

Network analysis with Cytoscape

presented by Kai, Stefan & Stephan

in lecture on systems bioinformatics

02/08/2012

Outline

1 Introduction

2 Code

3 Results

4 Literature

Outline

1 Introduction

2 Code

3 Results

4 Literature

Introduction

- using the tool CellDesigner to build a glycolysis/gluconeogenesis network,
- import the network into the tool Cytoscape,
- analyse the imported network and
- implement missing centralities and indices by coding some plugins

Glycolysis/Gluconeogenesis

- catabolic linear pathway of glycolysis deals with the breakdown and extraction of energy from glucose
- the reverse anabolic process - gluconeogenesis - is equally important
- gluconeogenesis helps to keep blood glucose levels within critical limits
- these processes provide many points for regulation

Glycolysis

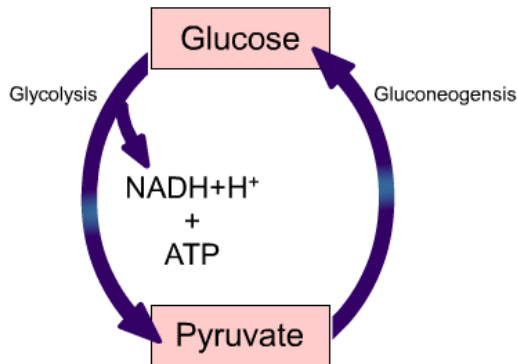
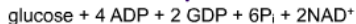


Fig. 1: Glycolysis vs. Gluconeogenesis

Glycolysis

The overall process of gluconeogenesis is energy intensive:



Compare this with glycolysis:

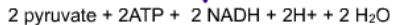


Fig. 2: Glycolysis vs. Gluconeogenesis

Glycolysis and Disease

- Genetic disease - mutations are generally rare due to importance of the metabolic pathway
- Other disease - disfunctioning glycolysis or glucose metabolism has been associated with some other diseases
- Cancer - typically tumor cells have glycolytic rates that are up to 200 times higher

- Ubiquitous gene overexpression appears to be restricted to glycolysis, in conclusion, Glycolysis is indeed special. [Gatenby and Gillies, 2004]
- this may also be of some interest for therapy, increased Glucose consumption can be observed with clinical tumour imaging
- Gene expression patterns in general can be modified by external factors such as drugs or components of nutrition
- one may envision substances that modify expression of glycolysis genes as complementary to conventional cancer therapies

- tried to model a glycolysis network
- tool very user unfriendly
- export into a format for cytoscape resulted in "strange" networks
- this task was skipped and the tool Cell Designer was not used

- provides many useful plugins
- plugin KGMLReader used to import KEGG Glycolysis network
- modified the network to make it user readable

Cytoscape import from KEGG

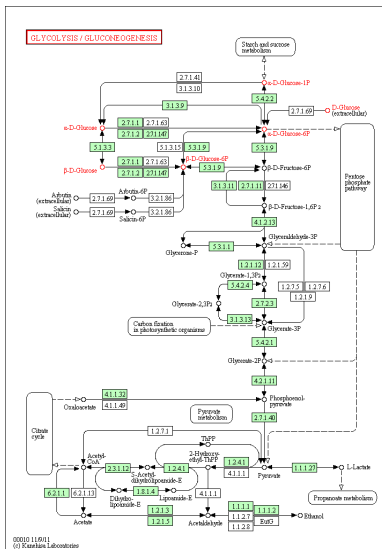


Fig. 3: Modelled Glycolysis Network

Cytoscape Network Model

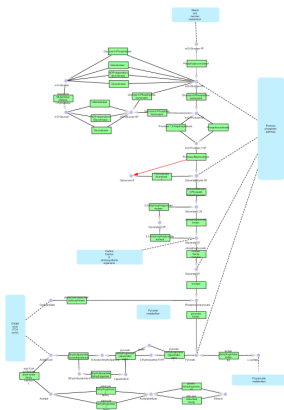


Fig. 4: example caption

Cytoscape Network Model

- Open Cytoscape Glycolysis Network

Outline

1 Introduction

2 Code

3 Results

4 Literature

Choice of programming language

Remark (supported languages)

Cytoscape supports various scripting language interfaces (e. g. Python, Ruby, Java, ...) for accessing its API through plugins.

But note: Depending on the version of the **plugin** and **cytoscape** there are incompatibilities, i. e. only a specific combination of **them** works.

Choice of programming language

Remark (supported languages)

Cytoscape supports various scripting language interfaces (e. g. Python, Ruby, Java, ...) for accessing its API through plugins.

