Ontology-guided exploration of biomedical data repositories

Fritz Lekschas

Freie Universität Berlin

Department of Computer Science and Mathematics
Freie Universität Berlin
Kaiserswerther Str. 16-18
14195 Berlin, Deutschland

Master of Science Bioinformatics, Master Thesis.

Eidesstattliche Erklärung

Ich versichere hiermit an Eides Statt, dass diese Arbeit von niemand anderem als meiner Person verfasst worden ist. Alle verwendeten Hilfsmittel wie Berichte, Bücher, Internetseiten oder ähnliches sind im Literaturverzeichnis angegeben. Zitate aus fremden Arbeiten sind als solche kenntlich gemacht. Die Arbeit wurde bisher in gleicher oder ähnlicher Form keiner anderen Prüfungskommission vorgelegt und auch nicht veröffentlicht.

Datum Unterschrift

The research for this master thesis has been conducted in collaboration with Peter J. Park's and Nils Gehlenborg's laboratories at the Department for Biomedical Informatics at Harvard Medical School in Boston. Prof. Dr. Nils Gehlenborg from Harvard Medical School and Prof. Dr. Peter N. Robinson from the Free University of Berlin have supervised this project.

Submitted: February 2, 2016

Abstract

Over the last decade, technologies for measuring biological characteristics in high-throughput fashion have advanced dramatically, numerous novel methodologies have been invented and data accumulates at an ever-increasing rate. The community needs to handle petabytes of so-called ‘OMICS’ data alone and recent efforts are encouraging a shift from local to global cloud-based storage solution to facilitate sharing. A principal challenge engendered by this development is how data can be easily found and explored by researchers. Classic information retrieval system focus on finding specific data given some input query like keywords and mostly rely on string matching against a textual content. While this works well for actual text-based content it is less effective for biological data which primary content is often numeric; thus, applications rely on search across associated metadata. Related to the explosion of biological data generation, the community realized a need for standard description of the data masses in order to provide interoperability and reuse of data. This led to the development of several large biomedical ontologies. Efforts have been put into automated mining and annotation of biological data using ontologies, providing rich descriptions of the experimental setup and findings. Up-to-date ontologies are mostly used for analysis of data sets and extraction of specific knowledge via semantic queries. This thesis assesses the possibilities of using ontological annotations to visually describe the nature of data collections, such as whole repositories or subsets like search results, and to provide means of exploration along the semantic relationships of annotation terms.

I describe an integrative approach of ontology-guided visualization of metadata annotations across a data repository in combination with a modern text-based search and data set previewing. A prototypical implementation demonstrates the feasibility and enhancements compared to existing system. Additionally, a case study evaluates the general effectiveness of the proposed method given a real-world data collection.

Acknowledgements

First and foremost, I would like to express my deep gratefulness to Nils Gehlenborg for his tremendous support and excellent supervision over the course of my master's thesis. Nils' broad knowledge in bioinformatics and data visualization, his excellent management skills and thoughtful advices have been a great source of inspiration throughout my stay. Second, I would like to thank Peter J. Park for making my visit at his group possible and for supporting me throughout my stay. It has been an unbelievably insightful visit. I would also to express my gratitude to Peter N. Robinson for co-supervising my thesis, prior guidance and in general many fantastic lectures throughout my undergraduate and graduate program.

Furthermore, I am grateful to the members of Peter J. Park's and Nils Gehlenborg's group as well as the whole Refinery Platform team for their support along this project. I am especially indebted to Burak Han Alver, Jean Fan, Youngsook Jung, Scott Kallgren, Minseok Kwon, Eunjung Lee, Semin Lee, Soohyun Lee and Jia Wang who volunteered for interviews and helped be gain insights in many small peculiarities in search behavior regarding different fields of research in computational biology. I would also like to thank Harald Stachelscheid for taking part in the interviews and giving me many thoughtful advices relating important characteristics in stem cell biology.

Additionally, I am deeply grateful to my parents who always encouraged and supported me with all the hurdles that are engendered when living and working abroad.

This thesis would not have been possible without the substantial help from all you great people. I deeply appreciate your help.

To my mom and dad.

Contents

Abstract iv

Acknowledgements v

Contents vii

List of Figures ix

List of Tables x

List of Acronyms xi

1 Introduction 1

1.1 The problem 1

1.2 Importance of the problem 2

1.3 Research questions and scope of the thesis 3

1.4 Proposed Solution 3

1.5 About this document 3

2 Background 5

2.1 Search principles 5

2.1.1 Information retrieval 5

2.1.2 Information seeking 9

2.2 Visualization principles 11

2.2.1 Graph visualization 16

2.2.2 Set-typed visualization 17

2.3 Ontologies 19

2.3.1 OBO and OWL 19

2.4 Graph databases 22

2.5 Related work 23

3 Methods 25

3.1 Data 25

3.1.1 Data abstraction 26

3.1.2 Data Processing 27

3.2 Design 28

3.2.1 Requirements & Tasks 28
3.2.2 Interviews 32
3.2.3 Tree map 37
3.2.4 List graph 40
3.2.5 Preview 44
3.2.6 Interactions 45
3.3 Implementation 46
3.3.1 OWL to Neo4J parser 47
4 Results **49**
4.1 Application 49
4.2 Case study 57
5 Discussion **60**
5.1 Conclusion 60
5.2 Limitations 60
5.3 Outlook 61
Bibliography **62**

List of Figures

2.1 Holistic process of information retrieval 6
2.2 Example analysis of an English sentence 8
2.3 Precision, recall and MAP 9
2.4 Summary of the classic information seeking process 11
2.5 Anscombe's Quartet. 12
2.6 Overview of basic visual marks and channels 12
2.7 Effectiveness of different visual channels 13
2.8 The nine Gestalt principle of perceptual grouping 15
2.9 Salience of marks 16
2.10 Network visualization methods 16
2.11 Screenshot of current 2D visualization techniques for tree data 17
2.12 Basic principles of set-typed visualization 19
2.13 Example and differences of RDF and RDFS 21
2.14 Example property graph 23
3.1 Abstract data model 26
3.2 Graph manipulations 28
3.3 Expected user exploration behavior 30
3.4 Graph to tree and tree to tree map conversions 37
3.5 Tree map zoom 38
3.6 Tree map coloring, labeling and spacing 40
3.7 Layout comparison of node-link diagrams 41
3.8 Term property visualization via superimposed bar charts 42
3.9 Tree map versus list graph 43
3.10 Different search result outcomes 44
3.11 Data set preview integration 45
3.12 Early mock-up of the final exploration interface 46
4.1 Landing page of the Refinery Platform 49
4.2 Exploration interface 51
4.3 Column-wise sorting 51
4.4 Data set-wise annotation term highlighting 52
4.5 Term hovering and locking 53
4.6 Browsing by branch 54
4.7 Tree map's level zoom 55
4.8 Annotation term querying and 2-bar visualization mode 56
4.9 Data set preview panel 57
4.10 Exploring high-level annotation sets 58
4.11 Annotation set overview 59

List of Tables

1 Ontology usage across Stem Cell Commons 25
2 Evaluation of data set characteristics 36
3 Graph database schema constraints. 48
4 Description of the four different query modes 55

List of Acronyms

BTO	BRENDA Tissue and Enzyme Source Ontology
CHEBI	Chemical Entities of Biological Interest
DBMS	Database Management Systems
EFO	Experimental Factor Ontology
EQP	Existential Quantification Property
FMA	Foundational Model of Anatomy
GO	Gene Ontology
IR	Information Retrieval
ISA	Investigation Study Assay
JS	JavaScript
MA	Mouse Adult Gross Anatomy
NCBITAXON	National Center for Biotechnology Information Taxonomy
NCIT	National Cancer Institute Thesaurus
OBO	Open Biomedical Ontologies
OWL	Web Ontology Language
RDF	Resource Description Framework
RDFS	RDF Schema
RO	OBO Relations Ontology
URI	Uniform Resource Identifier
W3C	World Wide Web Consortium

1 Introduction

The topic of this thesis is the exploration of biomedical data repositories through integrated search and visual exploration along text-based and ontology-guided metadata.

1.1 The problem

During the last decade technologies for measuring biological characteristics such as the genome, transcriptome or metabolome have evolved dramatically and the number of data sets generated is steadily growing. Today, the community is facing petabytes of so-called "-omics" data (Eisenstein 2015) alone. Managing and finding the right data becomes more and more involved. Large repositories like Gene Expression Omnibus (Edgar et al. 2002) or ArrayExpress (Kolesnikov *et al.* 2015) host a variety of different functional genomics data sets, including gene expression, genome variation, methylation, non-coding RNA, and many other profiles. The first step in managing the large amount of data was taken in 2001 when the Minimum Information About a Microarray Experiment guidelines have been introduced (Brazma *et al.* 2001), which later became the standard for microarray data annotations. Along with guidelines for the description of functional genomics data, controlled vocabularies have been developed that further increase the expressiveness and avoid ambiguities. Taxonomies and ontologies are the most commonly used controlled vocabularies for annotation of biomedical data. While taxonomies provide some structure and unambiguous naming, ontologies offer rich ways to semantically describe the relationships of concepts and enable the inference of new knowledge. The Gene Ontology (GO) (Ashburner *et al.*, 2000) was the first major biomedical ontology that quickly obtained high popularity and widespread in the biomedical community. Among others, the Experimental Factor Ontology (EFO) (Malone *et al.*, 2010) is extensively used to capture various experimental conditions underlying the data generation and is hosted at the European Bioinformatics Institute in the United Kingdom. The main advantage of utilizing ontologies for annotation of biomedical data is twofold: Frist, controlled globally unique vocabulary avoids naming ambiguities and fosters reuse of existing knowledge. Second, the underlying mathematical logics enable automated inference of new information. Having a controlled description of the data is the first crucial step to ensure the usefulness of data in the future.

Today most data repositories solely rely on textual representation of their content. Users have the possibility to query the application against keywords and retrieve results holding a minimal amount of information, e.g. title, author, date etc., to provide a first insight. Internally, systems match keywords against their document corpus and return a ranked list of decreasing relevance. The document's relevance is

determined via complex formulas that evaluate weighted frequencies of keyword matches within a document against the total collection. Though, some systems experiment with data-driven search algorithms (Fujibuchi *et al.*, 2007), the majority relies on text-based queries. Whether users find relevant data is subject to two characteristics: findability (Morville, 2005) and discoverability. The first term describes how easy it is to locate elements of information in a known space while the latter describes the efficiency to discover (new) information in an unknown space. Both criteria need to be met in order for a repository to be fully explorable.

A major challenge in finding the right data is that users are not directly operating on the biological data itself but instead on metadata of descriptive texts (e.g. summaries, protocols, etc.). There are many ways to express the same ideas and while controlled vocabularies are often used to simultaneously search for synonyms of query keywords, the relation between different search results remains hidden. It is not obvious why certain documents are ranked higher than others. Also, if the exact keyword to be searched is unknown it is challenging to find relevant data. While word ambiguities referring to the same concept can be solved well by inclusion of synonym collections such as the Medical Subject Headings (Rogers, 1963). Making data sets related to higher-level or lower-level terms explorable in purely text-based search applications is more challenging because it is not known in which situation the user wants to include related data. Also, even closely related concepts might have distinct names, making autosuggestions less intuitive. For example, should a search for *epithelial cell* retrieve or suggest data related to *podocyte*—also known as *renal glomerular visceral epithelial cell*—a subclass of *epithelial cell*? Another major problem is that there is no simple way to understand the nature of the repository or subsets of it, such as search results. Most repositories only specify the total number of hits or very generic statistics like the total number of data sets. Data repositories that have been thoroughly annotated with ontological terms have the potential to provide meaningful context, can aid better understanding of relationships to find hidden data, reveal trends, and ultimately facilitate semantic exploration. While there are many tools available for exploring data, most of them work in isolation or focus on a single task only. Up to date, there is no tool available that addresses the different aspects of search and exploration in an integrated manner.

1.2 Importance of the problem

The sheer amount of biomedical data and the high complexity of experimental setups increase difficulties in finding relevant data. Finding the right data is key to answer biological questions and deriving new hypothesis in computational biology and beyond. Having better ways of exploration can highlight patterns and improve overall reusability of data.

1.3 Research questions and scope of the thesis

The main research question of this thesis is how the exploration of ontologically annotated biomedical data collections can be improved in two ways: better understanding of search results and the composition of the repository as a whole. Hereby, exploration is meant to be the act of browsing and searching to find known and unknown data and to discover novel information. The goal of this thesis is to provide a method to visualize the semantic composition of annotations and allow for ontology-guided querying and filtering alongside with text-based search.

The scope of this thesis is to conduct a requirement analysis, review previous work on user interfaces that deal with repository exploration, and to develop new ways of browsing and exploring heterogeneous biomedical data repositories. The requirement analysis is intended to describe the needs and required tasks of the exploration approach and to categorize these according to different user groups. Related work in the field of repository exploration should provide an overview of the current state of research and point out known limitations that need to be addressed. A novel visual exploration approach should be developed, implemented and integrated into the Refinery Platform as a proof of principle. Finally, the prototype is to be evaluated using real-world data.

1.4 Proposed Solution

When searching for text documents search engines return the keywords found in their natural context, enabling the user to reason about the document's semantics derive relevance. Without a context it is difficult to judge whether a retrieved document really matches the user's needs and whether it fits to the user's mental model. An increasing number of biomedical data collections have been annotated with ontological term but in regards to text-based search those terms might not match or might not be familiar to the user. Similar to how the most relevant query keywords are presented within their context, the annotation terms themselves can be put into context by showing the semantic connection using their ontological relationships. In order to facilitate the understanding of search results, a novel list-like compact graph visualization has been combined with a tree map visualization to provide both: sensemaking of the current search results, providing an overview of the repository, and to enable ontology-guided querying to facilitate semantic exploration.

1.5 About this document

This thesis is roughly structured into three parts. Chapter 2 introduces the core concepts of search, visualization, and ontologies and describes related work. Chapter 3 introduced the data source used across this project, describes the design process and lists the user needs and required tasks for the exploration process. Chapter 4 presents the results in form of an implementation of two visualization methods and their integration into the host application–Refinery Platform–for searching,

browsing, and exploration. Furthermore, a real-world case study for evaluation of the novel exploration method is presented. Finally, chapter 5 discusses the results and conclusions, examines current limitations, and gives an outlook for future enhancements.

2 Background

Ontology-guided exploration of biological data repository is a cross-cutting challenge of how to find and visualize relevant data. The next chapters summarize the current state of research regarding search processes, visualization principles, bio-ontologies and related work.

2.1 Search principles

Information and data-driven search has become an omnipresent process in many digital activities. To highlight the great importance of information, George A. Miller coined the term "informavore" (Machlup, 1983) to characterize organisms that consume information in analogy to organisms that consume food to derive energy. The principles of digital search (subsequently just called *search*) include the extensively studied field of *information retrieval*, *information seeking behavior*, *human-computer interactions* as well as *information architecture*. *Information retrieval* is dealing with the technical and algorithmic aspects of search, i.e. finding ways to retrieve only relevant data and to optimally rank the documents in such a way that the highest relevant data are ranked first. The process of *information seeking* or more specific *information search* describes the act of how humans seek for data and interact with a given search system. *Human-computer interactions* study the generally behavior of how people interact with computer systems. In the context of search this means studying how people work with the search interfaces. The final component is *information architecture*, which concerns about ways how information is best structured in order to be found most easily by humans. All four components are highly intertwined and crucial to ensure high usability in terms of findability and discoverability.

2.1.1 Information retrieval

The technical process of retrieving relevant information from a collection of information resources, given a well-defined *information need* is called *information retrieval* (IR). The process is usually assumed to start with the user identifying a lack of information. The notion of *information need* describes the information that is needed to solve a specific problem and is typically being seen from two points of view: the IR system and the user who expressed the need. The first can be described as the *object information need* and characterizes the set of documents that are needed by the IR system to fully answer the given query. The latter is sometimes referred to as *subjective information need* of the user who suffers from the lack of information. It is generally nontrivial to capture *subjective information needs* unless users are asked directly; thus, in the context of IR, *objective information need* is often regarded only. *Information seeking* (covered in Chapter 2.1.2) studies the user's point of view in finding information (Marchionini, 1995, see Chapter 1). *Information need* can be fact-oriented, problem-oriented or a combination of both. For example, "What are the

four nitrogenous bases found in RNA?" is a fact-oriented information need and can easily be answered by "adenine, cytosine, guanine and uracil". On the other hand, questions such as "How do ribosnitches alter gene expression and what is their role in cancer?" normally require the IR system to retrieve more than one document to solve the problem. The technical aspect of IR starts when a query is submitted to an IR system (Figure 2.1). In most cases search queries consist of a set of keywords such as "riboswitches vs ribosnitches" but can generally be of any kind. Other frequently used query data types are images, audio messages of natural language, or even a document of the IR system itself. Usually, before the IR system issues a search, the query is normalized and transformed into a compatible query format (Figure 2.1, 2). Hereby, the query format strongly depends on the IR system. Next, the IR system retrieves all documents that match the given search query. The search can incorporate the document's content, metadata or both (Figure 2.1, 3). Subsequently, the retrieved documents are being sorted by decreasing relevance given an internal model (Figure 2.1, 4).

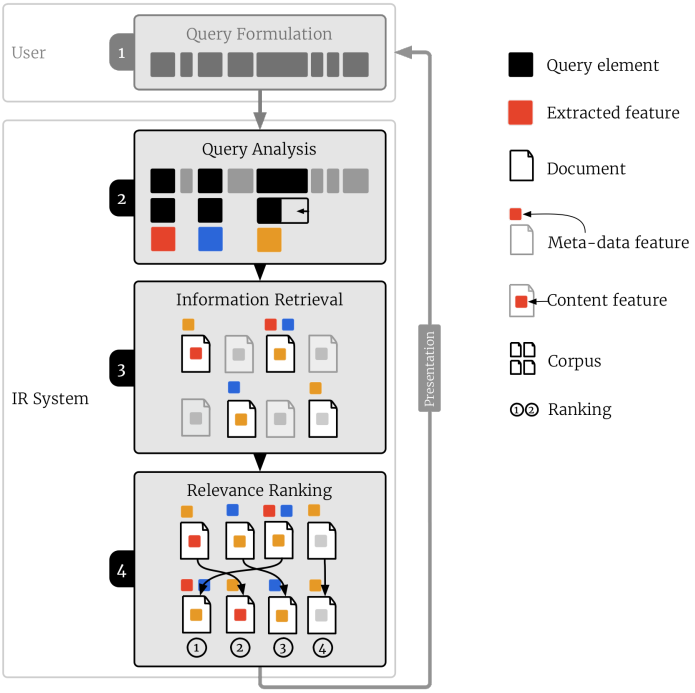

Figure 2.1: Holistic process of information retrieval. (1) The formulated query is depicted as rectangles of different length to symbolize different features. (2) The query is analyzed and transformed into a compatible format. (3) Documents with features similar to the query are extracted from the corpus. Features can match metadata or content traits. (4) The extracted documents are ranked according to their relevance.

The term relevance is used to describe how well a certain document matches the given search query in terms of its usefulness to answer the problem. Similar to

objective and *subjective information need*, relevance can be objective and subjective too. In the follow paragraphs relevance is used as a synonym for objective relevance.

The two most crucial components of any IR system are the ability to efficiently find the documents that match to the given input query and to rank these documents in such a way that it reflects the relevance best. Depending on the input query, the retrieval algorithms can vary dramatically. Since this thesis makes use of a text-based IR system only, other strategies will not be described here. Before anything can be retrieved from an IR system, the data needs to be imported from the original data store. The process consists of three main steps: translating the original data into a model that is suitable for searching, analyzing the content and metadata of each document, and finally indexing the documents. The first step is handled by an external application and depends on the structure of the documents to be indexed and which features are planned to be used for later retrieval. Once the data has been extracted, transformed, and handed over to the IR system, the data goes through a process called *analysis*, which translates text into tokens. A token could be considered the smallest data nugget in an IR system and represents a normalized word or parts of a word. The process of analyzing document content is also called *tokenization* and can be regarded as feature extraction since tokens ultimately represent the document's features. *Tokenization* can incorporate a variety of different steps such as ASCII conversion, lowercasing, split on punctuation and space, stop word removal, removal of possessive form, stemming or lemmatization. Figure 2.2 provides an example analysis. When a search is issued, the query must be analyzed in exactly the same way as the documents had been analyzed to ensure that tokens match. After tokenization, the final step is to index the data using a data structure called *inverted index*. At its core, the inverted index contains two objects: an alphabetically sorted list of all tokens and a dictionary containing the document IDs in which a token appears. The first is called token list and links tokens to their dictionary entry. Retrieving all documents that contain any of a given set of query tokens is then just a matter of finding the terms in the term list, moving to the containment dictionary to extract all documents with listed IDs. A thorough introduction to information retrieval is given by Manning *et al.* (2008) and a practical guide from Turnbull and Berryman (2016) provides insides into relevance engineering.

