Penerapan Algoritma Brute Force pada Permainan Tug of War

Vionie Novencia Thanggestyo - 13520006
Program Studi Teknik Informatika
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung, Jalan Ganesha 10 Bandung
E-mail (gmail): 13520006@std.stei.itb.ac.id

***Abstract*—Tug of War atau dalam bahasa indonesia tarik tambang adalah jenis permainan tradisional sederhana yang mengandalkan kekuatan dan kekompakan tim, dan menggunakan tali tambang sebagai alat permainannya. Di Indonesia, permainan ini sering dilombakan pada perayaan tertentu, salah satunya pada saat merayakan Hari Kemerdekaan Indonesia. Pada permainan ini, pemain dibagi menjadi dua tim. Masing-masing tim akan bekerja sama menarik tali tambang ke arah yang berlawanan. Tim yang berhasil menarik lawan ke sisinya dinyatakan sebagai pemenang. Pada makalah ini, konsep tarik tambang diadaptasi untuk membagi sebuah array of integer menjadi dua bagian, dengan jumlah nilai di tiap bagiannya kira-kira sama besar. Pembagian algoritma mejadi dua bagian akan dilakukan dengan menggunkan algoritma Brute Force. Algoritma Brute Force adalah pendekatan yang lempang (sraightforward) untuk memecahkan suatu persoalan. Pendekatan ini biasanya didasarkan pada pernyataan pada persoalan (problem statement), dan definisi konsep yang dilibatkan**

Kata kunci—Tug of War, Tarik Tambang, Brute Force

I. PENDAHULUAN

Gambar 1 Kegiatan Tarik Tambang
Sumber : https://kumparan.com/kabar-harian/permainan-tarik-tambang-cara-memainkan-dan-aturan-permainannya-1wTVZL8WAzg/4

Tug of War atau Tarik tambang adalah permainan yang umumnya dimainkan pada perayaan-perayaan tertentu seperti perayaan Hari Kemenrdekanaan Indonesia. Tarik tambang adalah permainan yang dilakukan secara beregu, yang mengandalkan kekompakan dan kekuatan tim. Tarik tambang berasal dari Yunani Kuno, sekitar tahun 500 SM. Tarik tambang biasanya dilakukan para atlet sebagai olahraga yang kompetitif. Terkadang tarik tambang juga dilakukan untuk latihan fisik tubuh sebelum melakukan olahraga lain. Tarik tambang dimainkan dengan cara tiap regu menarik tali tambang ke arah yang berlawanan. Regu yang berhasil menarik tim lawan ke arahnya dinyatakan sebagai pemenang.

Persoalan pada makalah ini diadaptasi dari konsep tarik tambang. Diberikan sebuah array of integer, array tersebut akan dibagi menjadi dua bagian yang apabila dijumlahkan, kedua bagian tersebut akan memiliki jumlah yang kira-kira sama besar. Pembagian array menjadi dua bagian dengan jumlah sama besar akan dilakukan dengan menggunkan algoritma Brute Force.

II. TEORI DASAR

A. Algoritma Brute Force

1. Definisi Algoritma Brute Force

Algoritma Brute Force adalah pendekatan yang lempang (straightforward) untuk memecahkan suatu persoalan. Algoritma ini biasanya didasarkan pada pernyataan pada persoalan (problem statement) dan definisi konsep yang dilibatkan. Algoritma ini memecahkan persoalan dengan sangat sederhana, langsung, dan jelas (obvious way).

2. Karakteristik Algoritma Brute Force

Karakteristik Algoritma Brute Force adalah sebagai berikut

1. Algoritma brute force umumnya tidak “cerdas” dan tidak mangkus, karena algoritma ini membutuhkan jumlah langkah yang besar dalam penyelesaiannya. Kata “force” mengindikasikan “tenaga” ketimbang “otak”. Algoritma brute force kadang-kadang juga disebut algoritma naif (naïve algorithm)

2. Algoritma brute force lebih cocok untuk persoalan yang berukuran kecil, sederhana dan implementasinya mudah. Algoritma brute force sering digunkan sebagai basis pembanding dengan algoritma yang lebih mangkus

3. *Kelebihan dan Kekurangan Algoritma Brute Force*

Kelebihan dari algoritma brute force adalah sebagai berikut:

a. Metode brute force dapat digunakan untuk memecahkan hampir sebagian besar masalah (wide applicability)

b. Metode brute force sederhana dan mudah dimengerti

c. Metode brute force menghasilkan algoritma yang layak untuk beberapa masalah penting seperti pencarian, pengurutan, pencocokan string, perkalian matriks.

