Visualization of Systems Biology Reaction Network

Haoran Yu and ZiXiao Zhang

Introduction

Systems biology researchers store their data in a conventional, machine-friendly format to permit information sharing across different user groups performing different tasks. One format, Systems Biology Markup Language (SBML), is a standard for storing quantitative models to simulate *reaction* between multiple entities also known as *species.* The interaction between species often form a complex network, which is difficult for a human user to interpret using the SBML language alone, therefore multiple visualization tools have been developed in the past, such as Cytoscape, to present a more user-friendly environment to analyze interaction network data in systems biology. However, current tools are mostly targeted to expert researchers with domain knowledge of the data, new users that have limited domain knowledge cannot make good use of these tools. Moreover, even the expert users cannot interpret distributed data across different data models easily. Therefore it is our job to build a tool that can visualize systems biology network data in a way that permits non-expert users to easily find and discover the reaction or species of interest from a complex network, and permits expert users visualize a network that consists of data from multiple sources.

Task and Data Abstraction

Our visualization task is to present network data to the user, and enable an interface to discover the topology of the network as well as paths within the network. The network is a directed graph, consisting of nodes of type *reaction* and *species.* A *reactant species* is a node that has an edge pointing into a *reaction* node, and a *product species* is a node that has an edge from a *reaction* node pointing into it. The *product species* may then act as a *reactant species* for another reaction, thus forming a complex network.

The dataset we will be using is the BioModels Linked Dataset stored in the European Bioinformatics Institute (EBI) database, originally stored in SBML format, but was transformed to Resource Description Framework

(RDF) for easy data storage and retrieval. The SBML data schema consists of 14 tables, the most relevant tables for this project are *SBMLModel*, *Compartment*, *Species*, and *Reaction*. The SBMLModel data consists of 18 attributes, Compartment 5, Species 9, and Reaction 11. Since BioModels is a subset of the data stored in the EBI database, it contains only 636 files stored in RDF. However, each file is an independent data model and forms a stand-alone network. Therefore one of our tasks is to integrate these 636 networks together as a connected network. We note that the RDF version of the data is more table-oriented, whereas the original SBML store network data without spacial coordinates. It is unclear how many nodes and edges, or how many table items are in the dataset; this number can only be figured out once we parse the RDF files. The tool that we build will permit users to locate and browse reaction of interest from the integrated network presented in a 2-dimensional space.

Moreover, it is also possible for our tool to present an interface to permit users to annotate the nodes and edges in the network, however this is left as an extra feature in the end.

Visual Encoding

Due to the complexity of the SBML schema, we cannot encode every single attribute as visual channels in our tool, we aim to select a subset of the attributes to visualize that are the most relevant to the task abstraction. We encode the *SpeciesReference* attribute of the Reaction table as the edges in the network, the SpeciesReference links between a Species and a Reaction. Each distinct Species item in the Species table is encoded as a node in the network, and each distinct Reaction item in the Reaction table is similarly encoded as a single node. It is possible for a Species or Reaction to appear multiple times across the 636 models, and this allows the different independent networks to link together.

In addition to the network, we permit navigation of the network. The different parts of the network can be highlighted by seletion, zoomed in by user interaction, and can be viewed in a separate window with more detailed information about the selected item. For example, the user can select a node of type Species, and request for all attribute information stored in the Species table. The user will be directed to a separate window, which displays the original RDF data.

We use the color channel to encode the origin of the data, i.e. which of the 636 data models the item comes from. It is impossible to use 636 distinct colors, therefore we only allow coloring as items are highlighted. For example, when the user selects a Reaction node, all Species nodes that are connected to the Reaction are also highlighted. Each Species is colored using categorical coloring scheme, according the data model the Species is from.

We permit filtering of the network. A facet of the multi-window view will be allocated for dynamic generation of slide bars that encode the *Stoichiometry* attribute of the Reaction table. As the network is navigated and zoomed in, the slide bars update dynamically to reflect the distinct species present in the visible part of the network, each slide bar permits value selection from the minimum value to the maximum value of each Species. As the values are changed, the network updates dynamically, displaying only the reaction paths that are possible with the specified quantity of each Species. This function is especially useful for scientists performing lab experiments. The scientist may only have a limited quantity of a specific Species available, and would want to discover what reaction paths are possible using the given quantity. The scientist may select a start and end node to visualize all possible reaction paths.

Finally, we also permit querying the data item in the network. As results are returned, we permit linked highlighting in the network window. If time permits, we will encode more attributes as channels in the visualization tool.

Implementation

We will be using D3.js as our software to develop the tool. The main challenge of this project is to effectively display the network in a 2-dimensional space. We plan to use the 4-stage implementation as presented in GraphViz. The GraphViz technique is sufficient to achieve our visualization tasks in our design.

Timeline

Haoran

(1) 5 hours - extract and parse RDF, design data structure that can be encoded as a network

(2) 30 hours - encode attributes as visual channels

(3) 30 hours - implement user interactive interfaces

(4) 5 hours - fix errors and ensure compatibility

ZiXiao

(5) 5 hour - prototype D3 interface for network data

(6) 30 hours - implement algorithm to draw network data using GraphViz technique

(7) 30 hours - add support or data querying, network filtering and path generation

(8) 5 hours - fix errors and ensure compatibility