Penggunaan Algoritma Backtracking Persoalan N-Ratu dalam Persoalan Peletakkan Kursi dalam Perkumpulan yang Harus Saling Berjauhan

Naufal Arfananda Ghifari 13518096
Program Studi Teknik Informatika
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung, Jalan Ganesha 10 Bandung
E-mail (gmail): 13518096@std.stei.itb.ac.id

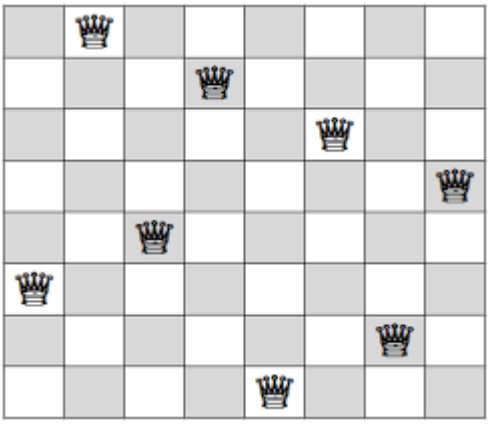
***Abstract*—Permasalahan N-Ratu merupakan salah satu permasalahan yang dapat diselesaikan dengan algoritma Backtracking. Permasalahan N-Ratu ini dapat digunakan dalam representasi lain contohnya adalah peletakkan kursi yang dikhususkan saling menjauhi satu sama lain.**

Keywords—BackTracking, N-Ratu, Kursi

I. PENDAHULUAN

Kemunculan virus COVID-19 di awal tahun 2020, meresahkan hampir seluruh masyarakat di seluruh dunia. Penyebaran yang cepat, luas dan massif dari virus yang diduga pertama kali muncul di Wuhan, China ini sudah masuk ke berbagai negara. Angka kematian virus ini pun terbilang tinggi. Sehingga setiap negara mempunyai caranya sendiri untuk bertahan dari kondisi ini. Pembatasan interaksi antara manusia dilakukan di semua tempat. Perkumpulan manusia pun hanya dapat dilakukan dalam skala kecil. Walaupun perkumpulan manusia hanya diperbolehkan dengan jumlah orang beberapa saja , mereka harus mempunyai jarak satu sama lain. Salah satu permasalahan yang perlu diatur adalah peletakkan tempat duduk orang-orang yang berkumpul agar memiliki jarak satu sama lain.

Dalam menangani permasalahan ini, penulis terinspirasi oleh salah satu materi yang penulis pelajari di mata kuliah Strategi Algoritma. Kasus ini dapat diatasi dengan memodifikasi persoalan N-Ratu pada algoritma *BackTracking*. Pada persoalan N-Ratu ini tidak boleh adanya pion Ratu pada papan catur yang berada dalam garis maupun diagonal yang sama. Dengan mengganti abstraksi papan catur disini sebagai lahan yang akan digunakan tempat duduk yang akan direpresentasikan dalam petak seperti catur berdasarkan nilai panjang dan lebarnya. Serta letak Ratu ini sebagai peletakkan kursi, diharapkan akan dihasilkan peletakkan kursi yang merupakan solusi dalam menangani permasalah ini

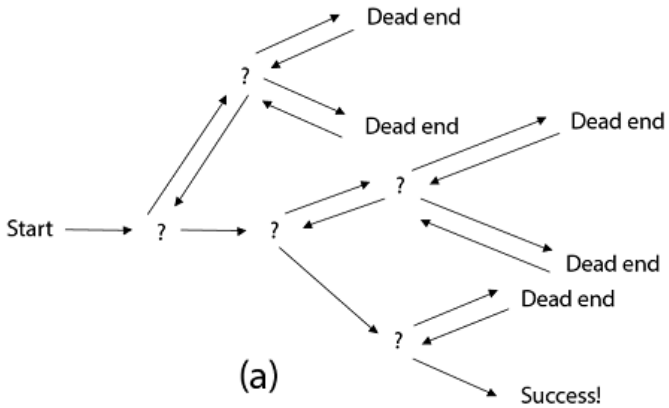


Gambar 1. Contoh persoalan N-Ratu (sumber : https://codepumpkin.com/n-queen-problem/ diakses pada 3 Mei 2020)

