Faceted Visual Exploration of Semantic Data

Philipp Heim, Jürgen Ziegler

Faceted Visual Exploration of Semantic Data

Philipp Heim[1] and Jürgen Ziegler[2]

[1] Visualization and Interactive Systems Group, University of Stuttgart, Germany
philipp.heim@vis.uni-stuttgart.de
[2] Interactive Systems and Interaction Design, University of Duisburg-Essen, Germany
juergen.ziegler@uni-due.de

Abstract. Tools for searching and interactively exploring the rapidly growing amount of semantically annotated data on the Web are still scarce and limited in their support of the users' manifold goals and activities. In this paper we describe a method and a tool that allows humans to access and explore Semantic Web data more effectively, leveraging the specific characteristics of semantic data. The approach utilizes the concept of faceted search and combines it with a visualization that exploits the graph-based structure of linked semantic data. The facets are represented as nodes in a graph visualization and can be interactively added and removed by the users in order to produce individual search interfaces. This provides the possibility to generate interfaces with different levels of complexity that can search arbitrary domains accessible through the SPARQL query language. Even multiple and distantly connected facets can be integrated in the graph facilitating the access of information from different user-defined perspectives. This approach facilitates searching massive amounts of data with complex semantic relations, building highly complex search queries and supporting users who are not familiar with the Semantic Web.

Keywords: Graph visualization, faceted search, query building, SPARQL, hierarchical facets, pivot operation

1 Introduction

The amount of semantically described data available on the Web has been growing considerably in recent years, turning the vision of a Semantic Web as proposed more than a decade ago by Tim Berners-Lee [1] more and more into reality. The growing volume of information in the Semantic Web can be seen in large information pools such as *DBpedia* [2] or in the *LOD initiative* (Linked Open Data [3]) which aims at semantically linking a large number of different open, publicly available data sets. In these datasets information is represented by semantic structures expressed in formal languages such as *RDF* (Resource Description Framework) or *OWL* (Web Ontology Language). The structures consist of statements about real world (e.g. 'Berlin') or virtual objects referenced by a *URI* (Uniform Resource Identifier), an assigned ontological class, such as 'city' or 'country', and an arbitrary number of properties that define links to other

objects, such as 'is capital of'. These techniques allow to search for information based on their semantic descriptions and to make data interoperable between different datasets by linking objects that have the same meaning.

The predominant view on the purposes of a Semantic Web was, and still is to enable machines to process distributed data on the Web more effectively. In recent years, however, there has been an increasing focus on the interactive use of semantic data. The Semantic Web can also help users directly to answer focused questions and to retrieve unknown factual information. In contrast to the conventional Web, however, there are a number of properties of semantic data that may impede human use unless appropriate support for searching and exploring the data is provided.

Conventional Web pages are predominantly structured and designed as coherent documents, intended and suitable for human consumption and perusal. Information units are embedded in other textual or multimedia content, providing context for clarification and a better understanding. Semantic data, on the other hand, are represented in the form of single subject-predicate-object statements and thus constitute more elementary pieces of information than standard Web pages. While the information contained in a single statement may indeed be the target of a focused search, for exploring and understanding the data, more context is usually required. The semantic relations between individual resources can be exploited to create such context. Since a pool of semantic data forms a large RDF graph, a context of interest can essentially by of arbitrary size. Users should therefore be able to define and visualize the context of some focal concept in a flexible and user-controlled fashion.

For this purpose, suitable tools are needed that support users throughout the information seeking process starting from an initial query formulation over the flexible selection of relevant context through to the identification and use of the information needed. One of the objectives of this paper is to propose a conceptual search model that is appropriate for describing the different user activities involved in searching and exploring semantic data. Since the information space to be searched can be very large, such a process typically requires the use of search and exploration activities in combination. Even if the target of the search is not well-defined at the outset, users will have to define a starting point for follow-up explorations based on some textual query. For experts, artificial query languages such as *SPARQL*[3] are available that allow to formulate precise queries on the basis of a well defined information need. For non-experts, free text search is more appropriate for initiating the search process. In contrast to Web search, however, the result is not comprised of Web pages but can be concepts that match the query terms. In the following stages, user activities are typically more of an exploratory nature, iteratively refining or modifying the search until the desired information is found.

A promising approach for supporting exploratory search processes is faceted search [4]. In faceted search, the search space is partitioned using orthogonal conceptual dimensions based on the properties of some set of objects and their

[3] http://www.w3.org/TR/rdf-sparql-query/

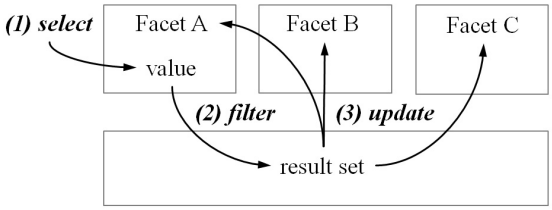


Fig. 1: In faceted search, selecting a value in a facet (1) filters the result set (2), with subsequent updating of the other facets (3).

corresponding values which serve as independent filters for narrowing down a result set. Selecting a value of a facet adds this value to the query and filters the result set accordingly. Whenever the result set changes, all facets are updated to show only those attributes that can be used for further filtering (Fig. 1). Therefore, users always see only those values that can further constrain the result, but which do not result in an empty set. While faceted search was originally developed for searching relational databases, the concept can be extended to semantic data, exploiting the well-defined classes and properties of semantic data for creating facets.

