Supplementary Information for GraphSpace: Stimulating Interdisciplinary Collaborations in Network Biology

Aditya Bharadwaj[1], Divit P. Singh[1], Anna Ritz[2], Allison N. Tegge[1,3], Christopher L. Poirel[4], Pavel Kraikivski[5], Neil Adames[6], Kurt Luther[1,7], Shiv D. Kale[8], Jean Peccoud[6], John J. Tyson[5], and T. M. Murali[1,9]

[1]Department of Computer Science, Virginia Tech, Blacksburg, VA, USA
[2]Biology Department, Reed College, Portland OR, USA
[3]Department of Statistics, Virginia Tech, Blacksburg, VA, USA
[4]RedOwl Analytics, San Francisco, CA, USA
[5]Department of Biological Sciences, Virginia Tech, Blacksburg, VA, USA
[6]Department of Chemical and Biological Engineering, Colorado State University, Fort Collins, CO, USA
[7]Center for Human-Computer Interaction, Virginia Tech, Blacksburg, VA, USA
[8]Biocomplexity Institute, Virginia Tech, Blacksburg, VA, USA
[9]ICTAS Center for Systems Biology of Engineered Tissues, Virginia Tech, Blacksburg, VA, USA

Contents

S1 Comparison to Other Systems **2**

S2 Other Features of GraphSpace **4**

S3 GraphSpace Use Cases **5**

S4 Interacting with GraphSpace **6**
S4.1 Initial Interaction 6
S4.2 Search 8
S4.3 Graph Interaction 11
S4.4 Layouts 14
S4.5 Layout Editor 17
S4.6 Filtering Nodes and Edges 21
S4.7 Node and Edge Editor 24

`user2@example.com`: This user owns no networks but can access `user1`'s networks.

S1 Comparison to Other Systems

In the main text, we described five challenges faced by interdisciplinary research groups in network biology, which we repeat here for the reader's reference:

Sharing. Computational biologists may generate hundreds or thousands of networks that they seek to share without manual overhead.

Organization. A research group working on multiple projects may seek to keep the networks from different collaborations private from each other.

Searching and Filtering. Collaborators may search within and across these networks for proteins and interactions of interest. Collaborators may also want to examine subgraphs of a network.

Layouts. Automatic layout algorithms incorporate almost no knowledge of the biological information underlying the networks. Thus, researchers need layout tools that can help them readily use their knowledge and intuition to modify node positions manually to bring out salient features.

Release. Upon the publication of a project, researchers may seek to make a subset of networks publicly available for access.

We are not aware of any existing systems that address all these challenges simultaneously. Many packages focus primarily on algorithms for laying out networks and/or analyzing their properties (2, 3, 4, 5, 6, 20). Several websites that support user queries on proteins and interactions are typically tied to a particular set of algorithms and analyses, or specific interaction networks or databases (e.g., (1, 7, 18, 21)). NetworkPainter (8) supports several features for collaboration, but focuses on manually created network diagrams and overlaying flow cytometry data on them. CellMaps is a web tool that allows display, editing, exploring, and analysis of biological networks (14). It does not appear to have features that permit sharing of networks and layouts on the CellMaps website.

The Cytoscape ecosystem provides solutions to some of the above challenges. Cytoscape (17) itself is a popular and powerful software environment for visualizing and interacting with networks. Cytoscape.js is a library of graph algorithms for analysis and visualization (5). CyNetShare (`http://idekerlab.github.io/cy-net-share`) provides a simple mechanism for a user of Cytoscape to export a network as a file and point the CyNetShare website to the location of this file, which allows users to share the URL of the network layout with collaborators. Since CyNetShare is designed for sharing networks one at a time, it does not readily scale to multiple networks. NDex (11) is a repository designed for storing networks that supports many file formats, network queries, and the creation of groups for collaboration. However, NDex does not allow users to modify, save, or share network layouts. While we developed GraphSpace groups and the notion of private collaboration vs. public sharing independently of NDex, the two resources are similar in this respect.

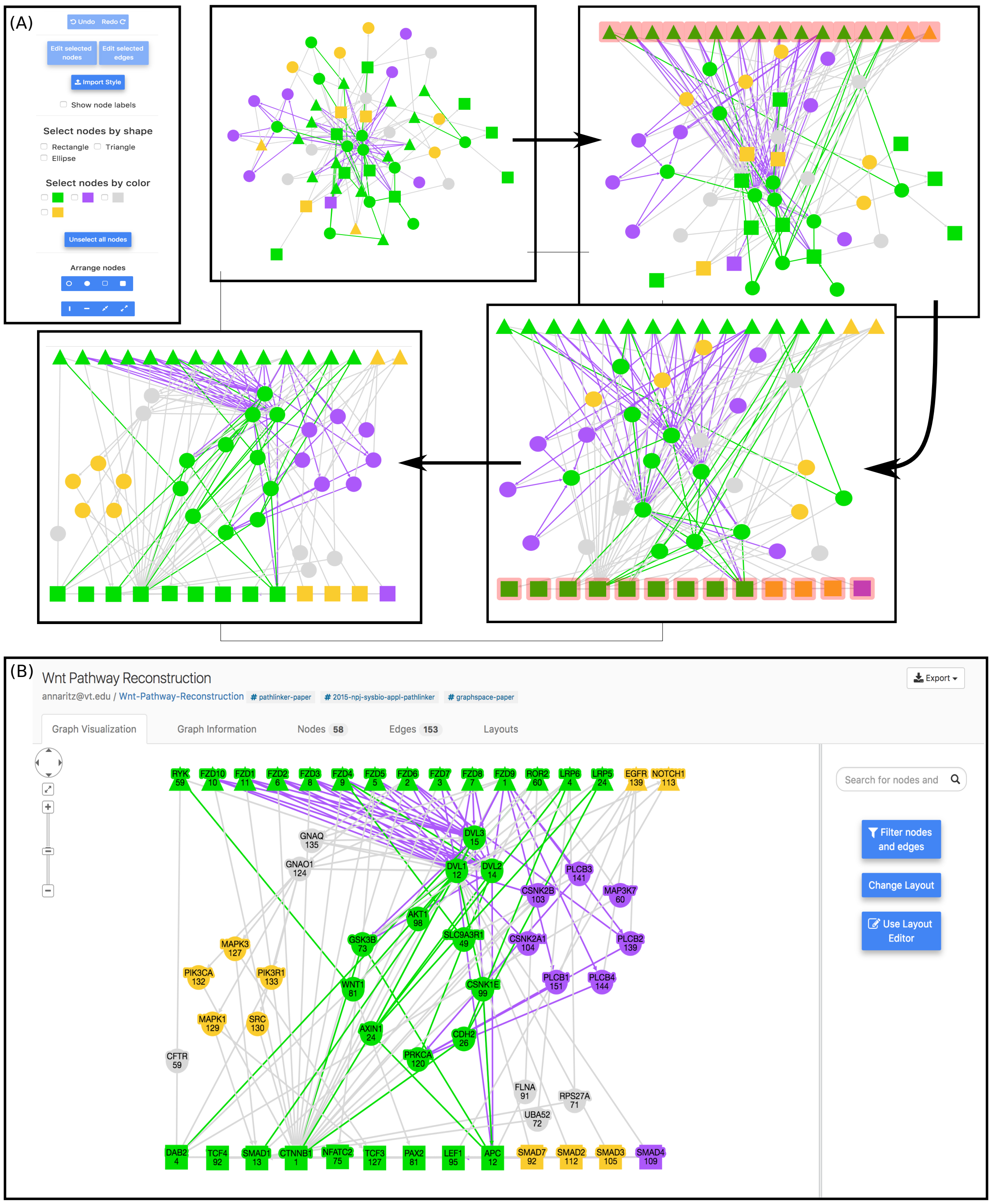


Figure S1: The reconstruction of the Wnt pathway by the PathLinker algorithm (13) available at `http://graphspace.org/graphs/public?tags=graphspace-paper`. (A) The layout editor provides an intuitive tool palette that allows the user to group nodes by colors and/or shapes and arrange them in blocks, circles, or lines to quickly build a custom-made manual layout of large networks. (B) The final layout of the network visualized in GraphSpace. The RYK-CFTR-DAB2 path referred to in the PathLinker use case in Section S3 appears on the left. The "Graph Information" tab displays a user-provided legend of shapes and colors. "Export" allows saving the visualization of a network in PNG and JPEG formats. This button also allows saving the structure and visual style of a network in JSON-formatted files.

S2 Other Features of GraphSpace

Here, we describe some other features of GraphSpace.

Network format. GraphSpace networks follow a Cytoscape.js supported JSON format that separates the specification of the network structure (nodes and edges) from the description of the visual styles of the nodes and edges (e.g., colors, widths, shapes, labels). The same format can be exported from or imported to Cytoscape (version 3.1.1 or greater). GraphSpace treats some attributes specially:

- Cytoscape.js supports a network-level "data" section in the JSON file specifying the network structure. In this section, we allow users to specify attributes such as "name", "title", "description", and "tags". GraphSpace displays the "name" attribute to identify networks in the list that a user can access and in the list that match search results. When the user accesses a specific network, GraphSpace displays the "title" above the layout of the graph and the content of the "description" attribute in the tab called "Graph Information" (Supplementary Figure S10) We also allow the "data" section to include network-specific attributes in this section, e.g., "PMID", "authors", and "organism". GraphSpace displays all these additional attributes in the "Graph Information" tab as well.
- Cytoscape.js also supports a "data" section within each node. Currently, GraphSpace recognizes an attribute called "aliases" in this section, which specifies a list of aliases for the node.
- GraphSpace adds a "popup" attribute to each node and edge that allows a user to add HTML-formatted information, e.g., Gene Ontology annotations and database links for a protein; or types, mechanism, and database sources for an interaction (Supplementary Figures S11 and S12).
- Each node and edge may also have an integer-valued attribute called "k", which denotes a rank. Through this attribute, GraphSpace allows the user to filter nodes and edges in a network visualization (Supplementary Section S4.6).

Search Syntax. GraphSpace supports three types of search terms:

1. a single string, e.g., "wnt". In this case, GraphSpace will return a network (a) if its "name" attribute contains the query as a substring or (b) if any node in the network has a "name", "label", or "aliases" attribute that contains the query as a substring.
2. two strings separated by a colon, e.g., "name:wnt". Here, GraphSpace will return a network if the "data" section of its JSON representation contains an attribute called "name" whose value contains "wnt" as a substring.
3. two strings separated by two colons, e.g., "wnt::fzd": This type of search term is only available when a user is searching a specific network. GraphSpace will highlight every edge that connects a node that matches "wnt" to a node that matches "fzd". This search ignores the direction of the edge.

All searches are case-insensitive. A user may specify more than one search term. When the user searches all networks, GraphSpace returns only those networks that match all the search terms. When the user is visualizing an individual network and searching within it, GraphSpace highlights nodes and edges that match *any* search term (Supplementary Figure S8).

Tags. GraphSpace uses tags as a mechanism for grouping a set of graphs, e.g., all the graphs that a user may desire to make public upon the publication of a paper. A graph may have any number of tags. GraphSpace includes an interface to search for networks that match one or more tags. Tags, which organize graphs, provide a system that is orthogonal to groups, which organize collaborators.

Programmatic access. The GraphSpace REST API provides endpoints for entities such as graphs, layouts, and groups that allow developers to interact with the GraphSpace website remotely by sending and receiving JSON objects. This API enables developers to create, read, and update GraphSpace content from client-side JavaScript or from applications written in any language. After a network is uploaded, the API allows the network owner to modify, delete, and share it. The API also allows a user to access several group management features available through the web interface. For example, a user can create or remove a group, add or remove members, obtain a list of groups he or she belongs to, and get information such as the membership on a group.

Python module. The GraphSpace software also includes a simple yet powerful Python library called "graphspace_python" that allows a user to rapidly construct a network, add nodes and edges, modify their visual styles, and then upload the network, all within tens of lines of code. Moreover, the user need not know the details of the REST API to use this module. It is very easy to integrate this library into a user's software pipeline.

GraphSpace implementation. GraphSpace uses Django (`https://www.djangoproject.com`), a popular, high-level Python web framework, for server-side functionality. It uses PostgreSQL for storing and querying its internal database of users, groups, graphs, nodes, edges, and layouts. GraphSpace also uses ElasticSearch as a NoSQL datastore for the JSON documents storing the network structure. This Apache Lucene-based indexing engine allows us to perform queries on the contents of the "data" attributes of networks and nodes. The distributed nature of ElasticSearch also offers us the ability to easily scale our systems to accommodate increased bandwidth in the future. GraphSpace takes advantage of three open-source JavaScript libraries for its client-side functions: (i) Bootstrap (`http://www.getbootstrap.com`), an HTML, CSS, and JavaScript framework for the user interface; (ii) JQuery (`http://www.jquery.com`) for easy access to the Document Object Model (DOM), fast event handling, and Asynchronous JavaScript and XML (AJAX) calls; and (iii) Cytoscape.js (5) for network layout and visual styles for nodes and edges.

S3 GraphSpace Use Cases

We provide two examples of how we have used GraphSpace in collaborations. First, we sought to determine regulatory mechanisms to extend a mathematical model of the budding yeast cell cycle (10). For several query proteins, we computed the 50 shortest paths in a yeast interactome to the proteins in the model. We uploaded these networks to GRAPHSPACE, annotating each node with the links to the Saccharomyces Genome Database and each edge with the source database and PubMed ID of the publication. These networks are available at `http://graphspace.org/graphs/public?tags=2013-jcb-linker`. Examination of the network for CDC5 (Polo kinase) revealed an unexpected connection between the proteins Bub2 and Tem1. The publication that reported this interaction provided evidence that allowed the modification of the mathematical model to correctly simulate the phenotype of yeast cells lacking Cdc5 and Bub2 (10).

Next, we designed an algorithm that reconstructs a human signaling pathway within an interactome when provided only the receptor and transcriptional regulator (TR) proteins in that pathway (13). For over 20 pathways, we visualized the 200 shortest paths from any receptor to any transcriptional regulator using GraphSpace. The filtering panel on the reconstruction of the Wnt signaling pathway revealed that the RYK-CFTR-DAB2 was the first receptor-to-TF path that involved a protein (CFTR) not already known to be a member of the Wnt pathway (left-most path in Figure S1(B)). Subsequent siRNA experiments in HEK290 cells confirmed CFTR's role in canonical Wnt signaling and suggested the RYK-CFTR-DAB2 path as an amplifier of Wnt signaling to β-catenin (13). These pathway reconstructions are available at `http://graphspace.org/graphs/public?tags=2015-npj-sysbio-appl-pathlinker`.

