CV Project 5: Connected Component Java Student: Zeehan Rahman

Project Due Date: 10/17/2021

Algorithmic Steps:

1. Open the input file and all the three output files (RfPrettyPrintFile, labelFile, propertyfile) 2. Dynamically allocate all the fields (numrows, numcols, minval, maxval) by reading the values from the input file
3. Dynamically allocate the zeroframedarray with size [numrows+2] [numcols+2], in order to padd the borders. Then initialize all the elements of array with 0
4. Dynamically allocate the eqarray with size (numrows+numcols)/4, and initialize only the 0^{th} element with 0. (Other elements are initialized later)
5. Load the zeroframedarray from the input file by calling the function loadimage
6. Then ask the user from console if he/she would like to use 4 connectedness or 8 connectedness algorithm to cluster connected sets of object pixels
7. Depending upon the user's choice, run the pass 1 for 8 component algorithm or for the 4 connected component algorithm. The algorithms modifies the zeroframedarray 8. Then call the imagereformat function to print the modified zeroframedarray to the RFPrettyPrintFile
9. Then call the printeqarray function to print the eqarray to the RFPrettyPrintFile 10. Then the pass 2 for either the 8 connected component or the 4 connected component algorithm is called, depending upon the user's previous choice
11. Then execute step 8 and 9 once for the modified zeroframedarray resulting from the pass 2 algorithm
12. Call the function manageEqarray to return the actual number of connected components resulting from the pass 2.
13. Then call the printeqarray function to print the eqarray to the RFPrettyPrintFile 14. Call the function pass3 to update the zeroframedarray pixel values with the actual values fromt the equivalence table. Also, assign the correct attributes(numpixels, minrow,mincol,maxrow, maxcol and the label) of each connected components to the Property object
15. Repeat steps 8 and 9 once for the modified zeroframedarray resulting from the pass 3 algorithm
16. Call the function printimage that writes the header and the resulting image from the pass 3 to the labelfile
17. Call the function printccproperty to output the attributes of each labels (numpixels, minrow, mincol, maxrow, maxcol and the label) to the propertyfile
18. Call the function drawboxes that draws the bounding box for different pixels. The results are stored in zeroframedarray
19. Call the function imagereformat to output the zeroframedarray to the RFPrettyPrintFile 20. Output the actual number of connected component in the image in the RFPrettyPrintFile 21. Finally, close all files

