A CUSTOMIZED FORCE-DIRECTED LAYOUT ALGORITHM FOR BIOLOGICAL GRAPHS WHOSE VERTICES HAVE ENZYME COMMISSION ATTRIBUTES

A THESIS SUBMITTED TO
THE GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES
OF
MIDDLE EAST TECHNICAL UNIVERSITY

BY

HASAN FEHMI DANACI

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR
THE DEGREE OF MASTER OF SCIENCE
IN
COMPUTER ENGINEERING

DECEMBER 2015

Approval of the thesis:

A CUSTOMIZED FORCE-DIRECTED LAYOUT ALGORITHM FOR BIOLOGICAL GRAPHS WHOSE VERTICES HAVE ENZYME COMMISSION ATTRIBUTES

submitted by **HASAN FEHMI DANACI** in partial fulfillment of the requirements for the degree of **Master of Science in Computer Engineering Department, Middle East Technical University** by,

Prof. Dr. Gülbin Dural Ünver _____
Dean, Graduate School of **Natural and Applied Sciences**

Prof. Dr. Adnan Yazıcı _____
Head of Department, **Computer Engineering**

Prof. Dr. Mehmet Volkan Atalay _____
Supervisor, **Computer Engineering Department, METU**

Examining Committee Members:

Prof. Dr. İsmail Hakkı Toroslu
Computer Engineering Department, METU _____

Prof. Dr. Mehmet Volkan Atalay
Computer Engineering Department, METU _____

Prof. Dr. Halit OĞUZTÜZÜN
Computer Engineering Department, METU _____

Assist. Prof. Dr. Aybar Can ACAR
Bioinformatics Department, METU _____

Assist. Prof. Dr. Mehmet TAN
Computer Engineering Department, TOBB-ETU _____

Date: _____

I hereby declare that all information in this document has been obtained and presented in accordance with academic rules and ethical conduct. I also declare that, as required by these rules and conduct, I have fully cited and referenced all material and results that are not original to this work.

Name, Last Name: HASAN FEHMI DANACI

Signature :

ABSTRACT

A CUSTOMIZED FORCE-DIRECTED LAYOUT ALGORITHM FOR BIOLOGICAL GRAPHS WHOSE VERTICES HAVE ENZYME COMMISSION ATTRIBUTES

Danacı, Hasan Fehmi

M.S., Department of Computer Engineering

Supervisor : Prof. Dr. Mehmet Volkan Atalay

December 2015, 48 pages

Force directed layout algorithm is popularly used to draw biological graphs. However, it employs only graph structure. When we would like to embed domain-specific knowledge, such as biological or chemical attributes related to the vertices, force directed layout algorithm should be modified. It is then important to draw more readable layouts for biologists without the dispose of aesthetically pleasing way that comes from force-directed algorithm's nature. This thesis aims to describe a modified and improved force-directed layout algorithm, EClerize, for biological graphs that represent pathways in which the vertices are identified with EC (Enzyme Commission) numbers. The vertices with the same EC class numbers are treated as members of the same cluster. Positions of vertices in clusters are affected by mainly two factors: biological similarity of each vertex in the same cluster and theoretical length between the vertices. EClerize is tested on a number of biological pathways and the improvement with respect to the original algorithm is presented.

Keywords: Information Visualization, Graph Visualization, Force-directed Graph Layout, Enzyme Commission Numbers

ÖZ

ENZİMLERİ TEMSİL EDEN DÜĞÜMLERE SAHİP ÇİZGELER İÇİN
ÖZELLEŞTİRİLMİŞ KUVVET YÖNELİMLİ YERLEŞİM ALGORİTMASI

Danacı, Hasan Fehmi
Yüksek Lisans, Bilgisayar Mühendisliği Bölümü
Tez Yöneticisi : Prof. Dr. Mehmet Volkan Atalay

Aralık 2015 , 48 sayfa

Kuvvet yönelimli yerleşim algoritmaları biyolojik çizgeleri çizmede popüler olarak kullanılır. Fakat sadece çizge yapılarını esas alır. Düğümlerle ilişkili olan biyolojik veya kimyasal özellikler gibi alan-özgül bilgileri çizge içerisine katmak istediğimizde kuvvet yönelimli yerleşim algoritmaları değiştirilmelidir. Bu noktada kullanıcılar için önemli olan kuvvet yönelimli yerleşim algoritmasının doğasından gelen estetiği bozmadan, daha okunabilir yerleşimler çizilmesidir. Bu tez, "Enzyme Commission" (EC) sayıları ile ifade edilen düğümlerden oluşan yolakları gösteren biyolojik çizgeler için değiştirilmiş ve geliştirilmiş bir kuvvet yönelimli yerleşim algoritması, EClerize'ı, tanımlamayı amaçlar. EC sınıf numaraları aynı olan düğümler aynı kümenin elemanı olarak muamele görürler. Aynı kümedeki düğümlerin pozisyonları iki ana faktörden etkilenir; aynı kümedeki düğümlerin biyolojik benzerlikleri ve düğümler arasında olması gereken teorik uzunluk. EClerize birçok biyolojik yolakların üzerinde test edilmiştir ve gelişmeler asıl algoritmayla karşılaştırılarak sunulmuştur.

Anahtar Kelimeler: Görselleme, Çizge Görselleme, Kuvvet Yönelimli Çizge Yerleşimi, Enzim Komisyonu Sayıları

To my dearest family

ACKNOWLEDGMENTS

I would like to express my gratitude to my supervisor Prof. Dr. Volkan Atalay for the useful comments, remarks and engagement through the learning process of this master thesis. None of this would have been possible without his help and guidance.

I would like to thank Assoc. Prof. Dr. Rengul Atalay for introducing me to the topic as well for the support on the way. I am also thankful to Assist. Prof. Dr. Aybar Can Acar for his valuable comments.

I am also grateful to Tugce Bulum for her patience. Very special thanks go to my friends Orhun Ozer and Seckin Kesici. Finally, I would like to thank my family for their endless love and support.

TABLE OF CONTENTS

LIST OF TABLES

TABLES

LIST OF FIGURES

FIGURES

CHAPTER 1

INTRODUCTION

Graphs are models representing relational data which consist of vertices (also called nodes) and edges. Vertices are used to represent objects and edges for relations between objects. Number of vertices in a graph may change from tens to large hundreds of thousands. Graphs which have small number of vertices can be drawn manually, however this is impractical for large graphs. Therefore, many automated layout algorithms, such as hierarchical layout, circular layout, force-directed layout, have been described in the literature [1]. Among these algorithms, the force-directed layout algorithms are the most popular ones since they are efficient and produce aesthetic results.

Force-directed layout algorithms treat vertices similar to charged particles and edges as springs. Particles repel each other while springs pull vertices together. This continues until the equilibrium is reached. Force-directed layout algorithms take into account only vertices and their relations.

Graphs are widely used in several areas from social networks to cartography to biology. Biological graphs consist of components such as genes, proteins and chemicals. Enzymes are proteins that have important roles in chemical reactions. An international committee (Nomenclature Committee of the International Union of Biochemistry and Molecular Biology) classifies enzymes based on the chemical reactions they catalyze. A numerical classification scheme for enzymes were defined and a class number, Enzyme Commission (EC) Number, have been assigned to each enzyme. EC numbers are in tree structure and the similarity of two enzymes could be determined through their number of common ancestors.

As we mentioned above, force-directed layout algorithms employ only graph structure and they do not take into account biological similarities. However, in the literature, there are studies that use force-directed algorithms for embedding domain-specific knowledge [2, 3, 4]. Nevertheless, none of them is specialized for EC Numbers. We modified and improved the force-directed layout algorithm in order to enhance visualization for biological graphs that contain vertices with EC Numbers.

Our algorithm, EClerize, is based on Kamada-Kawaii [5] algorithm which is a commonly used layout algorithm. EClerize treats vertices having the same EC class as if they are in the same cluster. EClerize adds virtual edges to create clusters. It is desired to minimize the distance between the vertices of the same cluster while to maximize the distance between two clusters. EClerize changes inter-cluster and intra-cluster distances according to parameters. Vertices without EC Numbers do not belong to any cluster and they are treated as regular vertices like they are treated in Kamada-Kawaii layout algorithm. Thus, EClerize draws more readable layouts for biologists keeping the aesthetically pleasing nature of force directed layout algorithms.

The remainder of this thesis is organized as follows. In Chapter 2, background information and related work are discussed. It covers basic information about the graphs and their layouts, related information in literature and biological background. In Chapter 3, our algorithm, EClerize, is presented. In Chapter 4, experimental results are given. It includes details of data sets used in experiments, discussion about the selection of parameters and the results. In Chapter 5, thesis is concluded and also future work is mentioned.

CHAPTER 2

BACKGROUND

2.1 Basic Information About Graphs

A graph $G = (V, E)$ is a structured model which is a representation of a set of vertices (also called "nodes") where some pairs of vertices are connected by edges. Graphs are popularly used in many applications, such as social networks, biological networks, navigation systems and database systems. Some basic definitions are given below.

Directed Graph also called digraph, is a graph whose edges are ordered pairs of vertices. In directed graphs, edges are drawn often as arrows.

Undirected Graph is a graph that all the edges are bidirectional. In undirected graphs, edges are drawn often as simple lines.

Weighted Graph is a graph that has a numeric value associated with each edge.

Unweighted Graph is a graph whose edges have no weights.

Shortest Path is the ordered set of edges in a graph such that the number of edges is minimized to traverse between two vertices.

Subgraph is a graph whose vertices are a subset of the vertex set of G, and whose edges are a subset of the edge set of G.

2.2 Graph Layout

Graphs are structural models that contain vertices and their relations. Mostly, graphs do not have location information, except map data. Graph layout algorithms aim to set vertices in right places to increase the readability of graphs. Hierarchical layout, circular layout, tree layout, orthogonal layouts and force-directed layout algorithms are popular graph layout algorithms [6].

Visual aesthetics of straight-lined undirected graph layouts is assessed by the following criteria: graph area, vertex placement, edge crossing and edge length [7, 6]. Graph area could be restricted to the size of screen or page. In this situation, the density of vertices would approach to massive numbers and the readability of the layout would get lower. On the other hand, drawing relatively small clusters to a large area makes relations of vertices untraceable when the size of an area is unlimited. Vertex placement also affects the graph area. Distribution of vertices affects readability as well. The distribution of vertices should be uniform and vertex overlapping should be avoided. Moreover, edge crossing reduces readability of graphs. It is important to draw graphs with minimum number of edge crossings. Finally, keeping edge lengths uniform is a generally accepted aesthetic criterion since it makes easier to trace paths in a graph. It is generally accepted that obeying visual aesthetic criteria makes graphs more readable and understandable [8]. An example random layout of a graph is given in Figure 2.1. Random layout of the graph does not point any information about graph. Result of the force-directed layout algorithm of the same graph is given in Figure 2.2 and it is much more readable than the random layout since it tries to follow the visual aesthetics criteria.

2.3 Force-directed Layout Algorithms

Force-directed layout algorithms or with well-known name spring-embedder algorithms are very popular among the graph layout algorithms. The force-directed layout algorithms simulate the graph as a physical system by assigning repulsive forces between vertices and attraction forces to edges.

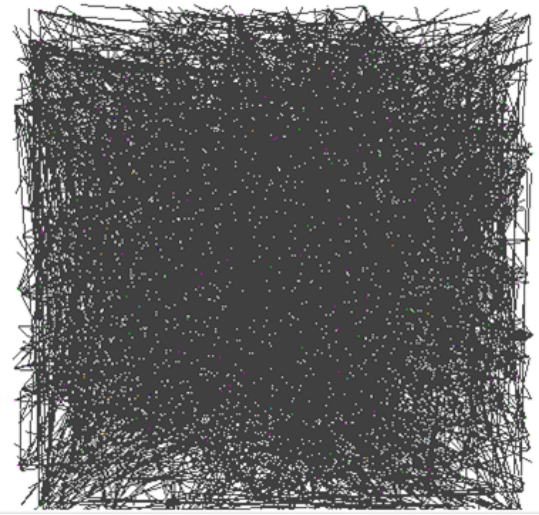


Figure 2.1: Random Layout of Signalling by PDGF Dataset [9]. It consists of 810 vertices and 1127 edges.

Figure 2.2: Force-directed layout of Signalling by PDGF Dataset. It is more readable than the random layout of the graph shown in Figure 2.1.

The approach described by Eades [10] is considered as one of the milestones of force-directed layout algorithms. In his method, vertices act like charged particles and repulse each other away. Moreover, the edges act like springs which bind the vertices together by an attraction force, as shown in Figure 2.3. Eades' algorithm takes edge lengths as uniform magnitudes and it reveals the symmetry in the graph. In Figure 2.4, pseudocode of spring algorithm of Eades is presented.

