Genotet: An Interactive Web-based Visual Exploration Framework to Support Validation of Gene Regulatory Networks

Bowen Yu, Harish Doraiswamy *Member, IEEE*, Xi Chen, Emily Miraldi, Mario Luis Arrieta-Ortiz, Christoph Hafemeister, Aviv Madar, Richard Bonneau and Cláudio T. Silva *Fellow, IEEE*

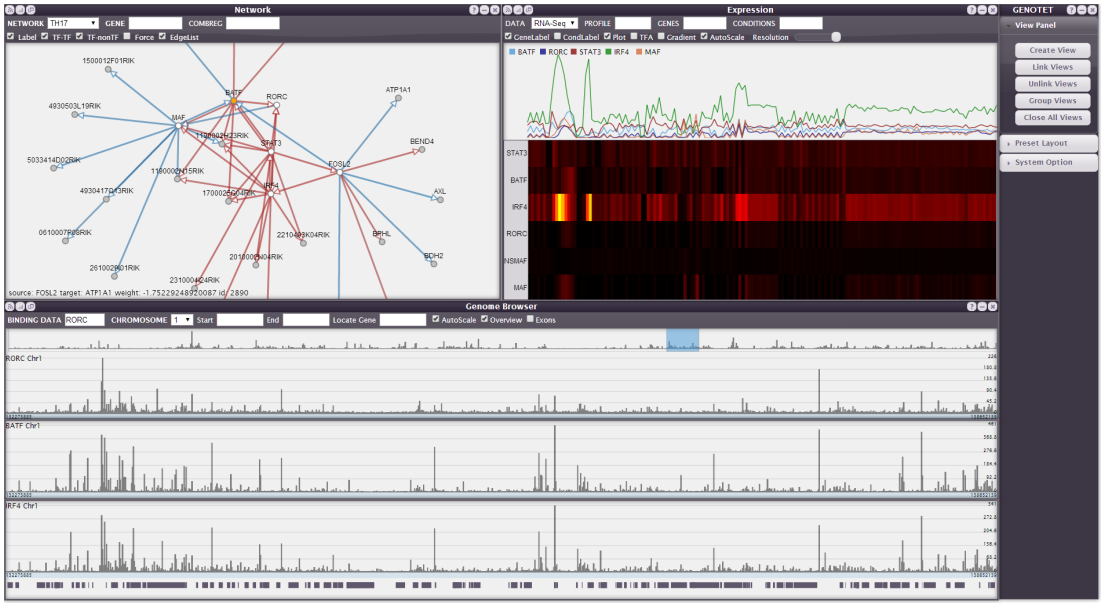


Fig. 1. An overview of Genotet interface. Genotet supports multiple views, where each view presents a different visualization metaphor for different types of gene regulation data. A powerful visual query interface achieved through dynamic linking / grouping of views enables an integrated visualization environment which significantly reduces the overhead involved during the analysis of this data.

Abstract— Elucidation of transcriptional regulatory networks (TRNs) is a fundamental goal in biology, and one of the most important components of TRNs are transcription factors (TFs), proteins that specifically bind to gene promoter and enhancer regions to alter target gene expression patterns. Advances in genomic technologies as well as advances in computational biology have led to multiple large regulatory network models (directed networks) each with a large corpus of supporting data and gene-annotation. There are multiple possible biological motivations for exploring large regulatory network models, including: validating TF-target gene relationships, figuring out co-regulation patterns, and exploring the coordination of cell processes in response to changes in cell state or environment. Here we focus on queries aimed at validating regulatory network models, and on coordinating visualization of primary data and directed weighted gene regulatory networks. The large size of both the network models and the primary data can make such coordinated queries cumbersome with existing tools and, in particular, inhibits the sharing of results between collaborators. In this work, we develop and demonstrate a web-based framework for coordinating visualization and exploration of expression data (RNA-seq, microarray), network models and gene-binding data (ChIP-seq). Using specialized data structures and multiple coordinated views, we design an efficient querying model to support interactive analysis of the data. Finally, we show the effectiveness of our framework through case studies for the mouse immune system (a dataset focused on a subset of key cellular functions) and a model bacteria (a small genome with high data-completeness).

Index Terms—Web-based visualization, gene regulatory network

1 Introduction

Regulation of gene expression underlies all cellular behaviors. Various types of regulatory rewiring allow the same set of genetic material (e.g. genes) to function in a spatially and temporally coordinated manner, giving rise to diverse phenotypes on both molecular and organismal levels. Transcription factors (TFs), proteins that bind to gene promoters and enhancers (gene-specific regulatory loci in the DNA) to alter gene expression, constitute a class of most-proximal regulators of gene expression. A network view of the regulatory dynamics involving TFs is essential in understanding how regulatory coordination takes place and how network components interconnect with each other as part of the dynamic system.

Recent breakthroughs in biotechnology have dramatically increased the scale (while reducing the cost) of collecting genome-wide datasets and have resulted in datasets large enough to allow for the automatic learning of regulatory networks from data. RNA-seq [32] and microarray [25] technologies enable global measurement of gene expression

- *Bowen Yu, Harish Doraiswamy, Cláudio T. Silva are at NYU Polytechnic School of Engineering. E-mail: {bowen.yu, harishd, csilva}@nyu.edu.*
- *Xi Chen, Emily Miraldi, Mario Luis Arrieta-Ortiz, Christoph Hafemeister, Richard Bonneau are at NYU Center for Genomics and Systems Biology, E-mail: {xchen, emiraldi, christoph.hafemeister, bonneau}@nyu.edu.*
- *Aviv Madar is at Cornell University. E-mail: am2472@nyu.edu.*

Manuscript received 31 Mar. 2014; accepted 1 Aug. 2014; date of publication xx xxx 2014; date of current version xx xxx 2014.
For information on obtaining reprints of this article, please send e-mail to: tvcg@computer.org.

levels, while ChIP-seq technology enables one to determine the binding of specific TFs across the genome. In addition, chromatin accessibility measurements [7, 13, 18] can measure what parts of the genome are actively bound by factors such as TFs. When paired with TF binding motifs (models of sequences of DNA patterns that specific TFs bind) these accessibility data and gene expression data can provide constraints on network structure (with each experiment providing several thousand priors on network structure). As input into mathematical modeling frameworks, these genome-scale datasets of TF binding and gene expression patterns provide an excellent opportunity to learn TF-target-gene regulatory networks requiring the need for visualization tools that can scale with these new experiments and the large network models that result from their analyses.

Several groups have used computational approaches to infer gene regulatory networks on global scales in a variety of organisms and cellular contexts, and subsets of these networks have been validated experimentally. Although thousands of TF-target gene relationships are inferred with statistical confidence, computationally inferred gene regulatory networks remain overwhelmingly *experimentally unvalidated* prior to publication. This is essentially due to limitations on time, resources, and scale of the experiments used to validate said networks. Integrated data and network visualization is a key element of workflows aimed at selecting the subset of predicted networks to be validated by costly biology experiments. Experimental biologists interested in model predictions generally access gene regulatory networks via multiple tools – one to visualize the gene regulatory networks, and another to visualize the primary data (gene expression and TF binding data). Analysis of this primary data in support of the various TF-target-gene relationships is crucial to motivating further model-hypothesis-driven experiments. While network visualization can be accomplished efficiently [26], visualization of primary data is performed through a genome browser [23], which is slow due to the size of the datasets. Note that the typical size of primary data corresponding to a relatively small network is usually larger than 100 GB. Also, since these tools require data to be stored locally, sharing data among biologists is difficult and further impedes post-publication experimental validation and utility of gene regulatory networks. Moreover, synchronization between the network and the primary data across different tools is non-trivial. Tools such as Gaggle [27] were developed to help perform this synchronization, and to allow connectivity between multiple tools. However, different datasets have different kinds of primary data, which makes it difficult to seamlessly use these tools. Several challenges make the use and maintenance of these tools problematic including: 1) reliance on client side active memory and computation that makes the tools cumbersome as datasets and network sizes increase, 2) difficulty in setting up these data-integration frameworks, and 3) the need to modify and use third party codes with mismatches in user interface functionality and modality.