Source Code

```
//Main
import java.io.*;
import java.util.Scanner;
public class Main {

	public static void main(String[] args) throws IOException
	{


		String inputfilename = args[0];
		FileReader inputImage = null;
		BufferedReader buffinImage = null;
		Scanner input = null;

		String outputname = args[1];
		FileWriter outputwriter = null;
		BufferedWriter output = null;

		String outputname2 = args[2];
		FileWriter outputwriter2 = null;
		BufferedWriter output2 = null;

		String outputname3 = args[3];
		FileWriter outputwriter3 = null;
		BufferedWriter output3 = null;

		try
		{
			inputImage = new FileReader(inputfilename);
			buffinImage = new BufferedReader(inputImage);
			input = new Scanner(buffinImage);

			Scanner in = new Scanner(System.in);
			System.out.print("Enter your choice: ");

			 String s = in.nextLine();
		System.out.println("You entered " + s);


			outputwriter = new FileWriter(outputname);
			output = new BufferedWriter(outputwriter);

			outputwriter2 = new FileWriter(outputname2);
			output2 = new BufferedWriter(outputwriter2);

			outputwriter3 = new FileWriter(outputname3);
			output3 = new BufferedWriter(outputwriter3);

			int numrows = 0, numcols = 0, minval = 0, maxval = 0;

			if(input.hasNextInt()) numrows = input.nextInt();
			else System.out.println("Invalid format of header");

			if(input.hasNextInt()) numcols = input.nextInt();
			else System.out.println("Invalid format of header");
			if(input.hasNextInt()) minval = input.nextInt();
```

```
else System.out.println("Invalid format of header");

if(input.hasNextInt()) maxval = input.nextInt();
else System.out.println("Invalid format of header");


boolean is4 = false;

if(s.contentEquals("4")) is4 = true;

CClabel CCobj = new CClabel(numrows, numcols, minval, maxval, is4); //array is dynamically
allocated in the constructor



CCobj.loadimage(input);

if(s.contentEquals("8"))
{
CCobj.Connect8Pass1();
 CCobj.ImageReformat(CCobj.zeroframedarray, output, "Using 8 Connected"
                          + " Component Algorithm\nPass 1");
CCobj.PrintEQArray(output, "Pass 1");

CCobj.Connect8Pass2();
 CCobj.ImageReformat(CCobj.zeroframedarray, output, "Pass 2");
CCobj.PrintEQArray(output, "Pass 2");
}

else if(s.contentEquals("4"))
{
        CCobj.Connect4Pass1();
         CCobj.ImageReformat(CCobj.zeroframedarray, output, "Using 4 Connected
Component "
                                    + "Algorithm\nPass 1");
        CCobj.PrintEQArray(output, "Pass 1");


        CCobj.Connect4Pass2();
         CCobj.ImageReformat(CCobj.zeroframedarray, output, "Pass 2");

        CCobj.PrintEQArray(output, "Pass 2");
}

else
        {
         System.out.println("Invalid entry"); System.exit(0);
         }

CCobj.truecc = CCobj.ManageEqArray();
 CCobj.PrintEQArray(output, "After Equivalence Table Management");

CCobj.Pass3();
 CCobj.ImageReformat(CCobj.zeroframedarray, output, "Pass 3");
CCobj.PrintEQArray(output, "Pass 3");

//CCobj.ImageReformat(CCobj.zeroframedarray, output2, "Final Result");
```

```
                    ///label 8 connectednesss or 4 connectedness
                    CCobj.PrintImage(output2);

                    CCobj.CCPropertyFile(output3);
                    CCobj.DrawBoxes();
                output.write("True number of Connected Components: "+CCobj.truecc+"\n");
                 CCobj.ImageReformat(CCobj.zeroframedarray,output, "Result of Draw Box");

                    System.out.println("Done, check the output files");

            }

            finally
            {
            if(input!=null) input.close();
            if(output!=null) output.close();
            if(output2!=null) output2.close();
            if(output3!=null) output3.close();
            }


    }

}
```

```
//CClabel
import java.io.BufferedWriter;
import java.io.IOException;
import java.util.LinkedList;
import java.util.Scanner;

public class CClabel
{
	public int numrows = 0, numcols = 0, minval = 0, maxval = 0;
	public int newLabel = 0;
	public int truecc = 0;
	public int minrow = 9999, mincol = 9999, maxcol = 0, maxrow = 0;
	public int newmin = 0 , newmax = 0;

	public boolean is4 = false;

	public int zeroframedarray[][];
	public int EQArray[];
	public Property PropertyFile[];

	public static LinkedList<Integer> link = new LinkedList<Integer>();

	public CClabel(int nr, int nc, int mv, int mxv, boolean isfour)
	{
		this.numrows = nr;
		this.numcols = nc;
		this.minval = mv;
		this.maxval = mxv;
		this.is4 = isfour;
		int eqsize = (numrows * numcols)/4;

		this.zeroframedarray = new int[numrows + 2] [numcols + 2];
		this.EQArray = new int[eqsize];


		for(int i = 0; i < this.numrows + 2; i++)
		{
			for(int j = 0; j < this.numcols + 2; j++)
			{
				this.zeroframedarray [i][j] = 0;

			}
		}

		this.EQArray[0] = 0;
	}

	public void loadimage(Scanner m)
	{
		for(int i = 1; i <= this.numrows; i++)
		{
			for(int j = 1; j <= this.numcols; j++)
			{
				if(m.hasNextInt())
					this.zeroframedarray[i][j] = m.nextInt();
```

```
                        }


                }
        }

        public void Connectedness()
        {
                int c = 1;
                if(c==4)
                {

                }

                else if(c==8)
                {

                }

                else System.out.println(c+" is a invalid choice");

        }

        public void Connect4Pass1()
        {


                for(int i = 1; i <= this.numrows; i++)
                {
                        for(int j = 1; j <= this.numcols; j++)
                        {
                                int pixel = this.zeroframedarray[i][j];

                                if(pixel>0)
                                {
                                        int a = this.zeroframedarray[i-1][j]; int b = this.zeroframedarray[i][j-1];// int
c = zeroarray[i-1][j]; int d = zeroarray[i-1][j];

                                        //Case 1
                                        if(a==0&&b==0)
                                        {
                                                newLabel++;
                                                this.zeroframedarray[i][j] = newLabel;
                                                this.EQArray[newLabel] = newLabel;
                                        }

                                        else if(a==b)
                                        {
                                                this.zeroframedarray[i][j] = a;

                                        }

                                        else if(a==0)
                                        {
                                                this.zeroframedarray[i][j] = b;
                                        }
```

```
                                        else if(b==0)
                                        {

                                                this.zeroframedarray[i][j] = a;
                                        }

                                        else if(a!=b)
                                        {
                                                int minlabel = Math.min(a, b);
                                                this.zeroframedarray[i][j] = minlabel;
                                                this.EQArray[Math.max(a, b)] = minlabel;
                                        }
                                }
                        }

                }

}

public void showArray()
{
        for(int i = 1; i <= this.numrows; i++)
        {
                for(int j = 1; j <= this.numcols; j++)
                {
                        System.out.print(this.zeroframedarray[i][j]+" ");

                }
                System.out.println();
        }
}

public void Connect4Pass2()
{
        for(int i = this.numrows; i >= 0; i--)
        {
                for(int j = this.numcols; j >= 0; j--)
                {
                        int pixel = this.zeroframedarray[i][j];
                        int minlabel =0;

                        if(pixel>0)
                        {
                                int a = this.zeroframedarray[i+1][j]; int b = this.zeroframedarray[i][j+1];

                                if(a==0&&b==0)
                                {

                                }

                                else if(Conn4Pass2Case2(pixel, a, b))
                                {

                                }

                                else if(a==0)
```

```
                                        {
                                                minlabel = Math.min(pixel, b);
                                                if(pixel>minlabel)
                                                {
                                                        this.zeroframedarray[i][j] = minlabel;
                                                        this.EQArray[pixel] = minlabel;
                                                }

                                        }

                                        else if(b==0)
                                        {
                                                minlabel = Math.min(pixel, a);
                                                if(pixel>minlabel)
                                                {
                                                        this.zeroframedarray[i][j] = minlabel;
                                                        this.EQArray[pixel] = minlabel;
                                                }

                                        }

                                        this.zeroframedarray[i][j] = this.EQArray[this.zeroframedarray[i][j]];

                                }
                        }

                }
        }

        public void Connect8Pass1()
        {
                for(int i = 1; i <= this.numrows; i++)
                {
                        for(int j = 1; j <= this.numcols; j++)
                        {
                                int pixel = this.zeroframedarray[i][j];

                                if(pixel>0)
                                {

                                        link = new LinkedList<Integer>();
                                        int a = this.zeroframedarray[i-1][j-1], b = this.zeroframedarray[i-1][j],
                                                        c = this.zeroframedarray[i-1][j+1], d =
this.zeroframedarray[i][j-1];


                                        if(a==0&&b==0&&c==0&&d==0)
                                        {

                                                newLabel++;
                                                this.zeroframedarray[i][j] = newLabel;
                                                this.EQArray[newLabel] = newLabel;
                                        }

                                        else if(this.IsSameLabel(a, b, c, d))
                                        {
                                                        this.zeroframedarray[i][j] = link.get(0);
```

```
                                        }

                                        else
                                        {
                                                int min = this.GetMin();
                                                this.zeroframedarray[i][j] = min;
                                                for(int k = 0; k<link.size();k++)
                                                {
                                                        this.EQArray[link.get(k)] = min;
                                                }
                                        }



                                }

                        }
                }
}


                public void Connect8Pass2()
                {
                        for(int i = this.numrows; i >= 0; i--)
                        {
                                for(int j = this.numcols; j >= 0; j--)
                                {
                                        int pixel = this.zeroframedarray[i][j];

                                        if(pixel>0)
                                        {
                                                link = new LinkedList<Integer>();
                                                int e = this.zeroframedarray[i][j+1], f = this.zeroframedarray[i+1][j-1],
                                                        g = this.zeroframedarray[i+1][j], h =
this.zeroframedarray[i+1][j+1];

                                                if(e==0&&f==0&&g==0&&h==0)
                                                {
                                                }

                                                else if(this.IsSameLabel(e, f, g, h)&&pixel==e)
                                                {

                                                }

                                                else
                                                {
                                                        link.add(pixel);
                                                        int min = this.GetMin();
                                                        if(pixel>min)
                                                        {
                                                                this.EQArray[pixel] = min;
                                                                this.zeroframedarray[i][j] = min;
                                                        }
                                                }
```

```
                                        this.zeroframedarray[i][j] = this.EQArray[this.zeroframedarray[i][j]];

                                }
                        }
                }
        }

        public boolean IsSameLabel(int a, int b, int c, int d)
        {
                if(a!=0) link.add(a); if(b!=0) link.add(b);
                if(c!=0) link.add(c); if(d!=0) link.add(d);

                if(link.size()==1)
                        return true;

                for(int i=0; i<link.size(); i++)
                {
                        for(int j=0; j<link.size();j++)
                        {
                                if(link.get(i)!=link.get(j))
                                        return false;
                        }
                }

                return true;
        }

        public int GetMin()
        {
                int min = 9999;
                for(int i=0; i<link.size(); i++)
                {
                        if(link.get(i)<min)
                                min = link.get(i);
                }

                return min;
        }
        public boolean Conn4Pass2Case2(int pixel, int a, int b)
        {
                if(a==0)
                        if(pixel==b)
                                return true;
                        else return false;
                if(b==0)
                        if(a==pixel)
                                return true;
                        else return false;

                 if(a==b||b==pixel||a==pixel)
                                 return true;
                 else return false;

        }

        //public boolean Case3()
```

```
public void ImageReformat(int zero[][], BufferedWriter outputimage, String caption)
{


	int maxlength = Integer.toString(this.newLabel).length();

	try
	{
		outputimage.write(caption+"\n");

		for(int i = 1; i <= this.numrows; i++)
		{
			for(int j = 1; j <= this.numcols; j++)
			{
				int input = zero[i][j];
				int length = Integer.toString(input).length();

				//outputimage.write(Integer.toString(input));

				if(input!=0)
					outputimage.write(Integer.toString(input));

				else
					length = 0;

				for(int k=length; k <= maxlength;k++)
				outputimage.write(" ");

			}
				outputimage.write("\n");
		}
	}

	catch(IOException e)
	{
		e.printStackTrace();
	}
}
public void PrintEQArray(BufferedWriter outputimage, String caption)
{

	try
	{
		outputimage.write(caption+"\nNew Label: "+this.newLabel+"\nEquivalence Table:\n");

		for(int i=1; i<=newLabel;i++)
		{

			outputimage.write(i+" "+this.EQArray[i]+"\n");
		}
			outputimage.write("\n");
	}
	catch(IOException e)
	{
		e.printStackTrace();
	}
```

```
}

public int ManageEqArray()
{
	int readlabel = 0;

	for(int index = 1; index <= this.newLabel; index++)
	{
		if(index != this.EQArray[index])
			this.EQArray[index] = this.EQArray[this.EQArray[index]];

		else
		{
			readlabel = readlabel + 1;
			this.EQArray[index] = readlabel;
		}

	}
	return readlabel;
}

public void Pass3()
{

	this.PropertyFile = new Property[this.truecc+1];
	for(int i=1; i <= this.truecc; i++)
	{
		this.PropertyFile[i] = new Property();
	}

	newmax = truecc;

	for(int i = 1; i <= this.numrows; i++)
	{
		for(int j = 1; j <= this.numcols; j++)
		{
			int pixel = this.zeroframedarray[i][j];
			if(pixel > 0)
			{
				this.zeroframedarray[i][j] = this.EQArray[pixel];
				pixel = this.EQArray[pixel];
				//System.out.println("pixel = "+pixel);

				this.PropertyFile[pixel].label = pixel;


				if(this.PropertyFile[pixel].minrow!=0&&i<this.PropertyFile[pixel].minrow)
					this.PropertyFile[pixel].minrow = i;

				if(this.PropertyFile[pixel].mincol!=0&&j<this.PropertyFile[pixel].mincol)
					this.PropertyFile[pixel].mincol = j;


if(this.PropertyFile[pixel].maxrow!=this.numrows&&i>this.PropertyFile[pixel].maxrow)
					this.PropertyFile[pixel].maxrow = i;
```

```
if(this.PropertyFile[pixel].maxcol!=this.numcols&&j>this.PropertyFile[pixel].maxcol)
                                                this.PropertyFile[pixel].maxcol = j;


                                        this.PropertyFile[pixel].numpixels = this.PropertyFile[pixel].numpixels + 1;

                                }
                        }
                }


}

public void PrintImage(BufferedWriter labelfile)
{

                try
                {
                        if(this.is4) labelfile.write("Using 4 Connected Component Algorithm\n");
                        else labelfile.write("Using 8 Connected Component Algorithm\n");

                        labelfile.write(this.numrows+" "+this.numcols+" "+this.newmin+" "+this.newmax+"\n");

                        int maxlength = Integer.toString(this.truecc).length();

                        for(int i = 1; i <= this.numrows; i++)
                        {

                                for(int j = 1; j <= this.numcols; j++)
                                {
                                        int pixel = this.zeroframedarray[i][j];

                                        int length = Integer.toString(pixel).length();

                                        labelfile.write(Integer.toString(pixel));

                                        for(int k=length; k <= maxlength;k++)
                                                labelfile.write(" ");
                                }
                                labelfile.write("\n");
                        }

                }

                catch(IOException e)
                {
                        e.printStackTrace();
                }

}

public void CCPropertyFile(BufferedWriter propertyfile)
{
                try
                {
```

```
                        if(this.is4) propertyfile.write("Using 4 Connected Component Algorithm\n");
                        else propertyfile.write("Using 8 Connected Component Algorithm\n");

                        propertyfile.write(this.numrows+" "+this.numcols+" "+this.newmin+" "+this.newmax+"\n"
                                        + "Number of Connected Components: "+this.truecc+"\n");

                        for(int i = 1; i <= this.truecc;i++)
                        {
                                propertyfile.write(this.PropertyFile[i].label+"\n"+this.PropertyFile[i].numpixels+"\n"+
                                 (this.PropertyFile[i].minrow-1)+" "+(this.PropertyFile[i].mincol
1)+"\n"+(this.PropertyFile[i].maxrow-1)+" "
                                +(this.PropertyFile[i].maxcol-1)+"\n_____\n");
                        }
                }

                catch(IOException e)
                {
                        e.printStackTrace();
                }


        }

        public void DrawBoxes()
        {
                for(int i = 1; i <= this.truecc;i++)
                {
                        int mincol = this.PropertyFile[i].mincol;
                        int minrow = this.PropertyFile[i].minrow;
                        int label = this.PropertyFile[i].label;

                        while(mincol <= this.PropertyFile[i].maxcol)
                        {
                                this.zeroframedarray[this.PropertyFile[i].minrow][mincol] = label;
                                mincol++;
                        }

                        mincol = this.PropertyFile[i].mincol;
```

```
                        while(mincol <= this.PropertyFile[i].maxcol)
                        {
                                this.zeroframedarray[this.PropertyFile[i].maxrow][mincol] = label;
                                   mincol++;
                        }

                        while(minrow <= this.PropertyFile[i].maxrow)
                        {
                                this.zeroframedarray[minrow][this.PropertyFile[i].mincol] = label;
                                   minrow++;
                        }

                        minrow = this.PropertyFile[i].minrow;

                        while(minrow <= this.PropertyFile[i].maxrow)
                        {
                                this.zeroframedarray[minrow][this.PropertyFile[i].maxcol] = label;
                                   minrow++;
                        }


                }

        }

}
```

```
//Property
public class Property
{
        public int label = 0;
        public int numpixels = 0;
        public int minrow= 9999;
        public int mincol = 9999;
        public int maxrow = 0;
        public int maxcol = 0;


}
```