In a domain such as 'football', for example, a faceted search would allow football players to be filtered by their clubs, their birthplaces, or their ages. Even so the concept of faceted search has been successfully applied in popular applications such as *Apple's iTunes*[4], using it to seek information in the Semantic Web entails several new problems that are not yet sufficiently solved. Most of the problems result from the vast number of objects, classes and properties. According to [5], the LOD cloud alone contained approximately two billion statements in 2008 already.

In this paper we introduce *Facet Graphs*, a new approach based on the concept of faceted search which allows the intuitive formulation of search queries for semantic data. Facet graphs are composed of set-valued nodes and labeled semantic relations between these nodes. Users can choose a node as the desired result type as well as related nodes that serve as facets for filtering the results, thereby producing a personalized search interface. The graph provides a coherent representation of multiple, even distantly connected facets on a single page, avoiding browsing and preventing users from losing track of the relationships. Each node in a facet graph contains a list of items that provides sorting, paging and scrolling functionalities and thereby enables an easy handling of even large amounts of objects. Users can build queries to filter the result list by just clicking on items in the facet nodes. Filtering is indicated by colored highlighting around the nodes affected, helping users to understand and keep track of the filtering constraints applied. Users can also at any time change their search perspective by turning results nodes into facets of related nodes.

[4] http://www.apple.com/de/itunes/

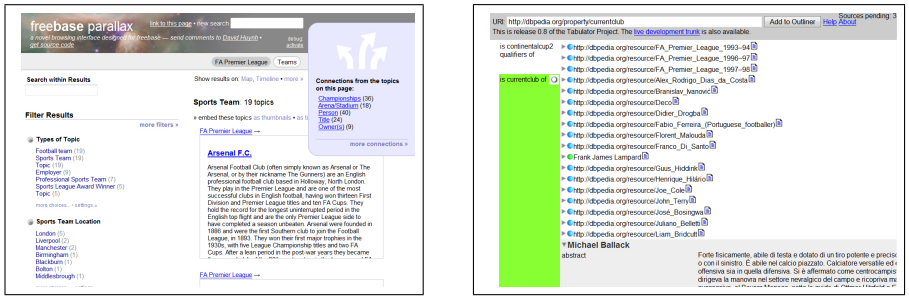
The combination of a graph visualization with faceted search allows for an easy formulation of even complex, semantically unique search queries and thus for seeking semantically annotated information without expert knowledge.

The rest of this paper is organized as follows. First we review related work and motivate our work based on shortcomings of the existing approaches. In section 3, we propose a general information seeking model for searching and exploring semantic data. We then describe the approach of Facet Graphs and the tool gFacet in detail, and evaluate its usability in a comparative user study. In the discussion, we relate our findings to the proposed search model and analyze Facet Graphs in terms of general requirements for information search. Finally, we present a conclusion and an outlook on future work.

2 Related Work

Several approaches using faceted search have been reported in the literature, both for relational and semantic data. Most approaches display the facets as well as the result set as lists at different positions on the screen. Examples are tools like *mSpace* [6], *Flamenco* [4], *Longwell* [7] and *Haystack* [8]. In most cases, only facets directly connected to the result set are used. For example, football players could be filtered by facets containing the clubs they are playing for, the city they are living in or the number on their shirts. By selecting, for example, the football club 'FC Chelsea' in a 'club' facet, football players would be filtered to only those players employed by this club. Hierarchical filtering, however, that allows also indirectly connected facets to be used for query building (e.g. get only players playing for clubs in the Premier League) is not supported by these tools.

Tools like *Parallax* [9] (Fig. 2a), *Humboldt* [10], *Tabulator* [11] (Fig. 2b) and the *Nested Faceted Browser* [12], by comparison, allow for hierarchical filtering. Therefore, football players could be filtered by the cities where clubs they are playing for are located. Thus the possible options for building a query are not restricted to the direct periphery around a result set but can include dimensions



(a) (b)

Fig. 2: a) Screenshots of Parallax and b) Tabulator.

that are distantly connected by other facets. Depending on the directness of their connections to the current result set, they can be integrated in a hierarchy and are therefore called *hierarchical facets*.

In Parallax, Flamenco and Humboldt, the hierarchy of facets is never completely visible to the user. By providing ways to browse through the facet hierarchy, users are able to include attributes of also distantly connected facets in their search query. For complex queries, it is getting more difficult for users to keep an overview of all included attributes since the corresponding facets are often scattered over several different pages.

Tools like Tabulator and the Nested Faceted Browser, on the other hand, allow the whole hierarchy to be displayed on one page. By using a tree structure to display all available hierarchical facets, users can open and close distantly connected facets in one coherent view and thus keep an overview of all attributes that are included in their queries. Since tree structures are used to depict all kinds of taxonomies in a wide range of well known applications, users need no extensive period of training to gain an understanding of how to use them.

In Tabulator and the Nested Faceted Browser, every attribute defines its own subtree that can be expanded by the user in order to see distantly connected facets and their attributes, which again can be expanded and so on (cp. Fig. 2b). So for the football players, one of the clubs they are playing for (e.g. 'FC Chelsea'), could be expanded by the user to see, for example, the city of its location (here: 'London'). Selecting 'London' would cause the list of players to be filtered to only those that are playing for 'FC Chelsea' or any other club located in this city. A combined list of all the cities that could be selected in order to filter the clubs and also the players, however, is not provided by the tree structure. The cities are partitioned in different subtrees, each for every club, that all need to be expanded in order to see all available cities. Having many subtrees opened, however, places attributes that actually belong to one facet at many different positions in the tree, leading to an increased tree structure that can possibly not be displayed on one screen. If an attribute is shared by more than one object (e.g. many clubs are located in London), it also occurs repeatedly in different subtrees.

