FleXplorer: A Framework for Providing Faceted and Dynamic Taxonomy-based Information Exploration

Yannis Tzitzikas, Nikos Armenatzoglou, Panagiotis Papadakos
Institute of Computer Science, FORTH-ICS, Crete, Greece, and
Department of Computer Science, University of Crete, Greece
{tzitzik | armenan | papadako}@ics.forth.gr

Abstract

Faceted and dynamic taxonomies are increasingly used nowadays in a plethora of applications. For developing user interfaces grounded on this interaction paradigm, it is advantageous to have a framework that enables the manipulation of the underlying information structure and provides the basic functionalities required. This paper introduces a formal model that captures faceted materialized taxonomies and the associated interaction-related notions. Subsequently, it discusses the current design (and implementation) of a general purpose framework grounded on this formal model. Finally, the paper reports some preliminary experimental and empirical results from using this framework and relational DBMSs.

1. Introduction

Faceted and dynamic taxonomies are used more and more nowadays in a plethora of application domains, and recently also in general purpose Web search engines[1]. In brief, faceted metadata search engines can switch easily between searching and browsing and allow users to see exactly the options that are available at any time. Features for faceted metadata search include[2]:

- Display of current results in multiple categorization schemes
- Display of populated categories only, that is categories leading to non-empty results
- Display a count of the contents of each category, informing the user of the number of results he/she will get if this category is selected
- Generation of groupings of results on the fly, based on metadata terms, such as size, price or date.

Examples of applications of faceted metadata-search include: e-commerce (e.g. ebay), library and bibliographic portals (e.g. DBLP), museum portals (e.g. MuseumFinland [5]), mobile phone browsers (e.g. FaThumb[6]), yellow pages portals (e.g. Veturi [8]), attempts to apply it over Semantic Web (e.g. [4, 7, 9]), general purpose web search engines (e.g. Google Base), and interaction frameworks (e.g. mSpace[12]).

To build applications according to this interaction paradigm, it would be advantageous to have a framework that (i) enables accessing and manipulating the information space (facets, taxonomies, object indices) and (ii) provides all functionalities required for building a user interface, e.g. functionalities for supporting zooming-in (or *iterative thinning*) and zooming-out, as well other auxiliary functions for exploring the information space. For instance, a human user (developer) could use this framework to define the desired facets and taxonomies or for importing existing taxonomies. Additionally, a user could use it for providing faceted access to a corpus of metadata records or to a structured source. Moreover it could be used by tools that mine facets and terms, e.g. [1, 2]. In general, we could say that such a framework can serve as the middleware between the presentation layer and the underlying information sources.

2. A Model for Facet-based Exploration

This section introduces a formal model aiming at capturing all key notions appearing in [11], [14], and [3]. Table 1 introduces basic notions and notations, like terms, terminologies, taxonomies, faceted taxonomies, interpretations and materialized faceted taxonomies (for details refer to [14, 13]). An example of a materialized faceted taxonomy, i.e. a faceted taxonomy accompanied by a set of object indexes, is shown in Figure 1.

[1] e.g. Google Base (http://base.google.com/)
[2] Source: http://www.searchtools.com/info/faceted-metadata.html

Name	Notation	Definition
terminology	$\mathcal{T}$	a set of names, called *terms* (they capture both categorical and numeric values)
subsumption	$\leq$	a partial order (reflexive, transitive and antisymmetric)
taxonomy	$(\mathcal{T}, \leq)$	$\mathcal{T}$ is a terminology, $\leq$ a subsumption relation over $\mathcal{T}$
broaders of t	$Br(t)$	$\{ t' \mid t < t' \}$
narrowers of t	$Nr(t)$	$\{ t' \mid t' < t \}$
direct broaders of t	$Br^{(1)}(t)$	$minimal_<(\{ t' \mid t < t' \})$
direct narrowers of t	$Nr^{(1)}(t)$	$maximal_<(\{ t' \mid t' < t \})$
faceted taxonomy	$\mathcal{F}$= $\{F_1, ..., F_k\}$	$F_i = (\mathcal{T}_i, \leq_i)$, for $i = 1, ..., k$ and all $\mathcal{T}_i$ are disjoint
compound term over $\mathcal{T}$	s	any subset of $\mathcal{T}$ (i.e., any element of $\mathcal{P}(\mathcal{T})$)
object domain	Obj	any denumerable set of objects
interpretation of $\mathcal{T}$	I	any function $I : \mathcal{T} \rightarrow 2^{Obj}$
model of $(\mathcal{T}, \leq)$ induced by I	$\bar{I}$	$\bar{I}(t) = \cup\{I(t') \mid t' \leq t\}$
materialized faceted taxonomy	$(\mathcal{F}, I)$	$\mathcal{F}$ is a faceted taxonomy $\{F_1, ..., F_k\}$, I is an interpretation of $\mathcal{T}$= $\bigcup_{i=1,k} \mathcal{T}_i$
extension of s in I and in $\bar{I}$	$I(s)$, $\bar{I}(s)$	$I(s) = \cap\{ I(t) \mid t \in s\}$ and $\bar{I}(s) = \cap\{ \bar{I}(t) \mid t \in s\}$

