Penggunaan Algoritma Exhaustive Search dalam Menyelesaikan Scrabble

Monica Adelia - 13520096
Program Studi Teknik Informatika
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung, Jalan Ganesha 10 Bandung
E-mail (gmail): 13520096@std.stei.itb.ac.id

***Abstract*—Algoritma *Exhaustive Search* merupakan sebuah pendekatan yang mudah untuk menyelesaikan permasalahan karena sifatnya yang sangat sederhana, langsung, dan cara yang jelas serta mudah dipahami. Implementasi dari *Exhaustive Search* ini adalah mencari kombinasi kata dari huruf-huruf tersedia seperti halnya pada permainan *Scrabble*. Setiap huruf pada permainan Scrabble memiliki poin-poinnya sendiri dan nilainya pun berbeda. Kata dengan huruf terpanjang belum tentu memberikan poin terbesar. Algoritma *exhaustive search* dimanfaatkan untuk menemukan kata dengan poin terbesar agar dapat memenangkan permainan.**

Kata Kunci—Brute Force; Scrabble; Algoritma; Exhaustive Search

I. PENDAHULUAN

Permainan papan adalah salah satu jenis permainan yang dimainkan dengan memanfaatkan papan sebagai sarana bermainnya. Keberadaan permainan papan sudah sejak zaman kuno. Seiring berkembangnya zaman, permainan papan sudah hadir dalam bentuk komputer atau permainan online sehingga tidak membutuhkan papan secara fisik. Permainan papan merupakan permainan berlawanan sehingga ada dua pihak dalam sekali permainannya. Namun karena perkembangan teknologi, permainan bisa dilakukan secara tunggal dengan melawan komputer. Permainan papa nada beragam, seperti Monopoli, Go game, Catur, Candy Land, Scrabble, Boggle, Sequence, Battleship, Guess Who, Pictionary, dan masih banyak lagi.

Scrabble merupakan permainan susun kata. Dalam satu gamenya, dapat dimainkan oleh dua sampai dengan empat orang. Dalam bermainnya, setiap pemain dapat mengambil tujuh buah kepingan huruf secara acak yang dimasukkan ke dalam kantung. Lalu, kepingan yang diambil tersebut, dibentuk menjadi sebuah kata yang tersusun secara horizontal maupun vertikal. Namun dalam pembuatan kata, kata baru tersebut harus bersambung dari salah satu huruf dari kata yang sudah ada pada papan. Kata-kata yang terbentuk harus sesuai dengan standar kamus, sesuai dengan Bahasa yang digunakan dalam permainan, baik Bahasa Indonesia, Bahasa Inggris, mauapun bahasa lain sesuai kesepakatan.

Gambar 1.1 Logo Scrabble
(Sumber: https://1000logos.net/scrabble-logo/)

Pemenang dari permainan ini adalah pemain yang dapat mengumpulkan poin sebanyak-banyaknya. Poin ini berada pada kepingan-kepingan huruf. Setiap kata yang dapat dibentuk, akan dijumlahkan nilai-nilai yang berada pada setiap hurufnya. Selain itu, pada papan permainan, terdapat bonus, seperti kata yang terletak pada bagian tersebut akan dikali lipatkan poinnya (bisa kali 2, dikali 3, dan lain-lain).

Pada makalah ini, penulis mencoba membuat program yang dapat memberikan kata yang menghasilkan poin tertinggi menurut poin pada kepingannya. Dalam pembuatan program, dimanfaatkan algoritma *exhaustive search* untuk menemukan kemungkinan kata yang terbentuk. Algoritma ini dipilih karena algoritma ini memang membutuhkan waktu yang cukup lama dan kompleksitas yang tidak terlalu baik, namun memberikan hasil yang optimal.

