar dapat diselesaikan dalam waktu bersamaan karena masih ada gambar yang tertutup dan pemain belum tahu apa gambar yang tertutup itu, hal inilah yang membuat pemain akan terjebak kedalam jalan buntu jika salah dalam menentukan gambar.

II.I Pemodelan Pohon

Pada bagian ini merupakan pemodelan dari permainan Mahjong Titans agar bisa diselesaikan dengan menggunakan algoritma Brute Force. Pemodelan memasukkan pola yang dapat di selesaikan dalam permainan ini kedalam bentuk pohon. Untuk setiap gambar di beri penomoran dari 1-36 karena satu gambar terdapat pada 4 buat mahjong sedangkan jumlah mahjong yang digunakan adalah 144 buah.

Untuk tahap awal kita masukkan semua persoalan mahjong tersebut kedalam array tiga dimensi yang berukuran sama. Jika di kotak tersebut tidak terdapat mahjongnya maka dalam matriks akan diisi dengan 0. Tetapi mahjong yang masih tertutup diisi dengan 37. Untuk satu mahjong mengambil tempat 2 kolom 2 baris. Karena lebar nya ada 15 mahjong dan tingginya ada 8 mahjong brarti ukuran matriksnya adalah 30x16.

Gambar 4. Soal Mahjong Titans



Gambar 5. Simulasi matriks teratas yang terdapat dalam array tiga dimensi

Selanjutnya dalam pemimplementasian pohon ini tentunya terlebih dahulu perlu membutuhkan fungsi yang menentukan berapa banyak mahjong yang bisa disamakan. Tapi sebelumnya dibuat dulu fungsi yang menghasilkan semua mahjong yang terletak dibagian yang bisa diselesaikan.

Fungsi ini memasukkan semua mahjong yang bisa terletak paling atas dan paling luar(terkiri atau terkanan tapi bukan yang paling atas maupun yang paling bawah) kedalam sebuat array yang nantinya akan dimasukkan kedalam pohon yang dirancang untuk menentukan yang mana saja yang bisa diselesaikan.

Ada beberapa kemungkinan yang terjadi pada 1 buah jenis mahjong ataupun 1 gambar mahjong, yaitu

1. Hanya satu gambar yang sama dalam waktu yang tertentu. Jika terjadi kasus ini berarti gambar di abaikan karena tidak memiliki pasangan dengan kata lain tidak di masukkan kedalam pohon.
2. Ada dua gambar yang bisa disamakan dalam waktu tertentu. Jika terjadi kasus ini maka gambar ini langsung dimasukkan kedalam pohon karena tidak ada kemungkinan lain yang bisa disamakan lagi.
3. Ada tiga gambar yang bisa disamakan pada waktu tertentu. Jika terjadi kasus ini maka akan terjadi tiga kemungkinan juga yaitu :
    a. Penyamaan gambar 1 dengan gambar 2
    b. Penyamaan gambar 1 dengan gambar 3
    c. Penyamaan gambar 2 dengan gambar 3

    Ketiga kemungkinan diatas dimasukkan semuanya kedalam pohon`sebagai semua kemungkinan yang bisa terjadi untuk disamakan. Karena jika terjadi kemungkinan diatas ada kemungkinan dari salah satu kasus tersebut yang akan membuat permainan tidak dapat diselesaikan, misalkan ada gambar yang sama dibawah salah satu gambar tersebut.
4. Ada empat gambar yang bisa disamakan dalam waktu yang bersamaan maka jadikan empat gambar tersebut menjadi dua pasang gambar yang sama secara bebas lalu masukkan ke dalam pohon karena bagaimana pun kombinasinya gambar tersebut sudah pasti akan selesai.

