

2016

Finding Optimal Route dengan A*

Laporan Tugas Program Kecerdasan Artifisial

Dosen Pengampu: Mahmud Dwi Sulistiyo, ST., MT.

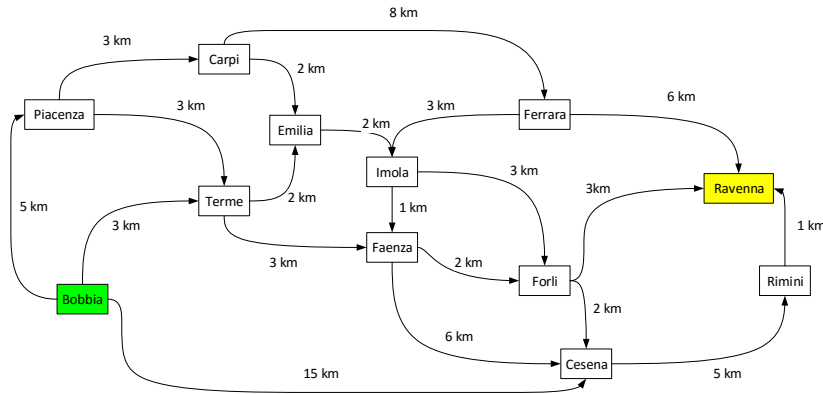
Nama: Putu Eka Budi Pradnyana

NIM: 1301144350

25 Februari 2016

1. Deskripsi Kasus

Kasus pada tugas program 1 ini merupakan *shortest path finding* atau menemukan jalur terpendek dari tiap jalur yang ada yang menghubungkan antara suatu tempat/kota asal hingga mencapai tempat/kota yang dituju.



Gambar diatas merupakan peta jalur-jalur yang menghubungkan tiap kota. Pada kasus yang disediakan ini akan dicari jalur terpendek dari Bobbia menuju Ravenna dimana tiap kota yang terhubung sudah disediakan jaraknya. Selain itu disediakan pula perkiraan jarak tiap-tiap kota menuju kota Ravenna atau jarak heuristic seperti gambar dibawah ini.

Kota	Ravenna	Rimini	Ferrara	Forlì	Cesena	Faenza	Imola	Emilia	Terme	Carpi	Piacenza	Bobbia
$h(n)$	0	0.5	5	2	4.5	4	5	6	7	8	10	10.5

Informasi berupa jarak tiap kota yang terhubung dan jarak heuristic tiap kota ke kota tujuan sangat diperlukan untuk menunjang pencarian rute terpendek dengan menggunakan algoritma A*.

2. Data yang Digunakan

Data yang digunakan pada tugas program 1 ini berupa data jarak antara satu kota dengan kota lain serta jarak heuristic dari tiap kota menuju kota tujuan yaitu Ravenna. Data tersebut dimasukan ke program secara hardcoded.

Rancangan data input dan outputnya adalah sebagai berikut :

Input	Jarak antar kota yang terhubung, jarak heuristic ke kota tujuan
Output	<p>Output pada tiap iterasi yaitu :</p> <ol style="list-style-type: none"> OPEN dan CLOSED ketika bestnode akan dievaluasi NILAI $f(n)$ untuk setiap node yang berada di OPEN dan CLOSED OPEN dan CLOSED ketika bestnode telah selesai dievaluasi BESTNODE baru yang akan dievaluasi di langkah selanjutnya <p>Output saat iterasi berhenti yaitu :</p> <ol style="list-style-type: none"> Solusi (rute) yang ditemukan Cost total

3. Desain Metode dan Implementasi Algoritma

Dalam mengerjakan desain metode dan implementasi algoritma, ada beberapa hal yang perlu diperhatikan diantaranya adalah :

a. Struktur data

Dalam pembuatan program ini digunakan struktur data berupa Array List yang librarynya disediakan Java melalui Netbeans. Array List digunakan untuk menampung node-node yang sudah dievaluasi (closed) dan node-node yang akan dievaluasi (open).

b. Algoritma A*

Adapun algoritma A* yang digunakan sebagai dasar/acuan pembuatan code program ini adalah sebagai berikut :

```

OPEN    // the set of nodes to be evaluated
CLOSED  // the set of nodes already evaluated
Add the start node to OPEN
While OPEN is not empty
    Current = node in OPEN with the lowest f_cost
    Remove current from OPEN
    Add current to CLOSED
    If current is the target node
        Return
    For each neighbour of the current node
        If neighbour is in CLOSED
            Skip to the next
        If neighbor is not in OPEN
            Set f_cost of neighbor
            Set parent of neighbour to current
            If neighbour is in OPEN
                Remove neighbour
            Add neighbour to OPEN

```

c. Penentuan Best Node


Penentuan best node pada program ini dilakukan dengan memperhatikan 2 kondisi yaitu :

1. Jika tidak terdapat node goal/Ravenna di arraylist open, maka akan dilakukan sortng array list open secara ascending berdasarkan value dari F_n tiap node yang ada di array list open menggunakan bantuan metode selection sort. Sehingga node yang memiliki value F_n paling kecil akan berada di array ke 0. Lalu best node bisa ditentukan dengan mengambil node yang berada di array list ke 0.
2. Jika terdapat node goal/Ravenna di arraylist open, maka akan dicari pada index ke berapa node goal itu berada dengan menggunakan fungsi `indexOf()` lalu ditampung disebuah variable position yang kemudian digunakan untuk mengambil node goal yang berada pada index array list sesuai dengan value variable position.

d. Penghentian Proses Iterasi

Pada program yang dibuat, loop berjalan selama array list open tidak kosong **dan** atribut isFound masih memiliki nilai false. Didalam loop ini akan dilakukan pengecekan terhadap nama dari best node dengan kondisi jika nama dari best node sama dengan nama dari goal node maka isFound akan diset ke true sehingga looping berhenti karena sudah mencapai goal node.