The aim of the algorithm is to minimize the energy function starting from random initial position. Because of various local minima of energy function, different initial configurations result in different local optima. Davis and Herald tried to avoid

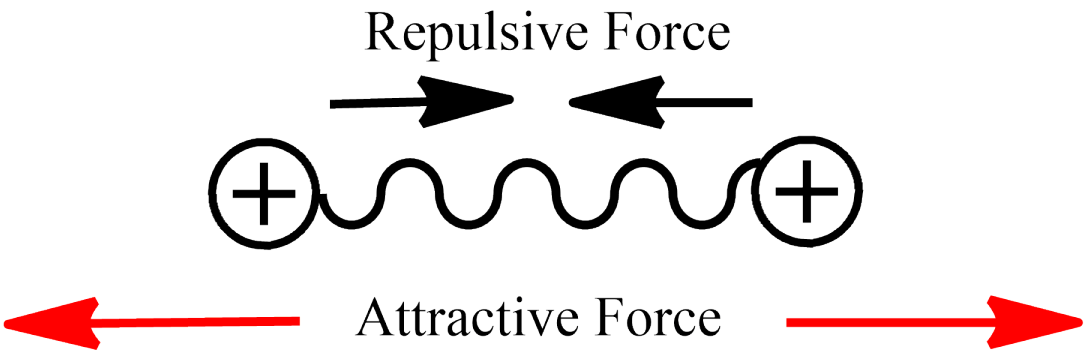


Figure 2.3: Repulsive and attractive forces in Spring-embedder algorithm.

Algorithm 1: Spring-Embedder

input: A Graph G =(V,E)
output: A nice layout of G

```
1:   for i := 1 to #Iterations do
2:        forces ← Calculate_Forces(G)
3:        Moves_Vertices(G,forces)
4:   end
```

Figure 2.4: Algorithm 1. Spring-embedder layout algorithm.

the local minima by using simulated annealing [11]. On the other hand, Kamada and Kawaii [5] proposed a different method that embeds graph theoretic distance approach into spring algorithm. They offer the following approach:

"The basic idea of our algorithm is as follows. We regard the desirable “geometric” (Euclidean) distance between two vertices in the drawing as the “graph theoretic” distance between them in the corresponding graph. We introduce a virtual dynamic system in which every two vertices are connected by a “spring” of such desirable length. Then, we regard the optimal layout of vertices as the state in which the total spring energy of the system is minimal.”

Kamada and Kawaii represented positions of vertices as $p_1, p_2, p_3 ....., p_n$ in a plane corresponding to vertices $v_1, v_2, v_3 ....., v_n \in V$ respectively. n is the number of ver-

tices, $|V|$. Energy function of Kamada-Kawaii layout algorithm based on the graph theoretic distance and it is defined as:

$$E = \sum_{i=1}^{n-1} \sum_{j=i+1}^{n} \frac{1}{2} k_{ij} (|p_i - p_j| - l_{ij})^2 \tag{2.1}$$

Pleasing layouts can be obtained by minimizing E. Minimum of E can be computed using the Newton-Raphson method [12]. l_{ij} is the desirable length between v_i and v_j and it is defined as

$$l_{ij} = L \times d_{ij} \tag{2.2}$$

where L is the desirable length of a single edge in the plane and d_{ij} is the shortest path between v_i and v_j.

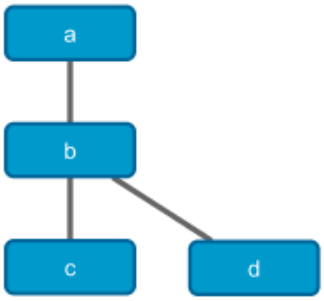
k_{ij} is the strength of springs between v_i and v_j and it is defined as

$$k_{ij} = \frac{K}{d_{ij}^2} \tag{2.3}$$

where K is the constant.

Kamada-Kawaii layout algorithm computes shortest distance of all-pairs (d_{ij}) and it is called "distance matrix" in this thesis.An example graph and its distance matrix shown in Figure 2.5. Kamada-Kawaii layout algorithm starts with a random initial configuration. Position of the vertex in the layout directly affects the movement requirement of that vertex and it is called stress of the vertex. Vertex that has the maximum stress is repositioned at each iteration until maximum number of iteration is reached or layout becomes stable. Pseudocode of Kamada-Kawaii layout algorithm is shown in Figure 2.6.

Force-directed layout algorithms draw small graphs efficiently. On the other hand, the force-directed layout algorithms might be problematic for drawing large graphs. Force-directed layout algorithms use multi-scale solutions to handle the problem of drawing large graphs. In multi-scale approach, vertices and edges transform into structures called super vertices containing lower level edges and vertices acting as



	a	b	c	d
a	0	1	2	2
b	1	0	1	1
c	2	1	0	2
d	2	1	2	0

Figure 2.5: A graph and its distance matrix.

Algorithm 2 Kamada-Kawaii

input: A Graph G = (V,E)
List of vertices V_{ignore} = (v ∈ V)
output: A nice layout of G

```
1:   Compute_Pairwase_Vertex_Distances(G)
2:   Compute_Pairwase_Ideal_Lengths(G)
3:   Compute_Pairwase_Spring_Strengths(G)
4:   for each v ∈ V do
5:        v.stress ← Calculate_Stress(G,v)
6:        if(v.stress > v_max.stress) do
7:             v_max ← v
8:        end
9:   end
10:  for i := 1 to #Iterations and v_max>threshold do
11:       v_max ← Move_Vertex_And_Find_Max(G,v)
12:  end
```

Figure 2.6: Algorithm 2. Kamada-Kawaii layout algorithm.

a group. After laid out, the super vertices are unpacked again and the full graph is obtained.

2.4 Basics of Clustering

Clusters are group of objects that satisfy the following conditions: objects in a group will be similar to each other and different from the objects in other groups. Cluster analysis helps to explore datasets and uncover hidden patterns. Revealed clusters changes with the clustering method. Cluster validation methods are used to compare how well different clustering algorithms perform on a dataset. There are many metrics to measure the quality in the cluster validation. Inter-cluster distance and intra-cluster distance are often used to asses cluster quality.

The inter-cluster distance measures distances between clusters. Different methods to measure the inter-cluster distances could be used such as, simple linkage, complete linkage, average linkage or centroid linkage [13]. In this thesis, the centroid linkage distance is used. The centroid linkage distance is described as:

$$D_{(inter-cluster)}(x, y) = c_x - c_y \tag{2.4}$$

where c_x and c_y are the centers of the clusters x and y, respectively.

Intra-cluster distance represents closeness between entities in the same cluster. Different methods to measure intra-cluster distances could be used such as, complete diameter, average diameter or centroid diameter [13]. In this thesis, the average diameter distance is used. The average diameter distance is described as:

$$D_{(intra-cluster)}(x) = \frac{1}{|x||x-1|} \sum_i \sum_j ||p_i - p_j|| \tag{2.5}$$

where x is the cluster and p_i and p_j are the positions of objects.

In a good cluster, the inter-cluster distance is maximized while the intra-cluster distance is minimized.

2.5 Literature

Traditional force-directed layout algorithms [4, 11, 10] do not consider size or shape of edges and vertices. The force-directed layout algorithms give insufficient layout quality in terms of vertex overlapping and edge-vertex crossing. Harel and Koren

provide a solution to the layout quality problem by considering size and shape of the vertices [1]. Moreover, a similar solution is proposed to prevent label overlapping [14, 15].

Force-directed layout algorithms generally start with random initial positions. Different initial configurations lead the algorithms into a final with a different local minimum at each time. Therefore, the initial configuration affects metrics like convergence, execution time and layout quality. Some algorithms offer methods to minimize the effects of initial configurations [16, 17]. Besides, Frick et al. suggest several new heuristics to improve the convergence, including local temperatures, gravitational forces and the detection of rotations and oscillations [18].

Calculation of position and velocity of n particles interacting with each other is an old problem in astrophysics and it is called as n-body problem. For large graphs, the calculation of force interaction between vertices takes a huge amount of time and it is similar to the n-body problem. The Barnes-Hut algorithm is a hierarchical $O(nlogn)$ force-calculation algorithm to solve the n-body problem [19]. Some studies attempt to solve the calculation of force interaction between vertices using the Barnes-Hut algorithm [20, 21, 22]. Furthermore, there are studies [21, 23, 24] in the literature that present multi-scale way to handle large graphs. For even better results, force-directed layout algorithms are adopted to the GPU architecture [25, 22, 26].

Visually analyzing large networks is an open problem because of its complexity. Clustered graph (also known as compound graphs) is a type of graph whose vertices are grouped into clusters, recursively. Visualizing large networks as clustered graphs is one of the most scalable and efficient solutions for handling the complexity [27, 28]. Generating layouts for clustered graphs is another problem. There are studies combining clusters and force-directed layout algorithms to resolve the layout problem [29, 30, 31, 32, 2]. So far, we mentioned methods that create clusters based on the graph structure. From now on, methods for adding clusters in the graph by using domain-specific knowledge will be discussed.

Creating clusters in graphs could be achieved by simply inserting virtual vertices which bound all the vertices in the same cluster. After inserting virtual vertices the vertices within a cluster will be positioned closer to each other than before [33].

Huang and Eades improved the force model by adding three different spring forces into their method, DA-TU [34]. Virtual spring forces of DA-TU model are: internal-spring, external-spring and virtual-spring. Internal-spring bounds vertices in the same cluster. External-spring connects different clusters. Each vertex in the cluster is connected with the virtual vertex via virtual-springs. Altogether, the virtual spring forces define the force model of DA-TU.

Customized force-directed layout algorithms could be used to construct an enhanced visualization of the biological graphs. Genc and Dogrusoz [2] designated an algorithm for signaling pathways. Their algorithm takes into account the directional and rectangular regional constraints which represent specific areas in the cell. Moreover, Becker and Rojas [3] stated that the customized spring embedding algorithm with a combination of circular and hierarchical layout algorithms give better results for metabolic pathways. Furthermore, GOlorize [35] is a customized layout algorithm that uses Gene Ontology(GO) annotations. Gene Ontology (GO) project provides a set of hierarchical controlled vocabulary split into 3 categories: Biological process, Molecular function and Cellular component [36]. GOlorize aims to emphasize the biological function of the vertices by using GO annotations as a source of the domain-specific knowledge. Each vertex can belong to none, a unique cluster or several clusters. The algorithm of GOlorize consists of three phases. At the first phase, the Freuchterman and Reinghold layout algorithm [4] is executed with the extra attraction force that is applied to the vertices belonging to the same cluster. At the second phase, clusters are separated by moving the cluster vertices away from the center of the graph. At the final phase, the Freuchterman and Reinghold layout algorithm is executed again but this time cluster vertices are fixed at the positions determined in the second phase. Hereby, final visualization of the GOlorize layout algorithm is generated.

2.6 Biological Background

Enzymes are proteins that accelerate or catalyze chemical reactions. Enzymes convert organic substrates, which are molecules at the beginning of the process, into different molecules called products [37]. An active site is the part of an enzyme and it has the

right shape to bind to one of the substrates. Active site of an enzyme is illustrated in Figure 2.7.

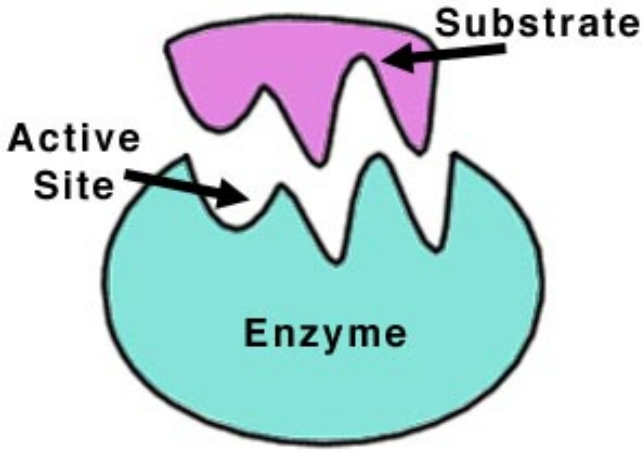


Figure 2.7: Active site of an enzyme is a perfect fit for a specific substrate [38].

Activation energy is the minimum energy required to start a reaction. Enzymes lower the activation energy of a chemical reaction, as a result, rate of the reaction is increased. It is reported that some enzymes could speed up the reaction many millions of times faster [39]. Many reactions would not be possible without an enzyme. A simple model that represents a reaction with an enzyme is given in equation 2.6.

$$Enzyme + Substrate \Longrightarrow Enzyme - SubstrateComplex \Longrightarrow Product + Enzyme \tag{2.6}$$

Enzymes do not alter inputs or outputs of a reaction, they only affect the reaction in terms of speed. An enzyme could be used at another reaction after the reaction is done. Enzyme activity could be affected by two kind of molecules: activators and inhibitors. Activators affects enzyme activity positively while inhibitors decrease enzyme activity. Many drug molecules are enzyme inhibitors.

