Strategi Permainan Hidden Panel Dengan Algoritma Greedy

Muhammad Fikri. N - 13519069
Program Studi Teknik Informatika
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung, Jalan Ganesha 10 Bandung
E-mail (gmail) : 13519069@std.stei.itb.ac.id

***Abstrak*—Algoritma greedy banyak digunakan untuk menyelesaikan permasalahan. Algoritma greedy berusaha mengambil langkah terbaik yang mungkin diambil. Dengan menggunakan algoritma greedy diharapkan akan memperoleh hasil yang optimal. Pada makalah ini akan dibahas pengaplikasian algoritma greedy untuk menyelesaikan salah satu permainan dalam game Brain Training yaitu hidden pannel.**

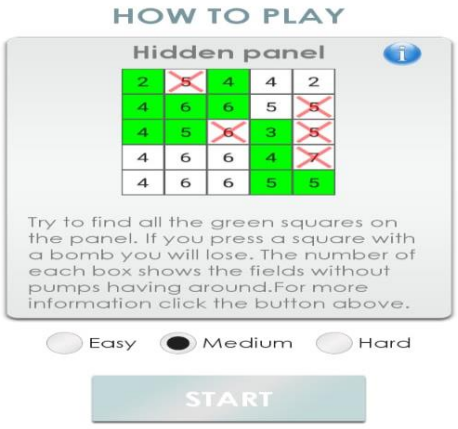
Kata Kunci—Brain; Training; Hidden; Panel; Greedy

I. PENDAHULUAN

Brain Training adalah sebuah permainan yang dikembangkan oleh untuk membantu pemain mengevalusi serta mengasah kemampuan dalam hal mengingat, menganalisis, ketepatan, ketajaman, dan persepsi. Dalam aplikasi ini, terdapat banyak kumpulan permainan yang menantang untuk dimainkan.

Gambar 1 : Tampilan game *brain training*.
Sumber : dokumen penulis

Salah satu permainan dalam aplikasi tersebut adalah *hidden panel*. Permainan ini tergolong dalam kelompok kategori *analyze*. Untuk memainkan permainan ini, pemain harus dapat menemukan kotak berwarna hijau tanpa membuka kotak yang berisi silang berwarna merah dengan melakukan analisis pada keseluruhan kotak. Jika pemain membuka kotak yang berisi silang berwarna merah tersebut maka permainan akan berakhir dan pemain gagal memenangkan permainan tersebut. Untuk menyelesaikan permainan ini, pemain harus membuka kotak yang tepat sehingga tidak memicu kotak yang berisi silang merah. Dalam menyelesaikan permainan *hidden panel* ini, pemain akan diberikan petunjuk angka-angka pada setiap kotak yang merepresentasikan banyaknya kotak hijau di sekitar kotak tersebut. Ketika permainan berlangsung, apabila pemain berhasil membuka kotak yang berwarna hijau ada kemungkinan untuk membuka kotak yang berisi silang merah di sekitar tersebut sehingga akan memberikan informasi tambahan bagi pemain untuk menemukan kotak-kotak hijau lainnya.



Gambar 2 : Panduan bermain *hidden panel*
Sumber : dokumen penulis

Pada awal permainan, pemain diberikan sekumpulan kotak dengan ukuran yang berbeda-beda tergantung tingkat kesulitan yang dipilih oleh pemain. Pada tingkat kesulitan *easy*, pemain diberikan 64 kotak dengan ukuran 8 x 8. Pada tingkat kesulitan *medium*, pemain diberikan sejumlah 72 kotak dengan ukuran 8 x 9. Pada tingkat kesulitan *hard*, pemain diberikan sebanyak 90 kotak dengan ukuran 9 x 10. Kotak-kotak tersebut jika dibuka maka ada dua kemungkinan yaitu kotak berwarna hijau (green) atau kotak yang berisi silang berwarna merah. Untuk selanjutnya, silang berwarna merah akan disebut sebagai *cross* dan kotak hijau akan disebut sebagai *green*.

