# Penerapan Algoritma Branch and Bound untuk Penentuan Jalur Wisata

Janice Laksana / 13510035
*Program Studi Teknik Informatika*
*Sekolah Teknik Elektro dan Informatika*
*Institut Teknologi Bandung, Jl. Ganesha 10 Bandung 40132, Indonesia*
*13510035@std.stei.itb.ac.id*

***Abstract*—Algoritma BFS merupakan teknik yang umum digunakan untuk melakukan tranversal pada sebuah graf. Algoritma *Branch and Bound* menggunakan skema algoritma BFS dalam membangun pohon ruang status. Algoritma ini merupakan metode pencarian secara sistematis di dalam ruang solusi yang direpresentasikan menjadi pohon ruang status. Salah satu aplikasi dari algoritma *Branch and Bound* adalah dalam menentukan jalur wisata. Penentuan jalur wisata menggunakan *Branch and Bound* bertujuan menentukan urutan tempat wisata yang menghabiskan jarak total yang paling minimum. Dengan urutan tempat wisata yang menghabiskan biaya(jarak) paling minimum, waktu berwisata yang terbatas dapat digunakan secara maksimal.**

***Kata Kunci*—*Branch and Bound, BFS, penentuan jalur wisata*.**

## I. PENDAHULUAN

Algoritma BFS merupakan teknik yang umum digunakan untuk melakukan tranversal pada sebuah graf. BFS akan diperoleh melalui pencarian dasar yang akan memproses simpul-simpul yang bertetangga dengan simpul yang sedang dikunjungi dengan menggunakan struktur data antrian (*queue*).

Algoritma *Branch and Bound* merupakan metode pencarian secara sistematis di dalam ruang solusi yang direpresentasikan menjadi pohon ruang status. Pembangunan pohon ruang status pada algoritma *Branch and Bound* menggunakan skema algoritma BFS. Perbedaannya adalah pada algoritma *Branch and Bound* pencarian simpul solusi akan berjalan lebih cepat karena setiap simpul pada pohon ruang status akan diberikan nilai ongkos (*cost*).

Pada makalah ini akan dibahas mengenai bagaimana menentukan rute perjalanan jalur wisata. Penentuan rute jalur wisata ini maksudnya adalah menentukan urutan mengunjungi tempat-tempat wisata yang ingin dikunjungi oleh pengunjung di negara terkait. Penggunaan algoritma *Branch and Bound* dalam menentukan rute perjalanan jalur wisata ini dimaksudkan agar meskipun waktu untuk berwisata di negara terkait terbatas, pengunjung dapat memanfaatkan waktu tersebut secara maksimal yakni meskipun waktu yang tersedia terbatas, dengan menggunakan algoritma *Branch and Bound* akan ditentukan urutan tempat-tempat yang ingin dikunjungi dengan jarak tempuh yang paling minimum. Dengan demikian tempat wisata yang dipilih oleh pengunjung dapat dikunjungi semua walaupun waktu yang tersedia tidak banyak. Pada makalah ini, negara yang dikunjungi adalah Hongkong.

## II. TEORI

### 2.1 Algoritma *Breadth-First Search* (BFS)

Algoritma BFS adalah teknik yang umum digunakan untuk melakukan tranversal pada sebuah graf. BFS diperoleh dari pencarian dasar yang akan memproses simpul-simpul yang bertetangga dengan simpul yang sedang dikunjungi dan menggunakan struktur antrian untuk menyimpan daftar dari simpul yang telah dikunjungi.

Dalam melakukan tranversal pada sebuah graf, BFS akan mengunjungi sebuah simpul (misal : simpul v), kemudian akan dikunjungi semua tetangga dari simpul v ini. Setelah semua simpul tetangga telah dikunjungi, simpul yang belum dikunjungi akan dikunjungi, dan kemudian akan dikunjungi semua simpul-simpul tetangga dari simpul ini, dan demikian seterusnya. Jika graf yang ditranversal oleh BFS merupakan sebuah pohon berakar, maka simpul pada aras d akan dikunjungi terlebih dahulu sebelum mengunjungi simpul pada aras d+1.

Antrian pada algoritma BFS akan digunakan untuk menyimpan simpul-simpul yang telah dikunjungi. Simpul-simpul ini perlu disimpan karena suatu saat akan diperlukan sebagai acuan untuk mengunjungi simpul-simpul yang bertetangga dengan simpul yang bersangkutan. Setiap simpul yang sudah dikunjungi masuk ke dalam antrian hanya satu kali.

**Deklarasi**

```
dikunjungi : array[1..n] of boolean
```

Elemen tabel ini akan bernilai *true* jika simpul *i* pada graf telah dikunjungi dan *false* jika simpul *i* belum dikunjungi.
Pada awalnya akan dilakukan inisialisasi tabel dikunjungi dengan nilai *false*.

```
for i←1 to n do
        dikunjungi[i] ← false
endfor
```

Yang berarti pada awalnya belum ada simpul yang dikunjungi pada graf. Diasumsikan graf yang ditranversal oleh BFS ini direpresentasikan dengan matriks ketetanggaan $A = [a_{ij}]$ dengan ukuran n x n. Nilai $a_{ij}$ akan sama dengan 1 apabila simpul *i* dan simpul *j* bertetangga dan akan bernilai 0 apabila simpul *i* dan simpul *j* tidak bertetangga.

Algoritma BFS :

```
procedure BFS(input v:integer)
{ Traversal graf dengan algoritma pencarian BFS.

  Masukan: v adalah simpul awal kunjungan
  Keluaran: semua simpul yang dikunjungi dicetak ke layar
}
Deklarasi
   w : integer
   q : antrian;

   procedure BuatAntrian(input/output q : antrian)
   { membuat antrian kosong, kepala(q) diisi 0 }

   procedure MasukAntrian(input/output q:antrian, input v:integer)
   { memasukkan v ke dalam antrian q pada posisi belakang }

   procedure HapusAntrian(input/output q:antrian,output v:integer)
   { menghapus v dari kepala antrian q }

   function AntrianKosong(input q:antrian) → boolean
   { true jika antrian q kosong, false jika sebaliknya }

Algoritma:
   BuatAntrian(q)          { buat antrian kosong }

   write(v)                { cetak simpul awal yang dikunjungi }
   dikunjungi[v]←true      { simpul v telah dikunjungi, tandai dengan
                             true}
   MasukAntrian(q,v)       { masukkan simpul awal kunjungan ke dalam
                             antrian}

  { kunjungi semua simpul graf selama antrian belum kosong }
   while not AntrianKosong(q) do
      HapusAntrian(q,v)    { simpul v telah dikunjungi, hapus dari
                             antrian }
      for w←1 to n do
         if A[v,w] = 1 then      { v dan w bertetangga }
            if not dikunjungi[w] then
               write(w)     {cetak simpul yang dikunjungi}
               MasukAntrian(q,w)
               dikunjungi[w]←true
            endif
         endif
      endfor
   endwhile
  { AntrianKosong(q) }
```