OUTPUT

RFprettyPrintFile for 4-connectness for data2

Using 4 Connected Component Algorithm
Pass 1

1 1 1 1 1 1 1 1 1 2 1 1 1 1 1 1 1 1 1 1 3 3 3 2 2 1 1 4 3 3 3 3
2
1 1 5 4 3 3
1 1 6 5 3 3
1 1 6 5 3 3
7 1 1 6 5 8 3 3
9 1 1 5 5 8 3
10 1 1 5 5 5 5 5
11 1 1 1 5 5 5
11 1 1 1 1
1
12 13 13 13
14 15 13 13 13 13
16 17 15 13 13 13 13 18 18 19 20 17 13 13 18 18 21 20 17 13 13
22 22 22 23 23 20 17 13 13 24 24 22 22 25 23 20 17 13 13 24 24 22 22 26
25 27 20 17 13 13 24 24 22 22 28 26 29 20 17 17 17 17 17 13 13 24 24
22 22 28 26 29 20 17 17 17 17 17 17 13 13 24 24 22 22 22 22 29 20 17 13 13
24 24 22 22 22 22 30 20 17 13 13 24 24 22 22 31 30 20 17 13 13 24 24 22
22 32 31 30 20 17 13 13 33 33 34 24 24 22 22 32 31 30 20 17 13 13 33 33 24
24 22 22 32 31 30 13 33 33 33 24 22 22 31 31 35 36 37 33 33 24 24 Pass 1
New Label: 37
Equivalence Table:
1 1
2 2
3 3
4 3
5 4
6 5
7 7
8 5
9 9
10 10
11 1
12 12
13 13
14 14
15 13
16 16
17 13
18 18
19 19
20 17
21 21
22 22
23 23
24 24
25 23
26 25
27 27

28 22
29 29
30 30
31 31
32 31
33 24
34 24
35 35
36 36
37 37

Pass 2

1 1 1 1 1 1 1 1 1 2 1 1 1 1 1 1 1 1 1 1 3 3 3 2 2 1 1 3 3 3 3 3
2
1 1 3 3 3 3
1 1 3 3 3 3
1 1 3 3 3 3
7 1 1 3 3 5 3 3
9 1 1 3 3 5 3
10 1 1 3 3 3 3 3 3
1 1 1 1 3 3 4
1 1 1 1 1
1
12 13 13 13
14 13 13 13 13 13
16 13 13 13 13 13 13 13 18 18 19 13 13 13 13 13 18 18 21 13 13 13 13
22 22 22 23 23 13 13 13 13 24 24 22 22 23 23 13 13 13 13 24 24 22 22 25
23 27 13 13 13 13 24 24 22 22 22 23 29 13 13 13 13 13 13 13 13 13 24 24
22 22 22 25 29 13 13 13 13 13 13 13 13 13 24 24 22 22 22 22 29 13 13 13 13
24 24 22 22 22 22 30 13 13 13 13 24 24 22 22 31 30 13 13 13 13 24 24 22
22 31 31 30 13 13 13 13 24 24 24 24 24 22 22 31 31 30 13 13 13 13 24 24 24
24 22 22 31 31 30 13 24 24 24 24 22 22 31 31 35 36 37 24 24 24 24 Pass 2
New Label: 37
Equivalence Table:
1 1
2 2
3 3
4 3
5 3
6 3
7 7
8 5
9 9
10 10
11 1
12 12
13 13
14 14
15 13
16 16
17 13
18 18
19 19
20 13
21 21
22 22
23 23
24 24
25 23
26 25
27 27
28 22
29 29
30 30
31 31
32 31
33 24
34 24
35 35
36 36
37 37

After Equivalence Table Management
New Label: 37
Equivalence Table:
1 1
2 2
3 3
4 3
5 3
6 3
7 4
8 3
9 5
10 6
11 1
12 7
13 8
14 9
15 8
16
10
17 8
18
11
19
12
20 8
21
13
22
14
23
15
24
16
25
15
26
15
27
17
28
14
29
18
30
19
31
20
32
20
33
16
34
16
35
21
36
22
37
23
Pass 3

1 1 1 1 1 1 1 1 1 1 2 1 1 1 1 1 1 1 1 1 1 3 3 3 2 2 1 1 3 3 3 3 3
2 1 1 3 3 3 3
1 1 3 3 3 3 1 1 3 3 3 3 4 1 1 3 3 3 3 3 5 1 1 3 3 3 3 6 1
1 3 3 3 3 3
1 1 1 1 3 3 3
1 1 1 1 1
1
7 8 8 8
9 8 8 8 8 8
10 8 8 8 8 8 8 11 11 12 8 8 8 8 11 11 13 8 8 8 8
14 14 14 15 15 8 8 8 8 16 16 14 14 15 15 8 8 8 8 16 16 14 14 15 15 17
8 8 8 8 16 16 14 14 14 15 18 8 8 8 8 8 8 8 8 8 8 16 16 14 14 14 15 18 8 8

8 8 8 8 8 8 8 16 16 14 14 14 14 18 8 8 8 8 16 16 14 14 14 14 19 8 8 8 8
16 16 14 14 20 19 8 8 8 8 16 16 14 14 20 20 19 8 8 8 8 16 16 16 16 16
14 14 20 20 19 8 8 8 8 16 16 16 16 14 14 20 20 19 8 16 16 16 16 14 14
20 20 21 22 23 16 16 16 16 Pass 3
New Label: 37
Equivalence Table:
1 1
2 2
3 3
4 3
5 3
6 3
7 4
8 3
9 5
10 6
11 1
12 7
13 8
14 9
15 8
16 10
17 8
18 11
19 12
20 8
21 13
22 14
23 15
24 16
25 15
26 15
27 17
28 14
29 18
30 19
31 20
32 20
33 16
34 16
35 21
36 22
37 23

True number of Connected Components: 23
Result of Draw Box

1 1 1 1 1 1 1 1 1 1 2 2
1 1 1 1 1 1 1 1 1 1 3 3 3 3 3 3 3 3 3 3 2
1 1 1 1 3 3 3 3 3 3 3 2
1 1 1 1 3 3 3 3 3 3
1 1 1 1 3 3 3 3
1 1 1 1 3 3 3 3
4 1 1 1 3 3 3 3 3
1 5 1 1 1 3 3 3 3 3
1 6 1 1 1 3 3 3 3 3 3
1 1 1 1 1 1 3 3 3 3 3 3 3 3 3
1 1 1 1 1 1 1
1 1 1 1 1 1 1 1 1 1
7 8 8 8 8 8 8 8 8
9 8 8 8 8 8 8 8
10 8 8 8 8 8 8 8 8 11 11
12 8 8 8 8 11 11
13 8 8 8 8
14 14 14 14 15 15 15 15 8 8 8 8 16 16 16 16 16 14 14 15 15 15 15 8 8 8 8
16 16 16
14 14 15 15 15 17 8 8 8 8 16 16 16
14 14 14 15 15 18 8 8 8 8 8 8 8 8 8 8 16 16 16
14 14 14 15 15 15 15 18 8 8 8 8 8 8 8 8 8 8 16 16 16
14 14 14 14 14 18 8 8 8 8 16 16 16
14 14 14 14 19 8 8 8 8 16 16 16
14 14 14 20 20 20 19 8 8 8 8 16 16 16

14 14 14 20 20 20 19 8 8 8 8 16 16 16 16 16
14 14 14 20 20 20 19 8 8 8 8 16 16 16 16 16
14 14 14 20 20 20 19 8 8 8 8 8 8 8 8 8 16 16 16 16 16 16   14 14 14 14 14 20 20
20 21 22 23 16 16 16 16 16 16

labelFile for 4-connectness for data2

Using 4 Connected Component Algorithm
30 35 0 23
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 1 1
1 1 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 2 0 0 0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 0 0 0 0 0 0 0
0 0 0 3 3 3 0 0 2 2 0 0 0 0 0 0 0 0 0 0 0 0 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 3 3 3 3 3 0 2 0 0
0 0 0 0 0 0 0 0 0 0 0 1 1 0 0 0 0 0 0 0 0 0 0 0 0 3 3 0 0 0 3 3 0 0 0 0 0 0 0 0 0 0 0 0 0 0
1 1 0 0 0 0 0 0 0 0 0 0 0 3 3 0 0 0 0 0 3 3 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 0 0 0 0 0 0 0 0 0
0 0 3 3 0 0 0 0 0 3 3 0 0 0 0 0 0 0 0 0 4 0 0 0 1 1 0 0 0 0 0 0 0 0 0 0 0 3 3 0 0 0 0 3 3 3
0 0 0 0 0 0 0 0 0 0 5 0 0 1 1 0 0 0 0 0 0 0 0 0 0 0 0 3 3 0 0 0 3 3 0 0 0 0 0 0 0 0 0 0 0 0
6 0 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 3 3 3 3 3 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 0 0 0 0 0 0
0 0 0 0 0 0 0 3 3 3 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 7 0 0 0 0 0 0 0 0 8 8 8 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 9 0 0 0 0 0 0
0 0 8 8 8 8 8 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 10 0 0 0 0 0 0 0 0 8 8 8 0 8 8 8 0 0
0 0 0 0 0 11 11 0 0
0 0 0 0 0 0 0 12 0 0 0 0 0 0 0 0 8 8 0 0 0 0 0 8 8 0 0 0 0 0 0 11 11 0 0 0 0 0 0 0 0 13 0 0 0 0
0 0 0 0 0 8 8 0 0 0 0 0 8 8 0 0 0 0 0 0 0 0 0 0 0 0 0 14 14 14 0 0 0 15 15 0 0 0 0 0 8 8 0 0 0
0 0 8 8 0 0 0 0 0 0 16 16 0 0 0 0 0 14 14 0 0 0 15 15 0 0 0 0 0 0 8 8 0 0 0 0 0 8 8 0 0 0 0 0 0
16 16 0 0 0 0 0 14 14 0 0 15 15 0 0 0 17 0 0 0 8 8 0 0 0 0 0 8 8 0 0 0 0 0 0 16 16 0 0 0 0 0
14 14 0 14 15 0 0 0 0 0 18 0 0 8 8 8 8 8 8 8 8 8 0 0 0 0 0 0 16 16 0 0 0 0 0 14 14 0 14 15 0
0 0 0 0 18 0 0 8 8 8 8 8 8 8 8 8 0 0 0 0 0 0 16 16 0 0 0 0 0 14 14 14 14 0 0 0 0 0 0 18 0 0 8
8 0 0 0 0 0 8 8 0 0 0 0 0 0 16 16 0 0 0 0 0 14 14 0 14 14 0 0 0 0 0 0 19 0 8 8 0 0 0 0 0 8 8 0
0 0 0 0 0 16 16 0 0 0 0 0 14 14 0 0 0 0 20 0 0 0 0 19 0 8 8 0 0 0 0 0 8 8 0 0 0 0 0 0 16 16 0
0 0 0 0 14 14 0 0 0 20 20 0 0 0 0 19 0 8 8 0 0 0 0 0 8 8 0 0 16 16 0 16 16 16 0 0 0 0 0 14
14 0 0 0 20 20 0 0 0 0 19 0 8 8 0 0 0 0 0 8 8 0 0 0 16 16 0 16 16 0 0 0 0 0 14 14 0 0 0 20 20
0 0 0 0 19 0 0 0 0 0 0 0 0 8 0 0 0 0 16 16 16 16 0 0 0 0 0 0 14 14 0 0 0 0 20 20 0 0 21 0 22
0 0 0 0 0 0 0 0 23 0 0 0 0 16 16 16 16 0 0

propertyFile for 4-connectness for data2

Using 4 Connected Component Algorithm
30 35 0 23
Number of Connected Components: 23
1
44
1 5
12 14
____
2
4
1 30
3 31
____
3
37
2 22
10 30
____
4
1
7 5
7 5
____
5
1
8 6
8 6

