Eindhoven University of Technology

MASTER

JuxtaSet
visualization of time-varying patterns and missing data

Drost, N.L.

Award date:
2017

Link to publication

Disclaimer
This document contains a student thesis (bachelor's or master's), as authored by a student at Eindhoven University of Technology. Student theses are made available in the TU/e repository upon obtaining the required degree. The grade received is not published on the document as presented in the repository. The required complexity or quality of research of student theses may vary by program, and the required minimum study period may vary in duration.

General rights
Copyright and moral rights for the publications made accessible in the public portal are retained by the authors and/or other copyright owners and it is a condition of accessing publications that users recognise and abide by the legal requirements associated with these rights.

- Users may download and print one copy of any publication from the public portal for the purpose of private study or research.
- You may not further distribute the material or use it for any profit-making activity or commercial gain

TU/e Technische Universiteit
Eindhoven
University of Technology

Department of Mathematics and Computer Science
Algorithms and Visualization

JuxtaSet: Visualization of Time-varying Patterns and Missing Data

Master Thesis

Niels Drost

Supervisors:
Prof.dr.ir. Jarke J. van Wijk
dr.ir. Wieke Altorf-van der Kuil
dr. Yingqian Zhang

Final version

Eindhoven, September 25, 2017

Abstract

Epidemiologists need to monitor complex resistance patterns of multi-drug resistant bacteria, and especially since bacteria are increasingly resistant to antibiotics. Their analysis, however, is limited to individual antibiotics. Hence, epidemiologists want to gain more insight in multi-drug resistance patterns and their relations with other attributes. The multivariate nature of the antibiotic susceptibility testing data and a multitude of missing data complicates this task. A new visualization system is designed that allows for rapid exploration of these patterns.

These patterns are combined patterns and exact patterns. Combined patterns and exact patterns are intersecting sets and exclusive set intersections in set terminology respectively. Using JuxtaSet, users can quickly analyze combined and exact patterns, and monitor data over time. Moreover, the users can ascertain how trustworthy the imputation of the data is, and they can get insight in attributes. JuxtaSet visualizes combined and exact patterns accompanied by sparklines depicting temporal fluctuations. Imputation of missing data is visualized as aggregates, and as distinct patterns on demand. Additional attributes are visualized with interactive bar charts. It uses scrollable non-overlapping matrix layouts to increase scalability and with linked highlighting, attributes and patterns can be explored. Epidemiologists at the Netherlands National Institute for Public Health and the Environment have shown during evaluation that JuxtaSet gives insight into these complex data in a convenient and user-friendly way. Given some practice or explanation, they were able answer relevant questions to their expertise.

Preface

Here I thank my supervisor Prof.dr.ir. Jarke J. van Wijk for the many inspiring and enthusiastic feedback I have received, MSc. Loes Soetens, dr.ir. Wieke Altorf-van der Kuil and dr. Tjalling Leenstra for their ideas, involvement in this study and their feedback, dr. Yingqian Zhang for being part of the assessment committee and participants of the user evaluation for participating in the user evaluation. I thank dr. Joos Buijs for providing a TU/e LaTeX template.

Contents

Chapter 1

Introduction

Bacteria are increasingly resistant to antibiotics [27], which is an emerging problem for public health, especially in the case of multi-drug resistance. It is necessary to enable epidemiologists to monitor multi-drug resistance patterns and the time trends of these patterns. Multi-drug resistance patterns describe the combinations of antibiotics to which bacteria are resistant. Epidemiologists at the Netherlands National Institute for Public Health and the Environment are daily operating with antibiotic resistance data. However, currently epidemiologists are limited to analyzing this data per individual antibiotic. Since epidemiologists want to gain more insight in the multi-drug resistance patterns, visualization can be a useful approach to this end. Typical questions epidemiologists need to answer: *What is the percentage of isolates resistant against both Piperacilline and Amoxicilline? Has this percentage increased over the past years?*

Bacterial resistance patterns are a form of set-typed data. The problem of visualization of set-typed data are addressed by set visualization techniques. In set-typed data, elements belong to multiple sets, here isolates are resistant against multiple antibiotics. Also, set-typed data sets are equivalent to datasets with multiple binary categorical variables [11]. Such set-typed datasets are abundant. For instance, movies have multiple genres, people practice multiple sports and supermarket transactions contain a list of products. Typically, there are a high number of elements (bacteria, movies, people, transactions) and a relatively low number of categorical variables (antibiotics, genres, sports, products). We define a pattern as a specific combination of memberships of sets, or, equivalently, a specific pattern of Boolean values for the different variables. There are a number of key terms related to the discussed concepts, which are described later in detail.

Given a pattern, we can count how many elements match this pattern. In some set-typed datasets the number of elements in a pattern can change over time, such as the increase of multi-drug resistant bacteria, or changing consumer behavior over time, for instance, people will buy more ice-cream in the summer. Furthermore, real world datasets are often not complete, because of incomplete or erroneous observations by humans. Since missing data can occur for every variable and element, missing data results in a high number of distinct patterns. There are two options to deal with the problem of missing data: imputation of the data versus removing incomplete cases from the data. Often there is a preference for imputation for a number of reasons: removing incomplete records introduces considerable bias and also results in a lower statistical power in statistical data analysis [32]. Yet the question remains after imputation whether the imputation is trustworthy. Visualization of imputed values of such data is necessary, such that researchers can validate and feel confident about the imputation of missing data or possibly adjust the imputation techniques. We propose a visualization technique that allows users to explore, monitor and analyze these complex, time-dependent datasets with missing data in detail.

Set-typed data can be described as (frequent) itemsets using the Market-Basket model described by Leskovec et al.. Following the Market-Basket model, there are many transactions with a relatively small number of items. Frequent itemsets are combinations of items that appear a certain minimum number of times in transactions [19]. However, since the notion of *items* is open for multiple interpretations, we use definitions that are more intuitive. *Sets*, *set-typed attributes*, *items*, *categorical variables*, *properties* all refer to the same concept. We find *properties* the most intuitive if an object in some context *has* a number of *properties*. An overview of the definitions used in literature is given in table 1.1. In set visualization literature *transactions* are called *elements* and the *items* are called *sets*, the *itemsets* are described as *intersecting sets* or as *overlapping sets* [36, 21, 4]. To keep in line with epidemiological and microbiological jargon, itemsets are called *combined patterns*. Hence, combined patterns with properties X refer to elements with *at least* all properties X. Elements or transactions in the dataset that have exactly the same properties are known as *exclusive intersections* in set visualization literature [21], whilst we call these *exact patterns*. The name exact patterns refers to the fact that these are patterns with exactly properties Y and exactly not properties Z. The *degree* of an *exact pattern* is the number of properties it positively encapsulates, hence the size of Y from the previous example. The *degree* of a *combined pattern* is equivalent to the number of properties the pattern describes, hence the size of X in the previous example. Next to properties, elements have attributes. Attributes give additional information about an element and are numerical or nominal variables.

Table 1.1: Overview definitions

Set visualization	**Market basket model**	**JuxtaSet**	**Example**
Set, set-typed attribute	Item	Property	Ingredient
Overlapping set, intersecting set	Itemset	Combined pattern	Recipes with at least the ingredients tomato and basil
Exclusive intersection	-	Exact pattern	Recipes with exactly the ingredients tomato, basil, mozzarella and pasta
Element	Transaction	Element	Single recipe
Attribute	Attribute	Attribute	recipe's rating
Set-degree, degree	-	Degree	Recipes with exactly 5 ingredients
Cardinality	Frequency	Frequency	10 recipes

Much work has been done to visualize set-typed data, but many fall short to address our challenge. In Chapter 3 an overview is given of related work. There we discuss three leading set-visualization techniques that tackle problems closest to ours. There are six categories of set-visualization techniques: Euler-based diagrams, overlays, node-link diagrams, matrix-based techniques, aggregation-based techniques and scatter plots [5]. Only three types of set-typed data visualization techniques have shown to be scalable: aggregation based techniques, matrix based techniques and scatter plots [5]. Scatter plots are less preferable, since the dots represent sets while people perceive them as elements [5]. Visualization of exact patterns is often either visualized cluttered, implicitly or not at all [3]. However, bacteria tend to have characteristic resistance patterns and gaining information which patterns exist is crucial. Visualization techniques that do explicitly visualize exact patterns are Upset and PowerSet [3, 21]. UpSet visualizes both exact patterns and combined patterns. AggreSet on the other hand focuses on the visualization of combined patterns. The work of this paper is inspired by AggreSet, Upset, PowerSet and RadialSets. These visualization techniques are well designed and offer great solutions for completing specific tasks, and we used interesting features for our system. The former three visualization techniques are discussed in more detail Chapter 3. Next to visualization of patterns, gaining insight in the relation between patterns and attributes is important. The visualization techniques discussed in this thesis are all able to visualize these relations to varying extent [3, 21, 36].

Furthermore, there is to our knowledge no set visualization technique that visualizes the time-varying frequency of the patterns and none of these techniques tackles the problem of missing data. We have therefore developed a new approach to set visualization, called JuxtaSet, and implemented a prototype application. Figure 1.1 shows a screenshot of the interface of JuxtaSet. JuxtaSet enables users to analyze exact and combined patterns, and to study time dependency and missing data. Since both combined patterns and exact patterns and their interplay are important, these patterns are visualized in juxtaposed, interactive, and interlinked matrices. The matrices are connected by linked highlighting and synchronous sorting and filtering. To show temporal variation, sparklines are given for each pattern. Moreover, the quantity of missing data is shown, hence users are informed when less confident conclusions can be drawn about patterns with more missing data. For detailed information on missing data, on demand a third matrix is constructed listing the unique patterns where data was missing.

In our work a different perspective on data is proposed than is used in many set visualization techniques. This system was originally designed for epidemiologists at the Netherlands National Institute for Public Health and the Environment. They are mostly interested in the patterns within the data and have less interest in the individual elements that underlie the data. Only in highly incidental cases, where a unique and dangerous resistance pattern is found, the epidemiologists would be interested in which patients these isolates have been measured, such that these patients can be warned and possibly quarantined. JuxtaSet therefore focuses on the visualization of patterns. This is in contrast to visualization systems that visualize for instance a movie database, a dataset used many times to present set visualization techniques. Users of these databases have more feeling for the elements than for the patterns. Upon selection and filtering, usually a list of elements appears, and users will find for instance that Pulp Fiction is a Crime/Drama movie. The users of JuxtaSet have no knowledge of and are hardly interested in single elements and will not use the system to find elements, but instead to explore and monitor patterns. This results in different priorities, and hence the analysis of patterns is central in our exploration pipeline.

In this thesis, first a background is given on the work of epidemiologists at the Netherlands National Institute for Public Health and the Environment. Second, a comparison of visualization techniques is given in section 3. In Chapter 4, JuxtaSet and its components are described in detail, and in Chapter 5 the implementation of JuxtaSet and its scalability are discussed. We show in Chapter 6 that JuxtaSet is generic. A specific use case and its evaluation are described in Chapter 7 and this paper concludes with a discussion and conclusion in Chapter 8 and Chapter 9.

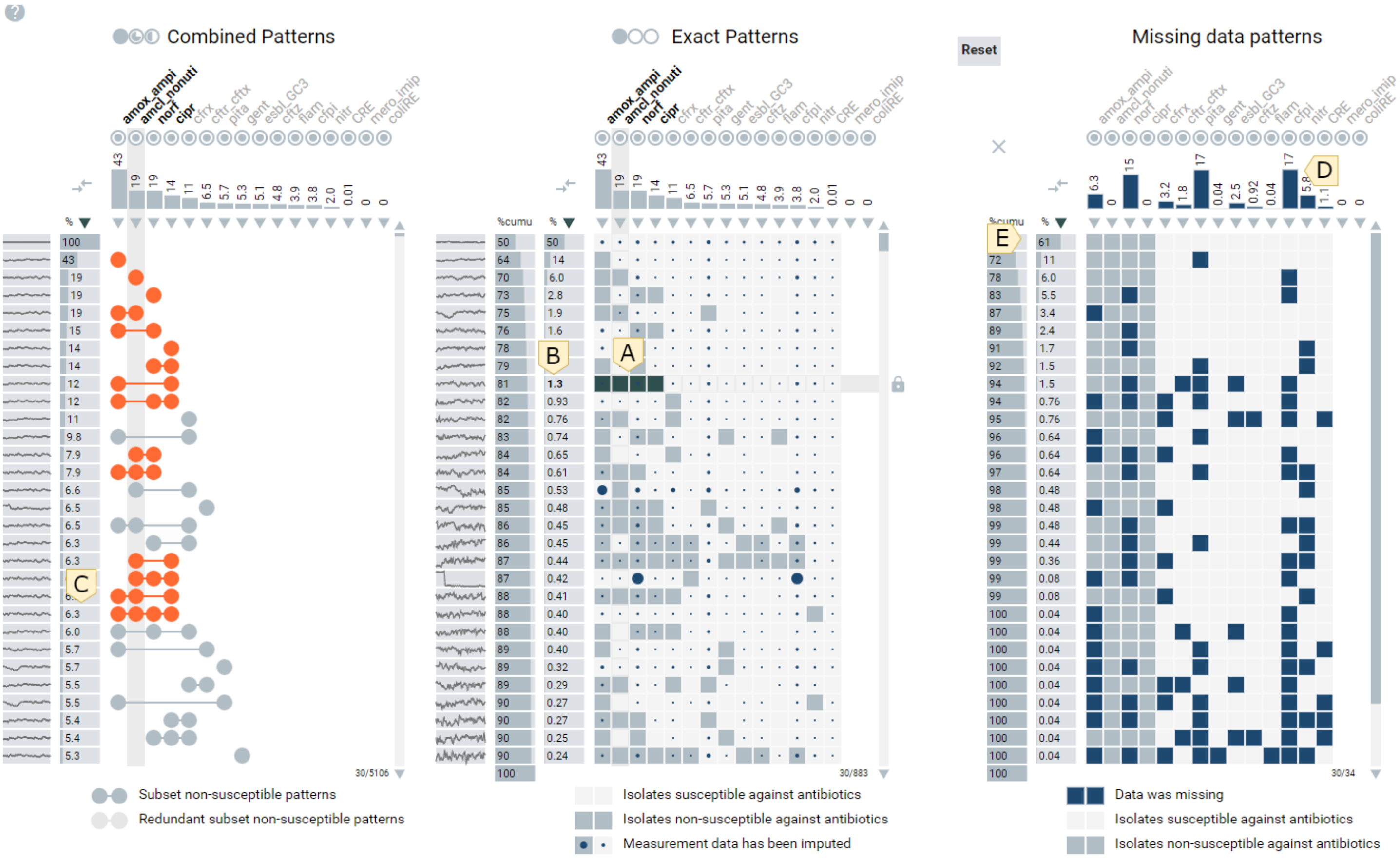

Combined Patterns
Exact Patterns
Reset
Missing data patterns
amox_ampi
amcl_nonuti
norf
cipr
cfrx
cftr_cftx
pita
gent
esbl_GC3
cftz
flam
cfpi
nitr
CRE
mero_imip
coliRE
43
19
19
14
11
6.5
5.7
5.3
5.1
4.8
3.9
3.8
2.0
0.01
0
0
%
%cumu
30/5106
30/883
30/34
A
B
C
D
E
Subset non-susceptible patterns
Redundant subset non-susceptible patterns
Isolates susceptible against antibiotics
Isolates non-susceptible against antibiotics
Measurement data has been imputed
Data was missing
Isolates susceptible against antibiotics
Isolates non-susceptible against antibiotics

Figure 1.1: The main screen of JuxtaSet, visualizing antibiotic resistance patterns using scrollable, non-overlapping matrix layouts. A number of patterns is highlighted in the left view. These patterns are combined patterns that match with the selected exact pattern in the middle view (A), consisting of the first four antibiotics. The selected exact pattern has a frequency of 1.3% (B), whilst the combined pattern with combined resistances *amox_ampi*, *amcl_nonuti*, *norf* and *cipr* has a frequency of 6.3% (C). Moreover, for the selected pattern more detailed information is shown where data was missing in distinct patterns in the right view. The right view show us that for 17% of the elements from the selected pattern data was missing for *pita* and *cfpi* (D), and 15% was missing for *norf*. 61% of the elements in this pattern had no missing data.

Chapter 2

Background

The data that has driven this project is from the Infectious Diseases Surveillance Information System for Antimicrobial Resistance (*ISIS-AR*), the Dutch national antimicrobial resistance surveillance system. ISIS-AR exists since 2008 and is a collaborative project between the Dutch Society for Medical Microbiology (NVMM) and the Netherlands National Institute for Public Health and the Environment (RIVM). It is maintained by the Centre for Infectious Disease Control (CIb) at the RIVM. The system has been set up to monitor the magnitude and trends of antimicrobial resistance, as well as outbreaks of antimicrobial resistance. Analysis of antimicrobial resistance data supports the development of empiric antibiotic treatment guidelines, and facilitates scientific research. Empiric treatment is treatment given based on experience and surveillance data, while the results of bacterial culture and other tests are awaited. The system contributes to quality, safety and reduction of health care. [18]

2.1 ISIS-AR

ISIS-AR consists of a collection of routine *Antimicrobial susceptibility testing* (AST) data from 40 Dutch medical microbiological laboratories comprising around 75% of Dutch hospitals [1]. More specifically, the database contains data on bacterial isolates from wounds, urine, blood and other human material, for which antibiotic resistance is tested. Additionally, information on the patients from which the isolates are sampled are stored. Bacterial isolates can be manually tested for antibiotic resistances using an antibiotic disk or gradient-test, or automatically using a machine. Based on the testing result the isolates are categorized for each antibiotic as susceptible, intermediately resistant or definitely resistant. Example questions about this dataset in which epidemiologists are interested to answer, are: *What are the most frequent resistance patterns? With what antibiotic is ESBL most frequently associated? Which exact resistance pattern is most frequently occurring with resistance for Norfloxacine? What percentage of the isolates is both resistant against at least Cefuroxim and Ceftazidim?* These and other questions were asked during user evaluation, which is discussed in Chapter 7.

In order to avoid overcomplication of the information visualization, we reduced the number of possible AST indications: intermediately resistant and definitely resistant are combined into one classification: *non-susceptible*. Hence, we retrieved from ISIS-AR a matrix of bacterial isolates versus antibiotics containing three possible values: susceptible, non-susceptible, and missing. This data was joined with general data per isolate, such as the age and gender of the patient from which the isolate was sampled, and the medical laboratory where the AST was conducted.

In the ISIS-AR database, test-data on 162 different antibiotics under supervision by the RIVM and 30 confirmation tests can be found. There are 51 different antibiotic classifications, classified according to the biological mechanism. Bacteria can be inherently resistant to specific mechanisms of classifications, and therefore be resistant against all antibiotics within the class. In ISIS-AR there is data of 2353 different micro-organisms. In NethMap by de Greeff and Mouton [10], only the 16 most clinically relevant different microbes are discussed. Most isolates in the 2017 NethMap study by RIVM are from 138k *E. coli* isolates followed by 44k *S. Aureus* isolates and 19k Coagulase negative *Staphylococci* (CNS) [10]. For development of JuxtaSet, we used a dataset with 189k *E. coli* isolates and 54 antibiotics or confirmation tests.

