Exploratory Visualizations of Rules for Validation of Expert Decisions

Protiva Rahman, Jian Chen, Courtney Hebert, Preeti Pancholi, Mark Lustberg, Kurt Stevenson and Arnab Nandi

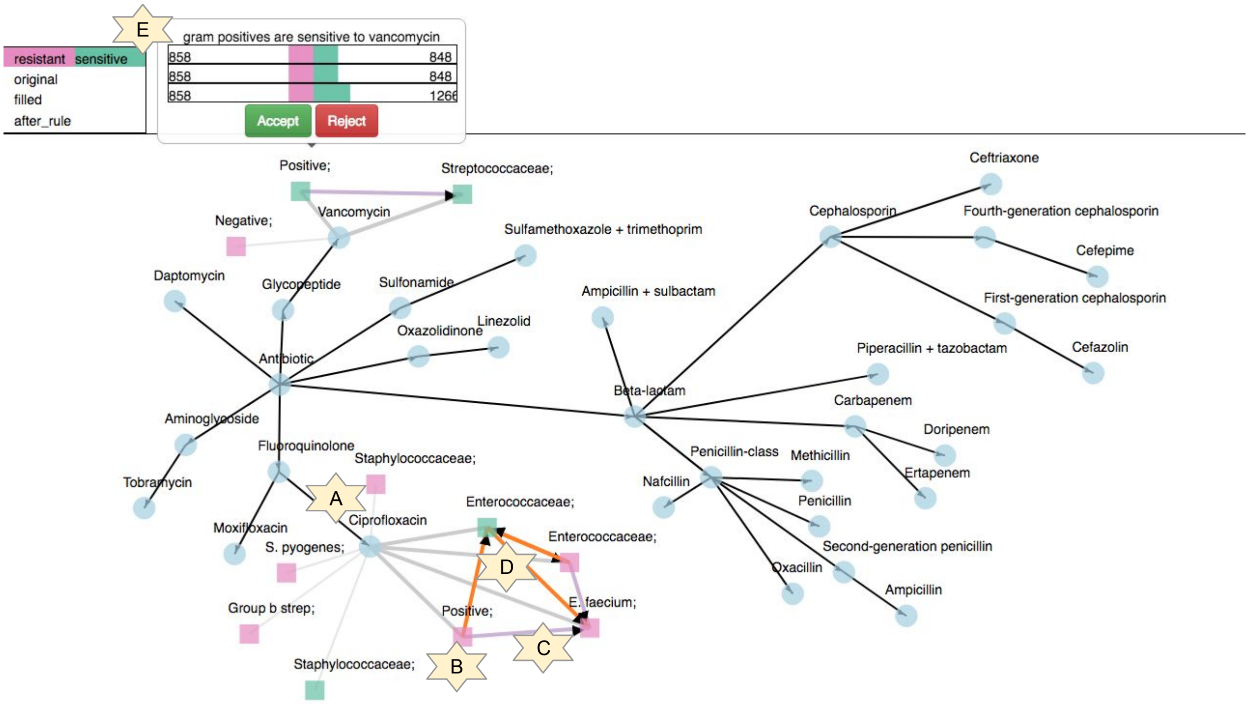


Fig. 1: Visualization of rules for filling in unreported microbiology data: (A) The blue circle nodes serve as navigation and show entities to be filled, in this case antibiotics. Each entity node can be expanded to show relevant rule nodes, which are represented as square nodes. (B) Rule nodes are labeled with predicates that the rule is conditioned on, while the node color encodes the value it fills in. Rule relationships are depicted through colored edges: (C) rules with result overlap are shown in purple and conflicting rules are shown in orange. (E) Each rule node also contains a data summary pop-up that shows how the dataset will be changed upon applying the rule. Users can interactively accept/reject rules to resolve conflicts and redundancies.

Abstract—Rule-based techniques are common in domains that require explainability and interpretability, such as the health-sciences. In certain cases, rules from multiple systems or humans have to be consolidated for consensus. When there is no objective function to measure rule validity and the rule generation process can contain errors, human involvement is required for rule consolidation. We present an interactive visualization system which allows users to explore relationships between rules, such as conflicts and redundancies, and see how applying the rule would affect the data. Our design allows users to interactively resolve conflicts by providing feedback on the correctness of each rule. Results of interacting with a rule are immediately applied to the data and redundant rules automatically removed.

Index Terms—Clinical Informatics, microbiology, rule-based, visual database exploration, network visualization

1 Introduction

Recently, there has been an increased demand for transparency and explainability [48, 58] in data-driven algorithms. In some cases, this has led to a preference for rule-based techniques over machine learning techniques, even at the cost of accuracy [45]. Further, many machine learning algorithms require human-in-the-loop intervention [21, 56]. Each of these interventions could be represented as rules and compared across users. Similarly, noisy data quality could lead to different heuristic optimizers or ensemble classifiers producing contradictory results. Other scenarios where conflict resolution between rules is required include consolidating policies across different government agencies, or when results and guidelines of different medical tests suggest different treatment plans. Moreover, in certain cases, due to the nature and domain of the task, machine learning is not an option at all and expert input in the form of rules is required, e.g., prior to the stage when hypotheses are formed [42]. All of these cases would need a human to resolve conflicts and redundancies between rules.

- *P. Rahman, J. Chen, and A. Nandi are with the Department of Computer Science and Engineering, The Ohio State University. Email: {rahman.92, chen.8028. nandi.9}@osu.edu.*
- *C. Hebert, P. Pancholi, M. Lustberg, and K. Stevenson are with The Ohio State University College of Medicine. E-mail: {courtney.hebert, preeti.pancholi, mark.lustberg, kurt.stevenson}@osumc.edu.*

DSIA: Data Systems for Interactive Analysis
21 October 2018, Berlin, Germany

As a concrete example, consider the case of unreported microbiology lab data, where multiple experts are required to arrive at consensus on a set of rules for filling in the missing values [40–42]. Automated methods cannot be used to impute the data due to the nature of the missing data. Thus, each expert individually formulates a set of rules while interacting with the data, which then have to be consolidated by removing conflicts and overlaps. While the number of rules to be consolidated will vary based on domain and number of missing values, in the case of filling in a microbiology dataset with 75,000 cells [42], we found that more than three hundred rules were written by three domain experts and among these rules only ten were the same across the three domain experts. Further, there is an inherent subjectivity to the process and experts can make errors due to the complexity of the task. Rules thus cannot be automatically resolved without expert involvement since it is unclear which rule is correct. Resolving them with heuristics such as majority vote is not an option either, since it is possible that multiple experts missed an outlying case that was considered by a fewer number of experts.

Hence, consensus from multiple experts is required to reduce bias and errors. Having the experts go through a list of conflicting rules is inefficient and cumbersome, because rules are conditioned on different variables and can have partially conflicting result sets. Two conflicting rules can both be accepted by the experts if their result sets are mostly disjoint, but a decision must be made on which rule applies to the overlapping result set, and that rule would have to be applied before the other. Thus, rules need to be seen in context of other rules, along with specific information on affected tuples. To address these, we build an interactive rule visualizer that allows users to see rule relationships and apply them in the correct order. While we describe our system in the context of a microbiology dataset, it is applicable to other cases where rules from multiple users, systems, models, etc. need to be consolidated with a human-in-the-loop.

Our interactive rule visualizer makes the following contributions:

1. Analysis and abstraction of problems in consolidating rules from multiple sources in the microbiology domain.

2. A visual encoding of rules and their related entities in collaboration with microbiology experts.

3. A new design that allows for quick identification of rule relationships such as conflicts and redundancies, and their impact on the data *in-place* to avoid loss of context.

