Copyright Warning & Restrictions

The copyright law of the United States (Title 17, United States Code) governs the making of photocopies or other reproductions of copyrighted material.

Under certain conditions specified in the law, libraries and archives are authorized to furnish a photocopy or other reproduction. One of these specified conditions is that the photocopy or reproduction is not to be "used for any purpose other than private study, scholarship, or research." If a, user makes a request for, or later uses, a photocopy or reproduction for purposes in excess of "fair use" that user may be liable for copyright infringement,

This institution reserves the right to refuse to accept a copying order if, in its judgment, fulfillment of the order would involve violation of copyright law.

Please Note: The author retains the copyright while the New Jersey Institute of Technology reserves the right to distribute this thesis or dissertation

Printing note: If you do not wish to print this page, then select "Pages from: first page # to: last page #" on the print dialog screen

ABSTRACT

DEVELOPING TECHNIQUES FOR ENHANCING COMPREHENSIBILITY OF CONTROLLED MEDICAL TERMINOLOGIES

by
Huanying Gu

A controlled medical terminology (CMT) is a collection of concepts (or terms) that are used in the medical domain. Typically, a CMT also contains attributes of those concepts and/or relationships between those concepts. Electronic CMTs are extremely useful and important for communication between and integration of independent information systems in healthcare, because data in this area is highly fragmented. A single query in this area might involve several databases, e.g., a clinical database, a pharmacy database, a radiology database, and a lab test database.

Unfortunately, the extensive sizes of CMTs, often containing tens of thousands of concepts and hundreds of thousands of relationships between pairs of those concepts, impose steep learning curves for new users of such CMTs. In this dissertation, we address the problem of helping a user to orient himself in an existing large CMT. In order to help a user comprehend a large, complex CMT, we need to provide abstract views of the CMT. However, at this time, no tools exist for providing a user with such abstract views. One reason for the lack of tools is the absence of a good theory on how to partition an overwhelming CMT into manageable pieces.

In this dissertation, we try to overcome the described problem by using a three-pronged approach. (1) We use the power of Object-Oriented Databases to design a schema extraction process for large, complex CMTs. The schema resulting from this process provides an excellent, compact representation of the CMT. (2) We develop a theory and a methodology for partitioning a large OODB schema, modeled as a graph, into small *meaningful* units. The methodology relies on the interaction

between a human and a computer, making optimal use of the human's semantic knowledge and the computer's speed. Furthermore, the theory and methodology developed for the schema-level partitioning are also adapted to the object-level of a CMT. (3) We use purely *structural similarities* for partitioning CMTs, eliminating the need for a human expert in the partitioning methodology mentioned above.

Two large medical terminologies are used as our test beds, the Medical Entities Dictionary (MED) and the Unified Medical Language System (UMLS), which itself contains a number of terminologies.

DEVELOPING TECHNIQUES FOR ENHANCING COMPREHENSIBILITY OF CONTROLLED MEDICAL TERMINOLOGIES

by
Huanying Gu

A Dissertation
Submitted to the Faculty of
New Jersey Institute of Technology
in Partial Fulfillment of the Requirements for the Degree of
Doctor of Philosophy

Department of Computer and Information Science

May 1999

APPROVAL PAGE

DEVELOPING TECHNIQUES FOR ENHANCING COMPREHENSIBILITY OF CONTROLLED MEDICAL TERMINOLOGIES

Huanying Gu

Dr. James Geller, Dissertation Advisor — Date
Associate Professor of Computer and Information Science, NJIT, Newark, NJ

Dr. Yehoshua Perl, Dissertation Co-Advisor — Date
Professor of Computer and Information Science, NJIT, Newark, NJ

Dr. Richard B. Scherl, Committee Member — Date
Assistant Professor of Computer and Information Science, NJIT, Newark, NJ

Dr. James J. Cimino, Committee Member — Date
Associate Professor of Medical Informatics, Columbia University, New York, NY

Dr. Michael Halper, Committee Member — Date
Assistant Professor of Computer Science, Kean University, Union, NJ

Dr. Waldemar G. Johanson, Jr., Committee Member — Date
Chairman and Professor of Medicine, New Jersey Medical School
University of Medicine and Dentistry of New Jersey, Newark, NJ

BIOGRAPHICAL SKETCH

Author: Huanying Gu

Degree: Doctor of Philosophy

Date: May 1999

Undergraduate and Graduate Education:

- Doctor of Philosophy in Computer Science,
  New Jersey Institute of Technology, Newark, NJ, 1999
- Master of Science in Computer Science,
  Nanjing University of Science and Technology, Nanjing, China, 1989
- Bachelor of Science in Computer Science,
  Huazhong University of Science and Technology, Wuhan, China, 1984

Major: Computer Science

Presentations and Publications:

H. Gu and Y. Perl and J. Geller and M. Halper and L. Liu and J. J. Cimino. Representing the UMLS as an OODB: modeling issues and advantages. Submitted for journal publication, 1999.

H. Gu and M. Halper and J. Geller and Y. Perl. Benefits of an OODB representation for controlled medical terminologies. *Journal of the American Medical Informatics Association,* in press, 1999.

H. Gu and Y. Perl and J. Geller and M. Halper and M. Singh. A methodology for partitioning a vocabulary hierarchy into trees. *Artificial Intelligence In Medicine,* 15(1):77-98, 1999.

H. Gu and J. Geller and L. Liu and M. Halper. Using a similarity measurement to partition a vocabulary of medical concepts. Submitted for conference publication, 1999.

H. Gu and Y. Perl and J. Geller and M. Halper and L. Liu and J. J. Cimino. Modeling the UMLS using an OODB. Submitted for conference publication, 1999.

H. Gu and L. Liu and M. Halper and J. Geller and Y. Perl. Converting an integrated hospital formulary into an object-oriented database representation. In C. G. Chute, editor, *Proceedings of the 1998 American Medical Informatics Association Annual Fall Symposium,* pp 770-774, Orlando, FL, November 1998.

H. Gu and Y. Perl and J. Geller and M. Halper and J. Cimino and M. Singh. Partitioning a vocabulary's IS-A hierarchy into trees. In D. R. Masys, editor, *Proceedings of the 1997 American Medical Informatics Association Annual Fall Symposium,* pp 630-634, Nashville, TN, October 1997.

L. Liu and M. Halper and H. Gu and J. Geller and Y. Perl. Modeling a vocabulary in an object-oriented database. In K. Barker and M.T. Ozsu, editor, *CIKM'96, Proceedings of the Fifth International Conference on Information and Knowledge Management,* pp 179-188, Rockville, MD, November 1996.

M. Halper and H. Gu and J. Cimino and J. Geller and Y. Perl. Comprehending the structure of a medical vocabulary using object-oriented database modeling. In *OOPSLA'96 Workshop on Object-Oriented Technology for Health Care and Medical Information System,* San Jose, CA, October 1996. Position paper.

H. Gu and J. Cimino and M. Halper and J. Geller and Y. Perl. Utilizing OODB schema modeling for vocabulary management. In J. J. Cimino, editor, *Proceedings of the 1996 American Medical Informatics Association Annual Fall Symposium,* pp 274-278, Washington, DC, October 1996.

Y. Perl and J. Geller and H. Gu. Identify a forest hierarchy in an OODB specialization hierarchy satisfying disciplined modeling. In *Proceedings of the First IFCIS International Conference on Cooperative Information Systems CoopIS'96,* pp 182-195, Brussels, Belgium, 1996

This work is dedicated to
Shenggang and Eric

ACKNOWLEDGMENT

I would like to extend my sincere gratitude to my advisors James Geller and Yehoshua Perl. Their invaluable guidance and encouragement have contributed significantly to the work presented in this dissertation. I would like to express a warm thanks to Dr. Michael Halper for his guidance, abundant help, and friendship throughout this research.

Special thanks to Dr. James J. Cimino for graciously sharing the MED with us, helping me understand the medical knowledge, and providing invaluable advice. I am thankful to Dr. Richard B. Scherl and Dr. Waldemar G. Johanson for serving on my committee and giving me helpful comments.

I would like to thank Li-min Liu for hanging tough together through the four years on OOHVR project. Thanks also due to Wenguang He, Zong Chen, David Ku, and Rajanikanth Tanikella for their help and friendship. I must thank all members of OOHVR project for our joyful days in AI & OODB lab.

I will always be indebted to my parents. Without their moral and intellectual guidance throughout my life, all these would be impossible. Also I wish to thank my brothers and parents-in-law for their continuous support and encouragement.

I dedicate this dissertation to my husband, Shenggang, and my lovely son, Eric, for their love, understanding, and support.

TABLE OF CONTENTS

Chapter **Page**

LIST OF TABLES

LIST OF FIGURES

Figure **Page**

Figure **Page**

CHAPTER 1

INTRODUCTION

1.1 Motivation

Controlled Medical Terminologies (CMTs) are collections of concepts that can be used to unify and consolidate disparate terminologies in the medical domain [26, 27, 33, 34]. CMTs have been used to encode drugs, diagnoses, procedures, etc. CMTs are core components of computer-based tools in the healthcare industry. They are used to reduce healthcare costs, provide better medical services, assess the quality of healthcare providers, and deliver healthcare services more efficiently. Large CMTs have been emerging as important resources for use in medical informatics applications, such as hospital department systems, electronic patient record systems, expert systems, and medical information systems [37].

While a CMT offers tremendous benefits, these benefits do come at a price. A CMT can be quite extensive and can contain an overwhelming amount of structural and semantic complexity. CMTs typically comprise on the order of tens of thousands to hundreds of thousands of interconnected concepts. The scopes and complexities of CMTs pose serious comprehension problems for users and even developers. It is difficult to maintain and use a CMT without a solid understanding of its overall structure and its content. Designers, maintainers, and users of CMTs will need tools to help with their work. There are tools for retrieval and manipulation of the contents of CMTs [113, 122, 123, 135, 138]. However, such tools are not sufficient. Rather, tools must be developed, that help professionals reach a level *comprehension* essential to performing their tasks.

The goal of our research is to make a large, complex CMT comprehensible. We aim at providing users of a large, complex CMT (both those whose job it is to maintain the CMT and those who build applications using it) with abstract views of the CMT to help them comprehend it. Our approach to achieve comprehension of a

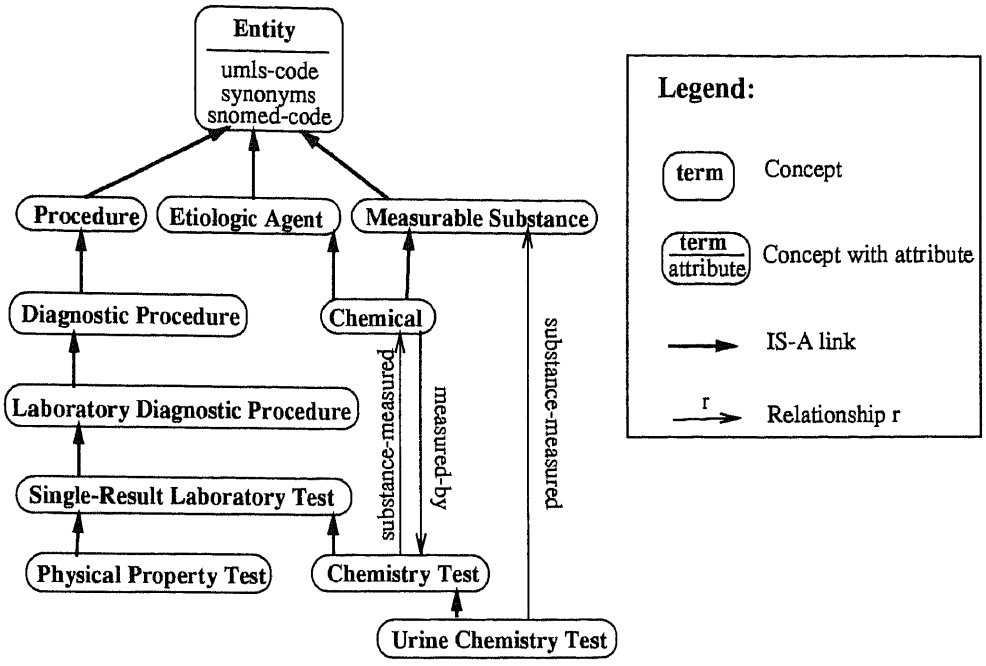

Figure 1.1 A small piece from a vocabulary

large, complex CMT is based on partitioning it into a collection of manageable and meaningful parts that can be used to derive an abstract view of the CMT. Thus, to comprehend a CMT, the user can start by studying the abstract view and then go on to study the details of parts that are interesting to him.

1.2 Background and Literature Review

A controlled terminology[1] [33, 34, 37, 38, 112] or an ontology [62, 63, 64] is a large network of concepts derived from a subject area of human knowledge. It typically contains an IS-A hierarchy which organizes the space of concepts (a *concept taxonomy*). In Figure 1.1, we show a small piece of a vocabulary. A concept should be seen as an unambiguous word sense. A concept is denoted by a primary term, while other terms referring to it are considered synonyms of the primary term.

[1] "Terminology" and "vocabulary" have similar meanings, referring to a body of related terms within a subject area. Both of them are used interchangeably in this dissertation.

In addition, vocabularies may contain (1) Attributes (holding literal data values) attached to a concept, which define the nature of the concept in more detail (umls-code, synonyms, etc. in Figure 1.1); (2) Relationships between concepts (storing references to other concepts, for example, measured-by, etc. in Figure 1.1); (3) Rules, constraints, or axioms, which add further grounding of a concept and make the vocabulary directly accessible to a reasoning engine.

Ontologies or controlled vocabularies appear in different areas. For instance, CYC [61, 85, 86, 87] is a general ontology for common sense knowledge to facilitate reasoning; WordNet [2, 103] is an online reference system that is one of the most comprehensive lexical ontologies; TOVE (Toronto Virtual Enterprise) [1, 47, 48, 49] is an ontology for enterprise modeling that is able to deduce answers to queries about the information in the model; and [13] describes an ontology from the law domain.

The healthcare informatics community has developed large concept taxonomy-based vocabularies for handling the ever-increasing glut of medical terms [37, 40, 112]. Examples of CMTs include Medical Subject Headings (MeSH) [105], the Systematized Nomenclature of Medicine (SNOMED) [42], SNOMED International [43], Physicians' Current Procedural Terminology (CPT) [6], International Classification of Diseases: 9th Revision, Clinical Modification (ICD-9-CM) [67], etc., that are integrated into the Unified Medical Language System (UMLS) [71, 73, 89, 136], GALEN's (General Architecture for Languages, Enclopedias, and Nomenclatures in Medicine) CORE (COncept REference) Model [120] (expressed in GRAIL (GALEN Representation And Integration Language) [121]), and the Medical Entities Dictionary (MED) [37, 40].

Most CMTs use structures that organize concepts into concept hierarchies. However, the hierarchies vary in the type of hierarchical relationship used between a parent and a child. In certain CMTs, such as MED [37, 40] and GRAIL [121], the relationship is called IS-A; in other CMTs, such as MeSH [105], the relationship is

unspecified and differs for different parent-child pairs. A child may have only one parent, such as in ICD-9-CM [67] and SNOMED [42]. Concepts in such terminologies form strict hierarchies or trees. On the other hand, a child may have more than one parent, as in terminologies or terminology models (e.g., MED and GRAIL) that reflect directed acyclic graphs.

Codes, names, and unique identifiers are common features in CMTs. In most CMTs, a code identifies a concept uniquely, such as in ICD-9-CM, SNOMED, MED, etc. Some of the older systems, such as ICD-9-CM and SNOMED, use the code not only to identify a concept uniquely, but also to indicate where in the hierarchy it is. This practice has the disadvantage of limiting placement of a concept to only one place in the hierarchy. If the code has a fixed number of digits, and each digit indicates a level in the hierarchy, the number of levels in the CMT will be limited. Newer systems, such as the MED and SNOMED-RT (SNOMED Reference Terminology) [134], do not use the code to indicate hierarchical location.

Several CMTs have long histories. For example, the International Classification of Diseases (ICD) was first initiated in 1853. It was intended as a list of causes of death, not as a full nomenclature of diseases or clinical findings. The World Health Organization assumed sponsorship of ICD in 1948 and continued to enrich its content, adding terms and codes for morbidity and for indexing hospital patients. In 1968, ICD was adopted for use in the United States. It was dramatically expanded by adding a large number of terms relevant to clinical medical practice. This modification was made to the 9th revision of ICD, and hence, has been referred to as ICD-9-CM (Clinical Modification). ICD-9-CM contains more than 10,000 terms for diagnoses and procedures. Terms have no cross-references. Each child has only one parent. All terms are in a strict, numerically coded hierarchy with five levels. Other examples of CMTs with long history are MeSH, dating back to 1960, and SNOMED

derived from the Systematized Nomenclature of Pathology, which itself was created in 1965 [41].

Among medical terminological systems, the UMLS (the Unified Medical Language System of the US National Library of Medicine) [71, 73, 89, 140] is of special importance, because it integrates a number of existing medical standards. The purpose of the UMLS is to aid the development of systems that help health professionals and researchers to retrieve and integrate electronic biomedical information from a variety of sources. The UMLS strategy focuses on the development of machine-readable *knowledge sources* that can be used by a wide variety of application programs to overcome the retrieval problems caused by differences in the way the same medical concept may be expressed in different sources [72]. That makes it easy for users to link disparate information systems, including computer-based patient records, bibliographic databases, factual databases, and expert systems.

The UMLS contains four knowledge sources: The Metathesaurus, the Specialist Lexicon, the Semantic Network, and the Information Sources Map. The UMLS knowledge sources are designed as multi-purpose tools, to facilitate the development of more effective biomedical information systems [98].

The UMLS Metathesaurus (META) provides a uniform, integrated distribution format for more than 40 biomedical vocabularies and classifications and links many sets of different names for the same concepts. The Metathesaurus is the largest and most complex among the four knowledge sources and is the foundation of the UMLS. It is a compilation of names, relationships, and associated information from a variety of biomedical naming systems representing different views of biomedical practice or research. The Metathesaurus is a machine-created, human edited and enhanced synthesis of authoritative biomedical terminologies. As such, it is a resource for maximizing the usefulness of existing vocabularies [107, 108, 125, 136, 137, 139].

The UMLS Semantic Network contains information about the types or categories (e.g., **Disease or Syndrome**, **Virus**) to which all Metathesaurus concepts have been assigned, and the permissible relationships among these types (e.g., **Virus** *causes* **Disease or Syndrome**) [95, 96, 97]. It provides a consistent categorization of all concepts found in the Metathesaurus and provides a set of useful links between these concepts at the level of the semantic types. Each concept in the Metathesaurus is assigned to one or more semantic types from the UMLS Semantic Network.

The UMLS knowledge sources have been applied in a wide variety of research and development environments to many different tasks, including vocabulary development, knowledge representation, clinical data capture, linking patient data to knowledge sources, curriculum analysis, natural language processing, automated indexing, and information retrieval. [28] describes the preliminary results of the attempt to reuse the UMLS Semantic Network as an ontology for the knowledge base of a patient education system. The MED [37], a controlled medical vocabulary developed and presently in use at Columbia-Presbyterian Medical Center (CPMC), was based on the UMLS. (Since we use it as a test bed in Chapter 2, we will describe details of the MED later.) [39] describes ways to apply an expert system approach to vocabulary integration and management by using the UMLS. [33] demonstrates that the UMLS structural model is appropriate for representing CPMC vocabularies and patient data and shows that the UMLS concepts provide excellent coverage of CPMC concepts in many areas. [75, 76, 77] show how to design conceptual models using the *conceptual graph formalism* [132], and how to implement computational models for information retrieval in large medical information databases. These models are based on the UMLS knowledge sources. In Chapter 3, we will describe how to model the Metathesaurus and Semantic Network as an OODB and what advantages there are to doing so.

CMTs have served diverse needs for many years. They are becoming larger and more complex as they evolve, and the demands placed upon them by the pursuit of the Electronic Medical Record (EMR) are enormous. It is difficult to maintain and use them. Tools are needed to support comprehension and maintenance of CMTs. At present, few commercial tools exist. All thoses tools and environments aid developers in construction of CMTs and provide support for managing and enhancing terminologies. Also, they facilitate distributed-development tasks. For example, [123] describes the Voser project for designing a CMT server. MEME II (Metathesaurus Enhancement and Maintenance Environment, Version II), described in [135], is a tool to support Metathesaurus maintenance and enhancement. It allows remote enhancements to a terminology to be incorporated locally, and local enhancements to be shared remotely. [114] describes how to manage the updating of large scale CMTs. In [93], K-Rep, a knowledge representation system based on a description logic, is used to model CMTs. This approach increases semantic consistency and inferential capability. Gálapagos is a configuration management and conflict resolution environment built on top of K-Rep [25]. It provides support for handling the inevitable conflicts generated by concurrent development of enhancements to a terminology. A proof-of-concept of Gálapagos is shown using an example in [25]. The use of semantic-based methods for managing concurrent terminology development to avoid disadvantages of traditional lock-based approaches common in database systems is presented in [24]. [29, 65] discuss problems in reconciling the needs of natural language understanding with more general requirements of concept representations for medical information. [65] introduces a knowledge-based approach to medical text understanding. [29] describes using syntactic-semantic tagging to transform a linguistic representation of surgical procedure expressions into conceptual representations.

An object-oriented knowledge representation framework has been used as a modeling vehicle for thesauri that were employed in (natural) language-to-language translation [45, 46]. A terminology editor called TEDI was also built in that context as a tool for extracting relevant information from hypermedia documents [104]. In [149], an electronic dictionary system (EDS) was developed with object-oriented database [15, 79, 148] techniques based on ObjectStore [82, 131].

1.3 Dissertation Overview

Controlled medical terminologies are fundamental to computer-based systems that support healthcare. However, CMTs are always large and complex, and working with them can be daunting. It is important to provide a means for orienting terminology designers and users to the terminology's contents. In order to help a user comprehend a large, complex CMT, we need to show him abstractions of the original CMT. This dissertation focuses on theories and methodologies for partitioning a CMT into manageable and meaningful pieces which form an abstract view of the original CMT.

This dissertation is an amalgamation of five papers. They are organized as follows. The published journal paper [56] is Chapter 2 which presents benefits of an OODB representation for CMTs. Chapter 3 contains the journal paper [58] (to be submitted shortly) which describes how to model the UMLS as an OODB and what the advantages of such a model are. The paper [57] (to be submitted to a journal shortly), which describes a theoretical framework and a human-computer interactive methodology for partitioning a large OODB schema into sub-schemas, is presented in Chaper 4. The published journal paper [59], adapting the theoretical framework and the methodology for partitioning an OODB schema to CMTs, appears in Chapter 5. Chapter 6 is the submitted conference paper [55] that describes another approach to partitioning a CMT into trees, based on structural similarity. Finally, conclusions of this dissertation will appear in Chapter 7.

1.3.1 An OODB Representation for Controlled Medical Terminologies

In our research, we are exploring the use of an object-oriented database (OODB) paradigm for representing CMTs to make large, complex CMTs understandable. We develop a theoretical framework for transforming a CMT into an OODB. OODBs have a data layer and a schema layer. In database systems, the schema always exists before the data. In our approach, the data (CMT) exist first, and a schema is abstracted from it.

An OODB schema which captures the structure of a controlled vocabulary in a compact way improves comprehension of the gestalt of a large controlled vocabulary. We use the MED [37] as our test bed and convert it into an OODB. Using the high-level view of a CMT afforded by the schema, one can gain insight into the CMT's overarching organization and better comprehend it. In Chapter 2, we show how the comprehension that the MED OODB schema provides uncovers some errors and inconsistencies made in the vocabulary's original modeling. This enabled designers to easily correct the mistakes and improve the original vocabulary content.

1.3.2 Representing the UMLS as an OODB

Since the UMLS of NLM combines many well established authoritative medical informatics terminologies in one knowledge representation system, it is very valuable to the healthcare community and industry. To support the comprehension and navigation of the UMLS, in Chapter 3, we use an OODB representation to represent the two major components of the UMLS, the Metathesaurus and the Semantic Network.

The UMLS OODB schema is based on the existing UMLS Semantic Network. To model the Metathesaurus and the Semantic Network as an OODB, we represent each semantic type in the Semantic Network as a *semantic type class* in the OODB schema. In Chapter 3, we will discuss why this straightforward approach to modeling

the UMLS is insufficient, and introduce a more sophisticated approach. All concepts assigned to only one semantic type become instances of the corresponding *semantic type class*. Each concept assigned to multiple semantic types becomes an instance of a new kind of class, called an *intersection class*. Furthermore, we introduce a rule to systematically define subclass relationships for all intersection classes.

As a result of this modeling, all classes abstract semantically uniform sets of concepts. The resulting UMLS OODB schema has a deeper and more refined structure than the Semantic Network of the UMLS. This is a modeling improvement which is completely in line with the design goals of the UMLS [98]. The UMLS OODB schema also supports the improved comprehension and navigation of the Metathesaurus. Furthermore, the intersection classes expose some problems existing in the current UMLS, such as concept omissions, classification errors, and ambiguities of concepts.

1.3.3 Partitioning Based on Semantics

OODB schemas are helpful in making CMTs comprehensible, but they are not sufficient. Sometimes even an OODB schema can be too large to understand (e.g., the UMLS OODB schema contains more than 1,000 classes). In order to maximize human comprehension, we need to partition a large OODB schema into disjoint meaningful, manageably sized parts (called contexts) which form a *macro structure* of the original OODB schema.

Our framework for partitioning an OODB schema is based on two concepts: *informational thinning* and *partitioning*. We first eliminate certain kinds of properties from the representation resulting in a directed acyclic graph (DAG) of all classes and subclass relationships between them. Then we find a forest of trees to replace this DAG, based on a set of three *rules of disciplined modeling*. We prove that, adhering to these three rules, such a forest can always be found. Based on

our partitioning framework, we develop a methodology that relies on an interaction between a user and the computer for finding a forest hierarchy. Such a hierarchy functions as a skeleton of the schema and supports comprehension and partitioning efforts. We will demonstrate our methodology by applying it to a subschema of a university database and the MED OODB schema.

Since the number of instances of classes may be large (e.g., an average class in the MED schema has 500 instances), it is still hard to understand. Thus, further partitioning efforts are needed to enhance comprehension. In Chapter 5, we adapt the theoretical paradigm and methodology developed for schemata to an extensive, complex vocabulary itself. By using the adapted methodology, we can partition a large, complex vocabulary into a collection of small contexts such that each context consists of a meaningful group of related concepts and fits onto a single screen. This enables the user to study and comprehend one of these groups (called contexts) at a time and step by step build a comprehension of the whole vocabulary.