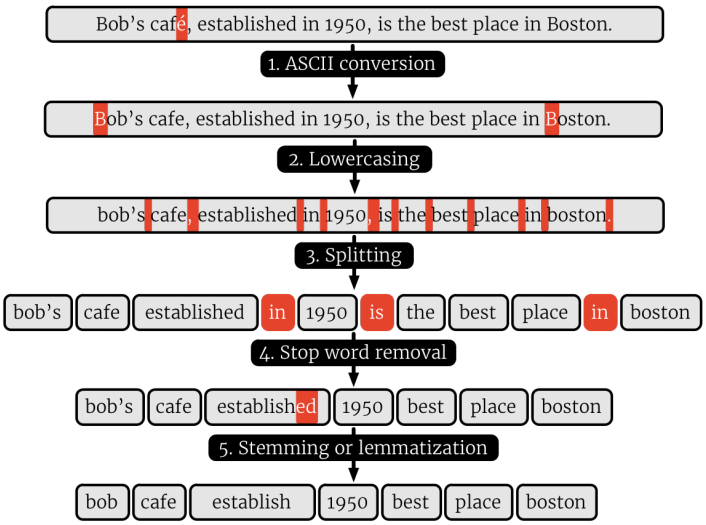

Figure 2.2: Example analysis of an English sentence. In each step the characters to be changed or removed in the next analysis step are marked in red. (1) The string's non-ASCII characters are translated into their ASCII counterparts or removed if no counterpart is available. (2) Uppercase characters are lowercased. (3) The token is split into smaller tokens at every white space and punctuation. (4) Common words that do add little value in retrieving relevant documents are called *stop words*. These *stop words* are typically removed. (5) Words are often transformed into their stem or lemma, in order to efficiently match the great variety of different grammatical forms. E.g. "establishing", "established" and "establishment" will all be lemmatized into "establish".

In order to evaluate the performance of an IR system a number of metrics have been defined. The most common metrics are *precision*, *recall*, *F-score* and *mean average precision* (MAP). *Precision* is the number of relevant documents divided by the total number of retrieved documents (Figure 2.3). On the other hand, recall describes the number of retrieved relevant documents divided by the total number of retrieved documents (Figure 2.3). Given a binary classificatory, *precision* and *recall* are equivalent to *positive predictive value* (i.e. *true positives* divided by *true positives* and *false positives*) and *true positive rate* or *sensitivity* (i.e. *true positives* divided by *true positives* and *false negatives*). *Precision* and *recall* are often negatively correlated, e.g. increasing the number of retrieved documents increases *recall* but most likely decreases *precision* since irrelevant documents will be returned too. *The F-Score* has been introduced to provide a measure for the tradeoff between *precision* and *recall* and is the (evenly) weighted harmonic mean of precision and recall). Because *precision* and *recall* do not take the ranking into account, i.e. it does not matter if relevant or irrelevant data is returned first MAP has been introduced to overcome this limitation. MAP reflects the mean of the totalized averaged precision up to a certain search result x (Figure 2.3).

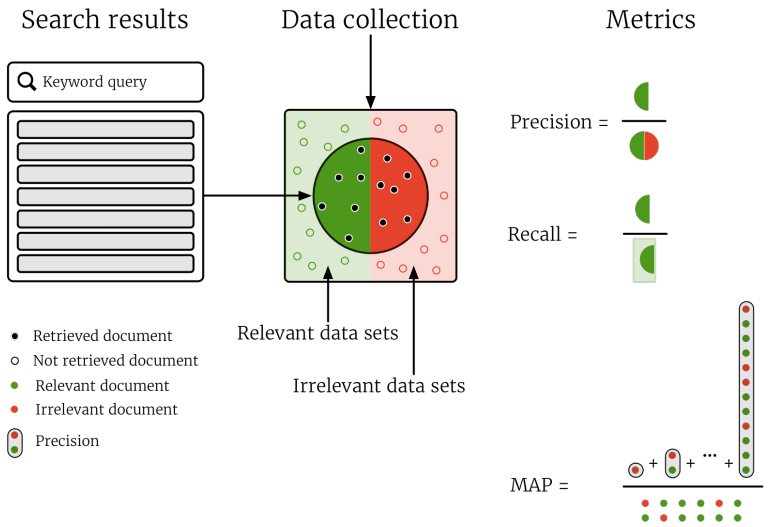

Figure 2.3: Precision, recall and MAP.

2.1.2 Information seeking

The process of how humans seek and search for information has been extensively studied over the past three decades. Many different theoretical models have been developed that attempt to explain the behavior. In contrast to *information retrieval*, which technically ends when results have been retrieved, *information seeking* implies a more open-ended process. The term itself was first coined by Wilson (1981) in order to better study the act of seeking for information. The standard model describes the process an potentially repetitive cycle of identifying the need for information, building a mental model and translating that mental model into words, followed by the specification of keywords for querying, evaluation of the retrieved results, and if necessary reformulation of the query until the desired information is found. This model assumes a clear task and need but the way to a satisfactory conclusion is not known, thus the query might be adjusted iteratively. The standard model is depicted in Figure 2.4 by the red arc indicating query term refinements. A more dynamic model called *berry-picking* describes search as an open-ended process, in which the need for information changes with every iteration. A clear final goal does not exist at the beginning and it changes over time as the user gains knowledge from previous results, hence the *information need* is not static but constantly changes in parallel to the search process. Figure 2.4 indicates the relationship between the knowledge gained through search and the change in information need by means of the yellow arc. Kuhlithau (1991) proposed *information search process*; an *information seeking* model consisting of six steps: *initiation*, *selection*, *exploration*, *formulation*, *collection* and *presentation*. Here, initiation marks the recognition of the information need. First the lack of information is put into context with what is already known and some general broad ideas for a possible direction of search are initiated. The second step (*selection*) describes the selection of a general topic, which will be explored in the third step in

order to gain a better understanding of the new information space. *Formulation* marks the fourth step and is mainly about specifying search and building a more focused search direction. The fifth step (*collection*) involves the collection of the needed information and is described as the most productive step. Finally, information is validated and summarized and characterizes the end of the search process. Throughout the six steps confidence is supposed to be build up along with satisfaction or disappointment depending on the success of search. *Information seeking* through steps is illustrated by the blue arc in Figure 2.4. Some *information seeking* processes can be summarized as strategic approaches (Hearst, 2009, see Chapter 3) to find information. *Information foraging* is a famous theory developed by Pirolli and Card (1995) as an attempt to model how the strategies and technologies for information seeking, searching, and consuming adapt along those processes. The rational is based on ideas of how animals succeed in foraging food. It is assumed that human users strive to consume the information with an optimal cost–benefit ratio. Applied to the processes of searching this theory argues that human users constantly evaluate their tactics on how to find the highest amount of information with the least input of mental energy. *Information foraging* picks up the idea of *informavores*, which characterize species that consume information. The most important aspect of the *information foraging* theory is the concept of *information scent* (Pirolli *et al.*, 2000). Similar to how animals rely on scent to evaluate the odds of finding food or prey in a given region, digital documents and user interfaces are scanned by human users for hints and guides on where to find more information. Research on *information scent* is subsequently strongly connected to *human-computer interaction* and *information architecture*. Related to *information scent* is the theory of the *patch-leaving model*, which studies conditions that make users leave the current page in belief to find information elsewhere more easily. *Information seeking* as a strategy also describes scenarios where the user gives up when finding information becomes too costly. The green arc in Figure 2.4 represents strategic processes. *Information seeking* can be seen as part for a broader process often referred to as *sensemaking* (Russell *et al.*, 1993), which describes the holistic behavior of how users transform information into meaningful knowledge.

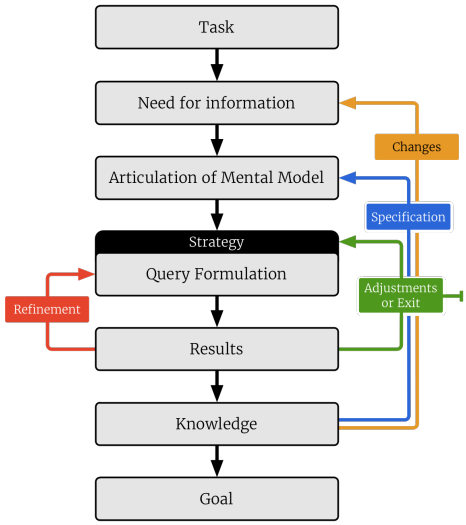

Figure 2.4: Summary of *information seeking* processes: *classic* (red arc), *berry-picking* (yellow arc), *information seeking as stages* (blue arc) and *strategic information seeking* (green arc).

A final note on *information seeking* strategies is the distinction between searching and browsing in which searching is denoted as the act of articulating desired information needs and browsing describes the strategy of recognizing a relevant option among presented navigational choices. When the browse navigation is well suited to a user's need the process of browsing is often cognitive less challenging than formulating one's needs for issuing a search query (Hearst, 2009, see Chapter 3).

2.2 Visualization principles

Visualization describes the process of graphically describing information to communicate a story. There are two major fields of visualization: scientific data visualization and information visualization. Both fields study ways to visually represent biomedical data. Scientific visualization deals with data that has an intrinsic real-world appearance or is at least strongly related to a real-world object. Examples are 3D animations of the blood flow within the human's heart or visualizations of molecules. On the other hand, information visualization is working with abstract data, such as DNA sequence alignments or gene regulatory networks. Since this project is dealing with abstract data only, the following paragraphs will solely focus on information visualization. Statistician Francis Anscombe has greatly illustrated why information visualization matters. He created the famous Anscombe's Quartet (Anscombe, 2012) (Figure 2.5), which illustrates the expressive power of visualization.

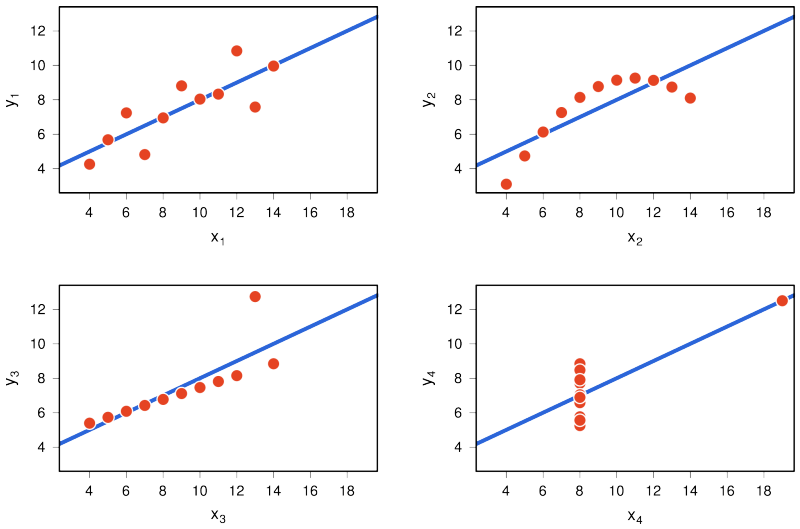

Figure 2.5: Anscombe's Quartet. An artificial data set comprised of four different sets of data points. Each set has almost identical characteristic: mean x value of 9, x variance of 11, mean y value of ~7.5, y variance of ~4.12, an x-y correlation of ~0.82 and a linear regression of $y = 3 + 0{,}5x$. Still, all four data sets are dramatically different.

At its core, each visual idiom (i.e. a chart or diagram) is composed of small marks, the primitive elements, such as points, lines, or complex shapes. Each of these marks has a number of different channels, which can be used for discrimination. Channels include the vertical and horizontal position (artificial as well as spatial), shape, color (hue, luminance and saturation), orientation (also called angle or tilt), size (length, area or volume) and motion. Figure 2.6 provides an overview of the most common marks and channels. Not every mark can be manipulated in all dimensions. For example, rotation of a circle does not create a new distinguishable mark.

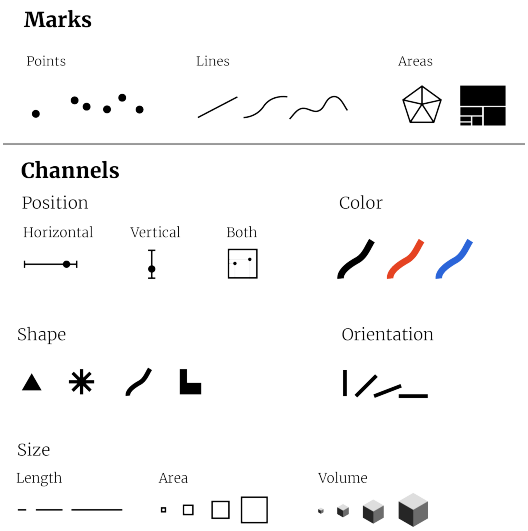

Figure 2.6: Overview of basic visual marks and channels. Adapted from Munzner (2015, see Chapter 5).

The channels can be divided into two groups (Munzner, 2015, see Chapter 5): magnitude channels and identity channels. Magnitude channels represent a data item's property value that follows some order. For example, this could be the number of reads that map to a specific region of the genome. Channels that are suitable for magnitude illustration are position (aligned or unaligned), size (length, size or area), orientation (tilt or angle), color (hue, luminance or saturation) or motion. Identity channels are used to group marks and thus require categorical data. Suitable channels are spatial region, color hue, motion and shape. The two channels color hue and motion can be used to some extend with both ordered and categorical data separately but not at the same time since. While motion is less common, color hue is often being applied to heat map visualization techniques. Munzner (2015, see Chapter 5) summarized the current state of understanding about the effectiveness of different channels (Figure 2.7).

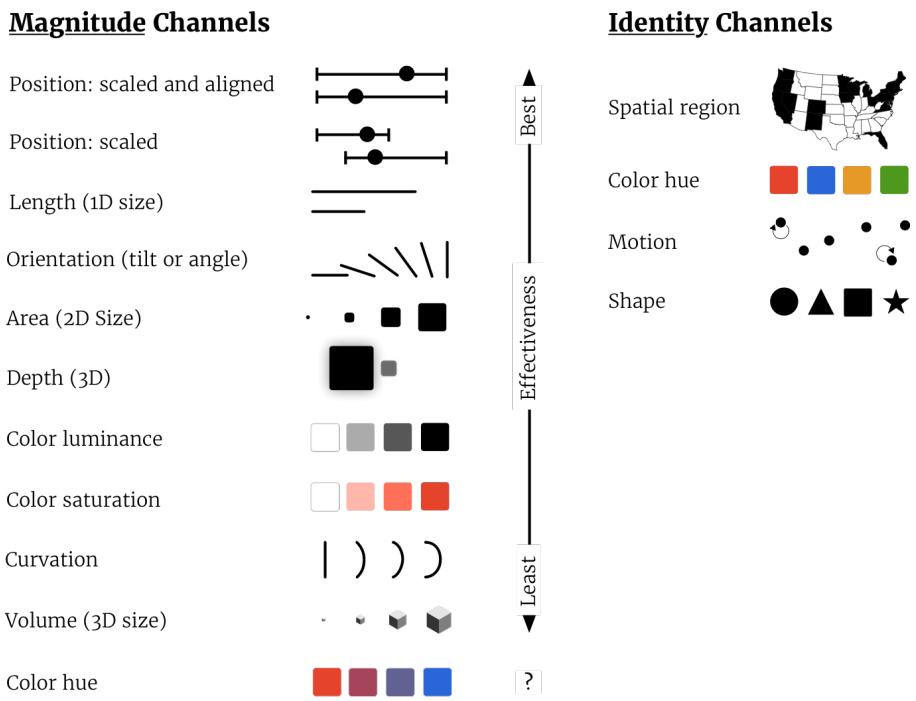

Figure 2.7: Effectiveness of different visual channels. Effectiveness has been measured according to the error rate by means of discrimination and perceptual speed. Color hue has been added as a magnitude channel since its extensive use in heat map visualization methods. While the effectiveness of hue as magnitude channel is not known, relative to this ranking it can be assumed to be low. Adapted from Munzner (2015, see Chapter 5).

Perceptual grouping is a process of mentally associating multiple marks or elements to groups or patterns. Research on perceptual grouping studies how human's visual system determines which elements of an image belong to a higher order concept. This is key to understand human's remarkable pattern recognition. In

real life almost no object is perceived as a continuous element duo to occlusion, different lightning and other criteria. In order to organize the visual clutter and successfully map regions of an image to known concepts, our visual system needs to link different fractions of an image together. In the early 20th century Gestalt psychologists investigated rationales behind perceptual grouping of elements within an image. Max Wertheimer described the first set of laws that drive grouping in 1923 (Wertheimer, 1923) and 1938 (Wertheimer, 1938). Those laws that are now more generally accepted as principles include *proximity*, *similarity*, *prägnanz* (i.e. "pithiness"), *common fate*, *closure*, *continuity*, *symmetry*, and *figure–ground*. Additionally, Stephen Palmer described two more principles in 1990's: *common region* (Palmer, 1992) and *connectedness* (Palmer and Rock, 1994). These modern principles are now commonly associated with the Gestalt principles as well. The following paragraph will describe the rationale behind each of the principles in brevity. A detailed evaluation has been given by Brooks (2014).

The *proximity* principle states (Figure 2.8, 1) that the distance between elements drives mental grouping. Elements that appear closer to each other are perceived as a group of a higher-order concept. The principle of *similarity* states (Figure 2.8, 2) that the higher the visual similarity of elements is the more likely they are regarded to be associated with each other and thus are seen as a group. Hereby, the degree of similarity heavily depends on the number of channels being different among the elements. The principle of *prägnanz* says that geometrically simple elements are favored over complex visual marks. As illustrated in Figure 2.8 (3), most people are likely to see two overlapping rotated rectangles rather than two complex non-overlapping polygons. The principle of *common fate* advocates that objects moving in the same direction will also form a mental group despites other principles such as *proximity* (Figure 2.8, 4). Another principle called *closure* states that closed objects, which are mentally connected to known concepts are favored over unknown open marks. As shown in Figure 2.8 (5) unconnected round lines that are centered around a common point are most unlikely seen as separated objects but united into an outlined circle. The principle of *continuity* is similar to *prägnanz* in that it states that visual marks are perceived as groups when following the form of the simplest line, indicting a possible trend or connection of the elements. The example from Figure 2.8 (6) shows points that form a cross. The principle argues that human's are most likely seeing two trends that can be described as two intersecting straight lines instead of two trends that follow a convex and concave line. The principle of *symmetry* states that multiple symmetric marks positioned around a common center are interpreted as a single object (Figure 2.8, 7). The *figure–ground* principle describes the interplay of foreground and background. Whether a certain region is regarded to foreground or background depends on the composition of many different visual channels. Figure 2.8 (8) illustrates the famous *face–cup* paradox, in which two separate well-known concepts are ambiguously identified, since the foreground–

background relation remains unclear. The principle of *common region* states that marks being enclosed by a line or are positioned in a homogeneously drawn region (e.g. same color or texture) tend to be perceived as a group of common elements (Figure 2.8, 9). Finally, the principle of *connectedness* expresses that elements which are connected by other similar elements–mostly lines–form a complex object or act as a group (Figure 2.8, 10).

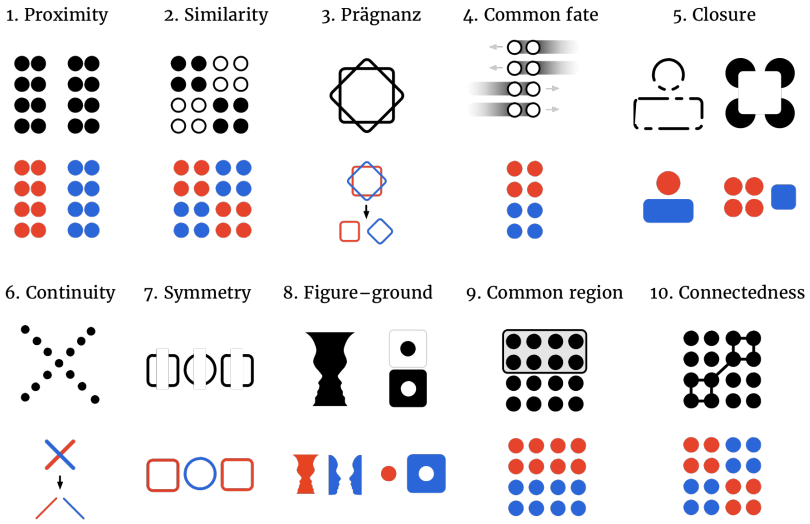

Figure 2.8: The ten Gestalt principle of perceptual grouping. The upper lines show the original visualization pictogram. The colored versions below illustrate the mental grouping by applying different hues (red and blue) to visualization primitives. The grey arrows used in common fate depict the direction of motion.

As an abstraction of data, visualization's ultimate goal is to highlight patterns for human users in a more efficient way than manual inspection of the raw data could. In this respect, good visualizations need to draw quick attention to the important details and help users locating specific elements. Salience is a property of how much an element pops out among a set of other visual marks. The greater salience is the easier it is for users to spot this mark. Salience itself can be expressed through any of the visual properties described above, i.e. mark type, channels or grouping. The effectiveness of salience depends on the overall composition of the visual properties used across the whole visualization. The more exclusive a property is the higher is its salience. Figure 2.9 illustrates some properties and how they convey salience.

