Università degli Studi di Catania
DIPARTIMENTO DI MATEMATICA E INFORMATICA

Data Mining and Visual Analytics Techniques

A dissertation submitted to the Department of Mathematics and Computer Science and the committee on graduate studies of University of Catania, in partial fulfillment of the requirements for the degree of doctorate in computer science.

Candidato:
Lorenzo Di Silvestro

Supervisore:
Chiar.mo Prof. Giovanni Gallo

Coordinatore:
Chiar.mo Prof. Vincenzo Cutello

Corso di Dottorato di Ricerca - XXVI Ciclo

Abstract

With the beginning of the *Information Age* and the following spread of the *information overload* phenomenon, it has been mandatory to develop a means to simply explore, analyze and summarize large quantity of data. To achieve this purposes a *data mining* techniques and *information visualization* methods are used since decades. In the last years a new research field is gaining importance: *Visual Analytics*, an outgrowth of the fields of scientific and information visualization but includes technologies from many other fields, including knowledge management, statistical analysis, cognitive science and decision science. In this dissertation the combined effort of the mentioned research fields will be analyzed, pointing out different way to combine them following the best practice according to several application cases.

Contents

Abstract **ii**

Introduction **1**

Dissertation outline 8

1 Social Network Analysis of Criminal Sentences **11**

1.1 Introduction 11

1.2 Data Collection 13

1.3 The Codebook 14

1.4 Information Extraction 16

1.4.1 Background 16

1.4.2 Information extraction for criminal sentences 16

1.4.3 Standard structure of an Italian criminal sentence 17

1.4.4 The finite state transducers 18

1.5 Social Network Analysis 19

1.5.1 Background 19

1.5.2 Social network analysis for criminal sentences 20

1.6 Conclusion 25

2 Information Visualization on Organized Crime Trials **26**

2.1 Introduction 26

2.2 Case study 27

2.2.1 Collecting data 28

2.2.2 Dataset description 28
2.2.3 Information extraction 29
2.2.4 Finding associations 30
2.2.5 Visualizing data 31
2.2.6 Clusters 32
2.3 Results 34
2.4 Conclusion 35

3 Visual Analysis of Multivariate Data from Dairy Industry 36

3.1 Introduction 36
3.2 Related Work 38
3.2.1 Animal Research and Dairy Industry Domain 38
3.2.2 Multivariate Data Visualization 40
3.2.3 Time-Varying Data Visualization 41
3.2.4 Interaction 41
3.3 Application Background 42
3.3.1 305-Days Lactation Yield 43
3.3.2 Test-Day Models 44
3.3.3 Means for Hypothesis Building 45
3.3.4 Multiple Timelines 46
3.4 System Overview 47
3.4.1 Data Conversion and Pre-Processing 48
3.4.2 Scatter Plots 48
3.4.3 Statistic Metrics and Density Plots 49
3.4.4 Multiple Timelines View 50
3.5 Case Study 52
3.5.1 Datasets Description 53
3.5.2 Analysis Process 54
3.5.3 Suggested Features 56
3.6 Expert Evaluation 57

3.6.1 Study Procedure 58
3.6.2 Participants 59
3.6.3 Study Results 60
3.7 Conclusion and Future Work 62

4 Visual Analytics System for Multiple Textual Data Sources 63
4.1 Introduction 63
4.2 Word Clouds 64
4.2.1 Single Clouds Layout Algorithms 66
4.3 RadCloud 70
4.3.1 Previous Works 71
4.3.2 Novel Technique 73
4.4 A Visual Analytics System 76
4.4.1 Input Data Processing 76
4.4.2 Clouds Viewer 77
4.4.3 Entity Exploration 80
4.5 Case Studies 82
4.5.1 Metals Description 82
4.5.2 Jules Verne Selection 84
4.5.3 Wes Andersons' Actors 85
4.6 Conclusions 86

5 Generalized Pythagoras Trees for Visualizing Hierarchies 87
5.1 Introduction 87
5.2 Related Work 89
5.3 Visualization Technique 91
5.3.1 Data Model 92
5.3.2 Traditional Pythagoras Tree 92
5.3.3 Generalized Pythagoras Tree 95
5.3.4 Excursus: Fractal Dimension 99
5.3.5 Visual Parameters 101

5.3.6 Analogy to Node-Link Diagrams 103
5.4 Case Studies 104
5.4.1 Phylogenetic Tree 104
5.5 Discussion 106
5.5.1 Geometry and Perception 107
5.5.2 Readability and Scalability 109
5.5.3 Aesthetics 110
5.6 Conclusion and Future Work 111

List of Figures

I - 1 Rise of the digital information age 2

I - 2 The visualization pipeline 4

I - 3 Visual analytics as a highly interdisciplinary field of research. 6

I - 4 The visual analytics process. 7

2 - 1 Pie diagram for people divided by their roles. 30

2 - 2 Histogram showing number of arches for weight's value. 31

2 - 3 Complete social network from data of sentences. 32

2 - 4 Clusters in boxes. 33

2 - 5 A cluster in detail. 34

3 - 1 Statistics on time-varying multivariate data. 47

3 - 2 System overview. 49

3 - 3 Relation between the quantity of milk and the percentage of protein. 50

3 - 4 The value of produced milk for each day-in-milk (relative timeline). . 52

3 - 5 Differences in lactation curve and fat contents. 56

3 - 6 Fat content for day-in-milk during fall. 61

3 - 7 Scatter plots show the relation between quantity of milk and fat. . . . 62

4 - 1 The effect of missing space space between words 65

4 - 2 Ebbinghaus Illusion 66

4 - 3 Sample result of the spiral layout algorithm. 68

4 - 4 Placing words at selected points on a spiral leads to gaps. 68

4 - 5 Explanation of the space station algorithm. 69

4 - 6 Sample result of the space station layout algorithm. 69
4 - 7 Vertical stack layout and synchronized word selection. 70
4 - 8 Examples of merged clouds 72
4 - 9 Intended position of a word in a 4-categories RadCloud. 75
4 - 10 Ambiguity of positions in RadCloud 75
4 - 11 Merged view, default visualization. 78
4 - 12 RadCloud with details 79
4 - 13 The RadCloud with (b) and without (a) intended position indicators. 80
4 - 14 MuWoC's entity explorer. 81
4 - 15 The source viewer with extracts from a single file. 81
4 - 16 RadCloud of metals' information data, 40 words per category. 83
4 - 17 RadCloud of Jules Verne's novels. 84
4 - 18 RadCloud of actors that worked in Wes Anderson's movies. 85

5 - 1 Extending Pythagoras Trees for encoding information hierarchies. . . 88
5 - 2 Illustration of the traditional Pythagoras Tree approach. 93
5 - 3 Fractal Pythagoras Tree using alternating 35° and 55° angles. 94
5 - 4 Random binary hierarchy visualized as a Pythagoras Tree. 96
5 - 5 Split of Generalized Pythagoras Trees creating an *n*-ary branch. . . . 97
5 - 6 Showing *n*-ary hierarchy: constant width; length stretch function. . . 99
5 - 7 Illustrating the fractal dimension of an *n*-ary branching hierarchy . . 100
5 - 8 Generalized Pythagoras Trees and node-link diagrams. 103
5 - 9 NCBI taxonomy hierarchically classifying species. 105

List of Tables

1.1 Collected Sentences. 14
1.2 JUNG Clustering. Top 5 clusters of initial graph - 204 clusters 21
1.3 JUNG Clustering. Top 5 clusters of final graph - 219 clusters 22
1.4 NodeXL Clustering. Top 5 clusters - 212 clusters 23
1.5 Cluster Tree [Metric 01]. Top 5 clusters - 30 clusters 23
1.6 Cluster Tree [Metric 01]. Top 5 clusters - 210 clusters 24
1.7 Cluster Tree [Metric 02]. Top 5 clusters - 30 clusters 24
1.8 Cluster Tree [Metric 02]. Top 5 clusters - 210 clusters 24

5.1 Exploring different parameter settings. 102
5.2 Comparison of hierarchy visualization approaches. 107

Introduction

"Just because I cannot see it, doesn't mean I can't believe it!"
- Jack Skellington [The Nightmare Before Christmas]

Since 1960 thanks to Miller's psychological studies [90] the term *information overload* has been used to refer to the difficulty that a person faces understanding an issue or taking a decision because of the availability of too much information.

As stated by Keim et al. in [71] this issue increase the probability of getting lost in data which may be:

- irrelevant to the current task at hand;
- processed in an inappropriate way;
- presented in an inappropriate way.

Although computer processing and storage capacity continuously increase, people that have to use the information do not improve their capabilities: human users act as a bottleneck in the process.

The information overload phenomenon has widely spread since the beginning of the so called *Information Age* that finds its roots in the *digital revolution.* Despite the production of digital data increased from the advent of the personal computer in the late 1970, according to Hilbert and López in [58] only in 2002 digital data storage surpassed non-digital for the first time. In 2007, at least 94% of all information on the planet is in digital form. They estimated the world's technological capacity to store, communicate, and compute information, tracking 60 analog and digital technologies during the period from 1986 to 2007. These results have been summarized in Figure I - 1 designed for an article published on *The Washington Post* in February 2011.

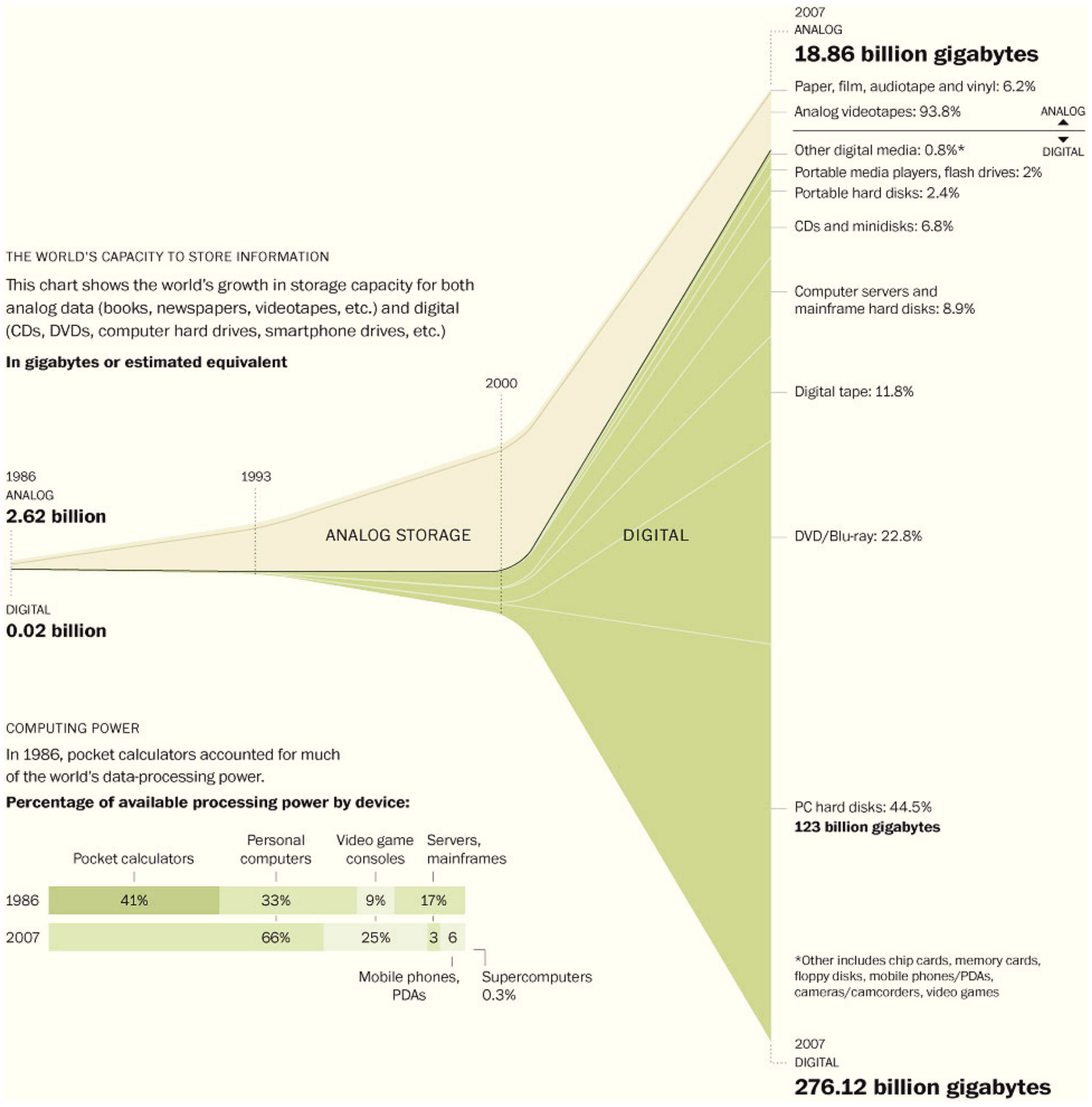


Figure I - 1: Rise of the digital information age. Hilbert and López (University of Southern California) took four years (1986, 1993, 2000 and 2007) and extrapolated numbers from roughly 1,100 sources of information. Credits: Todd Lindeman and Brian Vastag / The Washington Post

It is hence mandatory to develop a means to explore large quantity of data, analyzing them from different perspectives and summarizing it into useful information. The process that allows the user to achieve a better understanding of raw data as been labeled as *data mining*. Data mining is a research field that takes inspiration and techniques from *machine learning* but it differs from it in the intended goal. Machine learning focuses on prediction, based on known properties learned from the training data. Data mining focuses on the discovery of unknown properties in the data. Machine learning has been defined in 1959 by Arthur Samuel as a “Field of study that gives computers the ability to learn without being explicitly programmed”.

Some years later a new research field has emerged: *Information Visualization* is the study of visual representations of abstract data to reinforce human cognition. The modern study of visualization started with computer graphics, however in its early days the lack of graphics power often limited its usefulness. The recent emphasis on visualization started in 1987 with the special issue of Computer Graphics on *Visualization in Scientific Computing* [87].

Information visualization is visualization of abstract data. This is data that has no inherent mapping to space. According to this definition information visualization should be seen in contrast to *Scientific Visualization*, which deals with physically-based data. This kind of data is defined in reference to space coordinates, which makes it relatively easy to visualize in an intuitive way. The space coordinates in the dataset are mapped to screen coordinates. On the other hand visualization of abstract data is not straightforward. One has to find a good way to map data values to screen space [124].

Information and scientific visualization refer to any technique involving the transformation of data into visual information, using a well understood, reproducible process. It characterizes the technology of using computer graphics techniques to explore results from numerical analysis and extract meaning from complex datasets. Visualization is an important tool often used by researchers to understand the features and trends represented in the large datasets.

The process to create a valuable visualization from raw data is called *visualization pipeline* and it's shown in Figure I - 2.

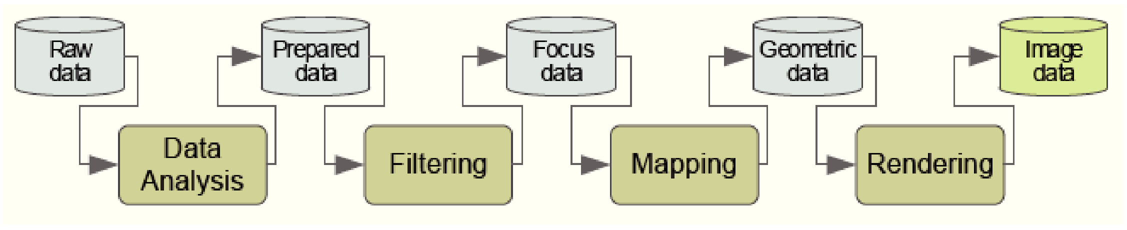


Figure I - 2: The visualization pipeline describes the process of creating visual representations of data.

According to the first formalization of the visualization pipeline by Card et al. [22], the process is made up by four subsequent phases:

1. **Data Analysis**: data are prepared for visualization by applying filters, interpolating missing values, or correcting erroneous measurements. This is the typical preprocessing phase that occurs in every data mining task. Usually there is no user interaction;
2. **Filtering**: selection of data portions to be visualized;
3. **Mapping**: focus data are mapped to geometric primitives (e.g. points or lines) and their attributes (e.g. color, position or size). This is the heart of the visualization process;
4. **Rendering**: geometric data are transformed to image data.

The mapping phase usually represents the most critical step for achieving *expressiveness* and *effectiveness*.

As stated by Card in [21] a visualization is said to be expressive if and only if it encodes all the data relations intended and no other data relations. In other words the expressiveness is achieved if the relevant information of a dataset (and only this) is expressed by the visualization. The term "relevant" implies that expressiveness of a visualization can only be assessed regarding a particular user working with the visual representation to achieve certain goals.

According to Mackinlay [84] the effectiveness criteria identify which graphical language, in a given situation, is the most effective at exploiting the capabilities of the output medium and the human visual system. Since perception, and hence the mental image of a visual representation, varies among users, effectiveness is user-dependent. Nonetheless, some general rules for effective visualization have been established in the visualization community.

As defined by Cook et al. in [33] *visual analytics* is an outgrowth of the fields of scientific and information visualization but includes technologies from many other fields, including knowledge management, statistical analysis, cognitive science and decision science. The term "visual analytics" gained importance shortly after the publication of *Illuminating the Path: The Research and Development Agenda for Visual Analytics* by Cook and Thomas in 2005 [120], in which visual analytics was initially proposed as a means to help United States intelligence analysts meet the challenge of dealing with the masses of security-related information made available to them following the terrorist attacks on September 11, 2001, on the World Trade Center and Pentagon. In [120] visual analytics is defined as the science of analytical reasoning facilitated by interactive visual interfaces. Furthermore, the authors describe the main motivations of using this approach:

> "People use visual analytics tools and techniques to synthesize information and derive insight from massive, dynamic, ambiguous, and often conflicting data; detect the expected and discover the unexpected; provide timely, defensible, and understandable assessments; and communicate assessment effectively for action."

Then they describe visual analytics as a multidisciplinary field that include the following focus areas:

1. analytical reasoning techniques that let users obtain deep insights that directly support assessment, planning, and decision making;
2. visual representations and interaction techniques that exploit the human eye's broad bandwidth pathway into the mind to let users see, explore, and understand large amounts of information simultaneously;
3. data representations and transformations that convert all types of conflicting and dynamic data in ways that support visualization and analysis;

4. techniques to support production, presentation, and dissemination of analytical results to communicate information in the appropriate context to a variety of audiences.

According to the assertions by Keim et al. [74] visual analytics is more than just visualization and can rather be seen as an integrated approach combining visualization, human factors and data analysis (Figure I - 3). With respect to the field of visualization, visual analytics integrates methodology from information analytics, geospatial analytics, and scientific analytics. Human factors (e.g. interaction, cognition, perception, collaboration, presentation and dissemination) especially play a key role in the communication between human and computer, as well as in the decision making process.

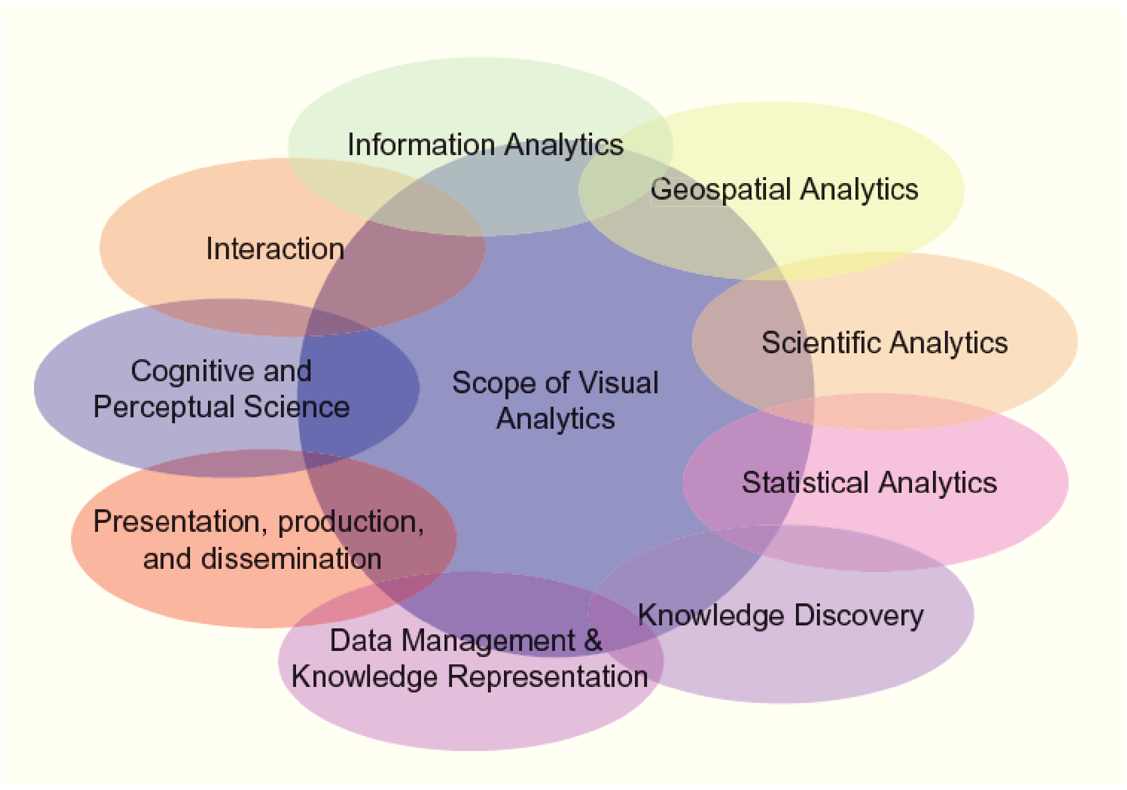

Information Analytics
Geospatial Analytics
Interaction
Scientific Analytics
Cognitive and Perceptual Science
Scope of Visual Analytics
Statistical Analytics
Presentation, production, and dissemination
Knowledge Discovery
Data Management & Knowledge Representation

Figure I - 3: Visual analytics as a highly interdisciplinary field of research.

The visual analytics process combines automatic and visual analysis methods with a tight coupling through human interaction in order to gain knowledge from data [73].

Analyst applies visual or automatic analysis methods. Automated analysis uses data mining techniques to generate models of the original data that can be visualized for evaluation and refinement. Alternating between visual and automatic methods is typical for the visual analytics process. Findings in the visualizations can be used to steer model building in the automatic analysis (Figure I - 4).

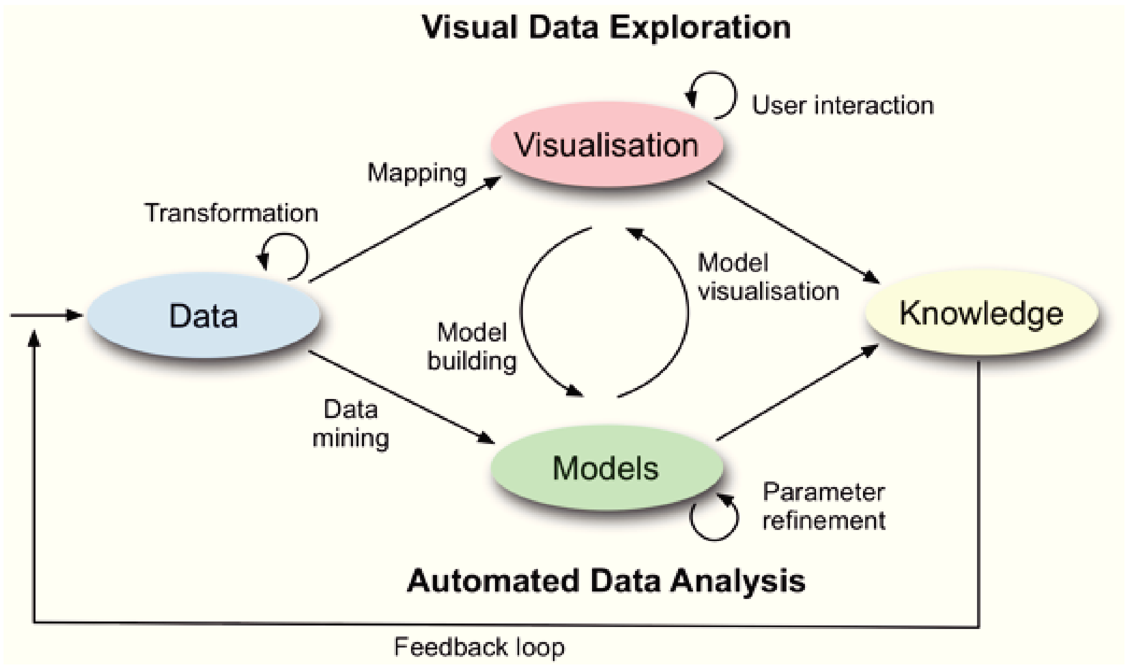


Figure I - 4: The visual analytics process.

As previously explained visual analytics builds on a variety of related scientific fields. At its heart, Visual Analytics integrates Information and Scientific Visualization with Data Management and Data Analysis Technology, as well as Human Perception and Cognition research [71].

Following this definition visual analytics can be seen as an evolution of the well-established cooperation of data mining and visualization since their first application in the modern digital era. In this dissertation the combined effort of the mentioned research fields will be analyzed, pointing out different way to combine them following the best practice according to several application cases, as explained in the following section.

Dissertation outline

In this dissertation we used different approach to face several real-world data problems. In this outline each approach will be summarized by identifying which technique has been used. We combined methods from Data Mining (DM) and Visualization (VIS) to exploit, then, the Visual Analytics process (VA).

Information Extraction and Social Network Analysis of Criminal Sentences [DM]

The first chapter reports a research based on analyzing criminal sentences on organized crime activities in Sicily pronounced from 2000 through 2006. For this case study we perform the analysis of the textual corpus by means of pure data mining techniques. Information have been extracted from text files obtained by OCR on the original papers collected in different Sicilian courthouses. The text files were parsed in order to extract the names of the actors involved in the facts and the relationships between them. The actors have been labeled with their role: judge, members of the court, prosecutor, defendants, lawyers. Relationships between actors were also extracted. Then, we modeled in a social network like style the information obtained in the previous stage. The social network has been analyzed using the JUNG Java library. In particular, the network has been inspected, in order to detect central nodes and sub-communities.

Information Visualization on Organized Crime Trials [DM for VIS]

In this chapter we report of our initial experiments and efforts to use InfoVis to make a specialized corpus of textual information more accessible and useful. In particular we apply graph visualization to a collection of organized crime sentences. The final objective of this research is to provide to the crime analysts a tool to pool existing information into an organized database in order to gain a better understanding and forecasting of crimes. Data mining has been used on the crime trials dataset to prepare data for a visualization phase performed to understand and to manage large amounts of textual data. We report the results obtained in so far clustering and visualizing data.

Visual Analysis of Time-Dependent Multivariate Data from Dairy Farming Industry [VIS for DM]

The third chapter addresses the problem of analyzing data collected by the dairy industry with the aim of optimizing the cattle-breeding management and maximizing profit in the production of milk. Interactive information visualization solution are used to improve and simplify any further data mining approach on this kind of data.

The amount of multivariate data from daily records constantly increases due to the employment of modern systems in farm management, requiring a method to show trends and insights in data for a rapid analysis. We have designed a visual analytics system to analyze time-varying data. Well-known visualization techniques for multivariate data are used next to novel methods that show the intrinsic multiple timeline nature of these data as well as the linear and cyclic time behavior. A qualitative expert user study conducted with animal researchers shows that the system is an important means to identify anomalies in data collected and to understand dominant data patterns, such as clusters of samples and outliers. The evaluation is complemented by a case study with two datasets from the field of dairy science.

Analyzing Textual Data by Multiple Word Clouds [VIS+DM=VA]

In the fourth chapter a complete visual analytics approach is presented. This section contains the fundamental ideas and provides insight into the development of a system for advanced text analysis. The visual analytics system MuWoC (Multiple Word Clouds) is based on a novel technique for the creation and visualization of word clouds. Word clouds are a good way to represent textual data with meta information. However, when multiple data sources have to be analyzed, word clouds suffer of poor comparability. To face this problem we propose a novel approach to create and visualize multiple word cloud. The so called RadCloud (Radial Cloud) merges multiple standard clouds into a single one, while retaining the information about the origin of the words. This is the core of our visual analytics approach, the novel visualization is used to access in an interactive environment to the original text that are processed with well-known data mining methods.

Generalized Pythagoras Trees for Visualizing Hierarchies [VIS]

The last chapter a novel information visualization technique is presented. Pythagoras trees are fractals that have been used to visually encode binary hierarchies. This visual encoding is a restriction for arbitrary branching hierarchies such as file systems or phylogenetic trees that are also branching into n subhierarchies. Although each hierarchy can be modeled as a binary one by subsequently dividing n-ary branches into a sequence of $n-1$ binary branches we follow a different strategy. In our approach we only allow a unique visual element for the n-ary branching instead of spreading the single binary branches along a major strand. Our novel technique extends the Pythagoras tree metaphor to arbitrarily branching trees by using convex polygons where the sides are used for placing subhierarchies in a recursive manner. We compare our approach with existing tree visual metaphors.

Chapter 1

Social Network Analysis of Criminal Sentences

"Leave the gun. Take the cannoli."
- Peter Clemenza [The Godfather]

The following contents are object of the paper *Information Extraction and Social Network Analysis of Criminal Sentences* by D. De Felice, L. Di Silvestro, G. Gallo, G. Giuffrida, G. Giura, C. Pennisi and C. Zarba, presented at the *Social Media: Risks and Opportunities in Military Applications (HFM-201) 2011.*

1.1 Introduction

In this chapter we study the juridical response to organized crime activities in Sicily, by analyzing a corpus of criminal sentences using techniques of information extraction and social network analysis. In particular, the analyzed criminal sentences were pronounced in the four courthouses of Sicily from 2000 through 2006, and were declared irrevocable for at least one defendant.

Although the corpus of criminal sentences is limited in both time and space, the results of the analysis are significant for three main reasons:

- there is not yet in the literature a comparative analysis of criminal sentences pronounced at the four Sicilian courthouses or in the rest of Italy: the present one is hence at the best of our knowledge a seminal work;
- there is not yet a digital database collecting data on the institutional response to the phenomenon of organized crime activities in Sicily: the present work may hence be used as a starting experience toward the creation of such knowledge base;
- in the case of organized crime activities, the Sicilian jurisprudence de-facto orients the Italian jurisprudence and it is hence relevant to better investigate the internal working of this activity.

The main objective of the analysis is to obtain a description of the socio-economic environment characterizing the trial leading to the criminal sentence, as well of the differences in the conduct of the trial between the different courthouses. Our research is composed of three main stages. In the first stage, we collected the criminal sentences from the courthouses of Sicily. Since there is not yet a digital archive of criminal sentences in Sicily, all sentences had to be collected in their paper format. The paper sentences have been scanned into PDF files, and then converted into TXT files by means of OCR technology. Furthermore, we have identified a codebook, which is basically a collection of well-thought variables to be devised from the text of each criminal sentence. In the second stage, the text files were analyzed using information extraction technology, in order to extract from the text of the sentences the actors involved in the facts and the relationships between them. In particular we extracted the judge, the members of the court, the prosecutor, the defendants, the lawyers, and the other people involved in the sentence facts. Relationships between actors were also extracted. The information extraction has been performed by implementing opportune finite state transducers (FST), which are automatons capable to recognize specific patterns in an input string. In the third stage, we constructed a social network using the information obtained in so far. The social network consists of a set of nodes and a set of edges. A node is any actor extracted in the second stage of the research.

