Using OODB Modeling to Partition a Vocabulary into Structurally and Semantically Uniform Concept Groups

Li-min Liu, Michael Halper, James Geller, and Yehoshua Perl

Abstract—Controlled Vocabularies (CVs) are networks of concepts that unify disparate terminologies and facilitate the process of information sharing within an application domain. We describe a general methodology for representing an existing CV as an object-oriented database (OODB), called an Object-Oriented Vocabulary Repository (OOVR). A formal description of the OOVR methodology, which is based on a structural abstraction technique, is given, along with an algorithmic description and a number of theorems pertaining to some of the methodology's formal characteristics. An OOVR offers a two-level view of a CV, with the schema-level view serving as an important abstraction that can aid in orientation to the CV's contents. While an OOVR can also assist in traversals of the CV, we have identified certain special CV configurations where such traversals can be problematic. To address this, we introduce—based on the original methodology—an enhanced OOVR methodology that utilizes both structural and semantic features to partition and model a CV's constituent concepts. With its basis in the notions of area and the recursively defined articulation concept, an enhanced OOVR representation provides users with an improved CV view comprising groups of concepts uniform both in their structure and semantics. An algorithmic description of the singly rooted OOVR methodology and theorems describing some of its formal properties are given. The results of applying it to a large existing CV are discussed.

Index Terms—Object-oriented databases, object-oriented models, object-oriented systems, knowledge representation, database models.

1 Introduction

A controlled vocabulary (CV) is a structure that houses knowledge in the form of concepts, subsumption links, and semantic relationships. CVs have become integral components of many information processing environments, particularly within the healthcare field. Among their primary benefits are their support for information sharing and integration, decision-support, and ad hoc querying of domain (e.g., medical) knowledge [7], [32]. Examples of such systems from the healthcare domain include MeSH [23], CPT98 [1], SNOMED [8], ICD9-CM [33] (all of which have been integrated into the Unified Medical Language System (UMLS) [16], [18]), GALEN's Core Model [29] (expressed in GRAIL [28]), and the Medical Entities Dictionary (MED) [6]. (References to related work on semantic networks [4], knowledge representation languages [22], and ontologies [24] can be found in [19], [20].)

One major aspect of many CVs is their enormous size and scope. A CV can easily consist of many thousands of concepts with a proportional number of interconcept relationships. Given this fact, it may be hard for potential users and even a CV's own designers to orient themselves to the vast content of a CV and exploit its many advantages.

• L.-M. Liu and M. Halper are with the Department of Mathematics and Computer Science, Kean University, Union, NJ 07083.
E-mail: {lliu, mhalper}@kean.edu.
• J. Geller and Y. Perl are with the Computer and Information Science Department, New Jersey Institute of Technology, Newark, NJ 07102.
E-mail: {geller, perl}@homer.njit.edu.

Manuscript received 19 May 1999; revised 4 Jan. 2001; accepted 30 Jan. 2001. For information on obtaining reprints of this article, please send e-mail to: tkde@computer.org, and reference IEEECS Log Number 109857.

In previous work, we have devised a novel technique for modeling a CV as an object-oriented database (OODB) [2], [3], [5], [12], [17], [21], [34], a form we call an Object-Oriented Vocabulary Repository (OOVR) [19], [20]. Using our methodology, we have constructed OOVRs based on the MED and the InterMED [25]. Both OOVRs are up and running in the ONTOS DB/Explorer [26], [31], a commercial OODB management system. Access to the InterMED OOVR is available on the Web in two forms [11], [27].

We have shown that the OOVR representation aids in vocabulary orientation and comprehension by providing an abstraction of the underlying CV contents. The schematic representation also helps in uncovering errors and inconsistencies that may have been introduced into a CV during its original development and subsequent refinement and expansion [13], [14].

In its original form, the OOVR methodology was presented as a two-phase process, with an initial phase followed by a refinement phase [19], [20]. In this paper, we first give a unified presentation of the methodology and prove some formal characteristics of OOVR representations. We also present a complete algorithmic description of the methodology.

An additional benefit of an OOVR is its support for more efficient browsing and traversal of a CV. However, during our experimentation with OOVR representations, we have encountered some special cases where small portions of a CV's concept configuration hindered the traversal process. The problems stemmed primarily from the fact that the OOVR methodology groups concepts together into an abstract entity when they have the same structure but not necessarily uniform semantics.

To address these issues, we present an enhanced OOVR methodology based on a revised partitioning scheme that performs a two-step decomposition of the source CV. As with the original OOVR technique, the first step breaks down a CV into collections of concepts, called *areas*, which have members exhibiting identical structure. In the second step, a special kind of area (called a *multirooted intersection area*) is further partitioned into collections of concepts, called *partial areas*, containing concepts uniform in their structure and their semantics. The partial areas are based on the recursively defined notion of *articulation concept*. The new kind of OOVR schema that emerges from this process has classes that are all "singly rooted," i.e., the subnetworks of the CV which are the classes' extensions each have a unique root concept. We will present the singly rooted OOVR methodology in its algorithmic form, along with theorems pertaining to various formal characteristics of singly rooted OOVR representations.

The remainder of this paper is organized as follows: In Section 2, we discuss the general structure of a CV. Section 3 presents the original OOVR methodology, including an algorithmic specification of its partitioning process, the details of the construction of an OOVR schema, and theorems that capture various formal characteristics of OOVR representations. In Section 4, we present a sample CV traversal using an OOVR in order to demonstrate the advantages of the extra abstraction layer afforded by the OOVR schema. Section 5 describes some difficulties that can arise in certain special cases of OOVR traversals. Then, in Section 6, we describe the formal aspects of the singly rooted OOVR methodology, including an algorithmic description and theorems about formal aspects of singly rooted OOVR representations. Section 7 presents the results of applying the enhanced methodology to the MED. Conclusions follow in Section 8.

2 Structure of a CV

A common formalism utilized in the construction of a CV is the semantic network, where each node is used to represent a unique concept from the knowledge domain. All concepts can exhibit two kinds of properties: 1) Attributes whose values are derived from some data types (such as integer or text string) and 2) relationships which are references to other concepts in the CV. Formally, an attribute is a mapping of a concept to a data type, while a relationship is a mapping of one concept to other concepts. For a concept v, we will use $P(v)$ to denote the set of all v's properties.

Each concept in a CV is defined with the attribute *name* that holds the concept's associated *term* (i.e., printable value) [10]. In order to satisfy the nonambiguity and synonymy criteria for CVs (proposed in [6], [7]), it is assumed that each concept also has the attribute *synonyms* whose value is the entire set of acceptable secondary names for a concept. The concept subsumption (IS-A) hierarchy is a fundamental aspect of a CV. Structurally, it is an acyclic collection of IS-A links, each of which connects a subconcept to a related superconcept. The multiple classification criterion [6], [7] requires that the IS-A hierarchy be a directed acyclic graph (DAG), allowing for any concept to have multiple parents.

The IS-A hierarchy plays two important roles. First, it supports subsumption-based reasoning. For example, a user wishing to know if a patient is taking antibiotics and knowing already that he is on Tetracycline can consult the CV to learn that **Tetracycline** IS-A **Antibiotic**.[1] The second aspect of the IS-A hierarchy is inheritance: A subconcept inherits all the properties exhibited by its superconcepts. For example, the concept **Sodium Test** IS-A **Test** and, therefore, the set of properties of **Sodium Test** is a superset of the properties of **Test**. If a concept has multiple parents, then it inherits properties from each of them.

Another assumption that we make, without loss of generality, is that a CV satisfies the following rule [6]:

Rule (Uniqueness of Property Introduction). A given property x can only be introduced at one concept in the CV.

A CV is also assumed, without loss of generality, to have a single root at the top of its IS-A hierarchy. We will refer to the root concept as **Entity**, which is defined to have the attributes *name* and *synonyms*. By inheritance, all other concepts in the CV will have these attributes, too.

We will be using the following notation when drawing a CV: A concept is a rectangle having rounded corners with its name written inside. Any attributes introduced by the concept (when shown) are listed below the name and are separated from it by a line. A relationship is a labeled arrow from the source concept to the target concept. Fig. 1 shows a portion of CV with three concepts: **Test**, **Glucose Test**, and **Substance**. The concept **Test** introduces the attributes *units* and *normal-value* and the relationship *measures* to **Substance**. **Substance** introduces the relationship *is-measured-by* (the converse of *measures*) but no attributes. **Glucose Test** IS-A **Test** and, therefore, inherits all of **Test's** properties.

3 OOVR Methodology

3.1 Partitioning a CV into Areas

Our OODB modeling of a CV is based on a structural abstraction of its network. The abstraction is derived from a partitioning of the network with respect to the notion of *area*. After defining area and other fundamental terminology, we prove some formal characteristics of the partition and its elements.

Definition 1 (Area). *An area of a CV is an induced subgraph [9] which contains all concepts that have the exact same properties.*

A CV is partitioned by its areas since each concept belongs to one and only one area. As we shall see, the partitioning of the CV into areas closely follows the property-introducing and inheritance patterns of the IS-A hierarchy and this partition can be automatically identified in a top-down manner.

Definition 2 (Property set of an area). *For an area A, $P(A)$ denotes the set of properties of any (and all) of its constituent concepts.*

1. Some typographical conventions: A boldface font will be used when writing the names of concepts. Properties of concepts will appear in italics and will be written strictly in lowercase letters. Object classes will be written in italics and will start with uppercase letters.

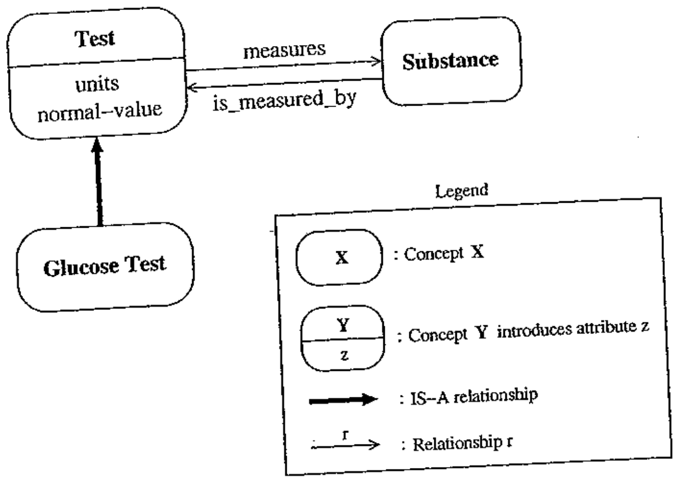

Fig. 1. Concepts **Test**, **Glucose Test**, and **Glucose**.

Definition 3 (Property-introducing concept). *A concept at which one or more new properties are introduced into the CV is called a property-introducing concept.*

An example of a property-introducing concept from the MED is **Pharmacy Items (Drugs and Nondrugs)** which, among other things, introduces the attribute *drug-trade-name*.

Definition 4 (Root of an area). *A concept* v *residing in area A is called a root of A if A contains no parents of* v.

The concept **Lab Diagnostic Procedure** is a root because its one parent **Diagnostic Procedure** belongs to a different area.

If an area has a single root, then the area is named after that concept. The area whose root is **Lab Diagnostic Procedure** is named "Lab Diagnostic Procedure Area."

Definition 5 (Property-introducing area). *An area containing a property-introducing concept is called a property-introducing area.*

An example is the Lab Diagnostic Procedure Area. In addition to property-introducing area, there is another kind of area defined in terms of *intersection concept*:

Definition 6 (Intersection concept). *Let* v *be a concept which is not a property-introducing concept and which has multiple superconcepts* $w_1, w_2, \ldots, w_n$ $(n > 1)$. *The concept* v *is called an intersection concept if the following condition holds:* $\forall i : 1 \leq i \leq n$, $P(v) \neq P(w_i)$. *That is, the set of properties of* v *differs from all of its parents' sets of properties. Note that* $P(v) = \bigcup_{i=1}^{n} P(w_i)$.

We use the designation "intersection concept" because v lies at the junction of (at least) two independent inheritance paths.

Definition 7 (Intersection area). *An area containing an intersection concept is called an intersection area.*

Fig. 2 shows an example with three property-introducing areas (A Area, B Area, and C Area) and an intersection area. The only concepts in the property-introducing areas with their names displayed are **A**, **B**, and **C**, the respective roots. **A** introduces the attribute a, **B**, the attribute b, and **C**, the attribute c. **A** also introduces the relationship r directed to **C**, which itself introduces relationship r', the converse of r.