4. Petunjuk Penggunaan Program

Untuk penggunaan program yang dibuat, pengguna hanya perlu membuka project menggunakan netbeans/IDE java lainnya. Kemudian pilih/klik class Main.java lalu run dengan menekan F6 (pada netbeans) atau klik  ikon yang ada pada toolbar.

5. Luaran Program

Berdasarkan algoritma yang digunakan yaitu A* maka output yang ditampilkan pada program dibagi menjadi 2 yaitu :

1. Output pada tiap iterasi yaitu :
 - OPEN dan CLOSED ketika bestnode akan dievaluasi
 - Nilai Fn untuk setiap node yang berada di OPEN dan CLOSED
 - OPEN dan CLOSED ketika bestnode telah selesai dievaluasi
 - BESTNODE baru yang akan dievaluasi di langkah selanjutnya
2. Output saat iterasi berhenti yaitu :
 - Solusi (rute) yang ditemukan
 - Cost total

Contoh solusi yang dihasilkan dari program yang dibuat adalah Bobbia, Terme, Faenza, Forli, Ravenna dan program juga akan menampilkan cost dari solusi yaitu 11.

6. Screenshot Program

Berikut ditampilkan beberapa screenshot program yang mencakup input dan output program.

```
Node node1 = new Node("Ravenna",0);
Node node2 = new Node("Rimini",0.5);
Node node3 = new Node("Ferrara",5);
Node node4 = new Node("Forli",2);
Node node5 = new Node("Cesena",4.5);
Node node6 = new Node("Faenza",4);
Node node7 = new Node("Imola",5);
Node node8 = new Node("Emilia",6);
Node node9 = new Node("Terme",7);
Node node10 = new Node("Carpi",8);
Node node11 = new Node("Piacenza",10);
Node node12 = new Node("Bobbia",10.5);
```

Gambar 1 Input kota dilakukan secara hardcoded pada Main.java

```

node12.connect(new Edge(node11,5));
node12.connect(new Edge(node9,3));
node12.connect(new Edge(node5,15));

node11.connect(new Edge(node10,3));
node11.connect(new Edge(node9,3));

node9.connect(new Edge(node8,2));
node9.connect(new Edge(node6,3));

node6.connect(new Edge(node4,2));
node6.connect(new Edge(node5,6));

node5.connect(new Edge(node2,5));
node4.connect(new Edge(node5,2));
node4.connect(new Edge(node1,3));

node7.connect(new Edge(node4,3));
node7.connect(new Edge(node6,1));

node8.connect(new Edge(node7,2));

node10.connect(new Edge(node8,2));
node10.connect(new Edge(node3,8));

node3.connect(new Edge(node7,3));
node3.connect(new Edge(node1,6));

node2.connect(new Edge(node1,1));

```

Gambar 2 Input keterhubungan antar kota

```

OPEN : [Bobbia(10.5)]
CLOSED : []
***BEST NODE : Bobbia***
OPEN : []
CLOSED : [Bobbia(10.5)]

OPEN : [Piacenza(15.0), Terme(10.0), Cesena(19.5)]
CLOSED : [Bobbia(10.5)]
***BEST NODE : Terme***
OPEN : [Piacenza(15.0), Cesena(19.5)]
CLOSED : [Bobbia(10.5), Terme(10.0)]

OPEN : [Piacenza(15.0), Cesena(19.5), Emilia(11.0), Faenza(10.0)]
CLOSED : [Bobbia(10.5), Terme(10.0)]
***BEST NODE : Faenza***
OPEN : [Emilia(11.0), Piacenza(15.0), Cesena(19.5)]
CLOSED : [Bobbia(10.5), Terme(10.0), Faenza(10.0)]

OPEN : [Emilia(11.0), Piacenza(15.0), Cesena(19.5), Forli(10.0)]
CLOSED : [Bobbia(10.5), Terme(10.0), Faenza(10.0)]
***BEST NODE : Forli***
OPEN : [Emilia(11.0), Piacenza(15.0), Cesena(19.5)]
CLOSED : [Bobbia(10.5), Terme(10.0), Faenza(10.0), Forli(10.0)]

OPEN : [Emilia(11.0), Piacenza(15.0), Cesena(19.5), Ravenna(11.0)]
CLOSED : [Bobbia(10.5), Terme(10.0), Faenza(10.0), Forli(10.0)]
***BEST NODE : Ravenna***
OPEN : [Emilia(11.0), Piacenza(15.0), Cesena(19.5)]
CLOSED : [Bobbia(10.5), Terme(10.0), Faenza(10.0), Forli(10.0), Ravenna(11.0)]

Rutenya : Bobbia Terme Faenza Forli Ravenna
Costnya : 11.0

```

Gambar 3 Output yang dihasilkan