1.1 Contributions

In this work, we develop Genotet (Figure 1), an interactive web-based framework to address the problems faced by biologists in validating gene regulatory networks. The software was the result of ten months of collaboration between three visualization researchers and six biologists (all co-authors) of the paper. Genotet is designed to specifically address the visualization tasks posed by the biologists. In particular, Genotet has the following properties:

- It visualizes the gene regulation network, their associated primary data including the gene expression and TF binding data, and supports queries driven by both genes (nodes) and regulatory interactions (directed edges).
- It provides interactive linking between different types of visualizations so as to provide seamless coordination across the visualizations of all data types.
- It is light weight and runs out-of-the-box on modern web browsers. Both, the memory and computational burdens of processing large gene regulation data is handled on the server-side.
- It uses level of detail rendering techniques together with a custom indexing scheme to help support interactive visualization and query execution over large binding data.

Finally, we demonstrate the utility of Genotet through several use cases attached to active genomics research efforts aimed at understanding regulatory networks in bacteria (B. subtilis) and mouse (Th17 cells, immune system).

2 Related Work

In this section, we first survey existing tools used by biologists for visualizations and analyses of gene regulation networks. We then briefly introduce other visualization systems developed to study other types of genomics data.

Biology tools for gene regulation data. The study of biochemical networks relies on iterating observation and experimentation, with biologists frequently checking and validating parts of gene regulatory network working models derived from the primary data. At the heart of this iterative model refinement are interactive interfaces for network data visualization. In practice, biologists can use any of a wide variety of existing network visualization tools, mostly local applications to visualize gene regulation data. Cytoscape [8, 11, 26] is one of the most widely used tools for gene network visualization, which facilitates large-scale network visualization. Many genome browsers exist for visualizing binding data. Local genome browsers such as Integrative Genomics Viewer (IGV) [23, 30], Ensembl [10], and GBrowse [16] are effective tools. There are also web-based genome browsers, e.g. the UCSC Genome Browser [24] aiming at visualizing genomics data mapped onto the chromosome such as the ENCODE data [12]. For expression profile data, the biologists typically run custom R scripts to generate visualization. An alternative is to use polyline and heat map plugins in another tool [17].

Although Cytoscape provides powerful graph visualization, it requires significant experience with the tool to find a satisfying Cytoscape configuration. Another key aspect of the problem is that biologists need to frequently switch between the visualizations of different data types. This can be achieved either manually or using plugins. Manual switching is extremely laborious and time consuming, thus plugins and automatic coordination of tools are preferred. Many plugins have been developed for Cytoscape. For example, Tang et al. introduced ContigScape [28] that facilitates gap closing of sequencing data from microbial genomes by presenting contigs relationships as a graph in Cytoscape. VistaClara [20] renders heat maps from matrix data. Despite the existence of those plugins, biologists find it troublesome to install multiple plugins and deal with potential compatibility issues. Besides plugins, bridging programs are written to provide a better connection between multiple applications. Gaggle [27] is a bridging framework that connects multiple biology tools that enables Cytoscape to communicate with other tools running locally. GenomeSpace [1] is another web framework for coordinating several biology visualization tools. The drawback of the bridging approach is that the user still needs to run multiple applications locally, which can be extremely resource consuming. We therefore seek to have a light weight visualization system that is easier to use, but is still integrative and effective in visualizing the gene regulation data. In this work, we thus focus on this requirement and develop a web-based framework that incorporates multiple visualization metaphors to help visualize gene regulatory networks as well as their associated primary data, and provide a simple query interface to interact between these data types which greatly reduces the burden on biologists.

Visual analytics systems. Advances in science and technology have enabled the generation of a large amount of biological data. In particular, with respect to genomic research, different biologists focus on different types of data related to genes. The diversity of these datasets and the wide range of requirements based on biologists' specific needs make it extremely difficult to have a "one size fits all" solution for the analyses of these datasets. Many visual analytics systems have been developed in recent times that cater to specific needs of biologists. We now list a few recent systems. Meyer et al. developed Pathline [22] and

Network	Alias	# Nodes	# Edges	Type	Binding	Expression	TF Activity
Th17	Th17	2218	4237	Mouse	79.8G/33 Genes wiggle files	21857×155	NA
Th17 Confidence	Confidence	15912	29394	Mouse			
B. Subtilis Prediction	Prediction	3119	4425	Bacteria	NA	3197×321	247×168
B. Subtilis Strength	Strength	4891	61535	Bacteria			

Table 1. Summary of the gene regulatory networks together with the corresponding primary data used in this paper.

Mizbee [21] to help biologists study the metabolic pathway and DNA sequence comparison, respectively. Barsky et al. [4] design Cerebal for cell network visualization based on experiment conditions. Kim et al. [19] proposed GeneShelf that handles changing gene expression data from public shared database and provides dynamic visual representations of these data. However, each of these systems cater to specific needs with respect to handling a single type of gene data, and do not lend themselves to be extended to be used simultaneously with multiple datasets of different types.

3 Data and Design Requirements

In this section, we discuss the different types of data used by the biologists to validate gene regulatory networks. We then summarize the primary design requirements to enable efficient analysis of these data.

3.1 Gene Regulation Data

Gene regulation data consists of a gene regulatory network together with its associated primary data. Gene regulatory networks [5] are essentially directed weighted networks. The nodes of the network correspond to genes (TF or non-TF). The edges are directed from TFs to target genes. The weight, which is positive for an activator and negative for a repressor, denotes the influence of a TF on its target gene.

Primary data is composed of one or more of gene expression, gene activity, and binding datasets. Gene expression data provides the expression levels of various genes for different experimental conditions (individual experiments). It is represented as a matrix, where the rows represent genes and columns represent experimental conditions. Each cell value in the matrix shows the expression level with respect to the gene of that row under the condition of that column. The gene activity data provides the TF activity of different genes for different experimental conditions, and has the same matrix format as the gene expression data. The binding data shows the binding frequency of a certain TF to a region of the chromosome. It is in the form of (location, count) pairs, where location refers to a coordinate on the DNA sequence for the full organism / genome, and count is the frequency of observed binding events. Table 1 provides a summary of the datasets used in this paper.

3.2 Design Requirements

We aim to create a single tool that can be used for validating gene regulatory networks through the use of visual queries on the primary data and the network. Multiple meetings were organized (between visualization researchers and biologists) to better understand the biologists' analyses needs. Based on these meetings, we finalized a set of tasks that is to be accomplished to meet the biologists' requirements. These are summarized below:

1. Support for visualizing the gene regulatory network.
2. Support for visualizing different types of primary data. This includes binding data, gene expression data, and gene activity data.
3. Ability to query different primary data through the gene regulatory network. For example, it should generate and execute automatic queries that are triggered by selecting directed edges in the network (the source node selects the binding data and the target node determines the genomic locus to be displayed in that data). This is one of the main operations performed by biologists to look at evidence for the presence of an edge in the network.
4. Should be able to handle multiple gene regulatory network datasets simultaneously. This allows comparison between different datasets.
5. Should be capable of handling large datasets. For example, binding data corresponding to a single gene is of size at least 1 GB. A regulatory network can have 1000s of genes.
6. Should be web-based and work in a browser. This enables biologists to have single repository for their data, which allows them to not only share their results, but also to carry out analyses with their collaborators.

4 Genotet System Design

Genotet is a web-based visualization framework and uses a client-server model that allows the computational burden of processing large gene data to be handled by a server. The client (web browser) is responsible only for rendering the data, and was implemented using JavaScript, predominantly with D3 [6] and jQuery [3] libraries. As modern browsers, irrespective of the operating system, support JavaScript, Genotet works out-of-the-box without imposing any overhead on the user. All data / computationally intensive tasks that result from queries and interactions performed on the client are handled by the server. The server-side component was implemented using node.js [2], a server-side JavaScript platform. The communication between the client and server is accomplished through HTTP requests and responses. In this section, we describe the design of Genotet and discuss the various choices made during its implementation.

4.1 User Interface Design

The user interface of the system consists of multiple data views, has the ability to dynamically link / group different views, and supports an interactive query interface. Figure 2 illustrates the different views and the various functionality supported by the interface.

Genotet employs a free-form layout that can be customized by the user. Each view corresponds to the visualization of a particular type of data. As mentioned earlier, the data consists of the gene regulatory network together with a set of primary data which includes binding data and gene expression data. The user can dynamically create, move, resize, and close the views so as to achieve the best desired layout. This design not only allows multiple datasets to be loaded simultaneously within Genotet, but it also makes it flexible to use.

The web-based client was designed using the model-view-controller architecture. Each data view consists of two components – data model (gene regulatory network, binding data, gene expression data) and its visualization (view). The controller is responsible for efficiently handling the interactions between different views and the resulting communications with the server. Two types of interactions between different views are supported – *linking* and *grouping* of views. The views are independent by default. The user can optionally link or group different views.