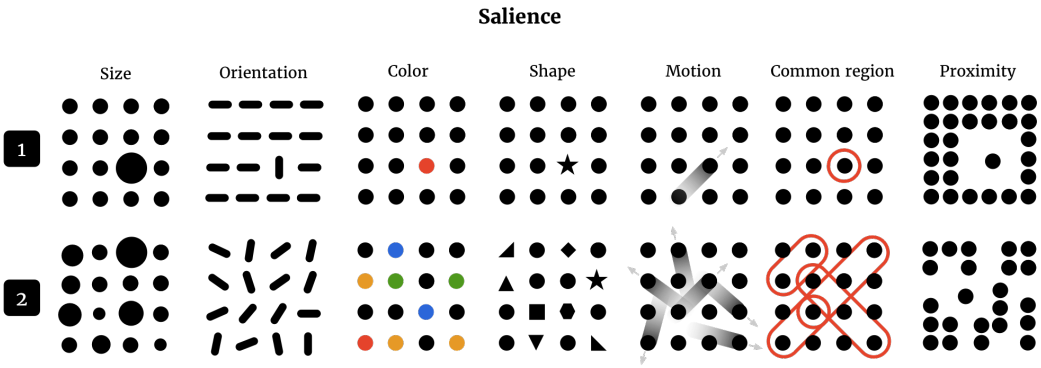

Figure 2.9: Salience of marks. (1) High salience of the element in the third column and third row makes that element pop out from the rest of the visual marks. (2) The same element from (1) has been randomly relocated. Other elements use the same visual property but different characteristics, which makes it harder for the relocated element from (1) to pop out.

2.2.1 Graph visualization

Graph or network-based data is omnipresent and as such does the field of graph visualization span multiple different disciplines. Visualizing graphs and networks involves the visual representation of vertices or nodes, their relation or linkage, and optionally properties of them. Typical tasks involve the understanding of the network or graph topology, clustering of nodes and finding paths. Two idioms are most commonly used: node-link diagrams and adjacency matrices (Figure 2.10).

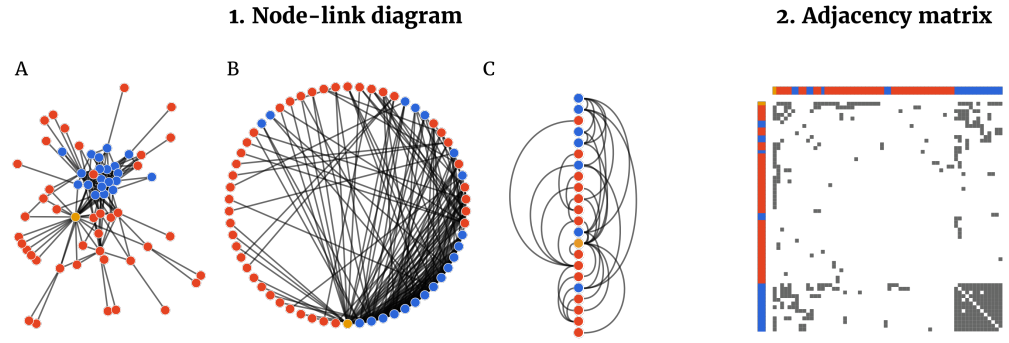

Figure 2.10: Network visualization methods. (1) Node-link diagrams are used most commonly and can be categorized according to their layout algorithm. Popular layouts are: (1a) force-directed, (1b) circular, or (1c) linear. (2) The adjacency matrix is a symmetric matrix M where nodes are displayed as columns and rows. A link between node i and j is depicted as a grey square at position $M_{i,j}$ and $M_{j,i}$. Adapted form Gehlenborg and Wong (2012).

Node-link diagrams are putatively the most popular visualization method for drawing graphs. In their basic form node-link diagrams are comprised of point marks and lines, representing nodes and relations respectively. The position of nodes is generally unconstraint and depends on the layout algorithm (Figure 2.10, 1). Concerns

that play a role in choosing the layout algorithm are the number of edge crossings or the relative distance from a chosen root node. The layout can also be motivated by the data type to be visualized. For example, interactions between mitochondrial DNA might be visualized using a circular layout. Force-directed node-link diagrams are highly effective in representing the topology–due to the combination of proximity and connectedness.

Matrix-based representations of graphs or networks show the relation of nodes via adjacency lists (Figure 2.10, 2). Matrix-based methods tend to be more scalable compared to node-link diagrams but do not convey the topology well.

In the special case that the graph represents a tree, i.e. an undirected and acyclic graph, containment or enclosure idioms, such as tree maps, can be used as well. The web page http://treevis.net (Schulz, 2011) contains a large and up-to-date collection of the various kinds of tree visualizations. A subset of 2D visualization methods for trees is illustrated in Figure 2.11.

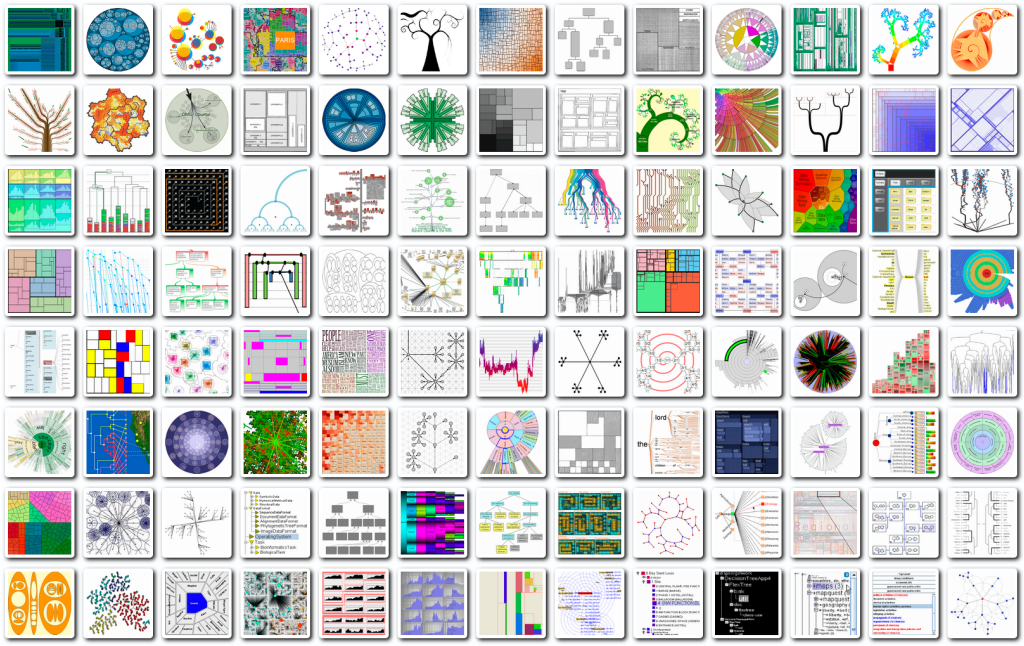

Figure 2.11: Screenshot of current 2D visualization techniques for tree data. Adapted from TreeVis.net (Schulz, 2011).

2.2.2 Set-typed visualization

Often the data to be visualized is categorical or set-typed. One could even argue that any kind of data can be categorized according to some attribute. Attributes that define membership could literally be anything. For example, biomedical data can often be categorized by gender, measuring technology or some expression level interval. The main reason for visualizing set-typed data is to help the user understand the nature of categorization.

There are two different types of questions that are of key importance. First, questions about the member elements such as which element is contained in most sets or which element is always associated with a specific attribute. Second, questions about the sets themselves, such as the set's size or relations between different sets like complement, containment, exclusion and intersection. Most often set-typed data visualization should facilitate answers about the membership across multiple different categories. Visualizing the membership of documents in different categories can help the user to quickly understand the nature of the data collection.

Categorizing data is especially very useful for browsing a large collection of elements as it allows the user to filter down before inspecting the relevance for each element separately. Also, hierarchical sets, i.e. sets that include other sets, can provide a level of abstraction and keep exploration of many elements perceptually feasible. A thorough review of set-typed visualization techniques is provided by Alsallakh *et al.* (2014).

The two most commonly used and oldest visualization techniques for set-type data are Euler and Venn diagrams as shown in Figure 2.12 (1 & 2). Leonard Euler has invented Euler diagrams in the 18th century, while John Venn invented the similar Venn diagram in 1880. Both techniques visualize the membership of a collection of elements through closed circles of any shape drawn on a plane. Elements that belong to multiple sets are visualized by overlapping sets. The main difference between Euler and Venn diagrams is that Venn diagrams require 2^n distinct zones for n sets, even when some zones represent empty set intersections. The greatest advantage of Euler and Venn diagrams is that they are typically understood without any explanations due to the principle of *common region* (Palmer, 1992). Humans tend to perceive special partitioning more easily when depicted by *closure* than *proximity* or *similarity*. The biggest limitation of both techniques is that they do not scale well beyond four categories. Many variations of Euler and Venn diagrams have been proposed but all are based on the same concept of using closed areas to visually indicate set membership. Node-link diagrams are another way to visualize set-typed data (Figure 2.12, 3). The membership can either be shown by drawing links between elements and sets (Figure 2.12, 3a) or by drawing a direct link between members of a set (Figure 2.12, 3b) (Alper *et al.*, 2011). Sometimes links are enlarge in such a way that they additionally include *common region* principle such as in Kelp diagrams (Dinkla *et al.*, 2012) or Bubble sets (Collins *et al.*, 2009). Techniques that solely rely on the *common region* principle are available too (Dinkla *et al.*, 2014) (Oelke *et al.*, 2014). A third approach to illustrate set-typed data is using matrix-based techniques. Matrix views are used in different ways to depict membership (Figure 2.12, 4). One way is to represent elements and sets by rows and columns and highlight a cell $M_{i,j}$ if element i is part of set j (Kim *et al.*, 2007). Another visual metaphor used in matrix-based visualization is plotting sets as rows and columns and creating a heat map, which represents the overlap between sets (Sadana *et al.*, 2014). Some approaches combines

different methods, such as UpSet (Lex *et al.*, 2014), which uses a matrix where each column represents a set and each row depicts set combinations.

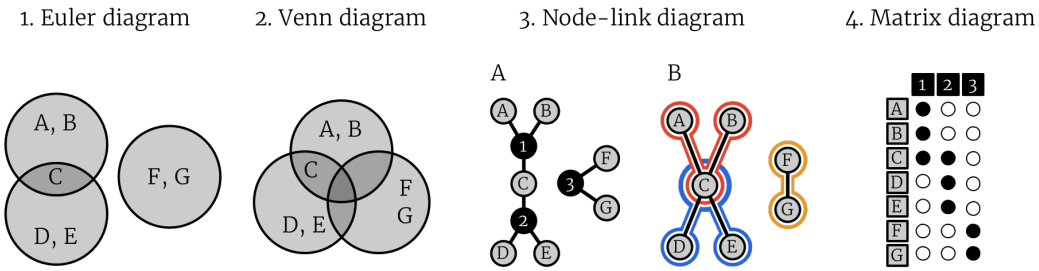

Figure 2.12: Basic principles of set-typed visualization. (1 & 2) represent methods following the *common region* principle. (3) Node-link diagrams facilitate sets via *connectedness*. (4) Matrix diagrams depict sets via columns and membership via rows.

2.3 Ontologies

Originally, ontology is a field of philosophy that is concerned about the how objects of the real world can be formally described, grouped, and related to each other given their characteristics. The study of ontologies is hundreds of years old and is rooted in Greek philosophy. Tom Gruber established ontologies in the field of computer science in 1995 as a model for sharing knowledge (Gruber, 1995). Gruber also coined the famous and shortest known definition of ontology:

> "An ontology is a specification of a conceptualization."
> Gruber (1993)

Thus, ontologies—in the domain of computer science—can be regarded as a set of definition over a set of concepts described with controlled vocabularies. The goal is to facilitate reuse and sharing of knowledge. As such, ontologies are a powerful tool for annotation of documents as well as semantic queries.

2.3.1 OBO and OWL

The two dominant languages for creating biomedical-ontologies are Open Biomedical Ontologies (OBO) and Web Ontology Language (OWL). The OBO ontology language was originally developed by Gene Ontology (GO) consortium and later emerged into the OBO Foundry. OWL has been developed by the World Wide Web Consortium (W3C) to provide a general tool for modeling web-based ontologies with the Semantic Web (Berners-Lee and Hendler, 2001) in mind. Generally, OWL is more powerful and features richer descriptiveness than OBO, since OBO had been designed for bio-ontologies only.

The OBO ontology model consists of classes (also called terms), instances and relationships. Classes model types of objects or real-world concepts and instances of classes represent individual objects of a class. Relationships link classes

and instances with each other. The most important class relationship type is *is_a* between classes, describing the subclass relationship. Thus if B *is_a* A then all instances that are of type B are also of type A. Other important relationships defined by the OBO Relation Ontology are *part_of*, *develops_from*, and *adjacent_to*. In general, relationships can be directional, symmetric, transitive or cyclic. Also, certain relationships can be used to relate other relationships. For example, the relationship *inverse_of* can be used to model *has_part* as the inverse of *part_of*. OBO also defines *domain* and *range* related, indirect subclass relationships. For example, given the fictional relationship *translated_by_RNA_polymerase*, the *domain* (i.e. source class) could be defined as the class *DNA* while the *range* (i.e. target class) would be *RNA*. Given the statement A *translated_by_RNA_polymerase* B, we can be inferred that A *is_a DNA* and B *is_a RNA*. OBO supports a number of different metadata attributes such as an identifier, a name, synonyms, cross-references, definitions, comments or a namespace to be associated with a term, relation or instance. A term's identifier is a strings in the form of IDSpace:ID, where IDSpace is the ontology's acronym. E.g. CL:0000653 stands for term number 0000653 of the Cell Ontology (CL).

OWL builds on top of the Resource Description Framework (RDF), which models the information about resources in the World Wide Web. In RDF information is defined as triples of the form: subject predicate object. Each triple defines a statement, i.e. one single piece of information. RDF statements can also be interpreted as a directed labeled graph with the subject representing the source node, the object being the target node and the predicate indicating the directed edge. (Figure 2.13, 1) The subject and predicate must be resources while the object can either be a resource or a literal value; e.g. string, number or Boolean. Resources are defined objects or concepts of any kind that are uniquely identifiable via a Uniform Resource Identifier (URI). Since RDF has been developed for the World Wide Web most resources correspond to web pages but they can literally be any kind of concept. Multiple RDF statements intrinsically form a directed, labeled, multi-relational graph. By itself RDF doesn't impose any semantics and is merely a data model. RDF schema (RDFS) defines some basic vocabulary to create ontologies. Specifically, RDFS provides means to describe classes, properties (i.e. relationships), enables direct sub- and super classing, and indirectly exposing a class hierarchy via *domain* and *range* rules. By being able to create a class hierarchy, RDFS also provides inference of new statements that are implied by asserted statements. The most prominent inference is engendered by the transitive nature of the class hierarchy (Figure 2.13, 2).

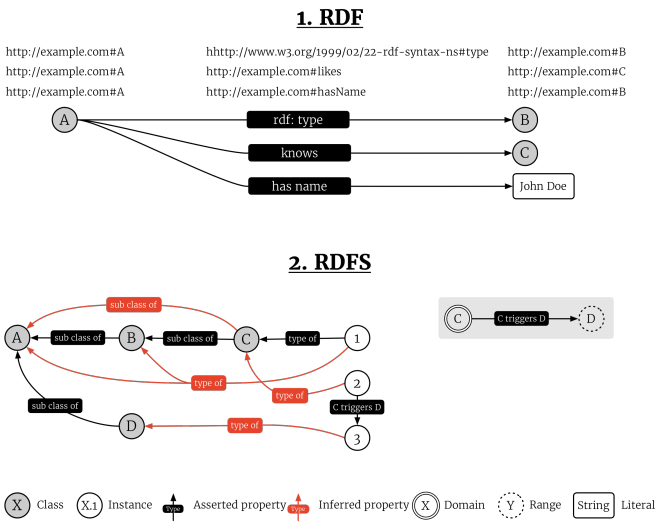

Figure 2.13: Example and differences of RDF and RDFS.

OWL is even more expressive compared to RDFS by extending the vocabulary and inference rules further. In addition to defining classes directly via a URI, it is possible to indirectly define classes via enumeration of all instances, property restrictions, intersections, union of multiple classes or the complement of another class. Especially property restrictions appear often in the biomedical domain. For example, the Cell Ontology (CL) has two essential object properties for expressing its class hierarchy: *subClassOf* and *develops_from*. The *develops_from* property is expressed by a property restriction. The following simplified expression is taken from CL and describes a typical *develops_from* relationship:

```
1    <owl:Class rdf:about="CL:0000005">
2      <rdfs:label>fibroblast neural crest derived</rdfs:label>
3      <rdfs:subClassOf>
4        <owl:Restriction>
5          <owl:onProperty rdf:resource="RO:0002202"/>
6          <owl:someValuesFrom rdf:resource="CL:0000333"/>
7        </owl:Restriction>
8      </rdfs:subClassOf>
9      <rdfs:subClassOf>
10       <owl:Restriction>
11         <owl:onProperty rdf:resource="RO:0002202"/>
12         <owl:someValuesFrom rdf:resource="CL:0000008"/>
13       </owl:Restriction>
14     </rdfs:subClassOf>
15   </owl:Class>
```

Here the *fibroblast neural crest derived cell* (CL:0000005) is described to develop from (RO:0002202) the *migratory neural crest cell* (CL:0000333) in lines 3 to 7

and from the *migratory cranial neural crest cell* (CL:0000008) in lines 9 to 14. An extensive introduction to bio-ontologies is provided by Robinson and Bauer (2011).

Currently the OBO Foundry (Smith *et al.*, 2007) lists 137 ontologies and BioPortal (Whetzel *et al.*, 2011) has 503 ontologies. The following twelve ontologies are used for data set annotation in this project: BRENDA Tissue and Enzyme Source Ontology (BTO), Chemical Entities of Biological Interest (CHEBI), Cell Ontology (CL), Experimental Factor Ontology (EFO), Foundational Model of Anatomy (FMA), Gene Ontology (GO), Mouse Adult Gross Anatomy (MA), NCBI Taxonomy (NCBITAXON), NCI Thesaurus (NCIT), Ontology for Biomedical Investigations (OBI), Phenotypic Quality (PATO), and Units of Measurement (UO).

2.4 Graph databases

Most database management systems (DBMS) are based on a relational modeling where data is grouped by relations and organized by tuples. Colloquially, relations stand for tables, which define a set of attributes (i.e. columns). Every data entry represents a tuple of information. Usually each tuple is uniquely identifiable by a unique attribute set called keys. The greatest deficit when working with relational data, e.g. graphs, is that relational DBMS require joins for each relation to be retrieved, which can quickly become a serious issue when more than a few relations are assessed. Graph databases build upon relational DBMS by explicitly storing object relationships instead of inferring them at query time. This characteristic makes graph database more suitable for relation intensive queries.

Neo4J (http://neo4j.com) is the graph DBMS that is being used throughout this project. It implements a property graph model, which consists of a graph $G = (V, E)$, a set of vertices V and a set of edges E. An edge $e \in E;\ x, y \in V;\ e = (x, y)$ is an ordered pair of vertices representing a directed relationship from x to y. Apart from a normal direct graph, vertices and edges of a property graph model can be associated with multiple key-value pairs. In addition, Neo4J adds labels and typed relationships to allow grouping of nodes. Even though relationships need to have a direction, the directionality does not matter. Neo4J allows traversing paths in any order. Figure 2.14 provides an example property graph and summarizes the ideas described above.

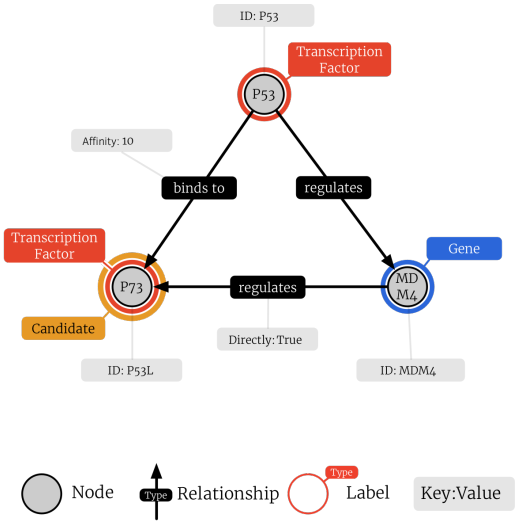

Figure 2.14: Example property graph. Nodes and relationships can be associated with multiple key-value pairs. The key-value pairs follow no explicit schema. Nodes can be grouped into sets by assigning multiple labels.

2.5 Related work

Visualization of data repositories or large document collections have similar requirements and goals compared with efforts of visualizing search results, since search results represent an arbitrary subset of the document corpus. Additionally, search visualization incorporates the notion of relevance. Ontology-guided visualization of biological data repositories intersects with a number of different research areas. In an attempt to provide a comprehensive overview, related work is categorized into two main groups: problem-focused and visualization-focused methods. Problem-focused projects are mostly concerned about applied challenges while visualization-focused work is trying to enhance existing or develop new visualization methods for a broader spectrum of applications.

Over the last two decades various different visualization methods have been developed to support search. The tools can be divided into those that attempt to visualize each result separately and those that try to provide an overview of the complete search results. For example, TileBars (Hearst, 1995) and successors Insyder (Reiterer *et al.*, 2005) and HotMap (Hoeber and Yang, 2006) visualize the approximated location of query term matches within each retrieved document and thus provide a visual notion of relevance. Others illustrate the relative similarity of search results by depicting each retrieved document as a glyph or simple visual mark in a 2D or 3D space. The spatial location is determined via dimensionality reduction. Similar documents should cluster together and form fuzzy groups or categories. Examples for glyph-based visualization techniques that operate search results are InfoSky (Andrews *et al.*, 2004) and xFind's VisIslands (Andrews *et al.*, 2001). InfoSky incorporates hierarchical classification of the documents and displays them in

circular weighted Voronoi tree maps. On the other hand, some visualization methods provide an abstract summary of the set of all retrieved documents. The RelationBrowser++ (Zhang and Marchionini, 2004) visualized the overall and search related abundance of categories using superimposed bar charts. The search engine Grokker[1] hierarchically categorized search results and provided a top-down filter mechanism via a circular tree map of topics and subtopics. A tool called ResultMaps (Clarkson *et al.*, 2009) groups search results according to a hierarchical classification and uses the tree map visualization to convey the hierarchy. Hearst (2009, see Chapter 10) provides a comprehensive overview of the efforts in visualizing search results.

