The Semantics of the Compound Term Composition Algebra

Yannis Tzitzikas[1,5], Anastasia Analyti[2], Nicolas Spyratos[3]

[1] *Istituto di Scienza e Tecnologie dell' Informazione, ISTI-CNR, Italy*
[2] *Institute of Computer Science, ICS-FORTH, Greece*
[3] *Laboratoire de Recherche en Informatique, Universite de Paris-Sud, France*
Email : tzitzik@isti.cnr.it, analyti@csi.forth.gr, spyratos@lri.fr

Abstract. In [TASC03], we proposed an algebra with four algebraic operators, whose composition can be used to generate valid compound terms in a given faceted taxonomy in an efficient and flexible manner. The positive operations allow the derivation of valid compound terms through the declaration of a small set of valid compound terms. The negative operations allow the derivation of valid compound terms through the declaration of a small set of invalid compound terms. Here, we formally define the model-theoretic semantics of the operations and the closed-world assumptions adopted in each operation. We prove that our algebra is monotonic with respect to both valid and invalid compound terms, meaning that the valid and invalid compound terms of a subexpression are not invalidated by a larger expression. The importance of this property is demonstrated through an example. We also show that our algebra cannot be directly represented in Description Logics. A metasystem on top of Description Logics is designed to implement our algebra.
Keywords: Faceted Taxonomies, Semantics, Description Logics.

1 Introduction

A faceted taxonomy is a set of taxonomies, each describing a given domain from a different aspect, or facet [Ran65]. The indexing of domain objects is done through conjunctive combinations of terms from the facets, called compound terms. For example, assume that the domain of interest is a set of hotel Web pages in Greece, and suppose that we want to provide access to these pages according to the *Location* of the hotels and the *Sports* facilities they offer. Figure

[5] Work done during the postdoctoral studies of the author at CNR-ISTI as an ERCIM fellow. The first part of this work was done when the author was at ICS-FORTH.

1 shows these two facets. Each object is described using a *compound term*. For example, a hotel in Crete providing sea ski and wind-surfing facilities would be described by the compound term $\{Crete, SeaSki, Windsurfing\}$.

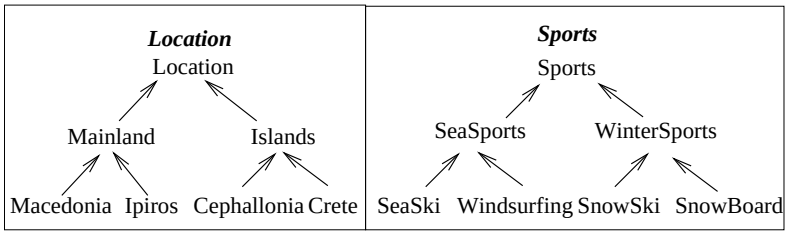

Fig. 1. Two facets

Faceted taxonomies carry a number of well known advantages over single hierarchies in terms of building and maintaining them, as well as using them in multicriteria indexing. A drawback, however, is the cost of avoiding *invalid* combinations, i.e. compound terms that do not apply to any object in the domain. For example, the compound term $\{Crete, SnowBoard\}$ is an invalid compound term, as there no hotels in Crete offering snow-board facilities.

In [TASC03], we proposed an algebra whose operators (two positive and two negative) allow the efficient and flexible specification of valid compound terms, thus alleviating the main drawback of faceted taxonomies. Following this approach, given a faceted taxonomy, one can use an *algebraic expression* to define the desired set of compound terms. In each algebraic operation, the designer has to declare either a small set of valid compound terms from which other valid compound terms are inferred, or a small set of invalid compound terms from which other invalid compound terms are inferred. Then, a closed-world assumption is adopted for the rest of the compound terms in the range of the operation. For example, if a user declares in a positive operation that the compound term $\{Crete, SeaSki\}$ is valid then it is inferred that the compound term $\{Crete, SeaSports\}$ is also valid. If a user declares in a negative operation that the compound term $\{Crete, WinterSports\}$ is invalid then it is inferred that the compound term $\{Crete, SnowBoard\}$ is also invalid. In our example, this means that the designer can specify all valid compound terms of the faceted taxonomy by providing a relatively small number of (valid or invalid) compound terms. This is an important feature as it minimizes the effort needed by the designer. Only the expression that defines the compound terminology has to be stored, as an inference mechanism with polynomial time complexity (given in [TASC03]) can check whether a compound term belongs to the compound terminology of the expression.

The proposed algebra can be used in order to construct taxonomies or thesauri (e.g. for Web catalogues like Yahoo! or ODP) which unlike existing thesauri, do not present the problem of missing terms or missing relationships.

In this paper, we emphasize on the semantics of the algebra. Specifically, we formally define the model-theoretic semantics of the operations and the closed-world assumptions adopted in each operation. First, intermediate semantics are defined for the particular operations, and then intermediate semantics are synthesized to define the semantics of the complete algebraic operation. Based on these, we define the models of an algebraic expression, and we prove that every *well-formed* algebraic expression is satisfiable. We also prove that our algebra is monotonic with respect to both valid and invalid compound terms, meaning that the valid and invalid compound terms of a subexpression are not invalidated by a larger expression. The importance of this property is demonstrated through an example.

In addition in this paper, we show that our algebra cannot be directly represented in Description Logics. A metasystem on top of Description Logics is designed to implement our algebra.

The remaining of this paper is organized as follows: Section 2 describes the algebra, and justifies the definition of a well-formed algebraic expression based on the monotonicity property. Section 3 gives a semantic interpretation of the algebra. Section 4 compares our approach with Description Logics. Finally, Section 5 concludes the paper.