An edge is a relationship between actors. The social network has been analyzed using the JUNG Java library . In particular, the network has been inspected in order to detect central nodes with high betweenness centrality. A node has high betweenness centrality if there are many shortest paths in the network that intersect the node. These nodes should be relative to pivotal character of the trials. Finally, the network has been inspected using a specialized clustering algorithm in order to detect community structures.

1.2 Data Collection

This research restricts in the analysis of criminal sentences to those sentences satisfying the following criteria:

- the sentences are relative only to mafia crimes: these are those encompassed by the Italian criminal procedure code, article 51, comma 3 bis;
- all sentences were declared final and irrevocable for at least one defendant;
- all sentences have been pronounced by Sicilian judicial authorities from January 1, 2000 to December 31, 2006.

Since in Italy there is not yet a unified digital database of criminal sentences, we needed to perform a complex and time-consuming data collection activity in order to gather the criminal sentences required for our analysis. Specifically a preliminary interrogation to the Italian computerized archive RE.GE. has been performed. This interrogation has been parameterized with the criteria above and produced a list of about 1,200 criminal sentences satisfying our query. Then a formal request to the applicable Sicilian courthouses has been made in order to gain physical access to the paper printed criminal sentences. After the authorizations were granted, we went physically to the various Sicilian courthouses, and xeroxed the files. Eventualy we collected 1,147 sentences; we performed an extensive manual data quality verification process. Due to misleading classification in the RE.GE. archive, only 726 criminal sentences really satisfied all our criteria. For instance, about 90% of the non-pertinent criminal sentences were initially recorded in the RE.GE. as concerning illicit drug smuggling. These sentences were later assessed as concerning the less grave crime of art. 73 of the same DPR, but without this correction being made in the RE.GE. The entire set of criminal sentences is made of about 55,000 pages, and the sentence length varies from a minimum of 2 pages to a maximum of 3,268 pages. Every page has been scanned producing PDF files, and processed with OCR technology in order to produce TXT files suitable to automatic computerized analysis. The entire process described was extremely time consuming, and it took more than two man-years work. The result of the process represents the first example in Italy of a large digital archive of criminal sentences. The collected sentences can be classified according to degree of judgment and proceedings format, as shown in the following table, where the rows indicate the degrees of judgments in the Italian judicial system, while the columns indicate the proceedings format.

Authority	**Standard**	**Abbreviated**	**Plea**	**Total**
GIP/GUP	23	135	118	276
Tribunale	122	9	7	138
Corte d'Assise	90	10	59	161
Corte d'Appello	5	1	0	6
Corte d'Assise d'Appello	71	4	70	145
Total	311	159	254	726

Table 1.1: Collected Sentences.

1.3 The Codebook

While analyzing the collected criminal sentences, we specified a codebook, which is a tabular tool whose purpose is to organize the results of the analysis. More precisely, the codebook consists of a table that contains one row for each criminal sentence. The columns denote classificatory variables that describe the features of the criminal sentences that are important for the analysis [15]. The process leading to the specification of the codebook started with an initial apriori definition of the variables. Then, each sentence has been read by human experts and analyzed in order to fill the codebook. While the criminal sentences were analyzed, the specification of the variables of the codebook has been gradually refined. Many sentences had to be reanalyzed several times. This manual approach is satisfactory to meet the designed goals, but it definitively limits additional analysis based on different variables. The lesson learned at this stage has been, hence, that an automatic process to harvest variables from free text is imperative in order to avoid long repetitive manual processes. At the end of the analysis, the final codebook is made of 44 variables belonging to four dimensions: temporal, procedural, social, and environmental. The temporal dimension includes variables describing the durations of each phase of the trial process, starting from the registration to the RE.GE., including the pronouncements of the verdicts at each degree of judgment, and terminating with the final declaration of irrevocability of the sentence. Therefore, the total duration of the trial process can be ascertained. The procedural dimension includes variables describing the legal events occurring during the temporal dimension. These events include custody measures, proceedings formats, contested crimes, modifications and integration of contested crimes, recognition of extenuating circumstances, and the final verdict. The social dimension includes variables describing the occupation or profession of the defendants, as well as their social and economic conditions. The environmental dimension includes variables describing the geographic, political, institutional, and political aspects of the events discussed in the sentence. They also identify the economic sector that is harmed by the contested crimes, and report the official quantification of the economic cost suffered because of the contested crimes.

1.4 Information Extraction

1.4.1 Background

Information extraction is generally the process of extracting structured data from unstructured ones. In this research it has been mostly the extraction of relational data from natural language documents [106]. Typically, given a document written in natural language, there are four kinds of information that can be extracted: entities, attributes, relations, and events. Entities can be individuals, things, dates, or measurements. Attributes are features associated to entities. For instance, an individual has attributes like birthdate, birthplace, profession, education, title, telephone number, email address. Relations are associations between entities. Events are relations where time is of primary importance. There are two main approaches to information extraction: deep and shallow. Deep information extraction is based on natural language processing. Information is extracted from the document by lexical analysis, semantic analysis, and the interpretation of the discourse [64]. Deep information extraction is quite effective, but too slow computationally. Furthermore, the (manual) construction of the model necessary to carry out the interpretation of the discourse is complex and laborious. Shallow information extraction does not aim at a human-like comprehension of the document, but aims only at the filling of the relational tables. This is done using a pipeline consisting of a finite number of finite state transducers (FSTs). A finite state transducer takes a sequential input and, if some conditions are verified, returns an output that depends on the input and on the internal state of the transducer [13]. Essentially, a finite state transducer performs a simple linguistic task. The idea is that a finite number of simple linguistic tasks is sufficient in order to fill the relational tables.

1.4.2 Information extraction for criminal sentences

We have implemented an automated analyzer that performs information extraction from our corpus of criminal sentences. Given a criminal sentence, our analyzer extracts the following entities representing individuals: judges, members of the court, defendants, lawyers, prosecutors, and other people involved. Our analyzer also extracts the crimes mentioned in the criminal sentence. Furthermore, it extracts, for each defendant, the lawyers(s) that represent them, and whether the defendant is convicted or acquitted. Finally, our analyzer extracts associations between entities representing individuals, by detecting when two distinct individuals co-occur in the same phrase of a criminal sentence. The extraction is performed by means of a pipeline of finite state transducers, and exploits the fact that Italian criminal sentences are written following a standard structure. We next describe this standard structure, and afterwards we describe the finite state transducers implemented.

1.4.3 Standard structure of an Italian criminal sentence

An Italian criminal sentence always start with the denomination of the legal authority, and the wording "Repubblica Italiana"', followed by "In nome del popolo italiano". The names of the members of the court follow. The first name mentioned is always that of the judge. The other names are the other members of the court. Then another section starts, where the defendants are listed. Each defendant has to be properly identified by his/her biographical data such as birthplace and birthday. In the criminal sentence, the defendant name is always followed by the wording "nato a" (i.e., born in). The name of each defendant is followed by the name(s) of the defending lawyer(s). The name of each lawyer is preceded by the title "avv.". The name of the prosecutor is preceded by the acronym "PM". The first name that is not preceded by "avv." or "PM", and is not followed by "nato a" indicates the end of the defendant list, and this name is an involved part in the events discussed by the sentence (for instance, it could be an injured party or a witness). The verdict of the sentence is always preceded by the acronym "PQM" or "PTM". For first-degree sentences (GIP/GUP), each defendant is either convicted or acquitted. In second-degree sentences, before the acronym PQM/PTM and after the defendant list, the first-degree verdict is described. Then, after the acronym PQM/PTM, it is explained how the first-degree verdict is modified.

1.4.4 The finite state transducers

We now list and describe the finite state transducers implemented in order to perform information extraction on our corpus of criminal sentences. People-FST. This transducers uses a dictionary of Italian first names and family names in order to recognize individuals. If necessary, the user can extend the dictionary. An individual is considered as a sequence of at least two names, or as a capital letter followed by a point, a space, and a name. Defendants-FST. Each defendant is always accompanied by its birthplace and birthday. If an individual is followed by the wording “nato a”, then we assume that he/she is a defendant. Lawyers-FST. Lawyers are individuals preceded by their title “avv.”. Judge-FST. It at this point the first individual appearing in the text of the sentence is not a defendant, then it must be the judge. If the first individual is a defendant, then the information about the judge is unavailable. Court-FST. If the name of the judge has been extracted, then all individuals comprised between the judge and the first defendant must be members of the court. Prosecutor-FST. The prosecutor is an individual preceded by the abbreviation “PM”. Other-FST. At this point, all individuals that are not the judge, members of the court, assistants, defendants, or lawyers, are categorized as “other people involved”. Defendants-lawyers-FST. This transducer associates each defendant to the list of lawyers that represent him/her. Crimes-FST. This transducer recognizes the crimes disputed in the trial using regular expressions. Verdict-FST. This transducer attempts to the deduce if a defendant has been condemned or absolved. This is done by analyzing the text of the sentence following the acronym “PQM” or “PTM”, and looking for words such as “condonna” or “assolve” written before the name of the defendant. Associations-FST. This transducer detects when two individuals co-occur in the same phrase of a criminal sentence. When this happens, we say that there is an association between the two individuals. These associations are then used in order to construct a graph, which will be then analyzed using techniques of social network analysis, as described in the next section.

1.5 Social Network Analysis

1.5.1 Background

A social network is a graph whose nodes are actors, and whose edges represent social relationships. The actors can be individuals or organizations. The social relationships are ties of various nature, such as friendship, kinship, or business connections. The study of social networks is useful in order to explain various social phenomena, such as the role of husband and wife in a household, causes of diffusion of epidemic illnesses, organization and behavior of terrorist cells. Social network analysis is modelled using the mathematical theory of graphs, in order to define and quantify various measures of social capital that show the benefit obtained and the role taken by the individuals in the social networks to which they belong [42]. These measures of social capital are, in general, centrality measures: they measure the importance of the individual and their capability to connect the network. In general, the term "centrality" can be interpreted as influence, prestige, or control. It is possible to highlight the centrality of an actor in a specific context simply using the various centrality measures. A node in the network has a certain importance depending of which centrality measure is used. The importance of the role of an actor in a social network can for instance be quantified by measuring the number of connections of the node (degree centrality), the proximity of the node to all the other nodes (closeness centrality), or the potentiality that the node has to intermediate the knowledge flow of the network (betweenness centrality). The last definition of centrality is among the most expressive. Nodes of a social network with high betweenness, commonly denoted as brokers, have great influence on the information flow travelling through the network. Formally, the degree of a node n in a graph G is the number of edges incident in n. The proximity of a node n in a graph G is the average distance between n and all nodes of G reachable from n. The betweenness centrality of a node n in a graph G is obtained by summing, for each pair of distinct nodes s, t, the ratio between the following quantities: (a) the number of minimal length paths between s and t that pass through n, and (b) the number of minimal length paths between s and t. An important task performed by social network analysis is the discovery and analysis of community structure inside a social network [28]. This task is the division of the network into disjoint or possibly overlapping groups, such that the groups found have a high clustering coefficient. The clustering coefficient of a group is given by the average of the local clustering coefficients of the actors in the group. The local clustering coefficient of an actor is the fraction of the actor's neighbors that are also neighbors of each other. Several practical applications of social network analysis are useful in order to explain different social phenomena, such as the diffusion of epidemic illnesses, the study of the behavior and roles of terroristic cells (a research supported by the government of the United States), and the study of the behavior of the employees of private companies in order to predict the performance of the company itself.

1.5.2 Social network analysis for criminal sentences

We have performed social network analysis to our corpus of criminal sentences. More precisely, we have analyzed a network graph generated using the information extracted with our finite state transducers. The network graph generated consists of a set of nodes and a set of edges. The nodes are the individuals extracted with the People-FST, Defendants-FST, Lawyers-FST, Judge-FST, Court-FST, Prosecutor-FST, and Other-FST finite state transducers. The edges are the associations extracted with the Association-FST finite state transducer. We recall that the Association-FST finite state transducer detects an association between two individuals when they co-occur within the same phrase of a criminal sentence. The generated graph contains 3,370 nodes and 30,489 edges. Each edge has a weight. More precisely, an edge between two individuals has a weight equal to the number of times that the two individuals co-occur in the same phrase. In the generated graph, the weight of edges varies from a minimum of 1 to a maximum of 613. The generated graph is very large, and this makes the social network analysis challenging, and the graphical visualization of the graph even more challenging. In order to ease the analysis, we have pruned the graph by removing those edges whose weight is smaller than 5. Removing these edges eases the social network analysis, and is a reasonable choice. In fact, if two individuals are not together in a phrase more than few times, we can assume that the co-occurrence is not significant for our analysis. The pruned edges represent 86.57% of the total number of edges. After pruning the edges and retaining only the nodes connected by the remaining edges, the graph consists of 1,390 nodes and 4,095 edges. We used three different clustering algorithms to highlight groups of entities closely connected. At first we have analyzed the pruned graph using the JUNG Java library. JUNG, which stands for Java Universal Network/Graph Framework, provides implementations of a number of algorithms from graph theory, data mining, and social network analysis, such as routines for clustering and computation of centrality measures. One of the clustering algorithms implemented in JUNG is an iterative algorithm based on edge betweenness [47]. At each iteration, the algorithm computes the edge betweenness for all edges in the graph, and then removes the edge with the highest edge betweenness. At the end of the process, the clusters detected are the connected components of the final graph. We have applied this clustering algorithm to our pruned graph, using 50 iterations. In the initial graph we find out 204 clusters while after the last iteration of the algorithm there are 219 clusters. The results of the clustering analysis are summarized in the following tables, where the first table contains the sizes and clustering coefficients of the largest five clusters in the initial graph, and the second table contains the sizes and clustering coefficients of the largest five clusters after removing 50 edges with highest betweenness centrality from the initial graph.

Size	**Clustering Coefficient**
186	0.7184
183	0.6287
34	0.6063
34	0.8365
32	0.7814

Table 1.2: JUNG Clustering. Top 5 clusters of initial graph - 204 clusters

Size	**Clustering Coefficient**
58	0.6927
56	0.7101
38	0.8618
36	0.7629
34	0.6063

Table 1.3: JUNG Clustering. Top 5 clusters of final graph (50 edges removed) - 219 clusters

Note that the initial graph contains clusters that are larger than the ones in the final graphs. Moreover, in general, as the edges are removed by the clustering algorithm, the clusters become more cohesive. The cohesion of a cluster is measured by its clustering coefficient, defined as the average of the local clustering coefficient of each node of the cluster. The local clustering coefficient of a node is defined as the fraction of the node's neighbors that are also neighbors of each other. For the second clustering algorithm we used a powerful free and open-source tool developed by the Social Media Research Foundation. NodeXL was created by Marc Smith's team while he was at Microsoft Research [116]. It is a template for Excel that allows to easily build a graph entering a network edge list. We used Clauset-Newman-Moore algorithm [28]. It is widely used in the community of complex network researchers, and was originally designed to analyze the community structure of extremely large networks. It is a hierarchical algorithm based on the modularity measure. Modularity is a quality index for clustering [12]. Modularity represents the number of edges falling within groups minus the expected number in an equivalent network with edges placed at random [93]. At each step the algorithm calculates the modularity increase for every possible join in the network then select the join that maximizes the increase in modularity and merge both communities. We have to repeat these steps until there's only one community. In this case we find out 212 clusters. Details of this clustering analysis is shown in the following table.

Size	**Clustering Coefficient**
101	0.6426
61	0.7177
51	0.8234
45	0.7250
40	0.5033

Table 1.4: NodeXL Clustering. Top 5 clusters - 212 clusters

For the third experiment we use a MATLAB implementation of an agglomerative hierarchical cluster tree. We use the complete linkage function, also called furthest neighbour, to build the cluster tree. It uses the largest distance between objects. So we had to calculate the distance between every entities in our dataset. We use two different metrics. In our first approach we calculate the distance considering the phrase location inside the text of the criminal sentence and counting the number of paragraphs that occurs between two entities. If two entities co-occur in the same phrase their distance is set to 0, while if they occur in two different criminal sentences the distance is set to a maximum value of 400 (we empirically find out that the number of phrases in a sentence is always lower than this value). We built a symmetric square matrix with 1390 rows. In the table below clustering results are shown.

Size	**Clustering Coefficient**
861	0.6387
38	0.5603
34	0.6949
29	0.8525
27	0.8581

Table 1.5: Cluster Tree [Metric 01]. Top 5 clusters - 30 clusters

The algorithm identifies 30 clusters in our network. The number of clusters is very small in comparison with the number of clusters highlighted by other algorithms we used. So we decide to force the algorithm to find out 210 clusters to make results more readable and easy to compare with our previous results.

Size	**Clustering Coefficient**
184	0.3206
27	0.8416
26	0.5968
23	0.8953
23	0.8830

Table 1.6: Cluster Tree [Metric 01]. Top 5 clusters - 210 clusters

We then used another approach to calculate distances between entities. We chose a similar method to the one used in text mining to represent a text in a very light form, called bag-of-word representation. In our model every entities is represented by a vector containing information on occurrences of the individuals. We extract a subset of phrases from our dataset, considering only paragraph in which occurs one or more entity. In this way we built 1390 vectors of 15865 cells filled with 0 if the entity is not present in the related paragraph otherwise is filled with the number of occurrences of the entity. Then we compute a pairwise euclidean distance between every couple of vector. In the tables below we report results for 30 and 210 clusters.

Size	**Clustering Coefficient**
1335	1.0000
5	1.0000
4	1.0000
3	1.0000
3	0.4929

Table 1.7: Cluster Tree [Metric 02]. Top 5 clusters - 30 clusters

Size	**Clustering Coefficient**
889	0.5841
21	0.6833
17	0.9811
11	0.7863
10	0.8382

Table 1.8: Cluster Tree [Metric 02]. Top 5 clusters - 210 clusters

Using this particular representation of entities and occurrences in phrases the agglomerative hierarchical tree is able to highlight a very large cluster of 1335 nodes with a maximum clustering coefficient. When we force the algorithm to find out more clusters we don't obtain the same results but they are comparable to previous execution of the same algorithm. This show that the simpler representation we used speeds up the algorithm execution maintaining reliability in results.

1.6 Conclusion

We have presented an analysis of criminal sentences on criminal trials on organized crime activity in Sicily pronounced from 2000 through 2006. The analysis was composed of three main stages. In the first stage we collected the criminal sentences in their paper format, and converted them in digital format by means of OCR technology. In the second stage we performed information extraction, in order to extract from the text of the sentences the actors involved in the sentences, and the relationships between them. In the third stage we used the information extracted in the second stage in order to build a social network. The social network was then anaylized in order to detect community structure. The main challenge of this research is the difficulty to perform information extraction on the corpus of available sentences. The difficulty stem mainly from the presence in the corpus of many errors done by the OCR software. Moreover, some of the sentences contain not only typed text, but also some handwriting, and the OCR is unable to satisfactorily process the handwriting. Future progress of this research will therefore need to use more sophisticated OCR tools, specifically tailored to the kind of criminal sentences that need to be analyzed.

Chapter 2

Information Visualization on Organized Crime Trials

"Every shadow no matter how deep is threatened by morning light."

- Izzi Creo [The Fountain]

The following contents are object of the paper *Information Visualization on Organized Crime Trials* by L.Di Silvestro, G. Gallo, G. Giuffrida and C. Zarba, presented at the *Eurographics Italian Chapter Conference 2011.*

2.1 Introduction

Since 1960 thanks to Miller's psychological studies [90] the term *information overload* has been used to refer to the difficulty a person faces understanding an issue or taking a decision because of the availability of too much information. In *digital age* an increasing number of people are connected to the Internet, they can use data and create news as well. They become active writers and produce more data for other viewers. Thousands of pages of text data are produced daily: now more than ever information overload is becoming a serious problem. It is easy to access information but too much of it is hard to manage and to understand and this can lead to misinformation. We need to find out efficient systems to manage large amount of information, exploring and analyzing the huge flow of new data gathered so far. Managing, exploring, and analyzing the flow of data are among the most important tasks for scholars of various disciplines.

One of the most efficient method to handle large amount of data and make them simpler to understand for people is using the visuo-spatial reasoning abilities of humans. [122] It is clear that visualization is the key for content analysis. In this scenario a new field of research has been developed, to design and study interactive visual representation of abstract data. Information Visualization (InfoVis) is a rapidly growing field that is emerging from research in human-computer interaction, computer science, graphics, visual design and psychology [114]. Text visualization is considered one of the big challenges of the newly defined field of visual analytics [56] [96] [55].

In this chapter we report of our initial experiments and efforts to use InfoVis to make a specialized corpus of textual information more accessible and useful. In particular we apply graph visualization to a collection of organized crime sentences. The final objective of this research is to provide to the crime analysts a tool to pool existing information into an organized database in order to gain a better understanding and forecasting of crimes. However, since we are still at the begin of this project our aim for the present is to gain some know-how about the major issues related to this specialized field.

2.2 Case study

Our data come from the legal domain. Legal scholars and social scientists are often interested in information extraction from a large number of texts. They need to analyze a very large amount of data to find out useful information to formulate and to verify social theories.

A valuable source of information about organized crime are the official trials' documentation. Empirical studies on whole trials are not, up today, practical, due to the huge size of the complete trials' documentation. It is hence wise to restrict the analysis only to the final sentences. This is reasonable because a sentence contains all relevant elements that allow judges to take a decision. Indeed reading it, we can reconstruct the decision process. From sentences moreover, we can extract data to find out statistical results on age, genders, locations, etc. on criminal activities.

2.2.1 Collecting data

Sociologists are interested in studying organized crime. In Sicily organized crime is largely connected to mafia affairs. Italy has yet no central digital database of past sentences. This makes expensive and difficult to gather this kind of data. Since sociologists' interest in this topic is high, a research group in Catania decided to invest into this data gathering activity.

It took about 30 months/man to get together all the information to create the dataset used in this study. The gathering of sentences have been done in the archives of all the major appellate justice courts in Sicily where the trials have been conducted.

Every paper sheet of the sentences has been xeroxed. The entire set is made of about 55000 pages (sentences length goes from 2 to 3268 pages). Every page has been scanned producing PDFs files; after that, an OCR system has been used to extract textual information. A set of (unchecked) text files has been produced [34].

This expensive work makes the dataset very interesting and important to use for social studies, because it represent the only example in Italy of digitalized crime sentences corpus on mafia topics.

2.2.2 Dataset description

Our dataset collects all criminal sentences of trials on crime activities in Sicily pronounced from 2000 to 2006. In this set are included only crime sentences that became definitive in that years, about mafia and drug dealing cases. According to these principles, 721 sentences have been included in our study.

These text are obtained by using OCR system on a PDF copy of the original papers of sentences. Those papers are often written using typewriters, so some characters are difficult to read and to recognize. Sometimes there are handwritten notes on the sheets that are obviously not recognized. For these reasons there are a lot of characters with no meaning in our digital text. This makes our work for automatic information extraction harder.

2.2.3 Information extraction

Information extraction is a method to obtain structured data from unstructured natural language texts. With these techniques we can extract four type of information: entity, attributes, associations, and events. Entities can be persons, objects, dates and measures. Attributes are characteristics of entities, like birthday and birthplace of people, or their job. Associations are relationships between two entities that link them to each other. Events are associations for which temporal dimension is important.

For this preliminary work about visualization data to make easier the work of sociologists and jurists we decide to use mainly entities. Among entities we decide to extract only those that represent people, leaving out places and other entity types. During extraction we use context to understand the role of the person we have just found in the text. Four kinds of roles are recognized: ”prosecutor”, ”judge”, ”lawyer” and ”defendant”.

Several finite state transducers (FST) are used to scan every document in sentences corpus. An FST is an automaton able to recognize specific patterns in an input string [10]. An automaton is build for each role we wish to identify in the text. For example layers are simple to recognize because their name come after an exact string that in Italian language denote their qualification (i.e. ”`avv.`”, ”`avv.to`”, ”`avv.ti`”, ”`avvocato`”, etc.).

During data extraction some important information about name’s position is saved. For every occurrence of a name we know the unique id of the paragraph in which the name is found and the id of the sentence.

In the whole dataset there are 2475 entities referring to people. A pie chart of the four roles for persons is shown in Figure 2 - 1.

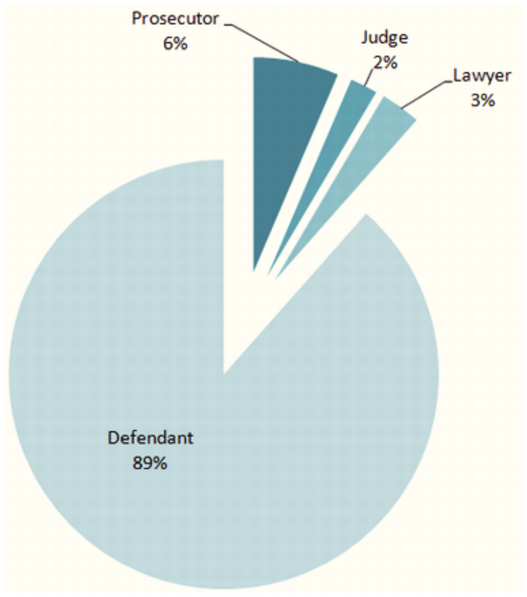


Figure 2 - 1: Pie diagram for people divided by their roles.

2.2.4 Finding associations

We use data on entities and their location to find out a correlation among them. For our experiments a co-occurrence relationship is defined. Two entities are related if they appear in the same paragraph or in the same sentence. A Python script is used to identify associations. If a couple of entities co-occurs more than one time, the association between them is weighted accordingly. There are 32537 association. The weight for an association go from 1 to 613. The simplest way to show entities and their relationships is to create a graph in which vertices represent entities and arches between vertices represent the relationship of co-occurrences.

It is very tricky to build and to show a readable and easy to handle graph with more than ten thousand arch, so we choose to prune arches with lesser relevance. As diagram in Figure 2 - 2 shows, the arches with weight smaller than 5 are about 87,12%. Those arches represent a weak relation between entities, representing a very rare co-occurrence of names in the same sentence; if two person are not together in a paragraph more than few times, we can assume that the co-occurrence is not significative for our goals.

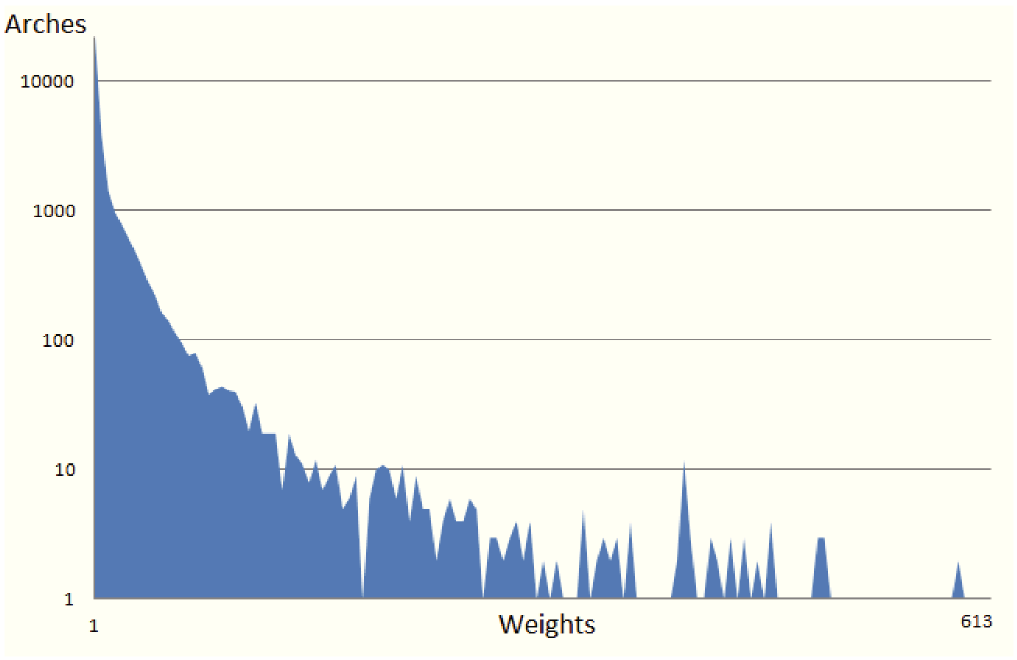


Figure 2 - 2: Histogram showing number of arches for weight's value.

If we maintain only arches with a weight grater or equal to 5, we have 4191 arches and 1436 entities connected in our graph.

2.2.5 Visualizing data

To build a graph with entities extracted from sentences in this initial study, we used a powerful free and open-source tool developed by the Social Media Research Foundation. NodeXL was created by Marc Smith's team while he was at Microsoft Research [115]. It is a template for Excel that allows to easily build a graph entering a network edge list. With this tool it is not difficult to filter vertices and edges or calculate some graph metrics. We use this tool to obtain a clustering on our data. To identify vertices that are clustered together into subgroups of interest is indeed of great help. In this case clusters could identify important aggregates among offenders. We report anecdotally a correlation between network's clusters and mafia families, and considering the nature of connections among clusters sociologists have been able to infer some new knowledge.

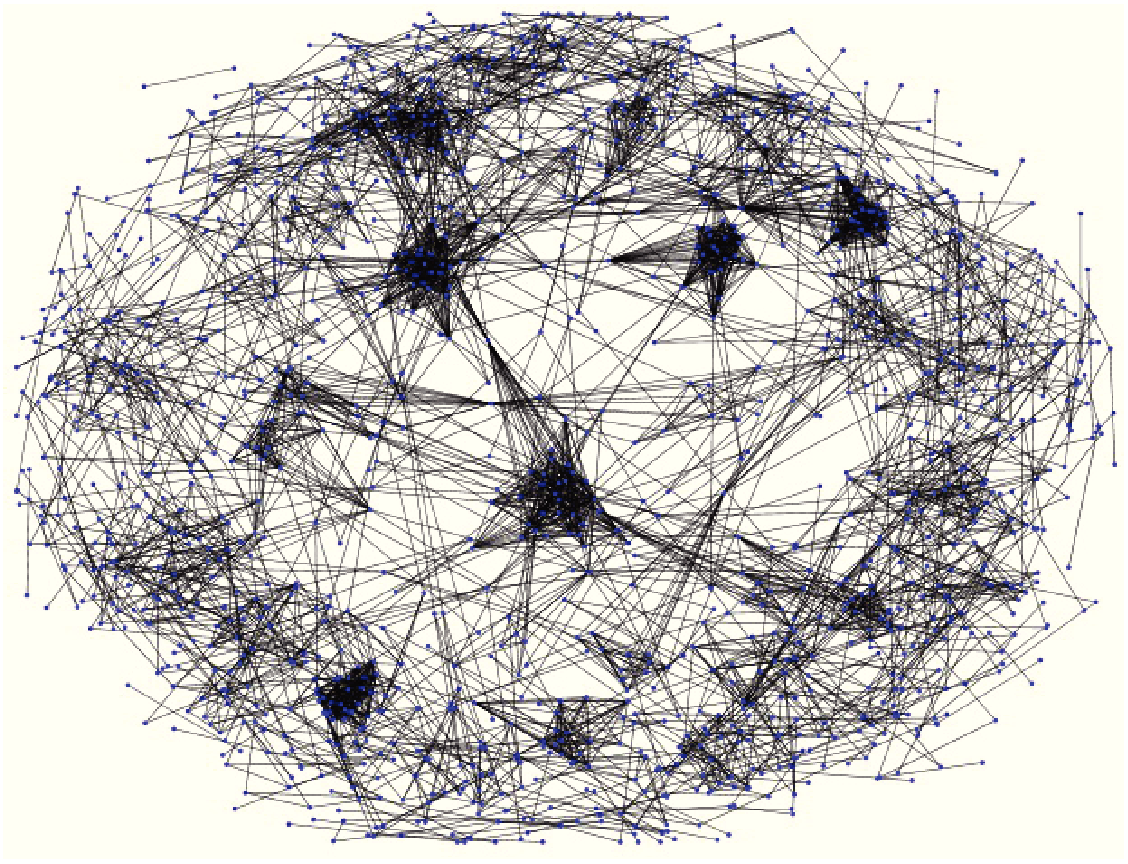


Figure 2 - 3: Complete social network from data of sentences.