Tree structures are well suited for visualizing and interacting with hierarchical data; however, when used for hierarchical facets, they tend to produce large and highly subdivided structures that cannot easily be overviewed by users. In this paper we therefore propose an alternative, graph-based approach to visualize and interact with hierarchical facets that aims at preventing large and confusing tree structures and hence facilitates an easy generation of semantically unique queries by the user.

3 A Three-Stage Model for Searching and Exploring Semantic Data

Searching and exploring linked semantic data differs in a number of aspects from conventional search paradigms as proposed, for example, in the area of informa-

tion retrieval. Conventional information search is usually either completely based on free-form textual search or uses a limited well-defined set of metadata. Free text search is the main paradigm in Web search while metadata-based search is a predominant model in more structured domains such as digital library search. Semantic data, on the other hand, are more structured than free text documents but may contain an arbitrary number of semantic relations which makes it hard or, in the case of large or open corpora, even impossible to build up a complete and coherent cognitive model of the underlying structure. This latter aspects also distinguishes semantic data search from querying conventional relational databases. A further important distinction refers to the granularity of the search results. While conventional search aims at retrieving a set of relevant information ressources at the document level, the size of a single chunk of semantic data is typically much smaller, containing individual factual statements describing relations between elementary units of information.

A variety of models have been proposed for describing user behavior when searching and exploring collections of information. Classical models from information retrieval focus either on the sequential process from the formation of the search goal to the use of the results or analyse the different levels of abstraction in the search process [17, 18]. Kuhlthau, for example, describes the information seeking process from a user perspective in six stages: task initiation, selection, exploration, focus formulation, collection and presentation [16].

While these models have demonstrated their usefulness in general information retrieval, they do not take into account the specific properties of semantic data which can be assumed to have a significant influence on the cognitive tasks and search strategies of the user. When searching semantic data, users can make use of the explicit semantic relations which link information instances among each other as well as with the conceptual structure of the dataset expressed in terms of ontologies describing the domain(s) at hand. The search process can therefore be separated in actions selecting the concept of interest and actions that explore information at the instance level. Once a conceptual starting point for the search has been determined, the semantic relations can be used as 'guiding rails' for a systematic exploration of the environment of the initial concept. Based on these observations, we can identify a model consisting of three stages of searching and exploring semantic data which forms the conceptual basis for the gFacet tool described in the next section.

1. **Goal formation and concept search:** In the initial phase, users form the search goal which can be specified at very different levels of concreteness. For explorative tasks, this goal will typically be at the concept level, specifying one or more concepts that determine the initial search space the user is interested in. Since the number of concepts in a datapool or linked data is far too high to start the search with a visual exploration, this goal (for instance 'find out more about German footballers' in our running example) is translated into textual search terms (e.g. 'football' that are used for identifying a relevant start concept 'e.g. German football clubs'. This concept (or class)

along with its instances, i.e. information elements of this type, constitute a point of departure for the explorative activities in the second stage.

2. **Construction of the exploration space:** At this stage, the user incrementally builds up a space for exploring the relevant semantic neighborhood of the starting concept. For this purpose, she can exploit the different semantic relations (object properties) linked to the starting concept. By incrementally adding relations and the instance sets (ranges) linked through these relations, users can flexibly construct subspaces of the entire dataset that are relevant to their information goal . Since these relations are independent of each other, they can be used for constructing independent facets and chains of facets. These facets can be used for providing visual tools that facilitate constructing and exploring the space.
3. **Multi-perspective exploration and sensemaking:** Once a relevant exploration space has been constructed by the user it can be used in various ways to explore it and to make sense of the information and relations contained in it. The semantic relations are a valuable asset to support the sensemaking process. First of all, they can help to filter the information by using related concepts as facets to narrow down the number of items of a target concept. Since semantic relations can be viewed in both directions by exchanging the roles of facet and target concepts, they support changes of perspective that are important for getting a better and more comprehensive understanding of the data and their relationships. At any time in this process, the user may choose to further expand or prune the exploration space when new information of interest is found. Stages 2 and 3 of this model are therefore closely intertwined and may be executed in an iterative fashion.

This model of searching and exploring semantic data forms the conceptual basis of the method and tool described in the next section. While other search and exploration strategies are possible and may be supported by other types of tools, the model focuses on the linking aspect of semantic data and provides a systematic method for exploring datasets the structure and content of which are largely unknown to the user. We therefore postulate that the model is particularly suited to searching open linked data due to their enormous volume and complex link structure for which users will only be able to build up partial mental models.

4 Visualizing Semantic Data with gFacet

In Facet Graphs, facets and result set are represented as nodes in a graph visualization (see also [19]). The semantic relations that exist between facets and result set as well as facets and other facets are represented by labeled directed edges between the nodes. Fig. 3 shows *gFacet*[5], a prototypical implementation of our approach of Facet Graphs.

[5] A description of an early version of this prototype with limited functionalities and access to dummy data only can be found in [13].