But note: Depending on the version of the **plugin** and **cytoscape** there are incompatibilities, i. e. only a specific combination of **them** works.

Java

We chose Java solely because the availability of a template plugin to us as well as it was suggested by a colleague.

Plugin for calculation of Katz's Index

Definition (Katz's Index)

Let A denote the adjacency matrix of a **directed** or **undirected** graph with $\dim(A) = n \times n$ and let α denote a non-negative damping factor.

$$I_{Katz}(i) = \sum_{k=1, \dots, \infty} \sum_{j=1, \dots, n} \alpha(A^k)_{ji} \quad (1)$$

\Leftrightarrow

$$I_{Katz} = ((\mathbb{I}_m - \alpha A^T)^{-1} - \mathbb{I}_m) \mathbb{I}_v \quad (2)$$

For calculations we used Eq. 2 and considered two distinct cases:

- 1 α small \Rightarrow emulation of degree centrality
- 2 $\alpha \in]0, \frac{1}{|\lambda_{max}|}] \Rightarrow$ influence of pathes weighted

Plugin for calculation of Katz's Index - $\alpha \equiv 0.1$

Listing 1: Necessary import statements

```
1 package katz.plugin;  
2  
3 import java.util.List;  
4 import java.awt.event.ActionEvent;  
5 import javax.swing.JOptionPane;  
6 import java.text.DecimalFormat;  
7  
8 import cytoscape.plugin.CytoscapePlugin;  
9 import cytoscape.util.CytoscapeAction;  
10 import cytoscape.Cytoscape;  
11 import cytoscape.CyNetwork;  
12 import cytoscape.CyEdge;  
13 import cytoscape.view.CyNetworkView;  
14  
15 import Jama.Matrix;
```

Plugin for calculation of Katz's Index - $\alpha \equiv 0.1$

Listing 2: Basic class definition

```
16 public class KatzPlugin extends CytoscapePlugin {
17
18     public KatzPlugin() {
19         KatzPlugin.MolbiPluginAction action = new KatzPlugin.MolbiPluginAction();
20         action.setPreferredMenu("Plugins.Molbi");
21         Cytoscape.getDesktop().getCyMenus().addAction(action);
22     }
23
24     public String describe() {
25         return "Plugin to calculate the topological Katz Index for networks";
26     }
27
28     public class MolbiPluginAction extends CytoscapeAction {
29
30         public MolbiPluginAction() {
31             super("Katz's Index (alpha = 0.1)");
32         }
33
34         @Override
35         public void actionPerformed(ActionEvent ae) {
36             run();
37         }
38     }
39 }
```

Plugin for calculation of Katz's Index - $\alpha \equiv 0.1$

Listing 3: Run method (I)

```
38     private void run() {
39
40         CyNetwork network = Cytoscape.getCurrentNetwork();
41         CyNetworkView view = Cytoscape.getCurrentNetworkView();
42         int N = network.getNodeCount() + 1;
43
44         if (N == 1) {
45             JOptionPane.showMessageDialog(view.getComponent(), "No network/view
46                                     loaded.");
47             return;
48         }
49
50         double [][] A = new double[N][N];
51         for (double[] row : A) Arrays.fill(row, 0.0);
52
53         for (CyEdge edge : (List<CyEdge>) network.edgesList()) {
54             int i = Math.abs(edge.getSource().getRootGraphIndex());
55             int j = Math.abs(edge.getTarget().getRootGraphIndex());
56             A[i][j] = 1;
57         }
58     }
```

Plugin for calculation of Katz's Index - $\alpha \equiv 0.1$

Listing 4: Run method (II)

```
57         Matrix M = new Matrix(A);
58         Matrix I = Matrix.identity(N, N);
59         Matrix IVec = new Matrix(N, 1, 1.0);
60
61         double alpha = 0.1; // emulates degree centrality
62         double [][] values = ((I.minus(M.transpose()).times(alpha)).inverse()).
            minus(I).times(IVec).toArrayCopy();
63
64         StringBuilder sb = new StringBuilder();
65         DecimalFormat df = new DecimalFormat("#0.000");
66
67         for (int i = 1; i < N; i++) {
68             if (i % 10 == 0) sb.append("\n");
69             sb.append("C(").append(i).append("): ").append(df.format(values[i
70 ][0])).append(" ");
71         }
72         JOptionPane.showMessageDialog(view.getComponent(), "All done.\n" + sb);
73         view.redrawGraph(false, true);
74     }
75 }
76 }
```

Plugin for calculation of Katz's Index - $\alpha \equiv \frac{1}{|\lambda_{max}|}$