2.2.6 Clusters

Clustering is among the main tasks of explorative data mining, and a common technique for statistical data analysis. NodeXL implements three clustering algorithms One of those is generally used to find community structure in very large networks: Clauset-Newman-Moore algorithm [29]. Using this algorithm we have assigned a different color to each cluster and bound every cluster in a box.

As is shown in Figure 2 - 4, each box has an area proportional to the number of vertices that are contained in the cluster.

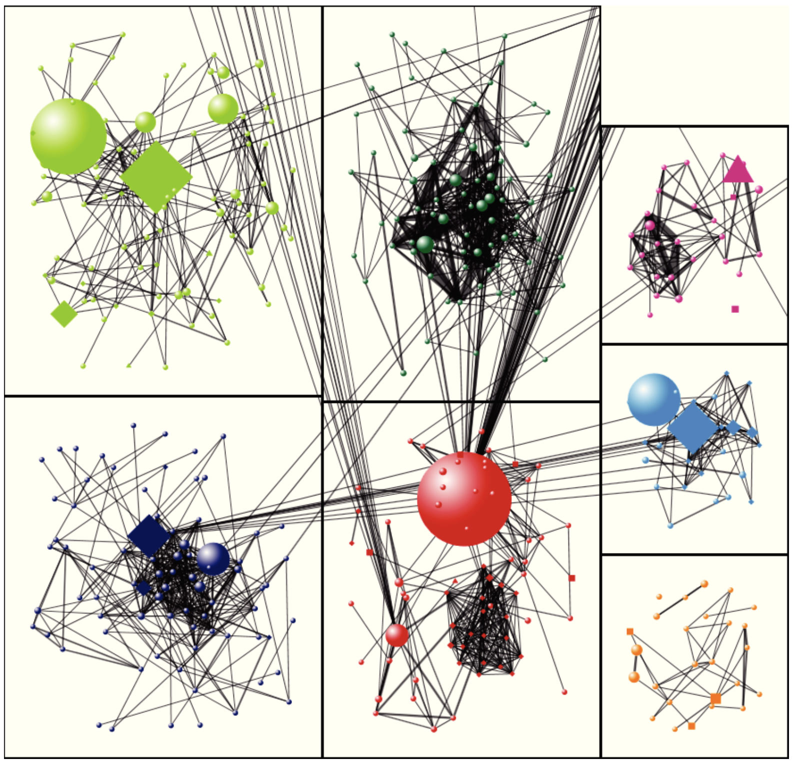


Figure 2 - 4: The diagram shows a portion of the whole graph. The seven rectangles bound seven clusters of persons. Bigger icons denote a higher betweenness centrality. The shape of a vertex represents the role of the corresponding person. See text for details.

We use different shapes to specify the role of person: circles for defendant, squares for lawyers, triangles for judges and diamonds for prosecutors.
The size of vertices depends on the *betweenness centrality* of the node [43]. Betweenness centrality is defined as follows:

$$g(v) = \sum_{s \neq v \neq t} \frac{\sigma_{st}(v)}{\sigma_{st}} \tag{2.1}$$

where σ_{st} is total number of shortest paths from node s to node t and $\sigma_{st}(v)$ is the number of those paths that pass trough v.

A vertex with high betweenness centrality often acts like a bridge between two clusters, perhaps indicating a key role of the person in the small society depicted by our sentences data.
The thickness of edges in the diagram is proportional to their weight.

It is possible to zoom in and visualize a cluster. In the example in Figure 2 - 5 we can see a cluster of defendants only labeled with their names.

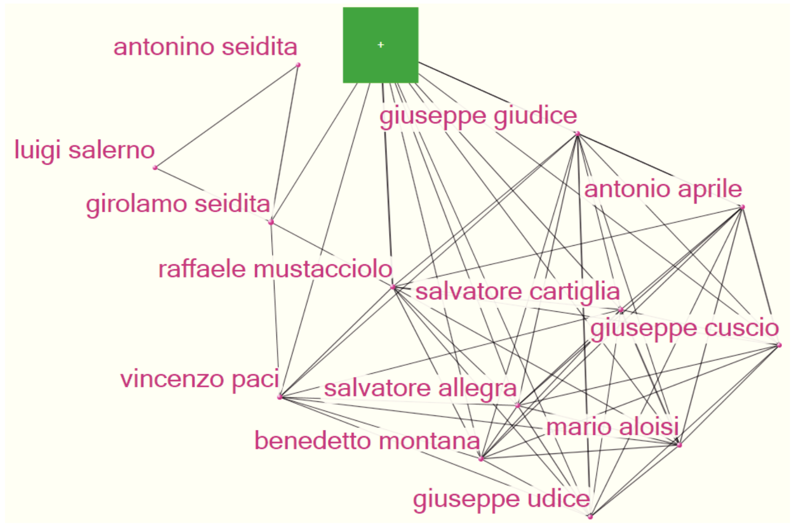


Figure 2 - 5: A cluster in detail.

They are connected to a lawyer with very high betweenness centrality. This shows that almost all people in this small group use to be defended in court by the same lawyer. This lawyer is, moreover, representative of another cluster of people, he acts like a link between two group of offenders.

2.3 Results

Working on this case study, we have been able to experiment on classic principles of information visualization. We have worked on data to achieve the five advantages defined in 2004 by Ware [125]:

1. Comprehension: Visualization provides an ability to comprehend huge amounts of data.
2. Perception: Visualization reveals properties of the data that were not anticipated.
3. Quality Control: Visualization makes problems in the data (or in the data collection methods) immediately apparent.

4. Focus + Context: Visualization facilitates understanding of small-scale features in the context of the large-scale picture of the data.
5. Interpretation: Visualization supports hypothesis formation, leading to further investigation.

Simple information extraction from sentences produces a rough network from the data (Figure 2 - 3). This is nearly useless. It is a way to represent data, but it is not readable: we can't understand data and it does provide very little insight in the structure that it wish to represent. We have tried several combinations of graphical accessory elements like color, size, shape, location, thickness, to code as much information as possible in a single image. By trial and errors we believe that we obtained a graphical representation that at a quick glance may help scholars to roughly grab many important information otherwise very hidden in our data.

Although a rigorous usability test of the proposed graphical layouts are still in progress, we may safely claim that these elements are of great help to navigate and understand network data. Definitive data about the testing will be produced soon.

2.4 Conclusion

This work has to be considered an attempt to realize a more powerful tool to handle large collections of data extracted from texts. Clustering is our first approach to data visualization because it appears to be the natural choice for our particular dataset. Working with person type entities is desirable to find out classes and groups reflecting those in the real world. With simple tools we have a way to visualize data and help scholars to manage thousands of entities and relationships, to identify cluster of people and highlight who of them is more important for his role in the small mafia sentences society. We intend to continue our study on information visualization adopting methods and knowledge we have learned during this work.

Future work will focus on: testing and improving the suggested graph layout; testing the proposed technique on other text collections; compare the proposed technique with other published methods.

Chapter 3

Visual Analysis of Multivariate Data from Dairy Industry

"Most days of the year are unremarkable.
They begin, and they end, with no lasting memories made in between.
Most days have no impact on the course of a life.
May 23rd was a Wednesday."
- Narrator [(500) Days of Summer]

The following contents are object of the paper *Visual Analysis of Time-Dependent Multivariate Data from Dairy Farming Industry* by L. Di Silvestro, M. Burch, M. Caccamo, D. Weiskopf, F. Beck and G. Gallo, presented at the *Conference on Information Visualization Theory and Applications (IVAPP) 2014*.

3.1 Introduction

This chapter addresses the problem of analyzing data collected by the dairy industry with the aim of optimizing the cattle-breeding management and maximizing profit in the production of milk. The amount of multivariate data from daily records constantly increases due to the employment of modern systems in farm management, requiring a method to show trends and other insights in data for a rapid analysis. To this end, we have designed a visual analytics system for animal researchers to analyze time-varying data. Well-known visualization techniques for multivariate data are used next to novel methods that show the intrinsic multiple timeline nature of these data as well as the linear and cyclic time behavior. Seasonal and monthly effects on production of milk and milk components (e.g. fat and protein) are displayed by aggregating data values on a cow-relative timeline according to the day of calving. Basic statistics on data values are dynamically calculated and a density plot is used to show how many samples are taken for each day, quantifying the reliability of a particular dataset. Dynamic filters are offered to the user for data exploration. A qualitative expert user study conducted with animal researchers shows that the system is an important means to identify anomalies in data collected and to understand dominant data patterns, such as clusters of samples, outliers, and trends. The evaluation is complemented by a case study with two datasets from the field of dairy science.

To increase the competitiveness of the dairy sector in the national economy, the dairy industry focuses on improving farm management. The information on farm productivity to support management on dairy farms is often collected by Dairy Herd Improvements agencies. Dairy farmers are usually visited once per month, during a day called *test-day.* Information on the herd such as breeding events, gender, and weight of new-born calves is collected. In addition, the milk production of each cow is measured and a milk sample is gathered to determine the fat and protein content, as well as somatic cell count.

Test-day records represent a valuable resource for animal researchers, but as data sources become larger, the analysis and exploration of patterns in data becomes more complex, representing a critical bottleneck in analytic reasoning. To address these problems, a visual analytics approach can be used to let users explore their data and interact with them with the aim to find interesting insights and eventually data anomalies and formulate hypothesis. Techniques that support the production and dissemination of analysis results may help researchers communicate to a variety of audiences.

We present an interactive system to analyze time-dependent multivariate data. A suite of visual analytics tools is designed for animal researchers, allowing them to perform an in-depth study of data in order to develop and explore their hypotheses. Since test-day records are multivariate data with an intrinsic time-varying nature, we devise techniques that are capable of representing linear and cyclic time behavior simultaneously, for example, by showing seasonal and monthly effects. Furthermore, a density plot is included to let the user evaluate the reliability of each dataset. Our system design is characterized by its conceptual simplicity that intends to be an easy-to-use tool for researchers with no background in visual analytics.

The utility of our system and its visualization methods is evaluated by a qualitative user study with domain experts and demonstrated by a case study. Two datasets from the dairy industry in Sicily (Italy) are used. In Sicily, approximately 125,000 dairy cattle are raised. Compared to other regions such those in Northern Italy, smaller traditional and less modernized farms are present that have to deal with hot climate, higher costs of feed and energy. Around 62% of Sicilian milk is produced in Ragusa province, representing therefore the most important production pole for the dairy sector in Sicily. In 1996, the dairy research center CorFiLaC was established to support the development of the dairy sector in Ragusa [19]. The datasets provided by CorFiLaC and used for our case study cover the typical range of (varying) sample rate and time span in which data are collected.

3.2 Related Work

As our system was designed to analyze different aspects of the same dataset, it merges visualization methods from several application domains. In this section, relevant topics from several areas will be covered, according to the different aspect of data analysis on which the user wants to focus.

3.2.1 Animal Research and Dairy Industry Domain

One of the main topics investigated by animal researchers is the study of movement of individual organisms or groups of animals. As stated by Nathan et al. [92], movement plays a major role in determining the fate of individuals; the structure and dynamics of populations, communities, and ecosystems; and the evolution and diversity of life. Furthermore, it is highly convenient to use a visual metaphor to represent and analyze this kind of data. Researchers studying moving objects, such as animals, ships, cars or pedestrians, face similar challenges in data analysis, interpretation, and visualization [111].

There is an increasing interest is movement ecology. This discipline combines expertise in a variety of fields, including biology, ecology, botany, environmental science, physics, mathematics, and virology. Accordingly, there are a few examples of visualization and visual analytics research geared toward data from movement ecology. Grundy et al. [51] exploit data obtained through tri-axial accelerometers to trace movement of wild animals. They make use of interactive spherical scatterplots and spherical histograms instead of the 2D time-series plots commonly used to study acceleration data. Benke et al. [8] use a geo-visual analytics approach, combining geo-spatial analysis with visualization.

Visual analytics is also used to study and model epidemic spread in herds. Afzal et al. [1] present a suite of predictive visual analytics tools to model the spread of Rift Valley Fever through a simulated mosquito and cattle population in Texas, providing an interactive environment for the investigation of multiple courses of responses as well as comparisons of the effectiveness of each component of a response plan.

However, we see only very little previous visualization work on animal research related to the dairy industry. In fact, there is almost no previous paper in which visualization tools and visual analytics methods would be used on milk production records or other data from dairy industry. The only exception known to us is the work of Galligan, who created *Cowpad* [45] and *Dairy Dashboards* [46], a suite of web tools to present analytical problems commonly occurring on dairy herds. Simple interfaces let the user evaluate the effect of changing variables in the process of dairy farm management. However, these tools do not use any advanced visualization or interaction technique to represent data involved in the simulation, only curve diagrams and gauge metaphors are proposed.

With this paper, we want to fill this gap in the literature, adopting visual analytics methods for this particular application domain.

3.2.2 Multivariate Data Visualization

We designed our visual analytics system to manage multivariate data and focus on the analysis of trivariate data (milk production, fat and protein values) varying over time. Therefore, our work fits in the major stream of research related to multivariate data visualization, as a specific type of information visualization that is an active research field with numerous applications in diverse areas [26].

The overarching design goal is to support non-visualization experts in their analysis work. Therefore, we adopt common, standard visual representations of multivariate data that do not need any or only little explanation for our target user audience. In particular, scatter plots, multiple function plots, bar charts, or similar are included. To reduce higher-dimensional data, geometric projection methods are employed [70]. Concerning trivariate data, modeling information in 3D space is most straightforward, but problems arise when displaying it in a 2D representation; indeed this method suffers from the well-known problem of occlusion and furthermore, it is hard to compare two points along the same axis. Therefore, we refrain from using 3D scatter plots, but apply projection methods to reduce the dimensionality. Similarly, we do not employ the scatter plot matrix [54] because this kind of diagrams might become hard to read due to small display space for each single plot, and according overplotting problems. Likewise, we do not use the scatter plot navigation by Elmqvist et al. [38], because animation can sometimes confuse novice users that have no background in visualization and that are not accustomed to use advanced charts and diagrams. To reduce learning efforts, we also refrain from employing parallel coordinates [66].

3.2.3 Time-Varying Data Visualization

For animal researchers, it is important to investigate how various factors on dairy production change over time. Data used for this contribution have a temporal dimension that makes it possible to highlight monthly, seasonal, and annual trends.

For time-dependent data, there is a large collection of different visualization techniques for different purposes of data exploration and depending on the characteristics of the temporal structure. Müller et al. [91] provide a survey of such techniques. Following the taxonomy by Aigner et al. [2], dairy production datasets exhibit a quite specific structure: they show a double structure of time, both linear and cyclic. For certain aspects, the analysts have to study the linear progress of data over time; in addition, they are interested in finding out seasonal and monthly effects. To highlight particular characteristics of data it is important to consider both structures simultaneously.

Van Wijk and van Selow [123] present a system to identify patterns and trends on multiple time scales simultaneously. It allows the visualization of univariate time-series on different levels of aggregation by clustering similar daily data patterns. We use different levels of aggregation to visualize two different timelines at once. Spiral approaches are a way to visualize large datasets and to support the identification of periodic structures in data. Previous solutions [23, 57, 127], however, are not suitable in cases in which the cyclic trends of data are not perfectly periodic, or the period is not known but subject to specific seasonal effects.

To the best of our knowledge, no previous work addresses the problem of visualizing linear and cyclic time data simultaneously.

3.2.4 Interaction

Interaction plays an important role in the design of our visual interface. We follow the Visual Information Seeking Mantra by Shneiderman [113]: "Overview first, zoom and filter, then details on demand". For the more detailed design of interaction, we used the refined taxonomies by Kosara et al. [79] and Yi et al. [133]; our system supports the different categories of tasks and interactions from those taxonomies. With a formative process and qualitative user feedback (see Section 3.6), we have been improving the interaction mechanisms. For example, the user feedback in particular highlighted the need for interactive filtering by a specification of properties of the desired subset (querying) [72].

Interactive, easy-to-use, and general-purpose systems for visual analysis have been gaining much attention and popularity. As typical examples, *Many Eyes* [65] and *Gap Minder* [48] are tools that are being used by a growing number of users that, for the first time, approach visual analytics. Similar to these tools, our system is designed as a simple visual interface for researchers with no particular background in visual analytics. Geared toward the needs of animal researchers, it has serves as a flexible tool of this class of analysis.

3.3 Application Background

Data on farm productivity is often collected by Dairy Herd Improvements (DHI) agencies. Data at herd level such as breeding events (calving and mating date), gender, and weight of new-born calves are collected usually every 3 months. Test records at individual cow level are usually collected every month by animal researchers at the DHI agency. Milk production of each cow is measured and a milk sample is analyzed to determine fat and protein content, and somatic cell count. These records are then processed and analyzed centrally (i.e., on DHI computers) by taking into account information generated during previous test-days at both herd and individual cow level. A few days after the test-day, the farmer receives a report containing information collected during the test-day. This report may also contain management information on individual cows and the herd as a whole. The information from the report can then be used by the farmer (or milk and cheese producer) for making decisions focusing on the improvement of farm management practices [19].

Although DHI data and information can contribute to improve management practices, the benefits only come about if the farm managers and/or the animal researchers that work as advisors at DHI spend considerable effort for analyzing the incoming information. This process can be time-consuming and complex due to the large amount of data. The quantity of information increases rapidly with the number of cows. For this reason, it is necessary to develop analytical tools that will accelerate and improve this task for the experts at DHI. Such tools should not only filter and pre-process the data, but also present the results in a way easy-to-use for the producer. The system should allow the advisor at DHI to rapidly analyze the data and communicate insights to the farmer, who then can draw conclusions for management decisions that will lead to improvement of performance in technical and financial terms.

3.3.1 305-Days Lactation Yield

Milk recording is recognized as a valuable means for breeding and herd management worldwide. Furthermore, these records are also used to perform a *genetic evaluation* for dairy production traits. Genetic evaluations are provided for different dairy breeds. Within each breed, all animals receive a genetic evaluation for a complete series of characteristics including milk, fat, and protein yield [44].

The genetic evaluation of dairy sires and cows has been based on the analysis of the 305-day lactation yield for many years. It uses the total amount of milk produced during a lactation (or milking) period. Usually, a dairy cow lactates for about 10 months ($\sim$305 days) each year before it dries up prior to giving birth to its next calf. The ideal method to estimate the 305-days lactation yield is to measure the amount of milk and milk components of each cow on a daily basis for 10 months after calving. However, one sample of milk is usually taken monthly for each cow. As stated by Schaeffer and Burnside [107], a common method of predicting 305-days yield is to compute the average production between two tests, multiply by the number of days between tests, and accumulate this quantity after each report. If the interval between two tests is much longer than 30 days, erroneous predictions of this value could result.

Several methods have been developed to improve the prediction of 305-day lactation yield. These methods base their prediction on the shape of the lactation curve and take into account different factors affecting this shape, such as herd environment and year of calving (i.e. the number of calving/lactation periods by the same cow). The term lactation curve is used to refer to the curve representing the rate of milk secretion with advance in lactation. Keown and van Vleck [75] categorized the lactation curve into stages and used the correlations of production among stages in a method for extending incomplete records to a 305-days production value. Schaeffer and Burnside [107] estimated the shape of the lactation curve by using a multiple-trait least squares analysis to calculate constraints for average milk production values.

3.3.2 Test-Day Models

After two decades, animal researchers started to use test-day yields for genetic evaluation rather than 305-days yield. By definition, a test-day (TD) model is a method of evaluating daily production of milk, fat, protein and somatic cell count considering effects for each test-day instead of one set of fixed effects over the lactation. The TD model estimates lactation curves and their changes.

Stanton et al. [117] estimate milk, fat, and protein lactation curves with a test-day model. Ptak and Schaeffer [98] use test-day yield for genetic evaluation of dairy sires and cows. Reents et al. [102] perform a genetic evaluation for somatic cell score with a test day model for multiple lactations.

In the early 1990's, Ptak and Schaeffer [98] identified some drawback in the new approach, such as the need to store all of the individual test-day yields on every cow, or the higher computational time needed to calculate a genetic evaluation by using more values per cow instead of only one.

With less than a decade, test-day models have been adopted in several countries. Schaeffer et al. [108] show that TD models provide 4% to 8% more accurate genetic evaluations of cows compared to evaluations from 305-days yields. In particular, they present a random regression TD model as an extension of the TD model that allows the shape of the lactation curve to differ for each cow by including random regression coefficients for each animal. Caccamo et al. [20], using random regression test-day model outputs, showed that herd curve variance for dairy cattle in Ragusa and Vicenza provinces (Italy) are extremely high, suggesting that variation could be explained by differences in management practice across herds.

3.3.3 Means for Hypothesis Building

Test-day models are used in most countries to perform national genetic evaluations for dairy cattle. Estimation of genetic, environmental, and herd effects can be used to predict future productions on individual cows. Deviation between predicted and actual production could be used to detect a disease at an early stage, i.e., before the cow shows clinical signs. For this reason, animal researchers need a system to identify abnormalities in lactation curves.

Halasa et al. [52] used the difference between the actual and the modeled milk production curve to identify subclinical mastitis (fibrocystic disease). A subclinical disease has no recognizable clinical findings. It is distinct from a clinical disease, which has signs and symptoms that can be recognized. Records from cows with clinical mastitis were excluded from the model building process in order to use predicted production based only on healthy cows. Jamrozik and Schaeffer [67] use a model based on test-day milk yield, fat-to-protein ratio, and somatic cell score to detect subclinical mastitis.

Animal researchers need a system to identify abnormalities in lactation curves. Nowadays, the storage of big amounts of data or the computational resources needed to handle these data do not represent a problem anymore. It is possible to use the complete set of test-day records collected over the years. Researchers can work directly on those data with no use of statistical or mathematical models that could lead to loss of precision or erroneous predictions.

A visual system to represent milk production data could be used to identify the characteristic traits of the lactation curve depending on various intrinsic features, such as breed, herd or number of lactation. It could be a rapid method to find out anomalies in data and to compare collected records with data coming from new productions. The visual analytics approach allows the animal researchers to build hypotheses on characteristic traits and anomalies alike—without being constrained by predefined statistical models or simplified models of lactation.

3.3.4 Multiple Timelines

The challenge for dairy producers is to interpret and utilize data from dairy production properly to improve decision making. Milk production data have an intrinsic dual timeline nature. Usually, lactation is represented according to a "cow-related" timeline: a curve shows the quantity of produced milk changing over time, from the day of calving to the moment in which the cow is *dried off*, and milking ceases. About sixty days later, one year after the birth of her previous calf, a cow will calve again, starting a new parity (period of lactation).

Since the "cow-relative" behavior is most important for temporal analysis, that *relative time*—represented as day-in-milk—is of highest relevance for analysis. Parities represent *cyclic time* characteristics in the data because the parities progress similarly. If one analyzes a single parity, there is *linear time* behavior within that parity. However, there is another temporal aspect that needs to be taken into account: the *absolute time* of the yearly calendar. Possible seasonal effects on production of milk and milk components ratio are due to yearly *cyclic* effects. However, within the year, we consider calendar dates as *linear time* because there are no weekly or daily effects on milk production data.

Mellado et al. [89] studied the effect of lactation number, year, and season of initiation of lactation on milk yield by using polynomial regression analysis.

In our contribution, a method to show and analyze the dual nature of time, with respect to the linear and cyclic time behavior is presented. A visualization system for time-varying data could also be used to handle data from *high production cows*. This kind of cows is more difficult to breed at a one-year interval. Many farms take the point of view that 13 or even 14 month cycles are more appropriate for this type of cow. With these data records, the traditional 305-days model cannot be used; consequently an adaptive and flexible analytics system could be helpful for the analysts.

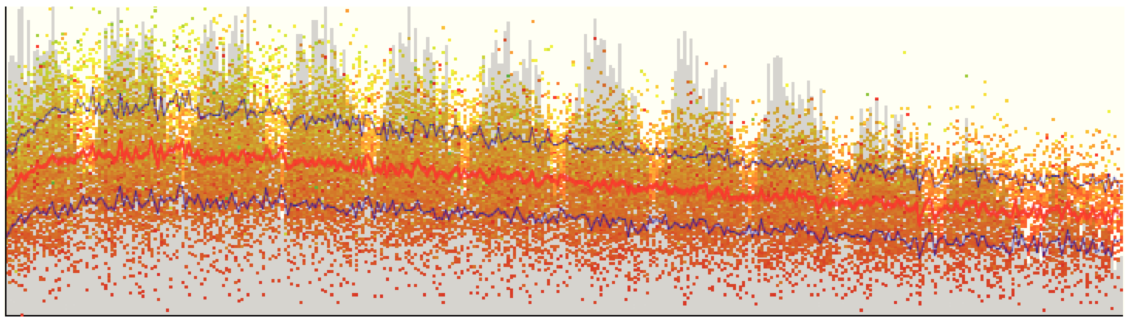


Figure 3 - 1: Statistics on time-varying multivariate data. The value of produced milk for each day-in-milk is shown. Scatter plot points are colored according to the value of fat contained in the sample of milk. The curve of density of samples in the dataset is represented in gray. The red line indicates the mean value of quantity of milk and the blue lines the standard deviation.

3.4 System Overview

In our system, several visualization methods are used. Well-known approaches are used besides some new techniques for visualizing two coexisting timelines simultaneously. The aim of the tool is to assist animal researchers in all the phases of their analysis process. A preliminary analysis could be performed for a data cleaning phase; then a deeper exploration lets the user study different aspects of the data, comparing different views.

To address facilitate multivariate and time-dependent data visualization, we designed to system to support several, complementary views. The user can choose from a list of four different views (see below for details). The views are linked by automatically updating and sharing all settings between views, so that it is easy to compare outputs. Every time a parameter changes, all the views are updated accordingly. In the visualization panel, a window to access the raw data through a spreadsheet table is available, but it is possible to hide this table or show the data view in full-screen mode hiding any other information besides the data selection view, and settings.

3.4.1 Data Conversion and Pre-Processing

It is possible to load and handle any kind of multivariate dataset with the constraint that a field containing a date (or time) must be included. Some of the functionalities will work only if a second field can be treated as a second timeline. Data can be loaded from a tab-separated values file (.txt) or in a comma-separated values file format (.csv). Data are loaded and shown in a tabular form similar to spreadsheets and application programs that animal researchers use to work with. The number of entries and fields of the dataset are shown and updated on every change made by the user. Rows can be ordered, by selecting the header following alphabetical or numerical order. Complete columns or rows can be removed from the dataset.

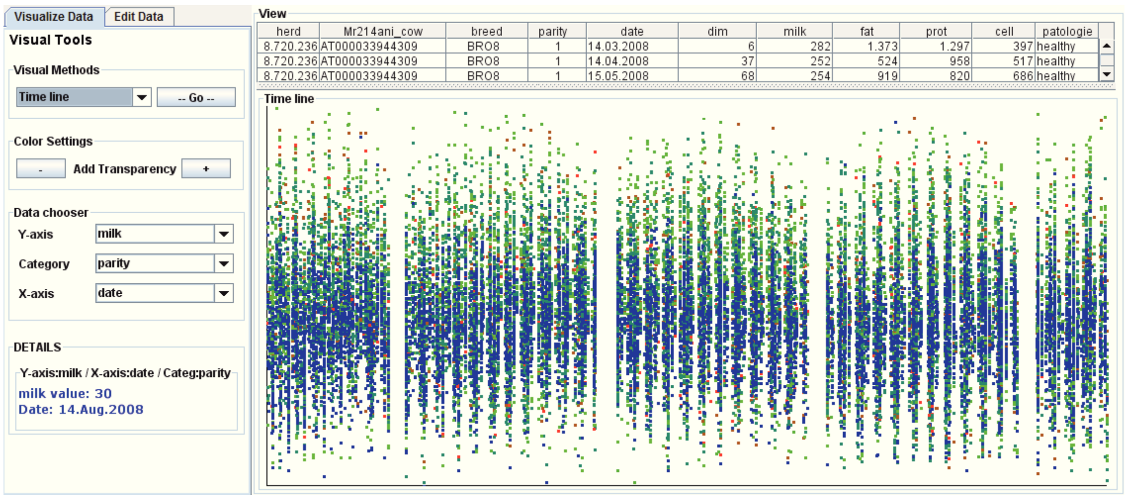
3.4.2 Scatter Plots

One of the visualization methods in the system is based on scatter plots. In this way, multivariate data can be shown, such as the attributes milk production, fat contents, protein values, and their time dependency. Any data field can be used as x-axis and y-axis. By default, the date field is used as x-axis, emphasizing the importance of time in our analysis approach.

A third data field can be used as a *category* to accordingly color each data point in the scatter plot. In this way, the colored scatter plot can show three data dimensions. If the *category* field chosen contains labels that assign class (categorical data), the user is asked to select a color for each label. It is possible to show data belonging to a certain class only, hiding uninteresting data from the view. If the field chosen as *category* contains continuous values, data points are colored automatically using a color scheme selected by the user from a suggested list. Furthermore, it is possible to filter data by *category* values. A slider with two handles lets the user choose the width and the endpoints of the interval of values to show. By moving the handle on the slider, the view interactively updates the plot, pointing out differences in data according to the changing values.

When a burst of aggregated data points overlaps making the scatter plot difficult to understand, transparency can be used to reduce visual clutter. The user can increase or decrease the alpha value of data points in the scatter plot.

Moving the mouse pointer over the plot, details on data points are displayed next to the view. Figure 3 - 2 shows the scatter plot, and the side panel. This panel is used for choosing data to plot, and environment settings. Details on data points selected by the mouse pointer are also shown.



Time line

Y-axis milk
Category parity
X-axis date

Y-axis:milk / X-axis:date / Categ:parity
milk value: 30
Date: 14.Aug.2008

herd	Mr214ani_cow	breed	parity	date	dim	milk	fat	prot	cell	patologie
8.720.236	AT000033944309	BRO8	1	14.03.2008	6	282	1.373	1.297	397	healthy
8.720.236	AT000033944309	BRO8	1	14.04.2008	37	252	524	958	517	healthy
8.720.236	AT000033944309	BRO8	1	15.05.2008	68	254	919	820	686	healthy

Figure 3 - 2: System overview. In the upper part of the window, raw data are shown in tabular form. In the bottom part, the view shows a scatter plot of selected data. In this view, the production of milk is shown for each day (absolute timeline). In the side panel, the user can change the visualization method and choose which data fields have to be plotted. In this example, the *parity* value (number of lactations) is used as category. First parity is colored blue, next parities are colored with a blue-to-light-green color gradient, and the last parity is colored red. During the first parity, milk production is lower than in the next and the data value range is narrower. Missing data points follow a regular pattern (recurring blank vertical stripes) that reflects the lack of data samples collected during summer vacation (August).

3.4.3 Statistic Metrics and Density Plots

A line chart of mean values as well as standard deviation ($\sigma+$ and $\sigma-$) can be superimposed on the scatter plot diagram, showing aggregated statistical quantities. Moreover, the density of data samples in the dataset for each y-axis value is calculated and shown as a histogram. The histogram allows the user to assess the quality of the original data indirectly; less density is related to higher variability of the data samples in that region. Therefore, the combination of density diagram with other diagrams allows us to include a measure of uncertainty in the overall visualization.