II. LANDASAN TEORI

A. *Algoritma Brute Force*

Algorima *Brute Force* merupakan pendekatan yang mudah untuk menyelesaikan suatu persoalan atau permasalahan. Algoritma ini biasa disebut pendekatan yang lempang atau *straightforward*. Algoritma *brute force* berfokus pada pernyataan yang ada dari suatu persoalan. Pemecahan persoalan dengan *brute force* memiliki sifat

1. Algoritma yang sederhana
2. Penyelesaian dengan cara langsung
3. Cara yang jelas

Persoalan yang dapat diselesaikan oleh algoritma *brute force* terbilang banyak karena sifatnya yang tidak memiliki

konsep dan langsung menyelesaikan saja. Contohnya, pencarian elemen terbesar atau terkecil dari suatu array, pencarian beruntun, menghitung perpangkatan, menghitung faktorial, perkalian matriks,

Karakteristik dari algoritma *brute force* adalah

1. Algoritma *brute force* bisa dikatakan tidak begitu cerdas dan tidak mangkus karena dalam pembuatannya tidak perlu pemikiran yang begitu mendalam, membutuhkan volume komputasi yang cukup besar, dan waktu atau kompleksitasnya yang cukup besar. Algoritma ini lebih memanfaatkan tenaga dalam penyelesaiannya daripada konsep atau metode penyelesaian yang efektif dan efisien.
2. Algoritma *brute force* lebih cocok untuk permasalahan dengan ukuran inputan yang kecil. Hal ini dikarenakan, algoritma *brute force* yang melakukan pengecheck-an satu persatu pada tiap elemen sehingga untuk persoalan yang inputannya cukup besar, akan memakan waktu yang cukup lama. Karena hal ini, algoritma *brute force* sering kali dimanfaatkan sebagai basis atau dasar pembanding dengan algoritma lain dalam menghitung kecepatannya.
3. *Brute* force bukanlah algoritma yang mangkus, namun dapat menyelesaikan hampir semua permasalahan. Karena sifatnya yang sederhana dan tidak memiliki banyak syarat, algoritma ini hampir dapat menyelesaikan semua persoalan yang ada. Bahkan ada beberapa permasalahan yang hanya dapat diselesaikan dengan menggunakan algoritma *brute force*. Ketika memiliki keraguan dengan, algoritma lain, pemrogram cenderung menggunakan algoritma ini karena sederhana dan dapat menemukan solusi dari permasalahan.

Algoritma ini memiliki kelemahan dan kelebihan. Kelemahan dari *brute force* adalah

1. *Brute force* jarang menghasilkan algoritma efisien
2. Lambat dan membutuhkan memori yang besar untuk inputan yang berukuran besar
3. Tidak kreatif seperti pemecahan masalah dengan strategi lainnya

Kelebihan dari *brute force* adalah

1. Menghasilkan algoritma layak untuk beberapa masalah penting, seperti pengurutan, pencarian, perkalian matriks, dan lain-lain.
2. Dapat digunakan untuk memecahkan hampir semua permasalahan yang ada (*wide applicability*)
3. Sederhana sehingga mudah dipahami
4. Menghasilkan algoritma dasar sebagai pembanding ataupun menyelesaikan komputasi seperti perkalian, penjumlahan, pencarian elemen minimum atau maksimum, dan lain-lain.

B. Echaustive Search

Exhaustive Search merupakan salah satu strategi dalam penyelesaian suatu permasalahan atau persoalan dengan cara *brute force* untuk persoalan-persoalan kombinatorik. Persoalan yang dimaksud dengan persoalan kombinatorik adalah persoalan yang berada di antara objek-objek kombinatorik seperti kombinasi, permutasi, atau himpunan bagian dari sebuah himpunan.

Langkah-langkah penyelesaian masalah dengan *exhaustive search* adalah

1. Mencari setiap kemungkinan solusi dengan cara yang sistematis.
2. Setiap kemungkinan solusi yang sudah ditemukan, evaluasi satu persatu. Simpan solusi terbaik yang ditemukan (the best solution found so far).
3. Jika semua kemungkinan solusi sudah dievaluasi, hasil dari pencarian adalah solusi terbaik yang tersimpan.

Secara teoritis, pencarian ini akan memberikan solusi, namun waktu dan memori yang dibutuhkan dalam pencarian *exhaustive search* ini cukup besar. Contoh kasus yang dapat diselesaikan permasalahan exhaustive search adalah *Travelling Salesperson Problem* (TSP), 1/0 *Kanpsack Problem*, kriptografi, dan lain-lain.