II. TEORI DASAR ALGORITMA BACKTRACKING

A. *Definisi algoritma backtracking*

Backtracking merupakan suatu metode pemecahan masalah yang mangkus, terstruktur, dan sistematis yang merupakan sebuah fase di dalam algoritma *Depth First Search*(DFS). Pada algoritma ini, hanya pilihan yang mengarah ke solusi yang dieksplorasi sementara pilihan-pilihan yang sudah dapat dipastikan tidak akan mengarah ke solusi akan dipangkas. Algoritma ini pertama kali diperkenalkan oleh D.H Lehmer pada tahun 1950.



Gambar 2. Ilustrasi salah satu algoritma Backtracking (sumber : https://www.javatpoint.com/backtracking-introduction diakses pada 3 Mei 2020)

B. *Properti Umum Metode Backtracking*

Berikut merupakan properti umum metode *backtracking* :

1. Solusi persoalan

   Solusi dinyatakan sebagai vector dengan n-tuple :

   $$X = (x_1, x_2, \ldots, x_n),\ x_i \in Si$$

   Dimungkinkan pula $S_1 = S_2 = \ldots = S_n$

   Contoh : $S_i = \{0,1\}$, $x_i = 0$, atau 1

2. Fungsi Pembangit nilai x_i

   Fungsi pembangkit dinyatakan sebagai predikat : T(k)

   Fungsi ini membangkitkan nilai x_k yang merupakan komponen vektor solusi.

3. Fungsi Pembatas

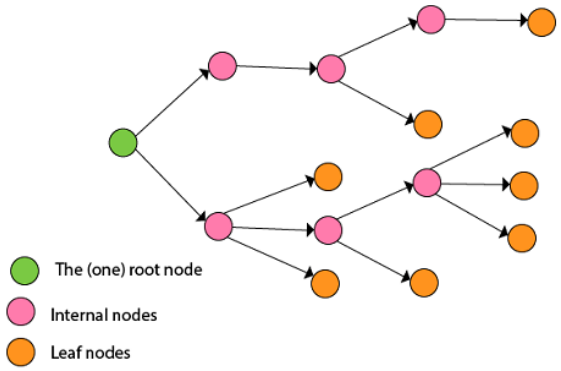
   Fungsi pembatas pada algoritma ini dinyatakan sebagai predikat $B(x_1, x_2, \ldots, x_k)$. B ini bernilai benar jika B mengarah ke solusi. Ketika B bernilai benar, maka akan dilanjutkan pembangkitan untuk x_{k+1}, sedangkan jika B bernilai salah, maka $(x_1, x_2, \ldots, x_k)$ akan dibuang.

C. *Konsep Algoritma BackTracking*

Pada Algoritma *Backtracking* ini semua kemungkinan solusi persoalan disebut ruang solusi. Ruang solusi direpresentasikan dalam struktur pohon. Tiap simpul pohon merupakan status persoalan sementara sisi/cabang pohon merupakan nilai-nilai x_i. Lintasan dari akan ke daun menyatakan solusi yang mungkin. Pengorganisasian pohon ruang solusi diacu pohon ruang status.

Sebuah pohon merupakan kumpulan dari simpul dan busurnya yang tidak mempunyai sirkuit. Ada tiga macam simpul yang terdapat pada pohon :

1. Simpul akar
2. Simpul dalam
3. Simpul daun



Gambar 3 Contoh Pohon. Simpul hijau merupakan simpul akar, merah muda merupakan simpul dalam, jingga merupakan simpul daun. (sumber : https://www.javatpoint.com/backtracking-introduction diakses pada 3 Mei 2020)

Dalam representasi pohon, *backtracking* ini merupakan pencarian yang digambarkan pada pohon untuk mencapai suatu simpul daun tertentu.