The user can hide one of the line plots or the histogram as they need. Otherwise, it is possible to hide data points to focus on the statistical quantities only. A low-pass filter can be used to smooth the line plots and density plots. The user can increase or decrease width of the low-pass filter interactively. In Figure 3 - 1, a dataset is displayed in which the density of data samples varies over time following a regular pattern: more samples are collected in certain days of the month. As shown in Figure 3 - 3, the density plot can be used to recognize that samples are normally distributed for a certain random variable (milk in this example).

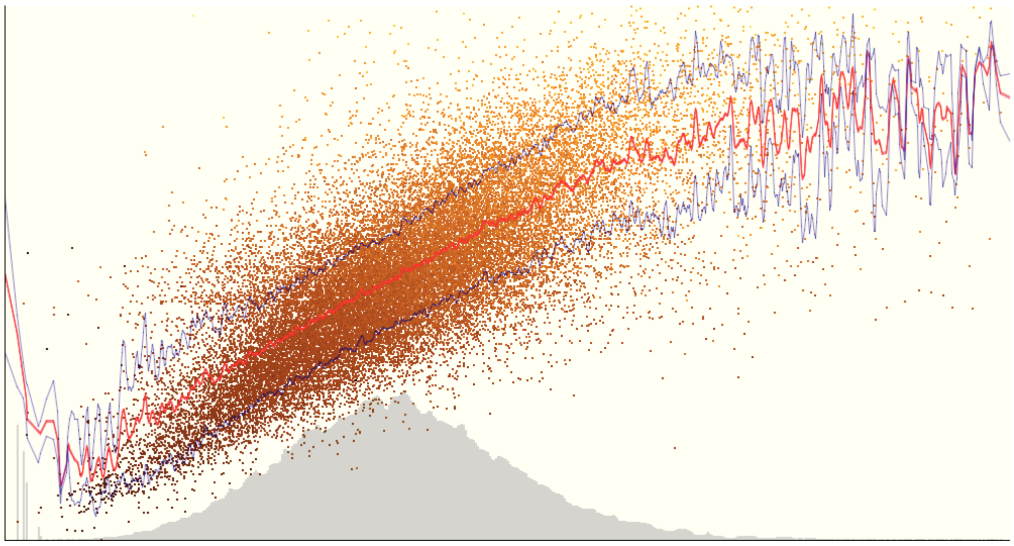


Figure 3 - 3: Relation between the quantity of milk (x-axis) and the percentage of protein (y-axis). Data points are colored according to the percentage of fat in milk samples (lighter color for higher values). The red line curve indicates the mean value of quantity of protein and the blue lines the standard deviation. The density plot shows that the number of samples for milk quantity follows a normal distribution.

3.4.4 Multiple Timelines View

We present two methods for showing multiple timelines simultaneously. As discussed in Section 3.3.4, the different temporal data characteristics make it difficult to analyze the temporal effects. In particular, depending on the focal point of analysis, cyclic and linear as well as relative and absolute time play a role. For analysts, it is useful to highlight cyclic data patterns as well as to point out differences between different periods of time. The system identifies an absolute timeline with linear behavior and a relative timeline that follows a cyclic behavior.

Stacked View

The first method aggregates data points for different periods of time over the relative timeline. Multiple scatter plots share the same *x*-axis that represents the relative timeline. A plot is shown for each period of the absolute timeline. The user can choose the time period to use from a list to aggregate data and build up a stacked view of several plots. It is possible to create a plot for each season, month, or year. In Figure 3 - 4, stacked scatter plots are shown for different months.

This method helps identify frequent patterns in data points or highlight cyclic behavior, but it is difficult to draw a comparison between data values at the same relative time point in different plots. The following data view is designed to help the analyst in this task.

Line Plots

In a single plot, different line charts are drawn, representing the mean value curve for aggregated data on different time intervals. Like in the data view presented before, the user can choose to aggregate data values for months, seasons, or years. In this view, each plot shares the same coordinate system, so that the line charts are aligned. Thus, it is simpler to identify differences in the curve of values for different time periods. The line charts can be rounded by the user as in the statistics view introduced before.

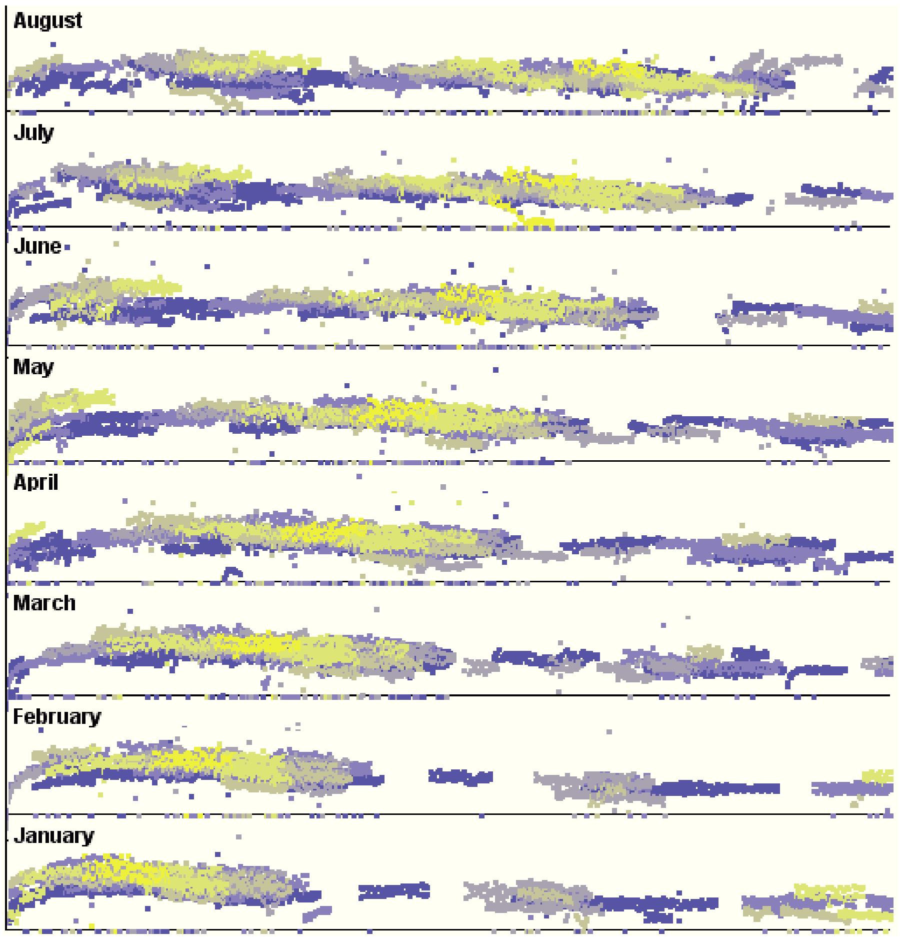


Figure 3 - 4: The value of produced milk for each day-in-milk (relative timeline). The absolute timeline (date) is shown by aggregating data for each month. Data points are colored from blue to yellow according to the parity (number of lactation). In first parities milk production is lower. Stacked scatter plots highlight differences in milk production and in data sampling: a gap in data points for a certain period in day-in-milk is visible from January to July shifting accordingly.

3.5 Case Study

We demonstrate the usefulness of our approach for two datasets with different features from the same application area.

Data used for our study was provided by the dairy research center CoRFiLaC (Consorzio Ricerca Filiera Lattiero Casearia). CoRFiLaC was estabilished in 1996 to support the development of the dairy sector in the Ragusa province (Sicily), where about 62% of the overall milk production and more than 50% of Sicilian cheese is produced. CoRFiLaC aims to deliver scientific and technical information to producers in a timely fashion to support management decision for dairy farms and to support the production of traditional Sicilian dairy products [19].

Analysis of collected data should allow the farmer to rapidly determine management decisions that will lead to improvements in technical and financial terms.

The analysis was conducted with an expert that was familiar with the datasets. During our study, hypotheses made by the domain expert were confirmed with the visual analytics approach, and the implementation of new features was suggested in the sense of a formative process.

3.5.1 Datasets Description

Data used for our study were collected and used by animal researchers to assist farmers in the management of dairy herds.

Dataset_1

The first dataset contains 175,689 records on 6,468 cows from over 40 herds. Samples were taken monthly at the test-day, for over 10 years. Each record provides the following information:

- Herd number
- Identification number of the cow
- Code for breed of the cow
- Number of lactation (also called parity)
- Date of the test-day
- Days-in-milk (i.e., the number of days between the calving of the actual parity and the test-day)
- Yield of milk, fat protein
- The somatic cell count measured at the test-day
- Detected pathology associated with the test-day

In the dataset, records from 6 parities are included with days-in-milk ranging from 5 to 365.

Dataset_2

The second dataset used contains data collected from one farm only, i.e., one herd. At this farm, production data are collected in autonomy, without the support of a DHI agency. It is a modern farm in which data are collected almost daily, and only healthy cows are registered. Records are structured like the ones in *Dataset_1* with the exception of some missing fields: herd number, breed code, and the detected pathology.

Data records regard a single herd of 90 *high production cows* that continue to produce a sufficient quantity of milk for 11 months after calving (5 to 395 days-in-milk).

3.5.2 Analysis Process

Using both datasets, we performed an analysis session with the help of the domain expert. The analysis started by showing an overview of data contained in *Dataset_1* (see Figure 3 - 2). By using the absolute timeline (date of test-day), a recurring gap is evident in data sampling. This gap can be explained by the period of vacation that was observed in the farm during years prior to 2010 in Sicily. The data from *Dataset_2* does not show this gap any more, which can be explained by a change in management policy of the farms: in recent time, the farm does no longer follow the same period of vacation.

Colors were used to mark data points according to the detected pathology associated with the test-day. Different views allowed us to spot significant changes in the overall shape of curves, from which we could derive some insights. A slight loss in milk production was recognized for cows suffering from mastitis, when using the relative timeline (days-in-milk). When using the absolute timeline, we noticed a lack of registered sick cows before 2005. The analyst decided to hide data records concerning the period 1995–2004 from *Dataset_1*. The next analysis phases focused on three years only (2005–2008).

The density plot revealed evident changes in number of data samples during time. In the dataset, an increase in data collected in the last years could be recognized, which we attributed to the employment of modern systems in farm management. By showing data according to the days-in-milk (see Figure 3 - 1), a regular pattern in the density plot could be highlighted. Usually, the same group of agents from the DHI agency visits all the farms in the same area, scheduling one visit per month for each one. Thus, each peak in the density plot corresponds to the recording day (test-day) in the biggest farm of the area. With our visual analytics system, it is then possible to use a subset of the original data, taking into account only the period of time considered to be more reliable for which more data records are available.

During the first analysis phase, the system turned out to be a useful and fast means for data cleaning—on top of direct visual analysis. With visual data cleaning, we can even support machine-based data analysis. For example, a (cleaned) subset of data can be used to train classification algorithms to recognize the healthy status of a cow from the lactation curve shape or the milk contents percentage to warn the farmer before clinical symptoms occur.

The multiple timeline views were used to find differences in milk production for different seasons. In winter, cows generally produce more milk than in spring, but no evident differences could be identified for different months. By plotting stacked data points for each month, some data points in the August plot are shown, which identifies an anomaly in data recording that could be confirmed after the session.

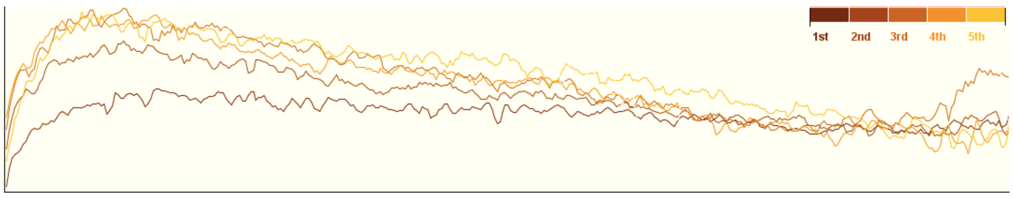
Other evident anomalies were found in *Dataset_1*: some record was filled with dummy values by the farmer while bursts of copies of the same record were used to fill some gap in data collection. Once again, this information can be used for cleaning the dataset to obtain a representative subset of reliable data.

3.5.3 Suggested Features

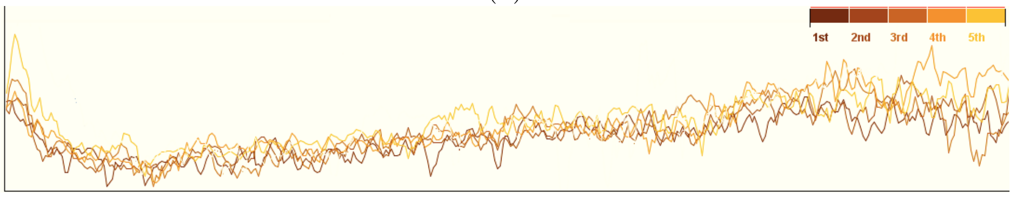
During the analysis process, some new features have been suggested by the domain expert. The multiple time line views were recognized as a good means to compare aggregated data values or mean value curves according to different characteristics.

The possibility to aggregate data and to visualize different plots according to the breed and the parity number were added in response to the expert's comments. By showing different curves for breeds, it is possible to find out which cow variety is more productive and how milk composition between different stocks is differing

Different lactation curves for each parity confirm well-known theories. Figure 3 - 5 shows different curves for the first 5 parities in the data from *Dataset_2*.



(a)



(b)

Figure 3 - 5: Multiple line plots highlight differences in lactation curve (a) and fat contents (b) for five parities. The quantity of fat decreases during the peak of milk production. The shape of the fat curve is equal for different parities, whereas the shape of lactation curve changes over parities.

Then it is asked to re-align data according to the beginning of the lactation period, so that it is possible to find differences in lactation curve depending on the moment of calving.

These features and other minor changes in the user interface were implemented before the evaluation by expert users was conducted.

3.6 Expert Evaluation

Our system, as well as the visualization methods used, was evaluated by a qualitative user study with domain experts. A small group of animal researchers was asked to use the system, with the aim to collect subjective feedback and opinions. The analysis of the feedback by potential target users describes preferences concerning views and features of the tool, and helps us to improve the software to meet users' needs.

We were mostly interested to follow the natural process of hypothesis building and problem-solving adopted by the animal researchers. To achieve this goal, it is fundamental to let the domain experts operate in their workplace as explained by Dunbar [36] proposing in-vivo studies. In such kind of studies, domain experts are observed in their workplace, working on genuine problems using familiar data and tools. For our study, we adopted the pair analytics approach [5]. This method is based on the in-vivo studies and the think-aloud method [81].

With think-aloud, participants are instructed to produce verbal reports about their thought process when performing specific tasks. Arias-Hernandes et al. [5] present a research tool for capturing reasoning process in visual analytics that addresses some of the limitations of the previous methods. Trickett et al. [121] report that one of the caveat of the think-aloud method is that once participants get absorbed in the task the amount of verbalization decreases and their reports on their thought process becomes scattered and fragmented. Furthermore, the imposed think-aloud activity increases the possibility of affecting the reasoning process, such as the *spontaneous insight* defined by Chang et al. [27] as "the process by which a problem solver suddenly moves from a state of not knowing how to solve a problem to a state of knowing how to solve it".

The pair analytics method generates verbal data about thought processes in a naturalistic human-to-human interaction with visual analytic tools. It requires two participants, a Subject Matter Expert and a Visual Analytics Expert [5]. With pair analytics, the problem of training the Subject Matter Expert to (technically) use the system is much reduced and some of the above problems of think-aloud are mitigated.

3.6.1 Study Procedure

As stated by Arias-Hernandes et al. [5], one of the drawbacks of in-vivo studies is that direct access to domain experts in their workplace is sometimes difficult to get. In our case, geographical distance was overcome by using remote communication tools. Every study session was performed by using a VoIP software for communication and each audio session was recorded. An application for remote desktop control was used to operate on a computer in the IT laboratory at CorFiLaC. The three volunteers worked directly on the computer used for the user study. The system was used on datasets presented in Section 3.5.1. These datasets were already analyzed by the participants for previous studies, so that they were familiar with its structure and general contents.

During a preliminary group session, the participants were informed about the procedure and the voluntary nature of the experiment. The system was presented in each component by using artificial data not used for the next sessions. Then, the participants were asked to fill out a questionnaire to collect anonymous information on their educational background and the level of confidence with computer science, statistics, and visualization topics. This questionnaire and others asked to be filled out after each study session were provided in a Web format. Answers were automatically stored in a spreadsheet.

Afterward, a private study session was performed with each of the participants left alone in the IT laboratory. For each session, the remote desktop was video recorded; then the video obtained was synced with the audio of the VoIP call and edited in a single file used to study feedback and suggestions collected. Each session was performed in conjunction with a Visual Analytics Expert that was able to control the remote computer by using mouse and keyboard. For these experiments, the Visual Analytics Expert lived a double nature, being himself the experimenter. He was disheartened to control remotely the system and let the Subject Matter Expert use it. The Visual Analytics Expert was instead encouraged to talk with the animal researcher and ask for comments and the motivation behind every choice made to perform the requested tasks.

At the beginning of the private session, the volunteer was asked to inspect the overview of a subset of records from *Dataset_1* (in Figure 3 - 2). For this task, records collected from years 2005 to 2008 were used. The analysis performed by the domain expert during the case study was used as ground truth to evaluate the correctness of the volunteers' answers. The first task was to explain the missing value pattern in the scatter plot. The second task was to freely comment the data points cloud according to the color coding used to produce the plot.

Then, the participant was asked to freely use the visual analytics approach on *Dataset_2*. With the help of the Visual Analytics Expert, they were instructed to use every available view.

Each study session took about 90 minutes for each participant. During the session, the experimenter annotated the user's behavior, preferred view, and interaction methods to complement audio and video recording that were analyzed afterward for the qualitative analysis of the system usage sessions.

After the experiment, the domain experts were asked to fill out 3 questionnaires on their experience. They were asked to evaluate the usability of the visual analytics approach by a standard System Usability questionnaire [14] using a 5-point Likert scale. Then, a questionnaire to evaluate the mental and temporal demand requested by the assigned tasks was provided. The third questionnaire asked the volunteers to point out the most and the less helpful features of the presented system and the missing feature useful to add.

3.6.2 Participants

For the qualitative user study, three domain experts from CorFiLaC (Sicily), i.e., potential target users, were asked to use our system. The domain expert that worked on the case study and in the preliminary design phase of the system was not involved in this study.

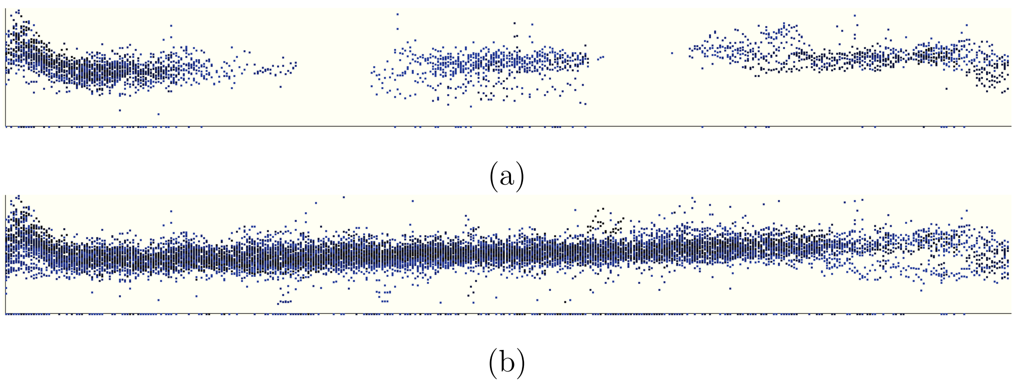
The volunteers for the user study have been working in CorFiLaC for one year and they are concluding their studies for a Master degree in agricultural and food science. They usually work with computers, but they only use spreadsheets and applications for numerical computation. Their answers to the first questionnaire confirm that they are confident with statistical concepts, but they do not have any background in visual data analysis. For their work, they use line charts and bar plots, but they have never used scatter plots or any advanced visualization methods. For this reason, the Visual Analytics Expert was asked to explain the correct way to read a scatter plot during the introductory session. However, due to the simple visual design of the system, the participants could pick up scatter plots as a visualization technique quickly. This part of the user study confirms appropriateness of the overall design goal of our system: to be easy-to-use by non-visualization experts.

3.6.3 Study Results

The goal of the analysis procedure was to evaluate the experience of potential target user of our visual analytics approach. We chose datasets that were already known to the users so that the new approach could be evaluated with familiar data. In this way, no difficulties in interpreting they data would arise during the study.

The simple tasks assigned were used to evaluate the effort required to understand the plots. Two of the participants readily explained the lack of data points and the regular pattern followed. They confirmed the analysis made by the expert involved in the case study and the preliminary design phases. One of the participants was initially not able to explain the pattern. Helped by the Visual Analytics Expert that remotely controlled the computer, the participant then examined the visual representation in more depth and, eventually, derived at the correct explanation of the pattern.

The second task involved the comprehension of the color coding used to mark data points according to the parity. None of the participants had any difficulties working on this task. Their comments revealed that they took advantage of using color to highlight different data behaviors. Initial difficulties to correctly interpret the scatter plot were overcome with the help of the Visual Analytics Expert. In the second part of the session, when the domain experts were free to use the tool and explore data, they showed growing confidence with the scatter plots.


(a)

(b)

Figure 3 - 6: (a) Fat content for day-in-milk during fall. (b) Fat content for day-in-milk for parity started in fall. Data points are colored from black to blue for the first three parities. Missing samples visible in (a) that can affect the analysis task are avoided by aligning data points on the season of calving, as shown in (b).

User opinions on ease of use and utility of our system were collected by the questionnaires sent to the domain expert after the user study sessions. The answers show that the overall user experience with the system was positive. The participants identified the multiple timelines view as the most useful when line charts are used. They also found the possibility to use the same view to highlight differences for different parity or breed interesting, referring to the features suggested by the domain expert involved in the case study. In particular, they indicated as an important means for analysis the possibility to re-align data on the day of calving. This method helps the analyst obtain information on the shape of complete curve, not considering gaps in data sampling as shown in Figure 3 - 6.

Each of them preferred to use line charts to compare the lactation curve shape, but they used scatter plots to identify anomalies in data, such as gaps in data sampling of artificial data values. They recognized the system as a means for simple graph creation, in order to show analysis results to farmers.

In the free comments section of the questionnaires, each of them asked to filter data interactively, without recurring to the editing interface of the system. An interactive filter was later implemented and the usage is shown in Figure 3 - 7.

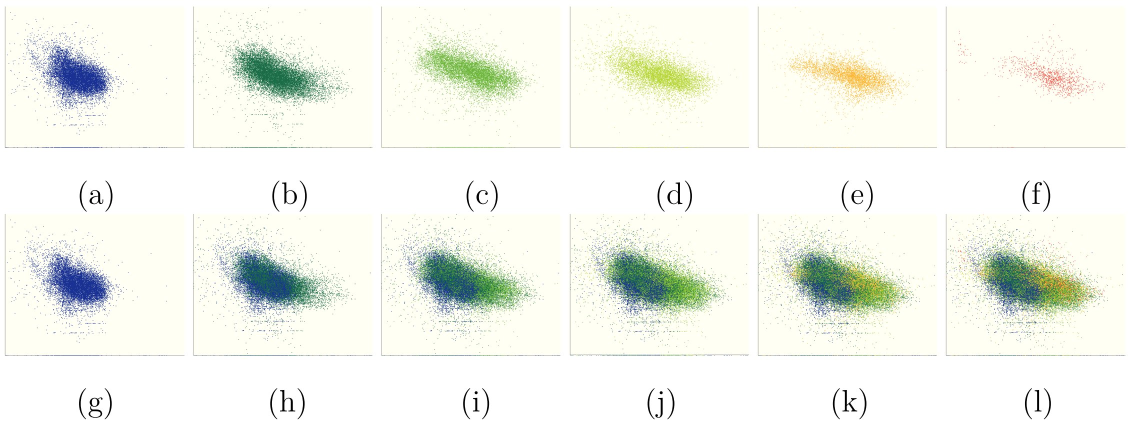


Figure 3 - 7: Scatter plots show the relation between quantity of milk (x-axis) and percentage of fat (y-axis). Data points are colored according to the parity (number of lactation). These image series show different uses of the interactive filter. Figures (a)–(f) show data for different parities (1 to 6). The interval width is constantly set to 1, its endpoints coincide and for each figure the interval is shifted from parity 1 to parity 6. In Figures (g)–(l), the first endpoint is constantly set to parity 1 and the width of the interval is sequentially increased, to add data points from the next parity to the plot.

3.7 Conclusion and Future Work

We used a visual analytics approach to support animal researchers in analyzing multivariate time-varying data. The system is designed to address the needs of the domain experts. By using real data from the dairy industry, we could prove the utility of the system as a means of identifying anomalies for a data cleaning phase, and as a tool for hypothesis building. Besides, an expert user study showed that researchers without background knowledge of visual analytics methods are able to adopt the system quickly.

For proper interpretation of lactation curves, milk production information has to be related to management practices and environmental conditions that might affect lactation curves. The individual cow level data have to be analyzed besides herd level data. The possibility to handle this kind of data could be added to our system in future work. Also, different case studies could be conducted on datasets from other domains. The multiple timeline nature of the data from dairy science might be present in other data as well, and the system might be applicable there, too.

Chapter 4

Visual Analytics System for Multiple Textual Data Sources

“Maybe we could express ourselves more fully if we say it without words.”
- Patricia Whitman [The Darjeeling Limited]

The following contents are object of the *Diplomarbeit* (master thesis) *Analyzing Textual Data by Multiple Word Clouds* by Nils Rodrigues. The thesis has been written in partial fulfillment of the final examination for a master degree in emphSoftwaretechnik at the Visualization Research Center (VISUS) of the University of Stuttgart. I advised this work with Prof. Dr. Daniel Weiskopf and Dr. rer. nat. Michael Burch during a period of research at VISUS.

4.1 Introduction

Vast amounts of textual data cannot be handled easily and quickly aiming at detecting commonalities as well as outliers in the data by just reading them. A huge effort is needed to find insights on a purely textual basis in the whole collection of texts. A promising way to tackle these problems is using visualization in general, and word clouds in particular, to summarize the textual contents and provide an overview. As Thomas Gottron has shown they provide a quick way of quickly finding the importance of texts [49]. They are readable and understandable while at the same time working with only a reduced context. With the advances of the world wide web and the mass emergence of personal blogs, word clouds often appear as a guidance system for visitors to find related or popular information.

However, when this visualization is to be used on multiple data sources it does not scale to the task. It is possible to create a single cloud for all of them, but the user looses the information about the origin of the words. In the context of a textual analysis task, this information in itself could be very relevant. The use of multiple independent clouds solves the issue of origin, but performs poorly when the observer wants to compare them. A possible solution for problems with the use of multiple data sources and their comparability is the development of a single visualization that still shows the words' provenance. However, a visual display alone sometimes is not enough for an in-depth analysis. It has to be embedded into a tool that provides the data to be displayed as well as ways of interaction.

This chapter contains the fundamental ideas and provides insight into the development of the software MuWoC. The name originated from the merging of *Multiple Word Clouds*. The system was designed and developed to answer an unasked question by employing word clouds in the context of textual data from multiple sources.

4.2 Word Clouds

Over time there have been multiple implementations of word clouds. The currently most prominent one is Wordle [40]. Through their website the user can upload any text and let the service create a word cloud from it. The result depends mostly on the frequency with which each word appears. The more often a word is used, the higher its importance is rated by Wordle algorithm.
This relevance is then used to select the topmost n words and accordingly assign a font size. The more important a word is, the bigger it will be drawn [39].

Concerning the colors to use to draw words on screen, the user can provide a palette from which the software will assign a color to each word. Furthermore, the algorithm can introduce small variations to the palettes.

The user can also select to have some words rotated, so that they are not horizontal anymore. This helps to reduce whitespace and packs them tighter together. However, a word's position inside a cloud is not always predetermined by grammatical rules and therefore it can differ from implementation to implementation.

As stated by the author, Wordle "was designed for pleasure" [39]. MuWoC however, follows a pragmatic approach to text analysis. Thus the user has to be able to easily find information. To achieve this goal guidelines were developed for all of MuWoC's visualizations to follow:

Letter casing

Words are more readable if their letters are not converted to lower or upper casing. The cause being that the human brain first recognizes whole words and only later individual letters [68]. At the same time, peripheral vision can recognize the shapes of words before the main visual focus gets to them [99]. As a result, the word *key* is first seen as ■. If a word is completely capitalized to only use upper case letters, the shape is also changed. The same word *KEY* would then appear to the peripheral vision in the shape of a rectangle ■. This means that humans would take longer to read because of the need to use the visual focus to recognize the words.

Separation

Words that have space around them can be read more easily [101]. If the vertical distance between the words *key* and *lock* is removed, the *y* from the former invades the space between the *l* and *k* of the latter. As a result the words don't form the shapes they normally would and the brain is forced to recognize the letters themselves.

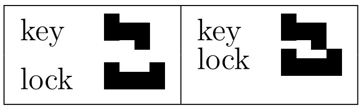


Figure 4 - 1: The effect of missing space space between words

Rotation

Words can be read best, when they are presented in the same direction we are accustomed to [63]. Since the majority of languages this means that horizontal words are easier to read than oblique ones. Therefore the cloud should not rotate them, as readability is more important than space utilization.

Placement of important words

In many word clouds the more important words can be found by observing their font size. However, if the words are distributed throughout large clouds, it becomes more difficult for the user to compare the sizes of objects far away from each other [30]. Even situations similar to the ones in the Ebbinghaus Illusion [104] can occur, which then also lead to false size impressions (Figure 4 - 2). Spatial distance also forces the user to scan the entire cloud to find the most relevant words. Therefore important words should be drawn near each other.

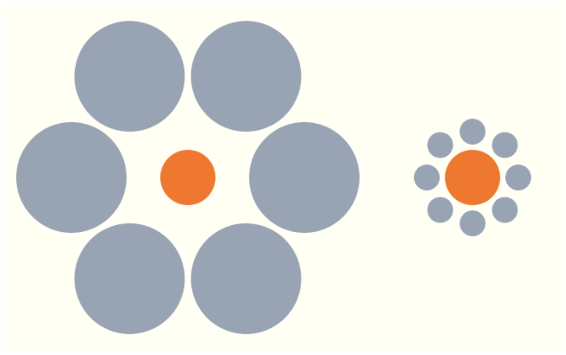


Figure 4 - 2: Ebbinghaus illusion. The two orange circles are exactly the same size; however, the one on the right appears larger.

4.2.1 Single Clouds Layout Algorithms

To accommodate different users needs, in our visual analytics system several layout algorithms has been included. In spite of their different approaches to the word placement, they still provide some continuity by only assigning a position to a word but not changing its appearance.

Spiral Layout

The first one was derived from Wordle and therefore has similarities as for example the use of a spiral. Other approaches have also recently used this spiral algorithm as it is relatively simple to implement and can produce satisfactory images [17]. MuWoC also follows Feinberg's *greedy* approach [39] as once a word has been placed, it will not move. It also uses a spiral that starts at the intended position of a word and goes outwards.

However MuWoC's implementation also has differences. There is no estimated cloud area, that is used to control the shape. This results in the placement of the words near their initial position, and the growth of the cloud into any shape. A second difference, is that the algorithm tries to put every word in the center of the cloud. Therefore, as the cloud grows, its shape becomes circular.