C. Teknik Heuristik

Teknik heuristik adalah teknik yang mengefisiensikan suatu strategi dalam proses pencarian. Algoritma *exhaustive search* dapat ditingkatkan efisiensinya dengan memanfaatkan teknik heuristik. Sehingga, pencarian solusi tidak perlu melakukan iterasi atau eksplorasi semua kemungkinan solusi.

Pada algoritma *exhaustive search*, teknik heuristik ini dimanfaatkan untuk mengeliminasi sehingga kemungkinan solusi tidak sebanyak hasil kombinasinya. Algoritma juga tidak harus mengeksplor semua kemungkinan solusi. Teknik ini menggunakan pendekatan yang tidak formal karena berdasarkan terkaan, penilaian intuitif, dan akan sehat.

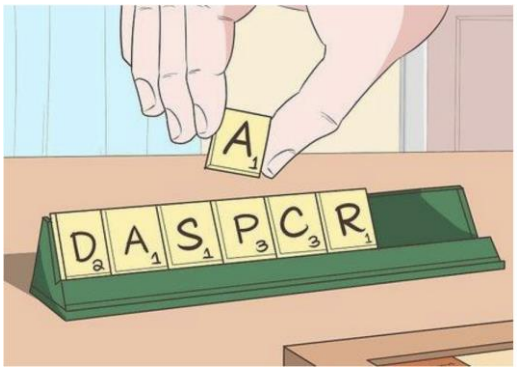
Teknik heuristik memecahkan suatu permasalahan bukan berdasarkan hitungan matematis melainkan berdasarkan pengalaman, proses pembelajaran, dan percobaan sebelumnya. Penemuan solusi tidak pasti optimal dengan teknik ini. Heuristik ini merupakan sebuah teknik bukanlah algoritma karena heuristik di sini hanyalah sebagai panduan atau tambahan informasi, sedangkan algoritma merupakan urutan langkah-langkah yang digunakan untuk menyelesaikan suatu permasalahan atau persoalan.

Teknik heuristik biasanya tidak dapat dibuktikan secara matematis karena sifatnya yang berbentuk terkaan, *common sense*, dan intuisi. Namun heuristik cukup berguna terlebih lagi pada data dengan inputan yang besar. Teknik heuristik cukup mengurangi waktu yang digunakan untuk memecahkan persoalan dengan mengeliminasi kebutuhan untuk mengevaluasi kemungkinan solusi yang tidak perlu. Penggunaan teknik heuristik sudah cukup lama terlebih lagi di dalam bidang *artificial intelligence*, contohnya dalam metode

best first search, hill climbing, algoritma A*, dan masih banyak lagi.

D. Scrabble

Poin dalam permainan *Scrabble* didapatkan dari poin yang berada pada tiap kepingannya. Poin pada tiap kepingan nilainya berbeda tidak selalu sama. Sehingga, kata yang panjang tidak berarti poin yang besar.



Gambar 2.1 Kepingan berpoin
(Sumber: https://www.thefastcode.com/id-idr/wiki/Bermain-Scrabble)

Selain dari poin yang terdapat pada kepingan-kepingan huruf, terdapat juga bonus atau pengali poin pada board. Pengali ditandai dengan kotak pada board yang berwarna. Tiap board pada game scrabble dapat berbeda-beda bonusnya. Bonus ini akan mengkali poin yang terletak di atasnya sehingga poin yang didapatkan akan lebih besar lagi. Oleh karena itu, yang dapat mempengaruhi poin yang didapatkan tidak hanya poin pada kepingan, tetapi juga nilai bonus pengali pada papan board.

Gambar 2.2 *Board Scrabble*
(Sumber: https://playscrabble.com/)

Cara memainkan permainan *Scrabble* adalah sebagai berikut:

1. Meletakkan kata pertama. Kata yang dibuat minimal terdiri dari dua huruf dan harus terletak di atas kotak bintang di tengah papan permainan. Peletakan kata dapat berbentuk horizontal dan juga vertikal, namun tidak boleh diagonal.
2. Hitung poin yang didapatkan dari kata yang terbentuk.
3. Ambil balok huruf baru sesuai jumlah huruf pada kata yang sudah dikeluarkan.
4. Lanjut ke orang berikutnya menjalankan langkah satu sampai tiga.