Table 1. Basic notions and notations

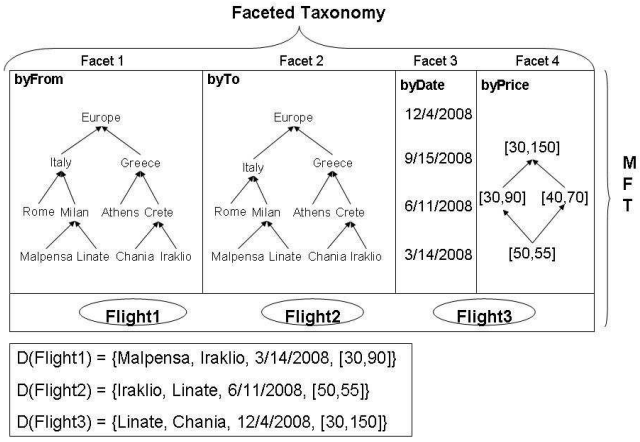

Figure 1. Example of a Materialized Faceted Taxonomy

Each facet F_i is associated with a name (a String) and a taxonomy. The same taxonomy may be associated with more than one facets (for instance, for indexing flights we may have two facets, named "from" and "to", associated with the same taxonomy "Location"). However, by prefixing the name of each term with the facet name, we may assume that all facet terminologies are disjoint (as stated in Table 1).

For our purposes, we need to adopt a minimal query language. A query is a compound term s (i.e. a conjunction of terms) and its answer is the set of objects $\bar{I}(s)$ (as defined in Table 1). Of course, boolean expressions and more complex query operators can be straightforwardly supported.

As interaction is of prominent importance, now we define formally the notions needed for describing interaction. Any subset of $\mathcal{T}$ is a possible *focus*. For reasons of minimality, we shall hereafter consider foci that are *redundancy free*. A focus ctx (i.e. $ctx \subseteq \mathcal{T}$) is redundancy free if $ctx = minimal_<(ctx)$. For example, $ctx = \{Greece, Athens\}$ is not redundancy free because $minimal_{\preceq}(ctx) = \{Athens\}$. The *content of a focus* ctx, is the set of objects $\bar{I}(ctx)$. We could also refine this notion and distinguish the *shallow* content $I(ctx)$, from the *deep* content $\bar{I}(ctx)$.

2.1. Zooming-in and Zooming-out

Now we will introduce elements allowing the refinement of a focus. To this end we introduce the notion of *zoom-in points*. A zoom-in point is actually a term that indicates where the user could zoom in. When building a GUI, an area is usually dedicated to each facet and the zoom-in points with respect to a facet F_i are actually those terms of T_i that should be shown in that area.

Given a focus ctx, we can define its *projection* to a facet F_i, denoted by ctx_i, as follows $ctx_i = ctx \cap T_i$.

Now we will define the zoom-in points with respect to a particular facet F_i. Consider a focus ctx and suppose that $ctx_i \neq \emptyset$. The *candidate zoom-in points* with respect to F_i, denoted by $CZ_i(ctx)$, are defined as:

$$CZ_i(ctx) = Nr^{(1)}(ctx_i)$$

The above definition can also be applied in cases where $|ctx_i| > 1$, assuming that $Nr^{(1)}$ is defined for sets of terms[3]. If $ctx = \emptyset$ then we assume that $ctx_i = \top_i$ where $\top_i$ denotes an auxiliary (internal and invisible) element standing for the top element of the taxonomy T_i.

