

SOFTWARE TOOLS FOR VISUAL ANALYSIS OF CANCER GENOMICS DATA IN THE CONTEXT OF PATHWAYS

A THESIS SUBMITTED TO
THE GRADUATE SCHOOL OF ENGINEERING AND SCIENCE
OF BILKENT UNIVERSITY
IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR
THE DEGREE OF
MASTER OF SCIENCE
IN
COMPUTER ENGINEERING

By
İstemi Rahman Bahçeci
August 2016

SOFTWARE TOOLS FOR VISUAL ANALYSIS OF CANCER
GENOMICS DATA IN THE CONTEXT OF PATHWAYS

By İstemi Rahman Bahçeci

August 2016

We certify that we have read this thesis and that in our opinion it is fully adequate,
in scope and in quality, as a thesis for the degree of Master of Science.

Uğur Doğrusöz(Advisor)

Ercüment Çiçek

Tolga Can

Approved for the Graduate School of Engineering and Science:

Levent Onural
Director of the Graduate School

ABSTRACT

SOFTWARE TOOLS FOR VISUAL ANALYSIS OF CANCER GENOMICS DATA IN THE CONTEXT OF PATHWAYS

İstemi Rahman Bahçeci

M.S. in Computer Engineering

Advisor: Uğur Doğrusöz

August 2016

Information visualization is concerned with effective visual presentation of abstract information, which reinforces human cognition. Graphs are structures that are well suited to represent relational information. Graph visualization is vital since the underlying relational information of the graph provides fine analysis and comprehension opportunities. Biological pathway visualization is one of the most popular areas, where graph visualization is highly favored.

Interactive analysis and visualization of cancer related pathways in the context of genomic data, such as those available through the TCGA project, might reveal valuable information for scientists about disease conditions and potential causes. As the size and complexity of such cancer pathways and associated genomic data increase, exchangeable in-silico representations and their effective, enhanced visualizations, and complexity management become crucial for effective analysis of such data to potentially discover cause-effect relations.

In this thesis, we designed and implemented software solutions to visualize cancer genomics data in the context of networks from simple gene interaction networks to process description diagrams within the cBioPortal for Cancer Genomics (cBioPortal). cBioPortal is a popular web portal, getting about 60.000 visits globally per month, providing visualization, analysis and download of large-scale cancer genomics data sets. The network view in cBioPortal presents neighborhood of genes of interest. The alteration data is overlaid on the network with numerous ways to filter and manage complexity of the network (e.g. by alteration percentage or by type or source of the interactions). Upon demand, the user can obtain a more detailed, mechanistic view of the interactions among gene

pairs, from Pathway Commons database with a live query using the SBGN process description notation. Finally, we also developed a new pathway visualization component, specifically for cancer pathways, using a uniform notation found in TCGA cancer publications. This tool also facilitates curation of pathways from scratch with support for collaborative editing.



Keywords: Bioinformatics, Biology, Pathway Visualization, Cancer Genomics Research, Complexity Management, Visualization Software, Web-based Software.

ÖZET

KANŞER GENOMİK BİLGİSİNİN YOLAKLAR DAHİLİNDE GÖRSEL ANALİZİNİ SAĞLAYAN YAZILIM ARAÇLARI

İstemi Rahman Bahçeci

Bilgisayar Mühendisliği, Yüksek Lisans

Tez Danışmanı: Uğur Doğrusöz

Ağustos 2016

Veri Görselleme soyut olarak temsil edilmiş bilginin, insan algısını destekleyecek, etkili bir biçimde görsel olarak temsil edilmesini hedefler. Çizgeler ilişkisel bilginin temsil edilmesi için tercih edilen uygun yapılardır. Bir çizgenin görsellenmesi çizgenin yapısındaki ilişkisel bilginin daha iyi anlaşılması ve çözümlenmesi bakımından fayda sağladığı için önemlidir. Biyolojik yolak görselleme çizge görsellemenin yaygın olarak kullanıldığı bir alandır.

Kanser ile ilişkili yolların ilgili kanser genomik bilgisi dahilinde interaktif bir biçimde çözümleme ve görsellenmesi, örn: TCGA projesi dahilinde erişilebilen yollar, araştırmacılar için hastalıkların olası nedenlerini anlamak, neden sonuç ilişkilerini gözlemek adına değerli bilgiler sunar. Bu yolların boyutu ve karmaşıklığı ve de ilgili genomik datanın boyutu arttıkça, daha etkili karmaşıklık yönetim teknikleri, etkili ve gelişmiş çizge görsellemeleri, bilgisayar ortamında kolay olarak paylaşımına olanak sağlayan bilgi temsilleri araştırmacıların neden sonuç ilişkilerini daha iyi çözümlemesine imkan sağlar.

Bu tez çalışması kapsamında genomik datanın yollar dahilinde görsellenmesini sağlayan yazılım bileşenleri tasarlanmış ve gerçekleştirilmiştir. Bu yazılım bileşenleri özel olarak 'cBioPortal for Cancer Genomics' (cBioPortal) isimli, kanser araştırmalarında yaygın olarak kullanılan web tabanlı bir yazılım dahilinde genlerin birbirleriyle ilişkilerini gösteren düzenleyici yolları ve moleküler seviyede prosesleri görselleyebilmek için hayata geçirilmiştir. cBioPortal dünya çapında aylık olarak yaklaşık 60.000 kişi tarafından ziyaret edilen, büyük çapta kanser genomik bilgisinin görsellenmesi, çözümlenmesi ve indirilmesi hizmetleri veren web tabanlı bir yazılımdır. cBioPortal dahilinde geliştirilen çizge görselleme

bileşeni, verilen bir gen listesinin , bir kanser tipi dahilinde etkileştiği diğer genleri içeren komşuluk yollarını kanser genomik bilgisiyle beraber etkili bir biçimde görselleyebilmektedir. Buna ek olarak çeşitli filtreleme ve karmaşıklık yönetimi olanakları da (örn: genomik alterasyona göre çizge köşelerini filtreleme, gen etkileşimlerini alındığı veri tabanına göre filtreleme.) bu yazılım bileşeni dahilinde hayata geçirilmiştir.

Istenildiği takdirde gen etkileşimleri daha detaylı bir şekilde, moleküler seviyede prosesler dahilinde Systems Biology Graphical Notation (SBGN) diye adlandırılan bir notasyonla görsellenebilmektedir. Bu detaylı proses seviyesinde olan yolak bilgisi Pathway Commons isimli yolak veritabanından bir sorguyla, SBGN proses dili diye adlandırılan bir dilde alınmaktadır.

Son olarak, özellikle TCGA yayınlarında sıkça rastlanan ve yaygın olarak kullanılan bir notasyonda yolların görsellenmesi ve oluşturulması amacıyla yeni bir yolak görselleme aracı tasarlanmış ve geliştirilmiştir. Bu yazılım aracı standard bir çizge görselleme ve oluşturma aracından farklı olarak, ortak olarak bir TCGA yolağının araştırmacılarca oluşturulmasına olanak sağlamaktadır.

Anahtar sözcükler: Biyoenformatik, Biyoloji, Yolak Görselleme, Kanser Genomik Araştırmaları, Karmaşıklık Yönetimi, Çizge Görselleme Yazılımı , Web Tabanlı Yazılım.

Acknowledgement

I would like to express my deepest gratitude to my supervisor Prof. Uğur Doğrusöz for his guidance, support and all the things he has done for me. I can honestly say that it was a valuable experience for me since I learned too many things from Prof. Doğrusöz about computer science and more importantly life.

I would like to thank to Assist. Prof. Dr. Ercüment Çiçek and Assoc. Prof. Dr. Tolga Can for reviewing and commenting on the manuscript of this thesis.

I would like to start my long list of thanks to the people that I seize the chance of knowing them during my graduate study in Bilkent University.

I would like to thank to Nail Akıncı, Necmi Acarsoy, Seher Acer, Umut Ağıl, Shatlyk Ashyralyyev, Ebru Ateş, Aytek Aman, Sermetcan Baysal, Çağdaş Cengiz, Merve Çakır, Gülfem Demir, Elif Eser, Can Fırtına, Begüm Genç, Eren Gölge, Emir Gülümser, Gülce Kale, Can Fahrettin Koyuncu, Caner Mercan, Muhsin Can Orhan, Mecit Sari, Metincan Siper, Ali Burak Ünal, Başak Ünsal and Bora Yalçın for their precious support and friendship. Words are not strong enough to express the value that they have brought to my life. It was a great pleasure and joy knowing all of them.

I would like to indicate my special thanks to *Simge Yücel* who is one of the most valuable people that I met in my entire life. It was a unique privilege to know her. I am grateful for her valuable support and everything she has done for me during the hardest times of my study.

My parents, Nefise and Hüseyin, and my brother İlteriş, I know this thesis and master's study will not be complete without your support and your faith in me.

Last but not least, I would like to indicate my greatest gratitude, respect and admiration to the founder of the Republic of Turkey, Mustafa Kemal Atatürk. He shed a light to path of mind, science and art by his great vision. I will always

be following the path he led.



Contents

1	Introduction	1
1.1	Motivation	3
1.2	Contribution	4
2	Background Information and Related Work	6
2.1	Graph Visualization	6
2.2	Biological Standards	11
2.2.1	BioPAX (Biological Pathways Exchange)	11
2.2.2	SBGN (Systems Biology Graphical Notation)	12
2.3	Related Tools, Libraries, and Resources	18
2.3.1	Cytoscape	18
2.3.2	Paxtools	22
2.3.3	Pathway Commons	22
2.3.4	ChiLay	24

2.3.5	TCGA	25
2.3.6	cBioPortal	25
2.3.7	BioGene	27
2.3.8	GraphML	27
2.3.9	Google Real Time API	27
2.3.10	SBGNViz.js	28
3	Network Visualization Component in cBioPortal	30
3.1	Visualization of Gene Interaction Pathways with Genomic Data in cBioPortal	31
3.2	Visualization of Process Level Details of Interactions in cBioPortal	39
3.3	Complexity Management Operations	44
3.3.1	Complexity Management Operations for Gene Interaction Pathways	44
3.3.2	Operations for Process Level Pathways	52
3.4	Architecture	58
4	TCGA Pathway Curation Tool	62
4.1	Visualization of TCGA Pathways	64
4.2	Architecture	69
4.3	Collaborative Usage	72

5 Conclusion 76

5.1 Future Work 77

5.2 Availability 78



List of Figures

1.1	Textual representation of a biological pathway (recruitment of repair and signaling proteins to double-strand breaks) in Pathway-Commons [1]	2
1.2	Visual representation of a biological pathway (recruitment of repair and signaling proteins to double-strand breaks) in Pathway-Commons [1]	3
2.1	A sample <i>directed (left)</i> and <i>undirected (right)</i> graph with four vertices: A, B, C, D and four edges that connects vertices to each other.	7
2.2	A sample <i>compound-graph</i> with hierarchically nested vertices. . .	8
2.3	Force-directed graph layout model [2].	10
2.4	A graph on the left before applying layout. Same graph after applying force-directed graph layout.	11
2.5	BioPAX provides a standard language for the efficient exchange of the pathway data [3].	12
2.6	Metabolical pathway of <i>Glycolysis</i> visualized in PD.	15

2.7	A sample metabolic process drawn in SBGN (left) and its associated representation in SBGN-ML (right)	16
2.8	Representation of SBGN PD language elements [4]	17
2.9	An example of <i>Cytoscape.js</i> usage.	20
2.10	A sample node removal of a compound node by querying node with id ‘n7’ and removing it from the graph that exemplifies manipulation and query features of <i>Cytoscape.js</i> . This graph is also visualized using <i>Cytoscape.js</i>	21
2.11	A sample graph query to the web service of Pathway Commons. .	24
2.12	cBioPortal web application home page.	26
3.1	Overview of visualization of gene interaction pathways with genomic data in cBioPortal	32
3.2	Overlay of genomic data on a node. Disks around the node represent genomic profiles in the context of copy number, mRNA expression and Mutation. The node is colored according to the alteration frequency using a white to red color gradient	33
3.3	Visualization of drugs targeting a gene in cBioPortal	34
3.4	Color coding of interactions in cBioPortal network visualization component	36
3.5	Genes tab	37
3.6	Detailed drug information of <i>RB1</i> gene	38
3.7	Process level interaction details of the interaction between <i>MDM2</i> and <i>RAD17</i> genes	40

3.8	Flow of events regarding visualization of process level interaction details	42
3.9	Detailed gene information of <i>MDM2</i> gene in SBGN view	43
3.10	Topology related filtering operations	45
3.11	Filtering operations in ‘Genes’ tab of network visualization tool in cBioPortal	46
3.12	Filtering out neighbours of seed genes by alteration percent	47
3.12	Filtering out neighbours of seed genes by alteration percent (cont’d)	48
3.13	Highlight operation that focuses on neighborhood of <i>CDK4</i> gene.	49
3.14	Filtering views by interaction type and source.	50
3.15	Interaction filtering by type.	51
3.16	Expand collapse operations	53
3.16	Expand collapse operations (cont’d)	54
3.17	Hide Operation.	56
3.18	Highlight Operation.	57
3.19	High level architectural design of network visualization component in cBioPortal.	59
4.1	Visualization of PI3KRAS signalling pathway in the context of high-grade serous ovarian cancer (HGS-OvCa) which is provided as part of a TCGA research article [5].	62

4.2	Pathway in Figure 4.1 curated and visualized by our tool along with the genomic data representation.	63
4.3	Visual representation of node types in TCGA curation tool	64
4.4	Visual representation of interaction types in TCGA curation tool	65
4.5	Genomic data overlay features of TCGA pathway curation tool . .	66
4.6	Representations of graph topology and genomic data that the tool accepts in simple tab delimited format.	67
4.7	Different context menus of the TCGA pathway curation tool. From left to right: edge context menu, node context menu and root context menu is visualized. Note that root context menu appears when a right click is performed on the canvas where no node or edge exists under mouse pointer.	68
4.8	High Level Architecture of TCGA Pathway Curation Tool	70
4.9	Flow of events regarding addition of new elements in collaborative usage mode	73
4.10	Collaborative initialization of TCGA pathway curation tool . . .	75

Chapter 1

Introduction

Information visualization is concerned with effective visual presentation of abstract information, which reinforces human cognition by exploiting the advanced processing power of human visual perception system. It aims to provide intuitive representations of abstract information allowing people to see, explore and comprehend large amount of information at one sight [6].

Graphs can be regarded as one of the most popular techniques in the field of information visualization. Graphs are abstract structures that are well suited to represent relational information. A graph represents some set of objects as *nodes* and relations among those objects as *edges* connecting nodes of the graph to each other.

Graph visualization is vital since the underlying relational information of the graph provides fine analysis and comprehension opportunities compared to the textual representation of such data (Figure 1.1 and Figure 1.2). Biological pathway visualization is one of the most popular areas, where graph visualization is highly favored for efficient utilization and intuitive representation of valuable information that is embodied in pathways (Figure 1.2).

Pathway: Recruitment of repair and signaling proteins to double-strand breaks

Authored: Matthews, L, 2003-11-18 00:00:00 [more...](#)

[Biochemical Reactions \(4\)](#) [Sub-Pathways \(1\)](#) [Molecules \(19\)](#)

Showing 1-4 of 4

1. [TP53B_HUMAN](#) + [gamma H2AX:MDC1/NFBD1 compl...](#) → [53BP1:H2AX complex at site ...](#) [Reactome](#)

At the beginning of this reaction, 1 molecule of '53BP1', and 1 molecule of 'gamma H2AX:MDC1/NFBD1 complex at site of DNA double-strand break' are present. At the end of this reaction, 1 molecule of '53BP1:H2AX complex at site of double-strand break' is present.

This reaction takes place in the 'nucleus'.

References:

- DNA damage activates ATM through intermolecular autophosphorylation and dimer dissociation. Nature 421:499-506 [[PubMed](#)]

2. [BRCA1_HUMAN](#) + [53BP1:H2AX complex at site ...](#) → [BRCA1:53BP1 complex at site...](#) [Reactome](#)

- [ACTIVATED](#) by [MDC1_HUMAN](#)

At the beginning of this reaction, 1 molecule of '53BP1:H2AX complex at site of double-strand break', and 1 molecule of 'Breast cancer type 1 susceptibility protein' are present. At the end of this reaction, 1 molecule of 'BRCA1:53BP1 complex at site of DNA double-strand break' is present.

This reaction takes place in the 'nucleus'.

References:

- Molecular views of recombination proteins and their control. Nat Rev Mol Cell Biol 4:435-45 [[PubMed](#)]
- A DNA damage-regulated BRCT-containing protein, TopBP1, is required for cell survival. Molecular and cellular biology 22(2):555-66 [[PubMed](#)]
- 53BP1, a mediator of the DNA damage checkpoint. Science 298:1435-8 [[PubMed](#)]

3. [ATM_HUMAN](#) (phosphorylated ¹⁹⁸¹) + [MDC1/NFBD1:gamma-H2AX complex](#) → [ATM associated with DNA dou...](#) [Reactome](#)

4. [MDC1_HUMAN](#) + [gamma H2AX-coated DNA doubl...](#) → [gamma H2AX:MDC1/NFBD1 compl...](#) [Reactome](#)

- [ACTIVATED](#) by [H2AX_HUMAN](#)

Figure 1.1: Textual representation of a biological pathway (recruitment of repair and signaling proteins to double-strand breaks) in PathwayCommons [1]

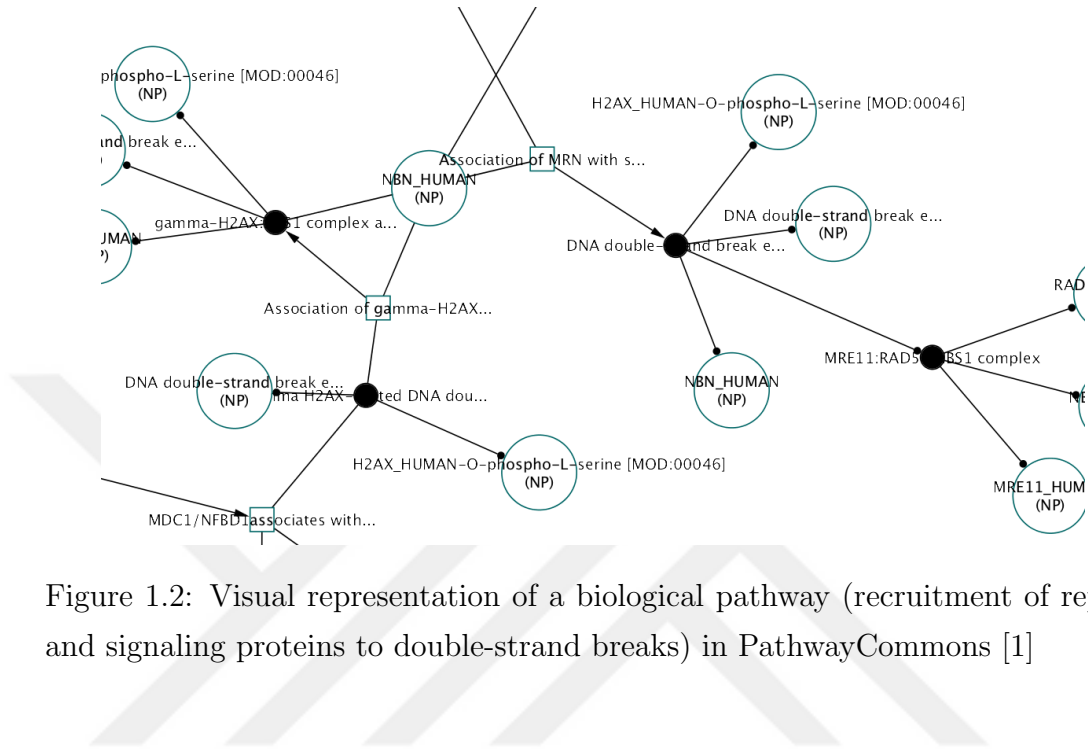


Figure 1.2: Visual representation of a biological pathway (recruitment of repair and signaling proteins to double-strand breaks) in PathwayCommons [1]

1.1 Motivation

A pathway represents series of molecular interactions in a cell that may lead to a certain product or a significant change in a cell. A pathway can trigger construction of new molecules such as protein, fat in a cell. Moreover, a pathway might play an important role in the context of the activation or deactivation of genes, signalling a cell to translocate.

It can be stated that pathway information captures valuable knowledge about molecular level biological processes. In this respect, pathways are vital in the context of genomic studies since they play important role about interpretation of increasing amount of biological data.

Interactive analysis and visualization of cancer related pathways in the context of genomic data, such as those available through the TCGA project [7], might reveal valuable information for the scientists about disease conditions and potential causes. As the size and complexity of such cancer pathways and associated

genomic data increase, exchangeable in-silico representations and their effective, enhanced visualizations, and complexity management become crucial for effective analysis of such data to potentially discover cause-effect relations.

1.2 Contribution

To aid the research efforts in cancer research, we designed and developed software components to visualize biological pathways from simple gene to gene interaction networks to detailed molecular process level networks along with the cancer genomics information.

Firstly, we provided a network visualization and analysis component for the popular cBioPortal web portal [8, 9]. cBioPortal facilitates analysis, visualization and download of large scale cancer genomics data sets. To enhance such features, a new visualization component is provided, which aims to visualize gene to gene interaction networks. Such networks represent neighborhood pathway of given set of genes of interest in the context of a specific cancer study. This component also features representation of genomic alteration data along with the useful features of filtering and managing the complexity in such gene to gene interaction networks.

Secondly in order to support information provided by visualized neighborhood pathways, we provided a complementary view that is capable of visualizing molecular process level details of an interaction between two genes from such visualized gene interaction pathways. Detailed process level views of interactions are visualized in Systems Biology Graphical Notation Process Description language (SBGN-PD) [10]. To visualize such SBGN-PD pathways, we integrated our previously implemented SBGN-PD visualization component SBGNViz.js [11] to cBioPortal. We extended SBGNViz.js' existing complexity management operations with operations of expand and collapse. Expand and collapse operations aims to reduce the complexity that is brought by the molecular complex structures and cellular compartments that frequently occur in SBGN-PD pathways.

Complex structures and cellular compartments are hierarchical compound structures that can include other entities in them; therefore, they may increase the size of the visualized graph enormously which makes it hard to focus on content of interest and decreases utilization of the information embodied in such pathways. Associated pathway information of an interaction of interest is obtained from Pathway Commons by a live query which returns the pathway in SBGN-ML format [12].

Pathway Commons [1] converts queried pathway information from BioPAX [3] format to SBGN-ML. To achieve such conversion popular Paxtools library is used in Pathway Commons. In the context of our work, we provided automated graph layout facility to the resulting SBGN pathways that Pathway Commons provides. Virtual representations of SBGN elements after the conversion from BioPAX models do not include any geometric information regarding geometric boundaries and location of SBGN elements, in the obtained pathways. This is an important aspect considering the efficient visualization of SBGN pathways. In this respect, we contributed to the Paxtools [13] library in a way that geometric boundaries of SBGN elements are estimated according to content of such elements. Regarding location information, we integrated CoSE [14] algorithm to Paxtools for automated graph layout of such SBGN pathways.

Finally, we also designed and developed a new pathway visualization component, specifically for cancer pathways, using a uniform notation found in TCGA cancer publications. This tool provides visualization and interactive editing of TCGA pathways according to a notation that respects the conventional representation of such TCGA pathways. The tool also lets users to import experimental cancer genomics data that can be overlaid on the visualized pathways. Besides visualization and interactive creation capabilities, this tool facilitates curation of such pathways in a collaborative manner using the Google Real Time technology [15]. This technology is behind popular collaborative document editing tool Google Docs.

Chapter 2

Background Information and Related Work

In this section related work and background information are discussed.

2.1 Graph Visualization

A graph $G = (V, E)$ is an abstract structure which represents a non-empty set of vertices (nodes) V and a set of edges E where each edge $e \in E$ connects a vertex pair (u, v) where $u, v \in V$.

A *directed graph* is a graph $G = (V, E)$ where each edge $e \in E$ is associated with a direction from a vertex u to vertex v where $u, v \in V$ (Figure 2.1). Similarly, an *undirected graph* is a graph $G = (V, E)$ in which edges are not associated with any direction information (Figure 2.1).



Figure 2.1: A sample *directed* (left) and *undirected* (right) graph with four vertices: A, B, C, D and four edges that connects vertices to each other.