D Area (Fig. 2) is an intersection area. Unlike a property-introducing area, an intersection area can have more than one root. D Area has two roots, **D** and **E**, both of which have two parents, one residing in B Area and the other in C Area. For a multirooted intersection area, the first identified root is used to name the area. The concept **D** was identified as a member of this area first and, hence, the area was named D Area. The concepts **F** and **G** are members of D Area because they are children of **D** and **E**, respectively. **F** and **G** are not roots of D Area. None of the concepts in D Area has any intrinsic properties. All properties are inherited from outside the area. It is not possible for an intersection area to contain a property-introducing concept since such a concept would induce a new property-introducing area.

In the following, we present some formal characteristics of the partition of a CV in terms of areas.

Lemma 1. *A property-introducing concept is a root of its area.*

Proof. Let v be a property-introducing concept that introduces property p. If v is the concept **Entity**, then v is clearly a root of its area. Otherwise, v has parents in the CV. Thus, the parents of v do not have the property p and the property sets of the parents must all be different from P(v). Hence, none of v's parents reside in v's area.□

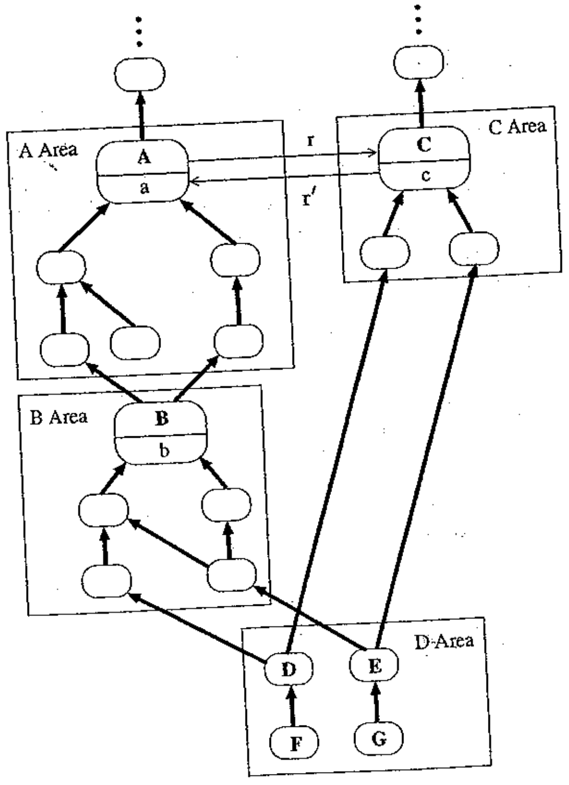

Fig. 2. Four areas of a CV.

Lemma 2. *A root of a property-introducing area is a property-introducing concept.*

Proof. Assume to the contrary that a root $\mathbf{r}$ of a property-introducing area A is not a property-introducing concept. Let $\mathbf{v}$ be a property-introducing concept contained in A and let p be a property that $\mathbf{v}$ introduces. By Lemma 1, $\mathbf{v}$ is also a root of A. Since $\mathbf{r}$ is a root of A, it is not a descendant of $\mathbf{v}$ and, thus, it does not have the property p. But, this implies that $\mathbf{r}$ and $\mathbf{v}$ are in different areas—a contradiction. □

Lemma 3. *All areas have at least one root.*

Proof. Areas are induced subgraphs [9] of the CV and, therefore, because the overall IS-A hierarchy of a CV is a DAG, each subhierarchy contained in an area will be a DAG, too. Since a DAG must have at least one root, the same is true of an area. □

Lemma 4. *A property-introducing area has exactly one root.*

Proof. By Lemma 3, an area must have at least one root. Suppose to the contrary that a property-introducing area A has at least two roots $\mathbf{r}_1$ and $\mathbf{r}_2$. By Lemma 2, the root $\mathbf{r}_1(\mathbf{r}_2)$ is a property-introducing concept introducing, say, a property p_1 (p_2). By the "uniqueness of property introduction" rule (see Section 2), $p_1 \neq p_2$. However, $\mathbf{r}_2$ is not a descendent of $\mathbf{r}_1$ since $\mathbf{r}_2$ is also a root of A. Hence, $\mathbf{r}_2$ does not inherit the property p_1 introduced by $\mathbf{r}_1$. Therefore, $\mathrm{P}(\mathbf{r}_1) \neq \mathrm{P}(\mathbf{r}_2)$ and $\mathbf{r}_1$ and $\mathbf{r}_2$ do not reside in the same area—a contradiction. □

An intersection area, in contrast, can have multiple roots (Fig. 2). Lemmas 1, 2, and 4 together give us:

Theorem 1. *There is a one-to-one correspondence between the property-introducing concepts, property-introducing areas, and the roots of these areas.*

Corollary 1. *The number of property-introducing areas is equal to the number of property-introducing concepts.*

By Corollary 1 and the uniqueness of property introductions, there is at most one property-introducing area for each property, which gives us:

Corollary 2. *The number of property-introducing areas is bounded by the overall number of different properties defined in the CV.*

Note that, when several properties are introduced at the same concept, there is only one corresponding area introducing them.

Lemma 5. *There are only property-introducing areas and intersection areas.*

Proof. Let A be an arbitrary area rooted at $\mathbf{r}_A$. Let $\mathbf{w}_1, \mathbf{w}_2, \ldots, \mathbf{w}_n (n \geq 1)$ be the parents of $\mathbf{r}_A$. Note that $\forall i : 1 \leq i \leq n, \mathrm{P}(\mathbf{w}_i) \neq \mathrm{P}(\mathbf{r}_A)$. If the union of the property sets of $\mathbf{r}_A$'s parents is different from $\mathbf{r}_A$'s property set, i.e., $\bigcup_{i=1}^{n} \mathrm{P}(\mathbf{w}_i) \neq \mathrm{P}(\mathbf{r}_A)$, then there is some property introduced at $\mathbf{r}_A$. In that case, $\mathbf{r}_A$ is a property-introducing concept and A is a property-introducing area. Otherwise, by Definition 6, $\mathbf{r}_A$ is an intersection concept since $\forall i : 1 \leq i \leq n, \mathrm{P}(\mathbf{w}_i) \neq \mathrm{P}(\mathbf{r}_A)$, and A is thus an intersection area. □

Theorem 2. *A CV is partitioned into disjoint areas which are either property-introducing areas or intersection areas.*

Proof. Areas are disjoint by definition. By Lemma 5, every area is either a property-introducing area or an intersection area. □

Below, we present the algorithm that partitions a CV into its respective areas. The algorithm operates in a top-down manner in its processing of the concepts of a CV. Its input is a complete CV and its output is the CV's entire set of areas. An area will be named after a property-introducing concept or a first-encountered intersection concept. We refer to these concepts as "naming concepts."

In the algorithm, $\mathcal{A}_\mathbf{v}$ will denote a set of concepts, each of which has the same set of properties, with $\mathbf{v}$ as its naming concept. $\mathcal{S}$ is a set which holds all naming concepts. $\mathcal{A}_{\mathrm{ALL}}$ is a set which will contain all $\mathcal{A}_\mathbf{v}$'s. At the end, $\mathcal{A}_{\mathrm{ALL}}$ will be returned. Each element $\mathbf{v}$ in $\mathcal{S}$ will later be used to name an area with the format *v_Area*. Every element $\mathbf{v}$ in $\mathcal{S}$ will have an associated set $\mathcal{A}_\mathbf{v}$ in $\mathcal{A}_{\mathrm{ALL}}$. Every concept $\mathbf{v}$ has an associated counter for unprocessed parents which is denoted as "p-counter[v]." This algorithm uses two auxiliary functions: "Num_parents_of" and "Is_proper_introducing." Num_parents_of takes a concept as input and returns its number of parents. Is_property_introducing takes a concept as input and returns "true" if it is a property-introducing concept and "false" otherwise. Statements with the same indentation are in the same block. A concept can be processed only if all its parents have been processed. Therefore, initially, the root of a CV is processed (see Table 1).

Let us illustrate the construction of a set $\mathcal{A}$ constituting an intersection area with multiple roots. Suppose the first concept processed in the intersection area D Area (Fig. 2) is **D**. We create a set $\mathcal{A}_{\mathrm{D}}$ with **D** as its first element. Later, we will visit the concept **E**, the other root of this area. We compare its property set to that of the concepts **A**, **B**, **C**, and **D** in the set $\mathcal{S}$ of naming concepts. The property sets of concepts **A**, **B**, and **C** do not match that of **E**, but **D**'s does match. Therefore, **E** will be inserted into the existing set $\mathcal{A}_{\mathrm{D}}$. When **E** is processed, the p-counter of **G** is reduced from 1 to 0 and **G** is inserted into the queue. Later on, when **G** is deleted from the queue, it has only one parent **E** and, thus, is added to set $\mathcal{A}_{\mathrm{D}}$.

3.2 OOVR Schema

In the OODB-version of the CV, each concept is represented by a unique object. The OOVR's schema is constructed automatically after the identification of all areas. There is a one-to-one correspondence between the areas in the CV and the classes in the OOVR's schema. That is, one class is defined to represent one area. The direct extension of a given class is identical to the set of concepts in the corresponding area in the CV. Due to this, we refer to the classes in the OOVR schema as *area classes*. If the area is a property-introducing area, then we have a *property-introducing class*. Likewise, for an intersection area, there is an *intersection class*. In an OODB schema, a class defines a set of objects whose structure and behavior are the same. In our

TABLE 1
Set_of_Areas FUNCTION AREA_Partition(*CV* **V**)

```
set_of_areas FUNCTION AREA_Partition($CV$ V)
BEGIN
    // initialization
    $\mathcal{Q}$ := newqueue();              // $\mathcal{Q}$ contains concepts ready to be processed.
    $\mathcal{A}_{ALL}$ := newset();             // $\mathcal{A}_{ALL}$ will hold all areas.
    $\mathcal{S}$ := newset();                // $\mathcal{S}$ will hold property-introducing concepts and
                                  // first-encountered intersection concepts.
    FOR EACH concept v in V DO   // Initialize p-counter for all concepts.
            p-counter[v] := Num_parents_of(v);
    enqueue($\mathcal{Q}$, $root$(V));           // Insert Entity into $\mathcal{Q}$.

    WHILE ( NOT emptyqueue($\mathcal{Q}$) ) DO
            v := dequeue($\mathcal{Q}$);
            // If v introduces new properties, we generate a new property-introducing area.
            IF ( Is_property_introducing(v) ) THEN
                    insert($\mathcal{S}$, v);
                    $\mathcal{A}_v$ := newset();      // Create a new set $\mathcal{A}_v$ to keep concepts in this area.
                    insert($\mathcal{A}_v$, v);       // Insert v into $\mathcal{A}_v$.
                    insert($\mathcal{A}_{ALL}$, $\mathcal{A}_v$);   // Insert the new set $\mathcal{A}_v$ into $\mathcal{A}_{ALL}$.
            // If v has only one parent, v is in the same area as its parent.
            ELSE IF ( Num_parents_of(v) = 1 ) THEN
                    Let w be the parent of v;
                    Let $\mathcal{A}_u \in \mathcal{A}_{ALL}$ be the set s.t. $w \in \mathcal{A}_u$;
                    insert($\mathcal{A}_u$, v);
            ELSE
                    // v has multiple parents $w_1, w_2, \ldots, w_n$ $(n > 1)$.
                    IF ( $\exists i$: $1 \leq i \leq n$ s.t. $P(v) = P(w_i)$ ) THEN   // v is not an intersection concept
                            Let $\mathcal{A}_u \in \mathcal{A}_{ALL}$ be the set s.t. $w_i \in \mathcal{A}_u$;
                            insert($\mathcal{A}_u$, v);
                    ELSE
                            //Concept v is an intersection concept.
                            Let flag $found$ := false;
                            // Determine whether v is the first-encountered intersection concept in its area.
                            // It is necessary to check the elements of $\mathcal{S}$ to determine whether this is the case.
                            FOR EACH element $c \in \mathcal{S}$ DO
                                    IF ( NOT $found$ AND $P(v) = P(c)$ ) THEN
                                    // Concept v is not the first-encountered intersection concept.
                                            insert($\mathcal{A}_c$, v);
                                            set flag $found$ := true;
                            IF ( NOT $found$ ) THEN
                                    // Concept v is the first-encountered intersection concept.
                                    insert($\mathcal{S}$, v);
                                    $\mathcal{A}_v$ := newset();
                                    insert($\mathcal{A}_v$, v);
                                    insert($\mathcal{A}_{ALL}$, $\mathcal{A}_v$);
            // After we process concept v, we need to decrease the p-counters of all v's children by one.
            // After the decrease, if any p-counter is equal to zero, we put the associated concept
            // into the queue since it is ready to be processed.
            FOR EACH child k of v DO
                    p-counter[k]−−;
                    IF ( p-counter[k] = 0 ) THEN
                            enqueue($\mathcal{Q}$, k);
    RETURN $\mathcal{A}_{ALL}$;
END □
```

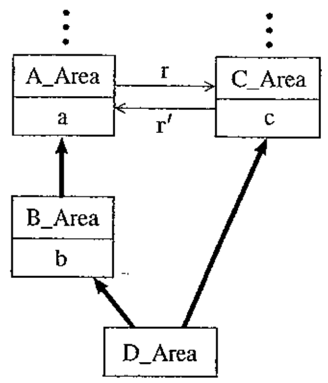

Fig. 3. Area classes for the areas in Fig. 2.

mapping, the instances of one class are exactly all those concepts that reside in a single area which, by definition, contains all concepts exhibiting identical properties.