III. Implementasi dan Analisis Pengujian

Penyelesaian permasalahan dalam permainan Scrabble, penulis implementasikan dengan memanfaatkan bahasa pemrograman Python beserta library-nya.

A. Implementasi

Implementasi solusi untuk menyelesaikan permasalahan ini dibagi menjadi beberapa bagian, yaitu:

1. Exhaustive Search

```
def createPossibleWord(word):
    y = [[a for a in p(word,y)] for y in range(1,len(word)+1)]

    this = []
    for i in y:
        while len(i)>0:
            this.append(i.pop())
    return [''.join(x) for x in this]
```

Gambar 3.1 fungsi createPossibleWord
(Sumber: penulis)

Fungsi createPossibleWord merupakan fungsi yang menerapkan algoritma *exhaustive search*. Fungsi ini menerima inputan kata yang tersusun atas huruf *random* yang nantinya akan dilakukan enumerasi. Hasil enumerasi adalah himpunan kata atau kumpulan huruf dari ukuran 0 sampai jumlah huruf inputan, dengan kombinasi yang berbeda. Himpunan ini merupakan kemungkinan solusi.

2. Teknik Heuristik

```
def fungsiHeuristik(list):
    for x in list:
        if(len(x) == 1):
            list.remove(x)
```

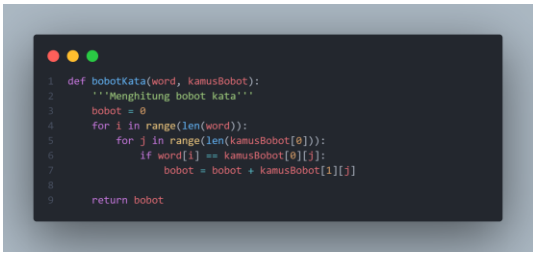
Gambar 3.2 fungsi heuristik
(Sumber: penulis)

Fungsi fungsiHeuristik ini adalah implementasi dari teknik heuristik. Teknik heuristik dimanfaatkan untuk mengoptimalkan proses pencarian, yaitu dengan membuat fungsi heuristik yang mengeliminasi beberapa kemungkinan solusi. Kemungkinan solusi yang dieliminasi oleh fungsi adalah kata yang terdiri dari nol sampai satu huruf.

Fungsi ini dibuat berdasarkan aturan permainan *Scrabble*. Setiap pemain yang ingin menaruh kata pada papan permainan, kata tersebut harus terhubung salah satu (atau lebih) hurufnya dengan kata lain yang sudah

ada pada papan kata. Dapat dikatakan bahwa huruf yang hanya terdiri dari satu huruf tidak mungkin digunakan pada permainan ini. Karena minimal satu huruf berasal dari kata yang sudah ada pada papan. Sama dengan kata yang terdiri dari nol huruf, ini sangat tidak mungkin digunakan pada permainan ini karena tandanya tidak ada kata yang dikeluarkan.

3. Bobot Kata



Gambar 3.3 fungsi bobotKata
(Sumber: penulis)

Fungsi ini merupakan fungsi yang digunakan untuk menghitung bobot kata yang sudah dievaluasi. Perhitungan bobot akan mengacu dari bobot per huruf hasil inputan *user* sesuai yang ada pada permainan *Scrabble*.

4. Main Program

Main program dari kode ini mengikuti langkah langkah sebagai berikut:

i. Pembacaan kamus kata

Kamus kata diperlukan pada program ini karena dalam proses evaluasi kumpulan huruf pada kemungkinan solusi, diperlukan acuan yang akan menentukan apakah suatu kemungkinan merupakan kata yang memiliki arti atau bukan. Kamus kata ini terdiri dari kosa kata sesuai dengan bahasa yang digunakan ketika bermain *Scrabble*.

ii. Menerima inputan huruf beserta bobotnya

Program menerima data mengenai huruf beserta bobotnya untuk pembuatan kemungkinan solusi. Inputan huruf beserta bobotnya disesuaikan dengan data yang ada pada kepingan permainan *Scrabble* yang sedang kita miliki.