2 The Compound Term Composition Algebra

In this section, we present in brief the *compound term composition algebra*, defined in [TASC03]. For more explanations, and examples the reader should refer to that article.

A *terminology* is a finite set of names, called *terms*. A *taxonomy* is a pair $(\mathcal{T}, \leq)$, where $\mathcal{T}$ is a *terminology* and $\leq$ is a reflexive and transitive relation over $\mathcal{T}$, called *subsumption*.

A *compound term* over $\mathcal{T}$ is any subset of $\mathcal{T}$. For example, the following sets of terms are compound terms over the terminology $Sports$ of Figure 1: $s_1 = \{SeaSki, Windsurfing\}$, $s_2 = \{SeaSports\}$, and $s_3 = \emptyset$.

A *compound terminology* S over $\mathcal{T}$ is any set of compound terms that contains the compound term $\emptyset$.

The set of all compound terms over $\mathcal{T}$ can be ordered using the *compound ordering* over $\mathcal{T}$, defined as: $s \preceq s'$ iff $\forall t' \in s' \ \exists t \in s$ such that $t \leq t'$.

That is, $s \preceq s'$ iff s contains a narrower term for every term of s'. In addition, s may contain terms not present in s'. Roughly, $s \preceq s'$ means that s carries more specific information than s'. For example, $\{SeaSki, Windsurfing\} \preceq \{SeaSports\} \preceq \emptyset$.

We say that two compound terms s, s' are *equivalent* iff $s \preceq s'$ and $s' \preceq s$. For example, $\{SeaSki, SeaSports\}$ and $\{SeaSki\}$ are equivalent. Intuitively, equivalent compound terms carry the same information.

Definition 1. A *compound taxonomy* over $\mathcal{T}$ is a pair $C = (S, \preceq)$, where S is a compound terminology over $\mathcal{T}$, and $\preceq$ is the compound ordering over $\mathcal{T}$ restricted to S.

Let $P(\mathcal{T})$ be the set of all compound terms over $\mathcal{T}$ (i.e. the powerset of $\mathcal{T}$). Clearly, $(P(\mathcal{T}), \preceq)$ is a compound taxonomy over $\mathcal{T}$.

Let s be a compound term. The broader and the narrower compound terms of s are defined as follows: $\mathrm{Br}(s) = \{s' \in P(\mathcal{T}) \mid s \preceq s'\}$ and $\mathrm{Nr}(s) = \{s' \in P(\mathcal{T}) \mid s' \preceq s\}$.

Let S be a compound terminology over $\mathcal{T}$. The broader and the narrower compound terms of S are defined as follows: $Br(S) = \cup\{\mathrm{Br}(s) \mid s \in S\}$ and $Nr(S) = \cup\{\mathrm{Nr}(s) \mid s \in S\}$.

We say that a compound term s is *valid* (resp. *invalid*), if there is at least one (resp. no) object of the underlying domain indexed by all terms in s. We assume that every term of $\mathcal{T}$ is valid. However, a compound term over $\mathcal{T}$ may be invalid. Obviously, if s is a valid compound term, all compound terms in $\mathrm{Br}(s)$ are valid. Additionally, if s is an invalid compound term, all compound terms in $\mathrm{Nr}(s)$ are invalid. The formal definition of validity is given in Section 3.

One way of designing a taxonomy is by identifying a number of different aspects of the domain of interest and then designing one taxonomy per aspect. As a result we obtain a set of taxonomies called *facets*. Given a set of facets we can define a *faceted taxonomy*.

Definition 2. Let $\{F_1, ..., F_k\}$ be a finite set of taxonomies, where $F_i = (\mathcal{T}_i, \leq_i)$, and assume that the terminologies $\mathcal{T}_1$, ... ,$\mathcal{T}_k$ are pairwise disjoint. Then the pair $\mathcal{F} = (\mathcal{T}, \leq)$, where $\mathcal{T} = \bigcup_{i=1}^{k} \mathcal{T}_i$ and $\leq \; = \bigcup_{i=1}^{k} \leq_i$, is a taxonomy which we shall call the *faceted taxonomy generated* by $\{F_1, ..., F_k\}$. We shall call the taxonomies $F_1, ..., F_k$ the *facets* of $\mathcal{F}$.

Clearly, all definitions introduced so far apply also to $(\mathcal{T}, \leq)$. For example, the set $S = \{\{Greece\}, \{Sports\}, \{SeaSports\}, \{Greece, Sports\},$ $\{Greece, SeaSports\}, \emptyset\}$, is a compound terminology over the terminology $\mathcal{T}$ of the faceted taxonomy shown in Figure 1. Additionally, the pair $(S, \preceq)$ is a compound taxonomy over $\mathcal{T}$.

Let $\mathcal{F} = (\mathcal{T}, \leq)$ be the faceted taxonomy generated by a given set of facets $\{F_1, ..., F_k\}$. The problem is that $\mathcal{F}$ does not itself specify which compound terms, i.e. which elements of $P(\mathcal{T})$, are valid and which are not. To alleviate this problem, we introduce an algebra for defining a compound terminology over $\mathcal{T}$ (i.e. a subset of $P(\mathcal{T})$) which consists of the valid compound terms.

To begin with we associate the terminology $\mathcal{T}_i$ of every facet with a compound terminology T_i that we call the *basic compound terminology* of $\mathcal{T}_i$. Specifically,

$$T_i = \cup\{\ \mathrm{Br}(\{t\}) \mid t \in \mathcal{T}_i\}$$

As every term t of a facet is considered valid, all compound terms in $\mathrm{Br}(\{t\})$ are valid. Thus, T_i is the set of compound terms over $\mathcal{T}_i$ that are initially known to be valid. We use the basic compound terminologies as the "building blocks" of our algebra[6].