Apart from that, a number of projects studied possibilities to visually summarize the corpus data repositories and enable exploration. The following examples focus on visual exploratory tools that utilize metadata or descriptive vocabulary, i.e. tools that visualize categorized or set-typed data. InfoSky (Andrews *et al.*, 2004) that has been described above can also be used to explore a whole collection of data. Hiérarchie (A. Smith *et al.*, 2014) is a tool for visualizing hierarchical topic models using sunburst charts to explore text documents. In a similar fashion, Phenoblocks (Glueck *et al.*, 2015) uses the SunBurst idiom to present the hierarchical structure of the Human Phenotype Ontology (Robinson *et al.*, 2008). The SunBurst technique is basically a tree map laid out radially. Nils Gehlenborg has studied ontology-guided exploration of ArrayExpress (Kolesnikov *et al.*, 2015) as part of his PhD dissertation. The prototype called ArrayExpress Explorer (Gehlenborg, 2010) uses the tree map idiom to visualize and query ArrayExpress' content by means of the EFO (Malone *et al.*, 2010).

Other work that indirectly relates to this thesis is more focused on visualization techniques for graph, tree, or containment data. The variety of tree visualizations alone is huge. As mentioned in Chapter 2.2.1 Hans-Jörg Schulz maintains an extensive collection of numerous different visualization methods for tree data (Schulz, 2011) (Figure 2.11). Tree maps (Johnson and Shneiderman, 1991) are one of the most space efficient ways to visualize hierarchical data and have been studied extensively. A major disadvantage of tree maps is that they do not communicate the tree's topology as well as node-link diagrams can. Elastic Hierarchies (Zhao *et al.*, 2005) has been developed to combine the strength of node-link diagrams and tree maps. GrouseFlocks (Archambault *et al.*, 2008) is another attempt to combine the node-link idiom with circular tree maps.

---

[1] Grokker had been shut down when Groxis ceased operation in 2009. The Internet Archive provides a copy of Grokker's tour, which contains screenshots and explanation of their visualization tool.
https://web.archive.org/web/20090509164021/http://www.groxis.com/service/grokker/grokker_tour.html

3 Methods

Throughout this project the what-why-how analysis framework developed by Brehmer and Munzner (2013) and later refined by Munzner (2015, see Chapter 2, 3 & 5) is used as a guidance for development. Hereby, what, why, and how correspond to the data abstraction, needs and tasks, and the decision for the visualization design.

3.1 Data

The data that is primarily being handled by the Refinery Platform as a case study comes from the Stem Cell Commons (Ho Sui *et al.*, 2013). The Stem Cell Commons have been initiated by the Harvard Stem Cell Institute to facilitate comparison of stem cell experiments at the molecular and semantic level and provide a community based platform for sharing data. The data had been manually curated and annotated with twelve ontologies (Table 1) and is stored in the investigation-study-assay (ISA) model (Rocca-Serra *et al.*, 2010). Up to date the Stem Cell Commons consist of 201 data sets of which 199 are valid ISA-Tab files.

Ontology	Number of terms	Number of data sets annotated at least once with the ontology (after importing)	Number of times the ontology was used for annotation across all data sets	Number of terms used for annotation
BTO	5809	5 (5)	9217	3
CHEBI	61550	144 (141)	22114	6
CL	4789	101 (81)	6020	59
EFO	17238	45 (198)	785	17
FMA	103902	132 (0)	2351	22
GO	44049	33 (32)	41	2
MA	3229	3 (3)	22	2
NCBITAXON	906907	201 (198)	4035	4
NCIT	116762	59 (58)	749	10
OBI	2932	201 (198)	462	9
PATO	2457	6 (3)	8	1
UO	331	58 (25)	14847	7

Table 1: Ontology usage across Stem Cell Commons. The difference of ontology usage per data set between before and after importing is due overlaps among ontologies, e.g. the EFO covers almost all terms from other ontologies that have been used for annotation, and annotation issues, e.g. sometimes ISA-Tabs ontology sources and identifiers do not match. Note that the FMA annotations could not match any data set because of outdated identifiers. The FMA uses at least three kinds of URI schemas and while mapping is certainly possible, it is out of the scope of this thesis.

3.1.1 Data abstraction

The goal of this project is to enhance exploration of biological data repositories by illustrating semantic relationships across the entire repository and search results. Since it is not the goal to visualize search or ontologies but their relationships the data needed to be abstracted to clarify their relationships. Each ISA file represents an investigation with studies and assays, each related by a one-to-many relationship, e.g. one investigation can have multiple studies and each study can host multiple assays. In terms of exploration a data sets is regarded as an atomic unit, meaning that this project only focuses on inter and not intra data set exploration. Thus, a data set can be seen as a set of raw data files (Figure 3.1, 1), which have been annotated with a set of ontology terms (Figure 3.1, 2). Additionally, studies and assay can have multiple annotations as well. Finally, only those ontology terms that are associated with data sets are of use for exploration. This leads to a complex annotation set hierarchy illustrated in Figure 3.1 (3). Terms that are not used for annotation and which are not compliant with equation 3.1 are removed.

Therefore, the abstracted data to be visualized can be described as an ontology-guided containment multi-hierarchy. The transitive nature of subclass relationships defines the subsumptive containment hierarchy of ontology terms. E.g. the term *podocyte* is a subclass of *epithelial cell*, which in turn is a subclass of *cell*. Hence, *podocyte* is also a subclass of cell. In terms of data set exploration this means that any data set that has been annotated with *podocyte* should also be listed when the higher-level term *cell* is browsed.

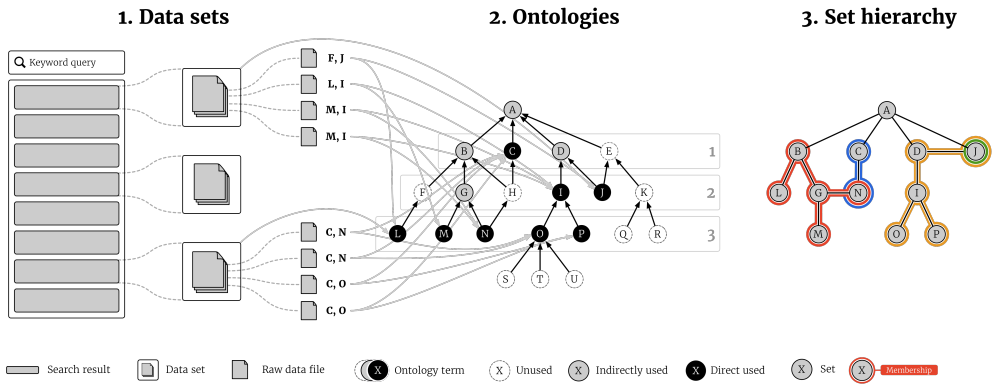

Figure 3.1: Abstract data model. (1) List of data sets. Each data set can consist of multiple raw data files. Raw data files, assays or studies can be annotated with ontology terms. (2) The original topology of the ontology used for annotation. (3) Compressed annotation set hierarchy. The four main branches are indicated by colored contour lines. The set hierarchy is not equivalent to a tree as some sets can be subsets of multiple other sets; e.g. N is a strict subset of G and C.

3.1.2 Data Processing

Ontologies can be represented as directed, and in most cases acyclic, graphs. The most important property in terms of exploration is the number of data sets that are associated with an ontology term. Given a graph $G = (V, E)$ with V representing the set of vertices and E representing the set of edges linking two vertices, we denote the number of times a term t has been used to annotate a data set as the size of the term related node v_t. Since only real-world concepts should be illustrated, satisfiable classes are considered only, i.e. nodes that are reachable from the global root node *OWL:Thing*. By having a unique root node, we expose an indirect order on the node set. The length of the shortest path of a node x to the root is defined as the distance of x.

The ontologies used for annotation of data sets each describe a large domain but the actual terms that have been used for annotation are relatively few compared to the overall number of terms. For example, the Stem Cell Commons uses 142 out of 1269955 terms only. Since the goal of an exploration tool is to provide means for finding data sets and understanding the composition of data collections, ontology terms that have not been used for annotations should not be shown unless they account for a stem to nodes of size greater than zero. Thus, to provide efficiently visualization methods the ontology graph needs to be pruned. A set is only of use if its size is different from its superset's size. Mathematically this means that we need to ensure a strict containment set hierarchy. Given three terms A, B and C where A is a subclass of B and B is a subclass of C. The terms' set representation S_A, S_B and S_C must be compliant with:

$$S_A \subset S_B \subset S_C \tag{3.1}$$

In addition, the tree map visualization method conveys the hierarchical order by containment; data to be visualized needs to come in form of a tree rather than a graph. Although the list graph methodology use the visual metaphor of node-link diagrams, the hierarchy is illustrated by placing nodes from left (the root) to the right (leaves). Depending on the complexity of the graph, it is possible that a node could be placed in multiple different columns, as there might be different path to the very root. To avoid visual clutter links only go in one direction: from superclass (left) to subclass (right). Thus, nodes with multiple parents whose distance are not equal are duplicated. (Figure 3.2)

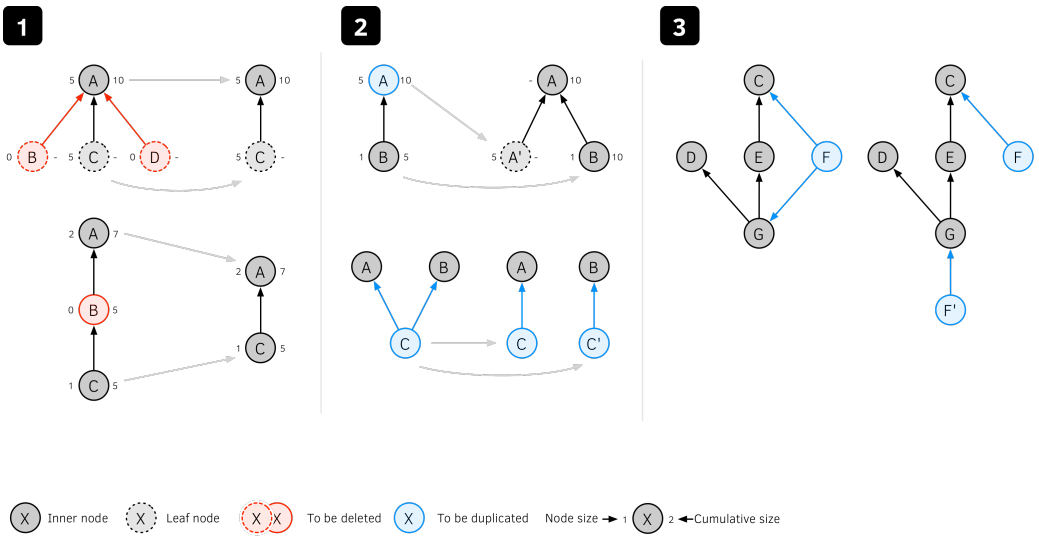

Figure 3.2: Graph manipulations. (1) Leafs and inner nodes of size zero are deleted. The cumulative size includes the sum of the size of all child nodes. (2) Node duplication for the tree map visualization. Inner nodes with a size greater zero need to be duplicated as child nodes to themselves. Also, nodes with multiple parents are duplicated for each parent to provide a unique path to the root. (3) The list graph visualization only requires nodes to be duplicated when their parents' distances to the root are not the same. Therefore, node *F* is duplicated because the distance of node *C* and *G* is not equal. On the other hand, node *G* is not duplicated because the distance of nodes *D* and *E* is the same.

3.2 Design

3.2.1 Requirements & Tasks

The first step in designing novel visualization methodologies is to compile a set of requirements that formalize the needs and tasks to be addressed. Requirements are often predefined to some degree by previous work and later refined by expert scientist working in the field of research associated with the problem. Therefore, prior to the requirements analysis the main target users need to be determined. In regards to exploration of biological data repositories, three primary and two secondary audiences have been identified: data scientists and analyst, project leaders, group leaders and funders, data curators, and developers. The five groups act as blue print characteristics for user roles with different needs regarding an exploration system. The roles are not meant to exclusively describe a user but rather the different roles a user can reflect. For example, a person can be a data scientist and project leader at the same time, having to fulfill two roles throughout the daily work.

The primary concern of data scientists and analysts is how data can be turned into information and subsequently transformed into knowledge. Thus, exploiting raw data to gain insides in the underlying biology mechanisms is of key importance. The main focus of data scientists and analysts is to find the most

relevant data as quickly as possible. Precise description and meta-information of data sets is crucial in order to evaluate the relevance. The characteristics that define relevance can vary greatly depending on the project. The main goal for finding relevant data is to compile a data collection that can be analyzed to potentially answer specific biological questions. Data might also be needed to expand in-house generated data. Generally there are two categories of data to be collected: data that is highly similar or data that differs significantly in some respect to existing data. For example, more data might be needed to increase the overall quality and validity of a finding. Data that differs could foster confidence about the robustness of a novel method or it might act as a negative control. The data scientist's exploration behavior is illustrated in Figure 3.3 (1). To summarize, the needs of data scientists and analysts are:

N1 Find data sets that match experimental characteristics.
N2 Find data sets that are similar or dissimilar to given data sets.

The project leader stands for the role model of somebody who leads a research project. The focus of a project leader is to find suitable characteristics of data sets that are crucial in successfully addressing the scientific challenges of a project. The difference between the role of project leaders and data scientists is that project leaders are more concerned about the general state of research and availability of data that is closely related to the project instead of directly analyzing raw data. Therefore, a project leader needs an overview of a collection of data sets that matches experimental conditions (Figure 3.3, 2):

N3 Get an overview of groups of data sets that match specific experimental conditions.

The role of the group leader or funder represents a person who manages and leads a group of people or guides research fields. A group leader needs to have a vision how the past, current, and future research work ties together and what the overall goals are. Hence, project leaders constantly assess the current state of research, find trends, develop novel hypothesis, and strategies how to further contribute to the research field of interest. Even though funders are not directly involved in research project, they need to evaluate the current state of research and discover trends in order to make confident decision about how research fields can be shaped in a way that the funding organization's goals and aspirations are best met. In conclusion both roles need to gain insights into the broader overview of a data collection to discover broader patterns. For an exploratory application, the most important requirement is to provide an overview over the whole or large parts of the repository (Figure 3.3, 3).

N4 Get an overview of the distribution the overall data collection or large subsets of it.

Data curators are responsible for the quality of metadata and integration of data sets across the repository, such as ontology annotations. Their needs are resolving around the current state of curation and how it's quality can be improved to increase the benefits of data. Data curators are not concerned in finding data sets or discovering trends in data generation but instead are interested in the overall distribution and usage of annotation terms. For example, while an overview of the term distribution describes the nature of the data collection it also gives insights in how well certain areas have been annotated. Reason for biased annotation diversity can be due to different annotation strategies over time or differently structured ontologies. It can also point out parts of ontologies that need further specifications. Observing those trends can facilitate better data curation (Figure 3.3 4).

N5 Get an overview of the annotation set hierarchy.

Finally, the developers of a biological data repository can benefit from an exploratory tool in order to evaluate the current usage of the system. This is needed to foresee future trends and plan appropriate changes ahead of time. Similar to group leaders and funder, their goal is not to find specific data sets or small collections of data sets but to understand the repository as a whole (Figure 3.3, 5). The need of developers in regards to an exploration system is very similar to *N4*; the major differences between developers, group leaders, or funders are the resulting actions only.

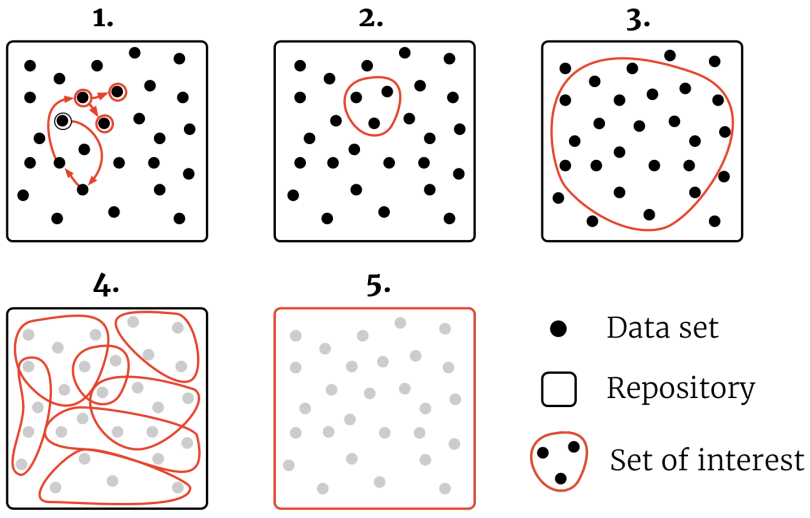

Figure 3.3: Expected user exploration behavior. (1) Data scientists and analysts aim at locating specific data sets. (2) Project leaders focus on a small specific group of data sets that potentially help to answer their project's biological questions. (3) Group leaders and funder

examine the bigger picture of the overall repository. Compared to project leaders they are less concerned about very specific details. (4) Data curators are primarily interested in annotations instead of data sets. (5) Developers care about the system as a whole to foresee trends in data generation.

The specific requirements are derived from the set of needs of the different roles from above. All needs require some degree of understanding of subsets of the repository. Subsets can either be retrieved through search or though term-based browsing and querying. Understanding the composition of characteristics, i.e. annotations, is crucial for understanding if and what to explore next. Hence, the requirements are separated in those that describe the needs for understanding set composition and those that are needed to successfully explore the data repository. The following tasks are related to understanding and sensemaking of data collections:

T1 Determine annotation terms of a data set.
T2 Determine abundance of annotation terms of a group of data sets.
T3 Determine abundance of sets of annotation terms among a group of data sets.
T4 Determine annotation term containment relationships.
T5 Preview data set's content.

The notion of relevant data sets (N1), i.e. data sets that significantly match a desired experimental setup, can be achieved through the illustration of annotations (T1) and by previewing a data set (T5). Showing the relationship between the ontology terms (T4) that have been used for annotation can facilitate finding of related data sets (N2). The abundance of single ontology terms (T2) and sets of ontology terms (T3) aids understanding of search results and highlights patterns (N3, N4, N5). Previewing certain details of a data set (T5) can further increase or decrease relevance and help to find the desired data (N1, N2).

The following tasks are related to the process of actively exploring data collections:

T6 Search for data sets.
T7 Query data repository by annotation term.
T8 Filter down a group of data sets according some annotation.
T9 Loosen annotation constraints.

Being able to search (T6) by keywords and query (T7) by ontology terms for data sets is crucial in finding specific data sets as well as groups of data sets (N1, N2, N3, N4). Drilling down search results by filtering according to some annotation terms (T8) and drilling up by loosening constraints (T9) supports exploration through an enriched search. Ranking annotations according to their abundance and size enhances both: understanding of the nature of data sets (N3, N4, N5) by highlighting most

abundant or most scarce terms, and facilitates exploration by providing a notion of *information scent* (Pirolli *et al.*, 2000):

T10 Rank annotations.

3.2.2 Interviews

Over the course of this project a small series of semi-structured interviews with data scientists have been performed in order to better understand how users actually search for data and to guide the design process. There are several ways to assess which information is most important for the personal relevance of data. The two main categories to distinguish are qualitative research and quantitative research. Qualitative research in terms of the design process can help to understand the problem and facilitate hypothesis generation. On the other hand, quantitative research is often used to verify or compare hypothesis. For this project semi-structured interviews have been chosen because the goal is to gain insights in search behavior to assist further design processes rather than validating a final approach. Semi-structured interviews offer a balance between open-ended and structured interviews. The interviews have been guided by the following questionnaire:

Q1 Do you usually start searching at (a) a data repository such as GEO or ArrayExpress, (b) a scientific journal database such as PubMed, or (c) both depending on the context?

Q1.1 In case of Q1.b: Do you usually read the whole paper or is the abstract (plus figures) sufficient for searching?

Q2 Briefly describe how you find relevant data through the steps described in Q1.

Q3 Rate the following characteristics of a data set or study by (a) importance and (b) how frequently it is important in regards to the relevance for search from 1 to 5 (1 = irrelevant | never; 2 = rarely important | rarely; 3 = helpful | sometimes; 4 = important | often; 5 = essential | always):

1. Technology
2. Sample size
3. Replicates
4. Data quality
5. Species
6. Organ or tissue
7. Cell type
8. Disease
9. Cell line
10. Marker genes
11. Protocol

12. Original problem or goal
13. Publication date
14. Generating laboratory or group
15. Prominence or awareness
16. Technical popularity

Q4 Are there other characteristics that are important to the relevance of a data set during search?

Q5 What are your primary purposes when searching for data?

Q6 Do you usually start your search (a) with a precise search query and loosen constraints, e.g. by reformulation when you cannot find anything relevant, or (b) with a broader search query and then specify the query?

Q7 Would it be useful to have an indicator of the number of data sets related to a certain category, which has only been found partially? E.g. a search for "liver" might not return all data sets related to "hepatic lobule" or "hepatocyte".