1.3.4 Partitioning Based on Structure

Since the above human-computer interactive methodology needs to use a human expert to make decisions, it is very expensive. We want to automate this process. A structural analysis of a large hierarchy of concepts shows that there are "natural breakpoints" in the hierarchy. Our structural method for automating the partitioning of a vocabulary is based on a definition of the similarity of a pair consisting of a child concept and its parent concept in the vocabulary. These kinds of similarities can be measured numerically. In Chapter 6, we compare both the structure and values of every concept with those of all its parents. In some cases, concepts will be very similar to their parents, in that they have the same structure (attributes, relationships) and inherit all values. In other cases, concepts may be quite different from their parents, because they introduce new structure, and they *override* inherited

values. A distribution over these similarities for all pairs is then computed. Based on this distribution, the vocabulary can be partitioned into manageable pieces. The results of applying this approach are similar to the results of applying the approach described in Chapter 5.

CHAPTER 2

BENEFITS OF AN OODB REPRESENTATION FOR CONTROLLED MEDICAL TERMINOLOGIES

2.1 Introduction

Controlled medical terminologies (CMTs) have been recognized as important tools in a variety of medical informatics applications ranging from patient-record systems to decision-support systems. CMTs are typically organized in semantic network structures consisting of tens to hundreds of thousands of concepts. This overwhelming size and complexity can be a serious barrier to their maintenance and wide-spread utilization.

In this chapter, we address some of the problems of terminology comprehension by presenting a methodology for representing a CMT, modeled using the semantic network paradigm [83, 133, 144], as an object-oriented database (OODB) [15, 79, 148]. We refer to such a representation as an Object-Oriented Healthcare Terminology Repository (OOHTR) [90, 91]. One of the most important components of the OOHTR is its schema, which provides an abstraction layer through which the CMT can be viewed and studied. This compact presentation of the CMT helps to shed light on its overarching structure.

We will use the Medical Entities Dictionary (MED) as our test bed. Studies have shown that users of the MED at Columbia-Presbyterian Medical Center (CPMC) have trouble navigating through its constituent semantic network to find desired concepts [70]. The complexity of the MED also presents challenges to its maintenance personnel who often find it difficult to add concepts or create links without a clear understanding of the underlying terminology structural model. Others approaching the MED have encountered similar difficulties [78]. In this chapter, we will demonstrate how the OOHTR's schema was used to uncover some conceptual errors and inconsistencies in the MED—some that had been introduced

initially and others that had crept in over time. These discoveries led directly to improvements in the MED's design.

The rest of the chapter is organized as follows. In the next section, we give an overview of the structural characteristics of CMTs, like the MED, that are modeled as semantic networks. Section 2.3 describes our methodology for modeling a CMT as an OODB, and presents the results of applying the methodology to the MED to produce an OOHTR. Finally, in Section 2.4, we discuss how the OOHTR's schema helped in the improvement of the MED design by exposing errors which were subsequently corrected. In Section 2.5 we compare our approach with approaches based on the use of Description Logics. Summary appears in Section 2.6.

2.2 Semantic Network CMTs

A CMT that is amenable to our methodology must have the structure of a semantic network. Such a CMT is a collection of medical concepts, each of which consists of properties that are either attributes (holding literal data values) or relationships (storing references to other concepts). One attribute needed in each concept contains the concept's associated *term* (or textual denotation) [46]. In the MED this attribute is called *name*. Another attribute, called *synonyms* in the MED, needs to hold additional denotations for a concept. As an example of a relationship in the MED, *is-measured-by* connects the concept **Chemical** to the concept **Chemistry Test**. Let us note that, throughout the chapter, terms will be written in a bold font. Property names will appear in italics and will be written strictly in lowercase letters.

The CMT's concepts must be organized into a concept subsumption hierarchy: a directed acyclic graph (DAG) composed of concepts (nodes) and IS-A links, each connecting a concept to its superconcept. The IS-A links provide the means for the inheritance of attributes and relationships, and they support subsumption-based reasoning. A concept may have more than one parent in the hierarchy. For

example, in the MED, **Chemical** IS-A **Measurable Substance** and **Chemical** IS-A **Etiologic Agent**. We will assume that all CMTs are singly-rooted. The MED's IS-A hierarchy is rooted overall at the concept **Medical Entity**.

CMTs tend to be large and complex in scope. At the time of our research, the MED comprised about 43,000 concepts,[1] which were connected by over 71,000 (non-hierarchical) relationships. The IS-A links totaled over 61,000. Figure 2.1 shows a small portion of the MED (68 concepts or about 0.16% of the entire MED). In Figure 2.2 we present the notational conventions used for semantic networks.

Figure 2.1 contains the six concepts **CPMC Drug: Benadryl 25MG Cap**, **Pancreatin**, **Calcification of Pericardium**, **Amylase**, **Allen Serum Specimen**, and **Allen Serum Amylase Measurement**, along with most of their ancestors in the IS-A hierarchy and some of the relationships between the respective concepts. Included are concepts for laboratory tests, medications, and diagnoses. For brevity, some details have been omitted, including: additional children of the ancestor concepts, all attributes, and some relationships. Furthermore, the names of relationships have been written as numerical codes, whose meanings can be found in Table 2.1 in Section 2.3.1 .

As discussed in [37], the content of a CMT should satisfy the following seven basic requirements:

1. Domain completeness: There should be no numerical limitation on the size of any of the CMT's dimensions. (E.g., no limit on the depth of the IS-A hierarchy.)
2. Synonymy: Concepts can be recognized by multiple names.
3. Nonvagueness: Each concept must have a well-formed meaning.
4. Nonredundancy: No two concepts may have the same meaning.
5. Nonambiguity: Each concept may have no more than one meaning.
6. Multiple classification: Concepts may have more than one superconcept in the IS-A hierarchy.

[1] This was the 1996 version; the MED has since grown to over 59,000 concepts.

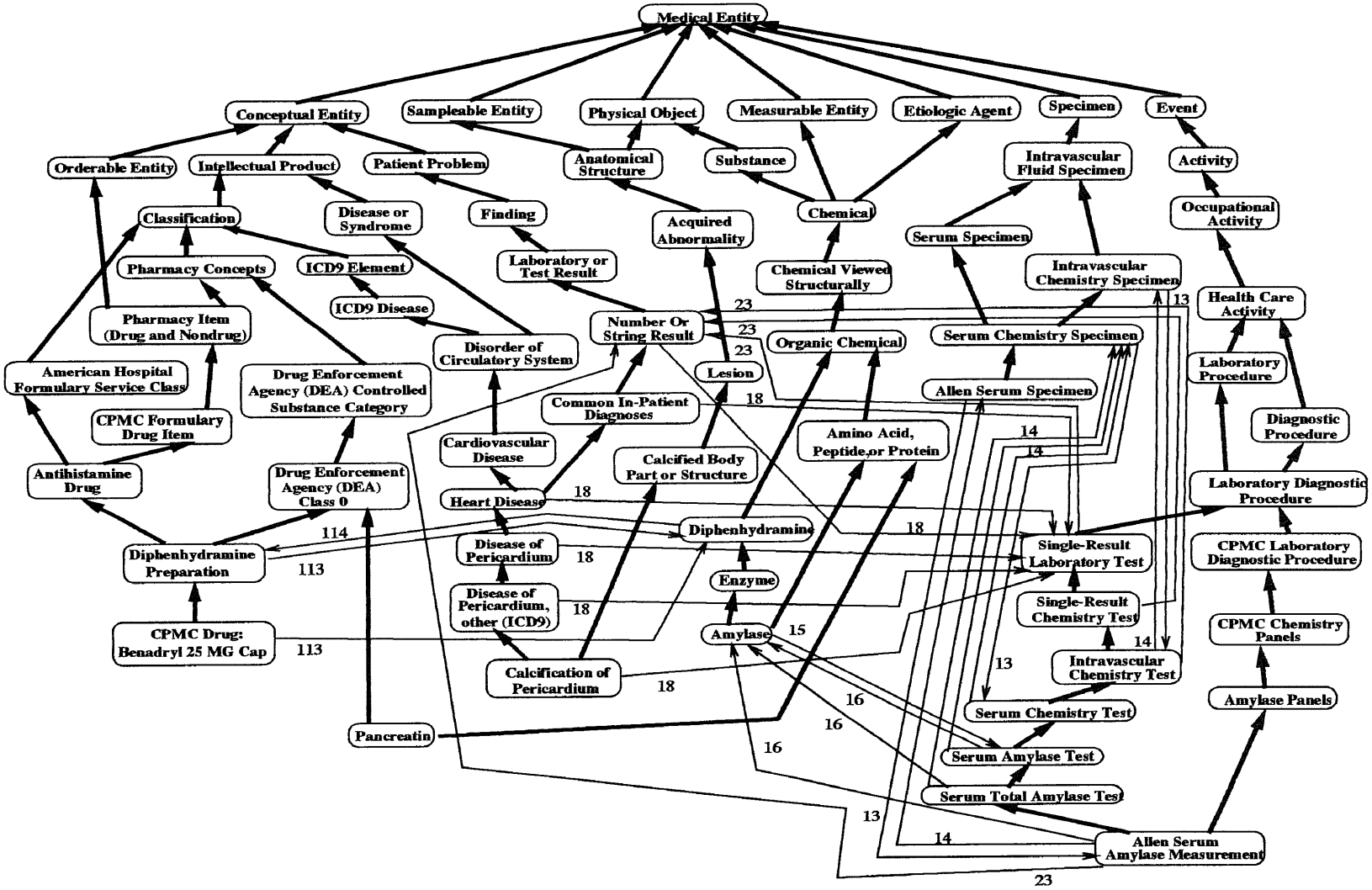

Figure 2.1 Sample of the MED content (see Figure 2.2 for legend of symbols)

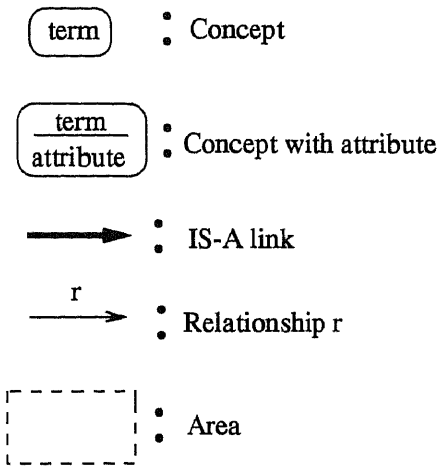

Figure 2.2 Legend of symbols in figures of semantic networks

7. Consistency of views: A concept should appear the same (and have the same properties and children) no matter how the concept is arrived at in the hierarchy.

We will also assume that a CMT satisfies the following rule regarding the introduction of properties.

Rule 1: A given property x (whether it be an attribute or a relationship) can be introduced at only one concept in the CMT. □

This requirement is not limiting, because if several concepts need to introduce the property x, then an "artificial" parent of them can be added to accommodate the unique introduction [35]. The MED, to which we will be applying our methodology, satisfies Rule 1.

2.3 The OOHTR Schema

2.3.1 Initial OOHTR Schema

The strategy that we chose for modeling a CMT as an OOHTR utilizes special concepts as the basis for the definitions of object classes in the schema. It, in fact, produces an abstraction of the underlying pattern in which properties are introduced into the CMT.

In general, the purpose of an object class in an OODB schema is to abstractly define a collection of properties for a group of objects (or instances) that exhibit those exact properties and have a common semantics. In a CMT, there are some concepts which function in an analogous role: Each introduces (defines) attributes and relationships that are exhibited by all its children and descendants in the IS-A hierarchy (due to the inheritance mechanism). We call such concepts *property-introducing concepts*. As will be discussed further below, very few concepts in a CMT are property-introducing. Almost all of them inherit all their properties.

A property-introducing concept also plays the role of the most general conceptual entity among its descendants. In this way, it captures the overarching semantics of the descendants.

Due to these facts, it is sensible to construct object classes with respect to all the property-introducing concepts appearing in the CMT. Toward that end, we define the notion of *area* to be a set containing one property-introducing concept plus all that concept's descendants which have the same properties. Note that some descendants can have more properties than the property-introducing concept; in such cases, the descendants do not belong to the area. The property-introducing concept of an area is called the area's *root* since it is the area's highest concept in the IS-A hierarchy (i.e., the property-introducing concept's parents are not in the same area because they lack the properties it introduces). An area will also be named by its property-introducing concept. An area with property-introducing concept **A** will be called "Area A" or "A Area."

To illustrate the notations of property-introducing concept and area, we show an excerpt from the MED in Figure 2.3. The figure contains six property-introducing concepts: **Medical Entity**, **Drug Allergy Class**, **Event Component**, **Radiology Term**, **Pharmacy Order Observation**, and **Pharmacy Allergy Observation**. It also contains the concept **Pharmacy Order Component** which is not property-

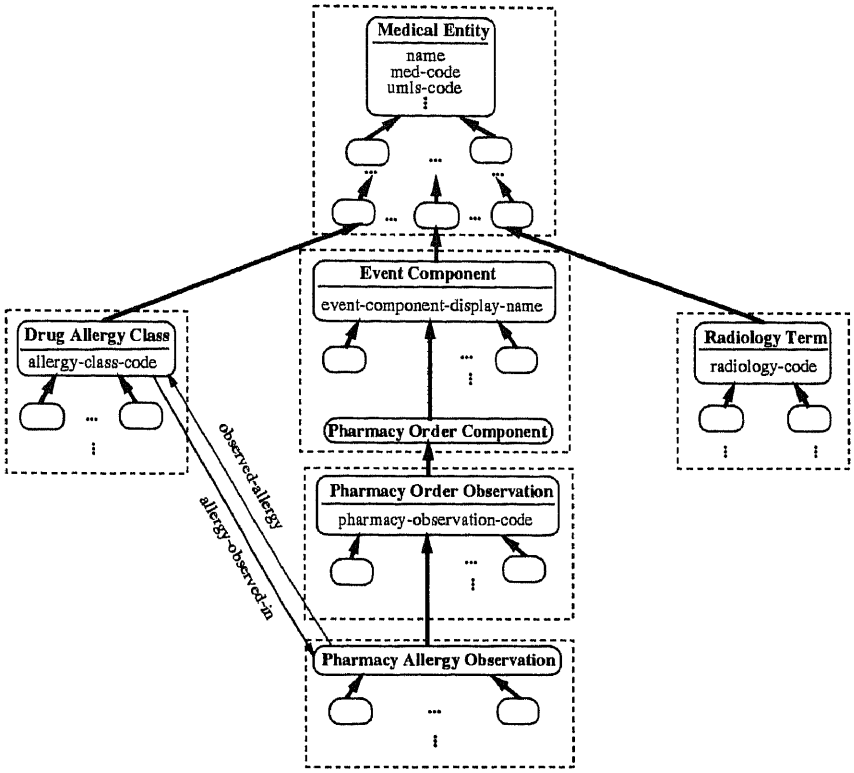

Figure 2.3 Six areas of the MED (see Figure 2.2 for legend of symbols)

introducing. Other concepts are left unlabeled. The concept **Medical Entity**, the root of the entire MED, introduces the attributes *name*, *med-code*, and *umls-code* (among others). The concept **Drug Allergy Class** introduces the attribute *allergy-class-code* and the relationship *allergy-observed-in* directed to the concept **Pharmacy Allergy Observation**. The concepts **Event Component**, **Radiology Term**, and **Pharmacy Order Observation** each introduce a single attribute. Finally, **Pharmacy Allergy Observation** introduces the relationship *observed-allergy*, the converse of *allergy-observed-in*.

There are six areas in Figure 2.3, each of which is enclosed in a large, dashed rectangle. The area rooted at concept **Medical Entity** extends down to, but excludes, concepts **Drug Allergy Class**, **Event Component**, and **Radiology Term**. Those three concepts are roots of areas of their own. Examples of some of the

Drug Allergy Class Area's concepts, not shown in the figure, are **Glucocorticoids**, **Codeine**, **Morphine**, **Barbiturates**, **Tetracyclines**, and **Phenothiazines**. The Event Component Area extends down to include the concept **Pharmacy Order Component**, which is the parent of **Pharmacy Order Observation**, the root of the Pharmacy Order Observation Area. The last area is rooted at **Pharmacy Allergy Observation**.

Once the property-introducing concepts and their respective areas have been identified, object classes can be created to represent them. Such a class will serve the dual purposes of defining the properties for an area and holding the area's concepts—all of which have identical structure and semantic similarity—as its instances. For this reason, we refer to a class in the OOHTR schema as an *area class*.

To be more precise: For each area in the CMT, we define an object class whose instances will be exactly the area's concepts, including its root. The class's name is formed by concatenating the name of the area's root and "_Area." The properties defined by an area class are identical to those introduced by the area's root within the CMT. So, for example, the Medical Entity Area would have the corresponding class *Medical_Entity_Area*, which would define the properties *name*, *med-code*, *umls-code*, etc.

Let us note that the root of an area exhibits all the properties that it itself introduces plus the properties that it inherits from its parent(s). The area's other concepts, of course, also have these same properties. To reflect this situation, we utilize the standard subclass inheritance of OODB schemata. A given area class *A_Area*, corresponding to Area *A*, is made a subclass of each area class that contains a parent of the root of Area *A*. As we discussed above, a concept may have more than one parent, and the subclass hierarchy induced by this process is therefore not necessarily a tree. That is, an area class can have multiple area classes as

superclasses. In addition to the properties that it defines intrinsically, an area class has all the properties of its superclasses through inheritance.

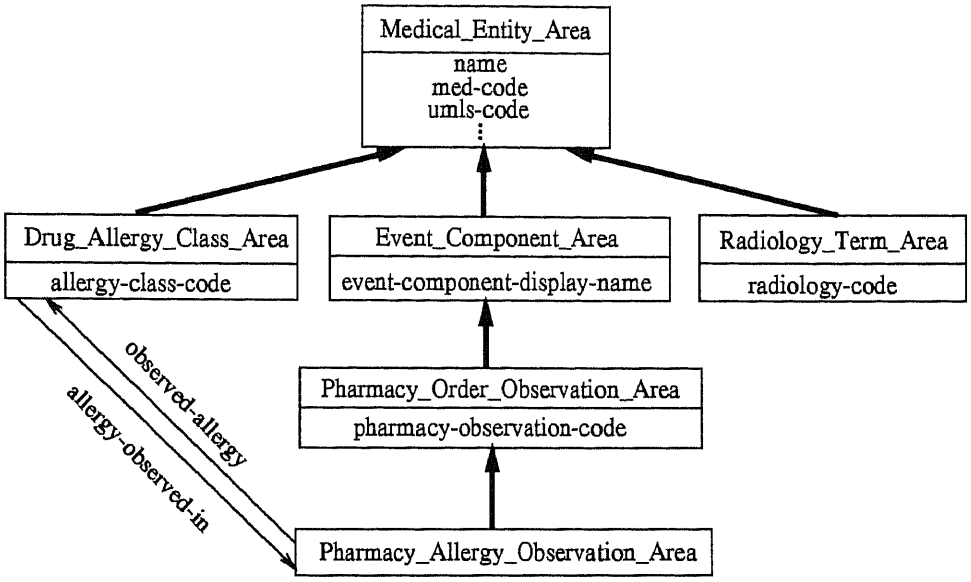

Figure 2.4 Area classes corresponding to the MED areas in Figure 2.3 (see Figure 2.5 for legend of symbols)

In Figure 2.4, we show the schema corresponding to the six areas in Figure 2.3. Note that the schema is represented using our OOdini-2 graphical notation [66]. In Figure 2.5 we show the notation used for OODB schemas. With OOdini-2, a class is represented as a rectangle, and a relationship, as a labeled arrow. A subclass relationship is drawn as a bold arrow directed upward from the subclass to the superclass. An attribute is listed inside its class rectangle beneath the

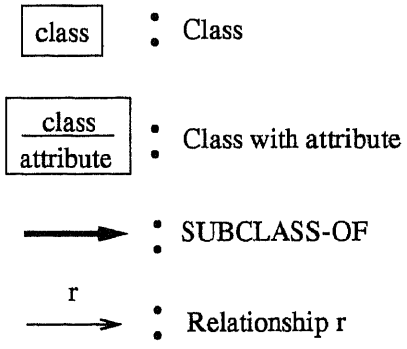

Figure 2.5 Legend of symbols in figures of OODB schemas

class's name. We can see that there are six object classes, one for each area in Figure 2.3. The classes have the properties introduced by the corresponding roots (again, see Figure 2.3). For example, the class *Medical_Entity_Area* has the attributes *name*, *med-code*, *umls-code*, etc. As another example, the class *Pharmacy_Allergy_Observation_Area* has the relationship *observed-allergy* directed to the class *Drug_Allergy_Class_Area*, and it is a subclass of *Pharmacy_Order_Observation_Area* from which it inherits the attributes *name*, *med-code*, *umls-code* (and so on), *event-component-display-name*, and *pharmacy-observation-code*.

The overall OOHTR schema produced by this mapping turns out to be very compact in its number of classes, particularly when one compares that number to the MED's tens of thousands of concepts. The compactness results from the fact that the total number of distinct properties in the MED is only 150. This implies that there are at most 150 property-introducing concepts, out of the 43,000 in the entire terminology. So, it can be seen that most concepts in the MED are not property-introducing. Because some concepts introduce multiple properties, the number of property-introducing concepts is actually just 53. The above process thus identifies 53 areas for the MED's 43,000 concepts, and the OOHTR schema consists of only 53 area classes.

Figure 2.6 presents the entire OOHTR schema obtained via the above described mapping. To save space, only numeric codes of attributes and relationships are shown. For example, attribute "9" is *lab-procedure-code*; relationship "18" is *result-of-tests*. Table 2.1 gives the codes and corresponding names for all attributes and relationships. The area class *Medical_Entity_Area* that corresponds to the MED's overall root **Medical Entity** becomes the top class in the OOHTR schema's class hierarchy. As we mentioned, an area class can have more than one superclass. This is demonstrated by the class *Chemical_Area* which has the superclasses *Measurable_Entity_Area* and *Etiologic_Agent_Area* (Figure 2.6).

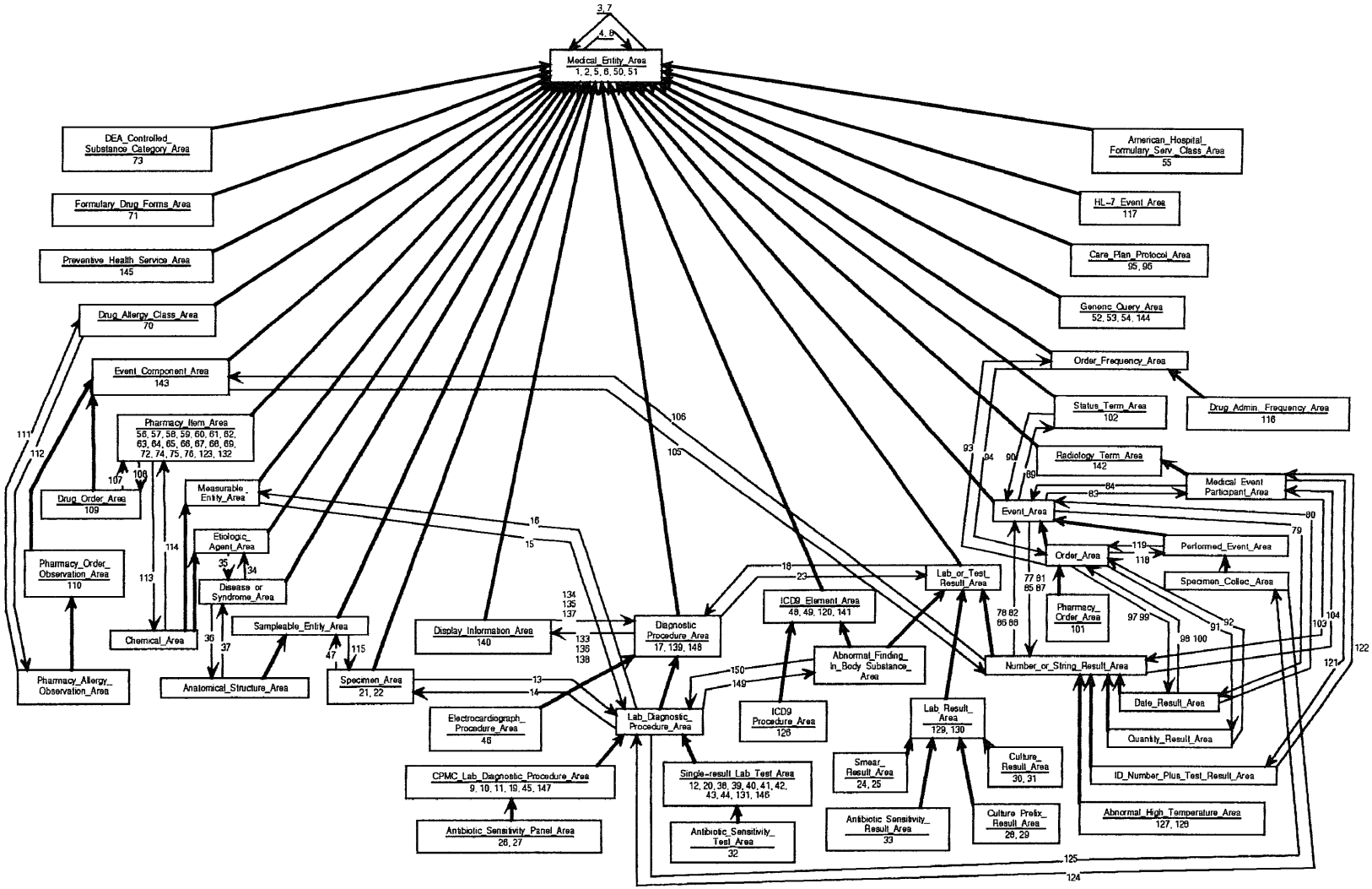

Figure 2.6 Schema derived from the areas of the MED (see Figure 2.5 for legend of symbols)

In [90], we presented a program called the *OOHTR Generator* which automatically generates the OODB schema for a given CMT. That program was used to build the MED's OOHTR schema shown in Figure 2.6. It has also been applied to the InterMED [90].

