Article

An executable Java software for visualizing networks

WenJun Zhang
School of Life Sciences, Sun Yat-sen University, Guangzhou 510275, China
E-mail: zhwj@mail.sysu.edu.cn, wjzhang@iaees.org

Received 13 January 2023; *Accepted 26 March 2023*; *Published online 15 April 2023*; *Published 1 March 2024*

Abstract
An executable Java software, netGen, for visualizing networks was developed in present study. An interactive network can be visualized from the specified text or csv data file by using netGen. netGen is useful to visualize both undirected and directed networks. Both netGen and demonstration data files are given.

Keywords software; network; visualization; Java.

Network Biology
ISSN 2220-8879
URL: http://www.iaees.org/publications/journals/nb/online-version.asp
RSS: http://www.iaees.org/publications/journals/nb/rss.xml
E-mail: networkbiology@iaees.org
Editor-in-Chief: WenJun Zhang
Publisher: International Academy of Ecology and Environmental Sciences

1 Introduction

Network biology dedicates to the construction, analysis, simulation and control of biological networks (Zhang, 2012b, 2016b, 2018). Network construction is the basis of network biology. Biological networks are always constructed from random sampling (Zhang, 2011, 2012c; Zhang et al., 2014), experimental or reported data (Li and Zhang, 2013; Jiang and Zhang, 2015; Jiang et al., 2015; Qi et al., 2018; Zhang and Zhang, 2019; Xin and Zhang, 2020, 2021; Zhang, 2021b; Yang and Zhang, 2022), or theoretical simulation (Zhang, 2016a; Zhang and Li, 2016).

Also, network visualization is an important aspect (Narad et al., 2017). Various visualization software and tools, including Pajek, have been developed and used (Zhang, 2007; Zhang, 2012a, b; Li and Zhang, 2013; Zhang and Zhang, 2019; Xin and Zhang, 2020, 2021; Yang and Zhang, 2022). In a recent study, I presented an online web tool (http://www.iaees.org/publications/software/netJa/netGen.htm) for visualizing user-interface interactive networks was presented (Zhang, 2021a). In the generated network, the user can mouse-press any of the nodes to drag the network, examine network topology, evaluate node centrality, etc. It can be freely used and run on popular web browsers as Chrome, etc. It is an animated and powerful tool but, the layout of the generated network cannot be freely controlled by the user.

Zhang (2007, 2012a) has presented Java tools for visualizing interactive networks. In the advanced version of the tool (Zhang, 2012a), the data must be loaded with ODBC database and the tool is just a Java application, which is harder to be manipulated by the users. In addition, the appearance of the application is not ideal. For these reasons, in this study I re-developed the executable Java software which uses text or csv data files only which will be much convenient for use.

2 Software and Data

2.1 Software and runtime environment

The software, netGen (version 2.0), was developed based on j2sdk1.4.2, in which several Java classes, NetGraph, NetVertex, NetEdge, NetPanel, and GraphicsFrame (Zhang, 2007; Zhang, 2012a) were included and loaded by the class, netGenerator. The following are Java codes of the class netGenerator:

```
import java.awt.*;
import java.io.*;

public class netGenerator extends Frame
{

    public netGenerator()
    {
        Panel panel = new Panel();
        panel.setLayout(new FlowLayout());
        cbg = new CheckboxGroup();
        cb1 = new Checkbox(" ", cbg, true);
        cb2 = new Checkbox(" ", cbg, false);
        Label label = new Label("Network Types?");
        cb1.setLabel("Undirected Network");
        cb2.setLabel("Directed Network");
        buttonok = new Button("OK");
        panel.add(label);
        panel.add(cb1);
        panel.add(cb2);
        panel.add(buttonok);
        add(panel);
        resize(350, 150);
        setLocation(250, 180);
        pack();
        show();
    }

    public boolean action(Event event, Object obj)
    {
        if(event.target == buttonok)
        {
            if(cbg.getCurrent() == cb1)
                sele = 0;
            else
            if(cbg.getCurrent() == cb2)
                sele = 1;
            hide();
            Frame frame = new Frame();
```

```
                FileDialog filedialog = new FileDialog(frame, "Open data file", 0);
                frame.setLocation(250, 150);
                filedialog.show();
                String s = filedialog.getDirectory() + filedialog.getFile();
                try
                {
                    DataInputStream datainputstream = new DataInputStream(new FileInputStream(s));
                    int i;
                    for(i = 0; datainputstream.readLine() != null; i++);
                    datainputstream.close();
                    args1 = new String[i + 1];
                    args2 = new String[i + 1];
                    ai = new int[i + 1];
                    DataInputStream datainputstream1 = new DataInputStream(new FileInputStream(s));
                    String s1;
                    for(int j = 0; (s1 = datainputstream1.readLine()) != null; j++)
                    {
                        String as[] = s1.split(",");
                        args1[j + 1] = as[0];
                        args2[j + 1] = as[1];
                        ai[j + 1] = Integer.parseInt(as[2]);
                    }

                    datainputstream.close();
                }
                catch(Exception exception) { }
                (new GraphicsFrame(new NetGraph(args1, args2, ai, sele), "Zhang WJ. 2024. An executable Java software for
visualizing networks. Network Biology, 14(1): 1-12")).resize(900, 680);
                return true;
            } else
            {
                return false;
            }
        }

        public static void main(String args[])
            throws IOException
        {
            new netGenerator();
        }

        public String args1[];
        public String args2[];
        public int ai[];
        public int sele;
        public CheckboxGroup cbg;
        public Checkbox cb1;
        public Checkbox cb2;
        public Button buttonok;
}
```