A graph $G = (V, E)$ is said to be a *compound graph* if it contains any vertex v such that it includes other vertices and edges inside itself (Figure 2.2). In compound graphs there exist parent-child relationships, where a compound vertex v is the parent of the vertices inside itself. Descendant vertices of compound vertex v is said to be *children* vertices of v . Children vertices of vertex v might in turn be a *compound vertex* which represents additional level of nesting (Figure 2.2). *Compound graphs* are well suited for the biological network visualization, where nested entities are present in the topology of networks.

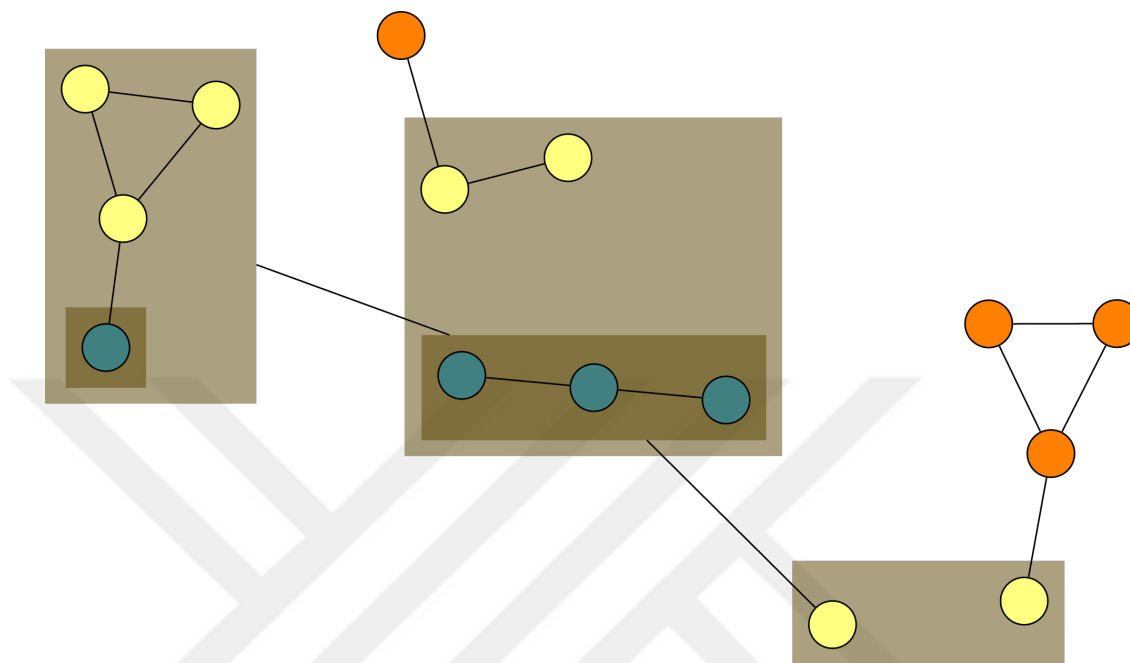


Figure 2.2: A sample *compound-graph* with hierarchically nested vertices.

Graphs are commonly used to represent relational information among a set of objects. Topological structure of a graph defines this relational information by presenting the set of objects as nodes and the relations between those objects as edges that links nodes in the graph to each other.

Since the graphs are abstract structures, they are not associated with any kind of geometric information which is required for visualization. However visualization of graphs are vital in the sense that underlying relational information of the graphs provide fine analysis and comprehension opportunities. Biological pathways and social networks are some of the most popular areas where graph visualization is heavily used. In this respect, as the complexity and size of represented relational information in graphs increase, the demand for effective and powerful graph visualization techniques also increases for fine analysis and comprehension of underlying relational information.

Graph visualization is concerned with the geometric representation of graphs. Such geometric representation of a graph is attributed by location, width, height,

and border along its topological structure. Many aesthetic properties like color of nodes, transparency, node shapes, edge arrow shapes etc. can be regarded as aspects of graph visualization. In other words, any visual attribute that has an influence on viewers' comprehension might be considered as an aspect of graph visualization.

Besides the aspects discussed above, there is another vital concept in graph visualization that is known as *graph layout*. *Graph Layout* is concerned with effective utilization of the relational information that is embodied with the visual representation of a graph via creating aesthetically pleasing views of such graphs according to the viewers' perspective. Even though the concept of aesthetically pleasing views might change from a person to person, there is some commonly accepted criteria for an aesthetically pleasing visualization of a graph in the literature [16]. According to the criteria a good layout favors following aesthetics:

- Vertices should be distributed evenly in the canvas where the graph is visualized.
- Edge crossings should be minimal.
- Edge lengths should be uniform.
- If there exists symmetries in the graph topology, such symmetries should be inherited by graph layout.
- Area that graph uses should be minimal.

Aesthetics above are competitive with each other and often leads to trade-offs where optimality of one aesthetic prevents other. Therefore, it is stated that optimization problems regarding the aesthetics above are NP-Hard [17] and therefore, general approaches introduced in the context of *graph-layout* algorithms are heuristic algorithms.

It can be stated that graph visualizations that do not adhere to the aesthetics discussed above can be associated with a bad graph layout and could potentially

confuse users, whereas a good layout would be more effective in terms of clearly presenting underlying relational information of a graph (Figure 2.3).

Force-directed graph layout algorithms are a class of graph layout algorithms that are regarded as one of the most flexible and successful algorithms in literature [18]. Also known as Spring Embedders, such algorithms are heuristic algorithms and model graph topology as a physical system, where every vertex is represented as a charged particle thus causing a repulsive force to each other and every edge is represented as a spring that is exerting an attractive force to the node pair that share this edge (Figure 2.4).

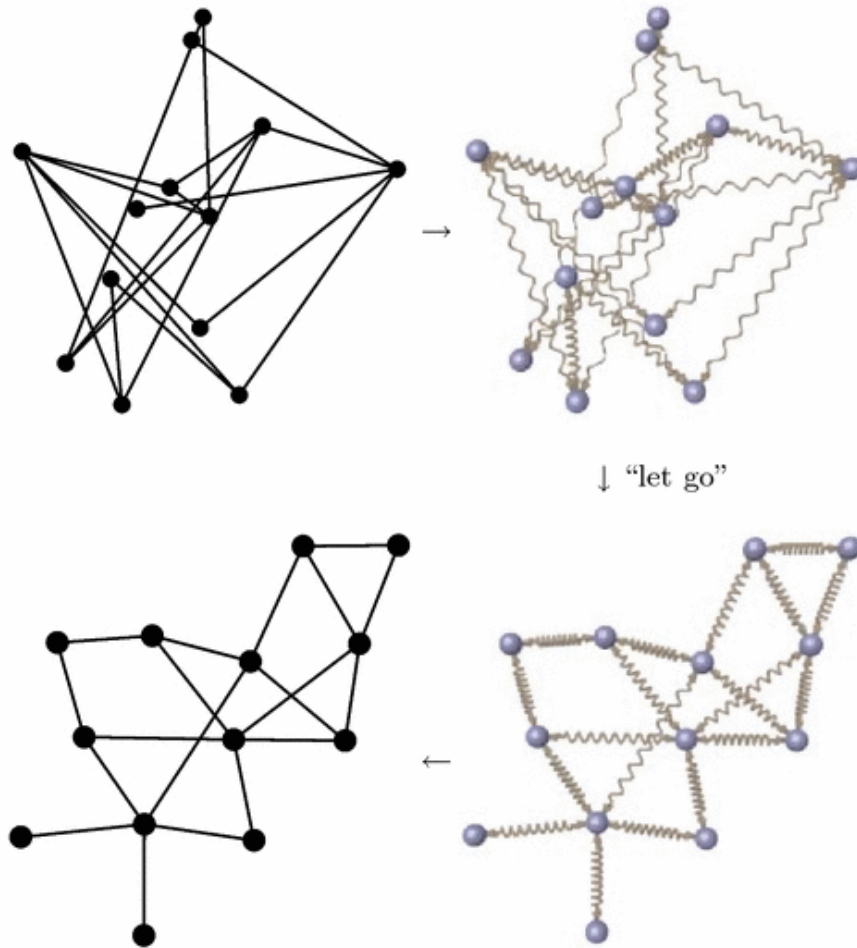


Figure 2.3: Force-directed graph layout model [2].

The graph is laid out as the consequence of repetitive iterations where repulsive and attractive forces are calculated for each vertex and vertices are moved accordingly until simulated system reaches a stable state where the total energy in the system is minimized (Figure 2.4).

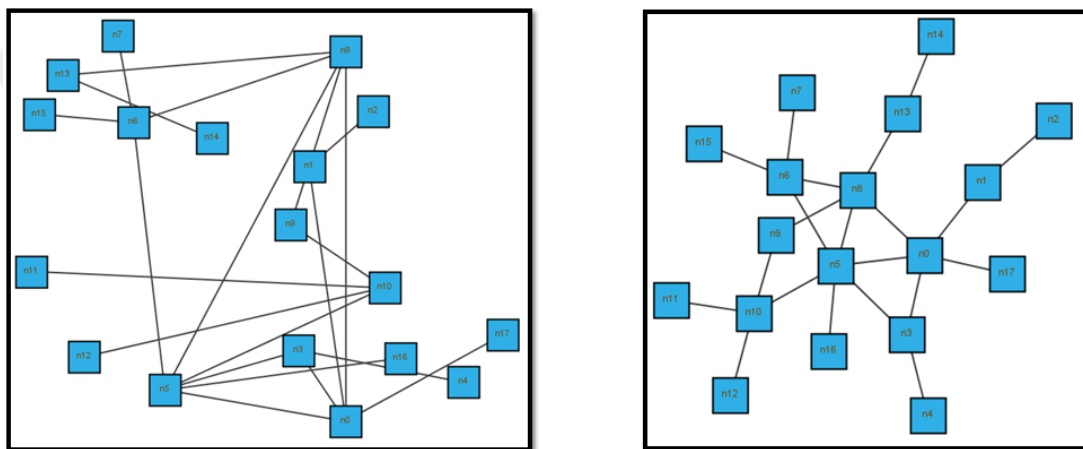


Figure 2.4: A graph on the left before applying layout. Same graph after applying force-directed graph layout.

2.2 Biological Standards

2.2.1 BioPAX (Biological Pathways Exchange)

BioPAX is an effort to standardize the biological pathway representation at molecular and cellular level [3]. As the consequence of the research and development in the field of bioinformatics, amount of pathway data has increased enormously with the diverse communities which in turn, led the emergence of vast number of related tools, huge pathway databases for the utilization of pathway data. However, this situation caused inconsistent representation of pathways in different databases which prevented effective analysis and biological discovery opportunities. BioPAX overcomes the problem of fragmented, inconsistent

pathway data by providing a standard language which provides efficient storing, exchanging and reuse of information of pathways such as signalling pathways, gene-regulation pathways, protein protein interaction pathways in an unified manner. With the help of BioPAX, integrity of representation of pathways in the community is increased that brings more opportunities to the field of pathway research 2.5.

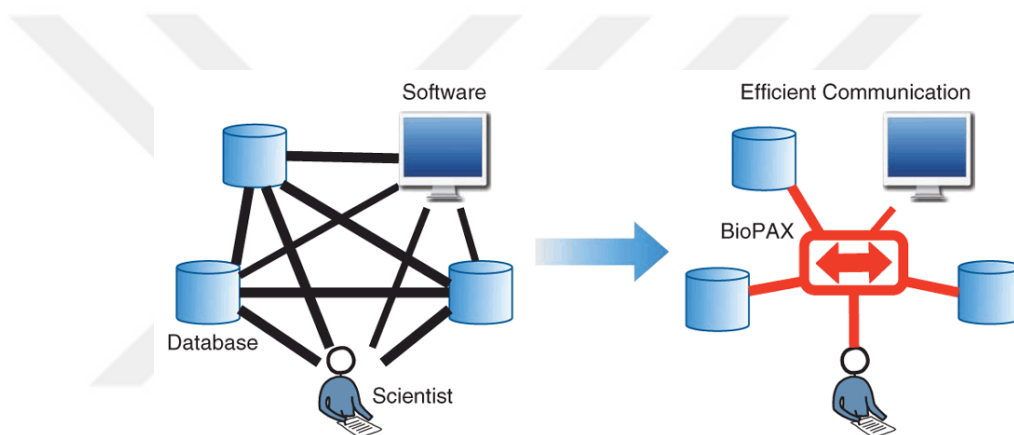


Figure 2.5: BioPAX provides a standard language for the efficient exchange of the pathway data [3].

2.2.2 SBGN (Systems Biology Graphical Notation)

As stated in the previous section, BioPAX was an effort to standardize the format in which the pathways are stored, exchanged without ambiguity. It is obvious that without a proper unified format, ambiguity and complexity decreases the efficiency.

Circuit diagrams and Unified Modeling Language (UML) [19] diagrams are two examples of standard visual languages. They provide regularity by reducing ambiguity and complexity. Moreover they improve communication of complex information by means of software tool support.

This also applies for the graphical representation of biochemical interactions.

Systems Biology Graphical Notation is introduced to resolve the ambiguity and the complexity that merits from non-existence of a unified graphical notation in the context of visual representation of biochemical interactions [20, 21]. It is developed by a community of biochemists, modelers and computer scientists. The hope for SBGN is to represent networks of biochemical interactions in a standard, unambiguous way to provide precise representation, storage, sharing and visualization opportunities according to the needs of diverse biology community.

SBGN consists of three languages that complement each other:

- **Process Description Language (PD):** The SBGN Process Description language aims to represent all biochemical entities and their interactions in the context of molecular processes. The diagrams visualized in SBGN PD language gives insight about how entities transform from one form to another as a result of different interactions. It plays important role regarding visualization of temporary states of molecules in the process of biochemical reactions.
- **Entity Relationship Language (ER):** The SBGN Entity Relationship language aims to reveal information about an entity and the relationships in which such an entity participates. Different than SBGN PD, it aims to provide information about the influences of entities to each other in the network, regardless of the temporary transformation states of entities as consequences of biochemical processes.
- **Activity Flow Language (AF):** The SBGN Activity Flow language is interested in the flow of information between biochemical entities in a network. It is not interested in representing information about the state transitions or entities. Instead it focuses to the activities among entities by ignoring biochemical process details like entity states and processes or relationships.

As discussed, PD diagrams are dedicated to visualize the transformation of molecular entities from one form to another by series of biochemical processes. Change is the essence of such diagrams.

PD language defines its graphical elements as illustrated in Figure 2.8. To represent any physical or conceptual entity in a PD diagram, *Entity Pool Nodes (EPN)* introduced in PD language. Entity Pool Nodes consists of six node types that describes physical entities: macromolecule, simple chemical, nucleic acid feature, multimer, unspecified entity and complex. Source and sink, and perturbing agents are conceptual entities in PD [10].

As stated change is the essence in the PD diagrams. In this regard EPNs are transformed to each other as the consequence of processes. Since processes are vital in PD, there exists different graphical definition for processes: omitted process, uncertain process, association, dissociation, and phenotype as seen in Figure 2.8.

Complex and Compartment nodes are compound nodes in PD diagrams. Complex node represents an entity that may contain other entities: macromolecules, simple chemicals, multimers, even other complexes as a compound structure. Compartment node represents a logical or physical structure such as a cellular compartment which can be composed of EPNs.

PD also offers auxiliary units that can decorate other EPNs, that provide additional information which may be useful to the viewer:

- Unit of Information: Represents abstract information about the functionality of the entity in the diagram that this unit of information belongs.
- State Variable: Represents different physical or informational configurations.
- Clone Marker: Used to indicate that decorated entity is cloned in the diagram.

There is also definition for the edges that connect PD nodes together. There exists nine edge definitions in PD. Symbols at the extreme ends of the edges define the meaning of edges in PD (Figure 2.8).

Figure 2.6 illustrates a Glucose, entity which initially participates in the metabolic reaction of Glycolysis with an ATP entity, where Hexokinase act as an inhibitor and as result of first process one ADP and Glucose 6P entities are produced. Here, Glucose and ATP molecules are linked with straight lines to the process to represent *consumption*, where Hexokinase is connected with a straight line and a ring at the end of this line to emphasize *catalysis*, where lines with arrows at their ends represent *production* of Glucose 6P and ADP

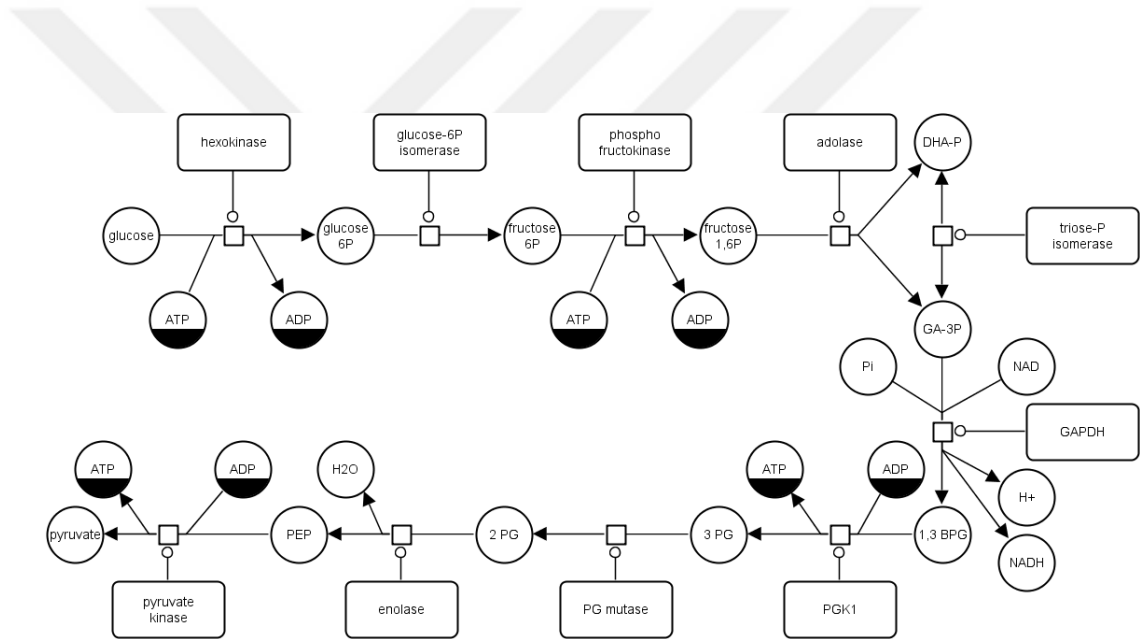


Figure 2.6: Metabolical pathway of *Glycolysis* visualized in PD.

Although SBGN is a promising effort in terms of representing biological pathway information in a unified and unambiguous way, it does not offer any standard about how the diagrams should be stored, exchanged between the tools that aims to support SBGN. Thus, to overcome this problem, *SBGN-ML* was developed which is a dedicated, lightweight XML-based file format [12]. SBGN-ML format includes all necessary geometric information for drawing, positioning of SBGN entities in the corresponding diagrams (Figure 2.7). SBGN-ML is provided as part of LibSBGN project. Hope for the LibSBGN is to provide a standard library that simplifies the storage, exchange and validation of SBGN diagrams in the context of software tools [22]. LibSBGN offers following useful features:

- Representation of SBGN maps programatically.
- Reading / writing in the SBGN-ML format.
- Validation of semantical and syntactical correctness of SBGN-ML format .
- Conversion to other formats such as SBML and BioPAX.
- Support for Java and C++.

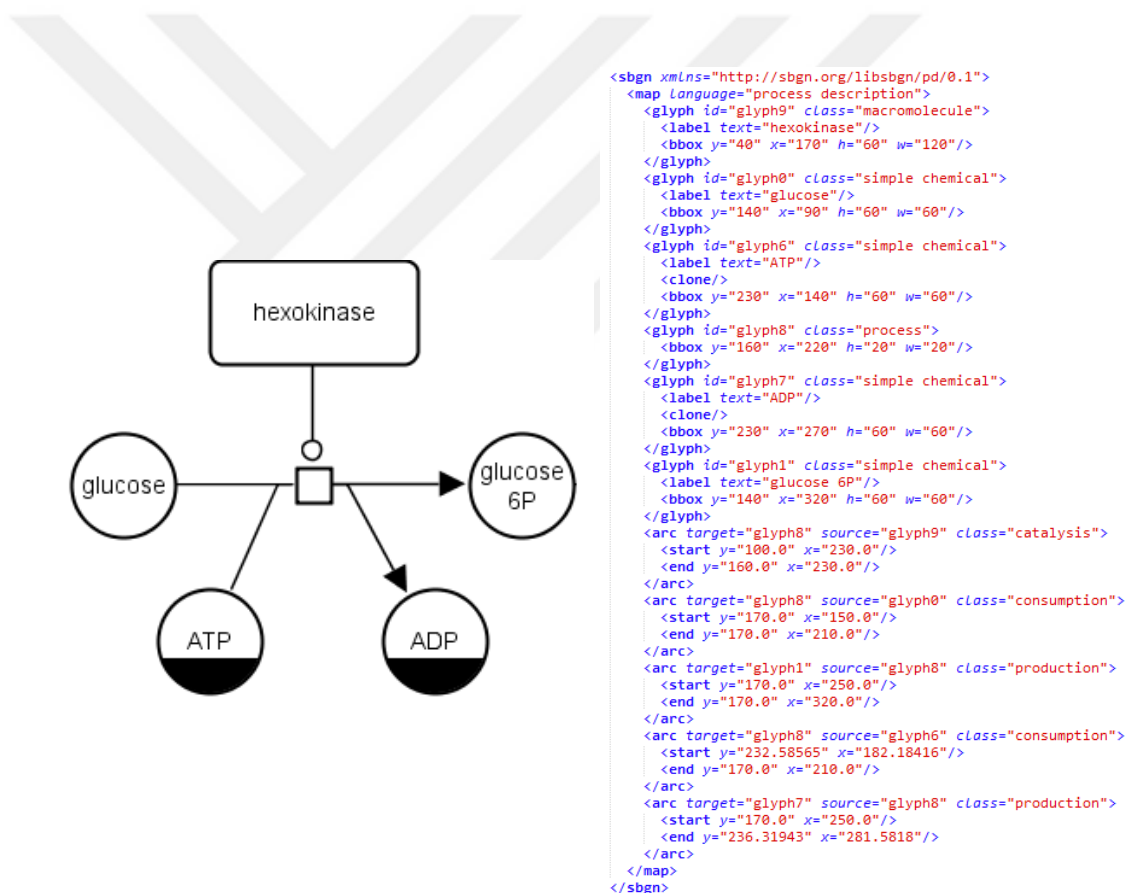


Figure 2.7: A sample metabolic process drawn in SBGN (left) and its associated representation in SBGN-ML (right)

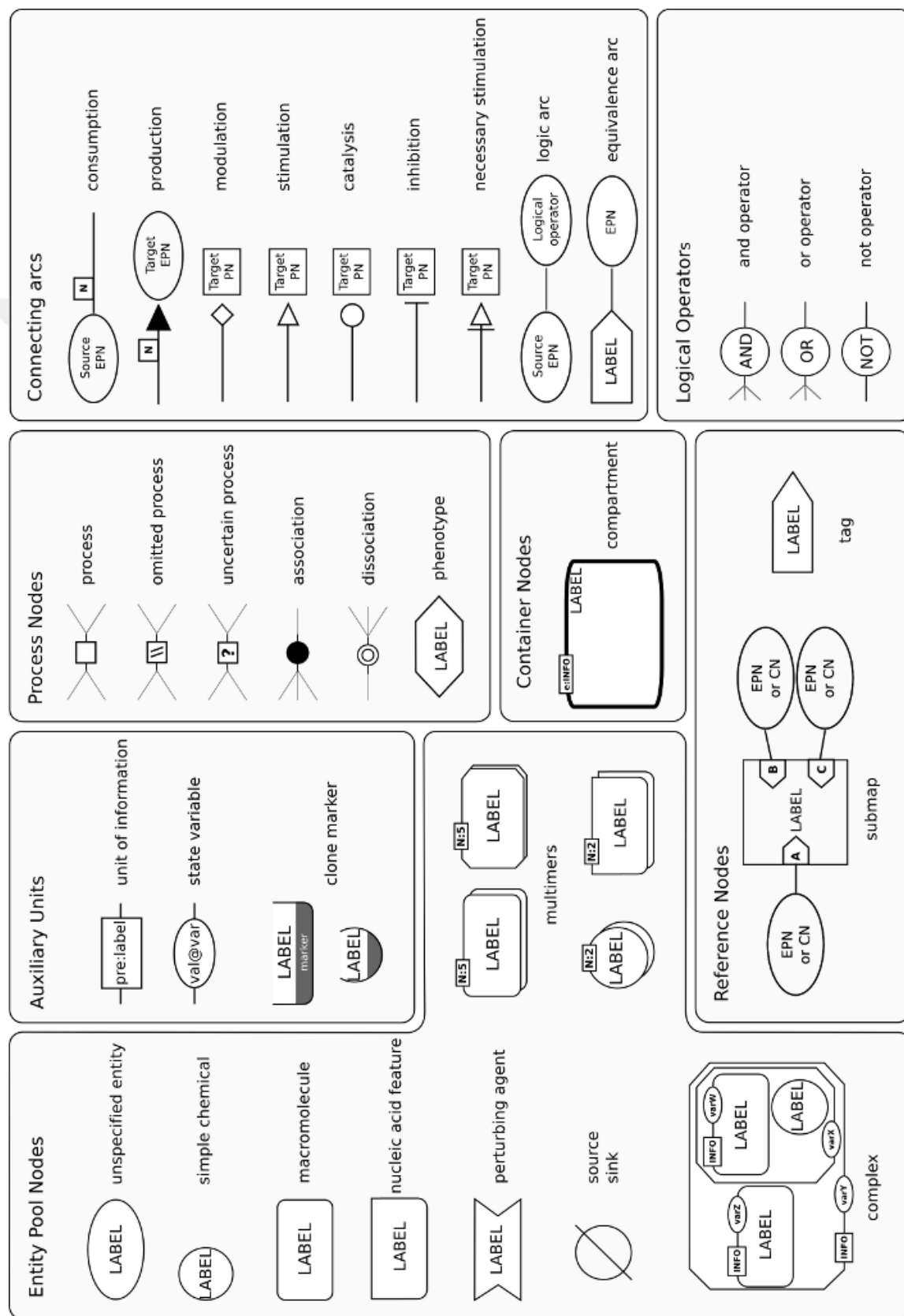


Figure 2.8: Representation of SBGN PD language elements [4]

2.3 Related Tools, Libraries, and Resources

2.3.1 Cytoscape

Cytoscape is an open-source software platform which mainly focuses on developing network visualization tools. *Cytoscape* platform offers three popular network visualization software:

- *Cytoscape*: Desktop application for network analysis and visualization.
- *CytoscapeWeb*: Web based network visualization and analysis library. Built on older web technologies (Flex and ActionScript) [23, 24].
- *Cytoscape.js*: Successor of CytoscapeWeb, built using modern web technologies along with new features for network visualization and analysis.