The approach with a common starting point also presents a problem in combination with the algorithm's greedy nature. The first word, would be in the center, whether it was important or not. Subsequent, maybe more relevant, words would be placed nearer to the outside and would be randomly distributed because they would have followed the spiral. This is not a wanted feature. The human vision has a focal point where it is sharpest, the so called *fovea*[100]. This point is at the center of the field of vision. The farther something is from the center, the more blurred it appears [99].
To keep important words together and most readable when looking at the entire cloud, they should be in the center. To achieve this, the words' list have to be ordered in a descending order, before they are placed in the cloud. This solves the previous problem and also add the distance from the center as an additional measure of importance for the user.

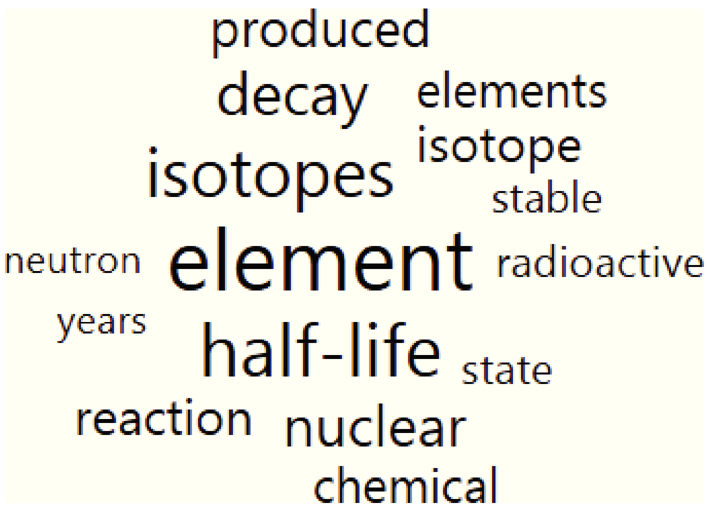


Figure 4 - 3: Sample result of the spiral layout algorithm.

Space station

The previous algorithm places words relatively near to the position they were supposed to be. However, following the spiral also meanes, that there might be space between the words, that could be filled.

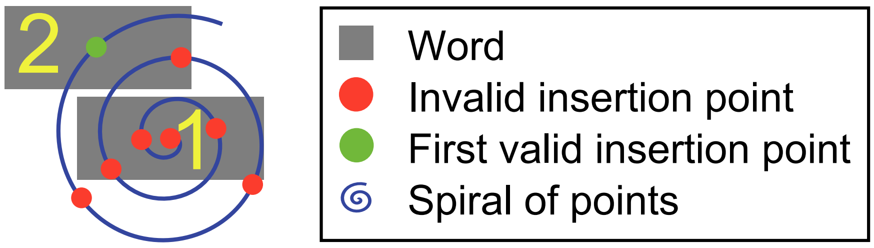


Figure 4 - 4: Placing words at selected points on a spiral leads to gaps.

Therefore a different layout algorithm was developed. The first basic rule is that a word's bounding box should always touch at least one other word's box to avoid wasting space. Further more, since the bounding box is a rectangle, it has four sides with which is can connect to other words.

To accommodate these rules, the 2-dimensional projection of a simplified space station can be used. Each word represents a module that has four docking ports: top, right, bottom and left. These ports are situated at the center of their respective sides of the bounding box. When words are placed in the cloud, they connect to other words by docking. To achieve a compact visual representation a new word should always dock at the port that is nearest to the center of the cloud.These rules lead are graphically explained in Figure 4 - 5.

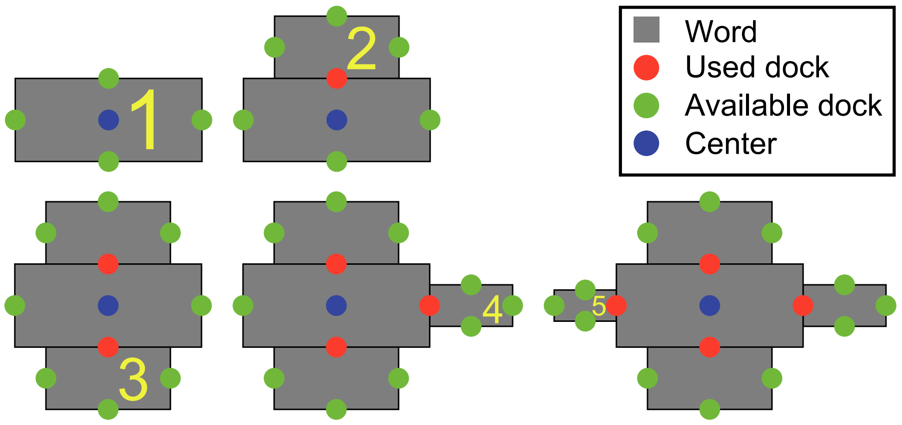


Figure 4 - 5: Explanation of the space station algorithm.

As with the previously mentioned spiral approach, the cloud's final size is not known before all words have been placed and therefore a coordinate translation is needed before the rendering phase (sample result in Figure 4 - 6).

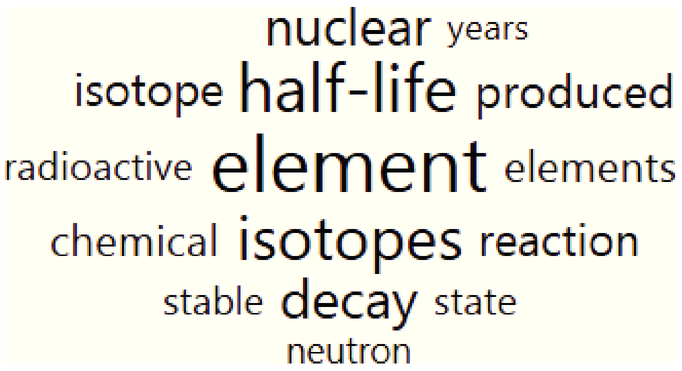

nuclear years
isotope half-life produced
radioactive element elements
chemical isotopes reaction
stable decay state
neutron

Figure 4 - 6: Sample result of the space station layout algorithm.

Vertical stack

All previously described placement algorithms have produced word clouds that form a dense cluster. There representations of words are well suited for standard clouds, but there are issues when working with multiple ones side by side. For example, it is difficult to compare the individual word's importance in one cloud with it's relevance in another. A vertical stack algorithm trades off the compact representation of a cloud for better comparability. As already mentioned above the words are ordered by relevance. Placing each word beneath the previous one therefore creates an ordered list. As visible in Figure 4 - 7, the user can choose a word and quickly compare its position in other clouds.

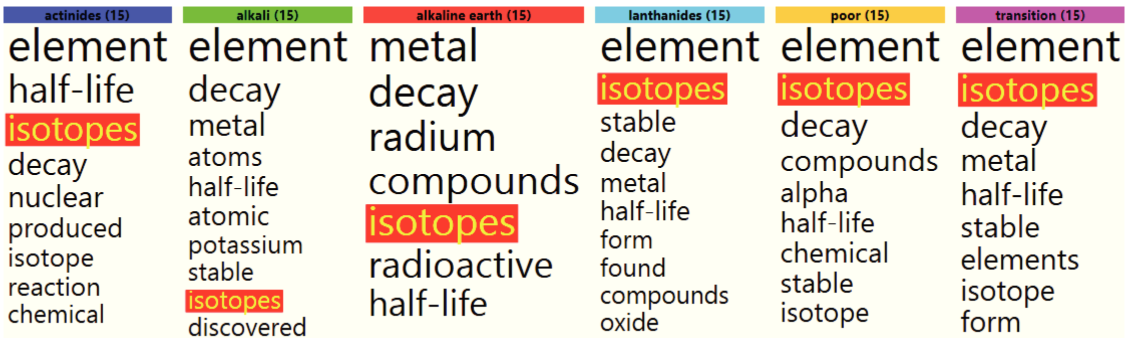


Figure 4 - 7: Vertical stack layout and synchronized word selection.

4.3 RadCloud

The algorithms previously presented produce standard separate clouds. The visualization of more than one single cloud in a single image can answer to simple comparison tasks. However, single clouds lack some basic, but useful, functionality:

Set comparison

Word clouds as mentioned until now are essentially sets of unique character tokens. Very prominent operations on sets are their comparisons. Euler and Venn diagrams provide visual representations that allow for fast comparability. But word clouds are always next to each other without overlapping, thus they get harder to compare as each single word from one cloud has to be searched in the others.

Relevance in categories

A word can be very important in one category and less relevant in another. To compare those importances the beholder first has to find the word in each cloud.

This can be very time consuming if there are many clouds or if they contain large amounts of words.

We develop a novel visualization techniques that add these functionality to a simple representation by using a unique word cloud. The visual analytics system allow the user to choose between two different visualization: multiple single clouds in one single image or a unique cloud that preserve and display information on the separate sets of words.

4.3.1 Previous Works

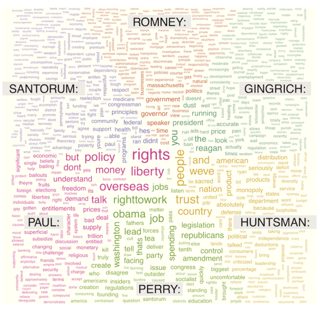
Castella and Sutton approached the issue of bad comparability by creating *word storms* with similarities between the clouds [25]. Their algorithm tries to maintain the words' size, position, color and orientation in all separate word clouds. This works well for a limited number of words and clouds. However, as the amounts increase it requires more and more comparative work from the users.

Collins and Carpendale created *VisLink* [31]. It helps with comparisons between many types of visualizations, as long as they represent the same data. To achieve this, the visualizations are drawn on a semi-transparent surface that can be moved in three dimensional space. The user can select rendered elements and VisLink then adds lines between those elements in one visualization and their representation in the other.

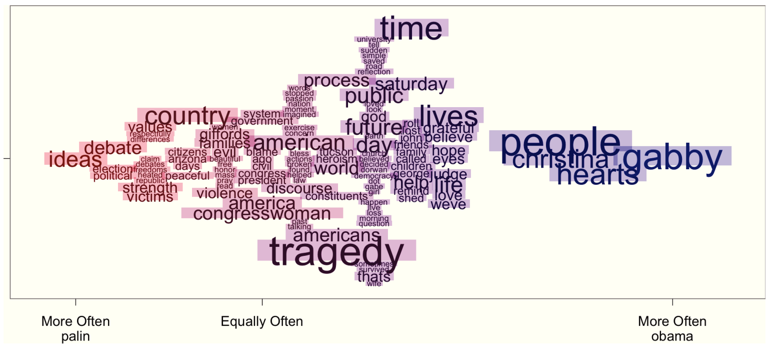
Collins et al. also developed *parallel tag clouds* [32]. This representation allows for fast comparisons while the individual clouds are laid out similarly to the vertical stack algorithm (see 4.2.1). However, the words still appear multiple times.

To solve both problems (*Set comparison* and *Relevance in categories*), the idea of a single cloud and separate ones for each category can be merged. Therefore providing only one cloud that contains the information about the belonging of words to the categories. Such a merged cloud should not contain multiple copies of the same words because it would force the user to search for all appearances: the merging should therefore be performed like the union operation in set theory.

The need for comparability between word clouds has risen especially in the context of sociology. In order to satisfy this requirement, there have already been multiple attempts for creating merged word clouds that contain the information about the source of the words. The origin can then in turn be used to deduct other information.



(a)



(b)

Figure 4 - 8: Examples of merged word clouds from **(a)** Ian Fellows' *Comparison Cloud* and **(b)** Winter Mason's *Comparative Word Cloud*.

Ian Fellows created the *comparison cloud* in the programming language of *R* [41]. *R* was developed for statistical calculations and is well suited for the task because of the similar calculations necessary for the analysis of statistical and textual data. The comparison cloud contains the differences between texts and merges them into one cloud. To separate the words from different sources, the available space is divided equally, resulting in a radial layout (see Figure 4 - 8(a)) [41].
However this visualization only depicts differences, thus it won't contain the same word in more than one cloud. This also means, that from a visual stand point each word belongs to exactly one cloud. When two arbitrary separate representations both contain the same word, this information is therefore lost in the process of combination.

Winter Mason developed a *Comparative Word Cloud* [86] and used on the transcripts of two speaches from Sarah Louise Palin and Barack Hussein Obama II (Figure 4 - 8(b)). Mason addressed the issues of merging with an implementation that also uses *R* and builds two clouds based on the appearance frequency of words from two source texts. These clouds can potentially have common words. When merging them, instead of using a random algorithm for an initial word position, he creates a horizontal scale. Each of the two documents represents an end of this scale. Collisions are then avoided by vertical movement. Mason's work therefore is very suitable to accomplish the goal of merging clouds, while also showing how much a word belongs to a category. But this approach is restricted to two clouds only.

A general distribution of information along an axis or scale has previously been described by Hoffman et al. as *dimensional anchors* [61]. These anchors are a generalization of Hoffman et al. previous work on *RadViz*, a visualization for multivariate data [60]. RadViz was created by merging two ideas:

- Ankerst et al. developed the *Circle segments* visualization, where the space in a circle was equally divided to contain single pixels [4].
- Olsen et al. were looking for a way to visualize documents in the context of a retrieval system. The result was *VIBE* [95].

VIBE's layout system attached a metaphorical approximation of physical springs between mobile documents and stationary points of interest (POI). The spring constants were then modeled by a measure of belonging or relevance of the documents in the context of certain interests [95]. To distinguish this type of springs from others that will be introduced later in this document, they will be labeled as being *positional*.

For RadViz, the positioning of points of interest was restricted to a circle. As in the *Circle Segments*, the points were equally distributed, to get a segmentation. Furthermore, Hoffman et al. transfered the calculation of spring constants from belonging to a POI to belonging to any value. This enables RadViz to visualize multidimensional data, where every rendered point has springs to all stationary dimension anchors [60].

4.3.2 Novel Technique

The RadViz visualization technique can be used to create merged word clouds: the RadCloud is born. Instead of rendering a single icon at a data points' position p_w, a word is drawn at p_w. Where Olsen et al. used points of interest and Hoffman et al.

dimensions, RadCloud uses categories. The spring constants are gathered by taking the word's relevance in each category.

RadCloud approaches the calculation of a word's position as follows. First when the data is loaded the words have to get a relative weight w'^{c}_{i} in each category:

$$
\begin{aligned}
\mathscr{C} &:= \{c_1, \ldots, c_u\} = \{\text{category}_1, \ldots, \text{category}_u\} = \text{all categories} \\
words^c &:= \text{ all words that belong to } \boldsymbol{category_c} \\
words &:= \bigcup words^c \\
w^c_i &:= \begin{cases} 0 & \text{if } w_i \notin words^c \\ weight(word_i, category_c) \in \mathbb{R} & \text{if } w_i \in words^c \end{cases} \\
w'^{c}_{i} &:= \frac{w^c_i}{\sum_{k=1}^{|words|} w^c_k} \quad \text{(normalized weight in category)}
\end{aligned}
\tag{4.1}
$$

The circle in which all words will be placed is centered around the coordinate system's origin. The categories are equally distributed along this circle. The individual positions are then obtained by rotating the point at the circle's top. The vectors V_c from the origin to the categories are then very simply created as:

$$
V_c := \overrightarrow{position(category_c)} \tag{4.2}
$$

The category positions and spring vectors are cached and reused as long as the render area, the category order or visibility do not change. The intended placement for each word is then obtained using:

$$
\begin{aligned}
w''^{c}_{i} &:= \frac{w'^{c}_{i}}{\sum_{k=1}^{|\mathscr{C}|} w'^{k}_{i}} \\
position(word_i) &:= \sum_{c=1}^{|\mathscr{C}|} \left(V_c \times w''^{c}_{i}\right)
\end{aligned}
\tag{4.3}
$$

Figure 4 - 9 illustrates the composition of the intended position by multiple separate vectors to the categories.

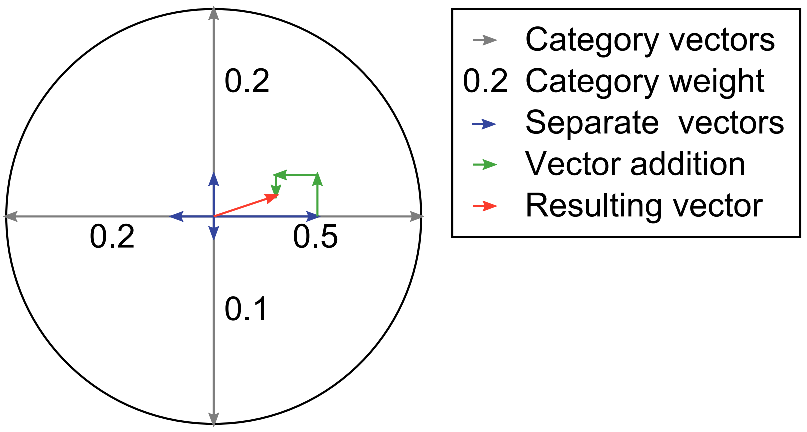


Figure 4 - 9: Intended position of a word in a 4-categories RadCloud.

While the RadCloud produces usable visualizations, it has room for improvement. Figure 4 - 10 show how the words' positions suffer from an ambiguity when more than three categories are displayed.

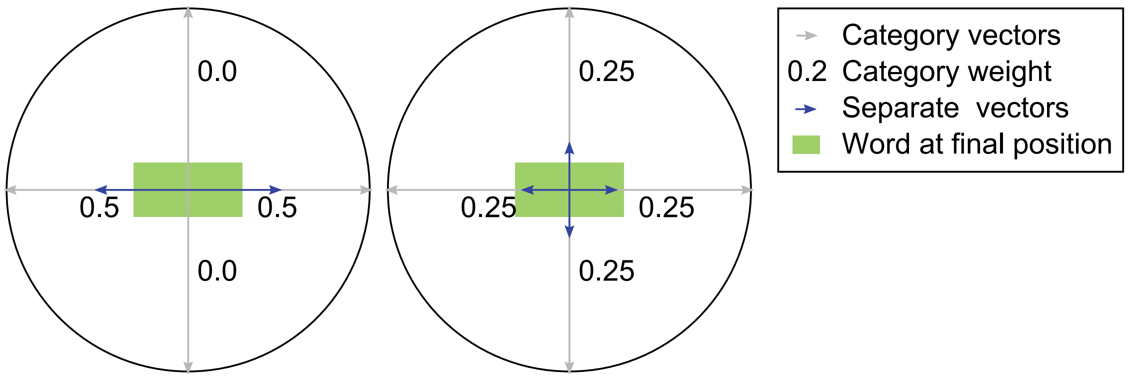


Figure 4 - 10: Example of the ambiguity of the positions in RadCloud (and also RadViz): two words at the same place but with different weights.

The difference between multiple possible interpretations of a word's placement could be decreased by rearranging the categories. For this, a new relation $\mathscr{R}_c$ would have to be defined on the set of categories $\mathscr{C}$. Categories with a high weight in their relations would be placed next to each other:

$$\begin{aligned} \mathscr{R}_c &:= \{(c_i, c_j) \colon c_i \in \mathscr{C} \wedge c_j \in \mathscr{C}\} \\ weight((c_j, c_k)) &:= \sum w_i^{c_j} + \sum w_i^{c_k} \end{aligned} \tag{4.4}$$

For our purpose additional features has been used to get rid of the ambiguity. The position of categories on the external circle is selected randomly. In the next paragraphs the use of color coding and interactive feature are shown.

4.4 A Visual Analytics System

A visual analysis tool for textual data was created, based on the word clouds presented in 4.2. This tool would extract words files and show them in either separate clouds or merged in the RadCloud. Furthermore it lists named entities that were previously recognized and also recognizes them itself. To show the original context of the single words or entities, it displays surroundings and entire documents.

To obtain visible word clouds, MuWoC uses four main phases: the recognition of named entities, the extraction of words, the determination of a word's relevance in a category and the final visualization.

4.4.1 Input Data Processing

The support for multiple source formats is part of the effort to make MuWoC a versatile and useful visual analytics tool for textual data. However, since texts are the main intended form of input, there has to be a way to distinguish documents from different categories. These categories are the base for the creation of multiple word clouds. Without the information about the belonging to categories, there would only be one cloud.

For a deeper understanding of the analyzed data it may be necessary to perform more complex tasks to get more information. A powerful means to gather further information is the named entity recognition.

Before plain text is used as a data source, the user can preprocess it with a named entity recognizer. This introduces new information for later use. The *Stanford Named Entity Recognizer* [119] from the Stanford University's *Natural Language Processing Group* has been used for this task.

After a standard phase of text extraction and text filtering (by using stop word removal) the system provide several metrics to determine words' relevance. Besides the well-known metrics used in text analysis as the *term frequency (TF)* and the *term frequency - inverse document frequency (TFiDF)*, a new one has been involved in the study of our VA system.

A new approach was implemented in which the TF and TFiDF serve as input for a support vector machine. Normally an SVM is used with textual data to sove a classification task [110]. But since SVM could determine the most relevant ones in a category as part of its classification process, is it possible to use the weights used as a metric of importance for a certain category. During normal operation with an SVM, it would first be trained and later the results would be used to categorize documents. However, to get the mentioned support vector the last step is not needed.

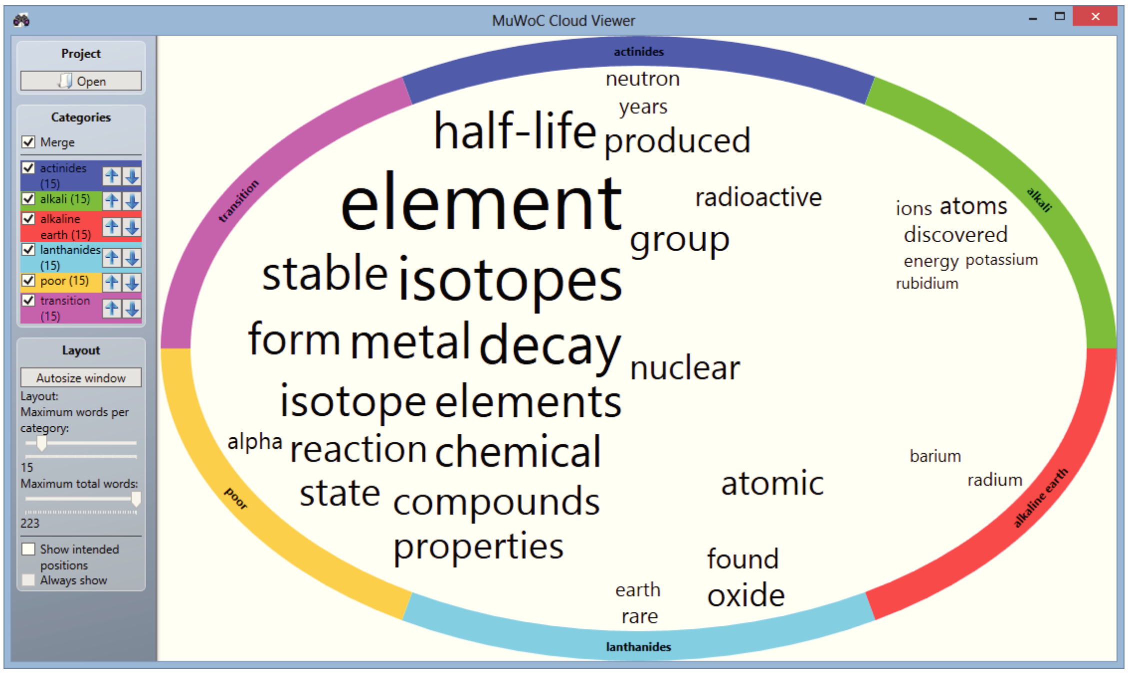
As mentioned the input documents for MuWoC are already categorized by their file location. An SVM expects vectors that are labeled as belonging to one or the other category. MuWoC creates a vector for each document. The dimensions are set by getting the entire list of all extracted words, essentially creating a bag of words representation [53] over all documents. The importance of each word in each category is then calculated as the sum of values that the SVM yielded for the vectors. After the training, the quality of the SVM's model is assessed by performing a five-fold cross validation on the training data itself. The result is then displayed, so the user can select other settings in case of an unsatisfactory accuracy. This is an important feature, because the more accurate the SVM classifies the documents, the better it discovered the words that distinguish them. Since Hirao et al. [59] have shown that an SVM is effective in finding the most important sentences, it should also be capable of discovering the most important words.

4.4.2 Clouds Viewer

Once the input text files are preprocessed, words and entities are extracted. Then the user can simply visualize in a single view a cloud for each text category. Otherwise, the user can merge the separate clouds into a single one: the system build up a unique cloud by using the *RadCloud* approach previously mentioned (Figure 4 - 11).

As the words' positions are dependent on how much they belong to each category, there should be a way to show this comparative information to the user in a less ambiguous manner.

MuWoC provides the possibility to underline each word with a horizontally stacked bar chart with predetermined width as shown in Figure 4 - 12. As it displays the relevance in each category, it is called *category bar*.



☑ Merge
☑ actinides (15)
☑ alkali (15)
☑ alkaline earth (15)
☑ lanthanides (15)
☑ poor (15)
☑ transition (15)
Layout:
Maximum words per category: 15
Maximum total words: 223
☐ Show intended positions
☐ Always show

Figure 4 - 11: Merged view, default visualization.

The viewer provide also a way to use color to encode information on the categories associated to each word. To obtain a foreground brush for the text display, the weight distribution across the categories has to be analyzed. If a word has equal relevance in each cloud equally, it does not belong specifically to only one. This results in an intended position near the middle. MuWoC maps this belonging and not-belonging to colors. When a word is only in one category, it inherits its color. As the word also belongs to other categories, the color gets less and less saturated, but does not change its hue. This way the color still shows to which category a word belongs most, but also signals that is present in others, too.

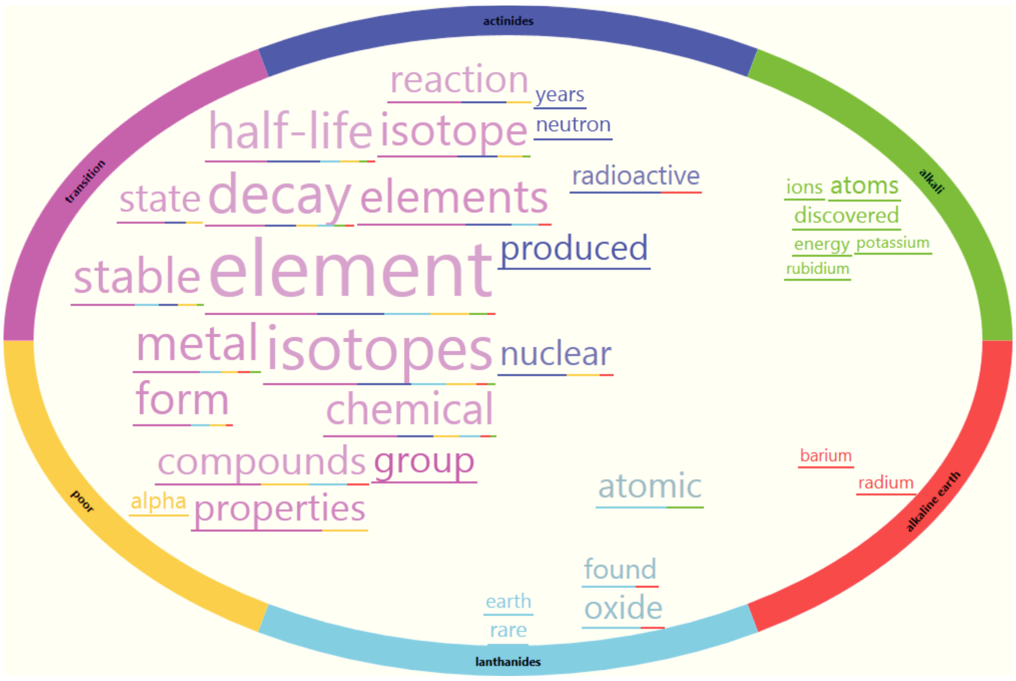
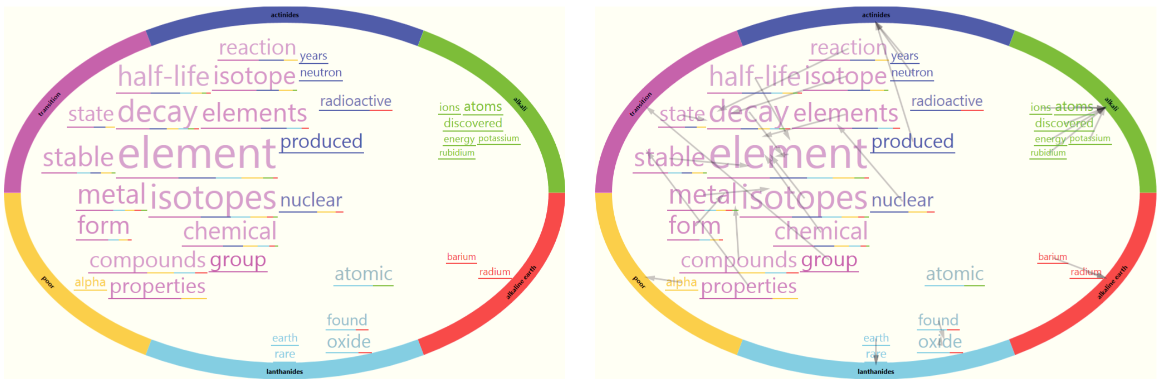


Figure 4 - 12: The merged view with word coloration, category bar and intended position display enabled.

The more a word belongs to multiple categories, the closer it is to the center and the less saturated it becomes. In order to calculate a metric of belonging to one or more categories, MuWoC uses the relative weights introduced in the RadCloud definition.

The main difficulty with the RadCloud is the avoidance of overlaps and the applied solution to this problem shifts words from there intended positions. To provide the user with the unmodified, original position, the side bar offers a check box that enables the display of an overlayed arrow to the selected word's intended location. A second option adds the same indicator to all words but renders them semi-transparently to preserve readability (see Figure 4 - 13). This feature also indicates how well the words were placed and therefore guide the user in the correct choice of RadCloud's settings.

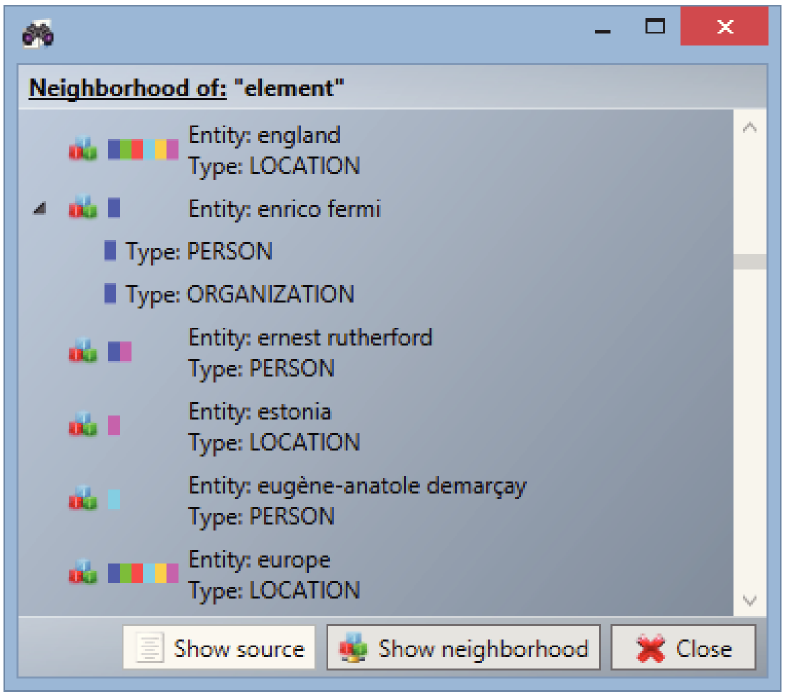


(a) (b)

Figure 4 - 13: The RadCloud with (b) and without (a) intended position indicators.