From the candidate zoom-in points we now filter out those that yield an empty content. The (good) *zoom-in*

[3]If $S \subseteq \mathcal{T}$ then $Nr^{(1)}(S) = \cup_{t \in S} Nr^{(1)}(t)$

points are defined as:

$$Z_i(ctx) = \{t \in CZ_i(ctx) \mid \bar{I}(ctx) \cap \bar{I}(t) \neq \emptyset\}.$$

So $Z_i(ctx)$ comprises the terms of T_i that should be shown in the GUI area dedicated to facet F_i if the user focus is ctx. For example, assuming the example of Figure 1, we have:

$$\begin{aligned} Z_1(\{Greece_1, Italy_2\}) &= \{Crete_1\} \\ Z_2(\{Greece_1, Italy_2\}) &= \{Milan_2\} \\ Z_3(\{Greece_1, Italy_2\}) &= \{6/11/2008\} \\ Z_4(\{Greece_1, Italy_2\}) &= \{[30, 150]\} \\ Z_1(\{Italy_1, Crete_2\}) &= \{Milan_1\} \\ Z_2(\{Italy_1, Crete_2\}) &= \{Iraklio_2, Chania_2\} \end{aligned}$$

When the user selects a zoom-in point t, then the current focus is updated, i.e. $ctx' = ctx \cup \{t\}$ (specifically, $ctx' = minimal_<(ctx \cup \{t\})$). Subsequently, all new zoom-in points are computed and presented.

The user can also *zoom out* by deselecting any term t of the corresponding focus. In that case t is replaced by its direct broader term(s) i.e. by $Br^{(1)}(t)$. Alternatively, the user may remove t without replacing it with any other term.

2.2. "Side" Zooming

Now we introduce another kind of zoom-in points. This kind of points is useful for taxonomy-based sources that: (a) comprise more than one taxonomy (i.e. they are faceted taxonomies), or (b) comprise a single taxonomy that is not a tree and we have single classification (i.e. an object is indexed with exactly one term), or (c) comprise a single taxonomy and we have multiple classification (i.e. an object can be indexed with more than one terms). Two terms are (extensionally) *related*, denoted by $R(t, t')$ if $\bar{I}(t) \cap \bar{I}(t') \neq \emptyset$. Notice that this relation is reflexive and symmetric (but not transitive). We will use $RT(t)$ to denote the terms that are related to t, i.e. $RT(t) = \{\ t' \mid R(t, t')\}$. Notice that if $t \leq t'$ and $\bar{I}(t) \neq \emptyset$, then it holds $R(t, t')$. For this reason we introduce the notion of *purely related terms*. The set of *purely related terms* of a term t, denoted by $PR(t)$, is defined as $PR(t) = RT(t) - (Nr(t) \cup Br(t))$, so it comprises the related terms of t except those which are broader or narrower than t.

We can now define the *related zoom-in points* w.r.t. a facet F_i, denoted by $ZR_i(ctx)$, as follows: $ZR_i(ctx) = maximal_<(PR(ctx_i))$. Note that for each t in $ZR_i(ctx)$ it holds $\bar{I}(ctx) \cap \bar{I}(t) \neq \emptyset$. Also note that if objects are indexed by at most one term from a facet F_i and F_i is a tree, then $ZR_i(ctx) = \emptyset$ for any ctx. In our running example we have $RZ_4(\{Milan_1, Iraklio_2, [30, 90]\}) = \{[40, 70]\}$.

2.2.1 Presentation and Ranking of Zoom-in points

Each zoom-in point t is usually accompanied by a number that indicates the number of objects that will be obtained if the user selects that zoom-in point. Specifically that number equals the cardinality of the set $\bar{I}(ctx) \cap \bar{I}(t) = \bar{I}(ctx \cup \{t\})$, which is certainly greater than zero (if $t \in Z_i(ctx)$ or $t \in ZR_i(ctx)$).

The zoom-in points can be ranked according to various criteria like, number of results if selected, user preferences, popularity, usage workload, etc.