Linking views. Linking views enables one visualization to respond to the user interaction in another visualization. A link relation is defined as an ordered pair of views $(v_1 \rightarrow v_2)$. The controller keeps track of all the links between the different views. User interactions in a view v_1 triggers a link message to the controller, which communicates this message to all the target views that are linked to v_1. This could potentially result in one or more queries originating from the target views (see Section 4.3).

Grouping views. A view group is essentially a set of views $\{v_1, v_2, \ldots, v_n\}$. Grouping multiple views allows for data as well as visual synchronization across them. Visual synchronization "snaps" the views together, and forces the views to resize proportionally. This allows grouped views to be moved and resized together. Data synchronization, on the other hand, allows for properties of the different views of the *same type* within a group to be synchronized.

Any operation on a view triggers a group message to the controller which broadcasts it to all views in its group. This enables the corresponding operation to be executed on all views in that group (see Section 4.3). Note that, if a view receives a link message causing it to

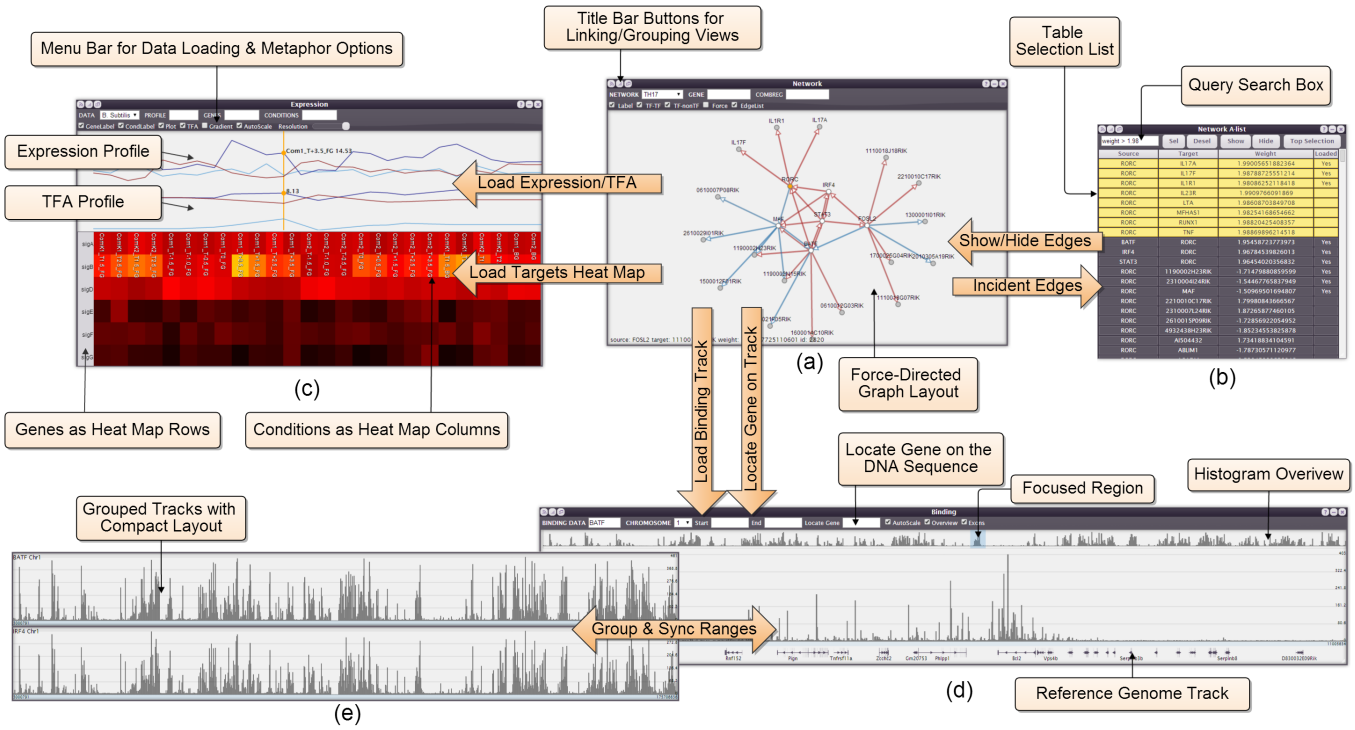


Fig. 2. Genotet views. Each view is associated with a title bar with shortcut buttons for view linking / grouping. Below the title bars are the menu bars that support queries and visualization options. **(a)** Network View: The graph plot applies force-directed layout that aims at iterative network study. **(b)** Table View: A table of incident edges that pops up when the user clicks a gene in the network. The table view supports filtering and selection of edges based on their properties. The user can thus iteratively add (remove) edges to (from) the graph. **(c)** Expression View: The expression view visualizes the gene expression data, with the expression matrix presented below as heat map, and selected expression profiles plotted as polylines above. TFA profile is also shown. **(d)** Genome Browser View: The genome browser view renders the binding data as a typical biology genome browser. Interactive track overview is located at the top of the view to ease navigation. Genes along with their names and exons are rendered at the bottom. The user may conveniently locate genes using the search box in the menu. **(e)** Genome browser views can be grouped to synchronize the displayed loci on the DNA sequence for comparison. Compact layout saves space for grouped views.

change its state, then it may trigger a group message where applicable so that all views within its group are synchronized.

UI interactions. There is a common menu panel for Genotet that supports view related operations including creating, deleting, linking / unlinking, and grouping / ungrouping views. In order to link views, the user has the option to select the source and target views from a list of available views. Grouping is accomplished similarly. Adding a new view to a group is accomplished by grouping that view with any view from the group.

Every view has a consistent layout for the title bar, supporting a set of common functionality. Specifically, at the top-left corner of each title bar, a set of shortcut buttons are provided that can quickly link / group views (see Figure 2). Additionally, hovering these shortcut buttons highlights views that are grouped or linked, which gives hints on the current linking / grouping configuration. Each view also has a menu bar that supports the specific functionality required by each visualization metaphor. The title bar and menu bar can be optionally hidden when not in use so as to construct a more *compact layout* (Figure 2(e)).

Extensibility. The linking and grouping behavior between various views depend on the views. It is up to a given view to act upon the link / group messages triggered by another view. In our design of the UI framework, we require each view to implement the API exposed by the controller that aids in this communication. Also, as mentioned earlier, the data and visualization components of a data view are decoupled in the design. Thus, it is possible to easily incorporate more functionality into Genotet in the future. For example, if a new visualization metaphor is required for an existing (or new) data type, then it is sufficient to just create a new view type (and data model) that implements the exposed API, and implement its functionality. The controller will automatically handle the communication between the new view and existing views.

4.2 Data Views

We now describe in detail the different visualization metaphors used to visualize the different data types along with the various options they support.

Network view. The gene regulatory network is a directed weighted graph that typically consists of thousands of nodes and edges. Biologists studying these networks start their analysis with a sub-network induced by a priori subset of genes of interest, and selectively expand the network when needed. Also, the size of the sub-network analyzed at any point does not get too large. Therefore, we focus on displaying small-sized graphs and improving the user experience of the biologists for such exploration instead of trying to layout the entire network.

Figure 2(a) shows the network view for a sample sub-network. We use a force-directed layout from D3 library to layout the sub-network containing the genes of interest. The direction of the edges are indicated by arrows, and the edges are color-encoded based on the edge weights (positive or negative). Biologists are interested in differentiating between TF and non-TF genes when viewing the network. Therefore nodes in the network are colored as either white or gray, depending on whether they are TF or non-TF genes, respectively. Several filters are provided to enable customization of the network visualization. The user can filter edges based on their types, i.e. they can selectively view only TF–TF edges, or TF–non-TF edges.

Certain nodes, especially those corresponding to TF genes have a large out-degree, typically several hundreds. A naïve method of expansion that displays all nodes adjacent to a TF node would quickly clutter the network visualization. In order to provide a cleaner method of exploration, we simply list the neighbors of a selected node as a table (Figure 2(b)). The user can then selectively add neighboring

edges using this table. The use of such a table has multiple advantages. It allows for showing additional attributes of these edges, such as edge weight, based on which the user can make an informed choice on whether to include the edge in the visualization or not. It also allows the use of traditional table primitives such as sorting the list based on a selected attribute, which makes it easy for the user to explore the large number of neighbors. Additionally, the table (and the network view) is also integrated into our query interface, which allows for users to search for genes (nodes) and edges of interest using comparison operators on attribute values and add them into the network.