4.4.3 Entity Exploration

The cloud window provides overviews on the categories, their similarities and differences. Word clouds do not show any entities or connections between the displayed items. Thus the visual analytics system offers a means for data exploration, following several level of details. The entity explorer represent the first level, as it allows selecting individual entities. The linkage by neighborhood enables a second step in entities and words exploration. The second level only displays the neighborhoods from the source files, while the third level shows them entirely. For instance if a single forename is present in a word cloud, it is not clear which person was being referenced in the original source text. To get this information, the program can use the entities that were detected before extracting the single text tokens. When the user double clicks on a word in a cloud, MuWoC searches all entities for the inclusion of this exact string and displays them in the entity explorer. As shown in Figure 4 - 14, the explorer is a relatively simple window. The entities found are displayed in a list that is ordered by length: the shorter the entity name, the closer it matches the contained word. If there are multiple types of entities with the same name, the entry can be expanded to reveal all of them. Each list item also has an indicator that shows in which category it appears. It is comprised of rectangular icons that stacked horizontally and colored in accordance with the categories they represent.


Neighborhood of: "element"

Entity: england
Type: LOCATION

Entity: enrico fermi

Type: PERSON

Type: ORGANIZATION

Entity: ernest rutherford
Type: PERSON

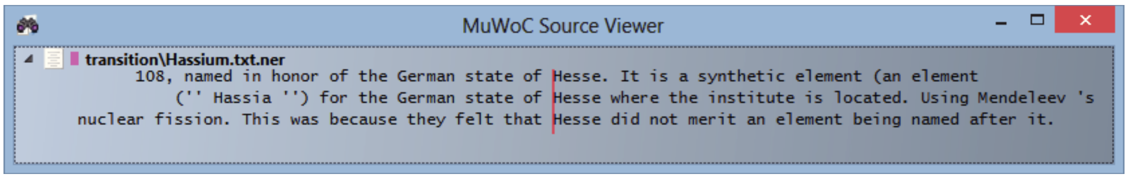
Entity: estonia
Type: LOCATION

Entity: eugène-anatole demarçay
Type: PERSON

Entity: europe
Type: LOCATION

Figure 4 - 14: MuWoC's entity explorer.

The entity neighborhood is calculated by using the same techniques addressed in Chapter 1. As a third level of navigation, the source viewer displays the neighborhoods of all appearances of a word or entity (see Figure 4 - 15). As it implements a concordance view [130], the main goal is to stack them vertically while aligning the target words. MuWoC places a vertical red line at the beginning of the target to highlight its position. To ensure good comparability between the lines of text, the UI font is replaced by a monospaced family.



transition\Hassium.txt.ner
```
        108, named in honor of the German state of Hesse. It is a synthetic element (an element
            ('' Hassia '') for the German state of Hesse where the institute is located. Using Mendeleev 's
 nuclear fission. This was because they felt that Hesse did not merit an element being named after it.
```

Figure 4 - 15: The source viewer with extracts from a single file.

Since the project might not contain all words that should be displayed, it has to access the original source files of the extraction process. MuWoC uses plug-ins to support a variety of file formats and therefore cannot read them directly on its own.

4.5 Case Studies

The main goal in MuWoC's development was to create a visual text analysis tool that uses word clouds and can provide useful information. To evaluate its ability to help find commonalities and differences between multiple texts, it has been tested on different source data sets.

4.5.1 Metals Description

The chemical elements that are categorized as metals belong to different groups. Alkali, alkaline earth, transition, actinides, lanthanides and poor metals are examples of such subcategories.

There is much scientific information available on chemical elements in general. It is therefore relatively easy to access it for this evaluation. Every groups contains multiple metals, which means that there are also multiple documents in each analysis category. The goal will be to find out whether a user can get information from the observation of the generated word clouds. The texts are taken from *Wikipedia, The Free Encyclopedia* using the public *MediaWiki* application programming interface. The source texts were first used for a named entity recognition and then for the word extraction. The support vector metric on the TFiDF yielded an accuracy of 89.5 %.

The merged view (see Figure 4 - 16) with 40 words from each category shows words like *alpha*, *beta*, *radioactive* and *years*. This could indicate that many metals are radioactive and mainly have half-lives in the spans of years. They probably emit mostly alpha and beta radiation when decaying.

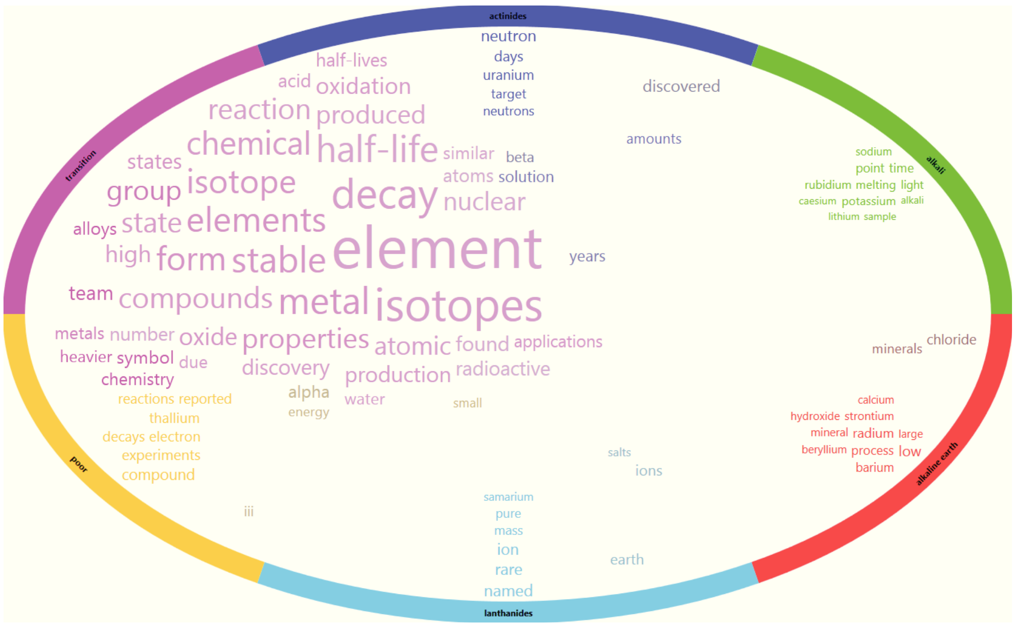


Figure 4 - 16: RadCloud of metals' information data, 40 words per category.

A word nearer to the actinide category is *neutron*. It indicates, that they also produce neutrons. This theory is supported by the fact, that Uranium is one of the metals in this group. It is well known that it is applied in various contexts of nuclear physics because of its emission of slow neutrons during the fission.
Another word near the actinide group is *target*. Further reading on Wikipedia reveals that elements with more Protons than Uranium were often produced by bombarding smaller atoms with alpha radiation.

The category of the alkali metals contains the word *light*. This suits them because of their relatively low density and firmness.

On the side of alkaline earth, words like *mineral*, *hydroxide* and *large* appear. This can probably be attributed to same reason, that also gave this group its name. Alkaline earth metals can be found in large quantities in the earth's crust. However, they usually don't appear in their pure form; oxides are much more common.

The group of poor metals did not contain special words that allowed fast access to its characteristics.

4.5.2 Jules Verne Selection

Jules Verne is an author of science fiction and adventure books. It would be interesting to find commonalities between the following selection of his works:

- Around The World In 80 Days
- 20,000 Leagues Under The Sea
- A Journey Into The Center Of The Earth
- The Moon-Voyage
- A Floating City and The Blockade Runners

The RadCloud on these texts is shown in Figure 4 - 17. Only three words are shared and often used in the contents of the books analyzed: *made* is a common word and its presence in the cloud is not significant, while *feet* and *time* denote the distinctive writing style of Jules Verne. The author write on science-fiction topics with a scientific approach, describing in details objects and vehicles' size or temporal aspects. Furthermore, for each novel the main topics are highlighted in the corresponding area of the cloud.

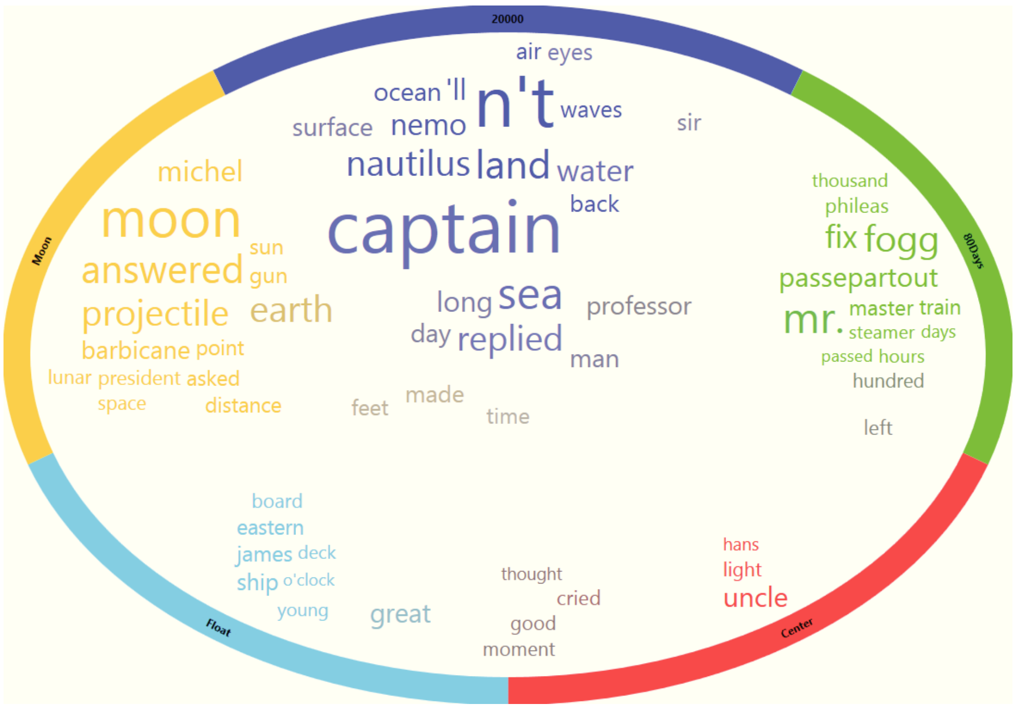


Figure 4 - 17: RadCloud of Jules Verne's novels.

4.5.3 Wes Andersons' Actors

A list of actors and their appearances in Wes Anderson's movies was manually created as a sample of MuWoC's ability to handle arbitrary CSV sources.

The file's columns correspond to the movies *Bottle Rocket*, *Rushmore*, *The Royal Tenenbaums*, *The Life Aquatic with Steve Zissou*, *The Darjeeling Limited*, *Fantastic Mr. Fox* and *Moonrise Kingdom*. Each actor is represented by a row: Noah Baumbach, Adrien Brody, Seymour Cassel, Brian Cox, Willem Dafoe, Michael Gambon, Jeff GoldBlum, Anjelica Huston, Harvey Keitel, Bill Murray, Edward Norton, Kumar Pallana, Jason Schwartzman, Andrew Wilson, Luke Wilson, Owen Wilson and Wallace Wolodarsky. The weights for each movie are binary, and code if an actor appear or not in a certain movie.

As Figure 4 - 18 shows, three actors appeared in only one film. The others were part of multiple projects. Bill Murray and Owen Willson even acted in six of the seven selected movies. When words converge on the same spot, it means that those actors worked together on the exactly the same films. An example are Gambon, Baumbach and Dafoe on *Fantastic Mr. Fox* and *The Life Aquatic with Steve Zissou*.

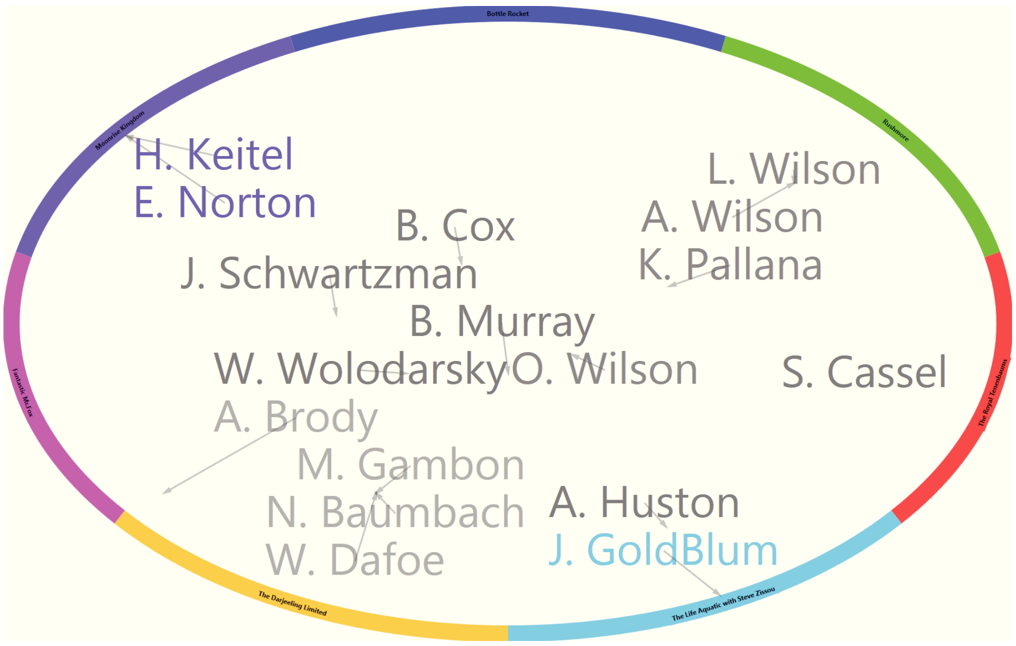


Figure 4 - 18: RadCloud of actors that worked in Wes Anderson's movies.

4.6 Conclusions

The visual analytics system presented in this chapter is shown to be a valuable resource to analyze texts by producing word clouds. The RadCloud novel approach did help in finding similarities and differences between multiple data sources or artificial categories found in data. Under certain hypothesis it seems to be more helpful to the user than separate traditional word clouds. The combination of a support vector machine weights as relevance metric and the radial visualization can give a good overview of textual data, then it simplify the task to search for significant insight into the source data. However, the quality is very dependent on the accuracy with which the support vector is able to categorize the documents. Further use case scenarios could compare how the TFiDF and the support vector weights perform in comparison. This could be especially interesting when the SVM does not produce satisfactory results.

Chapter 5

Generalized Pythagoras Trees for Visualizing Hierarchies

"There is no gene for fate."
- Vincent Freeman[Gattaca]

The following contents are object of the paper *Generalized Pythagoras Trees for Visualizing Hierarchies* by F. Beck, M. Burch, T. Munz, L. Di Silvestro and D. Weiskopf, presented at the *Conference on Information Visualization Theory and Applications (IVAPP) 2014*.

5.1 Introduction

Pythagoras Trees are fractals that can be used to depict binary hierarchies. But this binary encoding is an obstacle for visualizing hierarchical data such as file systems or phylogenetic trees, which branch into n subhierarchies. Although any hierarchy can be modeled as a binary one by subsequently dividing n-ary branches into a sequence of $n-1$ binary branches, we follow a different strategy. In our approach extending Pythagoras Trees to arbitrarily branching trees, we only need a single visual element for an n-ary branch instead of spreading the binary branches along a strand. Each vertex in the hierarchy is visualized as a rectangle sized according to a metric. We analyze several visual parameters such as length, width, order, and color of the nodes against the use of different metrics. The usefulness of our technique is illustrated by two case studies visualizing directory structures and a large phylogenetic tree. We compare our approach with existing tree diagrams and discuss questions of geometry, perception, readability, and aesthetics.

Hierarchical data occurs in many application domains, for instance, as results of a hierarchical clustering algorithm, as files organized in directory structures, or as species classified in a phylogenetic tree. Providing an overview of possibly large and deeply nested tree structures is one of the challenges in information visualization. An appropriate visualization technique should produce compact, readable, and comprehensive diagrams, which ideally also look aesthetically appealing and natural to the human eye.

A prominent visualization method are node-link diagrams, which are often simply denoted as *tree diagrams*; layout and aesthetic criteria have been discussed [103, 129]. Although node-link diagrams are intuitive and easy to draw, visual scalability and labeling often is an issue. An alternative, in particular easing the labeling problem, are indented trees [18] depicting the hierarchical structure by indentation. Further, layered icicle plots [80] stack boxes on top of each other for encoding a hierarchy, but waste space by assigning large areas to inner nodes on higher levels of the hierarchy. The Treemap approach [112], which is applying the concept of nested boxes, produces space-efficient diagrams but complicates interpreting the hierarchical structure.

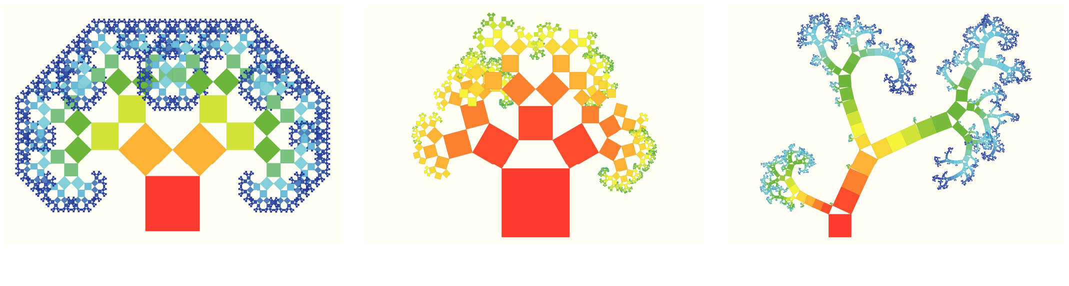


Figure 5 - 1: Extending Pythagoras Trees for encoding information hierarchies: (a) traditional fractal approach; (b) Generalized Pythagoras Tree applied to an n-ary information hierarchy; (c) additionally visualizing the number of leaves by the size of the inner nodes.

We introduce Generalized Pythagoras Trees as an alternative to the above hierarchy visualization techniques. It is based on Pythagoras Trees [11], a fractal technique showing a binary hierarchy as branching squares (Figure 5 - 1, a); the fractal approach is named after Pythagoras because every branch creates a right triangle and the Pythagorean theorem is applicable to the areas of the squares. We extend this approach to n-arily branching structures and use it for depicting information hierarchies (Figure 5 - 1, b). Instead of triangles, each recursive rendering step produces a convex polygonal shape where the corners are placed on a semi circle. The size of the created rectangles can be modified for encoding numeric information such as the number of leaf nodes of the respective subhierarchy (Figure 5 - 1, c). Different metrics and encodings exist, which we explore in detail.

We implemented the approach as an interactive tool and demonstrate its usefulness by applying it to large and deeply structured abstract hierarchy data from the NCBI taxonomy, a phylogentic tree that structures the living organisms on earth in a tree consisting of more than 300,000 vertices. Furthermore, a comparison to existing hierarchy visualization approaches provides first insights into the unique characteristic of Generalized Pythagoras Trees: a higher visual variety leads to more distinguishable visualizations, the fractal origin of the method supports identifying self-similar structures, and the specific layout seems to be particularly suitable for visualizing deep hierarchies. Finally, the created images are visually appealing as they show analogies to natural tree and branching structures.

5.2 Related Work

The visualization of hierarchical data is a central information visualization problem that has been studied for many years. Typical respresentations include node-link, stacking, nesting, indentation, or fractal concepts as surveyed by Jürgensmann and Schulz [69, 109]. Many variants of the general concepts exist, for instance, radial [7, 37] and bubble layouts [50, 82] of node-link diagrams, circular approaches for stacking techniques [3, 118, 132], or nested visualizations based on Voronoi diagrams[6, 94].

Although many tree visualizations were proposed in the past, none provides a generally applicable solution and solves all related issues. For example, node-link diagrams clearly show the hierarchical structure by using explicit links in a crossing-free layout. However, by showing the node-link diagram in the traditional fashion with the root vertex on top and leaves at the bottom, much screen space stays unused at the top while leaves densely agglomerate at the bottom. Transforming the layout into a radial one distributes the nodes more evenly, but makes comparisons of subtrees more difficult. Node-link layouts of hierarchies have been studied in greater detail, for instance, , Burch et al.[16] investigated visual task solution strategies whereas McGuffin and Robert [88] analyzed space-efficiency.

Indented representations of hierarchies are well-known from explorable lists of files in file browsers. Recently, Burch et al. [18] investigated a variant as a technique for representing large hierarchies as an overview representation. Such a diagram scales to very large and deep hierarchies and still shows the hierarchical organization but not as clear as in node-link diagrams. Layered icicle plots [80], in contrast, use the concept of stacking: the root vertex is placed on top and, analogous to node-link diagrams, consumes much horizontal space that is as large as all child nodes together.

Treemaps [112], a space-filling approach, are a prominent representative of nesting techniques for encoding hierarchies. While properties of leaf nodes can be easily observed, a limitation becomes apparent when one tries to explore the hierarchical structure because it is difficult to retrieve the exact hierarchical information from deeply nested boxes: representatives of inner vertices are (nearly) completely covered by descendants. Treemaps have been extended to other layout techniques such as Voronoi diagrams [6, 94] producing aesthetic diagrams that, however, suffer from high runtime complexity.

Also, 3D approaches have been investigated, for instance, in Cone Trees [24], each hierarchy vertex is visually encoded as a cone with the apex placed on the circle circumference of the parent. Occlusion problems occur that are solved by interactive features such as rotation. Botanical Trees [76], a further 3D approach, imitate the aesthetics of natural trees but are restricted to binary hierarchies, that is, n-ary hierarchies are modeled as binary trees by the strand model of of Holton [62]; it becomes harder to detect the parent of a node.

The term fractal was coined by Mandelbrot [85] and the class of those approaches has also been used for hierarchy visualization due to their self-similarity property [77, 78]. With OneZoom [105], the authors propose a fractal-based technique for visualizing phylogenetic trees; however, n-ary branches need to be visually translated into binary splits. Devroye and Kruszewski [35] visualize random binary hierarchies with a fractal approach as botanical trees; no additional metric value for the vertices is taken into account; instead, they investigate the Horton-Strahler number for computing the branch thicknesses.

The goal of our work is to extend a fractal approach, which is closer to natural tree structures, towards information visualization. This goal promises embedding the idea of self-similarity and aesthetics of fractals into hierarchy visualization. Central prerequisite—and in this, our novel approach differs from existing fractal approaches—is that n-ary branches should be possible. With respect to information visualization, the approach targets at combining advantages of several existing techniques: a readable and scalable representation, an efficient use of screen space, and the flexibility for encoding additional information. A downside of the approach, however, is that overlap may occur similar as in 3D techniques (though it is a 2D representation)—only varying the parameters of the visualization or using interaction alleviates this issue.

5.3 Visualization Technique

Our general hierarchy visualization approach extends the idea of Pythagoras Trees. Instead of basing the branching of subtrees on right triangles, we exploit convex polygons with edges on the circumference of a semi circle. This approach enables the visualization of n-arily branching hierarchies and introduces new degrees of freedom. But before we provide details on the generalized approach, we formally model the underlying data and give an overview of traditional Pythagoras Trees—these, in turn, are a special case of the proposed Generalized Pythagoras Trees.

5.3.1 Data Model

We model a hierarchy as a directed graph $H = (V, E)$ where $V = \{v_1, \ldots, v_k\}$ denotes the finite set of k vertices and $E \subset V \times V$ the finite set of edges, i.e., parent–child relationships. One vertex is the designated root vertex and is the only vertex without an incoming edge; all other vertices have an in-degree of one. We allow arbitrary hierarchies, that is, the out-degree of the vertices is not restricted. A maximum branching factor $n \in \mathbb{N}$ of H can be computed as the maximum out-degree of all $v \in V$. For an arbitrary vertex $v \in V$, H_v denotes the subhierarchy having v as root vertex; $| H_v |$ is the number of vertices included in the H_v (including v). The depth of a vertex v' in H_v is the number of vertices on the path through the hierarchy from v to v'. We allow positive weights to be attached to each vertex of the hierarchy $v \in V$ representing metric values such as sizes. We model them as a function $w : V \to \mathbb{R}$. The weight $w(v) \in \mathbb{R}^+$ of an inner vertex v does not necessarily need to be the sum of its children, but can be.

5.3.2 Traditional Pythagoras Tree

The Pythagoras Tree is a fractal approach describing a recursive procedure of drawing squares. In that, it was initially not intended to encode information, but its tree structure easily allows representing binary hierarchies: each square represents a vertex of the hierarchy; the recursive generation follows the structure of the hierarchy and ends at the leaves.

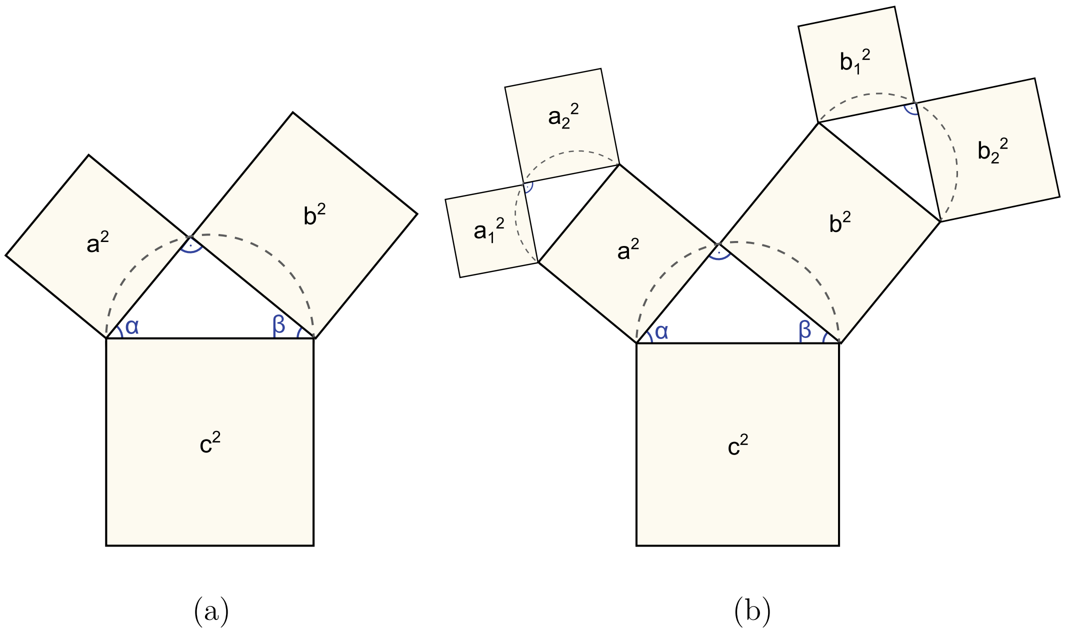


Figure 5 - 2: Illustration of the traditional Pythagoras Tree approach: (a) a single binary branch; (b) recursively applied branching step.

Drawing a fractal Pythagoras Tree starts with drawing a square of side length c. Then, two smaller squares are attached at one side of the square—usually, at the top—according to the procedure illustrated in Figure 5 - 2 (a): Then, a right triangle with angles α and β where $\alpha + \beta = \frac{\pi}{2}$ is drawn using the side of the square as hypotenuse, which also becomes a diameter of the circumcircle of the triangle. The two legs of the triangle are completed to squares having side lengths a and b. In the right triangle, the Pythagorean theorem $a^2 + b^2 = c^2$ holds, i.e., the sum of the areas of the squares over the legs is equal to the area of the square over the hypotenuse. Applying this procedure recursively to the new squares as depicted for the next step in Figure 5 - 2 (b) creates a fractal Pythagoras Tree (the recursion is only stopped for practical reasons at some depth). The angles α and β can be set to a constant value or be varied according to some procedural pattern. Figure 5 - 1 (a) provides an example of a fractal Pythagoras Tree where $\alpha = \beta = \frac{\pi}{4}$ and Figure 5 - 3 shows an instance with alternating angles α and β in each step; the depth of recursion is encoded as the color of the squares in both cases. Characteristic feature of these fractals is that visual objects are partially overlapping.

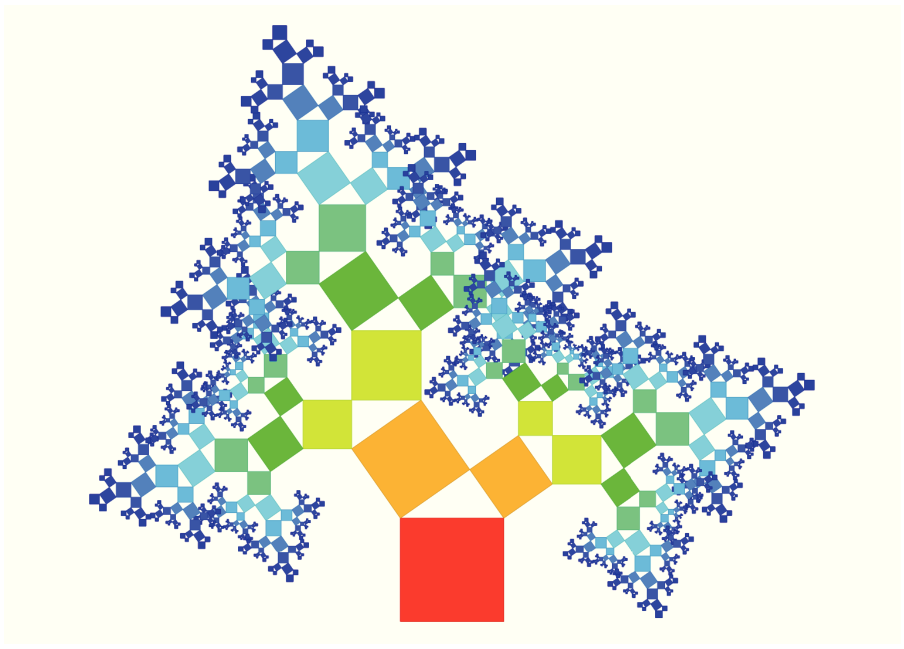


Figure 5 - 3: Fractal Pythagoras Tree using alternating 35° and 55° angles.

Transforming the fractal approach into an information visualization technique, the squares are interpreted as representatives of vertices of the hierarchy, called *nodes*. As a consequence, the fractal encodes a complete binary hierarchy, the recursion depth being the depth of the hierarchy. If the generated image should represent a binary hierarchy that is not completely filled to a certain depth, the recursion has to stop earlier for the respective subtrees. If the hierarchy is weighted as specified in the data model, the weights can be visually encoded by adjusting the sizes of the squares, i.e., the corresponding angles α and β.