Cytoscape [25] is an open source desktop application which is developed to visualize molecular interactions and biological networks. In addition to visualization features, Cytoscape provides easy integration of additional information such as annotations and gene expressions along with the visualized networks. Cytoscape was initially developed for biological research but with its rich analysis, visualization and integration features it is adopted as a generic platform for complex network analysis and visualization.

CytoscapeWeb [26] is web based graph visualization and analysis component. CytoscapeWeb was an effort to create a simplified version of Cytoscape for the web platform, not as a complete end user application, but as an independent component that could be used by developers. Core technology behind CytoscapeWeb is based on Flex framework and ActionScript programming language, both emerged as a result of Adobe Flash[27] technology. ActionScript and Flex were popular before HTML5 and they were powerful components for creating interactive applications such as drawing applications and games. As a result of its core technologies CytoscapeWeb is easy to integrate to any web application. Besides its core

technology, CytoscapeWeb provides a JavaScript API (Application Programming Interface) for client side integration, which makes it possible to manipulate and customize the visualized networks as desired.

Cytoscape.js [28] is the successor of CytoscapeWeb. Development of HTML5 brought many improvements in terms of infrastructures for creating interactive web applications without requiring any third-party dependencies such as Adobe Flash. As a consequence of these improvements, Cytoscape.js has been developed as an open source graph theory library for network analysis and visualization which is purely built using native web technology JavaScript. Cytoscape.js inherits all the functionality featured in CytoscapeWeb and offers more advanced, improved functionality that makes it easy to access and manipulate network topology, network style with the help of modern web technologies. It is funded by NRRNB (Natural Resource for Network Biology) [29] and NIH (National Institutes of Health) [30].

Cytoscape.js's architecture is built upon event-driven architecture model with a central core API. Cytoscape.js core provides all fundamental functionality as a JavaScript API regarding accessing and manipulating the graph topology, changing the style of visualized graph, applying layout, viewport related functions like zooming, panning etc. According to the central core design, there are several extensions which communicates with core by means of notifications to the core about required actions on the visualized graph. The core manages all the handling of notifications generated by extensions by performing actual desired actions. Client side usage follows the same behaviour as extensions through the JavaScript API of the core of Cytoscape.js but with a difference that client side can not directly communicate with extensions. Cytoscape.js also provides a flexible interface for custom plugin implementation and registration that enables the custom solutions regarding needs of the clients.

Cytoscape.js can be integrated to any web application with ease by its simple interface. Basically, Cytoscape.js can be initialized on a DOM (Document Object Model) element in a HTML page using initialization function *cytoscape()*. Cytoscape.js works with JSON format [31], which makes it easy to define graph

elements and corresponding style of the network, layout options and meta options along with some callback functions at initialization (Figure 2.9).

```
var cy = window.cy = cytoscape({
  container: document.getElementById('cy'),
  style: [
    {
      selector: 'node',
      style: {
        'content': 'data(id)',
        'text-opacity': 0.5,
        'text-valign': 'center',
        'text-halign': 'right',
        'background-color': '#11479e'
      }
    },
    {
      selector: 'node:parent',
      style: {
        'content': 'data(id)',
        'text-opacity': 0.5,
        'text-valign': 'bottom',
        'text-halign': 'center',
        'background-color': '#fff'
      }
    },
    {
      selector: 'edge',
      style: {
        'width': 4,
        'target-arrow-shape': 'triangle',
        'line-color': '#9dbaea',
        'target-arrow-color': '#9dbaea',
        'curve-style': 'bezier'
      }
    }
  ],
  elements: {
    nodes: [
      { data: { id: 'n0' } },
      { data: { id: 'n1' } },
      { data: { id: 'n2', parent: 'n7' } },
      { data: { id: 'n3', parent: 'n7' } },
      { data: { id: 'n4', parent: 'n7' } },
      { data: { id: 'n5', parent: 'n7' } },
      { data: { id: 'n6' } },
      { data: { id: 'n7' } }
    ],
    edges: [
      { data: { source: 'n0', target: 'n1' } },
      { data: { source: 'n0', target: 'n2' } },
      { data: { source: 'n2', target: 'n3' } },
      { data: { source: 'n2', target: 'n4' } },
      { data: { source: 'n2', target: 'n5' } },
      { data: { source: 'n1', target: 'n6' } }
    ]
  },
  layout: {
    name: 'cose'
  }
});
```

Figure 2.9: An example of *Cytoscape.js* usage.

Moreover, the manipulation and accessing features of Cytoscape.js is highly promising. Cytoscape.js follows selector and chaining conventions for querying and manipulating visualized networks. These conventions are well known in the web literature because of the JQuery framework [32] and CSS language which are well known for a typical web developer. A sample query and manipulation is illustrated in Figure 2.10.

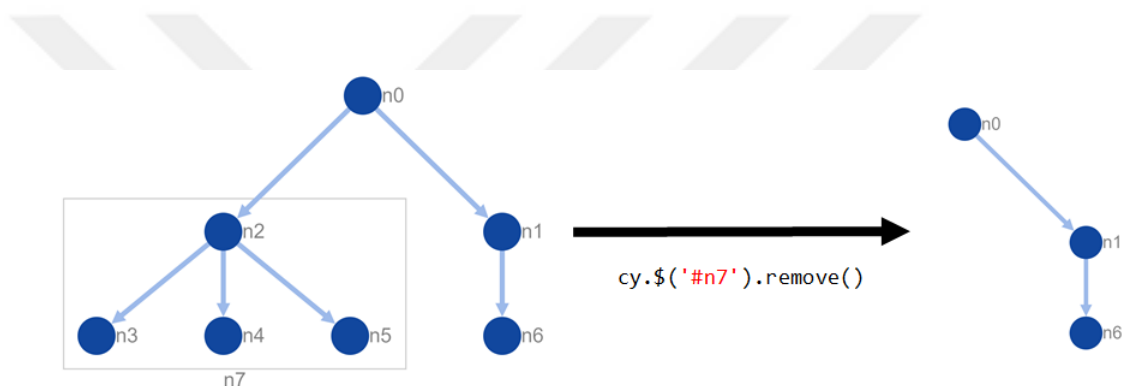


Figure 2.10: A sample node removal of a compound node by querying node with id ‘n7’ and removing it from the graph that exemplifies manipulation and query features of *Cytoscape.js*. This graph is also visualized using *Cytoscape.js*.

Besides rich accessing and manipulation features, there exists functionality for changing the visual style of the graph, animation and applying graph layout. More importantly, Cytoscape.js includes functionality regarding graph theoretic algorithms such as *breadth-first search*, *Kruskal’s algorithm*, accessing siblings of a node, populating neighbourhood of a node. Having such features greatly simplifies access and manipulation of graph topology and reduces complexity for developers.

Hierarchical structures in network visualization is frequently favored in the fields such as biological pathway visualization for visualizing molecular complex structures, cellular compartments. In this regard, Cytoscape.js supports compound nodes and offers corresponding functionality to access, traverse, and perform operations on compound nodes which makes Cytoscape.js a convenient

choice for visualizing networks that include hierarchically structured information (Figure 2.10).

2.3.2 Paxtools

Although BioPAX provided a unified standard for representation, exchange and reuse of pathway information, complex structure of BioPAX with its huge number of properties and classes makes it difficult to develop software tools that manipulates BioPAX. In this regard, Paxtools is introduced as an open-source Java library which includes effective algorithms and methods for common and easy but time consuming tasks of reading, writing, searching, merging, comparing and transforming pathway information between formats [13].

Paxtools provides conversion of pathway information from BioPAX format to SBGN-ML, Simple Interaction Format (SIF) and Gene Set Enrichment Analysis (GSEA) format [33].

Since BioPAX is an adopted format by community, Paxtools becomes a handy choice for providing useful functionality in terms of manipulating data in BioPAX format which promotes the improvement of software tools and systems that are dedicated to the exploration, analysis and visualization of pathways.

2.3.3 Pathway Commons

Pathway Commons [1] provides access to the collection of publicly available pathway information that is gathered from multiple pathway databases. Pathway Commons currently includes 42,000 pathways, 1,350,000 interactions from 22 pathway sources [34].

Pathway Commons offers a web service for scientists, that provide querying facilities of rich pathway information that is collected from different pathway databases in BioPAX and Proteomics Standards Initiative - Molecular

Interaction(PSI-MI) formats [35]. The web service offers a web API by which developers can query the rich collection of pathway information and download resultant pathway information in various formats that are adopted by the community.

Web service API of Pathway Commons offers an important feature of graph queries. Graph queries are vital for finding connections and neighborhoods of elements, such as the paths between two genes of interest or the neighborhood of provided genes of interest. The queried entities and type of the resulting pathways can be specified while making query requests to the web API. Supported graph types are as follows:

- COMMON STREAM: Common downstream or common upstream of a specified set of entities.
- NEIGHBORHOOD: Neighborhood of provided set of entities.
- PATHS BETWEEN: Paths between provided set of entities.
- PATHS FROM TO: Paths from specified set of entities to target set of entities.

The output format of resulting pathways is BioPAX format by default but it can be specified optionally :

- BINARY SIF: Simple Binary Interaction Format
- BIOPAX: BioPAX RDF/XML Format
- EXTENDED BINARY SIF: Extended Simple Binary Interaction Format
- GSEA: Gene Set Expression Analysis Format
- JSONLD: JSONLD format
- SBGN: Systems Biology Graphical Notation Format



Figure 2.11: A sample graph query to the web service of Pathway Commons.

2.3.4 ChiLay

Chisio Layout (ChiLay) is an open source Java library. It includes useful graph layout algorithms that provide automatic layout for flat, simple, compound and clustered graphs. ChiLay currently includes Compound Spring Embedder (CoSE) algorithm for compound graphs [14], Circular Spring Embedder (CiSE) for clustered graphs [36], SBGN-PD Layout algorithm for providing a specialized layout algorithm for the domain specific requirements of the SBGN PD graphs [37]. These algorithms are developed and maintained by our research group. ChiLay also offers several other popular layout algorithms for both directed and undirected graphs, such as Sugiyama Layout [38].

ChiLay was designed to be used both as a local Java component that can be integrated to any Java application or remotely by a web application by sending graph structure in a XML format to a dedicated server where ChiLay is deployed. This dedicated server in return sends the laid out graph structure to the client in the XML format. Details about remote usage, ChiLay architecture and customization features can be found in ChiLay programmer's guide [39].

As part of our work, we integrated ChiLay to the Paxtools as a local dependency which is responsible for providing graph layout support for the SBGN-PD pathways that are converted from BioPAX format to SBGN-ML format. This graph layout support is crucial for the community. Paxtools is used by Pathway

Commons to provide pathways in SBGN therefore automated graph layout for such pathways utilizes the associated information embodied in SBGN diagrams.

2.3.5 TCGA

The Cancer Genome Atlas (TCGA) is a collaborative effort that is supported by the National Cancer Institute (NCI) [40], the National Human Genome Research Institute (NHGRI) [41] and National Institutes of Health (NIH) [30]. TCGA provides genomic alteration information about 33 types of cancer. It is stated that TCGA data set contains more than two petabytes of genomic data which is publicly available and helps cancer research community to understand, prevent, diagnose and treat the cancer.

Hundreds of samples for each specific type of cancer is being analyzed by TCGA. Analyzing many different samples from many different patients gives better insight about what makes one cancer distinct from another cancer. This reveals vital information since diverse set of samples will be powerful in terms of observing the cancer behaviour and the different response of treatments for a specific cancer. Trailing down the outcomes of genomic data of TCGA, especially in the context of pathways, researchers are able to understand cancer better which leads development of more effective, specialized opportunities for cancer patients.

2.3.6 cBioPortal

cBioPortal is a popular web portal, getting about sixty thousand visits globally per month, featuring visualization, analysis and download of large-scale cancer genomics data sets [8, 9]. Given a set of genes of interest and associated cancer study, the portal provides graphical summaries of gene specific information from multiple platforms, network visualization and analysis, survival analysis, patient specific queries, and software programmatic access. Web interface of cBioPortal makes it easy to access complex genomic profiles without requiring expertise on

bioinformatics which facilitates biological advancements.

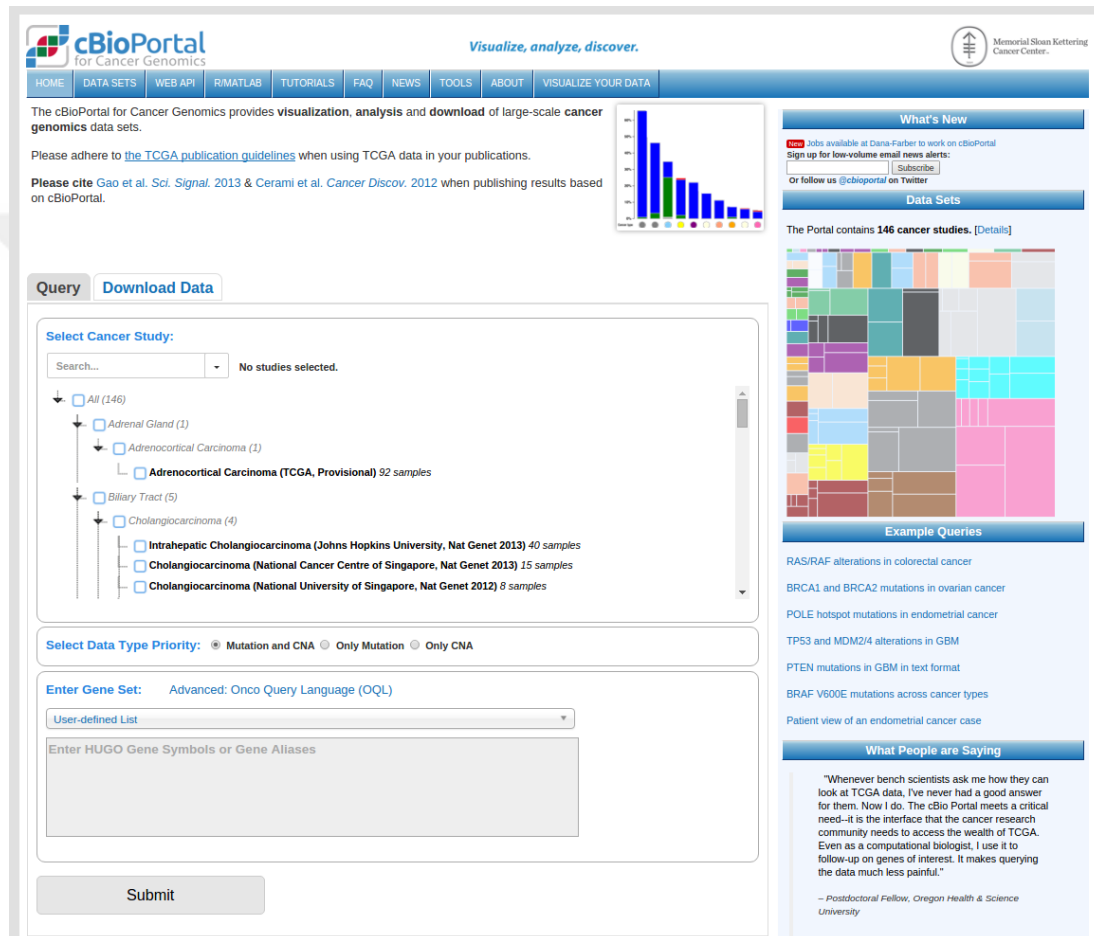


Figure 2.12: cBioPortal web application home page.

The cBioPortal for Cancer Genomics was originally developed at Memorial Sloan Kettering Cancer Center (MSK) [42]. The software is now developed and maintained by a multi-institutional team, consisting of MSK, the Dana Farber Cancer Institute [43], Princess Margaret Cancer Centre in Toronto [44], Children's Hospital of Philadelphia [45], The Hyve in the Netherlands [46], and our research group i-Vis at Bilkent University in Ankara, Turkey [47].

2.3.7 BioGene

BioGene [48] is a simple web service which lets scientist to make queries about a specific gene and provides description about its gene function, its aliases, its chromosome location. BioGene primarily uses Entrez Gene, a gene database provided by NCBI [49].

2.3.8 GraphML

GraphML is an XML based file format that is an effort to represent graph structure information in a common format for the efficient exchange of graphs. GraphML provides a core language that can represent structural properties of a graph and it lets addition of application-specific data which makes it a flexible and efficient choice for the exchange of graph information. Directed, undirected and compound graphs can be represented using GraphML [50].

2.3.9 Google Real Time API

Google Real Time API [15] is a technology that is behind the popular collaborative document editing tool Google Docs [51]. Google offers this technology as a public JavaScript API which basically lets developers to design a shared data model, just like a local in-memory data model. API offers data structures and objects such as lists, maps, objects and even custom JavaScript objects that can be used as part of a shared data model. Shared data model of each application is stored in a document called collaborative document which is stored in the Google Drive [52] folder of each user. Changes to the shared data model of an application on behalf of a user is propagated to the all users of the application remotely.

The behaviour of the shared data model objects are just like local in-memory JavaScript objects with a slight difference that a collaborative Google Real Time API data model may change as a result of modifications of other users rather than

current local user. To achieve this purpose applications using Google Real Time API attaches listeners to a collaborative document which stores collaborative data model, so that changes on shared data model is synchronized among all users.

Whenever associated shared data model of collaborative document is modified, the API sends a representation of the change to the server so that the change can be recorded in the collaborative document and synchronized to other collaborators. These changes are named as ‘mutations’ in the context of Google Real Time API. Design of the API provides automatic conflict resolution which means that users will not face problems about edit conflicts.

Google Real Time API makes sure that data models are ”eventually consistent” which means that when all the collaborators stop editing, eventually every user will have the same shared data model in their application. However, Google Real Time API does not guarantee in what order the changes will be delivered to the users or when all the users see the changes.

In our work, we are using Google Real Time API for allowing construction of TCGA pathways in a collaborative manner.

2.3.10 SBGNViz.js

SBGNViz.js [11] is a web based interactive visualization and analysis component for process description maps in SBGN. SBGNViz.js is purely written in JavaScript. It is capable of visualizing the pathway information stored in SBGN-ML format. SBGNViz.js provides full visualization support for SBGN-PD language including compound structures such as molecular complexes and cellular locations, using nested drawings. In addition to the visualization capabilities, it features loading and saving pathways in SBGN-ML format, automated graph layout support, zooming and scrolling on viewport, deletion of nodes and edges, and highlighting of desired portions of visualized pathways. On demand this tool also provides, detailed properties and external references to macromolecules or nucleic acid features that may be fetched from the BioGene service.

SBGNViz.js uses Cytoscape.js as core network visualization component. However, since Cytoscape.js offers generic shapes for representing nodes like ellipses, rectangles, etc. it is not possible to represent visual characteristics of SBGN elements that are defined in SBGN-PD language using such generic shapes that Cytoscape.js provides. Thus, SBGNViz.js provides an extension renderer that is registered to Cytoscape.js to visually represent the SBGN specific elements. SBGNViz.js also provides conversion of pathway information between JSON and SBGN-ML format, since JSON is accepted format by Cytoscape.js, SBGNViz.js parses pathway information from SBGN-ML and generates a JSON object that includes all pathway information. Similarly, for exporting visualized pathways SBGNViz.js converts such JSON representation of pathway to SBGN-ML format.

Chapter 3

Network Visualization Component in cBioPortal

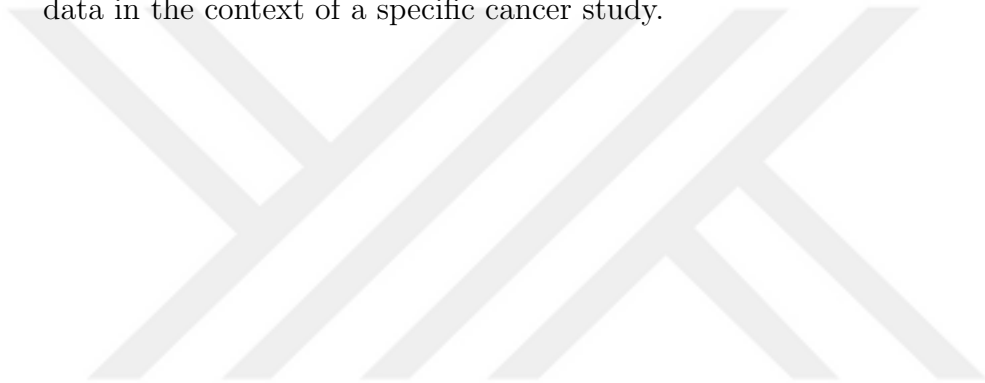
This chapter focuses on design and implementation of software tools that aim to visualize cancer genomics data in the context of networks from simple regulatory networks to process description diagrams within the cBioPortal for Cancer Genomics (cBioPortal), which is a popular web portal, providing visualization, analysis and download of large-scale cancer genomics data sets.

In this respect, we designed and developed a network visualization and analysis component in cBioPortal that presents neighborhood pathway of genes of interest in the context of specific cancer study. The genomic alteration data is overlaid on the visualized network with numerous ways to filter and manage complexity of the network (e.g. by alteration percentage or by type or source of the interactions).

Upon demand, the user is allowed to obtain a more detailed, mechanistic view of the interactions between gene pairs in such gene interaction networks, from Pathway Commons database with a live query that returns network of interest in the SBGN-PD notation in SBGN-ML format.

3.1 Visualization of Gene Interaction Pathways with Genomic Data in cBioPortal

One of the core feature that we have provided for cBioPortal focuses on efficient and intuitive visualization of neighborhood pathway of given genes of interest, where interactions among such genes is visualized along with genomic alteration data in the context of a specific cancer study.



