Implementation of GIS Tangle1.0 by JAVA

RUI ZHU

Geodesy and Geoinformatics

Royal Instiute of Technology

Contents

1. Proposal for *GIS Tangle 1.0*

2. Strategy

3. Results

4. Analysis and Discussion

Proposal for GIS Tangle

Create an interface to structure the software by SWT in Eclipse

Implement classes for different analysis models

1. Aspect Model
2. Slope Model
3. Shortest Path Model

Create input and output data file models for display

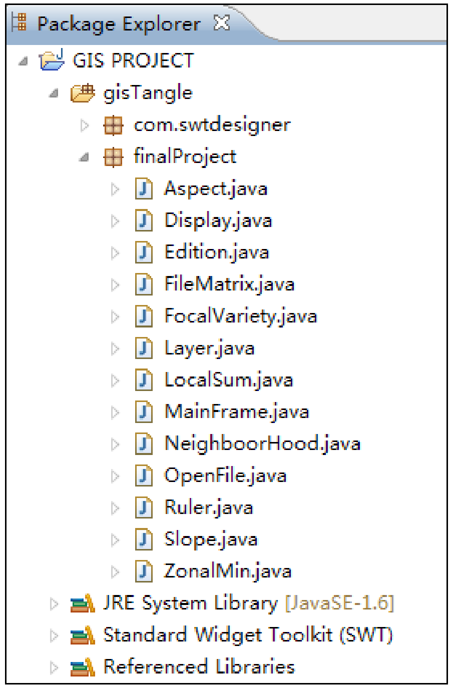
Create drawing model where pictures can be operated

1. Move
2. Renew
3. Distances Measurement

Strategy

Table 1 Structures

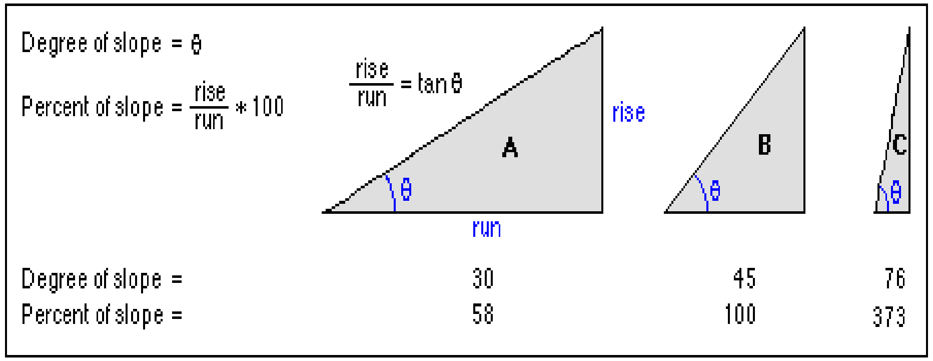
Layer 1	Layer 2	Layer 3
MainFrame Layer FileMatrix	OpenFile Model	OpenFile
	Raster Model	Slope Aspect Local Sum Focal Variety Zonal Minimum 3*3 Neighbor Hood
	Vector Model	Shortest Path
	Display Model	Display Ruler
	Edition Model	Edition

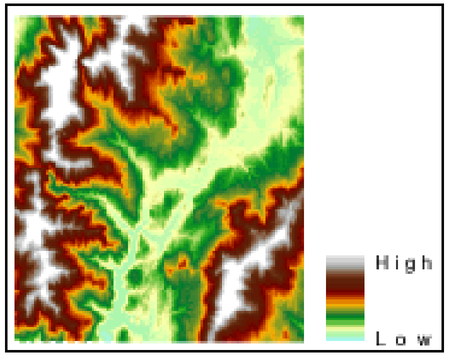


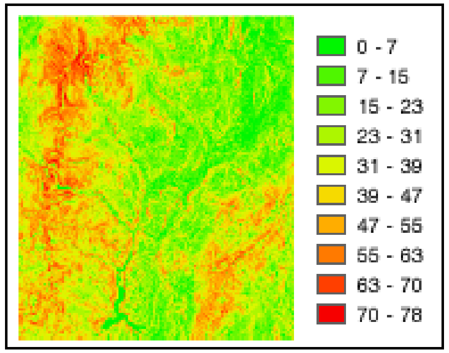
```
Package Explorer
GIS PROJECT
  gisTangle
    com.swtdesigner
    finalProject
      Aspect.java
      Display.java
      Edition.java
      FileMatrix.java
      FocalVariety.java
      Layer.java
      LocalSum.java
      MainFrame.java
      NeighboorHood.java
      OpenFile.java
      Ruler.java
      Slope.java
      ZonalMin.java
  JRE System Library [JavaSE-1.6]
  Standard Widget Toolkit (SWT)
  Referenced Libraries
```