The MED's IS-A hierarchy served as the basis for the mapping into the OOHTR schema. In fact, the mapping constituted the identification of the property-introducing concepts and a "collapsing" of the inheritance paths between them. Thus, the OOHTR schema can be seen as an abstraction of the property definitions and accompanying inheritance that occur within the MED. We call this kind of schema a *network abstraction schema.* In order to preserve the actual IS-A connections between concepts from within the source CMT, a pair of converse relationships *has-superconcept* and *has-subconcept* is added to the area class *Medical_Entity_Area* (see Figure 2.6). Due to this, the two properties are exhibited by all concepts in the OOHTR. The relationships are used to connect a given concept to its parents and children, respectively. If the concept **A** IS-A **B** in the CMT, then the object representing **A** in the OOHTR refers to the object denoting **B** via *has-superconcept.* Conversely, **B** relates to **A** through *has-subconcept.*

2.3.2 Extended OOHTR Schema

One complication in the above mapping arises because of the multiple inheritance which occurs in the CMT's IS-A hierarchy. (Recall that it is a DAG, not a tree.) The problem is illustrated for the MED in Figure 2.7, which expands Figure 2.3 to include the concept **Radiology Event Component** (and some of its descendants). It will be noted that **Radiology Event Component** is not a property-introducing concept. It is, however, a child of two property-introducing concepts, **Event Component** and **Radiology Term**, and inherits its properties from both of them. The latter point gives rise to the difficulty. Since **Radiology Event Component**

Table 2.1 Names of properties in Figure 2.6

1	umls-code	2	name	3	has-subconcept
4	has-superconcept	5	synonyms	6	print-name
7	has-parts	8	part-of	9	cpmc-lab-proc-code
10	service-code	11	cpmc-unit-names	12	cpmc-lab-test-names
13	specimen-of	14	specimen	15	measured-by
16	substance-measured	17	units	18	result-of-tests
19	cpmc-lab-proc-name	20	cpmc-lab-test-code	21	cpmc-lab-spec-code
22	cpmc-lab-spec-name	23	result-type	24	cpmc-smear-code
25	cpmc-smear-name	26	cpmc-panel-code	27	cpmc-panel-name
28	cpmc-prefix-code	29	cpmc-prefix-name	30	cpmc-result-code
31	cpmc-result-name	32	cpmc-sensitivity-name	33	cpmc-sensitivity-result-name
34	etiology	35	causes-diseases	36	site
37	site-of-diseases	38	normal-value	39	low-normal-value
40	high-normal-value	41	male-low-normal-value	42	male-high-normal-value
43	female-low-normal-value	44	female-high-normal-value	45	normal-ranges-text
46	cpmc-ecg-name	47	substance-sampled	48	icd9-code
49	icd9-entry-code	50	main-mesh	51	supplementary-mesh
52	question-type	53	english-question	54	brs-question
55	ahfs-class-code	56	dose-strength-units	57	dose-strength-number
58	formulary-name	59	short-formulary-name	60	formulary-code
61	drug-trade-name	62	drug-generic-name	63	drug-manufacturer
64	drug-rx-vs-otc	65	drug-form-code	66	drug-floor-stock
67	drug-route	68	drug-in-formulary	69	drug-volume
70	allergy-class-code	71	drug-description	72	drug-category
73	dea-code	74	drug-specifier	75	drug-generic-code
76	drug-interaction-codes	77	event-id	78	event-id-of
79	event-date	80	event-date-of	81	event-patient-id
82	event-patient-id-of	83	event-participant	84	participant-of
85	event-organization	86	event-organization-of	87	event-location
88	event-location-of	89	event-status	90	status-of
91	order-quantity	92	order-quantity-of	93	order-frequency
94	order-frequency-of	95	protocol-name	96	protocol-short-name
97	order-start-date	98	order-start-date-of	99	order-stop-date
100	order-stop-date-of	101	pharmacy-order-code	102	status-code
103	participant-id	104	participant-id-of	105	order-value
106	order-value-of	107	ordered-drug	108	ordered-in
109	drug-role-code	110	pharmacy-observation-code	111	observed-allergy
112	allergy-observed-in	113	pharmaceutic-component	114	pharmaceutic-component-of
115	sampled-by	116	admin-frequency-abbrev	117	hl7-event-code
118	event-object	119	object-of-event	120	old-icd9-code
121	participant-name	122	participant-name-of	123	drug-id
124	collected-for	125	collected-by	126	cpt4-code
127	lower-limit-for-input	128	upper-limit-for-input	129	lab-message-code
130	lab-message-text	131	cpmc-long-test-name	132	drug-alert-code
133	has-default-displays	134	default-display-for	135	displays-elements-of
136	elements-displayed-by	137	has-display-parameters	138	is-display-parameter-of
139	has-test-display-class-name	140	display-parameter-order	141	icd9-name
142	cpmc-radiology-code	143	event-component-display-name	144	query-fillers
145	preventive-health-name	146	lab-alt-test-name	147	lab-alt-proc-name
148	has-proc-display-class-name	149	defined-by-test	150	defines-abnormal-finding

inherits from **Event Component**, it has a different set of properties than its parent **Radiology Term** and is therefore not in the Radiology Term Area. Likewise, it is not in the Event Component Area, either. In fact, **Radiology Event Component** does not reside in any area! As such, it currently has no representation within the OOHTR. The same is true of its descendants.

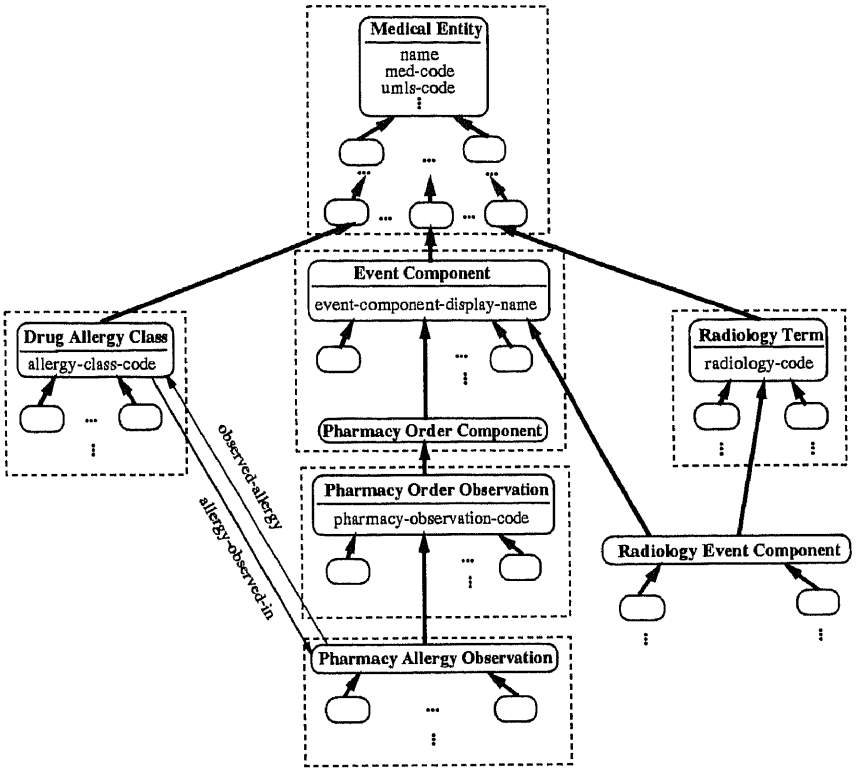

Figure 2.7 Expanded version of Figure 2.3 (see Figure 2.2 for legend of symbols)

Our solution is to introduce a new kind of area to include concepts like **Radiology Event Component**. In general, such a concept is characterized by the fact that its property set differs from the property sets of all property-introducing concepts in the CMT. While such a concept does not introduce any new properties of its own, it does lie at the juncture of "independent" inheritance paths and uniquely collects groups of properties. For this reason, we call such a concept an *intersection concept*. Note that we preclude a concept from being an intersection concept if it has an intersection concept ancestor with the same set of properties. For example,

in Figure 2.7, **Radiology Event Component**'s two children, whose names are **Radiology Report Event Component** and **Radiology Service Modifier**, are not intersection concepts.

We now define a new kind of area (called an *intersection area*) to be a set containing one or more intersection concepts having the same set of properties and all their descendants with the same properties. The intersection concepts residing in an intersection area will be called the roots of the area because they are the area's highest concepts in the IS-A hierarchy (i.e., their parents do not belong to the area). An example of an intersection area is the one containing the root **Radiology Event Component**, its two children **Radiology Report Event Component** and **Radiology Service Modifier**, and an additional 79 descendants. This intersection area contains just one root; however, an intersection area can be multi-rooted. As an example, the three concepts **Antihistamine Drugs**, **Anti-Infective Agents**, and **Autonomic Drugs** are children of the two property-introducing concepts **AHFS Service Class** and **Formulary Drug Item**. Hence, all three are intersection concepts, have the same set of properties, and root the same intersection area. In fact, there are 28 other intersection concepts that also have this property set, and therefore this particular intersection area has a total of 31 roots.

As with the areas rooted at property-introducing concepts, a separate class is created in the OOHTR schema for each intersection area. This new kind of class is referred to as an *intersection (area) class*. The concepts in the intersection area become instances of this intersection class, which, interestingly, does *not* define any new properties [just like its root(s)]. Instead, it gets its properties entirely via OO subclass inheritance. The subclass relationships for an intersection class are determined by the parentage of its root(s) in an analogous manner to that for ordinary area classes. Another interesting point is that an intersection class must have at least two superclasses in the schema's subclass hierarchy. Hence, the presence

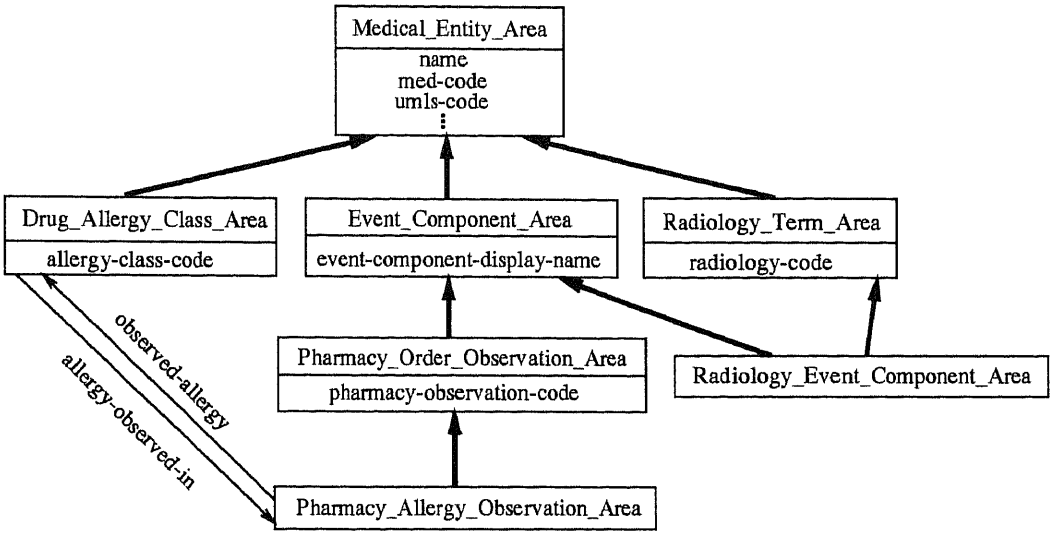

Figure 2.8 Schema for the areas in Figure 2.7 (see Figure 2.5 for legend of symbols)

of intersection concepts in the CMT implies multiple inheritance within the OOHTR schema.

If an intersection area has a unique root, then its corresponding intersection area class will naturally be denoted using the root's name (concatenated with "_Area"). Otherwise, one of the roots—say, the one appearing first in some search of the CMT—is arbitrarily selected as the name of the intersection class. In Figure 2.8, we show the classes for the areas appearing in Figure 2.7. The only addition to the schema from Figure 2.4 is the class *Radiology_Event_Component_Area*, representing the intersection area rooted at the concept **Radiology Event Component**. Note that it is a subclass of both *Event_Component_Area* and *Radiology_Term_Area*.

The entire OOHTR schema for the MED comprises a total of 90 area classes, 37 of which are intersection classes, and 131 subclass relationships. Of the 37 intersection classes, 22 contain a single root and 15 are multi-rooted. Even though the schema is large, one should bear in mind that it abstracts a CMT of 43,000 concepts—a network 632 times the size of the excerpt of 68 concepts shown in Figure 2.1. Each class contains about 477 concepts on average.

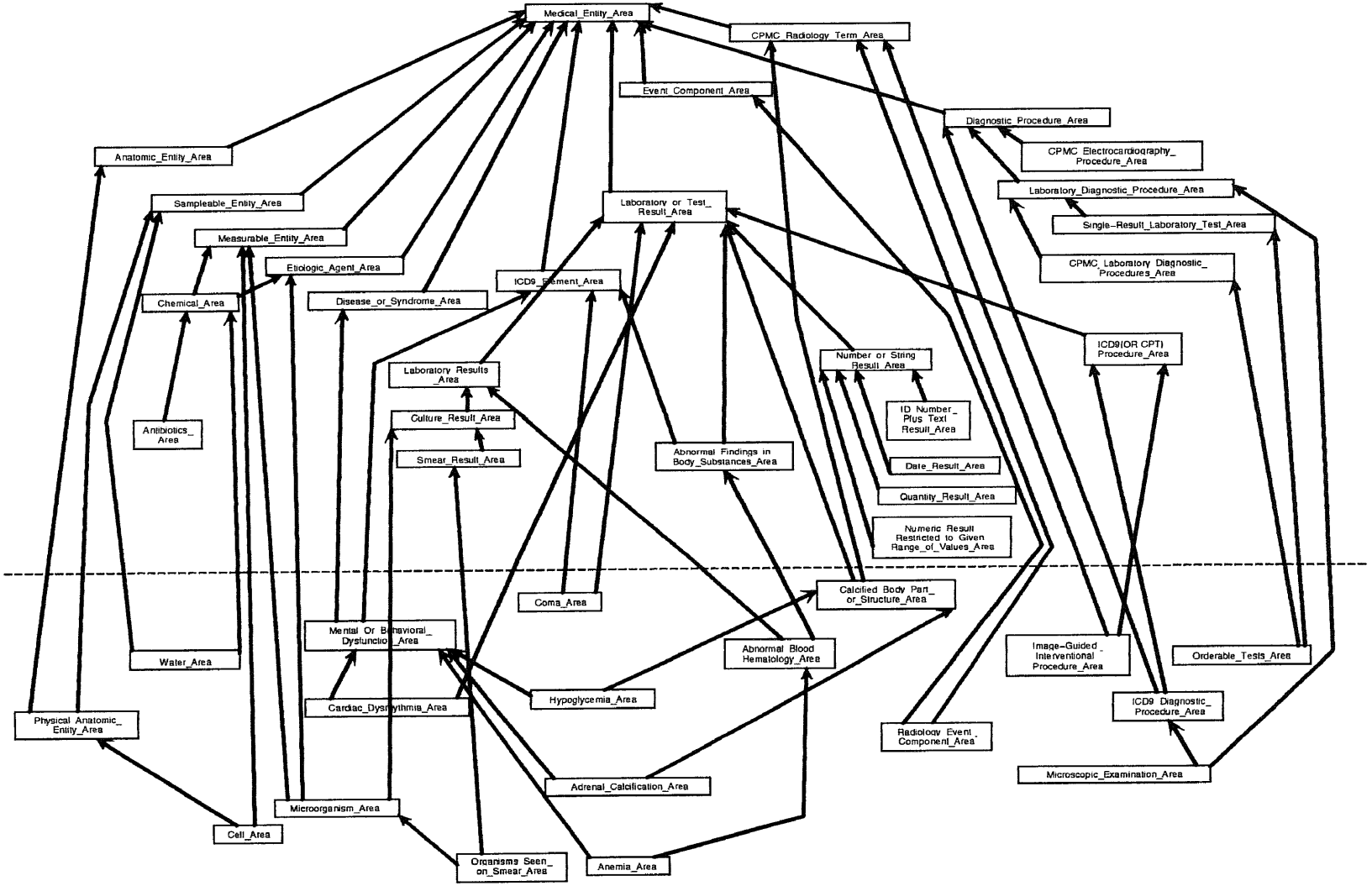

Figure 2.9 Excerpt from the OOHTR schema (see Figure 2.5 for legend of symbols)

In Figure 2.9, we show a large portion of the OOHTR schema's subclass hierarchy, with attributes and relationships omitted by applying "information thinning" [117, 59]. The figure contains about half of the property-introducing classes and the intersection classes. The area classes above the dashed line represent areas rooted at property-introducing concepts. Those below the line are intersection classes.

Let us point out that intersection concepts may lie at the juncture of three or more inheritance paths. The intersection class for such an area will be a subclass of at least three other classes. An example is *Microorganism_Area* which is a subclass of *Measurable_Substance_Area*, *Etiologic_Agent_Area*, and *Culture_Result_-Area* (Figure 2.9). It is also possible for an intersection class to be a subclass of another intersection class. For example, in Figure 2.9, *Anemia_Area* is a subclass of the intersection class *Abnormal_Blood_Hematology_Area*.

2.4 CMT Improvement Based on the OOHTR Schema View

The development of specialized views, such as network abstraction schemas, is of more than theoretical interest. The maintenance of a CMT like the MED at CPMC is a complex and difficult task. The challenges faced by maintenance personnel include updating the CMT (e.g., [122]), adding terms and relationships [138], and in general developing a change model for CMTs [113]. Furthermore, proper maintenance should include improving a CMT's organization, and uncovering and correcting inconsistencies and errors in its content. All of these require an understanding of the CMT's underlying structure. However, providing users of terminologies with comprehensible, comprehensive views remains difficult. This situation is true for terminology administrators as well as for those who would build applications or knowledge bases with respect to the CMTs.

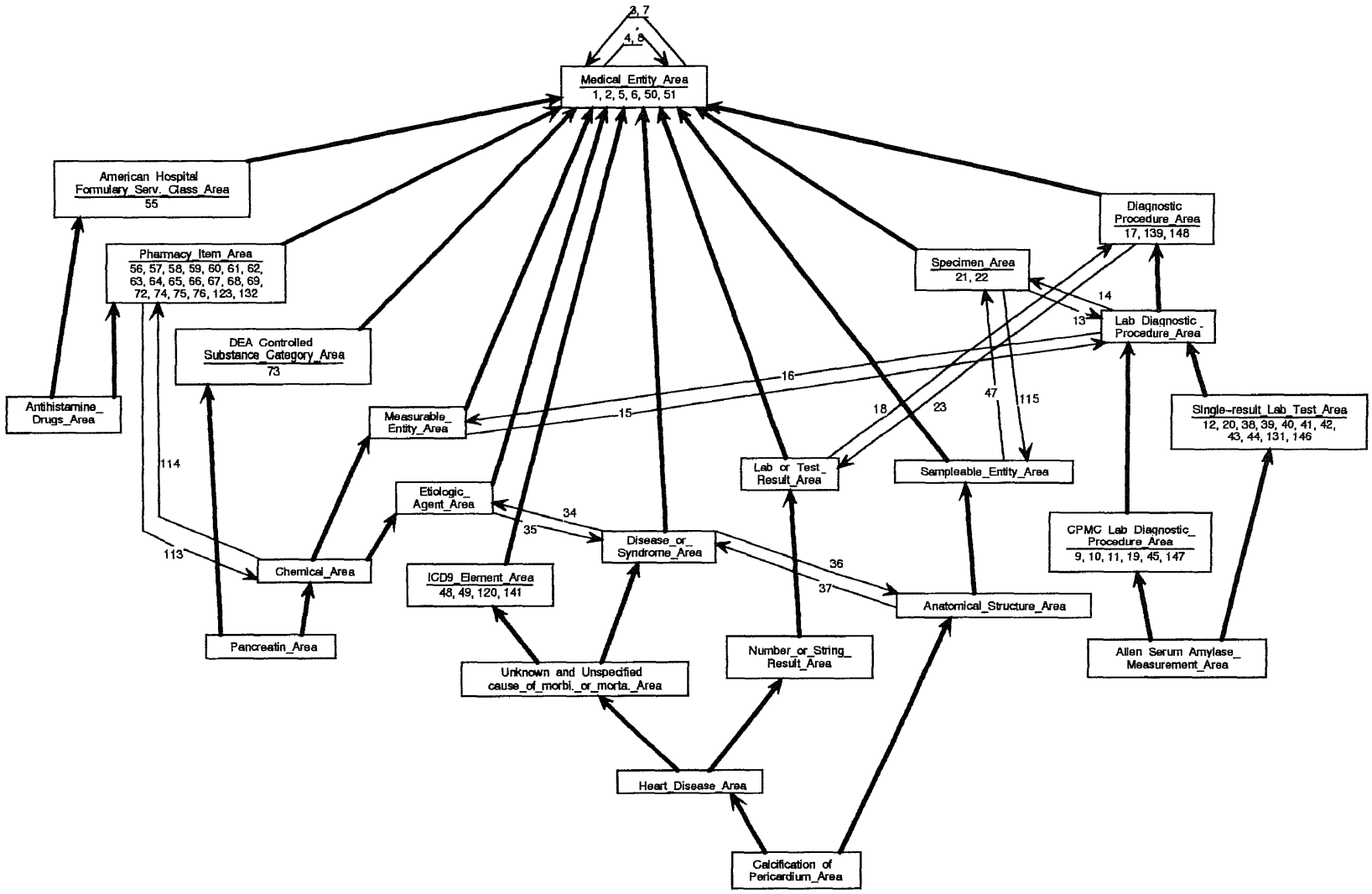

Figure 2.10 Partial OOHTR schema showing the area classes which account for Figure 2.1 (see Figure 2.5 for legend of symbols)

In our approach, an existing CMT is partitioned by using the OODB representation. This partitioning supplies an abstract view of the CMT, which helps the user understand the CMT. As we mentioned in the previous section, the OOHTR schema is very compact compared to the overall size of the MED: 90 area classes for 43,000 concepts. In Figure 2.10, we present another portion of the schema which comprises 24 area classes, corresponding to the 68 concepts of the MED shown in Figure 2.1. These 24 classes amount to 26% of the whole OOHTR schema and represent not only the 68 concepts in Figure 2.1 but also an additional 27,900 concepts or 65% of the entire MED. Compared to the complicated network in Figure 2.1 and, even more, compared to a semantic network of about 28,000 concepts, Figure 2.10 is much simpler and easier to understand; even so, it still completely and correctly captures the structure of a significant portion of the MED.

In the following subsections, we will discuss how the OOHTR schema facilitated various improvements of the MED [54] design. Specifically, we will present examples of support for updating the MED, improving its general design, and correcting errors.

2.4.1 Support for Updating the CMT

The MED comprises over 43,000 concepts with 88 different kinds of attributes, 62 different kinds of relationships (divided into 31 pairs of reciprocal relationships), 61,000 IS-A links, and 71,000 non-hierarchical links. Therefore, understanding the "big picture" of the MED is difficult. When new concepts are to be added, or when someone needs to find appropriate concepts in the MED, any lack of understanding becomes immediately apparent. The situation is often worsened because those people who maintain and use the MED may not be the same people who modeled a particular aspect originally.

Some ability to provide users with a manageable, high-level view of a CMT, like the MED, is needed to support user orientation. The OOHTR's network abstraction

schema affords such a view. By reducing the MED hierarchy about 500-fold, one can quickly see what the important areas (as represented by area classes in the schema) are and what attributes and relationships they exhibit. Someone looking to add a new concept to the MED can easily traverse these areas. During that traversal, the user can review the areas' properties to determine the appropriate area for the new concept. For example, a user faced with the task of adding a new laboratory panel to the MED can traverse the 90 class schema to find which area should contain the concept to be added; then the person can switch to traversing the concepts inside the area. Such traversal is easier and faster than a traversal of the whole terminology hierarchy of 43,000 concepts. This is analogous to commuting on the highways until reaching the vicinity of the destination and then taking an exit and continuing the ride on the local roads to the destination.

In our example (see Figure 2.6 in Section 2.2), we start in the *Medical_Entity_Area* and move to the *Diagnostic_Procedure_Area* and then to the *Lab_Diagnostic-_Procedure_Area.* This area has two children *Single_Result_Lab_Test_Area* and *CPMC_Lab_Diagnostic_Procedure_Area.* Scanning the attributes of these two candidate areas reveals that the latter has an attribute *lab-procedure-code* (encoded in Figure 2.6 as "9"). Since the concept to be entered is known by the user to have such an attribute, this area is clearly the appropriate one to choose. It has a child *Antibiotic_Sensitivity_Panel_Area* which is obviously not relevant to the new concept. We therefore switch to traversing the concepts within the *CPMC_Lab_Diagnostic_Procedure_Area* to find the proper position in the hierarchy where this new concept should be added. In this example, we need only traverse five areas to find the appropriate position for the new concept. Compared to a traversal of the CMT's hierarchy of concepts, the schema traversal is more efficient for such an update. Thus, the OOHTR schema can be seen to provide a valuable gestalt of the MED complexity, an understanding of which is needed to support updates.

2.4.2 Improving the CMT Organizational Structure

The MED content has grown steadily, averaging 500 additional concepts per month over the past ten years. Much of this growth has been the result of work by a variety of individuals, at times using automated mechanisms for adding concepts. When several people share the task of maintaining a content domain, and each has a slightly different organizational philosophy (e.g., "lumpers" versus "splitters"), it is easy for concepts to be characterized differently depending on who added them. The network abstraction schema provides a way for different people to share the same high-level view of the MED and to identify differences in their personal views. It also makes the MED's overall organization simpler to follow for all parties.

For example, the laboratory system at CPMC has concepts for individual laboratory tests (like **Serum Glucose Test**) and other concepts for orderable collections of tests (such as **CHEM-7**, a panel of 7 individual tests). These concepts are all represented in the MED with attributes appropriate to each (e.g., tests have units of measurement and normal ranges, while panels have codes used for billing). The concepts are linked to each other via relationships, e.g., **Tests** are *part-of* **Panels**, and **Tests** *measure* **Measurable Substances**. Users of the MED are often confused about the differences between tests and panels (the latter are also called "procedures" by some and "batteries" by others) [70, 78]. This confusion is exacerbated by the fact that individual tests can be ordered separately and can therefore take on the characteristics of both tests and panels.