4. Our system allows users to interactively make decisions on rules, which are immediately applied to the dataset. Redundant rules are automatically removed, reducing the burden on the user.

2 Related Work

Our work relates to prior systems in visualizing biological data, association rule mining, graph and set relations.

Visualizing Biological Data: There are multiple systems that have used visualizations to address specific problems in biological sciences. The closest to our work in terms of data domain is Garcia-Caballero et al. [20]'s visualization of antibiograms for clinical decision support. Antibiograms contain the percentage of a particular organism species that is *resistant* to an antibiotic, and are used for empiric antibiotic prescription. They compare three visualizations: sunburst model, bipartite graphs, and a tree model for the task of finding effective antibiotics. Even though they deal with the same data domain, they are addressing a different problem: they want to find the most effective antibiotic based on known resistance as opposed to validating resistances based on rules. Their methods are not applicable to visualizing rules, where we need to show relationships between rules as well as group them by antibiotics.

Other rule-based visualizers include Smith et al.'s RuleBender [46], which allows users to visualize the effects of rules on molecules for rule-based modeling (RBM) use cases where models are used to simulate cell signaling networks. Visualizing gene-interactions is addressed in Wu et al.'s EinVis [55] system, which represents them in a tree ring diagram and adjacency matrix simultaneously, using multi-view VisBubbles [32]. While all existing systems take advantage of different views to represent rules and their influence on each other, we use embedded views so that useful information can be accessed in close proximity in the same view.

Biological pathways is another common domain that requires relationship visualization [26, 31, 60]. Zhu et al.'s PathRings [60] uses linked sunburst visualization for pathways. Alternatively, Kashofer et al.'s enRoute [26] uses a node-link diagram to highlight path relationships and show it in the context of associated experimental data. Building on top of enRoute, Lex et al.'s Entourage system [31] partitions pathways into small sub paths, allowing the user to focus on one path. To enable better querying on paths, i.e. find paths connecting nodes A and B, Partl et al.'s Pathfinder system [37] allows users to visualize biological networks as node-link diagrams as well as ranked lists based on query criteria. Other biological querying systems include Partl et al.'s ConTour [38] system, which supports brushing and linking across multiple drug discovery datasets and Wu et al.'s path querying algebra [54], which allows users to query pathways through constructed examples. While these methods are related to ours in that they display and query interactive graphical information, they do not address conflict resolution between rules.

Visualizing Association Rules: Another area of research has focused on visualizing association rules. Blanchard et al.'s ARVis system [9] allows the user to focus on one rule and see other relevant rules which they refer to as neighborhood relations. Pilipczuk and Cariowa's NovoSpark system [39] uses cloud images to see how features affect a rule, and then use spectrum images to compare feature sets in rules based on the feature's importance. Treemaps [29], radial layouts [18, 28] and pyramids [27] have also been proposed for visualizing frequent item sets. However, most of these works address the goal of user validation of mined association rules purely based on the data, as opposed to visualizing rules for updating values. Hence, they do not allow the user to explore impacts of the rules or see relationships between them.

Graph Visualization: There has been plenty of work on graph visualizations [7, 23], with applications in dataflow [13, 49], distributed computing [14, 35, 36], user-interfaces [16], and recently, deep learning [53]. Adapting from these, we use a node-link diagram since the microbiology domain has inherent hierarchical classification, to which we want to add rule relationships. While treemaps [12] allow for more compact representation, it is difficult to represent additional attributes other than the hierarchy on it. Zhao et al.'s Elastic Hierarchies [59] combine treemaps with node-link diagrams, however this requires both dimensions to be hierarchichal and it is unclear how conflicts would be represented. Wongsuphasat et al.'s TensorFlow visualization [53] deals with challenges similar to ours in that nodes have high degrees, with edges crossing each other. Their approach to this involves bundling edges through nodes and removing auxiliary nodes. Since edge bundling leads to loss of comprehension in our case, we had to reduce the number of nodes initially presented to the user, allowing them to interactively explore more relations. We use a collapsible force-directed layout [19], a common technique for making graphs more readable [1, 4, 5, 51], where all node positions are fixed on initial load to allow users to maintain their mental models [34].

Recent work in graph visualizations includes Srinivasan et al.'s Graphiti system [47], which allows users to interactively create node-link diagrams by providing examples. This is an orthogonal problem to ours, however their ideas can be used to extend our work, to allow the expert to create the visualization that they want.

Visualizing Sets: Since rules can be considered as sets based on the entities they update and the entities they are predicated on, ideas from set visualization are relevant as well. Lex et al.'s UpSet system [30] is a visualization system for set data that allows the user to quickly see aggregates of intersections. This technique would allow us to see rules with overlapping resultsets, which is equivalent to listing conflicting rules, without providing context on the whole set of rules or the data.

The state of the art in set visualizations is summarized by Alsallakh et al. [3], which include Euler diagrams, overlays [15], node-link diagrams, matrices, aggregation, and scatter plot techniques. Rodgers et al. [43] propose using linear diagrams for visualizing set intersection, since these are easier for humans to identify and compare. Collins et al.'s BubbleSets [15] use isocontours based on energy of items imposed on pixels to show multiple set memberships, while Alper et al.'s LineSets [2] use colored edges instead of contours. Dinkla et al.'s Kelp diagrams [17] augment these by edge routing and finally, Xu et al. further improve on them by using glyphs to encode overlap [57]. These methods are not effective in cases where items belong to multiple sets which are not spatially close to each other. Further, while they might work for rule containment, they do not work for showing conflicts.

3 Domain Background

Our design addresses challenging issues in microbiology. Several microbiology concepts or attributes are related to edges in our rule-enabled network visualizations. For each culture (e.g., urine specimen), collected from a patient sent to the microbiology laboratory for testing, the laboratory reports the organism that is causing the infection and its sensitivity to a subset of antibiotics. If an antibiotic works against an organism, the organism is said to be *sensitive* to the antibiotic, otherwise it is *resistant* to it. Thus, for each culture, the antibiotics that are tested are reported as *sensitive* or *resistant*.

The subset of antibiotics which are tested depend on characteristics of the domain and institutional preference. When using the data from these reports for statistical analysis, sensitivities for a more comprehensive list of antibiotics, including the unreported ones, is desired. Domain experts such as infectious disease physicians, microbiologists and pharmacists are then needed to fill in these values through rules. Experts use their domain knowledge and the reported sensitivities to create rules that fill in the unreported values as *sensitive* or *resistant*. Rules can be expressed as standard SQL `update` queries and presented to the user as statements. An example user rule would be: *E. coli is resistant to Vancomycin*, which corresponds to the following `update` query:

`UPDATE SET` *Vancomycin* = *resistant* WHERE *organism* = *E. coli*;

Thus, each rule specifies a set of organisms that it applies to and updates a set of antibiotics as *sensitive* or *resistant*. The organism set can contain a single organism or be grouped by genus, species, family or gram stain [42]. Similarly, the antibiotic set can contain a single antibiotic or be grouped by antibiotic classes [42]. Along with being conditioned on a set of organisms, a rule can also be conditioned on its results to another antibiotic. For example, the rule, *If Staphylococcus is resistant to Cefazolin then it is resistant to Cefepime*, would correspond to the following `update` query:

`UPDATE SET` *Cefazolin* = *resistant* WHERE *organism* = *Staphylococcus* `AND` *Cefepime* = *resistant*;

Thus, rules have at most two `where` clause predicates.