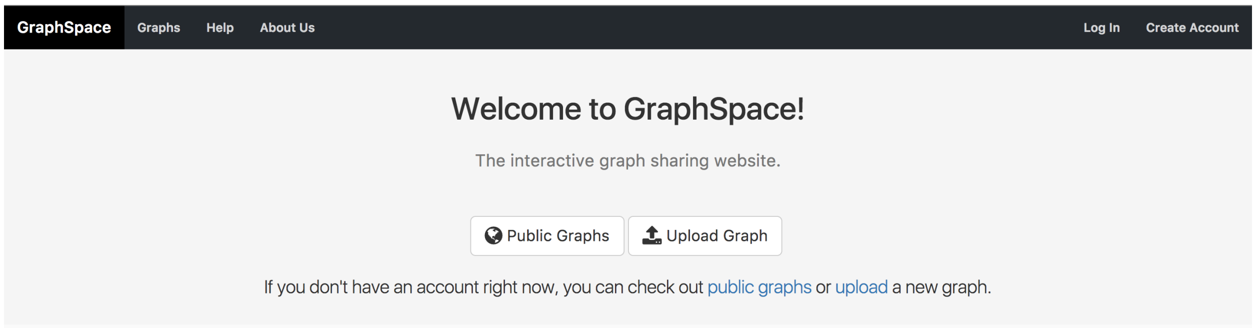
Since then, GraphSpace has proven useful for other projects involving remote collaborators spanning multiple scientific disciplines (13, 15, 19) and in diverse contexts, such as exploring significant transcriptional links among biological processes (9), studying multi-way molecular reactions (12), and exploring crowd-driven network layout methodologies (16). GraphSpace has also been an invaluable tool for visually debugging initial implementations of network biology algorithms. We have adopted GraphSpace for multiple undergraduate research and capstone projects, demonstrating its ease of use and potential as a teaching tool. Recently, several undergraduate students used the "graphspace_python" module to construct and post networks to GraphSpace as part of a computational biology course at Reed College. The majority of these students were biology majors with very little programming experience.

S4 Interacting with GraphSpace

In this section, we use screenshots and their captions to describe how a user may interact with the GraphSpace web interface.

S4.1 Initial Interaction

The welcome page greets a user when they visit GraphSpace (Figure S2). From this page, a user can view the networks that have been shared with the public ("Public Graphs") or upload their own graphs. Users may also create an account on GraphSpace.



Welcome to GraphSpace!

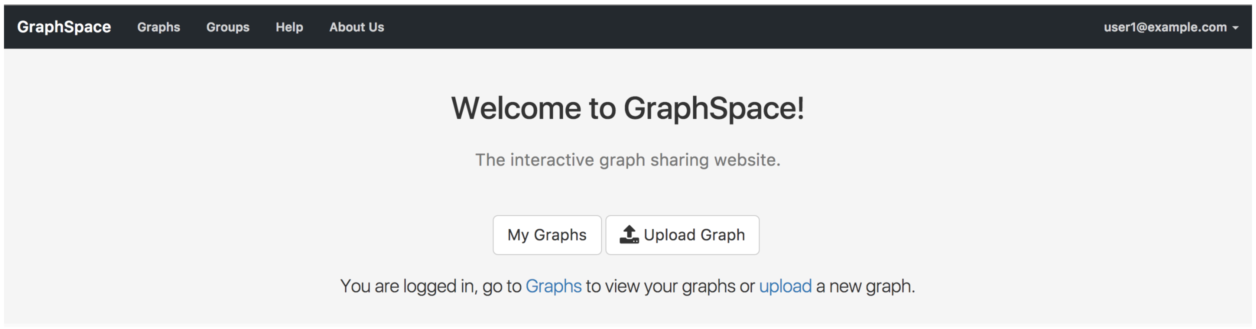
The interactive graph sharing website.

If you don't have an account right now, you can check out public graphs or upload a new graph.

Figure S2: The welcome page, before logging in.

For the rest of the tutorial, we assume that the user has already created an account. We also assume that the user belongs to a group with shared networks or that the user uploaded networks via the upload page or the RESTful API. The tutorial specifically steps through how `user1@example.com` interacts with GraphSpace after logging in.

After creating an account and logging in, a user is directed to the list of networks that they can access ("My Graphs" in Figure S3).



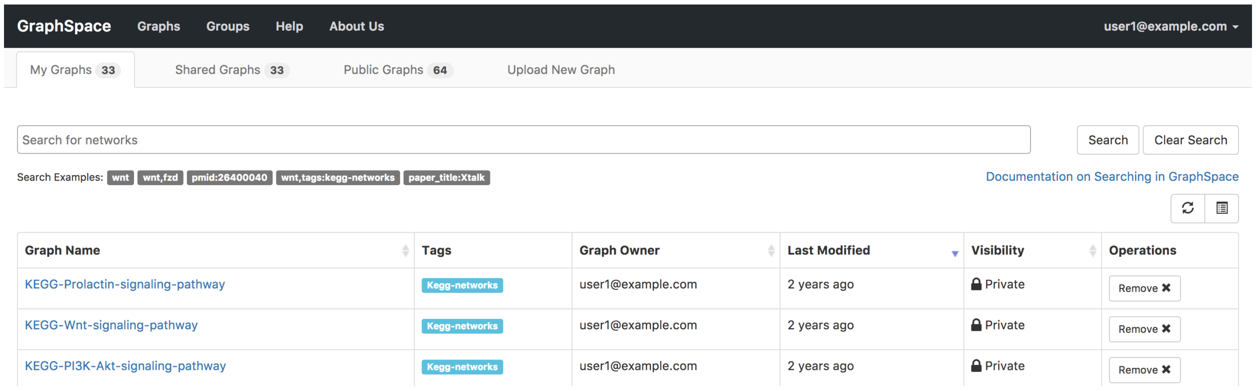
Welcome to GraphSpace!

The interactive graph sharing website.

You are logged in, go to Graphs to view your graphs or upload a new graph.

Figure S3: The welcome page, after logging in.

Clicking on the link titled "My Graphs" takes the user to a page that lists the networks accessible by the user. In this example, the user (`user1@example.com`) owns 33 networks, has 33 shared networks (this user has shared all owned networks with the group "Test Group"), and can access 64 public networks (Figure S4). The user can search this list of networks as shown in Figures S5–S8.



Search Examples: wnt wnt,fzd pmid:26400040 wnt,tags:kegg-networks paper_title:Xtalk

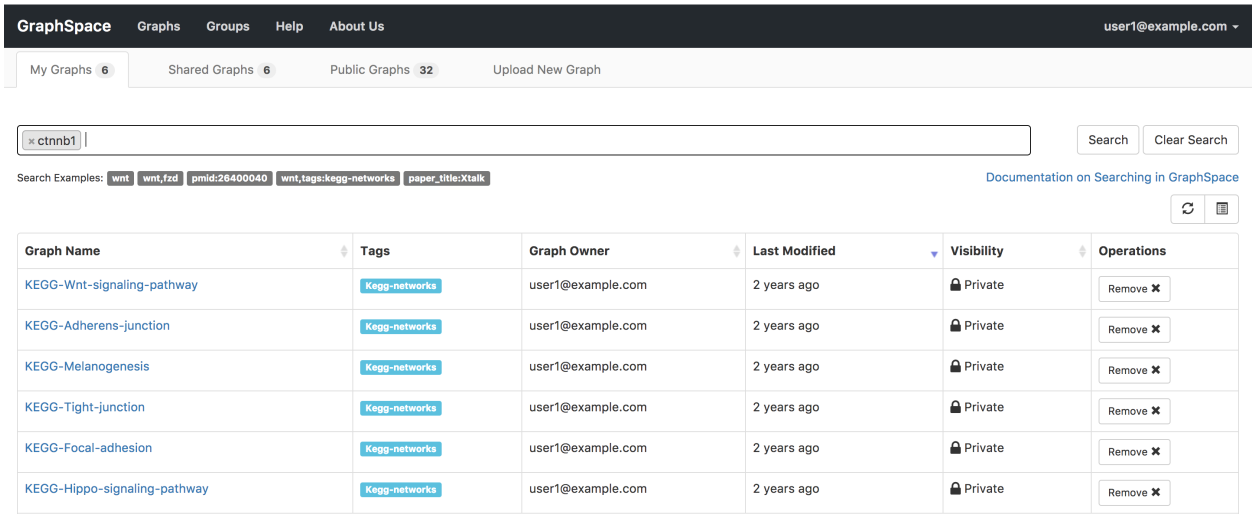
Documentation on Searching in GraphSpace

Graph Name	Tags	Graph Owner	Last Modified	Visibility	Operations
KEGG-Prolactin-signaling-pathway	Kegg-networks	user1@example.com	2 years ago	Private	Remove ✖
KEGG-Wnt-signaling-pathway	Kegg-networks	user1@example.com	2 years ago	Private	Remove ✖
KEGG-PI3K-Akt-signaling-pathway	Kegg-networks	user1@example.com	2 years ago	Private	Remove ✖

Figure S4: The Graphs page.

S4.2 Search

The user searches this list for networks that contain the protein CTNNB1, the symbol for β-catenin, a transcriptional regulator in the Wnt signaling pathway, obtaining a reduced list of networks. Any node whose "label", "name" or "aliases" attribute contains "ctnnb1" as a substring will match this search. The match is case-insenstive. There are six networks owned by the user and thirty-two public networks that contain this protein (Figure S5).



Graph Name	Tags	Graph Owner	Last Modified	Visibility	Operations
KEGG-Wnt-signaling-pathway	Kegg-networks	user1@example.com	2 years ago	Private	Remove ✖
KEGG-Adherens-junction	Kegg-networks	user1@example.com	2 years ago	Private	Remove ✖
KEGG-Melanogenesis	Kegg-networks	user1@example.com	2 years ago	Private	Remove ✖
KEGG-Tight-junction	Kegg-networks	user1@example.com	2 years ago	Private	Remove ✖
KEGG-Focal-adhesion	Kegg-networks	user1@example.com	2 years ago	Private	Remove ✖
KEGG-Hippo-signaling-pathway	Kegg-networks	user1@example.com	2 years ago	Private	Remove ✖

Figure S5: The Graphs page, after searching for "ctnnb1" as a substring within the networks.

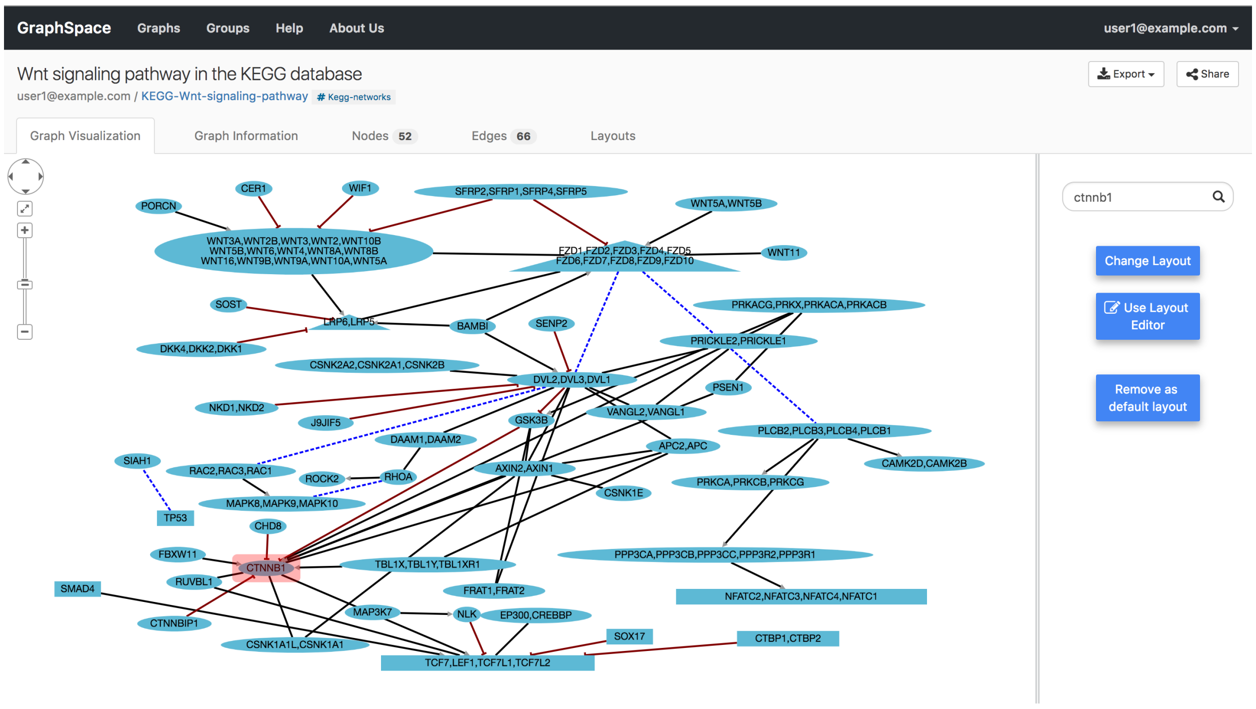


Figure S6: The user clicks on the network with the name "KEGG-Wnt-signaling-pathway" and reaches the network for the Wnt pathway with the searched node highlighted. This network is available at `http://graphspace.org/graphs/user1@example.com/KEGG-Wnt-signaling-pathway`

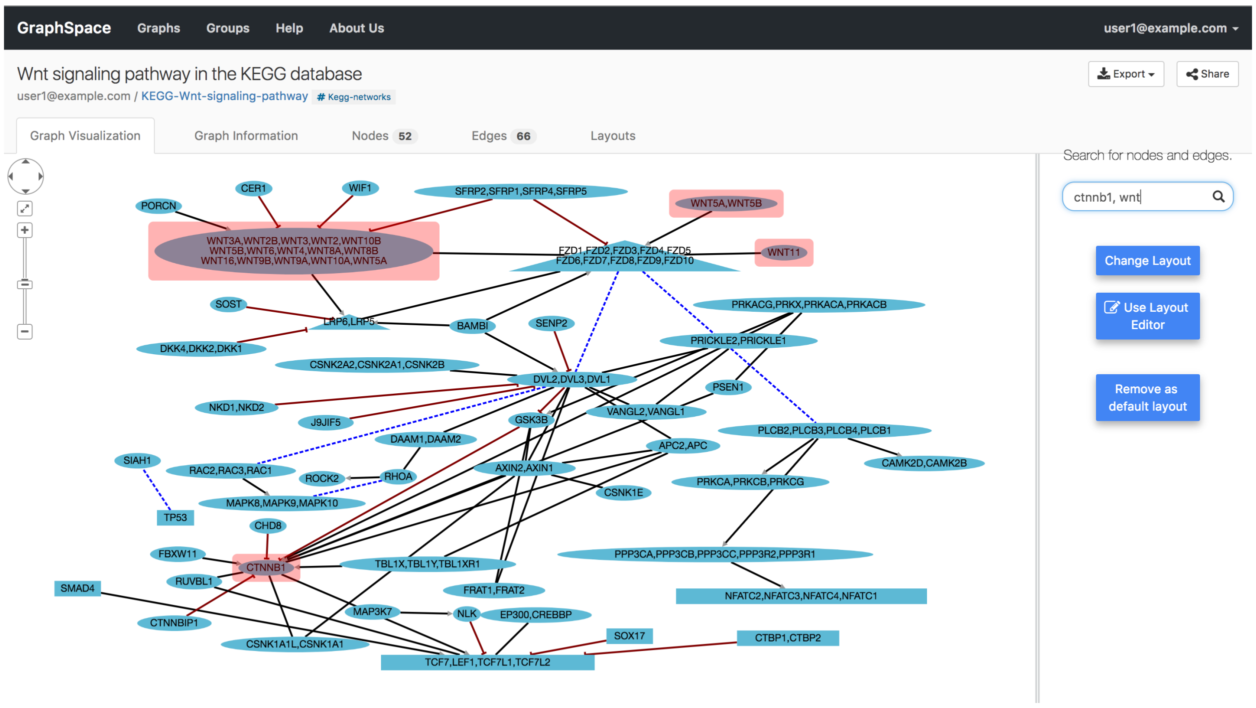


Figure S7: The search interface allows users to add more search terms using a comma-separated list. In this example, the user also searches for "wnt".