Enzymes are critical for metabolic reactions. The cell controls enzyme activities according to its demands. Through this mechanism, it decides what would be produced by using substrates. For example, pyvurate could be used to produce acetyl-CoA, fatty acid biosynthesis or different amino acids. In addition, pyvurate is used by some organisms for the generation of ethanol through converting into lactate [41]. Figure 2.8 shows the reference pathway of pyvurate metabolism that is taken from KEGG

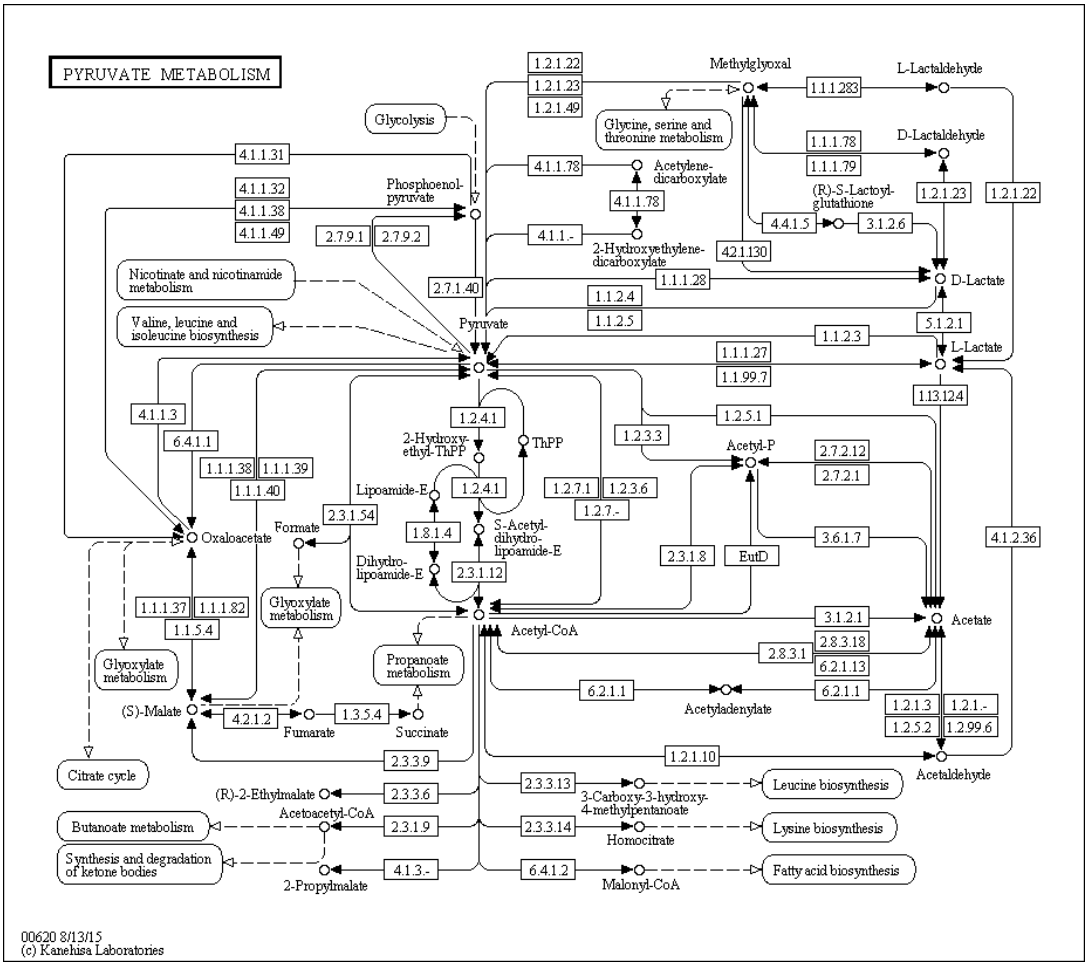


Figure 2.8: Pyvurate Metabolism [40]. EC numbers of enzymes that involved in reactions shown in boxes.

database [42].

Enzyme Commission Number (EC Number) is defined as a classification scheme for enzymes by the Nomenclature Committee of the International Union of Biochemistry and Molecular Biology, based on which reaction they catalyze. Tree structure of EC numbers is illustrated in Figure 2.9 and details of EC classes are given in Table 2.1. There are 6 trees that represent EC classes. The maximum depth of a tree is always 4.

Structure of EC number format is illustrated in Table 2.2. For example, the meaning of EC number "1.2.3.4" is as follows: the first number "1" represents the class of the enzyme which is "Oxidoreductase". "1.2" represents the subclass information of the enzyme which is "Acting on the aldehyde or oxo group of donors". "1.2.3" represents the sub-subclass of the enzyme which is "With oxygen as acceptor". The last number

Table 2.1: List of EC Numbers

Class	Name	Detail
1	Oxidoreductases	Catalyze oxidation-reduction reactions
2	Transferases	Catalyze transfer of functional groups
3	Hydrolases	Catalyze cleavage of bonds by addition of water
4	Lyases	Catalyze group elimination reactions to form double bonds
5	Isomerases	Catalyze racemization of optical or geometric isomers
6	Ligases	Catalyze formation of bonds between reactions couples with ATP hydrolysis

Table 2.2: Format of the EC Number "1.2.3.4"

1	2	3	4
Class	Subclass	Sub-subclass	Serial Number

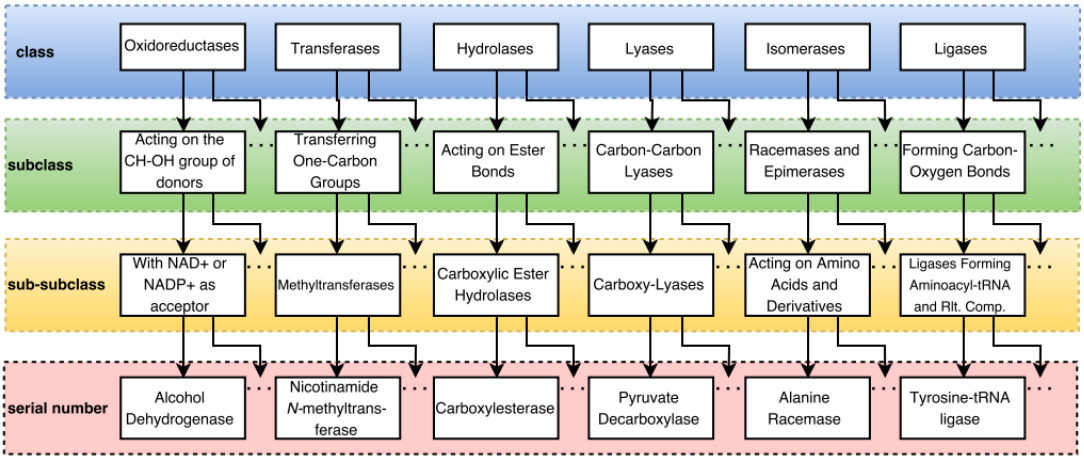


Figure 2.9: Tree structure of Enzyme Commission Numbers. Levels correspond to class number, subclass number, sub-subclass number and serial number respectively.

is the serial number and the EC number "1.2.3.4" is the final classification of that enzyme which is "oxalate oxidase".

Enzymes are the most crucial mechanism in the cell to sustain life. For example, the enzymes that add phosphate groups to proteins are known as kinases(EC 2.7). Serine/Threonine protein kinase(EC 2.7.11) enzymes have a role in the regulation of cell proliferation, programmed cell death, cell differentiation and embryonic development [43]. Because of the role in the cell, mutation of kinases could lead to cancer.

The term $in\ silico$ is used to refer to experiments that performed on computer or via computer simulation which are commonly used in molecular biology. $in\ silico$ methods could help researchers working on drug target discovery and diagnostic tests by reducing time and money. There is huge amount of data in molecular biology. In addition, graphs are widely used to represent relations of enzymes with other molecules. Visualising graphs by including domain-specific knowledge would help researchers to understand the relations between enzymes and the other structures.

CHAPTER 3

ECLERIZE ALGORITHM

Enzymes are proteins that play essential roles in life. It is important to understand enzymes and their function for researchers working on drug target discovery and diagnostic tests. An international committee (Nomenclature Committee of the International Union of Biochemistry and Molecular Biology) classifies enzymes based on what they catalyze. The numerical classification of an enzyme is called the Enzyme Commission (EC) Number and it shows the biological similarity of enzymes. Biological graphs such as metabolic pathways contain vertices that represent enzyme molecules. Force-directed layout algorithms are widely used and studied to draw biological graphs and they could be modified to enhance visualization. This thesis aims to create an improved and modified force-directed layout algorithm that would draw more readable layouts for biological graphs that represent pathways including enzyme molecules.

Our algorithm, EClerize, is based on Kamada-Kawaii force-directed layout algorithm and works on unweighted and undirected graphs. Vertices with the same EC class are treated as members of the same cluster. In addition, unclustered vertices are allowed since most of the vertices in biological graphs do not have EC Number attribute. Virtual edges are added between the vertices in the same cluster. Then, strengths of springs of virtual edges are modified according to a formula that represents biological similarity of enzymes. At the final layout, positions of vertices are affected by mainly two factors:

1. Biological similarity of each vertex in the same cluster and

2. Theoretical length between the vertices.

The pseudocode of EClerize is presented in Figure 3.1 . The following parameters are input to the algorithm;

- list of the vertices with EC numbers;
- strength of spring of EC vertices S (as described in Section 3.2);
- distance factor for clusters c (as described in Section 3.3).

3.1 Overview of EClerize

```
     Algorithm 3: EClerize
input: A Graph G=(V,E)
     List of vertices Vec = (v ? V), v has EC number
     constant S, strength of spring of EC edges
     constant c, distance factor for clusters
output: A nice layout of G

1:   Add_Virtual_Edges(G,Vec)
2:   dist ← Compute_Distance_Matrix(G)
3:   lengths ← Compute_Pairwase_Ideal_Lengths(G,dist)
4:   strn ← Compute_Pairwase_Spring_Strengths(G,dist)
5:   Apply_Spring_Model(G,Vec,S,strn)
6:   Layout_Algorithm(G, null, lengths, strn)
7:   if(c ≠ 0) do
8:        Increase_Inter_Cluster_Distance(G, Vec ,c)
9:        Layout_Algorithm(G, Vec, lengths, strn)
10:  end
```

Figure 3.1: Algorithm 3. Pseudocode of the EClerize layout algorithm.

Generally, force-directed layout algorithms attempt to minimize an energy function. Kamada-Kawaii layout algorithm is one of the most popular force-directed algorithms and it tries to find the minimum point of an energy function by taking the graph theoretic distance into account. Kamada-Kawaii layout algorithm uses distance-matrix which contains the shortest paths between all pairs of vertices. By using the dis-

tance matrix, the ideal vertex distances and the strengths of springs are also calculated. EClerize is a modified version of the Kamada-Kawaii layout algorithm. The strengths of springs in the original algorithm are replaced with a function that represents the graph theoretic distance between vertices and the biological similarity between enzymes.

In the first step of Figure 3.1, we add virtual edges between vertices in the same cluster. This is performed by the method $Add\_Virtual\_Edges(G, Vec)$. For example, Figure 3.2 presents the final layout without virtual edges while as an addition to Figure 3.2, red edges in Figure 3.3 are drawn to illustrate virtual edges. In an undirected complete graph, there is an undirected edge between each possible pair of vertices. Added virtual edges compose an undirected complete graph with the vertices in the same cluster.

After the first step, the structure of the graph stays constant during the execution of algorithm. The distance matrix is calculated in step 2 of the Algorithm 3. Ideal vertex distances and strengths of springs are obtained by using node distances in Steps 3 and 4. In step 5 function $Apply\_Spring\_Model(G, Vec, S, strn)$ adjusts the strengths of springs of vertices with EC numbers according to a method which will be described in details at Section 3.2.

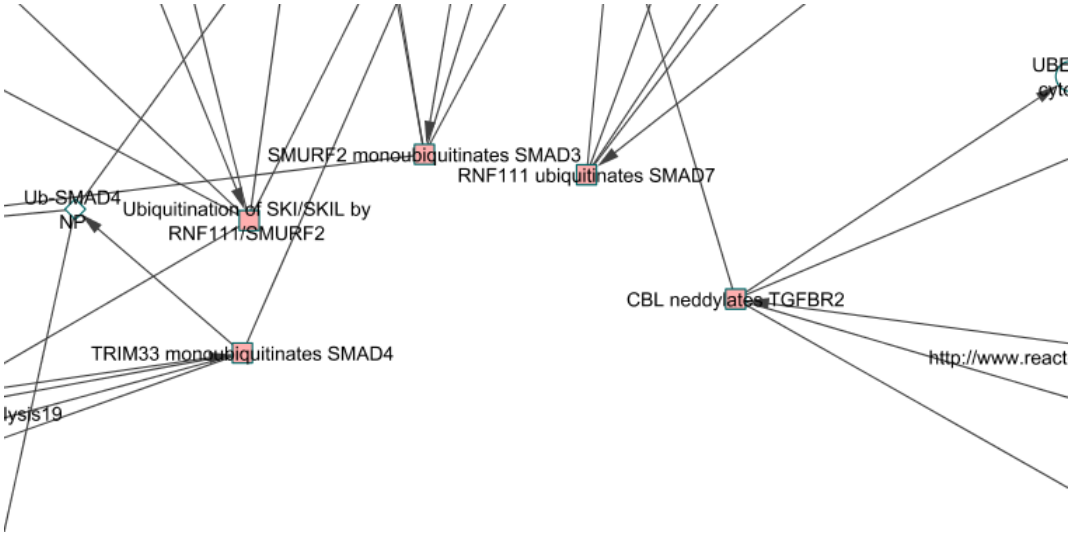


Figure 3.2: A part of final layout of EClerize layout algorithm on a graph. Vertices with orange color have the same EC class.