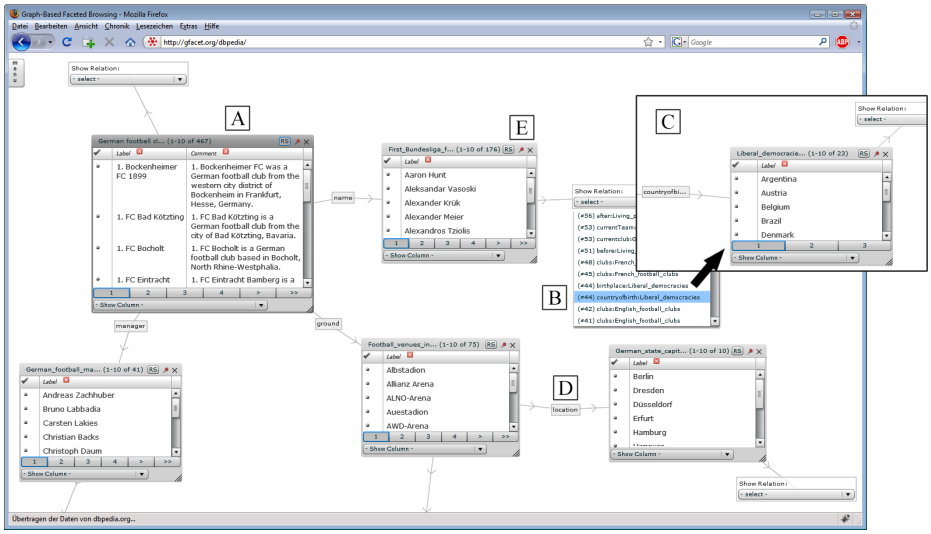


Fig. 3: In gFacet, the result set (A) and the facets are represented by nodes and connected by labeled edges (D). By selecting properties out of drop-down-lists (B), further facets can be added as new nodes to the graph (C).

In gFacet, the node representing the result set is marked by a darker background color (cp. Fig. 3, A) to better distinguish it from the nodes representing facets. The attributes of each facet are listed within one single node, which can be scrolled, paged and sorted to handle large sets of facet values. The graph can be interactively expanded by adding nodes that serve as additional facets. Facets can be added via a drop-down list from which related concepts can be selected (e.g. 'First Bundesliga Footballers' in Fig. 3, B). Facets in the drop-down lists are ordered by the number of values. Selecting a facet (e.g. the one containing the countries where footballers are born), adds it to the graph and connects it to the existing node by a labeled edge (cp. Fig. 3, C). In this way, the user can iteratively construct a graph of facets that serves as an exploration space and fosters the user's understanding of the relationships among items.

The nodes in the graph are positioned in an aesthetically pleasing way with a force-directed layout algorithm [14]. To prevent nodes from changing their position in the visualization too frequently, we apply a pinning mechanism that forces nodes to hold their position. When a new node is added, pinning is executed after a short time delay. This delay allows the force-directed algorithm to position new nodes in an appropriate way and at the same time prevents already existing information to change their location. Pinned nodes are indicated by a pin symbol in the upper right corner (cp. Fig. 3, E) which can also be used to control pinning manually.

The general benefits of representing hierarchical facets as nodes in a graph are:

1. The attributes for each facet are grouped into one node.
2. All nodes are shown in a coherent presentation on one page.
3. Semantic relations between the nodes are represented by labeled edges (Fig. 3, D).
4. Facets can be flexibly added and removed by the user (Fig. 3, C).

4.1 Extracting Facets and Searching for a Start Concept

To make facets selectable by the user, they first have to be extracted from the underlying data structure. In gFacet, we build SPARQL queries on the client side, send them to SPARQL endpoints using HTTP requests and extract the facets from the resulting XML data. The client-server communication as well as the graphical user interface are implemented in *Adobe Flex*[6] and compiled to a Flash movie, which runs in all Web browsers with Flash Player installed. An implementation of gFacet that uses DBpedia's SPARQL endpoint can be accessed online[7]. Since gFacet does not need any modification on the server side, it can be used to access information from other SPARQL endpoints as well.

Extracting facets from semantic structures is performed on the basis of semantic links. The links are defined by properties like 'plays for' and connect objects such as 'Franck Ribry' with, for example, 'FC Bayern Munich'. So given a list of football players, a possible facet to filter this list would be the football clubs at least one of the players in the list are playing for. Links from facet concepts to other concepts allow to construct facets hierarchically.

Since the number of concepts or classes as well as relations in a semantic data set is typically very large and cannot be shown simultaneously, users first have to identify a start concept or ontological class which is shown as the first node in the visualization. This step is done by entering textual search terms (e.g. 'german football' in Fig. 4). While entering terms, matching classes are immediately shown in a selection list.

The corresponding SPARQL query, which returns all the classes with labels containing the words entered by the users (here: "german football"), is given in the following:

```
SELECT DISTINCT ?class ?label COUNT(?o) AS ?numOfObj
WHERE { ?class rdf:type skos:Concept .
?o skos:subject ?class .
?class rdfs:label ?label .
?label bif:contains "german and football" .
FILTER (lang(?label) = "en") }
ORDER BY DESC(?numOfObj) LIMIT 30
```

The found classes are initially ordered according to the numbers of objects contained in each class. The class selected by the user becomes the first node in

[6] http://www.adobe.com/products/flex
[7] http://gfacet.semanticweb.org.

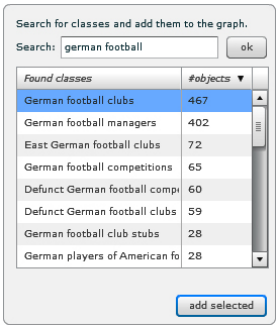


Fig. 4: List of suggested classes to define the initial search space based on user input.

the graph and also the current result set (e.g. 'German football clubs'), listing all the objects of the selected class (cp. Fig. 3, A).

