Penerapan Algoritma *Brute Force* dan *Backtracking* dalam mencari solusi *Tower of Hanoi*

Dito Rifadli Febrian - 1301213518
Muhammad Hermawan Alghozy - 1301213473
Aulia Khusnul Arief - 1301213425
Program Studi Informatika
Fakultas Informatika
Universitas Telkom, Jalan Telekomunikasi 1 Bandung
ditorifadlifebrian@student.telkomuniversity.ac.id
igoghozy@student.telkomuniversity.ac.id
auliakarief@student.telkomuniversity.ac.id

Abstrak - Makalah ini dibuat bertujuan untuk menganalisis dan memecahkan suatu masalah yang dikemukakan oleh Eduard Lucas, seorang ahli matematika Perancis pada tahun 1883 yakni Menara Hanoi *(Tower of Hanoi)*. Menara Hanoi adalah topik kuno yang menarik. Dalam laporan ini kami menggunakan metode *Brute Force* dan *Backtracking* untuk mencari sebuah solusi dari masalah tersebut. Hasil analisis akan dilaporkan dan ditunjukkan bahwa metode yang diusung dapat menemukan sebuah solusi untuk masalah Menara Hanoi.

Kata Kunci: Menara Hanoi, *Tower of Hanoi*, *Brute Force*, *Backtracking*

Abstract - This paper aims to analyze and solve a problem proposed by Eduard Lucas, a French mathematician in 1883, namely the Tower of Hanoi (Tower of Hanoi). The Tower of Hanoi is an interesting ancient topic. In this report we use Brute Force and Backtracking methods to find a solution to the problem. The analysis results will be reported, and it is shown that the proposed method can find a solution to the Tower of Hanoi problem.

Keywords: *Menara Hanoi*, Tower of Hanoi, Brute Force, Backtracking

I. PENDAHULUAN

A. Latar Belakang

Perkembangan teknologi informasi dan komunikasi sangatlah pesat, sehingga membawa suatu perubahan signifikan dalam berbagai aspek kegiatan manusia. Penerapan suatu sistem informasi masuk ke dalam berbagai sektor, salah satunya perusahaan manufaktur yang telah menjadi tren yang tak terhindarkan. Dalam lingkungan yang semakin ketat dan kompetitif ini, suatu perusahaan perlu memasukkan peran teknologi informasi terbaru untuk meningkatkan kualitas operasional mereka.

Salah satu tantangan yang dihadapi adalah oleh perusahaan manufaktur adalah pengelolaan rantai pasok yang kompleks dan penjadwalan produksi yang efektif. Dalam konteks ini, penyelesaian masalah Menara Hanoi telah menjadi subjek amsalah yang menarik untuk diteliti sebagai bentuk penerapan teknologi dan algoritma yang efisien.

B. Tujuan Penelitian

Penelitian ini bertujuan untuk menyajikan pendekatan strategi aloritma dalam mencari solusi untuk masalah Menara Hanoi. Kami akan menggunakan metode *Brute Force* dan *Backtracking*. Melalui penelitian ini, kami berharap dapat menunjukkan bahwa strategi algoritma yang kami pelajari di kelas dapat menjadi solusi dan metode yang cukup efektif dalam maalah Menara Hanoi. Hasil penelitian ini dapat memberikan wawasan baru dalam mengoptimalkan menyelesaikan masalah yang ada.

C. Metode Penelitian

Dalam penelitian ini, kami akan menggunakan metode *Brute Force* dan *Backtracking* dan akan mengimplementasikan strategi algoritma yang kami pelajari di kelas untuk mewakili keadaan dan pergerakan pada Menara Hanoi. Selain itu, kami

akan mencari solusi yang optimal. Laporan ini akan dilakukan menggunakan sampel kasus yang berbeda untuk menguji metode yang diusung.

II. DASAR TEORI

A. Pengertian Menara Hanoi *(Tower of Hanoi)*

Menara Hanoi adalah sebuah permainan matematis atau teka-teki. Permainan ini terdiri dari tiga tiang (terdiri dari tiang asal, tiang bantu, dan tiang tujuan) dan sejumlah cakram dengan ukuran berbeda-beda yang bisa dimasukkan ke tiang mana saja. Permainan dimulai dengan cakram-cakram yang tertumpuk rapi berurutan berdasarkan ukurannya dalam tiang asal, cakram terkecil diletakkan teratas, sehingga membentuk kerucut.