**2.1 Algoritma *Branch and Bound*(B&B)**

Algoritma *Branch and Bound*(B&B) merupakan metode pencarian secara sistematis di dalam ruang solusi yang direpresentasikan menjadi pohon ruang status. Pohon ruang status pada algoritma B&B dibangun dengan menggunakan skema algoritma BFS. Setiap simpul pada pohon ruang status akan diberi nilai ongkos (*cost*) sehingga pencarian simpul solusi pun akan berjalan lebih cepat. Simpul yang akan diekspansi berikutnya tidak lagi berdasarkan urutan pembangkitan (seperti pada BFS) akan tetapi simpul akan diekspansi berdasarkan nilai ongkos yang paling kecil di antara simpul-simpul yang hidup. Setiap simpul pada algoritma B&B akan diasosiasikan dengan nilai ongkos yang menyatakan nilai batas(*bound*). Untuk setiap simpul X, nilai batas ini dapat berupa :

1. Jumlah simpul pada upapohon X yang akan dibangkitkan sebelum simpul solusi ditemukan.
2. Panjang lintasan dari simpul X ke simpul solusi terdekat.

Untuk memilih simpul hidup yang akan menjadi simpul yang pertama masuk ke dalam antrian, simpul-simpul hidup akan diurutkan terlebih dahulu berdasarkan nilai batasnya dari yang paling kecil ke paling besar. Strategi pemilihan simpul ini disebut sebagai strategi pencarian berdasarkan biaya terkecil (*least cost search*).

Algoritma B&B adalah sebagai berikut :

1. Masukkan simpul akar ke dalam antrian Q. Apabila simpul akar merupakan simpul solusi (*goal node*) maka solusi telah ditemukan. STOP.
2. Jika Q kosong, tidak ada solusi. STOP.
3. Jika Q tidak kosong, pilih simpul *i* yang memiliki ongkos paling kecil. Jika terdapat lebih dari satu simpul *i* yang memiliki ongkos terkecil, pilih satu simpul secara acak.
4. Jika simpul *i* adalah simpul solusi, maka solusi telah ditemukan. STOP.
   Jika simpul *i* bukan merupakan simpul solusi, bangkitkan semua anak-anaknya. Apabila simpul *i* tidak memiliki anak, kembalilah ke langkah nomor 2.
5. Untuk setiap anak *j* dari simpul *i*, hitung ongkosnya, dan masukkan semua anak-anak tersebut berdasarkan urutan ongkos (dari yang paling kecil ke paling besar) ke dalam antrian Q dan ulangi langkah nomor 2.

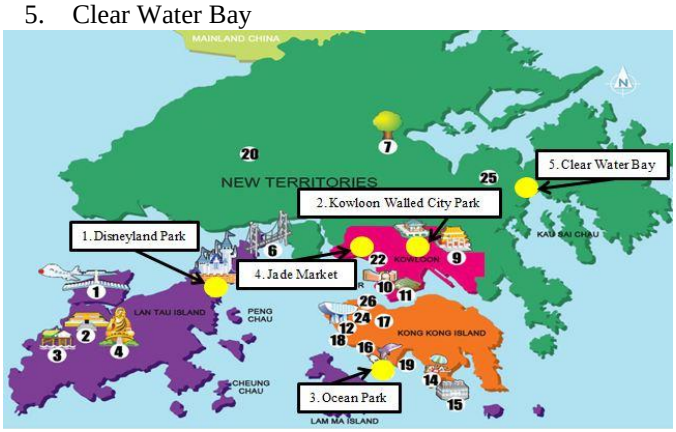
## III. PENERAPAN METODE BRANCH AND BOUND UNTUK PENENTUAN JALUR WISATA

Salah satu aplikasi penerapan dari metode *Branch and Bound* dalam kehidupan sehari-hari adalah penyusunan jalur wisata.

Pertama-tama yang harus dilakukan untuk mengaplikasikan metode *Brach and Bound* ini adalah dengan menggambarkan graf yang menyatakan tempat wisata yang ingin dikunjungi beserta dengan jarak yang harus ditempuh untuk mengunjungi tempat tersebut.

Misalkan terdapat lima tempat wisata yang ingin dikunjungi yakni :

1. Kowloon Walled City Park
2. Disneyland Park
3. Ocean Park
4. Jade Market
5. Clear Water Bay




Dengan jarak (dalam km) :

| | 1 | 2 | 3 | 4 | 5 |
|---|---|---|---|---|---|
| 1 | ∞ | 23 | 15 | 5 | 16 |
| 2 | 23 | ∞ | 36 | 21 | 36 |
| 3 | 15 | 36 | ∞ | 12 | 24 |
| 4 | 5 | 21 | 12 | ∞ | 17 |
| 5 | 16 | 36 | 24 | 17 | ∞ |

Matriks di atas merupakan matriks yang merepresentasikan jarak antartempat wisata di Hongkong. Dengan urutan nomor 1 sampai dengan 5 merepresentasikan Kowloon Walled City Park, Disneyland Park, Ocean Park, Jade Market, dan Clear

Water Bay. Pada matriks, baris ke satu dan kolom ke satu berisi nomor-nomor yang merupakan representasi tempat wisata. Pada baris ke dua kolom ke tiga terdapat angka 23 yang berarti jarak dari Kownloon Walled City Park menuju ke Disneyland Park adalah sebesar 23 km. Pada baris ke dua kolom ke tiga terdapat angka 15 yang berarti jarak dari Kownloon Walled City Park menuju ke Ocean Park adalah sebesar 15 km, dan begitu seterusnya.