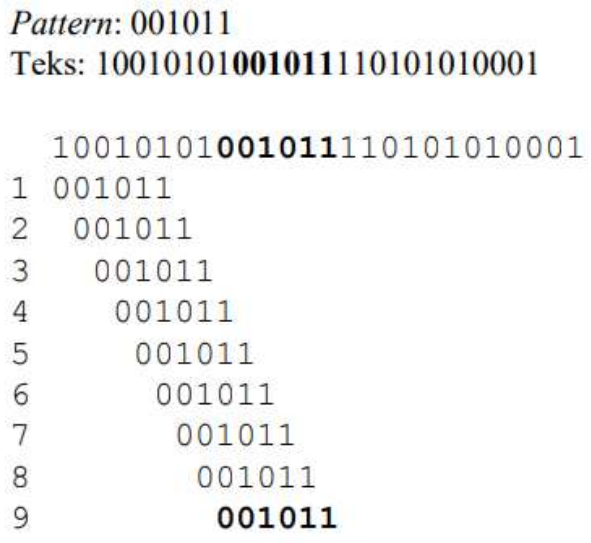
d. Metode brute force menghasilkan algoritma baku (standard) untuk tugas-tugas komputasi seperti penjumlahan/perkalian n buah bilangan, menentukan elemen minimum atau maksimum di dalam tabel (list).

Berikut ini adalah kekurangan dari algoritma brute force:

a. Metode brute force jarang menghasilkan algoritma yang mangkus

b. Beberapa algoritma brute force lambat sehingga tidak dapat diterima

c. Algoritma brute force tidak sekonstruktif/sekreatif yeknik pemecahan masalah lainnya



Gambar 2 Contoh brute force pada Pattern Matching

Sumber: https://informatika.stei.itb.ac.id/~rinaldi.munir/Stmik/2017-2018/Algoritma-Brute-Force-(2016).pdf

Dibawah ini merupakan contoh notasi algoritmik dengan algoritma brute force

```
procedure PencocokanString(input P : string, T : string,
 n, m : integer, output idx : integer)
{ Masukan: pattern P yang panjangnya m dan teks T yang
 panjangnya n. Teks T direpresentasika sebagai string
(array of character)
Keluaran: lokasi awal kecocokan (idx)
}

 Deklarasi
  i : integer
  ketemu : boolean
Algoritma:
  i ← 0
  ketemu ← false
  while (i ≤ n-m) and (not ketemu) do
     j ← 1
     while (j ← m) and (Pj = Ti+j ) do
           j ← j+1
     endwhile
     { j > m or Pj ≠ Ti+j }
     if j = m then { kecocokan string ditemukan }
           ketemu ← true
     else
           i ← i+1 {geser pattern satu karakter ke kanan teks }
     endif
  endfor
  { i > n - m or ketemu }
  if ketemu then
     idx ← i+1
```

```
else
   idx ← -1
endif
```

Meskipun bukan metode yang mangkus, hampir semua persoalan dapat diselesaikan dengan algoritma brute force. Sulit menemukan persoalan yang tidak dapat diselesaikan dengan algoritma brute force. Bahkan, pada beberapa persoalan hanya dapat diselesaikan dengan menggunakan algoritma brute force.

B. *Exhaustive Search*

Exhaustive search adalah teknik pencarian solusi secara brute force untuk persoalan-persoalan masalah kombinatorik. Biasanya diantara objek-objek kombinatorik seperti permutasi, kombinasi, atau himpunan bagian dari sebuah himpunan. Semakin banyak objek, semakin banyk pula kemungkinan solusinya. Walaupun solusi yang ditemukan dari exhaustive search pasti solusi terbaik, kompleksitas algoritma exhaustive search masih exponensial, sehingga algoritma ini cenderung dihindari.

Algoritma exhaustive search dapat diperbaiki kinerjanya sehingga tidak perlu melakukan pencarian terhadap semua kemungkinan solusi. Salah satu cara untuk mengoptimalkan pencarian solusi pada exhaustive search dapat digunakan teknik heuristik. Teknik ini mengeliminasi kemungkinan solusi yang tidak mungkin menjadi solusi terbaik sehingga pencarian yang dilakukan menjadi lebih sedikit.

