# Faceted Approach To Diverse Query Processing

Alessandro Agostini
Department of Computer Science
Prince Mohammad Bin Fahd University
Al-Khobar, Saudi Arabia
aagostini@pmu.edu.sa

Devika P. Madalli, A.R.D. Prasad
Documentation Research and Training Centre
Indian Statistical Institute
Bangalore, India
{devika,ard}@drtc.isibang.ac.in

## ABSTRACT

This paper presents a formal framework for implementing a query refinement method. The method uses general principles of facet analysis. Two key notions are advanced and discussed: diversity and focus. Diversity refers to the information needs of a querying user; it is captured by the notion of 'facet'. A focus refers to how diversity is captured from the documents as organized by the user; it provides a kind of context to the user query. The method is situated within the formal framework of the smallest propositionally closed description logic $\mathcal{ALC}$, thereby betting that $\mathcal{ALC}$ provides us with a suitable SAT solver to implement a facet engine, which is the main component of our method.

## Categories and Subject Descriptors

H.3.7 [**Information Systems**]: Digital Libraries; I.2.4 [**Computing Methodologies**]: Knowledge Representation Formalisms and Methods

## General Terms

Design, Human Factors, Algorithms.

## Keywords

Query refinement, facet-based search, text-based search, context-based search, user issues, description logic.

## 1. INTRODUCTION

Classical libraries had systems that processed subjects or domains and built representations such as subject indices. Among these system, the Colon Classification System (CCS) first proposed by S.R. Ranganathan [20] is currently widely used by almost all Indian libraries. The CCS had enough contextual information in the method of facetisation and synthesis so that it formed a semantic formalisation of the domain scope of the library collections.

In order to digitize CCS and similar facet-based systems, Prasad and Guha [18] demonstrate the applicability of faceted schema in describing resources in web directories and annotating resources in digital libraries using SKOS/RDF representation to express DEPA strings, according to faceted theory by Ranganathan [20] and DEPA facet analysis [7]. On the other hand, current keyword-based querying methods does not use DEPA strings to represent web directories and annotating resources in digital libraries, so they seem inadequate to search over digital repositories organized according to CCS and similar faced-based classification systems.

For answers to be relevant, a user must ask the appropriate query in order to retrieve the desired information and fulfill the information need (IN). For keyword-based search this means that a high number of keywords is necessary to the user to narrow down the search according to her information need. This is due the semantic ambiguity of querying languages, often built upon natural language, as it is the case of keyword-based querying. Unfortunately, the query length of keyword-based search on average is reported to be short, with 90% of the queries being less than four keywords [12]. As a consequence, the ambiguity of the query is somewhat mirrored in the relative relevance of search results [32, 3]; diversity in search results arises [15] and query refinement by the user is often the only solution. To resolve such ambiguity some authors advanced the notion of 'context' in web search, see for instance [14, 10] and references cited therein. However, in contect-based solutions the user is often assumed to know how data and information are organized in the search domain. This is often hard to happen in real-world, distributed scenarios like the Web, due to large amounts of heterogeneous data organized in an unknow structure.

In this paper we present a formal framework wherein we define a method for the extraction of DEPA facets from a user query. The facets are then used to refine the original query for search and retrieval purposes. The method is aimed to suggest the user a list of facets that the user would hardly be aware of by simply typing a keyword-based query into a search engine, without any query context. These automatically suggested new facets can be used by the user, for instance by clicking on one of the new facets, to narrow down the search space by expanding the original user query with the suggested facet.

This paper is organized as follows. In Section 2 we define basic concepts related to facet analysis. In Section 3 we discuss the first step of our method. In Section 4 we build a formal faceted ontology to formalize the focused terms and contexts that we successively process, in Section 5, to produce new facets to be shown to the user for query refinement. After building the faceted ontology and defining the facet engine, in Section 6 we present the three different yet related querying methods we offer to the user; these are keyword-based, by focus, and on subject. In Section 7 we discuss related work. In Section 8 we conclude the paper.

## 2. FACETS ANALYSIS

Facet analysis is essentially a conceptual analysis of the subject matter, or the topical content of a concept into distinct divisions that together constitute a semantic description of the concept. In order to build the facet repository available to a user to refine a query, in this section we present some elements of facet analysis.

Our facets repository is organized around two main notions of the DEPA paradigm for facet analysis [6, 7]: subjects and facets. A *subject of a concept* is the topical content of the concept, that is, the concept's overall semantics, as defined by the combination of extensional and intensional semantics of the concept term. The definition can be extended to a query, which in its simplest form can be thought of as a finite sequence of concept terms; see subsections 6.1 and 6.3. A *facet* consists of a "group of terms derived by taking each term and defining it, per genus et differentiam, with respect for its parent class." [31, p. 12]. According to Ranganathan [20], each domain is made of distinct divisions or facets that are groups of mutually exclusive concepts and many such facets together constitute a domain. The notion of such facetization has been extended by Bhattacharyya [7] to subject indexing by representing content as a string of fundamental categories DEPA (Discipline, Entity, Property and Action) that are conceptually equivalent to 'facets'. To illustrate, we rely on the following two examples.

EXAMPLE 1. *Consider a document titled 'Improving EU labour market access for Rome'. DEPA facet analysis of the title leads to facets such as: Labour Market (Entity), Access (Action), Rome (Space - from commonly applicable facet schedules across domains). The facet 'Discipline' is extrapolated from faceted document representation, and it is 'Economics'.*

Note that in case a concept would be classified within more than one discipline, as a homonymous or synonymous concept, then all such different combinations of facets are taken into account and presented to the user for further refinement.

EXAMPLE 2. *Consider a document titled 'Treating Apple trees for bacterial disease in Trentino'.*[1] *DEPA facet analysis provides a classification of the document into the following facets: Agriculture (D), Apple Trees (E), Treating (A), Disease (P), and Bacterial (as 'Modifier' to P, cf. [6]).*

We are now ready to define the facet repository for a given context. A *facet repository for a context* $\mathcal{C}$ is the set

$$FR(\mathcal{C}) = \{\langle C : d, e, p, a\rangle \mid C \text{ has DEPA facets } d, e, p, a\},$$

where $C$ is a concept description in description logic $\mathcal{ALC}$ (see subsection 4.2) of a concept or subject of interest in context $\mathcal{C}$, and $d, e, p, a$ are, respectively, a Discipline, Entity, Property and Action in DEPA classification system.