Since experts formulate rules individually, and there are multiple ways to fill in the same values, they might select different sets of rules but fill in the same values, giving a consensus dataset. In order to make expert decisions reproducible, it is not enough to have a consensus dataset; a set of consensus rules is also needed. Further, this allows for privacy-preserving knowledge sharing, since rules can be shared among institutions without sharing data. An automatic approach to consolidate rules would be to extract rules from the consensus dataset using decision trees, however, this is ineffective and still requires significant expert intervention [40].

Thus, the set of rules specified by each expert has to be consolidated by removing conflicts and redundancies. In this case, there are over three hundred rules, which can be grouped by the antibiotics and organisms it affects, but it is taxing for experts to go through this entire list without being able to see relationships between rules and their impact on the data. As an example of conflicting rules, consider the following:

- *If an organism is resistant to Cefazolin then it is also resistant to Ampicillin+sulbactam*
- *If an organism is sensitive to Penicillin then it is also sensitive to Ampicillin+sulbactam*

There could be a possible conflict between these two rules if there are cultures which are *resistant* to *Cefazolin* but *sensitive* to *Penicillin*, since the first rule would fill it in as *resistant* while the second one would fill it in as *sensitive*. Hence, one needs to assign an ordering to the rules or one of the conflicting rules needs to be discarded. Further, when deciding on a rule, experts want to see its impact on the dataset. For example, if there is more evidence in the reported data for the second rule, i.e., most of the organisms that are *sensitive* to *Penicillin* are mostly *sensitive* to *Ampicillin+sulbactam*, while there is an even distribution for *Ampicillin+sulbactam* for those that are *resistant* to *Cefazolin*, then they might be more likely to apply the *Penicillin* rule first.

3.1 Data and Task Abstraction

Our dataset consists of a set of entities and rules. In most domains, rules consist of the following three components: (1) a set of entities it updates, which we refer to as result set entities, (2) the value it is being updated with, and (3) a set of entities it is conditioned on, which we refer to as predicate entities. Two rules can have overlap in their predicate entities, in their result set entities and between the predicate entities of one rule and result set entities of the other. In the first two cases, the two rules can be conflicting or redundant, while the third case shows a cascading dependency. While talking to a domain expert on their experience in consolidating rules and observing four experts (Hebert, Pancholi, Lustberg, Stevenson who are co-authors on the team) at an hour long consensus meeting, we noticed that the last case was not considered when selecting rules. Hence, we identified the following tasks that needed to be addressed by our system:

1. Identify and resolve *complete* conflicts between rules by accepting correct rules.
2. Identify and resolve *partial* conflicts and redundancies by assigning an order to rules.
3. Compare updates made by rules against data that is available, i.e., has not been updated by rules.

3.2 Workflow

Before describing our design details, let us walk through an example usage scenario of our tool. Lucy is a microbiologist who wants to analyze newly collected lab results. She knows that Helen, Hannah and Harold have individually formulated three sets of rules to fill in the dataset and she needs to combine them to have a concise, correct and conflict-free set of rules. She knows that using traditional approaches would require her to compare the rule structure and result sets and go through a list of conflicting rules (around hundred conflicting pairs for our dataset). Further, if she wanted specific information on how the rules affect the dataset, she would have to go back and forth between the rules and the data.

Instead, Lucy decides to try our interactive rule visualizer (Figure 1). Upon navigating to the web application, Lucy first sees an overview (Figure 7), showing the hierarchy of antibiotics in her dataset, a view which she is familiar with. She decides to first tackle all rules that fill in *Ciprofloxacin*, and thus double clicks on the *Ciprofloxacin* (Figure 1A) node which shows all the rules that affect it (Figure 4). She then explores the rule that fills in *Ciprofloxacin* for the gram *positive* group of organisms. She notices the pink *positive* node (Figure 1B), which would fill in *resistant* for *gram positives* and *Ciprofloxacin*. Double clicking this node shows the rule's relationship to the other *Ciprofloxacin* rules through colored edges (Figure 10). Lucy can immediately see that it would subsume two other rules that fill in *resistant* for *Enterococcaceae* and *E. faecium*, which are subsets of *gram positive* organisms (purple edges shown in Figure 1C). Lucy also sees that it conflicts with a rule that fills in *resistant* for *Enterococcaceae* (orange edges shown in Figure 1D).

To make her decision, Lucy would like to know what the distribution of *resistant* and *sensitive* is for *Enterococcaceae* and *Ciprofloxacin* in the reported lab data. On clicking the two conflicting *Enterococcaceae* nodes, their data summary modals appear (Figure 9), containing bar charts showing the data distribution. The top row shows the distribution in the reported data from the lab, the middle row shows the distribution based on values filled in by other rules, and the bottom row shows how the distribution would change on applying the current rule. Lucy sees that there are slightly more *Enterococcus* in the data that are *resistant* than *sensitive* and hence chooses to apply the rule that fills in *resistant* for *Enterococcaceae*, by clicking on the *accept* button on the modal (Figure 9). This automatically removes the *Enterococcaceae* and *E. faecium* nodes since they will no longer fill in any values. She likes this feature since this removes the burden of having to manually reject them. The gram *positive* node still remains since there are *gram positives* other than *Enterococcus* that still have missing values for *Ciprofloxacin*. Lucy is confident about her decisions on rules since she can easily interpret the decisions of her colleagues and immediately apply them to this newly collected dataset.

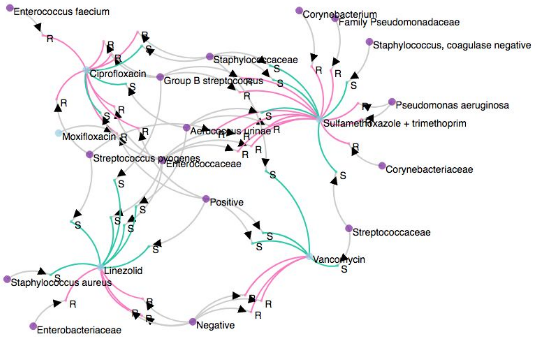


Fig. 2: Rule Representation A: Nodes for each rule (pink and green nodes), antibiotic (blue nodes) and organism (purple nodes). Grey edges link predicates to rules, while colored edges link rule to the result set entity.

4 Methods

In this section, we describe our visualization design developed collaboratively with microbiology experts and our algorithm for rule edits.

4.1 Rule Representation

First, we discuss our design decisions for representing rules, since this is the main and most atomic component of our system. Since the domain data, i.e., organism and antibiotics, has a semantic hierarchical structure with which we want to associate rules, we use a node-link diagram to represent them, in keeping with the experts' mental model of the data. Specifically, we use a directed force layout diagram [19], which tries to optimize the layout by distributing vertices evenly and maintaining uniform edge lengths by modeling nodes as charged particles and edges as springs.