Gambar 3 : Tampilan awal permainan *hidden panel*.
Sumber : dokumen penulis

hidden panel adalah permainan uang memanfaatkan analisis, logika, dan peluang (terutama ketika *current state* tidak memberikan keputusan yang pasti). Kita dapat menentukan pola-pola yang dapat membantu untuk memenangkan permainan ini serta menggunakan pendekatan algoritma greedy untuk mengambil langkah terbaik untuk setiap langkah yang dimainkan.

II. TEORI DASAR

A. *Algoritma Greedy*

Algoritma greedy merupakan metode yang paling populer dan sedehana untuk memecahkan persoalan optimasi. Persoalan optimasi adalah persoalan untuk mencari solusi optimal. Hanya ada dua macam persoalan optimasi yaitu maksimasi dan minimasi. Maksimasi yaitu persoalan untuk mencari nilai maksimum yang mungkin diperoleh untuk sebuah persoalan dan minimasi yaitu persoalan untuk mencari nilai minimum yang mungkin diperoleh untuk sebuah persoalan. Solusi dikatakan optimum adalah solusi yang bernilai maksimum untuk persoalan maksimasi atau bernilai minimum untuk persoalaan minimasi dari sekumpulan alternatif solusi yang mungkin.

Prinsip dari algoritma greedy adalah “take what you can get now ! ” . Prinsip ini berarti bahwa untuk setiap langkah yang dijalankan diambil pilihan yang terbaik yang dapat diambil saat itu tanpa mempertimbangkan konsekuensi kedepannya. Kemudian, dengan mengambil langkah yang terbaik di setiap langkah (optimum lokal) berharap akan berakhir dapat menghasilkan optimum global.

Elemen – elemen algoritma greedy :

1. Himpunan kandidat (C) : berisi kandidat yang akan dipilih pada setiap langkah.
2. Himpunan solusi (S) : berisi kandidat yang sudah dipilih sebagai solusi persoalan.
3. Fungsi solusi : menentukan apakah himpunan kandidat yang dipilih sudah memberikan solusi.
4. Fungsi seleksi (*selection function*) : memilih kandidat berdasarkan strategi greedy tertentu sehingga dapat menuju ke tujuan (maksimasi atau minimasi).
5. Fungsi kelayakan (*feasible*) : memeriksa apakah kandidat yang dipilih dapat dimasukkan ke dalam himpunanan solusi, maksudnya adalah dari himpunan kandidat yang dimiliki serta mempertimbangkan himpunan solusi yang sudah dimiliki tidak melanggar *constraint* yang ada. Jika kandidat tersebut layak maka akan dimasukkan ke himpunan solusi dan jika tidak layak maka tidak akan diperiksa lagi.
6. Fungsi obyektif : fungsi yang memaksimumkan atau meminimumkan nilai solusi.

Dengan menggunakan elemen-elemen di atas, maka dapat dikatakan bahwa, algoritma greedy melibatkan pencarian sebuah himpunan bagian S, dari himpunan kandidat C; yang dalam hal ini, S harus memenuhi beberapa kriteria yang ditentukan, yaitu S menyatakan suatu solusi dan S dioptimisasi oleh fungsi obyektif.

Skema umum algoritma greedy

```
function greedy(C:himpunan_kandidat) → himpunan_solusi

{ Mengembalikan solusi dari persoalan optimasi dengan algoritma greedy }

Deklarasi

   x : kandidat

   S : himpunan_solusi

Algoritma :

S ← {} { inisialisasi S dengan kosong }

   while (not SOLUSI(S)) and (C ≠ {} ) do

        x ← SELEKSI(C) { pilih sebuah kandidat dari C}
```

```
    C ← C – {x} { buang x dari C karena sudah dipilih }
    if LAYAK(S ∪ {x}) then { x memenuhi kelayakan untuk dimasukkan ke dalam himpunan solusi }
        S ← S ∪ {x} { masukkan x ke dalam himpunan solusi }
    endif
endwhile
{SOLUSI(S) or C = {} }
if SOLUSI(S) then { solusi sudah lengkap }
    return S
else
    write('tidak ada solusi')
endif
```