EXAMPLE 3. *Consider the previous two examples. We can assume that 'Improving EU labour market access for Rome' is represented by a concept description* $C_1$*, and 'Treating Apple trees for bacterial disease in Trentino' is represented by a concept description* $C_2$ *in a context* $\mathcal{C}$*. The facet repository* $FR(\mathcal{C})$ *contains* $\langle C_1 : Economics, LabourMarket, p, Access\rangle$ *for* $p$ *is unspecified, and* $\langle C_2 : Agriculture, AppleTrees, Disease, Treating\rangle$.

## 3. FOCUSED TERMS FROM TEXT

In the present work, we apply facetization as a technique to combine extensional and intensional semantics of concepts viz. queries, or equivalently to disclose the subject of concepts and queries to the querying user, for the purpose of query refinement and search assistance. We implement facetization in two related steps: 1. we produce certain "focused terms" from documents organized in a polyhierarchy, and 2. from focused terms we produc new facets to be shown to the user for the purpose of query refinement. We present step 1 in subsections 3.1 and 3.2 in this section, and step 2 in sections 4 and 5.

### 3.1 Organization of documents

Although our method can be adopted as integral part of digital libraries systems, both for describing the documents collection and for faceted querying over the collection or the web, in this paper we assumed the method assists a *querying user* in query refinement. As the method in this specific application uses a textual collection of documents stored in the user's querying machine, we stipulate the following convention.

CONVENTION 1. *We denote the set of available documents to a querying user by* $\mathcal{D}$*. All available documents are textual, that is, they can be processed by text information retrieval techniques as the variant of a standard technique discussed in Section 3.*

Intuitively, the domain $\mathcal{D}$ of documents can be thought of as the set of all documents the querying user has classified and stored in the querying machine.

CONVENTION 2. *We assume that the querying user organizes documents in* $\mathcal{D}$ *by using a 'polyhierarchical classification', or polyhierarchy.*

A *polyhierarchical classification* is a hierarchical classification permitting some concept terms to be listed in multiple categories of a taxonomy, or branches of a hierarchy [16]. An example of polyhierarchy can be found in Figure 1. Note that what makes the hierarchical classification in Figure 1 be polyhierarchical is the concept term 'Apple'.

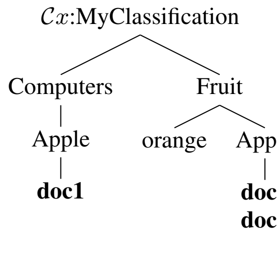

**Figure 1: A polyhierarchy, or polyhierarchical context $\mathcal{C}x$.**

A subset of documents is organized in 'contexts', each context be organized into related sets of documents. A *context* is a polyhierarchical classification composed by sets of documents, i.e., 'nodes' of the polyhierarchy, called *clusters*, and a relation over the nodes as defined by the polyhierarchy. Typical relations are the binary relations of subsumption, part-of, is-a, among others relations. Each cluster in a given context has a name composed by a finite sequence

[1] Trentino is a Province of the Italian North-east known for the Dolomites and for its quality production of red and yellow apples.

of words from a representation language, often a natural langiage thereby betting that clusters are named by a human—the querying user, who naturally applies her native language for clusters naming. A cluster's name in such representation language is referred to as *concept term*. A *concept* is a concept term provided with a semantics. Two kinds of semantics are provided to a concept term: an extensional semantics, defined over the documents in the cluster named by the concept term; and an intensional semantics, defined by the unique position of the concept term in a given 'focus'.

Contexts provide a way to define finite, ordered sequences of concept terms, each sequence called a *focus*. A *focus* consists of an ordered set of related concept terms, each concept term naming a cluster built upon the collection of documents in $\mathcal{D}$. Intuitively, a focus is a path of concept terms corresponding to a path in a given context. Figure 2 provides an example of both a context (left-hand side) and a focus (right-hand side). With reference to Figure 2, we write $\mathcal{C}x$:Fruit>Trentino>Apple to denote the focus named 'Apple' in the context $\mathcal{C}x$. In boldface are written two documents in the cluster 'Apple': **docRdoc** and **docGtxt**.

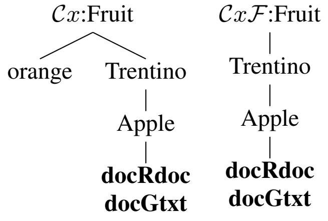

**Figure 2: An example of context (left) and focus (right).**

## 3.2 Concept terms grounded in documents

In this section, our goal is to automatically assign a 'label' to every cluster of a given context. Each cluster's label produced by Algorithm 1 below is a finite, simple concatenation of terms with maximum 'weight', extracted by using $\mathsf{Text}(\cdot)$. Formally, we proceed as follows.

Let $\mathsf{Text}(\cdot)$ be a text extraction function. In this paper, we refer to $\mathsf{Text}(\cdot)$ as a standard keywords extraction function, for instance see [25, Sec. 4]. Given a document $d$, $\mathsf{Text}(d)$ listes all the keywords in $d$, precisely, the most frequent 'tokens'. Applied to a document $d$, $\mathsf{Text}(\cdot)$ produces a set $\mathsf{Text}(d)$ of *terms* (or 'keywords'). Let $d$ be any document in $\mathcal{D}$. As terms are defined from documents, from now on we write $k \in \mathsf{Text}(d)$ to denote a generic term retrieved by using $\mathsf{Text}(\cdot)$ $d$. Given a document $d$, we rank a term $k \in \mathsf{Text}(d)$ by adapting IR standard TF/IDF ("Term Frequency / Inverse Document Frequency") method [22, 23] to deal with contexts and unique concept terms' *position*, i.e., focus, within a context. Observe that in the following, for a given context $\mathcal{C}$ we write '$C$ in $\mathcal{C}$' in place of '$C$ in $\mathcal{C}$' set of clusters' for every cluster $C$.