As mentioned in Section 3.1, there are three components to any rule that need to be represented: the result set entity, the value it is being updated with and the predicate entities. In our case, entities consist of different antibiotics and organisms. These can be represented in multiple ways. One possibility would be to create nodes for all organisms and antibiotics, as well as nodes for rules. Each rule node would then have a directed edge from it to the updated antibiotic, with the color representing if the updated value was *resistant* (pink) or *sensitive* (green). The rule node would also have incoming grey edges from its predicates. In our application there are at most two predicates (an organism and/or an antibiotic), but in a general case there might be more. This representation is shown for forty-nine rules of just four antibiotics in Figure 2. This view shows all the entities in the dataset and any rules that are related will be connected through a path. Thus, this visualization preserves and displays all the information on rules. However, this view is already very complex, without explicitly linking rule relationships through edges. Further, the user has to follow up to three edges to interpret the complete rules and follow paths to see rule relationships. Moreover, as the number of rules and predicates increases, the degree of each entity node will increase. For example, with eighty organisms and twenty antibiotics, we have a total of hundred entity nodes. If there are three hundred rule nodes, where each rule node is connected to at least two entities (predicate and result entity), then each entity node has a degree of six, assuming a uniform distribution of rules among entities. Since there are groups along two entity sets (organisms and antibiotics), there are multiple edge crossings and it is difficult to get insights.

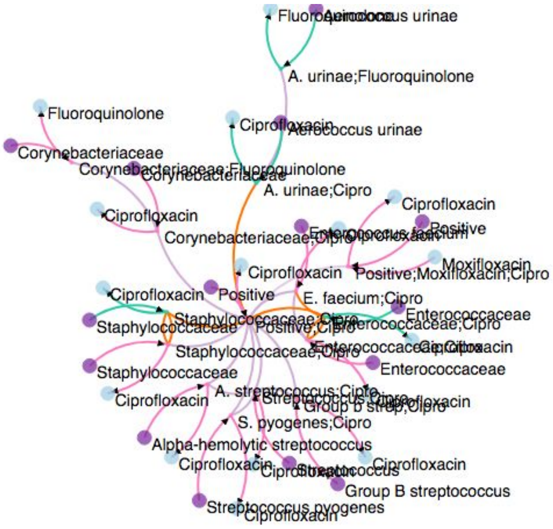


Fig. 3: Rule Representation B: Rule nodes have individual predicate and result set nodes, with purple and orange edges showing conflicts and redundancies between rules.

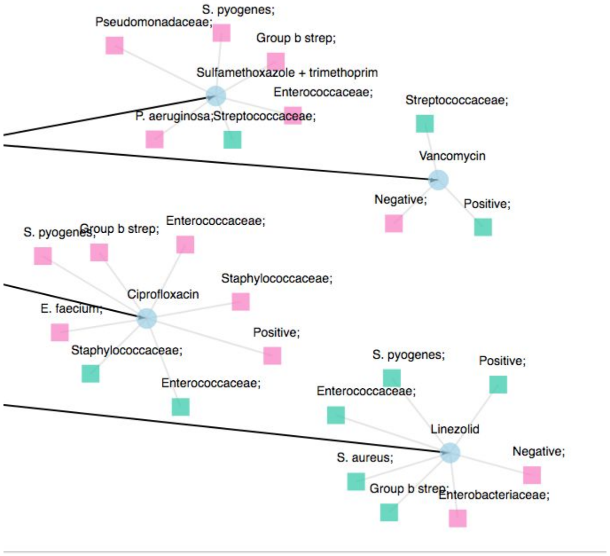


Fig. 4: Rule Representation C: Rules represented as square nodes, with predicates on label, color denoting the value, and connected via edges to the entity being filled in.

To reduce the degree and edge crossings, another possible representation would include each rule node having its own predicate and result nodes. For example, two rules that update the antibiotic *Ciprofloxacin* would have an edge to two individual *Ciprofloxacin* nodes, while in the prior representation, they were linked to the same node. The direction of edges is the same as the prior representation, i.e., from predicates to rule node and from rule node to result entity, with edge color representing rule value. While this increases the number of nodes, it significantly reduces the number of edge crossings, since rules have individual entity nodes. But with this simplification, we lose information on related rules and need explicit edges for rule relationships. Thus, conflicting rules are shown with an orange edge between them and rules that subsume one another have a purple edge between them. This representation is shown in Figure 3. While it improves on the prior representation, in that insights can now be gained on related rules through clusters in the graph, interpreting the rule still requires looking at three nodes, which has high cognitive load.

To further simplify this, we decided to include the predicates in label of the rule node, but have a separate node for the result entity. Hence, all the result entities, i.e., antibiotics are represented as nodes, similar to the first representation, but the predicates, i.e., organisms (and any additional antibiotic) serve as the label of rule nodes. This is cleaner than the first representation since rules are grouped only on the result entity as opposed to result and predicate, reducing edge crossings. This representation is also easier to interpret since only two nodes are required for a rule, and if users are looking at a group affecting the same result set, simply looking at the rule node is enough. Since we have two node types, result entity and rule, we differentiate them with shape: circle for entity nodes, and square for rule nodes, which is one of Bertin's [8] retinal variables for encoding nominal data. We use color, a separate visual channel [8], to represent the value being filled in by a rule: pink square nodes denote rules that fill in *resistant* while green square nodes denote rules that fill in *sensitive*. This is the representation that was selected, based on expert feedback.

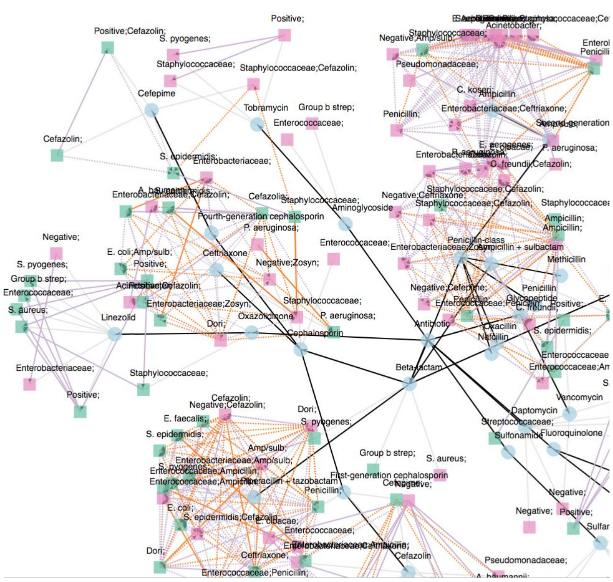


Fig. 5: Navigation View A: All information on screen - rule relationships showing clusters of result entities.

4.2 Navigation View

Given the chosen rule representation, the node-link diagram can still be quite complex, with hundreds of nodes and edges. Following Shneiderman's mantra of providing the user with an overview of the dataset first [44] and details on demand, we use an overview/navigation view to allow the user to choose what to explore. To ensure that the navigation view was comprehensible without overwhelming the user, we got expert feedback on three different views.

The first view showed all rules and relationships as shown in Figure 5, in keeping with Tufte's principle of increasing data density and presenting all information to the user [50]. This view shows the user clusters of rules that affect each other. Exploration through this navigation view would involve zooming into a cluster for a clearer view on rules in the cluster. However, it is overwhelming and hard to read at first glance. It takes time to process what items each cluster of rules affects.

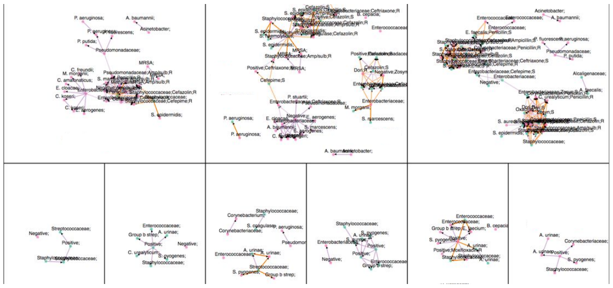


Fig. 6: Navigation View B: Small multiples view split by result entities.