The first question (Q1) helps to shed light on current usage of web-based data repositories and whether users have a general preference in searching data. Question 1.1 is an extension to the first question and is used to provide an estimate for how important a published manuscript is in regards to the data. The second question (Q2) is an informal follow-up to the previous two questions and is used to get an idea about the general search behavior of data scientist and analysts and is supposed to give insights in the different steps taken during search. Even though the search process is not assumed to vary significantly from what is already known, having an overview of the users' practices and preferences can be useful to ensure successful adoption of a new exploration system. The next question (Q3) is most structured and part of the questionnaire in order to assess importance of different experimental factors and meta-characteristics of a data set for relevance. Items Q3.1–Q3.4 are related to the technical aspect of a data set. The next six items (Q3.5–Q3.10) describe common characteristics of the biological sample. The last five aspects (Q3.11–Q3.14) reflect meta-characteristics of the overall study and laboratory that conducted the project. Question Q3.15 is about the overall prominence of a data set, the researchers behind the project or any other characteristic. In Q3.16 popularity is regarded as the number of views, downloads or analysis that have been run on a data set.

In total nine data scientists have been interviewed. Due to the small number of participants, the results presented here should be regarded as an indicator only. Regarding the starting point for search (Q1), every interviewee indicated to start searching for biological data by means of literature search at common literature databases such as PubMed. Four out of nine researchers stated that this is their only way of searching, while the other five data scientists said that they sometimes start a search directly at a data repository when looking for specific data types; e.g. large

consortium data like The Cancer Genome Atlas (Koboldt *et al.*, 2012) or the Genotype-Tissue Expression (Lonsdale *et al.*, 2013). When starting the search at a literature database, six out of the nine people indicated that they will focus on the authors, abstract and figures first and read the whole paper only if necessary to understand the data (Q1.1). They might also choose to read the whole paper regardless of the search process when it is of general interest. The other three scientists mentioned that they read the whole paper before downloading any data. The next question provided insights into putatively common search patterns. Three interviewees stated that when they are looking for large consortium data they normally go directly to the consortium's website directly since they know the data already or the general publication does not provide much information. On the other hand, when looking for data that has been generated by individual laboratories they tend to find the publication first in order to understand the data and find its location. One researcher mentioned that a publication also serves as a basic quality control, i.e. data without a publication appears to be less trustworthy. Question 3 highlights two aspects of the important characteristics of a data set during search: first, the importance heavily depends on the project and second, some characteristics are generally more relevant than others. According to Q3, key criteria for relevance of data sets are: *technology*, *data quality*, *species* and *tissue*. Furthermore, *cell type*, *disease state*, *cell line* and *prominence* have been highly rated. Characteristics are regarded to be important when at least more than half of the people consider it to be essential and more than two third of the interviewees consider this criterion to be of importance frequently. The results are also interesting as they showed differences between researchers trained or working in wet laboratories and dry laboratories. Harald Stachelscheid—an induced pluripotent stem (iPS) cell core facility group leader—mentioned the importance of marker gene expressions and differentiation or reprogramming protocols in iPS cell research but both characteristics have not been regarded as highly import by the nine computational data scientists. Table 2 provides an overview of the mean ratings, box percentages, standard deviation, Z-score and percentile of each sub question. Other characteristics (Q4) mentioned by interviewees are the connection between the user who is searching and the laboratory that publish or generated the data, the absolute proximity of the data to the user's laboratory and journal information in which the data set has been published. The information regarding connections between the user and authors is closely related to Q3.14. According to the data scientist, proximity is important for wet laboratory-related research as it makes potential cooperation easier. The journal in which data sets have been published acts as a quality indicator (Q3.4). Question five (Q5) was included to better understand what kind of data sets scientists are targeting in search. Apart from the fact that data scientists who do not have in-house generated data need to collect data prior to analyses, all researchers mentioned that they search for data that is similar in regards to most characteristics but differs in respect to few attributes in order to compare results. Four data

scientists mentioned that they sometimes search for data that is highly similar in all ways in order compare the quality among data sets. Question 6 indicated that both exploration principles–*drill-down* and *drill-up*–are important but a majority of interviewees prefers to start with a specific search query and only loosens constraints when no relevant data has been found. One data scientist uses the same tactic when looking for one specific data set but starts with a generic query and drills down when composing a collection of multiple data sets. Two interviewees mentioned that they also start with a generic search by means of finding a review paper if the targeted area of research is unfamiliar. One researcher stated that her search tactic heavily depends on the previous experience. The researcher either chooses to start specific and drill up or to start generic and drill down depending a personal estimation of the search efficiency. In regards to the last question (Q7), five scientists stated that it might help to have an estimate of undiscovered data that is still closely related but broader. Two people answered that they are not sure how much this kind of metric would help during search but are curious to test it. The other two researchers have been unable to answer this question.

	Q3.1		Q3.2		Q3.3		Q3.4		Q3.5		Q3.6		Q3.7		Q3.8	
	IMP	FREQ	IMP	FREQ	IMP	FREQ	IMP	FREQ	IMP	FREQ	IMP	FREQ	IMP	FREQ	IMP	FREQ
Mean	4.4	4.4	3.3	3.3	3.3	3.8	4,7	4.9	4,4	4.6	4,0	4.8	3,8	4.3	4,1	4.1
5	56%	56%	11%	22%	22%	33%	89%	89%	56%	67%	63%	75%	33%	33%	44%	56%
4	33%	33%	33%	11%	22%	33%	0%	11%	33%	22%	13%	25%	33%	67%	33%	11%
3	11%	11%	33%	44%	22%	11%	0%	0%	11%	11%	0%	0%	22%	0%	11%	22%
2	0%	0%	22%	22%	33%	22%	11%	0%	0%	0%	13%	0%	0%	0%	11%	11%
1	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	13%	0%	11%	0%	0%	0%
5 & 4	89%	89%	44%	33%	44%	67%	89%	100%	89%	89%	75%	100%	67%	100%	78%	67%
2 & 3	11%	11%	56%	67%	56%	33%	11%	0%	11%	11%	13%	0%	22%	0%	22%	33%
SD	0.73	0.73	1.00	1.12	1.22	1.20	1.00	0.33	0.73	0.73	1.60	0.46	1.30	0.50	1.05	1.17
Z-score	1.37	1.37	-0.11	-0.10	-0.09	0.27	1.22	4.32	1.37	1.53	0.34	2.81	0.25	1.77	0.63	0.57
Percentile	92%	92%	45%	46%	46%	61%	89%	100%	92%	94%	63%	100%	60%	96%	74%	72%
	Q3.9		Q3.10		Q3.11		Q3.12		Q3.13		Q3.14		Q3.15		Q3.16	
	IMP	FREQ	IMP	FREQ	IMP	FREQ	IMP	FREQ	IMP	FREQ	IMP	FREQ	IMP	FREQ	IMP	FREQ
Mean	3.3	4.1	1.9	4.2	3.0	4.2	3.0	3.8	3.0	3.4	3.0	4.1	3.2	4.3	2.7	4.4
5	11%	22%	0%	56%	0%	44%	11%	44%	22%	14%	0%	33%	0%	44%	0%	56%
4	44%	67%	11%	22%	44%	33%	11%	11%	22%	29%	33%	44%	56%	44%	22%	33%
3	22%	11%	11%	11%	22%	22%	44%	22%	11%	43%	33%	22%	11%	11%	33%	11%
2	11%	0%	33%	11%	22%	0%	33%	22%	22%	14%	33%	0%	33%	0%	33%	0%
1	11%	0%	44%	0%	11%	0%	0%	0%	22%	0%	0%	0%	0%	0%	11%	0%
5 & 4	56%	89%	11%	78%	44%	78%	22%	56%	44%	43%	33%	78%	56%	89%	22%	89%
2 & 3	33%	11%	44%	22%	44%	22%	78%	44%	33%	57%	67%	22%	44%	11%	67%	11%
SD	1.22	0.60	1.05	1.09	1.12	0.83	1.00	1.30	1.58	0.98	0.87	0.78	0.97	0.71	1.00	0.73
Z-score	-0.09	1.10	-1.48	0.71	-0.40	0.93	-0.45	0.25	-0.28	-0.02	-0.52	0.85	-0.23	1.25	-0.78	1.37
Percentile	46%	87%	7%	76%	34%	82%	33%	60%	39%	49%	30%	80%	41%	89%	22%	92%

Table 2: Evaluation of data set characteristics. IMP stands for importance and FREQ stands for frequency. Marks 4 and 5 have been combined as an indicator for general importance and marks 2 and 3 have been combined to represent minor importance. The Z-score is based on the overall mean of importance and frequency ratings.

3.2.3 Tree map

Tree maps (Johnson and Shneiderman, 1991) are a 2D space-filling visualization technique for hierarchical data that provides an concise overview of set sizes (N4). The design decisions described in this chapter build upon ideas presented in Nils Gehlenborgs PhD thesis (Gehlenborg, 2010). The tree map visualization technique is only able to handle mono hierarchies or trees, i.e. each nodes of the hierarchy can only have exactly one parent. Each node n of the hierarchical data is visually represented by a rectangular r_n. Each rectangular r_n is associated to a size $size_x(r_n)$ of a node property x; e.g. for set-typed hierarchical data this can be the set size (T2). But any other value can be chosen as well. For inner nodes r_{n^i} the rectangle's area is equivalent to the subsumed size of the node's children's sizes, i.e. $size(r_{n^i}) = \sum_k size(r_{k^c})$ where r_{k^c} represents the children. In order to successfully visualize a directed graph as a tree map, the graph needs to be transformed as described in Figure 3.2 (2). An example transformation and visualization is given in Figure 3.4.

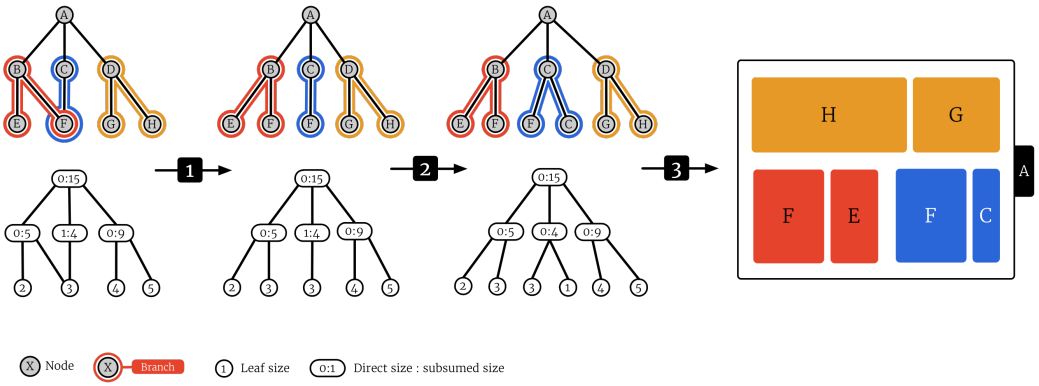

Figure 3.4: Graph to tree and tree to tree map conversions. For simplicity edge directions have been omitted. All edges are directed top-down and represent superclass to subclass relationships. (1) Node duplication per superclass, e.g. *F* is duplicated. (2) The tree map visualization technique represents the parent as a common area of the children and thus cannot be larger than the sum of the area of its children. Therefore, any node with a direct size, such as node C, needs to be cloned as a child of itself in order to visually accommodate its true size. (3) Example drawing of the ordered tree map of the final tree.

In regards to ontology-guided exploration the graph structure to be visualized is the pruned containment set hierarchy derived from the original ontology (Chapter 3.1.3). The number of data sets that have been annotated with a term is used as the property to be visualized by size. Originally the tree map visualization focused on the presentation of the whole tree, effectively only showing leafs. Since even a single ontology can have tens of thousands of terms it is not useful to show all leafs

at once. Therefore, recursive zooming is added. Hereby, the tree map layout is only constructed up to a certain depth and all nodes at that depth are considered to be a leaf. There are two ways to zoom into a tree map: branch-based zooming and level-based zooming. In branch-based zooming the user selects one of the rectangles as the new root node. The zoomed tree map then lays out in respect to the newly selected root node (Figure 3.5, 2). In level-based zooming the visible depth is increased; hence, showing farther nodes (Figure 3.5, 3). Providing means of zooming is important to address the user's needs to gain an overview at different levels of detail (N3 & N4).

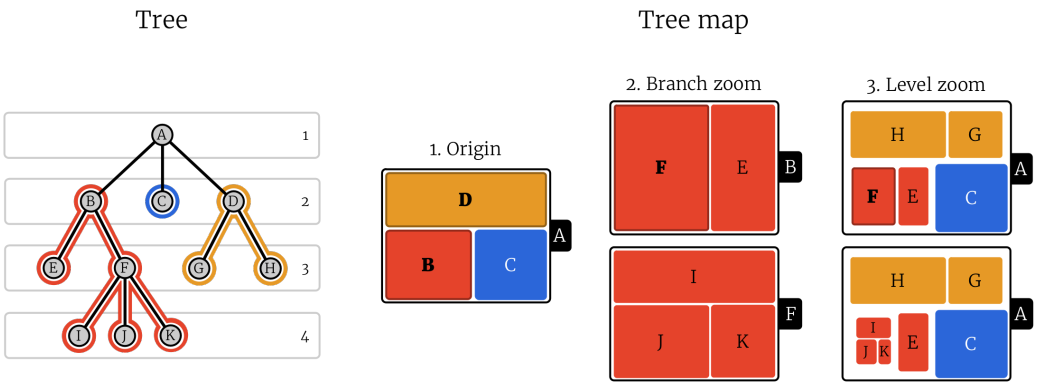

Figure 3.5: Tree map zoom. (1) Tree map showing the first level of child nodes of root *A*. Nodes with a light dark border and bold font indicate inner nodes. (2) Branch *B* has been zoomed in and now reflects a new tree map with the root being *B* and subsequently being *F*. (3) The visible levels of the original tree map has been increase from one to two and two to three, showing all inner nodes at this level and all leaf nodes up to this level.

In addition to the node's size, a second numerical property can be illustrated simultaneously by manipulation of another channel. Since tree maps already use size (i.e. area), position, and shape, there are only a few channels left such as color and motion. Motion is an highly effective channel to make an element pop out from the its surrounding when used sparsely but it quickly looses it's effectiveness when used often. It is also hard to recognize subtle differences in movement to judge the numerical value behind it. Therefore, color luminance, saturation and hue are better suited for visualizing a numerical value. Hue can be effective for continuous regions but is less well suited for comparisons across different spatial regions–as it is the case for tree maps. Also, hue is more effective to illustrate categorical data. Gehlenborg (2010) successfully used a branch color scheme in order to simultaneously visualize categories. Figures 3.4 and 3.5 show branch coloring by alternating the color's hue. This can be an effective tool in distinguishing different branches. A problem arises when the number of possible starting branches is too high or too low or when the starting branches are not well defined. Since this project does not aim at a very specific ontology but uses all or a subset of various ontologies for

visualization, it cannot be guaranteed that the initial branches truly mark different categories. Also, the final tree structure heavily depends on the annotations; some of the branches at the first level might not correspond the original categories due to pruning. Furthermore, introducing several hues comes at the expense of decreasing the salience effect of color for highlighting annotations when the user interacts with either the visualization or the list of data sets. Finally, luminance and hue are no independent channels and their perceived value heavily depends on the context in which they appear. To conclude, a grey scale is suited best for luminance comparison between different spatial regions and the color channel's salience remains high for highlighting nodes that the user interacts with. In respect to exploration the longest path from a node to its descendant leaf nodes is visualized using a restricted grey scale to indicate the size of the hidden explorable space (Figure 3.6). The grey scale ranges from a light grey that is used for leaf nodes to a dark grey used for nodes that are far away from leafs. Both extremes–black and white–are not used. White is already used as a background color and black is spared for the sake of highlighting nodes in interactions. In addition to the luminance, inner nodes are visually separated from leafs by adding a subtle border. The difference between the grey of a leaf nodes and nodes with one child node can be hard to perceive. A border helps to distinguish between nodes than provide further exploration options from leafs that mark the end of the exploration process in terms of annotation specificity. The coloring schema for text incorporates the Web Content Accessibility Guidelines (WCAG) 2.0[2] contrast ratio between a rectangle's background color and black or white to ensure the highest legibility of labels possible (Figure 3.4, 3.5, and 3.6). The guideline is a W3C recommendation and provides a formula to approximate the perceived contrast between two colors, which is not equal to their technical contrast level. Additionally, in order to be able to distinguish between different branches, the space between rectangle increases by the distance to the common ancestor node and subtly conveys the class hierarchy (T4). For example, in Figure 3.6 (2) the common ancestor of *A.1* and *B.1* is the root node and for both nodes the distance to the root is two. On the other hand, the common ancestor of *A.1* and *A.2* is *A*. The distance between *A.1* and *A.2* to A is only one. Therefore, *A.1* and *A.2* are placed closer together than *A.1* and *B.1*. According to the Gestalt principles discussed in Chapter 2.2, *proximity* provides a notion of togetherness, helping users to mentally group elements.

[2] https://www.w3.org/TR/WCAG20/

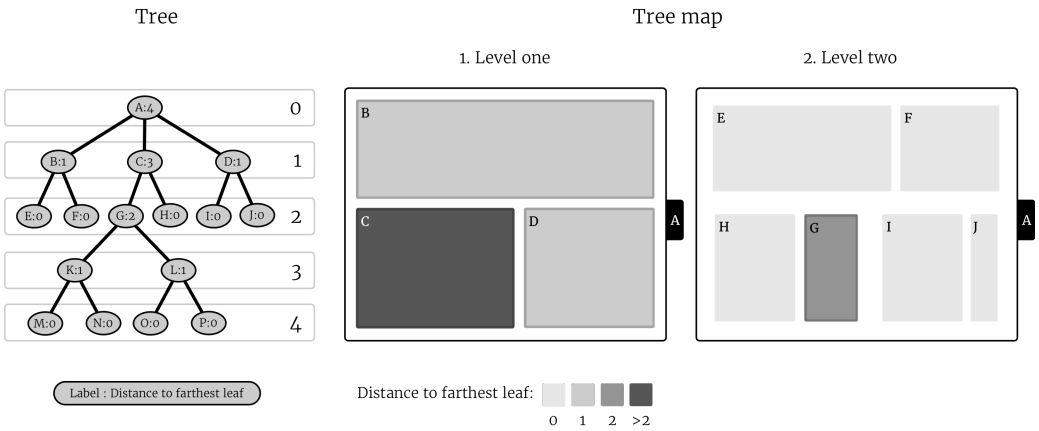

Figure 3.6: Tree map coloring, labeling and spacing.

The principles of the tree map visualization method have not changed much since their discovery (Johnson and Shneiderman, 1991). Most work related to this technique went into layout algorithms. The algorithms mostly differ in the average aspect ratio of the rectangle they generate, the spatial stability, and their ability to preserve the input order of nodes. The most commonly used algorithms are *slice-and-dice* (Johnson and Shneiderman, 1991), *Squarified* (Bruls *et al.*, 2000) and *Generalized Treemaps* (Vliegen *et al.*, 2006). Generally the desired aspect ratio should be low and both the spatial stability and ordering are ideally high but the three characteristics are negatively correlated. Also, a low aspect ratio can harm legibility (Bederson *et al.*, 2002), e.g. if text labels are longer than the rectangle, a line break needs to be introduced or the label is cut off. Higher aspect ratios allow for longer labels but too high aspect ratios make it harder to perceive the correct node size. In this project the *Squarified* tree map algorithm is used, given its relatively high stability and the low aspect ratio. Technically the tree map visualization does not need to be rectangular as demonstrated by Vliegen *et al.* (2006) but other shapes decrease space efficiency and introduce distorted shapes which size is harder to estimate. Additionally, having a rectangular layout adds the benefit of a natural element flow from the top left to the bottom right, which can be used to order rectangles according to their size.

3.2.4 List graph

The list graph is a novel visualization algorithm for flat horizontal node-link diagrams where each column or level can be seen as a list of nodes. The data to be visualized is the same as for the tree map, i.e. directed graphs that describe a containment hierarchy. The main difference to an ordinary node-link diagram is that the list graph requires a strict direction of node relationships. For example, given subclass relationships between ontology terms, it must hold that for all classes their superclass is positioned only left or only right but not both. For this project the root node is always located on the left and nodes representing subclasses are positioned to the right. Related nodes are visually linked via lines to convey the hierarchy (T4).