The intrinsic properties of a property-introducing class are defined to be exactly those introduced by the root concept of its corresponding area. In addition, all the concepts in a property-introducing area must have the properties inherited by the root from its parent(s) in the CV. To capture this situation, the property-introducing class is placed in subclass relationships with those other area classes to which the parents of the root belong. In this way, the property-introducing class obtains all necessary properties: Some are defined intrinsically, while the others are inherited from other classes.

In Fig. 3, we illustrate the above by showing the classes *A_Area*, *B_Area*, and *C_Area* that represent the corresponding areas in Fig. 2. The classes are boxes with their names and attributes written inside. An ordinary relationship is a labeled arrow, while a subclass relationship is a bold arrow pointing from the subclass to the superclass. The ellipses indicate the omission of the subclass relationships of *A_Area* and *C_Area*. All property-introducing classes have at least one subclass relationship. The only exception is *Entity_Area*, the root of the OOVR schema.

Since an intersection area does not contain any property-introducing concepts, and, in fact, all properties of its concepts are obtained via inheritance, an intersection class does *not* introduce any properties of its own. Instead, it is defined to be a subclass of all other area classes which contain one or more parents of its root(s). An intersection class *always* exhibits multiple inheritance, i.e., it inherits from two or more superclasses.

Referring to Fig. 3 again, we see the intersection class *D_Area*, representing D Area. *D_Area* is a subclass of both *B_Area* and *C_Area* because its roots (**D** and **E**) have parents residing in both those respective areas.

Our mapping technique may generate a "shortcut" of SUBCLASS_OF links. That is, it may happen that *Y* SUBCLASS_OF *Z*, *X* SUBCLASS_OF *Y*, and *X* SUBCLASS_OF *Z*. In this case, the SUBCLASS_OF link from *X* to *Z* is a shortcut of the two links connecting *X* to *Y* and *Y* to *Z*. We have made the decision to omit this kind of subclass relationship from the OOVR schema because it does not contribute to inheritance.

The final aspect of the mapping pertains to the IS-A hierarchy. All concepts have IS-A connections to other concepts (except for the root **Entity**). In the original network, **Entity** has the multivalued relationship "subconcept_of" that implements the IS-A hierarchy of concepts. In the mapping, this is translated into a multivalued, reflexive relationship *subconcept_of*, defined at the class *Entity_Area*. In this way, all concepts (objects) in the OOVR representation have their required IS-A connections.

All OODB schemas must have acyclic subclass structures to avoid circular definitions of properties. Since the OOVR schema is derived by an algorithm, it remains for us to prove that its induced subclass configuration is indeed acyclic. This result follows from the fact that the IS-A hierarchy of any CV is acyclic.

Theorem 3. *The subclass relationships of an OOVR schema are acyclic.*

Proof. Assume to the contrary that an OOVR schema contains a cycle of area classes $Z_0, Z_1, \ldots, Z_m$ (see Fig. 4) with respect to the SUBCLASS_OF relationship. Let the naming concepts of these classes be $\mathbf{r}_{Z_0}, \mathbf{r}_{Z_1}, \ldots, \mathbf{r}_{Z_m}$, respectively. According to the construction of the schema, for each class Z_i $(0 \leq i < m)$, there is an IS-A connection from its naming concept $\mathbf{r}_{Z_i}$ to a concept $\mathbf{w}_{i+1}$ in Z_{i+1}. (Also: $\mathbf{r}_{Z_m}$ IS-A $\mathbf{w}_0$ in Z_0.) Whether $\mathbf{r}_{Z_i}$ is a property-introducing concept or an intersection concept,

$$\mathrm{P}(\mathbf{w}_i) = \mathrm{P}(\mathbf{r}_{Z_i}) \supset \mathrm{P}(\mathbf{w}_{i+1}) = \mathrm{P}(\mathbf{r}_{Z_{i+1}})$$

and, therefore, $\mathrm{P}(\mathbf{r}_{Z_i}) \supset \mathrm{P}(\mathbf{r}_{Z_{i+1}})$. From this, we see that

$$\mathrm{P}(\mathbf{r}_{Z_1}) \supset \mathrm{P}(\mathbf{r}_{Z_2}) \supset \cdots \supset \mathrm{P}(\mathbf{r}_{Z_m}) \supset \mathrm{P}(\mathbf{r}_{Z_0}) \supset \mathrm{P}(\mathbf{r}_{Z_1}).$$

In other words, $\mathrm{P}(\mathbf{r}_{Z_1}) \supset \mathrm{P}(\mathbf{r}_{Z_1})$—a contradiction. □

Overall, the OOVR schema provides a structural abstraction of the underlying network of the CV [19], [20]. Concepts with the same properties are grouped into areas which, in turn, are modeled as object classes; the concepts themselves become the objects of the OODB. This schema represents a substantial reduction in size from the original CV. In Fig. 5, we show the InterMED OOVR schema. The InterMED has 2,820 concepts in its network, but its schema contains only 39 area classes, nine of them being intersection classes (below the dashed line). This schema can be used to gain an understanding of the InterMED.

4 Navigation Examples

In this section, we demonstrate how the schema helps to speed up traversals of a CV. Suppose that a user wants to search for some information, say, in the InterMED, but does not know the name of the concept for which the information is desired. For example, suppose a user is looking for a drug to treat fever and coughing in children. While the user does not remember the names of such drugs, he may recognize them when encountered. This is a natural application of an IS-A hierarchy traversal, with the user employing his knowledge about the target concept to guide the choices at the different levels of the hierarchy.

Using the InterMED OOVR representation, we enable a faster traversal involving both the schema and the underlying knowledge content. The depth of the InterMED's IS-A hierarchy is 11, while the depth of its OOVR schema's subclass hierarchy is just 4. Instead of traversing the

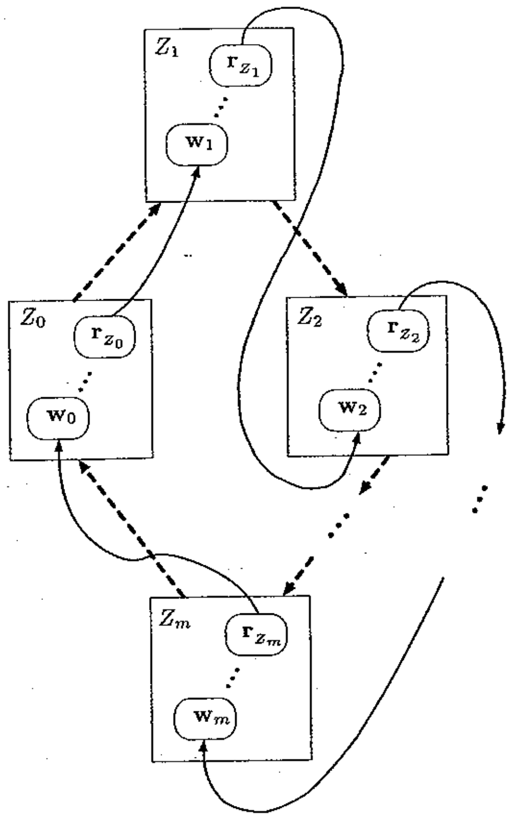

Fig. 4. Invalid subclass configuration: Subclass cycles do not exist in an OOVR schema. (SUBCLASS_OF drawn as a dashed arrow).

InterMED hierarchy through its many levels, we traverse the OOVR schema until the proper area class (say, *X_Area*) is identified. This is easier because the schema presents higher-level subject areas rather than detailed concepts. The user only needs to make a very general judgment about whether a desired concept fits into a given class or not. Once that judgment is made, the user will switch to that part of the concept network belonging to *X_Area*. The traversal will run through this subhierarchy until the desired concept is recognized (or its absence is noted).

This traversal is shorter since the number of traversing steps is bounded by the sum of the depth of the OOVR schema and the depth of the subhierarchy of *X_Area*. Furthermore, the traversal is faster because the number of subclasses of a given class in the OOVR is typically much smaller than the number of children of a concept in the InterMED. As a traversal requires scanning through a list of children and choosing one of them, it will be easier and faster at the schema level.

Let us demonstrate the above-mentioned traversal: looking for a drug to treat fever and coughing. First, let us perform the traversal at the concept level in the InterMED. The traversal starts at the root **Entity**, having 15 children. Since we are looking for a medication, **Pharmacy Items (Drugs and Nondrugs)** is chosen. The process continues in this manner all the way down to **Acetaminophen/Codeine Elixir Preparations**. The entire traversal path is illustrated in Fig. 6a. Alongside each concept, we list its number of children, indicating the range of choices encountered at that level. Overall, this traversal of a path of nine concepts required scanning a total of 83 children.

Let us now demonstrate the same traversal in the OOVR (Fig. 5). We start with *Entity_Area* and travel through *Pharmacy_Items_Drugs_And_Nondrugs_Area* to *Acetaminophen_Codeine_Tablet_Preparation_Area*, a leaf. At that point, the traversal switches to the concept level. Since *Acetaminophen_Codeine_Tablet_Preparation_Area* is an intersection class with only four roots, we can easily find the concept **Acetaminophen/Codeine Elixir Preparations.** This is illustrated in Fig. 6b, where the number beside a class is its respective number of subclasses. This traversal spans three classes, with a total of 26 subclasses, and four concepts. Thus, the total number of scanned items is 30, quite a bit fewer than the 83 required before.

To be formal in our comparison of the two traversal methods, we need to define the notion of *browsing path* on both the concept level and the area (class) level.

Definition 8 (Concept-level browsing path). *A concept-level browsing path is a sequence of concepts* $(\mathbf{c}_1, \mathbf{c}_2, \ldots, \mathbf{c}_n)$ *such that* $\mathbf{c}_{i+1}$ *IS-A* $\mathbf{c}_i$, $1 \leq i < n$.

Definition 9 (Schema-level browsing path). *A schema-level browsing path is a sequence of area classes* $(A_1, A_2, \ldots, A_k)$ *such that* A_{i+1} SUBCLASS_OF A_i, $1 \leq i < k$.

We can now properly state how the SUBCLASS_OF relationships at the OOVR schema level properly reflect the IS-A relationships in the CV. For every concept-level browsing path $(\mathbf{c}_1, \mathbf{c}_2, \ldots, \mathbf{c}_n)$, there exists a corresponding schema-level browsing path $(A_1, A_2, \ldots, A_k)$ which satisfies the following conditions:

1. Concept $\mathbf{c}_1$ is an instance of area class A_1.
2. Concept $\mathbf{c}_n$ is an instance of A_k.
3. There exists a partition of $(\mathbf{c}_1, \mathbf{c}_2, \ldots, \mathbf{c}_n)$ into disjoint subpaths of consecutive concepts, say, $\mathbf{c}_{i_1}, \ldots, \mathbf{c}_{i_e}$, which are paths in the induced subnetwork of an area A_j $(1 \leq j \leq k)$.

5 Inadequacy of the Multirooted OODB Modeling

5.1 Browsing Multirooted Intersection Areas

The traversal at the schema level is very effective when all areas are singly rooted. In such a case, the root concept subsumes all other concepts in the area and conveys the area's general semantics. However, only property-introducing areas are guaranteed to be singly rooted.