Genome browser view. The binding data is a set of (location, count) pairs which are visualized as a histogram in the genome browser view. The x-axis of the histogram represents the DNA sequence coordinates, and the y-axis shows the binding frequency count (Figure 2(d)). Since the DNA sequence is very long (its length is approximately between $[3 \times 10^6, 2 \times 10^8]$), rendering the histogram in full detail would be inefficient. We therefore perform a level of detail (LoD) rendering to minimize network overhead as well as rendering time. When the query range is large, we partition the range into a uniform set of sub-ranges, and display the most prominent peaks from each of the sub-ranges.

To assist easy exploration, an overview of the histogram is displayed at the top of the view. The user can then select a region to focus along the DNA sequence by dragging a region in the overview. The selected region is highlighted in the overview as a blue rectangle in Figure 2(d). The reference genome track is visualized below the x-axis of the histogram. Each gene is represented as a line segment along the genome track, indicating its sequence position on the DNA. The gene names are labeled below the genes. Biologists are also interested in identifying the locations of a subset of a given gene known as exons, which are visualized as a set of boxes along the gene.

The genome browser view (like all our other views) is integrated with the the query interface, allowing the user to efficiently search for a gene across all chromosomes. Searching for a gene along the DNA sequence results in automatically centering the coordinate range of the histogram to the locus of chosen gene. The user can also dynamically change the binding data of a genome browser view.

Expression View Recall that the gene expression data is represented as a real number matrix where the rows and columns correspond to genes and experiment conditions, respectively. This data is visualized using the heat map metaphor. Figure 2(c) shows an example of the expression view visualizing gene expression data. The biologists particularly want each cell of the matrix to be identifiable. So, instead of opting for a continuous heat map, we choose to render the heat map as a discrete set of rectangles for each cell.

An expression matrix contains up to tens of thousands of genes and hundreds of conditions. In order to allow effective exploration of this data, we allow the user to select and zoom into a region of interest in the heat map. Biologists are sometimes interested in visualizing only a selected set of genes and conditions. This is supported by allowing the user to manually change the rows and columns of the heat map through adding or removing genes and conditions. Alternatively, this operation can also be efficiently accomplished through the use of regular expressions to select the required groups of genes or conditions via the query interface. The latter is especially useful when the biologist already has a set of genes / conditions of interest. Again, in order to reduce both the network traffic and the rendering overhead, we use LoD rendering for the heat map.

When viewing only a few selected genes, the biologists prefer to plot the gene expressions as polylines. These polyline plots, also known as *expression profiles*, of the selected set of genes are rendered above the heat map. In case a selected gene is a TF, and has activity data associated with it, the activity profile is additionally visualized in the *TFA profile*. These two profiles are shown in Figure 2(c). The condition name along with the corresponding data values are displayed by hovering different parts of the polylines, as highlighted by the orange line and dot in Figure 2(c). Both the expression and the TFA profiles can optionally be hidden when not needed.

4.3 Query Interface

One of the main goals of this work is to enable efficient queries of the gene data. The linking and grouping features provide a powerful visual query interface to accomplish this. We now describe the different possible query types supported by Genotet in detail.

Link-based queries. In the analysis of gene regulatory networks, the most frequent queries used by the biologists are constituted of those between different data types. Linking multiple views assists in such queries across different data types and also provides a simple visual query interface that greatly simplifies the analysis of the gene regulatory networks. The different visual queries supported depend on the types of views that are linked together. We now discuss the possible queries based on view linking.

1. *Network view – Genome browser view.* The most common operations performed by the biologist are to view the binding data corresponding to nodes / edges in the gene network. Such queries are accomplished through the network view – genome browser view link by allowing querying the binding data directly through the network visualization. When the user selects a gene (node) in the network, the genome browser view automatically centers the DNA coordinate range to the gene's locus. When the user selects an edge in the network, the binding data of the source gene is loaded and the locus corresponding to the target gene in the binding data is displayed. This feature satisfies one of the main requirements of the biologists, since it saves intensive manual effort of going back and forth between multiple tools and searching for specific genes in them.
2. *Network view – Expression view.* While studying gene regulatory networks, biologists closely observe a TF and all its regulating genes. The expression profiles of a pair of regulator and target are then studied and compared. In order to accomplish this, when the user selects a gene in the network, all the genes that are targets of the selected gene are loaded as rows of the heat map in the expression view. If an edge is selected, then the expression profiles of the source and target genes are added to the expression profile plot.

Group-based queries. Grouping different views will synchronize any user interaction, such as selection, where applicable. This is especially useful for comparisons across different datasets, where focusing on a particular region in one of the views will automatically focus the other views in the group as well. For example, multiple genome browser views can be grouped together, so as to present uniform stacked layout equivalent to a typical multi-track genome viewer. Figure 2(e) shows an example where two genome browser views are grouped together. When grouped, all the genome browser views within the same group will synchronize their DNA sequence ranges. Thus, when one of the views has its range updated, the ranges are accordingly updated in the other views as well. This makes the comparison of different binding tracks straightforward. Also, when the user performs any query (or operation) on one view, then the same query is also performed on the other view. The compact layout option is especially useful for grouping views.

Text-based queries. Genotet extensively employs regular expression based searching in the various views. The biologists are interested in having a quick method to simultaneously load data related to multiple genes. Manual searching of these genes either through parsing a list, or manually entering their names is a cumbersome process. Also, in most cases, the names of genes searched for have common naming patterns. Thus the use of regular expression greatly simplifies the search procedure. Regular expression based search options are available in every view. Each of these views have a input text box in their menu using which the user can execute these queries. The regular expressions are associated with a command indicating the operation to be performed on the resulting set. For example, regular expressions are used to add / remove nodes into the graph (with command "add / rm regexp"). Similarly, it can be used to add / remove genes and conditions into / from the heat map.

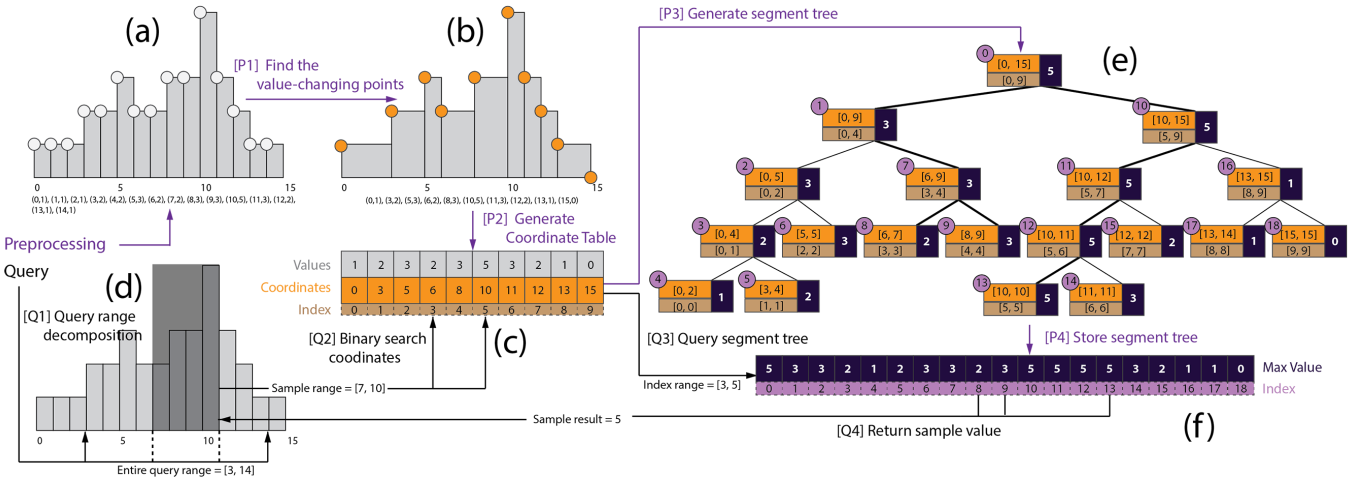


Fig. 3. The workflow for handling queries on binding data. It consists of two parts – preprocessing and querying, colored purple and black respectively. The different steps of the preprocessing stage are as follows: **[P1,P2]** The original data (a) is first compressed to generate the coordinate table (c) . **[P3,P4]** The segment tree (e) is then computed using the coordinate table and is stored as an array in the preorder traversal order (f). During user interaction, a range query is executed using the segment tree as follows: **[Q1]** We first sub-divide the query range into sample ranges according to the number of samples needed (d). **[Q2]** A binary search on the coordinate table (c) identifies the new range to be queried. **[Q3,Q4]** The updated range is then searched using the segment tree and the result is returned.