Another major difference is that nodes are aligned to the top instead of being centered vertically. The list graph's main purpose is to visualize ontology term abundance among a collection of data sets in regards to precision and recall to give an estimate of the most important annotations. The rational is that the term lists should act like ranked lists of search result; hence, the most relevant terms should be shown first. The benefit of this approach is that the user does not need to familiarize with a new concept since ranked list are omnipresent in search result interfaces and most file browsers. This aspect is considered to be highly important since new visualization techniques introduce new cognitive load and can be time consuming to understand, which might limit their use during the search process. While aligning nodes to the top does not make the overall visualization more space efficient but it can significantly decrease the space consumption for the top most important annotation terms. Similarly to the idea of (Pickens *et al.*, 2008) to distinguish between the retrieved search results and the results the user is actively looking at, the list graph is supposed to be an add-on visualization that provides relevant information regarding search in a limited amount of space. Since users have been found to focus on the first couple of search results only (Hotchkiss *et al.*, 2007), the same can be expected to hold for supportive search visualization techniques. For example, in Figure 3.7 the horizontal (2) and list-like (3) node-link diagram consume the same overall space as indicated by the grey border. In case the display space is limited only a fraction of the diagram is visible (shown by the dashed red border) and the space efficiency of the list-like node-link diagram is higher. Assuming that the nodes are sorted by relevance it is obvious why it is desirable to see as many top nodes as possible. Having higher space efficiency in terms of the relevant nodes that are visible, clearly comes at the cost of comprehensibility of the graph topology; the vertical (1) and horizontal (2) node-link diagrams better convey the topology compared to (3). Yet the topology of the list-like node-link diagram is still easier to grasp than the tree map's containment technique (N5). In addition to aligning nodes to the top, the list graph adds level dependent scroll bars to each level of nodes that exceeds the visible container. Here, a level corresponds to all nodes that have the same distance to the root. The usefulness of scroll bars has been studied by (Song *et al.*, 2010), which showed favorable results for a scroll bar to limit large fan-outs.

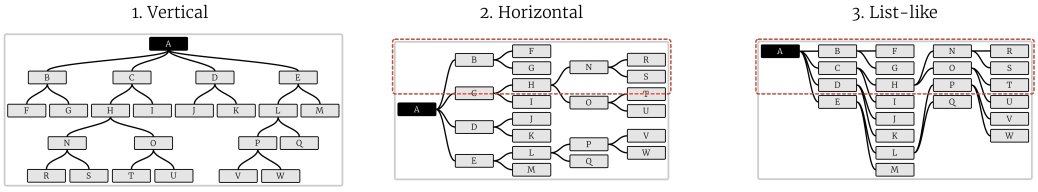

Figure 3.7: Layout comparison of node-link diagrams. Node size and spacing is constant among all three layouts.

The list graph focuses on the visual representation of two search metrics: the percentage of data sets retrieved with a specific annotation in relation to all retrieved data sets and the number of data sets with a specific annotated retrieved in relation to the overall number of data sets with that specific annotation. The first can be interpreted as precision and the latter represents recall. The list graph visualizes these two properties using superimposed bar charts (Figure 3.8). The list graph allows the user to sort each level of nodes, i.e. nodes with the same distance to the root node, according to these two properties (T10). In order to indicate which property a column is currently sorted by, a bar can be active or inactive. The visual representation of the active and inactive state depends on two bar chart modes (Figure 3.8, 1 & 2). The rational for having two modes is two-folded: First, having one bar compared to two is less distracting and reduces the cognitive load. On the other hand, it is harder to see the value of the inactive property that is only indicated by a superimposed vertical line, hence when both values are important two bars are favorable. The other reason for having two modes is that the final exploration tool should be beneficial for both: search and browsing. When browsing data according to ontology annotations, recall is always equal to one and thus does not need to be shown. In conclusion, having one bar is ideal for browsing while two bars are preferable during search. The reason for choosing bar charts is due to their high effectiveness for numerical data representation, especially when they are aligned and scaled–as it is the case for the list graph. Superimposing the bars over the nodes has the benefit of reduced space consumption, which is crucial as the overall screen space is limited.

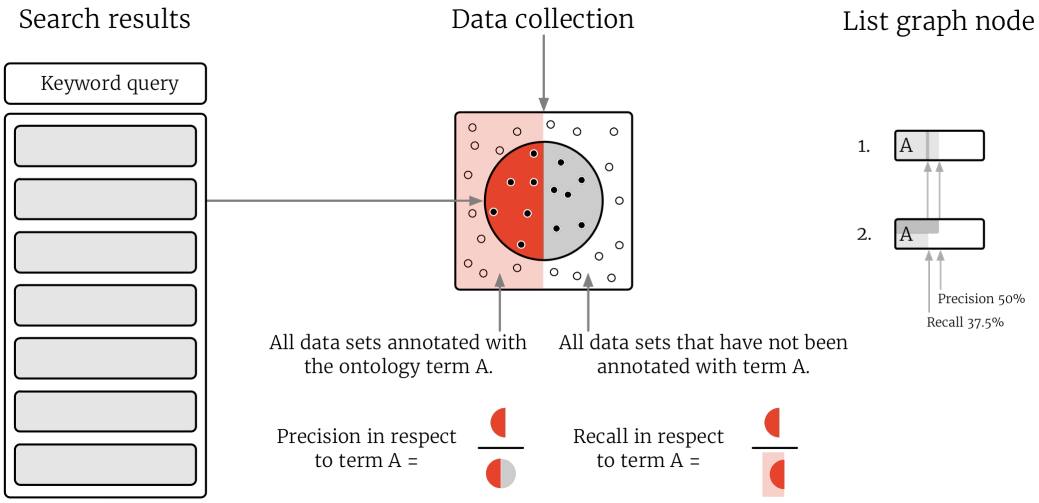

Figure 3.8: Term property visualization via superimposed bar charts. Currently two modes are supported: (1) *Precision* is set as the active property and its related bar makes up the whole node. *Recall*–the currently inactive property–is indicated as a superimposed line. (2) The two properties are visualized using two separate superimposed bars within the same node where each bar takes up half of the node's height. The active bar is filled with a slightly darker grey compared to the inactive bar.

A term's precision visualization and the size of its rectangle in a tree map are strongly related since both convey the relative abundance of data sets annotated with a term. The main difference is that the list graph can depict precision in relation to a fixed area while the tree map visualizes the ratio among all rectangles such that the total area is used. Figure 3.9 illustrates the difference between the tree map's and list graph's property visualization principles. In both cases (1) and (2) the total number of data sets that have been annotated with some ontology terms is equal to six and the number of data sets associated with term *A* is the same. While the tree map looks identical, the list graph's superimposed bar chart conveys the relative abundance of each term. The tree map does not provide any insights into the intersection of annotated data sets. As a data set can consist of multiple studies with different assays of various raw data files the intersection of term-related data collections can be high, especially the more generic a term is.

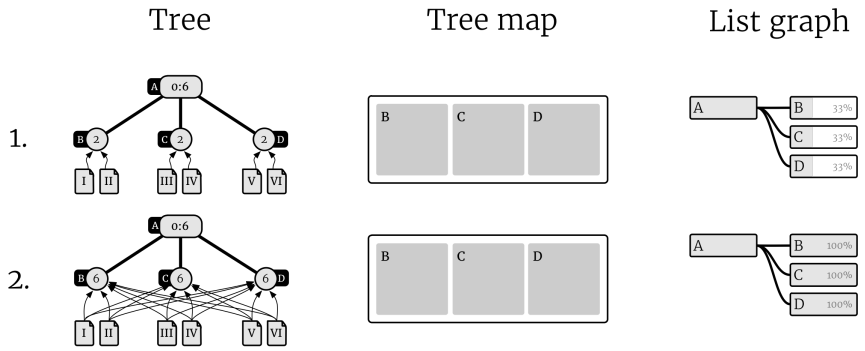

Figure 3.9: Tree map versus list graph. The size of the tree map's rectangles is compared to the list graph's precision bars.

Visualizing precision and recall can help to understand the outcome of a search in regards to ontology annotation terms. Figure 3.10 illustrates four general cases of a search in respect to one annotation term. The first case (Figure 3.10, 1) shows that the text-based search almost perfectly retrieved data sets of a certain characteristic, i.e. precision and recall are high. This can either be an indicator that the search process can be stopped, to further drill-down via annotation term filtering, or by further specifying the search query. In the second case (Figure 3.10, 2) the search returns documents that still match the annotation term perfectly in terms of precision but is too specific to match all data sets that have been annotated with this term. If this annotation term is of desire then a possible action could be to loosen constraints by generalizing the search query. In the third case (Figure 3.10, 3) all data sets of the specific annotation term are retrieved but this characteristic is not very specific in regards to the search query, i.e. recall is high but precision is low. Further specifying the search query or filtering down the search by ontology-guided quarying can help to exclude irrelevant results. In the final case (Figure 3.10, 4) the search

query is unspecific to the illustrated annotation term. Recall and precision are low. In this case a more thorough revision of the search query might be needed.

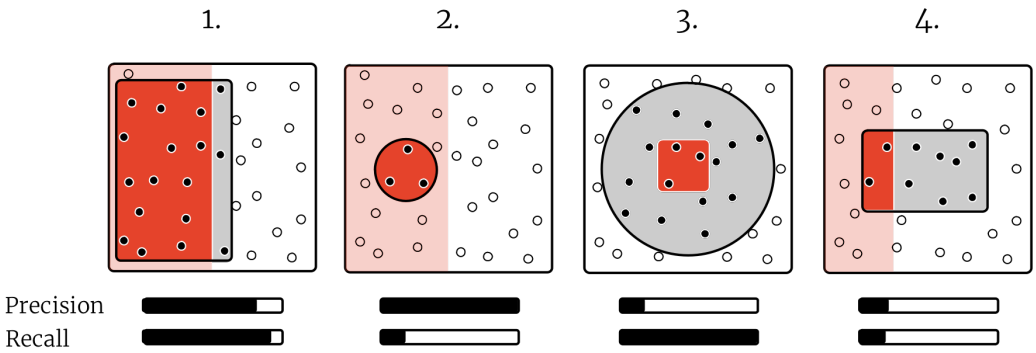

Figure 3.10: Different search result outcomes. The large rectangle represents the data repository. Contained circles stand for data sets. The area shaded in light red indicates data sets that have been annotated with a specific ontology term of interest. The inner rectangle or circle bordered in black illustrates search results. The light grey area stands for retrieved data sets that have not been annotated with the specific term, while the area filled with saturated red contains retrieved data sets with the desired annotation term.

3.2.5 Preview

When exploring data collections with the goal of finding relevant data (N1, N2) it is desirable to have a quick way to obtain a summary of a data set's content in order to verify whether this data set is really relevant or not (T5). Previewing the content of a data set can be achieved in many different ways. Important aspects to be considered are speed, simplicity and context preservation. Previewing a data set's content should be faster than opening the data set; otherwise the preview does not add any value. Simplicity relates to the actions needed to open the preview; i.e. the fewer actions needed the better. Context preservation helps to keep the cognitive load low when scanning the summarized content and returning to the search results. By keeping these aspects in mind, the data set preview has been integrated into the search interface as a slide-in window that is visually linked to the search results and the data set being previewed. Figure 3.11 illustrates the integration. A click on the right arrow on the right of a search result will slide in a preview panel, which in turn pushes out other content. The whole process is animated to assist comprehension of the transition. Keeping the search results visible at all times helps to preserve the current context. In general, the search interface is not altered in any way when previewing a data set's content.

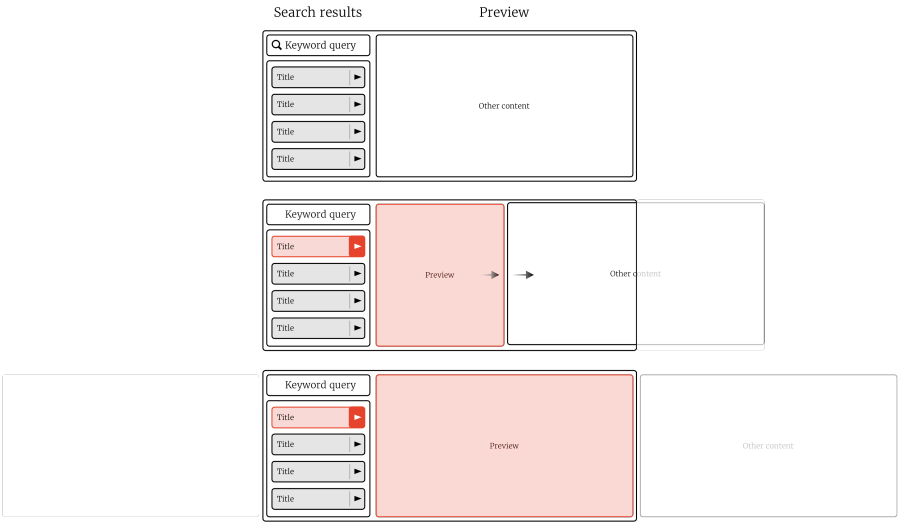

Figure 3.11: Data set preview integration.

The composition of the preview's content has been designed with the findings of the interviews (Chapter 3.2.2) in mind. The following metadata is shown in the given order:

1. Short description
2. Technology & measurement type
3. Sample source & species
4. Number of files
5. Analyses that have been run on the data set
6. Reference: title, authors, journal, manuscript source, PubMed and abstract
7. Wet and dry laboratory protocols

3.2.6 Interactions

The three aspects of data set exploration described in Chapters 3.2.3, 3.2.4 and 3.2.5 are combined with powerful text-based search (T6) to form the final exploration system to enhance findability and discoverability (N1). Figure 3.12 illustrates an early mock-up of the integrated interface. The list of results (Figure 3.12, 1a), list graph (Figure 3.12, 2a) and tree map (Figure 3.12, 2b) are linked via user interactions. Hovering over elements of either panel will highlight the related parts in the other two panels. The following is a list of interaction tasks:

I1 Hovering over a search result should highlights the corresponding annotation terms of this data set. (T1)

I2 Hovering over a node of the list graph or rectangle of the tree map should highlights all currently retrieved data sets that have been annotated with this term.

I3 The relative root of the tree map should be adjustable; i.e. zooming by

branch, and being reflected in the list graph.

I4 The visible depth of the tree map should be easily adjustable.

I5 Term highlighting should be lockable to allow comparison of different terms.

I6 The list graph should be scrollable by level and globally draggable and zoomable to uncover hidden areas.

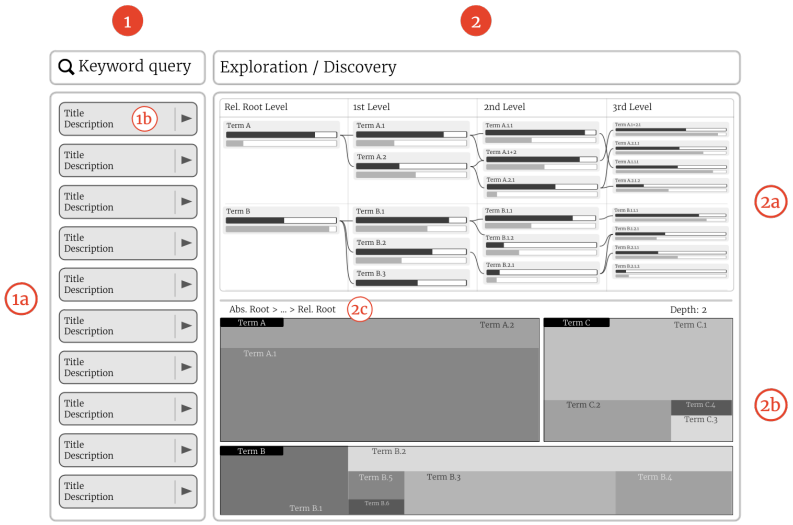

Figure 3.12: Early mock-up of the final exploration interface. (1) Search panel with query input field and results list (1a). Each retrieved data set is represented by an item in the list, consisting of (1b) its title, a very brief description and a button for previewing the data set. (2) Exploration panel comprised of the two visualization methods: the list graph (2a) and tree map (2b). The tree map panel includes a linked node list of the path from the absolute root term to the currently viewed root term.

3.3 Implementation

The exploration system is integrated into the Refinery Platform (http://refinery-platform.org), which consists of server-side back-end storage applications, a middleware controller application and a client-side front-end application. The middleware application is the central part that connects the different components and manages the logic. It is written in Python and builds upon the Django framework (http://djangoproject.com). PostgreSQL (http://postgresql.org)—a widely used open-source relational DBMS—is the main data storage for the Refinery Platform. It is complemented by Neo4J's graph database (http://neo4j.com), which additionally stores ontologies and annotation data. Solr (https://lucene.apache.org/solr) is being used as the IR system to provide a Google-like search experience. The server-side storage and IR systems feature application programming interfaces that support the representational state transfer architecture and communicate via JavaScript object notation messages. Communication is routed through the middleware application to

normalize requests. The front-end application handles most of the user interactions and utilizes AngularJS (http://angularjs.org) to provide a rich user experience. Visualizations are implemented in Hypertext Markup Language 5, Cascading Style Sheets 3 and Scalable Vector Graphics and are mainly orchestrated by JavaScript (JS) and Data-Driven Documents (Bostock *et al.*, 2011). The AngularJS architecture is similar to the server-side middleware application as it ties together the different visualizations and other user interface components and manages the information and event flow. The front-end application is build via Grunt (http://gruntjs.com). The source code of the Refinery Platform including the exploration system, extensive installation instructions, help, and background information is hosted on GitHub: http://github.com/parklab/refinery-platform. The list graph has been implemented as a separate application to support future reusability but is fully integrated into the Refinery Platform via an AngularJS wrapper application. The list graph is implemented in the latest version of JavaScript (known as ECMAScript 6 or 2015) and is build by Gulp (http://gulpjs.com). The source code is available on GitHub as well: https://github.com/flekschas/d3-list-graph. Both applications are continuously integrated via Travis-CI (https://travis-ci.org) and the Refinery Platform features several unit tests to ensure validity.

3.3.1 OWL to Neo4J parser

In order to access the ontology term hierarchy quickly, ontologies are converted in a simplified graph structure and imported into Neo4J. For that a Java-based parser has been implemented, which extracts the terms and their direct subclass relationships. The parse utilizes the OWL API (Horridge and Bechhofer, 2011) and currently supports OWL documents in the Extensible Markup Language format. Ontologies to be imported are checked for validity using HermiT (Glimm *et al.*, 2014) prior to processing. The parser is implemented in Java 7, is compiled into a Java Archive via Gradle (http://gradle.org) and continuously integrated via Travis-CI. The source is publicly available at https://github.com/flekschas/owl2neo4j. In addition, to simplify the collection and import of ontologies related to Stem Cell Commons a small utility repository holds batch download and import instructions at https://github.com/refinery-platform/ontology-imports.

In order to ensure data integrity, a minimal graph schema has been developed to guarantee uniqueness of ontology terms. As listed in Table 3, nodes that represent ontology classes are labeled by *Class* and must have a unique URI. Additionally an index on the name property is added to accelerate lookups. In order to keep track of imported ontologies, a meta node for each ontology is created and labeled *Ontology*. Nodes being labeled with *Ontology* must have a unique URI and acronym. OBO's *IDSPACE* is commonly used as an acronym; e.g. GO is the *IDSPACE* of the Gene Ontology and used as the acronym. OWL ontologies do not have the notion of an *IDSPACE* but support prefixes.

Label	Constraint
Class	URI must be unique
Class	Index on property name
Ontology	URI must be unique
Ontology	Acronym must be unique

Table 3: Graph database schema constraints.

Additionally, each imported term is labeled by its ontology's acronym. This is similar to OBO's namespace and helps to manage large numbers of ontologies. It should be noted that no term is being duplicated but instead reused when importing multiple ontologies. This has two benefits: First, many low level terms that are used across various ontologies are merged into one node, significantly saving disk space. Second, every query automatically includes all possible relationships over all ontologies. The parser extract satisfiable classes, i.e. classes that do not equal *OWL:Nothing*, and iterates over each class' direct superclasses. Direct subclass relationships do not include transitive relationships and are therefore preferable for visualization purposes, since the transitivity should be illustrated by the visualization explicitly. Additionally, the parser is able to extract existential quantification properties (EQP) of subclass relationships.

4 Results

The landing page of the Refinery Platform is illustrated in Figure 4.1. The dashboard provides three main panels for data sets, analyses and workflows. All three panels work similarly. The major difference of the data set panel is the ability to search and explore data sets. Since this project focuses on ontology-guided exploration of biological data only, other parts of the Refinery Platform will not be covered here.

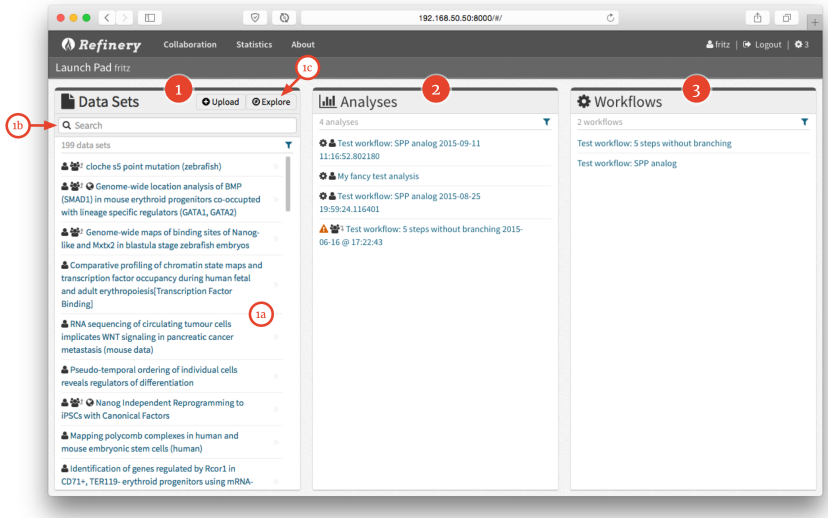

Figure 4.1: Landing page of the Refinery Platform. (1) Data set panel. (1a) All currently imported data sets are listed by their title. Glyphs depict basic ownership and sharing information. (1b) Search is incorporated into the same interface and consists of a simple query input field. (1c) The exploration panel can be activated via a click on *Explore*. (2, 3) The analyses and workflow panels behave in the same way as the data set panel but do not feature search or exploration tools yet.