Traversals in the context of multirooted intersection classes may not proceed so smoothly. This is because the name of the class is chosen arbitrarily from among the roots. Instead of conveying the general semantics for the whole area, the chosen root may capture only the essence of the concepts which are its descendants. But, some concepts in the area—aside from the other roots—may not even be descendants of that root. In fact, the roots may be very dissimilar from an interpretive viewpoint; grouping them together was strictly the result of structural similarity. Therefore, it is sensible to reexamine whether those concepts should have been grouped together in the first place.

As an example, let us look at the multirooted intersection area shown in Fig. 7a, which was gleaned from the MED. Overall, Fig. 7a contains six concepts. **ICD9 Disease** belongs to *ICD9_Element_Area* and **Disease or Syndrome** belongs to

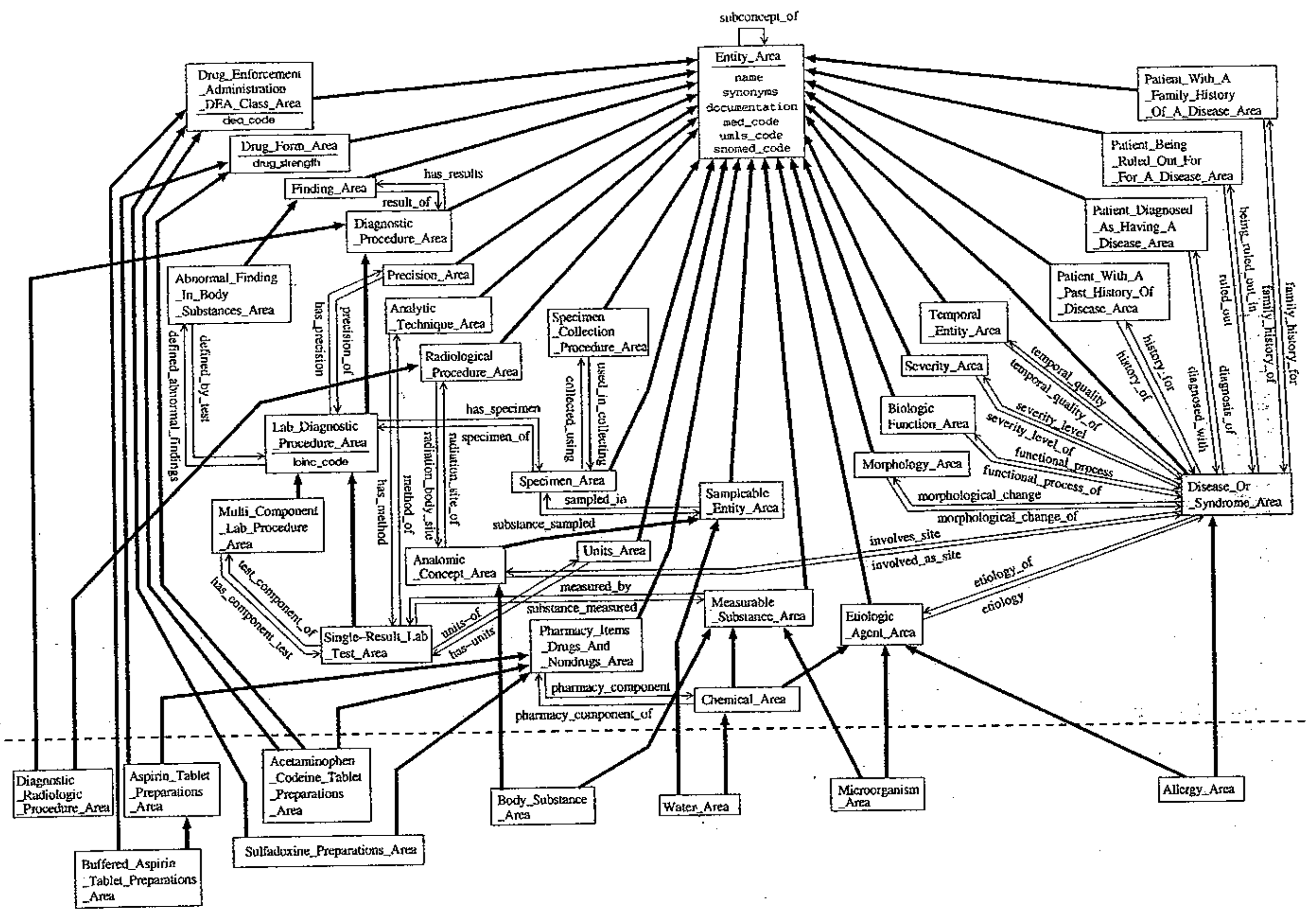

Fig. 5. Schema of the InterMED OOVR.

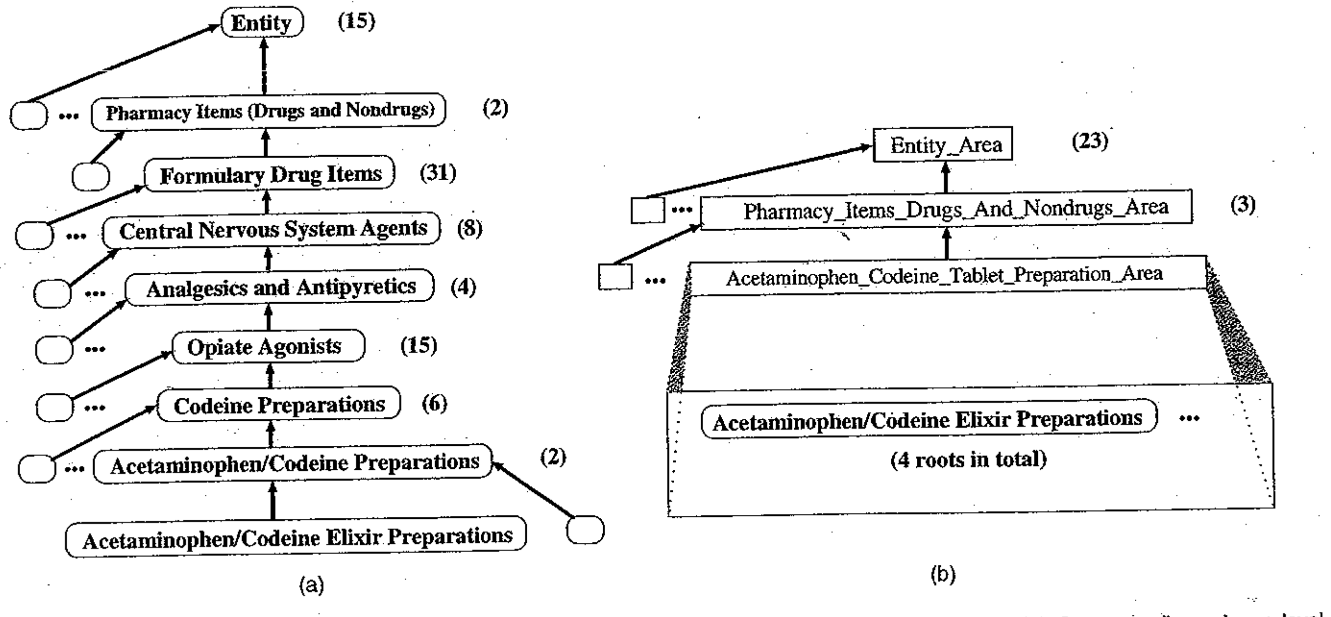
Entity
(15)
Pharmacy Items (Drugs and Nondrugs)
(2)
Formulary Drug Items
(31)
Central Nervous System Agents
(8)
Analgesics and Antipyretics
(4)
Opiate Agonists
(15)
Codeine Preparations
(6)
Acetaminophen/Codeine Preparations
(2)
Acetaminophen/Codeine Elixir Preparations
(a)
Entity_Area
(23)
Pharmacy_Items_Drugs_And_Nondrugs_Area
(3)
Acetaminophen_Codeine_Tablet_Preparation_Area
Acetaminophen/Codeine Elixir Preparations
(4 roots in total)
(b)

Fig. 6. (a) Concept-level browsing path between **Entity** and **Acetaminophen/Codeine Elixir Preparations**. (b) Corresponding schema-level browsing path.

Disease_or_Syndrome_Area (see Fig. 7b). The concepts **Mental or Behavioral Dysfunction** and **Neoplasm** are children of both **ICD9 Disease** and **Disease or Syndrome**. They have the same structure and, thus, are roots of the same intersection area, *Mental_or_Behavioral_Dysfunction_Area*. Actually, in the MED, there are 29 roots for this intersection area! These include **Infectious Disease, Disorder of Circulatory System, Disorders of Nervous System**, etc. Our mapping methodology randomly chooses one of them to be the area's name. In this example, **Mental or Behavioral Dysfunction** has been selected (Fig. 7b). However, any of the other 28 roots could have been selected.

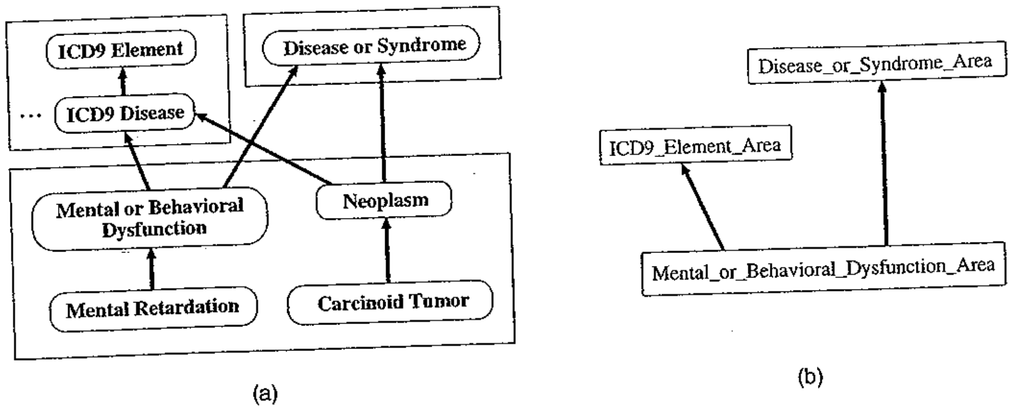

Fig. 7. (a) Three areas including an intersection area (in the bottom box) and (b) their OOVR schema.

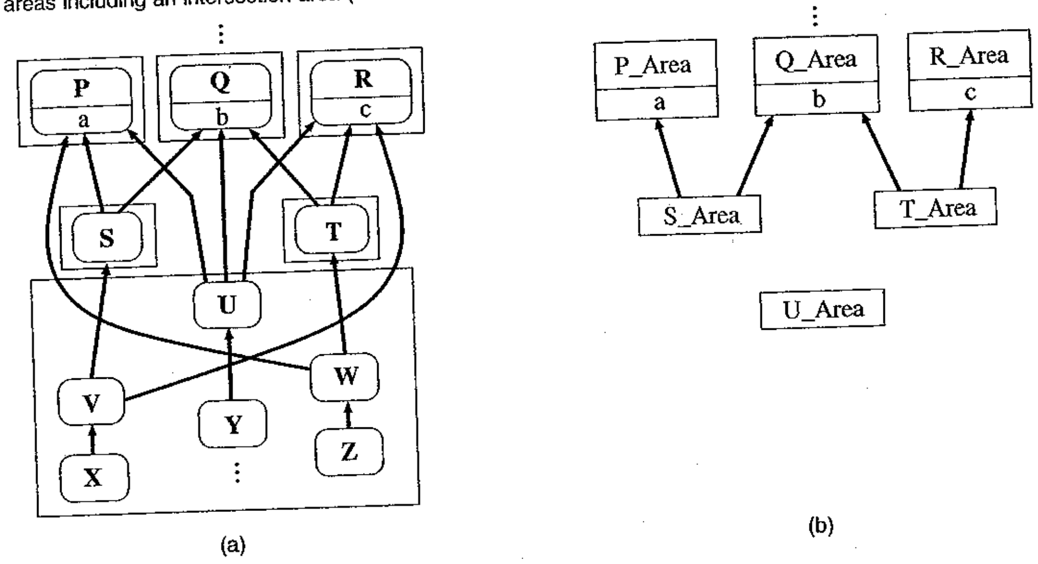

Fig. 8. (a) CV excerpt and (b) its OOVR schema with subclass relationships from *U_Area* omitted.