Additionally, the biologists are interested in studying the network by querying common targets of TFs, which represent combined regulation. We integrate this functionality into our querying interface, where the users can query for common targets using the search box present in the network view. Prior to using our tool, biologists could not perform such queries using the other genome visualization software. They usually resorted to writing scripts in R or Matlab to perform the required query, followed by manually loading the results into the network viewer. Our query interface, on the other hand, integrates multiple such cumbersome operations in a seamless manner, easing the burden on the user.

4.4 Handling Large Binding Data

A gene regulatory network typically consists of thousands of genes, and the matrix representing the gene expression data is composed of tens of thousands of genes (rows) and a few hundred conditions (columns). These data do not occupy much disk space. Even a brute-force linear search for a gene in these data would not impede the interactivity of Genotet. The binding data, on the other hand, is much larger in size. As mentioned earlier, it is composed of multiple binding tracks. Tracks are identified by gene and chromosome, or sequence ID. The length of each binding track is at least tens of millions and can be as large as a few hundred million. This results in approximately 2 GB of binding data per gene.

Effective handling of the binding data poses two main challenges. First, with increasing number of experiments being performed by the biologists, the amount of binding data becomes exceedingly large, making its maintenance difficult. Second, as mentioned in Section 4.2, we use LoD rendering of the histogram to avoid network overhead as well as rendering time. In order to obtain a good approximation, the samples should be chosen carefully so as not to miss any prominent "peaks" of the histogram. Interactive querying and sampling of such large data is non-trivial. Hence, it is necessary for Genotet to be able to handle such large data.

Using traditional databases does not serve our purpose, since the index overhead involved to support efficient queries on this data is also large, resulting in the consumption of more disk space. Also, large queries takes more than a second to execute. For example, a typical range query having range length of 500,000 takes close to 2 seconds to execute using PostgreSQL [29]. Note that, we could have queries whose range can be as large as a 100 million. When combined with the time taken to sample the results and transfer it back to the client, the total query execution time is not suitable for interactivity.

In order to handle data at this scale, we resort to using a custom storage and index data structure. We first store each binding track in a simple compressed format that does not alter ordering of the data. A custom index is then built directly on the compressed data. This index allows range queries used during LoD rendering to be interactively executed. An overview of this workflow is illustrated in Figure 3.

Data compression. The binding data is available as text wiggle files [31]. Figure 3(a) illustrates a small example of a wiggle file, where the data pairs are shown below the histogram. We observed a lot of redundancies across consecutive pairs of values. Therefore, we iterate through the entire data once and store only the necessary value-changing points of the histogram in a *coordinate table*. This results in a smaller set of (location, count) pairs whose corresponding histogram is *exactly the same* as the original histogram. This operation is illustrated in Figures 3(b) and 3(c).

Query execution. Let k be the maximum number of sample points that are to be returned after a DNA coordinate range update. Genotet uses a value of k equal to the width in terms of pixels of the genome browser view. Given a range query, we first divide the input range into a set of k equal sub-ranges. For example, consider the example query shown in Figure 3(d) having range [3, 14]. Let $k = 3$. The query range is equally subdivided into $k = 3$ sub-ranges. The maximum value from each of these sub-ranges are then returned as the result of the query. Using the maximum value within these sub-ranges ensures that the approximation closely matches the actual histogram without losing the peak details. This is a typical range maximum query (RMQ) that can be efficiently answered using the segment tree data structure [14] in $O(\log n)$ time per query, where n is the size of the tree. In this case, n equals to the size of the coordinate table.

The segment tree constructed for our running example is illustrated in Figure 3(e). We use the table index to represent the ranges while constructing the segment tree. The orange boxes of the segment tree show the original DNA coordinate range (for reference), the brown boxes show the table index range, and the eggplant boxes show the maximum value stored in the tree nodes. Since the range of table index is continuous, it allows us to flatten the segment tree as a linear array in the pre-order traversal order. The tree node number obtained by the pre-order traversal corresponds to its index in the flattened array (Figure 3(f)). This array stores only the maximum values. The segment tree is thus implicitly stored in this array as follows. Element 0 of the array corresponds to the root of the segment tree, and has range $[0, n-1]$, where n is the size of the coordinate table. In Figure 3 we have $n = 10$. For element i with range $[l_i, r_i]$, its left child is element $i+1$ with range $[l_i, \lfloor (l_i + r_i)/2 \rfloor]$ and its right child is element

$i + (\lfloor (l_i + r_i)/2 \rfloor - l_i + 1) \times 2$ with range $[\lfloor (l_i + r_i)/2 \rfloor + 1, r_i]$. Thus, a segment tree covering a range of length n has exactly $2n-1$ tree nodes, which is also the length of the flattened array.

In order to query for the maximum within a sub-range, the corresponding table indexes are first computed using a binary search on the coordinate table. Then searching for this range in the segment tree provides the result of the sub-range query.

The coordinate table and the segment tree array are both stored as binary files. Using this custom data structure, we are not only able to interactively execute range queries, but we also obtain orders of magnitude saving in space. For example, close to 80G of the Th17 binding data could be stored using only 8.47G using our custom index.

4.5 Discussion

Feature choice for LoD Rendering. As mentioned earlier, we use the maximum value within a given range to depict the value for that range when rendering the histogram / heat map in the genome browser / expression view. Alternatives would have been to use the mean or median frequency of the given range. We chose to use the maximum value because the biologists were interested in viewing the most dominating feature within a given range.

Layout choice. The process of studying the gene regulatory network is complicated and discussions with the biologists indicated that, depending on the aspect of the data they are looking at, they wanted a different set of views with differing interaction possibilities. For example, they sometimes wanted multiple coordinated genome browser views, and at other times they wanted a network view with either one or both of genome browser view and expression view. Even the number of views of each type they wanted varied depending on the application. Having a user interface with a fixed set of views has the disadvantage that multiple predetermined set of views have to be designed. This makes the system rigid, requiring developer intervention to add and configure newer setups in case of new requirements from the biologists. We therefore decided to have a more flexible interface, where the user can dynamically create and remove views and add the required interaction between them through linking and grouping the views.

Linking / grouping views. Currently, the user has to manually check the links / groups of each view by using the shortcut on the view's tool bar. However, when there are multiple views with several connections between them, it can get cumbersome to keep track of the existing links / groups. We plan to address this in the future by exploring different possibilities in which the relationship between views can be easily inferred.

Link functionality. We currently support only one possible type of linking between a pair of views. It is however possible for a pair of views to have more than one possible linking functionality. For example, when a node is selected in the network view, the user could choose to load the binding data of clicked node instead of the current functionality that searches for the locus of the selected gene in the loaded binding data. We intend to support the selection of the required functionality for view links in the future.

5 Use Cases

In this section we present three use cases that demonstrate the effectiveness of Genotet in visualizing gene regulation data and helping biologists validate their constructed gene regulatory networks.

5.1 Validating Confirmed Regulation in the Th17 Network

The mammalian immune system is composed of many functionally diverse cell types. In addition to being clinically relevant, the study of immune cell types provides a good model for studying mammalian gene regulation. In most, if not all immune cell types, the cooperative binding of a few TFs (termed master regulators) directs distinct transcriptional programs that lead to defined developmental cell fates. To understand the principles underlying mammalian gene regulatory networks, biologists previously used the lineage differentiation process of Th17, an inflammatory cell type characterized by the expression of cytokine IL17, as a model system for studying the networks that

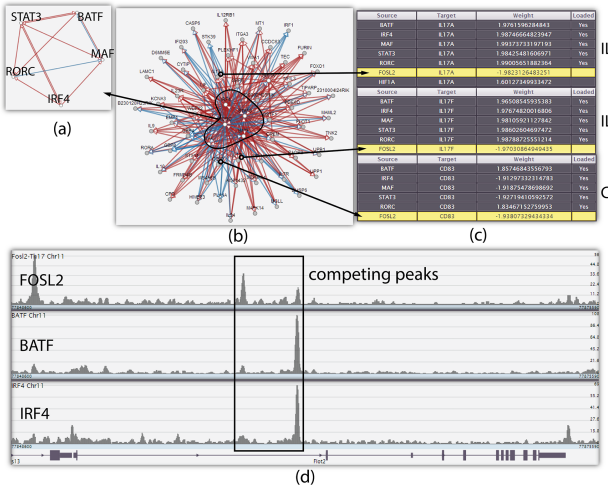


Fig. 4. Case 1: Validating the regulatory network found by [9]. **(a)** The five core TFs are visualized in the network view. **(b)** Using the combined regulation search of Genotet, the co-regulated targets of the core TFs are added to the network (the TFs are marked in the network). **(c)** We found support for FOSL2 mediated regulation at the locus by inspecting the incident edges of the targets. **(d)** Loading the binding data of FOSL2, BATF and IRF4 indicates that FOSL2 has competing binding signals against BATF and IRF4.

support immune cell fate specification. In Th17 cells (cells that play central roles in chronic inflamation), pioneer TF BATF and IRF4 bind to regulatory DNAs that alter chromatin accessibility and predispose regions of the chromatin for the cooperative binding of other factors that together shape cell fate. Biologists are interested in evidence of TF cooperativity in these cells and attempt to understand the network architecture extending from the core TFs.

