UNIVERSITATEA POLITEHNICA BUCUREŞTI
FACULTATEA DE AUTOMATICĂ ŞI CALCULATOARE
DEPARTAMENTUL CALCULATOARE

LUCRARE DE LICENŢĂ

Căutare adaptivă folosind fațete pentru magazine online

Coordonatori ştiinţifici:
Prof. dr. ing. Valentin Cristea
Prof. dr. ing. Paul-Alexandru Chiriță

Absolvent:
Mariana-Codruța Bădună

BUCUREŞTI
2013

UNIVERSITY POLITECHNICA OF BUCHAREST
FACULTY OF AUTOMATIC CONTROL AND COMPUTERS
COMPUTER SCIENCE DEPARTMENT

DIPLOMA PROJECT
Adaptive Faceted Search for Online Stores

Thesis supervisors:
Prof. dr. eng. Valentin Cristea
Prof. dr. eng. Paul-Alexandru Chirita

Student:
Mariana-Codruta Baduna

BUCHAREST
2013

Table of Contents

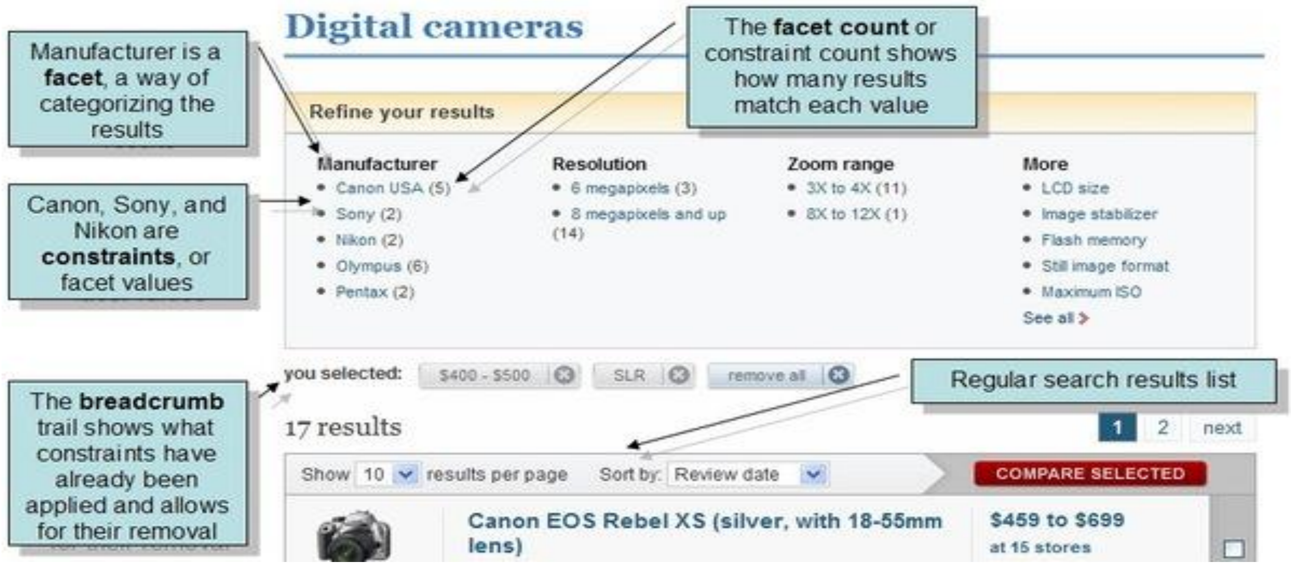
1. Introduction

1.1 Definition

According to Wikipedia, faceted search “is a technique for accessing information organized according to a faceted classification system, allowing users to explore a collection of information by applying multiple filters. A faceted classification system classifies each information element along multiple explicit dimensions, enabling the classifications to be accessed and ordered in multiple ways rather than in a single, pre-determined, taxonomic order”.

William Denton mentions in his article “How to make a faceted classification and put it on the web” [2] that facets are “a natural way of organizing things” and they come as a response to the question “In what ways can people see this data?”.

Let’s take a simple example to better understand what a faceted classification is. Books have titles, authors, subjects, publishers, dates of publication, maybe prices and so on. If we would make a list with all the books in a library ordering them by title,it would be a long and hard list to search. We should also consider the fact that someone may want to find all the books written by a specific author, not knowing their titles. Using facets, we choose some categories (title, author, publisher, price, and so on) that will combine to fully describe all the books in the specific library. We can find a specific book by choosing the terms in each category that match that book.



Manufacturer is a **facet**, a way of categorizing the results

The **facet count** or constraint count shows how many results match each value

Canon, Sony, and Nikon are **constraints**, or facet values

The **breadcrumb** trail shows what constraints have already been applied and allows for their removal

Regular search results list

Fig. 1.1. Example of a faceted search interface

In figure 1.1 we can see an example of a faceted search interface, where we can observe the facets and their values, the item’s count for every facet value, the constraint the user chose and the list of results.

As William Denton says, a faceted classification is “a set of mutually exclusive and jointly exhaustive categories, each made by isolating one perspective on the items (a facet), that combine to completely describe all the objects in question, and which users can use, by searching and browsing, to find what they need.”

1.2 Motivation

Faceted search is used to help users explore and navigate through a large collection of items and also help them locate the items they are looking for as fast and as easy as possible. The existing faceted search systems are manually constructed and are the same for all users. We propose an adaptive faceted search that can be used by any online store, with any type of products. The system adapts the results list, the facets and the facets values to each user’s needs.

Before faceted search, there were two main paradigms in the web search world. Those paradigms are the navigational search and the direct search.

In *navigational search*, the information is organized using a hierarchic structure, allowing users to narrow the scope of their request in a predetermined order. An example of navigational search is Yahoo! Directory. A drawback of this approach is the impossibility of classifying objects by more than one dimension. This is a problem for the users who want to search the objects by other aspect than the one used for the ordering of the collection.

Direct search is the model used by web search engines and it allows users to write their request in a text box. The problem with this approach is that a user of a web search interface is not always aware of the content of the collection he is searching in. He might not even know for sure what he is looking for or how to phrase the query.

The direct search was more popular in the last few years, but recently a new paradigm, the faceted search, has appeared. It overcomes the above mentioned drawbacks of the other two paradigms and has become the prevailing user interaction mechanism in e-commerce sites.

Faceted search combines the first two paradigms, by allowing users to “navigate a multi-dimensional information space by combining text search with a progressive narrowing of choices in each dimension” (Wikipedia).

Web browsers provide an easy way of navigating through many dimensions, so they go very well with facets. Barbara Kwasnick said that “the notion of facets rests on the belief that there is more than one way to view the world, and that even those classifications that are viewed as stable are in fact provisional and dynamic. The challenge is to build classifications that are flexible and can accommodate new phenomena."

In the next sections, we will see when it is appropriate to use a faceted classification, and we will present some of its advantages and disadvantages.

1.2.1 When to use faceted classification

Things can be classified in different ways, depending on the different purposes of the classification, and different structures can be used. According to Barbara Kwasnick, there are four different structures used for classifications, and choosing which one to use depends on the purpose of the classification. These structures are: hierarchies, trees, paradigms, and facets.

Hierarchies and trees should be used when the objects need to be classified by only one dimension. For hierarchies, groups are divided into subgroups and a subgroup inherits all its parent’s attributes. For trees, the subgroups don’t inherit attributes from their parents.

Paradigms should be used when the objects have two dimensions of classification. A paradigm can be seen as a spreadsheet.

Facets should be used when the object need to be classified by three or more mutually exclusive categories.

To better understand when it is appropriate to use faceted search, we will study some of it’s advantages and disadvantages.

1.2.2 Advantages

Faceted search has the great advantage of allowing the exploration of an unknown dataset. It also offers great flexibility, it doesn’t require complete knowledge of the collection, which is being classified, and you can add facets any time, as you need. Facets are expressive, they can help eliminate confusion regarding search terms. They are effective because they allow retrieval on whichever attributes of the item are important to the person who is searching. Facets allow many different perspectives on the classified items. Sites which show the number of hits for each option, make it more easy for the users to decide if they should further refine their queries of go straight to the item. Faceted search also prevents dead-end query, meaning that the facet values that would lead to an empty results list are not displayed.

1.2.3 Disadvantages

Faceted search systems also have some disadvantages. Choosing the right facets or adding new facets can be complicated. Another disadvantage is that the relationships between the facets cannot be expressed. Also, a faceted classification is not easy to visualize and it requires a commitment to the creation and maintenance of the classification over time. Also, it is more difficult to implement a faceted classification with unstructured and untagged data.

1.3 Overview

Nowadays, it is more and more important, when talking about Internet services, to help users find the documents or items they are looking for. Even though faceted search is a popular feature in e-commerce and digital libraries, current faceted search interfaces are manually constructed and are identical for every user. Not much research has been conducted on how to adapt the facet and their order, what facets to present to a user in order to improve the search experience. A pre-defined interface may not serve all users of the system adequately. In this paper, we will analyze strategies to create the appropriate facets and facet values and to customize the search interface to each user’s behavior.

The main problems that we will approach in this paper can be summarized as follows.

Extracting the facets and facet values. Most of the existing faceted classifications are manually constructed. We cannot manually select the facets and their values if we don’t know what items the collection contains. We will study how we can construct the faceted classification without prior knowledge of the items in the collection.

The vocabulary problem. Facets and facet values are official words that should be representative and intuitive. They have to be carefully chosen and maintained. Some other terms or user's queries may need to be translated into existing facet values. This problem also appears in the process of extracting the facets and facet values. Two items may have the same value for a facet, but the word describing it may differ.

Personalization and facet ranking. The order of facets and facet values displayed for a user should help that specific user to get to the result as quickly as possible. Some facets may be more relevant for some users than for others. We will try to personalize the faceted interface interaction using facet ranking and user's feedback.

Backend aspects. We will study different approaches to store the facets and their values, as well as retrieving the items that respect the constraints the user selects or searches.

Frontend design. In faceted search, the user interface has a great importance too. We will specify some ideas and good practices for a good interface design.

We will first go through some general information, principles of facets, and existing faceted classifications, we will then examine in more detail the problems mentioned above and in the end we will explain our implementation of an adaptive faceted search system.

2. Previous work

The academic community has showed interest in the faceted search subject, especially regarding library information and information retrieval research.

There are some well-known universal faceted classification, like Colon Classification and Bliss Bibliographic Classification, which can be used as a starting point when creating a faceted classification.

2.1 Colon Classification

A classic universal faceted classification system is S.R. Ranganathan's Colon Classification [5], which was the first faceted classification. It is especially used in libraries in India. It has five facets, known as PMEST:

- Personality (the something in question)
- Matter (what something is made of)
- Energy (how something changes, is processed, evolves)
- Space (where something is)
- Time (when it happens)

2.2 Bliss Bibliographic Classification

Henry E. Bliss (1870 - 1955) wanted to create a classification system that can adapt to any item a collection can contain. His idea was the concept of “alternative location”, meaning that an item could be put in more than one category.

The second edition of Bliss Bibliographic Classification [6] has the following facets:

- thing/entity
- kind
- part
- property
- material
- process
- operation
- patient
- product
- by-product
- agent
- space
- time

Vanda Broughton, one of the editors of BC2, said that "these fundamental thirteen categories have been found to be sufficient for the analysis of vocabulary in almost all areas of knowledge. It is however quite likely that other general categories exist.”