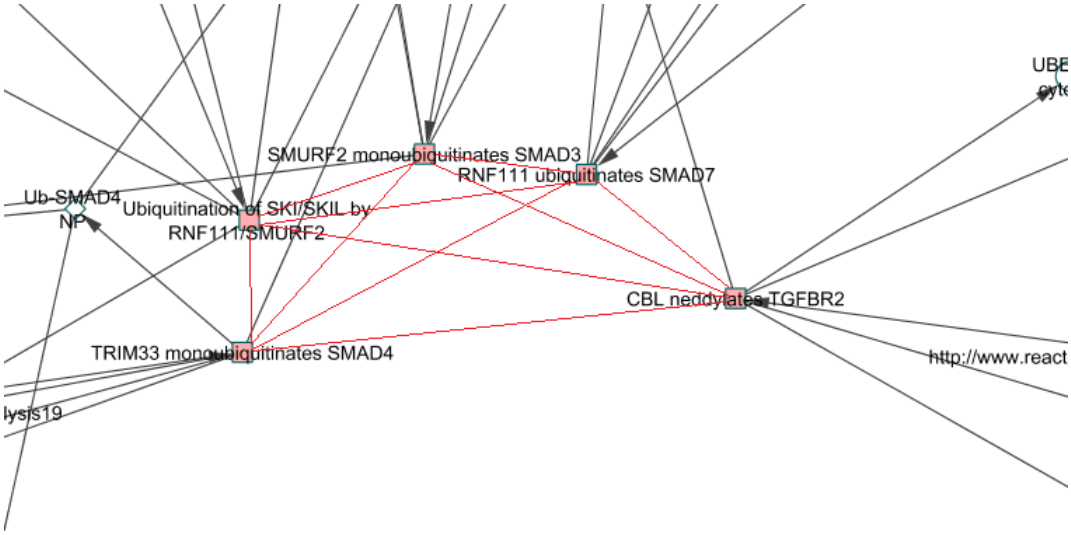


Figure 3.3: Virtual edges are added to the graph shown in Figure 3.2 and they are illustrated with red lines.

Until this point, we determine initial values that will be used in the customized force-directed algorithm. The Layout Algorithm which is given in Algorithm 4 is the core part of the layout methodology presented in this thesis, and it is based on the Kamada-Kawaii force-directed algorithm. Last part of Algorithm 3, steps 7 to 10, is focused on increasing inter-cluster distances. Increasing inter-cluster distances might not be desired in all cases. In the case of $c = 0$, the algorithm will stop the execution and final layout will be drawn. Otherwise the Layout Algorithm increases inter-cluster distances at step 8 in the method $Increase\_Inter\_Cluster\_Distance(G, V_{ec}, c)$ which will be described in detail at Section 3.3. After increasing inter-cluster distances, positions of EC vertices are fixed. Ultimately, the Layout Algorithm will execute. The aim of this repeated execution is to adjust positions of non-clustered vertices finely, not only according to EC class information, but also using their underlying pattern of interactions [35].

The Layout Algorithm, shown in Figure 3.4 terminates either when maximum number of iterations is reached or when maximum stress of all vertices is below a threshold. Parameters of Algorithm 4 are graph $G = (V, E)$, list of vertices $V_{ignore} = (v \in V)$. V_{ignore} is the list of vertices that are ignored by the Layout Algorithm, "lengths" is the graph theoretic distance of vertices and "strn" is the strengths of springs, respectively. The Layout Algorithm ignores vertices $v \in V_{ignore}$, so positions of vertices $(v \in$

Algorithm 4 Layout_Algorithm

```
input: A Graph G = (V,E)
       List of vertices V_ignore = (v ∈ V)
       lengths, ideal lenghts
       strn, strengths of springs
output: A nice layout of G

1:   for each v ∈ V do
2:        v.stress ← Calculate_Stress(G, v, lengths, strn)
3:        if(v.stress > v_max.stress and v ← V_ignore ) do
4:             v_max ← v
5:        end
6:   end
7:   for i := 1 to #Iterations and v_max>threshold do
8:        v_max←Move_Vertex_And_Find_Max(G,v, lengths, strn)
9:   end
```

Figure 3.4: Algorithm 4. Pseudocode of the core part of the EClerize.

V_{ignore}) are fixed. V_{ignore} is empty at first call and list of EC vertices is passed at second call from Algorithm 3.

First, stress of vertices are calculated. EClerize starts by repositioning vertices with the vertex that has the maximum stress. The vertex with the maximum stress is the most suitable candidate to lower the energy function. In the second part, steps 7 to 11, the most stressed vertex is repositioned until either the maximum number of iterations is reached or the maximum stress of vertices is below a threshold. The movement of one vertex affects all the others in the graph. Before moving the most stressed vertex, energy of that vertex is subtracted from other vertices. After the moving process, energy of the moving vertices affects others. The method $Move\_Vertex\_And\_Find\_Max(G, v, lengths, strn)$ moves the most stressed vertex and goes to the next vertex.

3.2 Spring Model

Kamada-Kawaii layout algorithm employs only graph structure. Edge set in the graph and the strengths of springs in the Kamada-Kawaii layout algorithm are modified to embed domain-specific knowledge. The modified model of the edge set and the strengths of springs are called as $Spring\ Model$ in this thesis. Spring Model consists of two phases. At first, virtual edges are added to the vertices that are in the same cluster to obtain a complete graph. Second, the strength of springs is modified to reflect biological similarity. The equation that determines the strength of springs in the original algorithm (Equation 2.3) is redefined as follows.

$$k_{ij} = \begin{cases} n_{ij}^2 * S * K & \text{if } i \in V_{ec} \text{ and } j \in V_{ec} \\ K/d_{ij}^2 & \text{otherwise} \end{cases} \tag{3.1}$$

where k_{ij} is the edge's strength of spring, between v_i to v_j; n_{ij} is the number of common ancestors of v_i and v_j; S is the parameter that determines the strengths of springs in cluster and K is the constant for strengths of springs. The term n_{ij}^2 is determined by inspecting the results of experiments. Algorithm for calculation of strengths of springs for EC vertices can be seen in Figure 3.5.

The number of common ancestors is determined by parsing EC numbers. An example of this can be seen in Figure 3.6. Class "1" and sub-class "1.1" are matching. But sub-subclass'("1.1.1" and "1.1.2") do not match with each other. Enzymes classified with EC 1.1.1.1 and EC 1.1.2.1 are in different sub-subclass and they have 2 common ancestors in the EC tree. Strength of the related edge's spring is adjusted according to Equation 3.1. So strengths of springs will be multiplied by $2^2 * S$. When S is 1, these vertices attract each other as if they were connected by 4 edges. An illustration of strengths of springs of edges between EC vertices when $S = 1$ is given in Figure 3.7

EClerize includes biological similarity into the energy function by means of proposed Spring Model. As a result of this, EClerize places vertices closer, which has more common ancestor than the other vertices in the EC tree.

Algorithm 5 Apply_Spring_Model

input: A Graph G = (V,E)
List of vertices V_{ec} = (v ∉ V), v has EC number
constant S, constant for strength of spring of EC edges
set strn, strengths of springs
output: Strengths of springs

```
1:   for each v_s ∈ V_ec do
2:        for each v_t ≠ v_s ∈ V_ec do
3:             n = Number_Of_Common_Ancestor(v_s, v_t)
4:             v_s.strength[v_t] ← n^2 * S * strn[v_s]
5:        end
6:   end
```

Figure 3.5: Algorithm 5. Spring model embeds the biological similarity into layout by changing strengths of spring of the EC vertices.

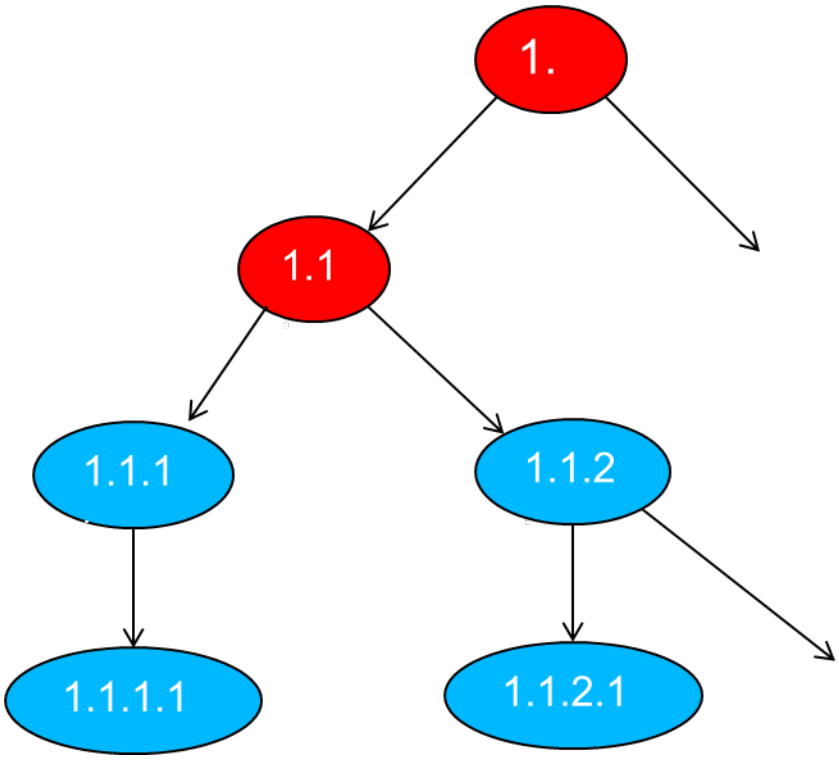


Figure 3.6: An example comparison of EC numbers on the EC number tree.

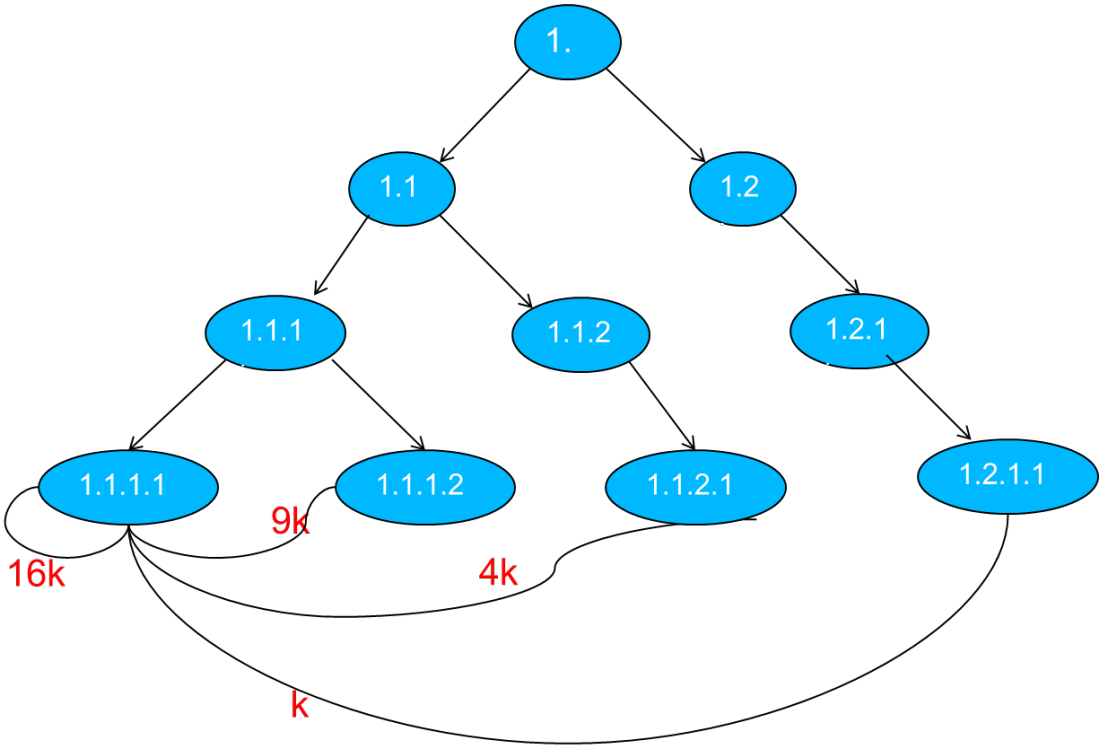


Figure 3.7: Illustration of strengths of springs of edges between EC vertices when $s = 1$.