Let querying user **u** organizing a context $\mathcal{C}$, cluster $C$ in $\mathcal{C}$, and term $k \in \mathsf{Text}(d)$ for a document $d \in \mathcal{D}$ be given. We define the *weight of $k$ in $C$* as follows:

$$\mathsf{W}_{\mathbf{u}}[k, C] = (\sum_{d \in C} \mathsf{TF}[k, d]) \cdot log \frac{\mathit{Card}(F\mathcal{C})}{\mathsf{doCK}_{\mathbf{u}}[k]}, \qquad (1)$$

where $\mathsf{TF}[k, d]$ is the total number of occurrences of $k$ in $d$, so that $\sum_{d \in C} \mathsf{TF}[k, d]$ is the total number of occurrences of $k$ in $C$; $\mathit{Card}(F\mathcal{C})$ is the number of focuses in $\mathcal{C}$ with leaf $C$, and $\mathsf{doCK}_{\mathbf{u}}[k]$ is the total number of clusters in the set

$$\mathcal{C} \setminus \{C' \mid C' \neq C \text{ is a cluster in a focus in } \mathcal{C} \text{ with leaf } C\} \qquad (2)$$

which contain $k$. Intuitively, (1) says that, in order to represent the extensional semantics of a focus, the importance of a retrieved term for a cluster, i.e., the value of $\mathsf{W}_{\mathbf{u}}[k, C]$, is inversely proportional to the number of different focuses with $C$ as leaf which contain the term.

The label of a cluster $C$ is the most representative term or sequence of terms for the cluster. Now we want compute the label of all clusters of a given context. For doing this, we process all documents stored in each cluster by considering the position of each cluster in the context. To define the process formally, we rely on the following technical definition. Let context $\mathcal{C}$ organize (a subset of) documents in $\mathcal{D}$ and cluster $C$ in $\mathcal{C}$ be given. We define

$$\mathit{IR}(\mathcal{D}, \mathcal{C}, C) = \{k \in \mathsf{Text}(d) \mid d \in C, C \text{ in } \mathcal{C}\}. \qquad (3)$$

We expect that the label of cluster $C$ in (3) is the most representative term or sequence of terms in $\mathit{IR}(\mathcal{D}, \mathcal{C}, C)$. The most representative term among terms in $\mathit{IR}(\mathcal{D}, \mathcal{C}, C)$ is the term with the highest weight among all terms in $\mathit{IR}(\mathcal{D}, \mathcal{C}, C)$ according to weighting measure 1. Formally, a term $k$ in $\mathit{IR}(\mathcal{D}, \mathcal{C}, C)$ is the most representative for the cluster $C$ in $\mathcal{C}$, and we say that $k$ is a *label of* $C$, if $\mathsf{W}_{\mathbf{u}}[k', C] \leq \mathsf{W}_{\mathbf{u}}[k, C]$ for all terms $k'$ in $\mathit{IR}(\mathcal{D}, \mathcal{C}, C)$. A sequence $k_1, k_2, ...k_n$ of terms in $\mathit{IR}(\mathcal{D}, \mathcal{C}, C)$ is a label of $C$ if (a) $\mathsf{W}_{\mathbf{u}}[k_i, C] = \mathsf{W}_{\mathbf{u}}[k, C]$ for $i = 1, 2, ...n$, and (b) $k$ is a label of $C$.

LEMMA 1. *Every cluster $C$ organized by a querying user* **u** *in a context $\mathcal{C}$ has a label if and only if $C$ contains a document $d$ such that* $\mathsf{Text}(d)$ *is nonempty.*

To compute a label of every nonempty cluster $C$ of a given context $\mathcal{C}$, we exhibit an algorithm that produces the label $l_C$ of $C$; see Algorithm 1. Set $\mathit{IR} = \mathit{IR}(\mathcal{D}, \mathcal{C}, C)$.

**Algorithm 1** Context-based cluster labeling.

```
Input: $\mathcal{C}, \mathcal{D} \neq \emptyset$
foreach $C$ in $\mathcal{C}$ with $C \neq \emptyset$ do
  foreach $k \in \mathit{IR}(\mathcal{D}, \mathcal{C}, C)$ do
    compute $\mathsf{W}_{\mathbf{u}}[k, C]$ according to formula (1) od;
  compute $M = \{k \in \mathit{IR} \mid \forall k' \in \mathit{IR}, \mathsf{W}_{\mathbf{u}}[k', C] \leq \mathsf{W}_{\mathbf{u}}[k, C]\}$;
  Let $n$ be the cardinality of $M$;
  Let $\{k_1, k_2, \ldots, k_n\}$ be the lexicographical ordering of $M$;
  Set $l_0 = \emptyset$;                          /* empty sequence */
  for $i = 1$ to $i = n$ do
    Pick $k_i \in M$;
    Set $l_i = l_{i-1} k_i$ od od;             /* simple concatenation */
Define $l_C = l_n$
Return : set of labels $\{l_C \mid C \text{ in } \mathcal{C},\ C \neq \emptyset\}$.
```

Observe: 1. If $C \neq \emptyset$ then $\mathit{IR} \neq \emptyset$. 2. The label $l_C$ computed by Algorithm 1 in not unique. In fact, $M$ in Algorithm 1 is assumed to be ordered according to lexicographical ordering. Other orderings of the elements in $M$ are possible and, as a consequence, a different label can be generated from each ordering.

EXAMPLE 4. *To illustrate how Algorithm 1 works, consider the context $\mathcal{C}x$ in Figure 2. The result of applying Algorithm 1 to*

*$\mathcal{C}x$, limited to focus $\mathcal{C}x\mathcal{F}$ in Figure 2 is depicted in Figure 3. Each label in the three, e.g., $l_{\text{Apple}}$, is a simple concatenation $k_1...k_n$ of terms extracted by Algorithm 1.*

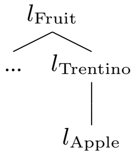

**Figure 3: A focus as labeled by Algorithm 1.**

We are now ready to define "focused terms." Let a focus $\mathcal{F}$ with concept term $C$ as leaf be given. A *focused term for $\mathcal{F}$* is any term that appears in a label $l_C$ of a cluster $C$ in $\mathcal{F}$. In symbols, the set of focus terms for $\mathcal{F}$ is

$$FT(\mathcal{F}) = \{k \mid k \text{ appears in } l_C, C \in \mathcal{F}\}.$$

A focused term for $C$ is any term that appears in $l_C$. A focus term for a concept term plays the role of a synonymous, or alias names, of the concept term. As we will see in Section 6, alias names are important to improve keyword-based querying.