2.3 Louise Spiteri’s model for faceted classification

According to William Denton, in his article “How to make a faceted classification and put it on the web”, Louise Spiteri created a set of principles that a faceted classification should respect. She divides the classification into three parts: "the Idea Plane, which involves the process of analyzing a subject field into its component part; the Verbal Plane, which involves the process of choosing appropriate terminology to express those component part; and the Notational Plane, which involves the process of expressing these component parts by means of a notational device".

The principles of Spiteri's model for creating faceted classifications, as mentioned by William Denton, are:

2.3.1 Idea Plane

- Principles for the Choice of Facets:

a) *Differentiation*: "when dividing an entity into its component parts, it is important to use characteristics of division (i.e., facets) that will distinguish clearly among these component parts".

b) *Relevance*: "when choosing facets by which to divide entities, it is important to make sure that the facets reflect the purpose, subject, and scope of the classification system".

c) *Ascertainability*: "it is important to choose facets that are definite and can be ascertained".

d) *Permanence*: facets should "represent permanent qualities of the item being divided".

e) *Homogeneity*: "facets must be homogeneous".

f) *Mutual Exclusivity*: facets must be "mutually exclusive," "each facet must represent only one characteristic of division".

g) *Fundamental Categories*: "there exist no categories that are fundamental to all subjects, and ... categories should be derived based upon the nature of the subject being classified".

- Principles for the Citation Order of Facets and Foci

a) *Relevant Succession*: "the citation order of the facets [and foci] should be relevant to the nature, subject, and scope of the classification system". The citation order can be: chronological order, alphabetical order, spatial or geometric order, simple to complex, complex to simple, canonical, increasing quantity, or decreasing quantity.

b) *Consistent Succession*: "once a citation order of facets has been established for a classification system, it should not be modified unless there is a change in the purpose, subject, or scope of the system".

2.3.2 Verbal Plane

a) *Context*: "the meaning of an individual term is given its context based upon its position in the classification system".

b) *Currency*: "the terminology used in a classification system should reflect current usage in the subject field". This means the system will require regular attention and revision.

2.3.3 Notational Plane

a) *Synonym*: "each subject can be represented by only one unique class number".

b) *Homonym*: "each class number can represent only one unique subject".

c) *Hospitality*: "notation should allow for the addition of new subjects, facets, and foci, at any point in the classification system".

d) *Filing Order*: "a notational system should reflect the filing order of subjects. This type of notation would reflect the citation order underlying the classification system".

2.4 William Denton's steps for creating a faceted classification

William Denton specifies in his article "How to Make a Faceted Classification and Put It On the Web" 7 steps to make a faceted classification, respecting the principles proposed by Louise Spiteri. These steps are:

1. *Domain collection*. This represents the step when we should choose the items that will be the training set for creating the faceted classification. If the domain is small, we can take all the items. Otherwise, we will take the most representative items, but we have to check that those items cover all the possibilities.

2. *Entity listing*. William Denton proposes to decompose every item's description into parts. The resulted words will be rearranged and we will choose from them the concepts that we will use for the creation of facets.

3. *Facet creation*. By analyzing the concepts obtained at the previous step, we will identify some high-level categories that appear in every item. Next, we can narrow down these high-level categories and obtain the facets that are "mutually exclusive and jointly exhaustive". Every term from the previous step must fit in these facets.

4. *Facet arrangement.* All the terms, meaning the facet values, will be ordered under the facets. After creating this classification, we have to verify that all the original items can be classified using facet values from the facets. If there are items that cannot be classified, we have to rebuild the classification.

During this step, we should also decide on a c*ontrolled vocabulary.* Facets and facet values are official words and phrases that will always be used for the concepts or things they represent. They have to be clear, unambiguous, everybody should understand them.

5. *Citation order*. This step refers to a default order of the items. This is optional, since on the web, the items can be ordered in multiple ways. This step was meant mostly for situations when a physical organization is need, for example printed bibliographies or library shelves. This default order will be used by the people who don't wish to make their own order, so it should be understandable and useful to as many users as possible.

6. *Classification*. This is the step when we use the classification system we have just created to classify all the items in the domain. We have to choose for every item, the facet values that match with it.

7. *Revision, testing, and maintenance.* If there are items that cannot be classified, we should go back as many steps as necessary to correct the problem. Maybe the facets and facet values we chose are not the right ones, maybe we need to add new ones. The order of the facets and facet values may need to be changed. William Denton suggests that the classification should be tested on users and modifications may need to be done afterwards.

Regular maintenance of the classification is also needed: as terminology changes, we should update it; we should verify that new items can be added to the classification, we may need to add, remove or rearrange facets and facet values as the domain of the classification changes.

2.5 Adaptive Faceted Search

Faceted search has become a popular technique, used in more and more commercial applications, particularly for online retailers and libraries. One shortcoming of the current method of constructing a faceted search system is that the faceted classification is manually constructed and is the same for every user. This paper studies methods of generating the faceted classification automatically and adapting the classification to each user's needs.

2.5.1 Creating the facets and facet values

The first step in creating a faceted search system is creating the facets and their values. This means breaking subjects into their component parts, which can be relatively easy for uniform collections, such as recipes, books or wines. In more complex and irregular collections, this is more complicated, but can give greater advantages.

As mentioned in [11], for e-commerce applications faceted classification it a good match because the facets are applicable to all products. For less unstructured data or documents, finding the facets can be much more complicated, but it can also be used. In [11] Glenda Browne mentions that "it is in these complex areas that faceting can bring important benefits by allowing users to search on separate aspects of topics. "

Most of the existing faceted classifications are manually constructed. The identification of facets is either a manual procedure or is based on an apriori knowledge of the facets and items that can appear in the collection. This paper proposes a system of creating a faceted classification for a collection that can contain any type of items. If we don't know from the start what items our collection will contain, we cannot manually construct the classification. This is a reason why we should use a method to automatically extract facets and facet values from the content or from the description of the items in the collection.

A faceted classification is based on a controlled vocabulary. The items in the collection must be indexed with metadata from the controlled vocabulary. If the items in the collection are already tagged, this step is simplified. Faceted search is more difficult to implement with unstructured and untagged data. However, according to [12], automated metadata extraction tools that recognise companies, people, products and other standard text can be used with unstructured items, so that they can be used for faceted classification searching. As mentioned in [11], studies have shown that subjects are better identified by computers than genres or predictive users. Manually adding metadata that general facets that are difficult to recognise for computers, may be a useful method for improving the quality of the classification.

Methods of creating facets and facet values

Facets can be derived from documents or item's descriptions in several ways:
1. We can use tools that automatically extract facets and facet values from unstructured content. When some unstructured content is added to a collection, the text is analyzed using natural language annotators and different parts of the text are tagged with keywords that may belong to already defined facets or to new facets.
2. The structured content added to the collection can be already tagged with facet-value pairs. This means that someone, maybe the person who adds the items in the collection, has to tag the appropriate text elements with the appropriate facet-value pairs. This method may improve the quality of the classification and reduce the complexity of the system since the text is already tagged, but it requires more human effort. The person who adds the metadata should also have some knowledge about the faceted classification so that the selected facets and facet values would respect the principles mentioned in a previous chapter (2.3.1 Idea Plane - Principles for the Choice of Facets).
3. We can define rules that associate text with keywords or facets. We can create dictionaries that associate keywords with facets and define synonyms for keywords. When unstructured content is added to the collection we can use the rules and dictionaries. This method requires human effort and it has a major drawback: we have to know what kind of items are going to be added to the collection so we can define a relevant dictionary. Another problem would be the fact that for some items new facets should be added or other synonyms appear in the text but we didn't define them in our dictionary so they will be ignored.

Tools for automatic extraction of facets and facet values

By using tools that extract metadata from unstructured and untagged data, faceted classifications can be made for any type of collection. We don't have to know what items the collection contains or what items will be added and the human effort is minimized since the text is automatically analyzed and tagged. A drawback of this method is the limited number and types of facets that a tool can generate.

One tool for automatic extraction of semantic information is *OpenClais* ([13], [14]). According to [3], it is used by the adaptive faceted search system for Twiter. It is a free web service and an open API that performs named-entity recognition and enables automatic metadata generation. Calais Web Service receives unstructured text and can identify entities, events, facts, that are returned in a RDF (Resource Description Framework) format.

As mentioned in [4], faceted search problem can be formulated in RDF terminology. A given RDF statement (subject; predicate; object) can be interpreted as follows:

- the subject is the item that should be returned by the search engine (the item that matches the selected facet value)
- the predicate is the facet
- the object is the facet value

The faceted query consists of a predicate-object pair or multiple predicate-object pairs. The faceted search engine should return all the subjects that correspond to those specific predicate-object pairs.

Other tools for extracting information from unstructured data are General Architecture for Text Engineering, also known as GATE ([15], [16]) and Machine Learning for Language Toolkit or Mallet ([17], [18]). GATE is a Java suite of tools which offer, among others, information extraction in many languages and Mallet is a Java-based package for statistical natural language processing, document classification, clustering, topic modeling, information extraction, and other machine learning applications to text.

A technique for automatic construction of facets

In [19] Wisam Dakka and Panagiotis Iperiotis propose a technique for automatic construction of faceted interfaces. The main idea behind their approach is the fact that the facet terms rarely appear in the documents so they propose using an external resource to "expand" the terms considered important (the facet values in the document) and find some "context" terms that don't appear in the document (the candidate facet terms). Their hypothesis is that the facet terms will appear after the expansion and then they can group together facet terms that belong to the same facet. They propose a technique to find the important terms in a document, a technique to expand those terms and identify facet terms that do not appear in the document and a technique to identify the candidate facet terms based on their frequency. In [21], two methods of selecting the facets from the set of candidate facet terms are presented and in [22] the problem of disambiguation is addressed and it gives us an approach to select the right facet for a term. These ideas suggest the following steps of the proposed technique for automatic generation of facets and facet values for any collection of documents:

1. Finding the important terms in a document which can represent the facet values. To find the terms in the document that carry important information, the authors of [19] propose using named entities, Yahoo terms, Wikipedia terms or domain specific vocabularies and ontologies. Named entities taggers can be used to obtain terms that represent important aspects of the content. Yahoo Term Extraction web service will return a list of significant words or phrases from the document. The authors of [19] use the idea that an important entity has its own Wikipedia page so they propose to mark a term as important if it matches a title of a Wikipedia entry. Furthermore, they exploit the redirect pages to capture variations of the same term, even if the term doesn't appear in the document in the same format as in the title of the Wikipedia page. Similarly, they exploit the anchor text from other Wikipedia entries to find different descriptions of the same concept. Domain specific vocabularies and ontologies can be used to find important terms for a domain. The methods above can be combined to find the important terms in a document.

2. Finding the context terms which will be used to find the set of candidate facet terms. To expand the important terms found in the previous step and find the "context" terms, the following methods are proposed: Google, WordNet hypernyms, Wikipedia Graph and Wikipedia synonyms. By making a Google query with a term, we can consider the context terms as the most frequent terms that appear in the returned snippets. WordNet hypernyms [20] could also be used to generate faceted hierarchies. A term X is a hypernym of a term Y, if every Y is a kind of X, for example plant is a hypernym of rose. According to studies, hypernyms are not that good with named entities, so other external tools should also be used. Another useful method to find context terms is considering the associations given by links between Wikipedia entries. Given a term, we can find the context terms as the top k linked terms with the highest score. This score is calculated using a tf-idf (term frequency–inverse document frequency[9]) method: for a link t1->t2 the score is log(N/in(t2))/out(t1), where N is the number of all Wikipedia entries, in(t2) is the number of incoming links that point to t2 from other entries and out(t1) is the number of outgoing links from t1 to other entries. To capture variations of the same term or synonyms, Wikipedia redirect pages and anchor text are used.