The following are Java codes of the class netPanel:

```
import java.awt.*;
import java.awt.event.*;
import java.awt.geom.GeneralPath;

class NetPanel extends Panel
        implements MouseListener, MouseMotionListener
{

        public NetPanel(int i)
        {
                sele = i;
                vertice = new NetVertex[10000];
                edges = new NetEdge[100000];
                if(nvertice >= 50 && nvertice < 2000)
                        nodeDiam = 10;
                else
                if(nvertice > 2000)
                        nodeDiam = 8;
                else
                if(nvertice < 50)
                        nodeDiam = 18;
                posColor = Color.black;
                addMouseListener(this);
                addMouseMotionListener(this);
        }

        public int findVertex(String s)
        {
                for(int i = 0; i < nvertice; i++)
                        if(vertice[i].lab.equals(s))
                                return i;

                return addVertex(s);
        }

        public int addVertex(String s)
        {
```

```
        NetVertex netvertex = new NetVertex();
        netvertex.x = 100D + 350D * Math.random();
        netvertex.y = 100D + 350D * Math.random();
        netvertex.lab = s;
        vertice[nvertice] = netvertex;
        return nvertice++;
    }

    public void addEdge(String s, String s1, int i)
    {
        NetEdge netedge = new NetEdge();
        netedge.from = findVertex(s);
        netedge.to = findVertex(s1);
        netedge.type = i;
        edges[nedges++] = netedge;
    }

    public void paintVertex(Graphics g, NetVertex netvertex, FontMetrics fontmetrics)
    {
        int i = (int)netvertex.x;
        int j = (int)netvertex.y;
        int k = fontmetrics.stringWidth(netvertex.lab) + 10;
        int l = fontmetrics.getHeight() + 4;
        netvertex.w = k;
        netvertex.h = l;
        g.setColor(Color.blue);
        g.fillOval(i, j, nodeDiam, nodeDiam);
        g.drawOval(i, j, nodeDiam, nodeDiam);
        g.setColor(Color.red);
        g.drawString(netvertex.lab, i - (k - 10) / 2, (j - (l - 4) / 2) + fontmetrics.getAscent() + nodeDiam + 10);
    }

    public void update(Graphics g)
    {
        Dimension dimension = getSize();
        if(offscreen == null || dimension.width != offscreensize.width || dimension.height != offscreensize.height)
        {
            offscreen = createImage(dimension.width, dimension.height);
            offscreensize = dimension;
            offgraphics = offscreen.getGraphics();
            offgraphics.setFont(getFont());
        }
        offgraphics.setColor(getBackground());
        offgraphics.fillRect(0, 0, dimension.width, dimension.height);
        for(int i = 0; i < nedges; i++)
        {
            NetEdge netedge = edges[i];
            int k = (int)vertice[netedge.from].x;
            int l = (int)vertice[netedge.from].y;
            int i1 = (int)vertice[netedge.to].x;
            int j1 = (int)vertice[netedge.to].y;
            if(sele == 0)
            {
                offgraphics.setColor(posColor);
                offgraphics.drawLine(k + nodeDiam / 2, l + nodeDiam / 2, i1 + nodeDiam / 2, j1 + nodeDiam / 2);
            }
            if(sele != 1)
                continue;
            offgraphics.setColor(posColor);
            if(netedge.type == 1)
            {
                if(i1 >= k)
                    paintArrow(offgraphics, k + nodeDiam / 2, l + nodeDiam / 2, i1, j1 + nodeDiam / 2);
                if(i1 < k)
                    paintArrow(offgraphics, k + nodeDiam / 2, l + nodeDiam / 2, i1 + nodeDiam, j1 + nodeDiam / 2);
            }
            if(netedge.type == -1)
                paintArrow(offgraphics, i1, j1 + nodeDiam / 2, k + nodeDiam / 2, l + nodeDiam / 2);
        }
        FontMetrics fontmetrics = offgraphics.getFontMetrics();