# 4. FACETED ONTOLOGY BUILDING

The result of extracting terms from documents and "facetizing" the concepts of a polyhierarchical classification by using them produces a basic kind of faceted taxonomy, provided that (1) the extracted terms or, often, a proper subset of these [9], are matched with a predefined set of facets, and (2) the clusters in a focus are related to each other by a subsumption relation. For a *faceted taxonomy* consists of: (a) a set of facets, where each facet consists of a predefined set of terms; and (b) a subsumption relation among the terms. In this section we provide the formal framework we need to formalize the focused terms and labeled contexts we have produced by Algorithm 1 by shallowly assuming (2)[2].

## 4.1 Description Logics

Description Logics (DLs) [5] are a family of logic-based knowledge representation formalisms designed to represent and reason about the knowledge of an application domain in a structured and well-understood way. In this paper, we use a basic description logic, called $\mathcal{ALC}$, thereby betting that $\mathcal{ALC}$ provides us with an efficient SAT solver to implement our facet engine (Section 5). $\mathcal{ALC}$ is the smallest propositionally closed DL, and provides the concept constructors

$$\neg C, C \sqcap D, C \sqcup D, \exists R.C, \forall R.C,$$

as well as concept inclusion (or subsumption) $C \sqsubseteq D$ and concept equality $C \equiv D$, where $C$, $D$ are concept descriptions and $R$ is a named role. A DL knowledge base (KB) consists of concept axioms (such as concept inclusion and concept equality axioms), role axioms (such as functional role axioms) and assertions of the form $C(a)$, $R(a, b)$ where $a$ and $b$ are named individuals. For the goal of this paper, we use a limited part of $\mathcal{ALC}$'s expressive power; in particular we do not use role axioms and assertions. Moreover, we write concept descriptions in lower case, as concept description from now on are terms extracted by Algorithm 1 from documents as explained. Due to the limitation of space, we do not provide a detailed introduction of Description Logics (DLs), but rather point the reader to [5, 4] and offer the reader an example.

EXAMPLE 5. *Consider the labeled focus in Example 4. We can represent it within $\mathcal{ALC}$ by a set of equality axioms, that we present as labels of the labeled focus in Figure 4. The concept descriptions $k_i^j$ that appear in the tree refer to the focused terms extracted by Algorithm 1 for each concept in the focus; $hasK$ is a named role, which is intuitively interpreted as 'has keyword'. For example, $\exists hasK.k_1^3$ intuitively means that concept term 'Fruit' in focus $\mathcal{F}$:Fruit>Trentino>*Apple *is extended with focused term (keyword) $k_1^3$. Each equality axiom that appears along the tree defines in $\mathcal{ALC}$ a concept term in $\mathcal{F}$; the focus itself is formalized by the equality axiom: FocusApple* $\equiv \text{Apple} \sqcap \exists \text{R}.(\text{Trentino} \sqcap \exists \text{R}.\text{Fruit})$. *An $\mathcal{ALC}$ KB for this example is the set of the three equality axioms depicted along the tree plus the equality axiom that defines 'FocusApple' as the 'focus Apple', i.e., the focus $\mathcal{F}$.*

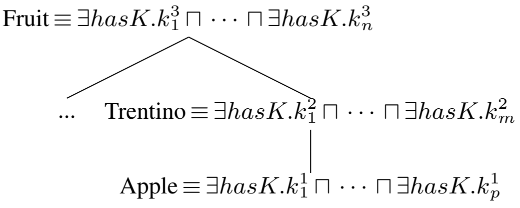
Fruit $\equiv \exists hasK.k_1^3 \sqcap \cdots \sqcap \exists hasK.k_n^3$
...
Trentino $\equiv \exists hasK.k_1^2 \sqcap \cdots \sqcap \exists hasK.k_m^2$
Apple $\equiv \exists hasK.k_1^1 \sqcap \cdots \sqcap \exists hasK.k_p^1$

**Figure 4: A labeled focus in $\mathcal{ALC}$.**

## 4.2 Formal Faceted Classifications

Now we generalize the example. Algorithm 2 below provides a way to build an $\mathcal{ALC}$ faceted knowledge base, or faceted ontology, for a given context. The algorithm works in two main steps.

First, it builds a knowledge base by adding $\mathcal{ALC}$ equality axioms that formally define the concept terms of an input context by using focused terms computed by Algorithm 1 over the same context. For maching purposes that we will see in Section 5, if strictly more or strictly less (but at least one) focused terms were computed for a concept term, then the algorithm adds to the knowledge base all the equality axioms defined over all possible combinations of four focused terms picked up, possibly with repetitions, from the computed terms.

Second, the algorithm adds to the knowledge base so obtained all $\mathcal{ALC}$ equality axioms that formally define DEPA facets of every concept as stored in the facet repository (see Section 2). These axioms have the form

$$C \equiv \exists FacetD.d \sqcap \exists FacetE.e \sqcap \exists FacetP.p \sqcap \exists FacetA.a, \quad (4)$$

where $C$ represents a concept $c$ available in the facet repository, $FacetD$, $FacetE$, $FacetP$, and $FacetA$ are named roles rapresenting the property of $c$ in terms of DEPA facet analysis paradigm.[3] The intended interpretation of these named roles relates to the facet repository. For example, $\exists FacetD.f$ means that there is a concept in the facet repository with facet 'Discipline' be $f$. By extension, equality axiom (4) means that there is a concept in the facet repository with facet 'Discipline' $d$, 'Entity' $e$, 'Property' $p$, and 'Action'

[2] That in our approach clusters in a focus are related to each other by a subsumption relation follows from Convention 2 by observing that polyhierarchical classifications are often subsumption hierarchies. However, we do not need to strictly assume (2) in this paper.

[3] To shorten notation, in algorithms we use $D$, $E$, $P$, $A$ instead.