Based on the properties of the objects in the result set, available facets are extracted automatically and displayed in a drop-down next to the result set. Each facet in this list consists of a property (e.g. 'ground') and a class of objects this property leads to (e.g. 'Football venues in Germany'). It is thus possible to have several facets with the same property but different target classes, or with different properties but the same target class. The corresponding query looks as follows:

```
SELECT DISTINCT ?prop ?newClass
COUNT(DISTINCT ?objNewClass) AS ?numOfObj
WHERE { ?objCurrClass skos:subject <URIofGermanFootbalClubs> .
?objCurrClass ?prop ?objNewClass .
?objNewClass skos:subject ?newClass .
?newClass rdf:type skos:Concept .}
ORDER BY DESC(?objNewClass) ?prop ?newClass LIMIT 40
```

Similarly to adding first order facets to the result set, also facets of second or higher order can be added to the graph and used for filtering. In higher order facets, only those objects are shown that are connected to objects in other visible facets that are in turn directly or indirectly connected to objects in the result set. Fig. 5 shows a result set on the left, a first order facet in the middle and a second order facet on the right together with a schematic representation of the visible (black) and invisible (gray) objects. The gray dots in the second order facet represent objects of that class which are not connected to any object in the result set and therefore not visible. In this way, all the facets in the graph only contain objects that can be used for filtering but will never result in an empty result set.

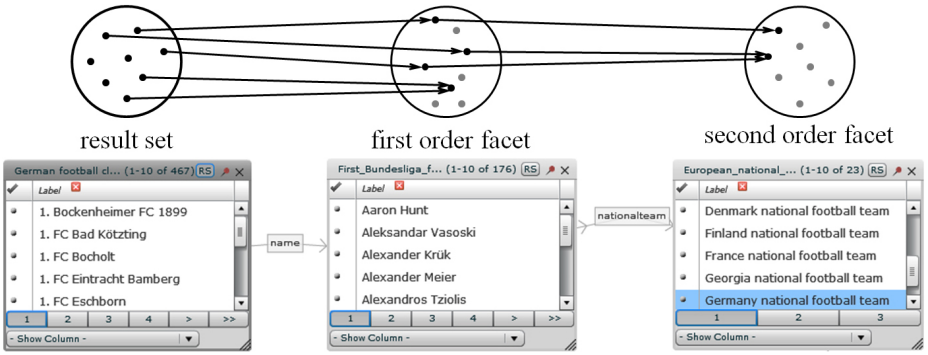


Fig. 5: Only objects that are connected to the result set are visible in the hierarchical facets: Here the national teams for which players of German clubs are playing.

4.2 Exploring the Search Space

After selecting at least one facet node in addition to the result node, it can be used to build semantically unique search queries. By selecting one of the objects in a facet, the result set is filtered to only objects that are directly or indirectly connected to the selected one. For example, the selection of 'Germany national football team' in the second order facet in Fig. 5 filters the football clubs to only those with players playing for this national team (see the new result set in Fig. 6).

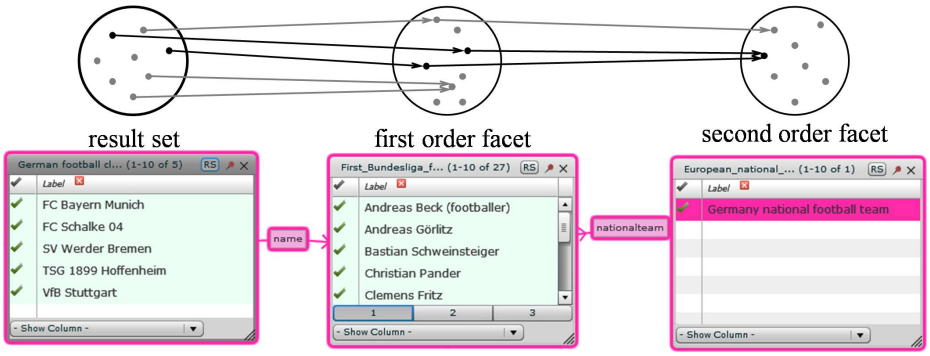


Fig. 6: Hierarchical facets can be used for filtering.

To support traceability and sensemaking, we color-code the border of each node according to the current filters in effect for this node. If a facet object is selected, the selected object itself, all the filtered nodes and all the edges between them are marked with the same color (Fig. 6). When several filters are in effect