iii. Pembuatan kemungkinan kata dari huruf inputan

Kemungkinan kata dari huruf inputan dibuat dengan memanfaatkan fungsi createPossibleWord. Lalu digunakan juga teknik heuristik yaitu pada fungsi fungsiHeuristik untuk mengeliminasi beberapa dari kemungkinan solusi.

iv. Evaluasi kemungkinan solusi kata dengan kamus

Evaluasi ini dilakukan dengan pencarian kemungkinan solusi pada kamus yang dimiliki. Bila kata ada pada kamus, masukkan kata pada himpunan hasil, jika tidak, kata tidak dimasukkan pada himpunan hasil.

v. Perhitungan bobot dari kata yang terbentuk

Himpunan kata yang sudah terkumpul hasil evaluasi, dihitung bobotnya sesuai dengan bobot huruf inputan. Hasil ini diurutkan berdasarkan bobot terbesar.

vi. Jawaban

Hasil dari permasalahan inputan sudah didapatkan dalam bentuk kata yang terbentuk beserta bobotnya terurut dari bobot terbesar hingga yang terkecil.

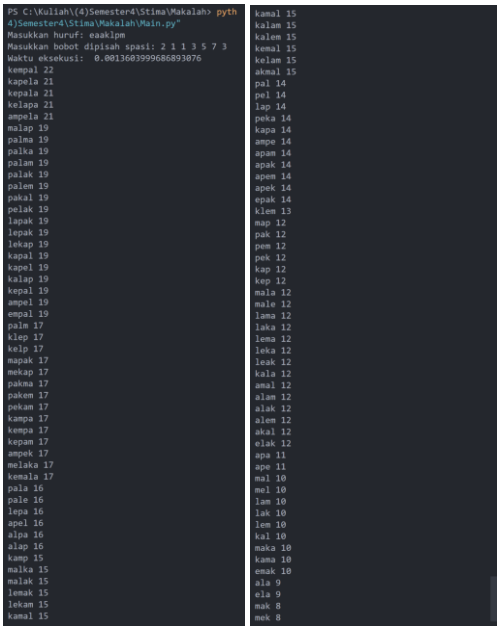
B. Analisis Pengujian

Pada proses pengujian ini, kamus kata yang digunakan adalah kamus kosa kata Bahasa Indonesia dengan jumlah kata sebanyak 31375. Kode program dapat digunakan untuk pencarian kata dengan bahasa apa saja hanya perlu menyesuaikan pada bagian kamus datanya saja. Jumlah huruf yang digunakan pada kasus uji ini berjumlah tujuh menyesuaikan dengan permainan *Scrabble* pada umumnya. Terdapat beberapa kasus uji yang digunakan dalam pengujian kali ini, yaitu sebagai berikut

1. Kasus Uji 1

Huruf	**Bobot**
e	2
a	1
a	1
k	3
l	5
p	7
m	3

Hasil pengujian dari kombinasi huruf dan bobot di atas adalah



Gambar 3.4 hasil uji kasus 1
(Sumber: penulis)

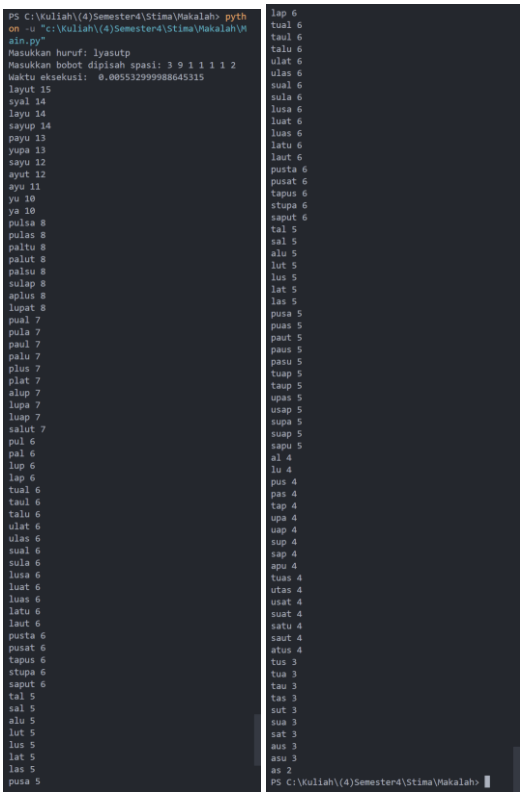
Waktu eksekusi yang diperlukan untuk mencari jawaban adalah 0.001360 sekon. Waktu perhitungan ini dikategorikan cukup singkat. Dalam satu permainan, satu pemain mendapatkan beberapa menit untuk menentukan kata, umumnya 5 menit. Sehingga, waktu ini seharusnya lebih dari cukup.