Seperti yang sudah disinggung sebelumnya bahwa algoritma backtracking mengacu pada aturan pencarian DFS. Berikut merupakan prinsip algoritma *backtracking*.

1. Simpul-simpul yang sudah dibangkitkan pada algoritma ini disebut simpul hidup. Simpul hidup ini diperluas sehingga lintasan yang dibangun bertambah banyak. Simpul hidup yang sedang diperluas ini dinamakan simpul-E.
2. Jika lintasan yang dibentuk tidak mengarah pada solusi, maka simpul-E yang terkait akan dibunuh sehingga menjadi simpul mati. Pembunuhan simpul ini diatasi oleh sebuah fungsi yang disebut fungsi pembatas. Simpul yang sudah dibunuh ini tidak akan diperluas lagi.
3. Ketika pembentukan lintasan berakhir dengan simpul mati, maka proses pencarian *backtrack* ke simpul aras sebelumnya.
4. Kemudian lanjutkan dengan membangkitkan simpul lainnya. Sehingga simpul yang dibangkitkan ini menjadi simpul-E yang baru
5. Pencarian ini terhenti ketika telah sampai pada status tujuan.

D. *Kompleksitas Algoritma BackTracking*

Setiap simpul dalam pohon ruang status saling berkait dengan sebuah pemanggilan rekursif. Jika jumlah simpul dari pohon ruang status adalah 2^n atau n!, maka untuk kasus terburuk algoritma ini memiliki kompleksitas waktu $O(p(n)\ 2^n)$ atau O(Q(n) n!). Dalam kasus ini p(n) dan q(n) adalah sebuah polinom derajat n yang menyatakan waktu komputasi setiap simpulnya.

E. Persoalan N-Ratu

Terdapat suatu papan catur yang ukurannya N x N dan N buah ratu. Permasalahan yang diajukan adalah bagaimana cara kita dapat menempatkan N buah ratu ini pada petak-petak papan catur ini sedemikian rupa sehingga tidak ada satupun ratu yang berada pada baris, kolom maupun diagonal yang sama.

Penyelesaian persoalan ini terdapat beberapa pendekatan.

1. *Brute Force* 1

   Mencoba semua kemungkinan solusi penempatan N buah ratu tersebut pada sebuah papan catur N x N.

   Pada kasus ini kompleksitas algoritma yang didapat adalah mengikuti aturan kombinasi C (N^2, N). Untuk kasus N = 8, maka kompleksitasnya C (64,8) = 4.426.165.368 kemungkinan solusi

2. *Brute Force* 2

   Meletakkan masing-masing ratu hanya pada baris-baris yang berbeda. Sehingga untuk papan catur N x N, kita mencoba menempatkan ratu mulai dari kolom 1 hingga N.

   Penyelesaian persoalan ini memiliki kompleksitas algoritma mengikuti perpangkatan N^N. Sehingga untuk kasus N = 8, kompleksitasnya adalah 8^8 = 16.777.216

3. *Brute Force* 3

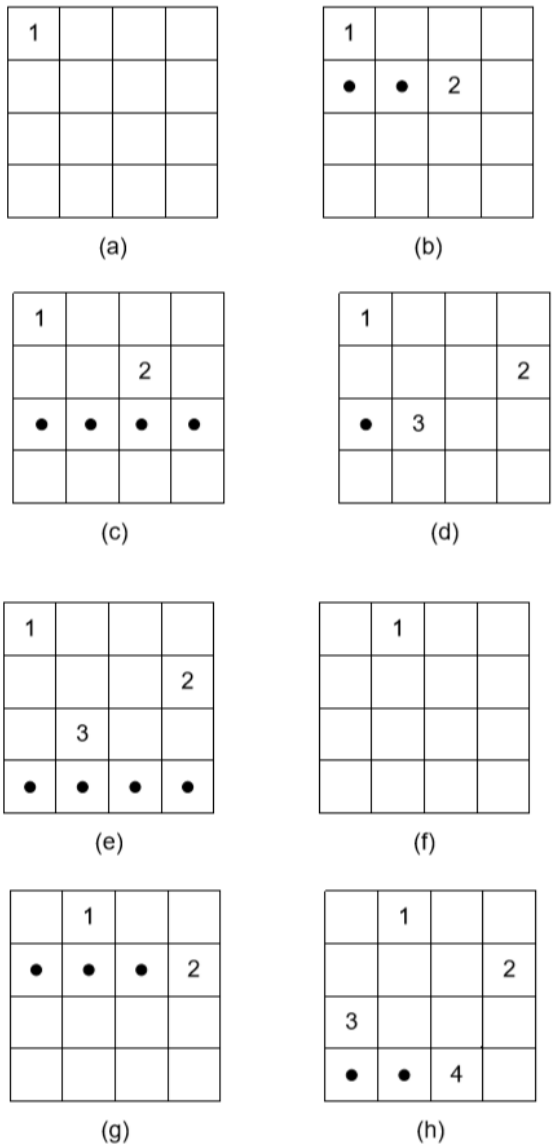
   Pada Brute Force 3, solusi dinyatakan dalam N-tupple.

