Penggunaan Algoritma *Brute Force* dan *Greedy* dalam Permainan Domino

Hary Fernando

Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung
Jalan Ganesa 10
e-mail: if15113@students.if.itb.ac.id

ABSTRAK

Perkembangan ilmu pengetahuan dan teknologi telah menghantarkan manusia ke zaman yang serba moderen seperti saat sekarang ini. Penggunaan internet, *mobile phone*, teknologi digital dan lain sebagainya tidak dapat dihindari lagi. Walaupun demikian tetap saja algoritma dasar seperti *brute force* dan *greedy* merupakan algoritma yang hingga kini masih tetap digunakan untuk menyelesaikan persoalan-persoalan klasik ataupun baru yang belum terdapat penyelesaian masalahnya oleh algoritma-algoritma baru yang lebih tinggi tingkat kompleksitasnya. *Brute force* pasti dapat menyelesaikan masalah baik dalam sekala kecil maupun masalah besar, walaupn tidak semangkus algoritma-algoritma lain seperti BFS, DFS, *Backtracking* dan lain sebaginya yang memang lebih mengutamakan kemangkusan. Bersama dengan algoritma *greedy* yang juga tergolong algoritma dasar, kedua algoritma ini dapat digunakan untuk penyelesain persoalan permainan domino, salah satu permainan yang sudah lama dimainkan di negeri kita. Walaupun banyak algoritma yang dapat digunakan untuk menyelesaikan masalah ini, disini hanya akan digunakan algoritma *brute force* dan *greedy* untuk menyelesaikannya.

Kata kunci: *brute force*, *greedy*, domino, balak.

1. PENDAHULUAN

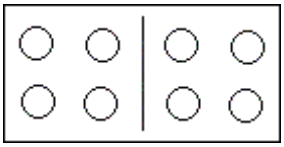
Domino adalah sebuah permanian yang menggunakan balok-balok yang pada satu sisinya terdapat tanda lubang /tanda yang menyatakan nilainya dari 1 sampai dengan 6 (Lihat gambar 1). Saat ini permainan domino dapat juga menggunakan kertas dan tanda yang berbentuk bulat digunakan untuk menyatakan nilai dari kartu domino tersebut. Jumlah kartu domino keseluruhan adalah 28 kartu. Permainan domino umunya dimainkan oleh 4 orang, namun dapat juga kurang atau lebih.

Perlu diperhatikan bahwa permainan domino identik sekali dengan perjudian dan hal-hal negatif lainnya.

Cara memainkan permainan domino sebenarnya banyak sekali, tergantung daerah dan keadaan masyarakatnya, tetapi permainan domino yang paling umum adalah permainan domino dengan meletakkan kartu domino yang bernilai paling kecil terlebih dahulu yaitu kartu kosong, kemudian diikuti oleh pemain lain dengan menyambung kartu domino dengan nilai yang bersesuaian sehingga membentuk suatu pola yang tidak terputus. Penurunan kartu dilakukan terus hingga kartu domino habis atau tidak ada lagi kartu yang dapat diturunkan.

Jika yang diturunkan adalah kartu balak, yaitu kartu yang kedua ujungnya bernilai sama, maka akan terbentuk percabangan dari kartu ini dan pemasukan kartu berikutnya dapat dimulai dari bercabangan ini dan bercabangan bertambah.

Dalam permainannya, mungkin saja terdapat cabang yang banyak sehingga nantinya pemain harus menentukan pilihan kartu mana yang dibuang.



Gambar 1. Contoh sebuah balok/kartu domino yang disebut juga balak 4

Permainan berakhir jika terdapat pemain yang dominonya telah habis atau semua pemain tidak dapat lagi menyambungkan balok-balok itu lagi. Pemain dengan nilai angka yang paling sedikit adalah pemenangnya.

Walaupun tidak menutup kemungkinan terdapat cara bermain yang lain, di sini hanya dibahas permainan domino standar yang meletakkan kartu domino hingga kartu dominonya habis seperti yang dijelaskan di atas.