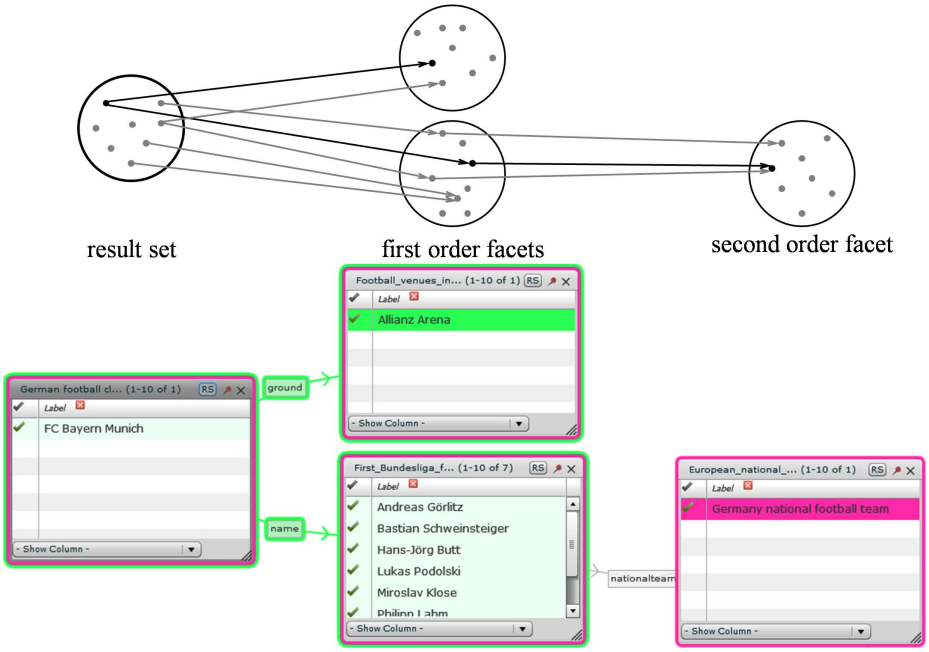


Fig. 7: Multiple selections.

for a node, several colored borders corresponding to the filters are shown. Nodes not affected by the filtering keep the normal border. The color coding provides an effective means to detect and understand the active restrictions on a node as well as the facet selections causing these restrictions.

As shown in Fig. 7, filtered nodes are surrounded by colored borders for each selection that affects them. New borders are added to a node like tree-rings when new filters come into effect. Deselecting the facet object that constrains the result also removes the corresponding border. In the current implementation, multiple filter selections represent a logical 'AND', constraining the result set to only those objects that are connected to all selections. Other logical operations would be possible, but have not yet been investigated for the current system.

4.3 Pivot Operation

While adding facets to the graph or filtering the result set in various ways, users may at any time change their minds about what they are interested in. gFacet supports such situated, exploratory behavior. When, for example, searching for 'German football clubs', users could become more interested in 'First Bundesliga footballers' and would like to explore which clubs they are playing for. They therefore might want to use clubs as a filter for footballers instead of the other way round. This operation is supported by a *pivot operation* which allows to

change the perspective and direction of the filtering process. The pivot operation is known from the field of data drilling [15], where data are represented according to different dimensions.

The Nested Faceted Browser [12] and Humboldt [10] are the only tools mentioned in the related work section that allow users to perform a pivot operation and thus to change the focus of their search. In Humboldt, the user can replace the current result set displayed in the centre of the screen, by one of its visible facets, which are arranged on the right-hand side of the display. The chosen facet becomes the new result set and vice versa, the replaced result set becomes a new facet. In Humboldt, only directly related facets are shown next to the result set and thus hierarchical facets can only be reached by operating pivots. Whenever a pivot is operated and a new result set is shown in the centre of the screen, the list of directly related facets is updated accordingly. This way, information is partitioned over multiple pages, placing substantial cognitive load on the users to keep track of former result sets and facets, which are not visible yet.

Our graph-based approach, by contrast, allows a pivot operation to be performed without any changes of the displayed information structure and thus reduces the cognitive load to keep track of them. Clicking the 'RS'-Button (RS = result set) next to the pinning needle of any of the turns this facet into the new result set and vice versa the current result set into a new facet. Whereas other approaches have to rebuild their complete layout to keep the displayed facets up to date, in our approach even distantly related facets will keep their position while operating pivots. The only aspect that changes in gFacet when operating a pivot is the number of objects in both the result set and the visible facets.

5 User study

To evaluate the usability of our approach, we conducted an initial user study that compared gFacet with Parallax [9], a tool that also uses a facetted approach to searching semantic data but which is not based on a graph visualization. For both tools we measured to what extent participants were able to solve the following three task types of different levels of difficulty:

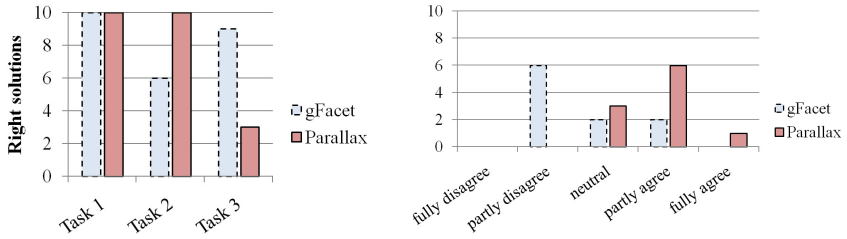
1. Find two players who are playing for a certain club.
2. Find two cities where players who are playing for a certain club are born.
3. Find one player who is playing for a certain club and is born in a certain city.

We applied a 2x3 (*tool type x task type*) within-subject design. To control for learning effects, each participant was assigned to one of two groups which used the two tools in reversed order.. After completing all three tasks with one tool, the participants were asked to fill out an evaluation sheet to rate this tool. In a final questionnaire, participants had to directly compare both tool types with each other.

Ten participants took part in the study, with an average age of 28.3 (ranging from 24 to 31). Eight of them were male; two were female with all ten participants having normal or corrected to normal vision and no color blindness. Education level of the participants was at least general qualification for university entrance and they were all familiar with computers. The functionalities of both, gFacet and Parallax were introduced by videos in which sample tasks were solved.

5.1 Results

Overall, gFacet performed very well for complex tasks. However, it performed less well for more simple tasks that could also be accomplished by just following links.