Tujuan dari teka-teki ini adalah untuk memindahkan seluruh tumpukan ke tiang yang lain, mengikuti aturan berikut:

1. Hanya satu cakram yang boleh dipindahkan dalam satu waktu.
2. Setiap perpindahan berupa pengambilan cakram teratas dari satu tiang dan memasukkannya ke tiang lain, di atas cakram lain yang mungkin sudah ada di tiang tersebut.
3. Tidak boleh meletakkan cakram yang lebih besar di atas cakram lain yang lebih kecil.

Teka-teki ini ditemukan oleh Édouard Lucas, ahli matematika Perancis di tahun 1883. Ada sebuah legenda tentang candi Indian yang berisi ruang besar dengan tiga tiang yang dikelilingi 64 cakram emas. Pendeta Brahma, melaksanakan tugas dari peramal di masa lalu, sesuai dengan aturan teka-teki ini. Menurut legenda ini, bila tekateki ini diselesaikan, dunia akan kiamat. Tidak jelas benar apakah Lucas menemukan legenda ini atau terinspirasi olehnya. Dengan berbagai macam metode penghitungan dan penelitiannya, permainan menara hanoi ini akhirnya terpecahkan dengan solusi: jika terdapat sejumlah *n* cakram pada permainan menara hanoi, maka banyaknya perpindahan yang terjadi (tetap mengikuti aturan permainan) hingga semua *n* cakram sampai di tiang tujuan adalah $2^n - 1$ perpindahan (Rinaldi Munir. 2004).

B. Pengertian *Brute Force*

Brute force adalah sebuah pendekatan yang lempeng *(straightforward)* untuk memecahkan suatu masalah, biasanya didasarkan pada pernyataan masalah *(problem statement)* dan definisi konsep yang dilibatkan.

Karakteristik Algoritma Brute Force:

- Algoritma brute force umumnya tidak "cerdas" dan tidak mangkus, karena ia membutuhkan jumlah langkah yang besar dalam penyelesaiannya. Kadangkadang algoritma brute force disebut juga algoritma naif (naïve algorithm).
- Algoritma brute force seringkali merupakan pilihan yang kurang disukai karena ketidakmangkusannya itu, tetapi dengan mencari pola-pola yang mendasar, keteraturan , atau trik-trik khusus, biasanya akan membantu kita menemukan algoritma yang lebih cerdas dan lebih mangkus.
- Untuk masalah yang ukurannya kecil, kesederhanaan brute force biasanya lebih diperhitungkan daripada ketidakmangkusannya. Algoritma brute force sering digunakan sebagai basis bila membandingkan beberapa alternatif algoritma yang mangkus.
- Meskipun brute force bukan merupakan teknik pemecahan masalah yang mangkus, namun teknik brute force dapat diterapkan pada sebagian besar masalah. Agak sukar menunjukkan masalah yang tidak dapat dipecahkan dengan teknik brute force. Bahkan ada masalah yang hanya dapat dipecahkan secara brute force. Beberapa pekerjaan mendasar di dalam komputer dilakukan secara brute force, seperti menghitung jumlah dari n buah bilangan, mencari

C. Pengertian *Backtracking*

Algoritma Backtracking dapat dimaknai menjadi 2 hal, yaitu:

a. Metode penyelesaian persoalan (optimasi maupun nonoptimasi) yang mangkus, terstruktur, dan sistematis.

b. Sebuah fase pada proses traversal pada algoritma DFS *(Depth-First Search)*

Lalu seperti apakah algoritma DFS? Algoritma DFS merupakan algoritma yang menyelesaikan suatu persoalan dengan merepresentasikannya dalam bentuk graf dan melakukan proses traversal dimana algoritma DFS akan mengunjungi tiap simpul yang ada pada graf secara sistematik. Pada algoritma DFS, penelusuran simpul dilakukan secara mendalam *(Depth)*. Dalam proses penyelesaian persoalan, representasi persoalan

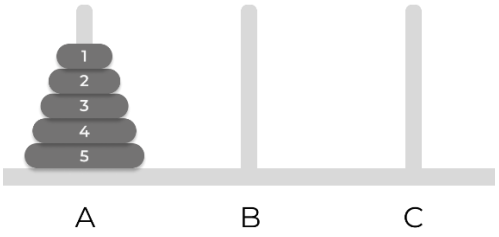
dalam graf memiliki dua pendakatan, yaitu graf statis dan graf dinamis. Graf statis merupakan graf yang sudah tersedia sebelum proses pencarian dilakukan sedangkan graf dinamis merupakan graf yang akan terbentuk saat proses pencarian dilakukan. Pada algoritma DFS, pencarian dimulasi dari simpul akar root dengan algoritma sebagai berikut:

1. Kunjungi simpul *root*
2. Kunjungi salah satu simpul yang bertetangga dengan simpul root.
3. Ulangi proses DFS pada simpul tetangga tersebut.
4. Ketika mencapai satu simpul yang seluruh simpul tetangganya sudah dikunjungi, proses pencarian dirunut-balik *(backtracking)* ke simpul sebelumnya yang sudah dikunjungi yang memiliki simpul tetangga yang belum dikunjungi
5. Apabila sudah tidak lagi terdapat simpul yang dapat dikunjungi atau sudah ditemukan penyelesaian persoalan pada salah satu solusi maka proses pencarian sudah selesai.

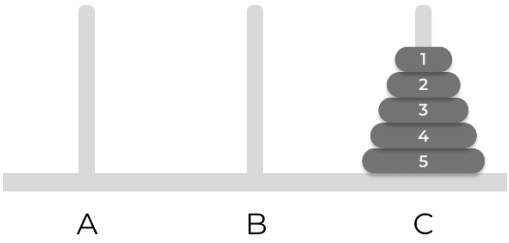
III. IMPLEMENTASI

A. Deskripsi Masalah *Tower of Hanoi*

.



Gambar 1. Initial State



Gambar 2. Goal State

Gambar 1: merupakan inisial state yang merupakan awal dari sebuah permainan. Semua cakram berada pada tiang atau menara A.

Gambar 2: merupakan goal state yang merupakan tujuan dari permainan tersebut. Semua cakram harus berpindah dari menara A menuju menara C.

Menara B sebagai menara bantuan untuk memindahkan cakram dari satu menara ke menara yang lainnya.

Masalah klasik pada Menara Hanoi yaitu seperti gambar di atas yang memiliki tiga menara, dilambangkan sebagai A, B, C, dan (n≥1), cakram-cakram dengan ukuran yang berbeda. Semua cakram pada awalnya bertumpu pada menara sumber dengan urutan cakram terbesar di bagian bawah, kedua terbesar di atasnya, dan seterusnya, dengan cakram terkecil di atasnya.

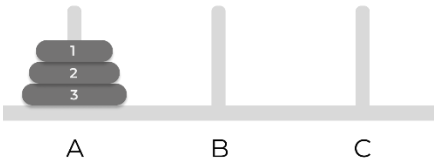
Tujuannya adalah untuk memindahkan cakram dari menara dari menara sumber ke menara tujuan, dengan aturan yang sudah ditentukan yaitu:

1. Hanya satu cakram yang boleh dipindahkan dalam satu waktu.
2. Setiap perpindahan berupa pengambilan cakram teratas dari satu manara dan memasukkannya ke menara lain, di atas cakram lain yang mungkin sudah ada di tiang tersebut.
3. Tidak boleh meletakkan cakram di atas cakram lain yang lebih kecil. Artinya yang besar selalu dibawah yang kecil.

Dengan aturan tersebut, bagaimana caranya agar semua cakram dapat berpindah dari menara awal ke menara tujuan.

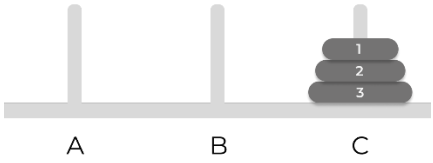
B. Solusi Menggunakan *Brute Force*

Dengan menggunakan algoritma *backtracking*, contohnya kita menggunakan 3 cakram seperti pada gambar di bawah ini:



Gambar 3. Initial State (backracking)

Seluruh cakram pada Menara A harus berpindah ke Menara C seperti gambar di bawah:



Gambar 4. Goal State State (backracking)

Algortima *Brute Force* mengandung konsep rekursif. Langkah penyelesaian dapat dinyatakan dalam relasi berulang yang juga dapat diselesaikan dengan notasi aljabar berikut:

$$
\begin{aligned}
S_n &= 2S_{n-1} + 1 \\
&= 2(2S_{n-2} + 1) + 1 = 2^2 S_{n-2} + 2 + 1 \\
&= 2^2(2s_{n-3} + 1) + 2 + 1 = 2^3 S_{n-3} + 2^2 + 2 + 1 \\
&= 2^3(2S_{n-4} + 1) + 2^2 + 2 + 1 \\
&= 2^4 S_{n-4} + 2^3 + 2^2 + 2 + 1 \\
&= ..... \\
&= 2^{n-2} S_2 + 2^{n-3} + ... + 2^2 + 2 + 1 \\
&= 2^{n-2} S_2 + \frac{2^{n-2} - 1}{2 - 1} \\
&= 2^{n-2} S_2 + 2^{n-2} - 1 \\
&= 2^{n-2} x3 + 2^{n-2} - 1 \\
&= 4x2^{n-2} - 1 \\
&= 2^{n-1}
\end{aligned}
$$