In addition, other criteria can be employed to suppress the visibility of some points. For instance, we may hide those zoom-in points leading to contexts with content size below a predefined threshold, or we may decide to present only the top-K zoom-in points for each facet.

3. `FleXplorer`

In brief, `FleXplorer` allows managing (creating, deleting, modifying) terms, taxonomies, facets and object descriptions. It supports both finite and infinite terminologies (e.g. numerically-valued attributes). In addition it supports explicitly and intentionally defined taxonomies. Examples of the former include classification schemes and thesauri, while examples of the latter include hierarchically organized intervals (based on the *cover* relation).

The implementation is in Java, so the predefined ordering of built-in types (e.g. of `int, float, String`), as well as the customized ordering defined for user-defined Java classes (e.g. through the `comparable` interface) is exploited. To allow intentionally defined partially ordered domains, a `partiallyComparable` interface has been introduced and can be used by the developer. The framework also supports parametric types.

Regarding, interaction, the framework provides methods for setting (resp. computing) the focus (resp. zoom-in points). In addition, the framework allows materializing on demand the relationships of a taxonomy, even if the domain is infinite and intentionally defined (e.g. between numbers, intervals, etc), as this can accelerate the computation of $Nr^{(1)}(t)$ at the cost of extra main memory space (to keep the relationships). Regarding deployment, the framework can be used either at the server side or at client side, depending on the case.

4. Experimental and Empirical Results

`Mitos` (formerly known as grOOGLE)[4] is a prototype Web search engine that is being developed by the Department of Computer Science of the University of Crete [10]. `FleXplorer` is used by `Mitos` for offering general purpose browsing and exploration services. Currently, only some general and content-independent facets are supported. Specifically, and on the basis of the top-L answer of each

[4] http://groogle.csd.uoc.gr:8080/mitos/

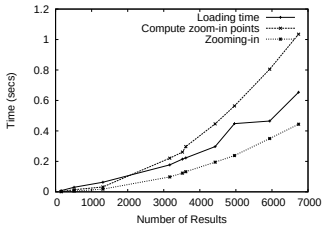

Figure 2. Experimental results of FleXplorer

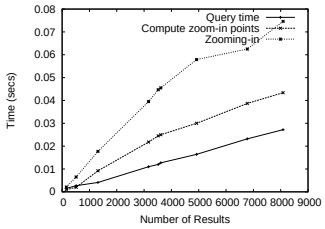

Figure 3. Experimental Results on DBMS

submitted query, the following facets/taxonomies are created and offered to users: (a) web domain, a hierarchy is defined (e.g. $csd.uoc.gr < uoc.gr < gr$), (b) format type (e.g. pdf, html, doc, etc), no hierarchy is created in this case and (c) encoding of a web page (e.g. utf-8, iso-8859-1). Notice that each page in the top-L answer is (straightforwardly) classified to one term of each of the above taxonomies.

Figure 2 shows the time to load to FleXplorer the top-L answer and the time to compute the zoom-in points (for three facets) and the content of the new focus (after the selection of a zoom-in point). As we can see for $L \leq 6700$ the time to load to FleXplorer the top-L answer is less than a second under the Mitos setting (Pentium IV 3.2GHz, 2GB RAM, Debian).

In the future we plan to extend the suite of available facets based on:

• last modification date of a web page. A hierarchy (year, month, day, time) will be created.

• content-based results clustering. Mitos already supports real-time results clustering. The derived clusters can be used to form an additional facet.

• user-defined facets. Users could define their own taxonomy. In this case any pair (n, q) where n is a user-provided name and q is any Mitos-query could be considered as a term.

• mining techniques. Techniques that will mine facets and taxonomies from the entire index of Mitos will be investigated. This can be considered as a special case of clustering.

As the index of Mitos is based on a DBMS (specifically PostgreSQL 8.3), an alternative approach would be to use directly SQL (as the above primitive facets correspond to columns of the schema), but this approach is not straightforwardly applicable in case of hierarchically organized terms. Figure 3 shows the corresponding results (for various result sizes) for computing the zoom-in points for only one facet (whose terms are not hierarchically organized). Notice that the time to compute the contents of the new focus is higher than the time to compute those of the original focus (because the corresponding query is longer). In general, this approach is very fast too.