In this case study, we try to validate a previously identified regulation in the Th17 network [9]. We reconstruct a gene regulatory network of Th17 gleaning evidence from gene expression data (RNA-seq), TF binding data (ChIP-seq), and chromatin accessibility data (FAIRE-seq). To find transcription regulators that function upstream of or in parallel with core Th17 TFs, we create a sub-network centered around the five core TFs: IRF4, BATF, STAT3, RORC, and MAF (Figure 4(a)) and look for candidate TFs that show significant target overlap with these core TFs, using the "combined regulation" search function (Figure 4(b)). Then, by listing the TF regulators for each target in the sub-network, we see that in addition to the five TFs, FOSL2 and HIF1A show up in the list recurringly (Figure 4(c) shows the occurrences of FOSL2). This suggests that FOSL2 and HIF1A regulate the same set of genes and potentially play important roles in Th17 lineage differentiation. This was later experimentally verified via gene knock-down and ChIP-seq experiments [9].

There were speculations that FOSL2 competes with BATF for binding at some loci function as a modulator repressing the expression of IL17A. To further investigate this we look at these common target loci to see if there is evidence of regulatory interactions in the ChIP-seq binding data. This is accomplished by loading and grouping the ChIP-seq tracks for FOSL2, BATF and IRF4. We found, among a few other examples, competing binding signals around FLOT2, a common target of the Th17 core TFs. This is shown in Figure 4(d), where FOSL2 presents a significantly lower peak where the peaks of BATF and IRF4 are high, and vice versa. FLOT2 is a gene that participates in signal transduction pathways related to cell growth, but there has not been many connections made between FLOT2 and Th17 specification. It therefore motivates further research on FLOT2.

In this example, Genotet enables biologists to perform an analysis that requires integration of multiple data types (network and binding data). The advantages of using Genotet in such analyses is reflected in the following comment from the biologist performing this experiment: *"Without Genotet, this combined regulation visualization requires rather complicated user operations. In existing tools such*

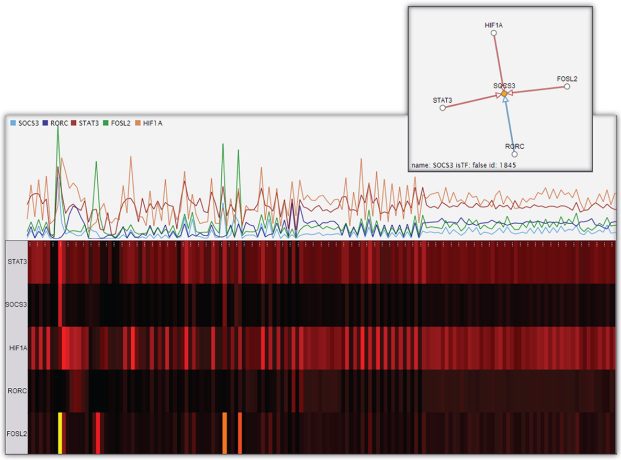


Fig. 5. Case 2.1: The regulators of SOCS3 are displayed with color-coded edges in the network view (top-right corner). The heat map and the profile plot show correlation between SOCS3 and its predicted regulators. Note that each gene's expression profile is normalized to that gene's maximum value so that correlations can be easily identified.

as Cytoscape it is not possible to execute such query without writing script to parse the data into a new network, which is laborious and time-consuming."

5.2 Exploring the Regulation of Single Gene in the Th17 Network

Biologists are often interested in studying only a subset of the gene regulatory network or even just the transcriptional regulation of a single gene. The difficulty of performing such analyses using existing software is reflected in the following comment by the biologist: *"Prior to using Genotet, much effort would have been required to, for example, find and evaluate network predictions about the regulation of a single gene."* The advantages of using Genotet for such analyses is demonstrated in this case study, where we focus on a specific gene – SOCS3. Specifically, we would like to study the Th17 network and use Genotet to assess: 1) whether SOCS3 plays a role in Th17 differentiation and any TFs are predicted to regulate its transcription, 2) the confidence in predictions by evaluating primary data, 3) the quality of the expression data through looking at the raw RNA-seq experiment data with timestamps.

If SOCS3 is in the Th17 network, it is likely to be involved in Th17 differentiation. To check whether SOCS3 is in the Th17 network, we first try to load the gene of interest, SOCS3, into the network view by querying for it. We find that SOCS3 is indeed present in the Th17 network. To check whether any TFs are predicted to regulate SOCS3's expression, we look at the other genes incident to SOCS3 in the network via the incident edge list table. Four predicted regulators of SOCS3 expression (RORC, STAT3, FOSL2, HIF1A) are then selected and added into the network for visualization. The edges are color-coded by weights so that the predicted type of regulatory interaction (positive or negative) can be easily identified. The resulting network is shown at the top-right corner of Figure 5.

In order to gain confidence in the regulatory predictions, we need to examine the primary data. For the Th17 network, two types of data were used to inform the computational network inference procedure: RNA-seq gene expression data and ChIP-seq binding data. First, we examine the gene expression data for evidence of an interaction. Specifically, we expect to see a correlation between regulator and target expression. To accomplish this, we load this data into an expression view, and limit the genes visualized in the heat map to SOCS3, STAT3, FOSL2, HIF1A, RORC. The heat map displays gene expression levels, and there appears to be some correlation between SOCS3 and its regulators. To further analyze this correlation, we add these five genes into the profile plot. The correlation becomes more apparent in the profile plot, as shown in Figure 5.

Next, we examine the binding evidence (ChIP-seq data) by creating

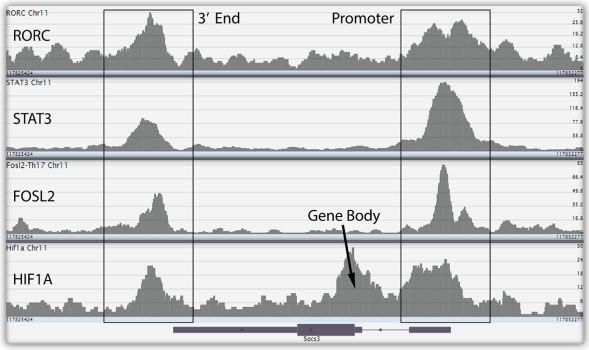


Fig. 6. Case 2.2: Finding evidence for SOCS3 regulatory predictions in the Th17 primary data by grouping the genome browser views. The four tracks are centered on the gene locus of SOCS3, showing the binding data of RORC, STAT3, FOSL2 and HIF1A respectively. Binding peaks at SOCS3's promoter, gene body, and 3' end provide evidence for all four regulatory predictions.