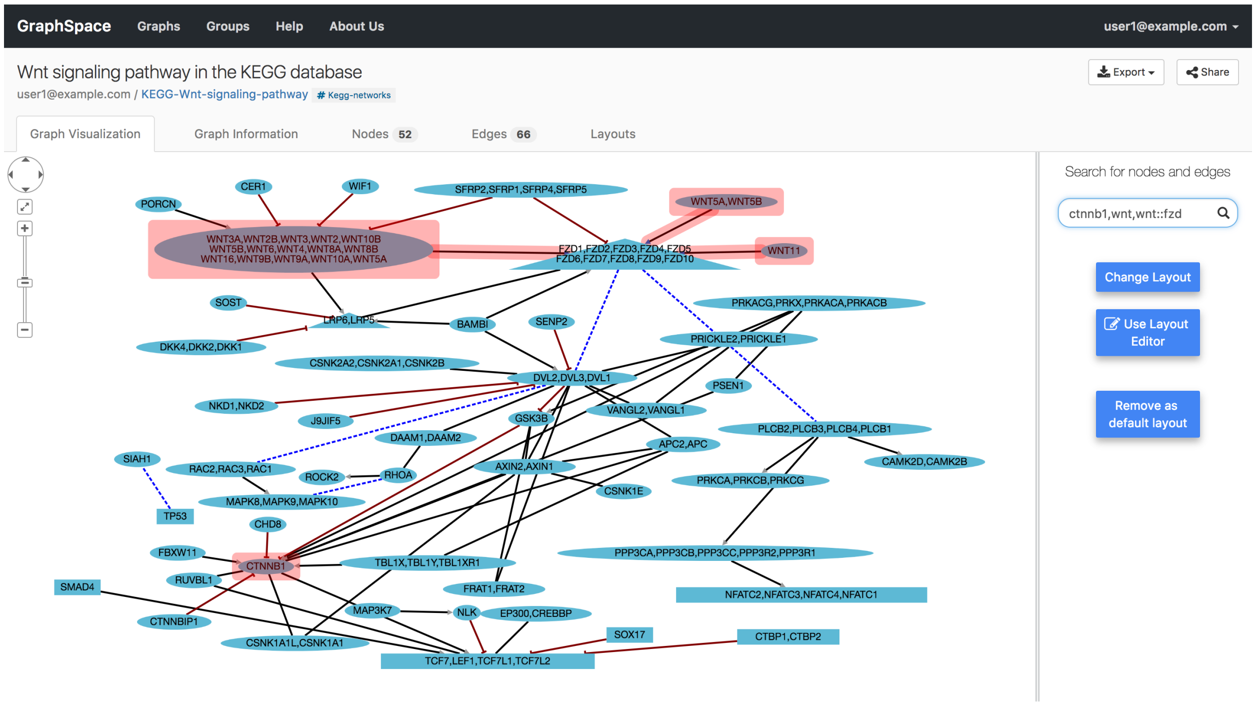


Figure S8: The user can also add search terms for edges ("wnt::fzd" in this example). GraphSpace highlights any edge whose tail node matches "wnt" and whose head node matches "fzd".

S4.3 Graph Interaction

In this section, we examine different ways to interact with an individual network on its page. The information that appears in Figures S9–S12 must be included in the JSON files that are uploaded by the network owner, as described in the “Network format” paragraph in Section S2.

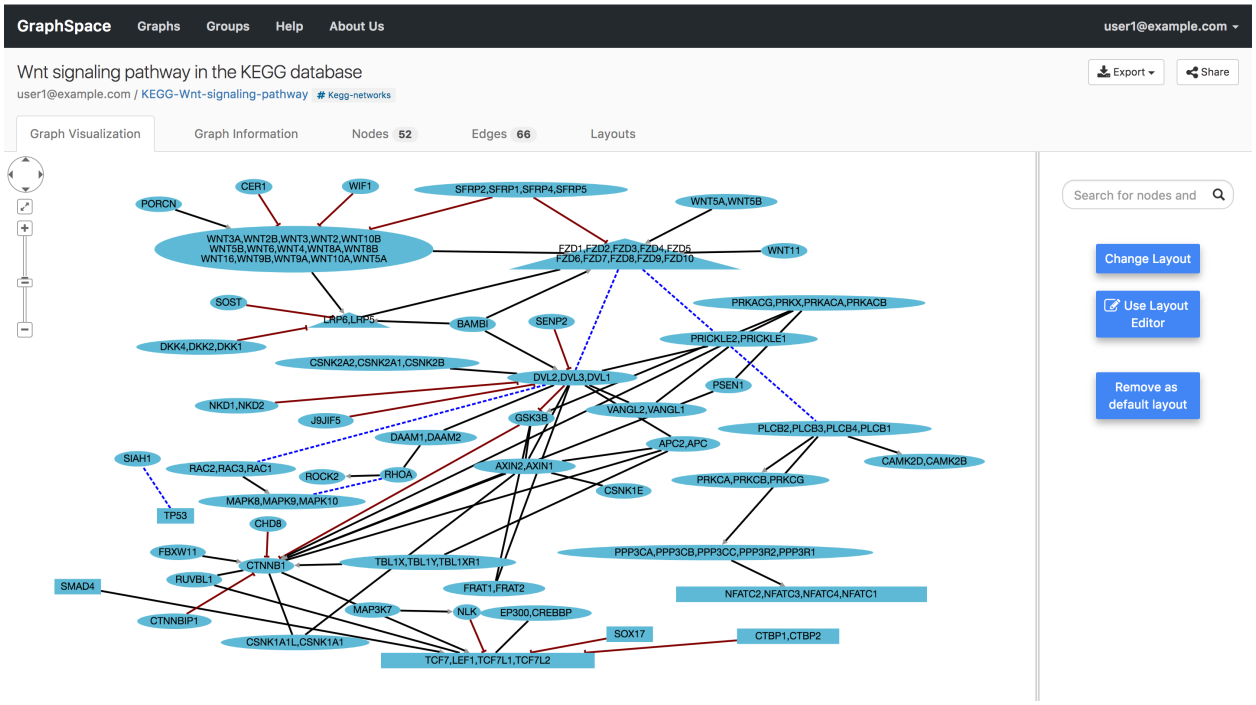
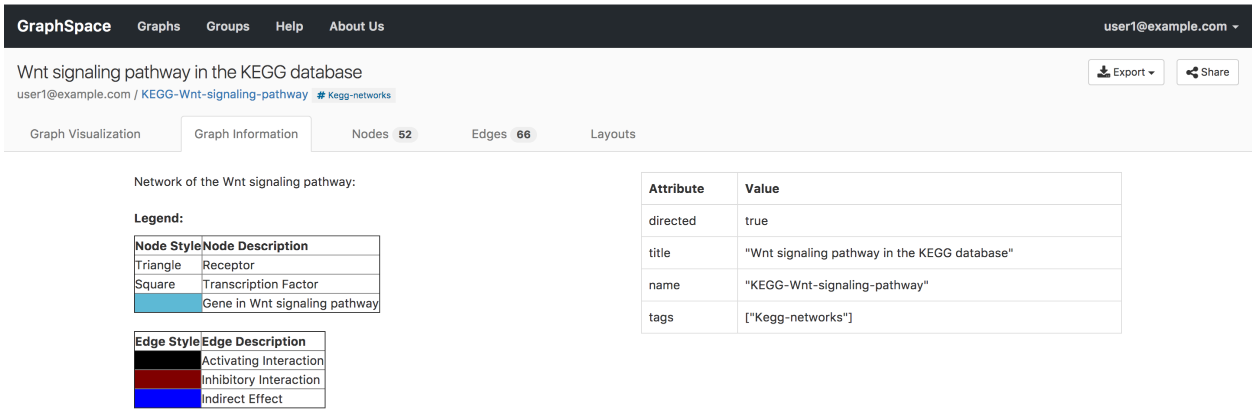


Figure S9: The “Graph Visualization” tab provides a visualization of the network.


GraphSpace Graphs Groups Help About Us user1@example.com

Wnt signaling pathway in the KEGG database

user1@example.com / KEGG-Wnt-signaling-pathway # Kegg-networks

Export Share

Graph Visualization | Graph Information | Nodes 52 | Edges 66 | Layouts

Network of the Wnt signaling pathway:

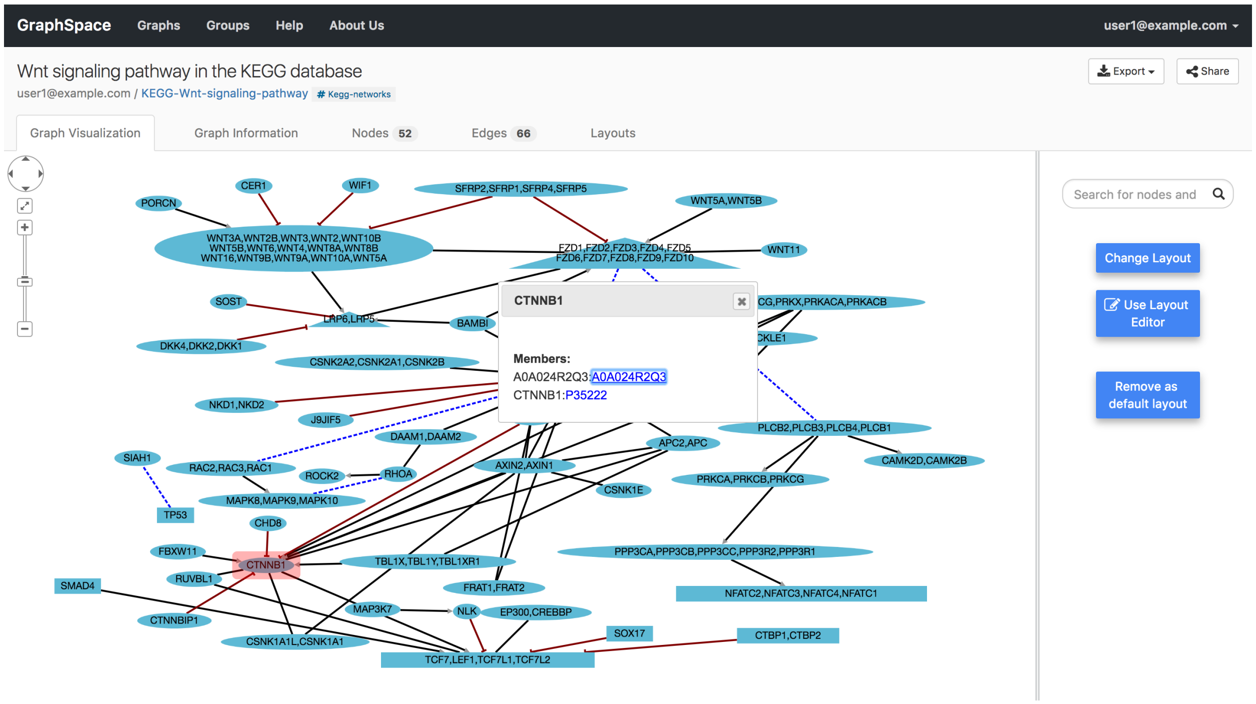
Legend:

Node Style	Node Description
Triangle	Receptor
Square	Transcription Factor
🟦	Gene in Wnt signaling pathway

Edge Style	Edge Description
⬛	Activating Interaction
🟥	Inhibitory Interaction
🟦	Indirect Effect

Attribute	Value
directed	true
title	"Wnt signaling pathway in the KEGG database"
name	"KEGG-Wnt-signaling-pathway"
tags	["Kegg-networks"]

Figure S10: The “Graph Information” tab displays information about the network, e.g., a legend of node and edge shapes and colors. The “description” attribute in the JSON for the network specifies this content.