(a) (b)

Fig. 8: a) Number of right solutions to the given tasks and b) comments to the statements 'It was difficult to understand the relations between the information'.

The bar chart in Fig. 8a shows the number of correct solutions for each of the three different task types using the two different tool types. For task 1 and task 2, Parallax performed equally or even better than gFacet. This was mainly because in Parallax participants could accomplish both tasks just by following links and did not have to filter at all. In gFacet, by contrast, they had to filter despite the simplicity of the first two tasks and hence did not accomplish task 2 in four cases. The unfamiliar approach of graph-based facets seemed partly too complex to be properly used in average exploratory tasks. However, in order to find the right solution for more complex tasks, as for example for task 3, gFacet performed significantly better than Parallax. It was almost impossible so to find the right solution for task 3 just by following links in Parallax. All the links had to be checked in a trial-and-error method to actually find the player that is born in the given city. Since this is quite time-consuming, participants were forced to apply filters in Parallax as well.

To find the right solution to task 3, participants had to filter the list of all football players to only players that are playing for a certain club and are born in a certain city. Even though Parallax generally supports applying multiple filters, participants felt uncertain about the solutions they found. A reason for this is

the strict separation in Parallax between exploration, which can be realized by clicking links on the right side of the content, and filtering, which can be done by clicking links on the left side (cp. Fig. 2a). If users explore a list of objects (e.g. cities where players are born) these objects cannot be used for filtering.

On the other hand, if users look at the filter options on the left side, they cannot further explore these options for hierarchical filters. To filter hierarchically, users first have to explore objects via links on the right side (e.g. cities where players are born) and then have to filter this list via links on the left side (e.g. to only cities located in Germany). Having to change from one side to the other can confuse users and thus can decrease their confidence in the found solutions. Moreover, clicking links to explore objects completely replaces the current content as well as the links at both sides. That way information gets partitioned over multiple pages and thus hampers users' understanding of the relations that exist between information (cp. bar chart in Fig. 8b). In gFacet, by contrast, all objects in all lists can be used for both filtering and the exploration of further facets and are all visible on one page.

6 Comparison of Tools Based on the Information Seeking Process

To discuss the advantages and limitations of our approach and to compare it with existing techniques, we use the general information seeking process (*ISP*) as described in [16] and determine to what extent existing tools support the different activities in the process. The ISP can be described in six stages: task initiation, selection, exploration, focus formulation, collection and presentation [16].. In the following, we propose a number of concrete requirements for each of these stages and analyze whether the tools fulfill the requirements (Fig. 9).

The first stage of the ISP, the *task initiation*, begins with awareness of lack of knowledge and leads to a concrete definition of the problem and its relation to prior experience and knowledge. Requirements are: A continuous support of the whole ISP including the problem definition (R1.1) and, based on this definition, suggestions on how to address the information need considering previously operated ISPs (R1.2).

The next stage, the *selection*, includes tasks like the identification and selection of the general topic to be investigated. Requirements are: An overview of all available topics (R2.1), for instance in form of a map that can be zoomed in and out, together with automatic suggestions of topics (R2.2) based on the entered problem description, entered keywords or an auto complete functionality. Both requirements aim at lowering the barrier to start an ISP.

The selection is followed by the *exploration* stage. It includes the investigation of the general topic, locating information and relating it to what is already known. Requirements are: A graphical representation of information that can be understood by the user (R3.1), interaction possibilities that are self-explanatory and easy to use (R3.2), the accessibility of details on demand (R3.3), sorting and

ISP Stages	Task initiation		Selec -tion		Exploration					Focus formulation						Collec -tion		Presenta -tion
Require-ments	R1.1	R1.2	R2.1	R2.2	R3.1	R3.2	R3.3	R3.4	R3.5	R4.1	R4.2	R4.3	R4.4	R4.5	R4.6	R5.1	R5.2	R6.1
mSpace	-	-	-	-	+	+	-	-	-	+	+	+	-	-	-	-	-	-
Humboldt	-	-	-	-	/	-	-	-	-	+	+	+	+	-	+	-	-	-
Parallax	-	-	-	+	+	-	+	+	-	+	+	+	+	-	-	-	-	-
Tabulator	-	-	-	+	/	-	+	+	-	-	-	+	+	-	-	-	-	+
gFacet	-	-	-	+	+	+	+	+	-	+	+	+	+	+	+	-	-	-

Fig. 9: Comparison of existing systems with respect to requirements supporting a six-stage information seeking process.

paging techniques to handle large datasets (R3.4) and zooming functionalities that are capable of showing information in different levels of detail (R3.5).

The focus *formulation* includes tasks like the identification and the selection of hypotheses that result in the formulation of certain filters and thus allows a focused perspective of the topic. It is rather an iterative process than one that is strictly linear. Requirements are: The interactive and intuitive formulation and change of filters (R4.1), their immediate execution on the data (R4.2), the combination of different filters (R4.3), the possibility to build hierarchical filters (R4.4), the traceability of effects caused by each of them (R4.5) and a possibility to change the focus (R4.6).

After the focus formulation, the *collection* takes place. Tasks are to gather and select information related to the focused topic. Requirements are: Easy mechanisms to select interesting findings (R5.1) and to export the selected information for further use in other systems (R5.2).

The last stage of the ISP, the *presentation*, consists of the task to present the found information. Requirements are: A broad range of opportunities to visualize the findings (R6.1).