Listing 5: Run method (II)

```
57      Matrix M = new Matrix(A);
58      Matrix I = Matrix.identity(N, N);
59      Matrix IVec = new Matrix(N, 1, 1.0);
60
61      double[] eigs = M.eig().getRealEigenvalues();
62      double alpha;
63      Arrays.sort(eigs);
64
65      if (eigs[eigs.length - 1] == 0) alpha = 0.1; // emulates degree
           centrality
66      else alpha = 1.0 / Math.abs(eigs[eigs.length - 1]);
67
68      double[][] values = ((I.minus(M.transpose()).times(alpha)).inverse()).
           minus(I).times(IVec).toArrayCopy();
69      StringBuilder sb = new StringBuilder();
70      DecimalFormat df = new DecimalFormat("#0.000");
71
72      for (int i = 1; i < N; i++) {
73          if (i % 10 == 0) sb.append("\n");
74          sb.append("C(").append(i).append("): ").append(df.format(values[i
           ][0])).append(" ");
75      }
76
77      JOptionPane.showMessageDialog(view.getComponent(), "All done.\n" + sb);
78      view.redrawGraph(false, true);
79  }
80 }
81 }
```

Plugin for calculation of Randic's Index

Definition (Randic's Index)

Let $v(i)$ denote the #neighbors of vertex v with index i and N #nodes.

$$I_{Randic} = \sum_{i=0}^{N-1} \frac{1}{\sqrt{v(i)}} \quad (3)$$

Cave

Randic's Index works only for **undirected** and **connected** graphs.

Plugin for calculation of Randic's Index

Definition (Randic's Index)

Let $v(i)$ denote the #neighbors of vertex v with index i and N #nodes.

$$I_{Randic} = \sum_{i=0}^{N-1} \frac{1}{\sqrt{v(i)}} \quad (3)$$

Cave

Randic's Index works only for **undirected** and **connected** graphs.

Application

Currently there is no known application of Randic's Index for the analysis of biological networks.

Plugin for calculation of Randic's Index

Listing 6: Necessary import statements

```
1 package randic.plugin;
2
3 import java.util.List;
4 import java.awt.event.ActionEvent;
5 import javax.swing.JOptionPane;
6 import java.text.DecimalFormat;
7
8 import cytoscape.plugin.CytoscapePlugin;
9 import cytoscape.util.CytoscapeAction;
10 import cytoscape.Cytoscape;
11 import cytoscape.CyNetwork;
12 import cytoscape.CyEdge;
13 import cytoscape.view.CyNetworkView;
14
15 import Jama.Matrix;
```

Plugin for calculation of Randic's Index

Listing 7: Basic class definition

```
16 public class RandicPlugin extends CytoscapePlugin {
17
18     public RandicPlugin() {
19         RandicPlugin.MolbiPluginAction action = new RandicPlugin.MolbiPluginAction();
20         action.setPreferredMenu("Plugins.Molbi");
21         Cytoscape.getDesktop().getCyMenus().addAction(action);
22     }
23
24     public String describe() {
25         return "Plugin to calculate the Index of Randic for networks";
26     }
27
28     public class MolbiPluginAction extends CytoscapeAction {
29
30         public MolbiPluginAction() {
31             super("Randic's Index");
32         }
33
34         @Override
35         public void actionPerformed(ActionEvent ae) {
36             run();
37         }
38     }
39 }
```

Plugin for calculation of Randic's Index

Listing 8: Run method

```
38     private void run() {
39         CyNetwork network = Cytoscape.getCurrentNetwork();
40         CyNetworkView view = Cytoscape.getCurrentNetworkView();
41         int N = network.getNodeCount();
42
43         if (N == 0) {
44             JOptionPane.showMessageDialog(view.getComponent(), "No network/view
45                                     loaded.");
46             return;
47         }
48
49         Double IRandic = 0.0;
50         for (CyNode node : (List<CyNode>) network.nodesList())
51             // otherwise Randic's Index will be +infinity!
52             if (network.getDegree(node) != 0)
53                 IRandic += Math.pow(network.getDegree(node), -0.5);
54
55         JOptionPane.showMessageDialog(view.getComponent(), "All done.\nRandic
56                                     Index " + IRandic);
57         view.redrawGraph(false, true);
58     }
```

Outline

1 Introduction

2 Code

3 Results

4 Literature

- determine the relative importance of a vertex within a graph
- centrality concepts were first developed in social network analysis
 - e.g. how influential a person is within a social network

Definition (Four measures of centrality)

- 1 degree centrality
- 2 eccentricity centrality
- 3 closeness centrality
- 4 betweenness centrality