One will note that there is almost no similarity between **Mental or Behavioral Dysfunction** and **Neoplasm**, even though they are in the same area. With **Mental or Behavioral Dysfunction** as the area's name, it is hard to imagine that **Neoplasm** also belongs there. In this case, the schema diagram does not provide a useful abstraction for assisting users in browsing an area of the CV.

Let us express the problem in terms of a browsing path. Consider the concept-level browsing path (**ICD9 Element, ICD9 Disease, Neoplasm, Carcinoid Tumor**). The corresponding schema-level browsing path is (*ICD9_Element_Area, Mental_or_Behavioral_Dysfunction_Area*). However, since **Carcinoid Tumor** is a descendent of **Neoplasm** and is not a descendent of **Mental or Behavioral Dysfunction**, a user will not know to choose this schema-level browsing path while searching for **Carcinoid Tumor**.

5.2 Establishing Subclass Relationships for Intersection Area Classes

We have discovered an additional problem in the modeling of multirooted intersection areas. Our mapping methodology yields a configuration of subclass relationships that does not reflect the pattern of IS-A links that cross the boundaries of such areas (the "cross-area IS-A relationships"). There are several equivalent OODB modeling alternatives that properly capture the structure of the areas, but none of these is sufficient enough to convey the full extent of the cross-area IS-A relationships. This makes it more difficult to effectively utilize the OOVR schema.

Consider Fig. 8a which contains 11 concepts, with only the top three, **P**, **Q**, and **R**, introducing new properties *a*, *b*, and *c*, respectively. **S**, **T**, and **U** have several parents which reside in different property-introducing areas and, thus, should belong to intersection areas. In fact, they should be roots of three different intersection areas since their property sets are different. **V** and **W** differ from **S**, **T**, and **U** in that one of their parents resides in an intersection area, while the other resides in a property-introducing area. However, **V** and **W** have the same property set as **U**. Our mapping methodology groups **U**, **V**, **W**, **X**, **Y**, and **Z** into the same intersection area. Its name is arbitrarily chosen to be *U_Area* (Fig. 8b).

There is a problem with defining *U_Area*'s subclass relationships (omitted from Fig. 8b). Consider the concept **V**: It has **S** and **R** as parents. Thus, one may set *U_Area*'s subclass relationships to point to *S_Area* and *R_Area* (Fig. 9a). However, the absence of a relationship between *U_Area* and *T_Area* may mislead the user into thinking that there is no IS-A link between any concepts of these two areas. Actually, **W** IS-A **T**. The schema of Fig. 9a has no schema-level browsing path corresponding to the

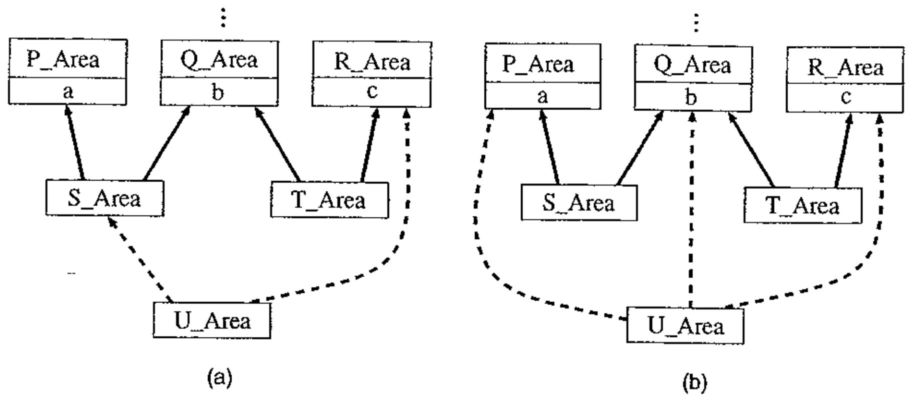

Fig. 9. Alternative schemas for the areas in Fig. 8a.

concept-level browsing path (**R**, **T**, **W**). A similar problem arises if **W**'s IS-A links are used to establish *U_Area*'s subclass relationships. Consider the concept **U**: It has parents **P**, **Q**, and **R**. Thus, one may set *U_Area* to be a subclass of *P_Area*, *Q_Area*, and *R_Area* (Fig. 9b). This schema might lead users to believe that there are no IS-A links between concepts in *U_Area* and those in *S_Area* or *T_Area*. Here, though, **V** IS-A **S** and **W** IS-A **T**. There is no schema-level browsing path corresponding to the concept-level browsing paths (**R**, **T**, **W**) and (**P**, **S**, **V**) in Fig. 9b.

Another alternative is to set *U_Area*'s subclass relationships to mirror all IS-A relationships of its roots. For each root *r*, we could define subclass relationships from *U_Area* to all area classes containing a parent of *r*. Using this approach, we obtain a set of parent classes for *U_Area* which is the union of the parent classes from the alternatives considered above. In Fig. 10, *U_Area* has five parents. The five subclass relationships from *U_Area* can, once again, be misleading. We do not know which relationship originated from which root. Furthermore, that same schema would be generated if there existed a single root in *U_Area* with five superconcepts in the areas *P_Area*, ..., *T_Area*. Thus, this is not desirable either. All these choices are structurally equivalent since the resulting property sets for *U_Area* are the same. However, what is lost is some of the OOVR schema's effectiveness in reflecting aspects of the IS-A hierarchy of the original CV.

6 Singly Rooted OOVR Methodology

The two problems that were presented in Section 5 arise from placing concepts of potentially widely varying semantics in the same intersection area and its corresponding area class. Recall that, in general, a class is a construct that gathers together objects with the same structure *and* semantics. In our mapping methodology, most classes satisfy this condition. Certainly, the structural aspect is satisfied by all area classes. Property-introducing classes are semantically cohesive due to their unique roots, which provide the areas' names. The same can be said for an intersection class having a single root. However, the synchronization of structure and semantics breaks down for multirooted intersection areas. All concepts of such an area have the same structure but not necessarily similar semantics because some concepts may be descendants of one root and not directly related at all to another root. It is unlikely that any single root provides appropriate "root semantics" for the entire area.

To preserve the ordinary interpretation of OODB classes as having objects with the same structure and semantics, and indeed to support effective CV access via the OOVR schema, we need to further partition a multirooted intersection area into separate singly rooted groupings, which we call *partial areas*. Once this is accomplished, the intersection area class can be replaced by a number of classes that have these partial areas as their respective extensions. This will ordinarily lead to the situation where several classes in the schema have the same structure, but that is not forbidden by the OODB paradigm. In the following section, we present a technique for carrying out this additional partitioning task.

6.1 Partial Areas of a Multirooted Intersection Area

It is natural to place roots of a multirooted intersection area into different partial areas since each root represents a distinct semantics. However, concepts may be descendants of more than one root. In such a case, they also represent distinct semantics. Thus, we create new partial areas for these kinds of concepts. Note that these newly created partial areas are considered distinct semantic groups as well. Therefore, concepts which are descendants of the roots of more than one distinct semantic group are also considered to exhibit new semantics in a recursive process.

In order to describe partial areas, we will need some new definitions. We will be using the term "path" to exclusively denote an *upward* path of IS-A links from some concept in the CV to one of its ancestors. The predicate "Desc" will be employed to indicate a descendant/ancestor relationship

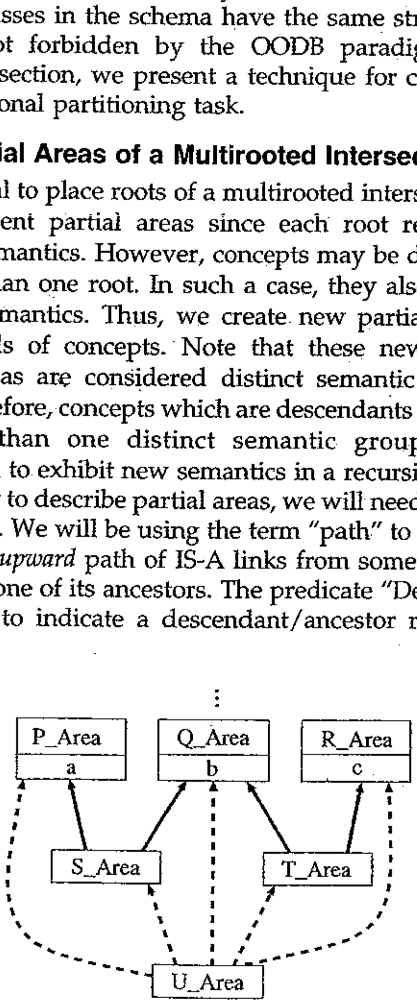

Fig. 10. Another alternative schema for Fig. 8a.

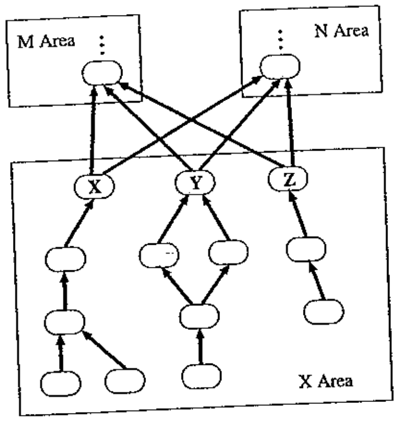

Fig. 11. Intersection area with three roots.

between a pair of concepts. That is, Desc(x, y) means that x is a descendant of y or, in other words, there exists a path from x to y. Two concepts x and y are called *independent* if x ≠ y, ¬Desc(x, y), and ¬Desc(y, x). In the following, the scope of the discussion is a multirooted intersection area *I*. The IS-A links pointing out of *I* are utilized only at the stage of setting the schema's SUBCLASS_OF relationships.

Definition 10 (Articulation concept). *An articulation concept (of intersection area I) is either:*

1. ***Base case:** An intersection concept or*
2. ***Recurrence:** A concept* w *for which there are two independent articulation concepts* x *and* y *such that: a)* Desc(w, x), Desc(w, y) *and b) no paths from* w *to* x *and no paths from* w *to* y *contain other articulation concepts.*

The role of an articulation concept in a multirooted intersection area is to root and name partial areas, just like the naming concepts were used for areas in the overall CV.

Definition 11 (Direct articulation descendant (DARD)). *Let* v *and* w *be articulation concepts (in I) such that* Desc(w, v). *The concept* w *is called a direct articulation descendant (DARD) of* v *if there exists a path from* w *to* v *that does not contain another articulation concept.*

With the definitions of articulation concept and DARD now in place, we can define the partial areas into which the multirooted intersection area is partitioned. Each intersection area will be divided into several partial areas (or *p-areas*, for short).

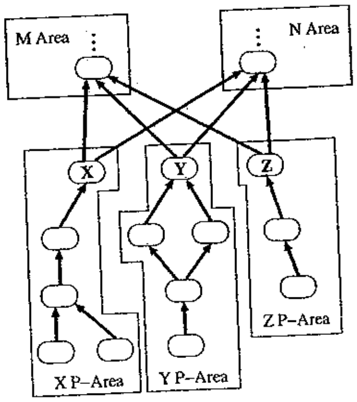

Fig. 12. Intersection area's p-areas.

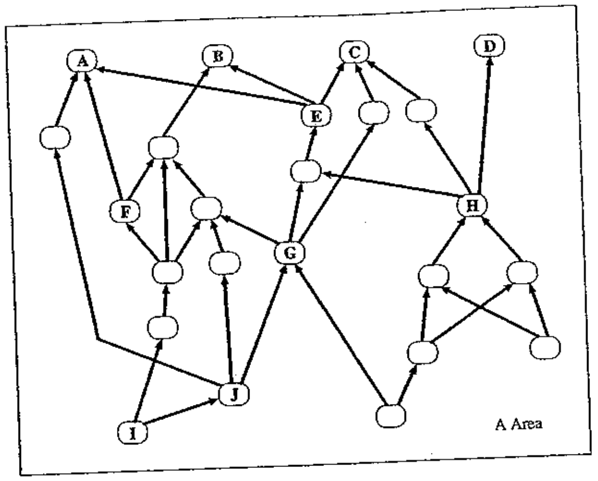

Fig. 13. Multirooted intersection area with descendant overlap.

Definition 12 (P-area). *A p-area (within an intersection area I) is a set of concepts containing an articulation concept* v *and all of* v*'s descendants (within I) excluding its DARDs and their respective descendants.*

Again, it is important to note that a p-area will contain a single articulation concept which will be the p-area's one and only root. Any descendants that are also articulation concepts will define new p-areas. As with the areas of the CV overall, the root concept is used as the name of the p-area.