Contoh:

Untuk 5 Cakram yaitu $n = 5$ dibutuhkan $2^5 - 1 = 31$ langkah minimum.

Code program menggunakan *Python:*

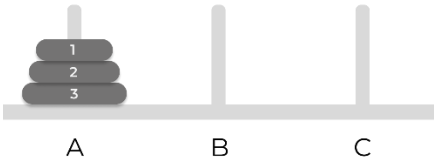
```
def TowerOfHanoi(n, source,
destination, auxiliary):

    if n > 0:

       TowerOfHanoi(n-1, source,
auxiliary, destination)

       print(f"Pindahkan Cakram {n} dari
{source} ke {destination}")

       TowerOfHanoi(n-1, auxiliary,
destination, source)

n = 3

TowerOfHanoi(n, 'A', 'C', 'B')
```

Menggunakan Algoritma *Brute Force* pada code di atas, berikut langkah-langkah dalam menyelesaikan solusi *Tower of Hanoi* dengan n cakram 3*:*

- Pindahkan Cakram 1 dari A ke C
- Pindahkan Cakram 2 dari A ke B
- Pindahkan Cakram 1 dari C ke B
- Pindahkan Cakram 3 dari A ke C
- Pindahkan Cakram 1 dari B ke A
- Pindahkan Cakram 2 dari B ke C
- Pindahkan Cakram 1 dari A ke C

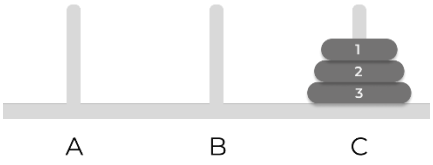
C. Solusi Menggunakan *Backtracking*

Dengan menggunakan algoritma *backtracking*, contohnya kita menggunakan 3 cakram seperti pada gambar di bawah ini:



Gambar 5. Initial State (backracking)

Seluruh cakram pada Menara A harus berpindah ke Menara C seperti gambar di bawah:

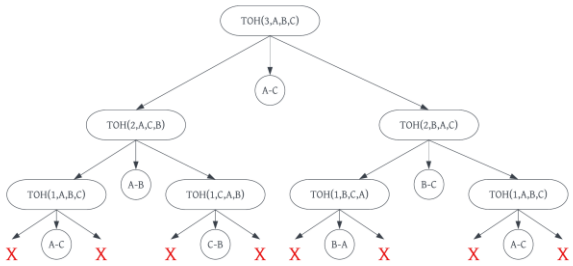


Gambar 6. Goal State State (backracking)

Perhatikan pseducode berikut ini:

```
Procedure TOH(n, A, B, C)

If n > 0 then

   TOH(n-1, A, C, B)

   Output("pindahkan cakram n dari A ke
C")

   TOH(n-1, B, A, C)

 Endif

Endprocedure
```

Dengan algoritma di atas, maka dapat di buat graf seperti di bawah ini:

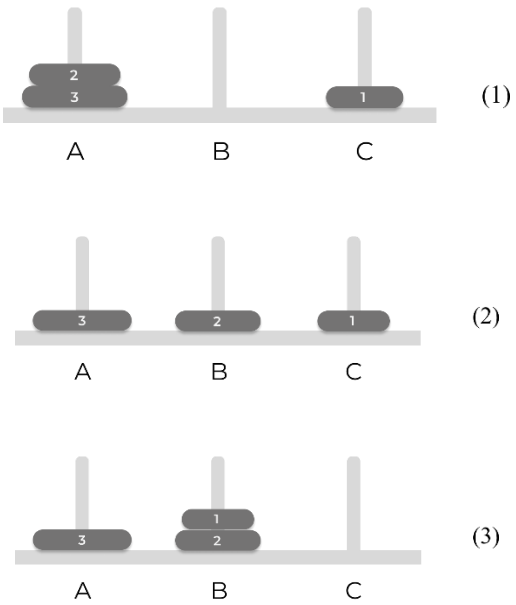


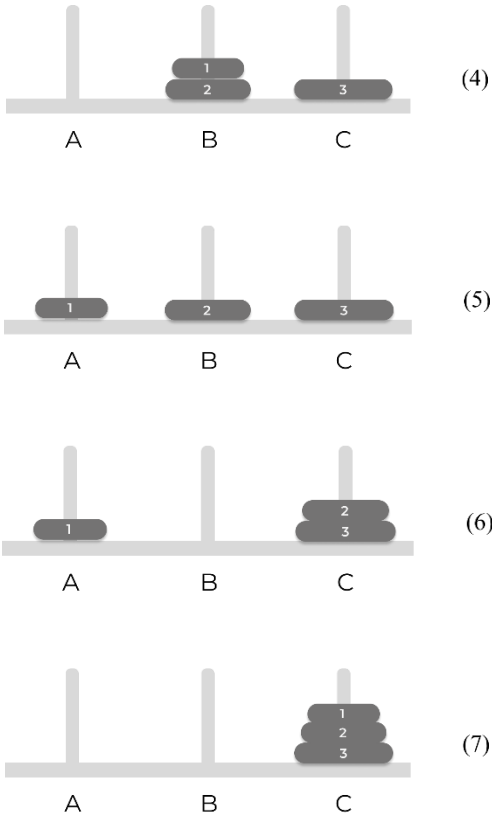
Gambar 7. Graf backtracking

Perhatikan graf di atas, dengan menggunakan DFS, maka di dapatkan hasil {(A-C), (A-B), (C-B), (A-C), (B-A), (B-C), (A-C)}.

Karena *backtracking*, maka di telusuri secara mendalam, ketika tidak memungkinkan untuk lanjut mendapatkan solusi, maka kembali ke simpul yang sudah dilalui.

Berikut langkah-langkah menggunakan gambar *Tower of Hanoi:*





Gambar 8. Langkah-langkah backtracking

IV. ANALISIS

Dengan menggunakan korelasi rekurens, jumlah langkah yang dibutuhkan untuk memindahkan n buah cakram dari menara asal ke menara tujuan pada permainan Menara Hanoi adalah $f(n) = (2^n - 1)$

a. Kompleksitas waktu menggunakan *Brute Force* yaitu $T(n) = O(2^n - 1)$ atau $O(2^n)$
b. Kompleksitas waktu menggunakan *Backtracking* yaitu $O(2^n)$

Pendekatan brute force dan backtracking akan memiliki kompleksitas waktu yang sama atau eksponensial, di mana n adalah jumlah disk. Hal ini karena dalam kasus terburuk, kedua algoritma perlu memeriksa semua kemungkinan pergerakan atau solusi, dan jumlah kemungkinan pergerakan atau solusi bertambah secara eksponensial dengan jumlah disk.

Running Time menggunakan brute force dengan n cakram 3 yaitu 3153672ns sedangkan Backtracking yaitu 2714428ns. Running time algoritma brute force lebih lama daripada backtracking

V. KESIMPULAN

Dari hasil pengujian, maka dapat disimpulkan beberapa hal sebagai berikut:

1. Algoritma *Brute Force* dan *Backtracking* mampu memberikan solusi pada permasalahan menara Hanoi *(Tower of Hanoi)*.
2. Algoritma *Brute force* dan *Backtracking* memberikan solusi yang sama yaitu memiliki kompleksitas waktu yang sama.

REFERENSI

[1] Munir. Rinaldi., *Algoritma dan Pemrograman Buku I (revsii ke-3)*. Informatika, 2004.

[2] Rinaldi Munir, *"Strategi Algoritmik"*. Teknik Informatika ITB, 2007

[3] G.S. Wulandari, S. Saadah. *Pengantar Strategi Algoritma*, KBM Indonesia, 2021.

LAMPIRAN

Code:

https://github.com/ditorifadli/tower-of-hanoi

Power Point:

https://telkomuniversityofficial-my.sharepoint.com/:p:/g/personal/ditorifadlifebrian_student_telkomuniversity_ac_id/EdoA2seJQ5hAmoNXz6NSQj8B3oPmE23Nhda-C6WIiIBiQw?e=6YjLR6

Gambar:

https://www.figma.com/file/IB5dcsZhTjPs5a9EIqizUs/Tower-Of-Honai?type=design&node-id=0%3A1&t=ZCVuMg1fNzxQKdoQ-1