Penerapan Algoritma *Brute Force* pada Teka-teki *Magic Square 3 x 3*

Dzar Bela Hanifa 13515007
Teknik Informatika
Institut Teknologi Bandung
Bandung, Indonesia
13515007@std.stei.itb.ac.id

***Abstract*— Teka-teki *magic square* merupakan teka-teki matematika yang telah ada sejak zaman dahulu. Pada zaman dulu *magic square* yang merupakan kotak dengan angka yang memiliki jumlah sama pada sisi horizontal, vertikal, serta diagonalnya seringkali dianggap sakral dan dikaitkan dengan legenda-legenda mistis. Di zaman modern, teka-teki ini dianggap mampu melatih kemampuan matematika serta penalaran pada anak-anak. Di makalah ini, akan dibahas pengimplementasian bidang ilmu strategi algoritma dalam permainan teka-teki *magic square* berukuran 3 x 3. Algoritma yang digunakan adalah algoritma *brute force*. *Brute Force* merupakan salah satu ilmu yang dipelajari pada pelajaran strategi algoritma. Algoritma tersebut memiliki kelebihan yaitu implementasinya yang mudah. Namun, algoritma ini juga memiliki kelemahan seperti lamanya waktu yang dibutuhkan untuk menemukan solusi.**

Keywords—brute force; magic square; strategi algoritma, teka-teki

I. PENDAHULUAN

Teka-teki merupakan sebuah permainan ataupun masalah yang menguji pengetahuan seseorang. Untuk dapat menyelesaikan suatu teka-teki, kepingan-kepingan penyelesaian masalah harus disusun secara logis. Teka-teki terdiri dari berbagai bentuk yang cocok untuk bermacam-macam kalangan. Beberapa contoh teka-teki adalah teka-teki silang, teka-teki angka, serta teka-teki logika. Selain digunakan sebagai bentuk kegiatan rekreasi, teka-teki juga dapat muncul dari permasalahan matematis.

Walaupun sebenarnya sudah mucul sejak ribuan tahun sebelum masehi, di dunia modern kata teka-teki(dalam Bahasa Inggris *puzzle*) baru pertama kali muncul di akhir abad 16. Kata tersebut berasal dari kata lain yaitu *pusle* yang berarti membingungkan. Kata *puzzle* diartikan sebagai permainan yang dibuat untuk menguji tingkat kecerdikan seseorang di masa tersebut. Definisi tersebut masih digunakan hingga edisi 1989 dari *Oxford English Dictionary.*

Salah satu bentuk teka-teki yang sudah ada sejak masa lampau adalah *magic square*. Teka-teki ini muncul di China sejak 650 SM. Dalam berbagai peradaban, teka-teki ini terkadang dikaitkan dengan hal-hal mistis dan simbol seni. Akhir-akhir ini, teka-teki *magic square* memiliki berbagai varian. Beberapa di antaranya adalah menambahkan batasan untuk menyelesaikan masalah, menggunakan perkalian, menggunakan bentuk selain dua dimensi, serta mengganti angka menjadi bentuk dan pertambahan menjadi operasi geometri.

Gambar 1—*Magic Square* di Katedral Sagrada Familia, Barcelona
(sumber:https://www.theguardian.com/science/2011/apr/03/magic-squares-geomagic-lee-sallows)

II. DASAR TEORI

A. Algoritma Brute Force

Brute force adalah sebuah pendekatan yang lempang (*straightforward*) untuk memecahkan suatu masalah, biasanya didasarkan pada pernyataan masalah (*problem statement*) dan definisi konsep yang dilibatkan. Algoritma *brute force* memecahkan masalah dengan sangat sederhana, langsung dan dengan cara yang jelas (*obvious way*).

Algoritma *brute force* umumnya tidak "cerdas" dan tidak mangkus. Algoritma ini membutuhkan jumlah langkah yang besar dalam penyelesaiannya, terutama bila masalah yang dipecahkan berukuran besar(dalam hal ini ukuran masukannya). Terkadang, algoritma *brute force* disebut juga algoritma naif (*naïve algorithm*).