Epidemiologists at RIVM check the monthly delivered data from the Dutch medical laboratories on exceptional results and give the labs feedback on these exceptional results. Exceptional results can be: exceptional resistance levels compared to national levels or compared to previous results from the specific laboratory; antibiotic usage by doctors that is deviant from guidelines; and exceptional microbial phenotypes. Based on the reply from the laboratory, any incorrect data is adjusted. Monitoring of laboratory data and giving feedback can lead to increased reliability of antibiotic resistance surveillance and possibilities to compare data [18]. Ultimately, it can lead to increased quality the medical laboratories. With the use of data from ISIS-AR, epidemiologists are able to monitor the emergence of antibiotic resistant bacteria and surveillance helps to improve empiric treatment [26]. Also, with ISIS-AR and statistical methods epidemiologists can gain insight in spread of highly resistant microbes (HRMOs) across hospitals and therefore contribute to prevention of HRMO spreading. Since ISIS-AR contains data from many hospitals and medical laboratories, antimicrobial outbreaks that have gone unnoticed in laboratories locally, can be detected by ISIS-AR [18]. Apart from ISIS-AR, there are several other surveillance programs in the Netherlands that monitor antibiotic resistance: programs monitoring antibiotic resistance in bacteria from other environments; monitoring specific bacteria; and monitoring specific resistance mechanisms [10]. Conclusions drawn from such surveillance programs are for instance, quoted from de Greeff and Mouton [10]:

- "For isolates from urine cultures a distinction was made for patients aged below and above 12 years of age in accordance with age categories used in the urinary tract infection guidelines of the Dutch College of General Practitioners (NHG). In general, resistance rates in the older age group were slightly higher than in the younger age group."

- "In *P. mirabilis*, there was a significant and clinically relevant decrease in resistance to amoxicillin/ ampicillin in patients aged 12 years and to co-amoxiclav in patients aged 12 years to 5% in 2016."

- "Intensive Care Units: In K. pneumoniae, resistance to piperacillin-tazobactam and gentamicin decreased significantly. In E. cloacae, significant decreases were found for ciprofloxacin, gentamicin, tobramycin, co-trimoxazole and percentage HRMO. In S. aureus, a significant decreasing trend was observed for ciprofloxacin and also in coagulase-negative."

- "Aminoglycoside resistance appears to have decreased slightly again but this does not have implications for therapy at present. High levels of resistance to amoxicillin, co-amoxiclav, cefuroxime, co-trimoxazole and ciprofloxacin, make these agents less suitable for empirical treatment in serious infections."

2.2 Task definitions

In ISIS-AR resistance is currently monitored by analyzing resistance of isolates against individual antibiotics. For instance, in figure 2.1 resistance levels are shown individually per antibiotic. There is a need by epidemiologists to analyze antibiotic resistance in a broader sense than individual antibiotics, the antibiotic resistance *patterns*. Together with epidemiologists at the Netherlands National Institute for Public Health and the Environment we have defined a number of tasks that need to be performed to be able to explore multi-drug resistance of bacteria. During meetings we have discussed about how they process their data and what they find important, and we questioned what they currently miss and would like to be able to do. Based on this information, we constructed a number of tasks and created a detailed process model of steps they would perform in an ideal situation. A schematic version of this process model is described later. Weekly progress meetings

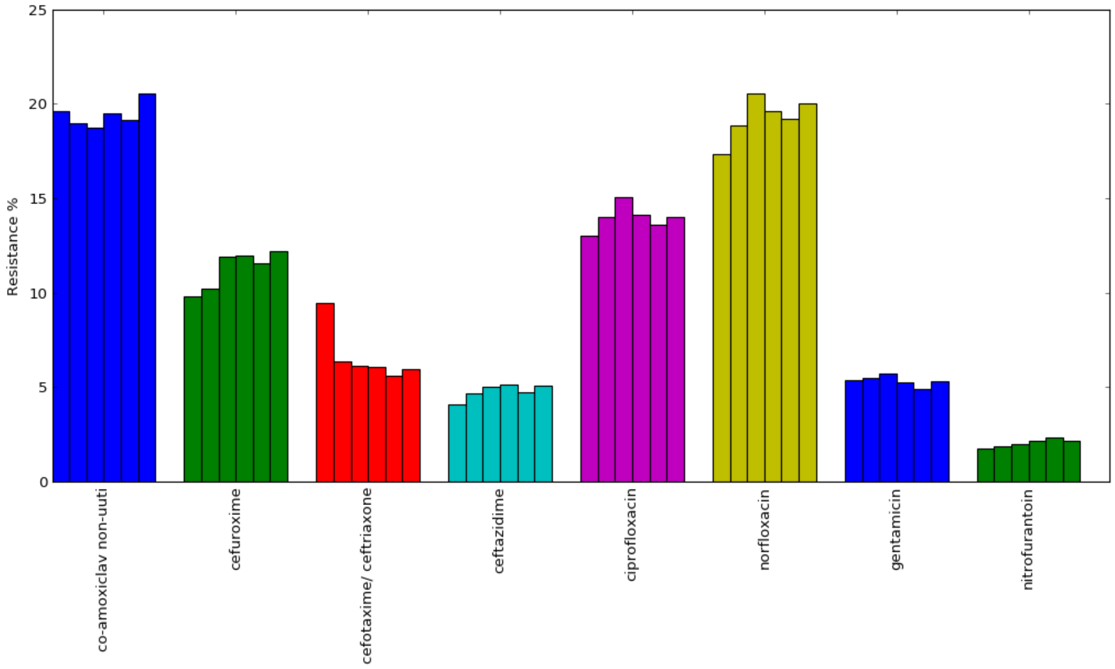


Figure 2.1: Resistance levels of several antibiotics from the dataset from the RIVM used for this thesis. Per antibiotic is shown how resistances have developed over the course of years from 2010 to 2015. This figure is made with Matplotlib [15].

with epidemiologists at RIVM helped steering the result to our goal. In close collaboration we defined a set of tasks:

T1 *Get insight in the most frequent exact resistance patterns and find resistance pattern with resistance against certain antibiotics.*

The most frequent exact resistance patterns should be directly visible. It should be visible to which antibiotics in the patterns the isolates are resistant, and what is the frequency of the patterns. Epidemiologists should be able to analyze resistance patterns more in depth using operators such as sorting and filtering on resistance to antibiotics.

T2 *Get insight in frequently occurring combinations of resistances against antibiotics and find resistance combinations with resistance against certain antibiotics.*

Equivalent to task T1, but for combined resistances.

T3 *Get an overview on which exact patterns data are missing and in what amount, and for which antibiotics data has been imputed.*

It should be directly clear - without user interaction - where data is missing and in what amount. Also, it should be visible for which antibiotics within the pattern there was missing data.

T4 *For a specific resistance pattern get detailed insight in which data was exactly missing.*

The different patterns of missing data within a specific resistance pattern should be shown on demand. It should be clearly visible where data is missing, for which antibiotic, how much was missing, and what values have been imputed. Also, it should be given how much data for each antibiotic was missing for the specific pattern.

T5 *Get insight in time trends of resistance patterns.*

For each of the resistance patterns that are visible on screen, the user should get an impression whether the frequency of the pattern has been decreasing or increasing over the past time, and if it was stable or fluctuating. The trend should be relative to the number of tested isolates.

T6 *Get insight in the distribution of degrees of patterns per antibiotic.*

In case of a multi-drug resistance pattern, the degree of the pattern is the number of antibiotics the pattern is resistant against. Resistance patterns have varying degrees. Epidemiologists should get an impression which antibiotics are more frequently occurring in patterns with high degrees, and which antibiotics are more frequently occurring in patterns with low degrees.

T7 *Get insight in how the attributes (gender, age, human material and laboratories) of a specific pattern are distributed with respect to the average distributions.*

Upon selection of a pattern, there should be the possibility to get an impression of the distribution of the attributes, compared to the average for all patterns together. Attributes should be visualized individually.

T8 *Get insight in the resistance patterns for a specific group of patients, e.g., patients older than 70 years old.*

Epidemiologists should be able to filter isolates based on resistance to multiple antibiotics or on multiple attributes. The visualization should show resistance patterns based on only the isolates in this filter.

T9 *Get insight in the exact isolates with a specific resistance pattern.*

In some cases, epidemiologists need to get a list of the exact isolates of a specific resistance pattern. The list should be accompanied with additional information that does not need to be visualized in other instances, such as the id of the isolate.

T10 *Is a pattern more frequently prevalent at a specific geographic location?*

Some bacterial resistance patterns can be more prevalent among hospitals that are geographically close to each other. It should be possible to visualize postal codes of patients infected by bacteria with a specific resistance pattern, on a map of the Netherlands.

2.3 Requirements

Epidemiologists are scientific researchers working with complex material. A list of tasks is presented in Section 2.2 and the system should be capable of providing epidemiologists the capability of completing these tasks. Since the material itself is already of some complexity and they need to answer highly technical questions, we find it necessary that the view should remain stable and intuitive with a minimum number of colors. Speed is of less importance, correctness of answering questions is much more important. This way, users are more likely to use the system in the future. The test dataset from RIVM contained 188.000 isolates with 5 attributes and 53 antibiotics (about 11M cells). The system should be able to deal with this large amount of data. Lastly, the system should work without any necessary software installation.

2.4 Summary

With specific database queries and statistical software, questions related to the domain of microbial resistances can be answered using the data from ISIS-AR. However, with current methodologies, patterns and trends of patterns are not directly visible. Epidemiologists need to be able to perform the previously listed tasks and an information visualization system, that is bound to a number of technical and design requirements, can aid in this process.

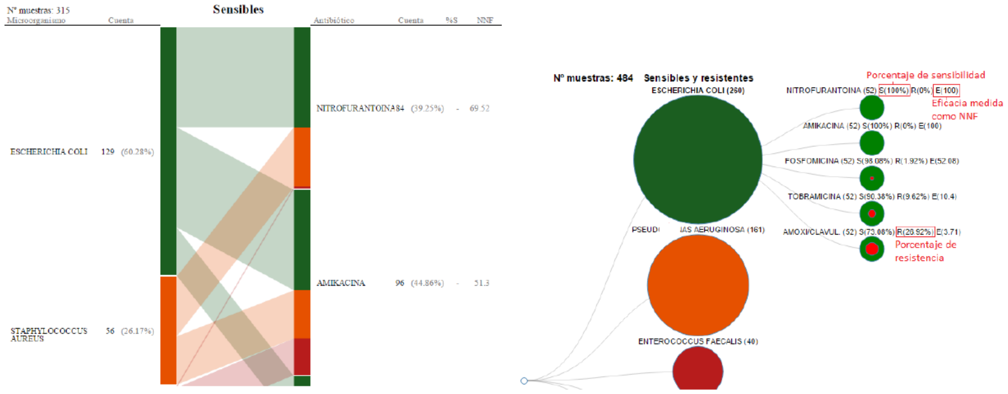
Chapter 3

Related Work

We first briefly explain research on microbial resistance with a strong visualization component. Next, we explain three set-visualization techniques in detail. There are many ways to visualize set-typed data, for a complete overview of set-typed data visualization see the review by Alsallakh et al. [5]. As explained in the Introduction, Euler-/Venn-diagrams, overlays and node-link diagrams are not scalable and scatter plots are not intuitive for set visualization. Matrix-based techniques are fairly scalable and aggregation-based techniques are definitely scalable [5]. We focus here on the approaches most similar to ours. Moreover, we give a short review of visualization techniques that deal with missing data.

3.1 Visualization of Antimicrobial Resistance

Garcia-Caballero et al. designed three visualization techniques to use in a clinical decision support system for empiric treatment [12]. The layouts of these three visualization techniques were: a sunburst layout; a bipartite graph; and a hierarchical tree layout. The bipartite graph and the hierarchical tree layout support most tasks, hence these two are shown here. In Figure 3.1(a) and 3.1(b) bacteria are ordered by their prevalence on the left side. The antibiotics are ordered on efficacy on the right side. In Figure 3.1(a) the bar between bacteria and antibiotics represent the individual efficacy of antibiotic on a bacteria. In Figure 3.1(b) the antibiotics on the second level



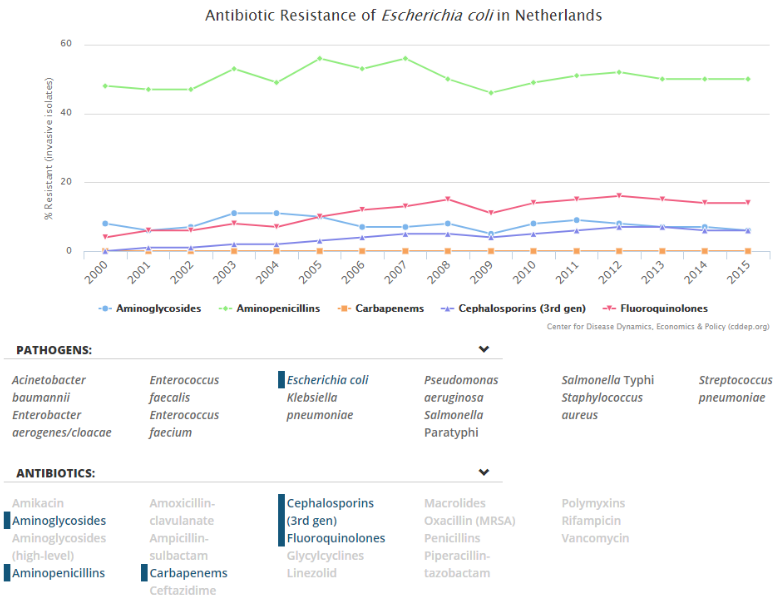
(a) Bipartite

(b) Hierarchical Visualization

Figure 3.1: Here, two visualizations by Garcia-Caballero et al. for a clinical decision support system are shown. In (a) a bipartite graph is shown, with on the left side bacteria and on the right side antibiotics. The bands show the efficacy of the antibiotic on the bacteria. In (b) antibiotics are branched under bacteria in a tree layout.

of the tree are antibiotics to which the parent bacteria is susceptible. The red circles in some antibiotics represent the resistance against the antibiotic.

ResistanceMap is a tool by the Centre for Disease Dynamics, Economics and Policy. They report international data on antimicrobial resistance and usage of antibiotics using chloropleths, trend lines and bar charts [34]. An example of trend lines is shown in 3.2.



PATHOGENS:

Acinetobacter baumannii
Enterobacter aerogenes/cloacae
Enterococcus faecalis
Enterococcus faecium
Escherichia coli
Klebsiella pneumoniae
Pseudomonas aeruginosa
Salmonella Paratyphi
Salmonella Typhi
Staphylococcus aureus
Streptococcus pneumoniae

ANTIBIOTICS:

Amikacin
Aminoglycosides
Aminoglycosides (high-level)
Aminopenicillins
Amoxicillin-clavulanate
Ampicillin-sulbactam
Carbapenems
Ceftazidime
Cephalosporins (3rd gen)
Fluoroquinolones
Glycylcyclines
Linezolid
Macrolides
Oxacillin (MRSA)
Penicillins
Piperacillin-tazobactam
Polymyxins
Rifampicin
Vancomycin

Figure 3.2: This image is from ResistanceMap, a public tool that visualizes bar charts, trend lines and cartographs of microbial resistances. In this case trend lines are shown. For instance, it can be seen that Aminopenicilines have the highest resistance levels. Below the trend lines, filters of pathogens and antibiotics are shown.

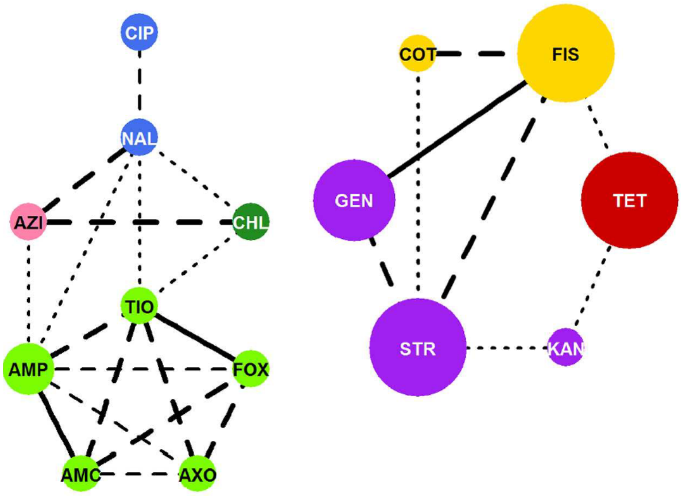
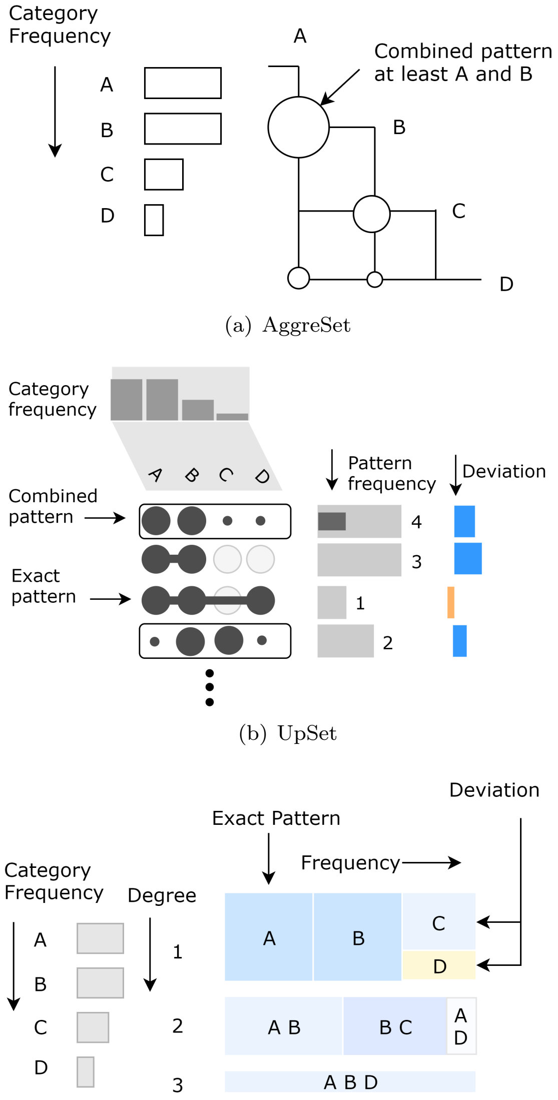


Figure 3.3: This image from Love et al. [22] shows a Markov Network of antibiotics from the year 2012. Sizes of circles represent resistance levels, edges represent magnitude of correlation and colors of circles represent the classification of the antibiotic. For instance, it can be seen that antibiotics of the same class have influence on each other's resistance levels.