        for(int j = 0; j < nvertice; j++)
            paintVertex(offgraphics, vertice[j], fontmetrics);

        g.drawImage(offscreen, 0, 0, null);
    }

    public void paintArrow(Graphics g, int i, int j, int k, int l)
    {
        double d = nodeDiam;
        double d1 = nodeDiam / 3;
        int i1 = 0;
        int j1 = 0;
        int k1 = 0;
        int l1 = 0;
        double d2 = Math.atan(d1 / d);
        double d3 = Math.sqrt(d1 * d1 + d * d);
        double ad[] = rotateVec(k - i, l - j, d2, true, d3);
        double ad1[] = rotateVec(k - i, l - j, -d2, true, d3);
        double d4 = (double)k - ad[0];
        double d5 = (double)l - ad[1];
        double d6 = (double)k - ad1[0];
        double d7 = (double)l - ad1[1];
        Double double1 = new Double(d4);
        i1 = double1.intValue();
        Double double2 = new Double(d5);
        j1 = double2.intValue();
        Double double3 = new Double(d6);
        k1 = double3.intValue();
        Double double4 = new Double(d7);
        l1 = double4.intValue();
        g.setColor(Color.blue);
        g.drawLine(i, j, k, l);
        Graphics2D graphics2d = (Graphics2D)g;
        GeneralPath generalpath = new GeneralPath();
        generalpath.moveTo(k, l);
        generalpath.lineTo(i1, j1);
        generalpath.lineTo(k1, l1);
        generalpath.closePath();
        graphics2d.fill(generalpath);
    }

    public double[] rotateVec(int i, int j, double d, boolean flag, double d1)
    {
        double ad[] = new double[2];
        double d2 = (double)i * Math.cos(d) - (double)j * Math.sin(d);
        double d3 = (double)i * Math.sin(d) + (double)j * Math.cos(d);
        if(flag)
        {
            double d4 = Math.sqrt(d2 * d2 + d3 * d3);
            d2 = (d2 / d4) * d1;
            d3 = (d3 / d4) * d1;
            ad[0] = d2;
            ad[1] = d3;
        }
        return ad;
    }

    public void paint(Graphics g)
    {
        repaint();
    }

    public void mousePressed(MouseEvent mouseevent)
    {
        double d = 1.7976931348623157E+308D;
```

```
            int i = mouseevent.getX();
            int j = mouseevent.getY();
            for(int k = 0; k < nvertice; k++)
            {
                NetVertex netvertex = vertice[k];
                double d1 = (netvertex.x - (double)i) * (netvertex.x - (double)i) + (netvertex.y - (double)j) * (netvertex.y -
(double)j);
                if(d1 < d)
                {
                    pick = netvertex;
                    d = d1;
                }
            }

            pickfixed = pick.fixed;
            pick.fixed = true;
            pick.x = i;
            pick.y = j;
            repaint();
            mouseevent.consume();
        }

        public void mouseReleased(MouseEvent mouseevent)
        {
            pick.x = mouseevent.getX();
            pick.y = mouseevent.getY();
            pick.fixed = pickfixed;
            pick = null;
            repaint();
            mouseevent.consume();
        }

        public void mouseDragged(MouseEvent mouseevent)
        {
            pick.x = mouseevent.getX();
            pick.y = mouseevent.getY();
            repaint();
            mouseevent.consume();
        }

        public void mouseClicked(MouseEvent mouseevent)
        {
        }

        public void mouseEntered(MouseEvent mouseevent)
        {
        }

        public void mouseExited(MouseEvent mouseevent)
        {
        }

        public void mouseMoved(MouseEvent mouseevent)
        {
        }

        public int sele;
        public int nodeDiam;
        int nvertice;
        int nedges;
        int w;
        int h;
        NetVertex vertice[];
        NetEdge edges[];
        NetVertex pick;
        boolean pickfixed;
        Image offscreen;
        Dimension offscreensize;
        Graphics offgraphics;
        Color posColor;
        Color negColor;
    }
```