Untuk masalah yang berukuran kecil, kesederhanaan *brute force* biasanya lebih diperhitungkann daripada ketidakmangkusannya. Algoritma *brute force* sering digunakan sebagai basis apabila membandingkan beberapa algoritma lain yang lebih mangkus.

Meskipun *brute force* bukan merupakan teknik pemecahan masalah yang mangkus, namun teknik *brute force* dapat diterapkan pada sebagian besar masalah. Nyaris semua permasalahan yang terdapat di bidang ilmu komputer dapat diselesaikan oleh algoritma *brute force*. Bahkan ada beberapa masalah yang hanya mampu diselesaikan oleh algoritma *brute force*. Beberapa pekerjaan elementer di komputer dilakukan oleh *brute force*, seperti menghitung jumlah dari n bilangan, mencari elemen terbesar di dalam tabel, dan sebagainya.

Algoritma *brute force* seringkali lebih mudah diimplementasikan jika dibandingkan dengan algoritma lainnya yang lebih kompleks. Karena kesederhanaannya, kadang-kadang algoritma *brute force* lebih mangkus apabila ditinjau dari segi implementasi.

Algoritma *brute force* memilike beberapa kelebihan dan kekurangan jika dibandingkan dengan algoritma lainnya.

Beberapa kelebihan dari algoritma *brute force* adalah :

- Dapat digunakan untuk memecahkan sebagian besar masalah
- Sederhana dan mudah dimengerti
- Menghasilkan algoritma yang dapat digunakan untuk berbagai masalah penting seperti pencarian, pengurutan, pencocokan *string*, serta perkalian matriks.
- Menghasilkan algoritma *standard* untuk tugas-tugas komputasi seperti penjumlahan / perkalian n buah bilangan, menentukan elemen minimum atau maksimum di dalam table (list).

Kelemahan dari algoritma *brute force* adalah :

- Jarang menghasilkan algoritma yang mangkus.
- Beberapa algoritma *brute force* lambat
- Tidak sekreatif pemecahan masalah lainnya

Exhaustive Search adalah salah satu teknik pencarian solusi yang menggunakan algoritma *brute force* sebagai dasarnya. *Exhaustive Search* digunakan untuk menyelesaikan masalah yang terkait dengan pencarian elemen dengan sifat khusus. Sifat khusus tersebut pada umumnya adalah objek-objek kombinatorik seperti permutasi, kombinasi, atau himpunan bagian.

Langkah-langkah metode *exhaustive search* adalah :

- Enumerasi setiap solusi yang mungkin dengan cara yang sistematis.
- Evaluasi setiap kemungkinan solusi satu per satu. Solusi yang tidak layak dikeluarkan, simpan juga solusi terbaik yang ditemukan sampai dengan sejauh ini.
- Apabila pencarian telah usai, umumkan solusi terbaik yang telah ditemui.

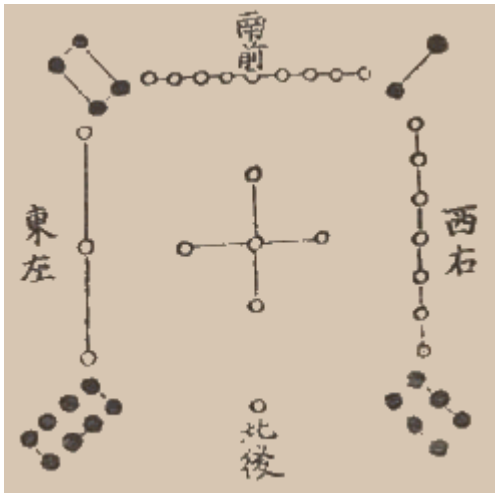
B. Teka-teki Magic Square

Magic Square merupakan matriks yang berisi bilangan bulat positif 1,2,…,n^2 yang berbeda satu sama lain dan disusun sedemikian rupa sehingga jumlah n bilangan di garis horizontal, vertikal, ataupun diagonal selalu sama. Jumlah n bilangan tersebut disebut juga sebagai *magic constant*.

$$M_2(n) = \frac{1}{n}\sum_{k=1}^{n^2} k = \frac{1}{2}n\left(n^2+1\right).$$

Magic square seringkali disebut juga sebagai *normal magic square*. Hal tersebut disebabkan ada juga *magic square* yang merupakan *non-normal magic square*. Yaitu *magic square* yang isinya tidak dibatasi oleh 1,2,…,n^2. Namun, terkadang *magic square* juga digunakan sebagai kata untuk menggambarkan segala jenis *magic square*, baik yang *normal* maupun *non-normal*.