Definition (Degree Centrality)

$$C_{deg}(v) = \{e | e \in E \text{ and } v \in e\}$$

- counts the number of edges attached to a vertex
- a local centrality measure
 - only direct neighborhood considered
- directed graphs:
 - indegree (interpreted as "popularity")
 - outdegree (interpreted as "gregariousness")

Results - Degree Centrality

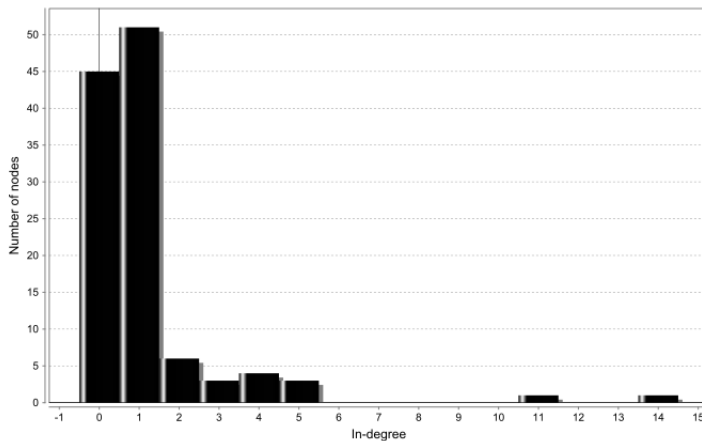


Fig. 5: Indegree Centrality

Results - Degree Centrality

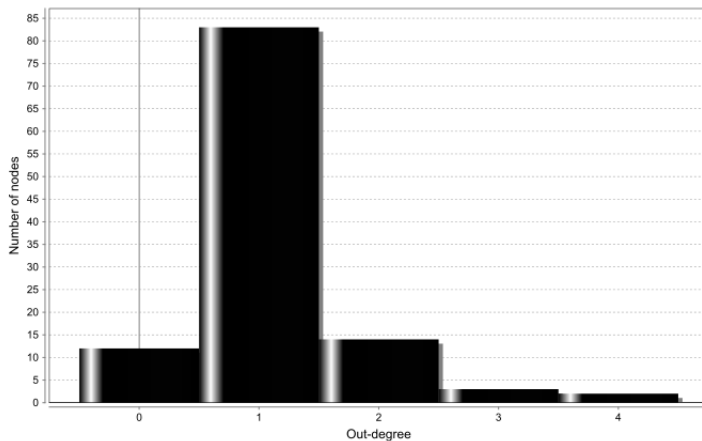


Fig. 6: Outdegree Centrality

Definition (Eccentricity Centrality)

$$C_{ecc}(s) = \frac{1}{\max\{d_{st} | t \in V\}}$$

- determine the maximum distance between every two vertices
- central vertices get low values
- centralities require high centrality values
 - reciprocal is used as centrality value
- only for CONNECTED networks

Results - Eccentricity Centrality

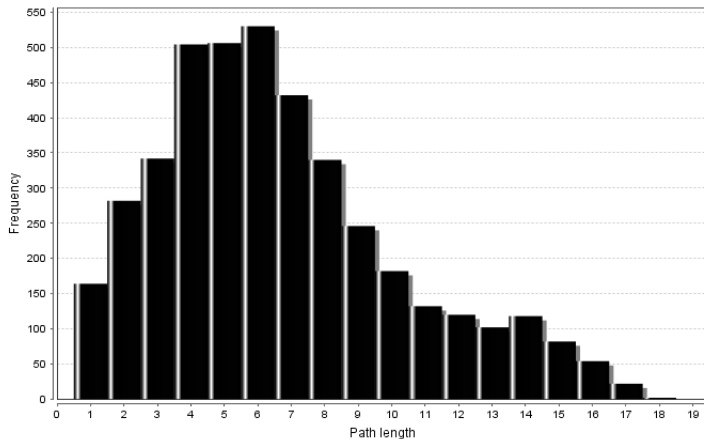


Fig. 7: Eccentricity Centrality

Definition (Closeness Centrality)

$$C_{clo}(s) = \frac{1}{\sum_{t \in V} d_{st}}$$

- C_{clo} of a vertex s is the sum of shortest path from s to all other vertices
- degree of interaction in the network
- the closer a point is to all the rest, the more effective and independent he can reach them