In the schema, the tests belong to the class *Single-Result_Lab_Test_Area* and the panels are contained in the class *CPMC_Lab_Diagnostic_Procedure_Area*. The schema grouped the tests that have the properties of panels into the intersection class which is a subclass of *Single-Result_Lab_Test_Area* and *CPMC_Lab_Diagnostic_Procedure_Area* (see Figure 2.10). In the case where an intersection class has a unique root, that concept's name is chosen to name the area class. Otherwise, one of the roots is

arbitrarily chosen as the name. In our example, the intersection area is indeed multi-rooted and is named the *Allen_Serum_Amylase_Measurement_Area*. When the intersection area classes were displayed, it was realized that an implicit, natural grouping of tests with panel properties exists. The MED could be simplified by making this group explicit. However, in the MED, there was no single concept which is the parent of these particular tests. This situation was achieved by creating a new concept in the MED called **Orderable Tests** as a child of both **Single-Result Lab Test** and **CPMC Lab Diagnostic Procedure**. All the tests in *Allen_Serum_Amylase_-Measurement_Area* (such as **Allen Serum Amylase Measurement Test**) were then linked to **Orderable Tests** as its children. When the schema was redrawn (see Figure 2.11), the *Allen_Serum_Amylase_Measurement_Area* took on the new name *Orderable_Tests_Area*, since that concept was now the single root of the area. Having such an intersection class in the schema as a subclass of its parent classes *Single-Result_Lab_Test_Area* and *CPMC_Lab_Diagnostic_Procedure_Area* helps to clear up for the user the confusion about tests, panels, and orderable tests. Interestingly, soon after the **Orderable Tests** concept was added to the MED, New York State required that CPMC make explicit to its physicians and computer systems how the previously "bundled" tests could be ordered and reported individually. Thanks to the *Orderable_Tests_Area* class, the transition was relatively painless and completely transparent to CPMC's information systems.

From the above, we derived a general rule for dealing with multi-rooted intersection areas in the MED and the OOHTR schema. Instead of picking an arbitrary concept to name the area class, we create in the MED a new general parent concept to summarize all the concepts in the area. This new area root will then be used in the OOHTR to name the area class.

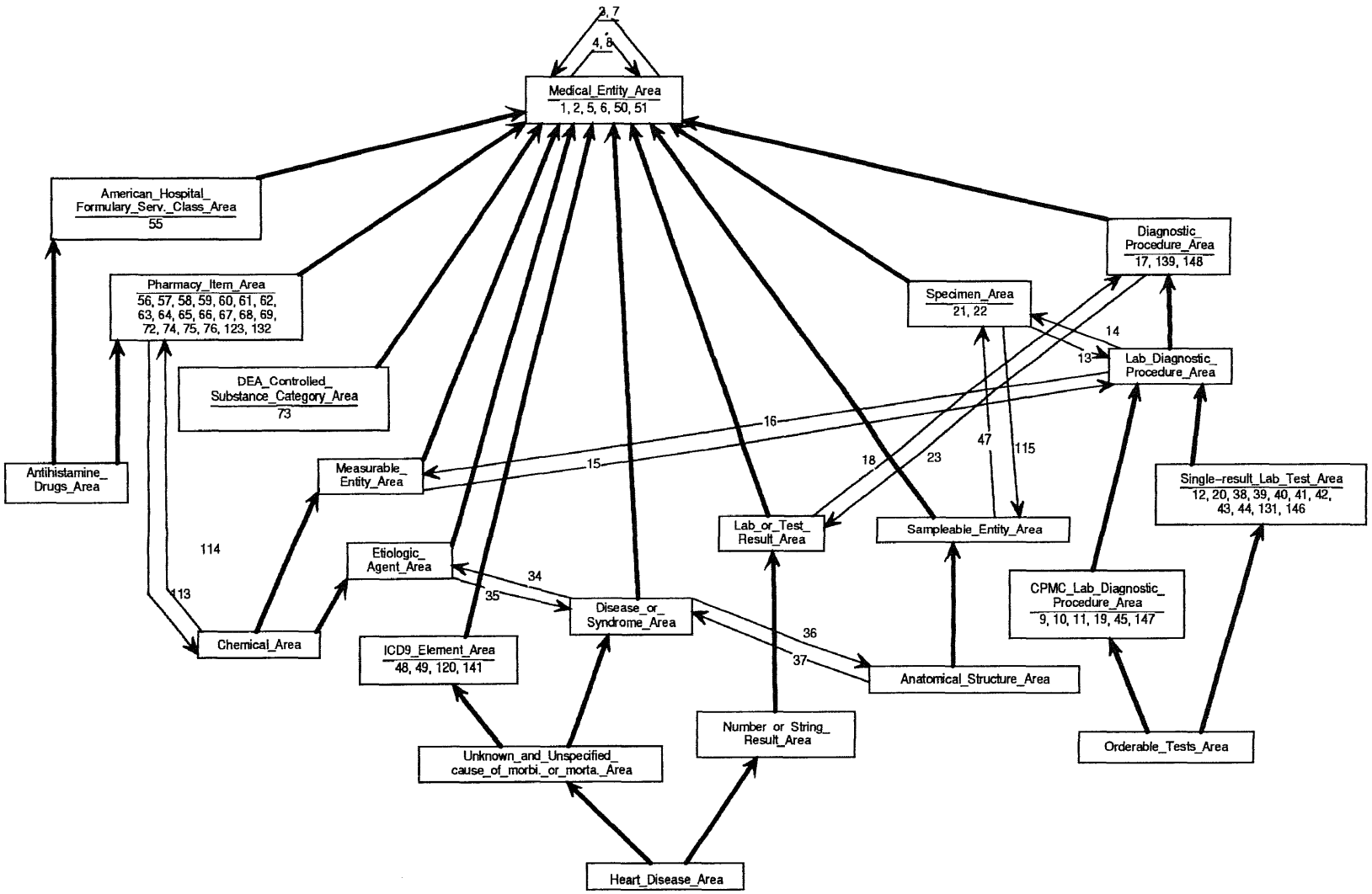

Figure 2.11 Improved version of schema from Figure 2.10 (see Figure 2.5 for legend of symbols)

2.4.3 Finding Inconsistencies and Errors in the CMT

Given the ambiguities that often occur in medical terminology, it is likely that the MED contains a concept with a name that has multiple meanings—in contradiction to the non-ambiguity condition required for the MED (see Section 2.2). Since the inception of the MED model [40], it was thought that such ambiguity could be detected through automated means. The intersection areas have provided the basis for such a method.

As an example, it can be seen from the schema in Figure 2.10 that *Calcification_of_the_Pericardium_Area* contains all concepts which are both heart diseases and anatomical structures (40 in all). Until the MED was viewed from this perspective, no one realized that the same concepts were listed as both diseases and anatomical structures! This is an example of ambiguity as the concept **Calcification of the Pericardium** (and its descendants) has one meaning as a body part and another meaning as a heart disease. This is not consistent with the original design of the MED in which a disease can be linked to body parts as the site of the disease, but can not itself be a body part. Thus, one or the other of the parent-child links had to be removed from the MED. Upon closer inspection of the *Calcification_of_the_Pericardium_Area*, we found that there are many such "Calcification of the X" concepts in the MED, all of which are included as descendants of **Calcification of Body Part**. This concept is a child of **Body Part**, and both are in the *Anatomical_Structure_Area*. **Calcification of Body Part** has 40 children (and three grandchildren) that are also classified as diseases, while other children are not. The discovery of this intersection class led directly to a study of these 40 concepts and their reclassification as either body parts or diseases, as deemed appropriate by external domain experts. So, for example, the link between **Calcification of the Pericardium** and **Heart Disease** was removed. Doing this caused **Calcification of the Pericardium** to be in a single area, namely, the *Anatomical_Structure_Area*.

Therefore, it no longer defined an intersection area of its own. When the schema was re-created (Figure 2.11), there was no longer any intersection area class that was a subclass of both *Heart_Disease_Area* and *Anatomical_Structure_Area.*

Let us look at another ambiguity example. In the schema (Figure 2.12), *Black_Piedra_Area* is an intersection area class which has two superclasses, *Smear_Result_Area* and *Wuchereria_Bancrofti_Area. Wuchereria_Bancrofti_Area* itself is an intersection area class which contains diseases caused by organisms. That means all concepts in *Black_Piedra_Area* are classified as smear results and diseases caused by organisms. After viewing the schema, the designer decided to disambiguate this situation by letting the concept **Black Piedra** refer only to the organism and not the disease caused by the organism (that disease now being called **Black Piedra Infection**). So, as an organism, **Black Piedra** is under the concept **Microorganism**. There is a class of things which are seen under the microscope on microbiology lab tests; they are called "smear results" because the specimens are "smeared" on the slides, stained, and examined. Not all smear results are organisms, and not all microorganisms (e.g., viruses) are seen on smears. Thus, the concept **Organisms Seen on Smear** was created. The concept **Black Piedra** became a child of **Organisms Seen on Smear** in the MED and sits in the intersection area class *Organisms_Seen_on_Smear_Area*, which is the subclass of *Microorganism_Area* and *Smear_Result_Area* in the schema (Figure 2.13).

The process of adding concepts to the MED was done by various experts in different fields, sometimes using automatic mechanisms to integrate concepts from a variety of sources. As a result, it is not surprising that inconsistencies and outright errors have crept into the MED. An example of an error discovered through the use of the schema was the *Pancreatin_Area* intersection area class (Figure 2.10). In the MED, it had been decided that medications (such as those classified by their DEA Controlled Substance category) would have chemicals as their pharmaceutic

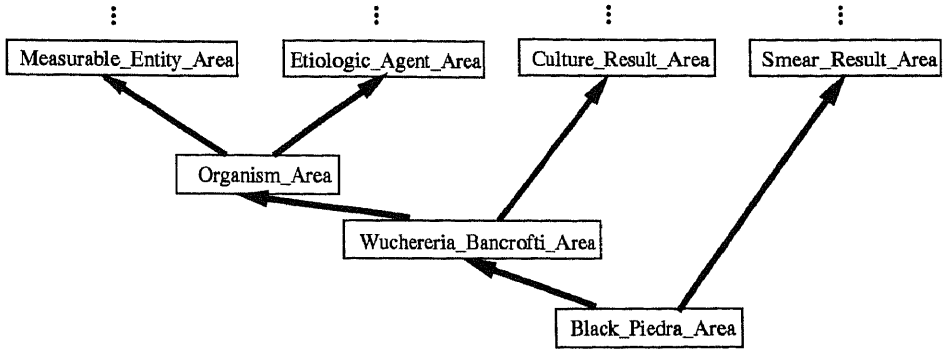

Figure 2.12 Partial OOHTR schema detecting the ambiguity of "Black Piedra" (see Figure 2.5 for legend of symbols)

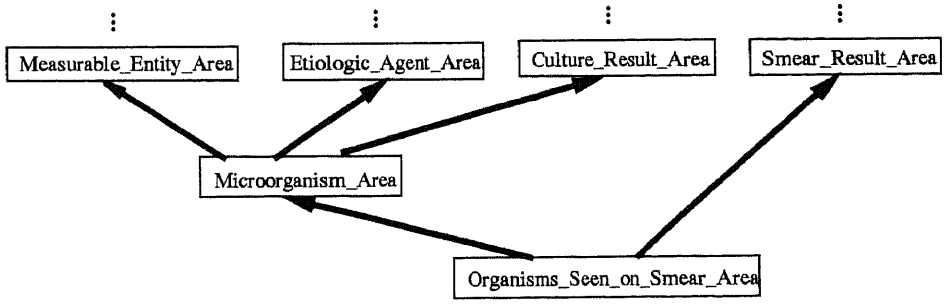

Figure 2.13 Improved version of schema from Figure 2.12 (see Figure 2.5 for legend of symbols)

components, but medications themselves would not *be* chemicals. The OOHTR schema (Figure 2.10) clearly shows that *Pancreatin_Area* violates this rule. On closer inspection, it was found that the concept **Pancreatin Preparations** was properly classified as a medication and that it was linked appropriately to the concept **Pancreatin**. However, the concept **Pancreatin** was classified not only as a chemical (which allowed it to have the *pharmaceutic-component-of* relationship to **Pancreatin Preparations**) but also as a medication (as shown in Figure 2.1). Once this error was seen, it was corrected quite easily by removing the IS-A link between **Pancreatin** and **DEA Class 0**. Since **Pancreatin** was the only concept in the MED to have attributes of both chemicals and medications, *Pancreatin_Area* had only one concept prior to the correction. After the correction, the intersection area class no longer existed since the concept **Pancreatin** was now included in *Chemical_Area* (Figure 2.11).

2.5 OODBs Versus Description Logics

A number of approaches to medical terminologies are based on the use of Description Logics. Description Logics are close descendants of KL-ONE [143, 18, 21] which is itself a descendant of Quillian's original semantic network [119]. Quite a number of KL-ONE descendants exist, which makes this probably the largest and most successful family of implemented knowledge representation systems. Two excellent overviews are [133, 84], while some of the family members are described in [17, 18, 19, 52, 106] [141, 145] [7, 8, 80, 115] [14]. Several other semantic networks that do not belong to the family of Description Logics exist, such as those described in [126, 142].

Our own choice of using Object-Oriented Databases instead of Description Logics is based on two kinds of reasons, which may be summarized under the labels of *abstraction ability* and *commercial viability*. Concerning *abstraction ability*, databases naturally come with two layers of representation, the schema layer and the

data layer. As was explained in Section 2.3.1, the schema is by orders of magnitude smaller than the data. This makes the schema a valuable orientation aid to users, maintainers, and newly hired or trained developers of a medical terminology.

In traditional database applications, the OODB schema is developed first, and then the database is populated by instances of the schema classes. It is the strength of our approach that we are deriving a schema *after the fact* from terminology data that already exists. Thus, we are supplying a road map for information that was not designed with a schema in mind, and which is therefore naturally harder to understand. We note that the abstraction supplied by a schema is different in nature from the abstraction supplied by the top level classes of a Description Logic network, because the schema registers "significant structural changes" in the terminology, no matter at what level they occur. This is not the case for Description Logic networks, because looking at the top levels gives just that, the top levels. Furthermore, intersections of top level concepts at lower levels are not reflected at the top levels, while they are reflected in our schema approach.

The *commercial viability* of OODBs seems to be better than that of Description Logics. There are a number of vendors that deliver "full service Object-Oriented Database systems." Those systems incorporate basic database features such as persistence and multiuser access. Some of them go far beyond these features, including, e.g., versioning and schema evolution. Documentation and help lines are standard for most of these systems. We mention as the most widely advertised products ORION/ITASCA, GemStone, ONTOS, ObjectStore, VERSANT, Jasmine and O_2. While Description Logics have recently become more available as commercial products, the balance still tilts towards OODB systems. We mention tools for the maintenance of Description Logic-based medical terminologies from Lexical Technologies[2] and Ontyx.[3] Some Description Logics, e.g., K-Rep [93], have been

[2]http://www.lexical.com
[3]http://www.ontyx.com

extended to include persistence and other OODB features. There are also freely available prototypes, both of OODBs (e.g., ODE [4]) and Description Logics (e.g. LOOM[4] [53]).

Naturally, the Description Logic approaches are superior to OODBs in what we might label *reasoning-based support*. They make better use of inheritance than OODBs, as their notion of inheritance is based on structure and values, while OODB inheritance is purely structural. The classification algorithm of Description Logics is an outstanding achievement which has not been duplicated in OODBs. However, a reasoning layer can be added on top of an OODB representation.

However, the nature of Description Logics themselves imposes severe limitations on their abilities in those areas that are considered their strengths. Specifically, as Levesque and Brachman discussed in their fundamental paper [20] on the tradeoff between representation and reasoning, "...subsumption of descriptions in FL is intractable,...." The only way to make the appropriate algorithms computable in polynomial time is to severely limit the power of the representation language. Secondly, Description Logics thrive in areas where the Aristotelian view of categorization, by necessary and sufficient conditions, is most applicable. When the number of natural kinds ("primitive concepts" in KL-ONE [147]), which cannot be defined that way, increases then classification algorithms lose some of their usefulness. In an area like medicine, many terms are highly subjective ("pain" comes to mind), and therefore cannot be defined by necessary and sufficient conditions.

Even in the face of these limitations, Description Logics are valuable and interesting experimental, scientific and commercial vehicles. Thus, we consider OODBs for medical terminologies as complementary to Description Logics.

[4]http://www.isi.edu/isd/LOOM

2.6 Summary

The job of maintaining a controlled medical terminology (CMT) can be daunting due to the typical CMT's large size and extensive scope. Among the tasks that need to be performed by maintenance personnel are updating the CMT with new concepts, re-organizing its design to enhance usability, and correcting mistakes which can arise from various sources. In order to handle these chores, a person must have a solid understanding of the overall structure of the CMT and its content.

Toward that end, we have proposed the use of the object-oriented database (OODB) paradigm for the representation of CMTs. We have introduced the notion of an Object-Oriented Healthcare Terminology Repository (OOHTR): A CMT represented in the form of an OODB. An OOHTR is derived from an underlying CMT through a partitioning process based on the pattern in which properties are introduced and distributed among the CMT's constituent concepts. In that context, we defined the notions of *property-introducing concept* and *intersection concept.* From these emerged the basic unit of the partitioning process called an *area*, a collection of concepts that have the same set of properties.

The partitioning process yields an OODB schema that captures the CMT's overall structure. The benefit of the schema is that it provides an extra level of abstraction and summarization for the CMT. After applying our methodology to the MED to produce an OOHTR, we demonstrated how the view afforded by the schema facilitated a variety of improvements. In general, the OOHTR schema can serve as an important mechanism for enhancing comprehension of a large CMT by users and maintainers alike.

CHAPTER 3

REPRESENTING THE UMLS AS AN OODB: MODELING ISSUES AND ADVANTAGES

3.1 Introduction

The Unified Medical Language System (UMLS) [71, 73, 89, 140] designed by the National Library of Medicine (NLM) combines many well established medical informatics terminologies in a unified knowledge representation system. It consists of four Knowledge Sources (the Metathesaurus, the Semantic Network, the Specialist Lexicon, and the Information Sources Map) that provide information about medical terminologies. The UMLS can be used by a wide variety of application programs to overcome the retrieval problems caused by differences in the way the same medical concept is expressed in different sources [72]. Such a resource is very valuable to medical researchers and the healthcare industry.

However, the UMLS is large and complex. The scope and complexity of the UMLS pose serious comprehension problems for users and even developers. The magnitude of presented knowledge is overwhelming for human comprehension capabilities. It is difficult to maintain and use the UMLS without proper comprehension. Designers, maintainers and users of the UMLS need tools to help with their work. There are tools for retrieval and manipulation of the content of the UMLS [27, 122, 125, 135, 138, 139]. However, such tools are insufficient. Rather, tools should also help professionals reach a level of *comprehension* essential to performing their tasks.

In previous chapter and [90, 91], we have developed a methodology for representing Controlled Medical Terminologies (CMTs) [33, 34] as Object-Oriented Databases (OODBs) to provide support for comprehending them. The comprehension support was achieved by the two layers of the OODB representation of a CMT, the schema layer and the concept layer. The schema layer gives an abstract

view of the source CMT, which aids in the comprehension of the structure and content of a large and complex CMT. At the concept layer, users can directly access objects which denote concepts of the original CMT and obtain any detailed medical knowledge they require.

In this chapter, we utilize an OODB representation to capture the knowledge of the two major components of the UMLS, the Metathesaurus and the Semantic Network, in a simplified and homogeneous way. The Metathesaurus is the largest and most complex of the Knowledge Sources. It is a compilation of terms, concepts, relationships, and associated information drawn from over 40 medical terminologies and classifications. In the 1998 release of the Metathesaurus, there are 1,051,901 term names mapped into 476,313 concepts. The Semantic Network contains information about types or categories (e.g., **Disease or Syndrome**, **Virus**) and the permissible relationships among these types (e.g., **Virus** "*causes*" **Disease or Syndrome**) [95, 96, 97]. Each concept in the Metathesaurus is assigned to one or more semantic types from the Semantic Network. The 1998 release of the Semantic Network contains 132 semantic types and 53 relationships.

To model the Metathesaurus and the Semantic Network as an OODB, we represent all semantic types in the Semantic Network as classes in the OODB schema. In the chapter, we will discuss why this straightforward approach to modeling the UMLS is insufficient. We introduce a more sophisticated approach. All concepts assigned to only one semantic type become instances of the corresponding class. Each concept assigned to multiple semantic types becomes an instance of a new kind of class, called intersection class. As a result of this modeling, all classes abstract semantically uniform sets of concepts. In this chapter, we also describe a rule to systematically define subclass relationships for all intersection classes. The resulting UMLS OODB schema has a deeper and more refined structure than the Semantic Network of the UMLS. We will explain why this is a modeling improvement which

is completely in line with the design goals of the UMLS [98]. The UMLS OODB schema also supports the improved comprehension and navigation of the Metathesaurus. Furthermore, the intersection classes expose some problems existing in the current UMLS, such as concept omissions, classification errors, and ambiguities of concepts.

The rest of this chapter is organized as follows. Sections 3.2 describes the derivation of the classes of the UMLS OODB schema. Section 3.3 presents the rule to specify the subclass relationships between classes. Benefits of the OODB representation of the UMLS are described in Section 3.4. Section 3.5 contains the summary.

3.2 OODB Class Representation of the Semantic Types

The Semantic Network and the Metathesaurus are two components of the UMLS, the connection between which is described in [98] as follows: "The Semantic Network encompasses and provides a unifying structure for the Metathesaurus constituent vocabularies." An OODB system also consists of two layers, the schema layer, describing the structure of the data and the instance layer, containing the data itself organized as objects with properties. This analogy suggests the use of an OODB to model the Semantic Network and the Metathesaurus. This modeling will unify the two components into one system which will offer several natural advantages for the UMLS. In this section, we will describe the process which derives the classes of the OODB.

3.2.1 The Semantic Type Classes

As previously noted, the Metathesaurus and the Semantic Network of the UMLS are related by associating each concept of the Metathesaurus with one or more semantic types. The Semantic Network provides a high level abstract view of the Metathe-

saurus. In general, a class in an OODB schema represents a group of objects (or instances) which exhibit the same properties and have a common semantics. The OODB schema gives an abstract view of a database. In order to model the UMLS as an OODB, it is sensible to represent the semantic types as classes in the OODB schema, and the concepts as instances of those classes. In other words, the Semantic Network, including its relationships (hierarchical and non-hierarchical), will serve as (a major part of) the OODB schema. In the next subsection, we will describe the remaining part of the classes.

The Semantic Network of the UMLS contains 132 semantic types which are arranged in a IS-A hierarchy. **Entity** and **Event** are two roots of the hierarchy. Figure 3.1 shows a few semantic types of the Semantic Network. In the modeling process, every semantic type in the Semantic Network is mapped into a class of the OODB schema. Names of classes are preserved. That is, the name of the class in the OODB schema is identical to the name of the corresponding semantic type in the Semantic Network. This kind of a class is called a *semantic type class.* Every IS-A link in the Semantic Network is mapped into a *subclass* relationship in the OODB schema. E.g., **Substance** IS-A **Physical Object** and **Physical Object** IS-A **Entity** in the Semantic Network are mapped as follows. In the OODB schema, "Substance," "Physical Object," and "Entity" are three semantic type classes. "Substance" is a *subclass* of "Physical Object" and "Physical Object" is a *subclass* of "Entity."

After we map all semantic types into the OODB schema, we obtain an OODB schema with two root classes "Entity" and "Event" since the hierarchy of Semantic Network concepts contains two roots. For traversal purposes, we assume a hierarchy to be singly rooted. Thus, we need to introduce an artificial root into the schema. A new class called "Thing" is added into the schema. The root classes mentioned above become the subclasses of "Thing." At this point, an OODB schema with 133

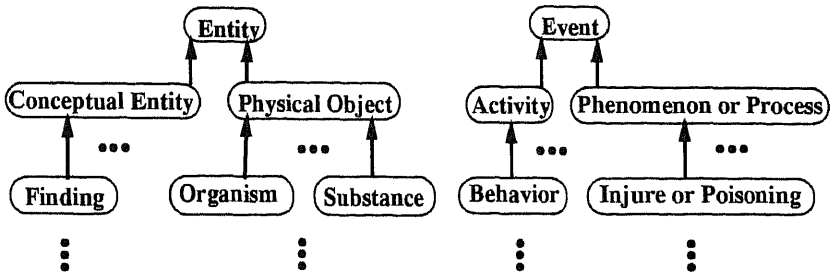

Figure 3.1 Extract of the Semantic Network. A semantic type is represented by a rounded-corner rectangle with its name written inside; an IS-A link is a bold arrow directed from a semantic type to a parent semantic type.

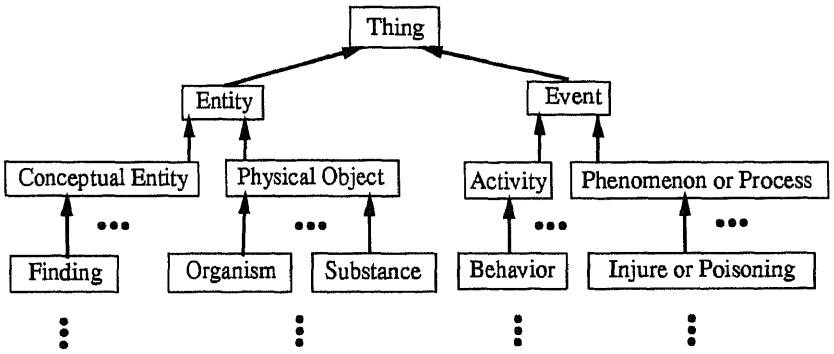

Figure 3.2 A subschema of the OODB schema corresponding to Figure 3.1. A class is represented as a rectangle and a *subclass* relationship is drawn as a bold arrow directed upward from the subclass to the superclass.

semantic type classes corresponding to all semantic types in the Semantic Network has been created. Figure 3.2 is a partial schema. Since the concept hierarchy of the Semantic Network consists of two disjoint trees, the corresponding OODB schema is also a tree.

Now, we need to assign the concepts of the Metathesaurus to classes. As mentioned before, each concept is assigned to at least one semantic type. If a concept is assigned to *only* one semantic type, we can immediately make it an instance of the corresponding semantic type class. For instance, the concept **Air** is assigned to the semantic type **Substance** and the concept **Chimera** is assigned to the semantic type **Organism**. After mapping, **Air** becomes an instance of the class "Substance" and **Chimera** becomes an instance of the class "Organism." In this way, 357,804

concepts in the Metathesaurus which are assigned to only one semantic type can be immediately represented as instances of the corresponding semantic type classes in the OODB schema.

However, concepts may belong to more than one semantic type. E.g., the concept **Cotton** belongs to two semantic types **Substance** and **Plant**; the concept **Norepinephrine preparation** belongs to four semantic types **Organism, Pharmacologic Substance, Neuroreactive Substance or Biogenic Amine,** and **Hormone**. Of the 476,314 concepts in the Metathesaurus (1998 release), 118,510 are assigned to two or more semantic types. For more details on the concept distribution, see Table 3.1.

Due to the possibility of additional semantic types, the concepts of one semantic type may be non-uniform. For example, the semantic type **Experimental Model of Disease** has 39 assigned concepts (see Table 3.2). Besides this semantic type, the concept **Radiation Injuries, Experimental** has one additional semantic type **Injury or Poisoning**. The concept **Water Deprivation** has one additional semantic type **Diagnostic Procedure**. Another 27 concepts have one additional semantic type, **Neoplastic Process**. The concept **Lesion, NOS** has two additional semantic types **Functional Concept** and **Sign or Symptom**. Only 9 concepts belong exclusively to the semantic type **Experimental Model of Disease**. It is difficult to comprehend and use the information contained in such a non-uniform semantic type. The problem we face is how to represent concepts with multiple semantic types in the OODB system.