Asumsi :

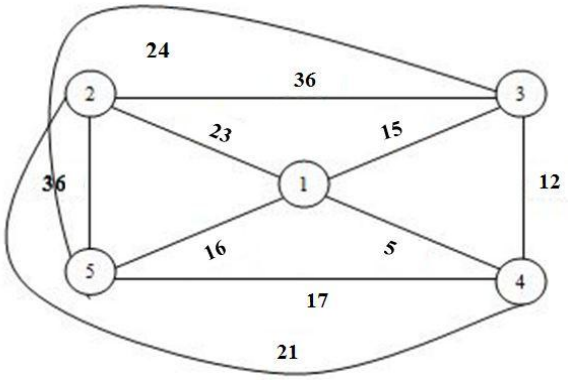
(i) G = (V,E) adalah graf lengkap yang merepresentasikan TSP.

(ii) |V| = n = jumlah simpul dalam graf G. Masing-masing simpul diberikan nomor 1, 2, 3, …, N.

(iii) Cij = bobot dari sisi i, j yang dalam kasus ini merupakan jarak antartempat wisata.

(iv) S adalah ruang solusi.

(v) |S| = (n-1)! yang merupakan jumlah dari banyaknya kemungkinan solusi.




Matriks berbobot untuk graf di atas adalah sebagai berikut :

| | | | | |
|---|---|---|---|---|
| ∞ | 23 | 15 | 5 | 16 |
| 23 | ∞ | 36 | 21 | 36 |
| 15 | 36 | ∞ | 12 | 24 |
| 5 | 21 | 12 | ∞ | 17 |
| 16 | 36 | 24 | 17 | ∞ |

Karena perjalan wisata di dalam graf melalui sisi (i,j) dengan *i* = 1, 2, 3, 4, 5 dan *j* = 1, 2, 3, 4, 5, maka mengurangkan setiap elemen pada suatu baris atau pada suatu kolom dengan konstanta *t* akan mengurangi panjang setiap perjalanan sebesar *t*. Jika *t* dipilih dari bobot minimum pada baris i (kolom j), maka mengurangkan seluruh elemen pada baris i (kolom j) akan menghasilkan sebuah nol pada baris i (kolom j) tersebut. Dengan terus melakukan proses ini, matriks bobot akan tereduksi. Jumlah total elemen pengurang dari semua baris dan kolom menjadi batas bawah dari perjalanan wisata dengan total bobot minimum. Nilai ini akan digunakan sebagai nilai ĉ untuk simpul akar pada pohon ruang status. Asumsi yang digunakan dalam kasus penentuan jalur wisata adalah :

1. Digunakan metode *Branch and Bound* TSP dengan anggapan tempat penginapan pengunjung terletak di daerah Lam Tin yang sangat dekat dengan tempat wisata Kownloon Walled City Park. Sehingga tempat yang asal adalah Kownloon Walled City Park dan tempat akhir dari perjalanan wisata ini adalah kembali pada Kownloon Walled City Park.
2. Akan digunakan struktur data *priority* queue di mana simpul yang memiliki *cost* yang sama memiliki ketentuan yakni simpul yang dibangkitkan terlebih dahulu akan memiliki prioritas yang lebih tinggi.
3. Akan dicari semua jalur wisata yang memiliki jarak yang optimum.

Lakukan reduksi baris :

| | | | | | |
|---|---|---|---|---|---|
| ∞ | 23 | 15 | 5 | 16 | R1-5 |
| 23 | ∞ | 36 | 21 | 36 | R2-21 |
| 15 | 36 | ∞ | 12 | 24 | R3-12 |
| 5 | 21 | 12 | ∞ | 17 | R4-5 |
| 16 | 36 | 24 | 17 | ∞ | R5-16 |

Setelah reduksi baris :

| | | | | |
|---|---|---|---|---|
| ∞ | 18 | 10 | 0 | 11 |
| 2 | ∞ | 15 | 0 | 15 |
| 3 | 24 | ∞ | 0 | 12 |
| 0 | 16 | 7 | ∞ | 12 |
| 0 | 20 | 8 | 1 | ∞ |

Kemudian lakukan reduksi kolom :

| | | | | | |
|---|---|---|---|---|---|
| ∞ | 18 | 10 | 0 | 11 | |
| 2 | ∞ | 15 | 0 | 15 | C2-16 |
| 3 | 24 | ∞ | 0 | 12 | C3-7 |
| 0 | 16 | 7 | ∞ | 12 | |
| 0 | 20 | 8 | 1 | ∞ | C5-11 |

Setelah reduksi kolom :

| | | | | |
|---|---|---|---|---|
| ∞ | 2 | 3 | 0 | 0 |
| 2 | ∞ | 8 | 0 | 4 |
| 3 | 8 | ∞ | 0 | 1 |
| 0 | 0 | 0 | ∞ | 1 |
| 0 | 4 | 1 | 1 | ∞ |

Total jumlah semua pengurang = (5+21+12+5+16) + (16+7+11) = 93. Nilai 93 ini adalah nilai batas untuk simpul akar,

ĉ (root) = 93

Sekarang kita akan melakukan reduksi dengan ketentuan
jika S bukan simpul daun, maka matriks berbobot reduksi untuk simpul S dapat dihitung sebagai berikut :

a. Ubah semua nilai pada baris *i* dan kolom *j* menjadi ∞. Ini untuk mencegah agar tidak ada lintasa yang keluar dari simpul *i* atau masuk pada simpul *j*.
b. Ubah A(j, 1) menjadi ∞. Ini untuk mencegah penggunaan sisi (j, 1).
c. Reduksi kembali semua baris dan kolom pada matris A kecuali untuk elemen ∞.

1. Simpul 2; Lintasan 1,2

| | | | | |
|---|---|---|---|---|
| ∞ | ∞ | ∞ | ∞ | ∞ |
| ∞ | ∞ | 8 | 0 | 4 |
| 3 | ∞ | ∞ | 0 | 1 |

| | | | | | |
|---|---|---|---|---|---|
| 0 | ∞ | 0 | ∞ | 1 | |
| 0 | ∞ | 1 | 1 | ∞ | C5-1 |

| | | | | |
|---|---|---|---|---|
| ∞ | ∞ | ∞ | ∞ | ∞ |
| 0 | ∞ | 8 | 0 | 3 |
| 3 | ∞ | ∞ | 0 | 0 |
| 0 | ∞ | 0 | ∞ | 0 |
| 0 | ∞ | 1 | 1 | ∞ |

Nilai batas untuk simpul 2 pada pohon ruang status :