Figure 1 classes in the project

Slope


Degree of slope = θ
Percent of slope = rise/run * 100
rise/run = tan θ
rise
run
A
B
C
θ
Degree of slope = 30 45 76
Percent of slope = 58 100 373





a	b	c
d	e	f
g	h	i

The basic algorithm used to calculate the slope is:

slope_radians = ATAN (sqrt((dz/dx)^2 + (dz/dy)^2))

Slope is commonly measured in degrees, which uses the algorithm:

slope_degrees = ATAN (sqrt((dz/dx)^2 + (dz/dy)^2)) * 57.29578

The slope algorithm can also be interpreted as:

slope_degrees = ATAN (rise_run) * 57.29578

where:

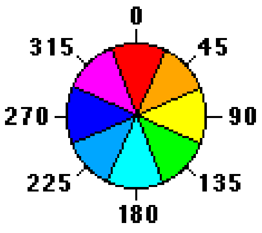
rise_run = sqrt((dz/dx)^2 + (dz/dy)^2)

The rate of change in the x direction for cell e is calculated with the following algorithm:

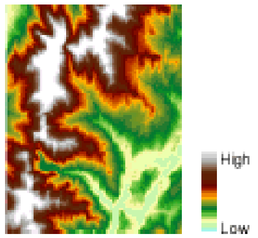
[dz/dx] = ((c + 2f + i) - (a + 2d + g)) / (8 * x_cellsize)

The rate of change in the y direction for cell e is calculated with the following algorithm:

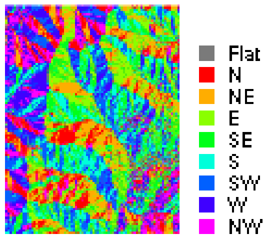
[dz/dy] = ((g + 2h + i) - (a + 2b + c)) / (8 * y_cellsize)



Aspect directions



Input elevation raster



Output aspect raster

a	b	c
d	e	f
g	h	i

Aspect

The rate of change in the x direction for cell e is calculated:

$$[dz/dx] = ((c + 2f + i) - (a + 2d + g)) / 8$$

The rate of change in the y direction for cell e is calculated:

$$[dz/dy] = ((g + 2h + i) - (a + 2b + c)) / 8$$

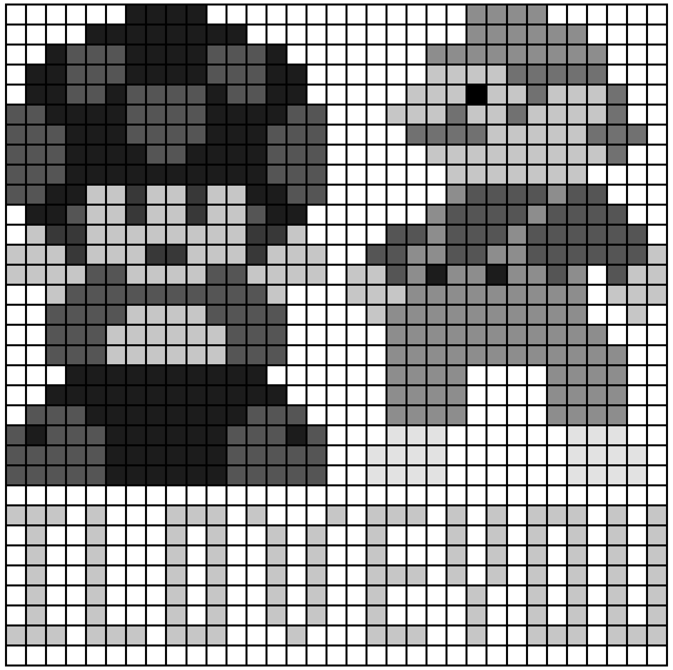
Taking the rate of change in both the x and y direction for cell e, aspect is calculated using:

$$aspect = 57.29578 * \operatorname{atan2}([dz/dy], -[dz/dx])$$

The aspect value is then converted to compass direction values (0-360 degrees), according to the following rule:

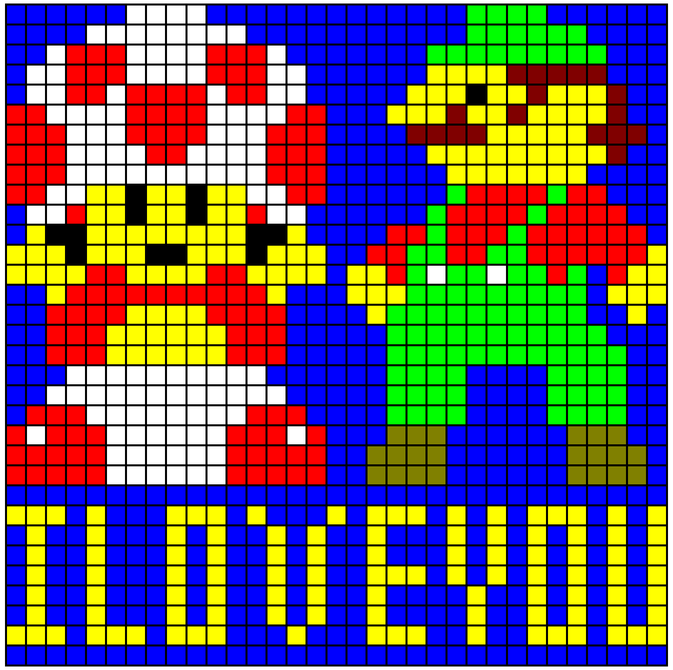
```
if aspect < 0
    cell = 90.0 - aspect
else if aspect > 90.0
    cell = 360.0 - aspect + 90.0
else
    cell = 90.0 - aspect
```

RGB Display



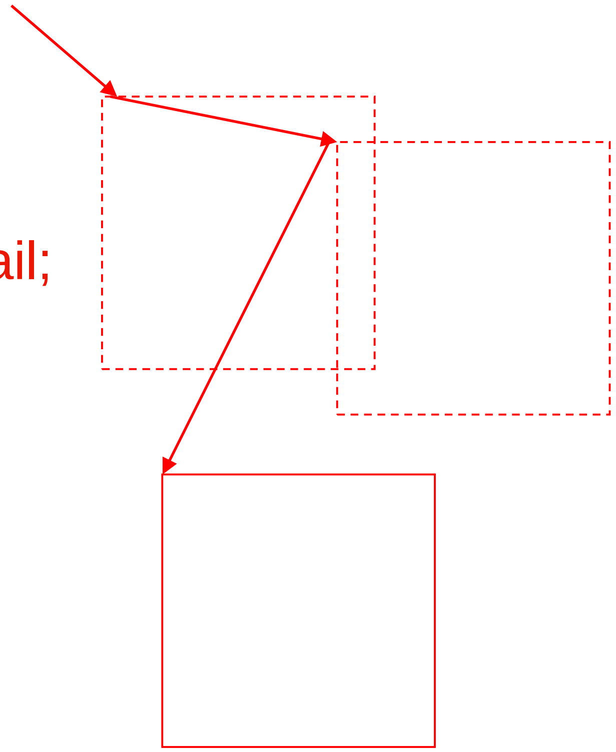
```
for(int i = 0; i<fm.getNcols(); i++)
        for(int j = 0; j<fm.getNrows() ;j++)
        {
                setpixel = (int)(fm.getMatrix()[i][j]);
                float min = fm.getMin();
                float max = fm.getMax();
                int grayValue = (int)(255 * (setpixel) / (max - min));
                RGB cyanRGB = new RGB(grayValue, grayValue, grayValue);
                Color color = new Color(display, cyanRGB);
                arg0.gc.setBackground(color);
                arg0.gc.fillRectangle(j*res, i*res, res, res);
                arg0.gc.drawRectangle(j*res, i*res, res, res);
        }
```