3.2.2 The Intersection Classes

Following the above approach, a concept which is assigned to more than one semantic type should be represented as an instance of more than one class in the OODB

Table 3.1 The distribution of concepts in the Semantic Network

Number of assigned semantic types	Number of concepts
1	357,804
2	108,905
3	9,262
4	331
5	10
6	2

schema. However, in OODBs all instances of a class must have the same structure and the same semantics.

In the UMLS using OODB, the semantics of a concept, describing its meaning, is provided by its semantic types. If a concept may belongs to only one semantic type, then it has a *simple semantics.* Otherwise, if a concept belongs to a set of semantic types, it has a *compound semantics* defined by the combination of its different semantic types. Thus, looking at the example we gave before, the concepts of the semantic type **Experimental Model of Disease** do not share the same semantics. E.g., the concept **Alloxan Diabetes** has the simple semantics of "Experimental Model of Disease" and the concept **Radiation Injuries, Experimental** has the compound semantics of "Experimental Model of Disease ∩ Injury or Poisoning." The symbol "∩" indicates the intersection, meaning that the concept **Radiation Injuries, Experimental** is both an "Experimental Model of Disease" and "Injury or Poisoning." In Figure 3.3, we show all intersections among six semantic types, **Experimental Model of Disease** and five other semantic types which it intersects with. Each intersection contains concepts which belong to two or more semantic types. From the figure, we see that all 39 concepts of **Experimental Model of Disease** are classified into five groups with different semantics. The concepts of

Table 3.2 All concepts assigned to the semantic type Experimental Model of Disease

Experimental Model of Disease
Alloxan Diabetes
Arthritis, Adjuvant
Avian Leukosis
Carcinoma 256, Walker
Carcinoma, Ehrlich Tumor
Carcinoma, Krebs 2
Carcinoma, Lewis Lung
Diabetes Mellitus, Experimental
Disease Models, Animal
Encephalomyelitis, Allergic
Hepatoma, Experimental
Hepatoma, Morris
Hepatoma, Novikoff
Lesion, NOS
Leukemia, Experimental
Leukemia L1210
Leukemia L5178
Leukemia P388
Liver Cirrhosis, Experimental
Liver Neoplasms, Experimental
Mammary Neoplasms, Experimental
Melanoma, B16
Melanoma, Cloudman S91
Melanoma, Experimental
Melanoma, Harding-Passey
Murine Acquired Immunodeficiency Syndrome
Neuritis, Experimental Allergic
Radiation Injuries, Experimental
Sarcoma 180
Sarcoma 37
Sarcoma, Avian
Sarcoma, Engelbreth-Holm-Swarm
Sarcoma, Experimental
Sarcoma, Jensen
Sarcoma, Rous
Sarcoma, Yoshida
Streptozotocin Diabetes
Tumor Virus Infections
Water Deprivation

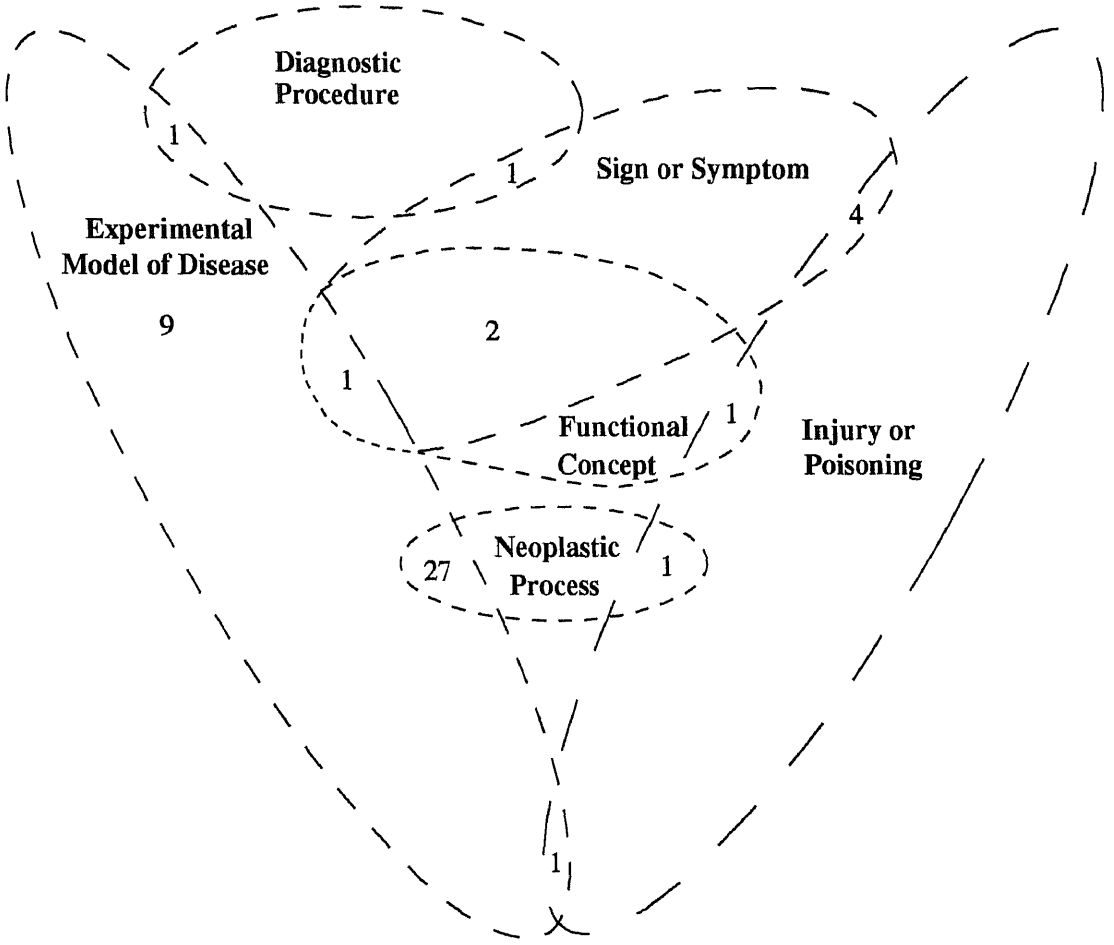

Figure 3.3 Semantic types and intersections among them

one group have a simple semantics, while the four other groups express compound semantics.

Thus, we cannot assign concepts of a given semantic type but with different semantics to the same class, corresponding to this semantic type. We need to differentiate these concepts and represent them as instances of different classes in the schema. Each of these classes needs have a uniform semantics.

In OODBs, each object must be an instance of one and only one class. Making concepts instances of more than one class violates this restriction. This OODB restriction follows from the need of semantic uniformity explained above. Thus, each concept, even if it has multiple semantic types, can only be represented as an instance

of one class. The current "semantic type classes" corresponding to the semantic types are therefore not sufficient for representing all concepts. An additional kind of classes is needed.

In this chapter, we call the set of instances of a class C the extent $E(C)$, the set of concepts of a semantic type S the extent $E(S)$, and the set of all concepts of the Metathesaurus M the extent $E(M)$. To keep the concepts corresponding to a semantic type class uniform, we remove all concepts belonging to several semantic types from the extent of all semantic type classes. Hence, each semantic type class will correspond to concepts belonging only to this semantic type. In order to fulfill the above goal of representing all concepts as instances of classes of uniform semantics, a new kind of class, called an *intersection class*, is introduced into our schema. Such a kind of class represents the combination of two or more semantic types.

To restate, the purpose of introducing intersection classes is to accommodate the concepts which are assigned to more than one semantic type. For this reason, the creation of intersection classes is based on those concepts. Every concept that belongs to more than one semantic type will be represented as an instance of one intersection class. In order to create intersection classes, all concepts with multiple semantic types are partitioned into groups such that each group contains the concepts belonging to the same set of semantic types. That means, the concepts in one group are uniform and have the same compound semantics. After we obtain the groups from the partitioning process, the corresponding intersection classes are created to represent all those concept groups. Furthermore, the concepts in each group become the instances of the corresponding intersection class. Table 3.3 lists the concepts belonging to the semantic type **Experimental Model of Disease**, partitioned into classes with uniform semantics. Table 3.3 represents a refined classification of the concepts of the original semantic type, listed in Table 3.2.

Table 3.3 Partitioning result of table 3.2

Experimental Model of Disease
Alloxan Diabetes
Arthritis, Adjuvant
Diabetes Mellitus, Experimental
Disease Models, Animal
Encephalomyelitis, Allergic
Liver Cirrhosis, Experimental
Neuritis, Experimental Allergic
Streptozotocin Diabetes
Murine Acquired Immunodeficiency Syndrome

Experimental Model of Disease ∩ Injury or Poisoning
Radiation Injuries, Experimental

Experimental Model of Disease ∩ Diagnostic Procedure
Water Deprivation

Experimental Model of Disease ∩ Neoplastic Process
Avian Leukosis
Carcinoma 256, Walker
Carcinoma, Ehrlich Tumor
Carcinoma, Krebs 2
Carcinoma, Lewis Lung
Hepatoma, Experimental
Hepatoma, Morris
Hepatoma, Novikoff
Leukemia, Experimental
Leukemia L1210
Leukemia L5178
Leukemia P388
Liver Neoplasms, Experimental
Mammary Neoplasms, Experimental
Melanoma, B16
Melanoma, Cloudman S91
Melanoma, Experimental
Melanoma, Harding-Passey
Sarcoma 180
Sarcoma 37
Sarcoma, Avian
Sarcoma, Engelbreth-Holm-Swarm
Sarcoma, Experimental
Sarcoma, Jensen
Sarcoma, Rous
Sarcoma, Yoshida
Tumor Virus Infections

Experimental Model of Disease ∩ Functional Concept ∩ Sign or Symptom
Lesion, NOS

In Figure 3.3, we show six semantic types and nine intersections among them. In our modeling process, all concepts residing in the intersections are removed from the extents of the original semantic type classes. All six original semantic types are represented as six semantic type classes "Experimental Model of Disease," "Neoplastic Process," "Injury or Poisoning," "Diagnostic Procedure," "Functional Concept," and "Sign or Symptom." Each concept belonging to only one of these six semantic types is represented as an instance of the corresponding semantic type class. Nine intersection classes are created to represent the nine intersections in Figure 3.3. They are "Experimental Model of Disease ∩ Diagnostic Procedure," "Experimental Model of Disease ∩ Neoplastic Process," "Experimental Model of Disease ∩ Injury or Poisoning," "Experimental Model of Disease ∩ Sign or Symptom ∩ Functional Concept," "Diagnostic Procedure ∩ Sign or Symptom," "Sign or Symptom ∩ Functional Concept," "Injury or Poisoning ∩ Sign or Symptom," "Injury or Poisoning ∩ Functional Concept," and "Neoplastic Process ∩ Injury or Poisoning." All concepts residing in the intersections of Figure 3.3 become the instances of the corresponding intersection classes.

After the creation of the intersection classes, all 476,314 concepts in the Metathesaurus are represented, each as an instance of one class in the schema. The whole schema consists of 1,296 classes. Among them, 1,163 are intersection classes. It may seem that with the intersection classes we lose the access to the extents of the original semantic types. However, in the next section, we will show that this information can be reconstructed upon demand from the OODB schema.

3.3 The Subclass Relationships in the UMLS OODB Schema

3.3.1 Straightforward Model: One Level of Intersection Classes

After introducing the intersection classes, we face the problem how to determine the *subclass* relationships originating with an intersection class in the schema. In other words, we need to decide what the superclasses of an intersection class are.

As we described previously, an intersection class represents the combination of more than one semantic type. Its semantics is more specific than that of each original intersected semantic type class. Now we need to decide what the superclasses of each intersection class are. One feasible approach is to use all its original intersected semantic type classes. We call this approach the "straightforward model." Thus, an intersection class is one level lower than its intersected semantic type classes in the initial schema. In this approach, there are no intersection classes which are superclasses of other intersection classes. The extended schema has only one more level than the initial schema. For example, in Figure 3.4 the intersection class "Sign or Symptom ∩ Functional Concept" has two superclasses "Sign or Symptom" and "Functional Concept." The intersection class "Experimental Model of Disease ∩ Sign or Symptom ∩ Functional Concept" has three superclasses "Experimental Model of Disease," "Sign or Symptom," and "Functional Concept." For Figure 3.4 one extra level of intersection classes and 19 additional *subclass* relationships are added to the original schema. Table 3.4 shows the distribution of classes in each level of the UMLS schema which is the result of the above approach. Table 3.5 lists the the number of the superclasses of all classes including both semantic type classes and intersection classes of the UMLS schema.

The initial schema of semantic type classes was a tree with a depth of 9. Tables 3.4 and 3.5 show that 1,163 intersection classes and 2,874 subclass relationships are added to the initial schema, resulting in a 10-level DAG schema. The designers of the UMLS considered desirable to increase the depth of the Semantic

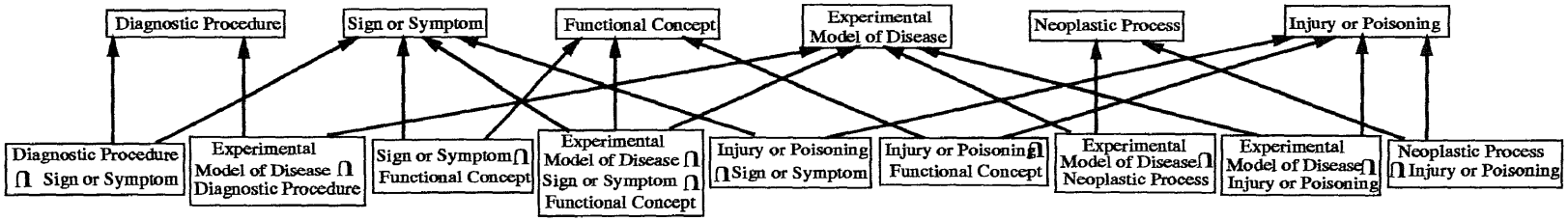

Figure 3.4 One feasible solution of adding *subclass* relationships

Table 3.4 The distribution of classes in each level of the UMLS schema in the straightforward model

Level	Number of classes	Number of intersection classes
1	1	0
2	2	0
3	4	0
4	20	0
5	41	62
6	23	218
7	23	172
8	17	240
9	2	401
10	0	70

Table 3.5 Number of superclasses for all classes in the UMLS schema in the straightforward model

Number of superclasses of a class	Number of classes
0	1
1	132
2	714
3	358
4	84
5	6
6	1

Network of the UMLS [98]. Thus, even the straightforward UMLS OODB schema represents a modeling improvement over the Semantic Network.

3.3.2 A Refined Model: Intersection Classes of Intersection Classes

In OODBs the *subclass* relationships are pointing from specific classes to general classes. By transitivity, every specific class is implicitly a subclass of all ancestors of its superclasses. (By ancestors we refer to classes reachable following a chain of superclass relationships.) Because of that, we do not need explicit subclass relationships to ancestors of the superclasses. For example, in Figure 3.4, we see intersection classes "Experimental Model of Disease ∩ Sign or Symptom ∩ Functional Concept," which is a subclass of three classes "Experimental Model of Disease," "Sign or Symptom" and "Functional Concept." The class "Sign or Symptom ∩ Functional Concept" is a subclass of "Sign or Symptom" and "Functional Concept." If we compare these two intersection classes, we see that the semantics of "Experimental Model of Disease ∩ Sign or Symptom ∩ Functional Concept" is more specific than the semantics of "Sign or Symptom ∩ Functional Concept." Hence, it is natural to have a subclass relationship from the more specific intersection class to the more general intersection class. In Figure 3.4, "Experimental Model of Disease ∩ Sign or Symptom ∩ Functional Concept" should become a child of "Sign or Symptom ∩ Functional Concept" (Figure 3.5).

We will now explain why the resulting modeling is correct. Since "Sign or Symptom ∩ Functional Concept" is a subclass of "Sign or Symptom" and "Functional Concept," the transitivity implies that "Experimental Model of Disease ∩ Sign or Symptom ∩ Functional Concept" is a subclass of both "Sign or Symptom" and "Functional Concept." Thus, there is no need to have an explicit subclass relationships from "Experimental Model of Disease ∩ Sign or Symptom ∩ Functional Concept" to "Sign or Symptom" and "Functional Concept" as in Figure 3.4.

Figure 3.5 shows the alternative modeling. In view of this example, we will discuss an alternative approach for defining subclass relationships for the intersection classes.

The refined model is designed to capture semantic relationships between intersection classes which were not reflected in the straightforward model. We make an intersection class a subclass of another intersection class. As a result, intersection classes appear in multiple levels. We do not want a class to have an unnecessary subclass relationship to a more general class if this relationship is implied by transitivity. For a class which is an intersection of two, it is necessary to make it a subclass of those two classes. For instance, in Figure 3.5, the intersection class "Sign or Symptom ∩ Functional Concept" is a subclass of "Sign or Symptom" and "Functional Concept." However, for the intersection of more than two, there may be more than one alternative to define the *subclass* relationships. In such a case, some subclass relationships which are unnecessary due to transitivity may be eliminated. In order to systematically define the *subclass* relationships of intersection classes, we need a rule to determine the superclasses of an intersection class.

Before we describe such a rule, we first need to give the definitions of the maximal subsets of a set and the minimal superclasses of a class.

Let U be a universal set of elements and let F be a given family of sets over U. (By family, we mean it is a set of sets). That is F is a subset of the power set of U ($F \subset 2^U$).

In the context of the UMLS, the universal set U is the set of all concepts of the Metathesaurus, and the universal family F is the family of the extents of all semantic types. ($F = \{E(S_1), E(S_2), \ldots, E(S_n)\}$; n is the number of semantic types.) For a given family of sets G ($G = \{E(S_{i_1}), E(S_{i_2}), \ldots, E(S_{i_k})\}(k < n)$) which is a subfamily of F, the *family intersection* I_G is the intersection of all extents in G. ($I_G = \bigcap_{1 \leq j \leq k} E(S_{i_j})$.)

In our OODB modeling of the UMLS a semantic type class C_{S_i} corresponds to the corresponding semantic type S_i. An intersection class corresponds to the intersection of several semantic types. (E.g., C_{I_G} corresponds to I_G.) When an intersection class is given, it is possible to identify all its *potential superclasses* for which there may exist an implied subclass relationship. E.g., for intersection class C_{I_G}, each of the semantic type classes $C_{S_{i_j}} (1 \leq j \leq k < n)$ is a potential superclass of C_{I_G}. Furthermore, for each D such that D is a subfamily of G, the intersection class C_{I_D} is a potential superclass of C_{I_G}.

Definition 1 (Maximal Subset): Let A and B be sets in F, such that A is a subset of B. If there does not exist any set C in F such that A is a proper subset C and C is a proper subset of B, then we call A a *maximal subset* of B in F. (E.g., if $\{X, Y, Z\}$, $\{X, Y\}$, and $\{X\}$ are three sets in F, $\{X, Y\}$ is a maximal subset of $\{X, Y, Z\}$ and $\{X\}$ is not.)

Now we define the notion of minimal superclass of an intersection class, corresponding to the above definition of maximal subset of a set.

Definition 2 (Minimal Superclass): Let C_{I_G} be an intersection class corresponding to the family intersection I_G. If C_{I_D} is a potential superclass of C_{I_G}, then C_{I_D} is a minimal superclass of C_{I_G} if D is a maximal subset of G.

Intuitively, a minimal superclass of a class C is a superclass that is most similar to C. As such it maximized the number of classes to which C does not need a direct link.

Note that as a special case, D may be a family of the extent of only one semantic type, say S_i. In this case, C_{I_D} degenerates to a semantic type class C_{S_i} rather than an intersection class. We have chosen simplify the above definition by not explicitly considering this special case. Assuming that $C_{I_D} = C_{S_i}$ is a degenerate intersection class, our definition still works.

Subclass Definition Rule: Let C_I be an intersection class in the schema. Then subclass relationships are defined from C_I to all its minimal superclasses in the schema.

This rule is guaranteed to increase the depth of the schema by transforming intersection classes of more than two semantic type classes into subclasses of other intersection classes. As McCray [98] notes, it is considered desirable to increase the depth of the Semantic Network. For example (see Figure 3.5), the classes "Sign or Symptom ∩ Functional Concept" and "Experimental Model of Disease" are the only two minimal superclasses of the intersection class "Experimental Model of Disease ∩ Sign or Symptom ∩ Functional Concept." Compared with Figure 3.4, with 19 subclass relationships, Figure 3.5 contains only 18 subclass relationships. In [58, 59, 117], we defined the complexity of a schema as the ratio between the number of relationships and the number of classes of the schema. Thus, when two schemas contain the same number of classes, the one with more relationships will be of higher complexity. Hence, the schema in Figure 3.5 is simpler than the one in Figure 3.4. Furthermore, the schema is more accurate, semantically, as it captures subclass relationships between intersection classes.

Unfortunately, there is no guarantee that the subclass definition rule will always result in a schema of lower complexity. It may result in a schema of higher complexity. For instance, if we assume that there is one intersection class which is an intersection of all six semantic type classes in Figure 3.4, six more subclass relationships are added, yielding a total of 25 subclass relationships when using the straightforward modeling approach. However, eight more subclass relationships are added, yielding 26 subclass relationships, if we use the refined modeling approach! Nevertheless, we shall see that in the UMLS schema obtained, the first situation occurs more often than the second and the total number of subclass relationships is reduced, resulting in a schema of lower complexity.

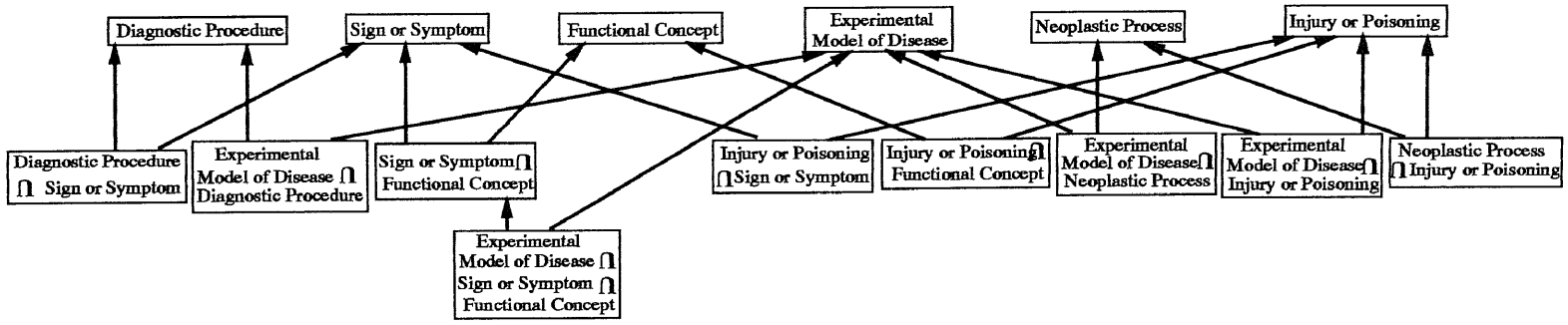

Figure 3.5 An improved solution of adding *subclass* relationships

Table 3.6 Number of classes in each level of the UMLS schema, using the refined modeling approach

Level	Number of classes	Number of intersection classes
1	1	0
2	2	0
3	4	0
4	20	0
5	41	56
6	23	203
7	23	163
8	17	186
9	2	234
10	0	212
11	0	89
12	0	16
13	0	3
14	0	1

Tables 3.6 and 3.7 show some details of the refined schema. Following the refined approach, we get an OODB schema with depth 14. To obtain this schema, 2,677 relationships are added. Comparing Tables 3.4 and 3.6, we see that intersection classes are pushed to lower levels in the refined schema. The numbers of intersection classes in the levels 5 to 9 are reduced, while the number of intersection class in level 10 grows from 70 to 212 classes. The new levels 11 to 14 contain 109 classes. Comparing Tables 3.5 and 3.7, we see a systematic reduction in the number of intersection classes with more than two superclasses. The number of intersection classes with 2 superclasses increases from 714 to 857. The number of intersection classes with 3, 4, and 5 superclasses is reduced. One class with 7 superclasses, not existing in the straightforward schema, is created. This class demonstrates the rare phenomenon of creating a class with an increased number of superclasses mentioned before.

Table 3.7 Number of superclasses for all classes in the UMLS schema, using the refined modeling approach

Number of superclasses of a class	Number of classes
0	1
1	132
2	857
3	267
4	36
5	1
6	1
7	1

To summarize, we created 1,163 intersection classes and added 2,677 new subclass relationships. All 476,314 concepts in the Metathesaurus are represented as instances of unique classes. The whole schema contains 1,296 classes. Compared with the straightforward approach where all intersection classes are subclasses of non-intersection classes, the refined approach adds more layers and fewer subclass relationships to the initial schema. Both approaches produce semantically more accurate schemas than the original Semantic Network. However, the refined approach produces a schema of lower complexity than the straightforward approach.

Figure 3.6 is a subschema of the resulting UMLS schema. It contains 15 semantic type classes and 6 intersection classes distributed over 11 levels. For comparison see Figure 3.7, where the same classes appear in a schema modeled by the straightforward approach. This schema has only 9 levels and 2 additional subclass relationships compared to Figure 3.6.