ĉ (S) = ĉ (R) + A( i, j ) + r

ĉ (2) = ĉ (1) + A( 1, 2 ) + 1

= 93 + 2 + 1 = 96

2. Simpul 3; Lintasan 1, 3

| | | | | | |
|---|---|---|---|---|---|
| ∞ | ∞ | ∞ | ∞ | ∞ | |
| 2 | ∞ | ∞ | 0 | 4 | |
| ∞ | 8 | ∞ | 0 | 1 | |
| 0 | 0 | ∞ | ∞ | 1 | |
| 0 | 4 | ∞ | 1 | ∞ | C5-1 |

| | | | | |
|---|---|---|---|---|
| ∞ | ∞ | ∞ | ∞ | ∞ |
| 2 | ∞ | ∞ | 0 | 3 |
| 0 | 8 | ∞ | 0 | 0 |
| 0 | 0 | ∞ | ∞ | 0 |
| 0 | 4 | ∞ | 1 | ∞ |

Nilai batas untuk simpul 3 pada pohon ruang status :

ĉ (S) = ĉ (R) + A( i, j ) + r

ĉ (3) = ĉ (1) + A( 1, 3 ) + 1

= 93 + 3 + 1 = 97

3. Simpul 4; Lintasan 1, 4

| | | | | | |
|---|---|---|---|---|---|
| ∞ | ∞ | ∞ | ∞ | ∞ | |
| 2 | ∞ | 8 | ∞ | 4 | R2-2 |
| 3 | 8 | ∞ | ∞ | 1 | R3-1 |
| ∞ | 0 | 0 | ∞ | 1 | |
| 0 | 4 | 1 | ∞ | ∞ | |

| | | | | |
|---|---|---|---|---|
| ∞ | ∞ | ∞ | ∞ | ∞ |
| 0 | ∞ | 6 | ∞ | 2 |
| 2 | 7 | ∞ | ∞ | 0 |
| 0 | 0 | 0 | ∞ | 1 |
| 0 | 4 | 1 | ∞ | ∞ |

Nilai batas untuk simpul 4 pada pohon ruang status :

ĉ (S) = ĉ (R) + A( i, j ) + r

ĉ (4) = ĉ (1) + A( 1, 4 ) + 3

= 93 + 0 + 3 = 96

4. Simpul 5; Lintasan 1, 5

| | | | | | |
|---|---|---|---|---|---|
| ∞ | ∞ | ∞ | ∞ | ∞ | |
| 2 | ∞ | 8 | 0 | ∞ | |
| 3 | 8 | ∞ | 0 | ∞ | |
| 0 | 0 | 0 | ∞ | ∞ | |
| ∞ | 4 | 1 | 1 | ∞ | R5-1 |

| | | | | |
|---|---|---|---|---|
| ∞ | ∞ | ∞ | ∞ | ∞ |
| 2 | ∞ | 8 | 0 | ∞ |
| 3 | 8 | ∞ | 0 | ∞ |
| 0 | 0 | 0 | ∞ | ∞ |
| 0 | 3 | 0 | 0 | ∞ |

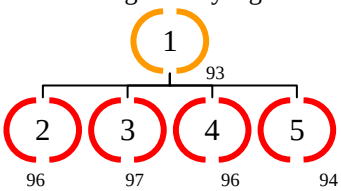
Nilai batas untuk simpul 5 pada pohon ruang status :

ĉ (S) = ĉ (R) + A( i, j ) + r

ĉ (5) = ĉ (1) + A( 1, 5 ) + 1

= 93 + 0 + 1 = 94

Pohon ruang status yang terbentuk sampai saat ini :




Nilai pada setiap simpul di dalam pohon ruang status di atas bermakna :

Jika perjalanan dimulai dari simpul 1, maka simpul berikutnya yang dikunjungi adalah :

- Simpul 2, dengan ongkos paling minimum 96
- Simpul 3, dengan ongkos paling minimum 97
- Simpul 4, dengan ongkos paling minimum 96
- Simpul 5, dengan ongkos paling minimum 94

Simpul yang hidup saat ini adalah 5, 2, 4, dan 3. Simpul 5 menjadi simpul hidup berikutnya karena memiliki nilai batas terkecil.

5. Simpul 6 ; Lintasan : 1, 5, 2

| | | | | |
|---|---|---|---|---|
| ∞ | ∞ | ∞ | ∞ | ∞ |
| ∞ | ∞ | 8 | 0 | ∞ |
| 3 | ∞ | ∞ | 0 | ∞ |
| 0 | ∞ | 0 | ∞ | ∞ |
| ∞ | ∞ | ∞ | ∞ | ∞ |

Nilai batas untuk simpul 6 pada pohon ruang status :

ĉ (S) = ĉ (R) + B( i, j ) + r

ĉ (6) = ĉ (5) + B( 5, 2 ) + 0

= 94 + 3 + 0 = 97

6. Simpul 7 ; Lintasan : 1, 5, 3

| | | | | |
|---|---|---|---|---|
| ∞ | ∞ | ∞ | ∞ | ∞ |
| 2 | ∞ | ∞ | 0 | ∞ |
| ∞ | 8 | ∞ | 0 | ∞ |
| 0 | 0 | ∞ | ∞ | ∞ |
| ∞ | ∞ | ∞ | ∞ | ∞ |

Nilai batas untuk simpul 7 pada pohon ruang status :

ĉ (S) = ĉ (R) + B( i, j ) + r

ĉ (7) = ĉ (5) + B( 5, 3 ) + 0

= 94 + 0 + 0 = 94

7. Simpul 8 ; Lintasan : 1, 5, 4

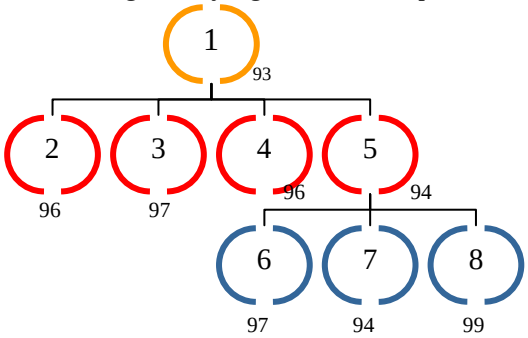
| | | | | | |
|---|---|---|---|---|---|
| ∞ | ∞ | ∞ | ∞ | ∞ | |
| 2 | ∞ | 8 | ∞ | ∞ | R2-2 |
| 3 | 8 | ∞ | ∞ | ∞ | R3-3 |
| ∞ | 0 | 0 | ∞ | ∞ | |
| ∞ | ∞ | ∞ | ∞ | ∞ | |

| | | | | |
|---|---|---|---|---|
| ∞ | ∞ | ∞ | ∞ | ∞ |
| 0 | ∞ | 6 | ∞ | ∞ |
| 0 | 5 | ∞ | ∞ | ∞ |
| ∞ | 0 | 0 | ∞ | ∞ |
| ∞ | ∞ | ∞ | ∞ | ∞ |