In the second candidate view, to segment out rule-clusters by the result entities, we use a small multiples view where each view shows rules affecting a particular set of entities. This is shown in Figure 6. Small multiples view is a common technique for simplifying complex information, while maintaining data density [50]. Through this view it is clear exactly which entities are affected by a set of rule. Even then the cognitive load on the user is high.

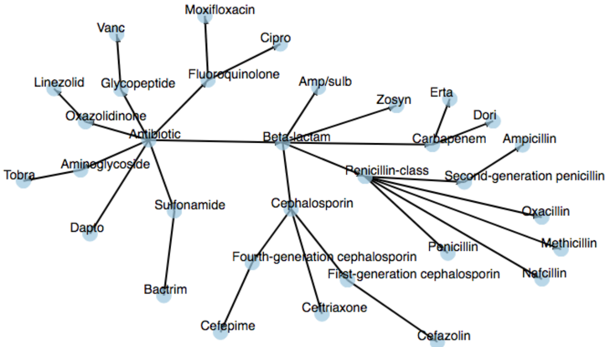


Fig. 7: Navigation View C: Only entity nodes are initially visible, which can be expanded to see rule nodes.

While the first two views used a bottom-up [52] approach where all the rules are shown first and then zooming into a cluster, the third view uses a top-down [44] approach. Shown in Figure 7, this view only shows entity nodes, which can be expanded to see relevant rules. This is another popular technique for simplifying complex graphs [53], where nodes are clustered up to their parents and available on demand. This view is the cleanest and lines up with the users' domain knowledge.

Based on expert feedback (Section 4.2) the third view was chosen as our navigation view. This approach reduces the cognitive load on the user, since the number of entity nodes will be much less than the number of rule nodes in most cases. This is because multiple rules will usually apply to the same entity. Nodes are expanded in place without triggering a recalculation of the force-directed graph, thus preserving context and the users' mental map of the visualization, a requirement for interactive exploration [34]. To avoid recalculation, all nodes are positioned when the visualization is loaded, but rule nodes are hidden.

4.3 Rule Relationships

Expanding an entity node shows all the relevant rule nodes, but seeing their relationships requires additional user interaction to maintain simplicity of the graph. When a user is interested in a rule, they can double click on the rule to see its relationship. Thus, the user is in control of how much information they want to see. Rule relationships are represented as directed edges, with edge color and stroke style encoding the relationship type. Rule relationships can be of four types, as shown in Figure 8:

1. *Conflict*: Two rules, which fill in opposite values either for the same result set or where one result set is a subset of the other, are said to be in direct conflict with each other. These relations are denoted by solid orange links.

2. *Subsumes*: When two rules fill in the same value and one result set is a subset of the other, the latter is said to subsume the former, i.e. $B \subseteq A \implies A$ *subsumes* B. If the result sets of two rules are identical and they fill in the same value, only one rule is shown. These are represented as purple solid lines.

3. *Partial Conflict*: Two rules which fill in *different* values and the result sets overlap, but neither one is a subset of the other, are said to be in partial conflict with each other. This is denoted by a dotted orange line.

4. *Overlap*: Two rules that fill in the *same* values and have overlapping subsets, but neither one is a complete subset of the other, are said to have an overlapping relationship. These are represented as purple dotted lines.

Thus, relationship edges between nodes of the same color will be purple while those between nodes of different colors will be orange.

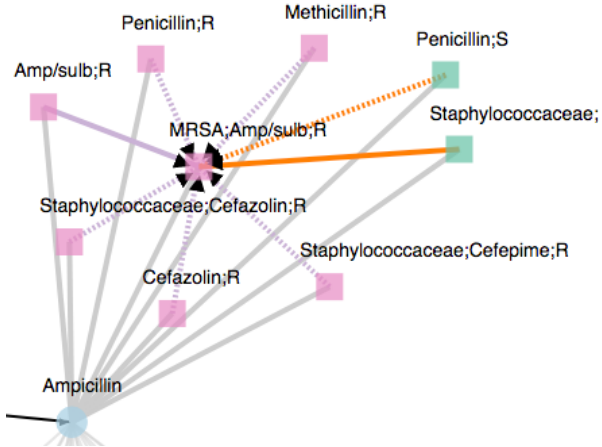
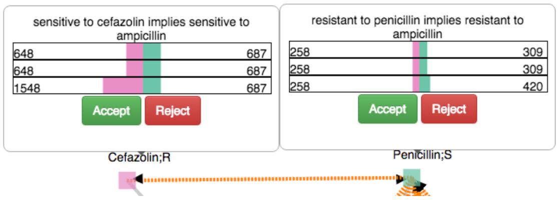


Fig. 8: Relationships between rules: conflicts shown in solid orange, partial conflicts shown in dotted orange, subsumes relation shown in solid purple and overlaps shown in dotted purple.

4.4 Data Summary

Along with seeing relationships between rules, users want to ensure that the rules for filling in unreported values match trends in the reported values. Additionally, if a rule fills in a large number of values, users might want to think more carefully about that rule. To enable such comparisons, we allow the user to preview the number of affected tuples and show the change in distribution of values. This is represented as three horizontal bar charts on a pop up, as shown in Figure 9, and is available on clicking on the rule node. The left side of the bar chart, in pink, represents the number of tuples which are *resistant* and right side in green represents the number of tuples which are *sensitive*. The top bar corresponds to the reported values in the data, the second bar corresponds to data filled in by other rules and the bottom one shows the distribution upon application of the current rule. If no rules have been applied, the first two bars are identical. The last bar adds the number of missing cells that will be changed upon rule application.


sensitive to cefazolin implies sensitive to ampicillin

648 687
648 687
1548 687

resistant to penicillin implies resistant to ampicillin

258 309
258 309
258 420

Fig. 9: On clicking a rule node, users can see how the number of tuples which are resistant versus sensitive will change upon rule application. The top bar shows the numbers in the reported data, the second one shows numbers filled in by other rules and the bottom bar shows the distribution, if the current rule were to be applied. All numbers are from a synthetic dataset. Users can also accept or reject a rule.

Thus, only one side of the bar will change depending on if the rule fills in *sensitive* or *resistant*. The raw numbers are also shown on the bars. Note that numbers shown in all images are from synthetic data.

While the bar chart representation is specific to rules that fill in binary values, providing tooltips with more information on the data is possible for most applications. The pop-up also contains the full rule text and buttons for accepting or rejecting a rule. The meaning of colors and bars is represented in a legend on the top of left corner of the screen (shown in Figure 1).

4.5 User Interactions

The user is able to interact with the graph in multiple ways going from the overview to increasing levels of detail, to get desired information, before making a decision [51]:

1. *Overview*: At the navigation level, the user is able to drag a node away from others before expanding it. Dragging a node also fixes it to its position to better enable this. Users can thus bring nodes of interest close together to compare.

2. *Node Detail*: From the navigation level, the user is able to expand entity nodes by double clicking. This shows and hides relevant rule nodes and also unfixes the node, so that entity nodes can be dragged with the rule nodes.

3. *Relationship Detail*: At the next level, once rule nodes are visible, single clicking on the rule node shows the data summary pop-ups, while double clicking reveals relationships. This way the user is able to first accept/reject rules based on the data, and additionally see relationships if required.

4. *Decision*: From the data summary pop-up, the user can provide decisions on the rule. Accepting a rule applies it to the entire dataset and updates the numbers on relevant data pop-ups. It automatically removes any conflicting or redundant rules from the visualizations. Similarly, inferred rejections from rejecting a rule are automatically removed.