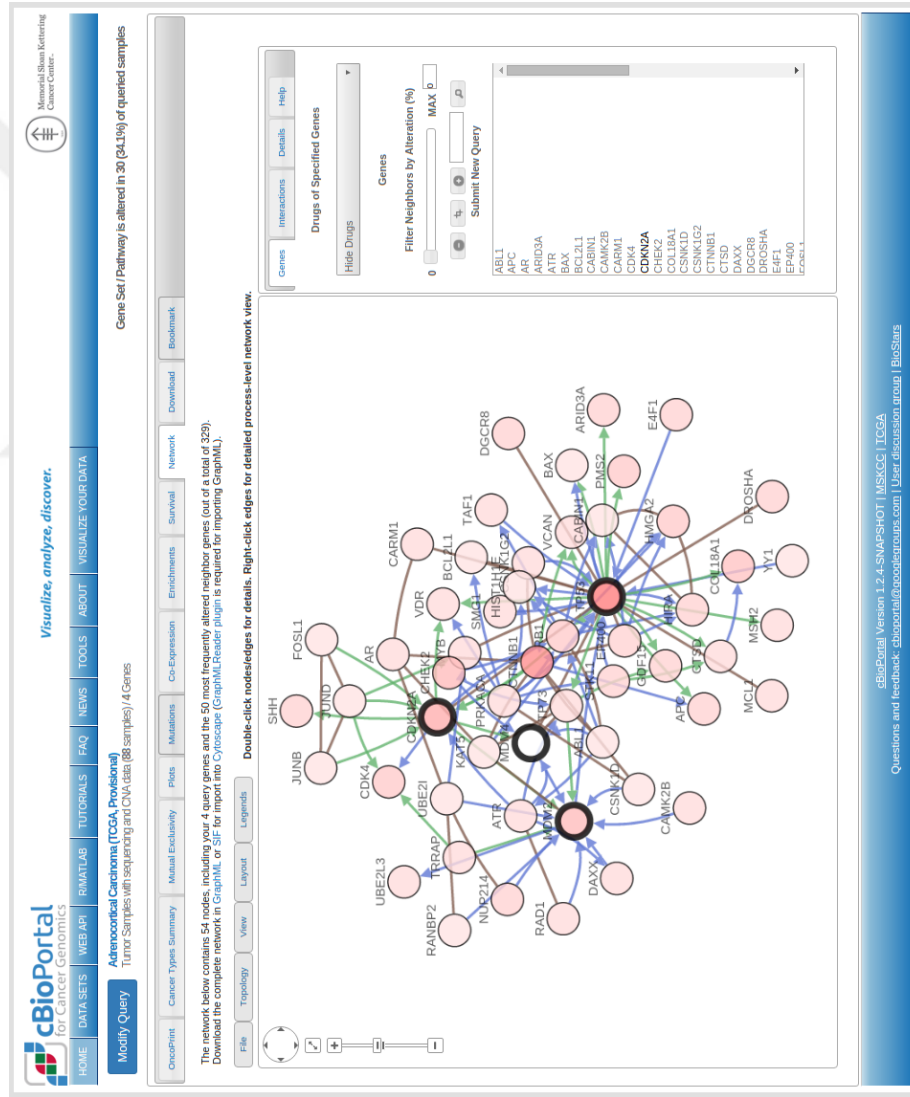


Figure 3.1 provides a sample visualization of neighborhood pathway of gene set of *MDM2*, *CDKN2A*, *MDM4*, *TP53* which is altered in 30 samples out of 88, in the context of *Adrenocortical Carcinoma (TCGA, Provisional)* study. The visualized pathway consists of 54 genes in total where 50 of these genes are most frequently altered neighbour genes of *MDM2*, *CDKN2A*, *MDM4*, *TP53* out of total 329 neighbours.

Genes are represented using circular node shapes in the visualized networks. Circular nodes with thick borders represent nodes that are in user provided gene set. These nodes are called as *seed nodes* since they can be considered as seed of the obtained gene neighborhood in the visualized network. Circular nodes with normal borders are called as *linker nodes* since they connect seed nodes and other linker nodes to each other in the neighborhood.

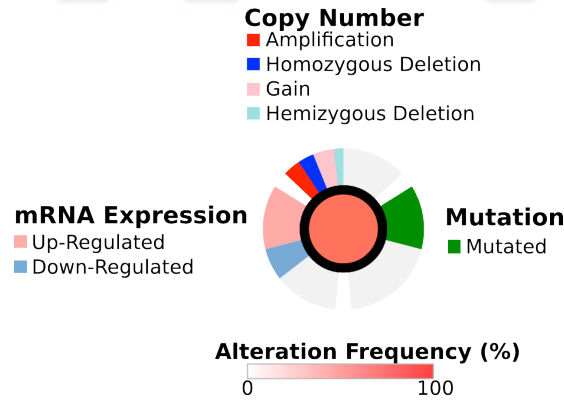


Figure 3.2: Overlay of genomic data on a node. Disks around the node represent genomic profiles in the context of copy number, mRNA expression and Mutation. The node is colored according to the alteration frequency using a white to red color gradient

By default, each gene in the network is color coded along a white to red color gradient, indicating the total frequency of alteration across the selected cancer study set where deeper red indicates higher frequency of alteration (Figure 3.2).

The exact genomic data displayed on the network depends on the genomic

profiles that user selected. For instance, user can chose to include mutation, copy number and mRNA expression profiles to visualize. We overlay such genomic data by rendering three additional disks around the genes in the network. Disk that is placed at the top of a gene is dedicated to visualize copy number profile of a gene in terms of *Amplification*, *Homozygous Deletion*, *Gain* and *Hemizygous Deletion*. Disk that is placed on the bottom left of a gene represents mRNA expression profile of the gene in terms of *Up-Regulation* and *Down-Regulation*. The disk that is placed at the bottom right of a gene is dedicated to visualize the mutation profile of a gene (Figure 3.2).

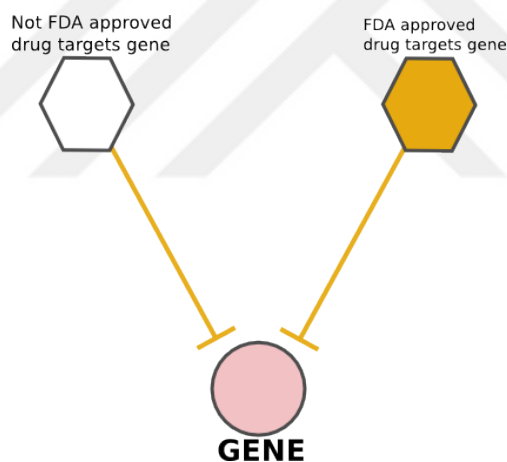


Figure 3.3: Visualization of drugs targeting a gene in cBioPortal

Drugs that are targeting genes in neighborhood pathways are also visualized by our tool. Note that drugs that are approved by FDA are filled in orange color whereas not FDA approved ones are not filled (Figure 3.3).

The interaction types are derived from the BioPAX to binary interaction mapping rules defined within Pathway Commons. They are encoded by different edge colors in visualized pathways [53]. The interaction types are:

- *Controls-state-change-of*: First gene controls a reaction that changes the state of the second gene.
- *Controls-transport-of*: First gene controls a reaction that changes the cellular location of the second gene.
- *Controls-phosphorylation-of*: First gene controls a reaction that changes the phosphorylation status of the second gene.
- *Controls-expression-of*: First gene controls a conversion or a template reaction that changes expression of the second gene.
- *Catalysis-precedes*: First gene controls a reaction whose output molecule is input to another reaction controlled by the second gene.
- *In-complex-with*: Genes are members of the same complex.
- *Interacts-with*: Genes are participants of the same molecular interaction.
- *Neighbor-of*: Genes are participants or controllers of the same interaction.
- *Consumption-controlled-by*: The small molecule is consumed by a reaction that is controlled by a gene
- *Controls-production-of*: The gene controls a reaction of which the small molecule is an output.
- *Controls-transport-of-chemical*: The gene controls a reaction that changes cellular location of the small molecule.
- *Chemical-affects*: A small molecule has an effect on the gene state.
- *Reacts-with*: Small molecules are input to a biochemical reaction.
- *Used-to-produce*: A reaction consumes a small molecule to produce another small molecule.

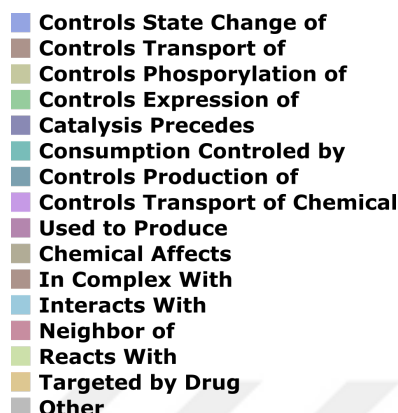


Figure 3.4: Color coding of interactions in cBioPortal network visualization component

Besides the canvas where the network is visualized, there exists complexity management functionality for visualized pathways, help content for usage instructions and documentation, and inspector content for more detailed information about a selected gene, interaction and drug from visualized pathways.

In this respect, there exists a menubar above visualization canvas which consists help content about visualization properties of network elements, topology and view related complexity management operations such as hiding selected elements, showing only selected elements, highlighting neighbour of selected nodes, toggling genomic data discs. It is also possible to perform graph layout to visualized network along with the capability of tuning layout parameters of applied graph layout algorithm CoSE (Figure 3.5).

There exists a panel right to the visualization canvas which also includes additional functionality in different tabs.

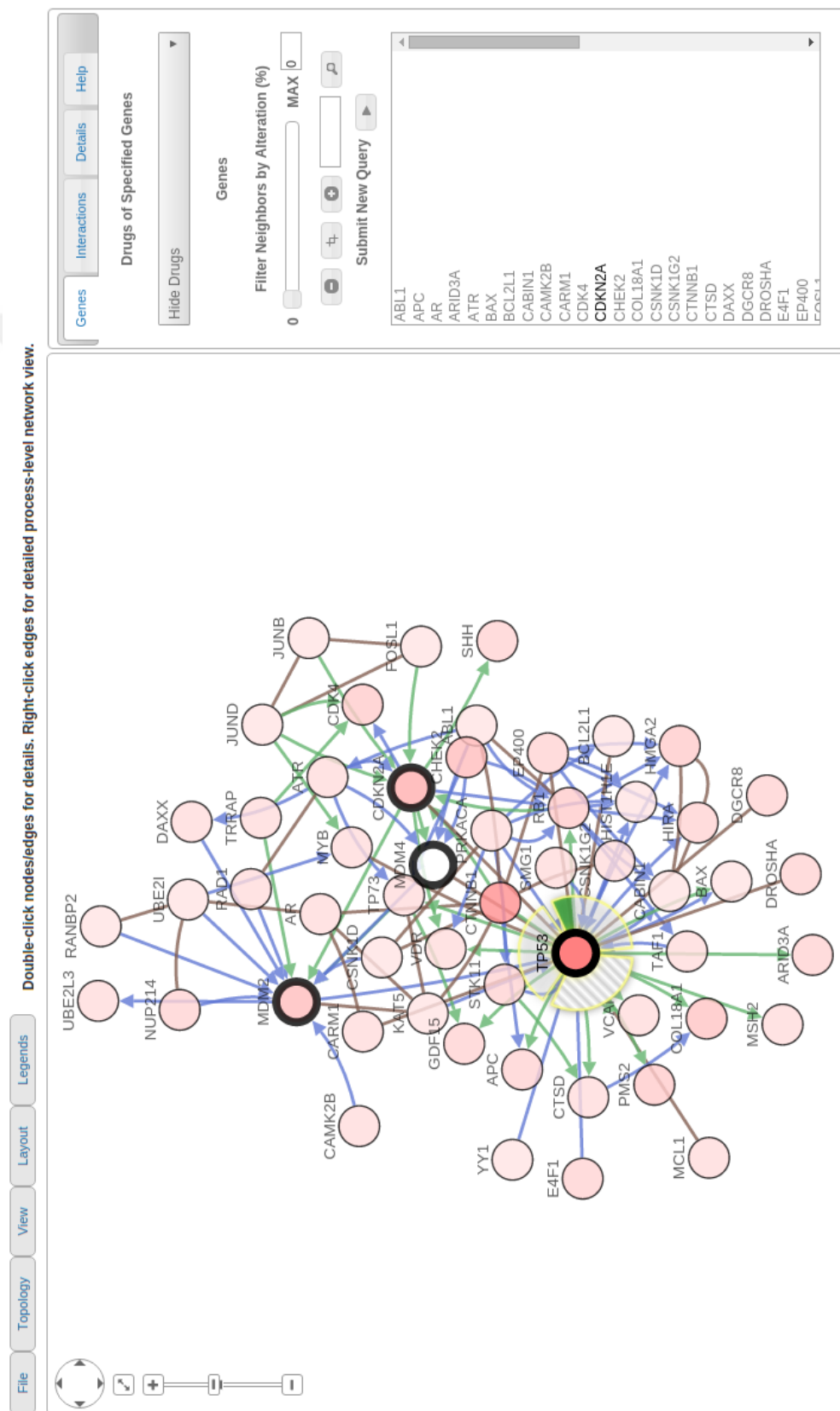


Figure 3.5: Genes tab

Genes and *Interactions* tabs provide complexity management operations regarding genes and the interactions in the network. Details about the complexity management operations that these tabs provide will be elaborated in Section 3.3.2.

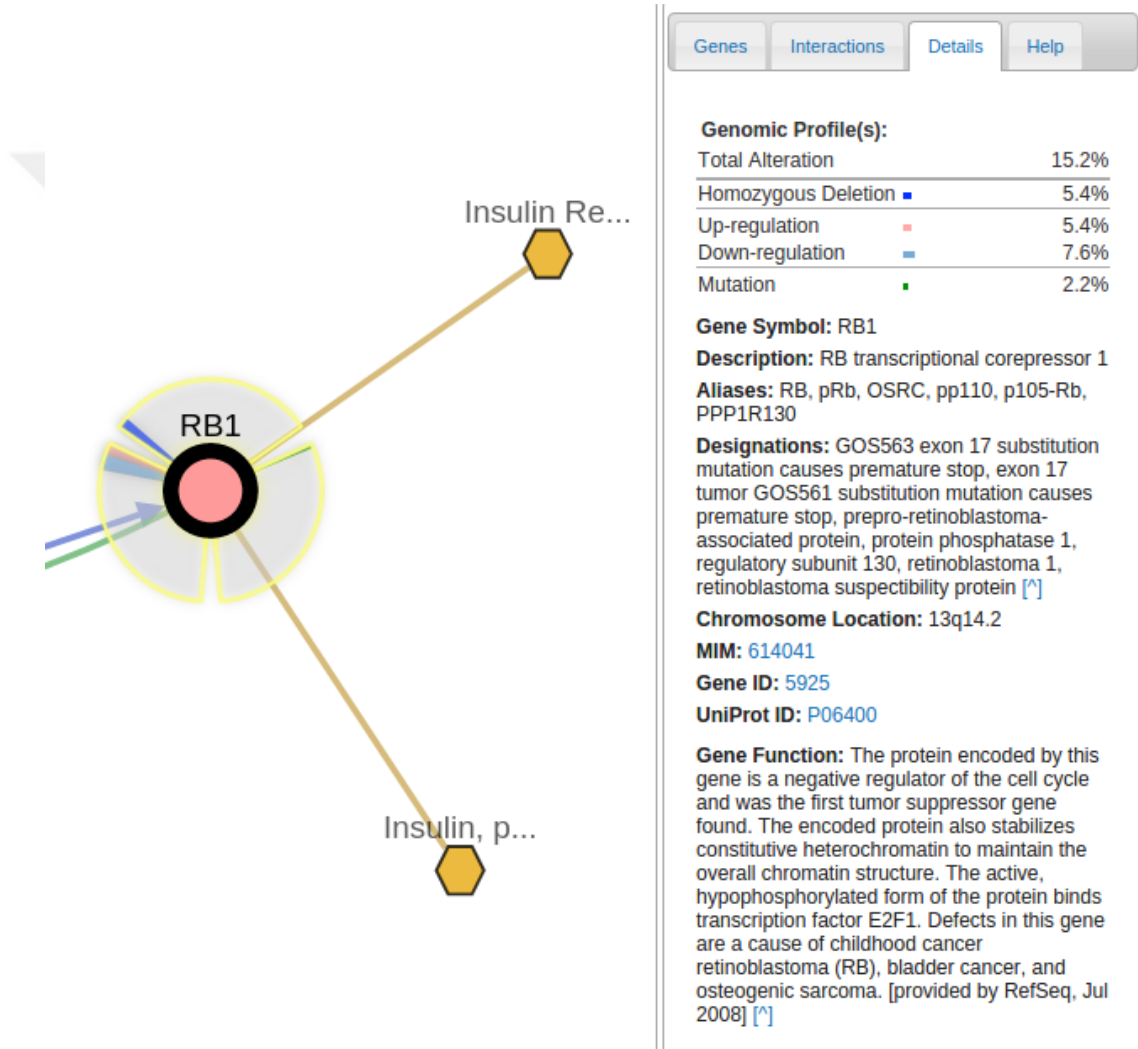


Figure 3.6: Detailed drug information of *RB1* gene

Details tab provides detailed information about the genomic profile of a selected gene in network. This inspector provides additional gene information which is obtained by a query to BioGene [48] service whenever a gene is double clicked on canvas. Similarly, whenever a drug or interaction is selected from visualized network, additional information about selected drug or interaction is provided in

the same tab (Figure 3.6).

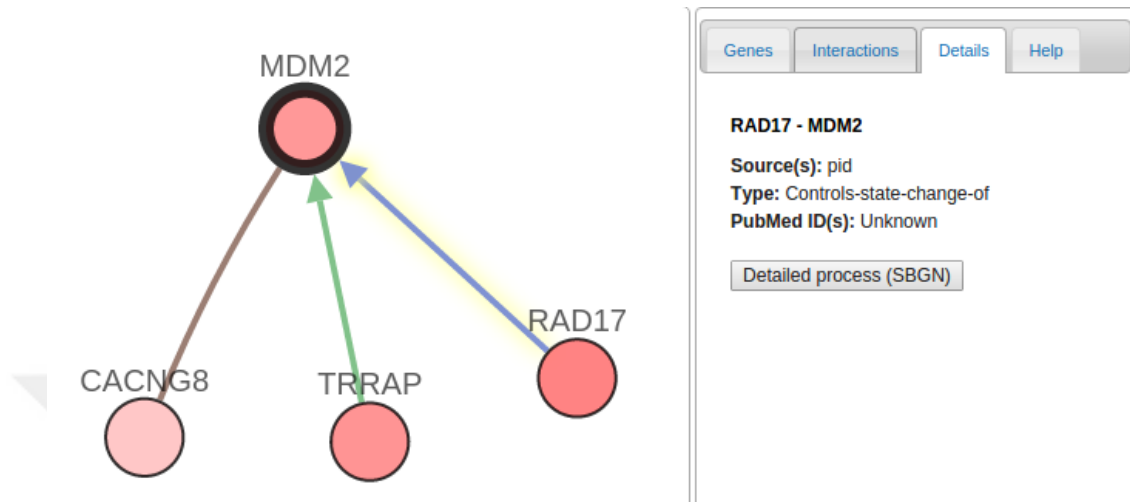
Help tab includes useful information about the usage details, visualization details and interaction details.

3.2 Visualization of Process Level Details of Interactions in cBioPortal

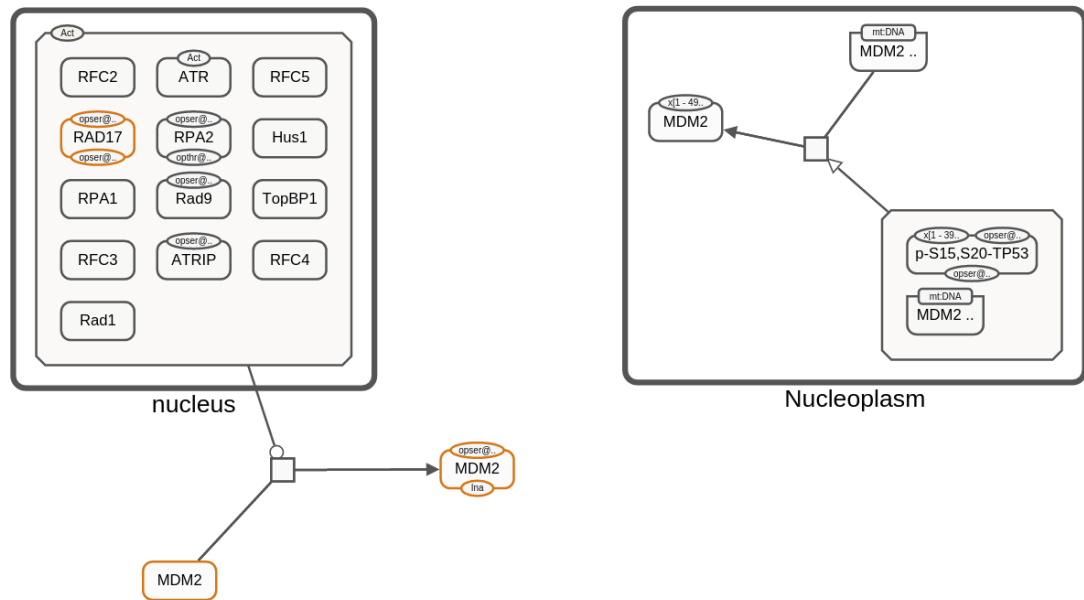
Gene interaction pathways derived from neighborhoods of *seed genes* visualize useful information about interactions of genes in the context of a specific cancer study. As discussed in Section 3.1, this view focuses on visualizing binary interactions among genes along with genomic alteration profiles of genes in the network. This view ignores the molecular process level interaction details of genes which could support insight about the role of interacting genes in the cancer study of interest.

In this regard, we provided a complementary view that is dedicated to visualize the molecular process level description of an interaction between gene pairs in the gene interaction pathways. To achieve this, we integrated an improved version of SBGNViz.js [11] to cBioPortal for visualizing, analyzing such complementary process description maps in SBGN. This SBGNViz.js instance is improved in a way to provide many additional useful features such as advanced complexity management operations.

Figure 3.7 illustrates process level details of *MDM2* and *RAD17*. Figure 3.7a indicates that there is an interaction between these two genes, where *RAD17* controls state change of *MDM2*. Upon request, view in Figure 3.7b can be obtained representing associated interaction details in SBGN, where the pathway information is taken by a live query to the Pathway Commons web service. In this process level view, we observe that *RAD17* belongs to a complex molecule structure, which catalyzes the metabolic process where *MDM2* enters as substrate and changes its state as a result.



(a) An interaction of interest from gene interaction pathway



(b) Process level details of interaction of interest in (a), visualized in SBGN

Figure 3.7: Process level interaction details of the interaction between *MDM2* and *RAD17* genes

This complementary view can be obtained for each interaction in the visualized neighborhood pathways. Users can perform a right click on the interactions or can use edge inspector to obtain such view (Figure 3.7a).

Pathway Commons web service is being used for retrieving paths between two genes of interest in SBGN-ML format to provide mechanistic details of interaction between these genes. Pathway Commons converts queried pathway information from BioPAX format to SBGN-ML. To achieve such conversion, Paxtools library is used in Pathway Commons core. In the context of our work, we provided automated graph layout facility to the resulting SBGN pathways that Pathway Commons provides. Virtual representations of SBGN elements after the conversion from BioPAX models do not include any geometric information regarding geometric boundaries and location of SBGN elements, in the obtained pathways which is an important aspect considering the efficient visualization of SBGN pathways. In this respect, we contributed to the Paxtools library in a way that geometric boundaries of SBGN elements are estimated according to the annotation content of such elements and regarding location information, we integrated CoSE algorithm to Paxtools for automated graph layout of such SBGN pathways (Figure 3.8).