Header untuk pembentukan pohon tersebut sebagai berikut:

```
Typedef MatSol:< bapak: int,
                 Tree : Matriks>


procedure TreeMaker
(input/output Tree:matriks,input
solusi: Matriks)

{procedure yang digunakan untuk
membuat pohon dengan
memanfaatkan procedure-procedure
lain seperti Terluar dan
PencariPasangan}


Procedure MahjongtoMartiks

(output Mahjong: Array Of
Matriks)

{procedure ini merupakan
procedure untuk membuat array of
matriks yang mensimulasikan
susunan mahjong kedalam Array of
Matriks}


function Terluar(input Mahjong:
array of Matriks, output
Terluar: Matriks)

{fungsi yang berguna membaca
array tiga dimensi yang
dimasukkan dan menghasilkan
Matriks yang berisi semua
mahjong yang terluar}


function PencariPasangan(input
Terluar: Matriks, output solusi
: MatSol)

{fugsi ini berfungsi untuk
menseleksi pasangan dari mahjong
terluar, fungsi ini menghasilkan
matriks yang berisi semua
kemungkinan pasangan gambar
mahjong tersebut}
```

```
procedure Update()

{procedure yang berfungsi untuk
meng-update kemungkinan yang
bisa setelah salah satu
kemungkinan yang ada dieksekusi}


procedure Open(input/output
mahjong: array of matriks, input
layer : int , input posisi :
point, input Yg_terbuka
:matriks)

{procedure yang berfungsi untuk
meng-update data mahjong yang
telah terbuka}

procedure FindPath(input solusi:
MatSol, output path : Matriks,
input possol:int)

{procedure ini merupakan
prosedur untuk menemukan path
dari solusi dengan menelusuri
bapak dari possol(posisi solusi)
hingga posisi yang tidak punya
bapaknya lagi atau Parent
utamanya}
```

Pada permainan ini masih ada bagian yang tertutup. Dan hal ini tidak bisa di tebak mahjong apa yang tertutup tersebut secara sembarangan karena hanya ada 4 mahjong yang mungkin dalam satu permainan. Jadi solusi yang di peroleh belum tentu sama juga. Untuk fungsi yang open tersebut akan meng-*generate* mahjong yang tertutup tersebut dari data mahjong yang telah terbuka. Dengan cara memilih secara random dari 32 kemungkinan jenis mahjong yang mungkin muncul dan membandingkan total mahjong yang telah terbuka(maksimal ada empat mahjong terbuka) . jika mahjong yang telah terbuka tersebut belum mencapai empat buah dari hasil random tersebut, maka angka tersebut akan dimasukkan kedalam *array of matriks* yang berisi data susunan mahjong tersebut. Agar tidak terjadi pengulangan data, misalkan tipe mahjong 1 sebenarnya sudah diketahui telah terbuka 4 kali, tapi 2 telah diselesaikan, maka dari hasil random, misalkan bisa di peroleh angka satu maka bisa saja tipe mahjong 1 tersebut digunakan lagi, untuk mencegah itu perlu juga di bandingkan dengan mahjong yang telah terbuka. Dan yang tentunya harus diisi adalah bapak dari pilihan tersebut yaitu susunan yang sebelum memilih pilihan tersebut.

Fungsi dan procedure yang ada tersebut di susun sedemikian rupa untuk bisa di looping hingga persoalan berhasil ditemukan, hal ini akan membentuk suatu pohon n-ner. Jika Pohon telah terbentuk maka kita bisa melanjutakan ketahap berikutnya yaitu tahap pencarian dengan metoda *brute force*.

II.II Algoritma Brute Force

Metoda Brute Force, dari artinya kita sudah bisa mengetahui metoda yang digunakan. Metoda ini memperoleh solusi dari suatu permasalahan dengan cara mencoba semua kemungkinan yang akan terjadi setiap pengambilan keputusan dalam penyelesaian permasalahan tersebut. Akibatnya jika mendapatkan kasus terburuk akan memperoleh waktu eksekusi yang sangat lama.

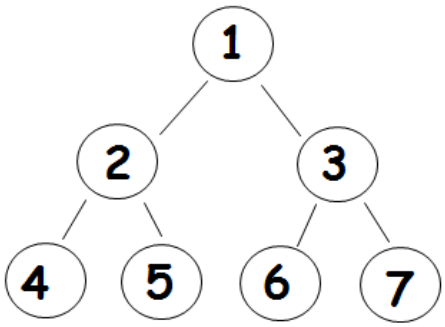
Sebelumnya kita telah membuat list yang berisi bagian-bagian mana salah yang bisa di samakan atau diselesaikan. Setelah itu dijadikan menjadi sebuah pohon. Pohon tersebut dibuat secara recursif dalam setiap pengulangan.