Let us now demonstrate the above formalism in the partitioning of two example, multirooted intersection areas from a CV. The first one is X Area shown in Fig. 11, where M Area and N Area are also shown. Note that X Area has three roots. Its p-areas appear in Fig. 12.

If there is no overlap among the descendants of the roots (intersection concepts) of a multirooted intersection area, then the only articulation concepts are the roots themselves. This is, in fact, the case with the intersection area of Fig. 11. In such a situation, every concept within the area is neatly grouped together with the unique root that is its ancestor. As a result, these groups form the p-areas of the original multirooted intersection area. Every p-area is singly rooted.

The partitioning becomes more complex when the descendants of the intersection concepts overlap and create

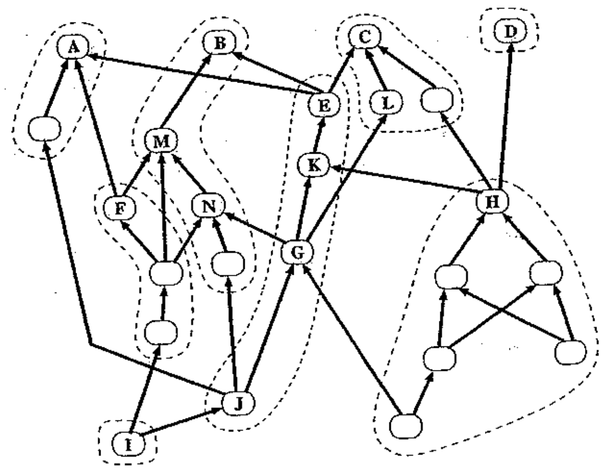

Fig. 14. P-areas for intersection area in Fig. 13 (with additional concepts labeled).

TABLE 2
Set_of_p-areas FUNCTION **P-AREA_Partition** (*area I*)

```
set_of_p-areas FUNCTION P-AREA_Partition(area $I$)
BEGIN
    // initialization
    $\mathcal{Q}$ := newqueue();                  // $\mathcal{Q}$ contains concepts ready to be processed.
    $\mathcal{A}_{\text{PAREA}}$ := newset();               // $\mathcal{A}_{\text{PAREA}}$ will hold all p-areas.
    FOR EACH concept v in $I$ DO      // Initialize p-counter and art[v].
        p-counter[v] := Num_parents_of(v, $I$);
        art[v] := ' ';
    FOR EACH root v of $I$ DO          // Put roots of $I$ into the queue since
        enqueue($\mathcal{Q}$, v);                 // they are ready to be processed.

    WHILE ( NOT emptyqueue($\mathcal{Q}$) ) DO
        v := dequeue($\mathcal{Q}$);
        $\mathcal{S}_{\text{ART}}$ := newset();             //$\mathcal{S}_{\text{ART}}$ will hold the arts of v's parents.
        FOR EACH parent c of v in $I$ DO
            insert($\mathcal{S}_{\text{ART}}$, art[c]);
        IF ( | $\mathcal{S}_{\text{ART}}$ | = 1 ) THEN
            // All parents of v are in one p-area.
            // Concept v will be in the same p-area as its parents.
            Let u be the (only) element in $\mathcal{S}_{\text{ART}}$;    // Concept v's parents are in the p-area
                                                 // rooted at u.
            insert($\mathcal{A}_{\mathbf{u}}$, v);              // Insert v into set $\mathcal{A}_{\mathbf{u}}$.
            art[v] := u;                    // Define art[v] to be the articulation concept u.
        ELSE IF ( | $\mathcal{S}_{\text{ART}}$ | > 1 AND
            $\exists$ u in $\mathcal{S}_{\text{ART}}$ which is a descendant of all other nodes in $\mathcal{S}_{\text{ART}}$ ) THEN
            // see Note 1 below
            insert($\mathcal{A}_{\mathbf{u}}$, v);
            art[v] := u;
        ELSE
            // Concept v is an articulation concept.
            // In case | $\mathcal{S}_{\text{ART}}$ | = 0, v is an intersection concept (root of $I$).
            $\mathcal{A}_{\mathbf{v}}$ := newset();            // Create a new set $\mathcal{A}_{\mathbf{v}}$ to hold concepts in this p-area.
            insert($\mathcal{A}_{\mathbf{v}}$, v);              // Insert articulation concept v into set $\mathcal{A}_{\mathbf{v}}$.
            insert($\mathcal{A}_{\text{PAREA}}$, $\mathcal{A}_{\mathbf{v}}$);         // Insert $\mathcal{A}_{\mathbf{v}}$ into $\mathcal{A}_{\text{PAREA}}$.
            art[v] := v;                    // Define art[v] to be v itself since v is an articulation concept.
        // After we process concept v, we need to decrease the p-counters of v's children by one.
        // After the decrease, if any p-counter is equal to zero, we put the associated concept
        // into the queue since it is ready to be processed.
        FOR EACH child k of v in $I$ DO
            p-counter[k]−−;
            IF ( p-counter[k] = 0 ) THEN
                enqueue($\mathcal{Q}$, k);
    RETURN $\mathcal{A}_{\text{PAREA}}$;
END □
```

additional articulation concepts. That case is demonstrated by the intersection area, A Area, shown in Fig. 13. This area has four roots: **A**, **B**, **C**, and **D**. By Definition 10, these are articulation concepts. The concepts **E**, **F**, **H**, and **I** are also articulation concepts. The eight p-areas for this intersection area are demarcated in Fig. 14 by dashed bubbles.

It is interesting to note that the concept **G** is not an articulation concept even though it has paths without articulation concepts to the independent articulation concepts **B** and **C**. However, the articulation concept **E** lies on a path from **G** to **B** (and also on a path from **G** to **C**). Thus, **G** is not an articulation concept and, in fact, belongs to the p-area rooted at **E**. This follows from item (2b) of Definition 10. For the same reason, the concept **J** is not an articulation concept. It, too, belongs to **E**'s p-area.

In the following, we prove that the p-areas partition the multirooted intersection area.

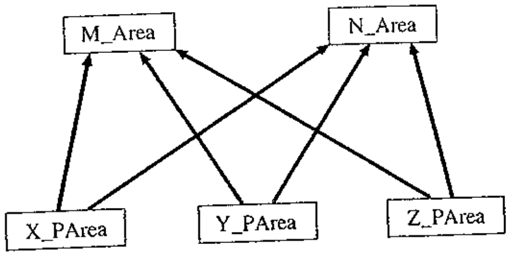

Fig. 15. Schema for intersection area in Fig. 11.

Lemma 6. *Let a nonarticulation concept* v *have multiple articulation ancestor concepts* $z_1, z_2, \ldots, z_n$ $(n > 1)$. *For any* i *and* j *such that* $i \neq j$, *if* z_j *is a descendant of* z_i, *then* v *does not belong to the p-area rooted at* z_i.

Proof If concept z_j is a descendant of z_i, then there must exist a DARD z_k (possibly z_j itself) of z_i on some path between z_i and z_j. Concept z_k is an ancestor of v since z_j is an ancestor of v. By Definition 12, v will be excluded from the p-area rooted at z_i as a descendant of the DARD z_k. □

Lemma 7. *Every concept belongs to at least one p-area.*

Proof. Case 1: If v is an articulation concept, then it is the root of its own p-area. Case 2: Assume to the contrary that a nonarticulation concept v does not belong to a p-area. Since v is in an intersection area, it must have one or more articulation ancestors $r_1, r_2, \ldots, r_n$ $(n \geq 1)$ which are roots of p-areas $P_1, P_2, \ldots, P_n$, respectively. By assumption, v does not belong to any of the P_i's. By Definition 12, v is excluded from P_i $(1 \leq i \leq n)$ because there exists a DARD of r_i (call it $r_j, i \neq j$) such that Desc(v, r_j). This implies Desc(r_j, r_i). Similarly, we see that there exists an articulation concept r_k to exclude v from the p-area P_j. Repeating this n times, we can form a sequence with $n+1$ articulation concepts starting at r_i. Each concept in this sequence is a descendant of its predecessor. However, by assumption, v has only n articulation ancestors. Therefore, some concept must appear more than once in the sequence. This implies there is a cycle in the IS-A hierarchy of the CV—a contradiction. □

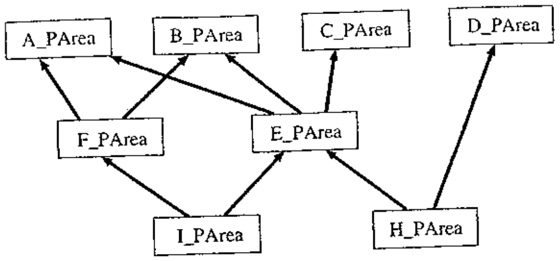

Fig. 16. Schema for intersection area in Fig. 13.

Lemma 8. *P-areas are disjoint.*

Proof. Case 1: If v is an articulation concept, then, by Definition 12, it is the root of a p-area of its own. Case 2: Assume to the contrary that a nonarticulation concept v belongs to two p-areas X and Y, rooted at r_X and r_Y, respectively. Since v belongs to the p-area X, then, by Definition 12, there is no path from v to r_X which contains other articulation concepts. Similarly, since v belongs to the p-area Y, there is no path from v to r_Y which contains other articulation concepts. Note that the roots r_X and r_Y must be independent concepts. Otherwise, if r_X is a descendant of r_Y, then, by Lemma 6, v does not belong to p-area Y rooted at r_Y. Similarly, r_Y cannot be a descendant of r_X. But, by Definition 10, if concept v has two independent articulation concepts ancestors r_X and r_Y such that no path from v to r_X or r_Y contains another articulation concept, then v itself is an articulation concept—a contradiction. □

Lemmas 7 and 8 together give us:

Theorem 4. *The p-areas of a multirooted intersection area partition the area.*

In the following, we present the algorithm that groups concepts in a multirooted intersection area I into p-areas. A_v is a set of concepts within one p-area rooted at v.

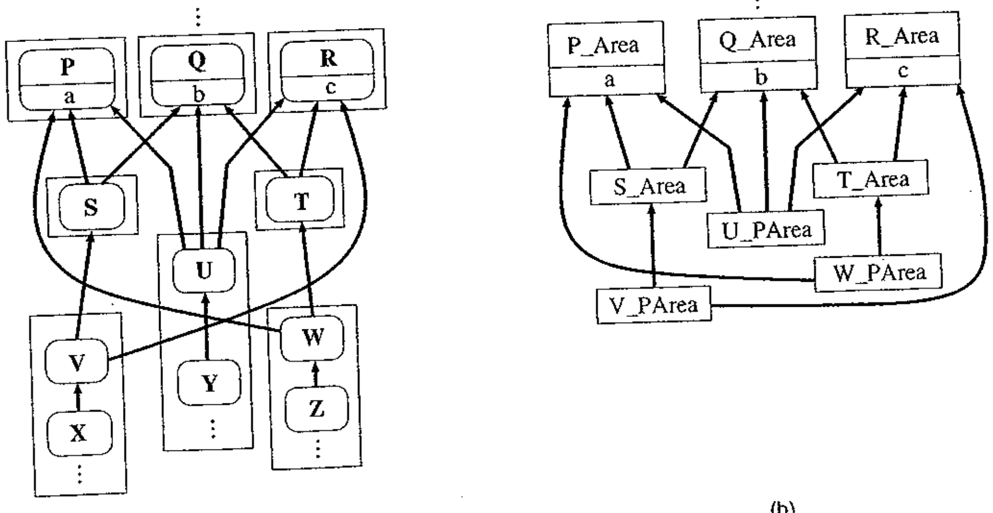

Fig. 17. (a) P-areas for Fig. 8a and (b) classes for those p-areas.

TABLE 3
Multirooted Intersection Classes in the Original MED OOVR Schema

Intersection class	Number of roots	Number of concepts
Intersection class 1	2	106
Intersection class 2	29	19,364
Intersection class 3	965	6967
Intersection class 4	31	324
Intersection class 5	11	182
Intersection class 6	9	170
Intersection class 7	15	40
Intersection class 8	42	43
Intersection class 9	2	175
Intersection class 10	3	3
Intersection class 11	2	52
Intersection class 12	3	96
Intersection class 13	124	124
Intersection class 14	16	16
Total: 14	Total: 1254	Total: 27,662

$\mathcal{A}_{\text{PAREA}}$ is a set of $\mathcal{A}_v$'s. At the end of the partitioning process, $\mathcal{A}_{\text{PAREA}}$ will be returned. Every concept v has an associated counter for "unprocessed parents" called "p-counter[v]." This algorithm uses one auxiliary function, "Num_parents_of," which takes a concept v and a multirooted intersection area I as input and returns the number of v's parents in I. Every concept v in p-area X rooted at x has an associated variable denoted as art[v] with value x. Indentation indicates a block. The algorithm does top-down processing to find p-areas in I. A concept can be processed only if all its parents have been processed. Therefore, initially, all roots of a multirooted intersection area are ready to be processed (see Table 2).