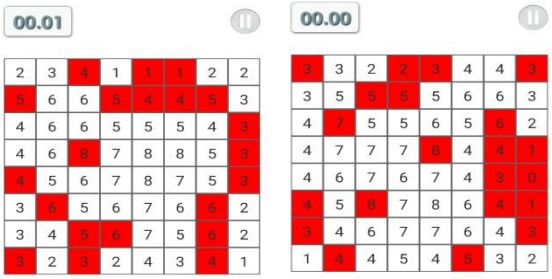
III. Strategi dan Analisis Dalam Permainan *Hidden Panel*

A. Kemunculan Cross

Pada permainan *hidden panel* terdapat tiga tingkat kesulitan yaitu *easy*, *medium*, dan *hard*. Tiap tingkat kesulitan memiliki perbedaan jumlah kotak, banyaknya *cross*, dan banyaknya *green*. Penulis melakukan pengujian apakah banyaknya *cross* dalam tingkat kesulitan yang sama akan berbeda atau tidak. Data pengujian dapat dilihat pada gambar di bawah ini namun untuk mengemat pemakaian ruang, hanya ditampilkan dua buah pengujian untuk tiap tingkat kesulitan.

Tingkat kesulitan :

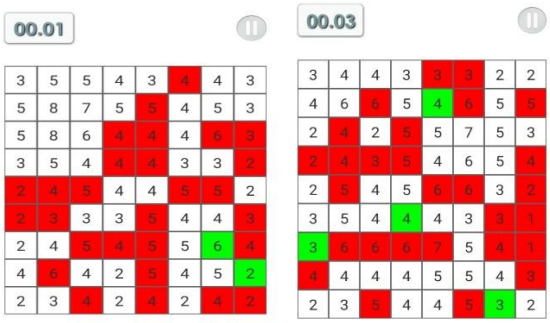
a) Easy



Gambar 8 : Data uji coba banyaknya *cross* pada tingkat *easy*
Sumber : dokumen penulis

Diperoleh bahwa pada tingkat kesulitan *easy* terdapat 21 *cross* dari total kotak sebanyak 64 kotak.

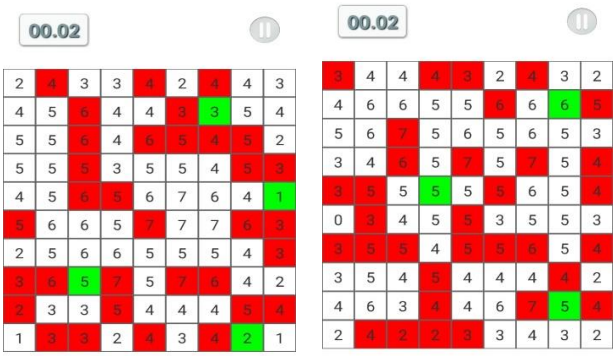
b) *Medium*



Gambar 9 : Data uji coba banyaknya *cross* pada tingkat *medium*
Sumber : dokumen penulis

Diperoleh bahwa pada tingkat kesulitan *easy* terdapat 28 *cross* dari total kotak sebanyak 72 kotak.

c) *Hard*



Gambar 10 : Data uji coba banyaknya *cross* pada tingkat *hard*
Sumber : dokumen penulis

Diperoleh bahwa pada tingkat kesulitan *easy* terdapat 33 *cross* dari total kotak sebanyak 90 kotak.

	Easy	Medium	Hard
Total Kotak	64	72	90
Frekuensi *cross*	21	28	33
Persentase banyaknya *cross*	32,8125 %	38,8888 %	36,6667 %

B. Pola - pola dalam *hidden panel*

Pada awal permainan diperlukan langkah awal untuk mulai membukat kotak – kotak yang belum diketahui. Dengan awalan yang baik akan membuat kotak – kotak semakin banyak yang terbuka dan kita dapat mengetahui letak *cross*.

A. Pola 8

Ketika kotak memiliki angka 8 maka dapat dipastikan semua kotak di sekitar kotak tersebut berisi *green*.