Pada setiap pengulangan dimasukkan semua kemungkinan yang di update terus sehingga semua kemungkinan tersebut tidak terlewatkan. Prosedure update berfungsi agar pembuatan daerah yang bisa disamakan lebih efisien. Dengan cara hanya memperhitungkan daerah yang berubah saja. Dengan demikian daerah yang terbentuk setiap looping akan bisa di tentukan. Update ini juga merupakan fungsi yang akan menerima gambar baru dari mahjong yang sebelumnya tertutup oleh mahjong lainnya. Sebelum melakukan update seharusnya dilakukan dulu procedure open untuk mengetahui meng-update susunan mahjong yang telah terbuka.

Selanjutnya akan dilooping terus sampai ditemukannya state dimana tidak array yang dihasilkan oleh procedure update tidak ada lagi atau hasilnya adalah array kosong. Tapi mahjong masih belum terselesaikan, maka akan lanjut ketahap berikutnya dan tahap yang sebelumnya akan diabaikan.

Pembuatan pohonnya yang terjadi karena terjadi looping ini di buat dari pohon induk trus anak paling kirin sampai anak paling kanan. Lalu selanjutnya pada kedalaman berikutnya, mulai dari anak paling kiri hingga anak paling kanan hingga ditemukan solusi. Jika ada pengabaian, maka

simpul tersebut akan diabaikan dan tidak akan dilanjutkan lagi.



Gambar 5. Simulasi matriks teratas yang terdapat dalam array tiga dimensi

Dengan menggunakan metoda *brute force* ini persoalan dalam penyusunan mahjong tersebut dapat selesaikan, tapi masih ada kemungkinan untuk permainan tidak bisa diselesaikan. Salah satunya yaitu saat ke empat mahjong saling menutupi sehingga tidak bisa diselesaikan.

Setelah di temukan solusi atau dimana semua mahjong telah disamakan maka persoalan kita belum selesai, kita harus menentukan path dari solusi tersebut. Dengan menggunakan bagian dari *MatSol* yang merupakan struktur yang berisi matriks dan integer berupa urutan bapaknya, kita bisa menemukan path dari solusi tersebut, setelah di tentukan maka selesailah permasalahan kita dengan menggunakan fungsi *FindPath*.

III. KESIMPULAN

Dalam memecahkan persoalan dalam **Mahjong Titans**, algoritma Brute Force dapat menjadi salah satu alternatif dalam menyelesaikan permasalahan tersebut. Tetapi algoritma ini juga tidak dapat dinilai paling bagus, karena masih banyak algoritma lain yang dapat menyelesaikan persoalan klotski, apa lagi algoritma ini mencoba semua kemungkinan yang ada. Dan tentunya juga memilki kasus terburuk dan kasus terbaiknya, jika hasil di temukan di ujung kanan bawah maka itu merupakan kasus terburuk sedangkan jika di temukan pada kiri bawah maka itu merupakan kasus terbaiknya. Hasil diperoleh tersebut pasti akan ditemukan pada kedalaman 72 karena pasti akan melakukan penyamaan angka sebanyak 72 kali. Hasil yang di peroleh oleh algoritma ini untuk setiap pengujian belum tentu sama, karena untuk soal yang sama ada kemungkinan kombinasi mahjong yang tertutup yang berbeda yang di generate secara random, hal inilah yang membuat hasil yang di peroleh berbeda setiap kali pengujian.

Pada dasarnya penyelesaian persoalan ini mirip dengan penyeselesaian menggunakan DFS, yaitu penelusuran dari cabang yang memiliki kedalaman yang sama, tapi karena semua kemungkinan diakses, atau di tentukan tanpa memperhitungkan telah muncul sebelumnya atau belum, maka lebih dekat penyelesaian persoalan ini dikatakan dengan metode *bruteforce* dari pada DFS.

REFERENSI

[1] Rinaldi Munir, "Diktat Kuliah IF2251 Strategi Algoritmik",2007.

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 2010

ttd