GraphSpace Graphs Groups Help About Us user1@example.com

Wnt signaling pathway in the KEGG database

user1@example.com / KEGG-Wnt-signaling-pathway # Kegg-networks

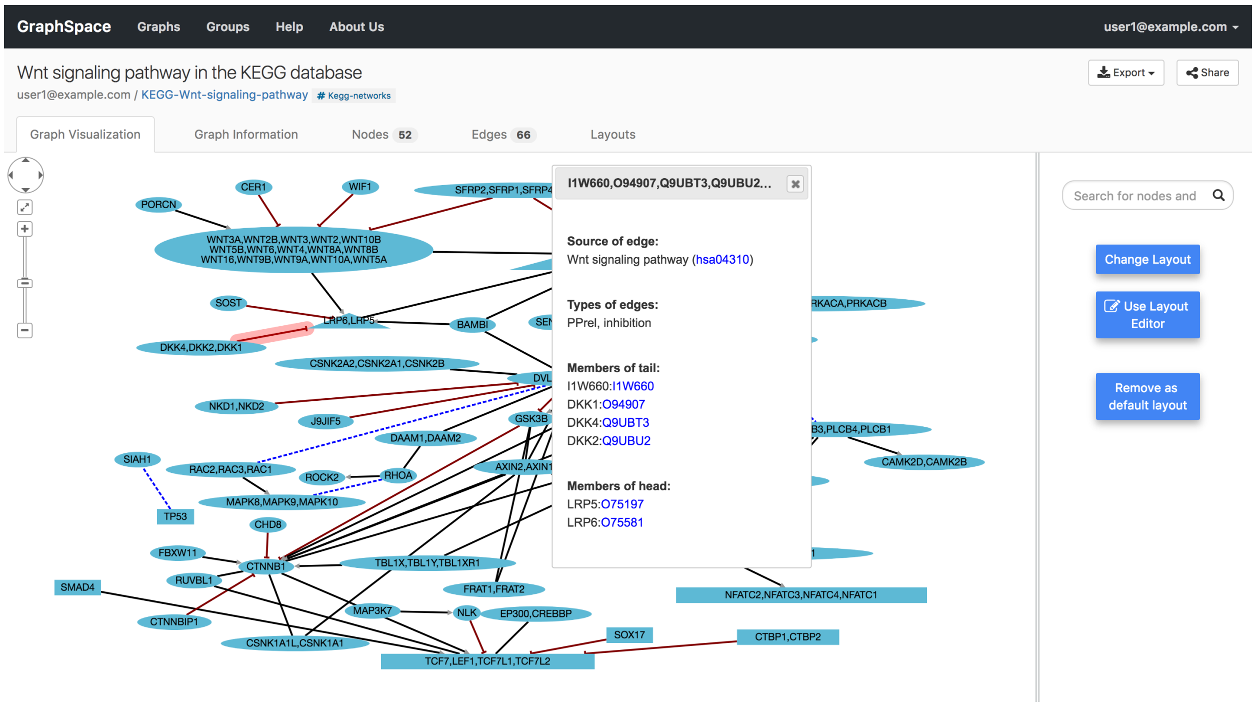
Export Share

Graph Visualization | Graph Information | Nodes 52 | Edges 66 | Layouts

CTNNB1

Members:
A0A024R2Q3:A0A024R2Q3
CTNNB1:P35222

Figure S11: Clicking on a node pops up a panel with information on that node. The “popup” attribute for the node in the network JSON specifies this content.



Wnt signaling pathway in the KEGG database

user1@example.com / KEGG-Wnt-signaling-pathway # Kegg-networks

I1W660,O94907,Q9UBT3,Q9UBU2...

Source of edge:
Wnt signaling pathway (hsa04310)

Types of edges:
PPrel, inhibition

Members of tail:
I1W660:I1W660
DKK1:O94907
DKK4:Q9UBT3
DKK2:Q9UBU2

Members of head:
LRP5:O75197
LRP6:O75581

Figure S12: Clicking on an edge pops up a panel with information on that edge. The "popup" attribute for the edge in the network JSON specifies this content.

S4.4 Layouts

Layouts provide a powerful means to organize nodes within a network. Figures S13–S16 illustrate how a user may access and view automatically generated and previously saved layouts.

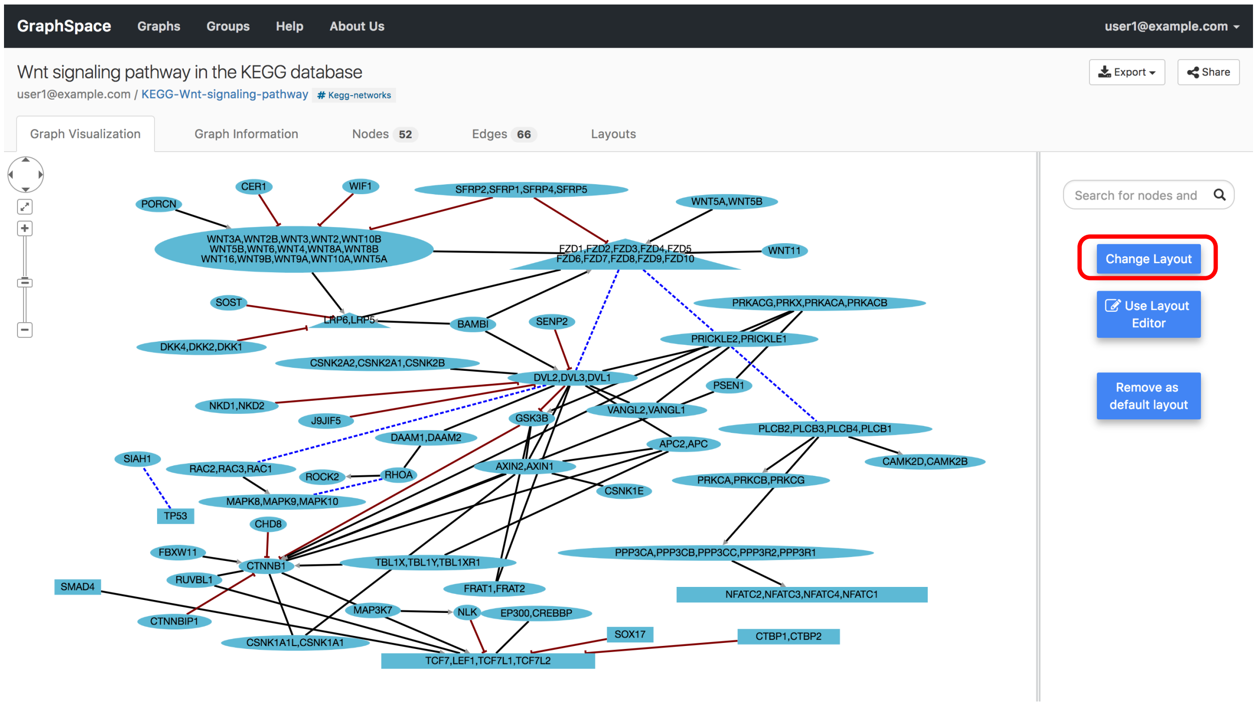


Figure S13: The user clicks on the “Change Layout” button.

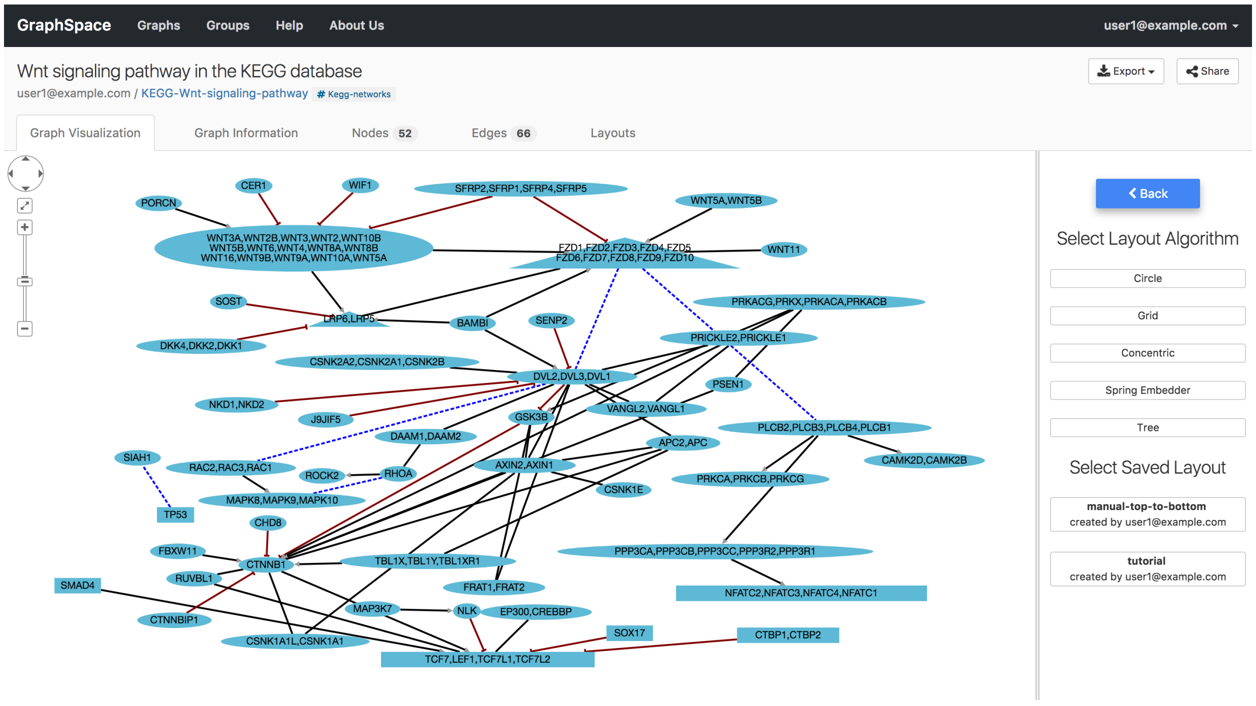


Figure S14: Clicking on the “Change Layout” button displays a layout panel which provides two alternatives. i) “Select Layout Algorithm”: This section lists all the automatic network layout algorithms supported by GraphSpace through its use of Cytoscape.js. ii) “Select Saved Layout”: This section lists layouts saved by the user. The user has created them in earlier sessions by using the layout editor or manually modifying the positions of nodes and edges created by some automatic layout algorithm and saving the layout.

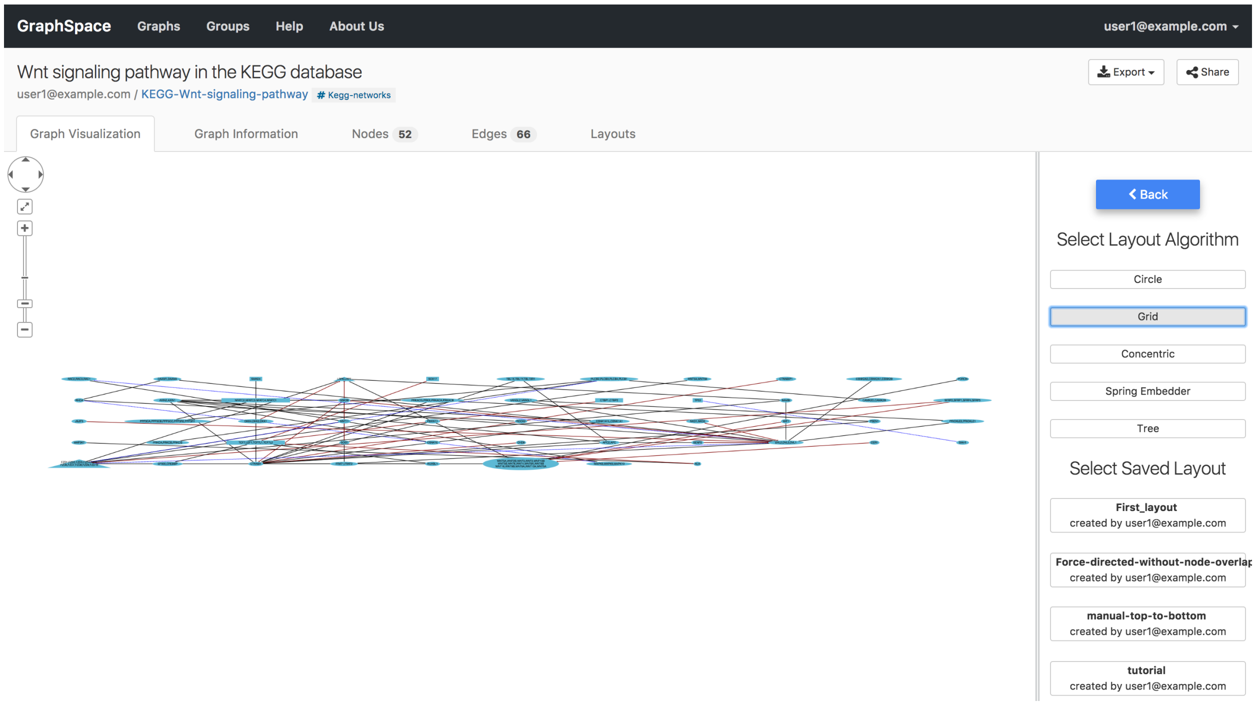


Figure S15: The user selects the "Grid" algorithm and obtains the displayed layout. The "Grid" layout algorithm places nodes in a well-spaced grid.

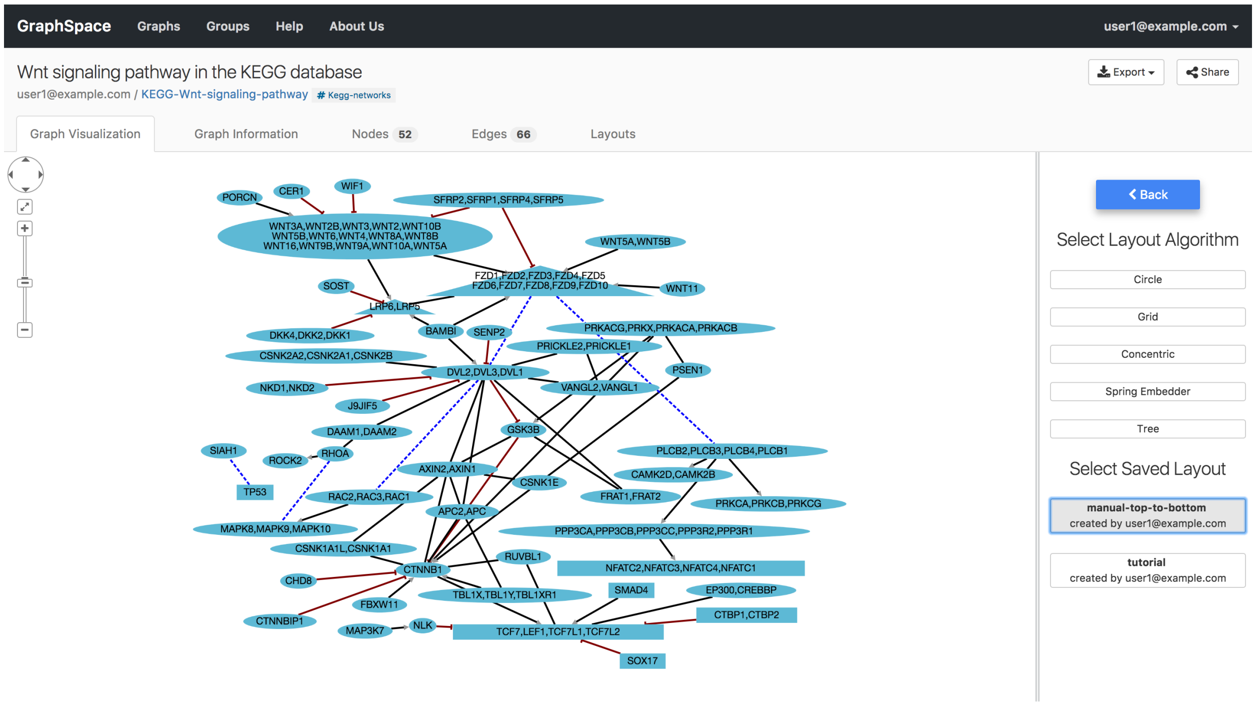


Figure S16: Here, the user opts to view the saved layout titled "manual-top-to-bottom". To create this layout, the user manually moved nodes so that extracellular ligands and receptors (triangles) appear on the top while transcription factors (rectangles) appear at the bottom. The user can click on the "Back" button to return to the main menu.