Gambar 11 : Contoh pola 8
Sumber : dokumen penulis

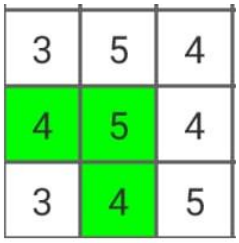
B. Pola 5

Ketika kotak yang memiliki angka 5 pada tepi kotak maka dapat dipastikan semua kotak di sekitar kotak tersebut berisi *green*.

Gambar 12 : Contoh pola 5
Sumber : dokumen penulis

C. Pola 3

Ketika kotak yang memiliki angka 5 pada pojok area permainan maka dapat dipastikan semua kotak di sekitar kotak tersebut berisi *green*.

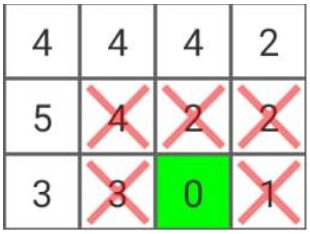


Gambar 13 : Contoh pola 3
Sumber : dokumen penulis

D. Pola 0

Ketika kotak memiliki angka 0 maka dapat dipastikan semua kotak di sekitar kotak tersebut adalah *cross* namun kotak belum tentu green. Hal yang dapat dipastikan adalah sekitar kotak 0 adalah *cross.*

.



Gambar 14 : Contoh pola 0
Sumber : dokumen penulis

C. Strategi dengan Menggunakan Pendekatan Algoritma Greedy

Dalam permainan hidden panel, algoritma greedy digunakan untuk menentukan langkah-langkah yang dilakukan ketika permainan sedang berlangsung sehingga dapat menemukan semua kotak berisi *green* tanpa menekan kotak yang berisi *cross.* Greedy yang dilakukan pada permainan ini adalah greedy by ObviousPanel. Artinya adalah melihat panel yang jelas kondisi state neighbor nya.

Elemen – elemen algoritma greedy yang diterapkan dalam permainan hidden panel adalah sebagai berikut :

1. Himpunan kandidat

   Himpunan kandidat nya adalah semua kotak / panel yang ada dalam permainana hidden panel.

2. Himpunan solusi

   Himpunan solusi nya adalah kotak/panel yang dapat dipastikan isinya (*green* atau cross).

3. Fungsi solusi

   Fungsi solusi nya adalah memeriksa apakah kotak yang dipilih berdasarkan kotak yang sudah pasti dapat diidentifikasi sekitarnya.

4. Fungsi seleksi (*selection function*)

   Fungsi seleksi nya adalah memilih kotak berdasarkan kotak yang sudah pasti dapat diidentifikasi sekitarnya.

5. Fungsi kelayakan (*feasible*)

   Fungsi kelayakan nya adalah memeriksa suatu kotak kemudian mempertimbangkan kotak tetangga nya yang sudah menjadi solusi (sudah diketahui *green* atau *cross*) serta nilai dari kotak tersebut yang menandakan banyaknya *g reen* di sekitar nya.