Nilai batas untuk simpul 8 pada pohon ruang status :

$\hat{c}(S) = \hat{c}(R) + B(i, j) + r$
$\hat{c}(8) = \hat{c}(5) + B(5, 4) + 5$
$= 94 + 0 + 5 = 99$

Pohon ruang status yang terbentuk sampai saat ini :




Nilai pada setiap simpul di dalam pohon ruang status di atas bermakna :
Jika perjalanan dimulai dari simpul 1, lalu simpul 5, simpul berikutnya yang dikunjungi adalah :
- Simpul 6, dengan ongkos paling minimum 97
- Simpul 7, dengan ongkos paling minimum 94
- Simpul 8, dengan ongkos paling minimum 99

Simpul yang hidup saat ini adalah 7, 2, 4, 3, 6, dan 8. Simpul 7 menjadi simpul hidup berikutnya karena memiliki nilai batas terkecil.

8. Simpul 9 ; Lintasan : 1, 5, 3, 2

$$\begin{bmatrix} \infty & \infty & \infty & \infty & \infty \\ \infty & \infty & \infty & 0 & \infty \\ \infty & \infty & \infty & \infty & \infty \\ 0 & \infty & \infty & \infty & \infty \\ \infty & \infty & \infty & \infty & \infty \end{bmatrix}$$

Nilai batas untuk simpul 9 pada pohon ruang status :
$\hat{c}(S) = \hat{c}(R) + C(i, j) + r$
$\hat{c}(9) = \hat{c}(7) + C(3, 2) + 0$
$= 94 + 8 + 0 = 102$

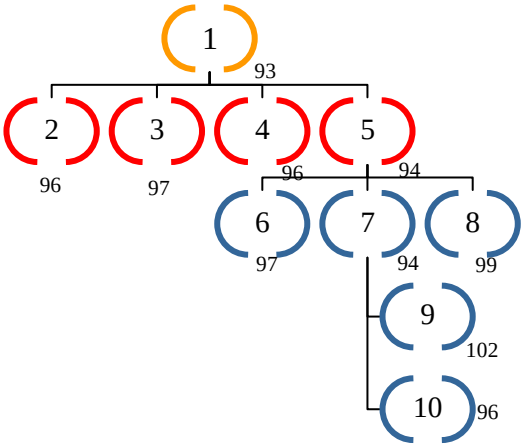
9. Simpul 10 ; Lintasan : 1, 5, 3, 4

$$\begin{bmatrix} \infty & \infty & \infty & \infty & \infty \\ 2 & \infty & \infty & \infty & \infty \\ \infty & \infty & \infty & \infty & \infty \\ \infty & 0 & \infty & \infty & \infty \\ \infty & \infty & \infty & \infty & \infty \end{bmatrix}$$ R2-2

$$\begin{bmatrix} \infty & \infty & \infty & \infty & \infty \\ 0 & \infty & \infty & \infty & \infty \\ \infty & \infty & \infty & \infty & \infty \\ \infty & 0 & \infty & \infty & \infty \\ \infty & \infty & \infty & \infty & \infty \end{bmatrix}$$

Nilai batas untuk simpul 10 pada pohon ruang status :
$\hat{c}(S) = \hat{c}(R) + C(i, j) + r$
$\hat{c}(10) = \hat{c}(7) + C(3, 4) + 2$
$= 94 + 0 + 2 = 96$

Pohon ruang status yang terbentuk sampai saat ini :




Nilai pada setiap simpul di dalam pohon ruang status di atas bermakna :
Jika perjalanan dimulai dari simpul 1, lalu simpul 5, kemudian simpul 7, simpul berikutnya yang dikunjungi adalah :
Simpul 9, dengan ongkos paling minimum 102
Simpul 10, dengan ongkos paling minimum 96

Simpul yang hidup saat ini adalah 2, 4, 10, 3, 6, 8, dan 9. Simpul 2 menjadi simpul hidup berikutnya karena memiliki nilai batas terkecil.

10. Simpul 11; Lintasan : 1, 2, 3

$$\begin{bmatrix} \infty & \infty & \infty & \infty & \infty \\ \infty & \infty & \infty & \infty & \infty \\ \infty & \infty & \infty & 0 & 0 \\ 0 & \infty & \infty & \infty & 0 \\ 0 & \infty & \infty & 1 & \infty \end{bmatrix}$$

Nilai batas untuk simpul 11 pada pohon ruang status :
$\hat{c}(S) = \hat{c}(R) + B(i, j) + r$
$\hat{c}(11) = \hat{c}(2) + B(2,3) + 0$
$= 96 + 8 + 0 = 104$

11. Simpul 12; Lintasan : 1, 2, 4

$$\begin{bmatrix} \infty & \infty & \infty & \infty & \infty \\ \infty & \infty & \infty & \infty & \infty \\ 3 & \infty & \infty & \infty & 0 \\ \infty & \infty & 0 & \infty & 0 \\ 0 & \infty & 1 & \infty & \infty \end{bmatrix}$$

Nilai batas untuk simpul 12 pada pohon ruang status :
$\hat{c}(S) = \hat{c}(R) + B(i, j) + r$
$\hat{c}(12) = \hat{c}(2) + B(2,4) + 0$
$= 96 + 0 + 0 = 96$

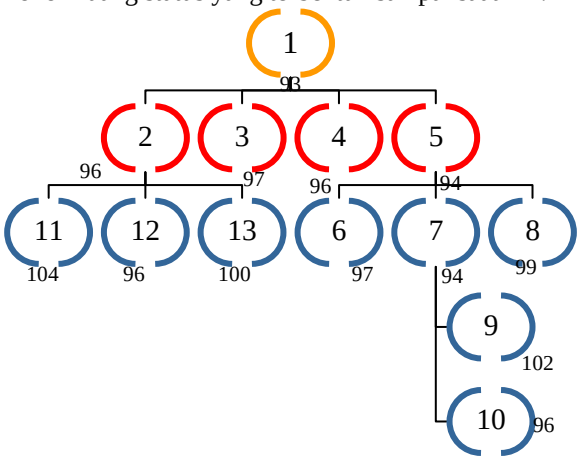
12. Simpul 13; Lintasan : 1, 2, 5

$$\begin{bmatrix} \infty & \infty & \infty & \infty & \infty \\ \infty & \infty & \infty & \infty & \infty \\ 3 & \infty & \infty & 0 & \infty \\ 0 & \infty & 0 & \infty & \infty \\ \infty & \infty & 1 & 1 & \infty \end{bmatrix}$$ R5-1

| | | | | |
|---|---|---|---|---|
| ∞ | ∞ | ∞ | ∞ | ∞ |
| ∞ | ∞ | ∞ | ∞ | ∞ |
| 3 | ∞ | ∞ | 0 | ∞ |
| 0 | ∞ | 0 | ∞ | ∞ |
| ∞ | ∞ | 0 | 0 | ∞ |