4.6 Rule Updates

All the edges and data summaries are precomputed during page load. This means all nodes and edge positions are designated at load time, however only the entity nodes and edges are visible, until the user expands these. When the user accepts a rule, any rules that conflicts with or is subsumed by the accepted rule is removed, since the result sets of these rules are now empty, having been filled with the accepted rule. The data summary is updated only for rules that overlap or partially conflict with the accepted rule. On rejecting a rule, rules that subsume the rejected rule are also removed. That is, if a particular rule is incorrect, any rule whose result set is a superset of it is also incorrect. Pseudocode for rule acceptance and rejections are shown in Algorithm 1.

Algorithm 1 Rule Decision

```
procedure ACCEPT_RULE($A$)
    APPLY_RULE($A$)
    $update\_list \leftarrow \{A\}$
    for $node \in update\_list$ do
        for $B \in children(node)$ do
            if $node$ subsumes $B$ or $node$ conflicts with $B$ then
                $update\_list.append(B)$
            end if
        end for
        Remove $node$
    end for
    for $B \in nodes$ do
        if $A$ overlaps with $B$ or $A$ partially conflicts with $B$ then
            update data summary tab for $B$
        end if
    end for
end procedure
procedure REJECT_RULE($A$)
    $remove\_list \leftarrow \{A\}$
    for $node \in remove\_list$ do
        for $B \in parent(node)$ do
            if $B$ subsumes $node$ then
                $remove\_list.append(B)$
            end if
        end for
        Remove $node$
    end for
end procedure
```

5 Evaluation

Our tool is built using Python's Django web framework and MySql database on the backend and HTML/Javascript and D3 [11] for frontend visualizations. The colors for nodes and links were selected using Color brewer [22]. In this section, we report feedback obtained as part of the design process and the latency of our rule update algorithm [24].

5.1 Dataset

Three researchers on the study team (Hebert, Lustberg, Pancholi) created rules as part of a data processing task for a microbiology urine culture result dataset for patients admitted to the OSU Wexner Medical Center between 2011-2016. This dataset, annotated with organism and antibiotic classification information from the Unified Medical Language System (UMLS) Metathesaurus [10], included over 10,000 cultures and 50 antibiotics.

5.2 Design Process and User Feedback

We iteratively designed the tool based on expert feedback during the design process. The first time, we got informal feedback on rule representation, between the versions shown in Figures 3 and 4, since the view in Figure 2 was incomprehensible. The representation in Figure 4 was preferred due to its simplicity.

During the second round we got feedback on the navigation views. The three visualizations in Figures 5,6,7 were shown to one domain expert. For each view, they were asked what they liked, what they disliked, what information was not captured in the visualization and any open-ended comments they had. They liked the third view showing the antibiotic hierarchy. They found this view the easiest to navigate since it matched their domain knowledge. The other two views were found overwhelming and hard to process at first glance. We also learned that the movement of the force layout was visually unpleasing and updated our layout to decrease movement, as many other network evaluation work has reported. They mentioned that they would like to get more information on the impacted data.

In the third round, we showed them our final prototype and asked them to find insights. Some examples of insights that they were able to see are shown in Figures 10 and 11. In Figure 10, they could see a rule conflict that revealed that one rule was overly general (*Cephalosporins* cover all *Staphylococcus aureus*) which would be incorrect for the portion of *Staphylococcus aureus* which is *resistant* to *Beta-lactams* (i.e., *MRSA*). In fact, the more general and correct rule would be that *MRSA* is *resistant* to all *Beta-lactams*, shown by the purple edge. However, for this insight, both the *Beta-lactam* and *Cephalosporin* nodes have to be expanded. Along with identifying errors, areas of true disagreements can also be seen in nodes with a high number of orange edges. This is shown in Figure 11: there was no consensus among the experts on the coverage of *Enterococcus* organisms for *Ciprofloxacin*.

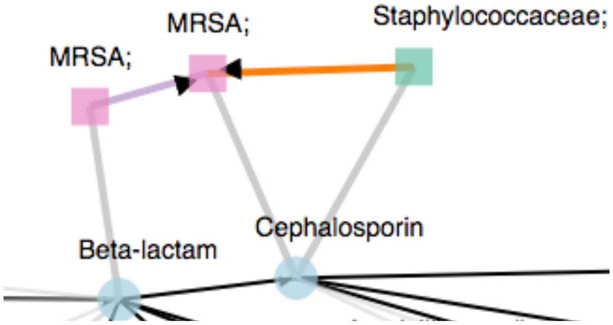


Fig. 10: Insights gained from visualization: The overly general rule: *Staphylococcaceae are sensitive to Cephalosporins* can be detected by its conflict with the correct rule: *MRSA is resistant to Cephalosporins.*

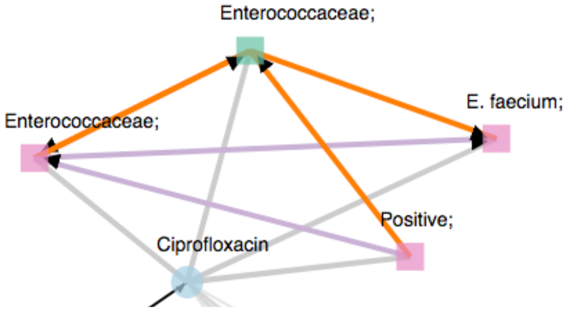


Fig. 11: Nodes with multiple conflict nodes draw attention to areas of subjectivity and disagreements. Whether *Enterococcaceae* organisms are *resistant* or *sensitive* to *Ciprofloxacin* is a debatable topic.

5.3 Latency

Since users are interactively applying rules, we need our rule update algorithm to maintain interactive performance. We simulated user interactions by iterating through each of the rules and accepting if it was a correct rule (provided by experts at a consensus meeting), otherwise rejecting it. The average rule application latency over each rule decision is 60*ms*, which is below the human threshold for perception [6]. Even with network latency, the total time will be well under 500*ms*, which is considered interactive [25, 33].

5.4 Discussion

While the expert liked the simplified antibiotic navigation view, it was too much effort to expand out various nodes and then see relationships. They would like hints on the overview nodes to see where they should start. For example, we could change the size of the antibiotic nodes to reflect those nodes which have the most missing data, or most rules, or most number of conflicts. Further, when there is a relationship between a visible node and an invisible node, the visible node can be linked to the parent of the invisible node, alerting the user to expand the parent node. In terms of improvements in design, the links can be changed to tethered edges to better show the direction of relationships.

This visualization shows the rules at the top level and then allows the user to get to the data. An alternate method would be to visualize the dataset and then see which rules filled in outlying values in the dataset. A controlled study comparing both these visualization techniques and seeing which one allows the experts to resolve conflicts and errors the quickest is another area we would like to explore.

We got positive feedback on the data summary pop ups, with comments indicating that more detailed information such as specific organisms affected would be helpful as well. After incorporating these edits, we hope to do a formal evaluation of our tool where we will observe the workflow of experts as they resolve conflicts in two new datasets by selecting and rejecting rules until no conflicts or overlaps remain.

6 Conclusion and Future Work

In this paper, we have described our design for visualizing relationships between rules, which are used for many tasks such as data cleaning and entity resolution. Our visualization provides a simple navigation view which matches the mental model of users and then allows them to expand nodes to see relevant rules. Details such as impact on dataset and conflicts and overlaps with other rules are available on demand. Through our system users can interactively find and resolve conflicts and dependencies. Initial feedback on our system is positive and we hope to do a formal evaluation after incorporating suggested changes.