Pack all classes into a JAR package and compile the package into an executable Java software, netGen.exe.

Before using the software, the Java Runtime Environment (JRE) should be installed. First, download the Java Runtime Environment software from the following website: https://www.java.com/en/. After downloading, click Install Java Runtime Environment. Alternatively, search for "Java Runtime Environment" on the Internet to find the appropriate platform version to download and install. Then, double-click netGen.exe to run the software (Fig. 1).

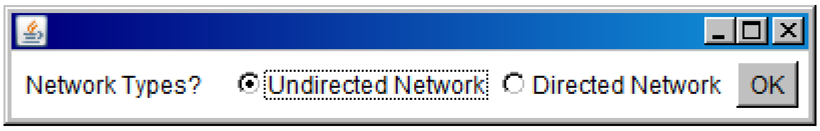

Network Types? [✓ Undirected Network] Directed Network

Fig. 1 An interface of netGen.

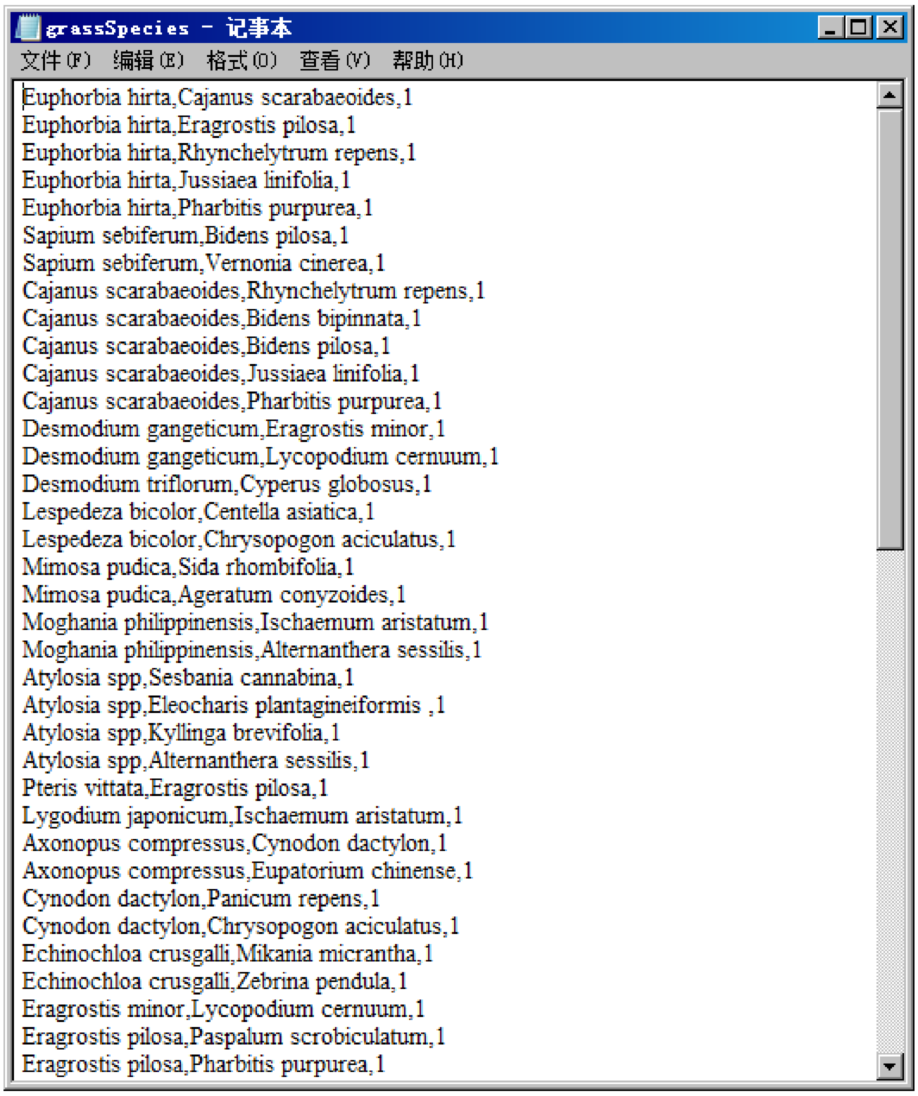
2.2 Data

The data for visualizing a network is stored in a text or csv file. Suppose there are totally *m* links in the network, then there are *m* rows in the data file. There are three items in each row: the first item (node) and the second item (node) represent a link between two nodes and the third item is an integer value *x*. For an undirected network, *x*=1 for each row. For a directed network, *x*=1 if it is a forward link (from the first item (node) to the second item (node)); *x*=-1 is for backward link (from the second item (node) to the first item (node)). However, if users choose to generate an undirected network (Fig. 1), *x*=1 and *x*=-1 are equivalent. The three items are separated by the comma "," (so by default, it means that any comma is not allowed in the first and second items), without any spaces.

The software and demo data files are included in the package: http://www.iaees.org/publications/journals/nb/articles/2024-14(1)/e-suppl/Zhang-Supplementary-Material.rar

3 Demonstration