For defining the desired compound taxonomy the designer has to formulate an algebraic expression e, using the operations *plus-product*, *minus-product*, *plus-self-product*, *minus-self-product*, and initial operands the basic compound terminologies $\{T_1, .., T_k\}$.

Let $\mathcal{S}$ be the set of compound terminologies over $\mathcal{T}$. First, we define the auxiliary n-ary operation $\oplus$ over $\mathcal{S}$, called *product*. This operation results in an "unqualified" compound terminology whose compound terms are all possible combinations of compound terms from its arguments. Specifically, let $S_1, ..., S_n$ be compound terminologies, we define:

$$S_1 \oplus ... \oplus S_n = \{s_1 \cup ... \cup s_n \mid s_i \in S_i\}$$

2.1 Algebraic operations

In this subsection, we describe each algebraic operation, in brief. Examples of the operations are given in [TASC03]. The definitions of the operations are semantically justified in Section 3.

We introduce two "variations" of the $\oplus$ operation, namely the *plus-product* and the *minus-product*. Each of these two operations has an extra parameter denoted by P or N, respectively. The set P is a set of compound terms that are certainly valid. On the other hand, the set N is a set of compound terms that are certainly invalid. These parameters are declared by domain experts that perform the indexing and allow to infer all compound terms that are valid or invalid.

To proceed we need to distinguish what we shall call *genuine compound terms*. Intuitively, a genuine compound term combines non-empty compound terms from more than one compound terminologies.

Definition 3. The set of *genuine* compound terms over a set of compound terminologies $S_1, ..., S_n$, denoted by $G_{S_1,...,S_n}$, is defined as follows:

$$G_{S_1,...,S_n} = S_1 \oplus ... \oplus S_n - \bigcup_{i=1}^{n} S_i$$

[6] Note that each basic compound terminology is a compound terminology over $\mathcal{T}$.

For example if $S_1 = \{\{Greece\}, \{Islands\}, \emptyset\}$, $S_2 = \{\{Sports\}, \{WinterSports\}, \emptyset\}$, and $S_3 = \{\{Pensions\}, \{Hotels\}, \emptyset\}$ then

$$\{Greece, WinterSports, Hotels\} \in G_{S_1,S_2,S_3},$$
$$\{WinterSports, Hotels\} \in G_{S_1,S_2,S_3} \text{ , but}$$
$$\{Hotels\} \notin G_{S_1,S_2,S_3}$$

Assume that the compound terms of $S_1, ..., S_n$ are valid. We are interested in characterizing the validity of all combinations of compound terms of $S_1, ..., S_n$. As we already know the validity of the compound terms of $S_1, ..., S_n$, we are basically interested in characterizing the validity of the compound terms in $G_{S_1,...,S_n}$. This is done through the following operations, plus-product and minus-product.

We now define the *plus-product* operation, $\oplus_P$, an n-ary operation over $\mathcal{S}$ ($\oplus_P : \mathcal{S} \times ... \times \mathcal{S} \rightarrow \mathcal{S}$), where the parameter P is a set of valid compound terms from the product of the input compound terminologies. Specifically, the set P is a subset of $G_{S_1,...,S_n}$, as we assume that all compound terms in the input parameters are valid.

Definition 4. Let $S_1, ..., S_n$ be compound terminologies and $P \subseteq G_{S_1,...,S_n}$. The *plus-product* of $S_1, ..., S_n$ with respect to P, denoted by $\oplus_P(S_1, ..., S_n)$, is defined as follows:

$$\oplus_P(S_1, ...S_n) = S_1 \cup ... \cup S_n \ \cup \ Br(P)$$

This operation results in a compound terminology consisting of the compound terms of the initial compound terminologies, *plus* the compound terms which are broader than an element of P. This is because, if a compound term p is valid then all compound terms in $\mathrm{Br}(p)$ are also valid.

For any parameter P, it holds: $\bigcup_{i=1}^{n} S_i \subseteq \oplus_P(S_1, ..., S_n) \subseteq S_1 \oplus ... \oplus S_n$.

Now we define the *minus-product* operation, $\ominus_N$, an n-ary operation over $\mathcal{S}$ ($\ominus_N : \mathcal{S} \times ... \times \mathcal{S} \rightarrow \mathcal{S}$), where the parameter N is a set of invalid compound terms from the product of the input compound terminologies. Specifically, the set N is a subset of $G_{S_1,...,S_n}$, as we assume that all compound terms in the input operands are valid.

Definition 5. Let $S_1, ..., S_n$ be compound taxonomies and $N \subseteq G_{S_1,...,S_n}$. The *minus-product* of $S_1, ..., S_n$ with respect to N, denoted by $\ominus_N(S_1, ..., S_n)$, is defined as follows:

$$\ominus_N(S_1, ...S_n) = S_1 \oplus ... \oplus S_n - Nr(N)$$

This operation results in a compound terminology consisting of all compound terms in the product of the initial compound terminologies, *minus* all compound

terms which are narrower than an element of N. This is because, if a compound term n is invalid then every compound term in $\mathrm{Nr}(n)$ is invalid.

For any parameter N, it holds: $\bigcup_{i=1}^{n} S_i \subseteq \ominus_N(S_1, ..., S_n) \subseteq S_1 \oplus ... \oplus S_n$.

The operators introduced so far allow defining a compound terminology which consists of compound terms that contain at most one compound term from each basic compound terminology. However, a valid compound term may contain any set of terms of the same facet (multiple classification). To capture such cases, we define the *self-product*, $\overset{*}{\oplus}$, a unary operation which gives all possible compound terms of one facet. Subsequently, we shall modify this operation with the parameters P and N.