getSystemColor() Display



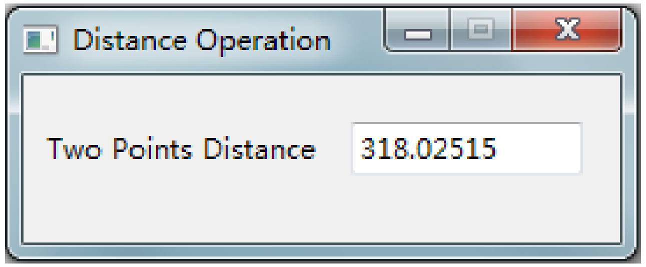
```
for(int i = 0; i<fm.getNrows(); i++)
    for(int j = 0; j<fm.getNcols(); j++)
    {
        setpixel = (int)(fm.getMatrix()[i][j]);
        arg0.gc.setBackground(display.getSystemColor(setpixel));
        arg0.gc.fillRectangle((int)(MainFrame.this.xLeftTop+j*res),
            (int)(MainFrame.this.yLeftTop+i*res), res, res);
        arg0.gc.drawRectangle((int)(MainFrame.this.xLeftTop+j*res),
            (int)(MainFrame.this.yLeftTop+i*res), res, res);
    }
```

Refresh the Picture



```
xChed = MainFrame.this.xMovePoint - MainFrame.this.xTail;
yChed = MainFrame.this.yMovePoint - MainFrame.this.yTail;

MainFrame.this.xLeftTop += xChed;
MainFrame.this.yLeftTop += yChed;

    FileMatrix fm = new FileMatrix(MainFrame.this.filePath);
    int res = fm.getCellsize();
    for(int i = 0; i<fm.getNrows(); i++)
        for(int j = 0; j<fm.getNcols(); j++)
        {
            setpixel = (int)(fm.getMatrix()[i][j]);
            arg0.gc.setBackground(display.getSystemColor(setpixel));
            arg0.gc.fillRectangle((int)(MainFrame.this.xLeftTop+j*res),
                (int)(MainFrame.this.yLeftTop+i*res), res, res);
            arg0.gc.drawRectangle((int)(MainFrame.this.xLeftTop+j*res),
            (int)(MainFrame.this.yLeftTop+i*res), res, res);
        }
MainFrame.this.xTail = MainFrame.this.xMovePoint;
MainFrame.this.yTail = MainFrame.this.yMovePoint;
```

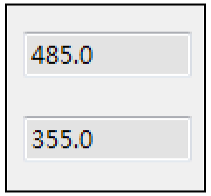
Distance Measurement



Two Points Distance 318.02515

```
public void showDis(float xs, float xe, float ys, float ye)
{
        text.setText( Float.toString( (float)
                Math.sqrt( (xs-xe)*(xs-xe)+(ys-ye)*(ys-ye) ) ) );
}
```

Coordinates Display


485.0

355.0

```
canvas.addMouseMoveListener(new MouseMoveListener()
{
    public void mouseMove(MouseEvent arg0)
    {
        if(MainFrame.this.filePath != null && !MainFrame.this.filePath.isEmpty() )
        {
            MainFrame.this.xMovePoint = arg0.x;
            MainFrame.this.yMovePoint = arg0.y;

            MainFrame.this.textX.setText( Float.toString(xMovePoint) );
            MainFrame.this.textY.setText( Float.toString(yMovePoint) );

            if( MainFrame.this.xStart >= 0 && MainFrame.this.xEnd < 0
                && MainFrame.this.moveEnable == true)
             canvas.redraw();
        }
    }
});
```