6. Fungsi obyektif

   memilih kotak yang jelas state dari tetangga nya atau pola -pola dalam hidden panel yang sudah dijelaskan pada bagian B.

```
// class Board

Attributes

baris,kolom,matrix,status

Operations

Board(baris,kolom) // constructor

NORTH(baris,kolom) // return matrix[baris-1][kolom] (North)

SOUTH(baris,kolom) // return south of panel

EAST(baris,kolom) // return east of panel

WEST(baris,kolom) // return west  of panel

NORTH_WEST(baris,kolom)  //  return  north west of panel

NORTH_EAST(baris,kolom)  //  return  north east of panel

SOUTH_WEST(baris,kolom)  //  return  south west of panel

SOUTH_EAST(baris,kolom)  //  return  south east of panel

CENTER(baris,kolom) // return the panel

display_matrix()  //  display  all  panel (value)

display_state() // display green panel (1) , cross panel (-1), has not been pressed (0)

get_neighbors(baris,kolom) // get adjacent of panel

is_final_board() // return true if not any state  =  0,  means  everything  has  been checked

is_init_board() // return true if all state = 0
```

```
// class Panel

Attributes

val,x,y,state

Operations

Panel(val) // constructor

setState(val) // set state to val {0 -> has not been pressed , 1 -> green panel , -1 -> cross panel}
```

```
// class GameStatus (enum)

InProgress

Completed
```

```
// class Solver
Attributes

board

Operations

Solver(baris,kolom) // constructor
solve() // to solve board
FirstMove() // initial move
obviousGreen()  //  panels  that  have  left neighbor  panels  that  haven't  been  pressed (state = 0) are equal to the value of those panels then set state of panel to 1
ObviousCross()  //  panels  that  have  left neighbor  panels  that  haven't  been  pressed (state = 0) are equal to (neighbor - value of those panels) then set state of panel to -1

procedure solve()

Kamus Lokal :
baris
kolom
mat = Board(baris,kolom)

   while mat.status = GameStatus.InProgress
      if (is_init_board) then
           FirstMove()

      obviousCross()
      obviousGreen()

   if (mat.status = GameStatus.Completed)
       display_state()
       write (‘Game Completed’)

procedure FirstMove()

{Untuk mengubah state menjadi green atau cross untuk  pola  –  pola  pada  hidden  panel  (refer bagian b)}

Kamus Data :

baris
kolom
mat = Board(baris,kolom)

Algoritma :

for i <- 0 to baris do
```

```
   for j <- 0 to kolom do
      if (i  = 0) then
          if (j  = 0) then
              if (mat.matriks[i][j].val = 3) //
3 pojok kiri atas
                  i = 0
                  j = 0
                  mat.EAST(i,j).setState(1)
                  mat.SOUTH(i,j).setState(1)
                  mat.SOUTH_EAST(i,j).setState(1)
          else if (j = mat.kolom - 1) then
              if (mat.matrix[i][j].val = 3)
                  mat.EAST(i,j).setState(1)
                  mat.NORTH(i,j).setState(1)
                  mat.NORTH_WEST(i,j).setState(1)
          else
              if (mat.matrix[i][j].val = 5) then
                  mat.WEST(i, j).setState(1)
                  mat.NORTH(i, j).setState(1)
                  mat.EAST(i, j).setState(1)
                  mat.NORTH_WEST(i,
j).setState(1)
                  mat.NORTH_EAST(i,
j).setState(1)
      else if (i = mat.baris - 1) then
          if (j = 0) then
              if mat.matrix[i][j].val = 3 then
                  mat.EAST(i,j).setState(1)
                  mat.NORTH(i, j).setState(1)
                  mat.NORTH_EAST(i,j).
   setState(1)
          else if (j = mat.kolom - 1) then
              if mat.matrix[i][j].val = 3 then
                  mat.WEST(i, j).setState(1)
                  mat.NORTH(i, j).setState(1)
                  mat.NORTH_WEST(i,
j).setState(1)
          else
              if (mat.matrix[i][j].val = 5)
   then
                  mat.WEST(i, j).setState(1)
                  mat.NORTH(i, j).setState(1)
                  mat.EAST(i, j).setState(1)
                  mat.NORTH_WEST(i,
j).setState(1)
                  mat.NORTH_EAST(i,
j).setState(1)
      else
          if (j = 0) then
              if mat.matrix[i][j].val = 5 then
                  mat.NORTH(i, j).setState(1)
                  mat.SOUTH(i,j).setState(1)
                  mat.EAST(i, j).setState(1)
                  mat.NORTH_EAST(i,
j).setState(1)
                  mat.SOUTH_EAST(i,
```

```
j).setState(1)
          else if (j = mat.kolom - 1) then
              if mat.matrix[i][j].val == 5
   then
                  mat.NORTH(i, j).setState(1)
                  mat.SOUTH(i, j).setState(1)
                  mat.WEST(i, j).setState(1)
                  mat.NORTH_WEST(i,
j).setState(1)
                  mat.SOUTH_WEST(i,
j).setState(1)

           else
              if  mat.matrix[i][j].val == 8
   then
                  mat.NORTH(i,j).setState(1)
                  mat.SOUTH(i,j).setState(1)
                  mat.WEST(i,j).setState(1)
                  mat.EAST(i,j).setState(1)
                  mat.NORTH_EAST(i,j).
   setState(1)
                  mat.NORTH_WEST(i,j).setState(1)
                  mat.SOUTH_EAST(i,j).setState(1)
                  mat.SOUTH_WEST(i,j).setState(1)
```