Let $\mathcal{BS}$ be the set of basic compound terminologies, that is $\mathcal{BS}= \{T_1, ..., T_k\}$.

The *self-product* of T_i is defined as: $\overset{*}{\oplus}(T_i) = P(\mathcal{T}_i)$.

The notion of genuine compound terms is also necessary here. The set of *genuine* compound terms over T_i is defined as: $G_{T_i} = \overset{*}{\oplus}(T_i) - T_i$.

Now we define the *plus-self-product* operation, $\overset{*}{\oplus}_P$, a unary operation ($\overset{*}{\oplus}_P$: $\mathcal{BS} \rightarrow \mathcal{S}$) where the parameter P is a set of compound terms that are certainly valid. The set P is a subset of G_{T_i}.

Definition 6. Let T_i be a basic compound terminology and $P \subseteq G_{T_i}$. The *plus-self-product* of T_i with respect to P, denoted by $\overset{*}{\oplus}_P(T_i)$, is defined as follows:

$$\overset{*}{\oplus}_P(T_i) = T_i \cup Br(P)$$

This operation results in a compound terminology consisting of the compound terms of the initial basic compound terminology, *plus* all compound terms which are broader than an element of P.

For any parameter P, it holds: $T_i \subseteq \overset{*}{\oplus}_P(T_i) \subseteq \overset{*}{\oplus}(T_i)$

The following definition introduces the *minus-self-product* operation, $\overset{*}{\ominus}_N$, a unary operation ($\overset{*}{\ominus}_N$: $\mathcal{BS} \rightarrow \mathcal{S}$) where the parameter N is a set of compound terms that are certainly invalid. The set N is a subset of G_{T_i}.

Definition 7. Let T_i be a basic compound terminology and $N \subseteq G_{T_i}$. The *minus-self-product* of T_i with respect to N, denoted by $\overset{*}{\ominus}_N(T_i)$, is defined as follows:

$$\overset{*}{\ominus}_N(T_i) = \overset{*}{\oplus}(T_i) - Nr(N)$$

This operation results in a compound terminology consisting of all compound terms in the self-product of T_i, *minus* the compound terms which are narrower than an element in N.

For any parameter N it holds: $T_i \subseteq \overset{*}{\ominus}_N(T_i) \subseteq \overset{*}{\oplus} T_i$

2.2 Algebraic Expressions

For defining the desired compound taxonomy, the designer has to formulate an expression e, where an expression is defined as follows:

Definition 8. An expression over a set of facets $\{F_1, ..., F_k\}$ is defined according to the following grammar:

$$e ::= \oplus_P(e, ..., e) \mid \ominus_N (e, ..., e) \mid \overset{*}{\oplus}_P T_i \mid \overset{*}{\ominus}_N T_i \mid T_i$$

The outcome of the evaluation of an expression e is denoted by S_e and is called the *compound terminology* of e. In addition, $(S_e, \preceq)$ is called the *compound taxonomy* of e.

As we will see in Section 3, all compound terms in S_e are *valid*, and the rest in $P(\mathcal{T}_e) - S_e$ are *invalid*, where $\mathcal{T}_e$ is the union of the terminologies of the facets appearing in e.

We are interested only in *well-formed* expressions, defined as follows:

Definition 9. An expression e is *well-formed* iff:

(i) each basic compound terminology T_i appears at most once in e,
(ii) each parameter P that appears in e, is a subset of the associated set of genuine compound terms, and
(iii) each parameter N that appears in e, is also a subset of the associated set of genuine compound terms.

For example, the expression $(T_1 \oplus_P T_2) \ominus_N T_1$ is not well-formed, as T_1 appears twice in the expression.

Constraints (i), (ii), and (iii) ensure that the evaluation of an expression is monotonic, meaning that the valid and invalid compound terms of an expression e increase as the length of e increases[7] (in other words, there are no conflicts). For example, if we omit constraint (i) then an invalid compound term according to an expression $T_1 \oplus_P T_2$ could be valid according to a larger expression $(T_1 \oplus_P T_2) \ominus_{P'} T_1$. If we omit constraint (ii) then an invalid compound term according to an expression $T_1 \oplus_{P_1} T_2$ could be valid according to a larger expression $(T_1 \oplus_{P_1} T_2) \oplus_{P_2} T_3$. Additionally, if we omit constraint (iii) then a valid compound term according to an expression $T_1 \oplus_P T_2$ could be invalid according to a larger expression $(T_1 \oplus_P T_2) \ominus_N T_3$.

This monotonic behaviour in the evaluation of a well-formed expression results in a number of useful properties. Specifically, due to their monotonicity, well-formed expressions can be formulated in a systematic, gradual manner (intermediate results of subexpressions are not invalidated by larger expressions).

[7] Proof of this property is given in Section 3.

A comprehensive example of the algebra that also demonstrates the benefits of monotonicity can be found in [TASC03].

In the rest of the paper, we assume that expressions are well-formed. In [TASC03], we presented the algorithm $IsValid(e, s)$ that checks the validity of a compound term according to a (well-formed) expression e, and the algorithm $wellFormed(e)$ that checks if an expression e is well-formed. Both algorithms have polynomial time complexity.

3 Semantic Interpretation

At first we shall give a model-theoretic interpretation to faceted taxonomies and to compound taxonomies. Using this framework, we shall formally define the validity of a compound term. In the sequent, we will define the models of the compound taxonomies that *satisfy* a (well-formed) algebraic expression. At that point, it will become evident that the algebraic operations and their parameters actually pose constraints to the models of the compound taxonomy $(P(\mathcal{T}), \preceq)$. Moreover, we will show that the operations as defined in Section 2, are also justified by the semantic interpretation of this section.