In Section 3.2.2, we discussed an apparent loss of information caused by our improved modeling. To recover the extent of a semantic type, we combine the extent of its semantic type class with the extents of all the intersection class descendants

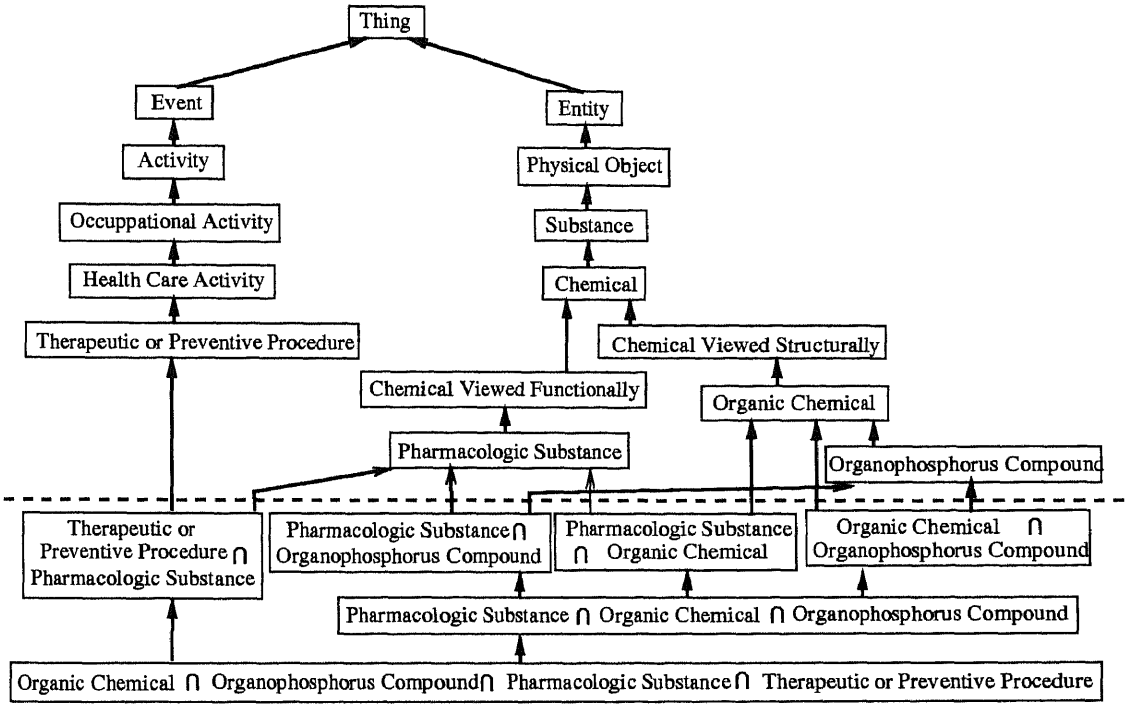

Figure 3.6 A subschema of the UMLS schema using the refined approach

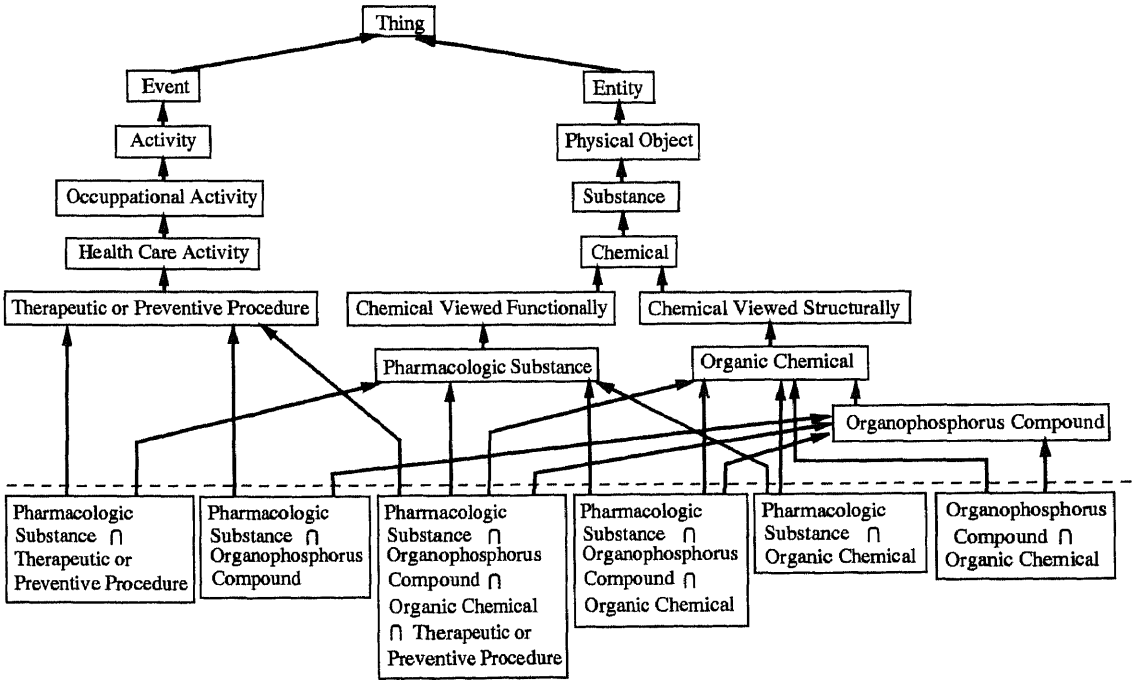

Figure 3.7 A subschema of the UMLS schema obtained by using the straightforward approach

(which, as ancestors, are defined with regards to the subclass relationships) of its semantic type class.

3.4 Advantages of the OODB Representation

3.4.1 Deeper Schema

In [98], McCray and Nelson say "The current scope of the [Semantic] Network is quite broad, yet the depth is fairly shallow. We expect to make future refinements and enhancements to the Network, based on actual use and experimentation." Introducing intersection classes and intersection classes of intersection classes provides extra refinement and extra layers to the information contained in the Semantic Network. The resulting UMLS schema has larger size and depth than the Semantic Network.

3.4.2 Uniform Semantic Classification

As we discussed before, the concepts belonging to a semantic type are not uniform since some of them belong to one or more additional semantic types. It is difficult for a user to comprehend and use the set of concepts of such a semantic type due to this lack of uniformity. Because all concepts are represented as instances of intersection classes, or of the semantic type classes, all extents extents are uniform since they contain only instances of one semantic type or one combination of semantic types. Having such classes simplifies the comprehension and use of the information contained in their concepts.

3.4.3 Reduced Average Extent Size

In the original Semantic Network, the sets of concepts of many semantic types are too large for comprehension. In the 1998 version, on average, every semantic type corresponds to about 5,000 concepts (Remember that many concepts belong to more

than one semantic type). Since the sizes of the extents of semantic types are not uniform, some of them corresponding to many more than 5,000 concepts. Thus, it is difficult for a user to comprehend those concepts.

Adding the intersection classes to the UMLS schema reduces the average number of concepts per semantic type class to about 2,700. The average number of concepts in each intersection class is 100, which is comparatively small. Having a schema with a reduced average number of instances per class improves comprehension and simplifies the use of the Metathesaurus.

3.4.4 Traversal

Since the Semantic Network and the Metathesaurus are unified into an OODB, the OODB representation enables a combined traversal of the schema layer and the instance (concept) layer. This combined traversal is faster and shorter than a traversal of the Metathesaurus itself since the OODB schema is smaller than the Metathesaurus by 2 orders of magnitude.

Suppose that a user wants to find an item of information which is stored in the UMLS, but he does not know the name of the concept of this item of information. He would however, recognize the information if he encounters it. For this purpose, the user needs to traverse the hierarchies of the Metathesaurus, using his knowledge about the target to guide his choices at different levels of the Metathesaurus. Instead of traversing the Metathesaurus through its many levels, we recommend a better approach. Utilizing the OODB representation of the UMLS, the user can traverse the OODB schema until the proper class, say S, is identified. A user will normally be able to do this, as he only needs to make a very general judgment about whether the concept that he is looking for fits into the given class or not. At this point, the user needs to switch to the subnetwork of the instance level which contains only

the concepts belonging to the class S. The traversal runs through the levels of this subnetwork until the desired concept is recognized (or its absence is noted).

As traversal requires repeated scanning through lists of children and chosing one of them, traversal is faster at the schema level than at the instance level. This is because the number of subclasses of a class in the schema is typically much smaller than the number of children of a concept in the Metathesaurus. To give an intuitive analog, think about driving on a major highway to reach a goal. Usually, after exiting the highway in the vicinity of the goal, a person will need to travel on local streets. Using the schema is like driving on a highway, while traversing the subnetwork of the Metathesaurus is comparable to driving on local roads. Traveling to a remote goal using local roads is usually slower than using a highway.

Let us demonstrate a traversal example, looking for the concept **Delusion of self-accusation**. We will now list a sequence of Metathesaurus concepts corresponding to this traversal. For each concept we list in "()" the number of its children. The user needs to scan this list to pick one child at every step of his traversal. Starting at **Medical Subject Headings** (15), traversing through **Diseases (MeSH Category)** (45), **Symptoms and General Pathology** (38), **Disease** (124), **Mental Disorders** (226), and **Delusions** (19), finally leading to the target **Delusion of self-accusation**. The traversal of this path of 7 concepts requires the user to scan a total of 467 children.

We will now contrast the above traversal with another traversal to the same target, using the OODB schema in the first phase of the traversal. We start with the root class, "Thing" (2), of the OODB schema. (The number inside the parentheses is the number of subclasses of the given class.) The traversal path from "Thing" is: "Event" (2), "Phenomenon or Process" (3), "Natural Phenomenon or Process" (1), "Biologic Function" (2), "Pathologic Function" (3), "Disease or Syndrome" (2), "Mental or Behavioral Dysfunction" (14), and the intersection class "Mental

or Behavioral Dysfunction ∩ Sign or Symptom." At this stage in our traversal, we switch to the concept level. The concept **Delusions** (19) is a root of the concept network of this intersection class. We continue on to the child **Delusion of self-accusation** which is the concept we are looking for.

This traversal passes through 9 classes with a total of 29 children and 2 concepts with a total of 19 children. The total number of scanned children (29+19=48) is much smaller than the number 467 that we found before. The combined traversal search path is longer than before, due to the fact that the Metathesaurus has many roots and the search starts at **Medical Subject Headings**. However, this disadvantage is clearly outweighted by the smaller number of children scanned. Altogether, the combined traversal supported by the UMLS schema enables a faster traversal.

3.4.5 Exposing Problems in the Current UMLS

Representing the intersection classes and their instances enables researchers to study the compound semantics of such intersection classes. In our previous experience [54, 56] with the CPMC MED [37] this has led to the identification of modeling problems. We have found a few such problems, which will be described below, and we conjecture that more problems will be found. The correction of these problems by domain experts would lead to a better new release of the UMLS.

3.4.5.1 Omissions: Let us give an example of omissions. In the UMLS schema, there is an intersection class "Body Part, Organ, or Organ Component ∩ Medical Device." Studying the extent of this class, we found that there are only four concepts in this class. They are **Dental abutments**, **Conduit with xenograft valve**, **Conduit with homograft valve**, and **Incubator.pediatric**. However, there are many more medical devices in body parts missing, e.g., heart valve. These missing concepts should be added as instances of this intersection class. The extents of

intersection classes will give the professionals in charge of the maintenance of the UMLS a useful view to discover omissions from the Metathesaurus.

3.4.5.2 Redundant Classifications: By creating intersection classes, we uncovered the phenomenon that 8,622 concepts in the Metathesaurus are assigned to several semantic types which stand in parent-child or ancestor-descendant relationships in the UMLS Semantic Network. For example, in Figure 3.6 the intersection class "Organic Chemical ∩ Organophosphorus Compound" has two superclasses "Organic Chemical" and "Organophosphorus Compound." However, "Organophosphorus Compound" is itself a subclass of "Organic Chemical." The creation of this intersection class was due to the fact that there are 127 concepts assigned to both the semantic types **Organic Chemical** and **Organophosphorus Compound**. This situation is not in line with the intentions of the UMLS designers. In [98], when discussing the assignment of concepts to semantic types, it is stated that "In all cases the most specific semantic type available in the hierarchy is assigned to a term." Therefore, those 127 concepts should only be assigned to the semantic type **Organophosphorus Compound**. As a result, the intersection class "Organic Chemical ∩ Organophosphorus Compound" will cease to exist. Thus, we get a new subschema (see Figure 3.8) replacing the previous one in Figure 3.6.

If all those redundant classifications are removed from the UMLS (that is, if all those 8,622 concepts are only assigned to one semantic type), 77 intersection classes will disappear from the UMLS schema. We believe that these redundant classifications resulted from the fact that the assignment of concepts to semantic types was done by different experts for the different UMLS sources. However, the use of intersection classes helped us to uncover such redundancies.

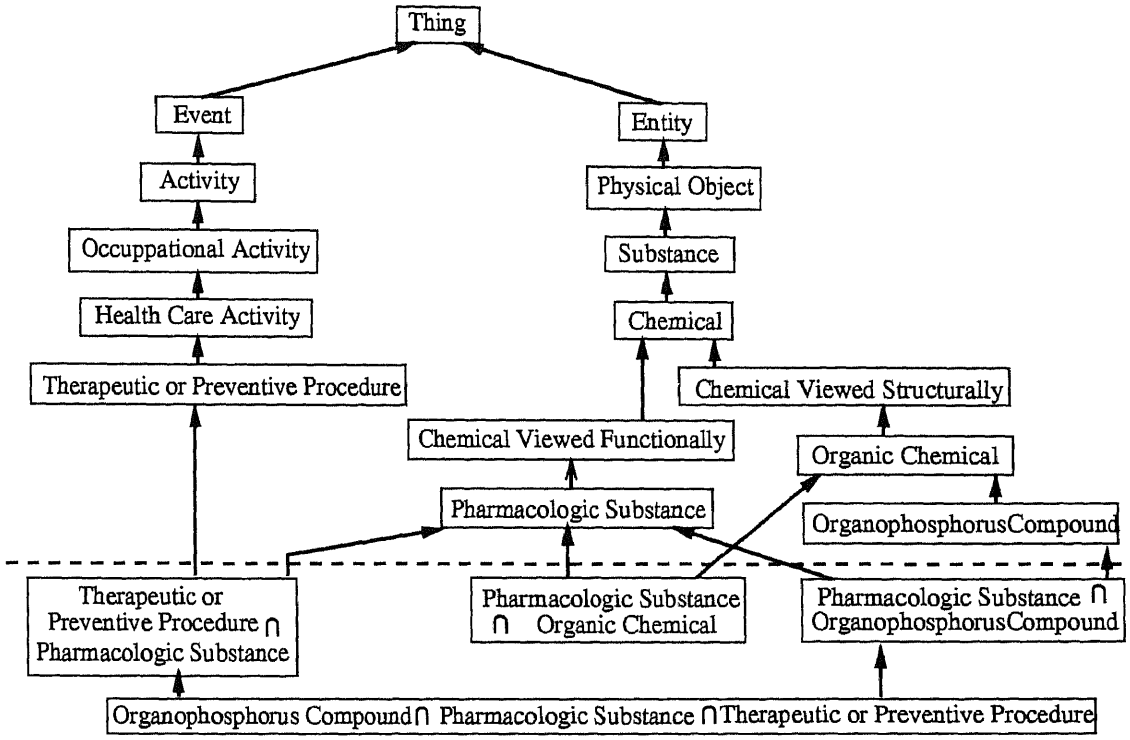

Figure 3.8 The subschema corresponding to Figure 3.6 after removing redundant classification

A list of the above 8,622 concepts and their correct semantic types was submitted to NLM. We have been notified that these redundant classifications will be removed from the next version of the UMLS.

3.4.5.3 Classification Errors: Intersection classes highlight some classification errors in the UMLS. For example, the concept **Encephalities Viruses** is the only instance of the intersection class "Virus ∩ Disease or Syndrome." But it is only a virus and should not be classified as a disease. Hence, it should be an instance of the "Virus" semantic type class. Furthermore, **Encephalities Viruses** is the only instance of the intersection class "Virus ∩ Disease or Syndrome," this intersection class is not needed.

Another example of a classification errors is the concept **Scotch Tape Mount**. It is the only instance of the intersection class "Bacterium ∩ Laboratory Procedure."

However, it is not a bacterium and should be an instance of the class "Laboratory Procedure." Thus, the intersection class "Bacterium ∩ Laboratory Procedure" should not exist either. Similarly, the concept **Urea formaldehyde resin** is the only instance of the intersection class "Organism ∩ Biomedical or Dental Material." But it is not an organism and should belong only to "Biomedical or Dental Material." Thus, the intersection class "Organism ∩ Biomedical or Dental Material" is not needed anymore. The concept **Alagille Syndrome** is the only instance of the intersection class "Congenital Abnormality ∩ Body Location or Region ∩ Disease or Syndrome." However, it is not a location. Thus, it should be an instance of the intersection class "Congenital Abnormality ∩ Disease or Syndrome" and the intersection class "Congenital Abnormality ∩ Body Location or Region ∩ Disease or Syndrome" is not needed either.

3.4.5.4 Ambiguity: Intersection classes helped us discover ambiguities of concepts in the UMLS. E.g., the intersection class "Plant ∩ Disease or Syndrome" has only one instance **Toxicodendron**. However, **Toxicodendron**, known popularly as poison ivy, refers to two different concepts, one is a plant and the other is the name of a disease. In order to differentiate them, two concepts should be created such that one is an instance of the class "Plant" and the other is an instance of the class "Disease or Syndrome." Since the intersection class has only this instance, it will be eliminated. Let us look at another example. The concept **Paronychia of toe** is the only instance of the intersection class "Anatomical Structure ∩ Disease or Syndrome." The classification exposes the need for two different concepts. One is the *diseased toe*, which is a body part and should be an instance of the class "Anatomical Structure." The other is the *disease of the toe* which should be an instance of the class "Disease or Syndrome." Thus, no such intersection class is necessary.

3.4.5.5 Non-uniform Classification: For some intersection classes their extents indicate that a non-uniform classification was employed for some concepts in the UMLS. For example, the concept **Prematurity** is the only instance of the intersection class "Organism Attribute ∩ Temporal Concept." The classification of **Prematurity** to both semantic types "organism attribute" and "temporal concept" is definitely legitimate. However, if this organism attribute is modeled as a temporal concept, then there exist other organism attributes which should be also classified as temporal conceps, e.g., the concept **Senility**. Hence, while the extend of the intersection class does not expose an error, it exposes non-uniformity in the way concepts were classified into semantic types in the UMLS. This non-uniformity is not surprising, when considering that many experts were involved in the classification of concepts into semantic types. Such feedback should be communicated to domain experts who should try to change the classification to be more uniform either by adding other relevant concepts to the intersection class or deleting the existing one in which case the intersection class will become empty.

3.4.5.6 Sample of Intersection Classes: In the UMLS schema, there are 422 intersection classes with only one instance. One of the authors, Dr. Cimino, checked the first 100 such intersection classes and their instances. For 11 intersection classes out of 100, the classification of concepts is correct. For 55 of these intersection classes, the multiple classifications are wrong. For 32 intersection classes the classified concepts indicate non-uniform classifications as explained in Section 3.4.5.5. There are 2 intersection classes which are redundant classification cases.

3.5 Summary

The Unified Medical Language System (UMLS) integrates many medical terminologies and coding systems. It plays a major role in overcoming terminological

differences in the design of computerized healthcare information systems. However, the size and complexity of the UMLS make it difficult to maintain and use. To help overcoming this problem, we have developed a methodology for representing two components of the UMLS, the Metathesaurus and the Semantic Network, as a unified OODB. the resulting UMLS OODB schema enhances the Semantic Network by adding more layers and providing more refinment than available in the Semantic Network. The UMLS OODB schema also supports a fast two level traversal of the Metathesaurus. It makes comprehension of the Metathesaurus easier, by partitioning it into semantically uniform classes. The latter, in turn, has led to the recognition of possible improvements of the UMLS.

CHAPTER 4

PARTITIONING AN OODB SCHEMA INTO CONTEXTS BY IDENTIFYING A FOREST HIERARCHY

4.1 Introduction

Object-Oriented Database (OODB) systems help manage complex, large bodies of information. However, comprehending the contents of an OODB system is a difficult task for a user, whenever it contains large amounts of complex information. The OODB schema, providing an abstraction of the OODB system, plays a major role as a tool for comprehending the contents of the OODB. However, an OODB schema itself may be large and hard to comprehend.

The graphical representation of a schema can help the user in obtaining an orientation in the schema. Thus, we assume that OODB schemas are represented graphically and manipulated with a graphical schema editor.

Unfortunately, for a large OODB schema whose graphical representation does not fit on a single screen (page), the advantages of the graphical representation are less significant. Thus, even with the help of a graphical representation, the user will still encounter schemas of large size and complexity which will cause comprehension problems.

Our work is motivated by the desire to comprehend large schemas of OODBs by developing methods to make them understandable for users. We present both a theoretical paradigm and a methodology to aid comprehension of existing large schemas. Our approach to achieve comprehension of graphical schemas is based on combining two concepts: *informational thinning* (i.e. display only high priority elements of the schema) and *partitioning*. A preliminary presentation of the theoretical paradigm only appeared in [117].

Our methodology is based on partitioning a large OODB schema into disjoint meaningful, manageably sized parts. Thus, to comprehend the schema, the user can

start to study the details of selected small parts. More specifically, the components of the partition are all trees, and each of these trees can ordinarily fit easily to a single computer screen.

In this chapter, we present a new technique for modeling, called *disciplined modeling.* Based on the rules of disciplined modeling, we develop a theoretical paradigm to support the existence of a meaningful forest hierarchy within the specialization hierarchy based on our paradigm. We present a methodology for finding such a forest hierarchy. This methodology relies on an interaction between a user and the computer. A user is asked to refine the specialization hierarchy of an OODB schema according to the rules of disciplined modeling. The resulting forest hierarchy represents a partitioning of the specialization hierarchy into trees. Such a hierarchy functions as a skeleton of the schema and supports comprehension efforts. We will demonstrate our methodology by applying it to the subschemas of a university database and the MED (Medical Entities Dictionary) [37]. In [59] we presented a methodology for partitioning a vocabulary modeled as a Semantic Network. The different model required a different theoretical paradigm although similar.

The rest of this chapter is organized as follow. In Section 4.2, we describe the notions of informational thinning and partitioning. The rules of disciplined modeling are introduced in Section 4.3. Section 4.4 presents the proof that the rules of disciplined modeling guarantee the existence of a forest structure. Our methodology for partitioning the OODB schema into trees are described in Section 4.5. In Sections 4.6 and 4.7, we apply the methodology to the subschemas of a university database and the MED. Section 4.8 describes how the forest structure helps comprehension. Section 4.9 contains our conclusions.

4.2 Informational Thinning and Partitioning

4.2.1 Schema Complexity

To get a handle on what it means for a schema to be large and complex, we measure the *size* of the schema by the number of its classes. Our experience is that comprehension difficulty for a large and complex schema stems more from the density of the relationships than from the volume of the classes. We define the *complexity*, **c**, of a schema as the ratio of the number of the relationships between classes to the number of classes. For two schemas of equal size we conjecture, based on our experience, that a more complex schema is more difficult to comprehend.

In this chapter, we will demonstrate our techniques on two schemas. In our previous work [99, 100] we used a large OODB schema describing a university environment. Figure 4.1 shows a subschema of the university OODB schema concentrating on some of the academic aspects of the university structure. It contains 36 classes and is about a third in size of the whole schema. (Note that the schema was simplified by omissions. For example, only few details of publications and education records appear.) In [56, 91, 90], we developed an OODB schema, containing 124 classes, for modeling a controlled medical vocabulary MED [37]. We will apply our methodology to the above two schemas.

One method that has been widely used to make schemas comprehensible is to use a graphical representation. We note, e.g., the popularity of the Entity Relationship model and its graphical representation [30] and graphical case tools for object-oriented programming such as OMT [124] and ROSE [16]. The two schemas are presented using our graphical OODB language OOdini [66]. In OOdini, a class is represented as a rectangle, and a set class, as a double line rectangle. A relationship is represented as a labeled thin arrow, and a multivalued relationship, as a double thin arrow. A subclass relationship is drawn as a bold arrow directed upward from

the subclass to the superclass. An attribute is listed inside its class rectangle beneath the class's name.

Figure 4.1 contains 36 classes and 74 relationships, and thus its complexity ratio is $\mathbf{c} = 74/36 = 2.06$. A user needs a substantial effort to obtain an orientation in this schema. This is rather distressing, because, besides the fact that this is only a subschema of the original schema, Figure 4.1 has been further simplified by not showing the attributes of the classes (except one).

One way for imposing an order on the human understanding process can be described as follows. We select a subschema of lower complexity. After the human has gained an understanding of this subschema he will find it easier to comprehend the original schema.

4.2.2 Informational Thinning

We will now discuss *informational thinning*. Informational thinning tries to eliminate information from the whole schema by prioritizing various kinds of properties of the classes and displaying only high priority kinds of properties.

One of the major concepts supported in OODBs is the notion of generalization/specialization. The specialization hierarchy of an OODB schema serves as the basis for property inheritance. It is the backbone of an OODB schema. Informational thinning lets us concentrate on the specialization hierarchy of a schema by removing all other properties and leaving only the *subclass* relationships. Since only the classes and the subclass relationships of the schema of Figure 4.1 are displayed in the schema of Figure 4.2, Figure 4.2 shows a specialization hierarchy of Figure 4.1. We call it the *hierarchical (sub)schema of a schema.* Note that the complexity of Figure 4.2 is $\mathbf{c} = 26/36 \approx 0.72$. This subschema has the same size as the original schema but a lower complexity and it is easier to comprehend. Furthermore, it is easier to comprehend due to the uniform nature of the hierarchy relationships, in

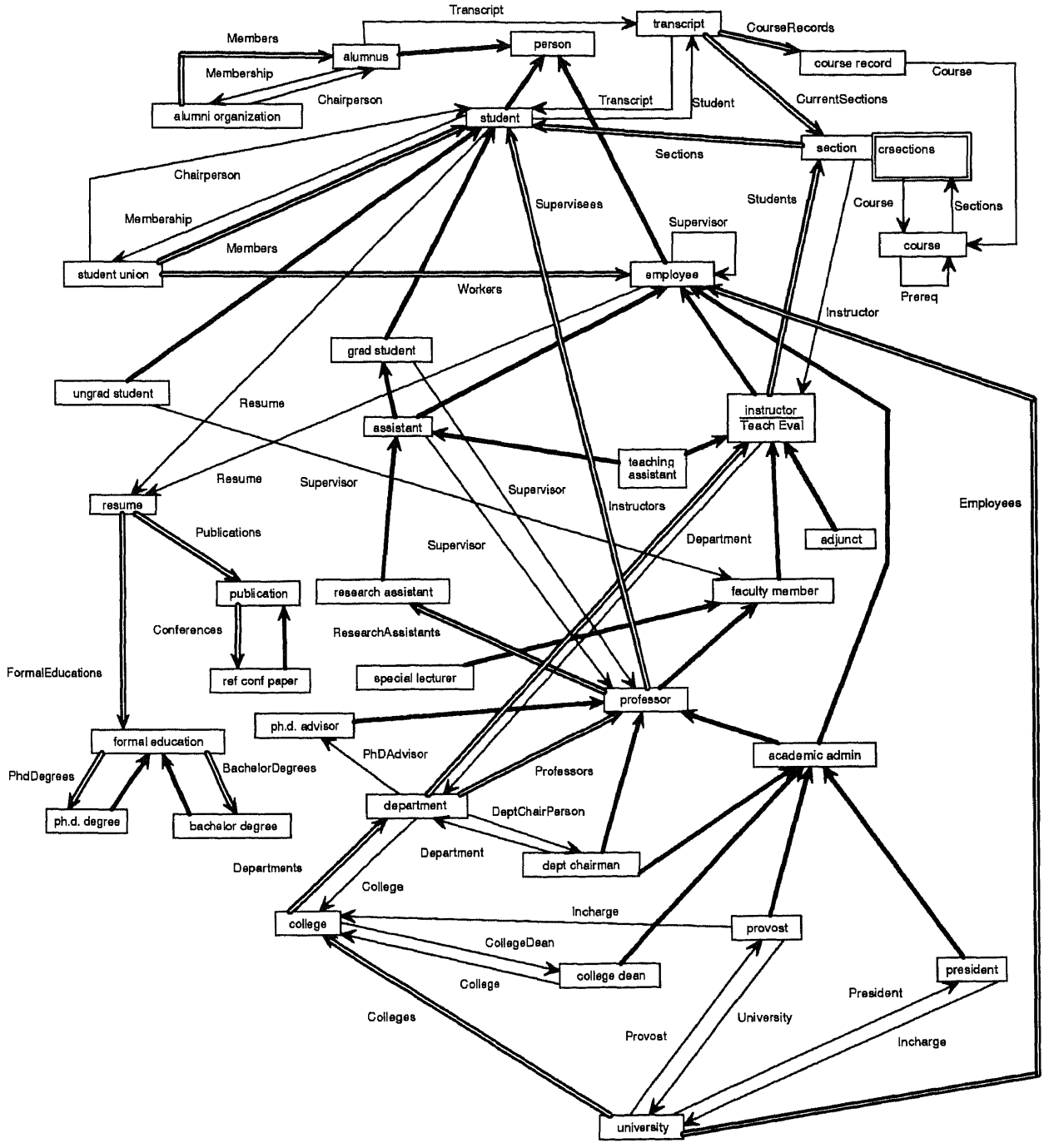

Figure 4.1 A subschema of a university database

contrast to the various semantics of the rest of the relationships which are user-defined. Due to this difference, the designer does not have to label the subclass relationship, which is identified by a special graphical icon, while for the user defined relationships, labels are necessary to specify their semantics.