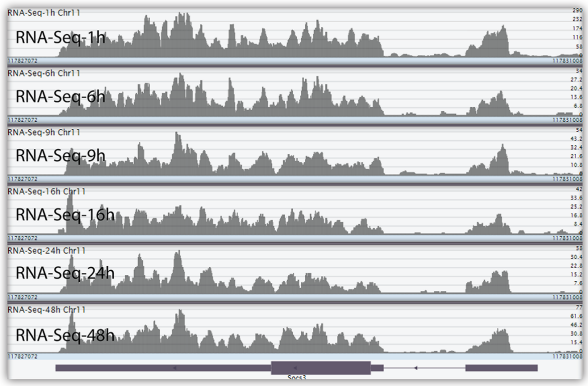


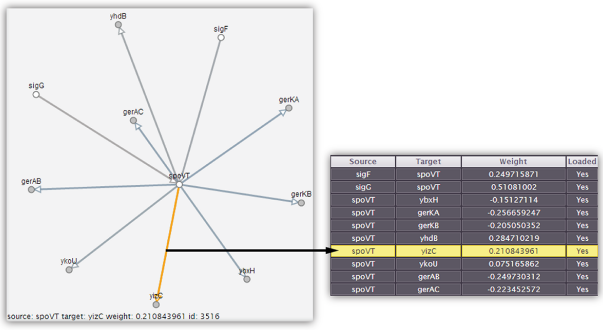
Fig. 7. Case 2.3: Assessing the quality of the binding data by loading the raw RNA-seq expression data with timestamps. It can be seen that the RNA-seq signals cover the entire gene (SOCS3 locus), which indicates that the gene's expression is well reflected in the experiment.

four genome browser views in Genotet and load the binding data corresponding to RORC, STAT3, FOSL2, and HIF1A. To gain confidence in the regulation interactions of those TFs with SOCS3, we check if there exists ChIP evidence for binding of these factors near SOCS3. The linking and grouping features of Genotet is convenient and effective in this analysis. After grouping the four genome browser views, all the tracks can be searched simultaneously. We search for SOCS3 to synchronize the ranges of the genome browser views around the SOCS3 locus. There is evidence displayed in the visualizations that all the four TFs bind to the promoter region of SOCS3 and also some binding downstream (Figure 6).

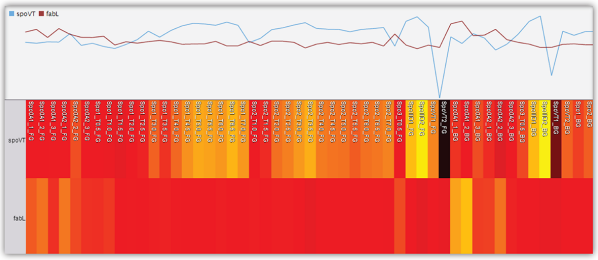
In addition to producing informative visualizations of gene expression data, it is also possible to assess the quality of the expression data by accessing the raw gene-expression RNA-seq files. Therefore the results of RNA-seq experiments corresponding to six timestamps are loaded (Figure 7). We see that the RNA-seq reads map across the entire gene, a quality-control hallmark that verifies the gene's expression. We could thus conveniently verify the raw data across multiple experiments for the time course of Th17 development.

5.3 Verifying the Regulation of Sporulation in *Bacillus Subtilis*

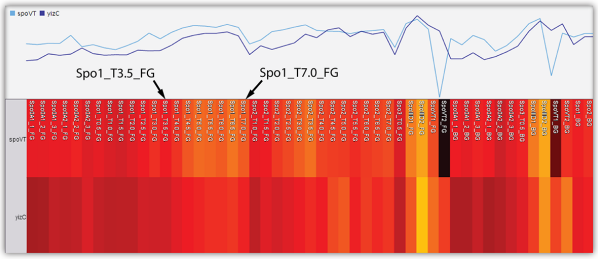
Bacteria are unicellular organisms widely studied for their key role in health and disease, and as model systems for learning new biology that is applicable to all systems. Bacteria have smaller genomes than plants and animals and both computationally (scale) and experimentally (genetics, lab protocol, cost) tractable systems for systems biology (genome-wide) studies of cellular regulation. *Bacillus Subtilis* (*B. Subtilis*) is the main model organism for Gram-positive bacteria. One of the most studied process in this species is sporulation. Through sporulation a cell is able to differentiate into a spore [15] that is resis-



(a)



(b)



(c)

Fig. 8. Case 3: Using Genotet to study the regulation of sporulation in *B. Subtilis*. **(a)** fabL, a known spoVT's target, was not identified in the predicted network. An existing edge from spoVT to yizC is highlighted. **(b)** The missing edge between spoVT and fabL can be explained by the low correlation observed between the expression of the two genes. **(c)** Presence of the edge between spoVT and yizC is verified through the observed correlation between their expression profile.

tant to hostile environmental conditions. Sporulation is an important survival strategy for *B. Subtilis*, and contributes to the persistence, antibiotic resistance, and spread of several pathogens. When the environment improves, the spore germinates producing a viable cell. In this case study, we would like to understand how the cell coordinates this process, with transcriptional regulation being one key layer of regulation for sporulation.

First a sporulation related gene spoVT is selected and displayed in the network view. The genes that are regulated by spoVT are listed and added into the network view (Figure 8(a)). By looking at the network, we realize that a known target gene, named fabL, is missing in the network. We therefore attempt to identify the reason behind it. To do this, it is necessary to observe the expression profile of spoVT and fabL, and check whether there exists a relationship between them under the sporulation-related experiment conditions (conditions containing the abbreviation "spo"). Using the linking feature of Genotet, the profile plot for spoVT is automatically generated by selecting spoVT in the network. In order to compare this plot with that of fabL, we search and add fabL into the gene expression view. To select all the sporulation related conditions, a regular expression search using "spo.*" comes in handy. The result of this operation is shown in Figure 8(b). It can be seen that there is no correlation between the profile of spoVT and fabL, which gives the reason for fabL missing in the constructed *B. Subtilis* network. Thus the missing edge of the network is verified. The difficulty of performing this analysis prior to using Genotet is indicated in the following comment by the biologist: *"Without the functionality of Genotet, performing the above experiment would require us to export the expression profile data for spoVT and fabL and either use a custom R script to plot them, or load the expression matrix in Cytoscape by installing extra plugins."*

We would also like to verify known regulatory interactions in light of the above observation. We find an unexpected target yizC in the network, of which the regulation is unknown. To check if this edge is correctly included by the network, we add the expression profile of yizC to the expression view. Again, due to the linking functionality, the comparison of the profile plot can be easily achieved with a single click on the edge from spoVT to yizC (highlighted in Figure 8(a)). We observe a correlation between spoVT and yizC in the profile plot as shown in Figure 8(c). The expression profile of spoVT shows a strong up and down trend with the profile of yizC, under a set of conditions with continuous timestamps (from Spo1_T3.5_FG to Spo1_T7.0_FG). Thus, the existence of this edge in the network is verified. The ease of using Genotet for such analyses is reflected in the following comment by the biologist performing this experiment: *"It is a friendly approach to explore the network while users can also evaluate how good additional information supports the novel edges."*

6 Expert Feedback

The biologists used Genotet for over a month after which we conducted informal face to face interview with them and collected their feedback. Their response from using Genotet was encouraging, as noted by some of their comments that are mentioned in the previous section. They were also enthusiastic about using it in their future research, and plan to use it as a effective instrument for dissemination of knowledge among not just their collaborators, but also to the research community. This is reflected by the following comments:

"The tool will help us share our results with biologists, who often don't have experience in exploring that sort of data. We also plan to use the tool to provide an interface to supplementary data accompanying publications."

"We could use this tool for sharing our findings with the scientific community. It is an easy way to explore the network. People can focus on their favorite genes and cellular processes."

While the feedback was broadly positive, they also felt that our system needs some minor refinements so that new users can easily get started using Genotet. One such request was to provide a user interface to upload data onto the web server. Currently, users need to manually copy the data onto the remote server, and run a script for creating the required indexes. One of the biologists also wanted support for viewing additional meta data such as information about known interactions and the annotation of each gene. Another request was to support different layout options (which the users can choose) for viewing the network. We intend to address these requests before making our tool public. We expect one of the significant outcomes of this collaboration to be the adoption of Genotet by the systems biology research community.

7 Conclusions

In this work we developed Genotet, a light weight web-based interactive visual exploration system for gene regulation data. A powerful visual query interface was achieved through dynamic linking and grouping of different types of views. Genotet has the capability of handling large datasets due to the use of custom index structures which also helps support interactive queries of the data. Interactive visualizations were handled through the use of level of detail rendering of the visualizations. In addition to making Genotet public, we will also be working with our collaborators in several biology driven papers resulting form its usage.

Acknowledgments

This work was supported in part by an IBM Faculty Award, the NYU School of Engineering, DOE, NSF awards CNS-1229185, IOS-1126971, and NIH awards PN2-EY016586, IU54CA143907-01, EY016586-06.

REFERENCES

[1] GenomeSpace. http://www.genomespace.org.

[2] Node.js. http://www.nodejs.org.

[3] The jQuery Foundation. http://jquery.com/, 2014.