We conceptualize the world as a set of objects, that is, we assume an arbitrary domain of discourse and a corresponding set of objects Obj. A typical example of such a domain is a set of Web pages. The only constraint that we impose on the set Obj is that it must be a denumerable set.

The set of objects described by a term is the *interpretation* of that term.

Definition 10. Given a terminology $\mathcal{T}$, we call *interpretation* of $\mathcal{T}$ over Obj any function $I : \mathcal{T} \rightarrow 2^{Obj}$.

Intuitively, the interpretation $I(t)$ of a term t is the set of objects to which the term t is correctly applied. In our discussion the set Obj will be usually understood from the context. So, we shall often say simply "an interpretation" instead of "an interpretation over Obj". Interpretation, as defined above, assigns to a term denotational or extensional meaning ([Woo91]).

Now, any interpretation I of $\mathcal{T}$ can be extended to an interpretation $\hat{I}$ of $P(\mathcal{T})$ as follows:

$$\hat{I}(\{t_1, ..., t_n\}) = I(t_1) \cap I(t_2) \cap ... \cap I(t_n)$$

Definition 11. Let $(\mathcal{T}, \leq)$ be a taxonomy. An interpretation I of $\mathcal{T}$ is a *model* of $(\mathcal{T}, \leq)$,
if for each $t, t' \in \mathcal{T}$: if $t \leq t'$ then $I(t) \subseteq I(t')$.

Proposition 1. Let S be a compound taxonomy over a taxonomy $\mathcal{T}$, and let s and s' be two elements of S. It holds:

$$s \preceq s' \text{ iff } \hat{I}(s) \subseteq \hat{I}(s') \text{ in every model } I \text{ of } (\mathcal{T}, \leq)$$

We can see that the compound ordering $\preceq$ is also justified semantically (it coincides with extensional subsumption).

Definition 12. Let $(\mathcal{T}, \leq)$ be a taxonomy. An interpretation $\hat{I}$ of $P(\mathcal{T})$ is a *model* of $(P(\mathcal{T}), \preceq)$,
if for each $s, s' \in P(\mathcal{T})$: if $s \preceq s'$ then $\hat{I}(s) \subseteq \hat{I}(s')$.

From the above, it easily follows that:

an interpretation I is a model of $(\mathcal{T}, \leq)$ iff $\hat{I}$ is a model of $(P(\mathcal{T}), \preceq)$

For brevity hereafter we shall denote by I both I and $\hat{I}$. Additionally, in the following, by model I we refer to a model I of $(\mathcal{T}, \leq)$.

Let us now define compound term validity according to an expression e. For simplicity, we consider only expressions of the form $e \oplus_P e'$ and $e \ominus_N e'$.

Before we define recursively the valid and invalid compound terms of $e \; op \; e'$, we define the *valid genuine compound terms* of $e \; op \; e'$ (denoted by $VG(e \; op \; e')$) and the *invalid genuine compound terms* of $e \; op \; e'$ (denoted by $IG(e \; op \; e')$). That is, we first define the validity of the genuine compound terms associated with each operation.

Below we define the valid genuine compound terms of $e \oplus_P e'$:

$$VG(e \oplus_P e') = \{ \; s \in G_{S_e, S_{e'}} \mid I(s) \neq \emptyset \text{ in every model } I \text{ such that: } \; I(s') \neq \emptyset, \forall s' \in P \}$$

Now we adopt a closed-world assumption for the invalid genuine compound terms, specifically we assume that all elements of $G_{S_e, S_{e'}} \setminus VG(e \oplus_P e')$ are invalid. Thus we write:

$$IG(e \oplus_P e') = G_{S_e, S_{e'}} \setminus VG(e \oplus_P e')$$

The following proposition holds:

Proposition 2. $VG(e \oplus_P e') = Br(P) \cap G_{S_e, S_{e'}}$

Below we define the invalid genuine compound terms of $e \ominus_N e'$:

$$IG(e \ominus_N e') = \{ \; s \in G_{S_e, S_{e'}} \mid I(s) = \emptyset \text{ in every model } I \text{ such that: } \quad I(s') = \emptyset, \forall s' \in N \}$$

Now we again adopt a closed-world assumption for the valid genuine compound terms, specifically we assume that all elements of $G_{S_e,S_{e'}} \setminus IG(e \ominus_N e')$ are valid. Thus we write:

$$VG(e \ominus_N e') = G_{S_e,S_{e'}} \setminus IG(e \ominus_N e')$$

The following proposition holds:

Proposition 3. $IG(e \ominus_N e') = Nr(N)$

Clearly, for any operation $e\ op\ e'$, the sets $VG(e\ op\ e')$ and $IG(e\ op\ e')$ *partition* the set $G_{S_e,S_{e'}}$.

One can easily see that the semantic interpretation that is given in this section, justifies the way that the algebraic operations were defined in Section 2. Indeed, it holds:

$$VG(e \oplus_P e') = (S_e \oplus_P S_{e'}) \cap G_{S_e,S_{e'}}$$
$$VG(e \ominus_N e') = (S_e \ominus_N S_{e'}) \cap G_{S_e,S_{e'}}$$

Until now, for every operation $e\ op\ e'$ we partitioned the set $G_{S_e,S_{e'}}$ to the sets $VG(e\ op\ e')$ and $IG(e\ op\ e')$. Let $\mathcal{T}_e$ denote the union of the terminologies of the facets that appear in e. We will show how for a given expression e we can partition the elements of $P(\mathcal{T}_e)$ into the set of *valid compound terms*, denoted by $VC(e)$, and the set of *invalid compound terms*, denoted by $IC(e)$. We define:

$$VC(T_i) = T_i, \text{ for } i = 1, ...k$$
$$VC(e\ op\ e') = VG(e\ op\ e') \cup VC(e) \cup VC(e')$$
$$IC(e) = P(\mathcal{T}_e) \setminus VC(e)$$

Clearly, the sets $VC(e)$ and $IC(e)$ constitute a partition of $P(\mathcal{T}_e)$.