Langkah-langkah metode exhaustive search sebagai berikut:

1. Enumerasi (list) setiap solusi yang mungkin dengan cara yang sistematis.
2. Evaluasi setiap kemungkinan solusi satu per satu, simpan solusi terbaik yang ditemukan sampai sejauh ini (the best solusi found so far).
3. Bila pencarian berakhir, umumkan solusi terbaik (the winner)

Meskipun exhaustive search secara teoritis menghasilkan solusi, namun waktu atau sumberdaya yang dibutuhkan dalam pencarian solusinya sangat besar.

C. *Heuristik*

Teknik heuristik digunakan untuk mengeliminasi beberapa kemungkinan solusi tanpa harus mengeksplorasinya secara penuh keseluruhan solusi. Selain itu, teknik heuristik juga membantu memutuskan kemungkinan solusi mana yang pertama kali perlu dievaluasi. Heuristik berbeda dari algoritma karena heuristik berlaku sebagai panduan (guideline), sedangkan algoritma adalah urutan langkah-langkah penyelesaian.

Heuristik mungkin tidak selalu memberikan hasil yang diinginkan, tetapi secara ekstrim ia bernilai pada pemecahan masalah. Heuristik yang bagus dapat secara dramatis mengurangi waktu yang dibutuhkan untuk memecahkan masalah dengan cara mengelimininasi kebutuhan untuk mempertimbangkan kemungkinan solusi yang tidak perlu. Heuristik tidak menjamin selalu dapat memecahkan masalah, tetapi seringkali memecahkan masalah dengan cukup baik untuk kebanyakan masalah, dan seringkali pula lebih cepat daripada pencarian solusi secara lengkap.

Contoh penggunaan heuristik untuk mempercepat algoritma exhaustive search ada pada persoalan anagram. Anagram adalah penukaran huruf dalam sebuah kata atau kalimat sehingga kata atau kalimat yang baru mempunyai arti lain. Contoh anagram misalnya:

- lived -> devil
- tea -> eat
- charm -> march

Bila diselesaikan secara exhaustive search, kita harus mencari semua permutasi huruf-huruf pembentuk kata atau kalimat, lalu memerika apakah kata atau kalimat yang terbentuk mengandung arti. Teknik heuristik dapat digunakan untuk mengurangi jumlah pencarian solusi. Salah satu teknik heuristik yang digunakan misalnya membuat aturan bahwa dalam Bahasa Inggris huruf c dan h selalu digunakan berdampingan sebagai selalu digunakan berdampingan sebagai ch (lihat contoh charm dan march), sehingga kita hanya membuat permutasi huruf-huruf dengan c dan h berdampingan. Semua permutasi dengan huruf c dan h tidak berdampingan ditolak dari pencarian.

D. *Tug of War*

Gambar 3 Permainana Tarik Tambang
Sumber : https://rakyatku.com/read/160876/tak-hanya-seru-ini-6-manfaat-permainan-tarik-tambang

Tug of War atau dalam bahasa indonesia tarik tambang adalah permainan tradisional sederhana yang mangandalkan kekuatan dan kekuatan tim. Permainan ini terdiri dari dua regu dan menggunakan tali tambang sebagai

alatnya. Permainan tarik tambang sudah dikenal sejak zaman penjajahan. Konon, bangsa Indonesia yang saat itu sedang dalam masa kerja paksa oleh penjajah, menjadikan tali tambang yang biasa digunakan untuk bekerja sebagai mainan untuk mengisi waktu luang dan bersenang-senang. Permainan tersebut kemudian berkembang menjadi ajang mengadu kekuatan, hingga akhirnya berkembang menjadi perlombaan.

Saat ini tarik tambang umumnya diadakan saat hari-hari perayaan seperti ketika merayakan Hari Kemerdekaan Indonesia. Permainan ini dimainkan dengan cara membagi pemain menjadi dua regu. Pada tengah area pertandingan dan tali tambang diberi penanda sebagai batas area lawan. Masing-masing regu menarik tali tambang ke arah yang berlawanan. Pemenang dari permainan ini ditentukan dari regu yang berhasil menarik regu lawan ke daerahnya.