However FleXplorer is able to handle also hierarchically organized values. To estimate the efficiency of SQL over such values, we created a synthetic dataset having the following schema (and instance size):
paper(pid, title, year, venue) (10^5 tuples)
author(authorId, authorName) (4×10^4 tuples)
paperAuthors(pid, authorId) (2.2×10^5 tuples)
subjectHierarchy(stId, name, parentId) (3906 tuples)
paperSubjects(stId, pid) (10^5 tuples)
where **subjectHierarchy** forms a balanced and complete tree with degree 5 and depth 5. Each paper is associated with one randomly selected subject term (that is a leaf) and with 1 to 4 randomly selected authors. All fields of the tables have been indexed with B-Trees and the size of the database is 30.1 MB (the indexes occupy 17.2 MB). In order to run the experiments we have installed PostgreSQL 8.3 (with shared buffers parameter set to 1 GB) on a Pentium IV machine with 3GHz CPU and 1 GB RAM.

Figure 4(a) shows: (t_a) the time for computing the answer of a query comprising one subject term from various term depths, (t_b) the time to compute the zoom-in points with respect to the venue attribute, (t_c) the time to compute the content of the new focus (we have selected one zoom-in point from venue facet). Furthermore, the cost t_a is included in both t_b and t_c, since we re-compute the results. The reported times are the average of 20 different runs of 5 randomly selected subject terms for each depth. Figure 4(b) shows the corresponding average result sizes.

We conclude that the query times increase, compared to the query times in Figure 3, for the same number of results. This was an expected result, since to support hierarchically organized values using the DBMS, more complicated queries had to be issued.

Here we report the results **for larger data sets**, i.e. for databases that do not fit in main memory. We used the following database:
paper(pid, title, year, venue) (2×10^6 tuples)
author(authorId, authorName) (5×10^5 tuples)
paperAuthors(pid, authorId) (5×10^6 tuples)
subjectHierarchy(stId, name, parentId) (111.111 tuples)
paperSubjects(stId, pid) (2×10^6 tuples)
subjectHierarchy2(stId, name, parentId) (111.111 tuples)
paperSubjects2(stId, pid) (2×10^6 tuples)

Tables **subjectHierarchy** and **subjectHierarchy2** form a balanced and complete tree with degree 10 and depth 5.

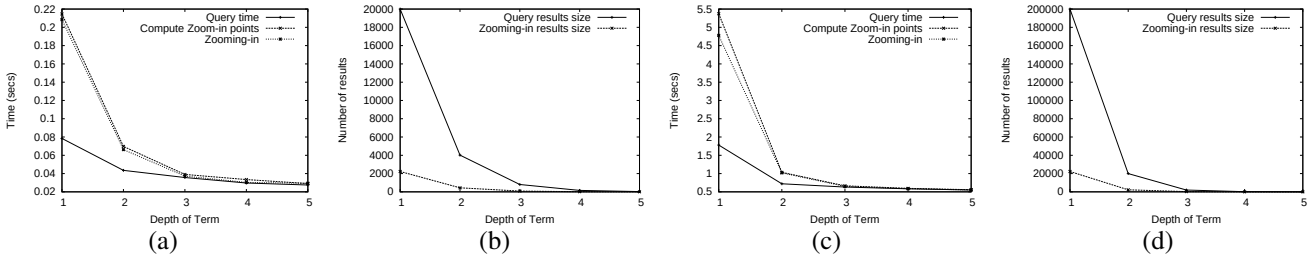

Figure 4. Experimental results on synthetic databases.

Again, each paper is associated with one randomly selected leaf subject term from each of the two hierarchies and with 1 to 4 randomly selected authors. All fields of the tables have been indexed with B-Tree access method and the size of the database is 1.24 GB (the indexes occupy 718 MB). The experiments run on the same machine as above but with the shared buffers parameter of postgreSQL set to 1 GB. Figure 4(c) shows the times of the same experiment as Figure 4(a) in a larger dataset. The reported times are the average times of 40 different runs for 10 randomly selected subject terms for each depth. Figure 4(d) shows the corresponding average result sizes. The aforementioned times were gathered using Java, meaning that the overhead of the JDBC driver is also included. This overhead also exists in the `Mitos` engine, since it is also written in Java.