S4.5 Layout Editor

In this section, we use a different network from the one shown in earlier figures to illustrate the features of the layout editor (Figures S17–S21). This public network is available at `http://graphspace.org/graphs/annaritz@vt.edu/Wnt-Pathway-Reconstruction`. The layout editor allows the user to group nodes by colors and/or shapes and arrange them in blocks, circles, or lines to quickly build a custom-made manual layout of large networks. The editor also allows the user to change the visual appearance of nodes and edges. We describe these features in Section S4.7.

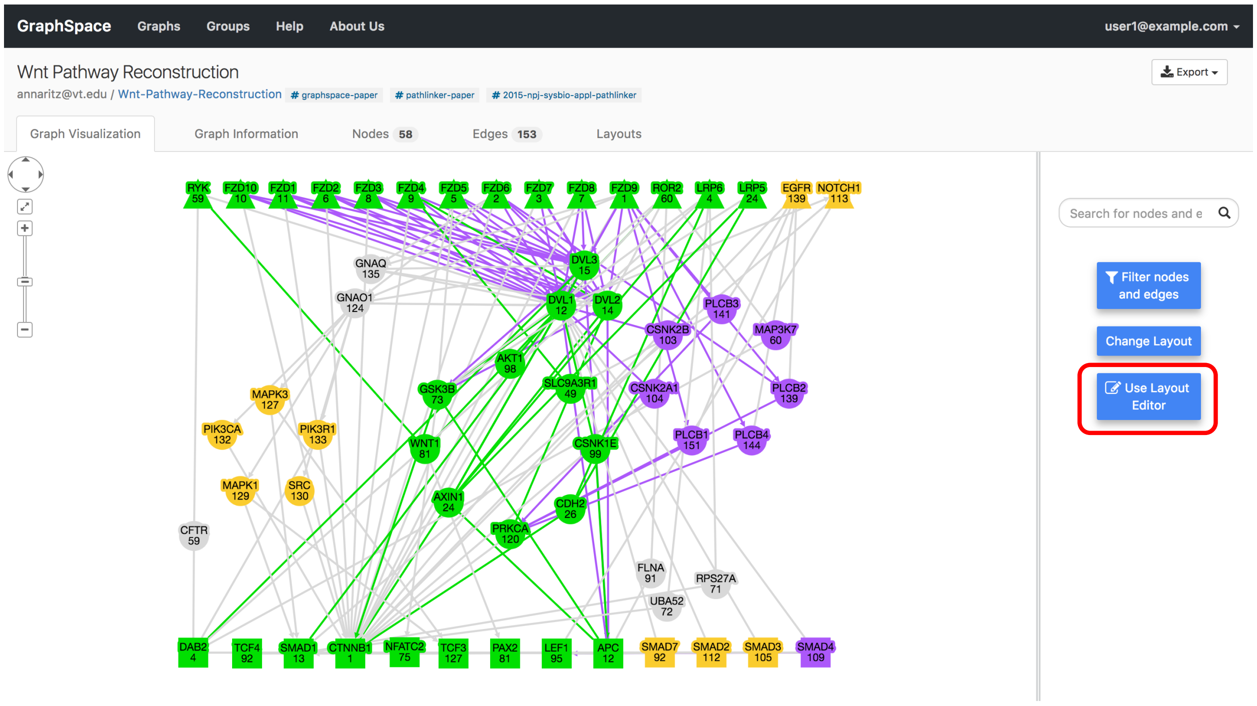


Figure S17: The user clicks on “Use Layout Editor” button available on the “Graph Visualization” tab to enter the layout editor mode.

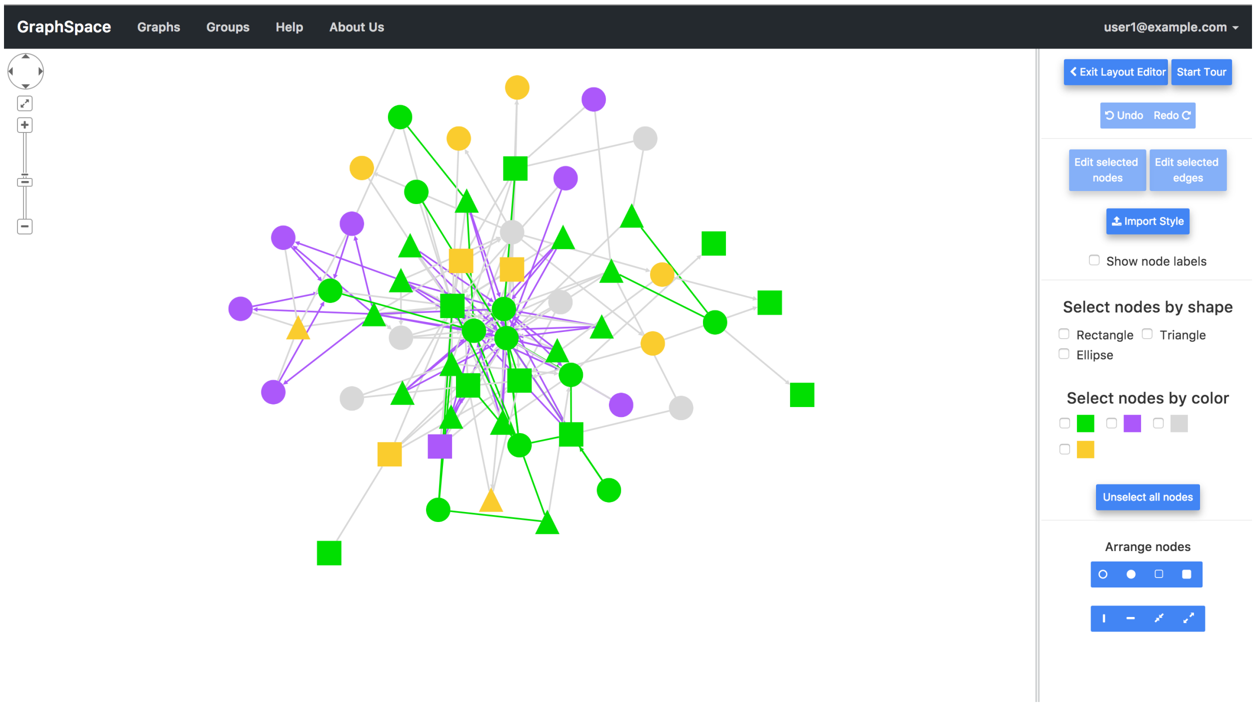


Figure S18: The user reaches this page after clicking the "Use Layout Editor" button. The panel on the right side of the page allows the user to select nodes by color (green/purple/gray/orange in this network), select nodes by shape (rectangle/triangle/ellipse in this network), or a combination of these choices. After selecting nodes, a user can arrange the nodes in an empty or filled circle, an empty or filled rectangle, or a horizontal or vertical line. The user can also adjust the spacing between nodes.

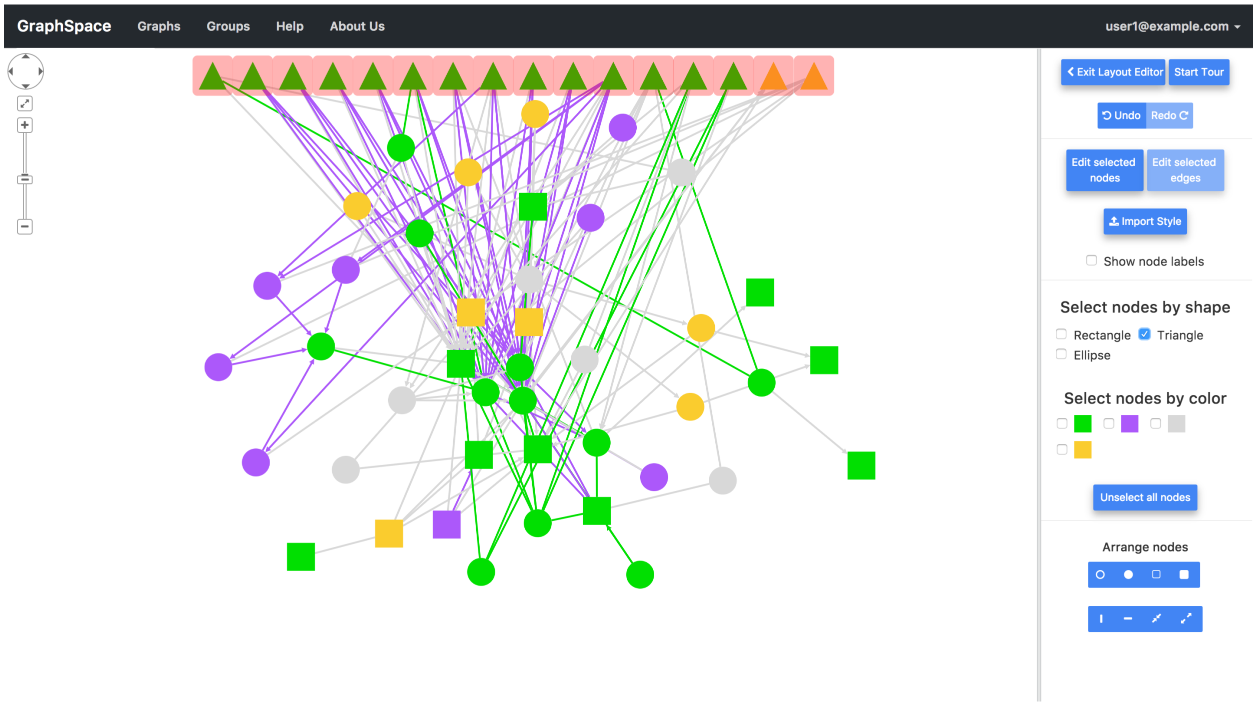


Figure S19: The user selects all triangles, arranges them in a horizontal line, and moves them to the top of the layout.

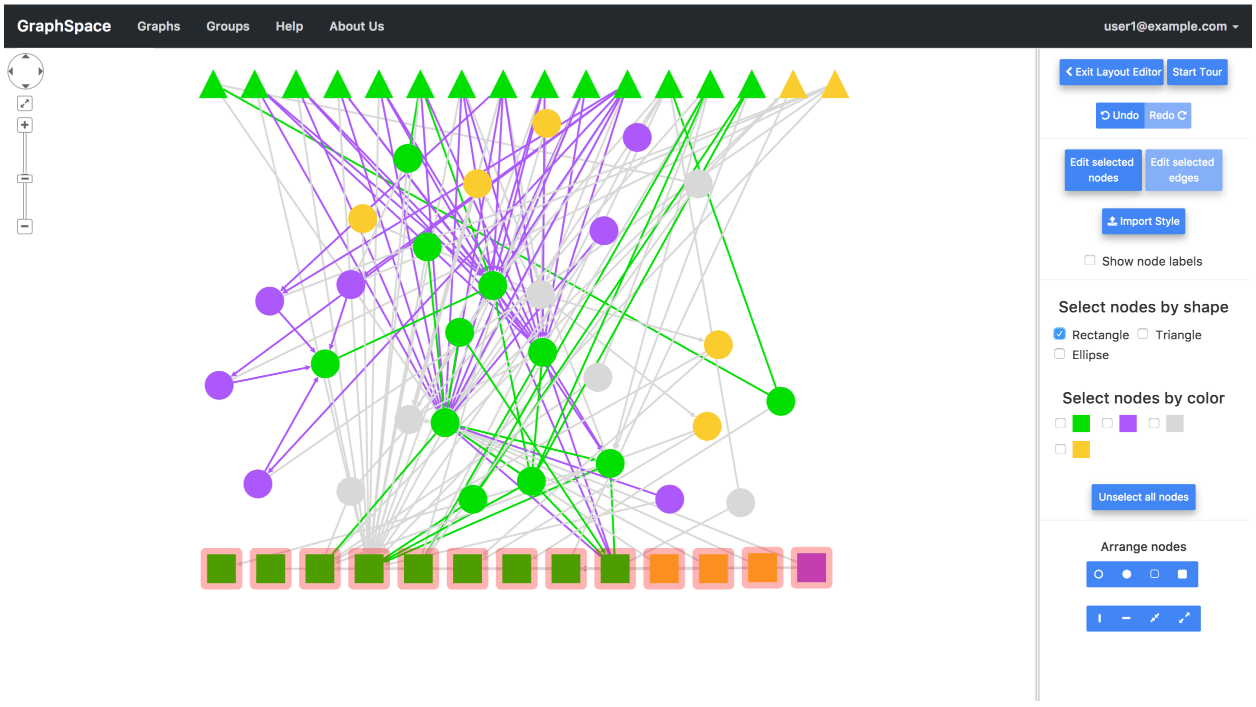


Figure S20: The user selects all rectangles, arranges them in a horizontal line, and moves them to the bottom of the layout.

