Penerapan Algoritma Brute Force dalam Penyelesaian Permainan Kartu SET

Ferawati Manurung 120140196
Program Studi Teknik Informatika
Institut Teknologi Sumatera, Jalan Terusan Ryacudu Lampung
E-mail : ferawati.120140196@student.itera.ac.id

***Abstrak*—Permainan kartu SET adalah salah satu permainan kartu yang sangat menguji kesabaran. Permainan kartu SET ini adalah permainan kartu yang belakangan ini sudah diimplementasikan ke dalam website setwithfriends.com . Permainan kartu SET ini bertujuan mencari 3 pasang kartu dengan 3 kesamaan atau 3 perbedaan dari 4 kriteria yang diberikan, yakni warna, bentuk, isi, dan angka. Memainkan kartu SET sangatlah membutuhkan kekuatan mental yang tinggi, bahkan sampai memenangkan America's Mensa Select pada 1991. Permainan kartu SET dapat dibuat mudah bila ada algoritma yang membantu dalam penyelesaian permainannya**

***Kata Kunci*—SET; Brute Force; Algoritma**

I. *PENDAHULUAN*

SET adalah permainan kartu dari negara Amerika Serikat yang ditemukan pada tahun 1991. Pada tahun yang sama, SET memenangkan nominasi dari *America's Mensa Select*. Namun, nyatanya permainan SET sangatlah sulit dimainkan. Kompleksitas dari permainan SET sendiri adalah NP-Complete. Hal ini menyebabkan permainan SET bila dilakukan hanya dengan kemampuan otak manusia, akan cukup relatif menantang. Bahkan *America's Mensa* yang merupakan kelompok manusia ber-IQ top 2% saja mengakui bahwasanya permainan ini sulit dimainkan.

Permainan SET sebenarnya memiliki aturan yang sangat simpel, yakni mencari set 3 kartu dari beberapa kartu yang ditampilkan, dengan semua syarat harus dipenuhi :

1. Terdiri dari 3 angka yang sama, atau dari 3 angka yang berbeda
2. Terdiri dari 3 warna yang sama, atau dari 3 warna yang berbeda
3. Terdiri dari 3 isian yang sama, atau dari 3 isian yang berbeda
4. Terdiri dari 3 bentuk yang sama, atau dari 3 bentuk yang berbeda.

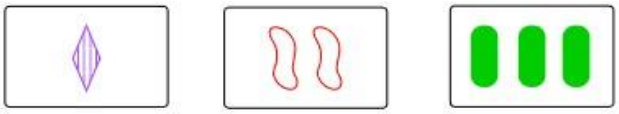
Aturan yang cukup simpel ini dapat dibuat lebih simpel dengan meninjau yang bukan set, yakni apapun yang dapat disusun dengan "dua kartu ____ dan satu kartu _____". Semisal "dua kartu wajik dan satu kartu bulat", bukan set.

Dengan cara yang sama, dapat dilakukan metode *Brute force* untuk menyelesaikan permainan dengan mudah.

II. *LANDASAN TEORI*

A. *SET*

SET merupakan permainan kartu yang dimainkan dengan dek khusus berjumlah 81 kartu, dengan 4 desain berbeda, yakni angka, isian, warna, dan bentuk.



Gambar 2.1 Ilustrasi SET

Sumber : https://homepages.warwick.ac.uk/staff/D.Maclagan/papers/set.pdf

Desain atau atribut tersebut masing-masing memiliki 3 jenis yang berbeda, yakni :

Angka : 1, 2, 3

Isian : Kosong, Arsir, Penuh

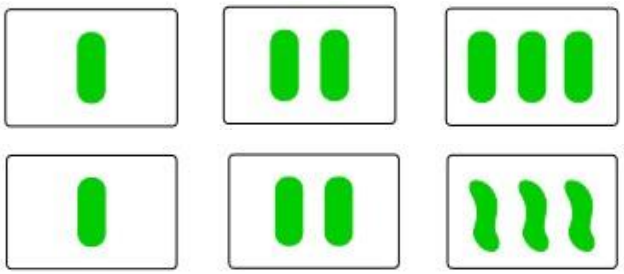
Warna : Merah, Hijau, Ungu

Bentuk : Oval, Wajik, dan Gelombang

Permainan SET sebenarnya memiliki aturan sebagai berikut:

1. Terdiri dari 3 angka yang sama, atau dari 3 angka yang berbeda
2. Terdiri dari 3 warna yang sama, atau dari 3 warna yang berbeda
3. Terdiri dari 3 isian yang sama, atau dari 3 isian yang berbeda
4. Terdiri dari 3 bentuk yang sama, atau dari 3 bentuk yang berbeda.