Nilai batas untuk simpul 13 pada pohon ruang status :
ĉ (S) = ĉ (R) + B ( i, j ) + r
ĉ (13) = ĉ (2) + B ( 2,5 ) + 1
= 96 + 3 + 1 = 100

Pohon ruang status yang terbentuk sampai saat ini :




Nilai pada setiap simpul di dalam pohon ruang status di atas bermakna :
Jika perjalanan dimulai dari simpul 1, lalu simpul 2, simpul berikutnya yang dikunjungi adalah :

- Simpul 11, dengan ongkos paling minimum 104
- Simpul 12, dengan ongkos paling minimum 96
- Simpul 13, dengan ongkos paling minimum 100

Simpul yang hidup saat ini adalah 4, 10, 12, 3, 6, 8, 13, 9, dan 11. Simpul 4 menjadi simpul hidup berikutnya karena memiliki nilai batas terkecil.

13. Simpul 14; Lintasan 1, 4, 2

| | | | | | |
|---|---|---|---|---|---|
| ∞ | ∞ | ∞ | ∞ | ∞ | |
| ∞ | ∞ | 6 | ∞ | 2 | R2-2 |
| 2 | ∞ | ∞ | ∞ | 0 | |
| ∞ | ∞ | ∞ | ∞ | ∞ | |
| 0 | ∞ | 1 | ∞ | ∞ | |

| | | | | |
|---|---|---|---|---|
| ∞ | ∞ | ∞ | ∞ | ∞ |
| ∞ | ∞ | 4 | ∞ | 0 |
| 2 | ∞ | ∞ | ∞ | 0 |
| ∞ | ∞ | ∞ | ∞ | ∞ |
| 0 | ∞ | 1 | ∞ | ∞ |

Nilai batas untuk simpul 14 pada pohon ruang status :
ĉ (S) = ĉ (R) + B( i, j ) + r
ĉ (14) = ĉ (4) + B( 4, 2 ) + 2
= 96 + 0 + 2 = 98

14. Simpul 15; Lintasan 1, 4, 3

| | | | | | |
|---|---|---|---|---|---|
| ∞ | ∞ | ∞ | ∞ | ∞ | |
| 0 | ∞ | ∞ | ∞ | 2 | C2-4 |
| ∞ | 7 | ∞ | ∞ | 0 | |
| ∞ | ∞ | ∞ | ∞ | ∞ | |
| 0 | 4 | ∞ | ∞ | ∞ | |

| | | | | |
|---|---|---|---|---|
| ∞ | ∞ | ∞ | ∞ | ∞ |
| 0 | ∞ | ∞ | ∞ | 2 |
| ∞ | 3 | ∞ | ∞ | 0 |
| ∞ | ∞ | ∞ | ∞ | ∞ |
| 0 | 0 | ∞ | ∞ | ∞ |

Nilai batas untuk simpul 4 pada pohon ruang status :
ĉ (S) = ĉ (R) + B( i, j ) + r
ĉ (15) = ĉ (4) + B( 4, 3) + 4
= 96 + 0 + 4 = 100

15. Simpul 16; Lintasan 1, 4, 5

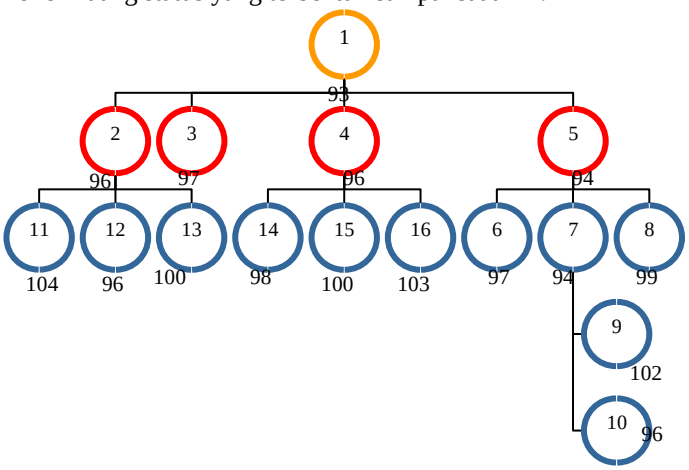
| | | | | | |
|---|---|---|---|---|---|
| ∞ | ∞ | ∞ | ∞ | ∞ | |
| 0 | ∞ | 6 | ∞ | ∞ | |
| 2 | 7 | ∞ | ∞ | ∞ | R3-2 |
| ∞ | ∞ | ∞ | ∞ | ∞ | |
| ∞ | 4 | 1 | ∞ | ∞ | R5-1 |

| | | | | | |
|---|---|---|---|---|---|
| ∞ | ∞ | ∞ | ∞ | ∞ | |
| 0 | ∞ | 6 | ∞ | ∞ | |
| 0 | 5 | ∞ | ∞ | ∞ | C2-3 |
| ∞ | ∞ | ∞ | ∞ | ∞ | |
| ∞ | 3 | 0 | ∞ | ∞ | |

| | | | | |
|---|---|---|---|---|
| ∞ | ∞ | ∞ | ∞ | ∞ |
| 0 | ∞ | 6 | ∞ | ∞ |
| 0 | 2 | ∞ | ∞ | ∞ |
| ∞ | ∞ | ∞ | ∞ | ∞ |
| ∞ | 0 | 0 | ∞ | ∞ |

Nilai batas untuk simpul 4 pada pohon ruang status :
ĉ (S) = ĉ (R) + B( i, j ) + r
ĉ (16) = ĉ (4) + B( 4, 5) + 6
= 96 + 1 + 6 = 103

Pohon ruang status yang terbentuk sampai saat ini :