Results - Closeness Centrality

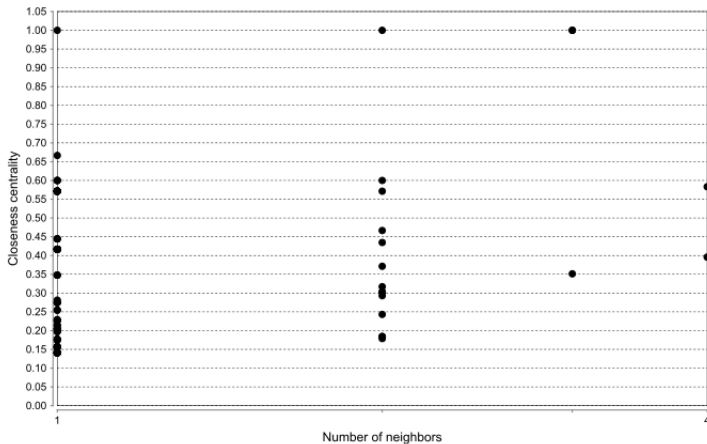


Fig. 8: Closeness Centrality

Definition (Betweenness Centrality)

$$C_{spb}(v) = \sum_{s \in V \text{ and } s \neq v} \sum_{t \in V \text{ and } s \neq v} \delta_{st}(v)$$

- center of attention: indirect relationships
→ Control of interaction
- v is the more powerful the more shortest paths between other vertices it can interrupt

Results - Betweenness Centrality

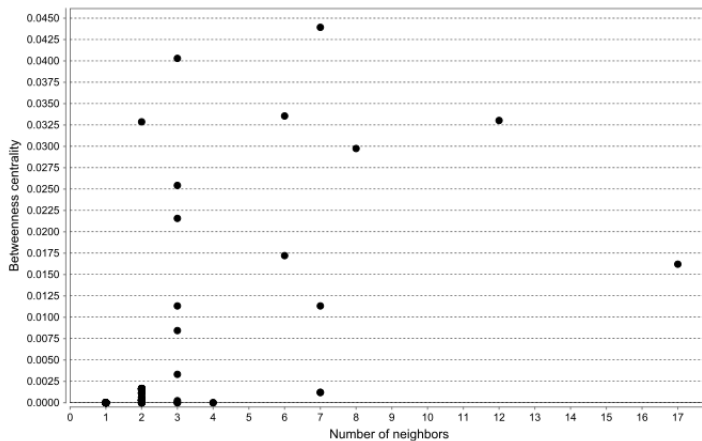


Fig. 9: Closeness Centrality

- the analysis tools are not very helpful
 - cannot see which vertex has which centrality
 - only give a statistical overview of the whole network
- **but** you can select a vertex in the network window and get an overview of all centralities from the selected vertex

Results - Betweenness Centrality

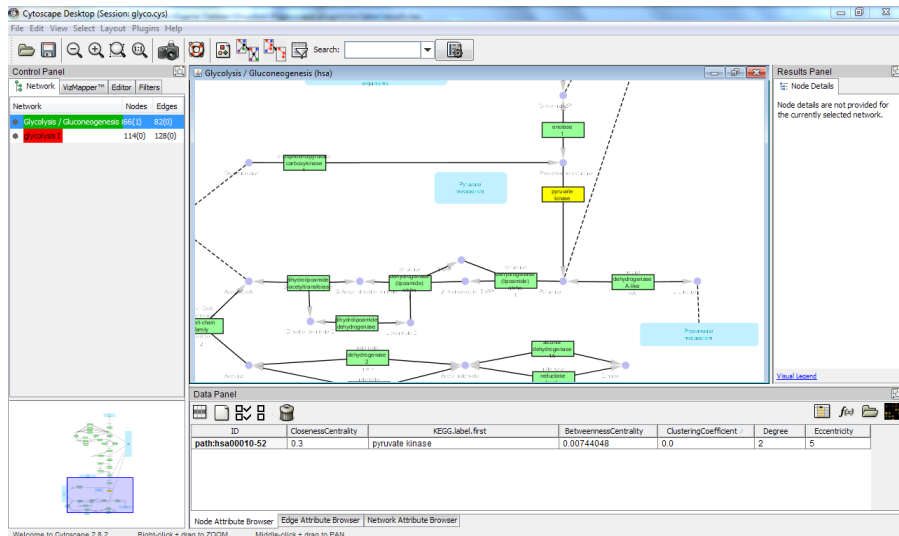


Fig. 10: Centrality Overview

Outline

1 Introduction

2 Code

3 Results

4 Literature



[Gatenby and Gillies]

Why do cancers have high aerobic glycolysis?

Nature Review Cance, Vol. 4, p. 891-899, November 2004.



[Altneberga and Greulich]

B. Altenberga and K.O. Greulich, Genes of glycolysis are ubiquitously overexpressed in 24 cancer classes.

Genomics Vol. 84, p. 1014-1020, September 2004.