Recently, Love et al. showed a new way of looking at antibiotic resistance data using Markov Networks. Whereas government agencies have had their focus on prevalence of multi-drug resistant bacteria, Love et al. propose a different perspective by providing information on the combined distributions of antibiotic resistance. Moreover, they visualized this information using Markov Networks. The Markov Networks show correlations between antibiotics, where the type of edge used determines the magnitude of the correlation. Sizes of antibiotics represent the percentage resistant and colors of antibiotics represent the antibiotics' classifications. Each year is presented individually in a Markov Network. Antibiotics of the same classification are related in the network since they have influence on each other's resistance levels, due to and here we quote "cross-resistance mechanisms, or pleiotropic mutations affecting the common site of multiple drugs" [22]. The edges between antibiotics of different classes are caused by exclusively co-resistance or pleiotropic resistance. An example of such a Markov Network is shown in Figure 3.3. The same information is presented in heatmaps that too show the correlation between pairs of antibiotics over the years using darker colors for higher correlations. However, here the classifications and resistance levels of antibiotics are absent.

Although there are a number of visualization systems that deal with antimicrobial resistance data, none seem to have the goal to visualize clear overviews of the complicated resistance patterns. Combinations of resistances are hardly or not at all discussed or visualized. Here it seems, there is a gap in visualization research.



(a) AggreSet

(b) UpSet

(c) PowerSet

Figure 3.4: AggreSet, UpSet and PowerSet are set visualization techniques. AggreSet visualizes combined patterns, UpSet visualizes combined patterns and exact patterns and PowerSet visualizes exact patterns. Schematic custom-made representations on a sample dataset are shown. The sample dataset used here can be found in the Appendix in table A.1. In (a) AggreSet shows that $A \cap B$ and $B \cap C$ are frequently occurring together, but $A \cap C$ does not occur. Combinations with D are not frequent, since D itself is not frequently occurring. In (b) UpSet shows us $A \cap B$ and $B \cap C$ are frequently occurring. It also shows us that there two exact patterns $A \cap B \cap \overline{C} \cap \overline{D}$ and $A \cap B \cap D \cap \overline{C}$ with a frequency of 3 and 1 respectively. Together they make the frequency of 4 of the combined pattern. In (c) PowerSet shows us the exact patterns grouped by degree. Here too we can find the exact patterns $A \cap B$ and $A \cap B \cap D \cap \overline{C}$. The tiles are marked with a color, which tells us something about whether the frequency of the exact pattern is over- or underrepresented.

3.2 Set Visualization

Alsallakh et al. explain that matrix-based set-visualization techniques are reasonably scalable and favor the fact that no edge crossings occur. Moreover, Yalcin et al. state that matrix-based solutions are scalable and easy-to-read [5, 36], supported by Lex and Gehlenborg [20]. However, the information gain from matrix based techniques is dependent on the ordering of the rows [5]. In this section we discuss three leading set visualization techniques, which are matrix- and aggregation based: AggreSet, Upset and PowerSet. They are presented in Figure 3.4. Here a short summary is given:

- AggreSet visualizes pairs of properties in a triangle-like matrix with circles representing the frequency of the overlap. These pairs are in fact combined patterns of degree two. More frequent properties are higher in the matrix. Attributes are visualized in charts aside.
- PowerSet visualizes exact patterns in juxtaposed tiles constructed by a modified Squarified Treemap algorithm. Colors of tiles represent the level of an attribute or the deviation[1]. The exact patterns are grouped by their degree. Attributes are visualized when hovering over a tile.
- UpSet visualizes exact patterns in a matrix form as rows by default. On demand exact patterns can be aggregated using various metrics for similarity, such as deviation, degree, combined patterns and properties. When aggregated, exact patterns are visually listed under the aggregation. Attributes are depicted row-wise.

AggreSet gives a clear overview of combined pattern sets with a degree of 2, combined patterns with two properties. It is a set-visualization technique designed by Yalcin et al., which tried to give a solution for three common set visualization deficiencies: a low scalability in number of sets, only a small number of supported set operations, and limited attribute visualization [36]. AggreSet focusses on the visualization of intersections of pairs of sets, creating set-pair intersections. These set-pair intersections are shown in a halved adjacency matrix, where the axes denote sets. The two equal axes can be sorted by a set similarity metric or by frequency of the set. The former allows the user to find sets that are highly related close together. The latter allows the user to find combinations that are surprisingly low in frequency together. For instance, in Figure 3.4(a) it is shown that the combined pattern $A \cap C$ has an unexpectedly low frequency compared to their individual frequencies. AggreSet cannot visualize the exact patterns and cannot visualize combined patterns of a degree higher than two. Combinations of patterns with a degree of three can only be inspected by hovering the mouse over a combined pattern of degree two and the circles of other combinations of two will then be partially filled. An example of this effect is show in Figure 3.5.

Furthermore, AggreSet tries to visualize the distribution of degrees of all combined patterns in the data is shown via a bar chart. Each bar in the bar chart represents a degree and the height of the bar represents the number of elements in patterns with that specific degree. An example is given in Figure 3.6.

UpSet by Lex et al. visualizes exact patterns using a matrix format showing patterns versus properties [21]. Moreover, the exact patterns can be aggregated on two levels with multiple choices: by combinations of properties with a specific degree, by set-degree and by deviation. When such an aggregation is done, the exact patterns that are part of the combined patterns are displayed hierarchically under the combined pattern. This shows clearly which exact patterns belong to a combined pattern. Moreover, aggregation by deviation splits the patterns in two groups: those that are more frequent than expected, and those that are less frequent than expected. UpSet visualizes combined patterns of all possible combinations of properties. However, only patterns with the same, user-defined, degree are shown. The patterns can be filtered and sorted and are provided

[1] the word *deviation* describes the difference between the expected frequency and the actual frequency of a pattern. The actual frequency of a pattern can be lower or higher than the expected frequency of the pattern based on the dataset. The deviation can be a complex formula such as in PowerSet [3].

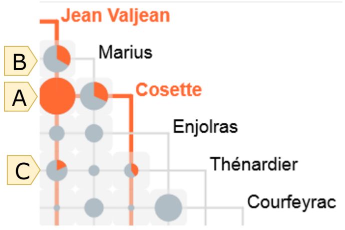


Figure 3.5: Here an example is shown on the *Les Misérables* dataset by Knuth [17]. This dataset has a list of co-occurrences of personages in the story of *Les Misérables* and the related volume, book and chapter in which the assemblence took place. AggreSet has visualized this dataset. Hovering over the combined pattern Jean Valjean ∩ Cosette (**A**) shows us that in this case Marius is relatively more frequently occurring with Jean Valjean (**B**) than Thénardier is with Jean Valjean (**C**) by the filled percentage of the circle.

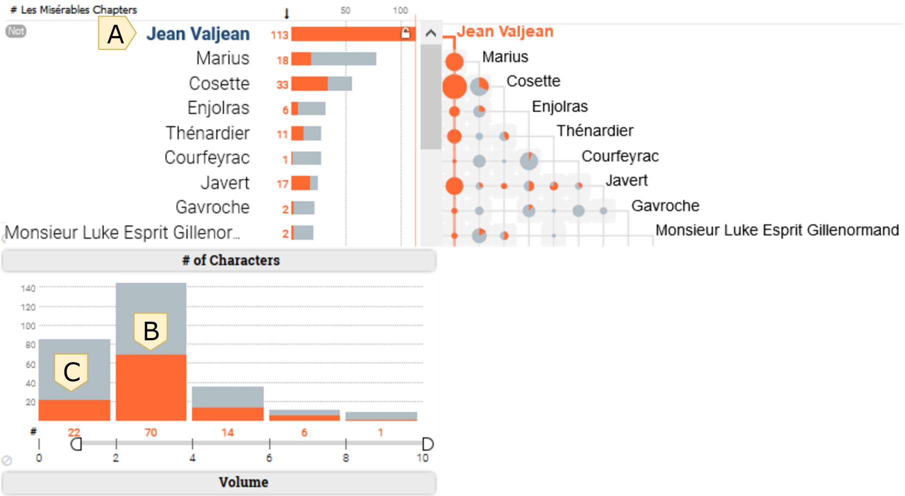


Figure 3.6: In this image the effect is shown of hovering with the mouse over AggreSet's *Jean Valjean* and its effect on the degree sharing bar chart. In this case we hover over *Jean Valjean* (**A**). This means that only chapters are regarded in which *Jean Valjean* is present. We can see how other views are updated with orange fills. For instance, in 70 chapters *Jean Valjean* is with 1-3 other persons (**B**), whilst in 22 chapters *Jean Valjean* is alone (**C**).

with a deviation metric to find unexpected patterns. Filters and sorts can only be performed on the exact patterns, and not on the combined patterns. Hence, the combined patterns cannot be sorted by frequency and cannot be filtered. Attribute distributions are visualized using boxplots for both combined patterns and exact patterns. For the distribution of degrees of properties in exact patterns, first by aggregation on sets and subsequently by degree is necessary.

PowerSet by Alsallakh and Ren is a set visualization technique that visualizes sizes of exact patterns and is built on top of UpSet [3]. The area of a tile represents the frequency of an exact pattern and labels on the larger tiles show which properties of the exact pattern it includes. A schematic example is shown in Figure 3.4(b). Hovering over a rectangle enlarges it, and shows the labels which were hidden before if the tile was small, and shows the summaries of attributes. Tiles are grouped vertically by degree into distinct regions (or rows) and each region is ordered horizontally by frequency by default. Other possible orderings are by deviation, properties or selection. The height of a complete region is proportional to the number of elements it comprises and regions can be collapsed. Exact patterns can be filtered by selecting multiple properties from the *set view*, which is a vertical list of all properties and their frequencies. Elements can be selected and their attributes can be visualized by clicking on a tile. PowerSet also enables finding association rules by requesting the right hand side of rules as input and visualizing all the possible left hand sides in association rules and their confidences. To new users it may not be directly clear whether combined patterns or exact patterns are visualized in PowerSet. However, PowerSet lacks the ability to visualize combined patterns [3]. Moreover, small tiles have no labels and therefore it is not immediately clear what the small tiles should represent. Lastly, we cannot "turn off" regions, which makes it harder to compare frequencies of exact patterns irrespective of their degree. For instance, in Figure 3.4(b) it is unclear whether B ∩ C or A ∩ B ∩ D is more prevalent. According to Munzner [25], using areas in the quantitative channel is unfavorable.

There are many visualization techniques that have succeeded in visualizing set-typed data. Three recent and powerful set-visualization techniques with attributes visualization have been discussed in detail. Linked highlighting has been used extensively showing relations between numerous objects and deviation metrics intuitively show unexpectedly low frequencies of patterns. Characteristically all three techniques avoid clutter of information, less important information is chosen to be hidden. However, these visualization techniques also have drawbacks. For instance, some do not clearly visualize exact patterns, others lack an overview of the combined patterns.

3.3 Visualization of Missing Data

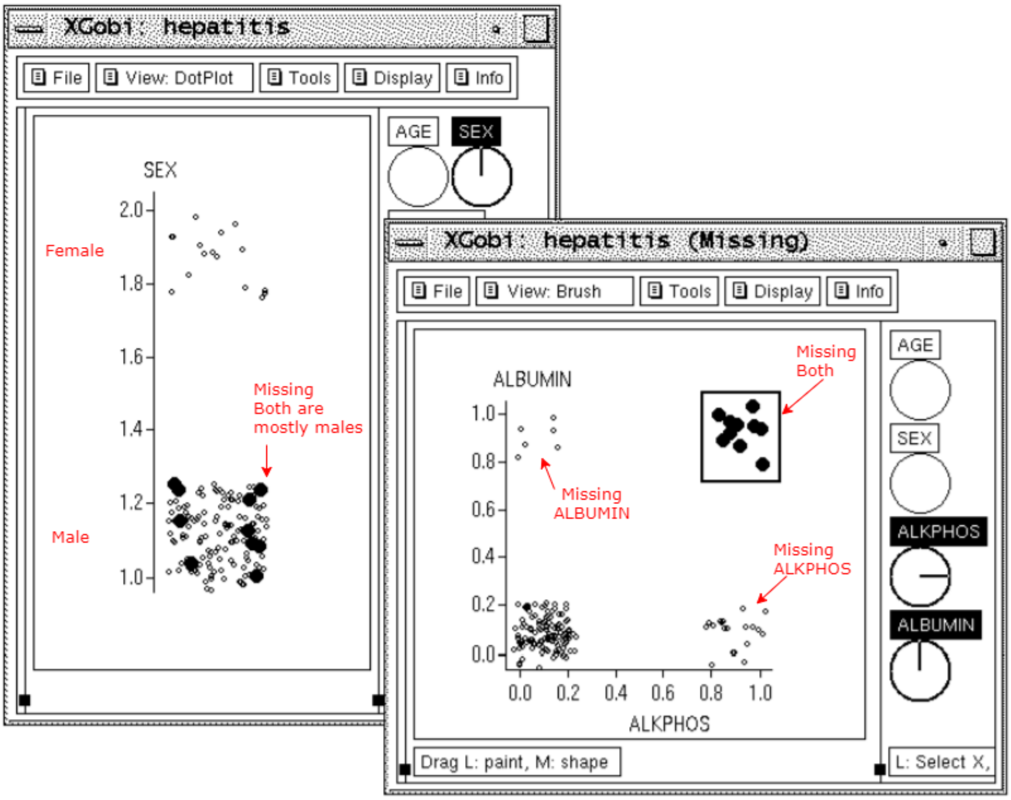


Figure 3.7: This image is from [33] and is adjusted with contextual information. It shows that there are six values missing for the variable *ALBUMIN* and there are more values missing for the variable *ALKPHOS*. The square in the top right corner is brushed. The brushed values are also brushed in the other diagram, clearly showing that missing values for both variables are typically men.

Little research has been done on visualization of datasets with missing data. Swayne and Buja

[33] visualized their multivariate datasets containing missing data with the intention to clarify which values were missing and their correlations with other variables. Furthermore, they also show imputed data to enable users to form an opinion of the quality of imputation on the dataset. Swayne and Buja visualized the effect of imputation of data using scatter plots. The easiest solution to clarifying missing values and correlations, was to impute the missing values to values of 20% below the minimum value of a variable. Consider the case of two variables where data values can contain missing values in both variables. This produced in a scatter plot two lines of missing data points, one horizontal and one vertical which show which points were missing, and one cluster in the intersection of these lines. This simplistic imputation is to explicitly show the user which original data points had missing data. An example is shown in Figure 3.7. However, this technique becomes more complex for multivariate data. Advanced imputation techniques have two problems according to Swayne and Buja: The original missing values are obfuscated and the user may wrongly believe the data he sees is correct. Hence, they provide three solutions to solve these problems. The simplest solution is an imputation scheme with fixed imputation values. The other two options are to show the location of missing values, and to make use of precomputed imputations, based on the data, using a more or less sophisticated algorithm. The user can choose with imputation scheme he would like to see or both.

Other visualization of missing data has been performed by Arbesser et al. [6]. Their objective was to design a visualization system, called Visplause, that allows for inspection of data quality and data problems. Hence, missing data, improbable values or values that violated specific constraints are visualized in detail. Moreover, overall data quality is reported on several scales in one view and reported in several time partitions in another view. However, Visplause does not impute missing data and hence only shows the location of missing data. An unmodified example image is currently not publicly available, hence the absence of one.

The discussed techniques have shown ways to visualize missing data. Missing data can be visualized per element, showing the original (missing) location of values and their imputed counterparts, or aggregated using overviews. When data is temporal, missing data be shown partitioned over time as well. Although there is not so much research on visualization of missing data, the discussed techniques provide complete and clear insights in how to tackle the problem om missing data. However, these techniques tend to aim for visualizing relations between missing data and variables, which is not the objective for epidemiologists. Epidemiologists are interested in the data visualization itself, and rely on the imputation of the data that it is correct. Imputation is necessary to reduce the number of patterns. If data was imputed they need to be able to find out if this imputation is trustworthy. Showing what values were missing and have been imputed can help to this end.

Chapter 4

JuxtaSet Technique

JuxtaSet is a visualization technique driven by the request from epidemiologists to gain more insight in multi-drug resistant bacteria. Ten tasks inherent to the problem statement were explained in Section 2.2. Formal definitions are given on the necessary data transformations to enable visualizations to complete the tasks. Moreover, the data format that JuxtaSet can import is explained. From this point on, JuxtaSet's visualization of combined patterns and exact patterns, how it deals with missing data and how it visualizes temporal variation are explained in detail. Lastly, JuxtaSet's exploration pipeline is discussed including a schematic overview of the layout and process model of exploration. Moreover, in this section filtering and visualization of attributes are described. The sections are accompanied with examples and models.

4.1 Definitions

In this section the mathematical background behind JuxtaSet is described. We start with formal descriptions of the exact and combined patterns. Next, we describe degrees in mathematical sense, a necessity to one of the epidemiologists. Lastly, we describe missing data and sparklines.

4.1.1 Exact and Combined Patterns

Formally, frequent itemsets have been described by Agrawal et al., where $\mathcal{Q}$ is the set of n properties:

$$\mathcal{Q} = \{p_1, p_2, ..., p_n\}. \tag{4.1}$$

We have a set of m elements E:

$$E = \{e_1, e_2, ..., e_m\}, \tag{4.2}$$

Elements and properties are connected by a label a_j with $a_j \in A$. In set visualization, $A = \{1, 0\}$ would be used to indicate whether an element belongs to a set. From this perspective, sets and properties are similar. Likewise, sets can be described as binary categorical variables [11]. However, we deviate from standard set visualization mathematics. Figure a binary matrix where sets are on columns and elements are on rows. Each cell indicates whether the element belongs to the set. Where set visualization describe relations on this matrix using a *vertical* perspective, we describe these relations using a *horizontal* perspective. We define a mapping which gives us for an element and a property their associated label:

$$v : E \times \mathcal{Q} \to A. \tag{4.3}$$

Using the powerset $\mathcal{P}$, we get the set of all possible subsets of the set of properties $\mathcal{P}(\mathcal{Q})$. Given a subset of properties C, thus $C \in \mathcal{Q}$, the set of elements that have a label a for all in C is given by the mapping $\mathcal{H}_C : \mathcal{P}(\mathcal{Q}) \times A \rightarrow \mathcal{P}(E)$:

$$\boxed{\mathcal{H}_C(C, a) = \{e \in E \mid \forall_{p_i \in C}[v(e, p_i) = a]\}} \tag{4.4}$$

The definition of frequent itemsets is defined by Leskovec et al. [19] with minimal support s as:

$$C_F = \{C \in \mathcal{P}(\mathcal{Q}) \mid |\mathcal{H}_C(C, 1)| \geq s\}. \tag{4.5}$$

We are interested in the patterns that occur at least once, thus with $s = 1$. These we call the combined patterns P_C:

$$P_C = \{C \in \mathcal{P}(\mathcal{Q}) \mid |\mathcal{H}_C(C, 1)| \geq 1\}. \tag{4.6}$$

Like the combined patterns, we are interested in the exact patterns. Exact patterns are the set of elements which have label 1 for properties C and label 0 for properties $\mathcal{Q} - C$. Therefore, we define a mapping that gives the set of elements that satisfy this property, $H_E : \mathcal{P}(\mathcal{Q}) \rightarrow \mathcal{P}(E)$

$$\boxed{\mathcal{H}_E(C) = \mathcal{H}_C(C, 1) \cap \mathcal{H}_C(\mathcal{Q} - C, 0)} \tag{4.7}$$

The exact patterns can then be defined with the equation:

$$P_E = \{C \in \mathcal{P}(\mathcal{Q}) \mid |\mathcal{H}_E(C)| \geq 1\}. \tag{4.8}$$

4.1.2 Degrees

In order to complete task T6, it is necessary to compute degrees of exact patterns. The degree of a pattern C, with $C \in P_E$, is defined with the mapping $d : P_E \rightarrow \mathbb{N}$:

$$d(C) = |C| \tag{4.9}$$

However, it is necessary to gain insight in the distribution of degrees per property. For this, all exact patterns are needed, and it needs to be checked for each degree k, which exact patterns have $d(C) = k$ and compute its number of elements. Hence, we define a mapping that takes a property and a degree and gives the number of elements, $d_E : \mathcal{Q} \times \mathbb{N} \rightarrow \mathbb{N}$:

$$d_E(p, k) = \{|C| \mid \exists C \in P_E(p \in C \land d(C) = k)\} \tag{4.10}$$

This way we can create a table that gives the distribution of degrees per property. Such a table is shown in Table 4.1.