Nilai pada setiap simpul di dalam pohon ruang status di atas bermakna :
Jika perjalanan dimulai dari simpul 1, lalu simpul 4, simpul berikutnya yang dikunjungi adalah :

- Simpul 14, dengan ongkos paling minimum 98
- Simpul 15, dengan ongkos paling minimum 100
- Simpul 16, dengan ongkos paling minimum 103

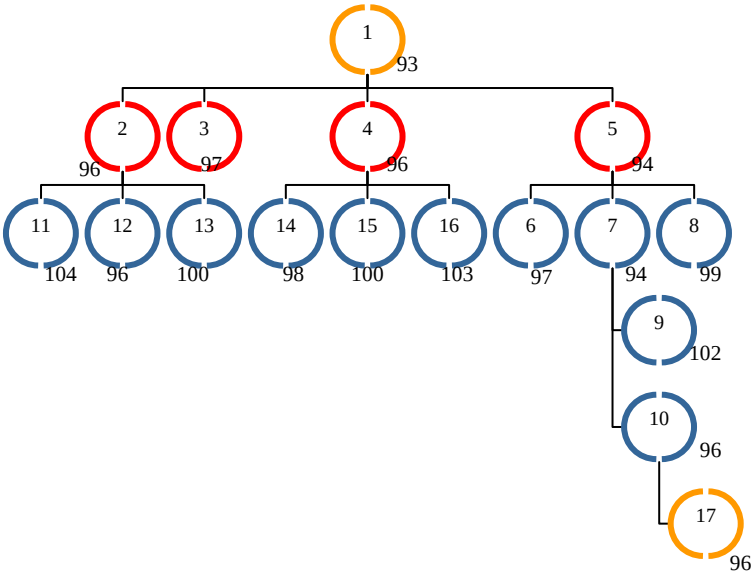
Simpul yang hidup saat ini adalah 10, 12, 3, 6, 14, 8, 13, 15, 9, 16, dan 11. Simpul 10 menjadi simpul hidup berikutnya karena memiliki nilai batas terkecil.

16. Simpul 17 ; Lintasan : 1, 5, 3, 4, 2

| | | | | |
|---|---|---|---|---|
| ∞ | ∞ | ∞ | ∞ | ∞ |
| ∞ | ∞ | ∞ | ∞ | ∞ |
| ∞ | ∞ | ∞ | ∞ | ∞ |
| ∞ | ∞ | ∞ | ∞ | ∞ |
| ∞ | ∞ | ∞ | ∞ | ∞ |

Nilai batas untuk simpul 17 pada pohon ruang status :
ĉ (S) = ĉ (R) + C( i, j ) + r
ĉ (17) = ĉ (10) + C( 4, 2 ) + 0
= 96 + 0 + 0 = 96

Pohon ruang status yang terbentuk sampai saat ini :




Karena simpul 17 adalah simpul daun, maka solusi pertama ditemukan, yaitu perjalanan dengan lintasan 1, 5, 3, 4, 2, 1 dengan panjang 96. Solusi ini merupakan solusi terbaik sejauh ini. Simpul yang hidup saat ini adalah 12, 3, 6, 14, 8, 13, 15, 9, 16, dan 11. Simpul 12 akan menjadi simpul hidup berikutnya karena memiliki nilai batas terkecil.

17. Simpul 18 ; Lintasan : 1, 2, 4, 3

| | | | | |
|---|---|---|---|---|
| ∞ | ∞ | ∞ | ∞ | ∞ |
| ∞ | ∞ | ∞ | ∞ | ∞ |
| ∞ | ∞ | ∞ | ∞ | 0 |
| ∞ | ∞ | ∞ | ∞ | ∞ |
| 0 | ∞ | ∞ | ∞ | ∞ |

Nilai batas untuk simpul 18 pada pohon ruang status :
ĉ (S) = ĉ (R) + C( i, j ) + r
ĉ (18) = ĉ (12) + C( 4, 3 ) + 0
= 96 + 0 + 0 = 96

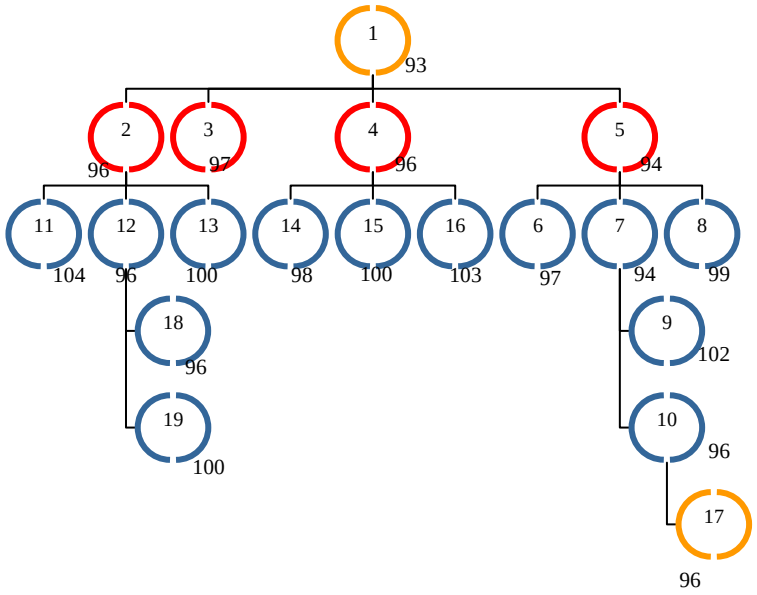
18. Simpul 19 ; Lintasan : 1, 2, 4, 5

| | | | | | |
|---|---|---|---|---|---|
| ∞ | ∞ | ∞ | ∞ | ∞ | |
| ∞ | ∞ | ∞ | ∞ | ∞ | |
| 3 | ∞ | ∞ | ∞ | ∞ | R3-3 |
| ∞ | ∞ | ∞ | ∞ | ∞ | |
| ∞ | ∞ | 1 | ∞ | ∞ | R5-1 |

| | | | | |
|---|---|---|---|---|
| ∞ | ∞ | ∞ | ∞ | ∞ |
| ∞ | ∞ | ∞ | ∞ | ∞ |
| 0 | ∞ | ∞ | ∞ | ∞ |
| ∞ | ∞ | ∞ | ∞ | ∞ |
| ∞ | ∞ | 0 | ∞ | ∞ |

Nilai batas untuk simpul 19 pada pohon ruang status :
ĉ (S) = ĉ (R) + C( i, j ) + r
ĉ (19) = ĉ (12) + C( 4, 5) + 4
= 96 + 0 + 4 = 100

Pohon ruang status yang terbentuk sampai saat ini :




Nilai pada setiap simpul di dalam pohon ruang status di atas bermakna :
Jika perjalanan dimulai dari simpul 1, lalu simpul 2, kemudian simpul 12, simpul berikutnya yang dikunjungi adalah :
- Simpul 18, dengan ongkos paling minimum 96
- Simpul 19, dengan ongkos paling minimum 100

Simpul yang hidup saat ini adalah 18, 3, 6, 14, 8, 13, 15, 19, 9, 16, dan 11. Simpul 18 menjadi simpul hidup berikutnya karena memiliki nilai batas terkecil.