Note 1. By Theorem 4, v must belong to one p-area. In this case, v is not an articulation concept. Concept v will belong to a p-area rooted at u which is a descendant of all other elements in $\mathcal{S}_{\text{ART}}$. We can always find such an element u in $\mathcal{S}_{\text{ART}}$ because of the following three conditions: $\mathcal{S}_{\text{ART}}$ has more than one element, there are no two independent concepts, and a CV is acyclic.

Let us illustrate the partitioning process of an intersection area (Fig. 14). Assume the roots of this area were inserted into the queue in the order: **A**, **B**, **C**, and **D**. When we process **M** (dequeue it from $\mathcal{Q}$), we will generate a set $\mathcal{S}_{\text{ART}}$ with one element **art[B]** (equal to **B**). Since there is only one element in $\mathcal{S}_{\text{ART}}$, **M** will be inserted into $\mathcal{A}_{\text{B}}$ and **art[M]** will be assigned **B**. Next, we decrement the p-counters of M's children. **F** and **N** are inserted into the queue since their p-counters are zero. When we process **F**, the $\mathcal{S}_{\text{ART}}$ of **F** will have two elements, **A** and **B**, which are **art[A]** and **art[M]**, respectively. Since **A** and **B** are independent, **F** is an articulation concept and will be inserted into a new set $\mathcal{A}_{\text{F}}$; **art[F]** will be assigned **F**. For **G**, the algorithm will generate $\mathcal{S}_{\text{ART}}$ with three elements: **art[N]** = **B**, **art[K]** = **E**, and **art[L]** = **C**. Since **E** is a descendant of **B** and **C**, **G** is not an articulation concept. Instead, **G** will be inserted into $\mathcal{A}_{\text{E}}$, and **art[G]** will be assigned **E**.

6.2 Singly Rooted Schema

After a multirooted intersection area is partitioned into its respective p-areas, a separate class (called a *p-area class*) is defined for each of these p-areas. The p-area classes are

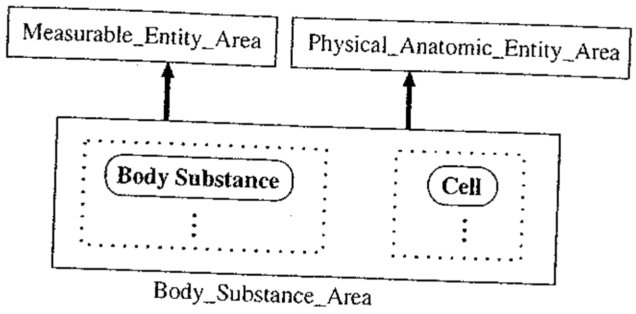

Fig. 18. Multirooted intersection class with disjoint descendants of roots in original MED OOVR schema.

intended as concept representations that promote better dissemination of the semantics of the underlying CV. To achieve this purpose, each class is singly rooted. The root captures the semantics of the class because all other concepts in the class are its descendents and are thus its conceptual specializations. Therefore, the root serves as a suitable name.

The subclass relationships of a p-area class are defined with respect to the parentage of the (unique) root as for an area class. The schemas for the intersection areas of Fig. 11 and Fig. 13 are shown, respectively, in Fig. 15 and Fig. 16. In Fig. 15, we see three p-area classes (along with two area classes) and six subclass relationships. Fig. 16 has eight p-area classes and nine subclass relationships. Shortcuts are omitted, as before. Also, in the context of this enhanced schema, we extend the notion of schema-level browsing path (Definition 9) to include p-area classes as well as area classes.

The enhanced partitioning of the multirooted intersection area classes solves the problem of establishing informative subclass relationships in the OOVR schema. That is, these subclass relationships more properly reflect the IS-A relationships which cross p-areas of the underlying CV. In Fig. 17a, we show the p-areas for the CV from Fig. 8a. Its schema appears in Fig. 17b. For any concept-level browsing path in Fig. 17a, there exists a corresponding schema-level browsing path in Fig. 17b. For instance, for the concept-level path (**P**, **S**, **V**, **X**), the corresponding schema-level path is (*P_Area*, *S_Area*, *V_PArea*).

The resulting schema is called the singly rooted schema. By having singly rooted p-areas, we achieve a schema where each class is named after its unique root and has an extension of semantically uniform concepts. The name properly captures the contents of the class and this, in turn, promotes a more accurate abstraction. Additionally, schema browsing is facilitated in all cases, solving the problems of Section 5. The growth of the size of the schema due to the inclusion of the p-area classes is not a concern as they offer a more refined abstraction of the CV.

Theorem 5. *There are no cycles in the singly rooted schema.*

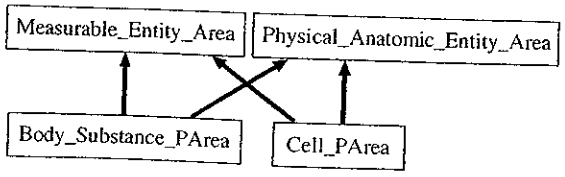

Fig. 19. Refined schema from Fig. 18.

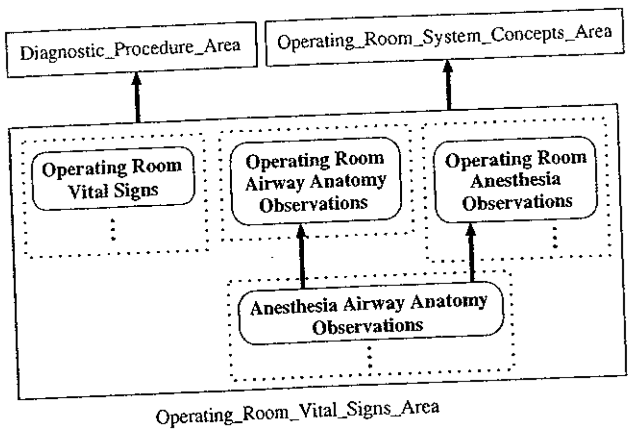
Diagnostic_Procedure_Area
Operating_Room_System_Concepts_Area
Operating Room Vital Signs
Operating Room Airway Anatomy Observations
Operating Room Anesthesia Observations
Anesthesia Airway Anatomy Observations
Operating_Room_Vital_Signs_Area

Fig. 20. Multirooted intersection class with nondisjoint descendants in original MED OOVR schema.

Proof. Suppose, to the contrary, that the schema contains a cycle of SUBCLASS_OF relationships. Classes represent p-areas or areas. Both kinds are singly rooted. Assume the cycle has $m+1$ classes $Z_0, Z_1, \ldots, Z_m$ such that Z_i is a SUBCLASS_OF Z_{i+1} $(i < m)$ and Z_m is a SUBCLASS_OF Z_0 (see Fig. 4). Let the roots of

$$Z_0, Z_1, \ldots, Z_m$$

be

$$\mathbf{r}_{Z_0}, \mathbf{r}_{Z_1}, \ldots, \mathbf{r}_{Z_m},$$

respectively. For each class Z_i $(0 \leq i < m)$, there is an IS-A connection from the root $\mathbf{r}_{Z_i}$ of Z_i to a concept w_{i+1} in Z_{i+1}. Also, $\mathbf{r}_{Z_m}$ IS-A w_0 in Z_0. Since $\mathrm{Desc}(\mathrm{w}_{i+1}, \mathbf{r}_{Z_{i+1}})$, then $\mathrm{Desc}(\mathbf{r}_{Z_i}, \mathbf{r}_{Z_{i+1}})$. Similarly, we see that

$$\mathrm{Desc}(\mathbf{r}_{Z_i}, \mathbf{r}_{Z_{i+1}}), \mathrm{Desc}(\mathbf{r}_{Z_{i+1}}, \mathbf{r}_{Z_{i+2}}), \ldots, \mathrm{Desc}(\mathbf{r}_{Z_{i-1}}, \mathbf{r}_{Z_i}).$$

But, this implies that a cycle of concepts exists in the CV—a contradiction. □

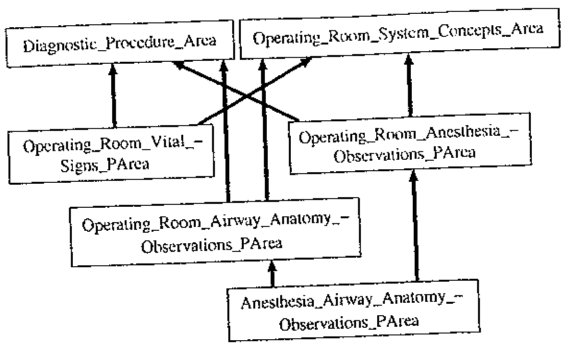
Diagnostic_Procedure_Area
Operating_Room_System_Concepts_Area
Operating_Room_Vital_-Signs_PArea
Operating_Room_Anesthesia_-Observations_PArea
Operating_Room_Airway_Anatomy_-Observations_PArea
Anesthesia_Airway_Anatomy_-Observations_PArea

Fig. 21. Singly rooted schema from Fig. 20.

7 Applying the Singly Rooted Methodology to a Large CV

In this section, we will apply our modeling approach to an existing OOVR with multirooted intersection classes [20]. This OOVR was originally obtained from the MED [6], which contains about 48,000 concepts and 61,000 IS-A links (1996 version). The MED OOVR's original schema consisted of 90 area classes: 53 property-introducing classes and 37 intersection classes. Of the 37 intersection classes, 14 are multirooted. Each of these is listed in Table 3 along with its number of roots and number of constituent concepts. The average size of these intersection classes is 1,975 concepts. The number of concepts belonging to such an intersection class can be quite large: Class 2 contains 29 roots and 19,364 concepts (Table 3).

In Table 3, there are nine classes in which the descendants of the various roots form disjoint sets. For example, Class 1, *Body_Substance_Area*, has two roots, **Body Substance** and **Cell**, and 106 total concepts (Fig. 18). Due to the disjointness, the only articulation concepts in the area are the roots. Thus, applying our methodology, we get two p-area classes *Body_Substance_PArea* and *Cell_PArea*, which together replace *Body_Substance_Area*. Both classes have the superclasses *Measurable_Entity_Area* and *Physical_Anatomic_Entity_Area* (Fig. 19). The other eight classes are 6, 7, 8, 9, 10, 11, 13, and 14 from Table 3.

TABLE 4
P-Area Classes in the Singly Rooted MED OOVR Schema

Intersection class in the original schema	Number of roots	Number of corresponding p-area classes in the singly-rooted schema	Average number of concepts per p-area class
Intersection class 1	2	2	53
Intersection class 2	29	168	115
Intersection class 3	965	1038	7
Intersection class 4	31	48	7
Intersection class 5	11	16	11
Intersection class 6	9	9	19
Intersection class 7	15	15	3
Intersection class 8	42	42	1
Intersection class 9	2	2	87
Intersection class 10	3	3	1
Intersection class 11	2	2	26
Intersection class 12	3	4	24
Intersection class 13	124	124	1
Intersection class 14	16	16	1
Total: 14	Total: 1254	Total: 1489	Average: 19

The remaining classes in Table 3 have roots whose descendants overlap. One of them is Class 12, *Operating_Room_Vital_Signs_Area*, which has three roots, **Operating Room Vital Signs**, **Operating Room Anesthesia Observations**, and **Operating Room Airway Anatomy Observations**, and a total of 96 concepts (Fig. 20). This class has four articulation concepts, three of them being the roots. The other is **Anesthesia Airway Anatomy Observations**. The four p-area classes which supplant the original intersection class are shown in Fig. 21.

Table 4 summarizes the results of applying the revised OOVR approach to the MED. Previously, we had 14 multirooted intersection classes with 1,254 roots in total. The new schema contains 1,489 p-area classes. Their average size is 19 concepts. This should be compared to the average size of 1,975 concepts of the intersection classes that were replaced. This more detailed abstraction level provides a set of smaller and more manageable semantic units and facilitates better navigation of the CV.