Since a class in an OODB schema can be specialized into a number of subclasses and can also be generalized into a number of superclasses, the hierarchical subschema of an OODB schema will be a directed acyclic graph (DAG). Thus, for large schemas even the hierarchical subschema may be difficult to comprehend due to its size and the existence of multiple superclasses for many classes. In graph theory, a hierarchical schema has a forest structure if no class has more than one superclass. If such a hierarchical schema is connected, then it forms a spanning tree of the DAG. It is generally considered to be easier to comprehend a forest hierarchical schema than a DAG schema of the same size, due to the fact that upward paths are not branching in a forest structure.

4.2.3 Schema Partitioning

A second approach to simplify the comprehension of a complex large schema is partitioning it into smaller subschemas. This applies to general and hierarchical schemas alike. From the technical side, only a limited size subschema can be displayed on a computer screen. From the conceptual side, human comprehension capacity is limited and is functioning much better on a small subschema. Hence, we are faced with the task of partitioning a large schema into smaller subschemas. In the partitioning we typically have two purposes:

1. To identify small subschemas which comprise logical units of the original schema.

2. To generate a minimal number of subschemas each of which fits on a computer screen by itself, and which together comprise the complete schema.

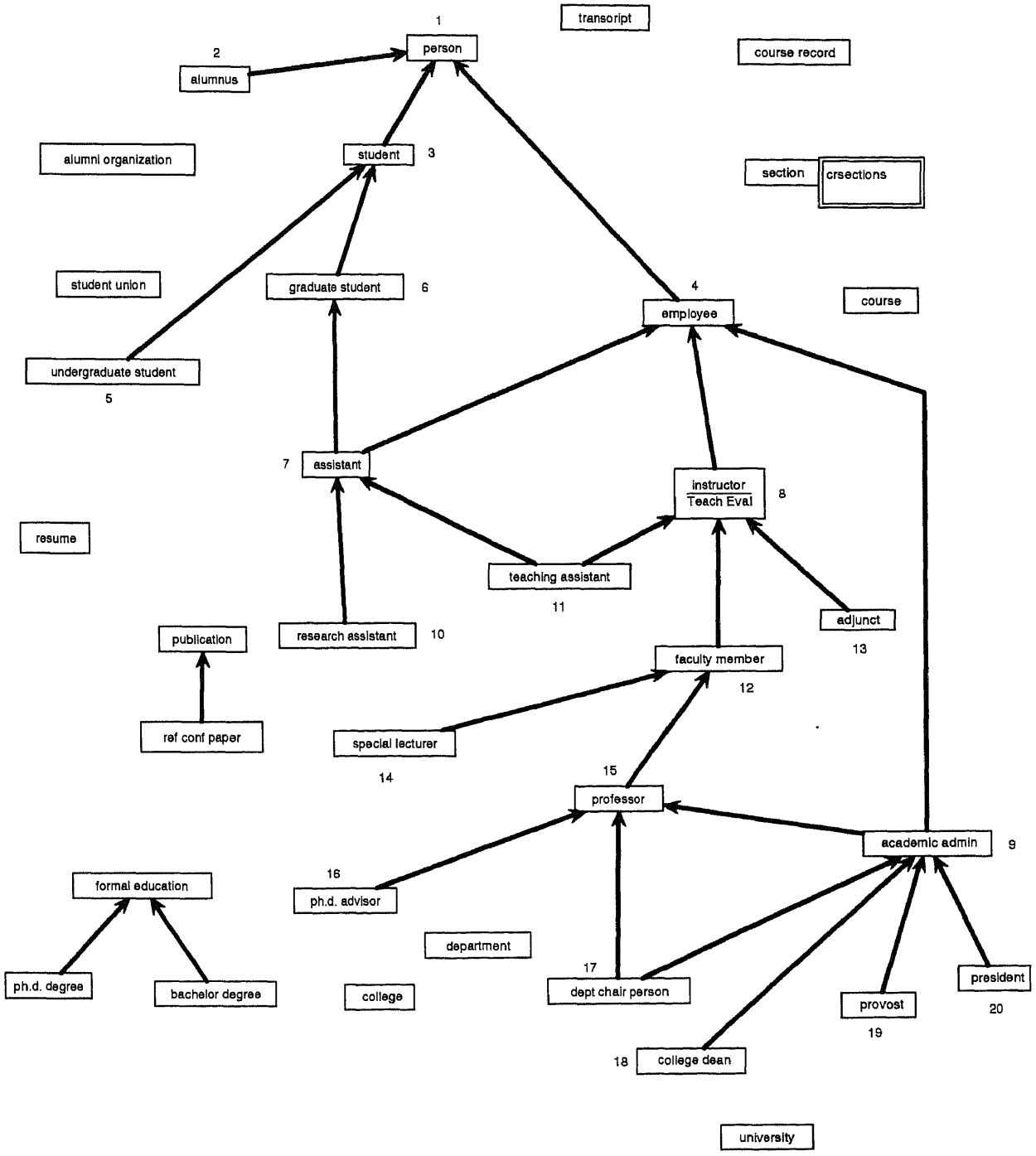

Figure 4.2 A subschema of a university database after applying informational thinning

Note that the need to achieve a logical partitioning of an existing schema introduces a vicious cycle, as one needs to comprehend the schema in order to partition it logically. Nevertheless, partitioning into logical subschemas tends to minimize the number of relationships between different subschemas, i.e., between a class from a subschema and a class of another subschema. Unfortunately, the problem of partitioning a schema according to the above criterion or some similar criteria is known to be NP-complete, that is, no efficient algorithm is known for it, and it is conjectured that no such algorithm exists [50].

A possible line of action is to combine the two comprehension approaches outlined above, informational thinning and partitioning, by trying to partition the hierarchical subschema and then use this partition to impose a partition on the original schema. Obviously, this partitioning problem is much simpler than the original partitioning problem since the schema has lower complexity. However, unless the hierarchical subschema has a forest structure, the partitioning problem in general is still an NP-complete problem. On the other hand, if the hierarchical subschema has a forest structure then there exist efficient algorithms for optimal partitioning according to various criteria [3, 9, 10, 11, 12, 81, 92, 118]. In this chapter we will describe an approach to show that in a hierarchical subschema of a general schema, there exists such a forest hierarchical schema, the semantics of which helps to support comprehension.

4.3 The Rules of Disciplined Modeling

4.3.1 The Category-of and Role-of Specialization Relationships

In order to identify a meaningful forest structure subschema of a hierarchical schema we shall look into the nature of the specialization relationship. In previous research, we as well other researchers [51, 109, 110, 111] identified two different kinds of specialization relationships, namely, *category-of* and *role-of*. According to our definition

category-of is a specialization relationship used for refinement in case that both the superclass and the subclass are in the same context. On the other hand, *role-of* is the specialization relationship used in the case the superclass and the subclass are in different contexts, and the subclass functions in a *role-of* the superclass.

How does a designer of an OODB schema determine whether a given specialization relationship is *category-of* or *role-of*? This depends on whether the two classes connected by the relationship are in the same context or not. For example, consider the classes in Figure 4.2. The class **student** is a subclass of the class **person**, and the class **graduate student** is a subclass of the class **student**. However, intuition tells us that information about the student and the graduate student are both in the same context of learning, while **person** is in a different context of personal life. Hence, the class **graduate** Student is *category-of* the class **student**, as it represents a refinement, which in turn is *role-of* the class **person**. However, this determination is not always so easy. In spite of extensive research, e.g., [22, 23, 60, 74, 94, 103, 128] there is still no definition of context which is widely accepted. One line of research on context comes out of the CYC project [87]. There, an attempt to build a gigantic knowledge base was found doomed to failure if contexts are not introduced for structuring. Work following this line [22, 60, 94] assumes that a context is a first class object used to parameterize axiom schemata. However, no clarity about the nature of contexts themselves is gained by this approach. As a workshop on the use of context in Natural Language Processing showed [74] the best thing researchers in Natural Language Processing can currently do is to agree that they disagree on what contexts are. Our approach is that we are not trying to define the notion of context. Rather we are making the pretheoretical (axiomatic) assumption that contexts exist in human thinking and we are trying to identify them.

We accept the situation that for some designers two classes are in the same context while for others they are in different contexts, due to different views of the

application, different emphasis and different levels of refinement. In our view, the designer of a schema should have the freedom to determine for each class, to which context it belongs.

Still, we believe that organizing for a user a complex schema into *reasonable* contexts is still preferable to leaving him without such an organization. What we are providing in this chapter is a theoretical paradigm for the existence of such assignments of classes to contexts which will result in a forest subschema of the DAG hierarchical schema, Also, we are introducing a methodology for finding such a forest. The forest subschema will support comprehension of the schema.

In order to ensure that a forest hierarchical subschema can be identified, the assignment of classes to contexts must always satisfy three rules which will be introduced below. We refer to modeling which satisfies these rules as *disciplined modeling.* As we shall show, a schema designer using the rules of disciplined modeling can still model every situation which is modeled when these extra rules do not apply. Only few modifications are required in the modeling, so that the rules of disciplined modeling are satisfied. As we shall see, all three rules are concerned directly or indirectly with the *category-of* relationship.

4.3.2 Contexts as Equivalence Relations

First we define a new mathematical relation *equicontext*, or "in the same context," between classes. A pair of two classes belongs to the equicontext relation if both classes belong to the same context.

It is interesting to contrast the two relations *category-of* and "in the same context." The *category-of* relation is directed and asymmetric while "in the same context" is an undirected relation since it is symmetric. The *category-of* relation between classes implies the equicontext relation between them, but the opposite is not necessarily true. For example, if two classes A and B are both *category-of* class

C, then A and B are in the same context since both are in the same context as class C, but A and B are not *category-of* one another.

Rule 1: The equicontext relation between classes is an equivalence relation, that is, it satisfies the three conditions of an equivalence relation: reflexivity, symmetry and transitivity.

An equivalence relation partitions the elements of a set into disjoint subsets, such that every two elements of the same subset are related and no two elements of different subsets are related. Hence, **Rule 1** implies **Rule 1'**.

Rule 1': The classes of a schema are partitioned by the equicontext relation into disjoint contexts.

Rule 1' will force the designer into explicit specification of the contexts in his schema and lead him to resolve some ambiguous situations in a systematic way. Due to the symmetry and transitivity of the "in the same context" relation, any two classes, between which there is a path of *category-of* relationships, where the relationships have *any orientation*, are in the same context.

We do not claim to have a unique way of assigning classes to contexts. As we are dealing with a problem of data modeling, there are usually different ways to model the same real world environment. We further do not claim that contexts in an application are naturally disjoint. To the contrary, in many applications and specially in complex ones, contexts overlap. However, in order to achieve our purposes, disciplined modeling requires the modeler to enforce disjoint contexts. Consider for example the class **teaching assistant** in Figure 4.2. From one side it belongs to the employment context as does its superclass **assistant**. On the other hand, it also belongs to the teaching context as its superclass **instructor** is the root of this class. However, **Rule 1** forces the class **teaching assistant** to belong to one context only. As we shall see later, this will be the employment context.

As will be seen, the partitioning of classes into disjoint contexts is a difficult task involving delicate analysis. It is possible that different modelers will make different decisions according to their different views of the application, different emphasis and different levels of refinement. We accept the possibility of differing partitions by different modelers as a natural result of the non-uniqueness of data modeling problems. However, we require each partitioning to satisfy the conditions for disciplined modeling, and we claim that any reasonable partitioning is better than a completely unstructured schema.

4.3.3 A Category-of Refinement is Exclusive

The *category-of* relationship is used when we need to refine the concept represented by the superclass when both the superclass and the subclass are in the same context. This means that instances of the superclass are divided into being also instances of the different *category-of* subclasses according to a distinction employed. In disciplined modeling, we further require from such a categorization that the refinement will be into mutually exclusive concepts.

Rule 2: Two classes which are *category-of* specializations of the same superclass cannot both contain an instance representing the same real world object.

This means that an instance that belongs to the extent of a superclass cannot belong simultaneously to the extent of more than one subclass. In other words, the real world objects corresponding to the instances of the different subclasses form disjoint sets. **Rule 2** is essential for achieving our purpose that the set of *category-of* relationships in a schema will form a forest structure.

Let us consider how to guarantee **Rule 2** in disciplined modeling. For a case of a class A which is a subclass of two classes B and C, by **Rule 2** it cannot be that both these subclass relationships are *category-of*. Thus we need to give the

disciplined modeler guidelines how to deal with the modeling of such a situation. Such guidelines will be discussed in Section 4.5.

Rule 3: For each context there exists one class R which is the *major* (or definitive) class for this context such that every class in this context is a descendent of R.

In other words, each context has one class which is a "root" for this context, i.e., there is a directed path of *category-of* relationships from each class of the context to this class. Note that we use here the notion of a directed tree where all the directions are towards the "root" rather than away from it. I.e., in graph theory terms the root is a sink.

Note that **Rule 3** does not actually limit the modeling of the application. In case that a context has several root classes, a new class R can be created with all the root classes as its subclasses. Thus, R will become the new unique root of the context.

4.4 Disciplined Modeling Results in a Forest Structure

In this section, we will prove the following theorem.

Theorem: Using disciplined modeling, a class has at most one superclass to which it has a *category-of* relationship.

Proof: Suppose, to the contrary, that there exists a class A which is *category-of* both class B and class C.

Hence class A and class B are in the same context. Similarly class A and class C are in the same context. By the transitivity of the equicontext relation (**Rule 1**) class B and class C are in the same context.

By **Rule 3** the joint context of the classes B and C has a root class D such that both classes B and C are descendents of D with regards to *category-of* relationships (Figure 4.3). In other words there is a sequence of *category-of* relationships from class $B(C)$ up to class D. Let $E(F)$ be a child class of the class D on the path of

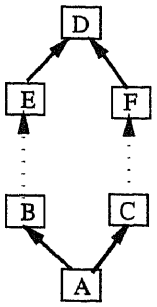

Figure 4.3 Schema demonstrating contradiction

category-of relationships from class $B(C)$ to class D. In case $E = F$, let E and F be the last classes to be distinct on the paths from classes B and C to the root class D; let D be the parent class of the classes E and F.

In the following discussion we are going to use a relation "*represents the same real world object*" defined for instances of a database. A pair of instances of the database belongs to this relation if both instances represent the same real world object. The relation *represents the same real world object* is obviously transitive.

Let a be an instance of class A. Then class B has an instance b_a corresponding to instance a of A, since A is *category-of* class B. In other words both instances a and b_a represent the same real world object.

Similarly, there exists an instance c_a of class C which represents the same real world object as the instance a of A. Thus both instances b_a and c_a represent the same real world object, due to the transitivity of the relation *represents the same real world object.*

As noted before, Class B (C) has a sequence of *category-of* relationships up to class E (F). Hence, class E (F) contains an instance e_a (f_a) corresponding to the instance b_a (c_a) of class B (C) where this correspondence is defined transitively from the correspondence along the sequence of *category-of* relationships. Hence, both the instances e_a and b_a (c_a and f_a) represent the same real world object. But, as was shown before, the instances b_a and c_a represent the same real world object. Thus

it follows again from the transitivity of the relation *represents the same real world object* that the instances e_a and f_a represent the same real world object.

But by **Rule 2** the extents of the classes E and F which are both *category-of* class D may not both contain an instance representing the same real world object, so the previous conclusion contradicts our assumptions. Hence the class A cannot be *category-of* the two classes B and C, however A can be only *category of* one class. ■

Corollary: The *category-of* hierarchy has a forest structure, i.e., consists of one or several tree structures.

Proof: A directed graph which contains no cycles and each vertex has at most one parent is a forest. Since the *category-of* hierarchy is a subhierarchy of the directed acyclic subclass hierarchy, it has no cycle. By the theorem, each vertex has at most one *category-of* superclass. Hence, the *category-of* hierarchy is a forest. ■

The tree structures of the forest serve as the backbones of the schema and will be critical in our efforts to comprehend the schema and partition it into manageable subschemas.

4.5 A Methodology for Finding a Forest Hierarchy

We have described a conceptual partitioning framework which guarantees that for the price of following the rules of disciplined modeling, there can be found a *forest structure subschema* of an OODB schema. This forest structure subschema serves as a skeleton supporting the comprehension of the schema. Furthermore, the trees of the forest represent contexts which are logical subschemas approximating all information relevant to a specific subject area, further supporting the comprehension of the original schema.

In this section, we will describe a methodology, based on our theoretical paradigm, to identify a forest structure subschema of a given schema. By a methodology we mean a process that involves human-machine cooperation. The

human domain expert is called upon to make some judgement decisions based on his understanding of the application while the computer supports the human by providing results of algorithmic procedures for tasks which do not involve complex intuitive decisions but might require many computational steps.

In the following description of the methodology, we will specify which parts are performed by a computer and which are performed by a human expert. The result of our methodology is a refinement of the specialization hierarchy of the OODB schema. Every subclass relationship becomes either a *category-of* or a *role-of*. We will differentiate between three kinds of *role-of* relationships. They are *regular role-of*, *role-of/intersection* and *role-of/category-of*. However, for partitioning purposes, they will all be treated in the same way. The *category-of* relationships will form a forest, and all *role-of* relationships will be deleted.

Step 1: Informational thinning. (Computer)

All attributes and relationships other than subclass relationships are removed from the OODB schema.

Step 2: Topological sort. (Computer)

The resulting subschema from **Step 1** is arranged in topological sort order.

Step 3: Identify roots of contexts. (Human)

The subschema is scanned top-down according to the order from **Step 2**. In this scanning, classes which are defining classes (roots) of contexts are identified. The decision should be made by the meaning and importance of the class in the application compared to its superclasses' meanings. These chosen classes start new contexts rather than refining the contexts of their superclasses.

After these classes are identified, the subclass relationships from them to their superclasses are changed to *role-of* relationships. This kind of *role-of* relationship is a *regular role-of*, where the relationship models a switch of context, that is, the relationship goes from a class in one context to a class in another context.

Step 4: Multiple superclasses. (Computer)

All classes with multiple non-*role-of* relationships to superclasses are listed in bottom-up order. (We will explain later why we are using bottom-up processing at this point.)

Step 5: Identify major superclass. (Human)

For each class identified in **Step 4**, the expert needs to identify at most one superclass which is in the same context as the class to conform with **Rule 2**. The subclass relationship to this superclass will be defined as a *category-of* relationship while all other subclass relationships of the class are defined as *role-of*.

In our experience, for most of the classes with multiple superclasses, an expert can easily determine which of the superclasses is the major one, i.e., which should have a *category-of* relationship directed to it. There is a minority of cases where the decision about a major superclass of a given class is not easy. In such cases, we try to distinguish which of the several superclasses, if any, should have a *category-of* relationship pointing to it, based on the partial context information we have already accumulated in our bottom-up processing. We provide the following guidelines.

Case 1: One of the superclasses is definitional, describing the essence or the definition of the subclass, while the other superclasses describe the functionality or usage of the subclass. Then we look at the partial context to which the class and its descendants belong. (This is the reason for the bottom-up processing). We try to determine whether the nature of the *category-of* relationships in this partial context is functional or definitional. If it is definitional, the definitional superclass is chosen as major superclass. If it is functional, then we will prefer the functional superclass. If there are several functional superclasses, we will prefer the one which matches the function appearing in the partial context of the class. If the class is currently the only class in its context, we will choose the definitional superclass. As a result, one superclass is chosen as major superclass. The class is made *category-of* this major

superclass and *role-of* the other superclasses. This kind of *role-of* relationship is called a *regular role-of* since a switch of context from the class to the superclass has occurred.

For example consider the class **teaching assistant** which has two superclasses **assistant** and **instructor** (see Figure 4.2). The superclass **assistant** is definitional (Q: What is a teaching assistant? A: An assistant) and the superclass **Instructor** is functional (Q: What does a teaching assistant do? A: He instructs). Thus, **teaching assistant** should be *category-of* **assistant** and *role-of* **instructor**.

Case 2: Both superclasses are structural, however it is possible to distinguish the major from the minor by linguistic analysis of the name of the subclass. For example, when the concept of one superclass is expressed in the subclass name as a noun while the concept of another superclass is expressed in the subclass name as an adjective then the noun defines the major superclass. As another example, if both concepts are expressed grammatically as nouns then the second noun is considered the major concept[1] In this case we follow the structure of a noun phrase consisting of a head noun, which appears last, and a modifier noun.

For example, reconsidering the above example, the class **teaching assistant** has two superclasses **assistant** and **instructor**. Since assistant is a noun and teaching is an adjective in the name of the class **teaching assistant**, according to **Case 2**, the class **assistant** is chosen as the major superclass and the **instructor** is a minor superclass.

Case 3: All superclasses are definitional, with the same importance or indistinguishable importance, as each of them contributes to the definition of the class in an equal or indistinguishable way. In that case, the semantics of the class is a combination of the semantics of all its superclasses. In such a situation, the class with multiple superclasses could belong to the context of any of its superclasses. However,

[1]There are well known exceptions to this rule. A toy gun is a toy and not a gun.

by **Rule 1** it cannot belong to more than one context. Also, we have no reason to prefer one over the others. Each choice of context will disassociate the class from the other contexts. This conflict is resolved by requiring that such a class starts a new context which represents the class as an intersection of its superclasses. Thus, this class is *role-of* all its superclasses. We call this type of *role-of* "*role-of/intersection*" represented as r/i in the figures. By this term, we emphasize that this is not an actual case of a switch of context but an artificial case due to the requirement of the theorem to forbid a class with two *category-of* superclasses. Without the theorem, we could probably leave the intersection concept in the context of its superclasses if all belong to one context. In Section 4.7, we will show such an example in partitioning of the MED schema. This concludes the three cases of **Step 5.**

Step 6: Identify diamond structures. (Computer)

For each class I in the resulting list of **Step 4** and each pair of superclasses $S1$ and $S2$ of I, find a lowest common ancestor A of both $S1$ and $S2$. For each pair of such classes I and A, output the structure (represented by $< I, A >$) containing I, A and all the classes which are descendants of A and ancestors of I. This is called a diamond or extended diamond structure. The class I is called the source of $< I, A >$, and the class A is called the sink of $< I, A >$.

Step 7: Resolve contradictions in the diamond structures. (Computer)

In order to fulfill **Rule 2** of disciplined modeling, each diamond or extended diamond structure must contain classes from more than one context. After executing the above steps, all the diamond structures already satisfy **Rule 2.** However, there is one case where we must artificially change additional *category-of* relationships to *role-of* relationships, to resolve a contradiction.

In this case, which we call *contradictory diamond case*, the source I of the diamond structure $< I, A >$ is a *role-of/intersection* of its superclasses. All other classes in the diamond structure belong to one context (see Figure 4.4). Since the

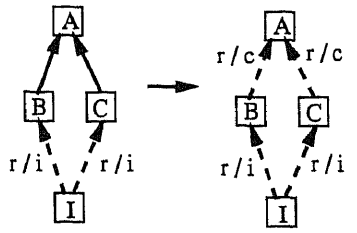

Figure 4.4 The diamond structure

source I is the intersection of two superclasses B and C, they cannot both belong to the same context of their superclass A. Otherwise, since the intersection of a context with itself will result in the original context, the intersection class must belong to this common context. Thus, the classes B and C are also defined as separate contexts. The *category-of* relationships from B and C to A are changed to *role-of*. However, we want to maintain the distinction between this *role-of* and the two other kinds. Therefore, we denote this kind of *role-of* as "*role-of/category-of*." It is represented by r/c in the figures.

Step 8: Get a forest hierarchy. (Computer)

After all subclass relationships are refined as either *category-of* or *role-of* relationship, a forest hierarchy of the *category-of* relationship is obtained, by deleting all three kinds of *role-of* relationships.

Note that the methodology used both top-down processing and bottom-up processing. The determination of the context of classes is performed top-down, since the context of the root class defines the context of its descendants. When scanning the schema top-down, an expert can identify which class defines a new context rather than continuing a context of one of its superclasses which has been processed already.

On the other hand, when determining bottom-up to which context a class belongs, choosing from among its superclasses, it is important to know the descendants of the class which belong to the same context. This knowledge will help to determine which of the superclasses fits best to the already constructed partial context.

4.6 Applying the Methodology to the Subschema of University Database

In this section, we will apply our methodology step-by-step to the subschema of a university database (Figure 4.1). Since the whole schema in Figure 4.1 deals with the academic context, if we insist on having this one context only, then every subclass relationships should be *category-of* and the schema would not have a forest structure. However, when the university environment is our application domain, having only one academic context does not help in comprehending the application. Naturally, we want to divide the schema into several contexts.