Table 4.1: Distribution of Degrees

	1	2	...
p_1	$d_E(p_1, 1)$	$d_E(p_1, 2)$	...
p_2	$d_E(p_2, 1)$	$d_E(p_2, 2)$	...
...	...	...	...

4.1.3 Missing Data

The data contains missing values and these missing values have been imputed. For each element it is registered using a mapping $u : E \times \mathcal{Q} \rightarrow A$ if data has been imputed for an element and property.

Missing Data Aggregated For Exact Pattern and Property

In the listed tasks in 2.2, it was described that information on missing data need to be retrieved per pattern aggregated and in detail. To get all the elements where data was missing for one specific property and exact pattern, we define a mapping $M_P : P_E \times \mathcal{Q} \rightarrow \mathcal{P}(E)$:

$$M_P(C, p_i) = \{e \in \mathcal{H}_E(C) \mid u(e, p_i) = 1\} \tag{4.11}$$

Missing Data in Detail For Exact Pattern

To compute the missing data patterns for a specific exact pattern, we need to be able to get the missing data for one specific element, and for which properties of this element data was imputed. Using the mapping u and a new mapping $M_F : E \to \mathcal{P}(\mathcal{Q})$, all properties can be computed where data was missing for an element:

$$M_F(e) = \{p_i \in \mathcal{Q} \,|\, u(e, p_i) = 1\} \tag{4.12}$$

Given we have a subset of elements E_s, where $E_s \in E$ and we would like to know for which properties in this set of elements, data was imputed. Hence, we define a mapping $M_A : \mathcal{P}(E) \to \mathcal{P}(\mathcal{P}(\mathcal{Q}))$:

$$M_A(E_s) = \{M_F(e) \,|\, e \in E_s\} \tag{4.13}$$

Using this equation we can get for any exact pattern $C \in P_E$, the detailed missing data patterns with $\boxed{M_A(\mathcal{H}_E(C))}$.

Example Let $\mathcal{Q} = \{A, B, C\}$, $E = \{e_1, e_2, e_3, e_4\}$ and $v(e_1, A) = 1$, $v(e_1, B) = 1$, $v(e_2, A) = 1$, $v(e_3, A) = 1$, $v(e_3, B) = 1$, $v(e_4, C) = 1$, other combinations are 0. Also, let $u(e_3, B) = 1$. We can compute $P_C = \{\emptyset, \{A\}, \{B\}, \{C\}, \{A, B\}\}$, $P_E = \{\{A\}, \{A, B\}, \{C\}\}$, $d_E(A, 1) = 1$, $d_E(A, 2) = 2$, $d_E(A, 3) = 0$, $d_E(B, 1) = 0$, $d_E(B, 2) = 2$, $d_E(B, 3) = 0$, $d_E(C, 1) = 1$, $d_E(C, 2) = 0$, $d_E(C, 3) = 0$. $M_P(\{C\}, C) = \{e_3\}$, $M_A(\mathcal{H}_E(\{C\}) = \{\emptyset, \{B\}\}$.

4.2 Data Definitions

JuxtaSet allows the possibility to import multiple datasets. These datasets are stored on the server. If there is for instance a dataset from a specific environment, epidemiologists might want to import this dataset exclusively from other datasets. Also, using smaller distinct datasets speeds up the application. There are a number of constraints on datasets imported into JuxtaSet. The data that can be imported into JuxtaSet is a CSV file with a form like in Table 4.2. Each element has attributes, an optional temporal variable, and information over properties. Attributes can be one of three data types:

- *numerical*: For instance age. Numerical attribute levels can be binned. The user is asked how many bins he would like. Eventually, the attributes are visualized in charts and binned data helps to create histograms.
- *nominal*: For instance hospital id's or gender. There exist a constraint amount of nominal values for the attribute. These attributes are visualized in bar charts. For instance, it can be visualized how many isolates are from each hospital.
- *strings*: For instance, patient id's. Attributes of type *string* are not visualized and are only helpful to get extra information about elements when elements are requested like in task T9. For instance, the isolate *id* from the original database, which helps the researchers quickly look up the isolate in their own database.

When importing a dataset, the user is asked for each attribute which data type it is. All values of an attribute must be of the same data type chosen for the attribute.

The dataset may contain a temporal variable, which is required to do task T5. Values in this column must be of type *string*. An important property that must hold for the temporal variable: sorting the temporal values string-wise must result in a list of values in chronological order. This holds for example for dates in American format *YYYY-MM-DD*, which are in chronological order if sorted string-wise.

Lastly, the dataset contains m properties. Possible values are 0, 1 and ε. The ε is an empty string in the CSV file and tells us data was missing for the cell.

Table 4.2: CSV format of file that can be imported into JuxtaSet.

attr 1	**..**	**attr n**	**temporal**	**cat 1**	**..**	**cat n**
num/nom/str	..	num/nom/str	str	0, 1, ε	..	0, 1, ε
num/nom/str	..	num/nom/str	str	0, 1, ε	..	0, 1, ε

4.3 Combined Patterns and Exact Patterns

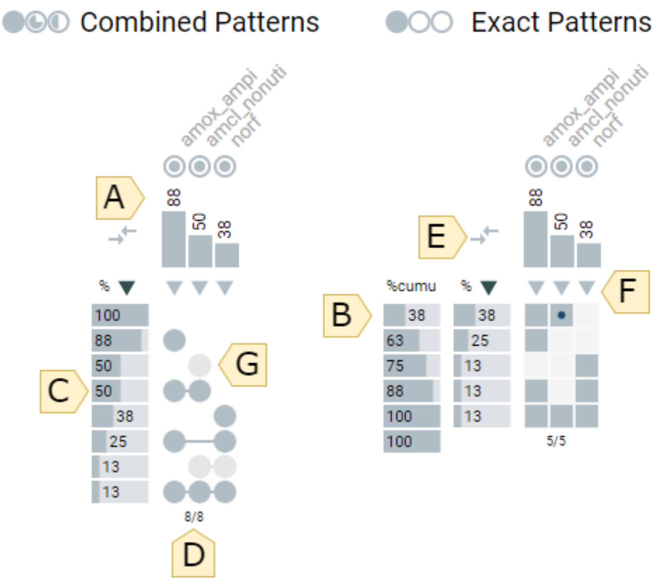


Figure 4.1: JuxtaSet's visualization of combined and exact patterns. We explain its core features using a sample dataset of fake isolates. In (A) it can be seen that 88% of the isolates is resistant against amox-ampi, and that in (B) that 38% of the isolates were exactly and only resistant against amox-ampi and amcl-nonuti. However, 50% of the isolates were resistant against at least amox-ampi and amcl-nonuti (C). In (D) it can be seen that there are 8 different combined patterns. The button next to (E) is used to sort antibiotics by frequency or name horizontally. The buttons next to (F) allow to sort the combined and exact patterns vertically. All isolates resistant against amcl-nonuti, are resistant against amox-ampi as well (G). This is indicated using a special color for these combined patterns.

The core of JuxtaSet is the presentation of combined and exact patterns. An example on a fake dataset is shown in Figure 4.1. First, we describe the core features of JuxtaSet's visualization of combined and exact patterns. Then, we describe the process of steps JuxtaSet takes to visualize its data.

4.3.1 Core Structure

JuxtaSet visualizes combined and exact patterns using matrices that very much look alike. Patterns are visualized row-wise and properties are visualized column-wise. Exact patterns are visualized characteristically using squares, whereas combined patterns are visualized using circles. This way these views are easily recognized and not mistaken. Moreover, lines between circles are drawn to make it easier to see which circles belong to the same combined pattern, and is from UpSet by Lex et al. [21]. In exact patterns, dark squares are used for containment of the property, and light squares are used for to indicate these elements did not contain the corresponding property.

Frequencies in percentage of patterns are visualized using horizontal bars with the percentage displayed. Patterns are sorted by frequency, with the most important patterns on top. For exact patterns, the cumulative percentage is also depicted using horizontal bars. Likewise, frequencies of properties are visualized using vertical bars. The columns are horizontally sorted by frequency, such that the most important properties are on the left side. Furthermore, when there are more than 30 patterns, scrollbars can be used to scroll to patterns with lower frequencies.

4.3.2 Other Features

Missing data is visualized on patterns and will be described in Section 4.4.4. Moreover, if the dataset is time-dependent, trends of patterns are visualized using sparklines. This is described in Section 4.5. Patterns can be sorted, filtered, locked and hovered. For instance, hovering over a pattern gives the absolute number and the distributions of the attributes of elements aggregated under the pattern. More detail on these features is described in Section 4.6.

4.3.3 Process of Imputation and Aggregation

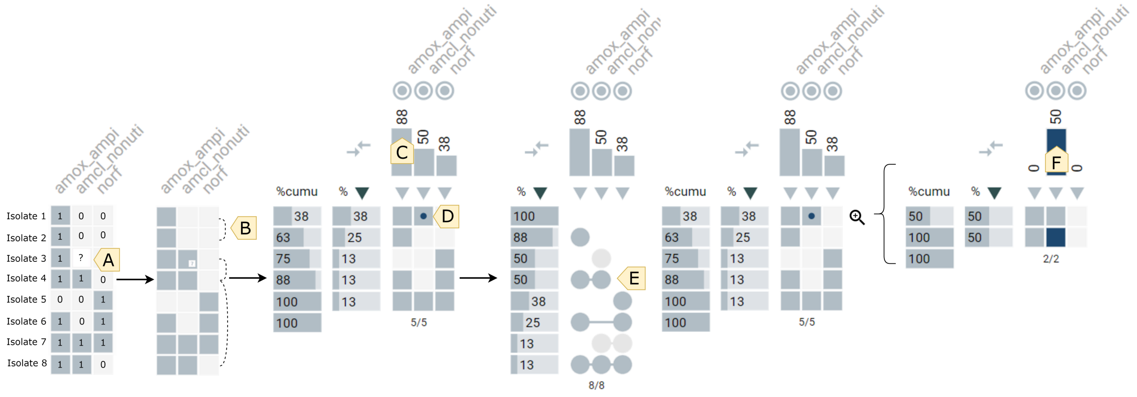


Figure 4.2: In this figure it is shown how JuxtaSet analyses a dataset, and how the patterns, frequencies and missing data are visualized. We start on the left side with a matrix of zeros, ones and missing data (A). Missing values are imputed, but it is kept track of which cells had missing data. Next, rows are aggregated that are exactly the same (B). JuxtaSet visualizes these patterns, with frequencies and missing data. For instance, like in Figure 4.1 in 88% of the elements there was resistance for amox-ampi (C). The top row has imputed data for amcl-nonuti (D). The combinations of patterns are shown and it can be seen that the combination of amox-ampi ∩ amcl-nonuti has a frequency of 50% (E), which is different from the exact pattern amox-ampi ∩ amcl-nonuti∩$\overline{\text{norf}}$. When zoomed into the pattern with missing data, it can be seen that it consists of two different aggregated patterns: one without missing data, and one with missing data. 50% of the elements with the zoomed in pattern has missing data (F).

In Figure 4.2 an example is shown of the process of steps that JuxtaSet undergoes when visualizing a dataset. First, when a dataset is imported, elements with missing data are imputed and it is registered which values have been imputed. To get insight in the most frequent exact resistance patterns, elements are aggregated which have exactly the same properties. To compute exact patterns, equation 4.8 is used. An example is shown in Figure 4.2, where in **B** two rows with exactly the same properties are aggregated. Similarly, using equation 4.6 all combined patterns are computed, a necessity for task T2 (*Get insight in frequently occurring combinations of resistances against antibiotics and find resistance combinations with resistance against certain antibiotics.*). Next, the combined patterns are visualized by JuxtaSet. JuxtaSet constructs a third matrix with detailed patterns of missing data when an exact pattern is selected.

4.4 Dealing with Missing Data

In a real life scenario, datasets are incomplete and for some elements it is unknown whether it has certain properties. There are three different possible scenarios per property of an element: we know an element has the property, we know it does not have the property or it is unknown. In a setting of antibiotic resistance this means that the isolate is resistant against an antibiotic, not resistant, or it is unknown. These unknown values result in patterns that otherwise would have been aggregated being displayed separately, since only elements with exactly the same cells can be aggregated. With the use of imputation, the number of patterns can be decreased. However, it is necessary to provide the epidemiologists insight in which data has been imputed and what the results were, such that they can feel confident on the conclusions drawn from the visualization. There are two tasks related to missing data, T3 and T4, that require to obtain information on missing data, on an abstract scale and on a detailed scale respectively.

4.4.1 Comparison Imputation Algorithms

To obtain insight in the efficacy and efficiency of different imputation methods and to test which imputation algorithm fits best to the antibiotic resistance dataset, we have performed an experiment. A generated test-dataset and a complete-cases real dataset are used (more detail in Section 4.4.3), varying amounts of data were removed, and it was studied to what extent the following four algorithms were able to impute the missing data:

- MEAN : replace missing value with *non-susceptible* if the antibiotic is more *non-susceptible* than it is *susceptible*. No antibiotic has a higher resistance than 50%, thus this essentially means that all missing values are replaced with *susceptible* [31].
- MICE: perform multiple imputation by chained equations. Impute a number of times, analyze their results and integrate these results back into the final result. Repeat the last two steps a number of times [9].
- SOFT-IMPUTE: a scalable imputation algorithm that for all columns replaces missing values repeatedly with an estimated guess and then updates the guess till it converges [23].
- KNN: replaces missing values with values from similar rows [7].

In Figure 4.3 the performances of four common imputation methods on both datasets is shown and compared to random imputation. Currently, in JuxtaSet, SOFT-IMPUTE is used. This algorithm is designed by Mazumder et al. [24] and implemented in Python by Rubinsteyn and Feldman [31]. Since the data comprises of a matrix of about 9 million cells and this method has the most scalable implementation in Python. Imputation methods with a lower error rate like MICE by Buuren and Groothuis-Oudshoorn [9] would be preferable, but also would require a substantial implementation effort, which was outside the scope of this project.

4.4.2 Cut-off Point

With an increasing percentage of missing data, imputation becomes less reliable. The chart in Figure 4.3 can help a modeller to define a cut-off point, a maximally acceptable percentage of missing data among cells that belong to a specific property. If the percentage of missing data in cells for the property is higher than this cut-off point, then no cell in this property is imputed. These values remain to be missing values and are visualized as white space in exact patterns. In JuxtaSet this cut-off point is by default 20%.

4.4.3 Data Simulation

We next describe in more detail how the test data were generated, a mask of missing values is generated and how performance of imputation is measured. In order to ascertain the performance of imputation algorithms on the isolates dataset, there are two possibilities for the initial dataset: use records from the isolates dataset with complete cases and in these cases randomly create missing data, or simulate a dataset comparable to the isolates dataset and randomly create missing data. The former may introduce bias, since records with complete cases may be much different from incomplete records, and the latter generates a dataset that may not fully represent the real dataset. We have chosen to do both, generate a dataset and use complete cases from the real dataset. Ultimately the implementation, configuration and validation of the imputation algorithm should be done in cooperation with a statistician. Here a proof of concept is presented.

Test Data Generation

Since some antibiotics have much higher resistance among isolates than other antibiotics, a uniform matrix of zeros (resistant) and ones (not-resistant) is not representative. A non-uniform Boolean matrix A of given size (n, m) was generated. Each cell in A was defined using Equation 4.14 with X being a uniform distributed random value between 0 and 1, k a constant parameter that determines the probability of True for the first column, and v a constant parameter that determines the slope.

$$A_{i,j} = \begin{cases} 1, & \text{if } X \geq \frac{j}{v(m-1)} + k \\ 0, & \text{otherwise} \end{cases} \tag{4.14}$$

A must be a matrix, since columns and rows need to represent antibiotics and isolates respectively. Imputation methods regard columns as variables and rows as records [31]. The values of the parameters in our simulations were $n = 300$, $m = 8$, $v = 2$, $k = 0.4$. Since some resistance patterns have a much higher frequency than other resistance patterns, each row in the matrix A was copied a number of times Z_i, where Z was a vector of n random values distributed according to the Zipf-distribution with parameter 2. This resulted in a new matrix A_z of size $(n \sum_{i}^{n} Z_i, m)$, simplified as (n_z, m). On the matrix A_z a mask of missing data is applied, which is described in the next paragraph.

Mask of Missing Data

Since we had chosen to do both a data simulation as well as using complete cases from real data, A_z was either a generated dataset or real data. On A_z we applied a mask of missing data. However, since some antibiotics resistances are hardly ever measured in laboratories, others are almost always measured, and others are in between, data is not missing completely at random. To incorporate this, a vector M of random values distributed according to the Zipf-distribution with parameter 2 was used and this vector was multiplied with a given fraction of missing data p and the number of rows n_z. This resulted in a vector that gave the number of cells with missing data for each column. Finally, this vector was applied by deleting M_j values on random positions in the matrix A_z, resulting in a matrix with missing values A_m.

Imputation Performance

For both the generated data as well as the real dataset, we apply each imputation method on A_m and compare the result to A_z. The percentages of incorrectly imputed values are computed. The whole process of generating or loading a dataset, applying a mask of missing data, impute the missing data and compute its performance is repeated 30 times for each imputation method in order to get more reliable results. The performance results per imputation method are depicted in Figure 4.3. From this figure it can be analyzed that imputation methods have more difficulty with imputing the generated data than with the complete cases from the real dataset. A possible reason is that the real dataset had more similar rows and thus were easier to impute, whilst the random generated dataset had more varying rows. Moreover, MICE has the highest performance for both datasets.