Aturan ini dapat dibuat lebih singkat dengan meninjau kartu yang bukan set, yakni apapun yang dapat disusun dengan "dua kartu ____ dan satu kartu _____". Semisal "dua kartu wajik dan satu kartu oval", bukan set.



Gambar 2.2 Contoh set dan bukan set

Sumber : https://homepages.warwick.ac.uk/staff/D.Maclagan/papers/set.pdf

Kartu yang akan ditampilkan di meja akan berjumlah 12 hingga kartu di dek habis. Namun bila kartu di meja tidak memungkinkan terjadinya set, maka akan diberi tambahan 3 kartu, hingga terjadi set. Pada dasarnya, bila kartu di dek berjumlah <20, maka terjamin ada minimal 1 set disitu. Bila kartu di dek sudah habis, permainan akan dilanjutkan hingga kartu di meja sudah tidak memungkinkan membentuk set atau kartu habis.

B. Brute Force

Brute force merupakan algoritma yang lempang (*straight forward)* dalam memecahkan suatu permasalahan.`Biasanya *brute force* memiliki beberapa pendekatan mendasar, yaitu sederhana, langsung, dengan cara yang jelas, dan cukup diselesaikan tanpa memerdulikan kompleksitasnya.

Terdapat beberapa karakteristik dari algoritma *brute force*, yaitu :

1. Algoritma *brute force* biasanya tidak 'cerdas' dan efektif karena memerlukan komputasi yang relatif banyak dan memakan waktu yang lama.
2. Baik digunakan dalam masalah berskala kecil
3. Hampir semua permasalahan dapat diselesaikan dengan *brute force*.
4. Biasa digunakan sebagai pembanding dari algoritma lain yang relatif lebih 'cerdas'.

Contoh implementasi dari algoritma *brute force* adalah metode penyusunan angka atau *sorting* menggunakan algoritma *bubble sort* yang mana akan secara lempang (*straight forward*) menukar angka yang berada di urutan yang tidak tepat. Hal ini menggunakan pendekatan yang secara simpel (kuli). Namun pada kenyataannya, hampir tidak ada permasalahan yang tidak dapat diselesaikan oleh algoritma *brute force* karena ke-*straight forward*-an ini.

C. Pencarian Beruntun

Pencarian beruntun merupakan salah satu permasalahan yang lumayan sering digunakan dalam visualisasi *brute force*. Permasalahan pencarian beruntun biasa meminta mengeluarkan indeks ditemukannya suatu item yang berada di dalam senarai.

Algoritma *brute force* yang sering digunakan adalah dengan membandingkan semua isi elemen senarai kepada nilai yang dicari, kemudian pencarian akan selesai ketika nilai tersebut ditemukan atau seluruh senarai telah diperiksa.

Pseudocode penyelesaian permasalah pencarian beruntun adalah sebagai berikut :

```
Procedure PencarianBeruntun(Input :a1, a2,…,an :integer, x :integer; Output idx : integer)
{ Mencari elemen bernilai x di dalam senarai a1,a2,…,an. Lokasi (indeks elemen) tempat x ditemukan diisi ke dalam idx. Jika x tidak ditemukan, maka idx diisi dengan 0.}

Deklarasi
   k : integer
Algoritma
   k ← 1
   while (k<n) and (ak ≠ x) do
       k ← k + 1
   endwhile
   {k = n or ak = x}
   if ak = x then
       idx ← k
   else

       idx ← 0
   endif
```

Jumlah operasi perbandingan elemen senarai maksimal sebanyak *n* kali. Kompleksitas waktu algoritma : *O(n)*.

III. PEMBAHASAN

Dalam permainan SET, karena ada 4 jenis perbedaan desain dengan masing-masing 3 desain, maka total kartu ada 3^4=81 kartu. Meninjau hal ini, langkah penyelesaian permainan SET dimulai dengan mendefinisikan kartu SET, kemudian mendefinisikan dek sebagai list of kartu, kemudian dari dek dikeluarkan kartu berjumlah 12 (atau lebih bila SET tidak ditemukan), dari kartu di meja, dilakukan *brute force* sedemikian rupa sehingga didapat set kartu yang tepat.

Dalam implementasinya, penulis membuat program *python* sederhana yang memuat permainan SET dan juga solusi dari permainan itu sendiri. Dengan asumsi dek yang dibuat benar, maka seharusnya akan ada tepat 27 subset ditemukan dari seluruh set yang memungkinkan.