$a$, and that concept has name $C$. Hence, as per second step, Algorithm 2 adds to the knowledge base all axioms of form as in (4) if and only if there is a concept (or a subject) with DEPA facets $d$, $e$, $p$, $a$ in the facet repository. We make the system insensitive to case and punctuation in the facets $d$, $e$, $p$, $a$ by adding additional axioms where variants of $d$, $e$, $p$, $a$ with the same meaning are used. We call the ontology produced by Algorithm 2 a *formal faceted classification* (FFC).

**Algorithm 2** Building a $\mathcal{ALC}$ faceted ontology $\mathcal{O}$.

```
Input: $\mathcal{C}, \mathcal{D} \neq \emptyset, FR(\mathcal{C})$
Set $\mathcal{O} = \emptyset$;                          /* $\mathcal{ALC}$ ontology to be built */
foreach $C$ in $\mathcal{C}$ with $C \neq \emptyset$ do
    $l_C := \langle k_1 k_2 \cdots k_n \rangle$;             /* $l_C$ computed by Algorithm 1 */
    for $i = 1$ to $i = \binom{n}{4}$ do
        $\mathcal{O} := \mathcal{O} \cup \{C \equiv \exists hasK.k_{i1} \sqcap \cdots \sqcap \exists hasK.k_{i4}\}$ od;
    if $\langle C : d, e, p, a \rangle \in FR(\mathcal{C})$
                        /* facets $d, e, p, a$ for $C$ in facet repository */
      then
            $\mathcal{O} := \mathcal{O} \cup \{\mathrm{C} \equiv \exists \mathrm{D}.d \sqcap \exists \mathrm{E}.e \sqcap \exists \mathrm{P}.p \sqcap \exists \mathrm{A}.a\}$;
                                          /* axiom of form in (4) added */
    fi od;
Return : $\mathcal{O}$.
```

## 5. FACET ENGINE

Now we design within our framework a facet engine that computes the matching between the focused terms of a input context and the predefined set of facets stored in the facet repository for a number of concepts. Intuitively, the facet engine looks at all keywords generated for each concept name in a focus for all focuses of the hierarchy, and browse through the focus from the root to the leaf to identify what keywords are DEPA facets stored in facet repository. The facet engine's main component is Algorithm 3. The basic steps of the algorithm are the following:

Step 1. Input a concept description $C$ that represents a user's query; the different possible queries that can be represented this way are presented in Section 6.

Step 2. Find and retrieve from the ontology built by Algorithm 2 all equality axioms that define $C$ in the ontology either by focused terms or DEPA facets. If no axioms do exist, that is, $C$ is not defined according to the knowledge stored in the ontology, the algorithm ends with no help to the user. This state means that the search engine cannot provide the user with help for query refinement by facets.

Step 3. For all retrieved axioms and for each axiom of the form $C \equiv \exists hasK.k_1 \sqcap \cdots \sqcap \exists hasK.k_n$, where $l_C = k_1 ... k_n$ is the label computed by Algorithm 1, the algorithm runs the $\mathcal{ALC}$ SAT solver in order to match (focused) terms $k_i$ in the axiom to all DEPA facets for $C$ possibly stored in the facet repository. Note that the performance of our method mainly dependents on this step, namely, the number and complexity of the matchings. Preliminary results suggested that the algorithm satisfies the requirements of a query refinement system in terms of real time performance. A complete study of the complexity of this step is in progress.

Step 4. For all successful matchings computed in Step 3, the retrieved DEPA facets are output and shown to the user.

**Algorithm 3** Query expansion with facets from focused terms.

```
proc QueryExpansion
  Input: $C, \mathcal{O}, FR(\mathcal{C})$   /* $C$ is meant to represent user query */
  Define $\Omega_K$ be the set of axioms in $\mathcal{O}$ of the form
  $C \equiv \exists hasK.k_1 \sqcap \cdots \sqcap \exists hasK.k_n$;          /* $k_1 ... k_n = l_C$ */
  Define $\Omega_F$ be the set of axioms in $\mathcal{O}$ of the form
  $C \equiv \exists D.d \sqcap \exists E.e \sqcap \exists P.p \sqcap \exists A.a$;
                                   /* $\langle C : d, e, p, a \rangle$ is in $FR(\mathcal{C})$ */
  if $\Omega_K \vee \Omega_F = \emptyset$
    then exit                      /* no query exspansion provided */
     else
        $s := Card(\Omega_K)$;              /* $\Omega_K$ cardinality is $s \geq 1$ */
        $t := Card(\Omega_F)$;              /* $\Omega_F$ cardinality is $t \geq 1$ */
        $FacetSet(C) := \emptyset$;    /* set of facets retrieved for $C$ */
        for $j = 1$ to $j = s$ do
            $F_{00} := \emptyset$;        /* different facets strings retrieved */
                                /* by using a single axiom in $\Omega_K$ */
            for $l = 1$ to $l = t$ do
                for $i = 1$ to $i = \binom{n}{4}$ do
                    if $\mathcal{O} \models \exists hasK.k_{i1} \sqcap \cdots \sqcap \exists hasK.k_{i4}\} \equiv$
                      $\exists D.d \sqcap \exists E.e \sqcap \exists P.p \sqcap \exists A.a$
                      /* focused terms and DEPA facets match */
                      then
                            $F_{li} := F_{li-1} \cup \{\langle C : d, e, p, a \rangle\}$
                                        /* $\langle C : d, e, p, a \rangle$ retrieved */
                                        /* depending on $k_{i1}$,...,$k_{i4}$ */
                    fi od
            od;
            $FacetSet(C)_j := FacetSet(C)_{j-1} \cup F_{li}$
                 /* all DEPA strings for $C$ in $FR(\mathcal{C})$ retrieved */
        od fi;
  Return : $FacetSet(C)_j$.
```

## 6. QUERY PROCESSING

After building the faceted ontology and defining the facet engine we are ready to use them to provide new facets to the user for query refinement. We allow the user to make three kind of query: keyword-based, by focus, and on subject. We discuss each querying method in turn.

### 6.1 Keyword-based querying

The user types one or more keywords in the search box. This method is the simplest one and it is often the only method available when the user does not know anything about the subject to search, or the user's knowledge on the query subject is not based on documents locally stored in the user querying machine, so that we can not use the ontology and facet engine we have advanced. This is also a tyipical case of keyword-based querying by common search engines, where the keywords used in the query are listed without a specific ordering on the only basis of the user's information need.