2. METODE

Terdapat beberapa metode yang dapat dilakukan untuk bermain permainan domino ini. Di sini akah dibahas penggunaan algoritma dasar diantaranya adalah algoritma *brute force* dan algoritma *greedy* untuk menyelesaikannya.

Pada pembahasan ini tidak digunakan algoritma yang kompleks tapi hanya algoritma *brute force* dan *greedy*. Dikatakan demikian karena pada permainan ini masalah waktu tidak terlalu dipersoalkan. Kita dapat saja menghabiskan waktu asalakan masih mempunya kartu untuk dikeluarkan.

2.1 Algoritma *Brute Force*

Algoritma *brute force* adalah algoritma yang memecahkan masalah dengan sangat sederhana, langsung, dan dengan cara yang jelas (obvious way) [1]. Penyelesaian permasalahan permainan domino dengan menggunakan algoritma *brute force* akan menempatkan kartu domino disembarang tempat asalkan dia dapat masuk dan diletakkan ditempat yang sesuai. Ini dilakukan hingga kartu yang dimiliki habis.

Algoritma *brute force* adalah algoritma yang lempang atau apa adanya. Pemain hanya perlu meletakkan kartu domino sehingga dapat dikeluarkan tapi tidak peduli urutan atau hal-hal lain.

2.1.1 Penyelesaian dengan Algoritma *Brute Force*

Secara garis besar *pseudo code*-nya dapat dirumuskan:

```
  procedure BruteSolve(input k : integer)→
boolean
  { k adalah kartu yang dimasukakan sehingga
  didapatkan diakhir boolean, yaitu menang
  atau kalah.
  dapat dimasukan artinya : sisi yang ada
  dengan sisi yang akan dimasukan cocok, sama
  jumlahnya.
  kalah = fales;
  menang = true;}

  Deklarasi :
  menang, main : boolan

  Algoritma :
  menang ← false
  main ← true          {main = true artinya
                       permainan          belum
                       berakhir}
  while( DapatDimasukan(k) ) or (main = false)
  do
      k=k+1
      menang = true
      BruteSolve(k+1)
  endwhile
  return menang
```

sedangkan *procedure* DapatDimasukan adalah :

```
  procedure     DapatDimasukan(input     k     :
integer)→ boolean
  Deklarasi :
  sisiMasukan : integer
  sisiPenerima : integer
  masuk : boolan       {tidak dapat dimasukan}

  Algoritma :
  masuk ← false
  if (sisiMasukan = sisiPenerima) then
      masuk =  true
  else
      masuk = false
  endif
```

Dari kode di atas jelas bahwa kita hanya perlu memasukkan/menurunkan kartu yang sesuai, tapi tidak memperhitungkan nilai kartu yang kita keluarkan. Yang kita lakukan hanya mengeluarkan kartu kita hingga habis. Terlihat proses rekursif terjadi karena pemasukan kartu akan dilanjutkan terus hingga kartu kita habis atau permainan telah berakhir.

Jika bertemu dengan persimpangan maka keluarkan kartu apapun yang masih bisa dikeluarkan, tanpa memperhatikan nilai dari kartu tersebut. Demikian algoritma *brute force* bekerja pada permainan domino ini.

Walaupun terlihat biasa saja tapi algoritma ini dapat menyelesaikan permasalahan permainan domino ini dengan pasti. Meskipun kemangkusan algoritmanya masih harus dipertanyakan.

Kompeksitas waktu yang digunakan :

1. karena waktu bukanlah faktor penting dalam permainan ini maka kompleksitas waktu dapat dikesampingkan, kecuali terjadi situasi dimana waktu diperlukan agar permainan domino ini lebih menarik
2. jika memang diperlukan maka dari kode di atas dapat dihitung bahwa kompleksitas waktunya yang diperlukan adalah O(n).

2.2 Algoritma *Greedy*

Penyelesaian persoalan permainan domino dengan metoda *brute force* memang tidak mangkus tetapi dapat menyelesaikan. Satu lagi algoritma yang dapat menyelesaikannya dan lebih baik dari *brute force* adalah algoritma *greedy*.