Algorithm 1 describes in greater detail how an arbitrary binary hierarchy (i.e., a hierarchy where each vertex either has an out-degree of 2 or 0) can be recursively transformed into a Pythagoras Tree visualization. It is initiated by calling **PythagorasTree**(H_v, S)**:** where $H_v = (V, E)$ is a binary hierarchy and $S = (c, \Delta s, \theta)$ is the initial square with center c, length of the sides Δs, and slope angle θ. The recursive procedure first draws square S and proceeds if the current hierarchy still contains more than a single node. Then, encoding the node weights in the size of the squares, the angles α and β are computed according to the normalized weight of the node op-

Algorithm 1 Pythagoras Tree Visualization

```
**PythagorasTree($H_v, S$):**
  // $H_v$: binary hierarchy
  // $S$: representative square $S = (c, \Delta s, \theta)$
    // $c = (x_c, y_c)$: center
    // $\Delta s$: length of a side
    // $\theta$: slope angle
  drawSquare($S$);    // draw square for current root vertex
  if $| H_v | > 1$ then
    // $v_1$ and $v_2$: children of $H_v$
    $\alpha := \frac{\pi}{2} \cdot \frac{w(v_2)}{w(v_1)+w(v_2)}$;
    $\beta := \frac{\pi}{2} \cdot \frac{w(v_1)}{w(v_1)+w(v_2)}$;
    $\Delta s_1 := \Delta s \cdot \sin\beta$;
    $\Delta s_2 := \Delta s \cdot \sin\alpha$;
    $c_1 := ComputeCenterLeft(c, \Delta s, \Delta s_1, )$;
    $c_2 := ComputeCenterRight(c, \Delta s, \Delta s_2)$;
    $S_1 := (c_1, \Delta s_1, \theta + \alpha)$;
    $S_2 := (c_2, \Delta s_2, \theta - \beta)$;
    PythagorasTree($H_{v_1}, S_1$);    // draw subhierarchy $H_{v_1}$
    PythagorasTree($H_{v_2}, S_2$);    // draw subhierarchy $H_{v_2}$
  end if
```

posed to the angle. The angles form the basis for further computing the parameters of the two new rectangles S_1 and S_2. The drawing procedure is finally continued by recursively calling **PythagorasTree(H_{v_1}, S_1)** and **PythagorasTree(H_{v_2}, S_2)** for the two children v_1 and v_2 of the current root vertex.

When, for instance, using the number of leaf vertices as the weight of each vertex, the algorithm produces visualizations such as Figure 5 - 4 that encodes a random binary hierarchy. Like the fractal approach, the visualization algorithm still produces overlap of subtrees that, however, becomes rarer through sparser hierarchies.

5.3.3 Generalized Pythagoras Tree

The Generalized Pythagoras Tree, as introduced in the following, can be used for visualizing arbitrary hierarchies, that are hierarchies allowing n-ary branches. Right triangles are replaced by convex polygons sharing the same circumcircle; the former

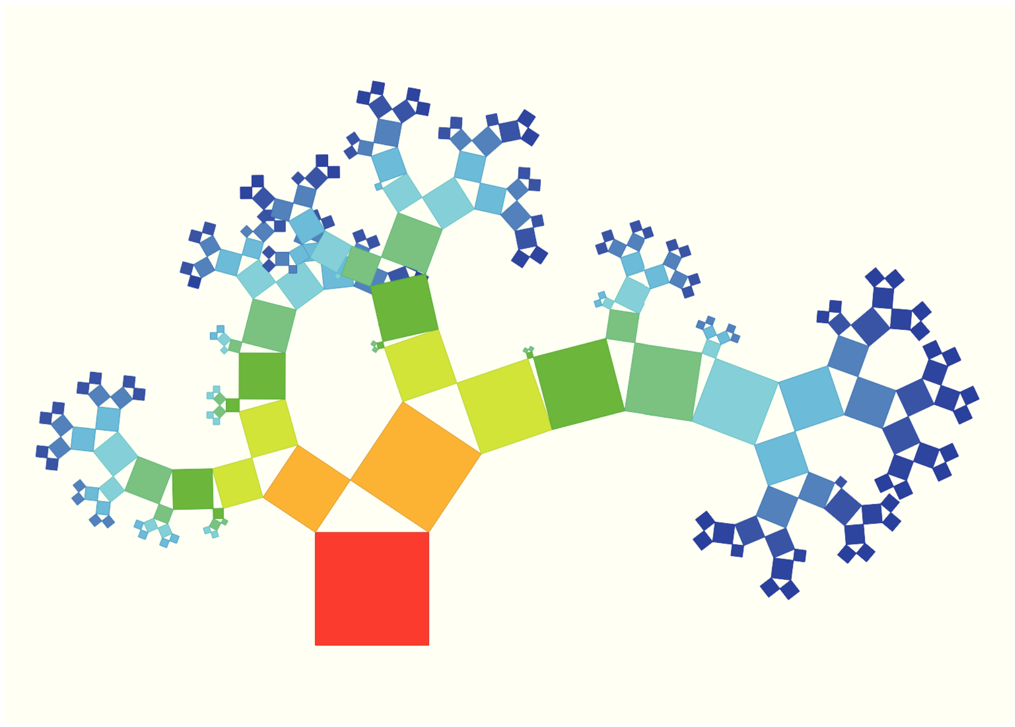


Figure 5 - 4: Random binary hierarchy visualized as a Pythagoras Tree that encodes the number of leaves in the size of the nodes.

hypotenuse of the triangle becomes the longest side of the triangle. For increasing the visual flexibility of the approach, squares are exchanged for general rectangles.

Figure 5 - 5 illustrates an n-ary branch, showing the polygon and its circumcircle. The polygon is split into a fan of isosceles triangles using the center of the circumcircle as splitting point. While the number of rectangles is specified by the degree of the represented branch, the angles and lengths can be modified to encode further information. In particular, we have two degrees of freedom:

- **Width function $w_x : V \to \mathbb{R}^+$ of rectangles**—Similar to binary hierarchies, the width Δx_i of a rectangle R_i can be changed, here, by modifying the corresponding angle α_i accordingly. The angle α_i should reflect weight $w_x(v_i)$ of a vertex v_i in relation to the weight of its siblings:

$$\alpha_i := \pi \cdot \frac{w_x(v_i)}{\sum_{j=1}^{n} w_x(v_j)} \quad .$$

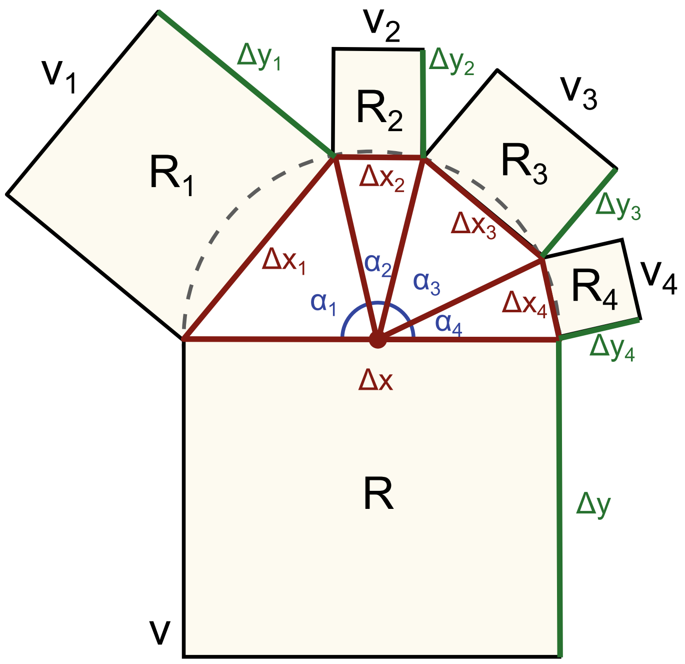


Figure 5 - 5: Polygonal split of Generalized Pythagoras Trees creating an *n*-ary branch.

The width of the rectangle is $\Delta x_i := \Delta x \cdot \sin\frac{\alpha_i}{2}$ where Δx is the width of the parent node.

- **Length stretch function w_y of rectangles**—Analogously, the length Δy_i of the rectangle R_i can be varied. This length, in contrast to the width Δx_i, does not underly any restrictions such as the size of a cirumcircle. Nevertheless, we formulate the length dependent on the length of the parent Δy and the relative width $\sin\frac{\alpha_i}{2}$ in order to consider the visual context (otherwise, it would be difficult to define appropriate metrics not producing degenerated visualizations): the length of the rectangle is $\Delta y_i := w_y(v_i) \cdot \Delta y \cdot \sin\frac{\alpha_i}{2}$.

Algorithm 2 extends Algorithm1 and describes the generation of Generalized Pythagoras Tree visualizations. Again, it is a recursive procedure and is initialized by calling **GeneralizedPythagorasTree**(H_v, R) where $H_v = (V, E)$ is an arbitrary hierarchy and $R = (c, \Delta x, \Delta y, \theta)$ represents the initial rectangle that, in contrast to the previous case, has a width Δx and a length Δy. For an *n*-ary branching hierarchy H_v with root vertex v, the algorithm first draws the respective rectangle before all children $v_1, \ldots, v_n$ are handled: for each child v_i, the computation of angle α_i forms

Algorithm 2 Generalized Pythagoras Tree Visualization

```
GeneralizedPythagorasTree($H_v, R$):
  // $H_v$: hierarchy branching into $n \in \mathbb{N}_0$ subhierarchies $H_{v_1}, \ldots, H_{v_n}$
  // $R$: representative rectangle $R = (c, \Delta x, \Delta y, \theta)$
    // $c = (x_c, y_c)$: center
    // $\Delta x, \Delta y$: width and length
    // $\theta$: slope angle
  drawRectangle($R$);    // draw rectangle for parent vertex
  for all $H_{v_i}$ do
    $\alpha_i := \pi \cdot \frac{w_x(v_i)}{\sum_{j=1}^{n} w_x(v_j)}$;
    $\Delta x_i := \Delta x \cdot \sin \frac{\alpha_i}{2}$;
    $\Delta y_i := w_y(v_i) \cdot \Delta y \cdot \sin \frac{\alpha_i}{2}$;
    $c_i :=$ComputeCenter$(c, \Delta x, \Delta y, (\alpha_1, \ldots, \alpha_{i-1}), \Delta x_i, \Delta y_i)$;
    $\theta_i :=$ComputeSlope$(\theta, (\alpha_1, \ldots, \alpha_i))$;
    $R_i := (c_i, \Delta x_i, \Delta y_i, \theta_i)$;
    GeneralizedPythagorasTree($H_{v_i}, R_i$);
  end for
```

the basis for deriving the width Δx_i and length Δy_i of the respective rectangle R_i as described above. Furthermore, the center and slope of the new rectangle need to be retrieved. Finally, **GeneralizedPythagorasTree**(H_{v_i}, R_i) can be recursively applied to subhierarchy H_{v_i} having rectangle R_i as root node.

Figure 5 - 6 shows a sample visualization created with the algorithm. For this initial image width function w_x is set to a constant value and the length stretch function w_y is defined as 1. As a consequence, the nodes are squares again, equally sized for each branch but n-arily branching. An example with a similar configuration can be found in Figure 5 - 1 (a); the same dataset is shown in Figure 5 - 1 (b) applying the number of leaf nodes as the width function w_x. Further configurations are discussed more systematically below. The discussion also includes the usage of color, which, in all figures referenced so far, visualizes the depth of the nodes. Furthermore, the order of rectangles can be modified and has an impact on the layout; in the generalized approach, we have a higher degree of freedom ($n!$ possibilities) than in the standard Pythagoras Trees where only a flipping between two angles can be applied.

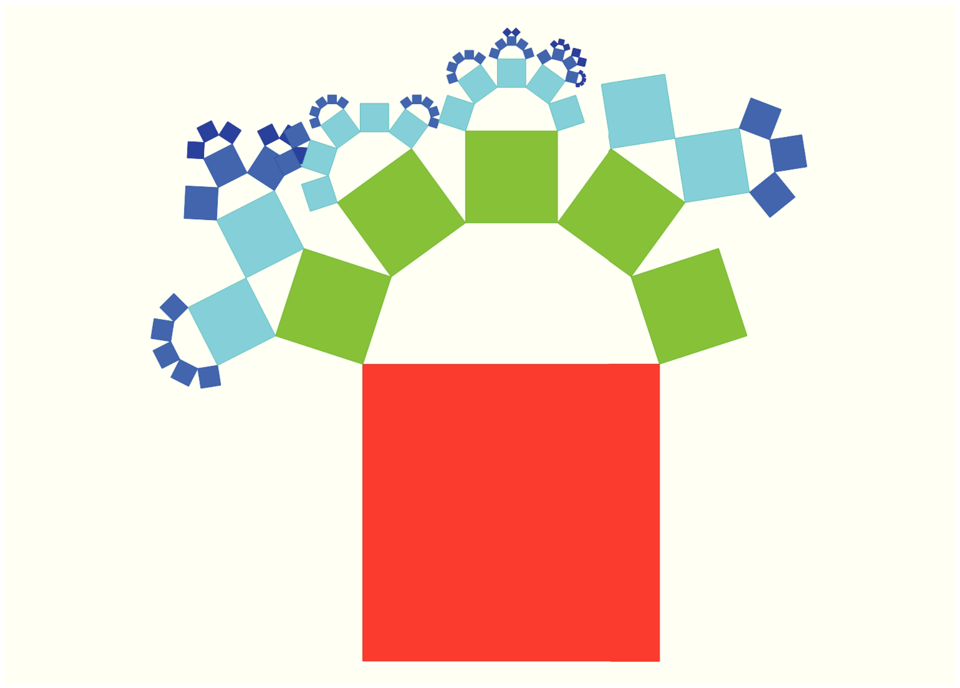


Figure 5 - 6: Generalized Pythagoras Trees showing n-ary hierarchy using a constant width and length stretch function.

5.3.4 Excursus: Fractal Dimension

The fractal dimension is typically used as a complexity measure for fractals. Looking back to the origin of the Generalized Pythagoras Tree visualization and interpreting it as a fractal approach, the extended fractal approach can be characterized by this dimension. To this end, however, not an information hierarchy can be encoded, but the approach needs to be applied for infinite n-arily branching structures; for simplification we do not consider scaling of rectangles. The following analysis shows that the fractal dimension, which is 2 for traditional Pythagoras Tree fractals, asymptotically decreases to 1 for a branching factor approaching infinity.

Any fractal can be characterized by its fractal dimension $D \in \mathbb{R}$ that is defined as a relation between the branching factor n and the scaling factor r given by $D = -\log_r n$. In our scenario, we have to first compute the scaling factor r depending on the branching factor n. Figure 5 - 7 illustrates the following formulas and shows an n-ary branch.

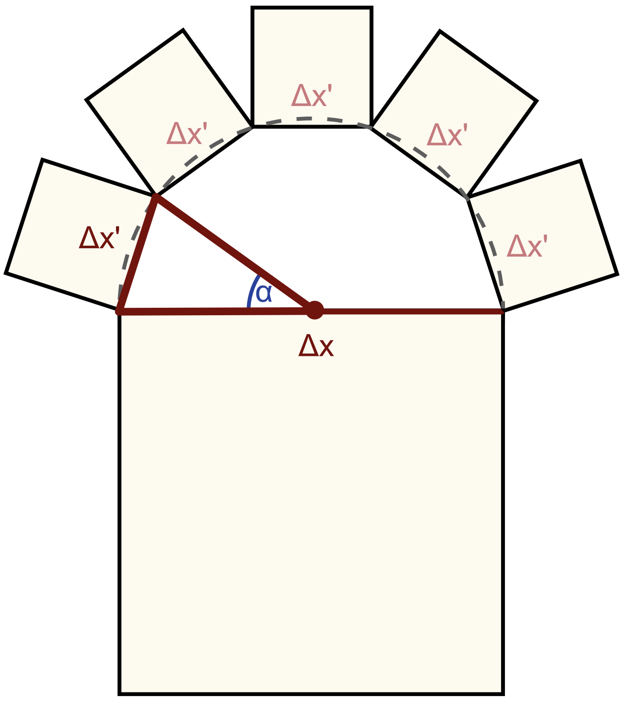


Figure 5 - 7: Illustrating the fractal dimension of an n-ary branching hierarchy by showing the splitting into equally sized angles.

First of all, the n-ary branch creates a convex polygon, which is split into isosceles triangles as described before. Since all rectangles have the same width, the angle at the tip of the triangle is $\alpha = \frac{\pi}{n}$. The width of the rectangle then is

$$\Delta x' = \Delta x \cdot \sin \frac{\alpha}{2} = \Delta x \cdot \sin \frac{\pi}{2n} \quad .$$

Relating the size of the square to the original square, the scaling factor can be derived as follows:

$$r = \frac{\Delta x'}{\Delta x} = \sin \frac{\pi}{2n} \quad .$$

The fractal dimension finally is

$$D_n = -\frac{\log n}{\log \sin \frac{\pi}{2n}} \quad .$$

This result confirms $D_2 = 2$ (traditional binary branches) and shows that the fractal dimension is approaching 1 for increasing n, i.e.,

$$\lim_{n \to \infty} D_n = 1 \quad .$$

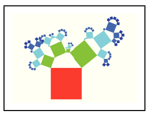
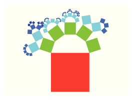
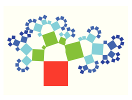
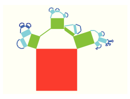
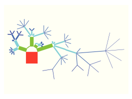
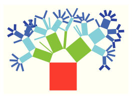
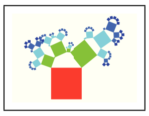
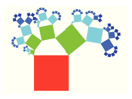
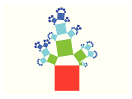
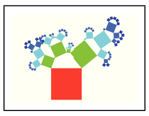
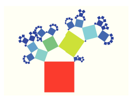
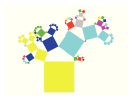
5.3.5 Visual Parameters

The visualization approach has been described precisely but still has some degrees of freedom that shall be explored in the following. For example, the size of the rectangles can be varied, the order of the subhierarchies in a branch is not restricted, or the coloring of the nodes is open for variation. These parameters help optimizing the layout and support the visualization by extra information in form of weights assigned to each node. For illustrating the effect, Table 5.1 shows the same random hierarchy (75 nodes; maximum depth of 5) in different parameter settings. As a weight, the number of leaf nodes is applied; but the metric is interchangeable, for instance, by the number of subnodes, the depth of the subtree, or a domain-specific weight. One setting (Table 5.1, S1 = O1 = C1), which seemed to work most universally in our experience, is selected as default and applied in all following figures of the chapter if not indicated otherwise.

Size

Already for the traditional Pythagoras Tree approach, rectangles can be split in uniform size or non-uniform size (Figure 5 - 3). For the generalized approach, we define a width function as well as a length function (Section 5.3.3). When employing the same metric for both, all nodes are represented as squares. Table 5.1 (S1) uses the number of leaf nodes as the common metric, which seems to be a good default selection because more space is assigned to larger subtrees. In contrast, when all subnodes are assigned the same size (i.e. a constant function is employed), small subtrees become overrepresented as depicted in Table 5.1 (S2). A variant of the approach, which is shown in Table 5.1 (S3), extends the approach from using semi circles to larger sectors of a circle.

Table 5.1: Exploring different parameter settings such as size, order, and color of rectangles for a sample dataset; framed images represent the default setting and are equivalent; the number of leaf nodes is applied as weight.

size (squares)	 (S1) sides: weight	 (S2) sides: equal size	 (S3) sides: weight; enlarged circle
size (rectangles)	 (S4) width: equal size; length: weight	 (S5) width: equal size; area: weight	 (S6) width: weight; area: weight
order	 (O1) external	 (O2) ascending weight	 (O3) maximum weight in the center
color	 (C1) depth	 (C2) weight	 (C3) category

Inserting different functions for width and length further increases the flexibility—nodes are no longer squares, but differently shaped rectangles. For instance, Table 5.1 (S4) encodes the number of leaf nodes in the height and applies a constant value to the width. When defining the length function relative to the (constant) width so that the area of the rectangle is proportional to the number of leaves, those leaf nodes are emphasized as depicted in Table 5.1 (S5). A similar variant shown in Table 5.1 (S6) has a constant length and chooses the width accordingly for encoding the number of leaf nodes in the area.

Order

The subnodes of an inner node of a hierarchy are visualized as an ordered list. While, for some applications, there exist a specific, externally defined order, many other scenarios do not dictate a specific order. In case of the latter, the subnodes can be sorted according to a metric, which again is the number of leaf nodes in this example. The sorting criterion mainly influences the direction in which the diagram is growing but also influences overlapping effects. Often the external order, at least in case it is random or independent of size, creates quite balanced views as depicted in Table 5.1 (O1). When, for instance, applying an ascending order, the image like the one shown in Table 5.1 (O2) grows to the right. More symmetric visualizations such as in Table 5.1 (O3) are generated when placing the vertices with the larger size in the center.

Color

The areas of the rectangular nodes can be filled with color for encoding some extra information. Selecting the color on a color scale according to the depth of the node in the hierarchy helps comparing the depth of subtrees: for instance, in Table 5.1 (C1) this encoding reveals that the leftmost main subtree, though being shorter, is as deep as the rightmost one. Alternatively, the weight of a node can be encoded in color like shown in Table 5.1 (C2), which, however, is more suitable if the size of the node not already encodes the weight. If categories of vertices are available, also these categories can be color-coded by discrete colors as depicted in Table 5.1 (C3).

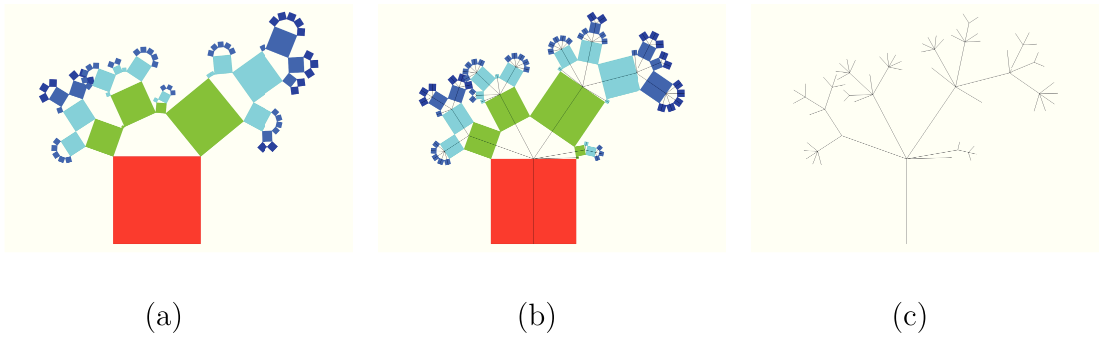


Figure 5 - 8: Relationship between Generalized Pythagoras Trees and node-link diagrams: (a) Generalized Pythagoras Tree; (b) Generalized Pythagoras Tree and analogous node-link diagram; (c) analogous node-link diagram.

5.3.6 Analogy to Node-Link Diagrams

Though being derived from a fractal approach, Generalized Pythagoras Trees can be adapted—without changing the position of nodes—to become variants of node-link diagrams. An analogous diagram can be created as illustrated in Figure 5 - 8 by connecting the circle centers of the semi circles of branches by lines. The circle centers become the nodes, the lines become the links of the resulting node-link diagram. Like the subtrees of a Generalized Pythagoras Tree might overlap, the analogous node-link drawing is not guaranteed to be free of edge crossings. We prefer the Pythagoras variant over the analogous node-link variant because it uses the available screen space more efficiently (which is important, for instance, for color coding) and shows the width of a node explicitly.

5.4 Case Studies

To illustrate the usefulness of our Generalized Pythagoras Tree visualization, we applied it to two datasets from different application domains—file systems with file sizes as well as the NCBI taxonomy that classifies species. In these case studies we demonstrate different parameter settings and also show how interactive features can be applied for exploration.

5.4.1 Phylogenetic Tree

Moreover, our approach is tested on a hierarchical dataset commonly used by the biology and bioinformatics communities. The taxonomy here used has been developed by NCBI and contains the names of all organisms that are represented in its genetic database [9]. The specific dataset encoding the taxonomy contains 324,276 vertices (60,585 classes and 263,691 species) and has a maximum depth of 42. The Generalized Pythagoras Tree visualization applied to this dataset (Figure 5 - 9 I) creates a readable overview visualization of the very complex and large hierarchical structure. The vertices of the tree have different sizes according to the number of leaves of their subtrees. Each inner vertex represents a class of species and it is easy to point out the class that contains more species. The root node is an artificial class of the taxonomy that contains every species for which a DNA sequence or a protein is stored in the NCBI digital archive.

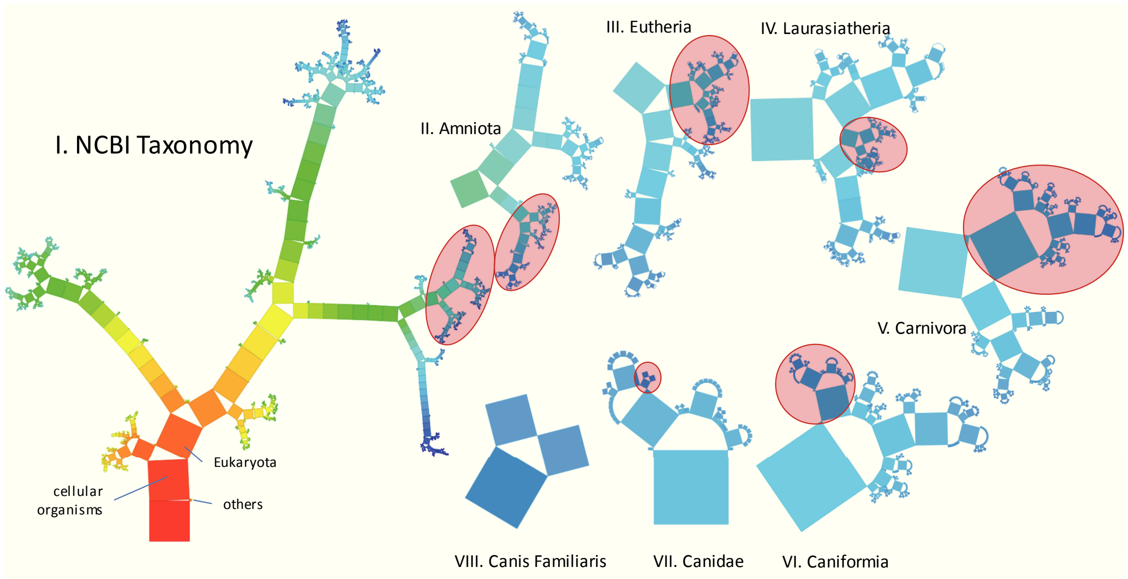


Figure 5 - 9: NCBI taxonomy hierarchically classifying species; rectangles sizes indicate the number of species in a subtree, color encodes the depth; an example for exploring the taxonomy by semantic zooming is provided.

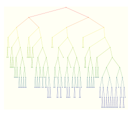
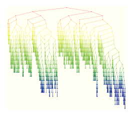
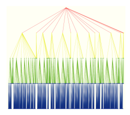
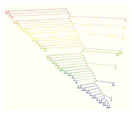
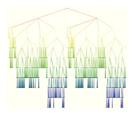
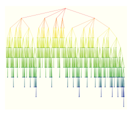
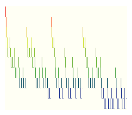
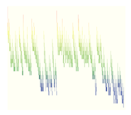
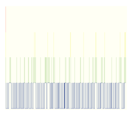
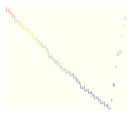
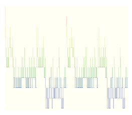
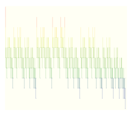
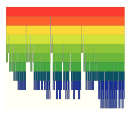
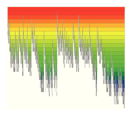
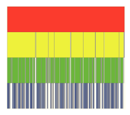
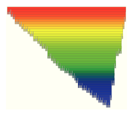
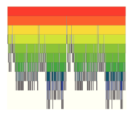
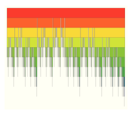
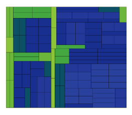
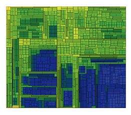
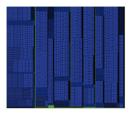
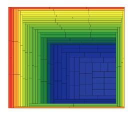
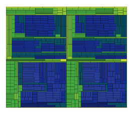
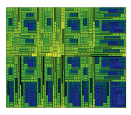
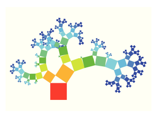
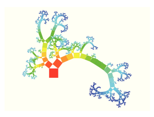
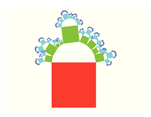
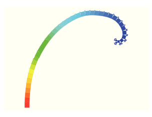
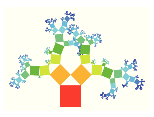
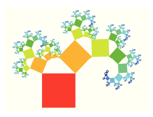
At the first level of the tree (see Figure 5 - 9 I), a big node represents *cellular organisms* and further nodes the *Viruses*, *Viroids*, unclassified species, and others (this information can be retrieved by using the geometric zoom). Selecting nodes and retrieving additional information facilitate the exploration of the tree. For instance, the biggest node at level 2 is the *Eukaryota* class, which includes all organisms whose cells contain a membrane-separated nucleus in which DNA is aggregated in chromosomes; it still contains 177,258 of the 263,691 species.

Besides gaining an overview of the main branches of the taxonomy, the visualization tool alllows for analyzing subsets of the hierarchy down to the level of individual species by applying semantic zooming. As a concrete example, we demonstrate the exploration process in the right part of Figure 5 - 9 (a more detailed documentation of this exploration is available as additional material); in each step we selected the subtree of the highlighted node (red circle): Figure 5 - 9 II shows the *Amniota* class, which belongs to the *tetrapoda vertebrata* taxis (four-limbed animals with backbones or spinal columns). In the next steps (Figure 5 - 9 III-V), we followed interesting branches until we reach the *Carnivora* class in Figure 5 - 9 V, which denotes meat-eating organisms; the subtree contains 301 species. From here, it is simple to proceed the exploration towards a well-known animal, such as the common dog, defined as *Canis Familiaris*, by zooming in the subtrees of *Caniformia*, literally “dog shaped” (Figure 5 - 9 VI), then through *Canidae*, the family of dogs (Figure 5 - 9 VII) with 45 species, and finally *Canis Familiaris*.

5.5 Discussion

The introduced technique for representing hierarchical structures is discussed by taking existing other hierarchy visualization approaches into account. We applied different standard hierarchy visualization techniques to a number of randomly generated artificial datasets. The results are listed in Table 5.2. Each column represents a different data set with some characteristic feature: a binary hierarchy with a branching factor of 2, a deep hierarchy with many levels, a flat hierarchy with a high maximum branching factor, a degenerated hierarchy that grows linearly in depth with the number of nodes, a symmetric hierarchy having two identical subtrees, and a self-similar hierarchy following the same pattern at each level. The rows show standard visualization techniques in comparison to Generalized Pythagoras Trees. Though the graphics can only act as previews in a printed version of the paper, they are included in high resolution and are explorable in a digital version (additionally, they are available as supplementary material). The following analysis considers multiple levels of abstraction from geometry and perception to readability and aesthetics.

Table 5.2: Comparison of hierarchy visualization approaches for representatives of a selected set of hierarchy classes.

	binary hierarchy	**deep hierarchy**	**flat hierarchy**	**degenerated hierarchy**	**symmetric hierarchy**	**self-similar hierarchy**
	node degree of 2	high number of hierarchy levels (25)	high maximum node degree (20)	linearly growing depth	two equivalent subtrees	self similar tree structure
node-link						
indented tree						
icicle plot						
Treemap						
Generalized Pythagoras Tree						

5.5.1 Geometry and Perception

Hierarchy visualizations aim at showing containment relationships between nodes and their descendants. Considering Gestalt theory [128], different approaches exist for visually encoding relationships: for instance, node-link diagrams use *connectedness* to express containment, while Treemaps are based on *common region* for showing that several nodes belong to the same parent. In contrast, Generalized Pythagoras Trees do neither directly draw a line between the nodes nor nest one node into the other, but they draw rectangles of decreasing size onto an imaginary curve. The human reader automatically connects the rectangles on the curve, which is denoted as the *law of continuation*. In all hierarchy visualization approaches shown in Table 5.2, *proximity* also plays a certain role (i.e., related nodes are placed next to each other) but should not be overinterpreted (i.e., nodes placed next to each other are not necessarily related).

