Penerapan Algoritma *Backtracking* dan Struktur Data Trie dalam Menyelesaikan *Word Search Puzzle*

Ahmad Saladin (*13519187*)
Program Studi Teknik Informatika
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung, Jalan Ganesha 10 Bandung
13519187@std.stei.itb.ac.id

***Abstrak*— Word search puzzle adalah sebuah puzzle yang terdiri dari huruf-huruf tersusun dalam bentuk grid yang mengandung sejumlah kata tersembunyi di dalamnya. Pencarian kata yang tersembunyi dapat dilakukan dengan menerapkan algoritma backtracking dan juga struktur data trie. Algoritma ini akan menemukan seluruh kata tersembunyi dalam puzzle jika diberika list kata-kata yang ada.**

Kata kunci—backtracking; trie; word search

I. PENDAHULUAN

Word search puzzle merupakan suatu permainan dengan tujuan mencari sejumlah kata yang tersembunyi dalam kumpulan huruf-huruf yang tersusun dalam grid. Dalam menyelesaikan puzzle ini banyak strategi yang dapat diterapkan. Misalnya mencari secara berurutan apakah terdapat huruf pertama dari kata yang dicari pada setiap baris dan kolom atau dengan mencari huruf-huruf yang jarang muncul, dalam bahas inggris misalnya J, B, K, Q, X, Y, atau Z.

Word search puzzle juga dapat diselesaikan oleh komputer. Beberapa pendekatan yang dapat digunakan untuk menyelesaikan word search puzzle adalah brute force, greedy, dan, backtracking.

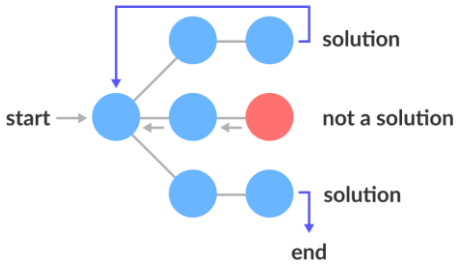
Makalah ini bertujuan untuk menjelaskan penerapan algoritma backtracking dan struktur data trie dalam menyelesaikan word search puzzle. Sistematika penulisian makalah terdiri dari lima bagian. Bagian pertama adalah pendahuluan, bagian kedua adalah landasan teori, bagian ketiga adalah analisis masalah dan implementasi, bagian keempat adalah pengujian, dan bagian kelima adalah kesimpulan

II. LANDASAN TEORI

A. *Algoritma Backtracking*

Algoritma Backtracking adalah suatu perbaikan dari algoritma exhaustive search. Berbeda dengan exhaustive search yang mengeksplorasi seluruh solusi yang mungkin dan kemudian dievaluasi satu persatu, pada algoritma backtracking hanya pilihan yang mengarah ke solusi yang dieksplorasi dan dievaluasi. Pilihan yang tidak mengarah ke solusi akan dipangkas dan tidak lagi dipertimbangkan[1].

Beberapa properti umum pada algoritma backtracking adalah sebagai berikut. Pertama Solusi persoalan dinyatakan sebagai vector dengan n-tuple: X = (x1, x2, …, Xn), Xi ∈ Si. Pada umumnya S1 = S2 = ... = Sn. Kedua terdapat fungsi pembangkit nilai xk yang membangkitkan nilai xk komponen vector solusi. Terakhir terdapat fungsi pembatas yang bertujuan untuk melakukan pemangkasan pada pilihan solusi. Fungsi ini akan bernilai true jika pilihan mengarah ke solusi (tidak melanggar constraint). Jika fungsi bernilai true pembangkitan nilai x akan dilanjutkan dan jika bernilai false akan dilakukan pemangkasan (simpul dibunuh oleh fungsi pembatas).