We deal with this method of querying as follows. Each keyword is mapped to zero or more concept terms in the context $\mathcal{C}$. We do that using an exact string match of the keyword to the concept term or one of its alias names, namely, its focused terms.

If no concept term and its alias names match any keyword, no concept description is available to the facet engine, and as a consequence no facets for query refinement are shown to the user.

If one concept term or its alias names match some keywords, then the concept description $C$ of the concept term is generated and processed by Algorithm 3 for query expansion. The facets that occur in the query expansion are shown to the user. When selecting one of the new facets, the user will narrow down the search by expanding the original query with the suggested facet.

If multiple concept terms match some keywords, then the concept description of each term is generated and processed by Algorithm 3 for query expansion. The facets that occur in the query expansion of every concept description are shown to the user. Alternatively, the user is given the option to refine their query to indicate which concept term, namely, keyword they meant the most.

## 6.2 Querying-By-Focus

Now suppose that the user knows at least something about the subject to search, and the user's knowledge comes from documents stored and polyhierarchically organized in the user's document collection. In this case, it would always be desiderable for the user to get better and better understanding of the hidden content of the query, as it is automatically generated by a suitable method, so as to discover new facets of the original query that the user was not aware of before. For example, suppose the query is 'apple' as contextualized in Figure 5. The user clicks on a concept term in a context $\mathcal{C}$. In doing that, the user selects a focus in $\mathcal{C}$. Alternatively, the user types some keywords as in keyword-based querying, but in a specific order to mean a focus in $\mathcal{C}$. For example, the user may click on (an appropriate graphic-version of) 'Apple' in context or either type keywords 'fruit', 'trentino', 'apple' *in this order*, as to mean $\mathcal{C}x$:Fruit>Trentino>Apple. In the example, by selecting the facet 'Fruit' the user would narrow down the search space by excluding all subjects about Apple Computers and related subjects as search results (see Figure 1). Similarly, by selecting facet 'Trentino' the user would be able to narrow down the search space by excluding all subjects about fruits that are not related to Trentino's production of apples. It follows that the keyword-based method and querying by focus are not equivalent for at least one reason, that is, in keyword-based querying the order of keywords does not matter, in querying by focus does. The other main difference between these two querying methods arises looking at query processing. The difference is that concept terms in a focus are not 'pure' keywords; a concept term is represented by a *string of similar keywords* as generated by Algorithm 1. Concept terms relate to documents in the user's repository, while keywords are usually unrelated to the user's documents.

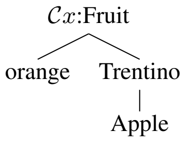

**Figure 5: A focus for query 'Apple'.**

A query-by-focus is similar to a query by example, yet it is more specific. In querying by example, a sample document (*the example*) is selected by the user to refine the query. On the other hand, in querying by focus the *position* of the sample document is also considered, that is, the place the document is stored within the user's documentary repository. To illustrate, suppose that a user stores his documents according two different structures, see Figure 6. Now suppose the user selects the document named **doc1** as the sample document. In classical querying by example, a relevant answer to the user would be any document about 'apple', as meant as either a fruit or a computer. In contrast, using querying by focus the only relevant answers to the user would be documents from one of the two focus Fruit>Apple and Computers>Apple.

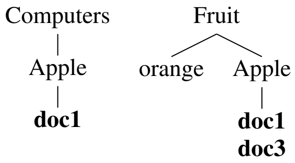

**Figure 6: Position of sample document doc1 matters.**

We deal with querying by focus as follow. First, a concept description $C$ of the concept term that is the leaf of the focus is generated and processed by Algorithm 3 for query expansion. The facets that occur in the query expansion are shown to the user. When selecting one of the new facets, the user will narrow down the search by expanding the original query with the suggested facet.

Note that the case where query by focus applies in practical situations is not as uncommon as it may seem, because almost all users start a search from a device storing text and text-annotated documents, and these are often organized by the user according to a polyhierarchical classification. More importantly, the fact that a user searches the Web does not mean that documents *from the Web* will be used for the purpose of querying by focus. The documents used for querying by focus are all and only the documents locally stored in the user's querying device, whatever the search objective is either to retrieve documents stored in the user's device or in the Web. As a consequence, querying by focus clearly scales to the size of the web. To understand a bit further, recall that our method is about query refinement, it is not a query search method. We use standard methods and search engines to search; the difference is that the keywords we let the search engines to use are automatically generated by our facetization technique.

## 6.3 Querying-On-Subject

Subject-based querying is the most common approach by specialized users, where 'subject' refers to the topical intent of a query (cf. Section 2). In our faceted approach to representation of documents in collection $\mathcal{D}$, 'subjects' are broken down into distinct divisions, the facets of subject. A typical 'query-on-subject' is deemed to relate to a specific subject of a preexisting faceted classification. For example, a subject-based query is: 'What are the documents on the effects of nitrogen fertilizers on rice plants?' The subject of the concept subsumed by this query is one of possibly many focuses, for example the following:

$$\mathcal{C}x\text{:rice plants}>\text{nitrogen fertilizers}>\text{effects.} \tag{5}$$

This is a partial focus, in the sense that the discipline subsumed by the query as provided by the DEPA facet analysis is

$$\mathcal{C}x\text{:Agriculture}>\text{rice plants}>\text{nitrogen fertilizers}>\text{effects.} \tag{6}$$

Another possible focus for the subject of query's concept is the following:

$$\mathcal{C}'x\text{:Agriculture}>\text{ effects of nitrogen}>\text{ fertilizers}>\text{rice plants.} \tag{7}$$

A number of different but equivalent focuses could exists for a given subject-based query. Note the the existance of a focus for this query as well as the focus form depend only upon the querying user's classification of documents. The take-away point is that by merging a subject to one or more focuses, by automatically transforming a query-on-subject to a query-by-focus, the method provides the user with assistance in query refinement. In fact, we compute the focuses generated from the query on subject, and for each focus we consider the concept description that represents the focus in $\mathcal{ALC}$ ontology computed by Algorithm 2. Then we proceed as in the case of querying by focus and compute the query expansion of the focus according to knowledge stored in the ontology. Finally, the retrieved facets are shown to the user. If multiple focuses are computed from the query's subject, the user is given the option to refine the original query to indicate which focus they meant for the searched subject.