show-how

The rest time is for UNSAL…

Dijkstra Shortest Path Algorithm

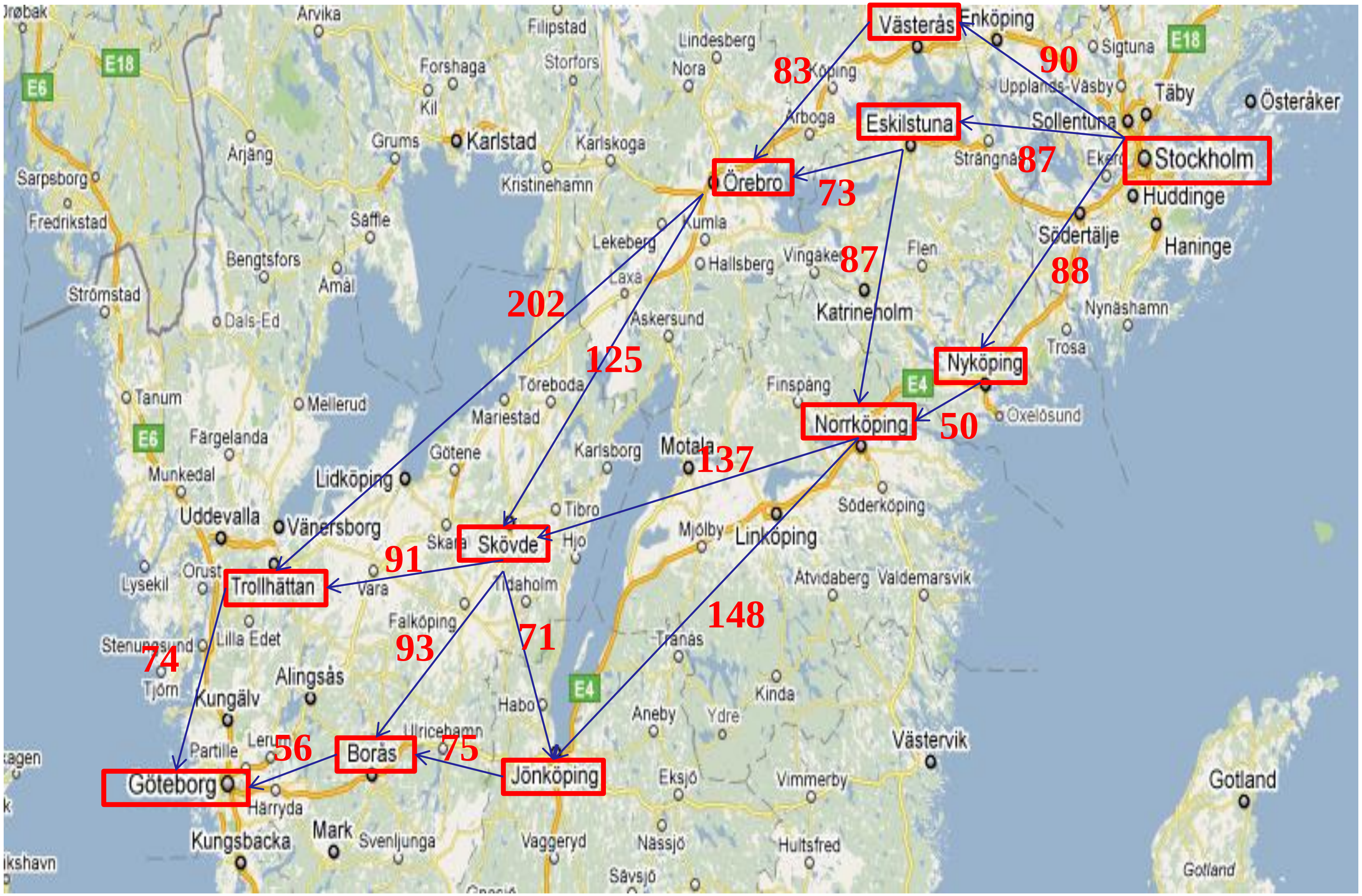
PROBLEM:

Finding the Shortest Highway Path and its Distance from Stockholm to Goteborg

CLASSES:

1. **Vertex Class**
2. **Distance Class**
3. **Graph Class**
4. **Path Class**

KTH
VETENSKAP
OCH KONST



The implementation of Dijkstra to my Path Java Class

1. Firstly I marked all vertexes as unvisited and initial vertex is current.
2. For current vertex with all its unvisited neighbors, we calculate their temporary distance
3. If these distance are less than the previously recorded distance overwrite the distance.
4. all vertexes have been visited, algorithms can be finished

All Possible Paths	km
1. Stockholm-Vasteras-Orebro-Trollhattan-Goteborg	449
2. Stockholm- Vasteras- Orebro-Skovde-Boras- Goteborg	447
3. Stockholm- Vasteras-Orebro-Skovde-Jonkoping-Boras- Goteborg	500
4. Stockholm- Eskillstuna-Orebro- Trollhattan-Goteborg	436
5. Stockholm- Eskillstuna -Orebro- Skovde-Boras- Goteborg	434
6. Stockholm- Eskillstuna -Orebro-Skovde-Jonkoping-Boras- Goteborg	487
7. Stockholm- Eskillstuna -Norrkoping-Skovde- Boras- Goteborg	460
8. Stockholm- Eskillstuna -Norrkoping-Skovde-Jonkoping-Boras- Goteborg	513
9. Stockholm- Eskillstuna - Norrkoping -Skovde- Trollhattan-Goteborg	476
10. Stockholm- Eskillstuna - Norrkoping - Jonkoping-Boras- Goteborg	453
11. Stockholm- Nykoping -Norrkoping-Skovde- Boras- Goteborg	424
12. Stockholm- Nykoping -Norrkoping-Skovde-Jonkoping-Boras- Goteborg	477
13. Stockholm- Nykoping - Norrkoping -Skovde- Trollhattan-Goteborg	440
14. Stockholm- Nykoping - Norrkoping - Jonkoping-Boras- Goteborg	417