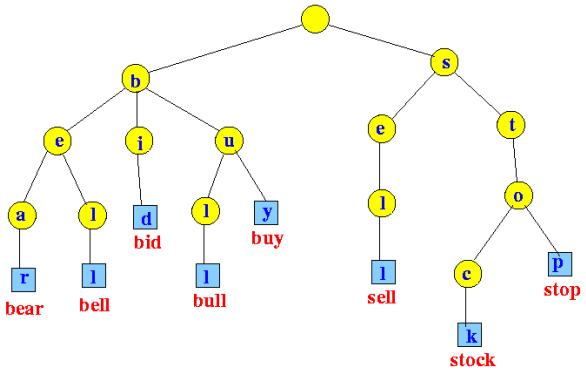


Gambar 1 Pohon Ruang Solusi Algoritma Backtracking
(Sumber: https://www.programiz.com/dsa/backtracking-algorithm)

Pada gambar 1 dapat dilihat contoh pohon ruang status suatu algoritma backtracking. Simpul-simpul yang berwarna biru adalah simpul yang memberikan nilai true pada fungsi pembatas sehingga simpul setelahnya dibangkitkan. Simpul yang berwarna merah memberikan nilai false pada fungsi pembatas sehingga simpul berikutnya tidak dibangkitkan dan proses pencarian akan backtrack ke simpul sebelumnya dan mencari simpul lain yang belum dievaluasi. Pencaria juga akan backtrack ke simpul sebelumnya jika sudah mencapai daun atau semua anak simpul tersebut sudah dievaluasi.

B. *Struktur Data Trie*

Struktur data trie adalah suatu struktur berbentuk pohon dimana setiap simpul merepresentasikan sebuah huruf dari kata tertentu. Tidak seperti pohon biner, setiap simpul pada trie dapat memiliki lebih dari 2 anak[3]. Pada sebuah trie suatu kata disimpan dalam bentuk lintasan dari akan ke daun seperti terlihat pada gambar 2.



Gambar 2 Trie
(Sumber: http://www.mathcs.emory.edu/)

Pencarian kata pada trie membutuhkan waktu yang lebih sedikit dibandingkan dengan kata yang disimpan dalam pohon biner. Kompleksitas waktu pencarian pada pohon biner adalah O(M * logN) sedangkan kompleksitas waktu pencarian pada trie adalah O(M) dimana M adalah Panjang kata yang dicari dan N adalah jumlah kata yang disimpan. Namun kekurangan dari trie adalah membutuhkan memory yang lebih besar[4].

C. *Word Search Puzzle*

Word search puzzle adalah sebuah puzzle yang terdiri dari huruf-huruf tersusun dalam bentuk grid yang mengandung sejumlah kata tersembunyi di dalamnya[1]. Kata-kata ini dapat menempati beberapa posisi berbeda yang dapat dilihat pada gambar berikut.

Gambar 3 Contoh *Word Search Puzzle*

III. Analisis Masalah dan Implementasi

Algoritma yang dibuat akan mencari posisi kata-kata yang telah diberikan dari persoalan di dalam puzzle. Kata-kata yang dicari akan disimpan oleh program dalam bentuk trie. Dengan menggunakan algoritma backtracking solusi persoalan berupa kumpulan huruf dalam puzzle yang memiliki salah satu dari posisi berikut.

- Horizontal dari kiri ke kanan.
- Horizontal dari kanan ke kiri.
- Vertikal dari atas ke bawah.
- Vertikan dari bawah ke atas.
- Diagonal dari kiri atas ke kanan bawah.
- Diagonal dari kiri bawah ke kanan atas.
- Diagonal dari kanan atas ke kiri bawah.
- Diagonal dari kanan bawah ke kiri atas.

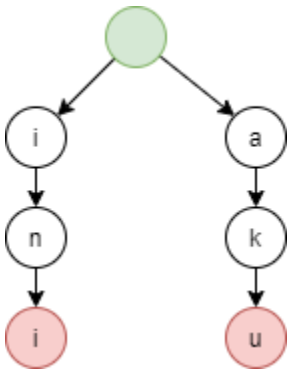
Fungsi pembangkit bekerja mulai dari pojok kiri atas puzzle (huruf 1) akan membangkitkan kumpulan huruf disekitarnya sesuai posisi solusi yang mungkin dengan huruf 1 sebagai awalnya. Terdapat dua buah fungsi pembatas yang akan dicek. Pertama jika pada posisi yang dicek (dari huruf 1 sampai ke pinggiran puzzle) jumlah huruf pada kumpulan huruf kurang dari jumlah huruf dari kata terpendek pada kata yang dicari fungsi akan bernilai false. Fungsi yang kedua akan mengecek apakah terdapat lintasan pada trie dari huruf 1 sampai huruf yang sedang dibangkitkan. Jika tidak terdapat lintasan fungsi ini akan bernilai false. Solusi ditemukan Ketika penelusuran huruf yang dibangkitkan pada trie mencapai node yang merupakan huruf terakhir dari kata yang dicari. Hal ini kemudian diulangi untuk huruf berikutnya dalam puzzle, bergerak secara horizontal dari kiri ke kanan lalu dilanjutkan dengan baris berikutnya.