# 7. RELATED WORK

There has been extensive work on automated facet construction motivated by query refinement, browsing and navigation over document collections, see for instance [29], [8, 9], [10], .[24], [30, 13]. The notion of context in these related works differ from the notion of focus; in [10] context is a piece of text, from a document the user is presented to, surrounding the query, which is marked by the user on the document. The structural nature of a focus contrasts with the plain, linguistic nature of query context as meant in [10]. The navigation trees discussed in [28] are similar to the focuses discussed in this paper. The formal approach of [28], moreover, as well as the use of faceted taxonomies is close in spirit, if not in the formal development to our work presented here. As far as we know, none of the foregoing approaches uses a DEPA facet schema.

Our method is a focused retrieval method, in the sense that focused retrieval addresses ways to provide a querying user a more direct access to relevant information [26]. Focused retrieval aims to identify not only documents relevant to a user information need, but also where within the document the relevant information is located. Our approach of querying-by-focus is similar to querying by focus on hierarchical classifications proposed by [1, 2].

In the Indian Context, faceted library systems, especially the Colon Classification System (CCS), has been adopted by majority of the academic libraries for organizing collections in semantic arrangement. However, there is a wide scope for use of the faceted theory behind systems such as CCS to other knowledge modeling efforts. Prasad and Guha [18] intoduced a facet-based method to formulate the descriptive domain metadata that could be used to annotate digital library resources. Prasad and Madalli [19] propose a generic model for building semantic infrastructure for digital libraries based on facets as used in traditional library classification systems.

Faceted taxonomies are extensively studied, see for instance [21, 27, 28] and references therein. Facet techniques include that studied by Tvarožek and Bieliková [27], who have proposed faceted navigation and its personalization in digital libraries. They follow a method of faceted browser adaptation based on an automatically acquired user model with support for dynamic facet generation J. Polowinski [17] argues for use of Faceted Browsing as a visual selection mechanism to browse data collections as it is deemed as being particularly suitable for structured, but heterogeneous data with explicit semantics. Normalized Formal Classifications (NFC) used in [11] does this by taking into account both the label of the node and its position using natural language processing techniques (see [11, sec 4]). On the other hand, we have used an information retrieval technique to find out the keywords that will successively represented in concept descriptions by using role names of the form $hasK.k$. This is an important difference with [11]. A focus is called "concept at a node" in [11, p. 70], although we believe that the two notions are not totally equivalent (to be investigated). The notion of Formal Faceted Classification (FFC) extends the notion of "lightweight ontology" of [11] to facets. A main difference with lightweight ontologies by [11] is that FFC's descriptive language is not propositional as the language used in [11]. Yet, it allows us to automate, through DL reasoning services (SAT), query refinement, as we did in this paper. Moreover, by our query language we allow a user to specify a query by selecting a sample document, to be interpreted of as the "information need" of documents similar to the sample selected. As a consequence, we provide a user with a mechanism of "querying by example" as a special case. On the other hand, in [11] it seems not easy to formalize querying by example, as the propositional language used does not allow to represent instances.

# 8. CONCLUSION

This paper presented a formal framework for a querying refinement method that enables the extraction of the diversity aspects, or facets, of a user query. The method uses the general principles of facet analysis in the DEPA paradigm of facetization and the notion of 'focus', which is used to infer new facets from the user query. The method provides a user with additional and essential contextual information, in form of new facets. When selecting one of the new facets, the user can narrow down the search by expanding the original query with the suggested facets. The proposed method of query refinement is based on diversity in querying and a multi-dimensionality of information. Three methods of querying weree discussed: keyword-based, by focus, and on subject. For each method, textual and structural dimensions were used to assist the user in query refining. The textual dimension allowed us to generate the top-k most relevant terms for each concept of a given polyhierarchy of text and text-annotated documents. The structural dimension of the polyhierarchy was used to match DEPA facets with the user query. We have situated our framework within the smallest propositionally closed description logic $\mathcal{ALC}$, and we have used $\mathcal{ALC}$'s solver to implement the facet engine as the main component of the method.

# 9. REFERENCES

[1] A. Agostini and P. Avesani. On the discovery of the semantic context of queries by game-playing. In H. Christiansen, M.-S. Hacid, T. Andreasen, and H. Larsen, editors, *Proceedings of the Sixth International Conference On Flexible Query Answering Systems (FQAS-04)*, pages 203–216, Berlin Heidelberg, 2004. Springer-Verlag LNAI 3055.

[2] A. Agostini and G. Moro. Identification of communities of peers by trust and reputation. In D. F. C. Bussler, editor, *Proceedings of the Eleventh International Conference on Artificial Intelligence: Methodology, Systems, Applications - Semantic Web Challenges (AIMSA-04)*, pages 85–95, Berlin Heidelberg, 2004. Springer-Verlag LNAI 3192.

[3] R. Agrawal, S. Gollapudi, A. Halverson, and S. Ieong. Diversifying search results. In *Proceedings of the Second ACM International Conference on Web Search and Data*

*Mining (WSDM-00)*, pages 5–14, New York, NY, 2009. ACM Press.

[4] F. Baader, D. Calvanese, D. McGuinness, D. Nardi, and P. Patel-Schneider, editors. *Handbook of Description Logics*, Cambridge, UK, 2002. Cambridge University Press.

[5] F. Baader and W. Nutt. Basic description logics. In F. Baader, D. Calvanese, D. M. Guinness, and P. P.-S. D. Nardi, editors, *Handbook of Description Logics*, pages 47–100. Cambridge University Press, Cambridge, UK, 2002.

[6] G. Bhattacharyya. POPSI: its fundamentals and procedure based on a general theory of subject indexing languages. *Library Science with a Slant to Documentation*, 16(1):1–34, 1976.

[7] G. Bhattacharyya. Subject indexing language: its theory and practice. In *Proceedings of the DRTC Refresher Seminar–13, New Developments in LIS in India*, Bangalore, India, 1981. DRTC, ISI Bangalore Centre.