Magic square, terutama yang berukuran 3 x 3 memiliki sejarah yang panjang. *Magic square* berukuran 3 x 3 sudah dikenal sejak zaman China kuno dengan sebutan Lo Shu. Menurut legenda di zaman China kuno, Lo Shu digunakan sebagai pola untuk mengatur sungai dan aliran air agar tidak terjadi banjir.



Gambar 2 Lo Shu Magic Square
(sumber:https://upload.wikimedia.org/wikipedia/commons/e/e2/Magic_square_Lo_Shu.png)

Teka-teki *magic square* merupakan teka-teki untuk melengkapi isi dari suatu *magic square* agar tercipta *magic square* dengan *magic constant* tertentu. Pengisian angka tersebut memiliki berbagai syarat misalnya angka yang diisikan haruslah unik serta jumlah angka horizontal, vertikal, serta diagonal haruslah sama. Berikut ini adalah contoh teka-teki *magic square* 3 x 3.

2	9	4
	5	

Gambar 3 Teka-teki *magic square* 3 x 3
(sumber:http://www.dr-mikes-math-games-for-kids.com/3x3-magic-square.html)

Pada contoh teka-teki tersebut, kotak-kotak yang kosong di *magic square* tersebut harus diisi dengan angka-angka agar jumlah bilangan di setiap sisi vertikal, horizontal, serta diagonal adalah 15.

III. PENERAPAN ALGORITMA BRUTE FORCE

Untuk mendapatkan angka-angka yang cocok untuk setiap kotak di dalam *magic square*, dilakukan implementasi teknik *Exhaustive Search*. Dalam teka-teki *magic square* yang digunakan sebagai implementasi, nilai *magic constant* serta tiga kotak yang terdapat di *magic square* akan diberi tahu. Oleh karena itu akan dicari tahu nilai-nilai angka yang terdapat di keenam kotak *magic square* sisanya.

2	a	4
c	5	b
d	e	f

Gambar 4 Contoh teka-teki magic square
(sumber:dokumen pribadi)

Contoh teka-teki *magic square* yang akan dipecahkan oleh algoritma *brute force* adalah yang terdapat di gambar 4. *Magic square* tersebut memiliki *magic constant* 15. Hal ini berarti jumlah setiap kolom, baris, serta diagonal yang terdapat di dalam kotak tersebut haruslah bernilai 15. Total ada 8 operasi matematika yang harus dicek.

```
baris 1 : 2 + a + 4 = 15
baris 2 : c + 5 + b = 15
baris 3 : d + e + f = 15
kolom 1 : 2 + c + d = 15
kolom 2 : a + 5 + e = 15
kolom 3 : 4 + b + f = 15
diagonal 1 : 2 + 5 + f = 15
diagonal 2 : 4 + 5 + d = 15
```

Berikut ini adalah langkah-langkah pemecahan masalah yang harus dilakukan untuk memecahkan teka-teki *magic square* tersebut.