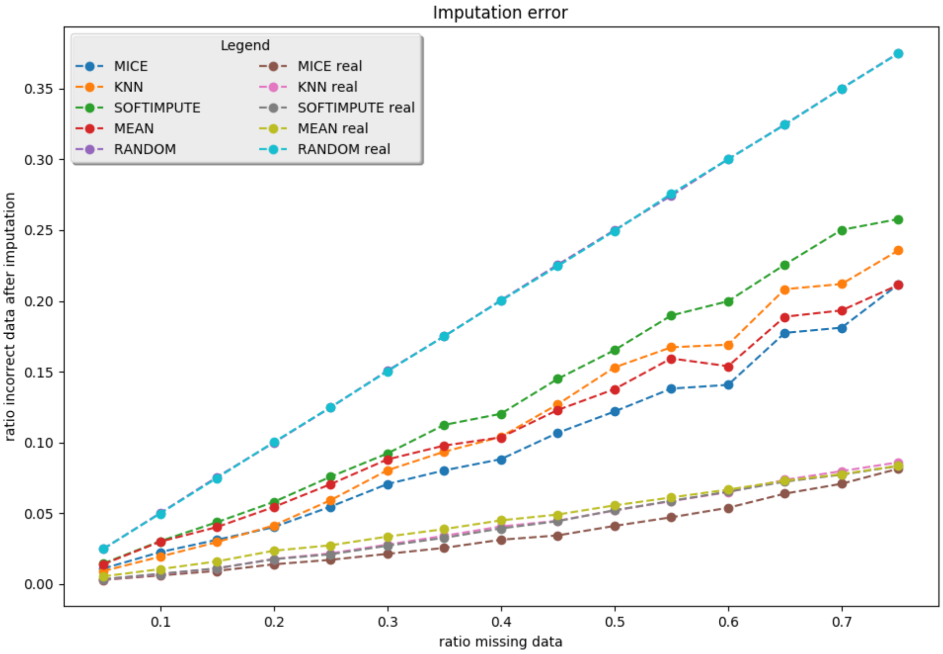


Figure 4.3: Performance of common imputation methods on simulated data. MICE performs best for higher ratios of missing data and SOFT-IMPUTE performs the worst. Random imputation has on average $p/2$ incorrect missing data in its end result. This figure is created with Matplotlib by Hunter [15].

4.4.4 Visualization in JuxtaSet

For the visualization in JuxtaSet the missing cells of the dataset are imputed using SOFT-IMPUTE, and it is kept track which cells originally had missing data. Next, the imputed elements are aggregated and shown in the exact patterns view as explained in Section 4.3. Also the patterns where originally data was missing are aggregated (using the tracked cells) and these patterns are assigned to the imputed pattern.

Missing Data Aggregated On Exact Patterns

Epidemiologists are interested in the amounts of originally missing data for each cell in the pattern and these are computed using the assigned patterns. Missing data is visualized in an aggregated form on top of the exact patterns, such that users of JuxtaSet can see which patterns and which properties have a lot of missing data. Missing data is indicated using circles on top of cells, where the area of the circle shows the amount of missing data. No circle is shown if there was no missing data. An example of how missing data aggregated on exact patterns is visualized in JuxtaSet, for a number of patterns is shown in Figure 1.1. In Figure 4.2, the highest exact pattern with label D has a blue dot for the antibiotic amcl-nonuti. The missing data is from the data element with a question mark and was imputed to *resistant*.

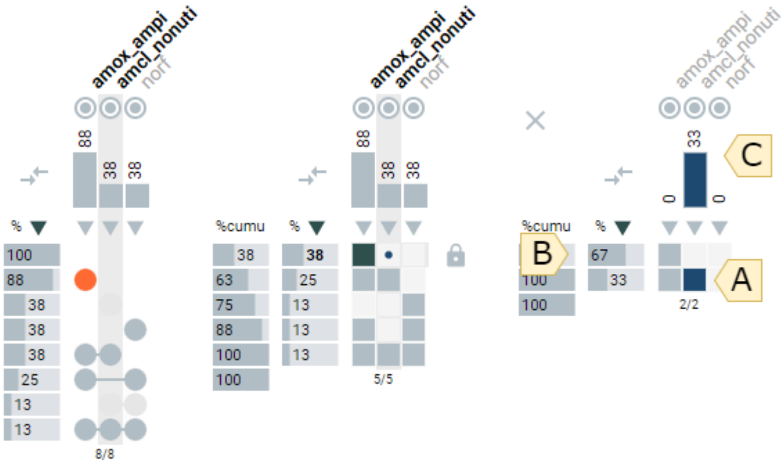


Figure 4.4: JuxtaSet visualization of visualization of missing data in detail for an exact pattern. The middle matrix show the exact patterns, the left matrix shows the combined patterns and the right matrix shows which exact different patterns exist for the selected row. Do notice this is not the same data as in Figure 4.2. Here, one patient is added to show that missing data rows are aggregated in this view as well. It can be seen that the missing data (A) has been imputed to not-resistant. There were three patients in the selected exact pattern. 2 out of 3 patients, 67% (B), had no missing data and one patient had missing data. The blue vertical bar indicates that 33% of the patients had missing data for the antibiotic amcl-nonuti (C).

Missing Data in Detail for Exact Pattern

Eventually, when the epidemiologist is interested in the missing data within one specific exact pattern, the assigned missing data patterns are shown individually. Elements with exactly the same missing data are still aggregated. Instead of grey bars that that show the percentage of each property, now blue vertical bars indicate the percentage of elements where data was missing for the selected pattern. Moreover, missing data in patterns are indicated with blue squares ■. Clicking an exact pattern results in visualization of the distinct patterns with missing data. An example is shown in Figure 4.4 where two distinct patterns are shown. In Figure 4.5, an example is displayed of a larger dataset. Here, the blue columns on top show the percentage that was missing for a selected resistance pattern (not visible here). For instance, 8% of the isolates was imputed to susceptible for the antibiotic *cefepine*.

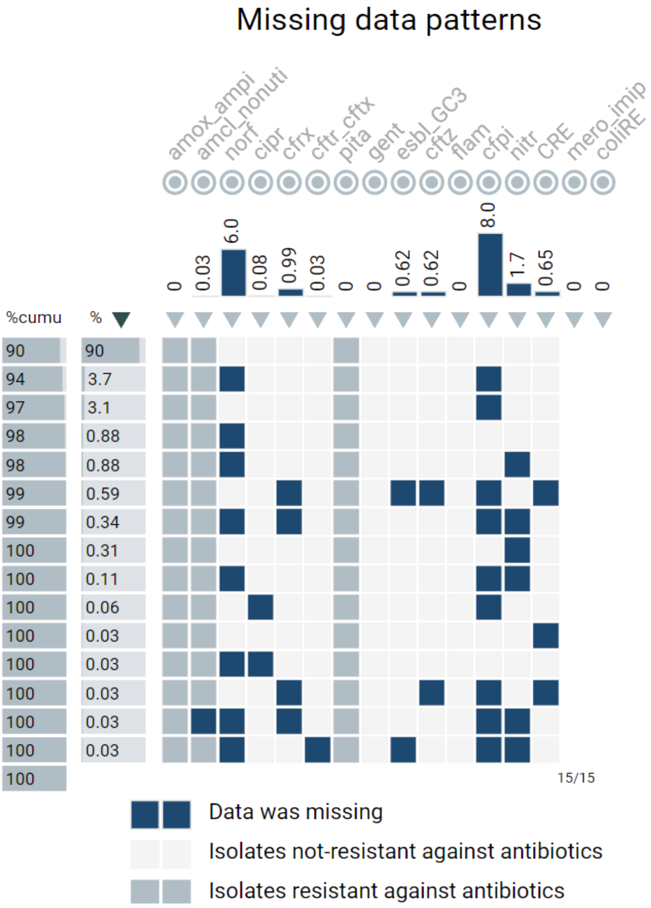


Figure 4.5: For a specific exact pattern, it is shown where data was missing and thus has been evidently imputed. In this case for 8% of the elements in the rows data was imputed for cfpi. 90% of the elements had no missing data.

4.5 Sparklines

Epidemiologists are interested in the developments of resistance patterns over time, a requirement defined in task T5. Hence, for both exact patterns and combined patterns, temporal variations are shown using sparklines. Sparklines are small and simple lines, originally designed by Tufte [35] with the idea to use these with the resolution of typography, such that they can be embedded in sentences. Tufte advises to use color indications with dots on special points on the sparkline, such as: the starting point; the most recent point; the highest point; and the lowest point. Moreover, a colored band can be used to show the points of the sparkline that are *normal.* In JuxtaSet the sparklines are visualized in small boxes, with the primary reason to get a quick but not detailed insight in trends. Although additional information can be displayed on the sparklines, we have chosen not to do this and aim for a minimal and stable visualization. Since the patterns are displayed in row-wise display, a sparkline is displayed next to each pattern. Sparklines show the user how the relative frequency of occurrences of a pattern evolves with respect to the rest of the data as described in equation 4.15. Hence, a flat sparkline shows that compared to rest of the dataset it has increased or decreased with a similar pace. This does not necessarily indicate that the absolute number of elements has not been increasing over time. Examples of sparklines are shown in Figure 1.1. JuxtaSet does not require the dataset to have time-dependent patterns, but it comes into full potential if this is the case.

4.5.1 Definition of Sparkline

Suppose the dataset is time-dependent, then each element has an associated time component. There are $t_1, t_2, ..., t_k$ time slots in T. A sparkline is drawn of a subset of elements E_s, all elements belonging to pattern P of which we intend to visualize sparklines. The sparkline is relative to a set of elements E_c (usually the complete database), where $E_s \in E_c$. Suppose, the variable $x = \{x_1, x_2, ..., x_k\}$ represents all the elements in E_c and the variable $y = \{y_1, y_2, ..., y_k\}$ represents all the elements E_s of pattern P. Thus, the number of elements in each time slot of pattern P is given by y_1 for time slot t_1, y_2 for time slot t_2, etc. The expected number of elements in time slot u for the elements E_s in pattern P is $\frac{|y|}{|x|}x_u$, which we simplify as z_u. We define the relative difference between the expected number of elements given the elements E_c for each time slot and the real number of elements in each time slot in pattern P with the mapping $f : T \rightarrow [-1, 1]$:

$$f(t_i) = \begin{cases} \frac{y_{t_i} - z_{t_i}}{max(y_{t_i}, z_{t_i})}, & \text{if } y_{t_i} > 0 \vee z_{t_i} > 0 \\ 0, & \text{otherwise} \end{cases} \tag{4.15}$$

Since the function f is bounded by $[-1, 1]$, the visualization of the sparkline is much easier to implement than using a relative difference function without predefined boundaries. Sparklines with only the value zero for each time slot is a complete flat line exactly in the vertical middle of the sparkline box. This holds, for example, for the combined pattern without any properties ($\emptyset \in P_C$), as shown in Figure 1.1 for the top combined pattern. This combined pattern takes all the elements in the dataset, and the relative difference to itself is zero.

4.6 Exploration Pipeline

We have now explained the main visual elements, in this section we describe how they can be used in combination to explore the data; describe a number of additional visual elements; and discuss the interaction. JuxtaSet's layout consists of a number of building blocks where each block supports at least one or more tasks:

- *combined patterns view* (T2, T5)
- *exact patterns view* (T1, T3, T5)
- *Missing Data Patterns view* (T4)
- *filters view* (T8)
- *attributes view* (T7)
- *degree sharing view* (T6)
- *table of elements* (T9)

A schematic overview is shown in Figure 4.6. Here the Missing Data Patterns view is dashed because this view is only visible when an exact pattern is selected. The *attributes view*, the *degree sharing view* and the *elements view* show additional information. These three views are collapsed by default to prevent an information overflow on screen, but can be expanded by clicking on them, as indicated by an icon. This helps users to solve the highest priority tasks first, and only if necessary do lower priority tasks (T6, T7 and T9). The filters are always in view, such that users can directly see on what properties and attributes their data has been filtered.

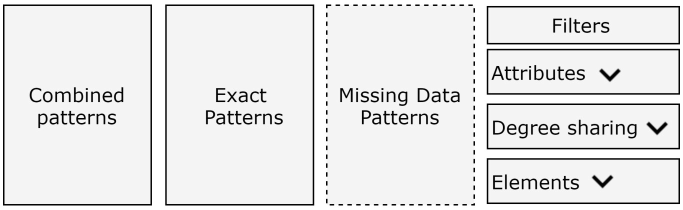


Figure 4.6: Schematic overview of JuxtaSet's layout. There is a dashed line around *Missing Data Patterns* since this view is only visible when an the missing data for a specific exact pattern is requested. The attributes view, the degree sharing view and the elements view can be expanded and collapsed, as indicated by the icon.

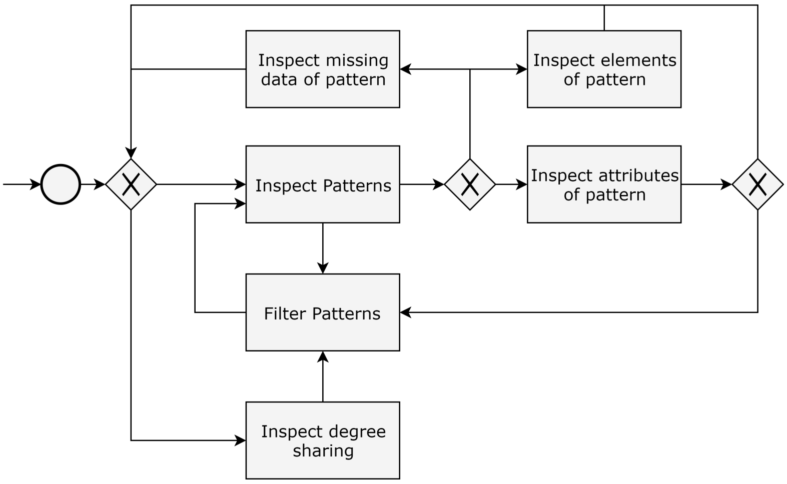

Inspect missing data of pattern
Inspect elements of pattern
Inspect Patterns
Inspect attributes of pattern
Filter Patterns
Inspect degree sharing

Figure 4.7: Process model of exploration in JuxtaSet.

4.6.1 Interaction Flow

In Figure 4.7 we show a simplified process model how a user can use JuxtaSet. They start with inspecting the patterns and possibly filter the patterns. When they have found an interesting pattern, they either inspect its attribute values, the missing data within the pattern, or in a very rare case they would like to see the elements that belong to the pattern. After inspection of attributes of the pattern, they might want to filter on attributes or specific parts of the pattern. Sometimes a user is interested in the degree sharing overview, and filter patterns based on the information.

JuxtaSet aims to exploration by supporting drill-down and filtering, linking and brushing, and sorting order, all via a limited set of interactions with visual objects shown. In JuxtaSet the following objects can be interacted with: exact patterns, combined patterns, sortables, toggles, attribute levels, filter buttons and degree sharing bars. The actions that follow when they are hovered and clicked upon are described in table 4.3. There are three different statuses for objects with corresponding colors:

- ● *default* status
- ● *hovered* status
- ● *locked* status

The colors are from AggreSet and were favoured by users during our user evaluation. When combined patterns or exact patterns are hovered, other objects are put in the *hovered* status as well, thereby supporting linking and brushing. Unlocking or moving out of an object will undo this

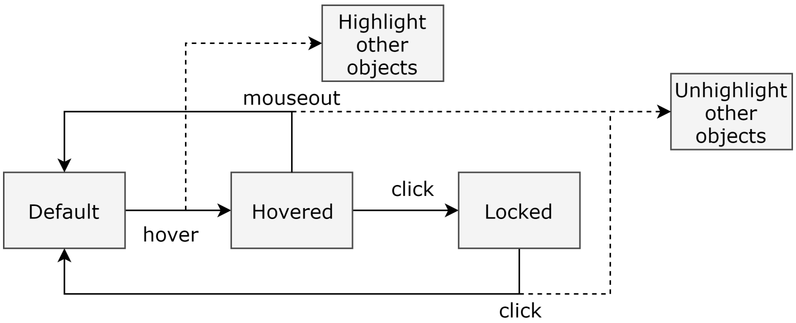


Figure 4.8: State diagram showing the states of an object (*default*, *hovered*, *locked*) and their transitions. Hovering an object possibly induces cascaded highlighting depending on the object that is hovered.

behavior. The process of flow between the defined statuses is shown in Figure 4.8. An example of the effect of hovering over an exact pattern is shown in Figure 4.9.

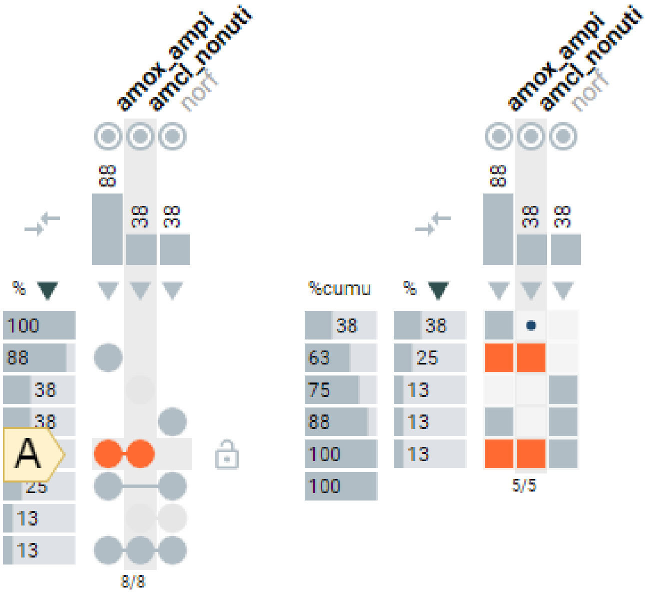


Figure 4.9: Hovering the mouse over an exact pattern, highlights the subsequent combinations in the combined patterns view. Equivalently, hovering over a combined pattern, highlights the subsequent patterns in the exact patterns view. In this specific case the combined pattern in the left diagram is hovered (A). Hence, the two exact patterns associated to it are highlighted as well.

4.6.2 Filtering

Because epidemiologists often need information on specific combinations of resistance for antibiotics or with specific attribute values, filtering of data is a necessary requirement. For instance, if they are interested in researching bladder infections in females, they would need to filter on gender and material. The result is a collection of elements from females where the measured human material is urine. This is an example of task T8. Filtering of data is integrated in JuxtaSet by clicking on objects that represent a set of elements. Clicking on an object is a natural way to select data, and gives the possibility to drill down to reduced number of elements. The

Table 4.3: Overview of elements that can be clicked, the action after a click, and to what purpose.

Object	Vis.	Hover	Click	Purpose
Combined pattern	●—●	Highlights the exact patterns of which the combined pattern is a subset.	Adds pattern to filter. Locks the pattern.	The possibility to select only elements with this pattern or inspect their attributes
Exact pattern	■■□	Highlights the combined patterns of which the exact pattern is a superset.	Locks the pattern. Opens the imputed patterns view.	The possibility to inspect imputed data within this pattern or inspect attributes
Toggle	◉ ○	Directly filters patterns for a specific property.	Locks the hovered state.	Inspect patterns ignoring the filtered antibiotic. See remaining resistance levels per antibiotic.
Attribute level	▯🟧 31-60	Show absolute number within the attribute level.	Locks the attribute level. Adds attribute's level to filter.	The possibility to filter elements with the specific attribute's level.
Drill down button	Drill down	-	Filter elements with the selected filters.	Dive deeper into a specific subset of elements that is interesting
Filters	age: 31-60 ⊗	Hints the filter can be removed.	removes planned filters. But to actually filter the elements, the *drill down* button has to be pressed again.	This is to take a broader view of the data.
Sortable	▼ ⇄	-	Sorts patterns or properties.	Reveals patterns that are interesting or puts properties in a different order.
Degree sharing bar	2 ▬ 3 ▮	-	Filters patterns with specific degree including the given antibiotic	The possibility to inspect patterns with certain properties of degree.