Dari hasil pencarian, kombinasi kata yang memberikan bobot terbesar adalah kata “kempal” dengan bobot sebesar 22. Banyak huruf kata pada kata “kempal” dan “kapela” sama-sama enam, tapi kedua kata tersebut memberikan bobot yang berbeda yaitu 22 dan 21.

2. Kasus Uji 2

Huruf	Bobot
l	3
y	9
a	1
s	1
u	1
t	1
p	2

Hasil pengujian dari kombinasi huruf dan bobot di atas adalah



Gambar 3.5 hasil uji kasus 2
(Sumber: penulis)

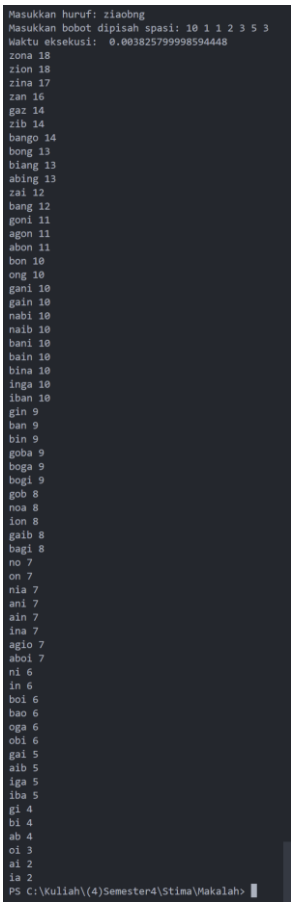
Waktu eksekusi yang diperlukan untuk mencari jawaban adalah 0.005532 sekon. Waktu perhitungan ini dikategorikan cukup singkat. Dalam satu permainan, satu pemain mendapatkan beberapa menit untuk menentukan kata, umumnya 5 menit. Sehingga, waktu ini seharusnya lebih dari cukup. Dari hasil pencarian, kombinasi kata yang memberikan bobot terbesar adalah kata “layut” dengan bobot 15.

Dari hasil uji kasus dua ini, dapat dilihat bahwa kata “ya” yang terdiri dari dua huruf menghasilkan bobot yang cukup besar yaitu 10. Sedangkan kata “sapu” yang terdiri dari empat huruf hanya menghasilkan bobot 5. Dapat dilihat bahwa panjang kata tidak selalu menjamin bobot akan selalu besar pada kasus permainan *Scrabble* ini.

3. Kasus Uji 3

Huruf	Bobot
z	10
i	1
a	1
o	2
b	3
n	5
g	3

Hasil pengujian dari kombinasi huruf dan bobot di atas adalah



Gambar 3.6 hasil uji kasus 3
(Sumber: penulis)

Waktu eksekusi yang diperlukan untuk mencari jawaban adalah 0.003825 sekon. Waktu perhitungan ini dikategorikan cukup singkat. Dalam satu permainan, satu pemain mendapatkan beberapa menit untuk menentukan kata, umumnya 5 menit. Sehingga, waktu ini seharusnya lebih dari cukup.

Dari hasil pencarian, kombinasi kata yang memberikan bobot terbesar adalah kata “zona” dengan bobot 18. Kata “bango”, “biang”, dan “abing” yang teridiri dari 5 huruf, namun bobotnya lebih kecil dari kata “zona”.