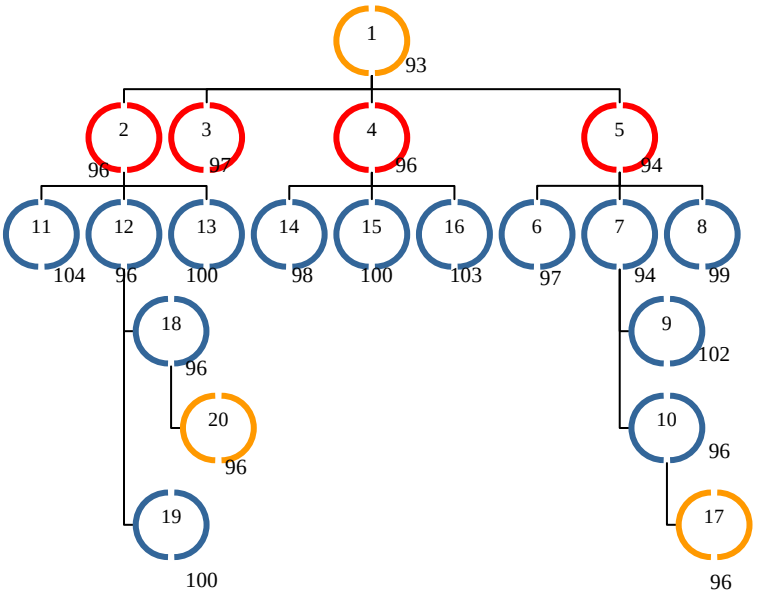
19. Simpul 20 ; Lintasan : 1, 2, 4, 3, 5

| | | | | |
|---|---|---|---|---|
| ∞ | ∞ | ∞ | ∞ | ∞ |
| ∞ | ∞ | ∞ | ∞ | ∞ |
| ∞ | ∞ | ∞ | ∞ | ∞ |
| ∞ | ∞ | ∞ | ∞ | ∞ |
| 0 | ∞ | ∞ | ∞ | ∞ |

Nilai batas untuk simpul 20 pada pohon ruang status :
ĉ (S) = ĉ (R) + C( i, j ) + r

$$\hat{c}(20) = \hat{c}(18) + C(3, 5) + 0$$
$$= 96 + 0 + 0 = 96$$

Pohon ruang status yang terbentuk sampai saat ini :




Karena simpul 20 adalah simpul daun, maka solusi kedua ditemukan, yaitu perjalanan dengan lintasan 1, 2, 4, 3, 5 dengan panjang 96. Solusi ini merupakan solusi terbaik juga. Dari penentuan jalur wisata menggunakan algoritma *Branch and Bound* telah ditemukan dua solusi dengan jarak yang optimum yaitu perjalanan dengan lintasan 1, 5, 3, 4, 2, 1 dengan panjang 96 dan perjalanan dengan lintasan 1, 2, 4, 3, 5 dengan panjang 96.

## IV. KESIMPULAN

Algoritma *Branch and Bound* ternyata dapat diterapkan dalam banyak aplikasi, bahkan aplikasi yang sangat dekat dengan kehidupan sehari-hari kita yaitu dalam menentukan jalur wisata. Penentuan urutan tempat yang dikunjungi pada saat berwisata penting untuk memaksimalkan waktu berwisata yang terbatas dengan jumlah tempat yang ingin dikunjungi. Sering terjadi pada saat berwisata, pengunjung ingin mengunjungi banyak tempat akan tetapi waktu yang disediakan terbatas. Dengan menggunakan algoritma *Branch and Bound* ini akan dicari urutan tempat yang dikunjungi sehingga total jarak yang ditempuh untuk mengunjungi tempat-tempat tersebut minimum dan waktu yang dipergunakan untuk mengunjungi tempat-tempat tersebut pun minimum.

Penggunaan *Branch and Bound* pada aplikasi ini diawali dengan menentukan kota/negara wisata yang ingin dikunjungi. Daftarkan tempat-tempat wisata yang ingin dikunjungi beserta jaraknya masing-masing. Setelah itu buatlah graf yang merepresentasikan tempat-tempat wisata yang ingin dikunjungi beserta jaraknya masing-masing. Kemudian dibuat sebuah matriks berbobot untuk graf tersebut. Lalu lakukan proses mereduksi matriks bobot hingga diperoleh urutan tempat-tempat wisata dengan jarak yang minimum. Dengan menggunakan metode *Branch and Bound*, terdapat dua jalur wisata mengunjungi Negara Hongkong yang memiliki jarak yang optimum yaitu :

1. Kowloon Walled City Park – Clear Water Bay – Ocean Park – Jade Market – Disneyland Park - Kowloon Walled City Park.
2. Kowloon Walled City Park – Disneyland Park – Jade Market – Ocean Park – Clear Water Bay - Kowloon Walled City Park.

## V. DAFTAR REFERENSI


[1] Ir. Munir, Rinaldi, N.P. , "Diktat Kuliah IF3051 Strategi Algoritma," Bandung: ITB, 2007, bab 5 dan bab 8.

[2] http://www.chinatourguide.com/ . Diakses pada tanggal 15 Desember 2012 pukul 22.00.

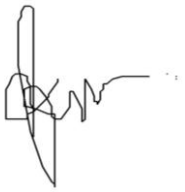
[3] http://courses.engr.illinois.edu/cs473/sp2011/Lectures/03_class.pdf . Diakses pada tanggal 15 Desember 2012 pukul 22.10.

[4] http://ww3.algorithmdesign.net/handouts/BFS.pdf. Diakses pada tanggal 15 Desember 2012 pukul 22.15 .


## 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, 20 Desember 2012



Janice Laksana
13510035