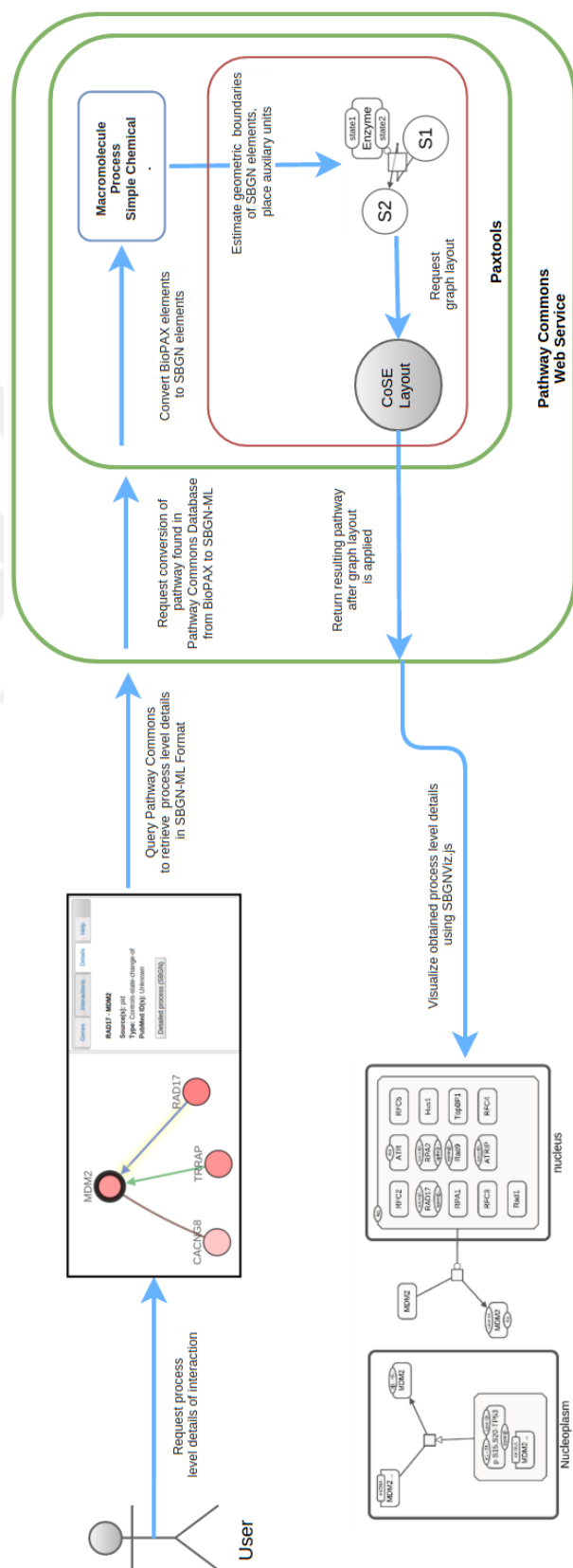


Figure 3.8: Flow of events regarding visualization of process level interaction details

In addition to the visualization capability in SBGN, this view provides complexity management operations such as expand and collapse operations, hiding and highlighting regarding specific requirements of SBGN maps along with useful functionality such as searching a gene, performing layout. There is a toolbar above visualization canvas where such functionalities are included for easy access. Moreover, there is also a menu bar above this toolbar that groups regarding complexity management functionalities, help content, graph layout support under associated menu items (Figure 3.7b). Corresponding complexity management operations will be elaborated in next section.

Upon request detailed information about genes are also available in this view which is obtained again from BioGene service.

MDM2	
Gene Summary:	This gene encodes a nuclear-localized E3 ubiquitin ligase. The encoded protein can promote tumor formation by targeting tumor suppressor proteins, (...)
Aliases:	HDMX, hdm2, ACTFS
Description:	MDM2 proto-oncogene
Chromosome Location:	12q14.3-q15
UniProt ID:	Q00987
Gene ID:	4193

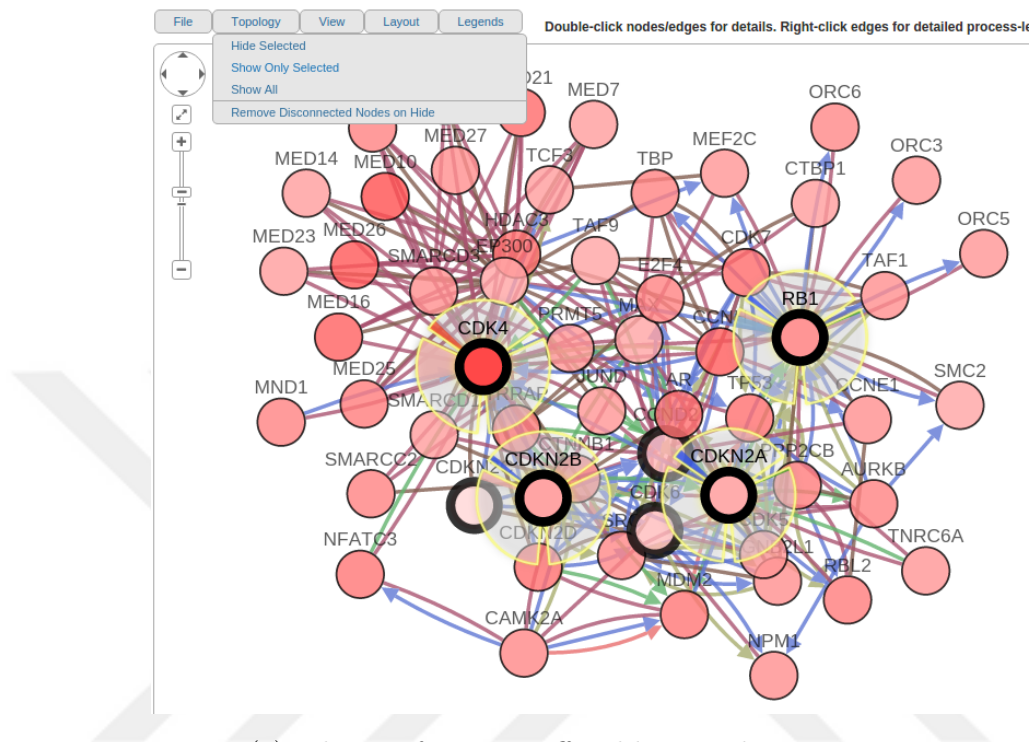
Figure 3.9: Detailed gene information of *MDM2* gene in SBGN view

3.3 Complexity Management Operations

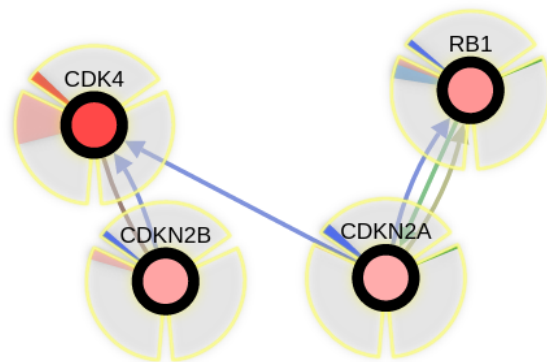
As the number of nodes, interactions along with the genomic data increase in visualized networks, utilization of the information represented in such networks decrease with the growing number of nodes and interactions, valuable information might become unnoticeable and obscured. In this respect, we provided some complexity management operations for the visualized networks to efficiently utilize and focus content of interest in both neighborhood networks and process description networks.

3.3.1 Complexity Management Operations for Gene Interaction Pathways

In the context of visualized neighborhood pathways along with genomic data. We provide node filtering operations to manage the complexity by the growing number of nodes in such networks. Topology related filtering operations of nodes can be found in the menubar above the visualization canvas under *Topology* menu item and in *Genes* tab of right panel. In the *Topology* menu, there exists functionality that controls the visibility of selected elements in the network such as showing only selected entities and hiding selected entities. These operations are handy to focus on content of interest in such networks. (Figure 3.10).



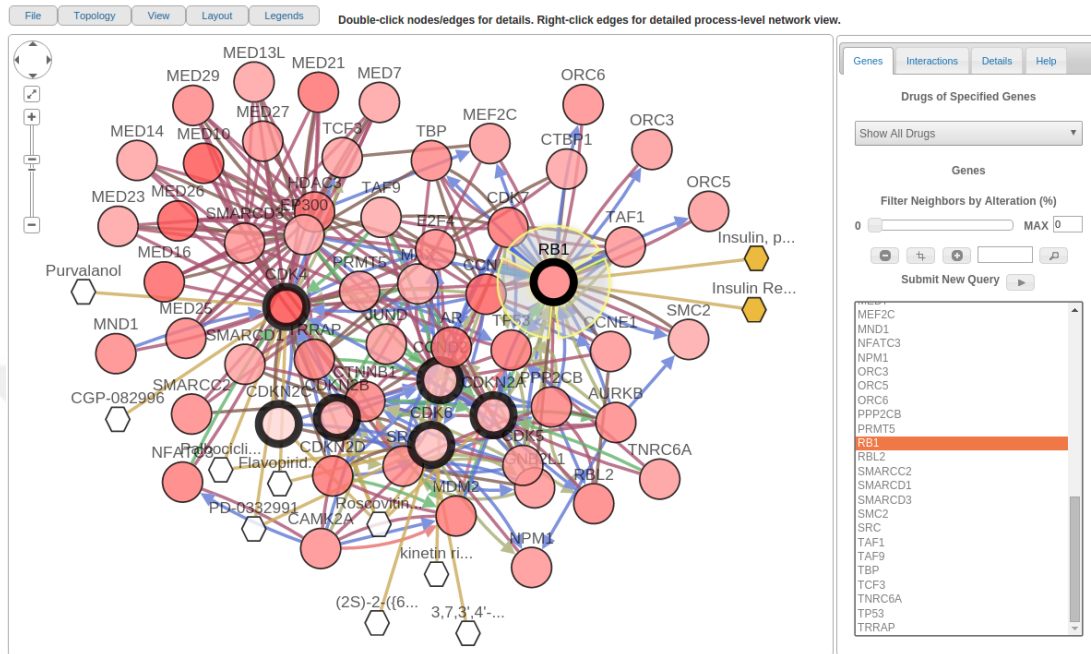
(a) Filtering functions offered by *Topology* menu item



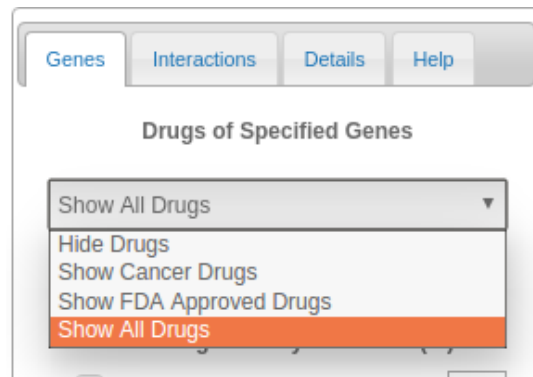
(b) Network in (a) after filtering operation of 'Show Only Selected' is performed

Figure 3.10: Topology related filtering operations

We provide another set of complexity management operations regarding filtering of nodes in *Genes* tab that is located in the panel right to the visualization canvas. Drugs that are targeting genes in the network are filtered out by default but user can change the visibility of drugs from the drop down list in *Genes* tab (Figure 3.11b).



(a) Filtering functions offered by *Genes* tab.



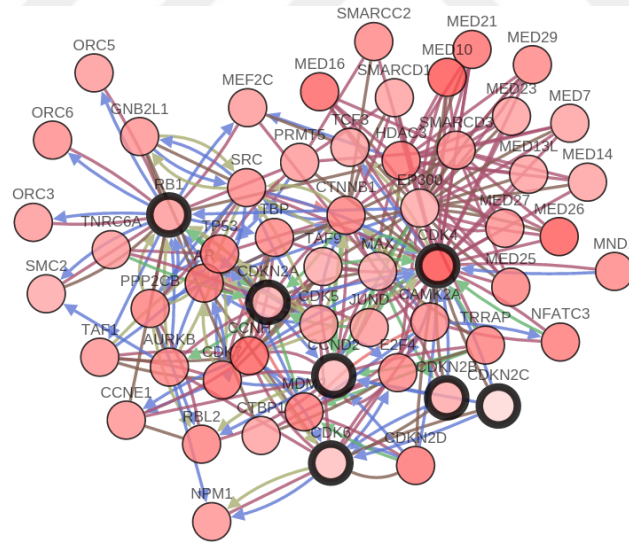
(b) Drug filtering operations.

Figure 3.11: Filtering operations in 'Genes' tab of network visualization tool in cBioPortal

Genes tab provides complexity management operations regarding pruning the neighbour of seed genes in the network by their alteration frequency. Additionally, all the gene names are listed in this tab as an overview with a search capability. Topology related filtering operations that exists under *Topology* menu are also included in this tab for easy access along with the functionality of performing

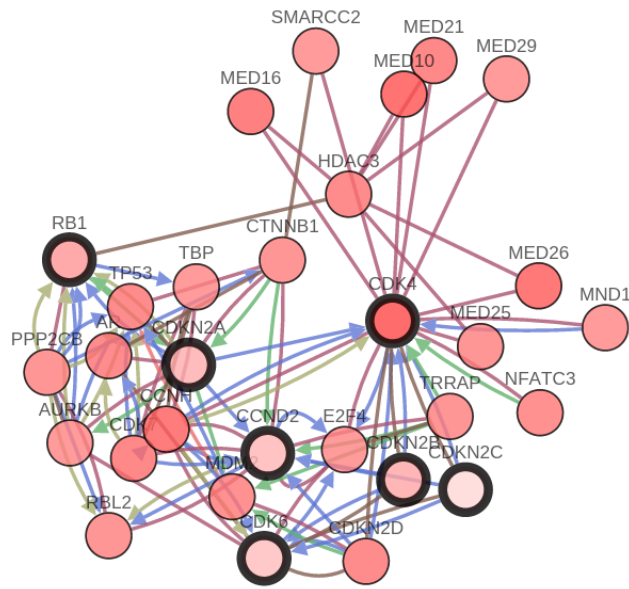
a query to obtain new neighborhood pathway according to selected genes in the gene list (Figure 3.11a).

Pruning the network according to the alteration frequencies can be performed using the slider located under ‘Filter Neighbors by Alteration (%)’ label in *Genes* tab. Bounds of this slider is defined by the minimum alteration percent, which is 0 by default, and the maximum alteration percent value of neighbours of seed genes in the network, that is denoted as ‘MAX’. Slider controls the visibility of neighbour genes by pruning the neighbor genes that have less alteration percent value compared to current slider value. The text box located on the right of ‘MAX’ label indicates the current alteration percent value that network is pruned accordingly (Figure 3.11a). An example pruning process is illustrated in Figure 3.12.

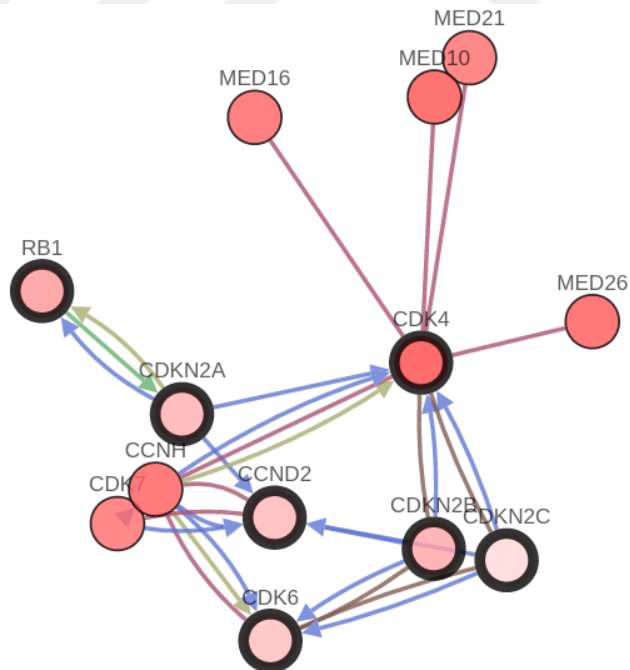


(a) Non pruned neighborhood pathway

Figure 3.12: Filtering out neighbours of seed genes by alteration percent



(b) Network is pruned according to the alteration percent value of 19.0% .



(c) Network is pruned according to the alteration percent value of 25.0% .

Figure 3.12: Filtering out neighbours of seed genes by alteration percent (cont'd)

There also exists highlighting functionality that aims to provide focused visualization to the neighborhood of selected nodes in the visualized networks. Associated operations can be found under ‘View’ menu item in the menu bar that is located above visualization canvas. A sample highlight operation is illustrated in Figure 3.13, here neighborhood of selected gene *CDK4* is highlighted by performing ghosting to the other nodes outside of the neighborhood of *CDK4*.

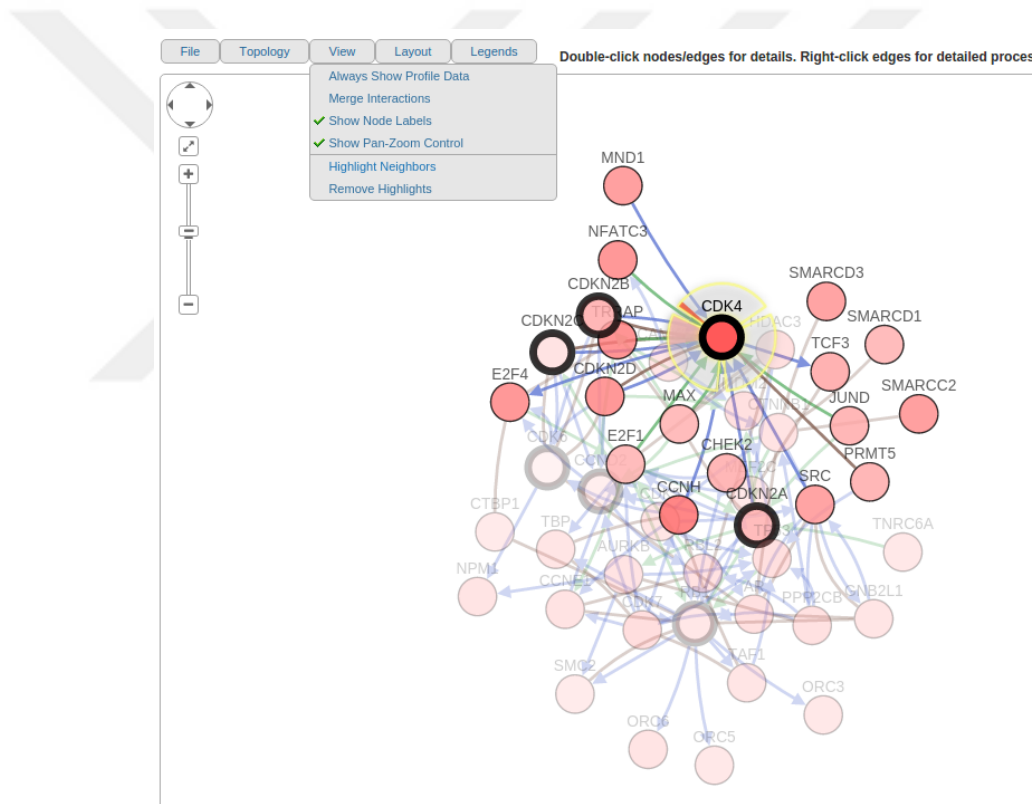


Figure 3.13: Highlight operation that focuses on neighborhood of *CDK4* gene.

Growing number of diverse interactions with more than one interactions between two genes makes it hard to utilize the genomic information and the relations of genes in the context of the visualized neighborhood networks. To manage the complexity that interactions bring, complexity management operations regarding interactions are provided in the *Interactions* tab (Figure 3.10a).

Interactions tab lists the interaction types that exists in the network along

with the list of sources where information about these interactions is taken. This tab offers filtering functionality according to type and source of interactions. (Figure 3.14).

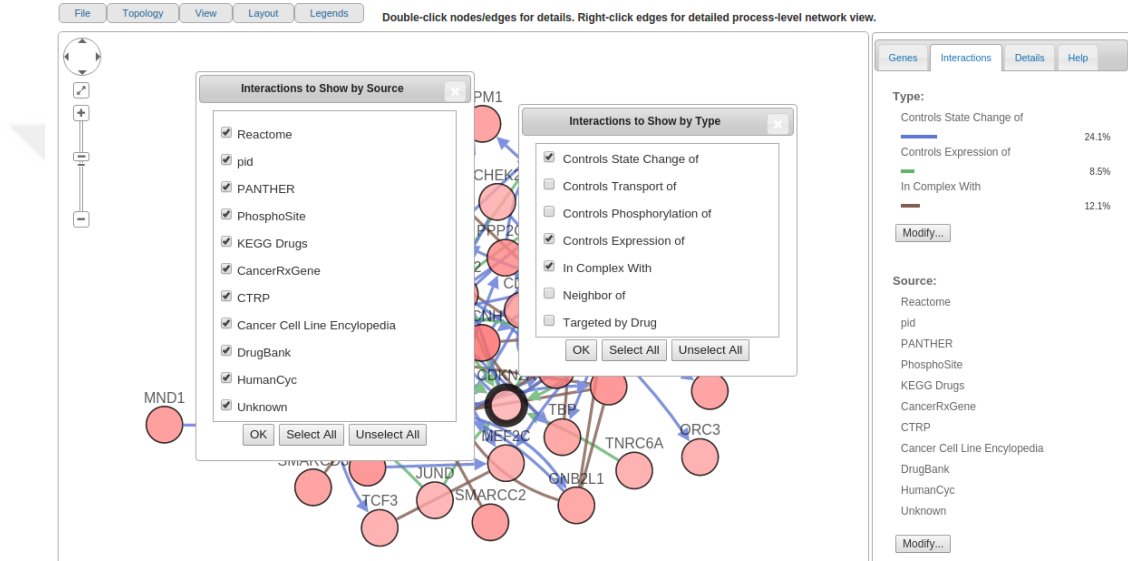
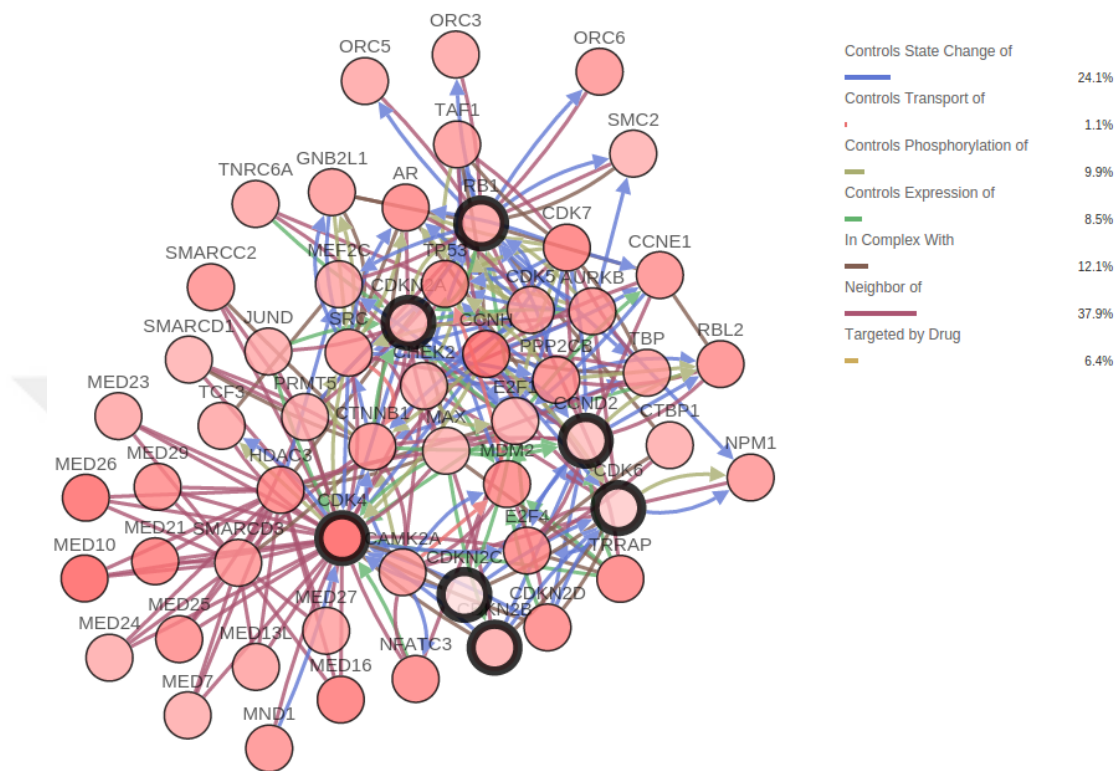
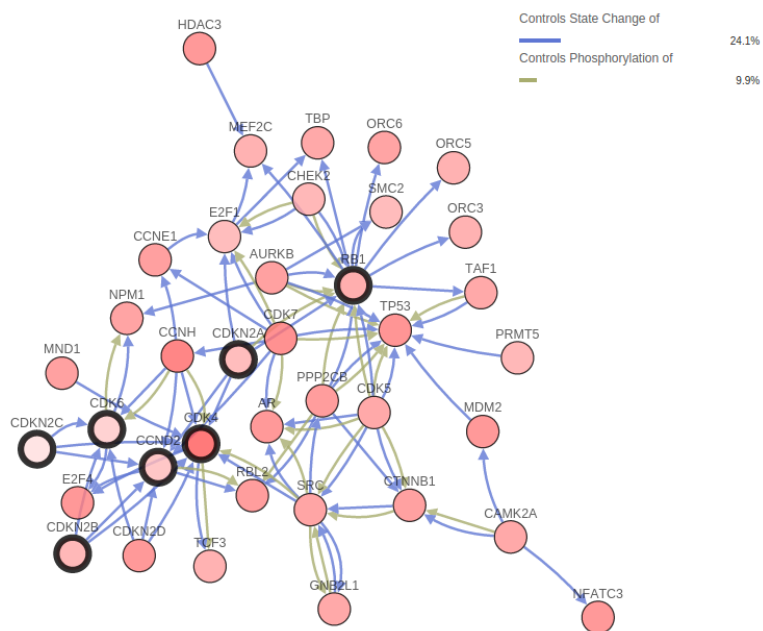


Figure 3.14: Filtering views by interaction type and source.