1. Identifikasi operasi penjumlahan yang harus dicek dari *magic square* tersebut.
2. Identifikasi operasi penjumlahan mana saja yang sudah memiliki konstanta.
3. Tentukan nilai maksimum bilangan yang dimungkinkan sebagai isi dari *magic square*. Nilai tersebut merupakan nilai dari *magic constant* – 3. Hal ini disebabkan bilangan minimum yang mengisi *magic square* adalah 1. Oleh karena itu, bilangan terbesar yang dimungkinkan untuk mengisi *magic square* tersebut adalah :
    - N max = *magic constant* – dua bilangan terkecil yang masih bisa digunakan untuk mengisi *magic square*
    - N max = *magic constant* – (1 + 2)
    - N max = *magic constant* – 3
4. Coba semua kombinasi angka yang mungkin dari 1 hingga *magic constant* – 3 untuk mengisi tiap kotak yang ada (kotak a,b,c,d,e,f). Lakukan pengujian untuk setiap operasi penjumlahan yang ada. Utamakan operasi yang telah memiliki nilai konstanta terlebih dahulu agar proses ini lebih cepat dilakukan

Untuk contoh kasus yang sesuai dengan gambar 4, langkah-langkah pemecahan masalah adalah sebagai berikut :

1. Operasi matematika yang telah memiliki dua konstanta adalah operasi pada baris 1 (2 + a + 4 = 15). Cari nilai a yang dapat digunakan untuk memecahkan operasi ini. Lakukan pengujian mulai dari 1 hingga angka *magic constant* - 3. Nilai tersebut adalah 9. Gunakan nilai ini sebagai nilai a.
2. Karena nilai a sudah diketahui, maka operasi penjumlahan apda kolom 2 sudah memiliki 2 konstanta (9 + 5 + e = 25). Lakukan pengujian mulai dari 1 hingga angka *magic constant* - 3. Nilai tersebut adalah 1. Gunakan nilai ini sebagai nilai e.
3. Tidak ada lagi operasi penjumlahan yang memiliki dua konstanta. Lakukan pengujian untuk operasi penjumlahan yang memiliki satu konstanta.
4. Ulangi langkah ketiga hingga semua nilai pada *magic square* berhasil diidentifikasi.

Berikut ini merupakan hasil penerapan algoritma *brute force* pada teka-teki yang terdapat di gambar 4.

2	9	4
7	5	3
6	1	8

Gambar 5 Hasil Penerapan Algoritma *Brute Force*
(sumber:dokumen pribadi)

IV. Analisis dan Pengujian Program

A. *Pengujian*

Testcase 1

Testcase 1 merupakan *magic square* yang sama dengan *magic square* pada gambar 4. Tiga angka konstanta adalah 2, 4, serta 5. *Magic constant* adalah 15. Program mampu menghasilkan solusi untuk *magic square* ini dengan baik. Untuk menemukan solusi pada *testcase* ini dibutuhkan waktu 7.859 detik.

```
Masukkan tiga konstanta serta posisinya di dalam magic square
Konstanta 1
Baris : 1
Kolom : 1
Nilai : 2
Konstanta 2
Baris : 1
Kolom : 3
Nilai : 4
Konstanta 3
Baris : 2
Kolom : 2
Nilai : 5
Masukkan magic constant
15

-----------------------------------
Magic Square 3 * 3
  |  2  |    |  -99  |    |  4  |
-----------------------------------
  |  -99  |    |  5  |    |  -99  |
-----------------------------------
  |  -99  |    |  -99  |    |  -99  |
-----------------------------------

-----------------------------------
Solusinya adalah
  |  2  |    |  9  |    |  4  |
-----------------------------------
  |  7  |    |  5  |    |  3  |
-----------------------------------
  |  6  |    |  1  |    |  8  |
-----------------------------------
Waktu yang dibutuhkan untuk mencari solusi adalah : 7.8590000000
```

Gambar 6 Pengujian Testcase 1
(sumber:dokumen pribadi)