_____
6
1
9 7
9 7
_____
7
1
13 10
13 10
_____
8
73
13 16
28 24
_____
9
1
14 9
14 9
_____
10
1
15 8
15 8
_____
11
4
15 31
16 32
_____
12
1
16 7
16 7
_____
13
1
17 6
17 6
_____
_ 14
31
18 3
29 7
_____
_ 15
8
18 7
22 10
_____
_ 16
33
18 27
29 32
_____
_ 17
1
20 12
20 12
_____
_ 18
3
21 13
23 13

_____
_ 19
5
24 14
28 14
_____
_ 20
9
25 8
29 10
_____
_ 21
1
29 13
29 13
_____
_ 22
1
29 15
29 15
_____
_ 23
1
29 24
29 24
_____

RFprettyPrintFile for 8-connectness for data2 Using 8 Connected Component Algorithm

Pass 1

1 1 1 1 1 1 1 1 1 1 2 1 1 1 1 1 1 1 1 1 1 3 3 3 2 2 1 1 3 3 3 3 3
2 1 1 3 3 3 2 1 1 3 3 2 2 1 1 3 3 2 2 4 1 1 3 3 2 2 2 4 1 1 3 3
2 2 4 1 1 3 3 3 2 2 1 1 1 1 3 2 2 1 1 1 1 1
1
1 5 5 5
1 5 5 5 5 5
1 5 5 5 5 5 5 6 6 1 5 5 5 5 6 6 1 5 5 5 5
7 7 1 8 8 5 5 5 5 9 9 7 1 8 8 5 5 5 5 9 9 1 1 8 8 10 5 5 5 5 9 9 1 1
8 8 10 5 5 5 5 5 5 5 5 9 9 1 1 8 8 10 5 5 5 5 5 5 5 5 9 9 1 1 1 1
10 5 5 5 5 9 9 1 1 1 1 10 5 5 5 5 9 9 1 1 1 1 10 5 5 5 5 9 9 1 1 1 1
11 10 5 5 5 5 12 12 9 9 9 1 1 1 1 11 11 10 5 5 5 5 12 9 9 9 1 1 11 11
10 5 9 9 9 9 1 1 11 11 10 10 5 9 9 9 9 Pass 1

New Label: 12

Equivalence Table:

1 1
2 2
3 2
4 1
5 5
6 6
7 1
8 1
9 9
10 10
11 11
12 9

Pass 2

1 1 1 1 1 1 1 1 1 1 2 1 1 1 1 1 1 1 1 1 1 2 2 2 2 2 1 1 2 2 2 2 2
2 1 1 2 2 2 2 1 1 2 2 2 2 1 1 2 2 2 2 1 1 1 2 2 2 2 2 1 1 1 2 2
2 2 1 1 1 2 2 2 2 2 1 1 1 1 2 2 2 1 1 1 1 1
1
1 5 5 5
1 5 5 5 5 5

1 5 5 5 5 5 5 6 6 1 5 5 5 5 6 6 1 5 5 5 5
1 1 1 1 1 5 5 5 5 9 9 1 1 1 1 5 5 5 5 9 9 1 1 1 1 10 5 5 5 5 9 9 1 1
1 1 10 5 5 5 5 5 5 5 5 9 9 1 1 1 1 10 5 5 5 5 5 5 5 5 9 9 1 1 1 1
10 5 5 5 5 9 9 1 1 1 1 10 5 5 5 5 9 9 1 1 11 10 5 5 5 5 9 9 1 1 11
11 10 5 5 5 5 9 9 9 9 1 1 11 11 10 5 5 5 5 9 9 9 9 1 1 11 11 10 5
9 9 9 9 1 1 11 11 10 10 5 9 9 9 9 Pass 2
New Label: 12
Equivalence Table:
1 1
2 2
3 2
4 1
5 5
6 6
7 1
8 1
9 9
10 10
11 11
12 9

After Equivalence Table Management
New Label: 12
Equivalence Table:
1 1
2 2
3 2
4 1
5 3
6 4
7 1
8 1
9 5
10 6
11 7
12 5
Pass 3

1 1 1 1 1 1 1 1 1 1 2 1 1 1 1 1 1 1 1 1 1 2 2 2 2 2 1 1 2 2 2 2 2
2 1 1 2 2 2 2 1 1 2 2 2 2 1 1 2 2 2 2 1 1 1 2 2 2 2 2 1 1 1 2 2
2 2 1 1 1 2 2 2 2 2 1 1 1 1 2 2 2 1 1 1 1 1
1
1 3 3 3
1 3 3 3 3 3
1 3 3 3 3 3 3 4 4 1 3 3 3 3 4 4 1 3 3 3 3
1 1 1 1 1 3 3 3 3 5 5 1 1 1 1 3 3 3 3 5 5 1 1 1 1 6 3 3 3 3 5 5
1 1 1 1 6 3 3 3 3 3 3 3 3 5 5 1 1 1 1 6 3 3 3 3 3 3 3 3 5 5 1 1
1 1 6 3 3 3 3 5 5 1 1 1 1 6 3 3 3 3 5 5 1 1 7 6 3 3 3 3 5 5 1 1 7 7
6 3 3 3 3 5 5 5 5 1 1 7 7 6 3 3 3 3 5 5 5 5 1 1 7 7 6 3 5 5 5 5 1
1 7 7 6 6 3 5 5 5 5 Pass 3
New Label: 12
Equivalence Table:
1 1
2 2
3 2
4 1
5 3
6 4
7 1
8 1
9 5
10 6
11 7
12 5

True number of Connected Components: 7
Result of Draw Box

1 1 1 1 1 1 1 1 1 1 1 1 2 2 2 2 2 2 2 2 2 2 1 1 1 1 1 1 1 1 1 1 1 2 2 2
2 2 2 1 1 1 1 2 2 2 2 2 2 2 1 1 1 1 2 2 2 2 2 2 1 1 1 1 2 2 2 2 2 1
1 1 1 2 2 2 2 2 1 1 1 1 1 2 2 2 2 2 2 1 1 1 1 1 2 2 2 2 2 2 1 1 1 1 1
2 2 2 2 2 2 2 1 1 1 1 1 1 2 2 2 2 2 2 2 2 2 2 1 1 1 1 1 1 1
1 1 1
1 1 1 3 3 3 3 3 3 3 3 3
1 1 1 3 3 3 3 3 3 3
1 1 1 3 3 3 3 3 3 3 3 4 4 1 1 1 3 3 3 3 4 4 1 1 1 3 3 3 3
1 1 1 1 1 1 3 3 3 3 5 5 5 5 5 5 1 1 1 1 1 3 3 3 3 5 5 5 1 1 1 1 6 6 6 6 3 3 3
3 5 5 5 1 1 1 1 6 6 1 6 3 3 3 3 3 3 3 3 3 5 5 5 1 1 1 1 6 6 1 6 3 3 3 3 3 3 3
3 3 5 5 5 1 1 1 1 6 6 1 6 3 3 3 3 5 5 5 1 1 1 1 6 1 6 3 3 3 3 5 5 5 1 1 7 7 7
6 1 6 3 3 3 3 5 5 5 1 1 7 7 7 6 1 6 3 3 3 3 5 5 5 5 5 1 1 7 7 7 6 1 6 3 3 3 3
5 5 5 5 5 1 1 7 7 7 6 1 6 3 3 3 5 5 5 5 5 5 5 1 1 1 1 1 7 7 7 1 6 6 6 6 3 3 3 3
3 3 3 3 3 5 5 5 5 5 5 labelFile for 8-connectness for data2

Using 8 Connected Component Algorithm
30 35 0 7
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 2 0 0 0 0
0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 0 0 0 0 0 0 0 0 0 0 2 2 2 0 0 2 2 0 0 0
0 0 0 0 0 0 0 0 0 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 2 2 2 2 2 0 2 0 0 0 0
0 0 0 0 0 0 0 0 0 1 1 0 0 0 0 0 0 0 0 0 0 0 0 2 2 0 0 0 2 2 0 0 0 0 0
0 0 0 0 0 0 0 0 0 1 1 0 0 0 0 0 0 0 0 0 0 0 2 2 0 0 0 0 2 2 0 0 0 0 0
0 0 0 0 0 0 0 0 0 1 1 0 0 0 0 0 0 0 0 0 0 0 2 2 0 0 0 0 2 2 0 0 0 0 0
0 0 0 0 0 1 0 0 0 1 1 0 0 0 0 0 0 0 0 0 0 0 2 2 0 0 0 0 2 2 2 0 0 0 0
0 0 0 0 0 0 1 0 0 1 1 0 0 0 0 0 0 0 0 0 0 0 0 2 2 0 0 0 2 2 0 0 0 0 0
0 0 0 0 0 0 0 1 0 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 2 2 2 2 2 0 0 0 0 0 0
0 0 0 0 0 0 0 0 1 1 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 2 2 2 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 1 1 1 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 3 3 3 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 3 3 3 3 3 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 3 3 3 0 3 3 3 0 0 0 0 0 0 0 4 4 0 0
0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 3 3 0 0 0 0 0 3 3 0 0 0 0 0 0 4 4 0 0
0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 3 3 0 0 0 0 0 3 3 0 0 0 0 0 0 0 0 0 0
0 0 0 1 1 1 0 0 0 1 1 0 0 0 0 0 3 3 0 0 0 0 0 3 3 0 0 0 0 0 0 5 5 0 0
0 0 0 1 1 0 0 0 1 1 0 0 0 0 0 0 3 3 0 0 0 0 0 3 3 0 0 0 0 0 0 5 5 0 0
0 0 0 1 1 0 0 1 1 0 0 0 6 0 0 0 3 3 0 0 0 0 0 3 3 0 0 0 0 0 0 5 5 0 0
0 0 0 1 1 0 1 1 0 0 0 0 0 6 0 0 3 3 3 3 3 3 3 3 3 0 0 0 0 0 0 5 5 0 0
0 0 0 1 1 0 1 1 0 0 0 0 0 6 0 0 3 3 3 3 3 3 3 3 3 0 0 0 0 0 0 5 5 0 0
0 0 0 1 1 1 1 0 0 0 0 0 0 6 0 0 3 3 0 0 0 0 0 3 3 0 0 0 0 0 0 5 5 0 0
0 0 0 1 1 0 1 1 0 0 0 0 0 0 6 0 3 3 0 0 0 0 0 3 3 0 0 0 0 0 0 5 5 0 0
0 0 0 1 1 0 0 0 0 7 0 0 0 0 6 0 3 3 0 0 0 0 0 3 3 0 0 0 0 0 0 5 5 0 0
0 0 0 1 1 0 0 0 0 7 7 0 0 0 6 0 3 3 0 0 0 0 0 3 3 0 0 5 5 0 5 5 5 0 0
0 0 0 1 1 0 0 0 0 7 7 0 0 0 6 0 3 3 0 0 0 0 0 3 3 0 0 0 5 5 0 5 5 0 0
0 0 0 1 1 0 0 0 0 7 7 0 0 0 6 0 0 0 0 0 0 0 0 3 0 0 0 0 5 5 5 5 0 0 0
0 0 0 1 1 0 0 0 0 7 7 0 0 6 0 6 0 0 0 0 0 0 0 0 3 0 0 0 0 5 5 5 5 0 0

propertyFile for 8-connectness for data2

Using 8 Connected Component
Algorithm 30 35 0 7
Number of Connected Components:
7 1
91
1 3
29 14