   $$X =(x_1, x_2, .., x_n)$$

   Dapar dilihat bahwa vektor solusi merupakan sebuah permutasi bilangan 1 hingga N atau dapat ditulis juga sebagai P(N,N). Pada kasus N =8 nilanya setara dengan P(8,8) = 40.320 buah.

4. *Backtracking*

   Algoritma dengan metode *backtracking* memperbaiki yang algotima Brute Force 3. Ruang solusinya merupakan permutasi angka-angka 1 hingga N. Setiap permutasi ini dinyatakn dengan lintasan dari akar ke daun. Sisi-sisi pada pohon ini dilabeli nilai x_i.



Gambar 4 Ilustrasi penyelesaian persoalan N-Ratu dengan backtracking dan N=4 . (sumber : Slide kuliah IF2211 Rinaldi Munir)

F. Algoritma Backtracking pada persoalan N-Ratu

Berikut kondisi-kondisi yang terjadi pada persoalan ini.

1. Tinjau dua posisi ratu pada (i,j) dan (k,l)
2. Jika dua buah ratu terdapat pada baris yang sama maka i =k
3. Jika dua buah ratu terletak pada kolom yang sama maka j=l
4. Dua buah ratu terletak pada suatu diagonal yang sama apabila
   |i-j | = | k-l |

Berikut algortima umum N-Ratu secara *backtracking*

```
{fungsi Tempat}
```

```
function Tempat(input k: integer) → Boolean
{true jika ratu dapat ditempatkan, sebaliknya tidak}
Deklarasi
  i : integer
  stop : boolean
Algoritma
  Kedudukan ← true
  i ← 1
  stop ← false
  while (i<k) and (not stop) do
    if(x[i] = x[k] or ABS (x[i]-x[k] )= ABS (i-k)){tidak dapat
ditempatkan}
    then
      kedudukan ← false
    else
      i← i+1
    endif
  endwhile
  return kedudukan
```

```
{prosedur N_Ratu_R}
procedure N_Ratu_R(input k: integer, input n : integer,
input/output selesai : boolean )
{Menempatkan ratu pada baris ke-k pada suatu papan catur N
x N secara backtracking dengan memanggil fungsi tempat
untuk mengecek kebolehan ratu ditempakan.}
Deklarasi
  stop : boolean
  subselesai : boolean
Algoritma
  Stop ← false
  While not stop and not selesai  do
    x[k] ← x [k] +1
    while (x[k] ≤ n and not Tempat(k)) do
      x[k] ← x[k] +1
    endwhile
    if (x[k] ≤ n )then
      if(k=n) then
        CetakSolusi(x,n)
        selesai ← true
      else
        N_Ratu_R(k,n, subselesai)
        selesai ← subselesai
      endif
    else
      stop ← true
      x[k] ← 0
    endif
  endwhile
```