Testcase 2

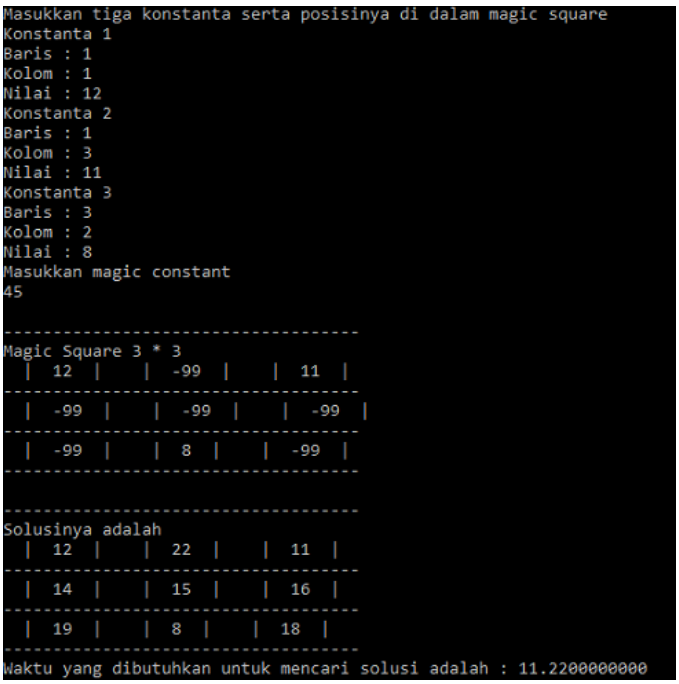
Tiga angka konstanta pada testcase 2 adalah 17, 7, serta 4. *Magic constant* adalah 60. Program mampu menghasilkan solusi untuk *magic square* ini dengan baik. Untuk menemukan solusi pada *testcase* ini dibutuhkan waktu 17.9 detik

```
Masukkan tiga konstanta serta posisinya di dalam magic square
Konstanta 1
Baris : 1
Kolom : 1
Nilai : 17
Konstanta 2
Baris : 1
Kolom : 3
Nilai : 7
Konstanta 3
Baris : 3
Kolom : 2
Nilai : 4
Masukkan magic constant
60

-----------------------------------
Magic Square 3 * 3
  |  17  |    |  -99  |    |  7  |
-----------------------------------
  |  -99  |    |  -99  |    |  -99  |
-----------------------------------
  |  -99  |    |  4  |    |  -99  |
-----------------------------------

-----------------------------------
Solusinya adalah
  |  17  |    |  36  |    |  7  |
-----------------------------------
  |  10  |    |  20  |    |  30  |
-----------------------------------
  |  33  |    |  4  |    |  23  |
-----------------------------------
Waktu yang dibutuhkan untuk mencari solusi adalah : 17.9820000000
```

Gambar 7 Pengujian Testcase 2
(sumber:dokumen pribadi)

Testcase 3

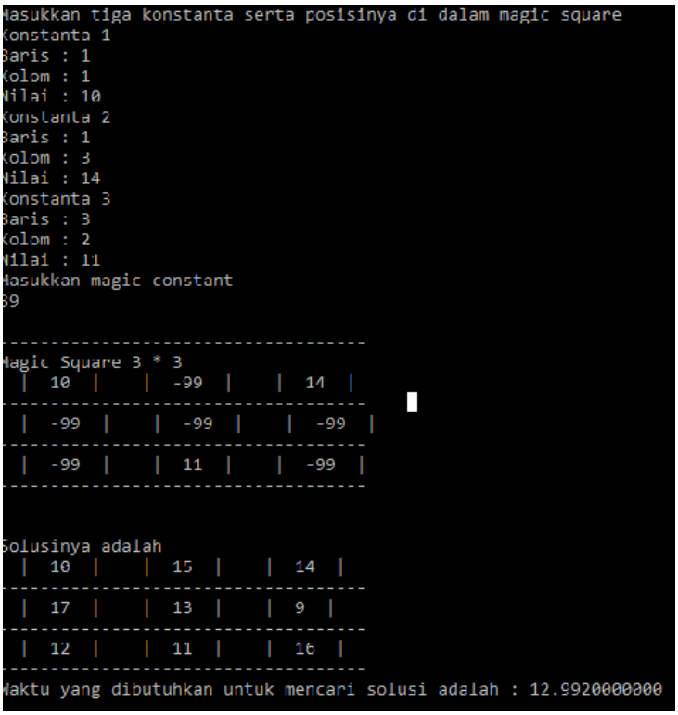
Tiga angka konstanta pada testcase 3 adalah 12, 11, serta 8. *Magic constant* adalah 45. Program mampu menghasilkan solusi untuk *magic square* ini dengan baik. Untuk menemukan solusi pada *testcase* ini dibutuhkan waktu 11.2 detik.