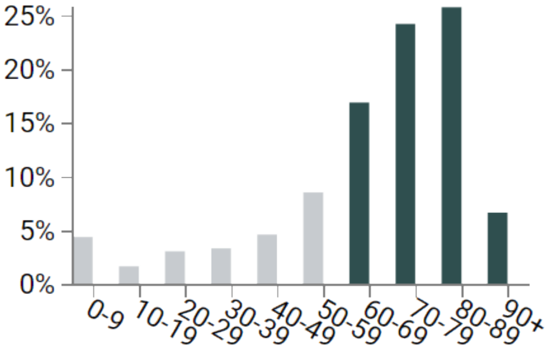
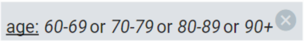
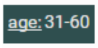
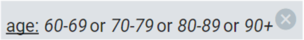


Figure 4.10: Datasets can be filtered in a range by shift+hold+click from a level to another level or by ctrl+click to select multiple individual levels.

question remains whether after a click data is directly filtered, or after a click data is just selected and highlighted, in combination with the possibility to filter when the user thinks the selection is complete. The latter gives the user the opportunity to think about their selection and possibly increase it with other selections before the screen is updated with new data. For JuxtaSet mainly the second option is chosen, *e.g.*, filters are only applied when pressed on the button *drill down*, a button in the *filters view*. When objects are selected, they appear in the *filters view*. The *filters view* shows two different classifications of filters:

- Currently selected filters that will filter elements when pressed on the button *Drill down*. These filters have not yet been executed on the elements. They can be removed and look like:  age: 60-69 or 70-79 or 80-89 or 90+ ⊗ .

- Filters that have already been executed by pressing the button *Drill down*. These filters are to show the user on what aspects specifically the data has been filtered. Hence, these filters can not be altered and look like:  age: 31-60 .

Data can be filtered on multiple filters of different attribute levels or different objects, but if one would like to select a certain level of the same attribute, either control+click can be used or shift+click. An example is shown in Figure 4.10. This behavior would not have been possible if clicks directly filter data like in AggreSet. The effect of selecting the four age bins from Figure 4.10 is a change in the summary of the currently selected filters  age: 60-69 or 70-79 or 80-89 or 90+ ⊗ . In JuxtaSet, the datasets used are often large and it may take between 30s and 55s to compute all the aggregations and statistics if they were not cached. This too is a reason to delay the execution of a filter with a separate button.

However, not all filtering operations require extensive computations, and some can be done directly without a performance penalty. We call this *direct filtering*, in contrast to *indirect filters*, which we described before. Direct filtering is supported by toggles and degree sharing objects (see Section 4.6.3).

Toggles ◉ are used to filter rows from the matrix and are direct filters, as explained above. When clicked upon they are highlighted and emptied ○ and the patterns which included the target property are filtered out. An example using data from the antibiotic resistance sample database is shown in Figure 4.11. The patterns with resistance for amcl-nonuti are no longer visible.

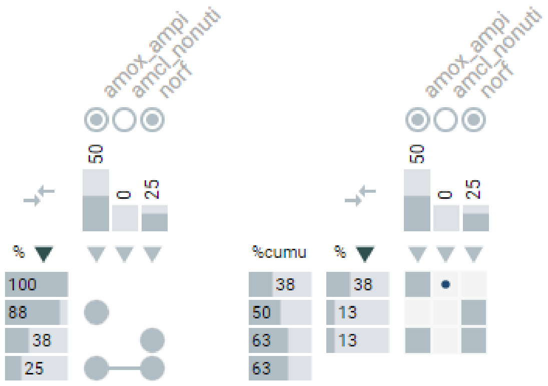


Figure 4.11: Toggles subtract patterns which have positive value for a specific property. In this case, patterns have been filtered on amcl-nonuti. filters and sorts work synchronously for combined and exact patterns, hence for both views the toggle filter is active. Compared to Figure 4.9, the patterns including resistance for amcl-nonuti have been removed. Also, it can be concluded from this visualization that of the isolates that were not resistant against amcl-nonuti, 50% is resistant against amox-ampi.

4.6.3 Degree Sharing

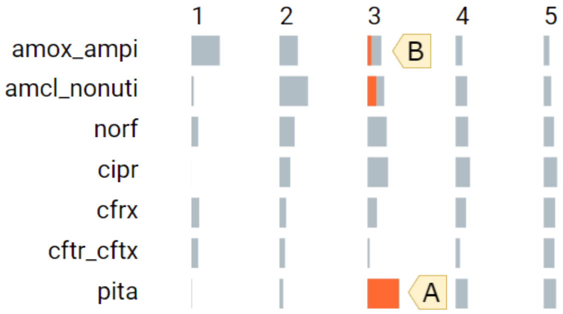


Figure 4.12: Degree sharing visualization, showing for specific antibiotics what the distribution is of the degrees of the patterns in which they occur. In this figure it is shown that the antibiotic *pita* almost exclusively is co-resistant with at least 2 or more antibiotics, and most frequently with exactly 2 other antibiotics A . Hovering over the exact pattern amox-ampi ∩ amcl-nonuti ∩ pita in the exact patterns view from Figure 1.1 highlights the bars in the degree sharing view (if it is not collapsed). We can interpret from this that all isolates that are resistant against pita and exactly two other antibiotics A , those antibiotics must be amox-ampi ∩ amcl-nonuti. However, not the other way around B .

For some antibiotics resistance always goes together with a number of other antibiotics and, on the other side, some antibiotics are hardly ever co-resistant with other antibiotics. Analysis of these effects is the aim of task T6. This information can not easily be obtained from the exact patterns. There can be hundreds or thousands of exact patterns and getting an idea of the average degree of patterns this way, is error prone. Hence, a visualization of the distribution of degrees among patterns for each antibiotic is required. Inspired by RadialSets by Alsallakh et al. and LineUp by Gratzl et al. [13] we used a baseline chart to visualize degree sharing existing of a regular table with embedded bars. Gratzl et al. explain that this way, the individual results are more easily compared between the antibiotics than using a stacked bar chart [13], an alternative solution to visualization of degree sharing. Additional reasons not to chose for a stacked bar chart, are that a stacked bar chart enforces the use of multiple colors, which is in JuxtaSet is reduced to a minimum. Besides, a stacked bar chart is unnecessary, since the bar charts above the pattern diagrams already show the frequency of a set.

An example on the antibiotic resistance database is shown in Figure 4.12: a table where each cell shows the frequency of exact patterns with a specific degree containing a specific antibiotic. We can see from this figure that the antibiotics amcl-nonuti, norf and cipr are more often co-resistant with other antibiotics than that they are the only antibiotic for which the resistance exists. In contrast, amox-ampi is mostly the only antibiotic isolates are resistant against. When we are interested which patterns occur for a specific sharing degree for a specific property, we can click on the bar and the patterns with these properties are filtered and shown. Additionally, hovering over either a combined pattern or an exact pattern shows the partition of the pattern within the degree sharing view by filling the bars of the degree sharing view. For example, hovering over the exact pattern amox-ampi ∩ amcl-nonuti ∩ pita, results in a largely filled bar for pita of set-degree three, and a little filled bar for amox-ampi and amcl-nonuti, indicating that many patterns with pita and exactly two other antibiotics, are mostly these two antibiotics. This is shown in Figure 4.12.

4.6.4 Set Attributes Exploration

Each element in the dataset is accompanied with a number of attributes. The attributes for isolates are the gender and age of the patient it originates from, date of sampling, laboratory of measurement, *etc.* The values of these attributes are aggregated and shown in charts for visual exploration as shown in Figure 4.10. Hovering a combined pattern or an exact pattern allows comparing the distributions of attributes of this pattern with all the data. The comparison is shown with an orange bar in the bar charts, like is done in Figure 4.13. The grey bars represent the average of the (possibly filtered) data.

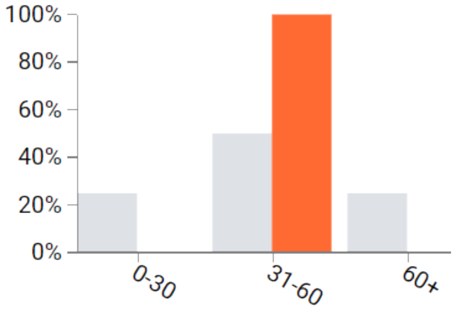


Figure 4.13: Hovering over a pattern shows the distribution of attribute values of elements within this pattern in orange bars. In grey bars the distribution of the complete dataset is shown. Here we can see that the hovered pattern has 100% of its isolates from patients between 31 and 60 years old.

Some attributes contain so many different values (for instance 42 different laboratories) that the labels can no longer be shown on the horizontal axis. AggreSet solves this problem by creating a scrollable bar chart and thereby hiding less frequent attribute levels. This however obfuscates many bars. For this reason, where hiding information from users seem a unfavorable solution, all the data is shown in the charts but bars may have small width. Hovering over a bar does show the label and the number of elements that are grouped under the level of the bar.

Chapter 5

Implementation

JuxtaSet is implemented using a web-based client-server model. Data is loaded from a database and is preprocessed and cached server-side. The visualization is shown and can be interacted with via a web-browser. User interaction with the visualization leads into communication with the server to request new data.

5.1 Technical Specifications

The visualization framework D3.js [8] is used. Python is used on a web-server for statistical computations and retrieving data. It consists of more than 5000 lines of JavaScript and 1800 lines of Python. A list of python packages that need to be installed on the server are:

- Flask (`http://flask.pocoo.org/`): a small web-framework for Python.
- NumPy by Jones et al. [16]: a statistical and numerical package for Python.
- scikit-learn by Pedregosa et al. [29]: a Python package with machine learning algorithms, data mining and data analysis tools.
- MySQLdb (`http://mysql-python.sourceforge.net/MySQLdb.html`): an interface between Python and MySQL databases.
- FancyImpute by Rubinsteyn and Feldman [31]: a Python package with a number of imputation methods.

The database must be a MySQL database. Most of the Python code deals with handling server requests from the client. Another part provides the ability to import new datasets. When a file is imported, new database tables are automatically created and the dataset gets automatically its own url. The dataset appears in a dropdown, such that the user can easily switch to other previously imported datasets. Alhough not a technical requirement for JuxtaSet, MatplotLib by Hunter [15] was installed and used for creating plots of the data for this thesis.

5.2 Scalability

Since the visualization system uses a web-server to analyze the large amounts of data and the results of these computations are stored in a cache, it is possible to retrieve results fast. However, the computations themselves may take a minute for a large dataset (100,000+ elements). The wait time to compute the cached and not-cache results have been benchmarked for the dataset from RIVM with 186.000 elements and 64 columns (attributes and antibiotics) and their results are shown in Table 5.1. Using a non-multithreaded server, the total time of these computations take

Table 5.1: Benchmark of computations

Id	After	Request	Not Cached	Cached	Size
1	-	883 Exact Patterns	8.63 seconds	0.10 seconds	132 KB
2	**1**	Sparklines for the top 30 exact patterns	11.04 seconds	0.36 seconds	71 KB
3	**1**	Aggregated imputation for the top 30 exact patterns	8.60 seconds	0.36 seconds	38 KB
4	-	Reference statistics of the complete database	4.62 seconds	0.24 seconds	18 KB
5	**1**	5106 Combined Patterns	2.64 seconds	0.24 seconds	342 KB
6	**5**	Sparklines for the top 30 combined patterns	19.31 seconds	0.42 seconds	71 KB

55 seconds, however commonly servers are multithreaded and this would take maximally 30.58 seconds $(8.63 + 19.31 + 2.64)$.

If requests have already been computed once, they are stored in a cache. Retrieving information the second time results in significantly lower wait times. The results from Table 5.1 are benchmarked on a laptop using Google Chrome with a Lenovo W541 ThinkPad with Intel Core i7 2.5GHz processor, 8,00GB Single-Channel DDR3 memory, 2047MB NVIDIA Quadro K1100M graphics card and SSD. However, a local development server is used and that is not representative for a real server. For instance, our development server does not have multithreading.

5.2.1 Combined Patterns Scalability

The website only retrieves data that is minimally necessary. New information is requested when the user interacts with JuxtaSet. There is however a limiting factor, the computation of all the possible combined patterns is exponential (2^n with n properties) and computing their frequency across each element in the database is cumbersome. The function that computes all combined patterns and then counts their frequencies is shown (modified) below:

```python
from itertools import combinations

def compute_combined_patterns(max_cardinality, antibiotics, resistant, exact_patterns):
    combined_patterns = []

    # Find all combinations with max cardinality = ...
    for cardinality in range(1, max_cardinality + 1):
        for comb in combinations(antibiotics, cardinality):
            # Add these combinations to combined_patterns list
            combined_patterns.append({
                'antibiotics': set(comb) | resistant,
                'ratio': 0,
                'absolute': 0
            })

    for exact_pattern in exact_patterns:
        for combined_pattern in combined_patterns:
            # if the combined pattern is a subset of the exact pattern
            # then increase the frequency of the combined pattern with
            # the frequency of the exact pattern.
            if combined_pattern['antibiotics'] <= exact_pattern.resistant:
                combined_pattern['ratio'] += exact_pattern.percentage / 100
                combined_pattern['absolute'] +=  exact_pattern.absolute

    # add the combined pattern with zero antibiotics with 100% frequency.
    combined_patterns.append({
        'antibiotics': resistant,
        'ratio': 1,
        'absolute': exact_patterns.no_records
    })

    # remove combined patterns with zero frequency
    combined_patterns = [a for a in combined_patterns if a['absolute']]
    return combined_patterns
```

The function above reduces the number of combinations using a *maximum cardinality* of combinations. However, if there is no maximum cardinality, the complete powerset would be computed with a cardinality of 2^n. If are 15 properties, there would be 32768 combinations. All these combinations need to counted across all the elements (which can be 100.000+). In JuxtaSet the maximum cardinality for combined patterns is set to six.

The Apriori algorithm could be used to compute combined patterns with a minimum frequency of one, since the Apriori algorithm does not compute all possible combinations [2]. An implementation of the Apriori algorithm in Python was used (the part which computes frequent itemsets, not association rules). However, it turned out that this implementation was slower than the Python function above. Faster techniques would be preferable, such as the algorithm by Orlando et al. [28], but are out of the scope of this project.

5.2.2 Scalability of Computation of Sparklines

The computation of the sparklines takes the most time in the benchmark from Table 5.1. For the top 30 combined patterns, 30 times the elements belonging to these patterns need to looked up in the database and their associated temporal attribute. Then these elements need to be grouped according to the temporal attribute and equation 4.15 is used to compute the sparklines. The bottleneck is consulting the database 30 times to find the corresponding elements belonging to the combined patterns. This piece in JuxtaSet would preferably be rewritten such that the database is only consulted once.

5.2.3 Scalability of Visualization in JuxtaSet

The layout of JuxtaSet limits the number of attributes it can visualize. Up to 6 attributes can be visualized, others are only visible when the window is scrolled. Also, the number of properties JuxtaSet can visualize is limited to 17 properties, since properties are visualized horizontally and shown twice (in combined patterns view and in exact patterns view).

Chapter 6

Generalization

In this section we give examples of JuxtaSet used on other common datasets, such that we can show that JuxtaSet is domain specific, in the sense that it can be used for epidemiological research, but also general purpose. Plaisant states that potential users may not feel confident on how well a generic visualization technique matches their own needs and tasks [30]. Therefore, the majority of this thesis aims to answer questions explicitly for epidemiologists working with antibiotic resistance data. JuxtaSet may interest other epidemiologists working with antibiotic resistance datasets. However, Plaisant also encourages that designing general-purpose visualization techniques as the visualization technique can then be easily ported to different datasets. Hence, we show evidence that JuxtaSet is able to visualize any set-typed dataset. JuxtaSet comes to full potential when a dataset is time-dependent and contains missing data. To prove that JuxtaSet can visualize other datasets, the *Les Misérables* dataset [17] and the (new) movies dataset with 27k movies [14] have been visualized in figures 6.1 and 6.2 respectively.

- *Les Misérables* is a novel consisting of 5 volumes with each a number of books and in total 365 chapters. The dataset by Knuth [17] consist of character meetings in each chapter. Within a single chapter there can be multiple meetings between different characters. In total there are 470 meetings. The meetings between multiple characters are visualized as patterns. *Les Misérables* is time-dependent as it consists of multiple books with multiple chapters in chronological order. Each single chapter is considered a *time-step*.
- The Movies database of 27k movies by Harper and Konstan [14] is a common database for research, education and industry. It has over thousands of downloads each year and 7500+ references in Google Scholar [14]. The database contains movies, ratings and tags. Each movie has a year in which it is produced and is classified to a number of genres. Moreover, users can give movies tags (such as *Nicolas Cage* if he appeared in the movie) and rate movies. Both actions are associated with timestamps. In our analysis we use the genres, ratings and compute a variable *watched*. The variable *watched* is the number of ratings the movie has received. Also, for each movie the average rating is computed. Individual ratings, tags and timestamps are not interesting to us. We visualize genres of movies as patterns. The Movies database is time-dependent because movies are from a specific year. Therefore, we can measure the popularity of specific genres over the course of years from 1880 to 2015.

Since these datasets do not contain missing data, there are no blue dots in the exact patterns diagrams. Short analyses are given that can be deduced from the static screen-shots. Of course, the visualizations are dynamic and provide much more information.