Permainan ini kemudian diadaptasi dan dijadikan sebuah persoalan. Misalkan diberikan sebuah array of integer berisi kekuatan pemain. Array tersebut kemudian dibagi dua bagian sedemikian rupa sehingga masing-masing subset jika dijumlah kan akan memiliki jumlah yang kira-kira sama besar.

III. Implementasi Bruto Force pada Tug of War

Langkah-langkah penyelesaian persoalan Tug of War dengan menggunakan exhaustive search. Disediakan sebuah array of integer. Array tersebut kemudian akan dibagi menjadi dua array (subset1 dan subset2) yang jika digabungkan akan menjadi array of integer semula

1. Bangkitkan semua subset dari array of integer
2. Masukkan subset tersebut ke dalam sebuah array
3. Iterasi array untuk mendapatkan subset1
4. Iterasi array of integer, periksa apabila tiap item pada array of integer sudah terdapat pada subset1, jika belum, masukkan item tersebut ke subset2
5. Hitung jumlah seluruh integer pada subset1 dan subset2
6. Jika selisih dari kedua jumlah subset tidak sama dengan nol, ulangi langkah 3
7. Jika selisih dari kedua jumlah subset sama dengan nol, hentikan iterasi, cetak hasil dari subset1 dan subset2 ke layar

Berikut ini dilampirkan penyelesaian persoalan Tug of War dalam bahasa python

```
#mendapatkan jumlah dari semua elemen
array
def sum_array(arr):
    sum = 0
    for item in arr:
        sum += item
    return sum

#dapatkan subset dari array
def generate_subset(arr):
    yield []
    for i in range(len(arr)):
        for j in
generate_subset(arr[i+1:]):
            yield [arr[i]] + j

#masukkan subset dari array ke sebuah
array baru
def getpowerset(arr):
    result = []
    for item in generate_subset(arr):
        result.append(item)
    return result

#dapatkan subset2 dari mengiterasi
array awal dengan subset1
def getSubset2(arr,sub1):
    sub2= []
    for item in arr:
        if item not in sub1:
            sub2.append(item)
    return sub2


def TugOfWar(arr):
    data = getpowerset(arr)
    selisih = -1
    track = 0
    subset1 = []
    subset2 = []
    while(track < len(data) and
    selisih != 0):
        sub1 = data[track]
        track += 1
        sub2 = getSubset2(arr,sub1)
        new_selisih =
abs(sum_array(sub1) -
sum_array(sub2))
        if(selisih == -1 or
new_selisih < selisih):
            selisih = new_selisih
            subset1 = sub1
            subset2 = sub2
    print("Subset1:",subset1)
    print("Jumlah Subset1:",
    sum_array(subset1))
    print("Subset2:",subset2)
    print("Jumlah Subset2:",
    sum_array(subset2))
```

Pada penyelesaian tersebut, jika subset1 berisi array kosong ([]), subset2 akan sama dengan array awal. Pembagian subset seperti itu tidak akan mengarah ke solusi, sehingga dapat diabaikan. Hal yang serupa akan terjadi jika subset1 sama dengan array awal dan subset2 merupakan array kosong. Oleh karena itu, dengan menggunakan teknik heurstik, kandidat solusi dapat dikurangi sehingga pencarian solusi akan lebih cepat. Teknik heuristik dapat digunakan pada fungsi getpowerset dengan cara mengecek jumlah item pada calon subset,

jika jumlah item sama dengan jumlah array awal atau jika calon subset merupakan array kosong, calon subset tersebut dapat diabaikan dan tidak dimasukkan ke array result. Sehingga penyelesaina Tug of War akan menjadi seperti berikut:

```
#mendapatkan jumlah dari semua elemen array
def sum_array(arr):
    sum = 0
    for item in arr:
        sum += item
    return sum

#dapatkan subset dari array
def generate_subset(arr):
    yield []
    for i in range(len(arr)):
        for j in generate_subset(arr[i+1:]):
            yield [arr[i]] + j

#masukkan subset dari array ke sebuah array baru
def getpowerset(arr):
    result = []
    for item in generate_subset(arr):
        if (len(item) != 0 and len(item) != len(arr)):
            result.append(item)
    return result

#dapatkan subset2 dari mengiterasi array awal dengan subset1
def getSubset2(arr,sub1):
    sub2= []
    for item in arr:
        if item not in sub1:
            sub2.append(item)
    return sub2


def TugOfWar(arr):
    data = getpowerset(arr)
    selisih = -1
    track = 0
    subset1 = []
    subset2 = []
    while(track < len(data) and selisih != 0):
        sub1 = data[track]
        track += 1
        sub2 = getSubset2(arr,sub1)
        new_selisih = abs(sum_array(sub1) - sum_array(sub2))

        if(selisih == -1 or new_selisih < selisih):
            selisih = new_selisih
            subset1 = sub1
            subset2 = sub2
    print("Subset1:",subset1)
    print("Jumlah Subset1:", sum_array(subset1))
    print("Subset2:",subset2)
    print("Jumlah Subset2:", sum_array(subset2))
```