At the end of this step, for each document we will have a list of important terms and a list of context terms for each important term in the document.

3. Selecting the candidate facet terms from the context terms. To select the candidate facet terms, a comparative term frequency analysis is made. The frequency of the term in the initial collection and the frequency of the term in the context terms collection are used. The selected terms are the ones more frequent in the context collection than in the initial collection.

4. Selecting the facets. After obtaining a set of facet term candidates, we must choose the ones which will be our facets. In [21], two methods for grouping together the candidates and select the facets are proposed. The first approach is to build subsumption hierarchies ([22], [23]) on top of the extracted candidate facet terms and the high level categories will be our facets. Another approach would be to cluster all the candidate facet terms and name each cluster; the names of the clusters will be the facets.

5. Selecting for a facet value the appropriate facet. A word can have different meanings according to its context. To establish the correct sense of a word in a document we have to consider the other terms in the document and their context terms. Since the set of candidate facet terms is selected based on the frequency of all the context terms in the document, the most appropriate context term for the word will probably be selected as a facet candidate. If the document is very large and a term is used with more than one sense, it can happen that a term has more than one facet associated. In this case, we can consider only the one with the higher frequency (the rank used for the selection of candidate facet terms) or we can consider all the facets and add the term as a facet value for each of those facets.

Even though the faceted classification is built, it has to be regularly maintained and updated. Adding or removing items may change the classification, facets or facet values may need to be added or removed. Therefore, regular maintenance of the classification is an important step in creating a good faceted search system.

2.5.2 The vocabulary problem

When thinking about vocabulary in the faceted search domain, there are several aspects to consider: the difference between the terms used by the designers or indexers of the faceted classification and its users, synonymy (multiple ways to express the same concept) and polysemy (one word or phrase corresponding to multiple distinct concepts).

In the paper entitled “The Vocabulary Problem in Human-System Communication” [28] the authors demonstrated how “the keywords that are assigned by indexers are often at odds with those tried by searchers.” They mention that people use a very large variety of words to describe the same thing and a no matter how carefully we choose a word, that word cannot cover all users’ attempts.

The vocabulary problem associated with the facet and facet value terms

Faceted refinement options offer the user guidance, which can be more useful as the user better understands the indexer’s vocabulary. The users may understand the information space in other terms than the facet terms of the classification.

Daniel Tunkelang mentions in his book “Faceted Search (Synthesis Lectures on Information Concepts, Retrieval, and Services)” [25] that one of the challenges of faceted search is helping users understand the refinement options presented to them: “For users to make effective use of refinement options, the refinements must offer users what Pirolli et al. call <<information scent>>: cues that indicate to users the value, cost of access, and location of available information content." When users are searching for a particular item, it is important for them to find one option that gives them the needed "information scent". Finding multiple similar options may create confusion and the user can be overwhelmed.

The method proposed by Daniel Tunkelang for increasing "the information scent associated with a refinement option" is to offer content preview for every option. This preview should be clear so that the users understands what that option refers to, but should also be concise. Another measure that should be taken in the case of too many refinement options that could confuse the user is avoiding duplication by "cleaning up" the facet values and and clustering similar facet values.

The vocabulary problem should be considered when creating the facets and their values. The terms used should be clear and intuitive and the hierarchy shouldn't confuse the users. Linguistic processes, that are not the scope of this paper, could be used in case of synonymy and polysemy. The faceted classification can be based on a controlled vocabulary and the items in the collection can be indexed with metadata from the controlled vocabulary [24]. According to Wikipedia, "controlled vocabularies solve the problems of homographs, synonyms and polysemes by a bijection between concepts and authorized terms. In short, controlled vocabularies reduce ambiguity inherent in normal human languages where the same concept can be given different names and ensure consistency."

The vocabulary problem associated with the users' text queries

As mentioned in the first chapter, faceted search enables users to "navigate a multi-dimensional information space by combining text search with a progressive narrowing of choices in each dimension" ([1]). The terms used for text queries may be different than the ones used for our faceted classification. This problem, of matching user-specified search terms against document text or system terms, considering synonymy and polysemy, is a larger topic in information retrieval and is not the purpose of this paper.

There is another type of search, category search, that is not a search among the items of the collection, but a search among the facet values. When a user wants to narrow down the result set using a text query, we can search among the facet values and return the items with the matching values. This approach is presented in [26], where dynamic category sets are used as the result of a category search. This approach is used as a solution for the vocabulary problem represented by the fact that a user may not understand the information domain in the terms of facets chosen by the designer of the classification. Dynamic category sets can be used in conjunction with linguistic techniques to solve the problems of synonymy and polysemy too.

Dynamic category sets [26] are sets of facet values that match with the terms in the user's query. For example, considering the query "history of Germany in 19th century", the search can return {Subject = History, Location = Germany, Time Period = 19th Century} and {Subject = Germany, Time Period = 19th Century}. We can make the restriction that a set has to match all the query terms or a minimum number of terms. The category sets should be minimal. For example, if the query is "fiction", minimal result sets can be {Genre = Fiction} and {Genre = Non-Fiction}, but the sets {Time Period = 20th Century, Genre = Fiction} and {Subject = History, Genre = Fiction} are not minimal. The search shouldn't return category sets that don't match any item in the collection (that lead to dead ends), just like in the user interface shouldn't be displayed facet values that don't match any items.

2.5.3 Personalization and facet ranking strategies

We will study two aspects regarding the personalization of faceted search interfaces: the facet ranking problem, which refers to which facets and in which order to present to the user, and ordering the result list. The goal is to help the users get to the items they want as fast as possible.

The facet ranking problem

One of the main challenges of the faceted search interface is selecting which facets and facet values to make available to the user at any time. This is especially important when the document domain is very large. Some systems show users all available facets and facet-values. This approach can quickly overwhelm the users and lead to diminished user performance. Other systems simply present the first few facet-values in an alphabetized list. Also, a predefined interface may not serve all users of the system adequately. This means that the facet-value pairs should be ranked so that the order in which they are presented to the users helps them narrow down and get to the result as quickly as possible.

In [3], a definition of the *facet ranking problem* is presented*,* using the RDF terminology explained in a previous chapter of this paper (2.5.1 Creating facets and facet values) : "Given the current query $\mathrm{F}_{\mathrm{query}}$, which is a set of facet-value pairs (predicate; object) ∈ $\mathrm{F}_{\mathrm{query}}$, the hit list H of resources that match the current query, a set of candidate facet-value pairs (predicate; object) ∈ F and a user u, who is searching for a resource t via the faceted search interface, the core challenge of the faceted search engine is to rank the facet-value pairs F. Those pairs should appear at the top of the ranking that restrict the hit list H so that u can retrieve t with the least possible effort." The goal is to minimize the effort a user has to invest in retrieving the items in which the user is interested in. This effort can be measured by click and scroll operations.

In the same paper [3], some facet ranking strategies for adaptive faceted search are proposed:

1. Non-Personalized Facet Ranking. This is a lightweight approach and it suggests that the facets or facet values should be ranked based on their occurrence frequency in the hit list (the set of items which match the query). This approach minimizes the risk of filtering out relevant items, by first showing the facets/facet values that appear in most of the items, but might increase the effort a user has to invest to narrow down search results.

2. Context-adaptive Facet Ranking. This strategy uses the concept of relationship between facets/facet values and the strength of the relationship. Given a partial query (the facet-value pairs the user has selected so far), the remaining facets/facet values are ranked based on the strength of their relationship with the facet-value pairs already selected. We can infer the relationship between two facets/facet values by considering the frequency of the queries or the number of selected items in which they appear together. This approach could be used per user or globally. The context-sensitive strategy can only be applied in situations where the user has already made one selection.

3. Personalized Facet Ranking. This strategy relies on a user model, which specifies a user's interest in a specific facet/facet value. The facets that are the most interesting for that specific user are displayed first. The relevance of each facet for a specific user can be learned by using a user's history, by letting the user modify the order of the facets using drag&drop, or by allowing the user to explicitly rank the facets. Another option is a

collaborative approach. As mentioned in [7], “based on the assumption that a user shares similar criteria, preferences, or action patterns with some other user, information can be borrowed from others in order to aid a new user. The idea of learning from others is called social learning in psychology [...] The research on social learning is more focused on explaining human behavior <<learned observationally through modeling: from observing others one forms an idea of how new behaviors are performed, and on later occasions this coded information serves as a guide for action>>.” This idea is used in information retrieval domain, to make recommendations to a user by computing the similarity between one user’s preferences and that of other users.

The above facet ranking strategies can be combined to create new ranking strategies.

Ordering the result list

The hit set, meaning the set of items which match the current query, should be ordered so that the most relevant items for that specific user are displayed first. To do this, we could use two approaches:

1. The first approach is to use the interest of the users in certain items. This can be accomplished by studying what items a user has searched before, what items he has clicked on or viewed, or in the case of an online store, what items he has purchased. Another option could be allowing the users to explicitly rate the items found for a query. The idea of using explicit user feedback is motivated by the success of online reviewing/rating systems such as Flickr, Netflix and YouTube, which have demonstrated that millions of users are willing to provide explicit ratings about items.

2. The second approach refers to learning strategies that exploit the similarities between users. We can infer relationships between users by comparing their queries or the items they have search, clicked or bought. We can use this as a measure of their similarity. Given a user, we can use the rankings or the items selected by similar users to order the hit list or to make suggestions to our current user. This is a collaborative approach which we have already mentioned when talking about the personalized facet ranking strategy.

User profile

As we mentioned above, we can personalize the faceted search interface by using user profile. This idea is not new, a domain in which it is used is the one of recommender systems. Search engines usually use recommender systems that identify user interests through various methods in order to provide information specific to the user’s needs. Some ideas and proposals regarding this problem are presented in [8]. One issue addressed in this paper [8] is that many existing recommender systems “suffer from the the cold-start problem of handling new items or new users. In a collaborative system, for example, new items cannot be reliably recommended until they have been rated by several users. In some cases the number of users might be small compared to the number of items, which causes sparsity problems in many algorithms. Recommendations for items that are new to the catalog are therefore considerably weaker than more widely rated products. To avoid this cold start problem, content-based recommendation systems recommend items based on item content and a profile of the user’s interests.”

Traditional content-based recommender systems use term frequency–inverse document frequency method (TF-IDF) [9] to find the top “n” words from each document a user visits and then use these keywords to locate recommendations. The recommendations for a user may be computed by using the top keywords in the user's history as a query to the search engine. The TF-IDF method uses a weight for each term that represents the importance of the term in the collection of documents:

weight(term, document) = tf(term, document) * idf(term),

where:

- tf(term, document) represents the frequency of the the term in the document
- idf(term) represents the relevance of the term in the collection of documents (the total number of documents divided by the number of documents containing the term).

We could use this method to weight the facets, considering tf(f) as the frequency of the facet f in the user's query (how many times the user selected a value for that facet) and idf(f) as the importance of the facet, meaning the order of the facet in the sequence of facets selected by the user. This way, we can make the user profile for the order of the facets. We can also use this idea for the ordering of the results, by assigning a value to each result as the sum of the facet weights that the item contains. We should also consider the frequency of the specific values that the item has for every facet. So, an item which has facets and facet value that the user has selected more frequently and are more important will have a bigger value. This way, we try to predict the next facet and facet value for the user's query, considering the user's history.