Penerapan algoritma *greedy* pada permainan ini adalah mengeluarkan domino yang bernilai terbesar sehingga diakhir diharapkan solus optimal, yaitu domino kita habis, atau jika tidak, akan didapatkan paling sedikit nilai kartu domino yang tertinggal.

Masalah utama dalam penggunaan algoritma *greedy* ini adalah menentukan kartu domino mana yang akan dikeluarkan sehingga nilia kartu yang ditinggalkan sesedikit mungkin.

Dalam prakteknya persimpangan-persimpangan yang ada akan menentukan langkah *greedy* yang akan diambil selanjutnya.

Namun jika pada saat ini kita telah mengeluarkan kartu dengan nilai terbesar yang kita miliki, hal tersebut belum menjamin bahwa kartu-kartu selanjutnya yang akan dikeluarkan akan mudah/cepat habis. Bisa saja natinya pemain membutuhkan kartu-kartu besar sebagai penghubung kartu yang satu dengan kartu yang lain sehingga semua kartu dapat keluar.

2.1.1 Penyelesaian dengan Algoritma *Greedy*

Rumusan masalahnya dapat ditulis sebagai berikut:

```
  procedure GreedySolve(input k : integer)→
boolean
  { k adalah kartu yang dimasukakan sehingga
  didapatkan diakhir boolean, yaitu menang
  atau kalah.
  dapat dimasukan artinya : sisi yang ada
  dengan sisi yang akan dimasukan cocok, sama
  jumlahnya.
  kalah = fales;
  menang = true;}

  Deklarasi :
  menang, main : boolan

  Algoritma :
  menang ← false
  main ← true          {main = true artinya
                       permainan          belum
                       berakhir}
  while ((masih_ada_yang_paling_besar(k)) and
  ( DapatDimasukan(k) ) or (main = false) do
      k=k+1
      menang = true
      GreedySolve(k+1)
  endwhile
  return menang
```

```
  procedure     DapatDimasukan(input     k     :
integer)→ boolean
  Deklarasi :
  sisiMasukan : integer
  sisiPenerima : integer
  masuk : boolan       {tidak dapat dimasukan}

  Algoritma :
  masuk ← false
  if (sisiMasukan = sisiPenerima) then
      masuk =  true
  else
      masuk = false
  endif
       return masuk
```

```
  procedure  masih_ada_yang_paling_besar(input
k : integer)→ boolean
  Deklarasi :
  sisiMasukan : integer
  sisiPenerima : integer
  masuk : boolan       {tidak dapat dimasukan}
  i : integer
  Algoritma :
  masuk ← false
  for (i = 1) to 6 do
  if (sisiMasukan[i] > sisiMasukan[i+1]) then
      sisiPenerima = sisMasukan[i]
      masuk =  true
  else
      sisi Penerima = sisiMaasukan[i+1]
      masuk = true
  endif
       return masuk
```

Dari kode di atas dapat dilihat bahwa penggunaan algoritmanya hampir sama dengan yang digunakan pada algoritma *brute force*, tapi dengan tambahan bahwa yang aka ndimasukan terlebih dahulu adalah yang terbesar yang dimiliki saat itu, diliat dengan *procedure* masih_ada_yang_paling_besar.

Kompleksitas waktu yang diperlukan oleh permainan domino ini adalah sama dengan algoritma *brute force* O(n).

Namun demikian, algoritma *brute force* berbeda dengan algoritma *greedy* yang akan memilah blog mana yang paling besar akan dikeluarkan sehinga pemain akan memiliki sedikit jumlah nilai ditangannya. Tetapi kedua algoritma ini tentu belum mangkus mengingat jika kita telah memasukan kartu yang besar-besar terlebih dahulu akan tertinggal kartu yang kecil yang mana pihak lawan memiliki kartu-kartu besar sehingga kondisi akan bertambah sukar

3. Studi Kasus

Akan dilihat contoh aplikasi langsung dari permainan ini menggunakan kedua algoritma di atas.