IV. Pembahasan

Pada sebuah permainan, pemain memiliki tiga tingkat kesulitan yaitu easy, medium, dan hard. Tiap tingkat kesulitan memiliki jumlah kotak dan jumlah cross yang berbeda-beda. Berdasarkan percobaan pada BAB 3, untuk tingkat kesulitan easy, ada 21 cross dari total kotak sebanyak 64 kotak. Untuk tingkat kesulitan medium, ada 28 cross dari total sebanyak 72 kotak. Untuk tingkat kesulitan hard, ada 33 cross dari total sebanyak 90 kotak.

Strategi greedy yang digunakan adalah greedy by ObviousPanel. Artinya adalah melihat panel yang jelas kondisi state neighbor nya berdasarkan nilai dari panel tersebut dan kondisi current state neighbor nya. Program menerima input file txt yang berisi angka-angka dari sebuah board permainan hidden panel.

Contoh isi file txt :

22342211
34665542
36665332
35445564
47643132
35434474
48765354
46554554
34442332

Kemudian, file txt tersebut dimasukkan ke atribut class Solver yaitu board. Nilai – nilai nya disimpan di tipe data matriks (array of array panel). Pada awal permainan, akan langsung dicari pola – pola hidden panel yang telah dijelaskan pada BAB 3 karena panel tersebut merupakan panel yang jelas state dari neighbor nya. Setelah itu, dicari panel yang obviousCross yaitu panel yang sudah dapat dipastikan state nya adalah cross. Setelah pencarian state yang cross, dilanjutkan dengan pencarian yang dapat dipastikan state nya adalah green. Selama Status dari game tersebut masih InProgress maka program akan terus berjalan hingga state nya berubah menjadi Completed atau Failed. Status Game Completed yaitu Ketika semua panel telah dipastikan state nya masing – masing. Setelah status game completed ditampilkan semua state panel.

V. KESIMPULAN

Algoritma greedy dapat digunakan untuk membantu strategi dalam menyelesaikan permainan hidden panel. Dengan adanya strategi tersebut, pemain yang memainkan permainan hidden panel dapat lebih jelas langkahnya dalam rangka untuk menyelesaikan permainan hidden panel sehingga semua kotak green diketahui tanpa membuka kotak cross.

VI. PENUTUP

Puji Syukur Alhamdulillah penulis ucapkan kepada Allah Swt.,yang telah memberikan nikmat ilmu, nikmat sehat secara jasmani,rohani dan nikmat lainnya yang tidak bisa disebutkan. Terimakasih penulis ucapkan juga kepada teman-teman seperjuangan dan orang tua yang selalu mendukung dalam penyelesaian makalah ini sehingga penulis dapat menyelesaikan makalah dengan tepat waktu. Terimakasih juga saya sampaikan kepada Dosen Strategi Algoritma K02, Nur Ulfa Maulidevi yang telah menyampaikan ilmu dan membimbing mahasiswa untuk menyelesaikan mata kuliah strategi algoritma.

VIDEO LINK AT YOUTUBE *(5)*

https://youtu.be/W8F1T-pz2Ow

REFERENCES

[1] Munir,R.2021.Algoritma Brute Force bagian 1. Diambil dari https://informatika.stei.itb.ac.id/~rinaldi.munir/Stmik/2020-2021/Algoritma-Brute-Force-(2021)-Bag1.pdf pada tanggal 11 Mei 2021

[2] Jones, Mathew.Solving Minesweeper with c# and LINQ," Diambil dari https://exceptionnotfound.net/solving-minesweeper-with-c-sharp-and-linq/ pada tanggal 10 Mei 20201

[3] Kemere, Chris F. PseudoCode class. Diambil dari https://www.researchgate.net/figure/Pseudocode-for-classes-in-TRADER-Version-10_fig2_3187951 pada tanggal 11 Mei 2021

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.

Payakumbuh, 11 Mei 2021

Muhammad Fikri. N 13519069