The network data for grass species community were obtained from Zhang (2012c), which is stored in a text file (grassSpecies.txt), as shown in Fig. 2.



```
Euphorbia hirta,Cajanus scarabaeoides,1
Euphorbia hirta,Eragrostis pilosa,1
Euphorbia hirta,Rhynchelytrum repens,1
Euphorbia hirta,Jussiaea linifolia,1
Euphorbia hirta,Pharbitis purpurea,1
Sapium sebiferum,Bidens pilosa,1
Sapium sebiferum,Vernonia cinerea,1
Cajanus scarabaeoides,Rhynchelytrum repens,1
Cajanus scarabaeoides,Bidens bipinnata,1
Cajanus scarabaeoides,Bidens pilosa,1
Cajanus scarabaeoides,Jussiaea linifolia,1
Cajanus scarabaeoides,Pharbitis purpurea,1
Desmodium gangeticum,Eragrostis minor,1
Desmodium gangeticum,Lycopodium cernuum,1
Desmodium triflorum,Cyperus globosus,1
Lespedeza bicolor,Centella asiatica,1
Lespedeza bicolor,Chrysopogon aciculatus,1
Mimosa pudica,Sida rhombifolia,1
Mimosa pudica,Ageratum conyzoides,1
Moghania philippinensis,Ischaemum aristatum,1
Moghania philippinensis,Alternanthera sessilis,1
Atylosia spp,Sesbania cannabina,1
Atylosia spp,Eleocharis plantagineiformis ,1
Atylosia spp,Kyllinga brevifolia,1
Atylosia spp,Alternanthera sessilis,1
Pteris vittata,Eragrostis pilosa,1
Lygodium japonicum,Ischaemum aristatum,1
Axonopus compressus,Cynodon dactylon,1
Axonopus compressus,Eupatorium chinense,1
Cynodon dactylon,Panicum repens,1
Cynodon dactylon,Chrysopogon aciculatus,1
Echinochloa crusgalli,Mikania micrantha,1
Echinochloa crusgalli,Zebrina pendula,1
Eragrostis minor,Lycopodium cernuum,1
Eragrostis pilosa,Paspalum scrobiculatum,1
Eragrostis pilosa,Pharbitis purpurea,1
```

Fig. 2 The network data for grass species community.

Run the software netGen.exe and load the data file above, an interactive network can be visualized. We can drag the nodes (species) in the network to exhibit a better layout of the network (Fig. 3 and 4).