2
41

1 22
10 31

_____

3
74
13 16
29 24

_____

4
4
15 31
16 32

_____

5
33
18 27
29 32

_____

6
11
20 12
29 15

_____

7
9
25 8
29 10

_____

RFprettyPrintFile for 4-connectness for data3

Using 4 Connected Component Algorithm
Pass 1
1 2 3 4
5 1 6 2 3 3 4
7 5 5 8 1 1 2 3 3 9 9 9 9 10 4 5 5 5 5 1 1 1 11 12 9 13 10 4 5 5 5 1 1 1 14 14 14 15 16 17 18 10 19 5 5 1 1 1 1 20
14 14 14 14 15 16 21 18 5 5 5 1 1 1 1 22 20 14 14 14 14 14 15 16 23 21 21 24 5 5 1 1 1 22 14 14 14 14 14 25
16 16 16 16 16 16 26 5 1 1 22 22 14 14 14 14 14 27 16 16 16 16 16 16 16 26 26 26 5 1 28 28 29 22 14 14 14 30
27 27 16 16 16 16 16 16 26 26 5 1 28 31 32 33 14 14 14 34 30 27 16 16 16 16 16 16 16 35 26 1 31 31 36 32 33 14
37 30 16 16 16 16 16 16 31 36 38 33 39 40 41 41 16 16 16 16 42 31 39 39 39 43 41
31 44 44 44
45 46 47 44 44 44 44 44 48 48 49 50 50 50 51 47 44 44 44 44 48 48 52 53 50 50 50 50 51 47 44 44 44 44
44 44
54 55 53 50 50 50 50 50 51 47 44 44 44 44 44
56 57 55 50 50 50 50 58 47 44 44 44
59 59 59 60 61 61 55 55 50 50 50 50 50 62 44 44 44 63 63 63 63 64 64 59 65 61 55 50 50 50 50 66 67 44 63 63 63 64 64 64
68 65 69 70 50 50 50 66 66 71 72 63 63 63 73 65 65 74 75 76 72 77 77 63 63 78 79 80 81 82 75 83 84 76 76 85 86 86 63 63
63 87 88 88 79 79 82 89 84 76 76 90 91 86 63 63 87 87 79 92 93 93 89 84 84 76 76 76 76 94 86 63 63 63 87 87 95 79 79
96 97 93 89 84 84 76 76 76 98 94 86 63 63 99 100 95 79 79 79 101 93 89 84 76 76 76 98 94 86 63 63 63 63 63 102 102 100
95 79 79 79 79 101 103 89 84 84 76 76 98 94 63 63 63 63 104 Pass 1
New Label: 104
Equivalence Table:
1 1
2 2
3 3
4 4
5 1
6 2
7 5
8 5

9 9
10 4
11 11
12 12
13 13
14 14
15 15
16 16
17 17
18 10
19 19
20 14
21 16
22 14
23 16
24 24
25 16
26 5
27 27
28 28
29 29
30 30
31 31
32 32
33 33
34 30
35 35
36 32
37 37
38 33
39 39
40 39
41 16
42 31
43 43
44 44
45 45
46 46
47 44
48 48
49 49
50 50
51 47
52 52
53 50
54 54
55 50
56 56
57 57
58 58
59 59
60 60
61 61
62 62
63 63
64 64
65 65
66 66
67 67
68 65
69 69
70 70
71 71
72 72
73 73
74 74
75 75
76 76
77 77
78 63
79 79

80 80
81 81
82 75
83 83
84 76
85 85
86 63
87 87
88 88
89 84
90 76
91 91
92 92
93 89
94 86
95 79
96 96
97 97
98 94
99 63
100 95
101
101
102
100
103
103
104
104
Pass 2
1 2 3 4
1 1 2 2 3 3 4
1 1 1 5 1 1 2 3 3 9 9 9 9 4 4 1 1 1 1 1 1 1 11 12 9 13 4 4 1 1 1 1 1 14 14 14 15 16 17 10 4 19 1 1 1 1 1 1 14 14
14 14 14 15 16 16 10 1 1 1 1 1 1 1 14 14 14 14 14 14 15 16 16 16 16 24 1 1 1 1 1 14 14 14 14 14 14 16 16
16 16 16 16 16 1 1 1 1 14 14 14 14 14 14 27 16 16 16 16 16 16 16 1 5 5 1 1 28 28 29 14 14 14 14 30 27 27 16
16 16 16 16 16 1 5 1 1 28 31 32 33 14 14 14 30 30 27 16 16 16 16 16 16 16 35 5 1 31 31 32 32 33 14 37 30 16
16 16 16 16 16 31 32 33 33 39 39 16 16 16 16 16 16 31 31 39 39 39 43 16
31 44 44 44
45 46 44 44 44 44 44 44 48 48 49 50 50 50 44 44 44 44 44 44 48 48 52 50 50 50 50 50 44 44 44 44 44 44
44 44
54 50 50 50 50 50 50 50 44 44 44 44 44 44 44 44
56 57 50 50 50 50 50 50 58 44 44 44 44
59 59 59 60 61 61 50 50 50 50 50 50 50 62 44 44 44 63 63 63 63 64 64 59 65 61 50 50 50 50 50 66 67 44 63 63 63 64 64
64 65 65 69 70 50 50 50 66 66 71 72 63 63 63 73 65 65 74 75 76 72 77 77 63 63 63 79 80 81 75 75 83 76 76 76 85 63 63
63 63 63 87 88 88 79 79 75 76 76 76 76 76 91 63 63 63 87 87 79 92 76 76 76 76 76 76 76 76 76 86 63 63 63 63 87 87 79
79 79 96 97 76 76 76 76 76 76 76 86 86 63 63 63 63 79 79 79 79 79 101 76 76 76 76 76 76 86 86 63 63 63 63 63 63 79 79
79 79 79 79 79 79 101 103 76 76 76 76 76 63 86 63 63 63 63 104 Pass 2
New Label: 104
Equivalence Table:
1 1
2 2
3 3
4 4
5 1
6 2
7 1
8 5
9 9
10 4
11 11
12 12
13 13
14 14
15 15
16 16
17 17
18 10
19 19
20 14
21 16
22 14
23 16
24 24

25 16
26 1
27 27
28 28
29 29
30 30
31 31
32 32
33 33
34 30
35 35
36 32
37 37
38 33
39 39
40 39
41 16
42 31
43 43
44 44
45 45
46 46
47 44
48 48
49 49
50 50
51 44
52 52
53 50
54 54
55 50
56 56
57 57
58 58
59 59
60 60
61 61
62 62
63 63
64 64
65 65
66 66
67 67
68 65
69 69
70 70
71 71
72 72
73 73
74 74
75 75
76 76
77 77
78 63
79 79
80 80
81 81
82 75
83 83
84 76
85 85
86 63
87 87
88 88
89 76
90 76
91 91
92 92
93 76
94 86
95 79

96 96
97 97
98 86
99 63
100 79
101 101
102 79
103 103
104 104

After Equivalence Table Management
New Label: 104
Equivalence Table:
1 1
2 2
3 3
4 4
5 1
6 2
7 1
8 1
9 5
10 4
11 6
12 7
13 8
14 9
15
10
16
11
17
12
18 4
19
13
20 9
21
11
22 9
23
11
24
14
25
11
26 1
27
15
28
16
29
17
30
18
31
19
32
20
33
21
34
18
35
22
36
20
37
23
38
21
39
24
40
24

41
11
42
19
43
25
44
26
45
27
46
28
47
26
48
29
49
30
50
31
51
26
52
32
53
31
54
33
55
31
56
34
57
35
58
36
59
37
60
38
61
39
62
40
63
41
64
42
65
43
66
44
67
45
68
43
69
46
70
47
71
48
72
49
73
50
74
51
75
52
76
53
77
54
78
41
79
55

80
56
81
57
82
52
83
58
84 53
85 59
86 41
87 60
88 61
89 53
90 53
91 62
92 63
93 53
94 41
95 55
96 64
97 65
98 41
99 41
100 55
101 66
102 55
103 67
104 68