Definition 13. Let e be an expression, and let I be a model of $(\mathcal{T}, \leq)$. We say that I *satisfies* e if
(1) $\forall\ s \in VC(e),\ I(s) \neq \emptyset$,
(2) $\forall\ s \in IC(e),\ I(s) = \emptyset$.

The following proposition expresses that every expression e is satisfiable.

Proposition 4. Let e be an expression. There always exists a model I of $(\mathcal{T}, \leq)$ that satisfies e.

The following proposition expresses that the compound taxonomy S_e of an algebraic expression e (as computed from our operations) consists of exactly those compound terms which are valid according to the semantic interpretation that we described in this section.

Proposition 5. Let e be an expression. It holds:

$$VC(e) = S_e \text{ and } IC(e) = P(\mathcal{T}_e) \setminus S_e$$

The following proposition gives a very important property of our theory, that is, intermediate results of subexpressions are not invalidated by larger expressions. Thus, expressions can be formed in a constructive, gradual manner, allowing use of intermediate results.

Proposition 6. Let e be an expression and e' be a subexpression of e. Then, it holds

$$VC(e') \subseteq VC(e) \text{ and } IC(e') \subseteq IC(e)$$

To see the significance of this proposition, let $\{F_1, ..., F_k\}$ be the facets of a faceted taxonomy and let e' be an expression that defines the current desired compound taxonomy. Assume that now the designer adds some new facets of interest. Then, he has only to extend (and not to rewrite) e' with a subexpression e'' such that the new expression, $e = e' \; op \; e''$, defines the new desired compound taxonomy.

The following proposition expresses that the valid and invalid genuine compound terms of a subexpression of an expression e are indeed valid and invalid compound terms, respectively.

Proposition 7. Let e be an expression and $e_1 \; op \; e_2$ be a subexpression of e. Then, it holds

$$VG(e_1 \; op \; e_2) \subseteq VC(e) \text{ and } IG(e_1 \; op \; e_2) \subseteq IC(e)$$

From the above proposition and propositions 2 and 3, it easily follows that for any parameter P and N of e, it holds: $P \subseteq VC(e)$ and $N \subseteq IC(e)$.

In Section 2, we informally indicated that if a compound term s is valid then every compound term in $\mathrm{Br}(s)$ is also valid. Additionally, if a compound term s is invalid then every compound term in $\mathrm{Nr}(s)$ is also invalid. This property is formally proved in the following proposition.

Proposition 8. Let e be an expression. It holds:

$$Br(VC(e)) = VC(e) \text{ and } Nr(IC(e)) = IC(e)$$

The semantic interpretation that we described can be extended in a straightforward manner, so as to also capture the plus-self-product operation and the minus-self-product operation.

4 Comparison with Description Logics

Below we investigate whether we can represent the compound taxonomies defined by our algebra, in Description Logics [DLNS96]. Recall that any Description Logic (DL) is a fragment of First Order Logic (FOL). In particular, any (basic) DL is a subset of $\mathcal{L}_3$ i.e. the function-free FOL using at most *three* variable names.

In DL, a knowledge base, also referred as a DL theory is formed by two components: the *intensional* one, called TBox, and the *extensional* one, called ABox. The former contains the definitions of predicates, while the latter contains the assertions over constants.

One can easily see that a faceted taxonomy $F = (\mathcal{T}, \leq)$ can be expressed as a TBox containing one primitive concept t for each term $t \in \mathcal{T}$, and one expression $t \dot{\leq} t'$ for each relationship $t \leq t'$. Let K_F denote the TBox that is derived in this way.

Table 1 describes a method for deriving a TBox K_e for a (well-formed) expression e of our algebra[8]. Note that a compound term $s = \{t_1, ..., t_n\}$ is written as a DL expression $d_s = t_1 \sqcap ... \sqcap t_n$. In the case of a plus-product operation we derive a new concept $x \dot{\leq} t_1 \sqcap ... \sqcap t_n$ for each element $\{t_1, ..., t_n\} \in P$, while in the case of a minus-product operation we derive an expression $t_1 \sqcap ... \sqcap t_n \dot{\leq} \bot$ for each element $\{t_1, ..., t_n\} \in N$.

e	K_e
$e_1 \oplus_P e_2$	$K_{e_1} \cup K_{e_2} \cup$ $\{ x \dot{\leq} t_1 \sqcap ... \sqcap t_n \mid \{t_1, ..., t_n\} \in P\}$
$e_1 \ominus_N e_2$	$K_{e_1} \cup K_{e_2} \cup$ $\{ t_1 \sqcap ... \sqcap t_n \dot{\leq} \bot \mid \{t_1, ..., t_n\} \in N\}$
$\overset{*}{\oplus}_P (T_i)$	$K_{T_i} \cup \{ x \dot{\leq} t_1 \sqcap ... \sqcap t_n \mid \{t_1, ..., t_n\} \in P\}$
$\overset{*}{\ominus}_N (T_i)$	$K_{T_i} \cup \{ t_1 \sqcap ... \sqcap t_n \dot{\leq} \bot \mid \{t_1, ..., t_n\} \in N\}$
T_i	$\{ t \dot{\leq} t' \mid$ for each $t, t' \in T_i,$ s.t $t \leq t'\}$

Table 1. Using DL for representing the compound terminology of an expression

Now, Table 2 sketches how we can check whether a compound term $s = \{t_1, ..., t_n\}$ belongs to the compound terminology S_e of an expression e by using K_e and the inference mechanisms of DL. In this table we consider that $s_1 = \{t \in s \mid F(t) \in F(e_1)\}$ and $s_2 = \{t \in s \mid F(t) \in F(e_2)\}$.