[8] W. Dakka, R. Dayal, and P. Ipeirotis. Automatic discovery of useful facet terms. In *Proceedings of the ACM SIGIR 2006 Workshop on Faceted Search*, New York, NY, 2006. ACM Press.

[9] W. Dakka and P. Ipeirotis. Automatic extraction of useful facet hierarchies from text databases. In *Proceedings of the 2008 IEEE 24th International Conference on Data Engineering (ICDE-08)*, pages 466–475, Washington, DC, USA, 2008. IEEE Computer Society.

[10] L. Finkelstein, E. Gabrilovich, Y. Matias, E. Rivlin, Z. Solan, G. Wolfman, and E. Ruppin. Placing search in context: The concept revised. In *Proceedings of the Tenth International World Wide Web Conference (WWW-2001)*, pages 406–414, New York, NY, 2001. ACM Press.

[11] F. Giunchiglia, M. Marchese, and I. Zaihrayeu. Encoding classifications into lightweight ontologies. In S. Spaccapietra and et. al., editors, *Journal on Data Semantics VIII*, pages 57–81. Springer-Verlag LNCS 4380, Berlin Heidelberg, 2007.

[12] B. J. Jansen, A. Spink, and T. Saracevic. Real life, real users, and real needs: a study and analysis of user queries on the web. *Information Processing & Management*, 36(2):207–227, 2000.

[13] K. Latha, K. R. Veni, and R. Rajaram. AFGF: An automatic facet generation framework for document retrieval. In *Proceedings of the 2010 International Conference on Advances in Computer Engineering (ACE-2010)*, pages 110–114, Washington, DC, USA, 2010. IEEE Computer Society.

[14] S. Lawrence. Context in Web Search. *IEEE Data Engineering Bulletin*, 23(3):25–32, 2000.

[15] V. Maltese, F. Giunchiglia, K. Denecke, P. Lewis, C. Wallner, A. Baldry, and D. Madalli. On the interdisciplinary foundations of diversity. In G. Boato and C. Niederee, editors, *Proceedings of the First International Workshop on Living Web at ISWC-09, Washington D.C., USA, October 26, 2009*. CEUR-WS, 2009.

[16] P. Morville and L. Rosenfeld. *Information architecture for the World Wide Web, 3rd edition*. O'Reilly Media, Inc., Sebastopol, CAe, 2006.

[17] J. Polowinski. Human interface and the management of information. Designing information environments. In M. J. Smith and G. Salvendy, editors, *Proceedings of the Symposium on Human Interface 2009, held as Part of HCI International 2009 (HCII-09), San Diego, CA, USA, July 19-24, 2009*, pages 601–610, Berlin Heidelberg, 2009. Springer-Verlag LNCS 5617.

[18] A. Prasad and N. Guha. Expressing faceted subject indexing in SKOS/RDF. In *Proceedings of the First International Conference of Semantic Web and Digital Libraries, Bangalore 21-23 February (ICSWDL-07)*, 2007.

[19] A. Prasad and D. Madalli. Semantic digital faceted infrastructure for semantic digital libraries. *Library Review*, 57(3):225–234, 2008.

[20] S. R. Ranganathan. *Prolegomena to Library Classification*. Asia Publishing House, London, 1967.

[21] G. Sacco and Y. Tzitzikas, editors. *Dynamic Taxonomies and Faceted Search*, The Information Retrieval Series, v. 25, Berlin Heidelberg, 2009. Springer-Verlag.

[22] G. Salton, editor. *The SMART Retrieval System—Experiments in Automatic Document Retrieval*, Englewood Cliffs, NJ, 1971. Prentice-Hall Inc.

[23] G. Salton and M. McGill. *Introduction to Modern Information Retrieval*. McGraw-Hill, New York, NY, 1983.

[24] E. Stoica, M. A. Hearst, and M. Richardson. Automating creation of hierarchical faceted metadata structures. In *Proceedings of the Human Language Technology Conference (NAACL HLT)*, pages 244–251, Rochester, NY, USA, 2007. Association for Computational Linguistics.

[25] P. Tonella, F. Ricca, E. Pianta, and C. Girardi. Using keyword extraction for web site clustering. In K. Wong, editor, *Proceedings of the Fifth International Workshop on Web Site Evolution (WSE-03)*, pages 41–48, Amsterdam, The Netherlands, 2003. IEEE Computer Society.

[26] A. Trotman, S. Geva, J. Kamps, M. Lalmas, and V. Murdock. Current research in focused retrieval and result aggregation. *Special Issue in the Journal of Information Retrieval*, 13(5):407–411, 2010.

[27] M. Tvarožek and M. Bieliková. Personalized faceted browsing for digital libraries. In L.ács, N. Fuhr, and C. Meghini, editors, *Research and Advanced Technology for Digital Libraries. Proceedings of the 11th European Conference on Digital Libraries (ECDL-07), Budapest, Hungary, September 16-21, 2007*, pages 485–488, Berlin Heidelberg, 2007. Springer-Verlag LNCS 4675.

[28] Y. Tzitzikas, N. Spyratos, P. Constantopoulos, and A. Analyti. Extended faceted taxonomies for web catalogs. In *Proceedings of the Third International Conference on Web Information Systems Engineering (WISE-02)*, pages 192–204, 2002.

[29] R. van Zwol and B. Sigurbjörnsson. Faceted exploration of image search results. In *Proceedings of the Nineteenth International World Wide Web Conference (WWW-10)*, pages 961–970, New York, NY, 2010. ACM Press.

[30] E. Vee, U. Srivastava, J. Shanmugasundaram, P. Bhat, and S. A. Yahia. Efficient computation of diverse query results. In *Proceedings of the 2008 IEEE 24th International Conference on Data Engineering (ICDE-08)*, pages 228–236, Washington, DC, USA, 2008. IEEE Computer Society.

[31] B. Vickery. *Faceted classification: A guide to construction and use of special schemes*. Aslib - Asia Publishing House, London, 1960.

[32] K. Weinberger, M. Slaney, and R. van Zwol. Resolving tag ambiguity. In *Proceedings of the 16th International ACM Conference on Multimedia (MM 2008)*, New York, NY, 2010. ACM Press.