Pass 3
1 2 3 4
1 1 2 2 3 3 4 1 1 1 1 1 1 2 3 3 5 5 5 5 4 4 1 1 1 1 1 1 1 6 7 5 8 4 4 1 1 1 1 1 9 9 9 10 11 12 4 4 13 1 1 1 1 1 1
9 9 9 9 9 10 11 11 4 1 1 1 1 1 1 1 9 9 9 9 9 9 9 10 11 11 11 11 14 1 1 1 1 1 9 9 9 9 9 9 9 11 11 11 11 11 11 11
1 1 1 1 9 9 9 9 9 9 9 15 11 11 11 11 11 11 11 1 1 1 1 1 16 16 17 9 9 9 9 18 15 15 11 11 11 11 11 11 1 1 1 1 16
19 20 21 9 9 9 18 18 15 11 11 11 11 11 11 11 22 1 1 19 19 20 20 21 9 23 18 11 11 11 11 11 11 19 20 21 21 24
24 11 11 11 11 11 11 19 19 24 24 24 25 11
19 26 26 26
27 28 26 26 26 26 26 29 29 30 31 31 31 26 26 26 26 26 26 29 29 32 31 31 31 31 31 26 26 26 26 26 26
26 26 33 31 31 31 31 31 31 31 26 26 26 26 26 26 26
34 35 31 31 31 31 31 36 26 26 26 26
37 37 37 38 39 39 31 31 31 31 31 31 31 40 26 26 26 41 41 41 41 42 42 37 43 39 31 31 31 31 31 44 45 26 41 41 41 42
42 42 43 43 46 47 31 31 31 44 44 48 49 41 41 41 50 43 43 51 52 53 49 54 54 41 41 41 55 56 57 52 52 58 53 53 53 59
41 41 41 41 41 60 61 61 55 55 52 53 53 53 53 53 62 41 41 41 60 60 55 63 53 53 53 53 53 53 53 53 53 41 41 41 41 41
60 60 55 55 55 64 65 53 53 53 53 53 53 53 41 41 41 41 41 41 55 55 55 55 55 66 53 53 53 53 53 53 41 41 41 41 41 41
41 41 55 55 55 55 55 55 55 55 66 67 53 53 53 53 53 41 41 41 41 41 41 68 Pass 3
New Label: 104
Equivalence Table:
1 1
2 2
3 3
4 4
5 1
6 2
7 1
8 1
9 5
10 4
11 6
12 7
13 8
14 9
15 10
16 11
17 12
18 4
19 13
20 9
21
11
22 9
23
11
24

64
42
65
43
66
44
67
45
68
43
69
46
70
47
71
48
72
49
73
50
74
51
75
52
76
53
77
54
78
41
79
55
80
56
81
57
82
52
83
58
84
53
85
59
86
41
87
60
88
61
89
53
90
53
91
62
92
63
93
53
94
41
95 55
96 64
97 65
98 41
99 41
100 55
101 66
102 55
103 67
104 68

True number of Connected Components: 68
Result of Draw Box
1 1 1 1 1 1 1 1 1 1 2 2 3 3 4 4 4
1 1 1 1 2 2 3 3 4 4

1 1 1 1 1 1 2 2 3 3 5 5 5 5 4 4 4 1 1 1 1 1 1 1 1 1 6 7 5 5 5 5 8 4 4 4 1 1 1 1 1 1 9 9 9 9 9 9 9 9 9 10 11 11 11 11 12 11 11
11 11 11 1 13 1 1 1 1 1 1 9 9 9 9 9 9 9 10 11 11 11 4 4 11 1 1 1 1 1 1 1 1 9 9 9 9 9 9 9 9 9 10 11 11 11 11 11 11 11 1 14 1 1 1 1 1
1 9 9 9 9 9 9 9 11 11 11 11 11 11 11 11 11 1 1 1 1 1 1 9 9 9 9 9 9 9 9 9 15 15 11 11 11 11 11 11 11 11 11 1 1 1 1 1 1 16 16 17
9 9 9 9 9 18 18 9 15 15 11 11 11 11 11 11 11 11 11 1 1 1 1 1 16 19 19 19 20 20 21 21 9 9 9 18 18 9 15 15 11 11 11 11 11 11 11
11 1 1 22 1 1 1 1 1 1 19 19 19 20 20 21 21 9 9 9 9 23 18 18 9 11 11 11 11 11 11 11 11 19 19 19 20 20 21 21 24 24 24 11
11 11 11 11 11 11 19 19 19 24 24 24 25 11 11 11 11 11 11 11 11 11 11 19 19 19 26 26 26 26 26 26 26
27 28 26 26 26 26 26 26 26 26 29 29 30 31 31 31 31 31 31 31 26 26 26 26 26 26 26 29 29 32 31 31 31 31 31 31
31 26 26 26 26 26 26 26 26
33 31 31 31 31 31 31 31 26 26 26 26 26 26 26 26
34 35 31 31 31 31 31 36 26 26 26 26 26 26
37 37 37 38 39 39 31 31 31 31 31 31 31 40 26 26 26 26 41 41 41 41 41 41 41 42 42 42 37 37 37 43 43 43 39 39 31 31 31 31 31 31
31 44 44 45 26 26 26 26 26 26 41 26 41 41 42 42 42 43 43 43 46 47 31 31 31 31 31 31 31 44 44 48 49 41 41 41 41 41 50 43 43 43 51
52 52 53 53 53 53 53 53 53 53 53 54 54 41 41 41 41 41 55 55 55 55 55 55 56 55 57 52 52 53 58 53 53 53 53 59 41 41 41 41 41 41
41 60 60 61 61 55 55 55 52 52 53 53 53 53 53 41 62 41 41 41 41 60 55 60 55 55 63 53 53 53 53 53 53 53 53 41 41 41 41 41 41
41 41 60 60 60 55 55 55 64 65 53 53 53 53 53 53 53 53 41 41 41 41 41 41 41 55 55 55 55 55 55 55 66 53 53 53 53 53 53 53
53 41 41 41 41 41 41 41 41 41 55 55 55 55 55 55 55 55 66 67 53 53 53 53 53 53 53 53 41 41 41 41 41 41 41 41 41 68

labelFile for 4-connectness for data3

Using 4 Connected Component Algorithm
30 40 0 68
0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 2 0 0 0 3 0 0 0 0 0 0 0 0 0 0 4 0 0 1 0 0 0 0 1 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 2 2 0 0 0 3 3 0 0 0 0 0 0 0 0 0 4 0 1 1 1 0 0 1 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 2
0 0 0 3 3 0 5 5 5 5 0 0 0 4 4 0 0 0 1 1 1 1 1 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 6 0 0 0 0 7 0 0 5 0 8 0
0 4 4 0 0 0 0 1 1 1 1 0 1 1 0 0 0 0 0 0 0 0 9 9 9 0 0 0 0 10 0 0 0 0 0 0 11 0 12 0 0 4 4 0 0
0 13 0 1 1 0 1 1 1 1 0 0 0 0 0 0 0 0 9 9 9 9 9 0 0 0 10 0 0 0 0 0 0 11 0 0 11 0 4 0 0 0 0 0 0 1 1 1 1 1 1 1 0 0 0 0 0 0 9
9 9 9 9 9 9 0 0 10 0 0 0 0 0 0 11 0 11 11 11 0 0 0 0 0 14 0 0 1 1 1 1 1 0 0 0 0 0 0 0 0 9 0 9 9 9 0 9 9 9 0 0 0 0 0 11
11 11 11 11 11 11 0 0 0 0 0 1 0 0 1 1 1 0 0 0 0 0 0 0 0 0 9 9 9 9 9 9 9 9 0 0 15 0 0 0 0 0 11 11 11 11 11 11 11 0 0 0
0 1 1 1 1 1 0 16 16 0 0 0 0 0 17 0 9 9 0 9 9 0 18 0 15 15 0 0 0 0 11 11 0 11 11 11 11 0 0 0 0 0 0 1 1 1 1 0 16 0 19
0 0 0 20 0 21 0 9 9 9 0 18 18 0 0 15 0 0 0 0 11 11 11 0 11 11 11 11 0 0 0 22 0 1 0 1 0 0 0 19 19 0 20 20 0 21 0 0
9 0 23 0 18 0 0 0 0 0 0 0 11 11 11 11 11 11 0 0 0 0 0 0 0 0 0 0 0 0 0 19 0 0 20 0 21 21 0 24 0 24 0 0 0 0 0 0 0 0
11 11 11 0 11 11 11 0 0 0 0 0 0 0 0 0 0 0 0 19 19 0 0 0 0 0 0 0 24 24 24 0 0 25 0 0 0 0 0 0 11 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 19 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 26 26 26 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 27 0 0 0 28 0 0 0 0
0 0 0 0 0 0 0 0 0 26 26 26 26 26 0 0 0 0 0 0 0 29 29 0 0 0 0 0 0 0 30 0 0 0 0 0 31 31 31 0 0 0 0 0 0 0 0 0 26 26
26 0 26 26 26 0 0 0 0 0 0 29 29 0 0 0 0 0 32 0 0 0 0 0 31 31 31 31 31 0 0 0 0 0 0 0 0 0 26 26 26 26 26 26 26 26
0 0 0 0 0 0 0 0 0 0 0 0 0 0 33 0 0 0 0 0 31 31 31 31 31 31 31 0 0 0 0 0 0 0 26 26 26 26 26 26 26 0 0 0 0 0 0 0 0 0 0
0 0 34 0 0 35 0 0 0 31 0 31 31 31 0 31 0 0 0 0 0 36 0 0 26 26 0 26 26 0 0 0 0 0 0 0 0 0 0 0 37 37 37 0 0 38 0 39 39
0 31 31 31 31 31 31 31 0 0 0 0 0 0 40 0 0 26 26 26 0 41 41 41 41 0 0 0 42 42 0 0 0 37 0 43 0 0 0 39 0 0 31 31
31 31 31 0 0 44 0 0 0 0 0 45 0 0 26 0 0 0 41 0 41 41 0 0 42 42 42 0 0 0 43 43 0 0 46 0 47 0 0 31 31 31 0 0 0 44
44 0 0 0 48 0 49 0 0 0 0 0 41 0 41 41 0 0 0 0 0 0 50 0 0 0 43 43 0 0 0 0 51 0 0 0 0 0 0 52 0 0 0 0 53 0 0 49 0 0 54
54 0 41 0 0 41 0 41 0 0 0 0 0 0 0 0 0 0 55 0 56 0 0 57 0 0 0 0 52 52 0 58 0 53 53 53 0 0 0 59 0 0 0 0 41 41 41 41
41 0 0 0 0 60 0 61 61 0 55 55 0 0 0 0 0 0 0 0 0 52 0 0 0 53 53 0 53 53 0 53 0 0 0 0 62 0 41 41 41 0 0 0 0 0 60 60
0 0 0 55 0 0 0 0 0 0 0 0 0 63 0 0 53 53 53 53 53 53 53 53 0 0 0 0 0 41 41 41 41 41 0 0 0 0 0 60 60 0 55 55 55
0 64 0 0 0 0 0 0 0 0 65 0 53 53 53 53 53 53 53 0 0 0 0 0 41 41 41 41 41 0 41 0 0 0 0 0 55 55 55 55 55 55 0 0 0 0 0
0 0 66 0 0 53 53 53 0 53 53 53 0 0 0 0 0 41 41 41 41 41 41 41 0 0 0 55 55 55 55 55 55 55 55 0 0 0 0 0 0 66
0 67 0 53 53 53 53 53 0 0 0 0 0 0 41 41 0 41 41 41 41 0 68

propertyFile for 4-connectness for data3

Using 4 Connected Component Algorithm
30 40 0 68
Number of Connected Components: 68
1
55
0 0
11 9