[8] It is important that the expression e be well-formed. Otherwise, the TBOX may be inconsistent.

Our approach	*A DL-based approach*
$IsValid(e_1 \oplus_P e_2, s)$=TRUE	$(K_{e_1 \oplus_P e_2} \models d_s) \vee$ $IsValid(e_1, s) \vee$ $IsValid(e_2, s)$
$IsValid(e_1 \ominus_N e_2, s)$=TRUE	$(K_{e_1 \ominus_N e_2} \not\models d_s \equiv \bot) \wedge$ $IsValid(e_1, s_1) \wedge$ $IsValid(e_2, s_2)$
$IsValid(\overset{*}{\oplus}_P (T_i), s)$=TRUE	$K_{\overset{*}{\oplus}_P(T_i)} \models d_s$
$IsValid(\overset{*}{\ominus}_N (T_i), s)$=TRUE	$K_{\overset{*}{\ominus}_N(T_i)} \not\models d_s \equiv \bot$

Table 2. Using DL for checking the validity of a compound term

It is evident that we *cannot* check the validity of a compound term s according to an expression e by just using the classical satisfiability check of DL, that is by just checking if $K_e \models d_s$. However, if we would like to use a DL-based system for implementing our algebra then we should write (on top of DL) a *metasystem* that parses the expression e and recursively calls the inference mechanisms of DL as described in Table 2. The recursive calls are based on our theory, and are exactly those of algorithm $IsValid(e, s)$ (given in [TASC03]). Indeed the only difference of the DL metasystem with algorithm $IsValid(e, s)$ is that the DL checks: $K_{e_1 \oplus_P e_2} \models d_s$ and $K_{\overset{*}{\oplus}_P(T_i)} \models d_s$ are replaced by: $\exists p \in P$ such that $p \preceq s$. Similarly, the DL checks: $K_{e_1 \ominus_N e_2} \not\models d_s \equiv \bot$ and $K_{\overset{*}{\ominus}_N(T_i)} \not\models d_s \equiv \bot$ are replaced by: $\not\exists n \in N$ such that $s \preceq n$. In addition, our theory is needed to guarantee that expressions are well-formed and thus, there are no inconsistencies. We do not describe here this metasystem in detail as this would not offer anything new in our discussion.

Alternatively, if we want to use the classical satisfiability check of Description Logics, then we cannot create the TBox by the method described in Table 1. Instead, we have to create a TBox that contains many more expressions, by translating each plus-product operation as described in Table 1, and by translating each minus-product operation $e_1 \ominus_N e_2$ to the following set of expressions:

$$K_{e_1} \cup K_{e_2} \cup \{\ x \dot{\leq} t_1 \sqcap ... \sqcap t_n \mid \text{ for each } \{t_1, ..., t_n\} \notin S_{e_1} \oplus S_{e_2} - Nr(N)\}$$

If we derive a TBox K_e of an expression e in this way, then it will hold $IsValid(e, s)$=TRUE iff $K_e \models s$. However, in this approach, it is necessary to pre-compute all S_{e_1} and S_{e_2}, which practically makes the approach useless. Moreover, K_e will be a very big TBox.

From the above discussion it is evident that if we want to check the validity of compound terms using the classical satisfiability check of DL, then we cannot

represent our algebraic expressions in DL in a straightforward manner. This is due to the closed-world assumptions inherent to our operations.

5 Concluding Remarks

In [TASC03], we proposed an algebra for specifying the valid compound terms of a faceted taxonomy. Although faceted classification was suggested quite long ago (by Ranganathan in the 1920s [Ran65], the associated issues have not received adequate attention by the computer science community. However, there are several works about facet analysis (e.g. see [Dun89], [Vic86],[LN77]). Facets have also been studied in library and information science (for a review see [Map95]). For instance, thesauri ([Int86]) may have facets that group the terms of the thesaurus in classes. Ruben Prieto-Diaz ([PD89,PD91]) has proposed "faceted classification" for a reusable software library. The contribution of our compound term composition algebra lies in enriching a faceted scheme with a rigorous method for specifying valid combinations of terms. This method can be used in order to construct taxonomies or thesauri which unlike existing thesauri, do not present the problems of missing terms or missing relationships (for more about this problem see [CTHD00]).

In this paper, we defined the semantics of the algebraic operations. Specifically, we justified the definition of the algebraic operations, based on the model-theoretic definition of the valid and invalid genuine compound terms. Having defined the valid (resp. invalid) genuine compound terms of a positive (resp. negative) operation, the invalid (resp. valid) genuine compound terms are computed based on a closed-world assumption. The valid compound terms according to an expression e is the union of the valid genuine compound terms of all operations of e.

Additionally, we defined the models of an algebraic expression. Intuitively, a model of an algebraic expression, is an intepretation which is non-empty for each valid compound term, and empty for each invalid compound term. We proved that every well-formed algebraic expression is satisfiable. Moreover, we proved that well-formed algebraic expressions are monotonic, which ensures that results of subexpressions are not invalidated by larger expressions.

We also showed that we cannot directly represent the compound taxonomies defined by our algebra directly in Description Logics, and a metasystem was designed on top of Description Logics to implement our algebra.