Misalkan terdapat sebuah word search puzzle berukuran 5x5

A	B	C	I	I
B	K	D	N	A
C	U	U	I	K
D	E	C	F	B
E	B	B	A	A

Dengan kata yang dicari adalah ini dan aku.

Trie yang dibuat untuk menyimpan kata yang dicari dengan node hijau merupakan root dan node merah menggambarkan huruf terkahir kata adalah sebagai berikut.



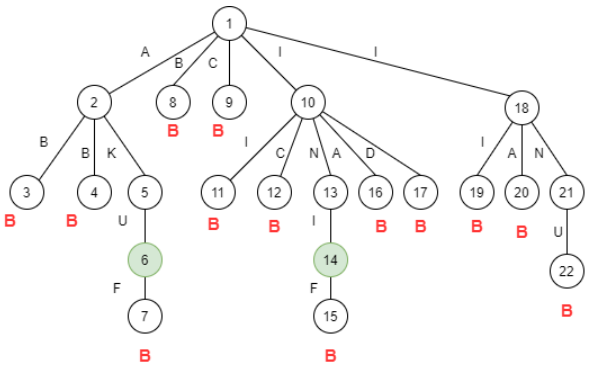
Gambar 4 Ilustrasi trie dalam program
(sumber: Arsip penulis)

Proses pembangkitkan simpul untuk huruf pada baris 1 kolom 1 ditunjukkan pada gambar 5. Pada ilustrasi tersebut huruf yang berwarna merah adalah simpul ekspan, dan huruf berwarna biru adalah huruf yang dibangkitkan. Pohon ruang status dengan B berwarna merah menandakan simpul yang

dibunuh oleh fungsi pembatas untuk Baris Pertama dari puzzle ditunjukkan pada gambar 6.

A	B	C	I	I
B	K	D	N	A
C	U	U	I	K
D	E	C	F	B
E	B	B	A	A

Gambar 5 Ilustrasi pembangkitan huruf untuk huruf pada baris 1 kolom 1
(sumber: Arsip penulis)



Gambar 6 Pohon ruang status baris pertama
(sumber: Arsip penulis)

A. *Pembentukan Trie*

Trie dibuat dalam bentuk sebuah tree yang setiap nodenya terdiri dari sebuah dictionary yang menyatakan anak dari node tersebut dan sebuah variabel Boolean yang menyatakan apakah node tersebut huruf terakhir kata atau bukan. Sebuah trie kosong adalah trie yang hanya berisi root yang tida memiliki anak.

Untuk menambahkan suatu kata ke trie akan dilakukan penelusuran dimulai dari root. Jika huruf pertama pada kata yang akan ditambahkan merupakan anak dari root, penulusuran akan dilanjutkan ke node tersebut. Jika tidak akan dibuat node baru (anak dari root) yang berisi huruf pertama dari kata yang ditambahkan dan penelusuran dilanjutkan ke node baru tersebut. Hal ini dilakukan sampai seluruh huruf dari kata yang ditambahkan sudah masuk ke dalam trie. Pada huruf terakhir variabel Boolean akan diset True yang menyatakan akhir dari kata.

Untuk melakukan pencarian pada trie akan dilakukan penelusuran dimulai dari root. Jika huruf pertama pada kata yang dicari merupakan anak dari root, penulusuran akan dilanjutkan ke node anak tersebut sampai huruf yang diperiksa habis. Jika tidak pencarian akan gagal dan berhenti.