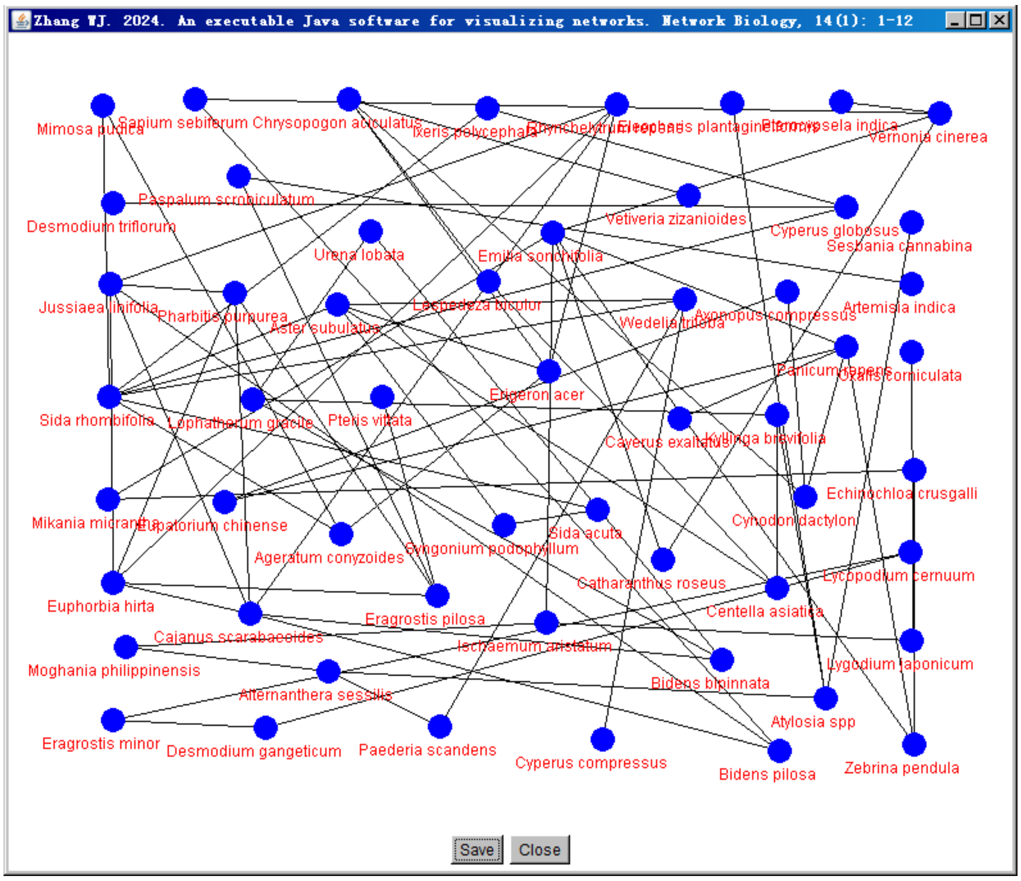


Fig. 3 The undirected network for grass species community, generated by the software, netGen.