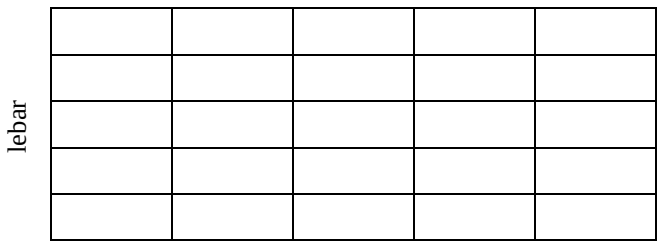
```
{program utama}
Deklarasi
  x : Array of integer [1..100]
  n : integer
Algoritma
  {inisisasi x dengan 0}
  for i ← 1 to 100 do
    x[i] ← 0
  endfor
  input(n) {menerima inputan nilai n}
  N_Ratu_R(1, n, false)
```

III. IMPLEMENTASI BACKTRACKING N-RATU PADA PERSOALAN PENGATURAN TEMPAT DUDUK

Persoalan yang diselesaikan pada kasus ini adalah bagaimana cara kita mengatur tempat duduk N orang yang berada pada suatu ruangan p x l sehingga tidak ada dua orang yang berada pada satu baris, kolom maupun diagonal.

Persoalan ini akan dicoba untuk diselesaikan dengan algoritma serupa yang digunakan dalam pemecahan masalah N-Ratu dengan cara backtracking. Sehingga hal pertama yang perlu dilakukan adalah bagaimana cara kita mengubah persoalan ini sebagai representasi N-Ratu sesederhana mungkin. Dalam hal ini penulis merepresentasikannya sebagai berikut.

1. Ratu pada konteks persoalan N-Ratu diubah menjadi kursi
2. Papan catur pada konteks ratu diubah menjadi ruangan yang dibagi Panjang dan lebarnya menjadi N bagian. Sehingga terdapat N^2 petak pada ruangan tersebut



Gambar 5 Contoh pembagian ruangan pada petak-petak (sumber : Dokumentasi penulisr\)

3. Solusi yang dicetak adalah kordinat tiap kursi

Sehingga algoritmanya sedikit mengalami perubahan dari algoritma yang tertulis di dasar teori mengenai N -Ratu

```
{fungsi tempat}

{sama persis dengan N-Ratu}

 function Tempat(input k: integer) → Boolean
{true jika kursi dapat ditempatkan, sebaliknya tidak}
```

```
Deklarasi
  i : integer
  stop : boolean
Algoritma
  Kedudukan ← true
  i ← 1
  stop ← false
  while (i<k) and (not stop) do
    if(x[i] = x[k] or ABS (x[i]-x[k] )= ABS (i-k)){tidak dapat
ditempatkan}
    then
      kedudukan ← false
    else
      i← i+1
    endif
  endwhile
  return kedudukan
```

```
{prosedur N_Kursi_R}
procedure N_Kursi_R(iinput k: integer, input n : integer,input
p:real, input l:real, input/output selesai : boolean )
{Menempatkan kursi pada baris ke-k pada suatu ruangan
berpetak N x N secara backtracking dengan memanggil fungsi
tempat untuk mengecek kebolehan kursi ditempakan.}
Deklarasi
  stop : boolean
  subselesai : boolean
Algoritma
  Stop ← false
  While not stop and not selesai  do
    x[k] <← x [k] +1
    while (x[k] ≤ n and  not Tempat(k)) do
      x[k] ← x[k] +1
    endwhile
    if (x[k] ≤ n )then
      if(k=n) then
        CetakSolusi(x,n, p, l)
        selesai ← true
      else
        N_Kursi_R(k,n, p,l, subselesai)
        selesai ← subselesai
      endif
    else
      stop ← true
      x[k] ← 0
    endif
  endwhile
```

```
{prosedur CetakSolusi}
procedure CetakSolusi(input k, n: integer, input p, l : real)
{mencetak solusi mengikuti format { (x1,y1), (x2,y2), … ,(xn,
yn)}}
print("{")
for i← 1 to N do
  print(" (", (x[i]-1) * p , ",",  (i-1) *l, ")")
  if( not (i =N))
    print(",")
  endif
endfor
print("}")
```

```
{program utama}
Deklarasi
  x : Array of integer [1..100]
  n : integer
  p,l : real
Algoritma
  {inisisasi x dengan 0}
  for i ← 1 to 100 do
    x[i] ← 0
  endfor
  input(n) {menerima inputan nilai n}
  input(p) {menerima inputan panjang ruangan}
  input(l)  {menerima inputan lebar ruangan}
  N_Kursi_R(1, n, p/n , l/n, false)
```