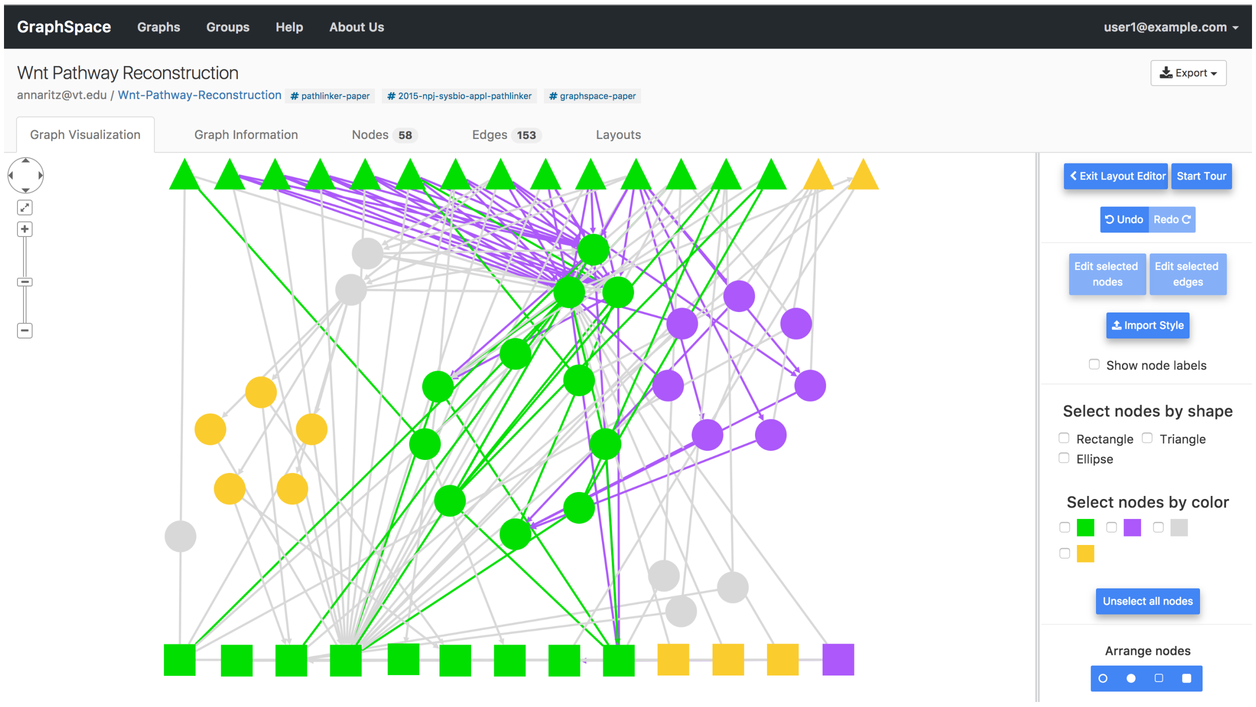


Figure S21: The user selects ellipses of each color, and arranges the nodes of each color in a circle, yielding the layout shown here. When the user presses the "Exit layout editor" button, GraphSpace prompts the user for a name for the layout. The user is then taken to the "Graph Visualization" tab (Figure S23). Clicking on the "Layouts" tab allows the user to manage the layouts, e.g., including sharing them with other users, deleting them, and renaming them. We do not display this functionality in this supplement.

S4.6 Filtering Nodes and Edges

Graph algorithms may output networks where nodes and edges can be ranked, e.g., by path index or by weight/score. GraphSpace allows each node and edge to have an integer-valued data-attribute called “k” that specifies the rank of the node or the edge. For any network that contains this attribute (and only for such networks), GraphSpace displays the “Filter nodes and edges” button. Selecting this button take the user to a page that shows a “Current rank” slider on the right. Changing the value in this slider hides all nodes and edges whose “k” values are larger than the value in the slider. The possible values in this slider range from 1 to the maximum value of “k” in the network. The figures in this section (Figures S23- S26) display what happens for different values of this slider. This interface element allows the user to unveil the network gradually in real time and gain intuition about how the network expands or contracts as this threshold changes.

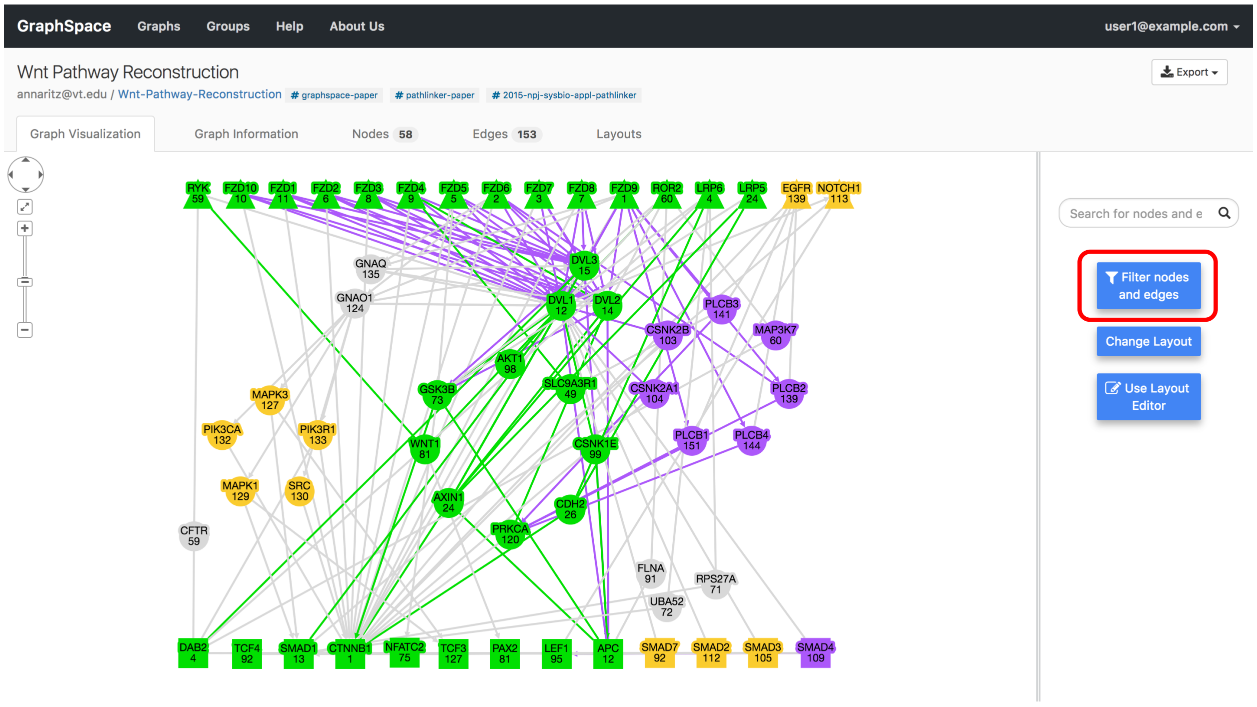
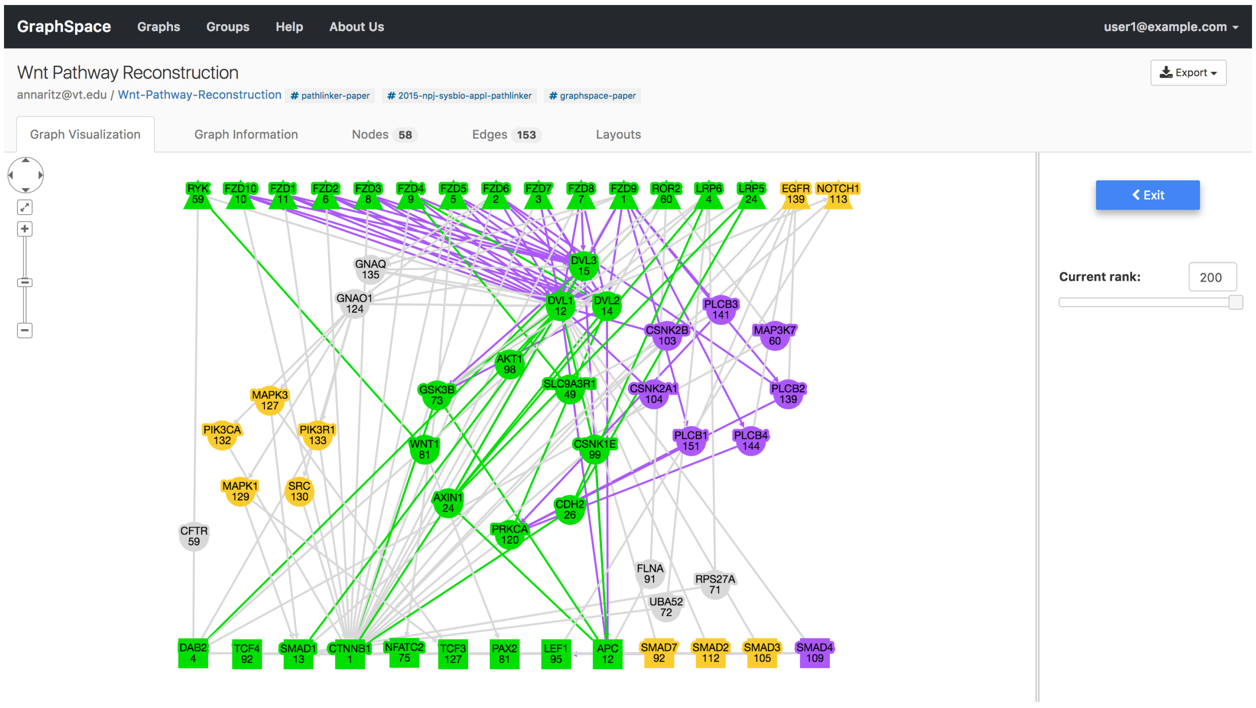


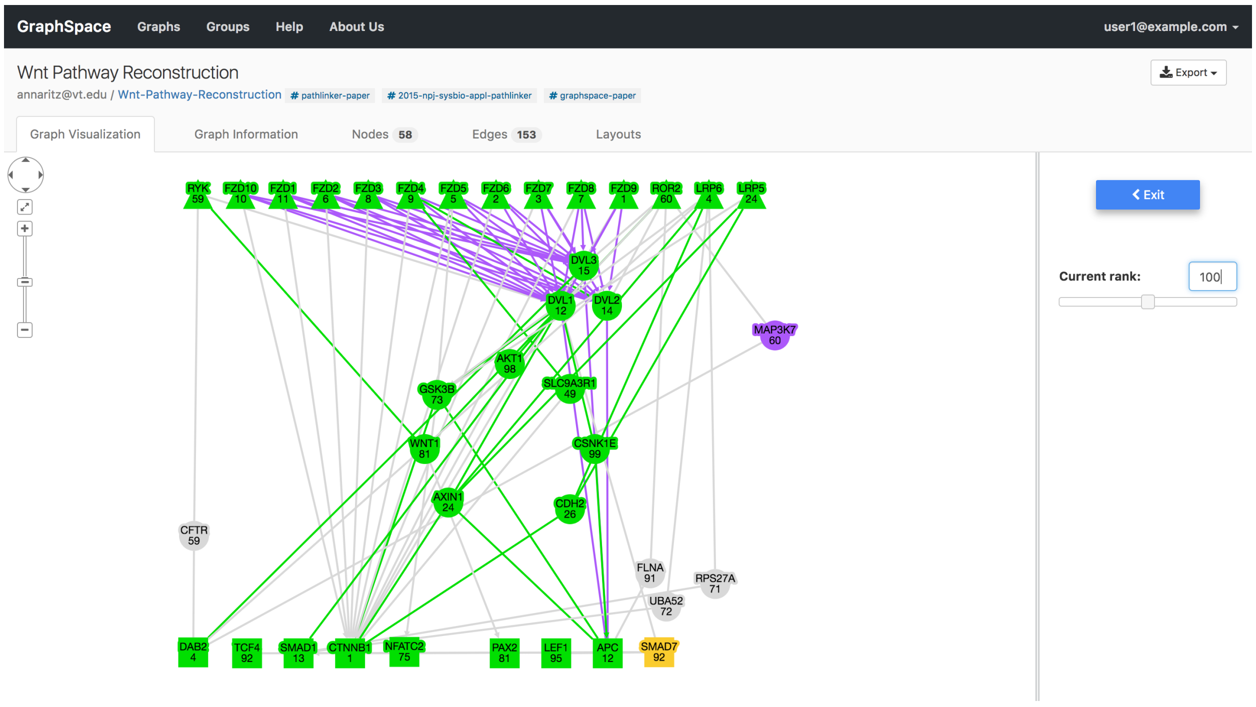
Figure S22: The user clicks on “Filter nodes and edges” button on the right hand side of the “Graph Visualization” tab to display a panel with the “Current rank” slider.



Wnt Pathway Reconstruction

annaritz@vt.edu / Wnt-Pathway-Reconstruction # pathlinker-paper # 2015-npj-sysbio-appl-pathlinker # graphspace-paper

Figure S23: Moving the slider to the left removes all nodes and edges with the “k” attribute larger than the slider value without changing the positions of the remaining nodes and edges.



Wnt Pathway Reconstruction

annaritz@vt.edu / Wnt-Pathway-Reconstruction # pathlinker-paper # 2015-npj-sysbio-appl-pathlinker # graphspace-paper

Figure S24: In this example, only nodes and edges with “k” ≤ 100 are visible.

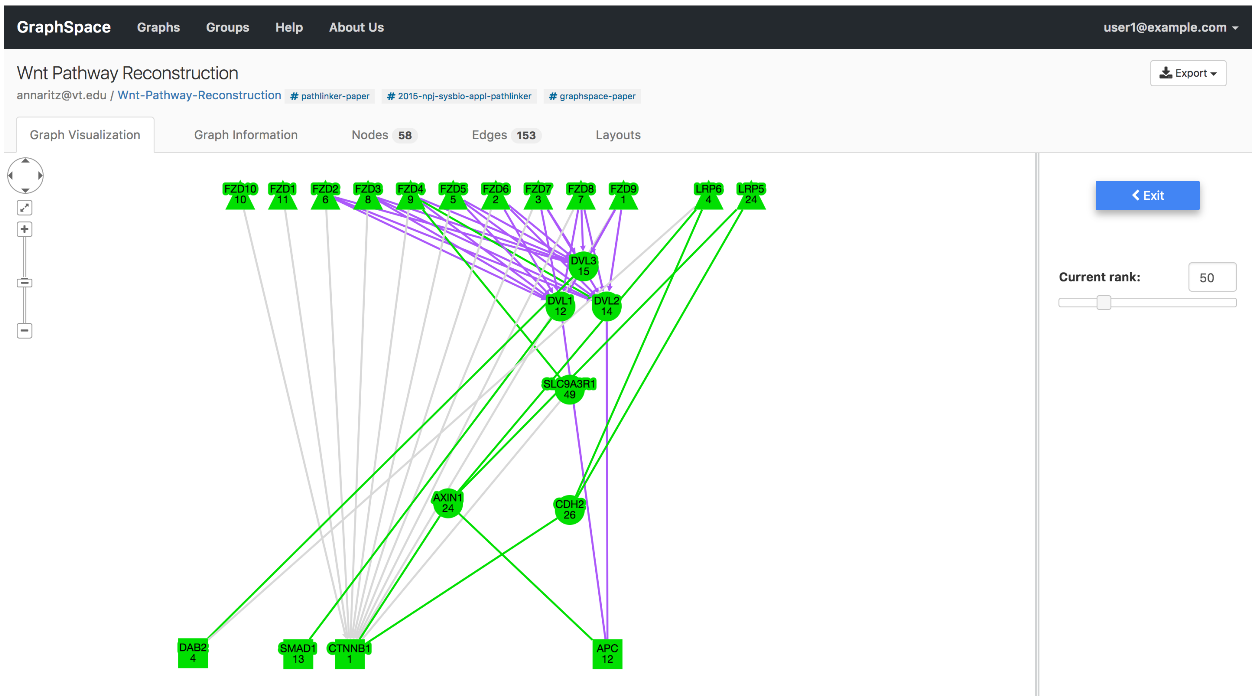


Figure S25: In this example, only nodes and edges with “k” ≤ 50 are visible.