Acknowledgment This work is supported by the NSF under awards IIS-1422977, IIS-1527779, CAREER IIS-1453582, IIS-1302755, DBI-1260765, DBI-1062057, CNS-1531491, NIST MSE-70NANB13H181, and by the NIAID of the NIH under award R01AI116975.

References

[1] J. Abello, F. Van Ham, and N. Krishnan. Ask-Graphview: A Large Scale Graph Visualization System. *IEEE transactions on visualization and computer graphics*, 12(5):669–676, 2006.

[2] B. Alper, N. Riche, G. Ramos, and M. Czerwinski. Design Study of LineSets, A Novel Set Visualization Technique. *IEEE Transactions on Visualization & Computer Graphics*, (12):2259–2267, 2011.

[3] B. Alsallakh, L. Micallef, W. Aigner, H. Hauser, S. Miksch, and P. Rodgers. The State-of-the-Art of Set Visualization. In *Computer Graphics Forum*, vol. 35, pp. 234–260. Wiley Online Library, 2016.

[4] D. Archambault, T. Munzner, and D. Auber. GrouseFlocks: Steerable Exploration of Graph Hierarchy Space. *IEEE transactions on visualization and computer graphics*, 14(4):900–913, 2008.

[5] D. Archambault, H. C. Purchase, and B. Pinaud. The Readability of Path-preserving Clusterings of Graphs. In *Computer Graphics Forum*, vol. 29, pp. 1173–1182. Wiley Online Library, 2010.

[6] B. P. Bailey, J. A. Konstan, and J. V. Carlis. The Effects of Interruptions on Task Performance, Annoyance, and Anxiety in the User Interface. In *Interact*, vol. 1, pp. 593–601, 2001.

[7] F. Beck, M. Burch, S. Diehl, and D. Weiskopf. A Taxonomy and Survey of Dynamic Graph Visualization. In *Computer Graphics Forum*, vol. 36, pp. 133–159. Wiley Online Library, 2017.

[8] J. Bertin. Semiology of Graphics: Diagrams, Networks, Maps. 1983.

[9] J. Blanchard, F. Guillet, and H. Briand. Interactive Visual Exploration of Association Rules with Rule-focusing Methodology. *Knowledge and Information Systems*, 13(1):43–75, 2007.

[10] O. Bodenreider. The Unified Medical Language System (UMLS): Integrating Biomedical Terminology. *Nucleic Acids Research*, 32(suppl_1):D267–D270, 2004.

[11] M. Bostock, V. Ogievetsky, and J. Heer. D^3 data-driven documents. *IEEE Transactions on Visualization & Computer Graphics*, (12):2301–2309, 2011.

[12] M. Bruls, K. Huizing, and J. J. Van Wijk. Squarified Treemaps. In *Data visualization 2000*, pp. 33–42. Springer, 2000.

[13] S. P. Callahan, J. Freire, E. Santos, C. E. Scheidegger, C. T. Silva, and H. T. Vo. VisTrails: Visualization Meets Data Management. In *Proceedings of the 2006 ACM SIGMOD international conference on Management of data*, pp. 745–747. ACM, 2006.

[14] C. Chambers, A. Raniwala, F. Perry, S. Adams, R. R. Henry, R. Bradshaw, and N. Weizenbaum. FlumeJava: Easy, Efficient Data-parallel Pipelines. In *ACM Sigplan Notices*, vol. 45, pp. 363–375. ACM, 2010.

[15] C. Collins, G. Penn, and S. Carpendale. Bubble Sets: Revealing Set Relations with Isocontours over Existing Visualizations. *IEEE Transactions on Visualization & Computer Graphics*, (6):1009–1016, 2009.

[16] E. Czaplicki. Elm: Concurrent frp for functional guis. *Senior thesis, Harvard University*, 2012.

[17] K. Dinkla, M. J. van Kreveld, B. Speckmann, and M. A. Westenberg. Kelp Diagrams: Point Set Membership Visualization. In *Computer Graphics Forum*, vol. 31, pp. 875–884. Wiley Online Library, 2012.

[18] P. M. Dubois, Z. Han, F. Jiang, and C. K. Leung. An Interactive Circular Visual Analytic Tool for Visualization of Web Data. In *IEEE/WIC/ACM International Conference on Web Intelligence (WI)*, pp. 709–712. IEEE, 2016.

[19] T. M. Fruchterman and E. M. Reingold. Graph Drawing by Force-directed Placement. *Software: Practice and experience*, 21(11):1129–1164, 1991.

[20] H. Garcia-Caballero, M. Campos, J. M. Juarez, and F. Palacios. Visualization in Clinical Decision Support System for Antibiotic Treatment. In *CAEPIA*, pp. 9–12, 2015.

[21] A. Graham, Y. Liang, L. Gruenwald, and C. Grant. Formalizing Interruptible Algorithms for Human Over-the-loop Analytics. In *Big Data (Big Data), 2017 IEEE International Conference on*, pp. 4378–4383. IEEE, 2017.

[22] M. Harrower and C. A. Brewer. ColorBrewer.org: An Online Tool for Selecting Colour Schemes for Maps. *The Cartographic Journal*, 40(1):27–37, 2003.

[23] I. Herman, G. Melançon, and M. S. Marshall. Graph Visualization and Navigation in Information Visualization: A Survey. *IEEE Transactions on visualization and computer graphics*, 6(1):24–43, 2000.

[24] L. Jiang, P. Rahman, and A. Nandi. Evaluating Interactive Data Systems: Workloads, Metrics, and Guidelines. In *Proceedings of the 2018 International Conference on Management of Data*, pp. 1637–1644. ACM, 2018.

[25] R. Jota, A. Ng, P. Dietz, and D. Wigdor. How Fast is Fast Enough?: A Study of the Effects of Latency in Direct-touch Pointing Tasks. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pp. 2291–2300. ACM, 2013.

[26] K. Kashofer, D. Kalkofen, M. Streit, A. Lex, C. Partl, and D. Schmalstieg. enRoute: Dynamic Path Extraction from Biological Pathway Maps for in-depth Experimental Data Analysis. In *2012 IEEE Symposium on Biological Data Visualization (BioVis)*, pp. 107–114. IEEE, 2012.

[27] C. K. Leung, V. V. Kononov, A. G. Pazdor, and F. Jiang. PyramidViz: Visual Analytics and Big Data Visualization for Frequent Patterns. In *14th International Conference on Dependable, Autonomic and Secure Computing, 14th International Conference on Pervasive Intelligence and Computing, 2nd Intl Conf on Big Data Intelligence and Computing and Cyber Science and Technology Congress (DASC/PiCom/DataCom/CyberSciTech)*, pp. 913–916. IEEE, 2016.

[28] C. K.-S. Leung and F. Jiang. RadialViz: An Orientation-Free Frequent Pattern Visualizer. In *Pacific-Asia Conference on Knowledge Discovery and Data Mining*, pp. 322–334. Springer, 2012.

[29] C. K.-S. Leung, F. Jiang, and P. P. Irani. FpMapViz: A Space-filling Visualization for Frequent Patterns. In *2011 11th IEEE International Conference on Data Mining Workshops*, pp. 804–811. IEEE, 2011.

[30] A. Lex, N. Gehlenborg, H. Strobelt, R. Vuillemot, and H. Pfister. UpSet: Visualization of Intersecting Sets. *IEEE Transactions on Visualization and Computer Graphics*, 20(12):1983–1992, 2014.