Pengujian bisa dikategorikan berhasil. Program dapat menampilkan hasil dari pencarian kata dengan baik. Kata juga terurut dari bobot yang terbesar ke yang terkecil sehingga memudahkan pemain untuk memilih kata yang memberikan poin terbesar.

Pada penggunaannya dalam permainan *Scrabble*, ditemukan beberapa kesulitan yaitu pada peletakan kata. Pemain tidak boleh sembarangan meletakkan kata pada bagian dari papan yang kosong. Melainkan, salah satu huruf atau lebih harus terhubung pada huruf lain yang sudah ada pada papan. Sehingga seluruh kata terhubung. Terkadang, kata yang memiliki bobot paling besar, tidak memiliki huruf untuk dihubungkan pada huruf yang ada pada papan. Sehingga, kata tersebut tidak dapat digunakan.

Pada kasus tertentu juga, kata dengan bobot terbesar pada hasil pencarian tidak menghasilkan poin terbesar pada papan *Scrabble*. Contohnya adalah pada kasus dimana kata lain dengan poin kepingan lebih kecil, namun karena peletakannya ada pada papan yang memiliki bonus pengali yang cukup besar, hasil poinnya pun berbeda. Seperti yang sudah disebutkan pada bagian landasan teori, hal ini wajar terjadi karena unsur yang mempengaruhi poin yang dapat didapatkan tidak hanya pada kepingan namun juga pada papan.

IV. KESIMPULAN

Algoritma *exhaustive search* atau *brute force* dengan pemanfaatan teknik heuristik dapat digunakan untuk menyelesaikan persoalan *Scrabble*. Namun, dalam penerapannya, terdapat beberapa unsur yang dapat memberikan poin lebih besar selain dari poin kepingan yaitu peletakan kata pada papan dengan bonus pengali yang besar. Sehingga faktor yang mempengaruhi poin tidak hanya satu. Selain itu, pemain tidak hanya mencari kata yang memberikan poin terbesar, namun juga harus bisa dihubungkan dengan kata lain yang sudah ada pada papan. Oleh karena hal ini, terkadang kata dengan bobot terbesar tidak dapat digunakan karena tidak ada penghubung dengan kata pada papan.

Algoritma ini cukup bermanfaat untuk memudahkan dalam mencari kata dengan bobot terbesar. Jika dimanfaatkan dengan baik, algoritma ini juga dapat memudahkan pemain untuk memenangkan permainan. Terdapat banyak algoritma lain yang dapat dimanfaatkan untuk menyelesaikan permasalahan ini. Eksplorasi lebih lanjut dari berbagai sisi dibutuhkan untuk menyelesaikannya.

Algoritma *exhaustive search* ini akan cocok untuk digunakan dalam persoalan pencarian kata dari huruf-huruf yang tidak beraturan.

VIDEO LINK AT YOUTUBE

Berikut ini penulis lampirkan video penjelasan terkait permasalahan yang diangkat pada permainan *Scrabble* dan solusinya dengan memanfaatkan algoritma *exhaustive search*. Video dapat diakses pada *link youtube* berikut, https://youtu.be/19Ah5daVxiA

UCAPAN TERIMA KASIH

Puji syukur penulis ucapkan kepada Tuhan Yang Maha Esa karena berkat dan rahmat-Nya penulis dapat membuat dan menyelesaikan makalah ini dengan baik dan lancar. Penulis juga mengucapkan terima kasih kepada Bapak Dr. Ir. Rinaldi, M.T. selaku pengajar mata kuliah Strategi Algoritma. Penulis juga mengucapkan terima kasih kepada keluarga yang telah

memberikan inspirasi dan teman-teman penulis yang membantu serta memberikan dukungan terhadap penulis selama pembuatan makalah ini.

REFERENSI

[1] https://informatika.stei.itb.ac.id/~rinaldi.munir/Stmik/2021-2022/Algoritma-Brute-Force-(2022)-Bag1.pdf diakses pada 23 Mei 2022 pukul 16.00

[2] https://informatika.stei.itb.ac.id/~rinaldi.munir/Stmik/2021-2022/Algoritma-Brute-Force-(2022)-Bag2.pdf diakses pada 23 Mei 2022 pukul 18.00

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

Monica Adelia 13520096