There are two buttons named as ‘Modify’ located at the end of the list of interaction types and sources. When clicked, these buttons show associated filtering views as shown in Figure 3.14. All of the interaction sources are visible by default. On the other hand, only the interactions with types of *Controls State Change of*, *Controls Expression of*, *In Complex With* are visible to provide a simpler and less dense initial network since there are lots of interaction types which might increase the level of complexity in visualized network when visualized at once.



(a) Network visualization where all available edge types are visible



(b) Same network where edges are filtered out by their type

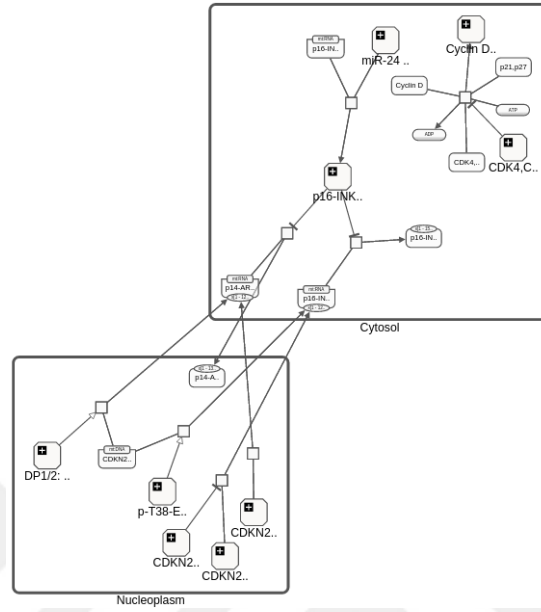
Figure 3.15: Interaction filtering by type.

3.3.2 Operations for Process Level Pathways

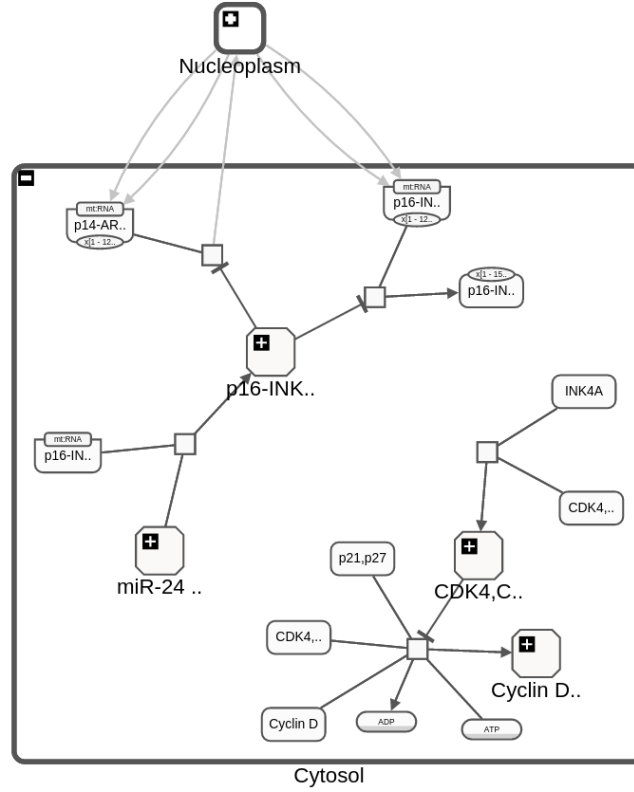
As discussed in section 3.2, process level details of an interaction between two genes in the neighborhood pathways are visualized in SBGN. These pathways are obtained from Pathway Commons web service. Obtained SBGN pathways include hierarchical structures such as molecular complexes and compartments as compound nodes in the obtained graph topology. SBGN pathways are compound graphs, whereas gene interaction pathways are flat graphs that do not include hierarchical structures. In this respect, increasing number of entities in such pathways along with the hierarchical structures bring complexity to the visualized networks which might obscure the information embodied in such SBGN pathways. To handle the complexity, operations of hiding, highlighting along with the expand, collapse utilities for compound structures are provided. To provide an efficient visualization for SBGN pathways in cBioPortal, an improved SBGN-Viz.js is constructed, enhanced by associated complexity management operations which will be discussed in this section.

One of the major complexity management operation that is provided is expand and collapse operations regarding compound structures of SBGN diagrams (*Complex* and *Compartment* structures). This utility is important since obtained process level pathways might include many compound structures with lots of entities in them, increasing the graph size and obscuring the valuable process level details of visualized pathways (Figure 3.16c).

Initially, *Complex* structures of the obtained SBGN pathways visualized in cBioPortal is collapsed to provide an effective initial visualization of such pathways (Figure 3.16a).

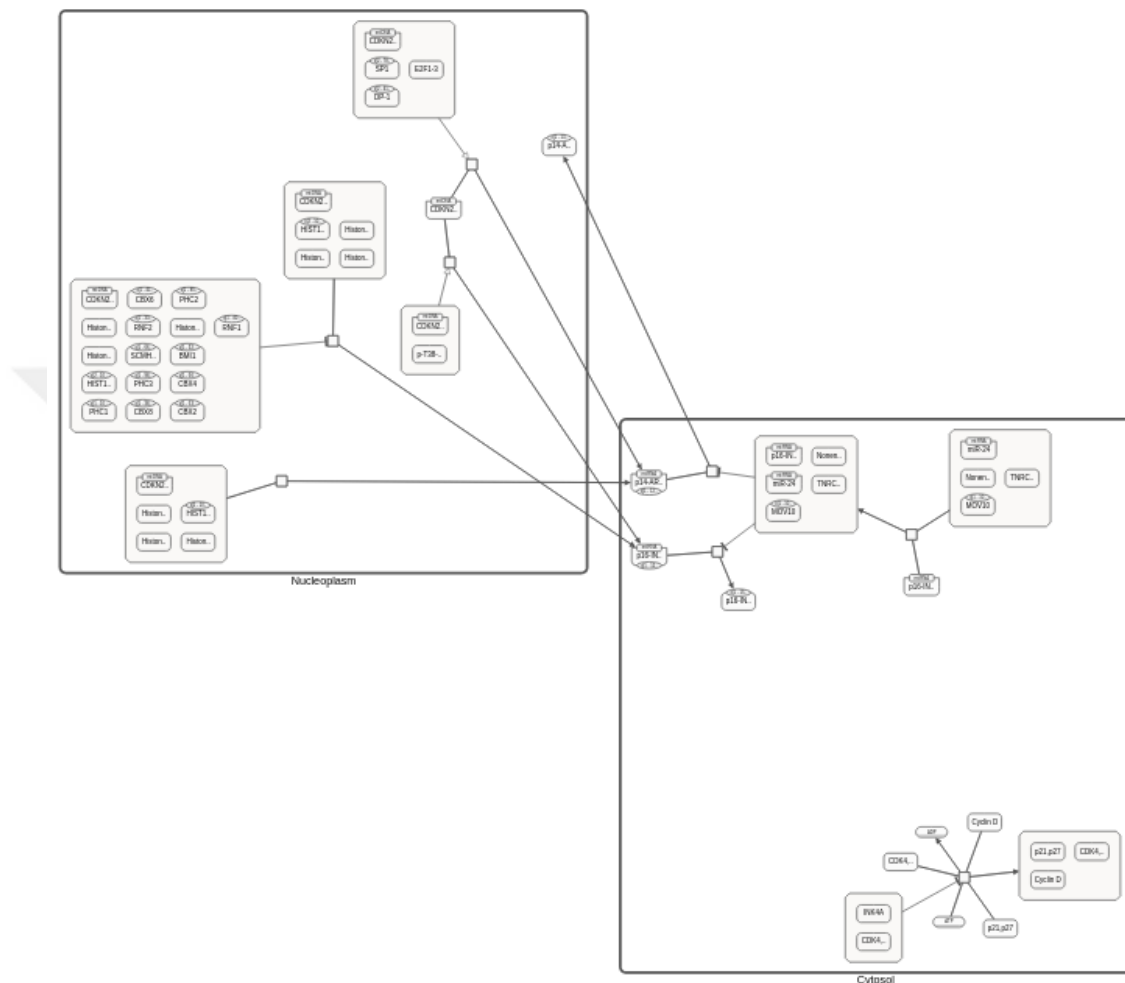


(a) Paths between *CCND1* and *CDKN2A* visualized in SBGN where all molecular complexes collapsed initially for a compact, intuitive view



(b) Same pathway in (a) where *Nucleoplasm* compartment is further collapsed

Figure 3.16: Expand collapse operations



(c) Same pathway in (a) where all molecular complexes expanded, resulting a complex view.

Figure 3.16: Expand collapse operations (cont'd)

Collapsing such compound structures by removing the child entities from visualized network and resizing these compound structures to a minimal dimension provides better usage of area and increases utilization of data. A collapsed node may include child nodes that have incoming and outgoing edges to the outside of such container compound node. In this case, incoming and outgoing edges of such child nodes of the container node which will be collapsed, are connected to the collapsed representation of container node to preserve the interaction relations (see *Nucleoplasm* compartment in Figure 3.16a and Figure 3.16b for reference).

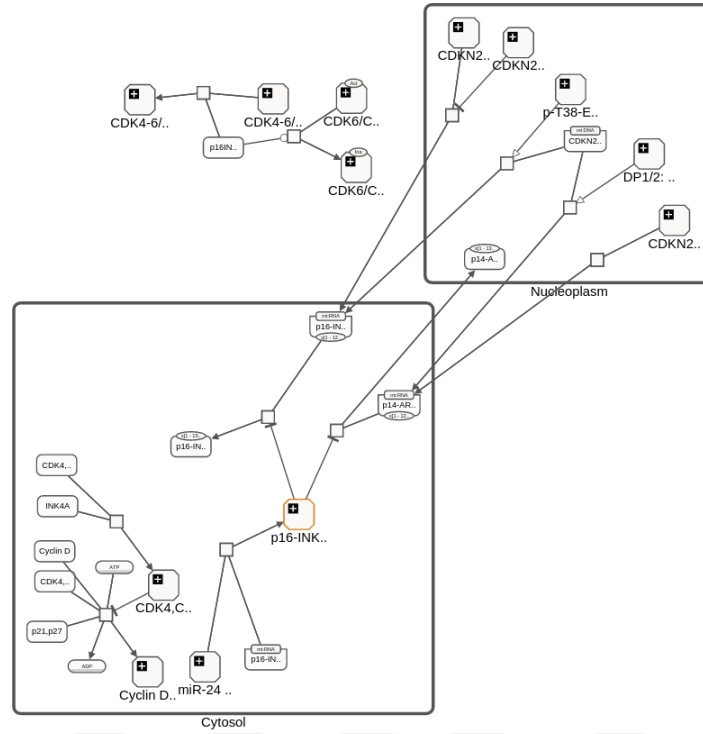
Expand operation undoes effect of collapse operation by adding child entities to the visualized network again, resizing compound structures according to their child entities and connecting the interactions to the actual target nodes.

It is possible to collapse and expand a clicked compound structure from canvas. A plus sign on top left corner of compound structures is rendered to indicate that this compound can be expanded. Similarly, a minus sign is rendered on the top left corner of compound structures to indicate that this compound can be collapsed (Figure 3.16b). Moreover, utility operations that expand or collapse all of the compound structures at once are also included.

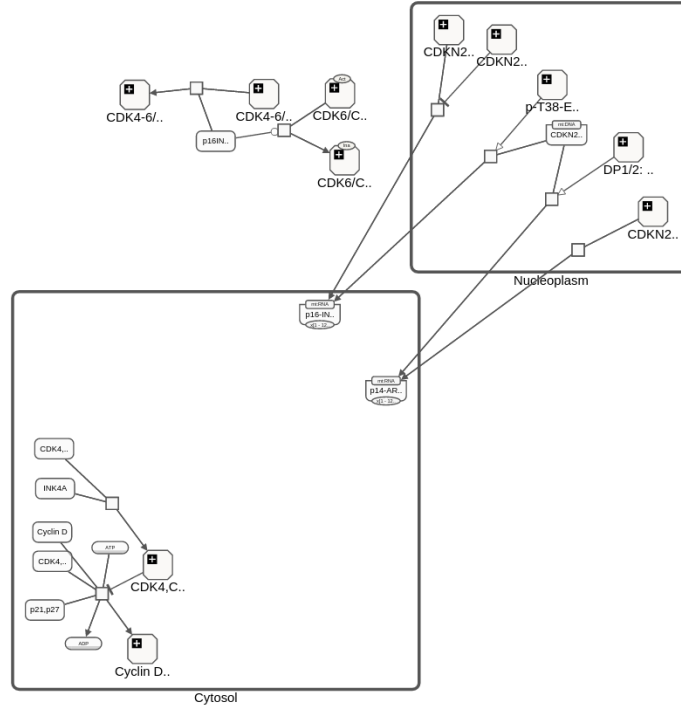
Highlighting and hiding operations also provided in this view to manage the complexity as in the neighborhood pathways drawn in cBioPortal. Hiding and highlighting operations in SBGN view are more complex regarding SBGN specific constraints compared to the neighborhood pathways.

When a hide operation on an entity is performed in visualized SBGN network, the processes in which this entity participates also removed with all the other entities that participates only to the removed process. This is performed mainly because a process with missing entity becomes meaningless and might confuse the user (Figure 3.17).

Basic highlighting operation works as in neighborhood pathways. Basically entities other than neighbours of selected entity in visualized SBGN pathway is shadowed (Figure 3.18a). Since the focus of visualized networks in SBGN view is process level details, there is also an option to highlight the processes that selected entity participates, along with all connected entities of such processes are highlighted to emphasize the related process details (Figure 3.18b).

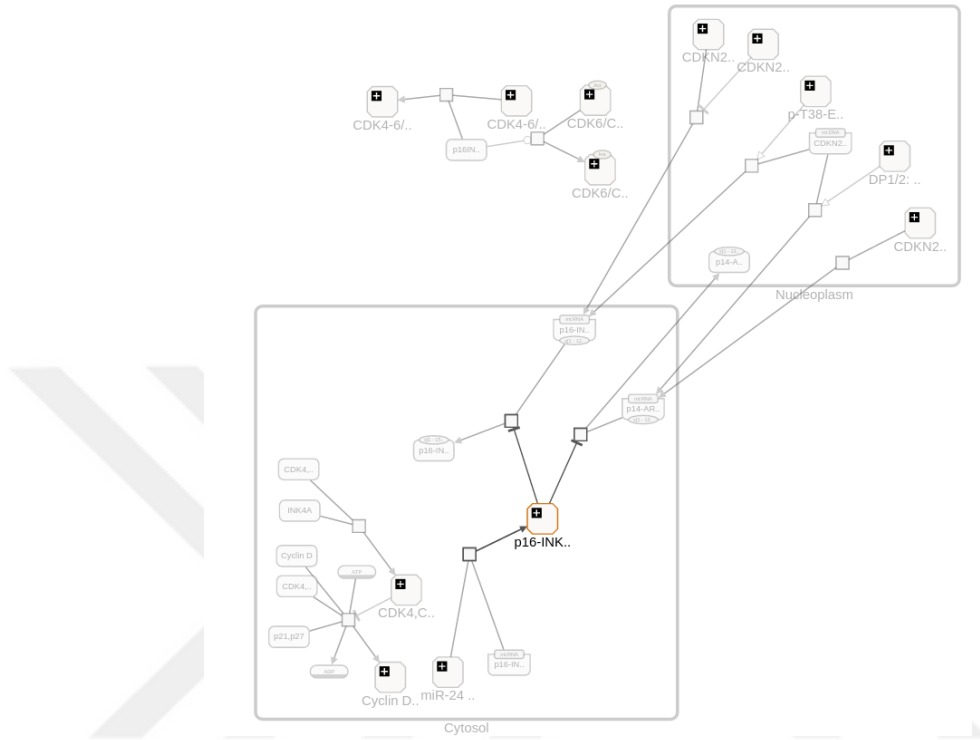


(a) Hiding operation of selected molecular complex from pathway

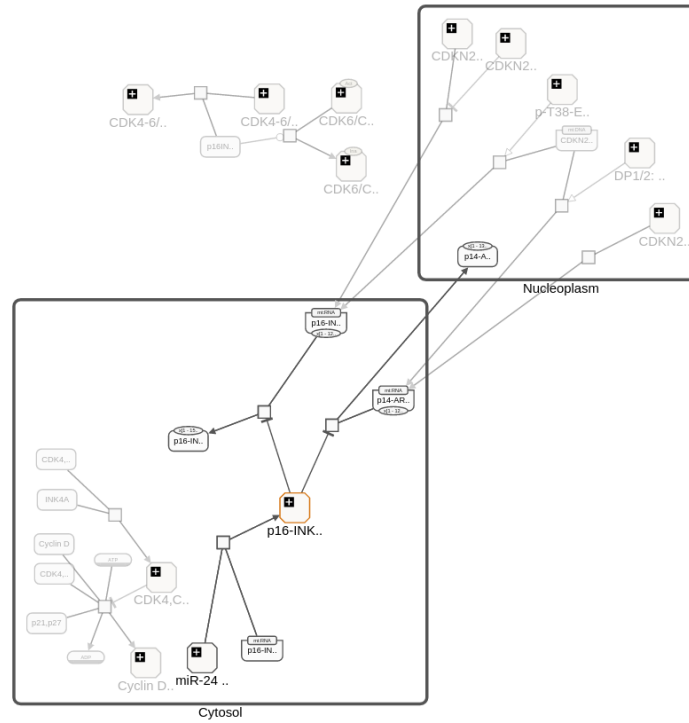


(b) Same pathway in (a) after selected molecular complex is hidden

Figure 3.17: Hide Operation.



(a) Highlight neighbour operation.



(b) Highlight processes operation

Figure 3.18: Highlight Operation.

3.4 Architecture

As discussed in 2.3.6, cBioPortal is an open source web application that provides graphical summaries of gene specific information from multiple platforms, network visualization and analysis, survival analysis, patient specific queries, and software programmatic access. Due to its rich and diverse features, there are many modern web technologies that are being used in the development of cBioPortal. Main technologies used in cBioPortal are Java Server Pages (JSP) [54] and Java Servlet [55] as backend technology and JavaScript as front-end technology. JQuery [32], Backbone.js [56], Angular.js [57], Bootstrap [58] and many other small JavaScript utility libraries along with HTML and CSS are used in the front-end side of cBioPortal. Besides that, for the data storage, cBioPortal uses MySQL [59].

cBioPortal is a collaborative effort by many developers from various research groups and institutes. In this respect, development of diverse features of cBioPortal is developed separately, where each feature is designed and developed independently by the associated research group or institute and integrated to the cBioPortal.

In the context of our work, we focused on design and development of network visualization feature of cBioPortal. We designed associated software solutions in a way that resulting network visualization tool will be easy to integrate to the existing cBioPortal architecture and codebase.

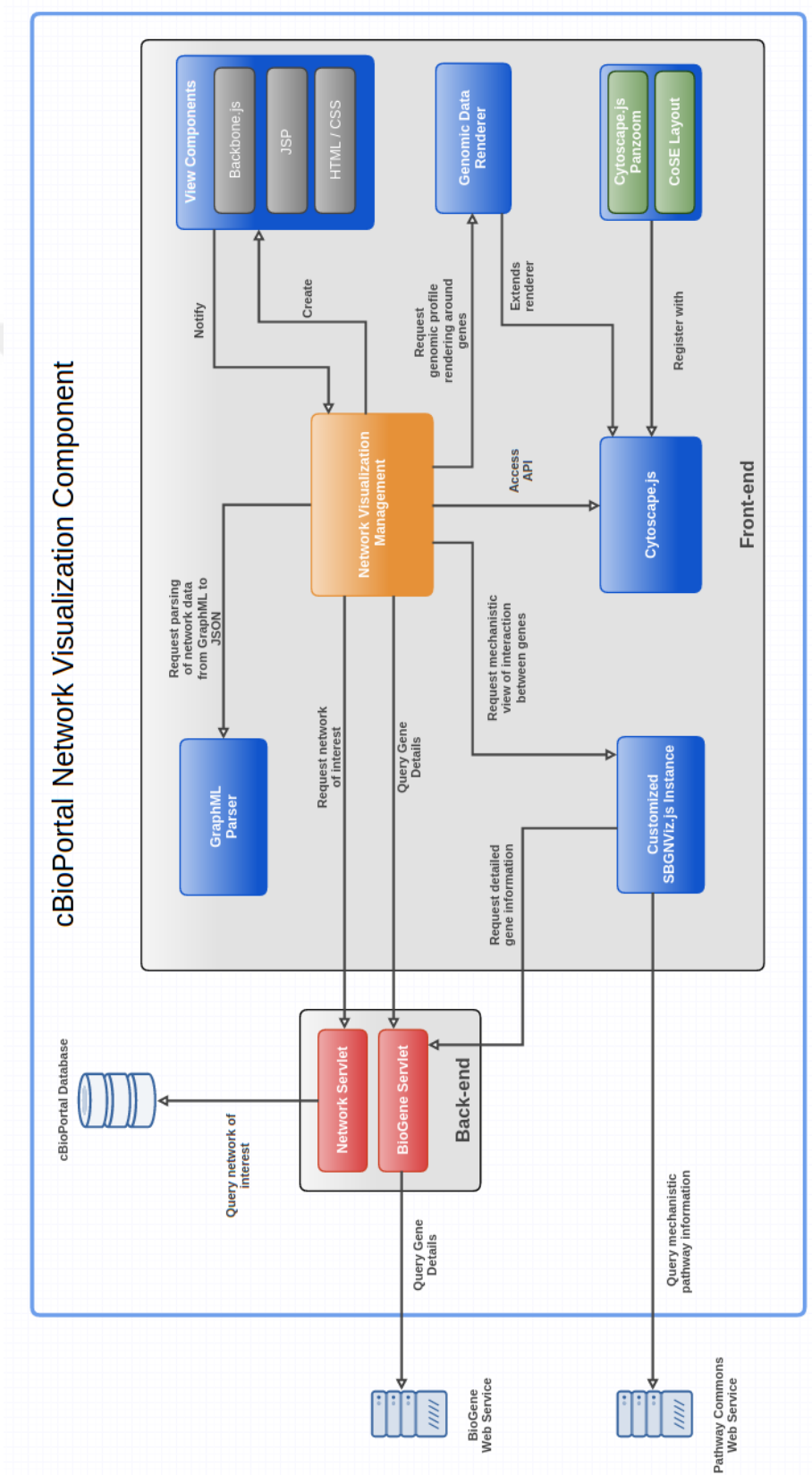


Figure 3.19: High level architectural design of network visualization component in cBioPortal.

As illustrated in Figure 3.19, Architectural overview of network visualization tool in cBioPortal mainly consists of back-end and front-end components as a typical web application.

In the back-end side, we have two components named as ‘Network Servlet’ and ‘BioGene Servlet’. These components are Java Servlets which are dedicated to respond to the requests emanating from front-end side. ‘BioGene Servlet’ communicates with BioGene service to fetch the detailed information about a gene of interest. ‘Network Servlet’ provides neighborhood pathway of a given set of genes of interest along with genomic data by querying the cBioPortal database which holds a copy of pathway information of Pathway Commons for fast response.

It can be stated that bigger portion of our work resides in the front-end side. We have a central management component called ‘Network Visualization Management’ which handles the business logic, dynamic content creation and event handling.

View related components of network visualization component consists of UI components that are implemented using standard web technologies of HTML, CSS along with JSP and Backbone.js. Handling of the events that are generated by view components and dynamic creation of such view components are handled by core controller ‘Network Visualization Management’.

There exists a utility component ‘GraphML Parser’. Back-end side of portal serves queried pathways in GraphML format [50] since it might be useful for the scientists to use exported pathways in other tools. Moreover, as discussed in Section 2.3.1, Cytoscape.js accepts JSON format for representing network topology that will be visualized. In this respect, ‘GraphML IO’ includes functionality in terms of conversion of network data from GraphML to JSON.