3.3 Distance Factor

Algorithm 6, shown in Figure 3.8, aims to create spacing between clusters to increase readability. In other words, it aims to increase inter-cluster distances. Before the execution of Algorithm 6, the energy of the graph should have already reached a local minimum. As we move vertices away, the energy of the graph would get away from the minimum point. We present a heuristic method to increase inter-cluster distance as follows: move "small cluster" more and to move "large cluster" less. By this means, energy function will fall in to minimum easier.

Clusters are determined with respect to the EC numbers. Maximum 6 clusters could exist because of the structure of EC tree. Then center of the gravity of the vertices with EC number is calculated. Each cluster moves out from the center of the gravity according to parameter c. Each cluster moves out as $\Delta_{cluster}$ and it is defined as follows:

```
     Algorithm 6: Increase_Inter_Cluster_Distance
input: A Graph G = (V,E)
     List of vertex Vec = (v ← V), v has EC number
     constant c, distance factor for clusters
output: A nice layout of G

1:   clusters ∈ Find_Clusters(Vec)
2:   center_ec ← Center(Vec)
3:   for each cl ← clusters do
4:        diff ∈ Center(cl) - center_ec
5:        increment ← diff * c
6:        Increase_Positions(cl, increment)
7:   end
```

Figure 3.8: Algorithm 6. The pseudocode of the part of the EClerize that increases inter-cluster distances.

$$\Delta_{cluster}(x, y) = (center_{V_{ec}} - center_{cluster}) * c \tag{3.2}$$

$$center_V(x, y) = \frac{\sum_{i=1}^{n} v_i(x, y)}{n} \tag{3.3}$$

The method, $Increase\_Inter\_Cluster\_Distance$, makes the vertices in each cluster move to their new location, according to equation below:

$$V(x, y) = V(x, y) + \Delta_{cluster}(x, y) \tag{3.4}$$

Effects of parameter c can be seen in Figure 3.9 to Figure 3.11. As distance factor increases, "small cluster" moves out from the center of gravity more than "large cluster".

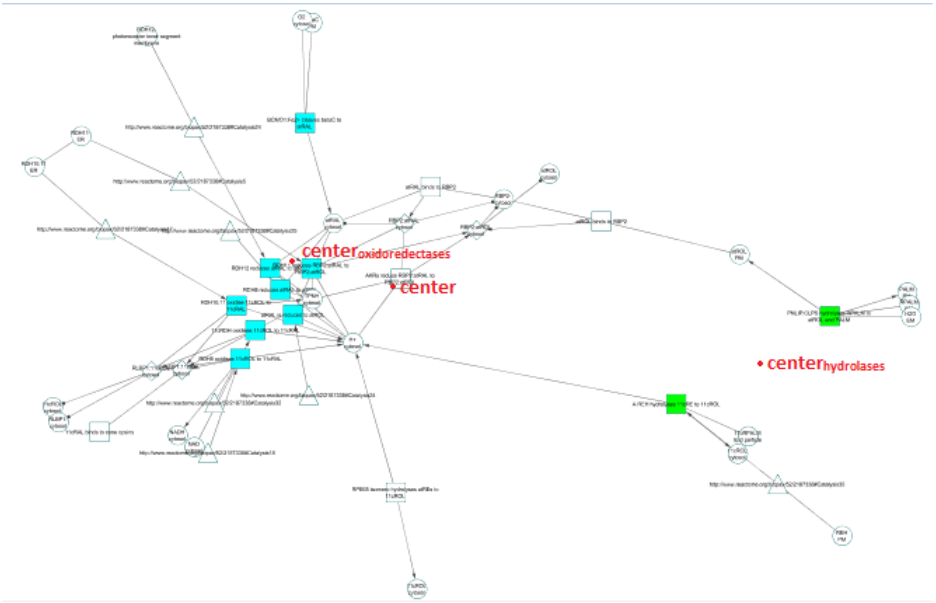


Figure 3.9: Inter-cluster distances are fixed with the distance factor $c = 0$.

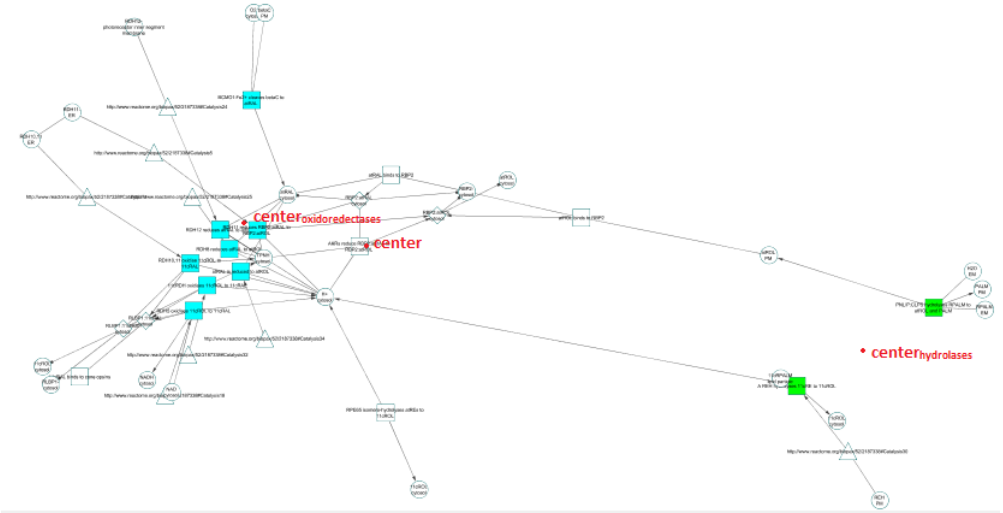


Figure 3.10: Inter-cluster distances are increasing with the distance factor $c = 0.5$.

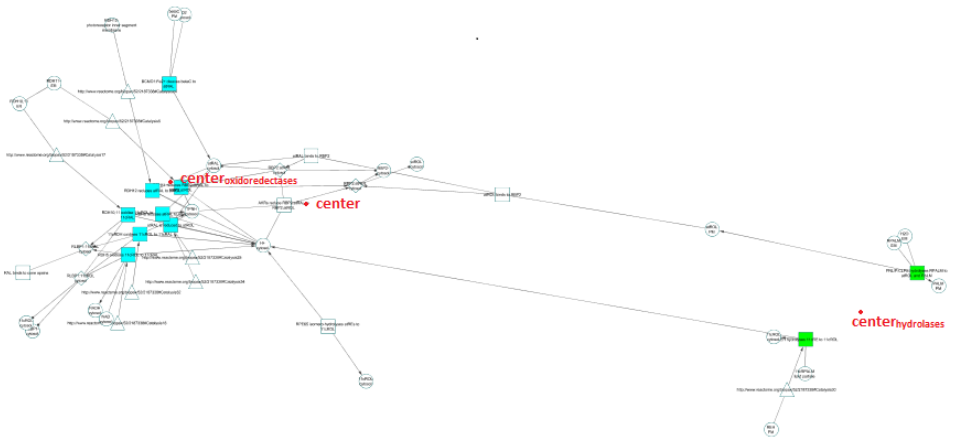


Figure 3.11: Inter-cluster distances are increasing with the distance factor $c = 1$.

CHAPTER 4

EXPERIMENTAL RESULTS

The proposed layout algorithm is developed under the framework of Cytoscape (version of 3.2), which is an open source software platform for visualizing molecular interaction networks and biological pathways and integrating these networks with annotations, gene expression profiles and other state data [44]. Experiments are carried out using 64-bit Windows 7 operating system and on an Intel Core i7-4700HQ 2.4GHz processor with 12 GB RAM. Our implementation is based on the spring embedded layout method of Cytoscape. Spring embedded layout method is also used as a reference to assess our results.

4.1 Datasets

Datasets used in this work are obtained from Reactome database. Reactome is a curated, peer-reviewed database of human biological resource. Reactome consists of units called as reaction. A reaction is any event that converts inputs into outputs. In this conversion process, physical entities are used as inputs and outputs. Reactions then compose pathways together [45]. Reactome database provides different download options to its users for pathways. BioPAX is one of those options and it is one of the common languages to exchange biological pathway data. First, pathways are downloaded under "BioPAX 3" Format and then imported into Cytoscape environment. We performed the experiments of EClerize on three different data sets: Visual phototransduction, Signaling by ERBB2 and Signaling by EGFR data. Details of data sets are shown in Table 4.1 and Table 4.2

Table 4.1: Details of datasets

Dataset	# of Vertices	# of Edges	# of EC Vertices
Visual phototransduction	542	767	28
Signaling by ERBB2	731	1109	38
Signaling by EGFR	779	1209	39

Table 4.2: Number of EC Vertices in Clusters

Dataset	Visual Phototransdution	Signaling by ERBB2	Signaling by EGFR
EC Class 1	9	0	0
EC Class 2	6	31	30
EC Class 3	11	2	3
EC Class 4	1	0	0
EC Class 5	1	0	0
EC Class 6	0	5	6
Total	28	38	39

4.2 Parameter Setup

EClerize works based on Kamada-Kawaii layout algorithm. Kamada-Kawaii layout algorithm requires some parameters to work on graphs: strength of spring constant for connected vertices, force constant for disconnected vertices, maximum iteration number or maximum iteration number per vertices. Selection of these parameters is not a concern of this study and these parameters are not changed during these experiments.

In addition to force-directed parameters, EClerize needs two extra parameters: S and c. S is the constant used for calculation of strengths of springs of EC vertices, which is described previously in Chapter 3. Parameter S affects inter-cluster distances. c is the constant for distance formula which will be used in the part of the main algorithm that will increase the inter-cluster distances between clusters. The aim of this section is to determine a meaningful interval for parameter c and S.

4.2.1 Spring constant for vertices that have EC numbers (parameter S)

S directly affects the intra-cluster distance since it will determine strengths of springs of the EC nodes in same cluster. $S = 0$ is added to experiments to show this effect. Algorithm with parameter $S = 0$ and $c = 0$ give us the same result as unmodified Kamada-Kawaii force-directed algorithm.

Figure 4.1, Figure 4.2 and Figure 4.3 present graphs that show the relation between parameter S and intra-cluster distance when $c = 0$. There are different methods to calculate intra-cluster distances. We use the average diameter distance which represents the average distance between all samples belonging to the same cluster. The algorithm produces the maximum intra-cluster distance when $S = 0$, which is expected since it means there is no attraction for the EC vertices in same cluster. Then even with the smallest value of S, intra-cluster distances fall down radically. In Figure 4.1 and Figure 4.3, clusters gather up with small values of S. On the other hand, in Figure 4.2, cluster 2 requires higher strengths of springs than others to gather up; since it is distributed well in the graph. We run EClerize until $S = 1$. Since after $S > 1$ intra-cluster distance does not change anymore. Intra-cluster distance becomes stable when S is bigger than 0.4 in all results.

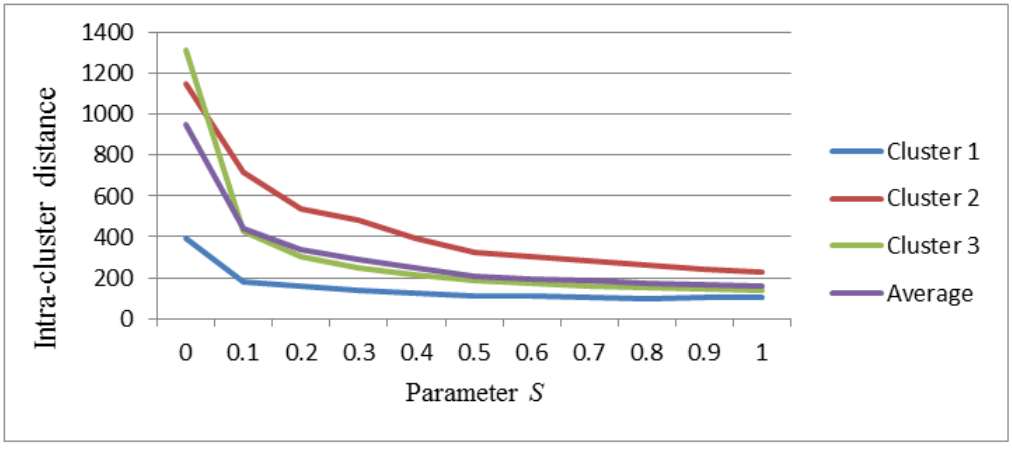


Figure 4.1: Relation between parameter S and intra-cluster distance when $c = 0$ on Visual Photo Transduction dataset.