B. *Algoritma Backtracking untuk Word Search Puzzle*

Langkah-langkah yang dilakukan pada algoritma Backtracking adalah sebagai berikut.

1. Cek huruf pertama apakah berada pada level 1 trie atau tidak.
2. Jika huruf tersebut ditemukan hitung Panjang kumpulan huruf maksimum dengan posisi horizontal kiri ke kanan.
3. Jika Panjang kumpulan huruf lebih besar atau sama dengan Panjang kata terpendek dalam trie lakukan penelusuran huruf dengan mencocokkan huruf-huruf pada wordsearch puzzle dimulai dari huruf pertama sesuai posisi yang sedang diperiksa dengan lintasan pada trie sampai ditemukan huruf yang tidak cocok atau mencapai huruf terakhir. Dalam penelusuran jika melewati node pada trie yang merupakan akhir sebuah kata artinya sebuah solusi ditemukan dan posisi huruf dalam wordsearch puzzle akan disimpan.
4. Ulangi Langkah 2 dan 3 untuk kumpulan huruf dengan posisi horizontal kiri ke kanan, vertikal atas ke bawah, vertikal bawah ke atas, diagonal diagonal kiri atas ke kanan bawah, diagonal kiri bawah ke kanan atas, diagonal kanan bawah ke kiri atas, dan diagonal kanan atas ke kiri bawah.
5. Ulangi Langkah 1-4 untuk semua huruf pada wordsearch puzzle.

IV. PENGUJIAN

Pengujian dilakukan pada dua buah word search puzzle berbeda. Puzzle pertama adalah puzzle dengan kategori medium berukuran 15x15 dan memiliki 11 kata tersembunyi. Puzzle kedua adalah puzzle dengan kategori hard berukuran 20x20 dan memiliki 28 kata tersembunyi.

Untuk pengujian pertama Puzzle dapat dilihat pada gambar 7 dan kata tersembunyi dapat dilihat pada tabel 1. Untuk pengujian kedua puzzle dapat dilihat pada gambar 8 dan kata tersembunyi dapat dilihat pada table 2.

E	R	Y	O	P	W	E	O	W	M	E	R	M	M	D
O	J	M	L	T	B	A	O	H	R	O	A	S	T	H
R	R	O	H	E	W	S	R	I	T	M	A	S	T	F
E	K	A	N	Y	C	S	G	T	E	I	T	H	R	A
H	A	V	E	F	Y	R	L	E	M	P	I	E	E	S
F	N	T	Y	X	R	D	V	M	P	K	P	W	H	O
O	E	S	S	U	O	G	B	U	O	H	N	H	C	E
H	G	O	D	E	D	O	R	A	N	G	E	I	E	A
F	G	R	A	Y	B	P	O	E	P	I	N	K	S	M
E	S	H	E	Y	L	C	W	X	D	L	E	G	B	F
E	T	I	C	E	N	T	N	M	N	C	A	E	L	E
H	T	A	I	L	N	B	H	H	D	O	U	C	A	E
N	T	L	E	L	U	L	J	G	T	L	E	E	C	A
E	W	A	M	O	I	U	D	O	P	A	H	N	K	S
L	E	M	O	W	I	E	L	R	C	S	O	T	B	D

Gambar 7 Word search Puzzle Kategori Medium
(Sumber: https://www.brainzilla.com/word-games/word-search/printable/)

BLACK	GREEN	RED
BLUE	ORANGE	WHITE
BROWN	PINK	YELLOW
GRAY	PURPLE	

Tabel 1 Kata Tersembunyi Kategori Medium
(Sumber: https://www.brainzilla.com/word-games/word-search/printable/)