From the pattern shown in Fig. 9 it becomes obvious that none of the tools supports the initial and the final stages of the ISP. gFacet apparently fulfills more requirements than any other tool. However, R3.5 is also not fulfilled by any of the tools. This is potentially problematic for the Facet Graph approach since extending the amount of information shown in the graph would result in very large images that are hard to overlook and handle. Introducing appropriate zooming techniques along with a better support for the initial and final stages are therefore possible extensions of the tool.

6.1 Conclusion and Future Work

In this paper we described Facet Graphs, a new approach for building semantically unique queries based on the concept of faceted search in combination with

graph visualization. In addition to the general advantages of faceted search, the visualization of facets as nodes in a graph allows the direct representation of relationships between nodes by labeled edges and thus a connected presentation of the result set together with all relevant facets on one page. The user can add and remove facets to the graph to produce a personalized interface including even distantly connected and multiple facets that can be used to filter the result set from different user-defined perspectives. All caused filtering effects are color-coded in the graph making them better understandable and traceable for users.

We introduced gFacet, a prototypical implementation of our approach that can query arbitrary SPARQL endpoints (e.g. DBpedia) to access information according to its semantics. We conducted a user study to compare gFacet with Parallax and found out that our tool is especially applicable in scenarios where multiple aspects from different domains need to be integrated in order to find certain information. Such scenarios seem to be particularly interesting for querying the Semantic Web because of its huge variety of domains with each containing large amounts of different classes, objects and properties. This opens up new and outstanding opportunities for users to access information; however, controlling such powerful opportunities remains a challenging task.

Future work includes the integration of appropriate zooming functionalities in combination with a focus and context technique to foster users to retain an overview even when using massive amounts of facets in one graph. Another interesting idea is to provide an opportunity to save especially helpful combinations of facets and share such search interfaces with other users. This would further lower the barrier of acceptance for using gFacet since users can load existing search interfaces that are built by more experienced users and thus do not need to start from scratch.

References

1. Berners-Lee, T. and Fischetti, M.: Weaving the Web: The Original Design and Ultimate Destiny of the World Wide Web by its Inventor. Harper, USA (1999)
2. Auer, S., Bizer, C., Kobilarov, G., Lehmann, J., Cyganiak, R. and Ives, Z.: DBpedia: A nucleus for a Web of open data. In Proc. ISWC 2008, LNCS, pp. 722–735, Springer, Heidelberg (2008)
3. Bizer, C., Heath, T., Kingsley, I. and Berners-Lee, T.: Linked data on the Web. In Proc. WWW 2008 Workshop: LDOW, (2008)
4. Hearst, M., English, J., Sinha, R., Swearingen, K. and Yee, P.: Finding the Flow in Web Site Search. Communications of the ACM, 45 (9), pp. 42–49, (2002)
5. Hausenblas, M., Halb, W., Raimond, Y. and Heath, T.: What is the size of the Semantic Web? In Proc. I-SEMANTICS 08, pp. 9–16, JUCS (2008)
6. Schraefel, m.c., Smith, D., Owens, A., Russell, A., Harris, C. and Wilson, M.: The evolving mSpace platform: Leveraging the Semantic Web on the trail of the memex. In Proc. Hypertext 2005, pp. 174–183, ACM Press (2005)
7. Longwell RDF Browser, SIMILE (2005). http://simile.mit.edu/longwell/.
8. Quan, D., Huynh, D. and Karger, D. Haystack: A Platform for Authoring End User Semantic Web Applications. In Proc. ISWC 2003, LNCS, pp. 738–753, Springer, Heidelberg (2003)

9. Huynh, D. and Karger, D.: Parallax and companion: Set-based browsing for the Data Web (2009)
10. Kobilarov, G. and Dickinson I.: Humboldt: Exploring Linked Data. In: Proc. WWW 2008 Workshop: LDOW, (2008)
11. Berners-Lee, T., Hollenbach, J., Lu, K., Presbrey, J., Prud'ommeaux, E. and Schraefel, m.c.: Tabulator Redux: Browsing and writing Linked Data. In Proc. WWW 2008 Workshop: LDOW, (2008)
12. Huynh D.: Nested Faceted Browser (2009). http://people.csail.mit.edu/dfhuynh/projects/nfb/.
13. Heim, P., Ziegler, J. and Lohmann, S.: gFacet: A Browser for the Web of Data. In Proc. SAMT 2008 Workshop: IMC-SSW, , pp. 49–58, CEUR-WS (2008)
14. Fruchterman, T. and Reingold, E.: Graph drawing by force-directed placement. In: Softw. Pract. Exper. 1991, pp. 1129–1164, John Wiley & Sons (1991)
15. Gray, J., Bosworth, A., Layman, A. and Pirahesh, H.: Data cube: A relational aggregation operator generalizing group-by, cross-tab, and sub-totals. In: Proc. ICDE 1996, pp. 152–159, IEEE Press, New York (1996)
16. Kuhlthau, C.C.: Developing a model of the library search process: cognitive and affective aspects, pp. 232–242, Reference Quarterly (1988)
17. Marchionini, G.: Information seeking in electronic environments. Cambridge, New York: Cambridge University Press (1997)
18. Bates, M. J.: Where should the person stop and the information search interface start? Information Processing and Management, 26(5), pp. 575–591, (1990)
19. Heim, P., Ertl, T. and Ziegler, J.: Facet Graphs: Complex Semantic Querying Made Easy. In: Proc. 7th Extended Semantic Web Conference (ESWC 2010), LNCS 6088, pp. 288–302, Springer, Heidelberg (2010)