8 Conclusions

We have presented a technique which allows a semantic network-based controlled vocabulary (CV) to be converted into an equivalent OODB representation called an OOVR. We described the theoretical aspects of our approach and gave an algorithmic specification for its implementation. At its foundation are the notions of area, articulation concept, and p-area, which are used to partition a CV and induce an OODB schema comprising structurally and semantically uniform units. The OODB schema consists of two kinds of classes, area classes and p-area classes. In our development of the methodology, we have solved the difficult problem of how to formally assign concepts of a multirooted area class to a set of singly rooted p-area classes. Each class in the OOVR schema contains concepts that have the exact same set of properties, making them all structurally uniform. Additionally, every class is singly rooted and, therefore, exhibits semantic uniformity.

A major advantage of the OOVR representation is the abstract layer provided by its schema. The singly rooted schema obtained guarantees that each class comprises a logical unit of concepts. The unique root is used as the name of a class to capture the overarching nature of the class's concepts. Utilizing this abstraction, a user can more readily browse the CV network and comprehend its content. We have presented the results of applying our OOVR approach to an existing CV called the MED. In the future, our methodology may help refine the organization of the metathesaurus [30] of the UMLS by partitioning the sets of concepts associated with semantic types into smaller logical units [15].

Acknowledgments

This research was (partially) done under a cooperative agreement between the US National Institute of Standards and Technology Advanced Technology Program (under the HIIT contract #70NANB5H1011) and the Healthcare Open Systems and Trials, Inc., consortium. This research was also partially supported by two grants from the New Jersey Commission for Science and Technology, one from the New Jersey Center for Software Engineering and the other from the Multi-lifecycle Engineering Research Center.

References

[1] Am. Medical Assoc., *Physicians' Current Procedural Terminology: CPT*, fourth ed. 1998.

[2] *Building an Object-Oriented Database System: The Story of O_2*, F. Bancilhon, C. Delobel, P. Kanellakis, eds., San Mateo, Cailf.: Morgan Kaufmann, 1992.

[3] E. Bertino and L. Martino, *Object-Oriented Database Systems, Concepts and Architectures*. Redwood City, Cailf.: Addison-Wesley, 1993.

[4] R.J. Brachman, "On the Epistemological Status of Semantic Networks," *Associative Networks: Representation and Use of Knowledge by Computers*, N.V. Findler, ed., pp. 3-50, 1979.

[5] *The Object Database Standard: ODMG 2.0*, R.G.G. Cattell and D.K. Barry, eds., San Francisco: Morgan Kaufmann, 1997.

[6] J.J. Cimino, P. Clayton, G. Hripcsak, and S. Johnson, "Knowledge-Based Approaches to the Maintenance of a Large Controlled Medical Terminology," *J. Am. Medical Informatics Assoc.*, vol. 1, no. 1, pp. 35-50, 1994.

[7] J.J. Cimino, G. Hripcsak, S.B. Johnson, and P.D. Clayton, "Designing an Introspective, Multipurpose, Controlled Medical Vocabulary," *Proc. 13th Ann. Symp. Computer Applications in Medical Care*, pp. 513-517, Nov. 1989.

[8] *SNOMED International: The Systematized Nomenclature of Human and Veterinary Medicine*, R.A. Côté D.J. Rothwell, J.L. Palotay, R.S. Beckett, and L. Brochu, eds. Northfield, Ill.: College of Am. Pathologists, 1993.

[9] S. Even, *Graph Algorithms*. Potomac, Md.: Computer Science Press, 1979.

[10] D.H. Fischer, "Consistency Rules and Triggers for Multilingual Terminology," *Proc. Third Int'l Congress Terminology and Knowledge Engineering (TKE '93)*, pp. 333-342, 1993.

[11] J. Geller, M. Halper, and Y. Perl, *Hybrid Diagram/Form Interface: A Two-Layered Web-Based Interface to an OODB Vocabulary*, in preparation.

[12] P.M.D. Gray, K.G. Kulkarni, and N.W. Paton, *Object-Oriented Databases: A Semantic Data Model Approach*. New York: Prentice Hall, 1992.

[13] H. Gu, J.J. Cimino, M. Halper, J. Geller, and Y. Perl, "Utilizing OODB Schema Modeling for Vocabulary Management," *Proc. 1996 AMIA Ann. Fall Symp.*, J.J. Cimino, ed., pp. 274-278, Oct. 1996.

[14] H. Gu, M. Halper, J. Geller, and Y. Perl, "Benefits of an OODB Representation for Controlled Medical Terminologies," *J. Am. Medical Informatics Assoc.*, vol. 6, pp. 283-303, 1999.

[15] H. Gu, Y. Perl, J. Geller, M. Halper, L. Liu, and J.J. Cimino, "Representing the UMLS as an OODB: Modeling Issues and Advantages," *J. Am. Medical Informatics Assoc.*, vol. 7, no. 1, pp. 66-80, Jan/Feb 2000.

[16] B.L. Humphreys, D.A.B. Lindberg, H.M. Schoolman, and G.O. Barnett, "The Unified Medical Language System: An Informatics Research Collaboration," *J. Am. Medical Informatics Assoc.*, vol. 5, no. 1, pp. 1-11, 1998.

[17] *Object-Oriented Concepts, Databases, and Applications*, W. Kim and F.H. Lochovsky, eds. New York: ACM Press, 1989.

[18] D.A.B. Lindberg, B.L. Humphreys, and A.T. McCray, "The Unified Medical Language System," *Methods of Information in Medicine*, vol. 32, pp. 281-291, 1993.

[19] L. Liu, M. Halper, J. Geller, and Y. Perl, "Controlled Vocabularies in OODBs: Modeling Issues and Implementation," *Distributed and Parallel Databases*, vol. 7, no. 1, pp. 37-65, Jan. 1999.

[20] L. Liu, M. Halper, H. Gu, J. Geller, and Y. Perl, "Modeling a Vocabulary in an Object-Oriented Database," *Proc. Fifth Int'l Conf. Information and Knowledge Management (CIKM '96)*, pp. 179-188, Nov. 1996.

[21] M.E.S. Loomis, *Object Databases: The Essentials*. Reading, Mass.: Addison-Wesley, 1995.

[22] E. Mays, C. Apte, J. Griesmer, and J. Kastner, "Experience with K-Rep: An Object-Centered Knowledge Representation Language," *Proc. IEEE AI Application Conf.*, Mar. 1988.

[23] *Medical Subject Headings*. Bethesda, Md.: Nat'l Library of Medicine, updated annually.

[24] N.F. Noy and C.D. Hafner, "The State of the Art in Ontology Design: A Survey and Comparative Review," *AI Magazine*, vol. 18, no. 3, pp. 53-74, Fall 1997.

[25] D.E. Oliver, E.H. Shortliffe, and InterMed Collaboratory, "Collaborative Model Development for Vocabulary and Guidelines," *Proc. 1996 AMIA Ann. Fall Symp.*, J.J. Cimino, ed., p. 826, Oct. 1996.

[26] *ONTOS DB/Explorer 4.0. Reference Manual.* Lowell, Mass.: ONTOS, Inc., 1996.
[27] The OOVR Browser, URL: http://object.njit.edu:8080/~new-oohvr/JBI/INTERMED/index.html, 2001.
[28] A.L. Rector, S. Bechhofer, C.A. Goble, I. Horrocks, W.A. Nowlan, and W.D. Solomon, "The GRAIL Concept Modelling Language for Medical Terminology," *Artifical Intelligence in Medicine,* vol. 9, pp. 139-171, 1997.
[29] A.L. Rector, W.A. Nowlan, and A.J. Glowinski, "Goals for Concept Representation in the GALEN Project," *Proc. 17th Ann. Symp. Computer Applications in Medical Care (SCAMC '93),* pp. 414-418, 1993.
[30] P.L. Schuyler, W.T. Hole, M.S. Tuttle, and D.D. Sherertz, "The UMLS Metathesaurus: Representing Different Views of Biomedical Concepts," *Bull. Medical Library Assoc.,* vol. 81, no. 2, pp. 217-222, 1993.
[31] V. Soloviev, "An Overview of Three Commercial Object-Oriented Database Management Systems: ONTOS, ObjectStore, and O_2," *SIGMOD Record,* vol. 21, no. 1, pp. 93-104, Mar. 1992.
[32] M.S. Tuttle and S.J. Nelson, "The Role of the UMLS in 'Storing' and 'Sharing' Across Systems," *Int'l J. Bio-Medical Computing,* vol. 34, pp. 207-237, 1994.
[33] *International Classification of Diseases: Ninth Revision, with Clinical Modifications.* Washington, D.C.: US Nat'l Center for Health Statistics, 1980.
[34] S.B. Zdonik and D. Maier, "Fundamentals of Object-Oriented Databases," *Readings in Object-Oriented Database Systems,* S.B. Zdonik and D. Maier, eds., San Mateo, Cailf.: Morgan Kaufmann, pp. 1-32, 1990.

Li-min Liu received the BS degree in computer and information science from Tamkang University, Taiwan, in 1990, the MS degree in computer information science from Syracuse University in 1994, and the PhD degree in computer and information science from the New Jersey Institute of Technology (NJIT) in 1999. He was a senior member of the technical staff at Pumpkin Networks, Inc., in California. Dr. Liu is an assistant professor of computer science at Kean University and a visiting researcher at the OODB & AI Laboratory of NJIT. His research interests include object-oriented (OO) database systems, OO modeling, knowledge representation, medical informatics, controlled vocabularies, and parallel databases.

Michael Halper received the BS degree (with honors) in computer science from the New Jersey Institute of Technology (NJIT) in 1985, the MS degree in computer science from Fairleigh Dickinson University in 1987, and the PhD degree in computer science from NJIT in 1993. During his graduate studies, he was the recipient of a Garden State Graduate Fellowship from the State of New Jersey. Dr. Halper is an associate professor of computer science at Kean University and a visiting researcher at the OODB & AI Laboratory of NJIT. His research interests include conceptual and object-oriented data modeling, part-whole modeling, extensible data models, object-oriented database systems, and medical informatics. He has worked on the OOHVR project—funded by the National Institute of Standards and Technology (NIST) Advanced Technology Program—to model controlled vocabularies and terminologies using object-oriented database technology. Dr. Halper has published numerous papers in international journals, conferences, and workshops. He is a member of The Honor Society of Phi Kappa Phi.

James Geller received an electrical engineering Diploma from the Technical University, Vienna, Austria, in 1979 and received the MS degree (1984) and the PhD degree (1988) in computer science from the State University of New York at Buffalo. He spent the year before his doctoral defense at the Information Sciences Institute (ISI) of University of Southern California in Los Angeles, working with their Intelligent Interfaces Group. He is currently a professor in the Computer and Information Science Department of the New Jersey Institute of Technology, where he is also director of the OODB & AI Laboratory and vice chair of the MS and PhD programs in biomedical informatics. Dr. Geller has published numerous journal and conference papers in a number of areas, including knowledge representation, parallel artificial intelligence, medical informatics, and object-oriented databases. His current research interests concentrate on object-oriented modeling of medical vocabularies and on Web mining. He is a past SIGART Treasurer.

Yehoshua Perl received the PhD degree in computer science in 1975 from the Weizmann Institute of Science, Israel. He was appointed a lecturer and senior lecturer at Bar-Ilan University, Israel, in 1975 and 1979, respectively. He spent a sabbatical at the University of Illinois from 1977 to 1978. From 1982 to 1985, he was a visiting associate professor at Rutgers University (New Brunswick). Since 1985, he has been in the Computer and Information Science Department at the New Jersey Institute of Technology (NJIT), where he was appointed a professor in 1987. Dr. Perl spent short research visits at the University of Capetown, University of Paris VI, University of Toronto, University of British Columbia, University of Rome, and GMD-IPSI. He received the Harlan Perlis Research Award of NJIT in 1996. Dr. Perl is the author of more than 80 papers in international journals and conferences. His publications are in the following areas: object-oriented databases, design and analysis of algorithms, data structures, design of networks, sorting networks, graph theory, and data compression. From 1995 to 1999, he was mainly involved in the OOHVR (Object Oriented Healthcare Vocabulary Repository) project supported by the US National Institute of Standards and Technology, Advanced Technology Program. Highlights of his research include, among others: the shifting algorithm technique for tree partitioning, analysis of interpolation search, the design of periodic sorting networks, modeling vocabularies using object-oriented databases, and enhancing the semantics of object-oriented databases.