The article "Conceptual Recommender System for CiteSeerx" [8] proposes a concept-based recommender system that recommends papers to general users of the CiteSeerx digital library of Computer Science research publications. Their system consists of three parts: a classifier, a profiler and a recommender.

The classifier uses all the items that have author-assigned tags as a training set to create a k-nearest neighbor classification method, which then will be used to classify all the other items. For each document, the classifier returns the top 3 concepts that are the most relevant for that document and a degree of association between the document and the associated concept.

The profiler tracks all the items that were viewed by a user, by tracking the user's click profile. For each of these documents, the top three concepts and their weights are retrieved. If one concept is found in more than one document, the weights from every document are added.

wt (cp, j) = ∑ wt (cp, i), for all documents i for user j,

where:

- wt (cp, j) = weight of concept cp in user profile j
- wt (cp, i) = weight of concept cp in document i

The final output of the profiler module is a list of concepts, the user might be interested in, and their corresponding weights arranged in decreasing order. Thus, the user profile is a list of concept-weight pairs, where the weights represent the amount of interest a user might have in a particular concept.

The recommender module retrieves the documents that have a specific concept as one of their top three concepts, as determined by the classifier. These documents are added to the list of possible recommendations. The weight of a document is calculated as weight associated with the concept in the user profile multiplied by the weight associated with the concept for the document. So the final weight of the document would be:

wt (i, j) = wt(cp, j) * wt(cp, i),

where:

- wt (i, j) = weight of document i for user j

- ○ wt (cp, j) = weight of concept cp in user profile j
- ○ wt (cp, i) = weight of concept cp in document i

In our case, we can apply this idea by choosing for each item 3 facet values that are the most representative for the specific item and assign them some weights. We can use the history of the user to extract the prefered facet values, the ones with the biggest weights. If a facet value is found in more than one item, the weights are added. So, we obtain a weight for every facet value in the user profile. We then need to compute the weights of the items and order them accordingly. We can use the formula mentioned above, for the recommender module.

The user profile can be improved using explicit user feedback.

Starting page

A faceted search system should present a good landing page for a user. Since a user profile describes the user's interests in the items, this information can be used to initially place users near their relevant items even before they begin to formulate a query. We can accomplish this by extracting from the user profile the facet-value pairs that are prefered by that user and making an initial query based on them. This query and the list of items that match the query creates the start state.

Some suggestions for creating the start page for a faceted search interface are presented in [7]:

1. Null Start State. In this method, each user begins in a state with no facet-value pairs selected by default and no pre-fetched documents. This is the simplest start state creation method.
2. Collaborative Start State. In this method the system automatically issues a query containing the facet-value pairs that are the mostly likely to be contained in a relevant document, considering all users. This query is then issued to the underlying retrieval algorithm and the matching documents are initially suggested to the user. Since the start state is globally created, every user is presented the same start state.
3. Personalized Start State. This method is similar to the method above, except that the default query is determined per user, using the profile of that specific user.

Adapting the order of facets and facet values is used to help the users find more rapidly and easier the items they want. There is still a risk associated with changing the initial order: the risk of confusing the users who might be overwhelmed by all the changes. Therefore, we should ask them if they want the system to adapt to their needs by changing the order of facets and facet values, or at least notify them.

2.5.4 Backend aspects

In this chapter, we will talk about some technologies and methods we can use for faceted search systems and some aspects concerning scalability and efficiency.

Scalability and efficiency

In [25] Daniel Tunkelang discusses about two of the backend concerns regarding faceted search: scalability and efficiency.

In his vision, when thinking about scalability in faceted search systems we have to think about the number of items, the number of facet values per item and the searchable text per item (in the case of documents). He roughly estimates the size of a faceted search system, considering for the implementation an inverted index that maps facet values or searchable text to documents and a document table that maps documents to facet values. In this situation, the size of the system is the product between the number of documents and the average width of a document (the number of facets per document plus the size of searchable text per document). For small collections, all this information can be kept in main memory, but for larger collections this is not the appropriate approach. Distributing the storage across multiple servers may be a scaling strategy, but this is not the scope of this paper.

Inverted index

Processing a faceted search query means performing two tasks: determine the set of results (the items that match the query constraints) and compute the counts for each facet-value pair (for the items in the result set). According to Daniel Tunkelang, the first task can be efficiently accomplished by using standard inverted index techniques [30], but the second task is the one which makes faceted search systems computationally demanding.

Inverted index "is an index data structure storing a mapping from content, such as words or numbers, to its locations in a database file, or in a document or a set of documents. The purpose of an inverted index is to allow fast full text searches, at a cost of increased processing when a document is added to the database. [...] It is the most popular data structure used in document retrieval systems, used on a large scale for example in search engines." [30] Thus, every facet-value pair will be mapped to a list of documents/items that match that constraint (have the respective value for the respective facet) which will be used to retrieve the items matching a query.

To accomplish the second task, Daniel Tunkelang mentions two approaches, both computationally expensive: a top-down approach and a bottom-up approach. The top-down approach uses the inverted index, looking up each candidate facet value to compute the intersection of the documents assigned to that value with the documents in the result set. The bottom-up approach iterates through the documents in the result set and then iterates through the facet values assigned to each document to accumulate them.

Bit vectors

Anne Schuth and Maarten Marx propose in their article “Fast faceted search in XML” [31], three methods for computing facet-value counts for a faceted search system using a XML database, named eXist-db.

The first method, which they call “the naive method” would be firing the current query together with every facet-value pair and record the length of each returned sequence. This is not a very suitable method, since its cost grows linearly with the number of facet-value pairs.

The second method was introduced by Van den Branden and it’s called “the rdbv method”. It exploits eXist-db specific range indexes.

The method they found to be much more efficient and scalable than the first two methods is based on bit-vectors [32]. Its performance is near constant when the size of the database grows. The use of bit-vectors is an efficient way to calculate counts. For each facet-value pair is used a bit-vector and every bit in a vector represents a document. The value of a specific bit is 1 if the document contains the facet-value pair. The bit-vectors for every facet-value pair can be constructed at index time and when a query is resolved a bit-vector for the result set is constructed. To calculate the counts, we have to intersect the last bit-vector with all other bit-vectors (corresponding to newly added constraints) and then calculate the cardinality. For example, if after executing a query we obtain the bit-vector 00010100, it means that the third and fifth documents are hits for the executed search query. To compare two result sets to each other and get the common results we will then use a bitwise AND operation. For example: 00010100 AND 00000100 means that only the third document is present in both result sets. If we count the number of bits turned on, we know how many documents match the query.

This can be calculated more efficiently by exploiting the capability of a processor to process a word in parallel.

XFML

Once you build a faceted classification, you can share it with others or you can reuse the faceted classifications made by others by using XFML. XFML [33] is a XML format for exchanging metadata in the form of faceted hierarchies. It provides a simple format to share classification and indexing data.

Once you have defined some facets and facet values, you can classify or index some web pages and add them to a XFML document so that your indexing efforts can be shared. You can only classify things that have a URI. XFML also provides ways to build connections between topics, information that lets you write clever tools to automate the sharing of indexing efforts.

The typical way to automate the reuse of indexing efforts is to write some code that regularly downloads an updated XFML file from sites with similar facets to our facets. We can then consider the relevant facets and copy the occurrences of relevant facet values to out XFML document.

Software for implementing faceted search

Since faceted search has become a popular technique in commercial search applications, an increasing number of enterprise search vendors provide software for implementing faceted search applications. One of the most known solution for implementing faceted search is Apache Solr Search engine platform based on Apache Lucene [34]. Another notable effort in faceted search is The CiteSeerX project at the Pennsylvania State University [35], which allows faceted search for academic documents and continues to expand into other facets such as table search.

According to Wikipedia [34] and [36], Solr is the most popular enterprise search engine. Its major features include: full-text search, faceted search, hit highlighting, dynamic clustering, database integration, and rich document handling. It is highly scalable, providing distributed search, index replication and load-balanced querying. Solr uses the Lucene Java search library at its core for full-text indexing and search, and has REST-like HTTP/XML and JSON APIs that make it usable from most popular programming languages. It offers very fast search, at the cost of an indexing overhead.

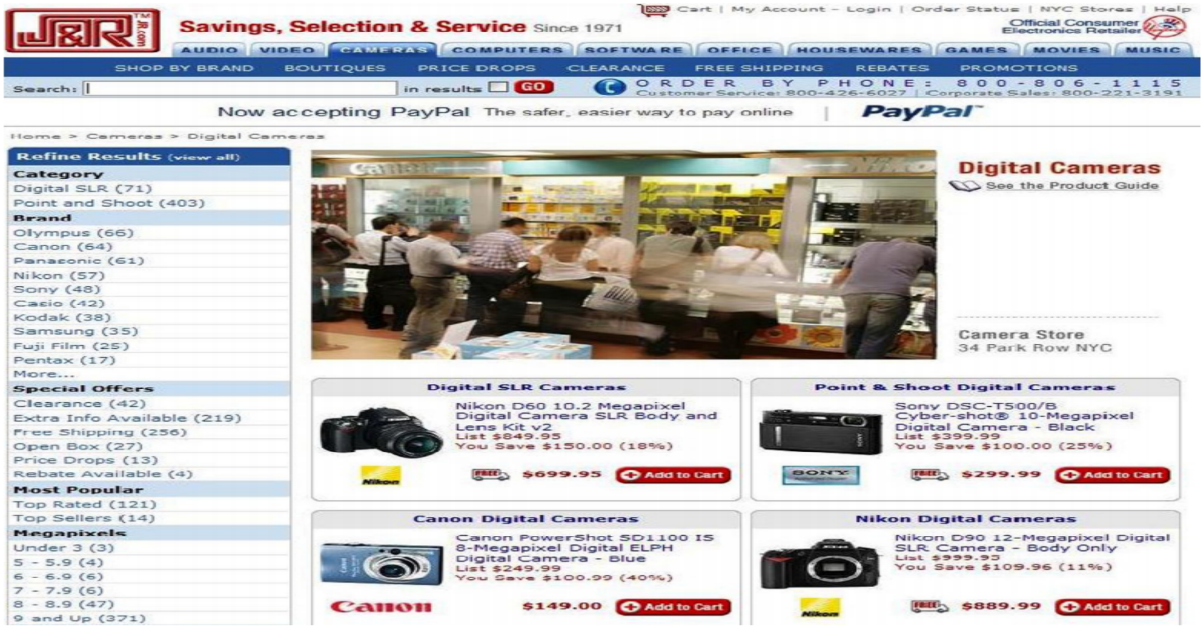
2.5.5 Frontend design

The design of the faceted search interface is also an important aspect in creating a faceted search system. The purpose of faceted search is to help users explore a large and unknown collection of items and also help them navigate as easy and as fast as possible to the items they are interested in. These needs cannot be met if the design of the interface is a poor one. Moreover, the power of faceted search can overwhelm and confuse users if it is implemented with a bad design.

In his book “Faceted Search (Synthesis Lectures on Information Concepts, Retrieval, and Services)” [25], Daniel Tunkelang addresses some of the most important and common interface challenges in the design of faceted search systems. In the following paragraphs, we will present some of the aspects mentioned in his book.

Facets and results

The first thing we should decide when we want to create a faceted search interface is how to display the set of facets and the set of results (items that match the query or the selected facet values). The decision depends on the type and scope of the interface. The sets could be displayed in the same view or in different views. Placing the two sets in different views is less common and may annoy the user, who has to make an extra effort to find the other view. Therefore, we will discuss the more common approach, placing the two sets in the same view. There are many options to do this as well.