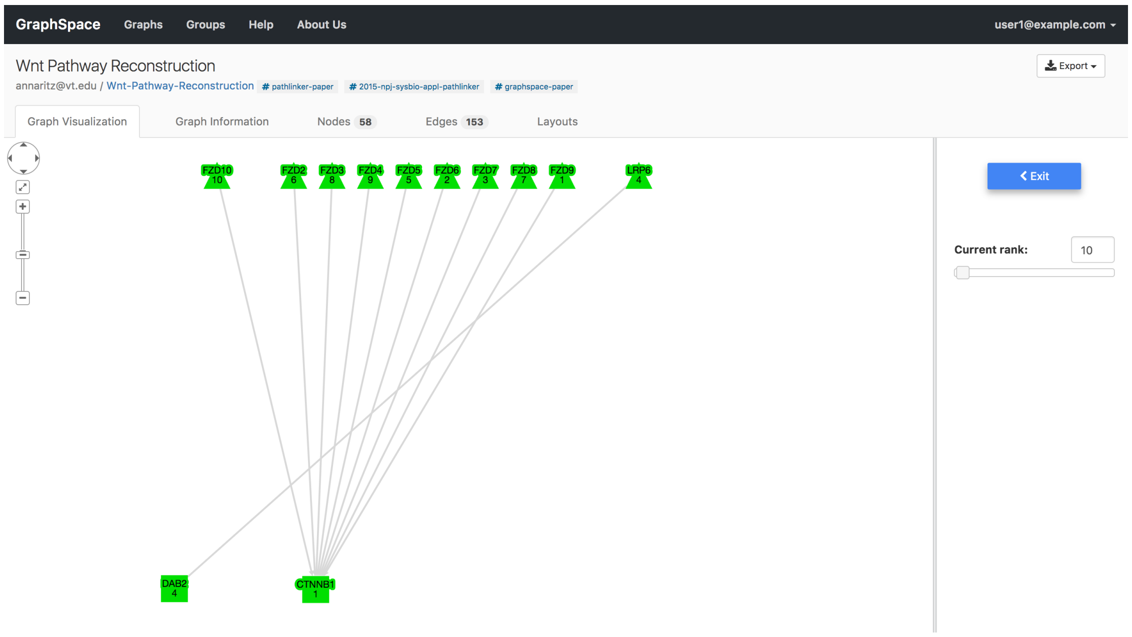


Figure S26: In this example, only nodes and edges with “k” ≤ 10 are visible. When the user clicks on “Back” button, they return to the main menu in the “Graph Visualization” tab.

S4.7 Node and Edge Editor

The layout editor includes features that allows the user to change the appearance of the nodes and edges using an easy-to-use editing interface. In this section, we use a subnetwork of the network in Figure S22 to illustrate the features of this editor. We generated this subnetwork by filtering nodes and edges with "k" > 10 using the procedure explained in Figures S22–S26. The users can access the node and edge editor by visiting the layout editor as explained in Figure S17.

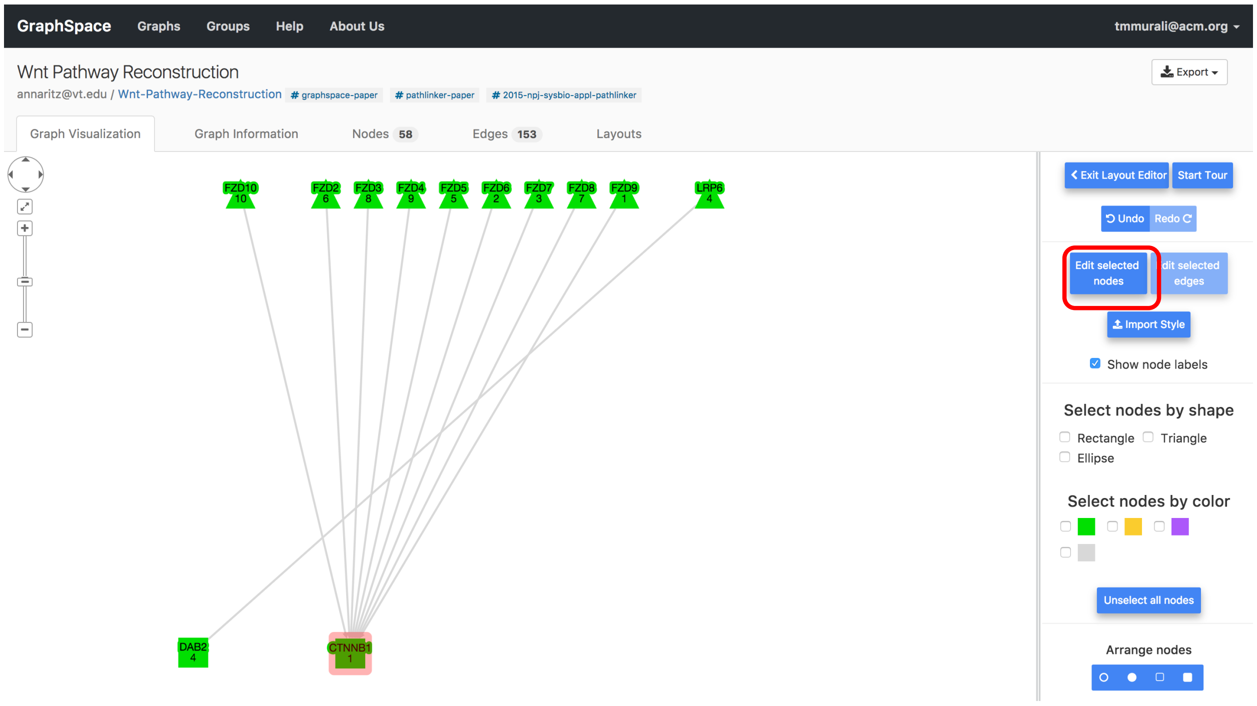
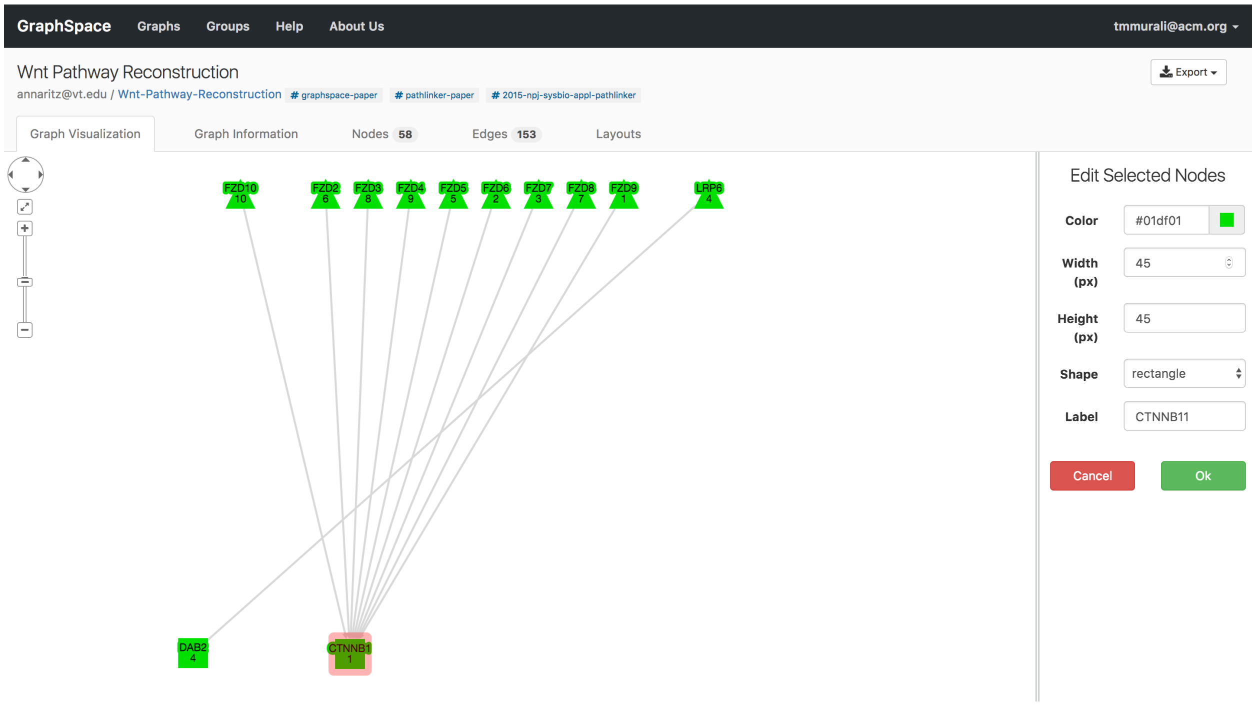


Figure S27: Once the user enters the layout editor, they can select nodes or edges using the gestures supported by Cytoscape.js (e.g., clicking to select one node or edge, using the "Shift" button on the keyboard and the mouse to select multiple nodes) or the tool palette for selecting nodes in the layout editor. In this example, the user selects the node called "CTNNB1" and opens the node editor by clicking on the "Edit selected nodes" button. The user may select more than one node.



Color: #01df01 🟩

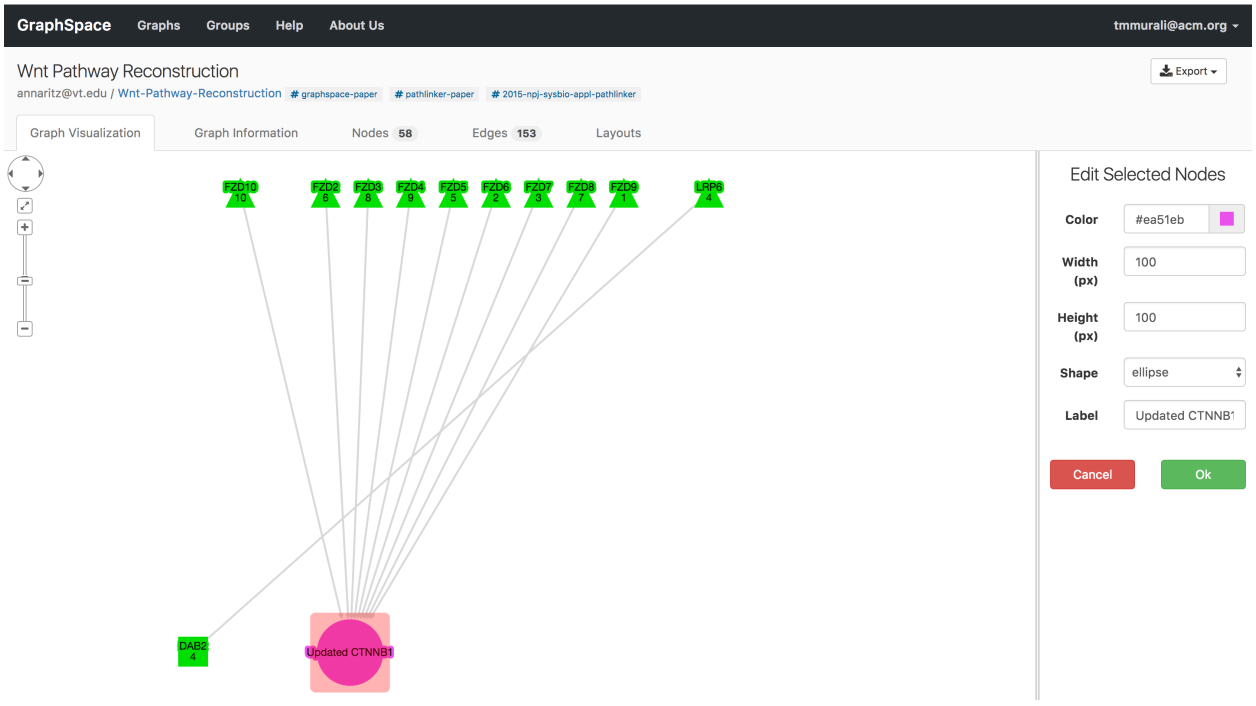
Width (px): 45

Height (px): 45

Shape: rectangle

Label: CTNNB11

Figure S28: The options that appear on the right allow the user to change the following style properties of the selected node(s): i) Color: The background color of the node. ii) Shape: The shape of the node. iii) Width: The width of the node in pixels iv) Height: The height of the node in pixels v) Label: The label inside the node.



Color	#ea51eb
Width (px)	100
Height (px)	100
Shape	ellipse
Label	Updated CTNNB1

Figure S29: The figure displays the layout after the user makes the following changes: (i) color from green to pink, (ii) shape from rectangle to ellipse, (iii) width and height from 45px to 100px, and (iv) label from “CTNNB1” to “Updated CTNNB1”. At this point, the user can apply these changes by clicking on the “OK” button or discard them by clicking on the “Cancel” button. In the remaining figures in this section, we assume the user discards these changes and goes back to the initial layout.