Figure 4.4, Figure 4.5 and Figure 4.6 show the relation between parameter S and number of edge crossing. At the beginning of the experiment increment of edge of

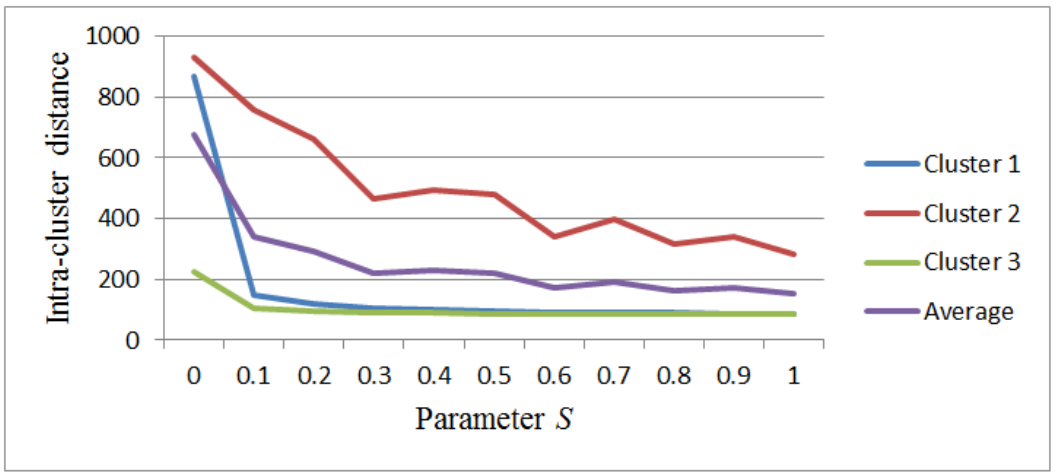


Figure 4.2: Relation between parameter S and intra-cluster distance when $c = 0$ on Signaling by EGFR dataset.

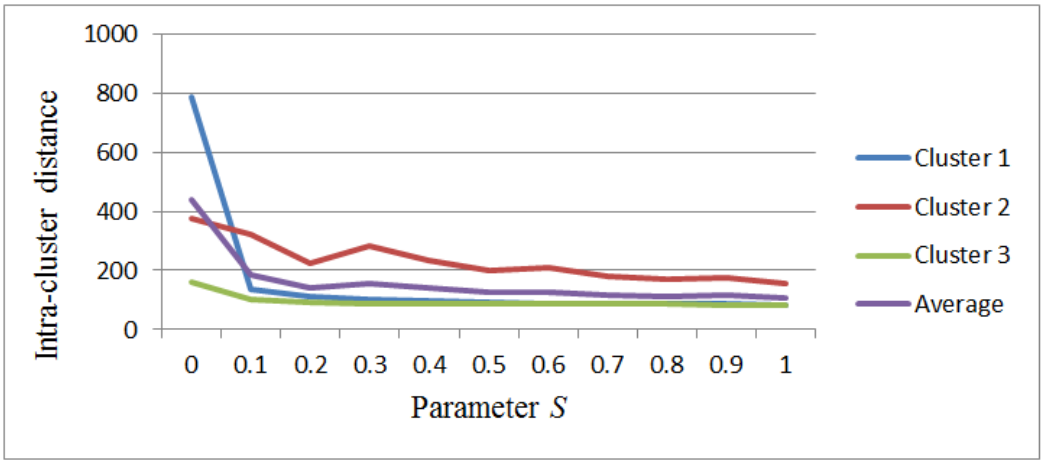


Figure 4.3: Relation between parameter S and intra-cluster distance when $c = 0$ on Signaling by ERBB2 dataset.

crossing is fast, and then became slower. We should find a range for parameter S, that would avoid higher number of edge of crossing and give satisfying intra-cluster distance. By looking at results, it is advised to use parameter S as: $0.1 < S < 0.4$.

4.2.2 Constant for Distance Factor (parameter c)

Table 4.3, Table 4.4 and Table 4.5 show us the effect of parameter c on size of drawing area, inter-cluster distance and execution time. One of the aesthetic goals of many

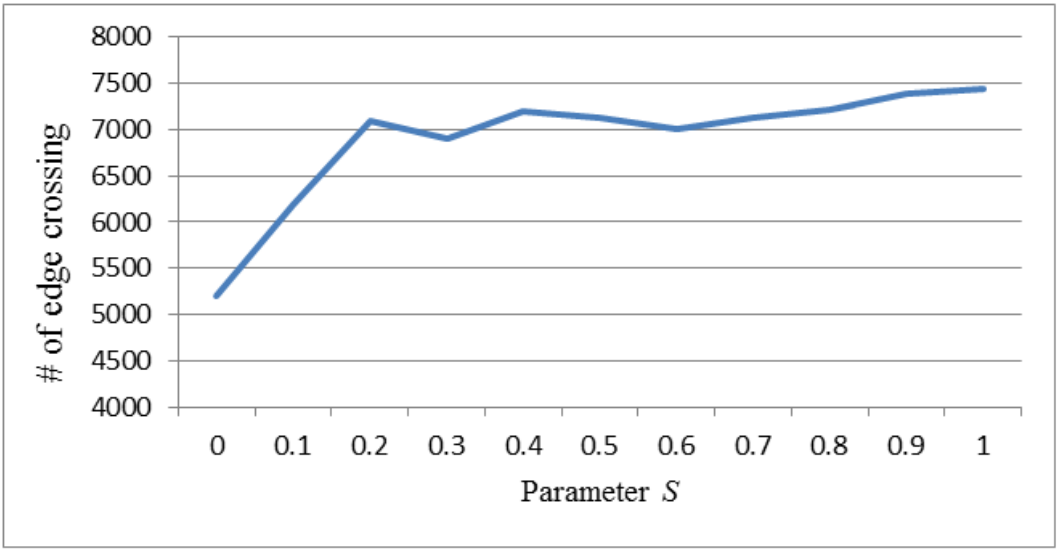


Figure 4.4: Relation between parameter S and number of edge crossing when $c = 0$ on Visual Photo Transduction dataset.

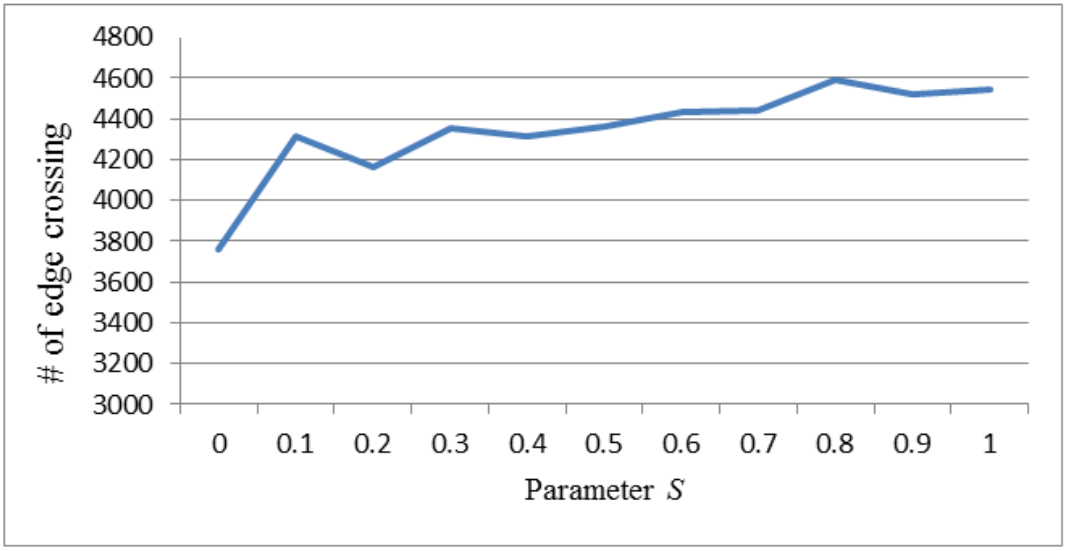


Figure 4.5: Relation between parameter S and number of edge crossing when $c = 0$ on Signaling by EGFR dataset.

layout algorithms is to minimize the drawing area. Increasing the drawing area is not desired. On the other hand increasing inter-cluster distance is a good thing. In this experiment average of centroid linkage distances, which reflects the distance between centers of two clusters, is used as inter-cluster distance. As parameter c increased, vertices are moving around far in the graph and thus the graph is getting away from

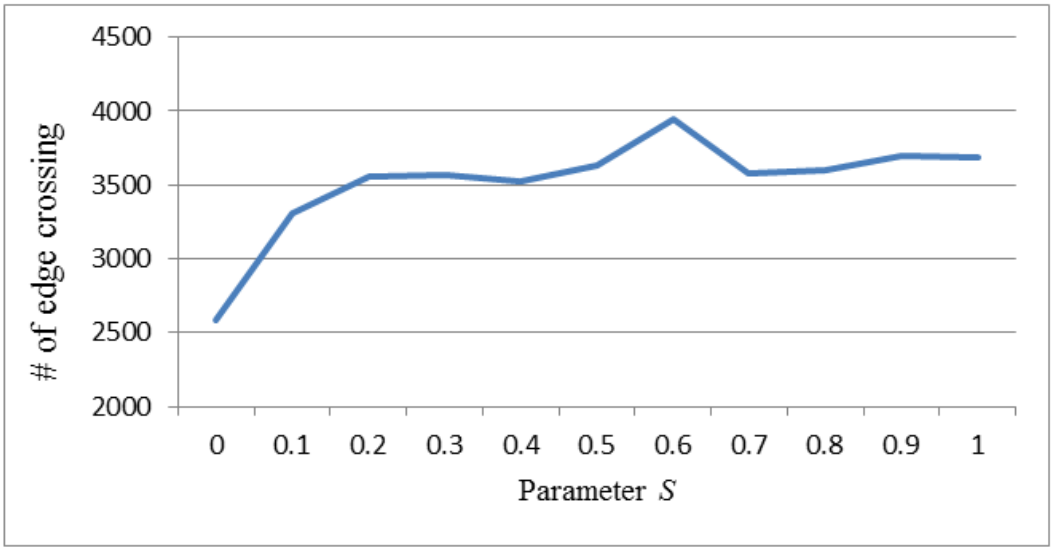


Figure 4.6: Relation between parameter S and number of edge crossing when $c = 0$ on Signaling by ERBB2 dataset.

Table 4.3: Drawing area, inter-cluster distance and execution time(ms) when $S = 0.1$ on Signaling by EGFR dataset

c	Drawing Area	Inter-Cluster Distance	Time(ms)
0.1	1060	1193	8854
0.2	1060	1302	8983
0.3	1057	1410	9155
0.4	**1058**	1519	9286
0.5	1069	1628	9178
0.6	1089	1736	9468
0.7	1107	1845	9459
0.8	1129	1953	9560
0.9	1160	2062	9687

local minima more. As a result of this, as c gets higher, more time will be required to find a minimum point in energy function. This can be seen in Table 2 and this deduction is also pointed out in Section 4.3. So it is crucial to choose a c value that will not increase the drawing area and time when inter-cluster distance is at maximum.

The drawing area of Visual phototransduction shows that after c value 0.4, drawing area is constantly increasing. This break point is 0.1 for Signaling by ERBB2 and 0.2 for Visual phototransduction. Exceeding these values tends to position clusters

Table 4.4: Drawing area, inter-cluster distance and execution time(ms) when $S = 0.1$ on Signaling by ERBB2 dataset

c	Drawing Area	Inter-Cluster Distance	Time(ms)
0.1	**1049**	1275	7926
0.2	1053	1391	8217
0.3	1079	1507	8133
0.4	1115	1623	8329
0.5	1151	1739	8220
0.6	1185	1855	8400
0.7	1213	1971	8446
0.8	1252	2087	8717
0.9	1287	2203	8669

Table 4.5: Drawing area, inter-cluster distance and execution time(ms) when $S = 0.1$ on Visual phototransduction dataset

c	Drawing Area	Inter-Cluster Distance	Time(ms)
0.1	1025	847	7379
0.2	**1025**	924	7454
0.3	1030	1001	7424
0.4	1033	1078	7405
0.5	1034	1155	7469
0.6	1037	1232	7461
0.7	1047	1309	7660
0.8	1062	1386	7648
0.9	1075	1463	7779

outside the drawing area. As well as it depends on the structure of the graph, it is advised to put an upper limit to c with value of 0.4.

4.3 Results

We run the EClerize multiple times with input graphs which have different number of vertices. We measured the execution time and the number of edge crossing of the EClerize at each run. In this way, we assessed the EClerize with respect to graph size. Since we could not find real life small data set samples to work on at different sizes,

we constructed a new method to create sub graphs (smaller graph) at specific sizes from a larger graph. This new method can be seen in Algorithm 7. It is based on Depth First Search (DFS) approach.