Gambar 8 Pengujian Testcase 3
(sumber : dokumen pribadi)

Testcase 4

Tiga angka konstanta pada testcase 4 adalah 10, 14, serta 11. *Magic constant* adalah 39. Program mampu menghasilkan solusi untuk *magic square* ini dengan baik. Untuk menemukan solusi pada *testcase* ini dibutuhkan waktu 11.2 detik.



B. Analisis

Dari hasil pengujian dapat disimpulkan bahwa algoritma *brute force* berhasil mendapatkan hasil yang akurat untuk setiap kasus uji. Waktu yang diperlukan untuk mencari solusi tergantung dari nilai *magic constant* semakin tinggi nilai *magic constant* maka akan semakin lama waktu program untuk menemukan solusi. Namun, karena *magic square* berukuran cukup kecil 3 x 3 waktu yang diperlukan tidak terlalu lama. Untuk menemukan setiap hasil pada operasi maka diperlukan operasi pengecekan sebanyak

(magic constant - 2) * (magic constant -2) *(magic constant – 2)

Oleh karena terdapat delapan operasi penjumlahan, maka pengecekan akan dilakukan sebanyak

8 * (magic constant – 2) * (magic constant -2) * (magic constant - 2)

Tentunya algoritma lainnya seperti *dynamic programming* akan mampu menyelesaikan teka-teki ini dengan waktu yang lebih singkat serta operasi yang lebih sedikit. Namun kelebihan penerapan algoritma *brute force* adalah implementasinya yang lebih mangkus serta mudah apabila dibandingkan dengan algoritma – algoritma lainnya.

V. KESIMPULAN

Algoritma Brute Force mampu menyelesaikan *magic square* berukuran 3 x 3 dengan waktu yang relatif cepat serta hasil yang akurat. Pada umumnya algoritma Brute Force tidak efektif dan efisien, akan tetapi karena kompleksitas masalah pengisian *magic square* tidak terlalu rumit maka pencarian solusi dapat dilakukan dengan waktu yang tak terlalu lama. Hal lain yang dapat ditambahkan untuk mempercepat *running time* program adalah menambahkan batasan-batasan *(pruning*) agar algoritma dapat berjalan lebih cepat lagi. Algoritma lain yang sebenarnya dapat diterapkan di dalam persoalan *magic square* adalah *dynamic programming*. Akan tetapi implementasi algoritma *brute force* lebih sederhana dibandingkan algoritma-algoritma lainnya

VI. UCAPAN TERIMA KASIH

Penulis mengucapkan terimakasih kepada Tuhan Yang Maha Esa atas berkat sehingga penulis dapat menyelesaikan makalah ini dengan tepat waktu. Penulis juga berterima kasih kepada Dr. Masayu Leylia Khodra ST,MT Dr. Nur Ulfa Maulidevi, S.T., M.Sc untuk bimbingannya selama mengikuti mata kuliah Strategi Algoritma sehingga penulis dapat menyelesaikan makalah ini .Selain itu, penulis juga tidak lupa berterima kasih kepada orangtua serta seluruh orang terdekat yang memberikan bantuan selama penulisan makalah ini.

VII. REFERENSI

[1] Munir, Rinaldi. 2009. Diktat Kuliah Strategi Algoritmik IF2251 Strategi Algoritmik. Departemen Teknik Informatika ITB.

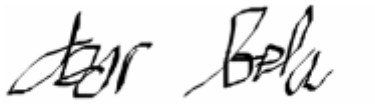
[2]. http://mathworld.wolfram.com/MagicSquare.html, diakses pada tanggal 17 Mei 2017, pukul 21.00

[3]. http://jeff560.tripod.com/m.html, diakses pada tanggal 18 Mei 2017, pukul 08.00

VIII. 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, 19 Mei 2017



Dzar Bela Hanifa / 13515007