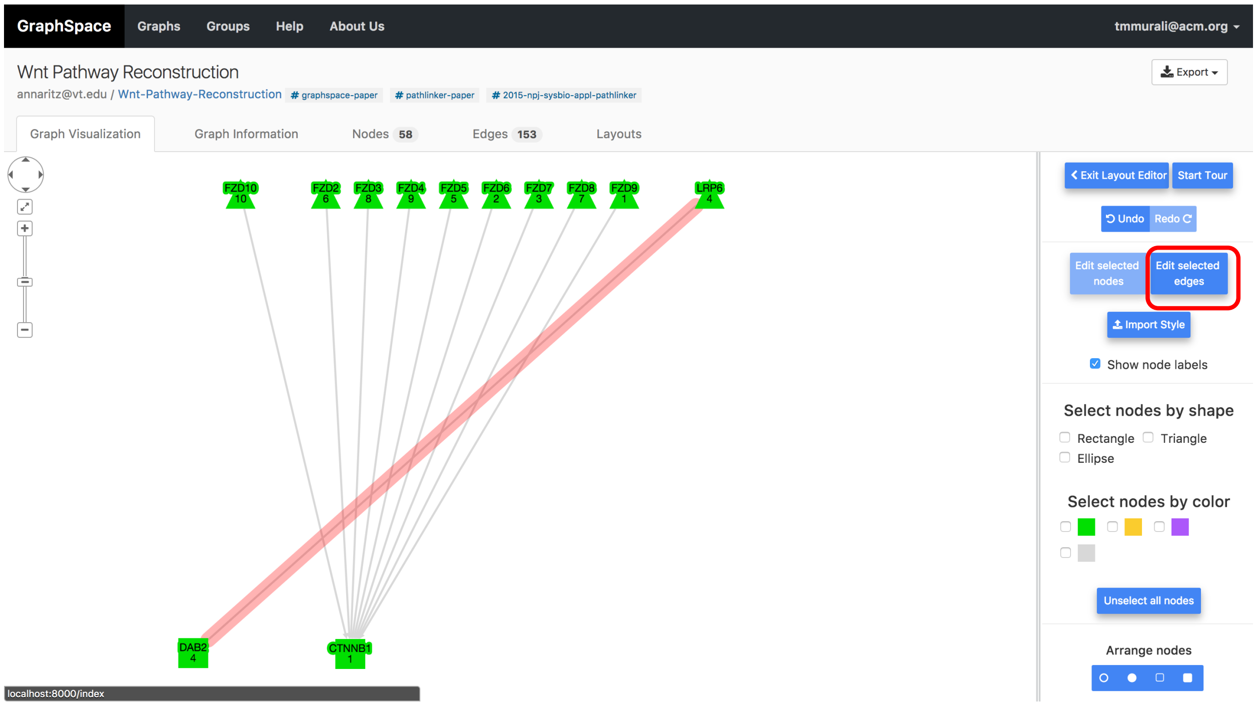
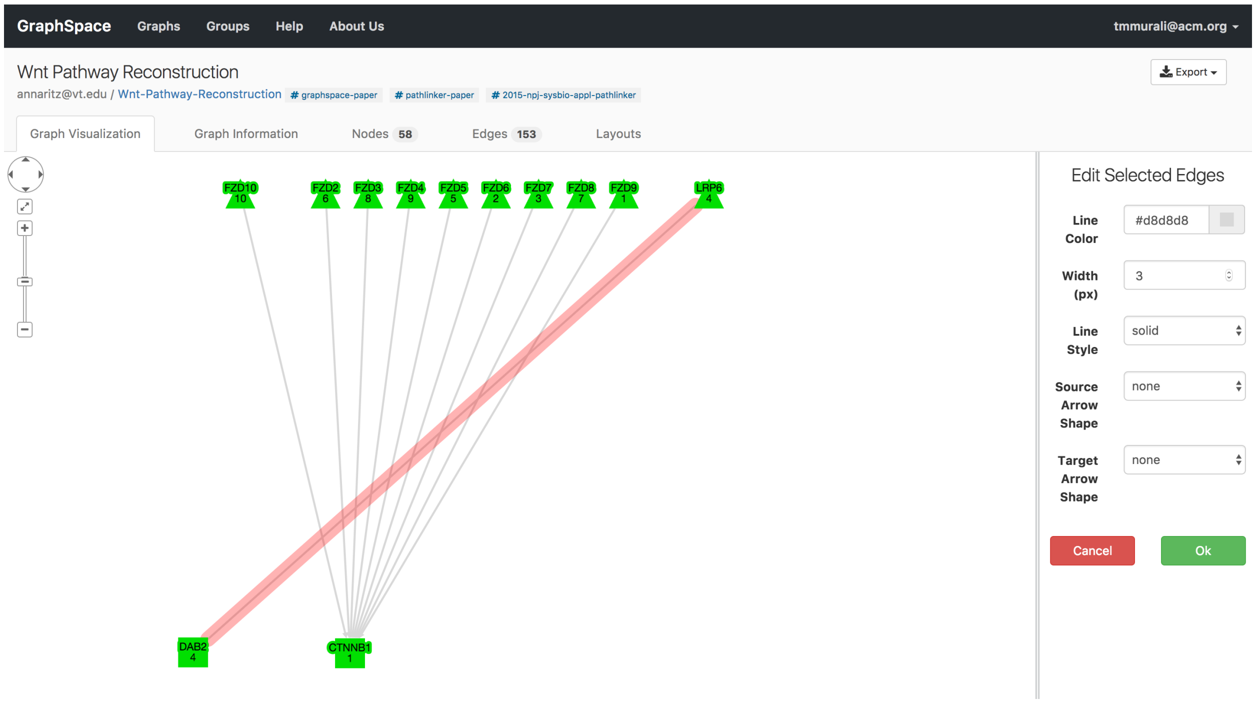


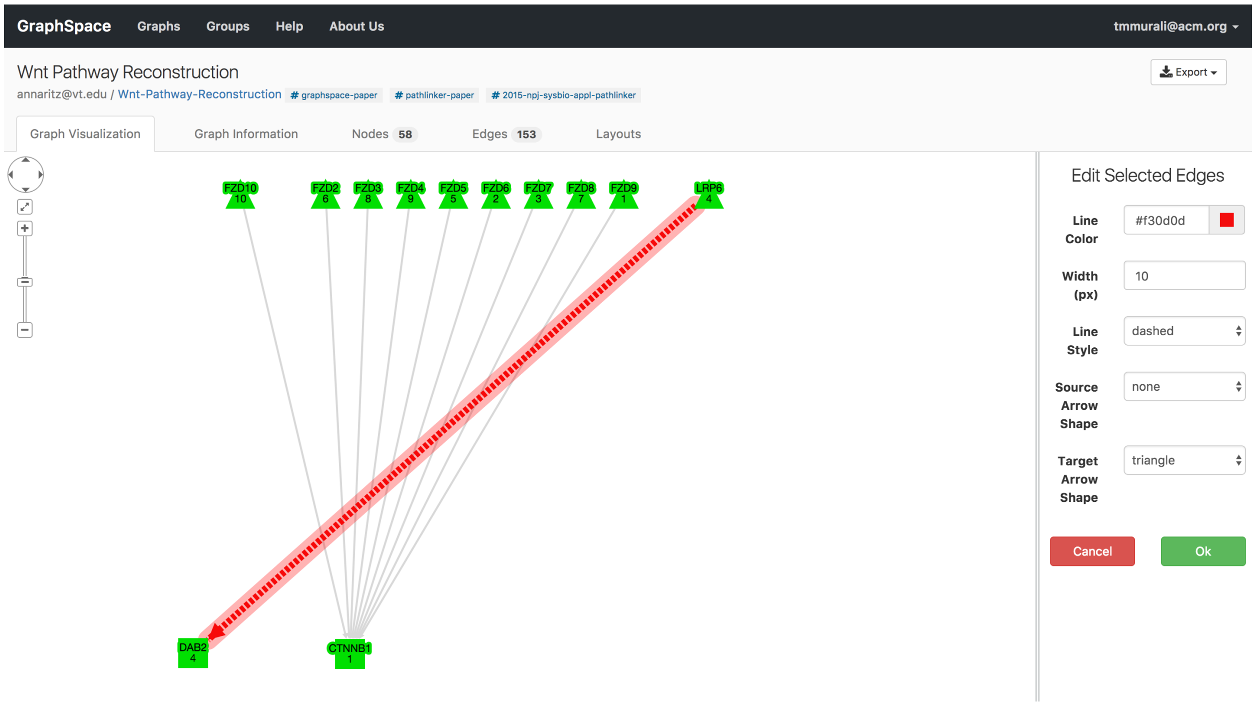
Figure S30: In this example, user selects the edge between "LRP6" and "DAB2" by clicking on it and opens the edge editor by clicking on "Edit selected edges" button. The user may select more than one edg.



Edit Selected Edges

Line Color	#d8d8d8
Width (px)	3
Line Style	solid
Source Arrow Shape	none
Target Arrow Shape	none

Figure S31: The panel on the right allows the user to change the following style properties of the selected edge(s): i) Line Color: the colour of the edge, ii) Line Style: the style of the edge. iii) Width: the width of the edge in pixels, iv) Source Arrow Shape: the shape of the edges source arrow, and v) Target Arrow Shape: the shape of the edges target arrow.



Line Color: #f30d0d 🟥

Width (px): 10

Line Style: dashed

Source Arrow Shape: none

Target Arrow Shape: triangle

Figure S32: The layout after the user makes the following changes: (i) line color from grey to red (ii) line style from solid to dashed (iii) width from 3px to 10px iv) target arrow shape from none to triangle. At this point, the user can apply these changes by clicking on the "OK" button or discard the changes by clicking on the "Cancel" button.

References

[1] Basha, O., Tirman, S., Eluk, A., and Yeger-Lotem, E. (2013). ResponseNet2.0: Revealing signaling and regulatory pathways connecting your proteins and genes–now with human data. *Nucleic Acids Res.*, **41**(Web Server issue), 198–203.

[2] Bastian, M., Heymann, S., and Jacomy, M. (2009). Gephi: an open source software for exploring and manipulating networks. *ICWSM*, **8**, 361–362.

[3] Cromar, G. L., Zhao, A., Yang, A., and Parkinson, J. (2015). Hyperscape: Visualization for complex biological networks. *Bioinformatics*.

[4] De Nooy, W., Mrvar, A., and Batagelj, V. (2011). *Exploratory social network analysis with Pajek*, volume 27. Cambridge University Press.

[5] Franz, M., Lopes, C. T., Huck, G., Dong, Y., Sumer, O., and Bader, G. D. (2015). Cytoscape.js: a graph theory library for visualisation and analysis. *Bioinformatics*, **32**, 309–311.

[6] Gerasch, A., Faber, D., Kuntzer, J., Niermann, P., Kohlbacher, O., Lenhof, H. P., and Kaufmann, M. (2014). BiNA: a visual analytics tool for biological network data. *PLoS ONE*, **9**(2), e87397.

[7] Hu, Z., Chang, Y.-C., Wang, Y., Huang, C.-L., Liu, Y., Tian, F., Granger, B., and DeLisi, C. (2013). VisANT 4.0: Integrative network platform to connect genes, drugs, diseases and therapies. *Nucleic Acids Research*, **41**(W1), W225–W231.

[8] Karr, J. R., Guturu, H., Chen, E. Y., Blair, S. L., Irish, J. M., Kotecha, N., and Covert, M. W. (2015). NetworkPainter: dynamic intracellular pathway animation in Cytobank. *BMC Bioinformatics*, **16**, 172.

[9] Lasher, C. D., Rajagopalan, P., and Murali, T. M. (2013). Summarizing cellular responses as biological process networks. *BMC Syst Biol*, **7**, 68.

[10] Poirel, C. L., Rodrigues, R. R., Chen, K. C., Tyson, J. J., and Murali, T. M. (2013). Top-down network analysis to drive bottom-up modeling of physiological processes. *Journal of Computational Biology*, **20**(5), 409–418.

[11] Pratt, D., Chen, J., Welker, D., Rivas, R., Pillich, R., Rynkov, V., Ono, K., Miello, C., Hicks, L., Szalma, S., Stojmirovic, A., Dobrin, R., Braxenthaler, M., Kuentzer, J., Demchak, B., and Ideker, T. (2015). NDEx, the Network Data Exchange. *Cell Syst*, **1**(4), 302–305.

[12] Ritz, A., Avent, B., and Murali, T. M. (2015). Pathway analysis with signaling hypergraphs. *IEEE Transactions on Computational Biology and Bioinformatics*. In review.

[13] Ritz, A., Poirel, C. L., Kim, H., Tegge, A. N., Powell, A., Simmons, K., Kale, S. D., and Murali, T. M. (2016). Pathways on demand: Automated reconstruction of human signaling networks. *NPJ Systems Biology and Applications*, **2**. Article number 16002.

[14] Salavert, F., Garcia-Alonso, L., Sanchez, R., Alonso, R., Bleda, M., Medina, I., and Dopazo, J. (2016). Web-based network analysis and visualization using CellMaps. *Bioinformatics*, **32**(19), 3041–3043.

[15] Santos, M. C. T., Tegge, A. N., Correa, B. R., Mahesula, S., Kohnke, L. Q., Qiao, M., Ferreira, M. A. R., Kokovay, E., and Penalva, L. O. F. (2015). mir-124, -128, and -137 orchestrate neural differentiation by acting on overlapping gene sets containing a highly connected transcription factor network. *Stem Cells*, **34**(1), 220–232.

[16] Singh, D. P. (2015). *GraphCrowd: Harnessing the Crowd to Lay Out Graphs with Applications to Cellular Signaling Pathways*. Master's thesis, Department of Computer Science, Virginia Tech.

[17] Smoot, M. E., Ono, K., Ruscheinski, J., Wang, P. L., and Ideker, T. (2011). Cytoscape 2.8: new features for data integration and network visualization. *Bioinformatics*, **27**(3), 431–432.

[18] Szklarczyk, D., Franceschini, A., Wyder, S., Forslund, K., Heller, D., Huerta-Cepas, J., Simonovic, M., Roth, A., Santos, A., Tsafou, K. P., Kuhn, M., Bork, P., Jensen, L. J., and von Mering, C. (2015). STRING v10: protein-protein interaction networks, integrated over the tree of life. *Nucleic Acids Res.*, **43**(Database issue), D447–452.

[19] Tegge, A. N., Sharp, N., and Murali, T. M. (2016). XTalk: a path-based approach to identifying crosstalk between signaling pathways. *Bioinformatics*, **32**(2), 242–251.

[20] Tripathi, S., Dehmer, M., and Emmert-Streib, F. (2014). NetBioV: an R package for visualizing large network data in biology and medicine. *Bioinformatics*, **30**(19), 2834–2836.

[21] Wong, A. K., Krishnan, A., Yao, V., Tadych, A., and Troyanskaya, O. G. (2015). IMP 2.0: a multi-species functional genomics portal for integration, visualization and prediction of protein functions and networks. *Nucleic Acids Res.*, **43**(W1), W128–133.