[31] A. Lex, C. Partl, D. Kalkofen, M. Streit, S. Gratzl, A. M. Wassermann, D. Schmalstieg, and H. Pfister. Entourage: Visualizing Relationships between Biological Pathways using Contextual Subsets. *IEEE Transactions on Visualization and Computer Graphics*, 19(12):2536–2545, 2013.

[32] G. Li, A. C. Bragdon, Z. Pan, M. Zhang, S. M. Swartz, D. H. Laidlaw, C. Zhang, H. Liu, and J. Chen. VisBubbles: A Workflow-driven Framework for Scientific Data Analysis of Time-varying Biological Datasets. In *SIGGRAPH Asia 2011 Posters*, p. 27. ACM, 2011.

[33] Z. Liu and J. Heer. The Effects of Interactive Latency on Exploratory Visual Analysis. *IEEE Transactions on Visualization & Computer Graphics*, (1), 2014.

[34] K. Misue, P. Eades, W. Lai, and K. Sugiyama. Layout Adjustment and the Mental Map. *Journal of Visual Languages & Computing*, 6(2):183–210, 1995.

[35] D. Moritz, D. Halperin, B. Howe, and J. Heer. Perfopticon: Visual Query Analysis for Distributed Databases. In *Computer Graphics Forum*, vol. 34, pp. 71–80. Wiley Online Library, 2015.

[36] D. G. Murray, M. Schwarzkopf, C. Smowton, S. Smith, A. Madhavapeddy, and S. Hand. CIEL: A Universal Execution Engine for Distributed Dataflow Computing. In *Proc. 8th ACM/USENIX Symposium on Networked Systems Design and Implementation*, pp. 113–126, 2011.

[37] C. Partl, S. Gratzl, M. Streit, A. M. Wassermann, H. Pfister, D. Schmalstieg, and A. Lex. Pathfinder: Visual analysis of Paths in Graphs. In *Computer Graphics Forum*, vol. 35, pp. 71–80. Wiley Online Library, 2016.

[38] C. Partl, A. Lex, M. Streit, H. Strobelt, A.-M. Wassermann, H. Pfister, and D. Schmalstieg. ConTour: Data-driven Exploration of Multi-relational Datasets for Drug Discovery. *IEEE transactions on visualization and computer graphics*, 20(12):1883–1892, 2014.

[39] O. Pilipczuk and G. Cariowa. The New Module for Rules Discovering and Visualization for NovoSpark® Visualizer software. *Przeglad Elektrotechniczny*, 91(11):197–200, 2015.

[40] P. Rahman, E. M. Hade, A. Nandi, P. Pancholi, M. Lustberg, K. Stevenson, and C. Hebert. Derivation of Expert Consensus Rules for Missing Antimicrobial Susceptibility Data. *AMIA*, 2018.

[41] P. Rahman, C. Hebert, and A. Nandi. Enabling Effective Data Interaction for Domain Experts. In *2018 IEEE International Conference on Healthcare Informatics (ICHI)*, pp. 465–466. IEEE, 2018.

[42] P. Rahman, C. Hebert, and A. Nandi. ICARUS: Minimizing Human Effort in Iterative Data Completion. *PVLDB*, 11(13), 2018.

[43] P. Rodgers, G. Stapleton, and P. Chapman. Visualizing Sets with Linear Diagrams. *ACM Transactions on Computer-Human Interaction (TOCHI)*, 22(6), 2015.

[44] B. Shneiderman. The Eyes Have It: A Task by Data Type Taxonomy for Information Visualizations. In *Visual Languages, 1996. Proceedings., IEEE Symposium on*, pp. 336–343. IEEE, 1996.

[45] R. Singh, V. V. Meduri, A. Elmagarmid, S. Madden, P. Papotti, J.-A. Quiané-Ruiz, A. Solar-Lezama, and N. Tang. Synthesizing Entity Matching Rules by Examples. *Proceedings of the VLDB Endowment*, 11(2):189–202, 2017.

[46] A. M. Smith, W. Xu, Y. Sun, J. R. Faeder, and G. E. Marai. RuleBender: Integrated Modeling, Simulation and Visualization for Rule-Based Intracellular Biochemistry. *BMC Bioinformatics*, 13(8):S3, 2012.

[47] A. Srinivasan, H. Park, A. Endert, and R. C. Basole. Graphiti: Interactive Specification of Attribute-Based Edges for Network Modeling and Visualization. *IEEE Transactions on Visualization & Computer Graphics*, (1):1–1, 2018.

[48] J. Stoyanovich, B. Howe, and H. Jagadish. Special Session: A Technical Research Agenda in Data Ethics and Responsible Data Management. In *Proceedings of the 2018 International Conference on Management of Data*, pp. 1635–1636. ACM, 2018.

[49] D. Thompson, J. Braun, and R. Ford. *OpenDX: paths to visualization; materials used for learning OpenDX the open source derivative of IBM's visualization Data Explorer*. Visualization and Imagery Solutions, 2004.

[50] E. R. Tufte. The Visual Display of Quantitative Information. *Quantitative Information*, 1983.

[51] S. Van den Elzen and J. J. Van Wijk. Multivariate Network Exploration and Presentation: From Detail to Overview via Selections and Aggregations. *IEEE Transactions on Visualization and Computer Graphics*, 20(12):2310–2319, 2014.

[52] F. Van Ham and A. Perer. Search, Show Context, Expand on Demand: Supporting Large Graph Exploration with Degree-of-Interest. *IEEE Transactions on Visualization and Computer Graphics*, 15(6), 2009.

[53] K. Wongsuphasawat, D. Smilkov, J. Wexler, J. Wilson, D. Mané, D. Fritz, D. Krishnan, F. B. Viégas, and M. Wattenberg. Visualizing dataflow graphs of deep learning models in TensorFlow. *IEEE transactions on visualization and computer graphics*, 24(1):1–12, 2018.

[54] K. Wu, L. Sun, C. Schmidt, and J. Chen. Graph Query Algebra and Visual Proximity Rules for Biological Pathway Exploration. *Information Visualization*, 16(3):217–231, 2017.

[55] Y. Wu, X. Zhu, J. Chen, and X. Zhang. EINVis: A Visualization Tool for Analyzing and Exploring Genetic Interactions in Large-scale Association Studies. *Genetic epidemiology*, 37(7):675–685, 2013.

[56] D. Xin, L. Ma, J. Liu, S. Macke, S. Song, and A. Parameswaran. Accelerating Human-in-the-loop Machine Learning: Challenges and Opportunities. In *Proceedings of the Second Workshop on Data Management for End-To-End Machine Learning*, p. 9. ACM, 2018.

[57] P. Xu, F. Du, N. Cao, C. Shi, H. Zhou, and H. Qu. Visual Analysis of Set Relations in a Graph. In *Computer Graphics Forum*, vol. 32, pp. 61–70. Wiley Online Library, 2013.

[58] K. Yang, J. Stoyanovich, A. Asudeh, B. Howe, H. Jagadish, and G. Miklau. A Nutritional Label for Rankings. In *Proceedings of the 2018 International Conference on Management of Data*, pp. 1773–1776. ACM, 2018.

[59] S. Zhao, M. J. McGuffin, and M. H. Chignell. Elastic Hierarchies: Combining Treemaps and Node-link Diagrams. 2005.

[60] Y. Zhu, L. Sun, A. Garbarino, C. Schmidt, J. Fang, and J. Chen. PathRings: A Web-Based Tool for Exploration of Ortholog and Expression Data in Biological Pathways. *BMC bioinformatics*, 16(1):165, 2015.