In node-link diagrams, indented tree diagrams, or icicle plots, each level in the hierarchy creates another layer in the visualization. As a consequence, the amount of (vertical) space available for a layer is reduced when adding further levels. In Generalized Pythagoras Trees, however, there are no global layers for levels of nodes: adding a level only produces a kind of local layer that is arranged on a semi circle. With respect to this characteristic, Generalized Pythagoras Trees are similar to Treemaps, which neither have global layers but split the area of a node for introducing the next level.

Like in icicle plots and Treemaps, larger areas are used to encode the nodes in Generalized Pythagoras Trees. This makes it easier to use color for encoding some metrics (such as the hierarchy level) in the nodes because colors are easier to perceive for larger areas [126] (*Color for Labeling*). In contrast to Treemaps (and complete icicle plots), Generalized Pythogoras Trees do not create space-filling images. Areas, however, might overlap, which is discussed in detail below.

Comparing the images shown in Table 5.2 with respect to uniqueness, Generalized Pythagoras Trees show a high visual variety: not only the subtrees vary in size, they are also rotated. Only the splitting approach in Treemaps creates similarly varying images, however, just with respect to texture but not shape. A positive effect of a high visual variety is that the different datasets can be distinguished more easily—the visualization acts as a fingerprint. Together with the fractal roots of the approach, the uniqueness helps detect self-similar structures: Table 5.2 (last column) shows a tree having a self-similar structure, which is generated according to the same recursive, deterministic procedure for every node; the self-similar property of the hierarchy is best detectable in the Generalized Pythagoras Trees because every part of the tree is just a rotated version of the complete tree.

5.5.2 Readability and Scalability

A hierarchy visualization is readable if the users are able to efficiently retrieve the original hierarchical data from it and easily observe higher-level characteristics. However, readability is also related to visual scalability, which means preserving readability for larger datasets. While, for smaller datasets, the exact information is usually recognizable in any hierarchy visualization, the depicted information often becomes too detailed when increasing the scale of the dataset. The visualization approach, hence, needs to use the available screen space efficiently and has to focus on the most important information.

Generalized Pythagoras Trees clearly emphasize the higher-level nodes of the tree (i.e., the root node and its immediate descendants): most of the area that is filled by the visualization is consumed by these higher-level nodes, which can be easily perceived because surrounded by whitespace. Lower-level nodes and leaf nodes, however, become very small and are not visible. But the visualization allows for sizing the nodes according to their importance by using the number of leaf nodes as a metric as done in Table 5.2. Node-link diagrams, indented trees, and icicle plots are similar in their focus on the higher-level nodes; as well, lower-level nodes become difficult to discern because of lack of horizontal space. Since the vertical space assigned to each level does not become smaller in these visualizations, it is easier to retrieve the maximum depth of a subtree. Treemaps focus on leaf nodes and show largely different characteristics.

The ability of a visualization technique to display also large datasets in a readable way considerably widens its area of application. As shown in the case study, Generalized Pythagoras Trees can be used for browsing large hierarchies such as the NCBI taxonomy. While it is possible to interactively explore large hierarchies in a similar way with the other paradigms listed in Table 5.2, Generalized Pythagoras Trees show some characteristic scalability advantages: for specifically deep hierarchies such as the one in the second column of Table 5.2, it adaptively expands into the direction of the deepest subtree, here in spiral shape. Comparing it to the other approaches, deep subtrees are still readable in surprising detail. In contrast for flat hierarchies, which have a specifically high branching factor, Generalized Pythagoras Trees do not seem to be as suitable: the size of the nodes decreases too fast which constrains readability.

For a degenerated hierarchy (Table 5.2, fourth column), which grows linearly in depth with the number of nodes, Generalized Pythagoras Trees create an idiosyncratic but readable visualization, similar as it is the case for the other visualization approaches. Also a symmetry in a hierarchy such as two identical subtrees (Table 5.2, fifth column) can be detected: the identical tree creates the same image, which is rotated in contrast to the other approaches, where it is moved but not rotated.

A problem limiting the readability of Generalized Pythagoras Trees is that, depending on the visualized hierarchy, subtrees might overlap. The other visualization approaches do not share this problem; only Treemaps also employ a form of overplotting: inner nodes are overplotted by its direct descendants. While Treemaps use overplotting systematically, overlap only occurs occasionally in Generalized Pythagoras Trees and is unwanted. A simple way to circumvent the problem using the interactive tool is selecting the subset of the tree that is overdrawn by another. Also, reordering the nodes or adapting the parameters of the algorithm could alleviate the problem.

5.5.3 Aesthetics

Fractals often show similarities to natural structures such as trees, leaves, ferns, clouds, coastlines, or mountains [97]. Among the images shown in Table 5.2, the Generalized Pythagoras Trees clearly show the highest similarity to natural tree and branching structures. Since, according to the biophilia hypothesis, humans are drawn towards every form of life [131], this similarity suggests that Generalized Pythagoras Trees might be considered as being specifically aesthetic. Also the property of self-similarity that is partly preserved when generalizing Pythagoras Trees supports aesthetics: *"fractal images are usually complex, however, the propriety of self-similarity makes these images easier to process, which gives an explanation to why we usually find fractal images beautiful."* [83]

5.6 Conclusion and Future Work

We introduced an extension of Pythagoras Tree fractals with the goal of using these for visualizing information hierarchies. Instead of depicting only binary trees, we generalize the approach to arbitrarily branching hierarchy structures. An algorithm for generating these Generalized Pythagoras Trees was introduced and the fractal characteristics of the new approach were reported. A set of parameters allows for customizing the approach and creating a variety of visualizations. In particular, metrics can be visualized for the nodes. The approach was implemented in an interactive tool. A case study demonstrates the utility of the approach for analyzing large hierarchy datasets. The theoretical comparison of Generalized Pythagoras Trees to other hierarchy visualization paradigms, on the one hand, showed that the novel approach is capable of visualizing various features of hierarchies in a readable way comparably to previous approaches and, on the other hand, reveals unique characteristics of the approach such as an increased distinguishability of the generated images and detectabiltiy of self-similar structures. Further, the approach has advantages for visualizing deep hierarchies and provides natural aesthetics.

An open research questions is how the overplotting problem of the approach can be solved efficiently and how the assumed advantages can be leveraged in practical application. Moreover, formal user studies have to be conducted to further explore the characteristics of the approach.

Bibliography

[1] S. Afzal, R. Maciejewski, and D.S. Ebert. Visual analytics decision support environment for epidemic modeling and response evaluation. In *Proceedings of the IEEE Conference on Visual Analytics Science and Technology (VAST)*, pages 191–200, 2011.

[2] Wolfgang Aigner, Silvia Miksch, Heidrun Schumann, and Christian Tominski. *Visualization of Time-Oriented Data*. Human-Computer Interaction Series. Springer, 2011.

[3] Keith Andrews and Helmut Heidegger. Information slices: Visualising and exploring large hierarchies using cascading, semicircular disks. In *Proceedings of IEEE Symposium on Information Visualization*, pages 9–11, 1998.

[4] Mihael Ankerst, Daniel A. Keim, and Hans-Peter Kriegel. Circle Segments: A Technique for Visually Exploring Large Multidimensional Data Sets. In *Visualization '96, Hot Topic Session, San Francisco, CA*, 1996.

[5] Richard Arias-Hernandez, Linda T Kaastra, Tera M Green, and Brian Fisher. Pair analytics: Capturing reasoning processes in collaborative visual analytics. In *Proceedings of the 44th Hawaii International Conference on System Sciences (HICSS)*, pages 1–10, 2011.

[6] Michael Balzer, Oliver Deussen, and Claus Lewerentz. Voronoi treemaps for the visualization of software metrics. In *Proceedings of Software Visualization*, pages 165–172, 2005.

[7] Giuseppe Di Battista, Peter Eades, Roberto Tamassia, and Ioannis G. Tollis. *Graph Drawing: Algorithms for the Visualization of Graphs.* Prentice-Hall, 1999.

[8] K. K. Benke, F. Sheth, K. Betteridge, C. J. Pettit, and J.-P. Aurambout. A geo-visual analytics approach to biological shepherding: Modelling animal movements and impacts. *ISPRS Annals of Photogrammetry, Remote Sensing and Spatial Information Sciences*, I-2:117–122, 2012.

[9] Dennis A. Benson, Ilene Karsch-Mizrachi, David J. Lipman, James Ostell, and Eric W. Sayers. Genbank. *Nucleic Acids Research*, 37(Database-Issue):26–31, 2009.

[10] Jean Berstel and Luc Boasson. Transductions and context-free languages. *Ed. Teubner*, pages 1–278, 1979.

[11] Albert E. Bosman. Pythagoras tree, reprinted in Bruno Ernst: Bomen van Pythagoras - Variaties van Jos de Mey Aramith Uitgevers Amsterdam, 1985, 2004.

[12] Ulrik Brandes, Daniel Delling, Marco Gaertler, Robert Görke, Martin Hoefer, Zoran Nikoloski, and Dorothea Wagner. On finding graph clusterings with maximum modularity. In *IN PROC 33RD INTL WORKSHOP GRAPH-THEORETIC CONCEPTS IN COMPUTER SCIENCE (WG'07*, 2007.

[13] Sergey Brin. Extracting patterns and relations from the world wide web. In *Selected papers from the International Workshop on The World Wide Web and Databases*, WebDB '98, pages 172–183, London, UK, UK, 1999. Springer-Verlag.

[14] John Brooke. SUS – a quick and dirty usability scale. In P. W. Jordan, B. Thomas, B. A. Weerdmeester, and A. L. McClelland, editors, *Usability Evaluation in Industry*, pages 189–194. Taylor & Francis, London, 1996.

[15] A. Bruschi. *Metodologia delle scienze sociali.* Sintesi (Bruno Mondadori (Firm))). Bruno Mondadori, 1999.

[16] Michael Burch, Natalia Konevtsova, Julian Heinrich, Markus Höferlin, and Daniel Weiskopf. Evaluation of traditional, orthogonal, and radial tree diagrams by an eye tracking study. *IEEE Transactions on Visualization and Computer Graphics*, 17(12):2440–2448, 2011.

[17] Michael Burch, Steffen Lohmann, and Daniel Weiskopf. Prefix tag clouds. In *Proceedings of the 17th International Conference on Information Visualisation (IV 2013)*, pages 45–50, Los Alamitos, CA, USA, 2013. IEEE.

[18] Michael Burch, Michael Raschke, and Daniel Weiskopf. Indented Pixel Tree Plots. In *Proceedings of International Symposium on Visual Computing*, pages 338–349, 2010.

[19] M. Caccamo. *Management Parameters from the Random Regressions Testday Model to Advice Farmers on Cow Nutrition.* 2012.

[20] M. Caccamo, R.F. Veerkamp, G. de Jong, M.H. Pool, R. Petriglieri, and G. Licitra. Variance components for test-day milk, fat, and protein yield, and somatic cell score for analyzing management information. *Journal of Dairy Science*, 91(8):3268–3276, 2008.

[21] Stuart Card. The human-computer interaction handbook. chapter Information Visualization, pages 544–582. L. Erlbaum Associates Inc., Hillsdale, NJ, USA, 2003.

[22] Stuart K. Card, Jock D. Mackinlay, and Ben Shneiderman, editors. *Readings in Information Visualization: Using Vision to Think.* Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1999.

[23] John V. Carlis and Joseph A. Konstan. Interactive visualization of serial periodic data. In *Proceedings of the ACM Symposium on User Interface Software and Technology (UIST)*, pages 29–38, 1998.

[24] S. Jeromy Carrière and Rick Kazman. Research report: Interacting with huge hierarchies: beyond cone trees. In *Proceedings of Information Visualization*, pages 74–81, 1995.

[25] Quim Castella and Charles Sutton. Word storms: Multiples of word clouds for visual comparison of documents. *arXiv preprint arXiv:1301.0503*, 2013.

[26] Winnie Wing-Yi Chan. A survey on multivariate data visualization. Technical report, Department of Computer Science and Engineering. Hong Kong University of Science and Technology, 2006.

[27] R. Chang, C. Ziemkiewicz, T.M. Green, and W. Ribarsky. Defining insight for visual analytics. *IEEE Computer Graphics and Applications*, 29(2):14–17, 2009.

[28] Aaron Clauset, M. E. J. Newman, and Cristopher Moore. Finding community structure in very large networks. *Physical Review E*, 70(6):066111+, December 2004.

[29] Aaron Clauset, M. E. J. Newman, and Cristopher Moore. Finding community structure in very large networks. *Phys. Rev. E*, 70:066111, Dec 2004.

[30] William S. Cleveland and Robert McGill. An experiment in graphical perception. *International Journal of Man-Machine Studies*, 25(5):491–501, 1986.

[31] Christopher Collins and Sheelagh Carpendale. Vislink: Revealing relationships amongst visualizations. *Visualization and Computer Graphics, IEEE Transactions on*, 13(6):1192–1199, 2007.

[32] Christopher Collins, Fernanda B. Viegas, and Martin Wattenberg. Parallel tag clouds to explore and analyze faceted text corpora. In *Visual Analytics Science and Technology, 2009. VAST 2009. IEEE Symposium on*, pages 91–98. IEEE, 2009.

[33] Kris Cook, Rae Earnshaw, and John Stasko. Guest editors' introduction: Discovering the unexpected. *IEEE Computer Graphics and Applications*, 27(5):15–19, 2007.

[34] D. DeFelice, G. Giuffrida, G. Giura, and C. Zarba. La descrizione dei reati di mafia nel testo delle sentenze. *Quaderni di sociologia*, LIV(3):57–80, 2010.

[35] Luc Devroye and Paul Kruszewski. The botanical beauty of random binary trees. In *Proceedings of Graph Drawing*, pages 166–177, 1995.

[36] Kevin Dunbar and Kevin Dunbar. Scientists build models invivo science as a window on the science mind. In *Model-Based Reasoning in Scientific Discovery*, pages 85–99. Kluwer Academic/Plenum Publishers, 1999.

[37] Peter Eades. Drawing free trees. *Bulletin of the Institute for Combinatorics and its Applications*, 5:10–36, 1992.

[38] N. Elmqvist, P. Dragicevic, and J. Fekete. Rolling the dice: Multidimensional visual exploration using scatterplot matrix navigation. *IEEE Transactions on Visualization and Computer Graphics*, 14(6):1539–1148, 2008.

[39] Jonathan Feinberg. Wordle. In Noah Iliinsky Julie Steele, editor, *Beautiful Visualization*. O'Reilly Media, Inc., 2010.

[40] Jonathan Feinberg. Wordle. Computer software, 2013.

[41] Ian Fellows. Words in politics: Some extensions of the word cloud fells stats. Website, 2012.

[42] L. C. Freeman. *The Development of Social Network Analysis: A Study in the Sociology of Science*. Empirical Press, 2004.

[43] Linton C. Freeman. A Set of Measures of Centrality Based on Betweenness. *Sociometry*, 40(1):35–41, March 1977.

[44] D. Funk. *The Animal Model: Genetic Evaluation Procedures for Dairy Production Limits*. University of Wisconsin–Extension, 1989.

[45] David Galligan. Cowpad – Visual Analytics. `http://cahpwww.vet.upenn.edu/node/89`. Accessed: 22/02/2013.

[46] David Galligan. Dairy Dashboards – Visual Analytics. http://www.dgalligan.com/galliganx/visualanalytics/visualanalytics.html. Accessed: 22/02/2013.

[47] M. Girvan and M. E. J. Newman. Community structure in social and biological networks. *Proceedings of the National Academy of Sciences*, 99(12):7821–7826, June 2002.

[48] Google 2008. Gap Minder. http://www.gapminder.org. Accessed: 22/02/2013.

[49] Thomas Gottron. Document word clouds: Visualising web documents as tag clouds to aid users in relevance decisions. In *Research and Advanced Technology for Digital Libraries*, pages 94–105. Springer, 2009.

[50] S. Grivet, D. Auber, J. Domenger, and G. Melançon. Bubble tree drawing algorithm. In *Proceedings of International Conference on Computer Vision and Graphics*, pages 633–641, 2004.

[51] Edward Grundy, Mark W. Jones, Robert S. Laramee, Rory P. Wilson, and Emily L. C. Shepard. Visualisation of sensor data from animal movement. *Computer Graphics Forum*, 28(3):815–822, 2009.

[52] T Halasa, M Nielen, APW De Roos, R Van Hoorne, G De Jong, TJGM Lam, T Van Werven, and H Hogeveen. Production loss due to new subclinical mastitis in Dutch dairy cows estimated with a test-day model. *Journal of Dairy Science*, 92(2):599–606, 2009.

[53] Zellig S Harris. *Distributional structure.* Springer, 1981.

[54] J.A. Hartigan. Printer graphics for clustering. *Journal of Statistical Computation and Simulation*, 4(3):187–213, 1975.

[55] Susan Havre, Ieee Computer Society, Elizabeth Hetzler, Paul Whitney, and Lucy Nowell. Themeriver: Visualizing thematic changes in large document

collections. *IEEE Transactions on Visualization and Computer Graphics*, 8:9–20, 2002.

[56] Marti A. Hearst. Tilebars: Visualization of term distribution information in full text information access, 1995.

[57] K.P. Hewagamage, M. Hirakawa, and T. Ichikawa. Interactive visualization of spatiotemporal patterns using spirals on a geographical map. In *Proceedings of the IEEE Symposium on Visual Languages*, pages 296–303, 1999.

[58] Martin Hilbert and Priscila López. The world's technological capacity to store, communicate, and compute information. *Science*, 332(6025):60–65, 2011.

[59] Tsutomu Hirao, Hideki Isozaki, Eisaku Maeda, and Yuji Matsumoto. Extracting important sentences with support vector machines. In *Proceedings of the 19th international conference on Computational linguistics-Volume 1*, pages 1–7. Association for Computational Linguistics, 2002.

[60] Patrick Hoffman, Georges Grinstein, Kenneth Marx, Ivo Grosse, and Eugene Stanley. Dna visual and analytic data mining. In *Visualization'97., Proceedings*, pages 437–441. IEEE, 1997.

[61] Patrick Hoffman, Georges Grinstein, and David Pinkney. Dimensional anchors: a graphic primitive for multidimensional multivariate information visualizations. In *Proceedings of the 1999 workshop on new paradigms in information visualization and manipulation in conjunction with the eighth ACM internation conference on Information and knowledge management*, pages 9–16. ACM, 1999.

[62] Matthew Holton. Strands, gravity, and botanical tree imaginery. *Computer Graphics Forum*, 13(1):57–67, 1994.

[63] Edmund B. Huey. Preliminary experiments in the physiology and psychology of reading. *The American Journal of Psychology*, 9(4):575–586, 1898.

[64] Kevin Humphreys, Robert Gaizauskas, and Saliha Azzam. Event coreference for information extraction. In *IN PROCEEDINGS OF THE WORKSHOP ON OPERATIONAL FACTORS IN PRATICAL, ROBUST ANAPHORA RESOLUTION FOR UNRESTRICTED TEXTS, 35TH MEETING OF ACL*, pages 75–81, 1997.

[65] IBM Research. Many Eyes. `http://www-958.ibm.com`. Accessed: 22/02/2013.

[66] A. Inselberg. *Parallel Coordinates: Visual Multidimensional Geometry and Its Applications*. Springer, New York, 2009.

[67] J. Jamrozik and L.R. Schaeffer. Test-day somatic cell score, fat-to-protein ratio and milk yield as indicator traits for sub-clinical mastitis in dairy cattle. *Journal of Animal Breeding and Genetics*, 129(1):11–19, 2012.

[68] Neal F. Johnson. On the function of letters in word identification: Some data and a preliminary model. *Journal of Verbal Learning and Verbal Behavior*, 14(1):17–29, 1975.

[69] S. Jürgensmann and H.-J. Schulz. A visual survey of tree visualization. *IEEE Visweek 2010 Posters*, 2010.

[70] D.A. Keim and H.-P. Kriegel. Visualization techniques for mining large databases: a comparison. *IEEE Transactions on Knowledge and Data Engineering*, 8(6):923–938.

[71] Daniel Keim, Gennady Andrienko, Jean-Daniel Fekete, Carsten Görg, Jörn Kohlhammer, and Guy Melançon. Visual analytics: Definition, process, and challenges. In Andreas Kerren, JohnT. Stasko, Jean-Daniel Fekete, and Chris North, editors, *Information Visualization*, volume 4950 of *Lecture Notes in Computer Science*, pages 154–175. Springer Berlin Heidelberg, 2008.

[72] Daniel A. Keim. Information visualization and visual data mining. *IEEE Transactions on Visualization and Computer Graphics*, 8(1):1–8, 2002.

[73] Daniel A. Keim, Joern Kohlhammer, Geoffrey Ellis, and Florian Mansmann, editors. *Mastering The Information Age - Solving Problems with Visual Analytics.* Eurographics, November 2010.

[74] Daniel A. Keim, Florian Mansmann, Jörn Schneidewind, and Hartmut Ziegler. Challenges in visual data analysis. In *In Proceedings of the Tenth International Conference on Information Visualization*, pages 9–16, 2006.

[75] JF Keown and L Dale Van Vleck. Extending lactation records in progress to 305-day equivalent. *Journal of Dairy Science*, 56(8):1070–1079, 1973.

[76] Ernst Kleiberg, Huub van de Wetering, and Jarke J. van Wijk. Botanical visualization of huge hierarchies. In *Proceedings of Information Visualization*, pages 87–94, 2001.

[77] Hideki Koike. Generalized fractal views: A fractal-based method for controlling information display. *ACM Transactions on Information Systems*, 13(3):305–324, 1995.

[78] Hideki Koike and Hirotaka Yoshihara. Fractal approaches for visualizing huge hierarchies. In *Proceedings of Visual Languages*, pages 55–60, 1993.

[79] Robert Kosara, Helwig Hauser, and Donna L Gresh. An interaction view on information visualization. *State-of-the-Art Report. Proceedings of EUROGRAPHICS*, 2003.

[80] J. Kruskal and J. Landwehr. Icicle plots: Better displays for hierarchical clustering. *The American Statistician*, 37(2):162–168, 1983.

[81] C. H. Lewis. Using the “thinking aloud” method in cognitive interface design. Technical Report RC-9265, IBM, 1982.

[82] C. C. Lin and H. C. Yen. On balloon drawings of rooted trees. *Graph Algorithms and Applications*, 11(2):431–452, 2007.

[83] Penousal Machado and Amilcar Cardoso. Computing aesthetics. In *Advances in Artificial Intelligence*, volume 1515 of *Lecture Notes in Computer Science*, pages 219–228. Springer Berlin Heidelberg, 1998.

[84] Jock Mackinlay. Automating the design of graphical presentations of relational information. *ACM Trans. Graph.*, 5(2):110–141, April 1986.

[85] B.B. Mandelbrot. *The Fractal Geometry of Nature*. W.H. Freeman and Company. New York, 1982.

[86] Winter Mason. Comparative word cloud. Website, 2012.

[87] Bruce H. McCormick, Thomas A. DeFanti, and Maxine D. Brown (ed). *Visualization in Scientific Computing*. ACM SIGGRAPH, New York, 1987.

[88] M.J. McGuffin and J.M. Robert. Quantifying the space-efficiency of 2D graphical representations of trees. *Information Visualization*, 9(2):115–140, 2009.

[89] M Mellado, E Antonio-Chirino, C Meza-Herrera, F G Veliz, J R Arevalo, J Mellado, and A de Santiago. Effect of lactation number, year, and season of initiation of lactation on milk yield of cows hormonally induced into lactation and treated with recombinant bovine somatotropin. *Journal of Dairy Science*, 94(9):4524–4530, 2011.

[90] James Miller. Information input overload and psychopatology. *Am J Psychiatry*, 116(8):695–704, 1960.

[91] W. Müller and H. Schumann. Visualization methods for time-dependent data – an overview. In *Proceedings of the 2003 Winter Simulation Conference*, pages 737–745, 2003.

[92] Ran Nathan, Wayne M. Getz, Eloy Revilla, Marcel Holyoak, Ronen Kadmon, David Saltz, and Peter E. Smouse. A movement ecology paradigm for unifying organismal movement research. *Proceedings of the National Academy of Sciences*, 105(49):19052–19059, 2008.

[93] M E Newman. Modularity and community structure in networks. *Proc Natl Acad Sci U S A*, 103(23):8577–8582, June 2006.

[94] Arlind Nocaj and Ulrik Brandes. Computing voronoi treemaps: Faster, simpler, and resolution-independent. *Computer Graphics Forum*, 31(3):855–864, 2012.

[95] Kai A Olsen, Robert R Korfhage, Kenneth M Sochats, Michael B Spring, and James G Williams. Visualization of a document collection: The vibe system. *Information Processing & Management*, 29(1):69–81, 1993.

[96] W. Bradford Paley. Textarc: Showing word frequency and distribution in text. *Poster presented at IEEE Symposium on Information Visualization*, 2002, 2002.

[97] Heinz-Otto Peitgen and Dietmar Saupe, editors. *Science of Fractal Images.* Springer-Verlag, 1988.

[98] Ewa Ptak and LR Schaeffer. Use of test day yields for genetic evaluation of dairy sires and cows. *Livestock Production Science*, 34(1):23–34, 1993.

[99] Keith Rayner. The perceptual span and peripheral cues in reading. *Cognitive Psychology*, 7(1):65–81, 1975.

[100] Keith Rayner and James H. Bertera. Reading without a fovea. *Science*, 206(4417):468–469, 1979.

[101] Keith Rayner, Martin H. Fischer, and Alexander Pollatsek. Unspaced text interferes with both word identification and eye movement control. *Vision Research*, 38(8):1129–1144, 1998.

[102] R Reents, JCM Dekkers, and LR Schaeffer. Genetic evaluation for somatic cell score with a test day model for multiple lactations. *Journal of Dairy Science*, 78(12):2858–2870, 1995.

[103] E.M. Reingold and J.S. Tilford. Tidier drawings of trees. *IEEE Transactions on Software Engineering*, 7:223–228, 1981.

[104] Brian Roberts, Mike G Harris, and Tim A Yates. The roles of inducer size and distance in the ebbinghaus illusion (titchener circles). *PERCEPTION-LONDON-*, 34(7):847, 2005.

[105] J. Rosindell and L.J. Harmon. OneZoom: A fractal explorer for the tree of life. *PLOS Biology*, 10(10), 2012.

[106] Sunita Sarawagi. Information extraction. *Found. Trends databases*, 1(3):261–377, March 2008.

[107] L. R. Schaeffer and E. B. Burnside. Estimating the shape of the lactation curve. *Canadian Journal of Animal Science*, 56(2):157–170, 1976.

[108] L.R. Schaeffer, J. Jamrozik, G.J. Kistemaker, and J. Van Doormaal. Experience with a test-day model. *Journal of Dairy Science*, 83(5):1135–1144, 2000.

[109] H.-J. Schulz. Treevis.net: A tree visualization reference. *IEEE Computer Graphics and Applications*, 31(6):11–15, 2011.

[110] Fabrizio Sebastiani. Machine learning in automated text categorization. *ACM computing surveys (CSUR)*, 34(1):1–47, 2002.

[111] Judy Shamoun-Baranes, Roeland Bom, E. Emiel van Loon, Bruno J. Ens, Kees Oosterbeek, and Willem Bouten. From sensor data to animal behaviour: An oystercatcher example. *PLOS ONE*, 7(5):e37997, 2012.

[112] B. Shneiderman. Tree visualization with tree-maps: 2-D space-filling approach. *ACM Transactions on Graphics*, 11(1):92–99, 1992.

[113] B. Shneiderman. The eyes have it: a task by data type taxonomy for information visualizations. In *Proceedings of the IEEE Symposium of Visual Languages*, pages 336–343, 1996.

[114] Ben Shneiderman and Benjamin B. Bederson. *The Craft of Information Visualization: Readings and Reflections.* Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 2003.

[115] Marc A. Smith, Ben Shneiderman, Natasa Milic-Frayling, Eduarda Mendes Rodrigues, Vladimir Barash, Cody Dunne, Tony Capone, Adam Perer, and Eric Gleave. Analyzing (social media) networks with NodeXL. In *C&T '09: Proceedings of the fourth international conference on Communities and technologies*, C&#38;T '09, pages 255–264, New York, NY, USA, June 2009. ACM Press.

[116] Marc A. Smith, Ben Shneiderman, Natasa Milic-Frayling, Eduarda Mendes Rodrigues, Vladimir Barash, Cody Dunne, Tony Capone, Adam Perer, and Eric Gleave. Analyzing (social media) networks with nodexl. In John M. Carroll, editor, *C&T*, pages 255–264. ACM, 2009.

[117] T.L. Stanton, L.R. Jones, R.W. Everett, and S.D. Kachman. Estimating milk, fat, and protein lactation curves with a test day model. *Journal of Dairy Science*, 75(6):1691–1700, 1992.

[118] J. T. Stasko and E. Zhang. Focus+context display and navigation techniques for enhancing radial, space-filling hierarchy visualizations. In *Proceedings of the IEEE Symposium on Information Visualization*, pages 57–65, 2000.

[119] The Stanford Natural Language Processing Group. Stanford named entity recognizer (ner). Computer software, 2013.

[120] James J. Thomas and Kristin A. Cook. *Illuminating the Path: The Research and Development Agenda for Visual Analytics*. National Visualization and Analytics Ctr, 2005.

[121] Susan B Trickett, Wai-Tat Fu, Christian D Schunn, and J Gregory Trafton. From dipsy-doodles to streaming motions: Changes in representation in the analysis of visual scientific data. In *Proceedings of the Twenty-Second Annual Conference of the Cognitive Science Society*, pages 959–964, 2000.

[122] Barbara Tversky. Visuospatial reasoning. In K. Holyoak and R. Morrison, editors, *The Cambridge Handbook of Thinking and Reasoning*, pages 209–240. Cambridge University Press, 2005.

[123] Jarke J. Van Wijk and Edward R. Van Selow. Cluster and calendar based visualization of time series data. In *Proceedings of the 1999 IEEE Symposium on Information Visualization*, pages 4–9, 1999.

[124] Robert Voigt. An Extended Scatterplot Matrix and Case Studies in Information Visualization. Published as Diplomarbeit. Master's thesis, October 2002.

[125] Colin Ware. *Information visualization: perception for design.* Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 2004.

[126] Colin Ware. *Information Visualization, Second Edition: Perception for Design (Interactive Technologies).* Morgan Kaufmann, 2nd edition, 2004.

[127] Marc Weber, Marc Alexa, and Wolfgang Müller. Visualizing time-series on spirals. In *Proceedings of the IEEE Symposium on Information Visualization*, pages 7–13, 2001.

[128] Max Wertheimer. Untersuchungen zur Lehre von der Gestalt. II. *Psychological Research*, 4(1):301–350, 1923.

[129] Charles Wetherell and Alfred Shannon. Tidy drawings of trees. *IEEE Transactions on Software Engineering*, 5(5):514–520, 1979.

[130] Wikipedia. Concordance (publishing) — wikipedia, the free encyclopedia. Website, 2013.

[131] Edward O. Wilson. *Biophilia.* Harvard University Press, 1984.

[132] Jing Yang, Matthew O. Ward, Elke A. Rundensteiner, and Anilkumar Patro. InterRing: A visual interface for navigating and manipulating hierarchies. *Information Visualization*, 2(1):16–30, 2003.

[133] Ji Soo Yi, Youn ah Kang, John Stasko, and Julie Jacko. Toward a deeper understanding of the role of interaction in information visualization. *IEEE Transactions on Visualization and Computer Graphics*, 13(6):1224–1231, 2007.