Our algebra can be used in any application that indexes objects using a faceted taxonomy. For example, it can be used for designing taxonomies for products, for fields of knowledge (e.g. for indexing the books of a library), etc. As we can infer the valid compound terms in a faceted taxonomy, we are able to generate navigation trees *on the fly*, having only valid compound terms as nodes. The algorithm for deriving navigation trees on the fly is given in [TASC03].

Our algebra can also be used for *configuration management*. Consider a product (e.g. a software module) whose functionality, or configuration, is determined by a number of parameters, where each parameter is associated with a finite number of values. However, some configurations may either make no sense, or not supported, or they may be "dangerous". For this purpose, the designer of the product can use an expression in order to specify all valid configurations. In this way, the user of the product can select one configuration from the valid ones. The advantage of using our approach is that the space needed for storing the valid configurations is low. This is quite important, as all information about configuration management has to be stored in the product itself, e.g. in the software module.

Interest in faceted taxonomies is also indicated by the development of XFML, Core-eXchangeable Faceted Metadata Language (http://xfml.org). XFML is a model to express topics organized in facets, and allows you to assign topics to any page on the web. XFML lets you publish this information in an open, XML based format. It can be easily seen that our algebra can also be applied in this context in order to prescribe the set of valid compound terms. This can prevent some of the indexing errors that may occur in an open and collaborative environment like the Web.

Currently, our algebra is been used for building the taxonomy of a tourist portal. The results that the designers report to us, concerning flexibility and ease of use, are so far very encouraging.

Finally, we have to note that the advantages of the compound faceted taxonomies that we propose (compactness, conceptual clarity, scalability, valid compound terms) can facilitate several other associated tasks. Specifically, they can certainly facilitate the design of *mediators* over several taxonomy-based sources (using the approach presented in [TSC01]), and the *personalization* of Web catalogs (using the approach presented in [STC02]).

As future work, we plan to study how updates on the faceted taxonomy, or changes to the desired compound terminology should update the expression that defines the compound terminology. This process can be automated. This is very important in practice, as it adds flexibility to the design process: the designer during the formulation of the expression e can update the faceted taxonomy, without having to bother that e will become obsolete. Additionally, the designer can add or delete compound terms from the desired compound terminology without having to worry that e will no longer reflect his/her desire.

In addition, we are going to study in depth, methodologies for the formulation of algebraic expressions that reflect the desire of the designer. Specifically, how the four operations of our algebra should be combined, so as the number of compound terms of the associated parameters P, N is minimal, and designer effort is minimized.

Acknowledgements

The first author wants to thank Tonia Dellaporta for inspiring him.

References

[CTHD00] Peter Clark, John Thompson, Heather Holmback, and Lisbeth Duncan. "Exploiting a Thesaurus-based Semantic Net for Knowledge-based Search". In *Procs of 12th Conf. on Innovative Applications of AI (AAAI/IAAI'00)*, pages 988–995, 2000.

[DLNS96] F.M. Donini, M. Lenzerini, D. Nardi, and A. Schaerf. "Reasoning in Description Logics". In Gerhard Brewka, editor, *Principles of Knowledge Representation*, chapter 1, pages 191–236. CSLI Publications, 1996.

[Dun89] Elizabeth B. Duncan. "A Faceted Approach to Hypertext". In Ray McAleese, editor, *HYPERTEXT: theory into practice, BSP*, 1989.

[Int86] International Organization For Standardization. "Documentation - Guidelines for the establishment and development of monolingual thesauri", 1986. Ref. No ISO 2788-1986.

[LN77] P. H. Lindsay and D. A. Norman. *Human Information Processing.* Academic press, New York, 1977.

[Map95] Amanda Maple. "Faceted Access: A Review of the Literature", 1995. http://theme.music.indiana.edu/tech_s/mla/facacc.rev.

[PD89] Ruben Prieto-Diaz. "Classification of Reusable Modules". In *Software Reusability. Volume I*, chapter 4, pages 99–123. acm press, 1989.

[PD91] Ruben Prieto-Diaz. "Implementing Faceted Classification for Software Reuse". *Communications of the ACM*, 34(5):88–97, 1991.

[Ran65] S. R. Ranganathan. "The Colon Classification". In Susan Artandi, editor, *Vol IV of the Rutgers Series on Systems for the Intellectual Organization of Information.* New Brunswick, NJ: Graduate School of Library Science, Rutgers University, 1965.

[STC02] Nicolas Spyratos, Yannis Tzitzikas, and Vassilis Christophides. "On Personalizing the Catalogs of Web Portals". In *15th International FLAIRS Conference, FLAIRS'02*, pages 430–434, Pensacola, Florida, May 2002.

[TASC03] Yannis Tzitzikas, Anastasia Analyti, Nicolas Spyratos, and Panos Constantopoulos. "An Algebraic Approach for Specifying Compound Terms in Faceted Taxonomies". In *13th European-Japanese Conference on Information Modelling and Knowledge Bases*, Kitakyushu, Japan, June 2003.

[TSC01] Yannis Tzitzikas, Nicolas Spyratos, and Panos Constantopoulos. "Mediators over Ontology-based Information Sources". In *Proceedings of the 2nd International Conference on Web Information Systems Engineering, WISE 2001*, pages 31–40, Kyoto, Japan, December 2001.

[Tzi02] Yannis T. Tzitzikas. *"Collaborative Ontology-based Information Indexing and Retrieval"*. PhD thesis, Department of Computer Science - University of Crete, September 2002.

[Vic86] B. C. Vickery. "Knowledge Representation: A Brief Review". *Journal of Documentation*, 42(3):145–159, 1986.

[Woo91] W. A. Woods. "Understanding Subsumption and Taxonomy". In *Principles of Semantic Networks*, chapter 1. Morgan Kaufmann Publishers, 1991.