Cart | My Account – Login | Order Status | NYC Stores | Help
Savings, Selection & Service Since 1971
Official Consumer Electronics Retailer
AUDIO
VIDEO
CAMERAS
COMPUTERS
SOFTWARE
OFFICE
HOUSEWARES
GAMES
MOVIES
MUSIC
SHOP BY BRAND
BOUTIQUES
PRICE DROPS
CLEARANCE
FREE SHIPPING
REBATES
PROMOTIONS
Search:
in results
GO
ORDER BY PHONE: 800-806-1115
Customer Service: 800-426-6027 | Corporate Sales: 800-221-3191
Now accepting PayPal The safer, easier way to pay online | PayPal
Home > Cameras > Digital Cameras
Digital Cameras
See the Product Guide
Camera Store
34 Park Row NYC
Add to Cart

Refine Results (view all)

Category
Digital SLR (71)
Point and Shoot (403)

Brand
Olympus (66)
Canon (64)
Panasonic (61)
Nikon (57)
Sony (48)
Casio (42)
Kodak (38)
Samsung (35)
Fuji Film (25)
Pentax (17)
More...

Special Offers
Clearance (42)
Extra Info Available (219)
Free Shipping (256)
Open Box (27)
Price Drops (13)
Rebate Available (4)

Most Popular
Top Rated (121)
Top Sellers (14)

Megapixels
Under 3 (3)
5 - 5.9 (4)
6 - 6.9 (6)
7 - 7.9 (6)
8 - 8.9 (47)
9 and Up (371)

Digital SLR Cameras
Nikon D60 10.2 Megapixel Digital Camera SLR Body and Lens Kit v2
List $849.95
You Save $150.00 (18%)
$699.95

Point & Shoot Digital Cameras
Sony DSC-T500/B Cyber-shot® 10-Megapixel Digital Camera - Black
List $399.99
You Save $100.00 (25%)
$299.99

Canon Digital Cameras
Canon PowerShot SD1100 IS 8-Megapixel Digital ELPH Digital Camera - Blue
List $249.99
You Save $100.99 (40%)
$149.00

Nikon Digital Cameras
Nikon D90 12-Megapixel Digital SLR Camera – Body Only
List $999.95
You Save $109.96 (11%)
$889.99

Fig. 2.1 *Faceted navigation with items area centered on the page and facets area to the left*

We can place the facets in a panel to the left/right of the result set, so that the user can see the facets all the time, but also can focus on the results which are centered in the page (as in figure 2.1).

Another option is to place the facets above the results, so that the user observes them, but this may require more effort to see the results (as in the figure 2.2). An upside may be that the users will make more elaborate queries and they will use the faceted search more efficiently, without rushing to the result set first. A less common approach is placing the facets below the results, which can make it difficult for users to observe or to use the facets, since they have to scroll to the bottom of the result list to see them.

A user's query can be ambiguous and the system may find different facet values that match with it. To establish the user's intent and not confuse him with a list of results that may not make sense for him, a clarification dialog should be presented.

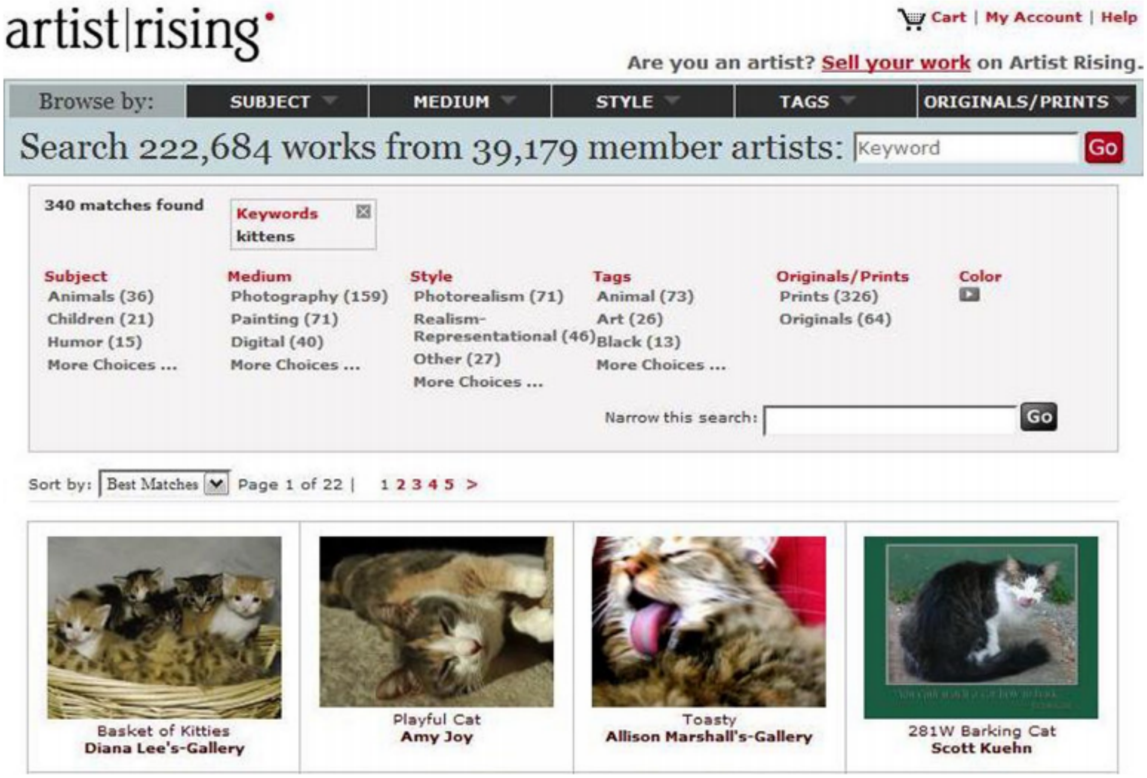


Fig. 2.2 *Faceted navigation with facets area above the results*

Organising the facets and facet values

Another important aspect in designing the faceted search interface is the organisation of facets and facet values. Faceted search systems are particularly useful in the case of large collections of items. The number of facets and facet values is also likely to be large in most situations. The wealth of information can overwhelm the users and we risk to loose what Daniel Tunkelang calls "the most valuable resource of all—the user's attention." To reduce this risk, we should reduce the number of facets and facet values presented and organize them effectively.

The strategies for organizing facets and facet values presented in [25] are:

- using a static pre-determined order which is the same for every user and doesn't change while the user navigates. This may help the user learn the model better since he always sees the same facets in the same order, but it might not help him find the results he wants as efficiently as possible. The facets presented first may not be the most relevant for the current user. This strategy is not suitable for a large number of facets, where only the first facets will be displayed by default and they may be irrelevant for the user.
- adapting the order of the facets and facet values, using rankings. Strategies for ranking the facet and the facet values were presented in a previous chapter (2.5.3 Personalization and facet ranking strategies) and the ideas mentioned there can be used to organize and order the facets in an adaptive and personalized manner.
- grouping together similar or related facets or facet values. This allows us to include more facets in a single display, using less space, by allowing users to collapse or

expand the groups as they want. An important aspect to consider here is the depth versus the breadth of the grouping hierarchies. We don't want the user to get stuck in an eternal drill down, which causes fatigue.

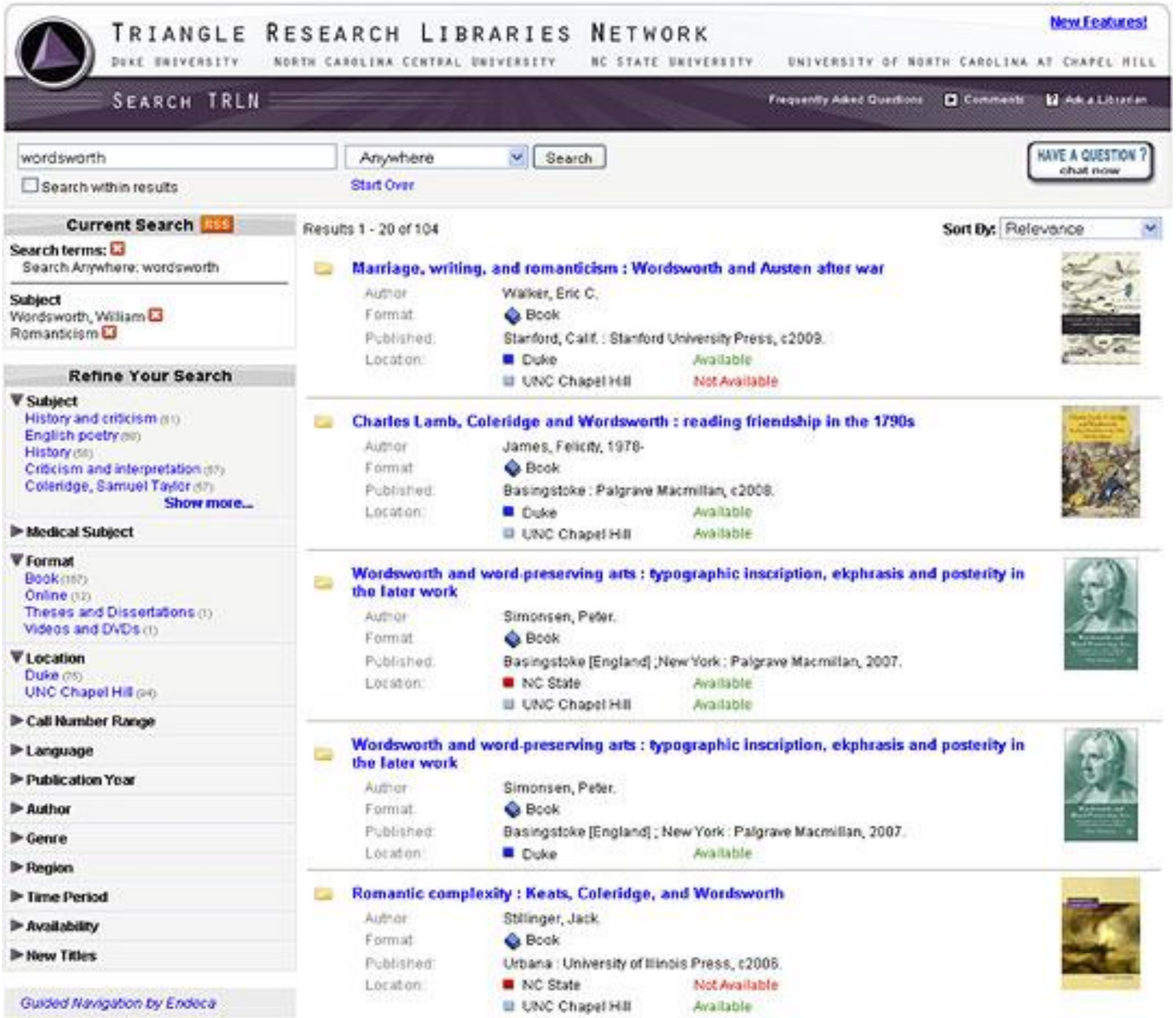
Search box

Since faceted search is still a type of search, the search box has an important role in the interface too. As mentioned in [25], a faceted search designer has many decisions to make regarding the search box. One decision is if the text query should complete the the current query filters or should start a new search from scratch. The conventional way is to clear all other filters, but we could let the user decide by clicking a checkbox. Another option is where to search: among the the description/title of the items, their content (in case of documents) or among the facets and their values. When making multi word queries or expanding queries, we should also consider precision and recall. The convention for multiword queries is that the search engine interprets the query as an unordered conjunction (AND) between the words. Regarding query expansion, users shouldn't worry about using the singular or the plural form of the word.