2
4
0 22
2 23

3
5
0 27
2 28

4
9

0 36
5 38
_____
5
5
2 30
3 33
_____
6
1
3 24
3 24
_____
_ 7
1
3 29
3 29
_____
_ 8
1
3 34
3 34
_____
_ 9
37
4 16
11 24
_____
_ 10
3
4 25
6 25
_____
_ 11
47
4 29
13 38
_____
_ 12
1
4 33
4 33
_____
_ 13
1
5 1
5 1
_____
_ 14
1
7 1
7 1
_____
_ 15
4
8 25
10 26
_____
_ 16
3
9 8
10 9
_____
_ 17
1

9 15
9 15

_____

_ 18
4
9 22
11 23

_____

_ 19
7
10 9
14 11

_____

_ 20
4
10 13
12 14

_____

_ 21
4
10 15
12 16

_____

_ 22
1
11 2
11 2

_____

_ 23
1
11 21
11 21

_____

_ 24
5
12 18
13 20

_____

_ 25
1
13 23
13 23

_____

_ 26
37
14 24
21 31

_____

_ 27
1
15 7
15 7

_____

_ 28
1
15 11
15 11

_____

_ 29
4
15 37
16 38

_____

_ 30
1

16 6
16 6

_____
_ 31
35
16 10
22 16

_____
_ 32
1
17 5
17 5

_____
_ 33
1
18 4
18 4

_____
_ 34
1
19 3
19 3

_____
_ 35
1
19 6
19 6

_____
_ 36
1
19 22
19 22

_____
_ 37
4
20 0
21 2

_____
_ 38
1
20 5
20 5

_____
_ 39
3
20 7
21 8

_____
_ 40
1
20 23
20 23

_____
_ 41
46
20 30
29 38

_____
_ 42
5
20 37
21 39

_____
_ 43
5

21 3
23 5
_____
_ 44
3
21 18
22 19
_____
_ 45
1
21 24
21 24
_____
46
1
22 7
22 7
_____
_ 47
1
22 9
22 9
_____
_ 48
1
22 23
22 23
_____
_ 49
2
22 25
23 25
_____
_ 50
1
23 1
23 1
_____
_ 51
1
23 10
23 10
_____
_ 52
4
23 16
25 17
_____
_ 53
36
23 18
29 26
_____
_ 54
2
23 28
23 29
_____
_ 55
20
24 2
29 9
_____
_ 56
1

24 8
24 8
_____
_ 57
1
24 11
24 11
_____
_ 58
1
24 19
24 19
_____
_ 59
1
24 27
24 27
_____
_ 60
5
25 1
27 3
_____
_ 61
2
25 3
25 4
_____
_ 62
1
25 31
25 31
_____
_ 63
1
26 15
26 15
_____
_ 64
1
27 9
27 9
_____
_ 65
1
27 17
27 17
_____
_ 66
2
28 16
29 16
_____
_ 67
1
29 18
29 18
_____
_ 68
1
29 39
29 39
_____

RFprettyPrintFile for 8-connectness for data3

Using 8 Connected Component Algorithm

Pass 1
1 2 3 4
5 1 2 2 3 3 4
5 5 5 1 1 1 2 3 3 6 6 6 6 4 4 5 5 1 1 1 1 1 2 3 6 6 4 4 1 1 1 1 1 1 7 7 7 2 6 6
4 4
8 1 1 1 1 1 1 7 7 7 7 7 2 6 6 4
1 1 1 1 1 1 1 7 7 7 7 7 7 7 2 6 6 6 4
9 1 1 1 1 1 7 7 7 7 7 7 2 6 6 6 6 4 4 4 9 1 1 1 7 7 7 7 7 7 7 2 6 6 4 4 4 4 4 9 9 1
1 1 1 1 7 7 7 7 7 7 2 2 6 4 4 4 4 4 1 1 1 1 1 1 7 7 7 7 7 7 7 2 4 4 4 4 4 4 4 1 1 1 1
1 7 7 7 7 7 7 4 4 4 4 4 4 1 7 7 7 7 7 10 4 4 4 4 4
1 1 7 7 7 11 4
1 12 12 12
13 1 12 12 12 12 12 12 14 14 13 1 1 1 12 12 12 12 12 12 14 14 13 1 1 1 1 1 1 12 12
12 12 12 12 12 12
13 1 1 1 1 1 1 1 1 12 12 12 12 12 12 12
13 15 1 1 1 1 1 16 12 12 12 12
17 17 13 15 15 15 1 1 1 1 1 1 1 1 16 12 12 12 12 12 12 12 18 18 13 15 15 1 1 1 1 1 19 16 12 12 12
12 18 18 18 13 13 15 15 1 1 1 19 19 16 16 12 12 12 20 13 13 15 19 16 16 21 21 12 12 22 13 23
15 19 19 24 16 16 16 21 12 12 12 12 12 12 25 26 26 13 13 19 16 16 16 16 21 12 12 12 12 25 25 13
19 27 16 16 16 16 16 16 16 16 12 12 12 12 12 25 25 13 13 13 28 27 16 16 16 16 16 16 16 12 12
12 12 12 12 13 13 13 13 13 27 16 16 16 16 16 16 12 12 12 12 12 12 12 12 29 13 13 13 13 13 13
13 27 16 16 16 16 16 16 16 12 12 12 12 12 12 12 Pass 1
New Label: 29
Equivalence Table:
1 1
2 2
3 3
4 4
5 1
6 4
7 2
8 8
9 1
10 4
11 11
12 12
13 13
14 14
15 13
16 16
17 13
18 18
19 19
20 20
21 16
22 12
23 13
24 16
25 13
26 25
27 16
28 13
29 13
Pass 2
1 2 3 4
1 1 2 2 3 3 4
1 1 1 1 1 1 2 3 3 3 4 4 4 4 4 1 1 1 1 1 1 1 2 3 4 4 4 4 1 1 1 1 1 1 1 2 2 2 2 4 4
4 4
8 1 1 1 1 1 1 2 2 2 2 2 2 4 4 4
1 1 1 1 1 1 1 2 2 2 2 2 2 2 2 2 4 4 4 4
1 1 1 1 1 1 2 2 2 2 2 2 2 2 4 4 4 4 4 4 4 1 1 1 1 2 2 2 2 2 2 2 2 2 4 4 4 4 4 4 4 1 1 1
1 1 1 1 2 2 2 2 2 2 2 2 2 4 4 4 4 4 4 1 1 1 1 1 1 2 2 2 2 2 2 2 2 2 4 4 4 4 4 4 4 1 1 1 1
1 2 2 2 2 2 2 4 4 4 4 4 4 1 2 2 2 2 2 4 4 4 4 4 4
1 1 2 2 2 11 4
1 12 12 12
13 1 12 12 12 12 12 12 14 14 13 1 1 1 12 12 12 12 12 12 14 14 13 1 1 1 1 1 1 12 12
12 12 12 12 12 12
13 1 1 1 1 1 1 1 12 12 12 12 12 12 12
13 13 1 1 1 1 1 16 12 12 12 12
13 13 13 13 13 13 1 1 1 1 1 1 1 1 16 12 12 12 12 12 12 12 18 18 13 13 13 1 1 1 1 1 19 16 12 12 12
12 18 18 18 13 13 13 13 1 1 1 19 19 16 16 12 12 12 20 13 13 13 19 16 16 16 16 12 12 12 13 13
13 19 19 16 16 16 16 16 12 12 12 12 12 13 13 25 13 13 19 16 16 16 16 16 12 12 12 12 13 13 13
19 16 16 16 16 16 16 16 16 16 12 12 12 12 12 13 13 13 13 13 13 16 16 16 16 16 16 16 16 12 12

12 12 12 12 13 13 13 13 13 16 16 16 16 16 16 16 12 12 12 12 12 12 12 13 13 13 13 13 13 13
13 16 16 16 16 16 16 16 12 12 12 12 12 12 12 Pass 2
New Label: 29
Equivalence Table:
1 1
2 2
3 3
4 4
5 1
6 3
7 2
8 8
9 1
10 4
11 11
12 12
13 13
14 14
15 13
16 16
17 13
18 18
19 19
20 20
21 16
22 12
23 13
24 16
25 13
26 13
27 16
28 13
29 13

After Equivalence Table Management
New Label: 29
Equivalence Table:
1 1
2 2
3 3
4 4
5 1
6 3
7 2
8 5
9 1
10 4
11 6
12 7
13 8
14 9
15 8
16 10
17 8
18 11
19 12
20 13
21 10
22 7
23 8
24 10
25 8
26 8
27 10
28 8
29 8

Pass 3
1 2 3 4 1 1 2 2 3 3 4 1 1 1 1 1 1 2 3 3 3 4 4 4 4 4 1 1 1 1 1 1 1 2 3 4 4 4 4 1 1
1 1 1 1 2 2 2 2 4 4 4 4 5 1 1 1 1 1 1 2 2 2 2 2 2 4 4 4 1 1 1 1 1 1 1 2 2 2 2 2 2 2 2
4 4 4 4 1 1 1 1 1 1 2 2 2 2 2 2 2 4 4 4 4 4 4 4 1 1 1 1 2 2 2 2 2 2 2 2 4 4 4 4 4 4 4
1 1 1 1 1 1 1 2 2 2 2 2 2 2 2 2 4 4 4 4 4 4 1 1 1 1 1 1 1 2 2 2 2 2 2 2 2 2 4 4 4 4 4 4 4 1

1 1 1 1 2 2 2 2 2 2 4 4 4 4 4 4 1 2 2 2 2 2 4 4 4 4 4 4 1 1 2 2 2 6 4
1 7 7 7
8 1 7 7 7 7 7 9 9 8 1 1 1 7 7 7 7 7 7 9 9 8 1 1 1 1 1 7 7 7 7 7 7 7
8 1 1 1 1 1 1 1 7 7 7 7 7 7 7
8 8 1 1 1 1 1 1 0 7 7 7 7
8 8 8 8 8 8 1 1 1 1 1 1 1 1 0 7 7 7 7 7 7 7 7 1 1 1 1 8 8 8 1 1 1 1 1 1 2 1 0 7 7 7 7 1 1 1 1 1 1 8
8 8 8 1 1 1 1 2 1 2 1 0 1 0 7 7 7 1 3 8 8 8 1 2 1 0 1 0 1 0 1 0 7 7 7 8 8 8 1 2 1 2 1 0 1 0 1 0 1 0
7 7 7 7 7 8 8 8 8 8 1 2 1 0 1 0 1 0 1 0 1 0 7 7 7 7 8 8 8 1 2 1 0 1 0 1 0 1 0 1 0 1 0 1 0 7 7
7 7 7 8 8 8 8 8 8 1 0 1 0 1 0 1 0 1 0 1 0 1 0 7 7 7 7 7 7 7 8 8 8 8 8 1 0 1 0 1 0 1 0 1 0 1 0 7
7 7 7 7 7 7 7 8 8 8 8 8 8 8 8 1 0 1 0 1 0 1 0 1 0 1 0 7 7 7 7 7 7 7 Pass 3
New Label: 29
Equivalence Table:
1 1
2 2
3 3
4 4
5 1
6 3
7 2
8 5
9 1
10 4
11 6
12 7
13 8
14 9
15 8
16 10
17 8
18 11
19 12
20 13
21 10
22 7
23 8
24 10
25 8
26 8
27 10
28 8
29 8