IV. Pengujian Program

1. Testcase 1
   Masukan : [23, 45, -34, 12, 0, 98, -99, 4, 189, -1, 4]
   Keluaran:

```
Subset1: [23, 0, 98]
Jumlah Subset1: 121
Subset2: [45, -34, 12, -99, 4, 189, -1, 4]
Jumlah Subset2: 120
```

Gambar 4 Test Case 1

Dari gambar 4, bisa dilihat kedua subset memiliki jumlah nilai yang hampir sama

2. Testcase 2
   Mauskan:[3, 4, 5, -3, 100, 1, 89, 54, 23, 20]
   Keluaran:

```
Subset1: [3, 4, -3, 100, 1, 23, 20]
Jumlah Subset1: 148
Subset2: [5, 89, 54]
Jumlah Subset2: 148
```

Gambar 5 Test Case 2

Dari gambar 5, bisa disimpulkan kedua subset memiliki jumlah nilai yang sama

V. Kesimpulan

Algoritma Brute Force dapat diaplikasikan pada hampir seluruh persoalan yang ada. Algoritma brute force cocok untuk persoalan kecil dan sederhana yang tidak membutuhkan komputasi yang besar, namun algoritma ini bukanlah algoritma yang mangkus. Walaupun demikian berdasarkan test case, algoritma ini mampu memberikan solusi yang tepat. Kompleksitas dari algoritma ini adalah 2^n, sehingga untuk array yang lebih besar dibutuhkan algoritma lain yang dapat memberikan waktu komputasi yang lebih cepat.

Ucapan Terima Kasih

Puji syukur kepada Tuhan Yang Maha Esa atas berkat dan rahmatnya, makalah ini dapat terselesaikan dengan tepat waktu.Penulis menyampaikan terima kasih kepada keluarga,

teman-teman, dan semua pihak yang telah mendukung studi dan proses pembelajaran dalam mata kuliah IF2211 Strategi Algoritma. Penulis juga mengucapkan terima kasih kepada dosen pengampu mata kuliah IF2211 Strategi Algoritma yaitu, Bapak Dr. Rinaldi Munir, MT atas ilmu yang telah diberikan selama semester 2 tahun ajaran 2021/2022.

REFERENCES

[1] https://informatika.stei.itb.ac.id/~rinaldi.munir/Stmik/2020-2021/Algoritma-Brute-Force-(2021)-Bag1.pdfdiakses pada 23 Mei 2022 pukul 15.00

[2] https://informatika.stei.itb.ac.id/~rinaldi.munir/Stmik/2020-2021/Algoritma-Brute-Force-(2021)-Bag2.pdf diakses pada 23 Mei 2022 pukul 17.00

[3] https://kumparan.com/kabar-harian/permainan-tarik-tambang-cara-memainkan-dan-aturan-permainannya-1wTVZL8WAzg/1 diakses pada 23 Mei 2022 pukul 19.00

[4] https://rakyatku.com/read/160876/tak-hanya-seru-ini-6-manfaat-permainan-tarik-tambang diakses pada 23 Mei 2022 pukul 19.24

[5] Kurniawan, Ari Wibowo (2019). *Olahraga dan Permainan Tradisional* (PDF). Malang: Penerbit Wineka Media. hlm. 81–82. ISBN 978-602-5973-94-9.

[6] https://www.geeksforgeeks.org/tug-of-war/ diakses pada 22 Mei 2022 pukul 14.35

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, 23 Mei 2022

Vionie Novencia Thanggestyo 13520006