OPTIONAL COMMAND

```
System.out.println(theGraph.GetVListelement(10));
    Distance node = theGraph.GetSpathelement(10);
    System.out.println(theGraph.GetVListelement(node.Vert));
    node = theGraph.GetSpathelement(node.Vert);
System.out.println();
```

```
City1         Distance    City2
Stockholm
Stockholm       90        Vasteras
Stockholm       87        Eskillstuna
Stockholm       88        Nykoping
Eskillstuna     160       Orebro
Nykoping        138       Norrkoping
Orebro          362       Trollhattan
Norrkoping      275       Skovde
Norrkoping      286       Jonkoping
Jonkoping       361       Boras
Boras           417       Goteborg
```

→ OUTPUT FILE-1

```
/*System.out.println(theGraph.GetVListelement(10));
    Distance node = theGraph.GetSpathelement(10);
    System.out.println(theGraph.GetVListelement(node.Vert));
    node = theGraph.GetSpathelement(node.Vert);
System.out.println();*/
```

```
Shortest Path
417

Stockholm

Nykoping

Norrkoping

Jonkoping

Boras

Goteborg
```

→ OUTPUT FILE-2

```
System.out.println(theGraph.GetVListelement(7));
    Distance node = theGraph.GetSpathelement(7);
```

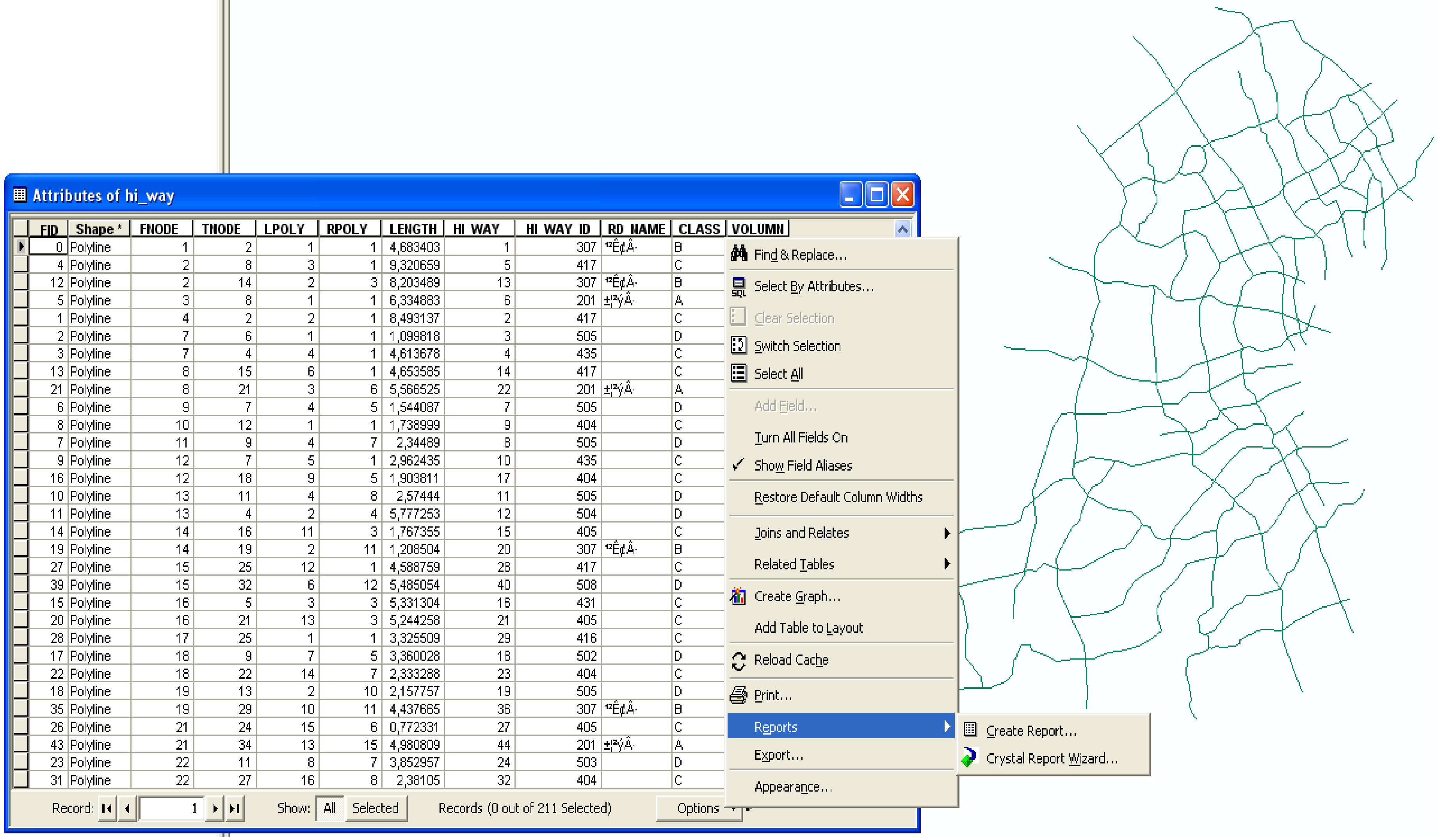
```
Shortest Path
275

Stockholm

Nykoping

Norrkoping

Skovde
```