Sehingga ketika program ini berhasil dieksekusi akan menampilkan keluaran berupa tuple dari koordinat kursi-kursi seperti :

{ (x_1,y_1), (x_2,y_2), … ,(x_n, y_n)}

Koordinat ini yang nantinya dijadikan acuan peletakkan kursi di ruangan orang – orang yang hendak berkumpul yang menjawab persoalan bagaimana cara kita mengatur tempat duduk N orang yang berada pada suatu ruangan p x l sehingga tidak ada dua orang yang berada pada satu baris, kolom maupun diagonal.

IV. KESIMPULAN

Backtracking merupakan algoritma yang memanfaatkan pencarian DFS yang membunuh simpul yang tidak mengarah kepada solusi. Persoalan N-Ratu merupakan persoalan dimana ada sebuah papan catur berukuran NxN yang dimaksudkan untuk menempatkan N buah ratu tanpa ada dua buah ratu yang berada di baris, kolom, dan diagonal yang sama.

Persoalan peletakkan kursi yang penulis coba pecahkan dapat diselesaikan dengan merepresentasikannya persoalan ini kepada permasalahan N-Ratu dengan metode backtracking yang pernulis pelajari dari mata kuliah Strategi Algoritma. Ini

merupakan salah satu metode yang dapat digunakan untuk menyelesasikan persoalan ini, dengan mengesampingkan metode lainnya yang mungkin juga dapat dicoba.

Persoalan ini direpresentasikan dengan N-Ratu dengan mengubah beberapa konsep N-ratu yang awalnya ratu menjadi kursi yang ditempatkan, dan papan catur sebagai ruangan yang berpetak N2 dengan membagi masing-masing lebar dan Panjang ruangannya dengan N. Penyelesaian ini membuat penulis sadar bahwa ketika kita dihadapi suatu persoalan akan lebih mudah menggunakan algoritma yang sudah kita kuasai sebelumnya dan merepresentasikan persoalan yang diahadapi kepada algoritma yang sudah dipelajari dibandingkan mendesain dan membangun algoritma penyelesaian dari awal. Keuntungannya, sebagai pemrogram kita dapat menghemat waktu dan mengurangi biaya desain dan pembangunan.

VIDEO LINK AT YOUTUBE

https://youtu.be/lBDChsoS-Dw

UCAPAN TERIMA KASIH

Penulis mengucapkan puji dan syukur kepada Tuhan Yang Maha Esa sehingga penulis dapat menyelesaikan makalah berjudul ” Penggunaan Algoritma Backtracking Persoalan N-Ratu Dalam Persoalan Peletakkan Kursi dalam Perkumpulan yang Harus Saling Berjauhan”. Penulis juga berterimakasih pada dosen pengajar IF2111 Straregi Algoritma, Dr. Ir. Rinaldi Munir, M.T, atas bimbingan beliau dalam memberikan pelajaran dan Pendidikan pada semester ini sehingga penulis dapat mencoba mengimplementasikan salah satu ilmu yang dipelajari dari beliau dengan terciptanya makalah ini. Penulis juga berterimakasih kepada semua orang di sekitar penulis yang turut mendukung penulis untuk menyelesaikan makalah ini.

REFERENSI

[1] Munir, Rinaldi. 2005. ”Diktat Kuliah IF 2211 Strategi Algoritmik”. Bandung:Program Studi Teknik Informatika STEI ITB

[2] https://codepumpkin.com/n-queen-problem/

[3] https://www.javatpoint.com/backtracking-introduction

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, 3 Mei 2020

Naufal Arfananda Ghifari
13518096