As core network visualization technology we are using Cytoscape.js. Along with the rich features of Cytoscape.js, we are using an extension for viewport manipulation features like zooming, panning etc. called ‘Cytoscape.js Panzoom’ [60]. Besides that, we are registering a graph layout algorithm called ‘CoSE Layout’ [61]

to Cytoscape.js for automatic graph layout. This algorithm is exact JavaScript counterpart of CoSE layout algorithm which is provided as part of Java Library ChiLay that is discussed in 2.3.4.

Although Cytoscape.js provides rich features regarding network visualization and analysis, it did not meet the requirements of our work in terms of rendering genomic alteration data along with visualized networks. To provide such rendering capabilities, we implemented an extension renderer component ‘Genomic Data Renderer’ and registered it with the Cytoscape.js. This renderer basically extends the node rendering function in the Cytoscape.js core in a way that it performs additional rendering regarding genomic alteration data after the original node rendering process in Cytoscape.js core.

All the components discussed so far is about visualization of neighborhood of genes of interest in the context of specific cancer study with genomic alteration data. To support this view, we also provided a secondary view that is dedicated to represent process level details of an interaction between two genes of interest in SBGN-PD. We integrated an improved and customised version of SBGNViz.js to our architecture for this purpose. Such SBGNViz.js instance is initialized by ‘Network Visualization Management’ component whenever the user wants to see this complementary view. Once initialized, this instance makes a live query to the Pathway Commons web service to obtain corresponding pathway information in SBGN-ML format and visualizes associated pathway in SBGN-PD.

Chapter 4

TCGA Pathway Curation Tool

A quick review of the literature yields TCGA pathways are represented in a conventional way that is more intuitive and attractive for the researchers of the associated publications than the ones using the standard notation such as SBGN (Figure 4.1).

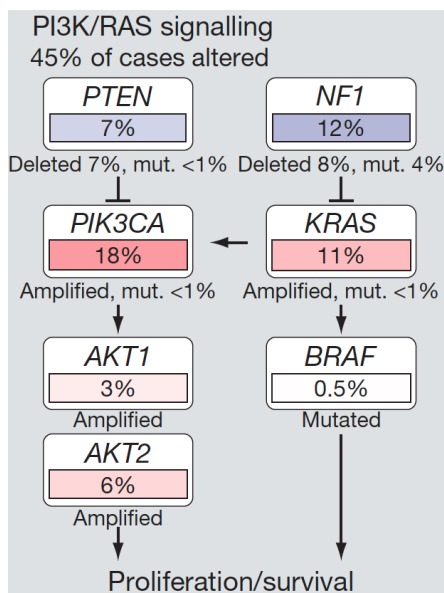


Figure 4.1: Visualization of PI3KRAS signalling pathway in the context of high-grade serous ovarian cancer (HGS-OvCa) which is provided as part of a TCGA research article [5].

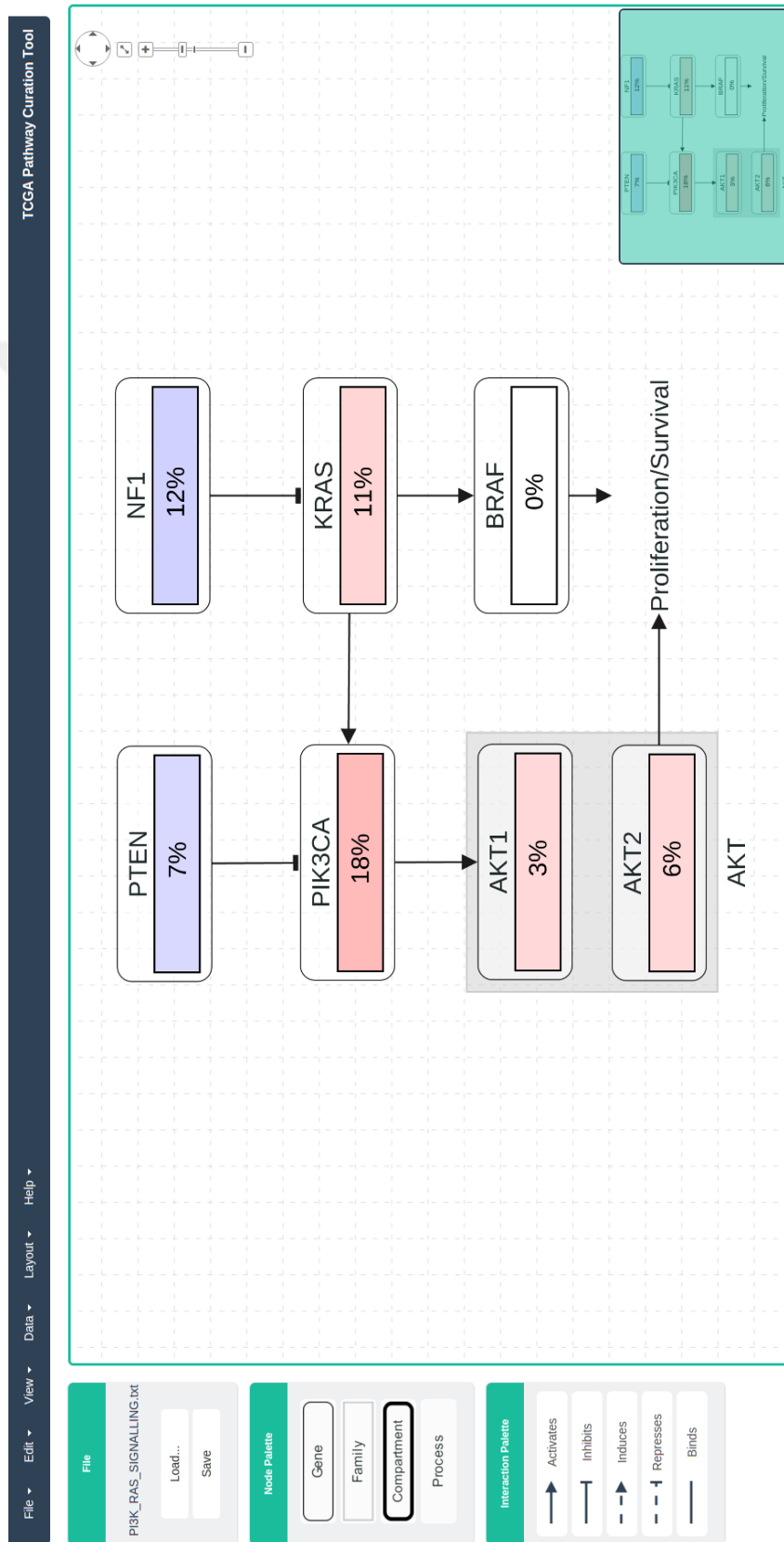


Figure 4.2: Pathway in Figure 4.1 curated and visualized by our tool along with the genomic data representation.

In this respect, we developed a new web based pathway visualization tool, specifically for cancer pathways, using a uniform notation based on the conventions found in TCGA cancer publications along with the overlay of experimental data. This tool facilitates interactive curation of such TCGA pathways from scratch with support for collaborative editing (Figure 4.2).

4.1 Visualization of TCGA Pathways

To capture the conventional visual characteristics in TCGA pathways, a visual notation was proposed for representing entities occurring in such TCGA pathways. According to this notation, node types are defined as follows:

- Genes: A simple node represents a single gene in pathway.
- Family of Genes: A compound node that is used to represent gene families in the pathway which are combined inside such node.
- Process: Represents a part of a pathway whose details are to be hidden under this node, represented as a compound node.
- Compartment: A cellular location in cell such as nucleus or cytoplasm, represented as a compound node.

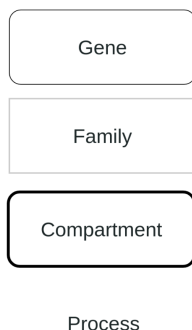


Figure 4.3: Visual representation of node types in TCGA curation tool

Edges that represents the interactions between entities are defined as follows:

- Activates: Source gene activates target gene, line with arrow at its end.
- Inhibits: Source gene inhibits target gene, line with T symbol at its end.
- Induces: Source gene induces target gene, represented by dashed line with arrow in its end.
- Represses: Source gene represses target gene, represented by dashed line with T symbol at its end.
- Binds: Source gene binds target gene, represented with standard line.

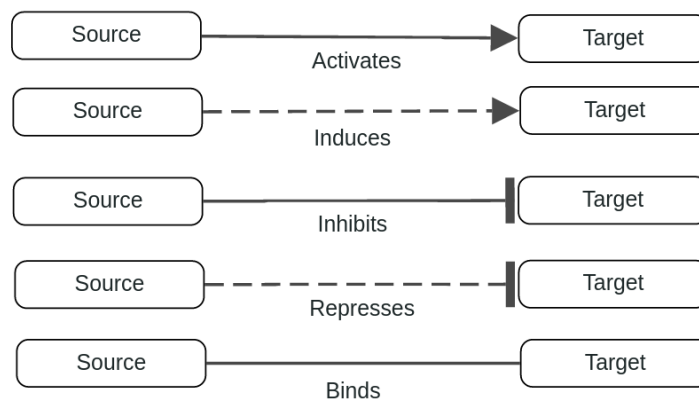
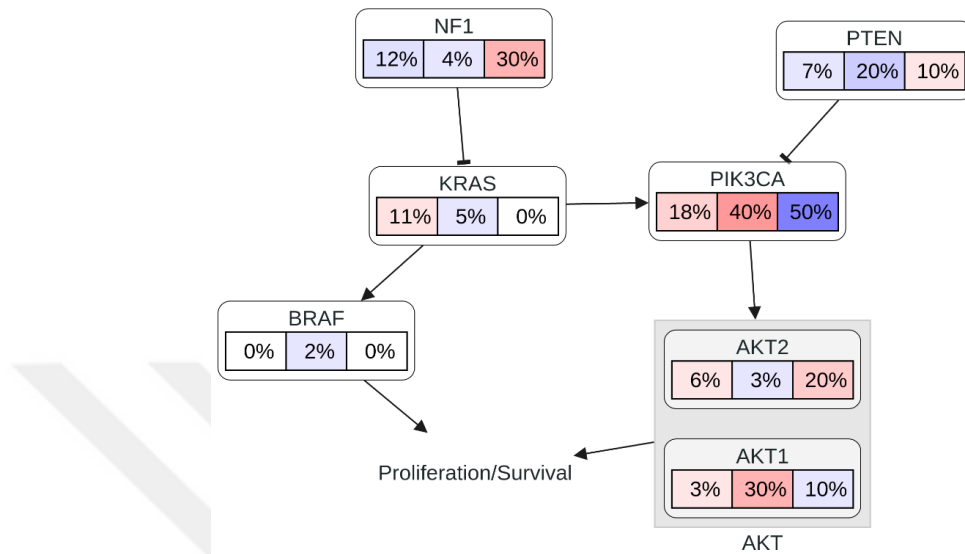


Figure 4.4: Visual representation of interaction types in TCGA curation tool

TCGA pathway curation tool facilitates overlay of genomic data regarding frequencies of activation and deactivation of a gene in the context of a specific cancer study. Activation frequencies are indicated using a white to red color scale, whereas deactivation frequencies are indicated using white to blue color scale. Genomic data is overlaid inside the boundaries of each visualized gene by additional rendering operation. The tool is capable of importing genomic data and visualizing such data. The data can consist multiple genomic frequencies of genes in different types of cancer (Figure 4.5a). It is also possible to choose the data sets to visualize from the loaded genomic data (Figure 4.5b).



(a) Genomic data from three different cancer types is overlaid on visualized network

×

Genomic Data Explorer

Data Set To Show:

☒ cancer1
☒ cancer2
☒ cancer3

Save Changes

(b) Three cancer types overlaid on genes of the pathway above. Note that cancer types listed here are visualized from left to right on the associated genes in the pathway

Figure 4.5: Genomic data overlay features of TCGA pathway curation tool

In addition to the visualization capabilities, the tool features import and export of curated pathway topology in a simple tab delimited file format (Figure 4.6a), overlay of genomic data and import of genomic data that is represented in a simple tab delimited format (Figure 4.6b), export of visualized pathways in JPEG and PNG formats, interactive addition and removal of pathway elements, view related vertical or horizontal alignment operations of selected elements, automatic graph layout support and help content that are included in the menubar above visualization canvas (Figure 4.2).

--NODE_NAME	NODE_ID	NODE_TYPE	PARENT_ID	POSX	POSY--
TP53	n1	GENE	-	100	100
MDM2	n2	GENE	-	120	100
AKT	n4	FAMILY	-	100	150
AKT1	n5	GENE	n4	100	160
AKT2	n6	GENE	n4	100	170

--EDGE_ID	SOURCE	TARGET	EDGE_TYPE
e1	n1	n2	ACTIVATES
e2	n2	n4	BINDS

(a) Representation of a sample graph topology that is exported from the tool

gene	cancer1	cancer2	cancer3
AKT1	-7	-20	10
AKT2	-12	-4	30
TP53	18	40	-50
MDM2	18	40	-50

(b) Representation of genomic data in a simple tab delimited file format

Figure 4.6: Representations of graph topology and genomic data that the tool accepts in simple tab delimited format.

Import and export functionality of curated pathway graphs along with interactive addition of pathway entities are provided, through the toolbar on the left of visualization canvas for easy access. Interactive node addition can be performed by dragging elements of 'Node Palette' and dropping them to canvas. When user selects an interaction type from 'Interaction Palette', tool automatically switches to the interaction addition state. If the user drags the mouse pointer from source node to target node while left mouse button is pressed, a new interaction between source and target node is created as a result of this action. Interactive edge addition mode can be turned off by clicking the selected interaction type from 'Interaction Palette' again. These interactive addition operations can also be found under 'Edit' menu item in the menubar.

To increase the effectiveness of interactivity, we provide some useful topology related operations as context menus that appear when a right click is performed on canvas, edges or nodes. A different context menu appears according to the element under the mouse pointer when a right click is performed (Figure 4.7). According to the selected element from such menus the topology of the visualized pathway is changed.

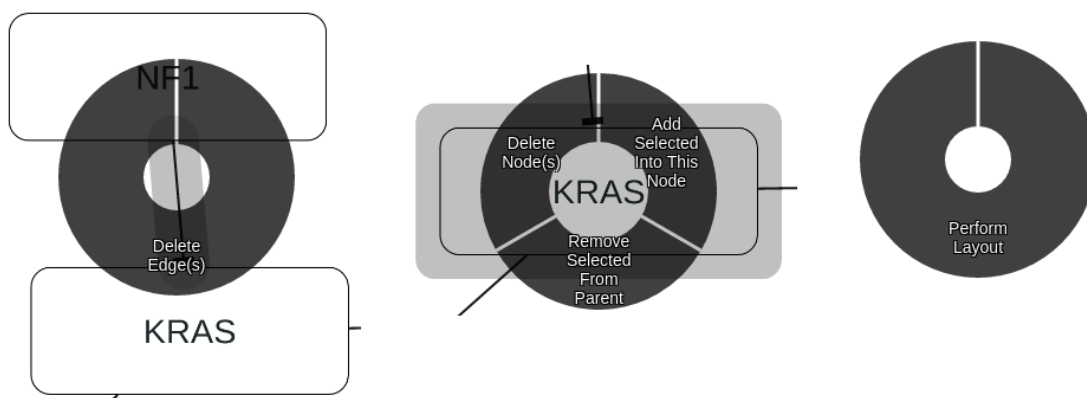


Figure 4.7: Different context menus of the TCGA pathway curation tool. From left to right: edge context menu, node context menu and root context menu is visualized. Note that root context menu appears when a right click is performed on the canvas where no node or edge exists under mouse pointer.

Design of the tool provides two choices of usage for the users. One is ‘Local’ usage, which lets user to create pathways individually, whereas ‘Collaborative’ usage lets user to create a shared pathway that can be created collaboratively by multiple remote users. The tool uses Google Real Time service [15] for collaborative usage; therefore, Google Drive authentication is required to use the tool in collaborative mode. In local usage, TCGA pathway curation tool is instantiated as an ordinary web application where the application data of the tool is locally stored. On the other hand, collaborative usage creates a shared data model that represents current graph topology, imported genomic data in a shared manner so that edits of different collaborators modify this shared data and when synchronized among all collaborators, same view of curated pathways will be obtained for all collaborators. Collaborative curation facility of the tool will be discussed in a separate section later in this chapter.

4.2 Architecture

TCGA pathway curation tool is a web based application that consists front-end and back-end components (Figure 4.8). In the back-end, there exists a Node.js [62] based server side component that serves the tool and includes basic back-end operations. TCGA pathway curation tool provides client side visualization and curation facilities in the front-end part of its architecture that are purely implemented in JavaScript (except view related components).

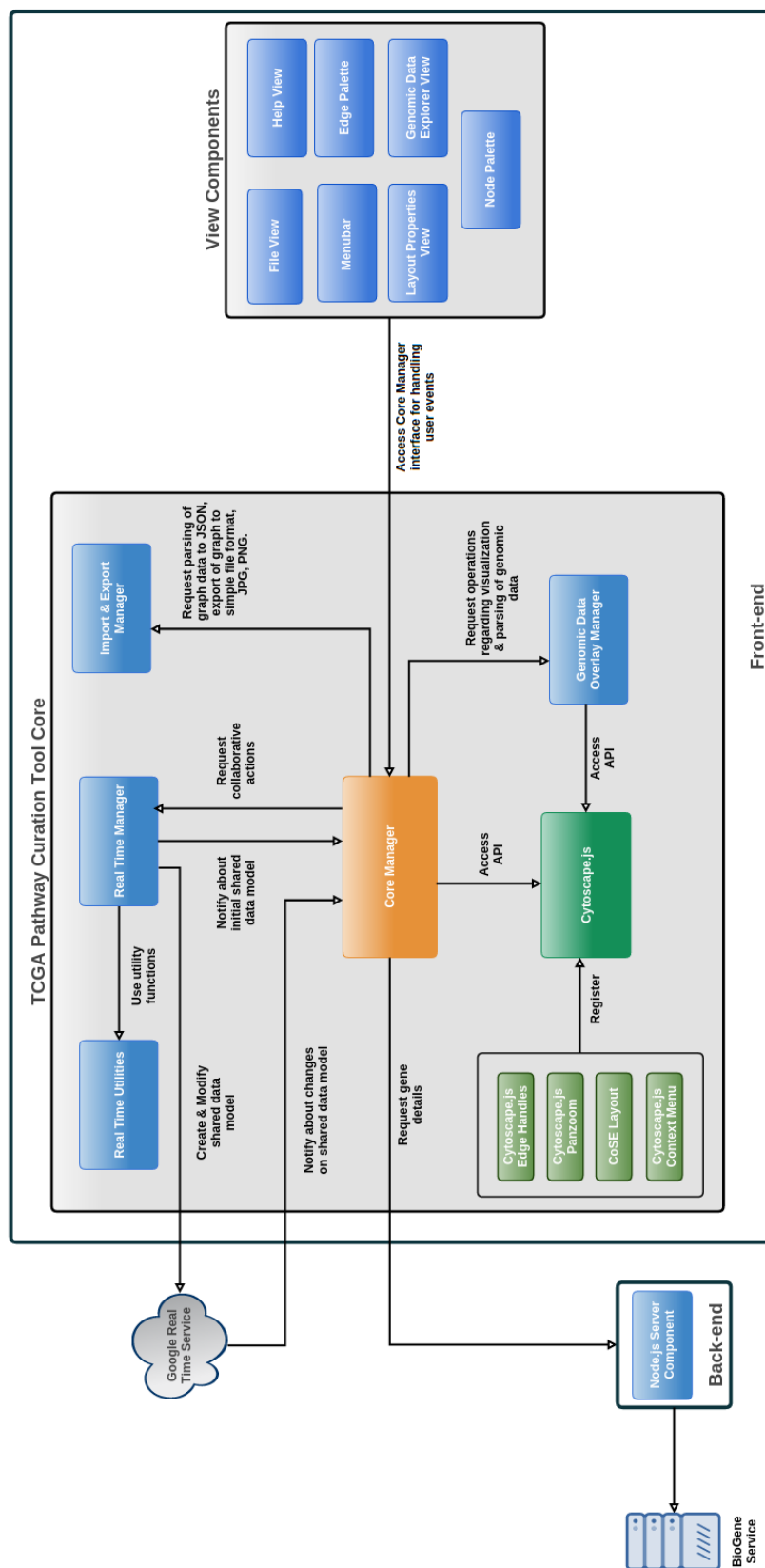


Figure 4.8: High Level Architecture of TCGA Pathway Curation Tool

TCGA pathway curation tool uses Cytoscape.js as a core graph visualization technology. Cytoscape.js extensions of ‘Cytoscape.js Edge Handles’ for interactive edge addition, ‘Cytoscape.js Panzoom’ for viewport related operations such as zooming and panning, ‘CoSE Layout’ for automatic graph layout, ‘Cytoscape.js Context Menu’ for context menu support in visualization canvas, are used in the tool.

TCGA Pathway Curation tool consists of two main parts of ‘View Components’ and ‘TCGA Pathway Curation Tool Core’. ‘View Components’ include UI related components that are implemented using HTML, CSS and popular Backbone.js framework. Each component in ‘View Components’ provides its own event handler for the associated view. These handlers use the interface provided by the ‘Core Manager’ to perform desired actions.

‘TCGA Pathway Curation Tool Core’ consists related components that handles the all the business logic of the tool. ‘Core Manager’ component provides an interface to outside of the core so that components outside of the core only communicates with ‘Core Manager’ for required actions using the Façade design pattern. The ‘Core Manager’ then fulfills the required action or communicates with the responsible manager component in the core, to perform the required action. This minimizes the coupling between the core and view related components and increases cohesion in the core which provides better architecture facilitating the re-usability of the core component.

‘Import & Export Manager’ provides functions regarding parsing graph topology from a simple tab delimited format to JSON format that Cytoscape.js accepts. This tool also provides conversion from JSON format to such simple tab delimited format that makes it possible to export the visualized graphs.

‘Genomic Data Overlay Manager’ facilitates parsing, maintaining and rendering of genomic data. Once the genomic data is loaded by user, this component stores the data and generates the visual representation of genomic data programmatically for each gene in SVG format [63]. It then, requests Cytoscape.js to

render such newly genomic overlays inside the boundaries of genes. This component also facilitates choosing of genomic data to be visualized on the genes from the imported genomic data set, which immediately takes effect on visualized pathway.

As stated earlier, the tool can be used individually or collaboratively. In terms of individual usage, 'Core Manager' component directly performs requested actions to reflect the effects immediately on the in-memory graph topology, genomic data etc.

In the case of collaborative usage, the functions provided by the 'Core Manager' interface acts differently compared to standard individual usage. In this respect, if the tool is in collaborative usage mode, to fulfill the requested action, 'Core Manager' requests associated change from 'Real Time Manager' on the shared data model of the tool which residing in Google Real Time Service. These changes on shared data model later notify the 'Core Management' component of each instance of the tool that is running on the browsers of different collaborators to synchronize the changes among collaborators.

Google Real Time service stores the shared data model of the tool in Google Drive folder of each collaborator, which requires utility functions like Google account authentication. These functions are provided as part of 'Real Time Utilities' component.

4.3 Collaborative Usage

As discussed in previous sections, the tool facilitates individual and collaborative creation of pathways. In the individual usage, an instance of the TCGA pathway curation tool is instantiated as a normal web application, where actions that user performs directly changes in-memory graph topology, genomic data, layout options etc. On the other hand, when the user chooses to use collaborative pathway creation mode, user actions are not directly reflected to the local in-memory

data. Instead, required changes are performed on corresponding objects in the shared data model. Changes on the shared data model, trigger callbacks that notify each instance of tool running on each collaborator's browser. According to the different notifications, each instance of the tool synchronizes associated local, in-memory model of graph topology, genomic data, layout properties, some global options such as zoom level and pan level of viewport (Figure 4.9).