Misalkan permainan telah berlangsung dan kartu domino yang kita miliki telah kita gunakan beberapa. maka dari kedua algoritma tersebut dapat dipreoleh dua kemungkian hasil yang berbeda.

3.1 Studi Kasus dengan Algoritma *Brute Force*

Misalkan kartuPenerima yang tersedia adalah 2, 4, dan 5. Kartu yang kita punya(kartuMasukan) ada yang bernilai 4, 5, 4, 3, dan 6. Dapat dilihat pada tabel di bawah ini :

Tabel 1 Tabel contoh penggunaan algoritma *brute force*

	kartuPenerima		
kartuMasukan	**2**	**4**	**5**
4		4	
5			5
4		4	
3			
6			

Maka, yang akan dikeluarkan dan yang mungkin adalah kartu yang bernilai 4 dan 5. karena metode yang digunakan adalah *brute force*, maka masukan kita tidak harus 4 atasu 5 yang dahulu. Tetapi tergantung pemain.

Demikian pula selanjutnya untuk persimpangan-persimpangan yang lain. Pemain hanya harus mengeluarkan sebuah kartu berapapun nilainya, asal cocok, maka permainan dilanjutkan hingga pemain telah kehabisan seluruh kartunya atau tidak ada lagi kartu yang dapat dikeluarkan.

3.2 Studi Kasus dengan Algoritma *Greedy*

Kalau diambil contoh pada studi kasus dengan algoritma *brute force* di atas, maka kartu yang akan dikeluarkan adalah pasti kartu bernilai 5 dan tidak memperhitungkan langkah selanjutnya..

Maka tabel yang bersesuaian dengan studi kasus ini adalah :

Tabel 2 Tabel contoh penggunaan algoritma *greedy*

	kartuPenerima		
kartuMasukan	**2**	**4**	**5**
4			
5			5
4			
3			
6			

Demikian juga selanjutnya, dengan contoh-contoh lain, selama algoritmanya adalah *greedy*, maka pemain akan mengeluarkan kartu terbesar yang mungkin masuk pada saat itu.

IV. KESIMPULAN

Penyelesaian permainan domino dapat dilakukan dengan penggunaan algoritma *brute foce* dan *greedy*.

Dalam permainan domino ini, baik algoritma *brute force* dan *greedy* tidak terlalu jauh perbedaannya, baik dari segi kemangkusan maupun waktu dikarenakan hal-hal sebagai berikut :

1. masalah waktu tidak terlalu diperhatikan.
2. kalaupun algoritma *greedy* lebih baik, jika di dapatkan bahwa kartu bernilai besar yang kita buang pertama menyebabkan kartu-kartu lain sukar untuk keluar, maka hasilnya mungkin lebih buruk dari algoritma *brute force* yang jelas-jelas pasti menghasilkan solusi permasalahan.

Walaupun terlihat sederhana, algoritma *brute force* pasti bisa menyelesaikan permasalahn permainan domino ini. Diharapkan kedepannya didapatkan algoritma-algoritma baru yang tentu saja lebih baik dari algoritma *brute force* ini untuk menyelesaikan persoalan-persoalan yang sepertinya terlihat sepele seperti permainan domino ini.

Jika persimpangan yang ada banyak maka algoritma *greedy* akan semakin banyak membandingkan. berbeda dengan algoritma bruteforce yang langsung menurunkan nilai kartu domino yang ada.

Dimungkinkan juga untuk menggabungkan algoritma *brute force* dan algoritma *greedy* menjadi suatu algoritma untuk menyelesaikan persoalan permainan domino ini. Tapi penulis tidak membahas mengenai penggabungan kedua algoritma ini untuk menyelesaikan persoalan ini.

REFERENSI

[1] Munir. Rinaldi, “IF2251 STRATEGI ALGRORITMIK Diktat Kuliah Strategi Algoritmik”, Departemen Teknik Informatika, 2005.