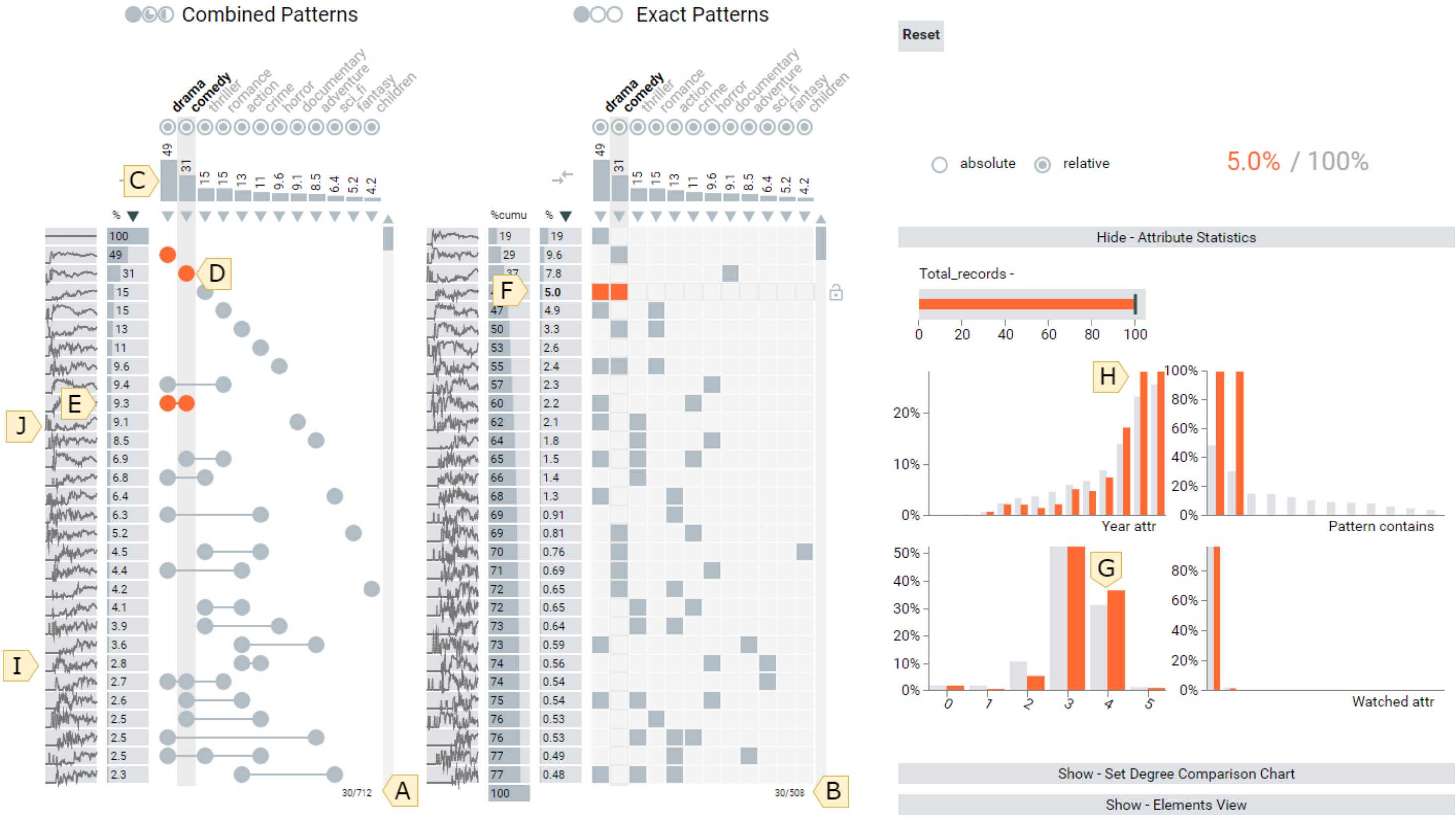


Figure 6.1: Visualization of the Movies dataset [14]. There are 712 combined patterns (A) and 508 exact patterns (B). 49% of the movies is a drama movie (C) and 31% of the movies is a comedy (D). 9.3% of the movies is both a comedy and drama movie (E), whilst 5% of the movies only drama and comedy (F). Drama and comedy movies generally have a bit more 4/5 ratings than is usual (G) and are nowadays more popular than they were before (H). Movie genres are typically not stable in popularity, the sparklines are not flat. Typically, almost all genres have a flat line on the bottom in the beginning of the sparkline indicating there were an extremely low number of movies of those genres (I). Exceptions are romance, sci-fi and documentary (J) movies which apparently were popular a very long time ago.

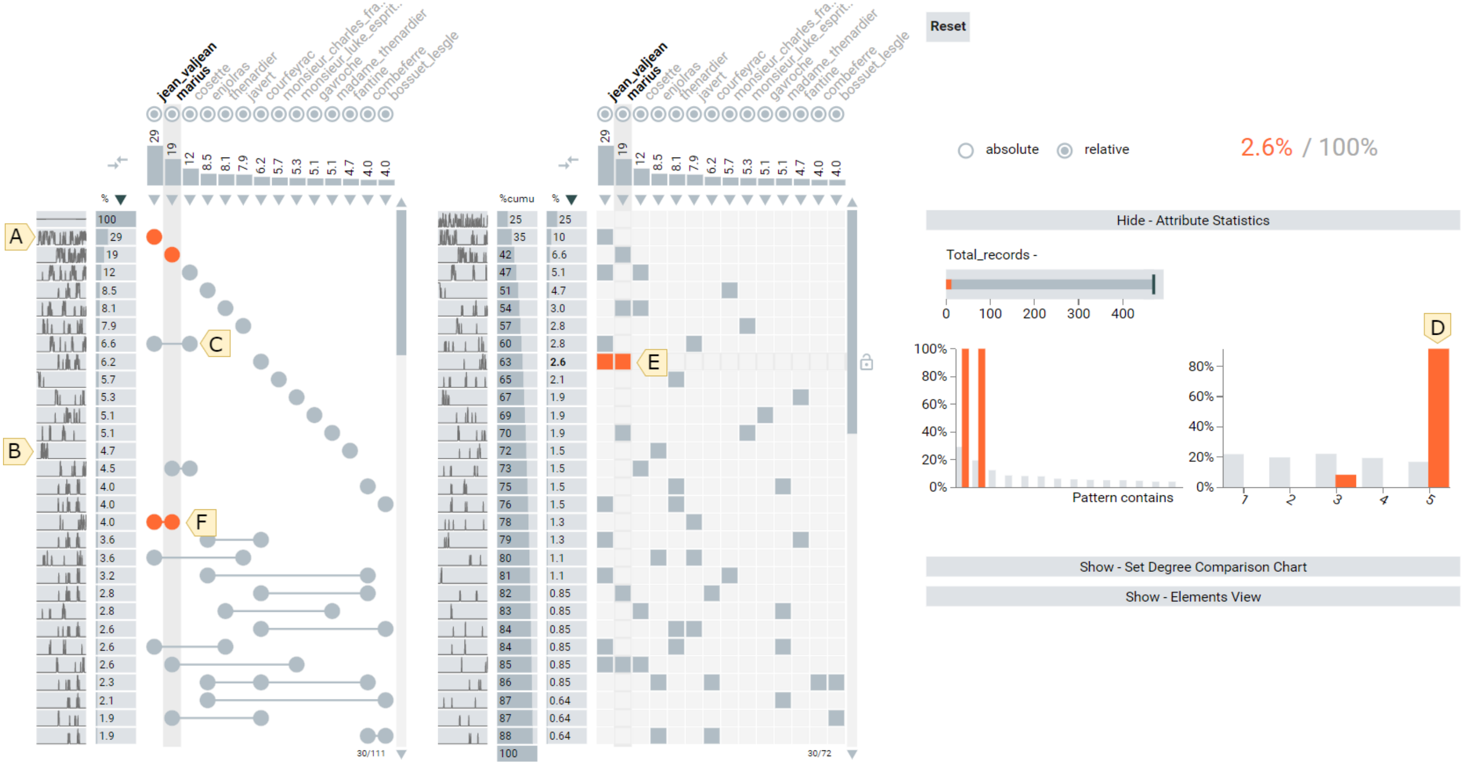


Figure 6.2: Here, a screenshot is shown of the visualization of the *Les Misérables* using JuxtaSet. It can be seen that Jean Valjean is present many times throughout the books, in some chapters absent, in others abundantly present (A). Characters are usually present a lot in a number of succeeding chapters and absent in many other parts of the play (B). Jean Valjean and the girl Cosette are more present together than most other characters are in the play at all (C). The exact pattern with only Jean Valjean and Marius is hovered, and it can be seen that Jean Valjean and Marius meet mostly and abundantly in the last volume (D). Meetings with only Jan Valjean and Marius (E), is about 2.6% of all meetings between characters, whilst meetings between characters with at least Jan Valjean and Marius, is 4% of all meetings (F). Furthermore, it can be seen that combinations of three characters are not so much prevalent. The user needs to scroll the combined patterns view, since combinations of one and two characters are dominating.

Chapter 7

User Evaluation

In this section we describe what choices we made in our user evaluation design. Then we report on comprehensiveness and readability, and results from open evaluations. Lastly we report on design evaluation results.

7.1 User Evaluation Design

In Section 2.3 we defined the requirements for our tool, and obviously, these have to be evaluated in practice, to verify whether experts can perform their tasks. Moreover, we explain which type of evaluation setup was used.

7.1.1 Requirements Testing

During the development of JuxtaSet, it was questioned what should be the focus: for instance the speed in which users can answer their questions or how happy they are when they are using the tool? Or what is the exploratory power? Many more questions are alike and listed in Visualization Analysis and Design by Munzner [25]. JuxtaSet was designed for monitoring of antibiotic resistance in the first place and hence its focus should be evaluated in the user evaluation. As described in the requirements in Section 2.3, it is necessary that JuxtaSet is stable and intuitive, and that using the tool is joyful. Above all, epidemiologists need to be able to answer questions correctly. These quality attributes were tested in the user evaluation.

Although it is more common that evaluations consist of simple tasks [30], doing so would not represent the real setting for epidemiologists. The system is required to complete the tasks listed in 2.2, and hence it is evaluated on these tasks despite their complexity. The tasks given in the user evaluation matched most of the tasks defined in Section 2.2. Some features related to tasks were not fully implemented during the user evaluation. For instance, task T5 is not evaluated, since sparklines were not implemented in the prototype version. Likewise are task T6, T9 and task T10.

7.1.2 Evaluation Set-up

Usability evaluation and controlled experiments are the most common types of evaluation [30]. There are two different types of controlled experiments: comparing design elements or comparing complete tools. We have chosen to compare design elements in order to obtain information on which types of visualization techniques are most preferred among users. Especially, since the system tested was a prototype version, it was tried to gain as much points of improvements as possible. Additionally, we have performed a usability evaluation in order to find out what slows down the user and what is unclear or not intuitive. During the evaluation session, the epidemiologists were instructed to answer three warming up questions, thirteen technical questions, eleven design and extent of apprehension questions and five open evaluation questions. The complete list of questions is given in Appendix A. Here, we give summaries of each phase of questions:

- *Warming up questions*: simple questions to get used to the way questions are asked in the technical part. Without the warming up questions, participants might feel overwhelmed. Moreover, with these warming up questions participants might already get an idea of what they see and the structure of the JuxtaSet. No intelligent user interaction is required for these questions.
- *Technical questions*: questions of increasing complexity based on the tasks from Section 2.2. Intelligent user interaction is required to answer these questions. The questions are representative for the questions epidemiologists will have when using JuxtaSet. Epidemiologists at RIVM are seasoned researchers and are expected not to feel stressed from complex questions, but might feel enthusiastic that they can answer these questions with JuxtaSet.
- *User evaluation and design questions*: questions that check how well participants understood information presented in specific views, and questions in which different designs are compared. Since a prototype version is evaluated, it is necessary to find out what should be improved and what is preferred.
- *Open evaluation*: questions that require an explanation. Like the *user evaluation and design questions*, these questions aim to find out what should be improved and what epidemiologists like about JuxtaSet.

No explanation was given before the evaluation sessions, but if the participants got stuck during the session they were given explanations to enable them to continue the questionnaire. There were seven participants with ages from about 30 to 50, of which five are female.

7.2 User Evaluation Results

During the user evaluation it became clear that users do require some introductory explanation of the main views in JuxtaSet. For instance, because not everybody directly understood the patterns can be sorted. Since explanation was given during the sessions, exact measurements of time and error rates of questions are not representative and are not analyzed statistically for this evaluation. Meanwhile, the extent to which users were able to independently answer the questions and the speed in which they were able to solve the technical questions varied. Three participants, who were already very familiar with the ISIS-AR database, needed little or no explanation. Two participants asked explanation on the interaction idiom (such as sorts and filters), and two participants needed an extensive explanation of the system before they started answering the questions. In the end, most people found the answers to almost all questions, except for a question, where an implication (if X then Y) with regard to antibiotic resistance was asked (question 7). However, question 7 has little priority, and was eventually removed from the questionnaire. Two people gave incorrect answers on the question on finding the absolute number of isolates in a pattern (question 5). Given the importance of this question, significant changes have been made in the system to present the absolute number of a hovered pattern explicitly on screen.

7.2.1 Improvement Suggestions During Evaluation.

During the evaluation of the initial prototype it became clear that there were a number of things that were not intuitive. In the preceding chapters the final version was described, here we indicate which elements were introduced or changed based on the evaluation:

- Filtering was not intuitive and hence the filtering system was updated after the evaluation. Clicking on a pattern or an attribute level now updates the set of filters on screen. Moreover, a distinction between planned filters and already executed filters is introduced.

- Abbreviations of antibiotics were unknown to some people, hence hovering over an antibiotic now provides the full name.
- Epidemiologists wanted to sort the columns which was impossible at first. Two different column sorting possibilities have been introduced: by frequency and by alphabet.
- Hovering with the mouse over a pattern directly requested information from the server. This implied that hovering with the mouse fast across the screen, many objects moved. Now, new information is not requested without hovering with the mouse for at least 400ms over the pattern.

7.2.2 Open Evaluation

When participants were questioned what they find positive about JuxtaSet, they responded (translated from written text):

> “You can see patterns that are normally not directly visible and normally you have to search for them specifically.”
> “Very beautiful visualization of data that is made insightful. Its interactive possibilities.”
> “All information is visible in one screen. You can directly update data by clicks and hovers.”
> “Calm view, nice overview, explanation button.”
> “It gives a very complete overview of all the possible patterns and their frequencies. It gives insight in the quality of the data (missing data etc.).”
> “Different tables and views and moving selections.”
> “Nice form, you can obtain a lot of information but it is not for 10 minutes. Nice how imputation is processed, it is a lot but it is also useful when you need it.”

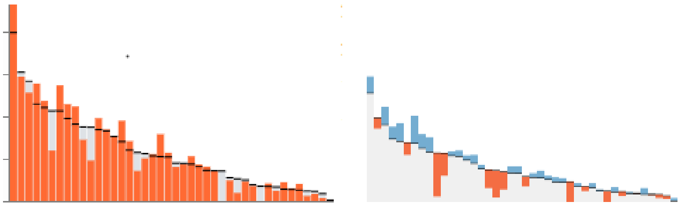
When participants were questioned whether they found JuxtaSet intuitive, they responded:

> “Yes, but it is somewhat complicated.”
> “Not completely, it takes some time to figure out what everything means, but the question tool helps in this process.”
> “Yes, even more when you know this is a complicated topic.”
> “Yes, but you need some practice.”

7.2.3 Design Evaluation

During the design evaluation a number of design choices were evaluated. The results are listed:

- 6 out of 6 users found the text readable.
- On the question whether they understood what is shown in the combined patterns view, participants responded with an average score of 3.67/5 on a Likert-scale.
- Equivalently they were asked whether they understood what is shown in the exact patterns view, and they responded with a score of 4/5.
- Every participant confirmed that they understand how JuxtaSet deals with missing data.
- Different color palettes were evaluated: AggreSet’s color pattern ●●●●●, UpSet’s color pattern ●●●●●, Brewer’s *PuBuGn* ●●●●● and Brewer’s *RdYlBu* ●●●●●. There is a strong preference for AggreSet’s color pattern and on average no preference for the last three colors patterns.
- Two different visualization techniques of bar charts were evaluated: one where two bars are drawn over each other and the other where the difference between the overlapping bars



(a) Overlap

(b) Difference

Figure 7.1: Two different bar chart layouts were tested during the design evaluation. These bar charts do not represent the same dataset. In (a) the orange bars that belong to one pattern overlap the reference bars of the rest of the dataset. The top of the reference bar charts are indicated by a small line. In (b) differences between bars that belong to one pattern and the reference bars are shown. The blue colored bars represent bars of the pattern that were higher than the reference bars. The orange bars represent lower values than the reference values.

is highlighted. The participants have unanimously shown their preference over the former. Examples are shown in Figure 7.1.

- The preference for usage of circles or squares in the pattern views has been evaluated. The participants have unanimously shown their preference of circles in the combined patterns view and squares in the exact patterns view. Their reasoning was that these two views should have different encoding otherwise they will think these views are the same, or they are connected more than they are. Additionally, in the right view they liked squares, because the imputation is depicted with circles, and they find circles within circles less clear. An example of (the prototype version of) JuxtaSet where both the combined patterns view used circles and the exact patterns view used circles.

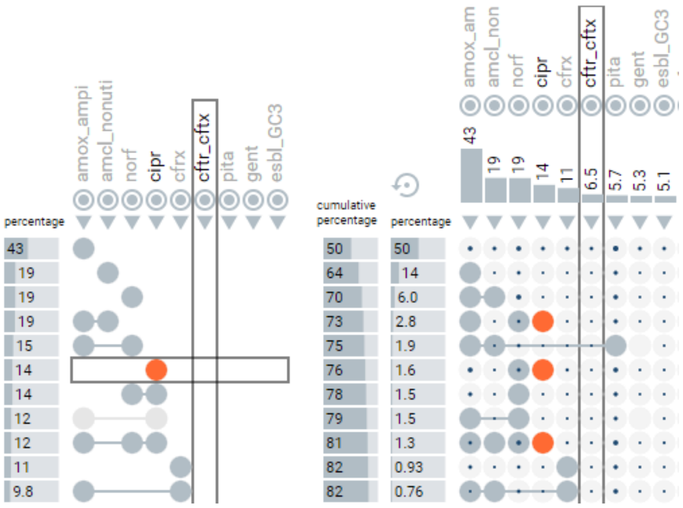


Figure 7.2: Three different layouts were shown to users. Circles and circles, squares and squares, circles and squares used by combined and exact patterns respectively. Here one of those layouts is shown: circles for both combined patterns and exact patterns. Although it looks nicely, people said they got confused and expected both views to represent the same. Hence their preference for circles for combined patterns and squares for exact patterns.

- The lines between circles in the combined diagram were evaluated on how distracting they found these lines. These lines were based on the design of UpSet. The participants have decided on a Likert-scale with a score of 4.3/5 on average that they are not distracted by the lines between the circles. Hence, they clearly liked the connecting lines between the circles. The lines helps to quickly see which circles are on the same row.

Chapter 8

Discussion

First, we discuss an assumption in the mathematical models from Section 4.1. Then we discuss a number of limitations of JuxtaSet. Finally, we discuss possibilities for future work.

8.1 Completeness of Formal Definitions

The equations for exact and combined patterns, Equation 4.4 and Equation 4.7, assume all missing data is imputed and take the value of 1 or 0 in A. However, in some cases there is so many missing data for a property that the percentage of missing data for one property is greater than the cut-off point, and imputation cannot reliably be performed. Only in these cases, a third category that represents *missing and not imputed*, an extension to A, should be used. However, integration of this category has impact on the equation of exact patterns, since the relation between elements and properties is no longer Boolean. The equations become much more complex and have therefore not been taken into the scope of this thesis.

8.2 Limitations of JuxtaSet

We present a number of issues JuxtaSet struggles with:

I1 For large datasets the computation of all statistics, sparklines, combined patters and exact patterns can take tens of seconds. The computation of the combined patterns and counting the frequencies among all elements, was the most limiting factor. In order to overcome this problem, only combinations of up to 6 properties are computed and counted by default, and take less than 3 seconds. If datasets are small and users are interested in combinations of higher degrees, they can choose the limit themselves.