When selecting multiple facets values for the same facet, the system can consider disjunctive selection (OR) or conjunctive selection (AND). One of the challenges is to make the users understand whether selecting multiple values from a particular facet is disjunctive or conjunctive, particularly if the site offers both alternatives.

Example of a good interface design

In "Search Patterns" [29], Peter Morville and Jeffery Callender present an image with what they call "a successful implementation of faceted navigation" (figure 2.3). They point out the regions in which the facet values are placed, regions which can be expanded or collapsed. By default, only the most important facets are opened and each open facet reveals only top four or five most heavily populated values. The number of items matching each facet value is displayed next to that value, which gives the user a scent of the distribution of items and helps him decide if he should apply more filters or he can go to the result list directly. Only facet values with at least one corresponding item are displayed, so that dead ends are avoided. Another useful aspect is displaying the facet values that have been selected so far and allowing the user to delete any of them at any time.



TRIANGLE RESEARCH LIBRARIES NETWORK

DUKE UNIVERSITY NORTH CAROLINA CENTRAL UNIVERSITY NC STATE UNIVERSITY UNIVERSITY OF NORTH CAROLINA AT CHAPEL HILL

New Features!

SEARCH TRLN

Frequently Asked Questions Comments Ask a Librarian

wordsworth Anywhere Search

□ Search within results

Start Over

HAVE A QUESTION ? chat now

Current Search RSS

Search terms: ☒
Search Anywhere: wordsworth

Subject
Wordsworth, William ☒
Romanticism ☒

Refine Your Search

▼ **Subject**
History and criticism (81)
English poetry (80)
History (58)
Criticism and interpretation (57)
Coleridge, Samuel Taylor (57)
Show more...

▶ **Medical Subject**

▼ **Format**
Book (107)
Online (12)
Theses and Dissertations (1)
Videos and DVDs (1)

▼ **Location**
Duke (75)
UNC Chapel Hill (94)

▶ **Call Number Range**

▶ **Language**

▶ **Publication Year**

▶ **Author**

▶ **Genre**

▶ **Region**

▶ **Time Period**

▶ **Availability**

▶ **New Titles**

Guided Navigation by Endeca

Results 1 - 20 of 104

Sort By: Relevance

Marriage, writing, and romanticism : Wordsworth and Austen after war
Author Walker, Eric C.
Format Book
Published Stanford, Calif. : Stanford University Press, c2009.
Location Duke Available
UNC Chapel Hill Not Available

Charles Lamb, Coleridge and Wordsworth : reading friendship in the 1790s
Author James, Felicity, 1978-
Format Book
Published Basingstoke : Palgrave Macmillan, c2008.
Location Duke Available
UNC Chapel Hill Available

Wordsworth and word-preserving arts : typographic inscription, ekphrasis and posterity in the later work
Author Simonsen, Peter.
Format Book
Published Basingstoke [England] ;New York : Palgrave Macmillan, 2007.
Location NC State Available
UNC Chapel Hill Available

Wordsworth and word-preserving arts : typographic inscription, ekphrasis and posterity in the later work
Author Simonsen, Peter.
Format Book
Published Basingstoke [England] ; New York : Palgrave Macmillan, 2007.
Location Duke Available

Romantic complexity : Keats, Coleridge, and Wordsworth
Author Stillinger, Jack.
Format Book
Published Urbana : University of Illinois Press, c2006.
Location NC State Not Available
UNC Chapel Hill Available

Fig. 2.3 *Faceted navigation at the Triangle Research Libraries*

Faceted search is a powerful solution which, if it's well constructed, allows users to navigate through a large collection of items based on what is important to them. If it's not considered from the user's perspective and it's not designed with usability principles in mind, a faceted search interface can make the users lives more complex and frustrated. So, to really help the user, you need to understand the user's mental model and design the interface accordingly.

3. Our Implementation

In this chapter we will mention the choices we made for the implementation of a generic faceted search system, considering every aspect mentioned in the previous chapters. We will detail the aspects that we took into consideration when making a decision and possible advantages and disadvantages of the choices we made.

3.1 Choosing the facet and facet values when adding the product

As we previously mentioned, the first step in creating the faceted system is choosing the facets and facet values. We want to implement a faceted search system that will work regardless of the type of items the collection will contain.Two main options we have at this point are:

1. use some tools to automatically extract the facets and facet values from the unstructured and untagged information about the items in the collection.

2. structure the information about the items in the collection by adding metadata or tags and then use those tags to automatically extract the facets and facet values.

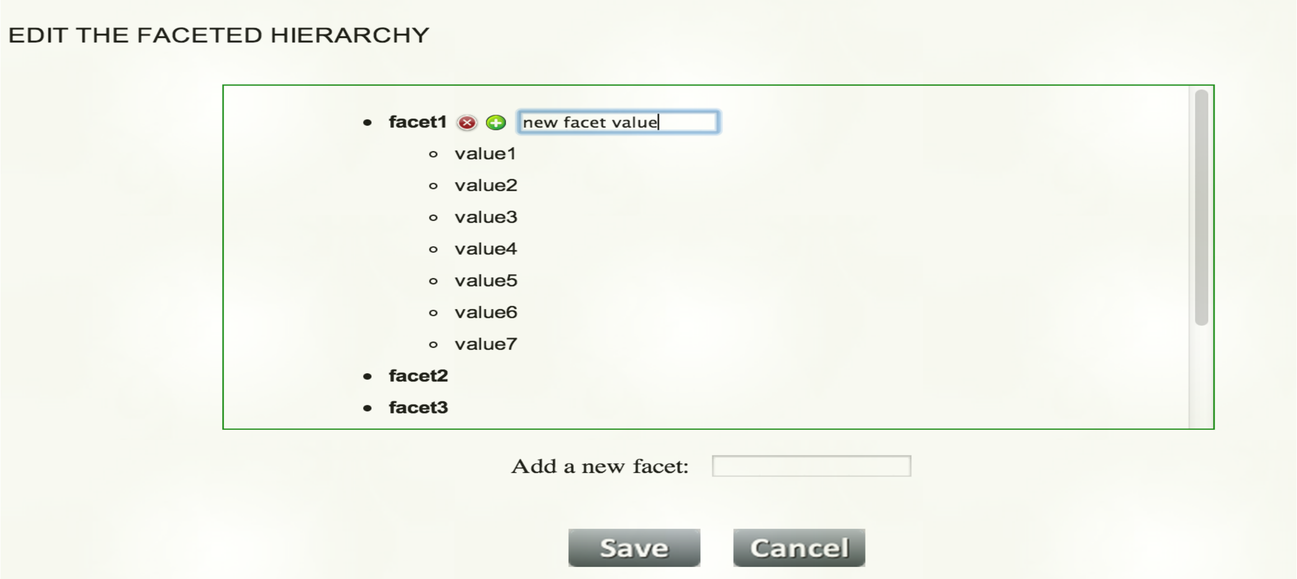
3. when a product is being added, instead of tagging it or its description with metadata, choose the facets and facet values corresponding to it from the list of currently existing facet and facet values or even adding new ones.

We chose the third option and we will try to explain why. One thing we considered when choosing how to implement this aspect was the fact that we want to create a system that is used for online stores, in which there is always a person who adds the products in the collection which is being classified. We can use the information the person who adds the products has about the respective item and make them tagg the products with metadata or choose the facets/facet values. We considered that trying to extract the facets and facet values without using any human interaction at all may not bring us too many advantages, it may be an overkill task that may not even generate an appropriate and relevant classification as a human can do. Since this task involves natural language processing and context disambiguation, involving a human may generate more reliable results. If we choose to make a person tagg the product, we should afterwards use some tools to extract the facets/facet values using those tags. Since we already decided that we will use some human effort, why not use that effort in a similar direction, not tagging the products but choosing for the product the corresponding facets/facet values from the list of existing faces/facet values or give the option to add new facets/facet values if the existing ones don’t match.

The main disadvantage of this approach is the fact that human effort is required. Another drawback is the fact that the person who adds/selects the facets/facet values may not have much experience in this task and may not choose the appropriate facets/facet values. We try to minimize this negative aspect by providing the user some relevant and clear information on how to choose the facet/facet values.

One advantage of this approach is the fact that the classification is more reliable since we avoid the errors that may arise from trying to process natural language or the inherent errors of context disambiguation. Another advantage is the fact that we use the information the user already has about the product, instead of trying to get that information from the product's description, that may not even be very comprehensive or relevant.

We considered that there are two types of users: privileged users, meaning the users who can administer the online store and who can add, remove or modify products, and normal users, who want to find and buy certain products from the online store. The faceted search system is addressed to the second type of users, it is built to help the normal users to find the products they are looking for more easy and faster. The two type of users can use the faceted search system in two different ways. The privileged users can add, remove or modify the products and the faceted classification (the facets and the facet values). When adding a product, they can specify the facets and facet values for that specific product, beside other details about the product, such as price, quantity, description. They can choose a product's facets and facet values from the already existing facets and facet values or they can add new facets and facet values that will be added to the list of global facets. A privileged user can also modify the faceted hierarchy, by adding and removing facets and facet values. They can also change the order of the facets and facet values by drag&drop the respective facets/facet values in the faceted hierarchy list.



EDIT THE FACETED HIERARCHY

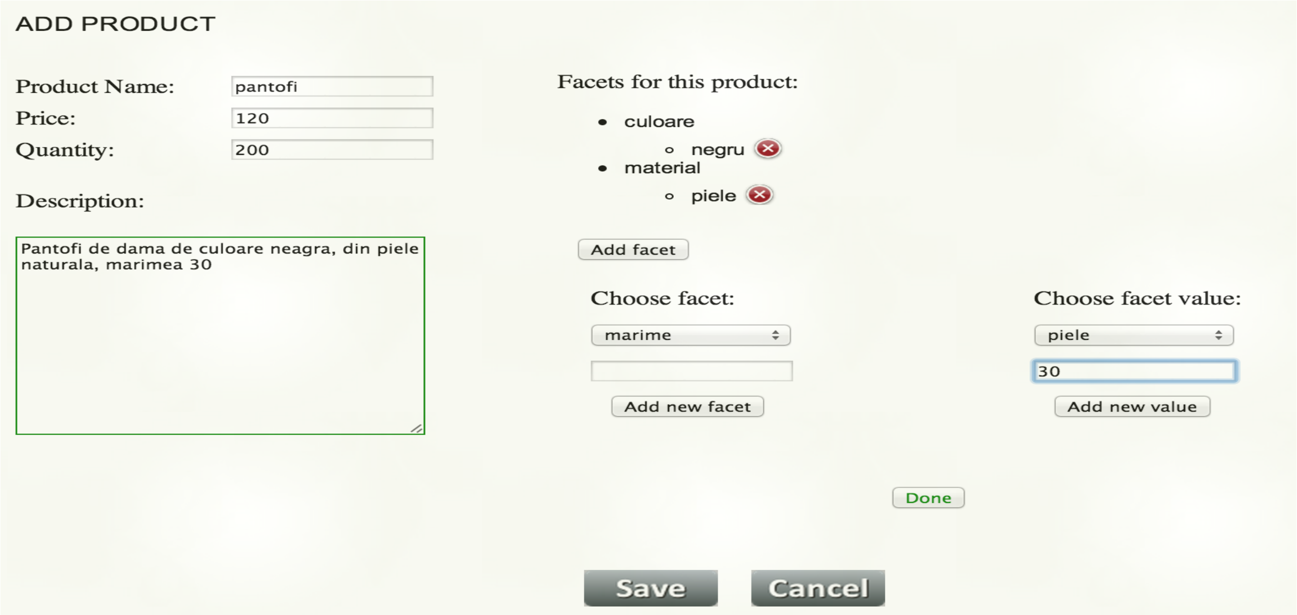
- **facet1** new facet value
  - value1
  - value2
  - value3
  - value4
  - value5
  - value6
  - value7
- **facet2**
- **facet3**

Add a new facet:

Figure 3.1 Interface for managing the faceted classification

In figure 3.1 we can see how the interface in which the privileged users can add, remove or change the order of facets and facet values looks like. Any facet or facet value can be removed by clicking the buttons that appear when hovering over the respective line. The order of facets and facet values inside a facet can be modified by dragging and dropping the respective facet/facet value. We can also modify the name of a facet or facet value. This is very useful because after the faceted classification is built, it has to be regularly maintained and updated. This also means that the names of the facets or facet values may need modifications. Therefore, this interface window is useful not only for building the classification, but also maintaining it, keeping it up to date.