True number of Connected Components: 13
Result of Draw Box
1 1 1 1 1 1 1 1 1 1 1 1 1 2 2 2 2 2 2 2 2 2 2 2 2 2 2 3 3 4 4 4 4 4 4 4 4 4 1 1 1 2 1 2 2 2 3 3 4
3 4
1 1 1 1 1 1 2 1 2 2 3 3 4 3 4 4 4 4 4 1 1 1 1 1 1 1 1 2 1 2 2 2 3 3 4 3 4 4 4 4 1 1 1 1
1 1 1 2 1 2 2 2 2 2 4 4 4 4 4 4
1 5 1 1 1 1 1 1 2 1 2 2 2 2 2 2 2 4 4 4 4 4 1 1 1 1 1 1 1 1 2 1 2 2 2 2 2 2 2 2 4 4 4 4 4
4 1 1 1 1 1 1 1 2 1 2 2 2 2 2 2 2 2 4 4 4 4 4 4 4 4 4 1 1 1 1 1 2 1 2 2 2 2 2 2 2 2 2 4 4 4 4 4
4 4 4 4 4 1 1 1 1 1 1 1 1 2 2 1 2 2 2 2 2 2 2 2 4 4 4 4 4 4 4 4 4 1 1 1 1 1 1 1 2 2 1 2 1 2 2 2
2 2 4 4 4 4 4 4 4 1 1 1 1 1 1 2 2 1 2 1 2 2 2 2 2 4 4 4 4 4 4 4 4 4 1 1 2 2 1 2 1 2 2 2 2 4 4 4 4 4 4
4
1 1 1 2 2 2 2 2 2 2 2 2 2 6 2 2 2 4 4 4 4 4 4 4 4 4 4 1 1 1 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 8 8 8 8 8 8 8 8
8 8 8 1 7 7 7 7 7 7 9 9 7 8 8 8 1 1 1 1 7 7 7 7 7 7 9 9 7 8 8 8 1 1 1 1 1 7 7 7 7 7 7 7 7 7 8 8 1 8 1 1 1 1
7 7 7 7 7 7 7 7 8 8 8 1 8 1 1 1 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 7 8 8 8 8 8 8 1 8 1 1 1 1 1 0
1 0 7 7 7 7 1 0 7 7 7 7 1 1 1 1 1 1 8 8 8 8 8 1 1 1 1 2 1 2 1 2 1 2 1 2 7 7 1 0 7 7 7 1 1 1 1 1 8 1 1 1 1 1 1 1 1 1 1
8 1 1 1 1 2 1 0 1 2 1 2 1 0 7 1 0 1 0 7 7 7 7 8 1 3 8 8 8 8 1 2 1 0 1 2 1 2 1 0 7 1 0 1 0 1 0 7 7 7 7 8 8 8 8 1 2 1 0 1 2
1 2 1 0 1 0 1 0 7 1 0 1 0 7 7 7 7 7 7 8 8 8 8 8 8 8 1 2 1 0 1 2 1 0 1 0 1 0 7 1 0 1 0 7 7 7 7 7 8 8 8 8 8 1 2 1 2 1 2 1 2
1 2 1 0 1 0 1 0 7 1 0 1 0 1 0 7 7 7 7 7 7 8 8 8 8 8 8 8 8 1 0 1 0 1 0 1 0 1 0 1 0 7 1 0 1 0 7 7 7 7 7 7 7 8 8 8
8 8 8 8 1 0 1 0 1 0 1 0 7 1 0 1 0 7 7 7 7 7 7 7 7 7 8 8 8 8 8 8 8 8 8 8 8 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0
1 0 1 0 1 0 1 0 7 7 7 7 7 7 7 7 7

labelFile for 8-connectness for data3

Using 8 Connected Component Algorithm
30 40 0 13
0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 2 0 0 0 3 0 0 0 0 0 0 0 0 0 0 4 0 0 1 0 0 0 0 0 1 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 2 2 0 0 0 3 3 0 0 0 0 0 0 0 0 0 4 0 1 1 1 0 0 1 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 2
0 0 0 3 3 0 3 4 4 4 0 0 0 4 4 0 0 0 1 1 1 1 1 1 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 2 0 0 0 0 3 0 0 4 0 4 0
0 4 4 0 0 0 0 1 1 1 1 0 1 1 0 0 0 0 0 0 0 0 2 2 2 0 0 0 0 2 0 0 0 0 0 4 0 4 0 0 4 4 0 0 0 5 0 1 1 0 1 1
1 1 0 0 0 0 0 0 0 0 2 2 2 2 2 0 0 0 2 0 0 0 0 0 4 0 0 4 0 4 0 4 0 0 0 0 0 1 1 1 1 1 1 1 1 0 0 0 0 0 0 2 2 2 2
2 2 2 0 0 2 0 0 0 0 0 4 0 4 4 4 0 0 0 0 0 1 0 0 1 1 1 1 1 0 0 0 0 0 0 0 2 0 2 2 2 0 2 0 2 2 2 0 0 0 0 0 4 4
4 4 4 4 4 0 0 0 0 0 1 0 0 1 1 1 0 0 0 0 0 0 0 0 2 2 2 2 2 2 2 0 0 2 0 0 0 0 0 4 4 4 4 4 4 4 4 0 0 0 1 1

1 1 1 0 1 1 0 0 0 0 0 2 0 2 2 0 2 2 0 2 0 2 2 0 2 0 0 0 0 4 4 0 4 4 4 4 0 0 0 0 0 1 1 1 1 0 1 0 1 0 0 0 2 0
2 0 2 2 2 0 2 2 0 0 2 0 0 0 0 4 4 4 0 4 4 4 4 4 0 0 0 1 0 1 0 1 0 0 0 1 1 0 2 2 0 2 0 0 2 0 2 0 2 0 0 0 0
0 0 0 4 4 4 4 4 4 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 2 0 2 2 0 2 0 2 0 0 0 0 0 0 0 0 4 4 4 0 4 4 4 0 0 0 0
0 0 0 0 0 0 0 0 0 1 1 0 0 0 0 0 0 0 2 2 2 0 0 6 0 0 0 0 0 0 4 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 7 7 7 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 8 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0
0 7 7 7 7 7 7 0 0 0 0 0 0 0 0 9 9 0 0 0 0 0 0 0 8 0 0 0 0 0 0 1 1 1 0 0 0 0 0 0 0 0 0 7 7 7 0 7 7 7 0 0 0 0 0
0 9 9 0 0 0 0 0 0 8 0 0 0 0 0 0 1 1 1 1 1 0 0 0 0 0 0 0 0 0 7 7 7 7 7 7 7 7 0 0 0 0 0 0 0 0 0 0 0 0 8 0 0 0
0 0 1 1 1 1 1 1 1 1 0 0 0 0 0 0 0 0 7 7 7 7 7 7 7 7 0 0 0 0 0 0 0 0 0 0 0 0 8 0 0 8 0 0 8 0 0 0 1 0 1 1 1 0 1 0 0 0
0 0 10 0 0 7 7 0 7 7 0 0 0 0 0 0 0 0 0 0 0 0 8 8 8 8 0 0 8 0 8 8 8 0 1 1 1 1 1 1 1 1 0 0 0 0 0 0 0 1 0 0 0 7 7 7 0 7
7 7 7 0 0 0 11 11 0
0 0 8 0 8 0 0 0 8 0 0 1 1 1 1 1 0 0 12 0 0 0 0 0 10 0 0 7 0 0 0 7 0 7 7 0 0 11 11 11 0 0 0 8 8 0 0 8 0 8 0 8 0 0
1 1 1 0 0 0 12 12 0 0 0 10 0 10 0 0 0 0 0 7 0 7 7 0 0 0 0 0 0 13 0 0 8 8 0 0 0 0 8 0 0 0 0 0 0 12 0 0 0 0
10 0 0 10 0 0 10 10 0 7 0 0 7 0 7 0 0 0 0 0 0 0 0 0 0 8 0 8 0 0 8 0 0 0 0 12 12 0 10 0 10 10 10 0 0 0 10 0
0 0 0 7 7 7 7 7 0 0 0 0 8 0 8 8 8 0 8 8 0 0 0 0 0 0 0 0 0 12 0 0 0 10 10 0 10 10 0 10 0 0 0 0 7 0 7 7 7 0 0 0 0
0 8 8 0 0 0 8 0 0 0 0 0 0 0 0 0 12 0 0 10 10 10 10 10 10 10 10 10 0 0 0 0 0 0 7 7 7 7 7 0 0 0 0 0 8 8 0 8 0 8 8
0 8 0 0 0 0 0 0 0 0 10 0 10 10 10 10 10 10 10 0 0 0 0 0 0 7 7 7 7 7 0 7 0 0 0 0 0 0 8 8 8 8 8 0 0 0 0 0 0 0 10
0 0 10 10 10 0 0 10 10 10 0 0 0 0 0 7 7 7 7 7 7 7 7 0 0 0 8 8 8 8 8 8 8 8 0 0 0 0 0 0 0 10 0 10 0 10 10 10 10
10 0 0 0 0 0 0 7 7 0 7 7 7 7 0 7 propertyFile for 8-connectness for data3
Using 8 Connected Component Algorithm
30 40 0 13
Number of Connected Components: 13
1
103
0 0
22 16
_____
2
68
0 13
13 26
_____
3
7
0 27
3 30
_____
4
62
0 29
13 38
_____
5
1
5 1
5 1
_____
6
1
13 23

13 23

_____

7
85
14 24
29 39

_____

8
52
15 0
29 11

_____

9
4
15 37
16 38

_____

10
50
19 16
29 29

_____

11
5
20 37
21 39

_____

12
8
21 15
26 19

_____

13
1
23 1
23 1

_____