OUTPUT FILE-3

CRITICS ABOUT SHORTEST PATH

- The most difficult part is the finding an input file for Dijkstra

CREATING AN ASCII INPUT FILE FROM ArcGIS

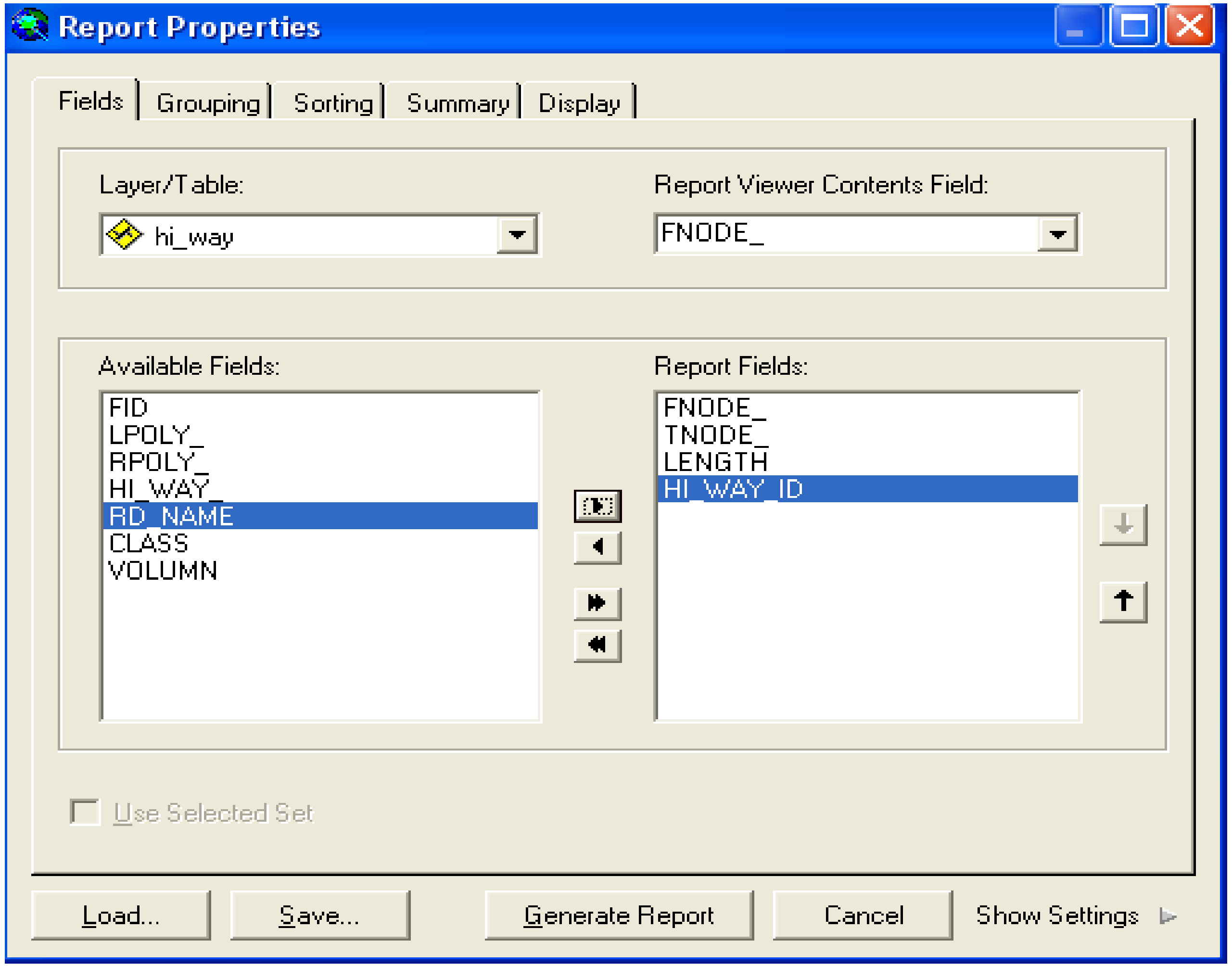


Attributes of hi_way

FID	Shape *	FNODE	TNODE	LPOLY	RPOLY	LENGTH	HI WAY	HI WAY ID	RD NAME	CLASS	VOLUMN
0	Polyline	1	2	1	1	4,683403	1	307	¹²Ê¢Â·	B	
4	Polyline	2	8	3	1	9,320659	5	417		C	
12	Polyline	2	14	2	3	8,203489	13	307	¹²Ê¢Â·	B	
5	Polyline	3	8	1	1	6,334883	6	201	±¦²ýÂ·	A	
1	Polyline	4	2	2	1	8,493137	2	417		C	
2	Polyline	7	6	1	1	1,099818	3	505		D	
3	Polyline	7	4	4	1	4,613678	4	435		C	
13	Polyline	8	15	6	1	4,653585	14	417		C	
21	Polyline	8	21	3	6	5,566525	22	201	±¦²ýÂ·	A	
6	Polyline	9	7	4	5	1,544087	7	505		D	
8	Polyline	10	12	1	1	1,738999	9	404		C	
7	Polyline	11	9	4	7	2,34489	8	505		D	
9	Polyline	12	7	5	1	2,962435	10	435		C	
16	Polyline	12	18	9	5	1,903811	17	404		C	
10	Polyline	13	11	4	8	2,57444	11	505		D	
11	Polyline	13	4	2	4	5,777253	12	504		D	
14	Polyline	14	16	11	3	1,767355	15	405		C	
19	Polyline	14	19	2	11	1,208504	20	307	¹²Ê¢Â·	B	
27	Polyline	15	25	12	1	4,588759	28	417		C	
39	Polyline	15	32	6	12	5,485054	40	508		D	