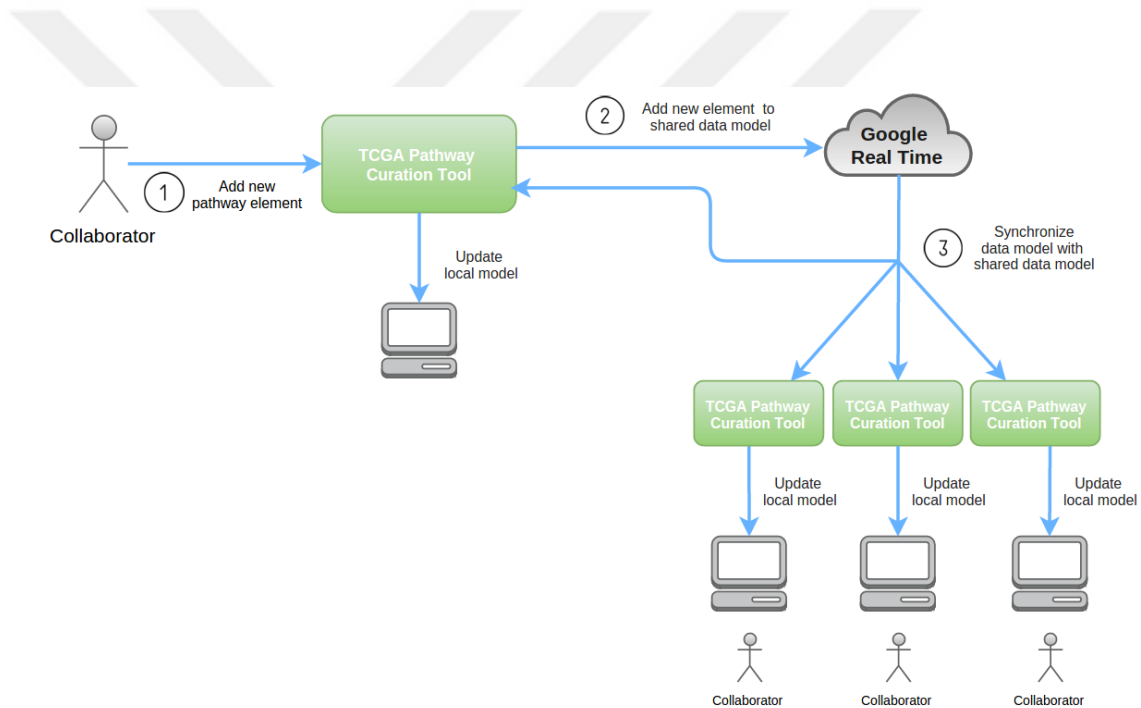


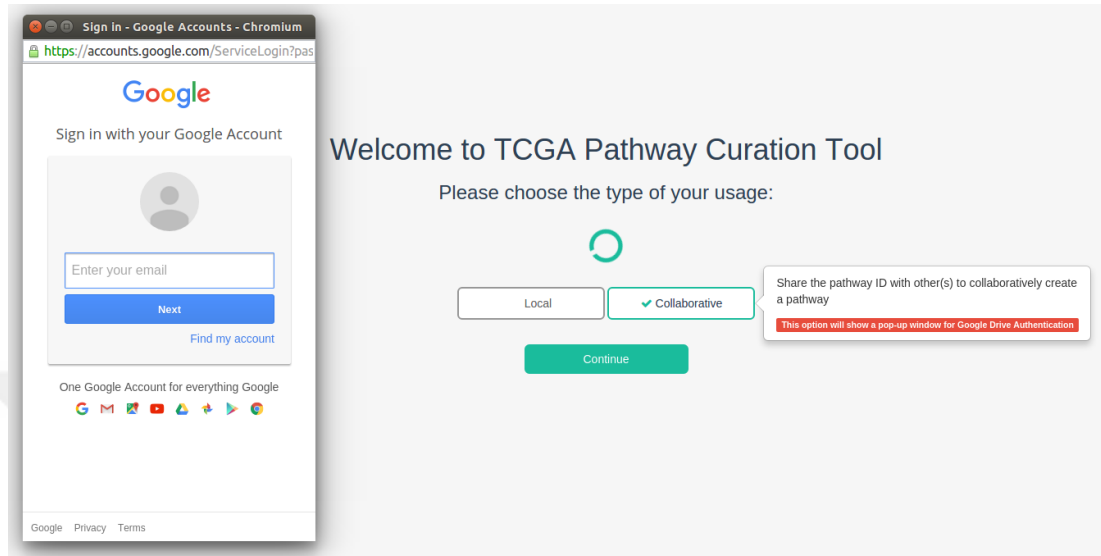
Figure 4.9: Flow of events regarding addition of new elements in collaborative usage mode

To create a shared pathway, the tool first prompts the user for Google account authentication since the shared data model will be stored in a shared document at Google Drive folder of the user. When the tool is initialized successfully after Google account authentication, a shared document is created in user's Google Drive folder. This shared document can be accessed and modified by any other instance of the tool that accesses to such link.

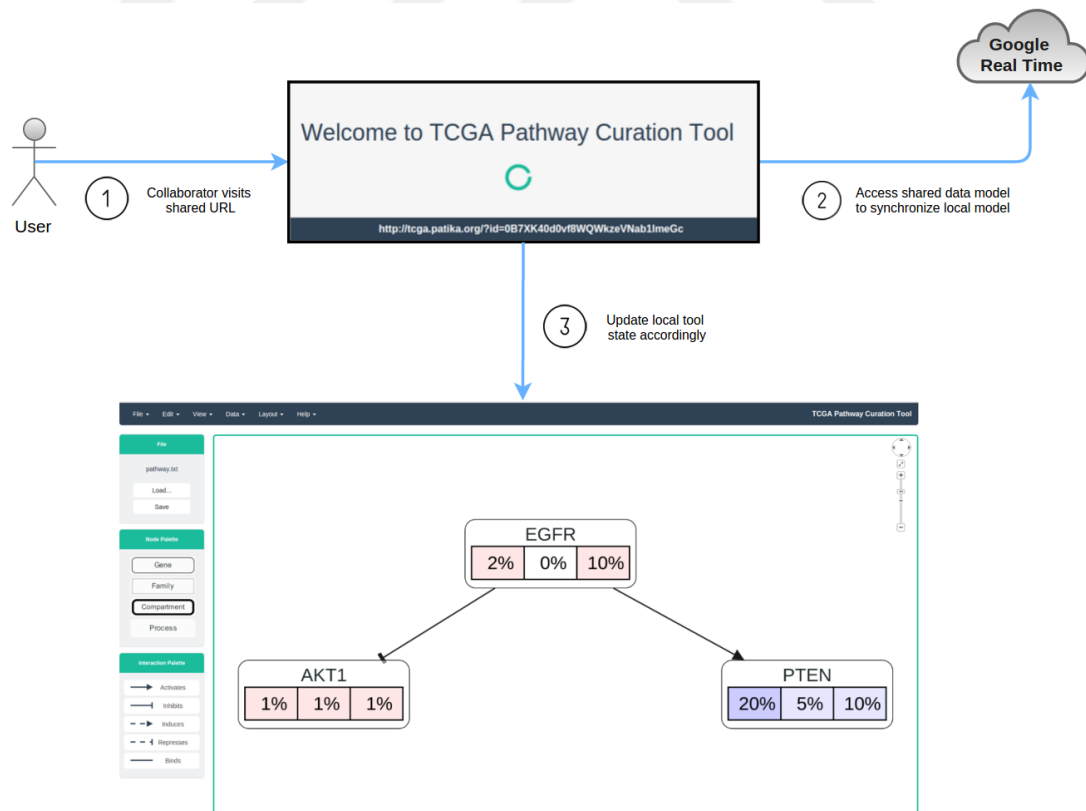
Initial shared data model which stores graph topology, genomic data, layout

properties and some global state information such as zoom level, pan level of the viewport, is constructed when such shared document is created for the first time. A unique id is generated for the associated shared document by Google Real Time service and the tool generates a unique URL using this id to represent a shared pathway that can be collaboratively curated by other collaborators when they visit this URL.

When other collaborators visit a shared pathway URL that is generated by the tool, the instance of the TCGA pathway curation tool running on the browser of visitor accesses the shared document whose id is appended in the shared URL generated by the tool. The tool then, synchronizes local in-memory model according to the shared data model that exists in the Google Real Time service. At this point visitor of the URL becomes a collaborator on the shared pathway and changes of each collaborator are sent to the Google Real Time service to update the shared data model accordingly.



(a) TCGA pathway curation tool where collaborative usage mode selected. Google account authentication is required to use collaborative mode



(b) Flow of events regarding synchronization of the tool when a collaborator visits a shared URL

Figure 4.10: Collaborative initialization of TCGA pathway curation tool

Chapter 5

Conclusion

In this study, we designed and developed pathway visualization solutions to aid cancer research.

We developed a network visualization component for the cBioPortal web portal. This component makes it possible to observe and analyze gene to gene interactions of given set of genes of interest in the context of a specific cancer study with the genomic alteration information. Visualized networks correspond to the neighborhood pathway of given genes of interest. This component also provides useful complexity management features in such gene to gene interaction networks through filtering.

We also provided a supportive pathway visualization view regarding the molecular process level details of an interaction of interest between gene pairs from visualized gene interaction pathways. Such detailed process level views are visualized in SBGN-PD. To provide such SBGN visualizations, we integrated our previously developed SBGN-PD visualizer, SBGNViz.js, to the cBioPortal. We improved SBGNViz.js complexity management features with more advanced complexity management operations of expand and collapse. Using expand and collapse, users can simplify compound structures (molecular complexes and compartments) into single collapsed smaller nodes as desired. Such structures frequently occur in

SBGN pathways as obtained from Pathway Commons.

To retrieve the SBGN-PD pathways in SBGN-ML format, Pathway Commons web service was used. In the context of our work, CoSE graph layout was integrated to Paxtools library, used by Pathway Commons web service to convert the queried pathways from BioPAX representation to SBGN-ML representation. Such graph layout support is vital for the community, since automated graph layout for such pathways utilizes the associated information embodied in SBGN diagrams.

Finally, we developed a new web based pathway visualization tool named ‘TCGA pathway curation tool’ specifically for cancer pathways, using the conventional notation found in TCGA cancer publications along with the overlay of experimental data. More importantly, this tool lets remote users create a pathway collaboratively.

5.1 Future Work

As future work, experimental genomic data overlay facility could be added to complementary molecular process level details of interactions visualized in SBGN in cBioPortal.

TCGA pathway curation tool offers basic functionality for creation, interactive editing, import and export of TCGA pathways in a unified notation. To make it a full featured pathway curation and editing tool, functionality for undo-redo could be added. In addition, currently edge types offered by the tool are straight edges, edges with bend points could be more useful in terms of generating more effective, manageable pathways.

5.2 Availability

Source code of network visualization component in cBioPortal can be found in GitHub repository of cBioPortal [64]. Similarly, contributions to Paxtools can also be found in associated GitHub repository [65]. Finally, source code of TCGA pathway curation tool is developed and maintained by our research group and can be found in corresponding GitHub repository of our research group [66].

cBioPortal periodically updates its web portal, network visualization feature can be found in the latest release of cBioPortal [9]. In addition, TCGA pathway curation tool is periodically deployed and made available at [7].

Bibliography

- [1] E. G. Cerami, B. E. Gross, E. Demir, I. Rodchenkov, Özgün Babur, N. Anwar, N. Schultz, G. D. Bader, and C. Sander, “Pathway Commons, a web resource for biological pathway data,” *Nucleic Acids Research*, vol. 39, no. Database-Issue, pp. 685–690, 2011.
- [2] U. Brandes, *Drawing on Physical Analogies*, pp. 71–86. Springer, 2001.
- [3] E. Demir, M. P. Cary, S. Paley, K. Fukuda, C. Lemer, I. Vastrik, G. Wu, P. D’Eustachio, C. Schaefer, J. Luciano, F. Schacherer, I. Martinez-Flores, Z. Hu, V. Jimenez-Jacinto, G. Joshi-Tope, K. Kandasamy, A. C. Lopez-Fuentes, H. Mi, E. Pichler, I. Rodchenkov, A. Splendiani, S. Tkachev, J. Zucker, G. Gopinath, H. Rajasimha, R. Ramakrishnan, I. Shah, M. Syed, N. Anwar, O. Babur, M. Blinov, E. Brauner, D. Corwin, S. Donaldson, F. Gibbons, R. Goldberg, P. Hornbeck, A. Luna, P. Murray-Rust, E. Neumann, O. Reubenacker, M. Samwald, M. van Iersel, S. Wimalaratne, K. Allen, B. Braun, M. Whirl-Carrillo, K.-H. Cheung, K. Dahlquist, A. Finney, M. Gillespie, E. Glass, L. Gong, R. Haw, M. Honig, O. Hubaut, D. Kane, S. Krupa, M. Kutmon, J. Leonard, D. Marks, D. Merberg, V. Petri, A. Pico, D. Ravenscroft, L. Ren, N. Shah, M. Sunshine, R. Tang, R. Whalley, S. Letovksy, K. H. Buetow, A. Rzhetsky, V. Schachter, B. S. Sobral, U. Doğrusöz, S. McWeeney, M. Aladjem, E. Birney, J. Collado-Vides, S. Goto, M. Hucka, N. L. Novere, N. Maltsev, A. Pandey, P. Thomas, E. Wingender, P. D. Karp, C. Sander, and G. D. Bader, “The BioPAX community standard for pathway data sharing,” *Nature Biotechnology*, vol. 28, no. 9, pp. 935–942, 2010.

- [4] “Systems Biology Graphical Notation Reference Card.” <http://www.sbgng.org/images/4/4a/Refcard-PD.png>. (Accessed on 07/30/2016).
- [5] “Integrated genomic analyses of ovarian carcinoma,” *Nature*, vol. 474, pp. 609–15, 2011 Jun 30 2011.
- [6] “Information visualization - Wikipedia, the free encyclopedia.” https://en.wikipedia.org/wiki/Information_visualization. (Accessed on 08/14/2016).
- [7] “TCGA Pathway Curation Tool.” <http://tcga.patika.org/>. (Accessed on 08/15/2016).
- [8] J. Gao, B. A. Aksoy, U. Dogrusoz, G. Dresdner, B. Gross, S. O. Sumer, Y. Sun, A. Jacobsen, R. Sinha, E. Larsson, E. Cerami, C. Sander, and N. Schultz, “Integrative Analysis of Complex Cancer Genomics and Clinical Profiles Using the cBioPortal,” *Science Signaling*, vol. 6, no. 269, pp. pl1–pl1, 2013.
- [9] “cBioPortal for Cancer Genomics.” <http://www.cbioportal.org/>. (Accessed on 08/13/2016).
- [10] S. Moodie, N. Le Novère, E. Demir, H. Mi, and F. Schreiber, “Systems Biology Graphical Notation: Process Description language Level 1,” *Nat Precedings*, 2011.
- [11] M. Sari, I. Bahceci, U. Dogrusoz, S. O. Sumer, B. A. Aksoy, O. Babur, and E. Demir, “SBGNViz: A Tool for Visualization and Complexity Management of SBGN Process Description Maps,” *PLoS ONE*, vol. 10, pp. 1–14, 06 2015.
- [12] M. P. van Iersel, A. Villger, T. Czauderna, S. E. Boyd, F. T. Bergmann, A. Luna, E. Demir, A. A. Sorokin, U. Doğrusöz, Y. Matsuoka, A. Funahashi, M. I. Aladjem, H. Mi, S. L. Moodie, H. Kitano, N. L. Novre, and F. Schreiber, “Software support for SBGN maps: SBGN-ML and LibSBGN,” *Bioinformatics*, vol. 28, no. 15, pp. 2016–2021, 2012.

- [13] E. Demir, O. Babur, I. Rodchenkov, B. A. Aksoy, K. I. Fukuda, B. Gross, O. S. Smer, G. D. Bader, and C. Sander, “Using Biological Pathway Data with Paxtools,” *PLoS Comput Biol*, vol. 9, p. e1003194, 09 2013.
- [14] U. Doğrusöz, E. Giral, A. Cetintas, A. Civril, and E. Demir, “A layout algorithm for undirected compound graphs,” *Inf. Sci.*, vol. 179, no. 7, pp. 980–994, 2009.
- [15] “Google Realtime API.” <https://developers.google.com/google-apps/realtime/overview>. (Accessed on 08/01/2016).
- [16] P. Eades and R. Tamassia, “Algorithms for Drawing Graphs: An Annotated Bibliography,” tech. rep., Providence, RI, USA, 1988.
- [17] D. S. Johnson, “The NP-completeness column: an ongoing guide,” *JOURNAL OF ALGORITHMS*, vol. 8, pp. 434–451, 1987.
- [18] T. M. J. Fruchterman and E. M. Reingold, “Graph Drawing by Force-directed Placement,” *Softw. Pract. Exper.*, vol. 21, pp. 1129–1164, Nov. 1991.
- [19] G. Booch, J. Rumbaugh, and I. Jacobson, *Unified Modeling Language User Guide, The (2Nd Edition) (Addison-Wesley Object Technology Series)*. Addison-Wesley Professional, 2005.
- [20] N. L. Novére, M. Hucka, and H. e. a. Mi, “The systems biology graphical notation,” *Nature Biotechnology*, vol. 27, pp. 735–741, Aug 2009.
- [21] “Main Page - sbgn.org.” http://www.sbgn.org/Main_Page. (Accessed on 08/03/2016).
- [22] “LibSBGN is a library that deals with SBGN maps.” <http://www.sbgn.org/LibSBGN>. (Accessed on 07/30/2016).
- [23] “Free, open-source framework — Adobe Flex.” <http://www.adobe.com/products/flex.html>. (Accessed on 08/10/2016).
- [24] “ActionScript Technology Center — Adobe Developer Connection.” <http://www.adobe.com/devnet/actionscript.html>. (Accessed on 08/10/2016).

- [25] “Cytoscape, Network Data Integration, Analysis, and Visualization in a Box.” <http://www.cytoscape.org>. (Accessed in June 2016).
- [26] “Easily embed interactive networks in your website with Cytoscape Web.” <http://cytoscapeweb.cytoscape.org>. (Accessed on 07/30/2016).
- [27] “Adobe Flash Technologies — flash ActionScript and component - Adobe Labs.” <http://labs.adobe.com/technologies/flash/>. (Accessed on 08/10/2016).
- [28] “Cytoscape.js: An open-source JavaScript graph theory library for analysis and visualisation.” <http://cytoscape.github.io/cytoscape.js>. (Accessed on 07/30/2016).
- [29] “National Resource for Network Biology.” <http://nrnb.org>. (Accessed on 07/30/2016).
- [30] “National Institutes of Health.” <http://www.nih.gov>. (Accessed in August 2016).
- [31] “JSON.” <http://www.json.org/>. (Accessed on 08/10/2016).
- [32] “jQuery.” <https://jquery.com/>. (Accessed on 08/03/2016).
- [33] A. Subramanian, P. Tamayo, V. K. Mootha, S. Mukherjee, B. L. Ebert, M. A. Gillette, A. Paulovich, S. L. Pomeroy, T. R. Golub, E. S. Lander, *et al.*, “Gene set enrichment analysis: a knowledge-based approach for interpreting genome-wide expression profiles,” *Proceedings of the National Academy of Sciences*, vol. 102, no. 43, pp. 15545–15550, 2005.
- [34] “Pathway Commons Web Interface.” <http://www.pathwaycommons.org/>. (Accessed on 07/30/2016).
- [35] S. Kerrien, S. Orchard, L. Montecchi-Palazzi, B. Aranda, A. F. Quinn, N. Vinod, G. D. Bader, I. Xenarios, J. Wojcik, D. Sherman, M. Tyers, J. J. Salama, S. Moore, A. Ceol, A. Chatr-aryamontri, M. Oesterheld, V. Stümpflen, L. Salwinski, J. Nerothin, E. Cerami, M. E. Cusick, M. Vidal, M. Gilson, J. Armstrong, P. Woollard, C. Hogue, D. Eisenberg, G. Cesareni,

- R. Apweiler, and H. Hermjakob, “Broadening the horizon – level 2.5 of the HUPO-PSI format for molecular interactions,” *BMC Biology*, vol. 5, no. 1, pp. 1–11, 2007.
- [36] U. Dogrusoz, M. E. Belviranli, and A. Dilek, “CiSE: A Circular Spring Embedder Layout Algorithm,” *IEEE Transactions on Visualization and Computer Graphics*, vol. 19, no. 6, pp. 953–966, 2013.
- [37] B. Genç and U. Dogrusoz, “An algorithm for automated layout of process description maps drawn in SBGN,” *Bioinformatics*, vol. 32, no. 1, pp. 77–84, 2016.
- [38] K. Sugiyama, S. Tagawa, and M. Toda, “Methods for Visual Understanding of Hierarchical System Structures,” *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 11, pp. 109–125, Feb 1981.
- [39] “ChiLay Programmer’s Guide.” <http://www.cs.bilkent.edu.tr/~ivis/chilay/ChiLay-2.0-PG.pdf>. (Accessed on 07/30/2016).
- [40] “NCI: National Cancer Institute.” <http://www.cancer.gov/>. (Accessed on 08/18/2016).
- [41] “National Human Genome Research Institute (NHGRI) - Homepage.” <https://www.genome.gov/>. (Accessed on 08/18/2016).
- [42] “Memorial Sloan Kettering Cancer Center.” <https://www.mskcc.org/>. (Accessed on 07/30/2016).
- [43] “Dana-Farber Cancer Institute - Cancer Treatment and Research in Boston, MA.” <http://www.dana-farber.org/>. (Accessed on 07/30/2016).
- [44] “Welcome to the Princess Margaret.” <http://www.uhn.ca/PrincessMargaret>. (Accessed on 07/30/2016).
- [45] “The Children’s Hospital of Philadelphia.” <http://www.chop.edu/>. (Accessed on 07/30/2016).
- [46] “The Hyve: Open source solutions for bioinformatics.” <http://thehyve.nl/>. (Accessed on 07/30/2016).

- [47] “Bilkent i-Vis Research Group.” <http://www.cs.bilkent.edu.tr/~ivis/>. (Accessed on 07/30/2016).
- [48] “Biogene, A simple web service for querying gene information from Entrez Gene.” <http://cbio.mskcc.org/biogene/>. (Accessed on 07/30/2016).
- [49] D. Maglott, J. Ostell, K. D. Pruitt, and T. Tatusova, “Entrez Gene: gene-centered information at NCBI,” *Nucleic Acids Res*, vol. 33, pp. 54–58, January 2005.
- [50] “The GraphML File Format.” <http://graphml.graphdrawing.org/>. (Accessed on 08/11/2016).
- [51] “Google Docs - create and edit documents online, for free..” <https://www.google.com/docs/about/>. (Accessed on 08/01/2016).
- [52] “Google Drive - Cloud Storage & File Backup for Photos, Docs & More.” <https://www.google.com/drive/>. (Accessed on 08/01/2016).
- [53] “cPath2::Formats (Pathway Commons).” <http://www.pathwaycommons.org/pc2/formats>. (Accessed on 08/13/2016).
- [54] “JavaServer Pages Technology.” <http://www.oracle.com/technetwork/java/javaee/jsp/index.html>. (Accessed on 08/03/2016).
- [55] “Java Servlet Technology.” <http://www.oracle.com/technetwork/java/index-jsp-135475.html>. (Accessed on 08/03/2016).
- [56] “Backbone.js.” <http://backbonejs.org/>. (Accessed on 08/03/2016).
- [57] “AngularJS Superheroic JavaScript MVW Framework.” <https://angularjs.org/>. (Accessed on 08/03/2016).
- [58] “Bootstrap The world’s most popular mobile-first and responsive front-end framework..” <http://getbootstrap.com/>. (Accessed on 08/03/2016).
- [59] “MySQL.” <https://www.mysql.com/>. (Accessed on 08/03/2016).

- [60] “cytoscape/cytoscape.js-panzoom: Panzoom extension for cytoscape.js.” <https://github.com/cytoscape/cytoscape.js-panzoom>. (Accessed on 08/03/2016).
- [61] “cytoscape/cytoscape.js-cose-bilkent: The cose layout for cytoscape.js by bilkent with enhanced compound node placement..” <https://github.com/cytoscape/cytoscape.js-cose-bilkent>. (Accessed on 08/03/2016).
- [62] “Node.js.” <https://nodejs.org/en/>. (Accessed on 08/15/2016).
- [63] “W3C SVG Working Group.” <https://www.w3.org/Graphics/SVG/>. (Accessed on 08/19/2016).
- [64] “cBioPortal/cbioportal: cBioPortal for Cancer Genomics.” <https://github.com/cBioPortal/cbioportal>. (Accessed on 08/15/2016).
- [65] “Paxtools/sbgn-converter/src at master biopax/paxtools.” <https://github.com/BioPAX/Paxtools/tree/master/sbgn-converter/src>. (Accessed on 08/15/2016).
- [66] “iVis-at-Bilkent/TCGA-Curation-Tool: An interactive graphical editing / curation tool for TCGA cancer pathways.” <https://github.com/iVis-at-Bilkent/TCGA-Curation-Tool>. (Accessed on 08/15/2016).