Figure 3.2 shows the page in which a privileged user can add or edit a product. Observe how the user can add the facets and facet values for the product. Any facet or facet value that currently exists for a product can be deleted by clicking the button that appears when hovering over the respective line. You can add a facet from the drop-down with the existing facets or add a new facet by typing its name in the corresponding textbox (that facet will be automatically added in the drop-down also). The facet values in the second drop-down correspond to the facet selected in the first drop-down. You can also add new facet values for the selected facet, by typing its name in the corresponding textbox.



ADD PRODUCT

Product Name: pantofi

Price: 120

Quantity: 200

Description:

Pantofi de dama de culoare neagra, din piele naturala, marimea 30

Facets for this product:

- culoare
  - negru
- material
  - piele

Choose facet: marime

Choose facet value: piele

30

Figure 3.2 Interface for adding a product in the collection

3.2 The vocabulary problem

The vocabulary problem is an issue that should be considered when creating the facets and their values. Since, in our implementation, the facet and facet values are chosen by the privileged user, the responsibility of choosing the right terms for that facets and theirs values falls on his shoulders. The vocabulary problem in the context of choosing the facets and facet values terms refers to the fact that those terms should be clear and intuitive and they shouldn’t confuse the user. In our implementation, we try to minimize the risk of having this problem, by trying to make the privileged user that will choose these terms aware of the responsibility he has. We do this by adding a info message about the aspects that they should consider when choosing the facets and the properties the facet and facet value terms should have.

Another measure we take to avoid the problems of ambiguity and synonymy is showing the privileged users all the facets and facet values a product has and all the facet and facet values that already exist in the classification, before allowing them to add new facets and facet values. By making the users aware of all the existing facets and by showing them the info message with the things they should take into consideration when building the faceted classification, we try to avoid that the users add another facet that already exists but has a different name or adding facet values that are synonyms.

Over time, some terms may need to be replaced to make them more clear and intuitive or because they are no longer used and other terms should be added. Our implementation allows the privileged users to modify the facet and facet value terms, making it easy to maintain and update the classification's vocabulary over time.

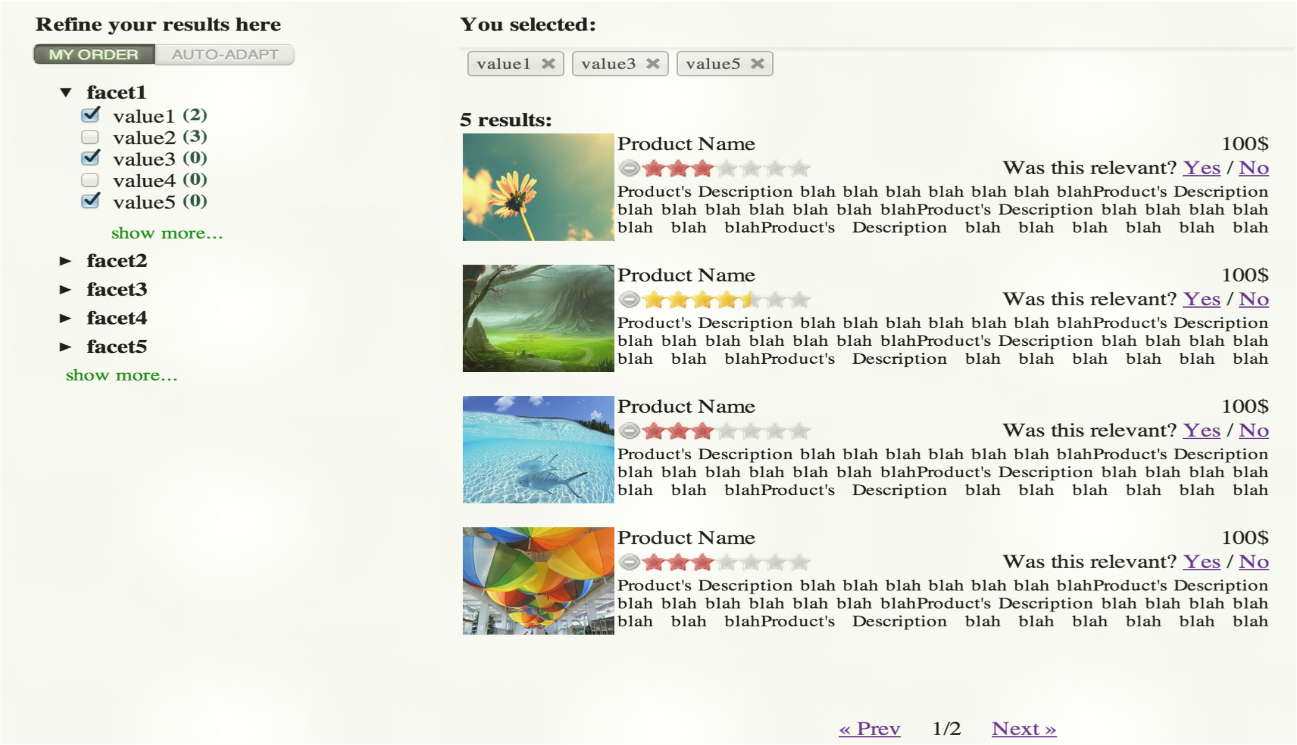
3.3 Personalization by using the user's profile

One of the most important aspects of the faceted search system we propose is the system's ability to adapt to the a specific user. This adaptability refers to the personalization of the facets and facet values and the personalization of the results list.

We will first discuss the personalization of the facets and facet values. In chapter 2.5.3 Personalization and facet ranking strategies, we described some facet ranking strategies that we can use to personalize the faceted classification. The strategy we chose to use in our implementation is the personalized facet ranking. We let each user modify the order of the facets and the values of each facet by dragging and dropping the respective facet/facet value to the desired position. This way, each user can create the order that best suits them and can modify that order any time. The predefined order of the facets and facet values is the order established by the privileged user when creating and editing the faceted classification. A normal user can use this predefined order, which is the same for everyone, or can create his own personalized order. Moreover, if the users want an automatic personalized order, they can choose to let us personalize their interface by creating their profile. We create the user's profile by ranking each facet/facet value considering the frequency of that facet/facet value in the user's queries. The more frequent a user uses a specific facet value, that facet value and the corresponding facet has a greater rank. In figure 3.3 we can see the interface where the users can select if they want us to automatically personalize the facets and facet values or if they want us to use the order they established by dragging and dropping the facets/facet values to the desired position.

The second aspect of the faceted search system personalization is the personalization of the results list. We use many methods of ranking the items in the results list, which we combine to get a final rank and then we sort the list in the descending order of that rank.

The first method to rank an item is allowing the users to rate the specific item. This rating system consist of a maximum number of five stars a user can give to a product. This type of rank is globally used, meaning that for each product, the average number of stars given by all users is computed and that rank is used for every user. We use this method to rank the quality of a product, as it is perceived by as many users as possible. The specific rank that a user sets for a product will have a greater influence when adapting the results list for that user, in the idea that if all the other users appreciate a product, but the current user doesn't, the opinion of the current user is more important when presenting him the results. This rank is also useful for a new user who hasn't ranked yet any products, but the products can still be sorted based on other users ratings.


Refine your results here

MY ORDER | AUTO-ADAPT

▼ facet1
- ☑ value1 (2)
- □ value2 (3)
- ☑ value3 (0)
- □ value4 (0)
- ☑ value5 (0)

show more...

► facet2

► facet3

► facet4

► facet5

show more...

You selected:

value1 ✖ value3 ✖ value5 ✖

5 results:

Product Name 100$
Was this relevant? Yes / No
Product's Description blah blah blah blah blah blah blahProduct's Description blah blah blah blah blah blah blahProduct's Description blah blah blah blah blah blahProduct's Description blah blah blah blah blah blah

Product Name 100$
Was this relevant? Yes / No
Product's Description blah blah blah blah blah blah blahProduct's Description blah blah blah blah blah blah blahProduct's Description blah blah blah blah blah blahProduct's Description blah blah blah blah blah blah

Product Name 100$
Was this relevant? Yes / No
Product's Description blah blah blah blah blah blah blahProduct's Description blah blah blah blah blah blah blahProduct's Description blah blah blah blah blah blahProduct's Description blah blah blah blah blah blah

Product Name 100$
Was this relevant? Yes / No
Product's Description blah blah blah blah blah blah blahProduct's Description blah blah blah blah blah blah blahProduct's Description blah blah blah blah blah blahProduct's Description blah blah blah blah blah blah

« Prev 1/2 Next »

Figure 3.3 Interface for the faceted search

The second method for product ranking that we use is allowing a user to specify if a product is relevant his query. This rank is used to personalize the results list of a user for each of his queries. If a user says that a specific product is relevant for the current query, we retain each facet value of the query and consider that the product is relevant for each of the facet values used. The relevance of a product for a facet value is incremented every time the users specifies the product is relevant for a query that contains the respective facet value and is decremented every time the users specifies the product is not relevant. We chose to use this method due to the success of rating systems such as YouTube or Flickr, which shows that millions of users are willing to provide ratings for items. In figure 3.3, we can see how a user can rate the item or specify if the item was relevant for his query.

Another method we use to rank the items is the frequency a item has been clicked on, buyed or rated by a specific user, in the context of specific query.

We also give every item a rank that represents the sum of the weights of the facet values corresponding to that item. The weight of a facet value is calculated as the order of the corresponding facet multiplied by the order of the facet value. We also multiply this value with the frequency the user used the specific facet value in his queries. So, an item which has facet values that the user has selected more frequently and are more important will have a bigger value. This way, we try to predict the next facet and facet value for the user's query, considering the user's history.

We combine all the ranks mentioned above to obtain a final rank for the product and then we sort the results list in the descending order of this rank.

We chose to personalize the starting page of a user too. For this, we use the facet values that were the most frequently used by the user. We make a query with the first three of those facets and we display the sorted results list. The facet values used will not be checked, so the user doesn't know that we used them to generate the first displayed products. This way we try to place the users near their relevant items before they begin to formulate a query.

3.4 Backend aspects

The main concerns when talking about backend aspects of a faceted search system are the following: determining the results set for a query, meaning the items that match the query constraints, and computing the counts for each facet value, meaning the number of items in the result set that match the respective facet value.

To accomplish these tasks, our implementation combines the inverted index method and the bit-vectors method described in a previous chapter, 2.5.4 Backend aspects. For each facet-value pair we retain a bit-vector that represents the items that match the specific facet value pair. Every bit in a vector represents a product. The value of a specific bit is 1 if the product matches the facet-value pair.