Algorithm 7 Create_Subgraph

```
input: A Graph G = (V,E)
     Starting point v ← V
     Maximum number of vertices, m
output: A Subgraph S = (V,E)

1:   e_list = find_unvisited_edges(v)
2:   for each e ∈ e_list do
3:        if(S.size < m) do
4:             if(!v.visited) do
5:                  v.visited = true
6:                  add_to_subgraph(v);
7:             end
8:             if(!e.visited) do
9:                  e.visited = true
10:                 add_to_subgraph(e);
11:            end
12:            Create_Subgraph(G,e.other,m);
13:       end
14:       if(!e.visited and e.nodes.visited) do
15:            e.visited = true
16:            add_to_subgraph(e);
17:       end
18:  end
```

Figure 4.7: Algorithm 7. The pseudocode of the method $CreateSubgraph$ that creates new subgraphs from a real biological dataset for comparisons.

Method $CreateSubgraph$ is called with parameters that are gradually increased. Greater subgraphs are created with each call and results are shown in Figure 4.8 and Figure 4.9.

The execution time of EClerize with parameter $c = 0$ and the execution time of unmodified force-directed algorithm are almost same. On the other hand EClerize with parameter $c = 0.1$ requires more time than the unmodified version; since after increasing inter-cluster distances, the main layout algorithm is called again as described in Section 3.1.

Figure 4.10, Figure 4.11 and Figure 4.12 shows the layout of EClerize while S is increasing and $c = 0$ on Visual Photo Transduction dataset. Intra-cluster distance is minimized while parameter S is increasing. In addition, Figure 4.13, Figure 4.14 and Figure 4.15 shows the layout of EClerize while c is increasing and $S = 1$ on Visual

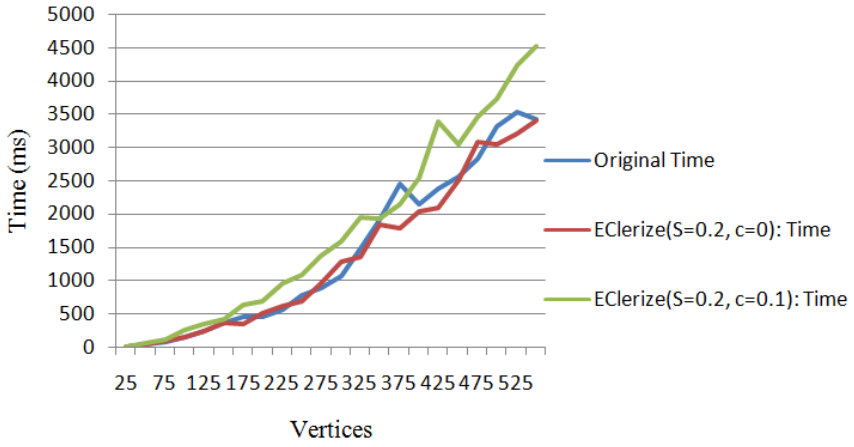


Figure 4.8: Graph size vs execution time. Blue line, red line and yellow line shows the execution time(ms) of the Kamada-Kawaii Layout Algorithm, EClerize with parameters $S = 0.2$, $c = 0$ and EClerize with parameters $S = 0.2$, $c = 0.1$ respectively.

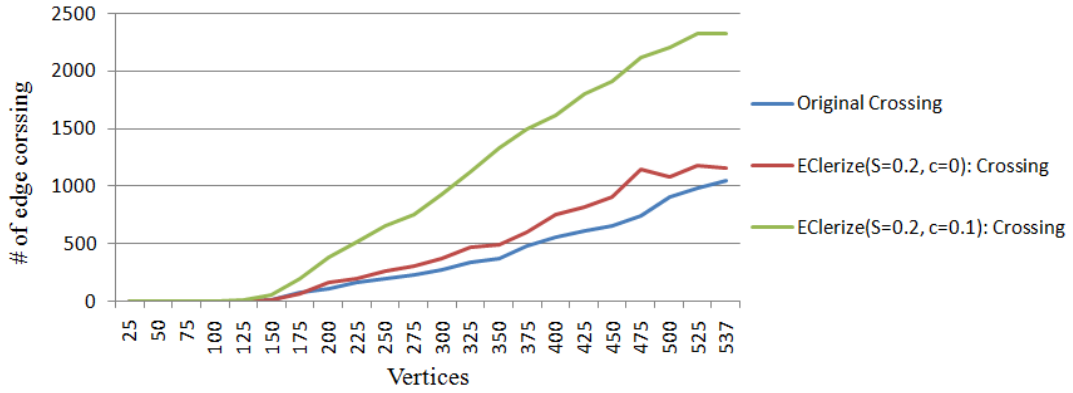


Figure 4.9: Graph size vs number of edge crossing. Blue line, red line and yellow line shows the number of edge crossing of the Kamada-Kawaii Layout Algorithm, EClerize with parameters $S = 0.2$, $c = 0$ and EClerize with parameters $S = 0.2$, $c = 0.1$ respectively.

Photo Transduction dataset. By looking these results, we can validate inter-cluster distance is maximized as c is increasing.

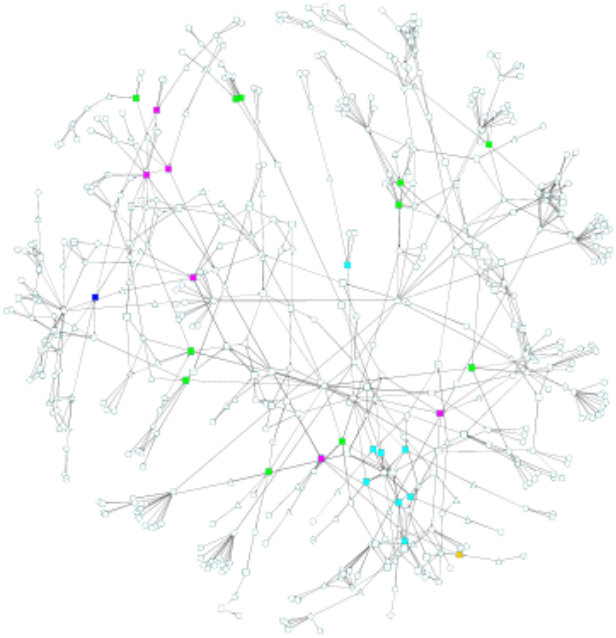


Figure 4.10: Layout of EClerize with parameters $c = 0$ and $S = 0$ on Visual Photo Transduction dataset.

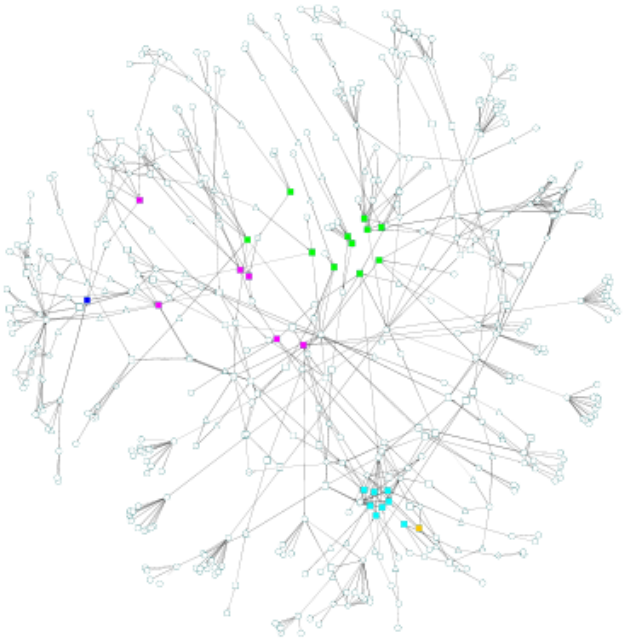


Figure 4.11: Layout of EClerize with parameters $c = 0$ and $S = 0.1$ on Visual Photo Transduction dataset.

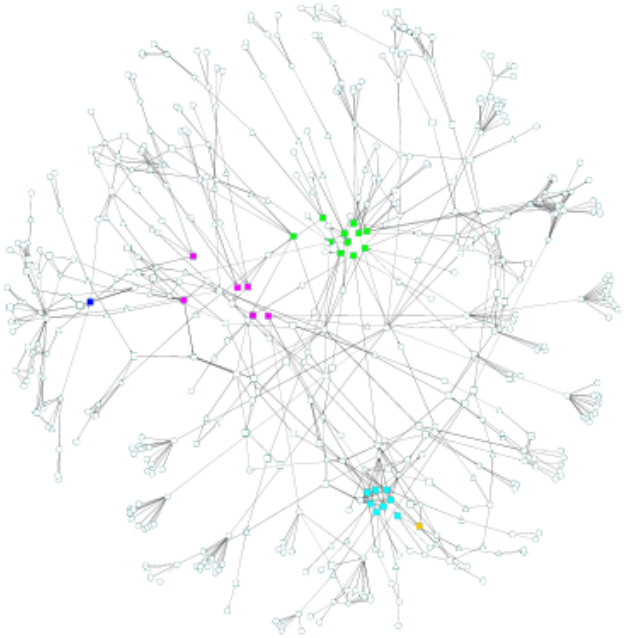


Figure 4.12: Layout of EClerize with parameters $c = 0$ and $S = 1$ on Visual Photo Transduction dataset.

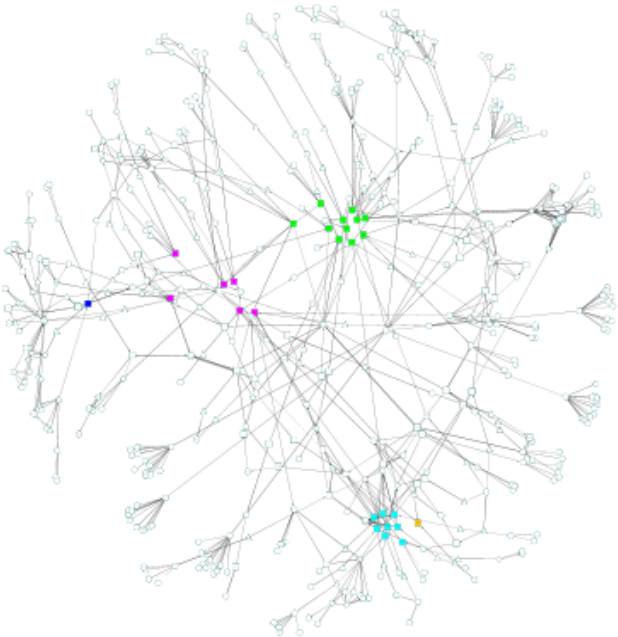


Figure 4.13: Layout of EClerize with parameters $c = 0.1$ and $S = 1$ on Visual Photo Transduction dataset.

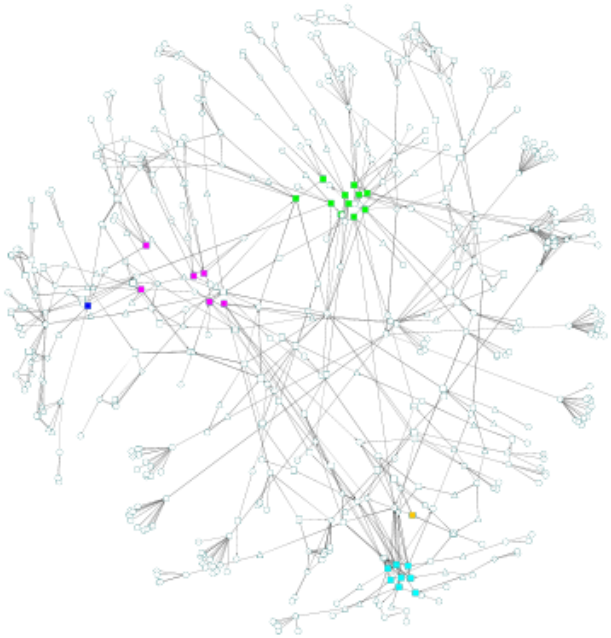


Figure 4.14: Layout of EClerize with parameters $c = 0.4$ and $S = 1$ on Visual Photo Transduction dataset.

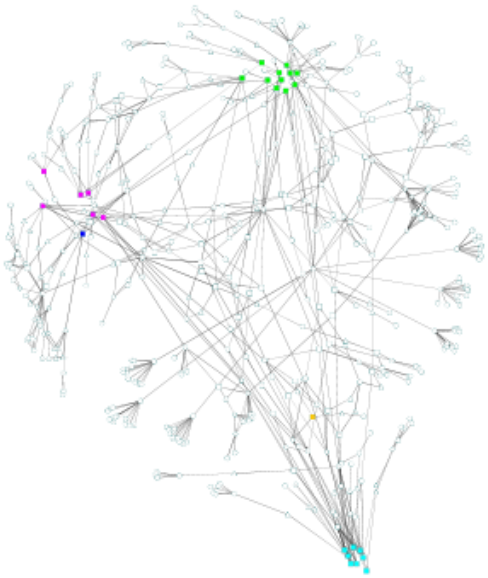


Figure 4.15: Layout of EClerize with parameters $c = 1$ and $S = 1$ on Visual Photo Transduction dataset.