A. Mendefinisikan Kartu dan Dek

Dalam mendefinisikan kartu dan dek, dapat dilakukan metode looping, sedemikian rupa sehingga untuk setiap elemen angka, setiap elemen warna, setiap elemen isi, dan setiap elemen bentuk, masing-masing muncul di kartu sebanyak 27 kali. Dari hal ini, dibuatlah prosedur mengisi dek sebagai berikut :

```
    def isiDek():
        kartu = ''
        angka   = ['S','D','T']
        warna   = ['H','M','U']
        isi     = ['A','K','P']
        bentuk  = ['O','W','G']
        for i in range(3):
            for j in range(3):
                for k in range(3):
                    for l in range(3):
                        kartu = angka[i] +
warna[j] + isi[k] + bentuk[l]
                        Dek.append(kartu)
```

Sedemikian rupa sehingga isi dek awal akan berisi 81 buah kartu dengan segala jenis elemen tepat muncul sekali yakni SHAO, SHAW, SHAG, dan seterusnya hingga TUPG. Masing-masing karakter menandakan karakteristik masing-masing, contohnya S menandakan satu, U menandakan ungu, P menandakan penuh, dan O menandakan Oval, sedemikian rupa sehingga SUPO berarti sebuah kartu berangka satu berwarna merah penuh dalam bentuk oval.

Gambar 3.1 Ilustrasi kartu SUPO

Sumber : Dokumentasi pribadi

B. Mengisi Meja dengan Kartu dari Dek

Dalam mengisi meja, dapat menggunakan *library* `random` dari bawaan bahasa pemrograman *python*. Ide yang diinginkan adalah dengan secara acak (*random*) memindahkan kartu dari dek ke meja sampai kartu di meja minimal berjumlah 12 dan kartu di dek belum habis. Bila kartu di meja melebihi 12 atau kartu di dek sudah habis, maka prosedur tidak melakukan apa-apa.

```
    def isiMeja():
        countMeja = len(Meja)
        countDek = len(Dek)
        while   (countMeja   <   12)   and
(countDek != 0):
            Meja.append(Dek.pop(
random.randint(0,(len(Dek)-1))))
            countMeja = len(Meja)
            countDek = len(Dek)
```

Demikian rupa sehingga akan terjadi looping terus menerus hingga kondisi tertentu terpenuhi, dalam kasus ini countMeja harus berjumlah minimal 12 atau countDek bernilai 0.

C. Mengecek Keberadaan Set dari Tiga Kartu

Dalam mengecek keberadaan Set dari tiga kartu, dapat dilakukan *brute force* dengan mengecek untuk setiap angka, warna, isi, dan bentuk, tidak ada elemen berjumlah 2. Yang dalam kata lain 1 dan 3 diterima. Penulis menggunakan *dictionary* bawaan python yang mana dapat digunakan untuk mensortir isi, warna, bentuk, maupun angka. Setelah itu menggunakan *set* dari python yang berarti tidak ada *item* yang sama disebutkan 2 kali.

```
    def isSet(tigaKartu):
        karakteristik = {
            "angka" : [],
            "warna" : [],
            "isi"   : [],
            "bentuk": []
        }
        for kartu in tigaKartu:

karakteristik["angka"].append(kartu[0])

karakteristik["warna"].append(kartu[1])

karakteristik["isi"].append(kartu[2])

karakteristik["bentuk"].append(kartu[3])
        uniqueness = {}
        for       key,       value       in
karakteristik.items():
            uniqueness[key]               =
list(set(value))
        for       key,       value       in
uniqueness.items():
            uniqueCount                   =
len(uniqueness[key])
            if uniqueCount == 2 :
                return False
        return True
```

Demikian rupa sehingga seluruh uniqueness dari masing-masing kriteria (warna, isi, angka, maupun bentuk) akan berjumlah antara 1 atau 3 bila 3 kartu tersebut merupakan set. Bila ada uniqueness dari salah satu kriteria yang berjumlah 2, maka secara otomatis 3 kartu tersebut bukanlah set. Keluaran dari fungsi isSet adalah True bilamana tiga kartu merupakan set, dan false bila tiga kartu bukan merupakan false.

D. Mengeluarkan Set dari Meja

Dari fungsi sebelumnya (isSet), kita dapat mengetahui apakah tiga buah kartu merupakan set. Meninjau hal ini, dapat dibuat sebuah fungsi yang akan mengecek apakah terdapat set dari meja.