[4] A. Barsky, T. Munzner, J. Gardy, and R. Kincaid. Cerebral: Visualizing multiple experimental conditions on a graph with biological context. *Visualization and Computer Graphics, IEEE Transactions on*, 14(6):1253–1260, Nov 2008.

[5] R. Bonneau. Learning biological networks: from modules to dynamics. *Nature Chemical Biology*, 4(11):658–664, Oct. 2008.

[6] M. Bostock, V. Ogievetsky, and J. Heer. D3: Data-driven documents. *IEEE Transactions on Visualization and Computer Graphics*, 2011.

[7] J. D. Buenrostro, P. G. Giresi, L. C. Zaba, H. Y. Chang, and W. J. Greenleaf. Transposition of native chromatin for fast and sensitive epigenomic profiling of open chromatin, DNA-binding proteins and nucleosome position. *Nature methods*, 10(12):1213–8, Dec. 2013.

[8] R. Christmas, I. Avila-Campillo, H. Bolouri, B. Schwikowski, M. Anderson, R. Kelley, N. Landys, C. Workman, T. Ideker, E. Cerami, R. Sheridan, G. D. Bader, and C. Sander. Cytoscape: A software environment for integrated models of biomolecular interaction networks. *Amercian Association of Cancer Research*, pages 12–16, 2005.

[9] M. Ciofani, A. Madar, C. Galan, M. Sellars, K. Mace, F. Pauli, A. Agarwal, W. Huang, C. N. Parkurst, M. Muratet, K. M. Newberry, S. Meadows, A. Greenfield, Y. Yang, P. Jain, F. K. Kirigin, C. Birchmeier, E. F. Wagner, K. M. Murphy, R. M. Myers, R. Bonneau, and D. R. Littman. A validated regulatory network for Th17 cell specification. *Cell*, 151:289–303, 2012.

[10] M. E. Clamp, T. D. Andrews, D. Barker, P. Bevan, G. Cameron, Y. Chen, L. Clarke, T. Cox, J. A. Cuff, V. Curwen, T. Down, R. Durbin, E. Eyras, J. Gilbert, M. Hammond, T. J. P. Hubbard, A. Kasprzyk, D. Keefe, H. Lehvslaiho, V. Iyer, C. Melsopp, E. Mongin, R. Pettett, S. C. Potter, A. Rust, E. Schmidt, S. M. J. Searle, G. Slater, J. Smith, W. Spooner, A. Stabenau, J. Stalker, E. Stupka, A. Ureta-Vidal, I. Vastrik, and E. Birney. Ensembl 2002: accommodating comparative genomics. *Nucleic Acids Research*, pages 38–42, 2003.

[11] M. S. Cline, M. Smoot, E. Cerami, A. Kuchinsky, N. Landys, C. Workman, Rowan, I. Avila-Campillo, M. Creech, B. Gross, K. Hanspers, R. Isserlin, R. Kelley, S. Killcoyne, S. Lotia, S. Maere, J. Morris, K. Ono, V. Pavlovic, A. R. Pico, A. Vailaya, P.-L. Wang, A. Adler, B. R. Conklin, L. Hood, M. Kuiper, C. Sander, I. Schmulevich, B. Schwikowski, G. J. Warner, T. Ideker, and G. D. Bader. Integration of biological networks and gene expression data using Cytoscape. *Nat Protoc*, 2:2366–2382, 2007.

[12] T. E. P. Consortium. The ENCODE (ENCyclopedia Of DNA Elements) project. *Science*, 306(5696):636–640, 2004.

[13] G. E. Crawford, I. E. Holt, J. Whittle, B. D. Webb, D. Tai, S. Davis, E. H. Margulies, Y. Chen, J. A. Bernat, D. Ginsburg, D. Zhou, S. Luo, T. J. Vasicek, M. J. Daly, T. G. Wolfsberg, and F. S. Collins. Genome-wide mapping of dnase hypersensitive sites using massively parallel signature sequencing (mpss). *Genome Research*, 16(1):123–131, 2006.

[14] M. de Berg, M. van Kreveld, M. Overmars, and O. Schwarzkopf. *Computational Geometry: Algorithms and Applications*. Springer-Verlag, second edition, 2000.

[15] M. J. de Hoon, P. Eichenberger, and D. Vitkup. Hierarchical evolution of the bacterial sporulation network. *Current Biology*, 20(17):R735 – R745, 2010.

[16] M. J. Donlin. Using the generic genome browser (gbrowse). *Curr Protoc Bioinformatics*, 2007.

[17] S. Dudoit, R. C. Gentleman, and J. Quackenbush. Open source software for the analysis of microarray data. *Biotechniques*, Suppl:45–51, Mar. 2003.

[18] P. G. Giresi, J. Kim, R. M. McDaniell, V. R. Iyer, and J. D. Lieb. FAIRE (Formaldehyde-Assisted Isolation of Regulatory Elements) isolates active regulatory elements from human chromatin. *Genome Research*, 17(6):877–885, 2007.

[19] B. Kim, B. Lee, S. Knoblach, E. Hoffman, and J. Seo. GeneShelf: A web-based visual interface for large gene expression time-series data repositories. *Visualization and Computer Graphics, IEEE Transactions on*, 15(6):905–912, Nov 2009.

[20] R. Kincaid, A. Kuchinsky, and M. Creech. VistaClara: An expression browser plug-in for Cytoscape. *Bioinformatics*, page 368, Aug. 2008.

[21] M. D. Meyer, T. Munzner, and H. Pfister. Mizbee: A multiscale synteny browser. *IEEE Transactions on Visualization and Computer Graphics*, 15(6):897–904, 2009.

[22] M. D. Meyer, B. Wong, M. P. Styczynski, T. Munzner, and H. Pfister. Pathline: A tool for comparative functional genomics. *Computer Graphics Forum*, 29(3):1043–1052, 2010.

[23] J. T. Robinson, H. Thorvaldsdttir, W. Winckler, M. Guttman, E. S. Lander, G. Getz, and J. P. Mesirov. Integrative Genomics Viewer. *Nature Biotechnology*, 29:24–26, 2011.

[24] K. R. Rosenbloom, C. A. Sloan, V. S. Malladi, T. R. Dreszer, K. Learned, V. Kirkup, M. C. Wong, M. Maddren, R. Fang, S. G. Heitner, B. T. Lee, G. P. Barber, R. A. Harte, M. Diekhans, J. C. Long, S. P. Wilder, A. S. Zweig, D. Karolchik, R. M. Kuhn, D. Haussler, and W. J. Kent. ENCODE data in the UCSC genome browser: year 5 update. *Nucleic Acids Research*, 41(Database-Issue):56–63, 2013.

[25] M. Schena, D. Shalon, R. W. Davis, and P. O. Brown. Quantitative monitoring of gene expression patterns with a complementary DNA microarray. *Science (New York, N.Y.)*, 270(5235):467–70, Oct. 1995.

[26] P. Shannon, A. Markiel, O. Ozier, N. S. Baliga, J. T. Wang, D. Ramage, N. Amin, B. Schwikowski, and T. Ideker. Cytoscape: A software environment for integrated models of biomolecular interaction networks. *Genome Research*, 13:2498–2504, 2003.

[27] P. Shannon, D. Reiss, R. Bonneau, and N. Baliga. The Gaggle: An open-source software system for integrating bioinformatics software and data sources. *BMC Bioinformatics*, 7(1):1–13, 2006.

[28] B. Tang, Q. Wang, M. Yang, F. Xie, Y. Zhu, Y. Zhuo, S. Wang, H. Gao, X. Ding, L. Zhang, G. Zhao, and H. Zheng. ContigScape: A Cytoscape plugin facilitating microbial genome gap closing. *BMC Genomics*, 14(1):289, 2013.

[29] The PostgreSQL Global Development Group. PostgreSQL. http://www.postgresql.org/.

[30] H. Thorvaldsdttir, J. T. Robinson, and J. P. Mesirov. Integrative Genomics Viewer (IGV): High-performance genomics data visualization and exploration. *Briefings in Bioinformatics*, 14(2):179–182, April 2012.

[31] UCSC Genome Bioinformatics. Wiggle track format (wig). http://genome.ucsc.edu/goldenPath/help/wiggle.html.

[32] Z. Wang, M. Gerstein, and M. Snyder. RNA-Seq: a revolutionary tool for transcriptomics. *Nature reviews. Genetics*, 10(1):57–63, Jan. 2009.