4.1 Application

The different components of the exploration tool are shown in Figure 4.2. The data set panel is explicitly kept identical to the landing page to provide a familiar context and keep visualization methods as an add-on exploration tool. Hearst (2009, Chapter 1 & 3) states that search is an mentally intensive task by means of finding data and sensemaking of results. Hence, the user interface should try to keep the cognitive load minimal. The visualization panel (Figure 4.2, 2 & 3) expands from the data set panel and pushes out the other content. The list graph (Figure 4.2, 2) and the tree map (Figure 4.2, 3) share about two third of the screen. On the initial page load without any search, the list graph and tree map show the repository-wide abundance of annotation terms. Exemplarily, Figures 4.2–4.8 show the usage of cell type related

ontology term annotations from CL. The exploratory visualization interface follows the famous *Visual Information-Seeking Mantra* (Shneiderman, 1996) by providing overview first and details on demand. Both visualizations are initialized with the absolute root term, which in this case is *native cell* (CL:0000003). The currently active root node is highlighted via bold black text and a softly glowing blue underline (Figure 4.2, 3b). The list graph does indirectly highlight the root nodes as the nodes being in the top left corner of the visualization. The list graph supports multiple root nodes to be shown. Since not all data sets have been annotated with some ontology terms a pseudo term called *No annotations* is added. *No annotations* is linked with the actual root node *native cell* via a pseudo root node that is not shown as it simply represents that repository as a whole. In order to help the user find the link between the tree map's rectangles and list graph's nodes, the currently visible layer of inner nodes in the tree map is shown in the list graph as the column with a light grey background and slightly darker node borders.

Both visualization idioms feature a top bar for controls and extra information (Figure 4.2, 2a & 3a). The list graph's top bar consists of six global options: sorting by precision, recall, and name as well as an option for each of the two bar chart visualization modes and a button to show the whole graph at once. In addition, a click on the right most arrow in the top bar will uncover options for sorting each level of nodes by precision and recall individually (Figure 4.3). The current state of each top bar setting is depicted by an icon next to it. Active settings are highlighted by increased font weight, text underline and a darker font color.

The tree map illustrates the number of data sets associated with a term by the size of a rectangle and sorts them in descending order; hence, the largest rectangle is listed top left. The rectangles' area reflects the ratio between all visible terms' sizes. The color of a rectangle indicates the distance of the farthest leaf node. Both channels provide an immediate motion for how many data sets are associated to a term and how much many more sub terms can be explored. The list graph directly visualizes related sub terms via a line between nodes and the precision bar depicts the number of data sets relative to the active root term. Subclass relationships are all drawn in the same direction from left (superclass) to right (subclass). Terms with multiple parents are duplicated when the parents' distance to the root is not equal. Duplicated nodes are illustrated by a dashed rather than a solid node border. To avoid clutter, sub graphs of duplicated nodes are omitted, instead the first instance, which is closest to the root node, provides the sub graph only.

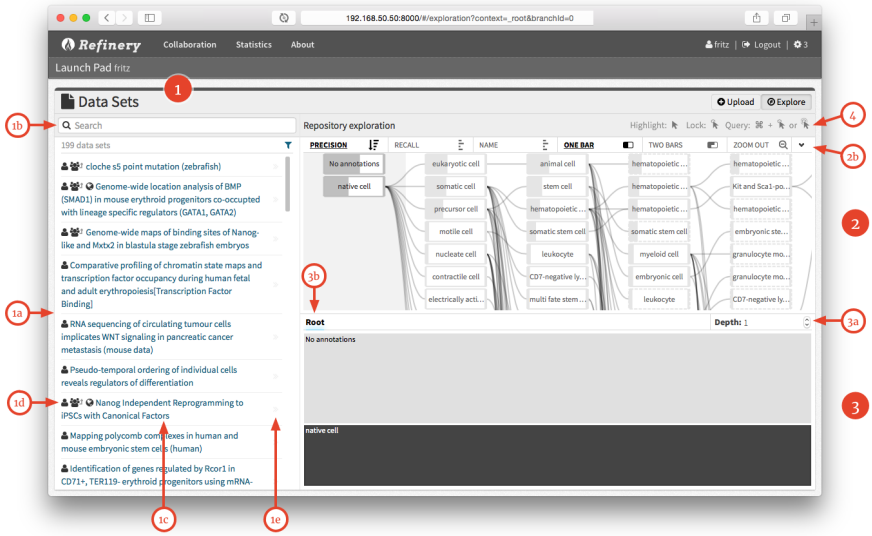

Figure 4.2: Exploration interface. (1) The left part of the interface remains unchanged compared to the landing page. The right part is divided into (2) the list graph and (3) the tree map panel. The list graph consist of the main visualization and a top bar (2a). The tree map is similarly composed of a top bar and the main visualization below. The top bar features breadcrumb-inspired path view to the absolute root term (3b) and an input field to increase the depth for zooming (3a).

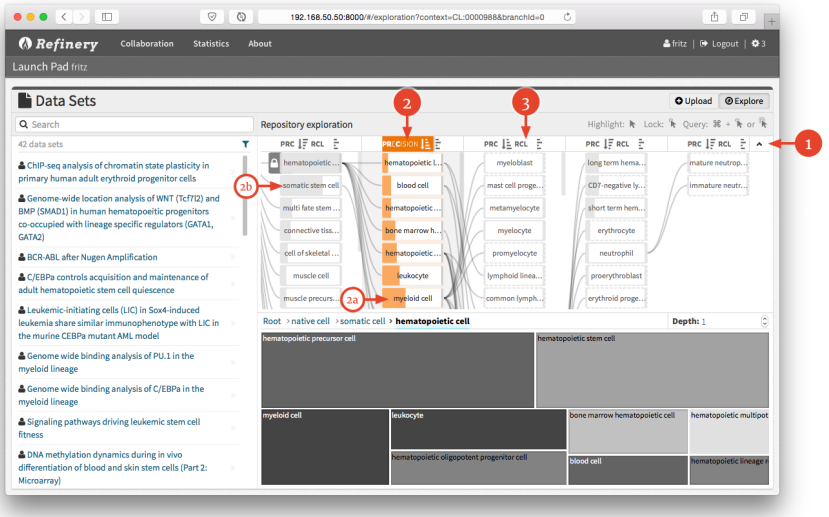

Figure 4.3: Column-wise sorting. (1) A click on the arrow uncovers column-wise sorting settings. Each column of nodes can be individually sorted by precision (2) or recall (3).

Hovering a button will highlight that column's nodes' attribute (2 & 2a). A click on either button will toggle through ascending and descending sorting modes. The current sort order is depicted by an icon next to the button that triggers sorting (2 & 3). Other columns will remain unaffected (2b).

There are many interactions that connect the list of data sets, the list graph, and the tree map. Since novice users do not know how to interact with the exploration tool, a list of basic interaction guide is provided in the top right corner of the exploration panel (Figure 4.2, 4). Interactions can broadly be categorized in highlighting, browsing and querying. Highlighting works in two way: it reveals a data set's annotation by showing related ontology terms in the list graph and tree map and all data sets related to a term are highlighted when interacting with a term in the tree map or list graph (T1). Mousing over the relevant parts of the exploration interface triggers both interactions. For example, to reveal the annotations of a data set, the user can mouse over a list entry and the related annotation terms in the list graph and tree map are simultaneously highlight by changing the grey scale color to an orange hue (Figure 4.4, 1). Hereby, the saturation of a highlighted rectangle in the tree map is kept constant to the luminance of grey, thus, lighter shades of orange indicate that a node is closer to a leaf than a more saturated orange. The tree map highlights direct or indirect annotation terms; e.g. the data set being hovering in Figure 4.4 is annotated with *neuronal stem cell*, which is not directly visualized at the current tree map level, hence the parental terms *somatic cell* and *precursor cell* are highlighted. The list graph visually distinguishes between direct and indirect annotations such that nodes related to direct annotation terms are fully colored in orange and indirect terms are only bordered in orange.

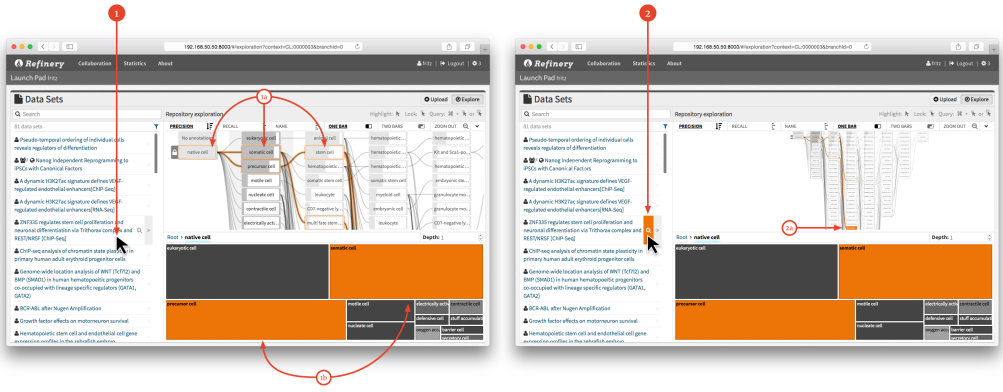

Figure 4.4: Data set-wise annotation term highlighting. (1) Normal mousing highlights relevant nodes in the list graph (1a) and tree map (1b). (2) Pointing the mouse cursor on the small magnifier icon will zoom out the list graph to the extent that all direct annotations are visible (2a).

Hovering an ontology term in respect to the list graph and tree map is illustrated in Figure 4.5 (1). Nodes and rectangles are highlighted by a black border (Figure 4.5, 1, 1a & 1c). When pointing the mouse over a node in the list graph, the directly hovered node's color is additionally inverted completely–i.e. the background turns from white to black–to make it pop out from indirectly hovered super and sub terms (Figure 4.5, 1c). List entries are shifted a little bit to the right, the font color turns black and an extra black border is added to the left, hence using two channels: color and motion to make related data sets pop out. Locking a term has been implemented to provide a possibility for comparing different annotations. While normal highlighting will be reverted as soon as the user leaves the hovered node or rectangle, a locked term will remain highlighted until the user unlocks it. A small lock icon depicts whether a term is locked (Figure 4.5, 2 & 2a). The tree map colors locked terms in orange while the list graph adds a thick orange border to the node. A term can be locked by a single mouse click on a rectangle or a click on the lock icon that appears right to a node when hovering it. Data sets that are associated to the locked term appear with a light orange background and a thick left orange border (Figure 4.5, 2b).

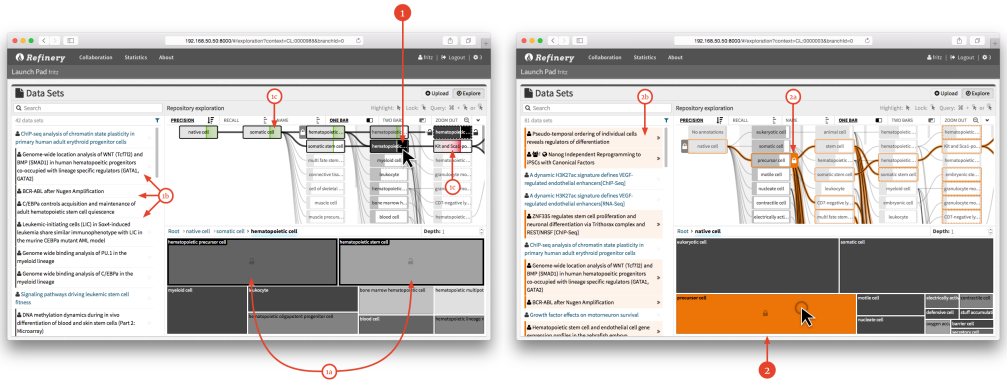

Figure 4.5: Term hovering and locking. (1) Hovering a node of the list graph (or rectangle of the tree map) highlights the term itself as well as related super and sub terms in the list graph and the tree map (1a). Data sets that have been annotated with the term being hovered are highlighted as well (1b). Additionally, super and sub terms of the directly hovered node in the list graph feature a superimposed partial bar to compare the value of the directly hovered node with the indirectly hovered nodes. Green bars (1c) indicate an increased and red bars (1c) indicate a decreased value compared to the hovered term. Locked nodes are highlighted in orange (2).

There are two ways of browsing the term hierarchy: expanding a sub tree by re-rooting the list graph and tree map or increasing the number of levels seen simultaneously. In addition to visually browsing a branch, the data collection will be queried according the new root. Thus, only data sets that are associated with the new

root term will be retrieved. Figure 4.6 shows the transition from *native cell* (Figure 4.6, 1) to *stem cell* (Figure 4.6, 2). Browsing along the term hierarchy can be triggered by a double click on a tree map's rectangle or by a single click on the lock icon left to a node in the list graph. The lock icon appears individually for each node when the user hovers over the node of interest.

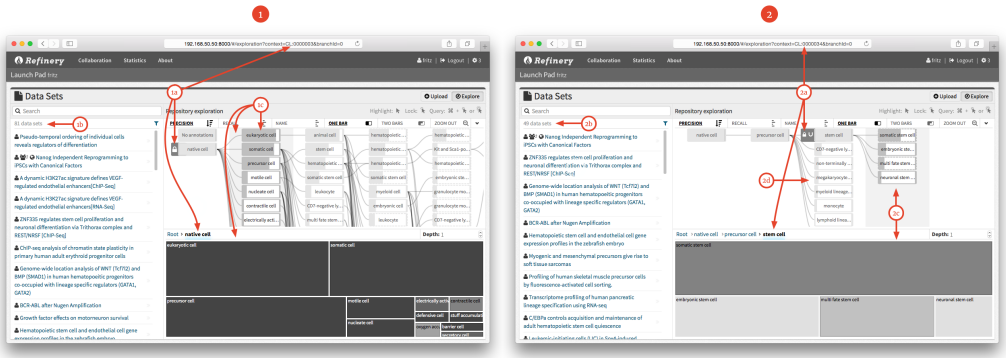

Figure 4.6: Browsing by branch. (1) The data collection is currently queried for *native cell* (1a), indicated by the URL, a small lock next to the corresponding node in the list graph and the highlighted term in breadcrumb-like navigation panel. 81 data sets (1b) are associated to *native cell*. After drilling-down to *stem cell* (2 & 2a) the number of retrieved data sets decreased to 59 (2b). Terms that are visualized simultaneously in both diagrams are indicated by (2c). During drill-down the list graph shows siblings of the current root node (2d) for quick comparison but hides siblings of higher-level terms in order to avoid visual clutter.

In addition to browsing by branch, the user can increase the visible depth to show nodes farther away from the root node (Figure 4.7). Given a visible depth of i, all inner nodes at distance i (from the root node) and all leaf nodes with a distance of $<= i$ are shown. Completely white areas such as in Figure 4.7 (2) indicate very small terms such that the branch-related extra padding prevents the rectangle form being shown.

Sometimes it is desired to querying the data collection (T7) and filtering subsets (T8) according to ontology annotations without wanted to browse just one specific branch of the class hierarchy. The exploration tool provides set theoretic queries across the whole annotation set hierarchy via the list graph visualization. Pointing the mouse cursor over a node will display two icons to the left of the node. The lock icon triggers browsing by branch and the union icon stands for querying. A click on the latter or on the node will initiate a *union* query. Multiple clicks on the same node will toggle through four different query modes as described in table 4. The two modes *union* and *intersection* produce the same results when only one term is queried at a time. For example, making a union query for *Mus musculus* and an intersection query for *stem cell* will retrieve only data sets that are annotated with

both terms. When a third intersection query for *neural cell* is issued the retrieved data sets must be annotated with *Mus musculus* and either or both of the terms *stem cell* or *neural cell*. Figure 4.8 illustrates the query capabilities of the exploration tool.

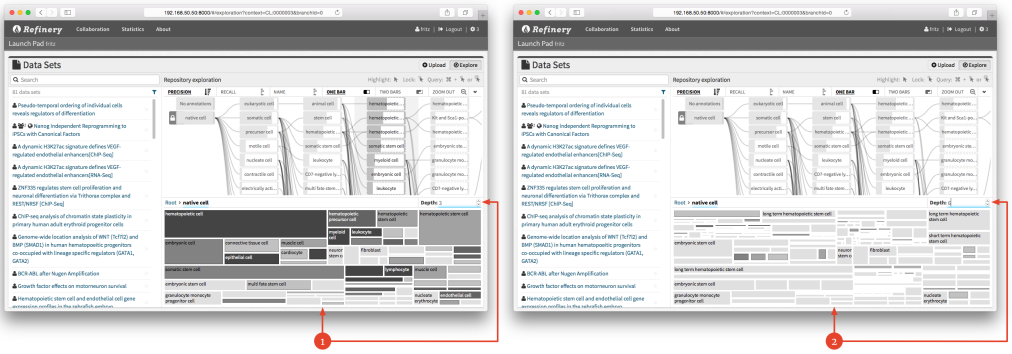

Figure 4.7: Tree map's level zoom. (1) The visible depth is set to three. (2) The visible depth is increased to nine.

Number of clicks modulo four	Query mode
0	Inactive: retrieved data sets might be annotated with this term.
1	Union: retrieved data sets must be annotated with either of the terms queried by union.
2	Intersection: retrieved data sets must be annotated with this term.
3	Exclusion: retrieved data sets must not be annotated with this term.

Table 4: Description of the four different query modes.

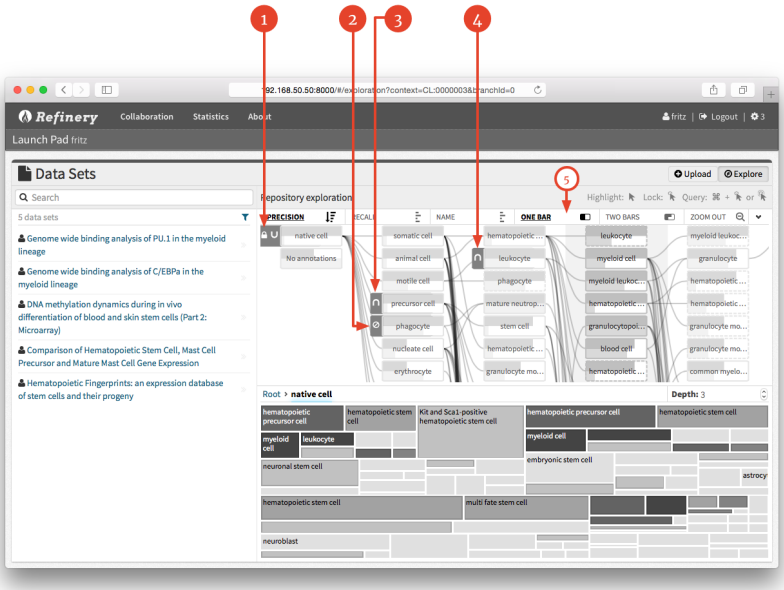

Figure 4.8: Annotation term querying and 2-bar visualization mode. (1) List graph has been re-rooted to *native cell*, which simultaneously acts as a *union* query. Additionally, (2) all data sets related to *phagocytes* are excluded and only the intersection between (3) *precursor cell* and (4) *leukocyte* related data sets are retrieved. Recall of annotation term sets are visualized next to precision by enabling the 2-bar visualization mode (5).

During the exploration process the user can choose to preview a data set before having to navigate to it. Figure 4.2 (1e) indicates the location of the button to open a data set's preview panel. Once a user hovers over a result list the preview button becomes more apparent. The layout of the preview is shown in Figure 4.9. The different parts of the preview panel have been carefully designed and adjusted according the results of the interviews (Chapter 3.2.2). The technology and biological sample description are listed right after the short description. As they generally seem to be most important to relevance. The analyses panel provides a notion of how popular a certain data set is and even though it isn't regarded as a key characteristic for exploratory search it is immensely important for navigational search and the broader purpose of the Refinery Platform as an integrated data management, analyses and visualization system.

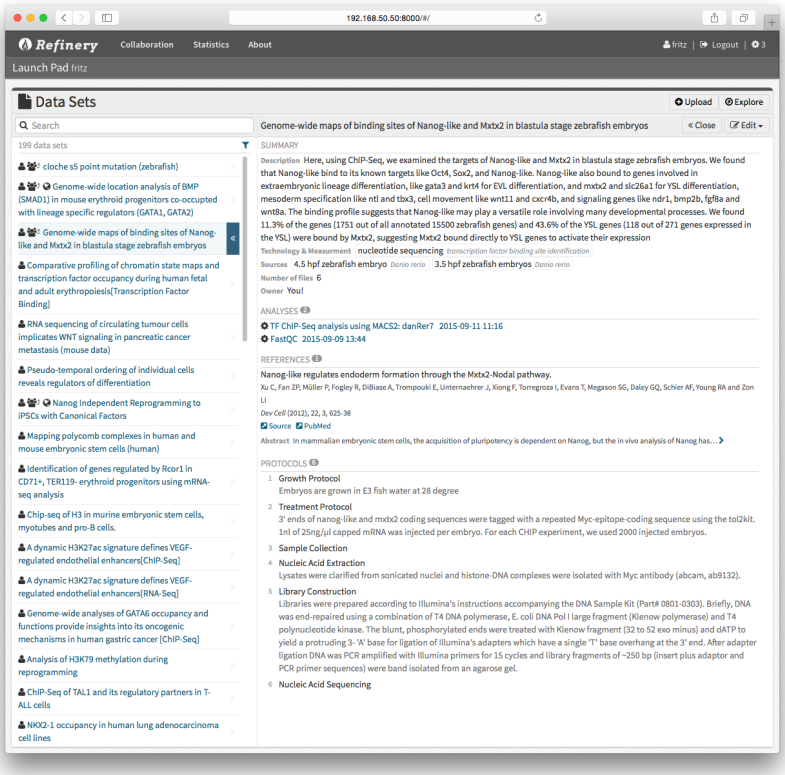

Figure 4.9: Data set preview panel. The panel is divided into four main parts: summary, analyses, references and protocols. The *summary* panel is comprised of a short descriptive paragraph, information regarding the technologies and measurement types, biological sources, the number of data files, and ownership. The *analyses* panel holds all analyses that have been run on the data set from within the Refinery Platform. The *references* panel provides the title, author, journal, publication date, a source link, and a link to PubMed as well as an expandable abstract if available. Finally, the *protocols* panel lists sample treatments, data generation, and post processing actions taken prior to importing into the Refinery Platform.