Penulis menggunakan modul *combination* dari *library itertools* untuk mendapat kombinasi 3 kartu dari seluruh isi meja. Namun, tentu saja tanpa menggunakan modul tersebut, dapat menggunakan *looping* biasa. Setelah mendapatkan kombinasi 3 kartu, dilakukan pengecekan menggunakan pencarian beruntun (*brute force*) apakah 3 kartu tersebut merupakan set, dilakukan terus menerus hingga seluruh kombinasi habis atau hingga ditemukan set. Bila pada akhir looping tidak ditemukan set dari meja, dapat disimpulkan, dari kartu-kartu yang ada di meja, tidak ditemukan set 3 kartu yang memenuhi syarat set. Maka agar permainan berlanjut, ditambahkan 3 kartu dari dek ke meja agar ditemukan set (ada

kemungkinan set tetap tidak ketemu). Bila ditemukan set 3 kartu dari meja, maka set 3 kartu tersebut akan dikeluarkan dari meja sehingga kartu-kartu yang dipakai tersebut tidak dapat digunakan kembali.

```
   def findSet():
       threeCard                          =
list(combinations(Meja, 3))
       for i in threeCard:
           if isSet(i):
               for card in i:
                   Meja.remove(card)
               return i
       if(len(Dek)!=0):

Meja.append(Dek.pop(random.randint(0,(len
(Dek)-1))))

Meja.append(Dek.pop(random.randint(0,(len
(Dek)-1))))

Meja.append(Dek.pop(random.randint(0,(len
(Dek)-1))))
       return   "tidak   ada   set   yang
ditemukan, menambahkan kartu ..."
```

Keluaran dari fungsi findSet adalah set yang ditemukan, atau pesan bahwasanya tidak ada set ditemukan.

E. Memainkan SET

Dari seluruh fungsi-fungsi yang didefinisikan sebelumnya (isiDek, isiMeja, findSet, isSet), dapat dibuat sebuah permainan SET sederhana dengan CLI dari python. Sebelumnya, perlu didefinisikan variabel Dek dan Meja terlebih dahulu. Setelah itu pada permainan SET, lakukan isiDek dan isiMeja sedemikian rupa sehingga terbentuk awalan permainan berupa meja berisi 12 kartu dan dek yang berisi 81-12=69 kartu. Kemudian selama jumlah kartu di meja tidak sama dengan 3 (yang mana pasti sebuah set), tampilkan isi meja dan lakukan findSet. Jika tidak ditemukan set, dan sisa kartu di dek adalah 0, maka game selesai. Di tiap perulangan, lakukan isiMeja agar kartu terus diperbaharui sampai dek habis.

```
   def main():
       isiDek()
       isiMeja()
       while(len(Meja)!=3):
           input()
           print("sisa dek : ",len(Dek))
           print("isi meja :")
           for i in Meja:
               print(i)
           print(len(Meja))
           print("_______________")
           foundSet = findSet()
           if(foundSet == "tidak ada set
yang ditemukan, menambahkan kartu ..."
and len(Dek)==0):
               break
           else:
               print("set yang ditemukan
: ", foundSet)
           isiMeja()
       print("Game Selesai")
```

Sedemikian rupa sehingga, ketika kita memanggil fungsi main(), maka akan dilakukan looping pengecekan set, dan akan menampilkan set yang ditemukan bilamana ditemukan. Hal ini menjamin jumlah kartu di meja maupun kartu di dek, bila dibagi 3, akan menyisakan 0. Dan ketika kondisi dimana set tidak ditemukan dan dek berjumlah 0, maka akan menampilkan pesan "Game Selesai".

Ada keterjaminan bahwa set pasti ditemukan ketika jumlah kartu di meja dan jumlah kartu di dek dijumlahkan berjumlah > 20. Berdasarkan kriteria ini, validitas dari hasil permainan SET sederhana ini dapat teruji benar atau tidaknya.

IV. KESIMPULAN

Berdasarkan kepada seluruh fungsi yang dibuat, maka akan menjadi sebuah *source code* utuh yang dapat dijalankan dalam bahasa pemrograman *python*. Permainan akan menjalankan source code dan permainan akan siap untuk digunakan. Berdasarkan permainan ini, dapat disimpulkan bahwasanya algoritma menjalankan permainan secara benar dan tepat tujuan. Metode brute force yang digunakan-pun berjalan semestinya dan menghasilkan keluaran yang tepat guna.

UCAPAN TERIMA KASIH

Demikian implementasi sederhana brute force untuk tugas kali ini, semoga penjelasan yang saya sampaikan bisa mudah dicerna dan bisa digunakan untuk selanjutnya ataupun bisa dikembangkan lebih lanjut lagi. Mohon maaf bila ada salah kata atau kurang kata. Sekian Terimakasih.

REFERENSI

https://homepages.warwick.ac.uk/staff/D.Maclagan/papers/set.pdf, diakses pada 29 Maret 2022

https://informatika.stei.itb.ac.id/~rinaldi.munir/Stmik/2020-2021/Algoritma-Brute-Force-(2021)-Bag1.pdf, diakses pada 29 Maret 2022

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.

Lampung, 29 Maret 2022

Ferawati Manurung 120140196