Summarizing, when the volume of data increases and performance of the DBMS approach degrades. We should also mention that the DBMS approach is feasible only if we a-priori know the depth of the taxonomies involved. For these reasons in future we plan to experiment with an hybrid approach where all hierarchically organized values are loaded to `FleXplorer` while the rest may reside to a DBMS.

5. Concluding Remarks

This paper presented the design of a framework for aiding the development of general purpose information exploration applications. Some preliminary experimental and empirical results were presented. Future work (apart from the issues already mentioned) includes extending `FleXplorer` so that it can act as a mediator over remote information sources.

References

[1] W. Dakka, R. Dayal, and P. Ipeirotis. "Automatic Discovery of Useful Facet Terms". *SIGIR Faceted Search Workshop*, Aug. 2006.

[2] W. Dakka, P. G. Ipeirotis, and K. R. Wood. "Automatic Construction of Multifaceted Browsing Interfaces". In *Procs of the 14th ACM CIKM '05*, pages 768–775, New York, NY, USA, Nov. 2005.

[3] S. Ferre and O. Ridoux. Logical information systems: from taxonomies to logics. In *Procs of FIND'2007 (at DEXA '07)*, pages 212–216, Regensburg, Germany, 3-7 Sept. 2007.

[4] M. Hildebrand, J. van Ossenbruggen, and L. Hardman. "/facet: A Browser for Heterogeneous Semantic Web Repositories". In *Procs of ISWC '06*, pages 272–285, Athens, GA, USA, Nov. 2006.

[5] E. Hyvönen, E. Mäkelä, M. Salminen, A. Valo, K. Viljanen, S. Saarela, M. Junnila, and S. Kettula. "MuseumFinland – Finnish Museums on the Semantic Web". *Journal of Web Semantics*, 3(2):25, 2005.

[6] A. K. Karlson, G. G. Robertson, D. C. Robbins, M. P. Czerwinski, and G. R. Smith. "FaThumb: a Facet-Based Interface for Mobile Search.". In *Procs of the Conference on Human Factors in computing systems, CHI'06*, pages 711–720, New York, NY, USA, Apr. 2006.

[7] E. Mäkelä, E. Hyvönen, and S. Saarela. "Ontogator - A Semantic View-Based Search Engine Service for Web Applications.". In *Procs of ISWC '06*, pages 847–860, Athens, GA, USA, Nov. 2006.

[8] E. Mäkelä, K. Viljanen, P. Lindgren, M. Laukkanen, and E. Hyvönen. Semantic yellow page service discovery: The veturi portal. *In poster paper at ISWC '05*, Nov. 2005.

[9] E. Oren, R. Delbru, and S. Decker. "Extending Faceted Navigation for RDF Data". In *Procs of ISWC '06*, pages 559–572, Athens, GA, USA, Nov. 2006.

[10] P. Papadakos, G. Vasiliadis, Y. Theoharis, N. Armenatzoglou, S. Kopidaki, Y. Marketakis, M. Daskalakis, K. Karamaroudis, G. Linardakis, G. Makrydakis, V. Papathanasiou, L. Sardis, P. Tsialiamanis, G. Troullinou, K. Vandikas, D. Velegrakis, and Y. Tzitzikas. "The Anatomy of Mitos Web Search Engine", http://arxiv.org/abs/0803.2220, Mar. 2008.

[11] G. M. Sacco. "Dynamic Taxonomies: A Model for Large Information Bases". *IEEE Transactions on Knowledge and Data Engineering*, 12(3), May 2000.

[12] M. Schraefel, M. Karam, and S. Zhao. "mSpace: Interaction Design for User-Determined, Adaptable Domain Exploration in Hypermedia". In *Procs of Workshop on Adaptive Hypermedia and Adaptive Web Based Systems*, pages 217–235, Nottingham, UK, Aug. 2003.

[13] Y. Tzitzikas and A. Analyti. "Mining the Meaningful Term Conjunctions from Materialised Faceted Taxonomies: Algorithms and Complexity". *Knowledge and Information Systems Journal (KAIS)*, 9(4):430–467, May 2006.

[14] Y. Tzitzikas, A. Analyti, N. Spyratos, and P. Constantopoulos. "An algebra for specifying valid compound terms in faceted taxonomies". *Data & Knowledge Engineering*, 62(1):1–40, 2007.