CHAPTER 5

CONCLUSION

5.1 Summary

Proteins are complex molecules that play vital role in living organisms and enzymes are special kinds of proteins that speed up chemical reactions. Enzymes are classified according to what they catalyze and a numerical classification number (EC number) is assigned to each enzyme. Most of the biological graphs contain vertices that represent enzyme structure. It is crucial to understand an enzyme's mechanism in the pathway, which is a form of graph, for researchers who work on drug target discovery and diagnostic tests. At this point, an enhanced data visualisation is needed.

Force-directed layout algorithm is very popular among the graph layout algorithms and it could be used to draw biological graphs. Simulating graphs like physical systems is the perspective of force-directed layout algorithms. However, modification is required when we would like to embed domain-specific knowledge into layout, since the force-directed layout algorithm employs only the graph structure. In this study, a modified and improved Kamada-Kawaii force-directed layout algorithm, EClerize, is proposed to generate more readable layouts for biological graphs that contain enzymes.

EClerize creates clusters from enzymes having the same EC class. Virtual edges are added to graph to create clusters. The equation in the original algorithm that assigns the strengths of spring is redefined to include biological similarity between enzymes. EClerize depends on two parameters: S and c. First parameter S is the spring constant for vertices that have EC number. As parameter S increases, intra-

cluster distances gets minimized. The second parameter, c is a constant for distance factor. c directly affects inter-cluster distances. Selection of these parameters is also discussed. According to results of the experiments, it is advised to use $0.1 < S < 0.4$ and $c < 0.4$. Finally, EClerize is tested on different datasets using the suggested parameters. Effects of EClerize on execution time, graph area and number of edge crossing are discussed. EClerize gives good results in terms of visual aesthetic and generates enhanced layouts for biological graphs that contain enzymes.

5.2 Improvements

Creating clusters in graphs could be accomplished by inserting virtual vertices. An edge is added between the virtual vertex and each vertex in the clusters. Thus, the vertices within a cluster will be closer to each other than before. However, this idea is effective on small graphs. Virtual force that bounds vertices in the same cluster loses its effect as the graph gets larger.

In the literature, there are studies that define new forces to create clusters, such as attractive force for the vertices in the same cluster and repulsive force for different clusters. Clusters are given manually and it reduces usability. Methods are implemented that navigate clusters in the graph as an animated interactive system. So, users can change structures of graphs at runtime. However, all of the solutions based on that a vertex must be part of a cluster.

GOlorize [35] enhances visualization of the graph by taking Gene Ontology annotation of vertices into consideration. Biological similarities affect positions in final layout. But, there is no way to give another domain-specific knowledge like EC numbers. Other layout algorithms that addressed biological graphs also do not support for EC numbers [46, 3].

EClerize allows unclustered vertices since most of the vertices in biological graphs do not have EC Number attribute. The inter-cluster and the intra-cluster distance defined as parametric. The effects of the parameters is examined in terms of quality of the layout. As a result of this, the optimal interval for parameters is determined. A heuristic method is used to improve convergence while increasing the inter-cluster

distances. Finally, positions of vertices in the final layout is affected by two factors: biological similarity between enzymes and theoretical length between the vertices.

5.3 Future Work

EClerize, like most of the force-directed algorithms, starts with random initial positions. It means the algorithm might end with different local minima at each time. This affects metrics that measure quality of the layout. Initial points can be determined to increase quality [16, 17].

EClerize is based on the simulation of physical laws. Vertices behave like same charged particles in EClerize. They repel each other away. Repelling forces are calculated after each of iteration. Complexity of the algorithm depends on the cube of number of vertices in the graph. Barnes and Hut proposed a fast approximation algorithm for dynamical system of particles under the influence of physical forces [19]. In the Barnes-Hut method, particles that are close enough to each other group into one particle. Then, when the group is sufficiently far away, we could approximate its physical effects by using its center of mass. Complexity of Barnes-Hut is $O(NlogN)$, where N is the number of particles. The Barnes-Hut algorithm could be used in the calculation of the force interaction between vertices to reduce complexity of EClerize [20, 21, 22].

Graphical Processing Units (GPU) are becoming powerful and affordable day after day. EClerize algorithm can be implemented in GPU environment [22, 26, 25]. Most of real world applications of graph visualization tools support labeling of vertices and edges. It could be applied to enhance readability of graphs [14, 15].

Although force-directed algorithms give good results for small to medium size graphs, multi-scale solutions could be used for large graphs [21, 23, 24].

REFERENCES

[1] D. Harel and Y. Koren, "Drawing graphs with non-uniform vertices," in *Proceedings of the Working Conference on Advanced Visual Interfaces*, pp. 157–166, ACM, 2002.

[2] U. Dogrusoz, M. E. Belviranli, and A. Dilek, "Cise: a circular spring embedder layout algorithm," *Visualization and Computer Graphics, IEEE Transactions on*, vol. 19, no. 6, pp. 953–966, 2013.

[3] M. Y. Becker and I. Rojas, "A graph layout algorithm for drawing metabolic pathways," *Bioinformatics*, vol. 17, no. 5, pp. 461–467, 2001.

[4] T. M. Fruchterman and E. M. Reingold, "Graph drawing by force-directed placement," *Softw., Pract. Exper.*, vol. 21, no. 11, pp. 1129–1164, 1991.

[5] T. Kamada and S. Kawai, "An algorithm for drawing general undirected graphs," *Information processing letters*, vol. 31, no. 1, pp. 7–15, 1989.

[6] G. Di Battista, P. Eades, R. Tamassia, and I. G. Tollis, "Algorithms for drawing graphs: an annotated bibliography," *Computational Geometry*, vol. 4, no. 5, pp. 235–282, 1994.

[7] D. Dubé, *Graph layout for domain-specific modeling*. PhD thesis, McGill University, Montréal, Canada, 2006.

[8] C. Bennett, J. Ryall, L. Spalteholz, and A. Gooch, "The aesthetics of graph visualization.," in *Computational Aesthetics*, pp. 57–64, Citeseer, 2007.

[9] "Signaling by pdgf." `http://www.reactome.org/content/detail/REACT_16888`. Accessed: 2015-11-23.

[10] P. Eades, "A heuristics for graph drawing," *Congressus numerantium*, vol. 42, pp. 146–160, 1984.

[11] R. Davidson and D. Harel, "Drawing graphs nicely using simulated annealing," *ACM Transactions on Graphics (TOG)*, vol. 15, no. 4, pp. 301–331, 1996.

[12] S. C. Chapra and R. P. Canale, *Numerical methods for engineers*, vol. 2. McGraw-Hill, 2012.

[13] M. Halkidi, Y. Batistakis, and M. Vazirgiannis, "Clustering validity checking methods: part ii," *ACM Sigmod Record*, vol. 31, no. 3, pp. 19–27, 2002.

[14] E. R. Gansner and S. C. North, "Improved force-directed layouts," in *Graph Drawing*, pp. 364–373, Springer, 1998.

[15] R. Klapaukh, D. J. Pearce, and S. Marshall, "Towards a vertex and edge label aware force directed layout algorithm," in *Proceedings of the Thirty-Seventh Australasian Computer Science Conference-Volume 147*, pp. 29–37, Australian Computer Society, Inc., 2014.

[16] W. Dong, F. Wang, Y. Huang, G. Xu, Z. Guo, X. Fu, and K. Fu, "An advanced pre-positioning method for the force-directed graph visualization based on pagerank algorithm," *Computers & Graphics*, vol. 47, pp. 24–33, 2015.

[17] C. Walshaw, "A multilevel algorithm for force-directed graph drawing," in *Graph Drawing*, pp. 171–182, Springer, 2001.

[18] A. Frick, A. Ludwig, and H. Mehldau, "A fast adaptive layout algorithm for undirected graphs (extended abstract and system demonstration)," in *Graph Drawing*, pp. 388–403, Springer, 1995.

[19] J. Barnes and P. Hut, "A hierarchical o (n log n) force-calculation algorithm," 1986.

[20] A. Quigley and P. Eades, "Fade: Graph drawing, clustering, and visual abstraction," in *Graph Drawing*, pp. 197–210, Springer, 2001.

[21] Y. Hu, "Efficient, high-quality force-directed graph drawing," *Mathematica Journal*, vol. 10, no. 1, pp. 37–71, 2005.

[22] A. Godiyal, J. Hoberock, M. Garland, and J. C. Hart, "Rapid multipole graph drawing on the gpu," in *Graph Drawing*, pp. 90–101, Springer, 2009.

[23] D. Harel and Y. Koren, "A fast multi-scale method for drawing large graphs," in *Graph drawing*, pp. 183–196, Springer, 2001.

[24] P. Gajer, M. T. Goodrich, and S. G. Kobourov, "A multi-dimensional approach to force-directed layouts of large graphs," in *Graph Drawing*, pp. 211–221, Springer, 2001.

[25] Y. Frishman and A. Tal, "Online dynamic graph drawing," *Visualization and Computer Graphics, IEEE Transactions on*, vol. 14, no. 4, pp. 727–740, 2008.

[26] M. Mrzek and B. J. Blazic, "Fast network communities visualization on massively parallel gpu architecture," in *Information & Communication Technology Electronics & Microelectronics (MIPRO), 2013 36th International Convention on*, pp. 269–274, IEEE, 2013.

[27] P. Eades and Q.-W. Feng, "Multilevel visualization of clustered graphs," in *Graph drawing*, pp. 101–112, Springer, 1997.

[28] V. Batagelj, F. J. Brandenburg, W. Didimo, G. Liotta, P. Palladino, and M. Patrignani, "Visual analysis of large graphs using (x, y)-clustering and hybrid visualizations," *Visualization and Computer Graphics, IEEE Transactions on*, vol. 17, no. 11, pp. 1587–1598, 2011.

[29] F. Bertault and M. Miller, "An algorithm for drawing compound graphs," in *Graph Drawing*, pp. 197–204, Springer, 1999.

[30] J. Hua, M. L. Huang, and Q. V. Nguyen, "Drawing large weighted graphs using clustered force-directed algorithm," in *Information Visualisation (IV), 2014 18th International Conference on*, pp. 13–17, IEEE, 2014.

[31] X. Wang and I. Miyamoto, "Generating customized layouts," in *Graph Drawing*, pp. 504–515, Springer, 1996.

[32] Y. Frishman and A. Tal, "Dynamic drawing of clustered graphs," in *Information Visualization, 2004. INFOVIS 2004. IEEE Symposium on*, pp. 191–198, IEEE, 2004.

[33] M. Kaufmann and D. Wagner, *Drawing graphs: methods and models*, vol. 2025. Springer, 2003.

[34] M. L. Huang and P. Eades, "A fully animated interactive system for clustering and navigating huge graphs," in *Graph Drawing*, pp. 374–383, Springer, 1998.

[35] O. Garcia, C. Saveanu, M. Cline, M. Fromont-Racine, A. Jacquier, B. Schwikowski, and T. Aittokallio, "Golorize: a cytoscape plug-in for network visualization with gene ontology-based layout and coloring," *Bioinformatics*, vol. 23, no. 3, pp. 394–396, 2007.

[36] "uniprot." `http://www.uniprot.org/help/gene_ontology`. Accessed: 2015-10-31.

[37] L. S. Jeremy M. Berg, John L. Tymoczko, *Biochemistry*. No. Ed. 5, W. H. Freeman, 2002.

[38] "Drug strategies to target hiv: Enzyme kinetics and enzyme inhibitors." `http://www.chemistry.wustl.edu/~edudev/LabTutorials/HIV/DrugStrategies.html`. Accessed: 2015-11-04.

[39] A. Radzicka and R. Wolfenden, "A proficient enzyme," *Science*, vol. 267, no. 5194, pp. 90–93, 1995.

[40] "Pyruvate metabolism - reference pathway." `http://www.genome.jp/kegg/pathway/map/map00620.html`. Accessed: 2015-11-16.

[41] E. O. Voit, *A first course in systems biology*. Garland Science, 2012.

[42] M. Kanehisa and S. Goto, "Kegg: kyoto encyclopedia of genes and genomes," *Nucleic acids research*, vol. 28, no. 1, pp. 27–30, 2000.

[43] A. M. Edelman, D. K. Blumenthal, and E. G. Krebs, "Protein serine/threonine kinases," *Annual review of biochemistry*, vol. 56, no. 1, pp. 567–613, 1987.

[44] P. Shannon, A. Markiel, O. Ozier, N. S. Baliga, J. T. Wang, D. Ramage, N. Amin, B. Schwikowski, and T. Ideker, "Cytoscape: a software environment for integrated models of biomolecular interaction networks," *Genome research*, vol. 13, no. 11, pp. 2498–2504, 2003.

[45] G. Joshi-Tope, M. Gillespie, I. Vastrik, P. D'Eustachio, E. Schmidt, B. de Bono, B. Jassal, G. Gopinath, G. Wu, L. Matthews, *et al.*, "Reactome: a knowledgebase of biological pathways," *Nucleic acids research*, vol. 33, no. suppl 1, pp. D428–D432, 2005.

[46] B. Genç and U. Dogrusoz, "A layout algorithm for signaling pathways," *Information Sciences*, vol. 176, no. 2, pp. 135–149, 2006.