Sparklines now take the most time. It can be configured per dataset whether the sparklines are computed. This way, users can decide if they are willing to wait for them. Preferably, this setting can be dynamically changed on screen by the user. The sparklines are computed only for the patterns visible on screen, in order to minimize computation time. Thus, if new patterns emerge due to sorting and filtering, sparklines are computed for these patterns individually. Furthermore, results are cached allowing instantaneous responses for returning users. Ultimately, the waiting time could be decreased if the most probable information requests are precomputed directly after importing a dataset.

I2 JuxtaSet is designed to visualize 17 properties correctly on a screen with at least 1600x900 pixels. Then, users do not need to scroll horizontally. If the user would like to analyze more than 17 properties on screen, software changes are necessary and a wider screen.

I3 To show time trends in resistance patterns, JuxtaSet visualizes sparklines next to each resistance pattern. These sparklines are calculated relative to the rest of the dataset and in a way that lines in the exact middle of the box mean that the trend is similar to the rest of the dataset. This holds even if the percentages are very low or very high. This may complicate the interpretation of percentages through the years. The formula that computes the relative difference from the rest of the dataset is also of moderate complexity. Although users are not bothered with a formula, it is of utmost necessity that they understand what the sparklines really represent. However, next to the sparkline the percentage is given for all years together, and epidemiologists are mainly interested to know whether a time trend is increasing or decreasing, or stable. Therefore, this way visualizing sparklines was considered most functional. To create more detailed insight in exact percentages over the years, an extra feature could be programmed visualizing the sparklines in a large plot.

I4 JuxtaSet is specifically designed for the tasks and requirements of epidemiologists. Although it has been shown that JuxtaSet is general-purpose, it has no intention to support all the possible set related tasks described by Alsallakh et al. [5].

I5 Our simulations in chapter 4.4 showed that the algorithm MICE might give more reliable results with respect to imputation of missing data. However, because MICE requires a substantial implementation effort, which was out of the scope of this thesis, and because SOFT-IMPUTE was a more scalable alternative, SOFT-IMPUTE was used for imputation of data in JuxtaSet. In order to get more reliable results, in future an effort is necessary to implement MICE.

I6 The color scale ●●●●● from AggreSet that is used in JuxtaSet has two colors that have much resemblance, namely the color that represents the locked state of an object ● and the color that represents missing data ●. Therefore, one of these two colors should preferably be changed. For instance, one alternative could be changing the dark green color to red ●●●●●. Colors can easily be changed in JuxtaSet's settings.

I7 It was concluded in Section 4.4.3, that the complete cases dataset that was based on real data had less incorrectly imputed data, irrespective of the imputation algorithm used. Possibly, this was caused by a lower deviation between rows within the real dataset compared to the random generated dataset. This however, needs to be examined more carefully and will help the statistician to define a cut-off point.

8.3 Concepts From Related Work

Combined and exact patterns that are not frequent and also less frequent than expected are not directly visible. Like in other Set-visualization techniques discussed in this paper, a deviation variable could be introduced. Patterns could be colored based on the deviation of its frequency with respect to the expected frequency. The expected frequency is calculated using the complete dataset. Visualization of deviation of patterns using colors is, for instance, implemented by PowerSet. Another possibility is, for instance, showing the deviation next to each pattern like in UpSet. Ultimately, combined and exact patterns should be able to be filtered and sorted on deviation.

Furthermore, in this thesis it was shown that the theory of frequent itemsets is remarkably close to theory discussed in set visualization. Additionally, the concepts *frequent itemsets* and *association rules* are inseparably connected. Not surprisingly, mining association rules could be helpful to epidemiologists. Information such as, *given a set of isolates with resistance to antibiotics X and Y, for 80% of these isolates there was resistance to antibiotic Z*, could be interesting to epidemiologists. Although, it is possible to retrieve this information given two antibiotics X and Y, a clear overview of these association rules is not implemented. PowerSet visualized association rules, where the right hand side of the association rule must be chosen (in this case Z), and the left hand side and their confidences are visualized.

The tasks defined in Section 2.2 have a considerable overlap with the complete list of set visualization tasks from the review by Alsallakh et al. [5]. However, as discussed in Issue I4, JuxtaSet has had no intention to support all set-related tasks. For instance, the task where combinations of sets can be found that have no sharing element, is unsupported by JuxtaSet since there was no direct need for this information. Tasks like these and others can still be implemented if users ask for it.

8.4 Future Work

At the end of this thesis, there are some questions left that cannot be answered using the current version of JuxtaSet. These questions are described first. Next, a large scale evaluation is proposed and other small points of discussion are described.

8.4.1 Unanswered Questions

For users that do have geographical information about elements in their datasets, a feature can be implemented showing the geographical location for each element on a map (task T10), which was built in early versions of JuxtaSet. However, for a good implementation, a recent and complete database on postal code coordinates for each year is necessary, which was not available at that time. Therefore, and because JuxtaSet is a general-purpose tool and datasets must be set-typed, we have chosen not to implement this feature.

Furthermore, meetings and the user evaluation with epidemiologists gave rise to several new questions, which are stated below:

Q1 Given the isolates are resistant against a certain antibiotic, to what antibiotics are the isolates most probably susceptible?

Q2 What are the steps if an imputation of missing data is found that can not (biologically) be correct?

An answer to Q1 is useful when epidemiologists would like to know what possible treatments are plausible if a certain antibiotic has been confirmed. Answers to this question would help in developing empiric antibiotic treatment guidelines. Currently, we have enabled epidemiologists to answer the opposite of this question (*Given the isolates are resistant to a certain antibiotic, to what antibiotics are the isolates most probably resistant?*) to give them insight in the existing resistance combinations. Instead of computing combinations where cells have the value 1 as is done in the current versions of JuxtaSet, here combinations are requested where cells have the values 0. Hence, this would not seem to be difficult to implement.

Imputation is based on an algorithm and does therefore not take into account biological impossibilities. After identification of a faulty imputation, manual adaptations could be made in the source code of the imputation method using certain rules that correct the faulty imputations. More sophisticated efforts would allow the user to dynamically define these kind of rules during interaction with the visualization.

8.4.2 Validation

Commonly, like in this study, untrained users are participating in user evaluations of visualization techniques [30]. Longitudinal studies may help getting a better insight in the evaluated visualization technique [30]. A large scale evaluation in practice should be conducted when the users are trained, such that it can be proven that tasks can be performed correctly and fast, whether extra features are necessary.

8.4.3 Other Points of Discussion

In JuxtaSet sparklines for the missing data patterns view are implemented, but are incorrect due to a software bug. Hence, these have been turned off in the current version of JuxtaSet. This bug will be fixed in the near future. Additionally, in Section 5.2.2 it was described that, at the time of writing, the computation of sparklines takes an unnecessarily long time. Faster implementations that do not consult the database as many times as it does would speed up the waiting time.

Chapter 9

Conclusions

We presented a new set visualization technique called JuxtaSet, which visualizes combined patterns and exact patterns interactively. Additionally, the time-dependency of patterns is visualized. Missing data is visualized both aggregated and in detail. Sorting, filtering and linked highlighting provide the user possibilities to gain more insight in the data. A user evaluation has been conducted with expert epidemiologists, who provided information on which parts of JuxtaSet provided the needed information in a user friendly way and which parts needed to be adapted, which was done accordingly. JuxtaSet is an application-specific system, since it is specifically built to support epidemiologists working with microbiological resistance data. However, it is also general-purpose. This is proven with examples on the movies and Les Misérables databases. New datasets in CSV format can easily be imported into JuxtaSet. These files can consist of millions (10M+) of cells. JuxtaSet uses a client-server model and can be visited using a web-browser. Limiting factors in scalability in JuxtaSet are the number of properties, attributes and the exponential number of combinations of properties.

Bibliography

[1] Infectieziekten surveillance informatie systeem-antibiotica resistentie (isis-ar). Website RIVM. URL http://www.rivm.nl/Onderwerpen/S/Surveillance_van_infectieziekten/Infectieziekten_Surveillance_Informatie_Systeem_Antibiotica_Resistentie_ISIS_AR. 5

[2] Rakesh Agrawal, Tomasz Imieliński, and Arun Swami. Mining association rules between sets of items in large databases. In *Acm sigmod record*, volume 22, pages 207–216. ACM, 1993. 19, 39

[3] Bilal Alsallakh and Liu Ren. Powerset: a comprehensive visualization of set intersections. *IEEE transactions on visualization and computer graphics*, 23(1):361–370, 2017. 2, 3, 15, 16, 17

[4] Bilal Alsallakh, Wolfgang Aigner, Silvia Miksch, and Helwig Hauser. Radial sets: Interactive visual analysis of large overlapping sets. *IEEE Transactions on Visualization and Computer Graphics*, 19(12):2496–2505, 2013. 2, 34

[5] Bilal Alsallakh, Luana Micallef, Wolfgang Aigner, Helwig Hauser, Silvia Miksch, and Peter Rodgers. The state-of-the-art of set visualization. In *Computer Graphics Forum*, volume 35, pages 234–260. Wiley Online Library, 2016. 2, 11, 15, 52

[6] Clemens Arbesser, Florian Spechtenhauser, Thomas Mühlbacher, and Harald Piringer. Visplause: Visual data quality assessment of many time series using plausibility checks. *IEEE transactions on visualization and computer graphics*, 23(1):641–650, 2017. 18

[7] Lorenzo Beretta and Alessandro Santaniello. Nearest neighbor imputation algorithms: a critical evaluation. *BMC medical informatics and decision making*, 16(3):74, 2016. 24

[8] Michael Bostock, Vadim Ogievetsky, and Jeffrey Heer. D^3 data-driven documents. *IEEE transactions on visualization and computer graphics*, 17(12):2301–2309, 2011. 37

[9] Stef Buuren and Karin Groothuis-Oudshoorn. mice: Multivariate imputation by chained equations in r. *Journal of statistical software*, 45(3), 2011. 24

[10] de Greeff and JW Mouton. Consumption of antimicrobial agents and antimicrobial resistance among medically important bacteria in the netherlands in 2016. Governmental annual report, June 2017. URL http://rivm.openrepository.com/rivm/bitstream/10029/620877/1/2017-0056.pdf. 5, 6

[11] Wolfgang Freiler, Kresimir Matkovic, and Helwig Hauser. Interactive visual analysis of set-typed data. *IEEE Transactions on Visualization and Computer Graphics*, 14(6), 2008. 1, 19

[12] Humberto Garcia-Caballero, Manuel Campos, Jose M Juarez, and Francisco Palacios. Visualization in clinical decision support system for antibiotic treatment. In *Actas de la XVI Conferencia de la Asociación Española para la Inteligencia Artificial, CAEPIA*, pages 9–12, 2015. 11

[13] Samuel Gratzl, Alexander Lex, Nils Gehlenborg, Hanspeter Pfister, and Marc Streit. Lineup: Visual analysis of multi-attribute rankings. *IEEE transactions on visualization and computer graphics*, 19(12):2277–2286, 2013. 34

[14] F. Maxwell Harper and Joseph A. Konstan. The movielens datasets: History and context. *ACM Trans. Interact. Intell. Syst.*, 5(4):19:1–19:19, December 2015. ISSN 2160-6455. doi: 10.1145/2827872. URL http://doi.acm.org/10.1145/2827872. 41, 42

[15] J. D. Hunter. Matplotlib: A 2d graphics environment. *Computing In Science & Engineering*, 9(3):90–95, 2007. doi: 10.1109/MCSE.2007.55. 7, 26, 37

[16] Eric Jones, Travis Oliphant, Pearu Peterson, et al. SciPy: Open source scientific tools for Python, 2001–. URL http://www.scipy.org/. [Online; accessed ¡today¿]. 37

[17] Donald Ervin Knuth. *The Stanford GraphBase: a platform for combinatorial computing*, volume 37. Addison-Wesley Reading, 1993. 16, 41

[18] T. Leenstra, S.F.T. Thijsen, and A.K. van der Bij. Isis-ar inzicht in 7 jaar lokale en nationale antibioticaresistentietrends. *NVMM*, 1(23):22–28, 2015. 5, 6

[19] Jure Leskovec, Anand Rajaraman, and Jeffrey David Ullman. *Mining of massive datasets*. Cambridge University Press, 2014. 1, 2, 20

[20] Alexander Lex and Nils Gehlenborg. Points of view: Sets and intersections. *nature methods*, 11(8):779, 2014. 15

[21] Alexander Lex, Nils Gehlenborg, Hendrik Strobelt, Romain Vuillemot, and Hanspeter Pfister. Upset: visualization of intersecting sets. *IEEE transactions on visualization and computer graphics*, 20(12):1983–1992, 2014. 2, 3, 15, 22

[22] William J Love, Kelson A Zawack, James G Booth, Yrjo T Grhn, and Cristina Lanzas. Markov networks of collateral resistance: National antimicrobial resistance monitoring system surveillance results from escherichia coli isolates, 2004-2012. *PLoS computational biology*, 12 (11):e1005160, 2016. 12, 13

[23] Rahul Mazumder, Trevor Hastie, and Robert Tibshirani. Spectral regularization algorithms for learning large incomplete matrices. *Journal of machine learning research*, 11(Aug):2287–2322, 2010. 24

[24] Rahul Mazumder, Trevor Hastie, and Robert Tibshirani. Spectral regularization algorithms for learning large incomplete matrices. *Journal of machine learning research*, 11(Aug):2287–2322, 2010. 24

[25] Tamara Munzner. *Visualization analysis and design*. CRC Press, 2014. 17, 45

[26] Thomas F O'Brien and John Stelling. Integrated multilevel surveillance of the world's infecting microbes and their resistance to antimicrobial agents. *Clinical microbiology reviews*, 24 (2):281–295, 2011. 6

[27] World Health Organization. Antimicrobial resistance, 2016. URL http://www.who.int/mediacentre/factsheets/fs194/en/. 1

[28] Salvatore Orlando, Paolo Palmerini, and Raffaele Perego. Enhancing the apriori algorithm for frequent set counting. In *DaWaK*, volume 1, pages 71–82. Springer, 2001. 39

[29] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011. 37

[30] Catherine Plaisant. The challenge of information visualization evaluation. In *Proceedings of the working conference on Advanced visual interfaces*, pages 109–116. ACM, 2004. 41, 45, 53

[31] Alex Rubinsteyn and Sergey Feldman. fancyimpute: Version 0.0.16, May 2016. URL `https://doi.org/10.5281/zenodo.51773`. 24, 25, 37

[32] Jonathan AC Sterne, Ian R White, John B Carlin, Michael Spratt, Patrick Royston, Michael G Kenward, Angela M Wood, and James R Carpenter. Multiple imputation for missing data in epidemiological and clinical research: potential and pitfalls. *Bmj*, 338:b2393, 2009. 1

[33] Deborah F Swayne and Andreas Buja. Missing data in interactive high-dimensional data visualization. *Computational Statistics*, 13(1):15–26, 1998. 17, 18

[34] Economics & Policy The Center For Disease Dynamics. Resistancemap. Visualization software, 2010-2015. URL `https://resistancemap.cddep.org/`. 12

[35] Edward R Tufte. Beautiful evidence. *New York*, 2006. 28

[36] M Adil Yalcin, Niklas Elmqvist, and Benjamin B Bederson. Aggreset: Rich and scalable set exploration using visualizations of element aggregations. *IEEE transactions on visualization and computer graphics*, 22(1):688–697, 2016. 2, 3, 15

Appendix A

User Evaluation Questionnaire

Three questions were asked during the warm-up phase of the questionnaire.

1. What is the percentage of females in the dataset?
2. What is most frequent resistance pattern?
3. What is the characteristic color of JuxtaSet when you move your mouse?

Twelve questions were asked during the technical phase of the user evaluation.

1. What is the percentage of isolates resistant against Amoxicilline/Ampicilline?
2. What is the percentage of females of the isolates that are resistant against Norfloxacine?
3. What percentage of isolates is resistant against Cefuroxim?
4. What percentage of isolates is resistant both Cefuroxim and Ceftazidim?
5. What is the absolute number of isolates that are resistant against both Cefuroxim and Ceftazidim?
6. Which is more prevalent: resistance for Gentamicine; or combined resistance for Amoxicilline/Ampicilline, Amoxicilline/Clavulanic acid, Norfloxacine and Ciprofloxacine?
7. Give one if-then relation with regard to antibiotic resistance. For example, if there is resistance for antibiotics X and Y then there is resistance to antibiotic Z.
8. With what other antibiotic is ESBL most prevalent?
9. What is the most frequent resistance pattern of isolates with resistance against Norfloxacine?
10. Filter on the combined resistance for Amoxicilline/Ampicilline and Amoxicilline/Clavulanic acid. Inspect the second exact pattern, what do you find surprising if you compare these results with the combined patterns?
11. What is the meaning of the blue dots in the exact patterns view?
12. Select the fifth exact pattern.

    A What is the number of distinct patterns where data was missing?

    B What percentage of the data of the selected resistance pattern was complete and what percentage was missing?

    C Figure we are interested in a summary of the listed patterns. How many resistance patterns do you need to capture 95% of the data and for which antibiotics has there been missing data in these patterns?

13. Give a resistance pattern that is relatively more prevalent among children of 0 to 9 years old.

Eleven questions were asked during the design phase.

1. The visualization is calm.(Likert-scale)
2. The following color palette is most appealing to me. (and order them by favourability)
   (a) AggreSet
   (b) UpSet
   (c) Brewer green-blue
   (d) Brewer red-blue
3. Of the two different bar chart designs, I find the most favourable:
   (a) Option 1 (overlay bars)
   (b) Option 2 (stacked bars)
4. Of the three design possibilities for the heatmap, I find the most favourable:
   (a) Combined patterns circles, exact patterns circles
   (b) Combined patterns squares, exact patterns squares
   (c) Combined patterns circles, exact patterns squares
5. For the circles in the combined patterns view, I find the lines between the circles distracting.
6. The texts are readable:
   (a) Yes
   (b) No
7. It is clear to me what is shown in the combined patterns view (Likert-scale)
8. It is clear to me what is shown in the exact patterns view (Likert-scale)
9. It is clear to me what the grey circles mean in the combined patterns view (Likert-scale)
10. It is clear to me how is dealt with missing data.
    (a) Yes
    (b) No
11. The question tool has clarified me things I did not understand before. (Likert-scale)

The last part of the user evaluation is an open evaluation.

1. What are strenghts of this system?
2. What would you change?
3. Is there something that is missing?
4. Is the visualization Intuitive?
5. Would you like to use the visualization on your own dataset?
   (a) Yes
   (b) No

Table A.1: Sample dataset.

Element	**A**	**B**	**C**	**D**
e1	1			
e2	1			
e3	1			
e4	1			
e5		1		
e6		1		
e7		1		
e8			1	
e9			1	
e10				1
e11	1	1		
e12	1	1		
e13	1	1		
e14		1	1	
e15		1	1	
e16	1			1
e17	1	1		1