15	Polyline	16	5	3	3	5,331304	16	431		C	
20	Polyline	16	21	13	3	5,244258	21	405		C	
28	Polyline	17	25	1	1	3,325509	29	416		C	
17	Polyline	18	9	7	5	3,360028	18	502		D	
22	Polyline	18	22	14	7	2,333288	23	404		C	
18	Polyline	19	13	2	10	2,157757	19	505		D	
35	Polyline	19	29	10	11	4,437665	36	307	¹²Ê¢Â·	B	
26	Polyline	21	24	15	6	0,772331	27	405		C	
43	Polyline	21	34	13	15	4,980809	44	201	±¦²ýÂ·	A	
23	Polyline	22	11	8	7	3,852957	24	503		D	
31	Polyline	22	27	16	8	2,38105	32	404		C	

Record: 1 Show: All Selected Records (0 out of 211 Selected) Options

CREATING AN ASCII INPUT FILE FROM ArcGIS



Report Properties

Fields | Grouping | Sorting | Summary | Display

Layer/Table: hi_way

Report Viewer Contents Field: FNODE_

Available Fields:
FID
LPOLY_
RPOLY_
HI_WAY_
RD_NAME
CLASS
VOLUMN

Report Fields:
FNODE_
TNODE_
LENGTH
HI_WAY_ID

□ Use Selected Set

Analysis and Discussion

1 Datasets are used appropriate for the analysis.

2 Results are reliable when considering the datasets and methods used.

3 It would have been more appropriate if time is enough.

Thank you and any questions? ^-^