4.2 Case study

In order to demonstrate the usefulness of the proposed ontology-guided exploration system, the Stem Cell Commons data has been used as a real-world case study. The scope of this study has been limited to the exploration along the EFO given that 198 out of 199 Stem Cell Commons ISA-Tabs are directly or indirectly annotated with a high number of EFO terms. The EFO provides a suitable topology for exploration since it's main focus is on structured descriptions of experimental factors of biological experiments.

Data scientists have a strong need to find data sets that match experimental characteristics (N1) and explore data sets with similar experimental factors (N2). The

exploration tool addresses both needs by an integrated single-page application of text-based search, data set previewing and the two visualization techniques: the list graph and tree map. Often exploration is a multi-step process of identifying putatively interesting data sets by sequentially scanning information. For example, when searching for *blood stem cell* only four out of seven results contain information about blood in the result list item. Hovering the cursor over the results immediately provides feedback on the annotations. Also, a search *zebrafish stem cell* retrieves only one query. While it is often unknown if truly everything is found, a quick exclusion query on *Danio rerio* reveals that only one stem cell related zebrafish data set exist. Generally, desired or undesired experimental factors can quickly be requested or excluded after issuing a search and help to filter down results. Most importantly, the data set preview and visualization interface does not interfere with search panel, providing easy-to-use extension. Especially the data set preview is helpful to understand search results when the search result's keyword in context preview and visualization do not reveal sufficient information.

Exploring the repository via branch-related queries and investigation of the tree map at different levels of depth provides a quick overview of the annotation set hierarchy of collections of data sets (N3) and helps to find higher-level sets that are hidden by normal search.

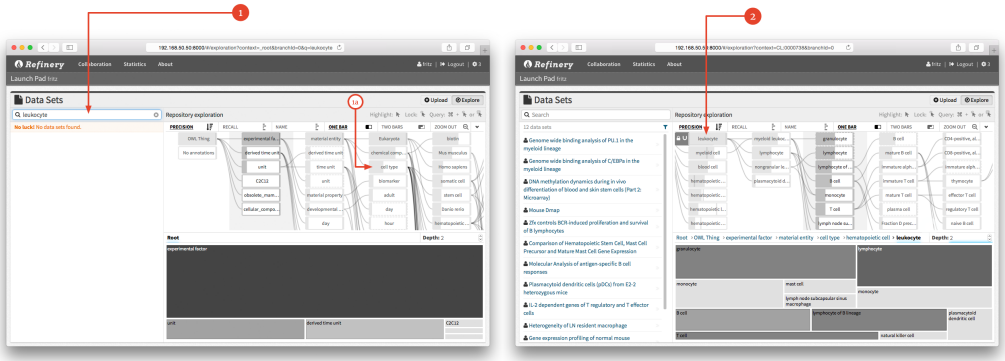

Figure 4.10: Exploring high-level annotation sets. (1) Text-based search does not retrieve any data sets related to *leukocyte*. Exploring Stem Cell Commons *cell type* related data sets (1a) reveals *leukocyte* annotation set (2).

For example, given that a project leader, group leader or funder wants to get an overview of all leukocyte-related data sets, searching for *leukocyte* leads to no results (Figure 4.10, 1) but exploring different cell types quickly reveals a group of 12 data sets that are associated with leukocytes via subclass relationships (Figure 4.10, 2); e.g. granulocytes, lymphocytes, B cells, T cells, monocytes, etc. The same holds for many other higher-level terms that have not been used for direct annotation but describe larger collections of data sets due to the transitive nature of subclass

relationships. Hereby, the combination of the list graph and tree map empowers rich exploration that is hardly possible with text-based search only.

Finally, data curators and ontology engineers would like to get insights in the overall annotation structure to identify areas that can be improved in terms of annotations or term specificity. Looking at the abundance of *No annotation* or the *OWL:Thing* can quickly identify the overall annotation coverage. In this case study, a group of two data sets have been found instantaneously that are annotated with the obsolete class *obsolete_mammary gland*, which should ideally be avoided. Browsing *experimental factor* shows that most annotations are related to material entities and provide the deepest exploration paths. Increasing the visual depth to three gives a clearer overview of the annotation set topology (N5). Most of the largest annotation sets are close to *experimental factor*, indicating little to explore (Figure 4.11, 2). For example, when exploring *chemical compound* annotations a clear imbalance is apparent. Two-thirds of all data sets are annotated with a chemical compound and almost all correspond to *Biotin* (Figure 4.11, 2b). Hence the mutual information—a measure for the reduction of uncertainty when knowing specific information—of *biotin* is relatively low compared to most other terms. While it is expected that many experiments include biotin due to it's high binding affinity to streptavidin, which is commonly used for isolation, purification or separation and called biotinylation. Exploration-wise, those terms are less interesting, as they do not help to find specific data sets. For a data curator it could be explored what other chemicals are associated with data sets to provide a richer description.

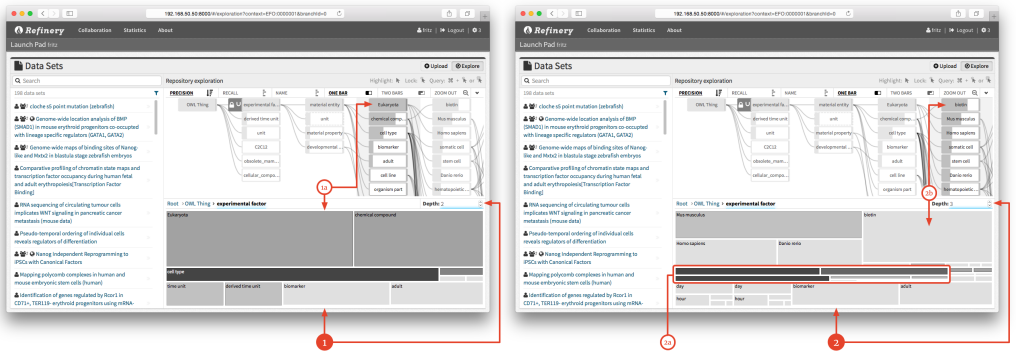

Figure 4.11: Annotation set overview. (1) The tree map provides holistic overview of all terms of distance two. (1a) The tree map and list graph visualization metrics—precision and area—complement each other. (2) Depth of three reveals that most of the largest annotation sets are close to the experimental factor. (2a) *Cell type* seems to provide most exploration depth. (2b) Two-thirds of all data sets are annotated with *biotin*.

5 Discussion

5.1 Conclusion

The ontology-guided exploration tool present throughout this thesis enables users to explore large biological data repositories by means of text-based search, visual browsing, and term-related querying. This application enriches discoverability and findability of data collections that have been ontologically annotated by visualizing semantic relationships of the annotation terms. The main contributions of this project include the conversion of ontologies into a simplified global graph structure that allows easy and quick access to the class hierarchy across multiple ontologies and is far less resource intensive than triple stores. Furthermore, ontology-compliant pruning of the original topology in accordance to the available data sets provides an annotation set hierarchy, which is highly optimized for exploration. The visual exploration interface is a unified single-page application that combines powerful search, data set previewing with the tree map and list graph visualization to provide an integrated exploration experience. The visualization serves two purposes: sensemaking and pattern discovery as well as ontology-guided querying of the data collection and could be regarded as a user-friendly semantic query tool. Finally, two key limitations of ontology-guided exploration described by Gehlenborg (2010) have been successfully overcome: multiple ontologies can be visualized and queried and the user interface integrates both keyword-based querying and visual browsing, giving the user the ability to use both simultaneously.

5.2 Limitations

The current exploration system should be regarded as a proof of principle only. The reason is twofold: first, the decisions drawn from the needs and required tasks as well as the interview lack a large-scale evaluation. While the current application enriches existing system, it needs to be proven that users adapt these contributions in their search process. Also, the implementation provided needs further improvements that are outside of the scope of this thesis. For example, to allow the exploration of many large ontologies simultaneously a server-side component needs to preprocess the ontology structure to relieve the client-side visualization application and ensure responsiveness. Furthermore, the current integration of the search system and visual exploration tools is limited in two ways. First, there is only one incremental exploration step possible, i.e. when the user initiates a search query the only additional step possible is to filter down these search results. The user cannot initiate a second search after filtering without resetting previous actions. Second, there is currently no possibility to loosen constraints of search results via annotation term queries as described in (T9). Also, comparing different groups of annotations terms (T3) is currently only possible indirectly.

Apart from the application side, the currently greatest concern regarding an ontology-guided exploration approach is that its usefulness dramatically depends on the quality of data curation. Most data collections provide none or only very few ontology annotations. And even if ontology annotations are available, other metadata are currently not incorporated. Fortunately, using this exploration tool it is easily possible to evaluate the current state of annotation and find areas for improvements. Also, having successfully addressed previous limitations and allowing an arbitrary number of ontologies to be visualized introduces a novel concern: how to determine meaningful entry points for each ontology in an automated fashion? Quite often terms close to the absolute root term (*OWL:Thing*) are very technical and provide little to no use for browsing.

Another limitation arises from pruning the original term hierarchy to a strict containment set hierarchy where each subclass must represent a strict subset in terms of the data sets associated to it. While this is desperately needed to provide a data structure for exploration that avoids meaningless browse steps, this heavily obscures the original structure and hides intermediate terms. For example, the data used in the case study described in Chapter 4.2 originally consists of 17239 ontology classes of which only 83 terms have been used to annotate the Stem Cell Commons data. Including all parental terms up to *OWL:Thing*, a total of only 156 terms link to data sets and thus are useful for browsing. Users might wonder what happened to unused or intermediate terms.

5.3 Outlook

The current exploration tool can be extended in multiple ways to further increase the interactions between classic search and exploration. An application that captures the individual steps of the whole exploration process could address the currently limited interplay between the search and the two visualization-based browsing methods. Often the users go back and forth, change query keywords or browse in different directions depending the previous results. Having a way to look at previous steps, link individual results, and provide an overview of the area already discovered without having to leave the current view could facilitate understanding of the consequences of a new search step and also point out undiscovered areas of the data repository. Given Shneiderman's *Information Seeking Mantra* (Shneiderman, 1996), this feature would relate to *history*. One of the principal reasons of a data scientist or analyst for searching for data is to compile a collection of data sets to address biological questions and to compare quality of findings. Thus, in most cases more than a single data set is needed. Providing a way to store highly relevant data sets within the exploration interface could support compilation and enhance comparability across data sets that have been found along an extended exploration process. The idea is related to what Shneiderman called the *extraction* task (Shneiderman, 1996) and originates from the omnipresent shopping card that every online shop provides. For

example, similar to how users explore books, clothing and other things, clinical data scientists could collect patient data for studies or clinical trials. Another useful extension could be a simultaneous search across data sets and ontology terms. This would facilitate finding annotation terms of interest more efficiently.

Bibliography

Alper,B. *et al.* (2011) Design study of LineSets, a novel set visualization technique. *TVCG*, 17, 2259–2267.

Alsallakh,B. et al. (2014) Visualizing Sets and Set-typed Data: State-of-the-Art and Future Challenges. *The Eurographics Association*.

Andrews,K. *et al.* (2001) Search result visualisation with xFIND. *UIDIS-01*, 50–58.

Andrews,K. *et al.* (2004) The visualization of large hierarchical documents spaces with InfoSky. *Proceedings of CoData'04-Workshop on Information Visualization, Presentation, and Design.*

Anscombe,F.J. (2012) Graphs in Statistical Analysis. *The American Statistician*, 27, 17–21.

Archambault,D. *et al.* (2008) GrouseFlocks: Steerable Exploration of Graph Hierarchy Space. *IEEE Trans. Visual. Comput. Graphics*, 14, 900–913.

Ashburner,M. *et al.* (2000) Gene Ontology: tool for the unification of biology. *Nat. Genet.*, 25, 25–29.

Bederson,B.B. *et al.* (2002) Ordered and quantum treemaps: Making effective use of 2D space to display hierarchies. *TOG*, 21, 833–854.

Berners-Lee,T. and Hendler,J. (2001) The semantic web. *Scientific american, 284(5), 28-37.*

Bostock,M. *et al.* (2011) D^3: Data-Driven Documents. *TVCG*, 17, 2301–2309.

Brazma,A. et al. (2001) Minimum information about a microarray experiment (MIAME)-toward standards for microarray data. *Nat. Genet.*, 29, 365–371.

Brehmer,M. and Munzner,T. (2013) A multi-level typology of abstract visualization tasks. *TVCG*, 19, 2376–2385.

Brooks,J.L. (2014) Traditional and New Principles of Perceptual Grouping. In: Wagemans,J. (2015) Oxford Handbook of Perceptual Organization. *Oxford University Press*.

Bruls,M. *et al.* (2000) Squarified Treemaps. *VisSym*, 33–42.

Clarkson,E. *et al.* (2009) ResultMaps: Visualization for Search Interfaces. *IEEE Trans. Visual. Comput. Graphics*, 15, 1057–1064.

Collins,C. *et al.* (2009) Bubble sets: revealing set relations with isocontours over existing visualizations. *IEEE Trans. Visual. Comput. Graphics*, 15, 1009–1016.

Dinkla,K. *et al.* (2014) eXamine: exploring annotated modules in networks. *BMC Bioinformatics*, 15, 201.

Dinkla,K. *et al.* (2012) Kelp Diagrams: Point Set Membership Visualization. *Computer Graphics Forum*, 31, 875–884.

Fujibuchi,W. *et al.* (2007) CellMontage: similar expression profile search server. *Bioinformatics*, 23, 3103–3104.

Gehlenborg,N. (2010) Visualization and exploration of transcriptomics data.

Gehlenborg,N. and Wong,B. (2012) Points of view: Networks. *Nature Methods*, 9, 115–115.

Glimm,B. *et al.* (2014) HermiT: An OWL 2 Reasoner. *J Autom Reasoning*, 53, 245–269.

Glueck,M. *et al.* (2015) PhenoBlocks: Phenotype Comparison Visualizations. *IEEE Trans. Visual. Comput. Graphics*, 22, 101–110.

Gruber,T.R. (1993) A translation approach to portable ontology specifications. *Knowledge Acquisition*, 5, 199–220.

Gruber,T.R. (1995) Toward principles for the design of ontologies used for knowledge sharing? *International Journal of Human-Computer Studies*, 43, 907–928.

Hearst,M.A. (2009) Search User Interfaces. *Cambridge University Press*.

Hearst,M.A. (1995) TileBars: visualization of term distribution information in full text information access. *Proceedings of the SIGCHI conference on Human factors in computing systems*. ACM Press/Addison-Wesley Publishing Co.

Ho Sui,S. *et al.* (2013) The Stem Cell Commons: an exemplar for data integration in the biomedical domain driven by the ISA framework. *AMIA Jt Summits Transl Sci Proc*, 2013, 70.

Hoeber,O. and Yang,X.D. (2006) The Visual Exploration of Web Search Results Using HotMap. *Information Visualization, 2006.* IEEE, 157–165.

Horridge,M. and Bechhofer,S. (2011) The OWL API: A Java API for OWL ontologies. *Semantic Web*, 2, 1, 11-21.

Hotchkiss,G. *et al.* (2007) Search engine results: 2010. *Enquiro Research*.

Johnson,B. and Shneiderman,B. (1991) Tree-maps: a space-filling approach to the visualization of hierarchical information structures. *Visualization, 1991, Visualization'91, Proceedings., IEEE Conference on*. IEEE.

Kim,B. *et al.* (2007) Visualizing Set Concordance with Permutation Matrices and Fan Diagrams. *Interact Comput*, 19, 630–643.

Koboldt,D.C. *et al.* (2012) Comprehensive molecular portraits of human breast tumours. *Nature*, 490, 61–70.

Kolesnikov,N. *et al.* (2015) ArrayExpress update--simplifying data submissions. *Nucleic Acids Research*, 43, D1113–6.

Kuhlithau,C.C. (1991) Inside the search process: Information seeking from the user perspective. *JASIS*, 42, 5, 361-371.

Lex,A. *et al.* (2014) UpSet: Visualization of Intersecting Sets. *TVCG*, 20, 1983–1992.

Lonsdale,J. et al. (2013) The Genotype-Tissue Expression (GTEx) project. *Nature*, 45, 580–585.

Machlup,F. (1983) The study of information: Interdisciplinary messages. *Wiley*.

Malone,J. *et al.* (2010) Modeling sample variables with an Experimental Factor Ontology. *Bioinformatics*, 26, 1112–1118.

Manning,C.D. *et al.* (2008) Introduction to information retrieval. *Cambridge University Press*.

Marchionini,G. (1995) Information Seeking in Electronic Environments. *Cambridge University Press*.

Morville,P. (2005) Ambient findability: Libraries at the crossroads of ubiquitous computing and the internet. *O'Reilly Media, Inc.*

Munzner,T. (2014) Visualization Analysis and Design. CRC Press.

Oelke,D. *et al.* (2014) Comparative Exploration of Document Collections: a Visual Analytics Approach. *CGF*, 33, 201–210.

Palmer,S. and Rock,I. (1994) Rethinking perceptual organization: The role of uniform connectedness. *Psychonomic Bulletin & Review*, 1, 29–55.

Palmer,S.E. (1992) Common region: a new principle of perceptual grouping. *Cogn Psychol*, 24, 436–447.

Pickens,J. *et al.* (2008) Algorithmic mediation for collaborative exploratory search. *SIGIR*, 315–322.

Pirolli,P. and Card,S.K. (1995) Information Foraging in Information Access Environments. *CHI*, 51–58.

Pirolli,P. *et al.* (2000) The effect of information scent on searching information: visualizations of large tree structures ACM, New York, New York, USA.

Reiterer,H. *et al.* (2005) INSYDER: a content-based visual-information-seeking system for the Web. *Int J Digit Libr*, 5, 25–41.

Robinson,P.N. *et al.* (2008) The Human Phenotype Ontology: a tool for annotating and analyzing human hereditary disease. *Am. J. Hum. Genet.*, 83, 610–615.

Robinson,P.N. and Bauer,S. (2011) Introduction to bio-ontologies. *CRC Press*.

Rocca-Serra,P. *et al.* (2010) ISA software suite: supporting standards-compliant experimental annotation and enabling curation at the community level. *Bioinformatics*, 26, 2354–2356.

Rogers,F.B. (1963) Medical subject headings. *Bull Med Libr Assoc*, 51, 114–116.

Russell,D.M. *et al.* (1993) The cost structure of sensemaking. *INTERCHI*, 269–276.

Sadana,R. *et al.* (2014) OnSet: A Visualization Technique for Large-scale Binary Set Data. *IEEE Trans. Visual. Comput. Graphics*, 20, 1993–2002.

Schulz,H.J. (2011) Treevis.net: A Tree Visualization Reference. *IEEE Comput Graph Appl*, 31, 11–15.

Shneiderman,B. (1996) The eyes have it: a task by data type taxonomy for information visualizations. IEEE Comput. Soc. Press, pp. 336–343.

Smith,A. *et al.* (2014) Hierarchie: Visualization for Hierarchical Topic Models. Association for Computational Linguistics, Stroudsburg, PA, USA, pp. 71–78.

Smith,B. *et al.* (2007) The OBO Foundry: coordinated evolution of ontologies to support biomedical data integration. *Nature Biotechnology*, 25, 1251–1255.

Song,H. *et al.* (2010) A comparative evaluation on tree visualization methods for hierarchical structures with large fan-outs. *CHI*, 223–232.

Turnbull,D. and Berryman,J. (2016) Relevant Search. *Manning Publications*. (in press)

Vliegen,R. *et al.* (2006) Visualizing Business Data with Generalized Treemaps. *IEEE Trans. Visual. Comput. Graphics*, 12, 789–796.

Wertheimer,M. (1938) Laws of organization in perceptual forms. *A source book of Gestalt psychology*.

Wertheimer,M. (1923) Untersuchungen zur Lehre von der Gestalt. *Psychological Research*, 4, 1, 301-350.

Whetzel,P.L. *et al.* (2011) BioPortal: enhanced functionality via new Web services from the National Center for Biomedical Ontology to access and use ontologies in software applications. *Nucleic Acids Research*, 39, W541–5.

Wilson,T.D. (1981) On user studies and information needs. *Journal of Documentation*, 37, 3–15.

Zhang,J. and Marchionini,G. (2004) Coupling browse and search in highly interactive user interfaces: a study of the relation browser++ ACM, New York, New York, USA.

Zhao,S. *et al.* (2005) Elastic hierarchies: combining treemaps and node-link diagrams. *Information Visualization*, 2005. IEEE, pp. 57–64.