S	K	I	N	D	I	V	I	N	G	S	C	I	T	S	A	N	M	Y	G
R	E	E	D	V	A	F	O	G	N	I	C	A	R	R	A	C	E	S	E
I	E	A	L	D	H	D	I	G	C	G	N	I	L	C	Y	C	E	F	A
S	T	Y	E	H	R	T	N	N	G	I	I	T	L	N	B	T	H	A	W
I	G	N	I	I	K	S	G	I	N	R	O	L	A	B	L	P	E	H	W
N	S	B	F	A	S	S	I	C	U	O	A	E	B	N	S	A	O	T	H
N	O	O	D	A	N	O	O	A	N	B	W	Y	T	T	K	R	H	U	P
E	F	W	N	E	O	D	S	R	T	P	O	L	O	H	S	A	U	Q	S
T	T	L	A	E	W	R	A	E	M	S	O	A	O	L	L	G	H	D	M
E	B	I	K	L	B	T	K	L	G	S	O	L	F	R	H	L	O	I	M
L	A	N	C	L	O	S	B	C	N	O	H	C	U	H	D	I	R	S	G
B	L	G	A	A	A	W	S	Y	I	G	W	G	C	Y	E	D	S	E	N
A	L	L	R	B	R	J	P	C	V	A	B	Y	C	E	T	I	E	H	I
T	G	N	T	Y	D	O	A	R	I	Y	E	O	N	K	R	N	R	A	M
E	N	T	G	E	I	G	I	O	D	T	O	U	V	C	D	G	A	N	M
E	P	T	T	L	N	G	U	T	A	S	E	E	C	O	N	T	C	D	I
M	E	C	T	L	G	I	U	O	B	T	I	L	T	H	G	D	I	B	W
A	C	E	W	O	O	N	R	M	U	S	L	A	H	E	S	L	N	A	S
Y	T	P	V	V	L	G	O	D	C	R	R	I	D	T	U	E	G	L	F
D	I	K	A	P	F	R	B	A	S	E	B	A	L	L	A	H	W	L	S

Gambar 8 Word search Puzzle Kategori Hard
(Sumber: https://www.brainzilla.com/word-games/word-search/printable/)

ATHLETICS	GYMNASTICS	RUGBY
BASEBALL	HANDBALL	HOCKEY
BASKETBALL	SNOWBOARDING	SKIING
BOWLING	HORSERACING	SKINDIVING
CARRACING	SCUBADIVING	JOGGING
CYCLING	MOTORCYCLERACING	SOCCER
FOOTBALL	PARAGLIDING	SOFTBALL
GOLF	TABLETENNIS	SQUASH
SWIMMING	TRACKANDFIELD	POLO
VOLLEYBALL		

Tabel 2 Kata Tersembunyi Kategori Hard
(Sumber: https://www.brainzilla.com/word-games/word-search/printable/)

Program yang dibuat akan membaca kedua file berisi puzzle tersebut kemudian menyelesaikan word search puzzle. Puzzle yang telah selesai akan ditampilkan dengan kata-kata yang dicari berwarna merah. Hasil pengujian adalah sebagi berikut.

Gambar 9 Hasil Pengujian 1
(Sumber: Arsip Penulis)

Gambar 10 Hasil Pengujian 2
(Sumber: Arsip Penulis)

Berdasarkan hasil pengujian, pada uji pertama program berhasil menemukan seluruh 11 kata yang tersembunyi dan pada uji kedua program berhasil menemukan seluruh 28 kata tersembunyi. Posisi setiap kata yang ditemukan oleh algoritma

backtracking dapat dilihat pada tabel 3 untuk uji pertama dan tabel 4 untuk uji kedua. Pada kedua tabel tersebut letak kata memiliki format (x,y) dimana x adalah nomor baris dan y adalah nomor kolom yang keduanya dimulai dari 1. Posisi dalam tabel ditulis sesuai dengan penomoran berikut:

1. Horizontal dari kiri ke kanan.
2. Horizontal dari kanan ke kiri.
3. Vertikal dari atas ke bawah.
4. Vertikan dari bawah ke atas.
5. Diagonal dari kiri atas ke kanan bawah.
6. Diagonal dari kiri bawah ke kanan atas.
7. Diagonal dari kanan atas ke kiri bawah.
8. Diagonal dari kanan bawah ke kiri atas.