The use of bit-vectors is a scalable and efficient way of refining the results list, but especially of calculating the facet counts. The bit-vectors are constructed when the products are added or edited. The number of these operations is much lower than the number of search operations, so we considered that the search operation should be optimized, even with the cost of adding extra computation when adding or editing a product.

When selecting a facet value as part of the current query, we can refine the results list in the following way: we AND the bit-vector corresponding to the current query (the current results list) and the bit-vector corresponding to the newly added facet value. This practically represents the intersection between the collection of products that match the current query and the collection of products that match the newly added facet value. We obtain a new bit-vector that represents the products contained in the new results set. This method is faster than iterating over all the products in the results list and verifying if a product matches the newly added facet value.

To compute the counts of every facet value, meaning the number of items in the results list that match the respective facet value, we do the same thing: AND the bit-vector of the results list and the bit-vector of the facet value. Then, we calculate the cardinality of the resulting bit-vector, meaning we count the bits with the value 1. The resulting number represents the number of products in the result set that match the current facet value. The use of bit-vectors is especially efficient and scalable for this operation, because the alternative is to iterate through the results list for every facet value in the classification. When the number of facet values increases, this can become really inefficient.

Since we offer the possibility of editing a product, therefore changing its facets and facet values, we also need a fast way of determining what facets and facet values a product has. For this, we also use a bit-vector, for every product, that represents the facet value pairs that the product matches.

3.5 Frontend design

The user interface of a faceted search system is an important part of the system, because it should help the user explore a large collection of items and navigate as easy and as fast as possible to the items the user is interested in. The interface shouldn't overwhelm or confuse the user and it shouldn't loose the users attention.

There are many aspects to be considered when designing a faceted search interface, aspects that we mentioned in a previous chapter, 2.5.5 Frontend design. In the following paragraphs, we will specify our choices for every of that aspects.

Placing the facets and results

We choose to place the facets in a panel to the left of the results area, so they are visible to the user all the time. Seeing the facet area all the time may determine the user to refine the results even more, making his search more efficient. The results area occupies the most of the page, making the user's navigation through the list as easier as possible.

Organising the facets and facet values

The number of facets and facet values is usually rather large, since the faceted search is used to explore a large collection of items. We cannot display all the facets and facet values at any time, because we risk to overwhelm and confuse the user. We chose to display only the first five facets by default and we give the user the possibility to expand the list of facets, by clicking on a "show more" link.

The default displayed facets are the most important facets for the current user, since we choose the first five facets from the order list of facets. The list is sorted in the descending order of the ranks assigned to facets, as we already discussed in the previous chapter 3.3 Personalization by using the user's profile.

The same thing applies to the facet values. For each facet, we only display by default the first five values and we give the user the possibility of expanding that list.

Another important aspect is the fact that we don't display facets or facet values that can lead to dead ends, meaning that there are no products in the results list that match them.

4. Results

Our implementation proposes an interaction with the users of the faceted search system through a web interface, since the faceted search system is going to be used for online stores.

As we mentioned before, our system takes into consideration two different types of users: the privileged users (the ones who manage the products and the faceted classification) and the normal users (the ones who use the faceted classification to explore the collection and find the items they are looking for).

The normal users will have access only to the faceted search page, which is the main page of our proposed faceted search system. They can use this page to explore the collection and find the items they want. The design of this page was built considering all the frontend aspects that we mentioned in the previous chapter.

The privileged users will have access to the following pages of the web interface:

1. a page where they can see and manage all the products in the collection. This page will look similar to the faceted search page a normal user can see, but will have some extra features that allow a privileged user to remove and modify products. We decided to use the same faceted search system to manage the products, because the system is built for large collections of items and managing those items using only a big list and no refining mechanism at all would be an almost impossible task.
2. a page where a privileged user can add or modify a product. This page contains fields for adding/editing product's details, as well as for adding/editing the facets and facet values for the respective product. The proposed page is just an example of page for generic products. Any online store can create its own page containing the product's details needed for its products.
4. a page where a privileged user can build/modify the faceted classification. New facet and facet values can be added, others can be removed. The order of the facets and facet values can be changed by dragging and dropping the respective facet/facet value to its desired location. The facets and facet values terms can also be modified.

This pages constitute the proposed adaptive faceted search system that can be used by any online store. Our method of creating the faceted classification allows any person with no technical knowledge to use the system and build the faceted hierarchy. This leads not only to savings in the budget allocated to build the faceted search for the online store, but also to savings in the time allocated to build and maintain the faceted classification.

The faceted search page is more interactive than the existing faceted search pages. It offers the users the opportunity to adapt their faceted classification by themselves or to use the automatic adaptive component of the system. It also uses different kind of rankings for the products, explicit (given by the users) and implicit rankings (computed by the system) to adapt the results list to the current user.

5. Conclusion

Faceted search is the most used search method in e-commerce. Online stores use this type of search due to its advantages over the other types of search. The main advantages of the faceted search are: it allows exploration of an unknown dataset, provides a guided means to navigate, or drill, and in any order, facets improve findability and are effective because they allow retrieval on whichever attributes of the item are important to the person who is searching.

We proposed a faceted search system that can be used for any online store, with any type of products. The method we use to create and maintain the faceted classification saves time and effort, as a nontechnical person can do it, in a very simple way.

An important part of our proposed system is the adaptability component. The goal of faceted search in general is to help users find the items they want or explore the collection of items. Every user is different, so a pre-defined and unique interface may not serve every user adequately. Our system adapts the order of the facets and their values to each user’s needs and also gives the users the possibility to establish their own order. We also adapt the results list using different types of product ranking: explicit ranking (given by the user; global and per user) and implicit ranking (computed by the system, based on users actions and history).

The research on this subject can be continued and some improvements and working directions include generating the facets and facet values automatically, using tools that can solve problems like natural language processing, synonymy and polysemy, combining the faceted search with textbox search and also improving the mechanisms for the facets and results personlaization.

References

[1] *Faceted search.*
http://en.wikipedia.org/wiki/Faceted_search

[2] William Denton, *How to make a faceted classification and put it on the web,* Nov. 2003.
http://www.miskatonic.org/library/facet-web-howto.html

[3] Ilknur Celik, Fabian Abel, Patrick Siehndel, *Adaptive Faceted Search on Twitter, 2011.*
http://ceur-ws.org/Vol-730/paper4.pdf

[4] Eyal Oren, Renaud Delbru, Stefan Decker, *Extending faceted navigation for RDF data, 2006.*
http://ir.library.nuigalway.ie/xmlui/bitstream/handle/10379/521/iswc2006-facets.pdf?sequence=1

[5] *Colon Classification.*
http://en.wikipedia.org/wiki/Colon_classification

[6] *Bliss Bibliographic Classification.*
http://en.wikipedia.org/wiki/Bliss_bibliographic_classification

[7] Jonathan Koren, Yi Zhang, Xue Liu, *Personalized Interactive Faceted Search,* 2008.
http://wwwconference.org/www2008/papers/pdf/p477-korenA.pdf

[8] Ajith Kodakateri Pudhiyaveetil, Susan Gauch, Hiep Luong, Joshua Eno, *Conceptual Recommender System for CiteSeerx,* 2009.
http://citeseer.uark.edu/projects/citeseerX/papers/recommender%20system.pdf

[9] *tf-idf* (term frequency–inverse document frequency).
http://en.wikipedia.org/wiki/Tf%E2%80%93idf

[10] Giovanni Maria Sacco, Yannis Tzitzikas, *Dynamic Taxonomies and Faceted Search,* 2009.
http://books.google.ro/books?id=KS0Y2UMC7XkC&pg=PA104&lpg=PA104&dq=personalized+faceted+search&source=bl&ots=r0yo9yvSqQ&sig=hb1EcV2WwMU8jWnGTGsogjl81t8&hl=en&sa=X&ei=clojUffGPIrotQbe3oDgCw&redir_esc=y#v=onepage&q=personalized%20faceted%20search&f=false

[11] Glenda Browne, *Faceted Classification,* 2003.
http://webindexing.biz/faceted-classification/

[12] *Faceted Metadata Search and Browse.*
http://www.searchtools.com/info/faceted-metadata.html

[13] *Calais (Reuters product).*
http://en.wikipedia.org/wiki/Calais_(Reuters_product)

[14] *Calais.*
http://www.opencalais.com/

[15] *General Architecture for Text Engineering.*
http://en.wikipedia.org/wiki/General_Architecture_for_Text_Engineering

[16] *GATE.*
http://gate.ac.uk/

[17] *Mallet (software project).*
http://en.wikipedia.org/wiki/Mallet_(software_project)

[18] *Mallet.*
http://mallet.cs.umass.edu/index.php

[19] Wisam Dakka, Panagiotis G. Ipeirotis, *Automatic Extraction of Useful Facet Hierarchies from Text Databases, 2008.*

http://www.ipeirotis.com/wp-content/uploads/2012/01/icde2008.pdf

[20] *WordNet.*
http://en.wikipedia.org/wiki/WordNet

[21] Wisam Dakka, Rishabh Dayal, Panagiotis G. Ipeirotis, *Automatic Discovery of Useful Facet Terms,* 2006.
http://www.ipeirotis.com/wp-content/uploads/2012/01/sigir_faceted2006.pdf

[22] Wisam Dakka, Panagiotis G. Ipeirotis, Kenneth R. Wood, *Automatic Construction of Multifaceted Browsing Interfaces,* 2005.
http://www.ipeirotis.com/wp-content/uploads/2012/01/cikm2005.pdf

[23] *Subsumption hierarchies, taxonomies and generalisation.*
http://coral.lili.uni-bielefeld.de/DATR/inherlexweb/node7.html

[24] *Controlled vocabulary.*
http://en.wikipedia.org/wiki/Controlled_vocabulary

[25] Daniel Tunkelang, *Faceted Search,* 2009.
http://disi.unitn.it/~bernardi/Courses/DL/faceted_search.pdf

[26] Daniel Tunkelang, *Dynamic category sets: An approach for faceted search,* 2006.

https://www.researchgate.net/publication/228498406_Dynamic_category_sets_An_approach_for_faceted_search

[27] Osma Suominen, Kim Viljanen, Eero HyvÄnen, *User-Centric Faceted Search for Semantic Portals,* 2007.

http://rd.springer.com/content/pdf/10.1007%2F978-3-540-72667-8_26

[28] G. W. Furnas, T. K. Landauer, L. M. Gomez, S. T. Dumais, *The Vocabulary Problem in Human-System Communication,* 1987.
http://courses.washington.edu/info320/au11/readings/Week3.Furnas.et.al.1987.The.vocabulary.problem.in.human-system.communication.pdf

[29] Jeffery Callender, Peter Morville, *Design Patterns: Faceted Navigation,* 2010.

http://alistapart.com/article/design-patterns-faceted-navigation

[30] *Inverted Index.*
http://en.wikipedia.org/wiki/Inverted_index

[31] Anne Schuth, Maarten Marx, *Fast Faceted Search in XML,* 2011.
http://ilps.science.uva.nl/PoliticalMashup/uploads/2011/02/fast-faceted-search.pdf

[32] *Bit array.*
http://en.wikipedia.org/wiki/Bit_array

[33] Peter Van Dijck, Introduction to XFML.
http://www.xml.com/pub/a/2003/01/22/xfml.html?page=2

[34] *Apache Solr.*
http://en.wikipedia.org/wiki/Apache_Solr

[35] *CiteSeer.*
http://en.wikipedia.org/wiki/CiteSeer

[36] *Apache Solr.*
http://lucene.apache.org/solr/