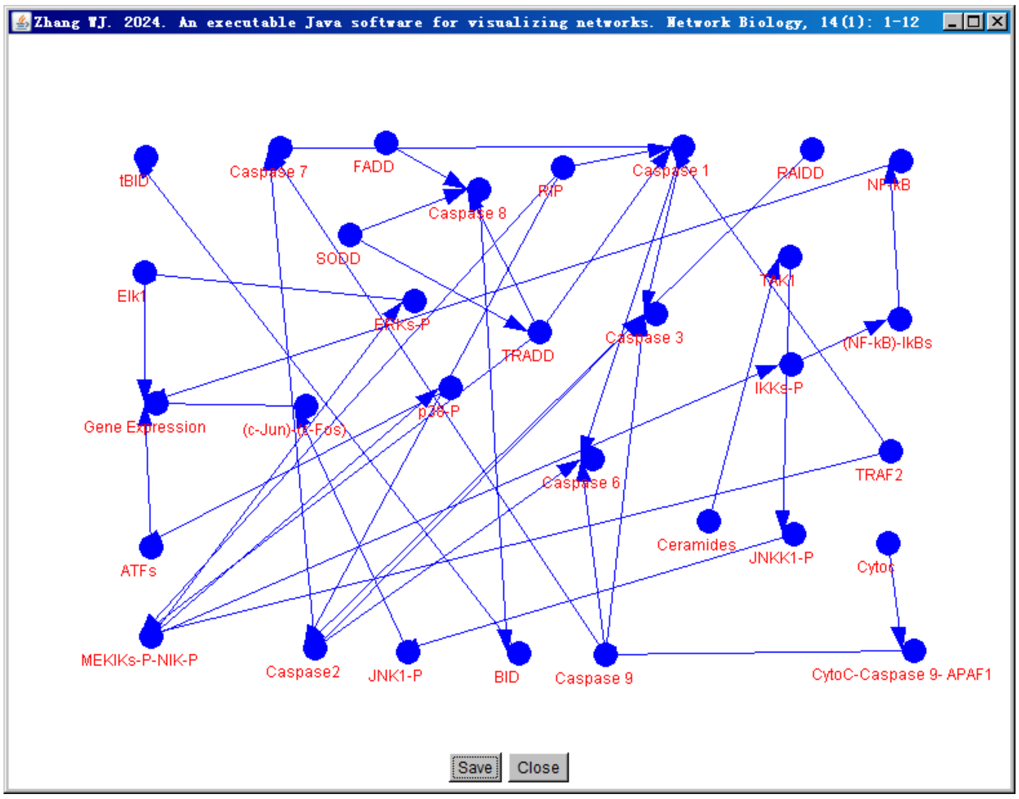


Fig. 4 The directed network for TNF signaling pathway, generated by the software, netGen.

References

Jiang LQ, Zhang WJ. 2015. Determination of keystone species in CSM food web: A topological analysis of network structure. Network Biology, 5(1): 13-33

Jiang LQ, Zhang WJ, Li X. 2015. Some topological properties of arthropod food webs in paddy fields of South China. Network Biology, 5(3): 95-112

Li JR, Zhang WJ. 2013. Identification of crucial metabolites/reactions in tumor signaling networks. Network Biology, 3(4): 121-132

Narad P, Upadhyaya KC, Som A. 2017. Reconstruction, visualization and explorative analysis of human pluripotency network. Network Biology, 7(3): 57-75

Qi YH, Liu GH, Zhang WJ. 2018. Analysis of word occurrence frequency and word association in English text file: A big data analytics method. Network Biology, 8(3): 126-136

Xin SH, Zhang WJ. 2020. Construction and analysis of the protein-protein interaction network for the olfactory system of the silkworm *Bombyx mori*. Archives of Insect Biochemistry and Physiology, 105(3): e21737

Xin SH, Zhang WJ. 2021. Construction and analysis of the protein-protein interaction network for the detoxification enzymes of the silkworm, *Bombyx mori*. Archives of Insect Biochemistry and Physiology, 108(4): e21850

Yang S, Zhang WJ. 2022. Systematic analysis of olfactory protein-protein interactions network of fruitfly, *Drosophila melanogaster*. Archives of Insect Biochemistry and Physiology, 110(2): e21882

Zhang GL, Zhang WJ. 2019. Protein-protein interaction network analysis of insecticide resistance molecular mechanism in *Drosophila melanogaster*. Archives of Insect Biochemistry and Physiology, 100(1): e21523

Zhang WJ. 2007. Computer inference of network of ecological interactions from sampling data. Environmental Monitoring and Assessment, 124: 253–261

Zhang WJ. 2011. Constructing ecological interaction networks by correlation analysis: hints from community sampling. Network Biology, 1(2): 81-98

Zhang WJ. 2012a. A Java software for drawing graphs. Network Biology, 2(1): 38-44

Zhang WJ. 2012b. Computational Ecology: Graphs, Networks and Agent-based Modeling. World Scientific, Singapore

Zhang WJ. 2012c. How to construct the statistic network? An association network of herbaceous plants constructed from field sampling. Network Biology, 2(2): 57-68

Zhang WJ. 2016a. A random network based, node attraction facilitated network evolution method. Selforganizology, 3(1): 1-9

Zhang WJ. 2016b. Selforganizology: The Science of Self-Organization. World Scientific, Singapore

Zhang WJ. 2018. Fundamentals of Network Biology. World Scientific Europe, London, UK

Zhang WJ. 2021a. A web tool for generating user-interface interactive networks. Network Biology, 11(4): 247-262

Zhang WJ. 2021b. Construction and analysis of the word network based on the Random Reading Frame (RRF) method. Network Biology, 11(3): 154-193

Zhang WJ, Li X. 2016. Generate networks with power-law and exponential-law distributed degrees: with applications in link prediction of tumor pathways. Network Pharmacology, 1(1): 15-35

Zhang WJ, Wang R, Zhang DL, et al. 2014. Interspecific associations of weed species around rice fields in Pearl River Delta, China: A regional survey. Selforganizology, 1(3-4): 143-205