Kata	Posisi	Letak Kata Pertama	Letak Kata terakhir
BLACK	3	(10,14)	(14,14)
BLUE	3	(12,7)	(15,7)
BROWN	3	(7,8)	(11,8)
GRAY	1	(9,2)	(9,5)
GREEN	5	(8,2)	(12,6)
ORANGE	1	(8,7)	(8,12)
PINK	1	(9,10)	(9,13)
PURPLE	7	(6,10)	(11,5)
RED	5	(8,8)	(10,10)
WHITE	3	(1,9)	(5,9)

Tabel 3 Letak Kata ditemukan pada pengujian 1
(Sumber: Arsip Penulis)

Kata	Posisi	Letak Kata Pertama	Letak Kata terakhir
ATHLETICS	8	(20,16)	(12,8)
BASEBALL	1	(20,8)	(20, 15)
BASKETBALL	6	(13,5)	(4,14)
BOWLING	3	(6,3)	(12,3)
CARRACING	2	(2,17)	(2,9)
CYCLING	2	(3,17)	(3,11)
FOOTBALL	4	(10,14)	(3,14)
GOLF	3	(17,6)	(20,6)
GYMNASTICS	2	(1,20)	(1,11)
HANDBALL	3	(13,19)	(20,19)
HOCKEY	4	(17,15)	(12,15)
HORSERACING	3	(9,18)	(19,18)
JOGGING	3	(13,7)	(19,7)
MOTORCYCLERACING	4	(18,9)	(3,9)
PARAGLIDING	3	(5,17)	(15,17)
POLO	1	(8,11)	(8,14)
RUGBY	7	(10,15)	(14,11)
SCUBADIVING	4	(20,10)	(10,10)
SKIING	2	(5,7)	(5,2)
SKINDIVING	1	(1,1)	(1,10)
SNOWBOARDING	3	(6,6)	(17,6)
SOCCER	5	(9,11)	(14,16)
SOFTBALL	3	(6,2)	(13,2)
SQUASH	2	(8,20)	(8,15)
SWIMMING	4	(18,20)	(11,20)
TABLETENNIS	4	(14,1)	(4,1)
TRACKANDFIELD	4	(14,4)	(2,4)
VOLLEYBALL	4	(19,5)	(10,5

Tabel 4 Letak Kata ditemukan pada pengujian 2
(Sumber: Arsip Penulis)

V. KESIMPULAN

Algoritma backtracking dan struktur data trie dapat digunakan untuk mencari seluruh solusi dari sebuah word seach Puzzle. Hasil penerapan algoritma tersebut berhasil menemukan seluruh kata yang dicari pada sebuah puzzle dengan kategori medium berukuran 15x15 dengan 11 kata tersembunyi dan sebuah puzzle dengan kategori hard berukuran 20x20 dengan 28 kata tersembunyi.

TAUTAN

Video di youtube dapat diakses pada:

https://youtu.be/-HBoy2Qbd4Y

Kode Source program dapat diakses pada:

https://github.com/Saladin21/Word_Search

UCAPAN TERIMA KASIH

Pertama-tama penulis mengucapkan rasa syukur kepada Allah SWT karena atas berkat dan rahmat-Nya penulis dapat menyelesaikan makalah ini. Penulis juga mengucapkan terima kasih kepada keluarga yang selalu memberikan semangat selama penulisan makalah ini. Terakhir penulis berterima kasih kepada seluruh dosen pengampu mata kuliah Strategi Algoritma, Dr. Ir. Rinaldi Munir, Dr. Eng. Rila Mandala, Dr. Nur Ulfa Maulidevi S.T, M.Sc, dan Prof. Dwi Hendratmo

Widiyantoro, Ph.D atas ilmu yang telah diberikan sehingga penulis dapat menyelesaikan makalah ini

REFERENSI

[1] https://informatika.stei.itb.ac.id/~rinaldi.munir/Stmik/2020-2021/Algoritma-backtracking-2021-Bagian1.pdf, diakses tanggal 10 Mei 2021.

[2] https://www.collinsdictionary.com/dictionary/english/wordsearch, diakses tanggal 10 Mei 2021.

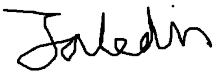
[3] https://towardsdatascience.com/implementing-a-trie-data-structure-in-python-in-less-than-100-lines-of-code-a877ea23c1a1, diakses tanggal 10 Mei 2021.

[4] https://www.geeksforgeeks.org/trie-insert-and-search/, diakses tanggal 10 Mei 2021.

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, 11 Mei 2021



Ahmad Saladin
13519187