Step 1 of our methodology is informational thinning. Figure 4.2 is the result of **Step 1**.

Step 2, topological sort, is applied to the hierarchy of Figure 4.2. Since some hierarchies in Figure 4.2 are singleton or contain small numbers of classes, we only show the processing of the big hierarchy, rooted at the class **person**. Since there are degrees of freedom in applying topological sort to a DAG, the order we used is left-to-right breadth first search [5]. The class numbering from 1 to 20 in Figure 4.2 reflects this order.

In **Step 3**, the hierarchy is scanned top-down by a human expert to identify all classes which define new contexts. All subclass relationships from such identified classes to their superclasses will be refined as *role-of* relationships. The class **person**, since it is the unique root of this hierarchy, starts a new context. Let us call it the personal context, but it does not have any superclasses.

Working top-down, we can see that the classes **alumnus**, **student**, and **employee** are subclasses of the class **person**. The class **alumnus** describes the context of former students. The class **student** defines the learning context. The class **employee** starts the employment context. These three contexts are different

from the personal context defined by **person**. Thus, the three classes are considered to start three new contexts. Thus, we make them *role-of* their superclass **person**.

The other class which is considered to start a new context is the class **instructor**. It defines the teaching context. This context is different from the employment context, as it concentrates on a specific activity. Thus, it is identified as the root of the context teaching. Thus, we make **instructor** a *role-of* its superclass **employee**.

According to human expert judgement, there are no other classes in this hierarchy, which start new contexts. Altogether, five classes have been identified as root classes (see Figure 4.5).

Step 4 of our methodology inspects all the classes which have more than one superclass in bottom-up order (reversing the order of the topological sort in Figure 4.2). These classes are **dept. chairman**, **teaching assistant**, **academic admin**, and **assistant**. Since each class with multiple superclasses can have at most one *category-of* relationship, a human expert needs to identify at most one major superclass for each of those four classes in **Step 5** of our methodology. As we mentioned before, there is a possibility of ambiguity in choosing roots for new contexts. Such ambiguities exist in many modeling situations. One may argue that **academic admin** is also the root of a context. However, we will not chose this option.

The class **dept. chairman** has two superclasses, one is **professor** and the other is **academic admin**. The main function of a chairman is to lead the department. He also functions as a professor, for example in teaching a course, but this is a secondary function for him. Hence, by **Case 1**, the class **dept. chairman** belongs to the same context as **academic admin**. We make it *category-of* **academic admin** and *role-of* **professor**.

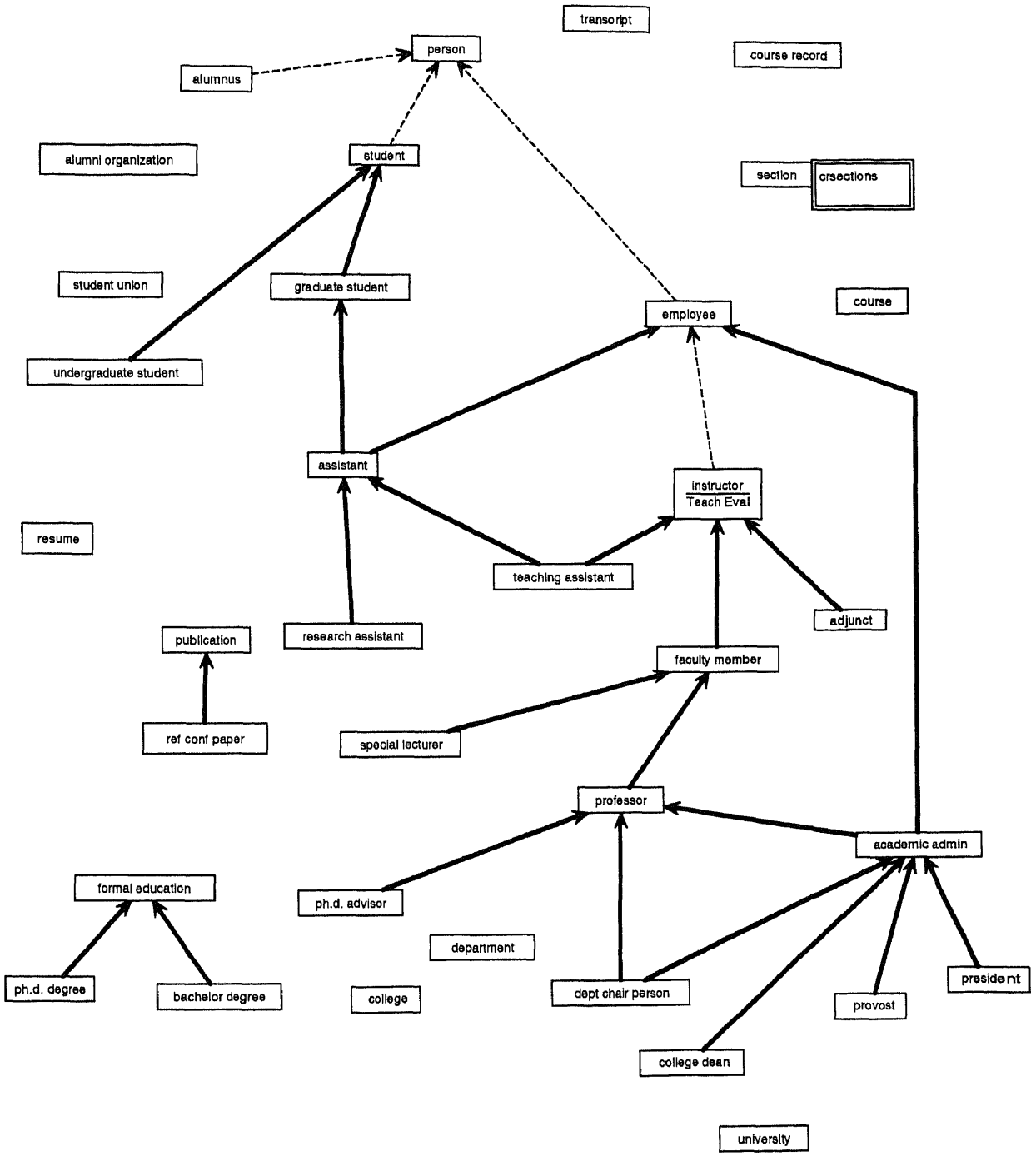

Figure 4.5 The subschema in Figure 4.2 after **step 3** of the methodology

The class **teaching assistant** has two superclasses **assistant** and **instructor**. Since **assistant** is a definitional superclass, while teaching is the functionality of a **teaching assistant**, by **Case 1**, **assistant** is chosen as the major superclass. As a matter of fact, we could come to the same conclusion using **Case 2**, as demonstrated above.

The class **academic admin** has two superclasses **professor** and **employee**. The purpose of having a professor's appointment for an academic administrator is to provide him with a tenured position for the case of his resignation as an administrator. Hence, by **Case 1** the superclass **employee** is the major superclass.

The class **assistant** has two superclasses, **grad student** and **employee**. Since **assistant** describes the student employment rather than his studies, it is in the same context as employee and is *category-of* **employee** and *role-of* **grad student**.

After applying steps 1-5 of the methodology, each class with multiple superclasses is *category-of* at most one superclass. Figure 4.6 shows the result of applying our methodology to the schema in Figure 4.1. *Role-of* links are marked by dashed lines.

The next two steps (**Step 6** and **7**) of our methodology deal with finding diamond structures and resolve contradictory cases. There are three diamond structures in Figure 4.6. They are <**dept chair person, employee**>, <**teaching assistant, employee**>, and <**assistant, person**>. None of these three diamond structures are contradictory cases. Thus, all subclass relationships in Figure 4.2 are refined as either *category-of* or *role-of* in Figure 4.6.

In **Step 8**, the forest hierarchy of the *category-of* relationships of Figure 4.6 is obtained by removing all *role-of* relationships. Figure 4.7 shows the different contexts of trees of the forest. To summarize, the hierarchy rooted at **person**, containing 19 classes, is partitioned into five trees, two of which, **person** and **alumnus**, contain only one class. The learning context contains three classes; the employment context

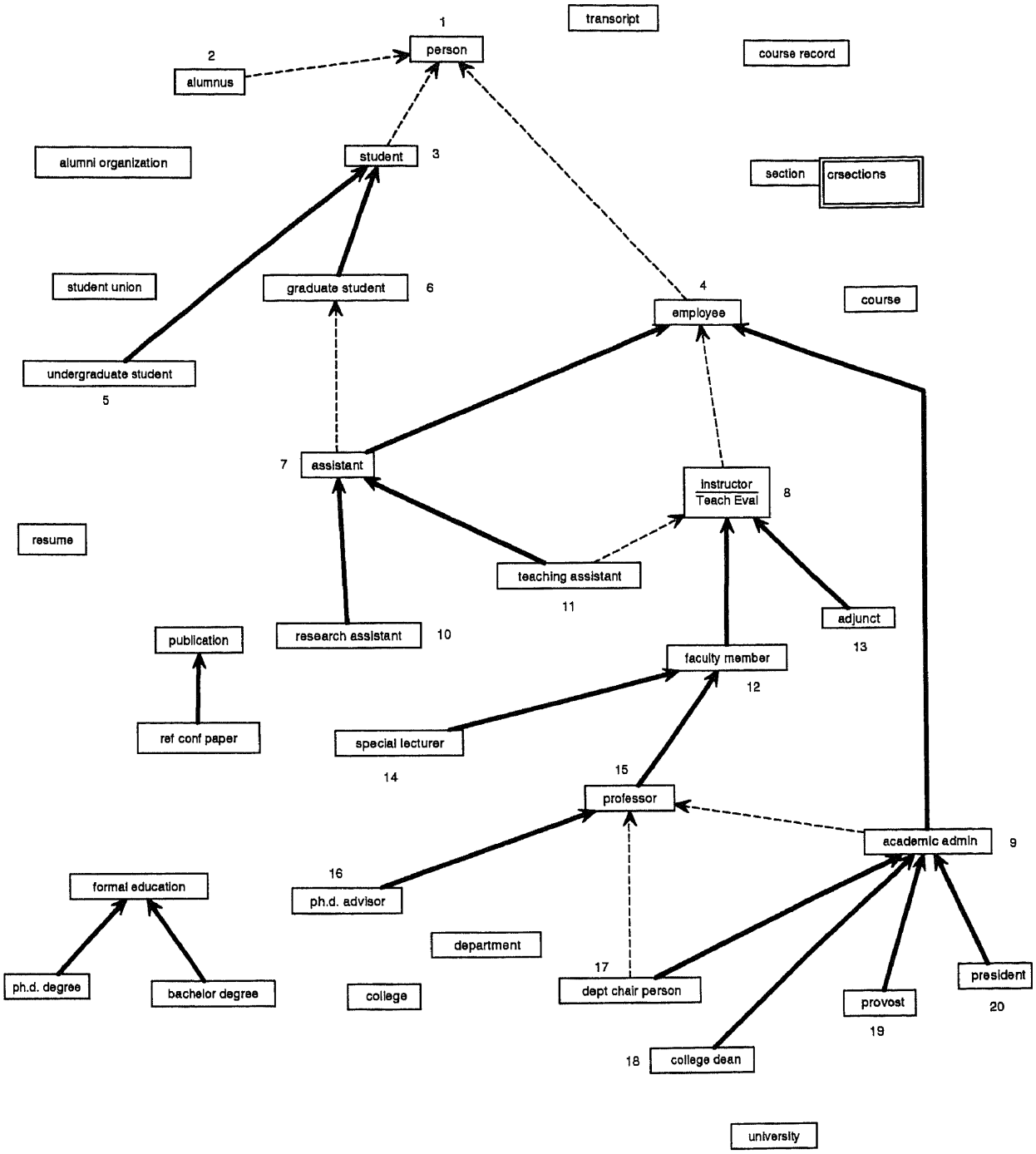

Figure 4.6 The result of applying the methodology to Figure 4.1

contains nine classes; and the teaching context contains six classes. Thus, Figure 4.7 gives a concise abstract representation of Figure 4.1. After achieving a comprehension of Figure 4.7, the user is ready to study the intricacies of the schema, by first comprehending the internal relationships inside each context and then considering the *role-of* relationships connecting different contexts as shown in Figure 4.6. To this aim, the user may pick one context at a time and study the relationships between classes of this context and classes of the other contexts.

4.7 Applying the Methodology to the MED Schema

In the last section, we applied our methodology to a subschema of the university database. In this section, we demonstrate our methodology by applying it to the larger and more complex MED schema. The MED was built at Columbia Presbyterian Medical Center [36, 37] and contains about 56,000 concepts. The MED schema consists of 124 classes and 190 subclass relationships. Figure 4.8 shows the schema after applying **Step 1** of our methodology. Since the schema in Figure 4.8 is too large and complex for displaying on one screen[2], we selected a subschema (see Figure 4.9) which contains 34 classes and 52 subclass relationships to demonstrate our methodology.

Step 2 of our methodology is topological sort. In Figure 4.9, all classes are numbered from 1 to 34. The order we used is left-to-right breadth first search.

In **Step 3**, following the topological sort ordering, the MED subschema is scanned top-down to identify all classes which define new contexts. All subclass relationships from identified classes to their superclasses are refined as *role-of*.

The class **MEDICAL ENTITY** (1) is the unique root of the MED subschema. It starts a new context. It does not have any superclass.

[2]If the reader felt that Figure 4.8 is unreadable, this is exactly proving our point.

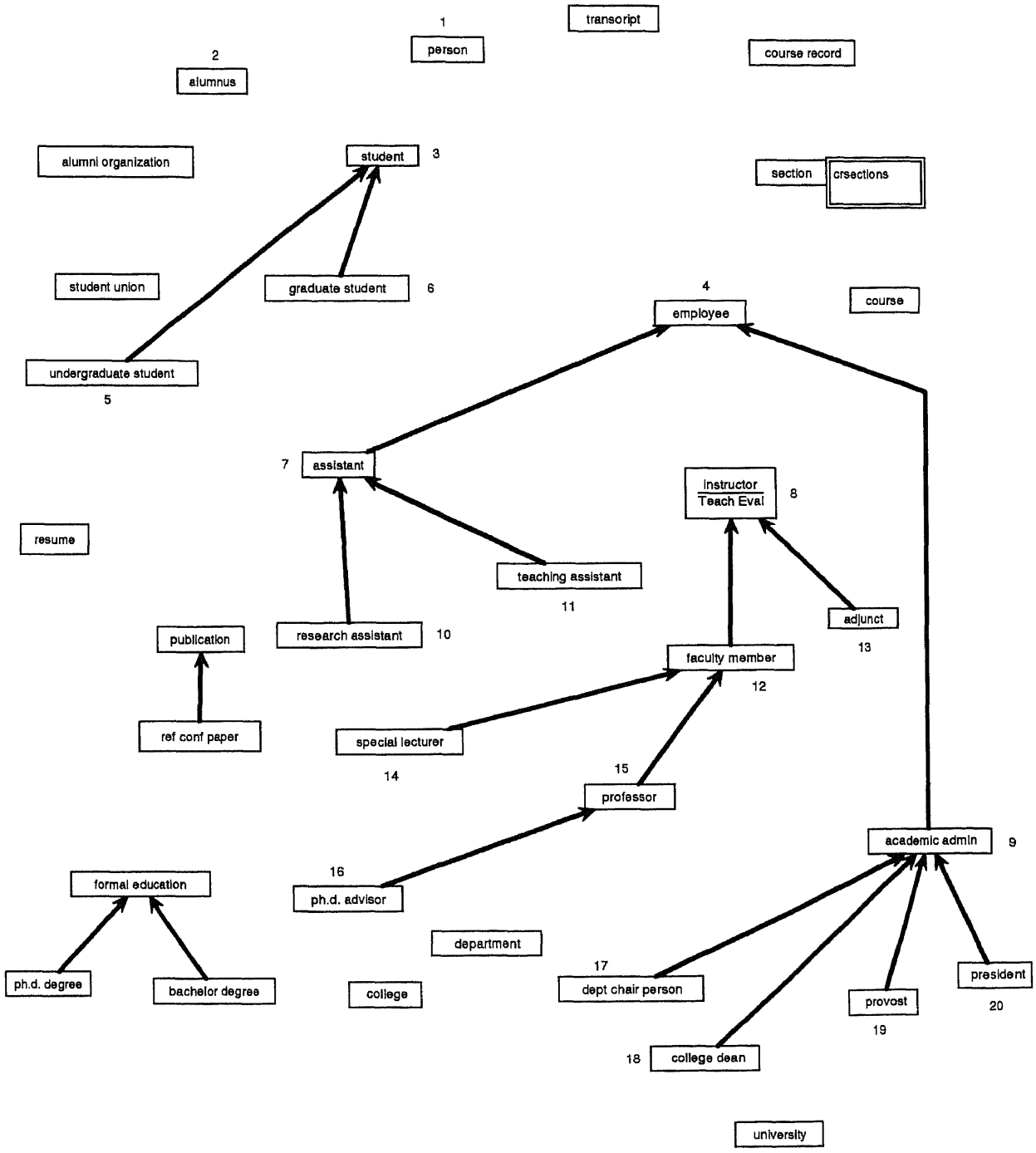

Figure 4.7 Removing all *role-of* from Figure 4.6

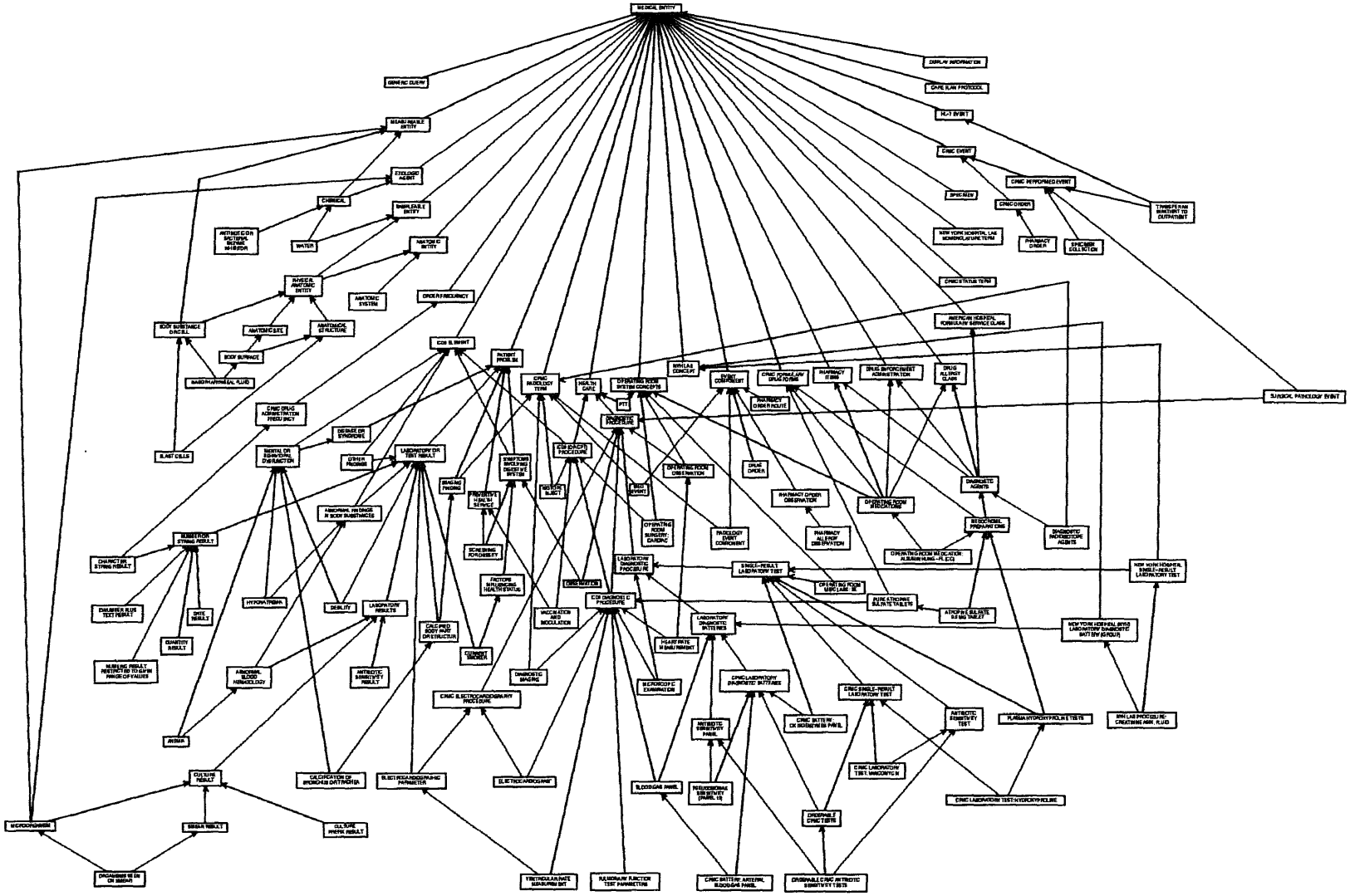

Figure 4.8 The MED schema

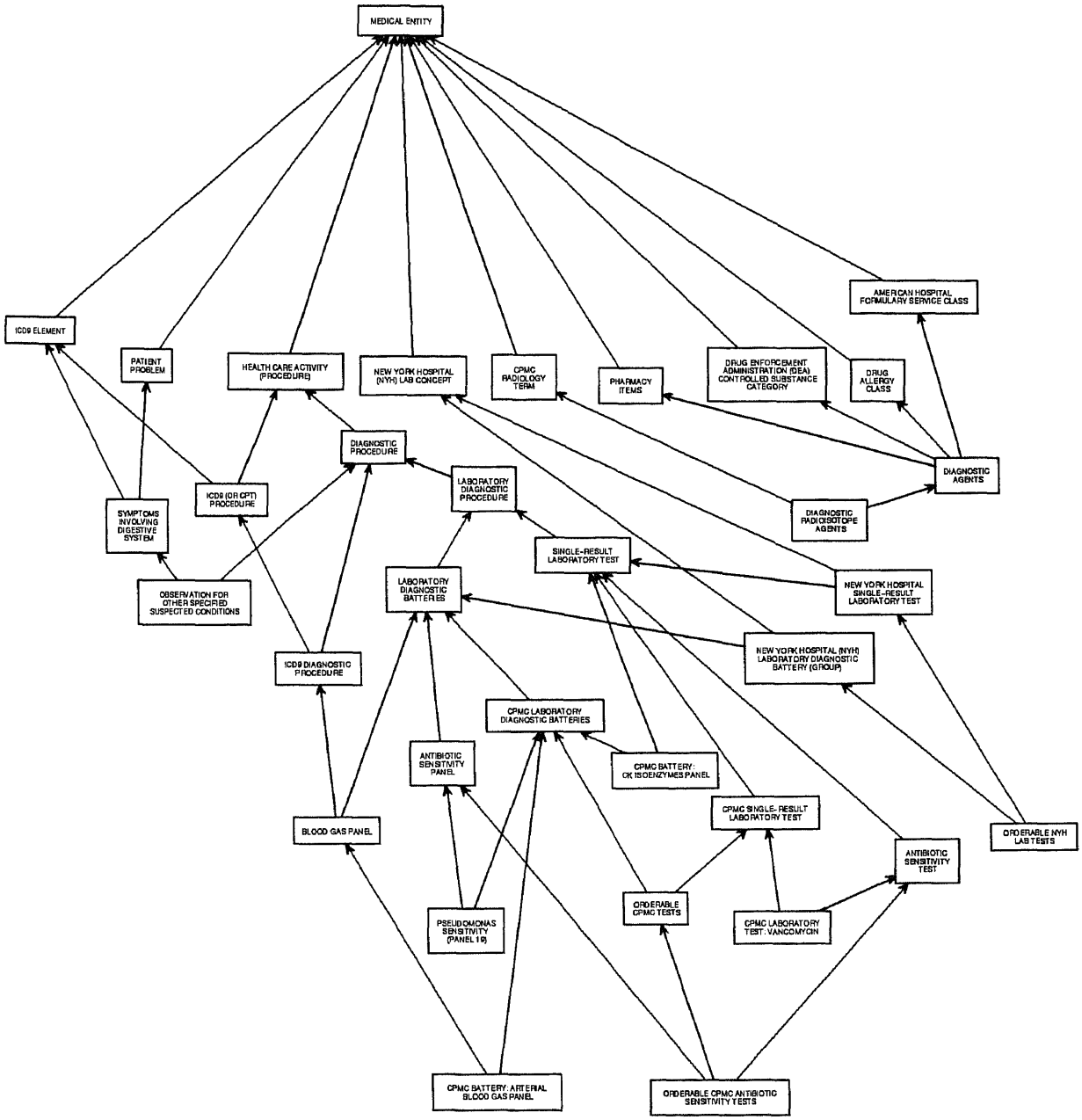

Figure 4.9 The MED subschema

The class **DIAGNOSTIC PROCEDURE** (14) which has one superclass **HEALTH CARE ACTIVITY (PROCEDURE)** (5) starts a new context. **HEALTH CARE ACTIVITY (PROCEDURE)** describes all the activities of a health care plan and **DIAGNOSTIC PROCEDURE** specifically focuses on the procedures which are for diagnostic purposes. Thus, **DIAGNOSTIC PROCEDURE** is a *role-of* its superclass **HEALTH CARE ACTIVITY (PROCEDURE)**.

The class **BLOOD GAS PANEL** (22) has two superclasses **ICD9 DIAGNOSTIC PROCEDURE** (19) and **LABORATORY DIAGNOSTIC BATTERIES** (23). **BLOOD GAS PANEL** refers the tests of concentrations of various gases, for example, oxygen and carbon dioxide in blood samples. Blood oxygen and carbon dioxide are controlled by adjusting the rate and depth of ventilation and respiration. Both are factors affecting the acid-base balance of blood. However, **ICD9 DIAGNOSTIC PROCEDURE** and **LABORATORY DIAGNOSTIC BATTERIES** both describe general diagnostic procedures. Thus, the class **BLOOD GAS PANEL** defines a new context and is a *role-of* its superclasses **ICD9 DIAGNOSTIC PROCEDURE** and **LABORATORY DIAGNOSTIC BATTERIES**.

In **Step 3** of our methodology, 12 classes, including the root class **Medical Entity**, are chosen to define new contexts. Figure 4.10 shows all these classes except the root class as *role-of* of their superclasses.

In **Step 4** of our methodology, all classes which have more than one non *role-of* superclass are identified in a bottom-up order (reversing the order of the topological sort in Figure 4.9). There are 13 classes with multiple non-*role-of* relationships to their superclasses in Figure 4.10. They are (34), (33), (32), (31), (25), (21), (19), (18), (16), (15), (13), (12), and (11).

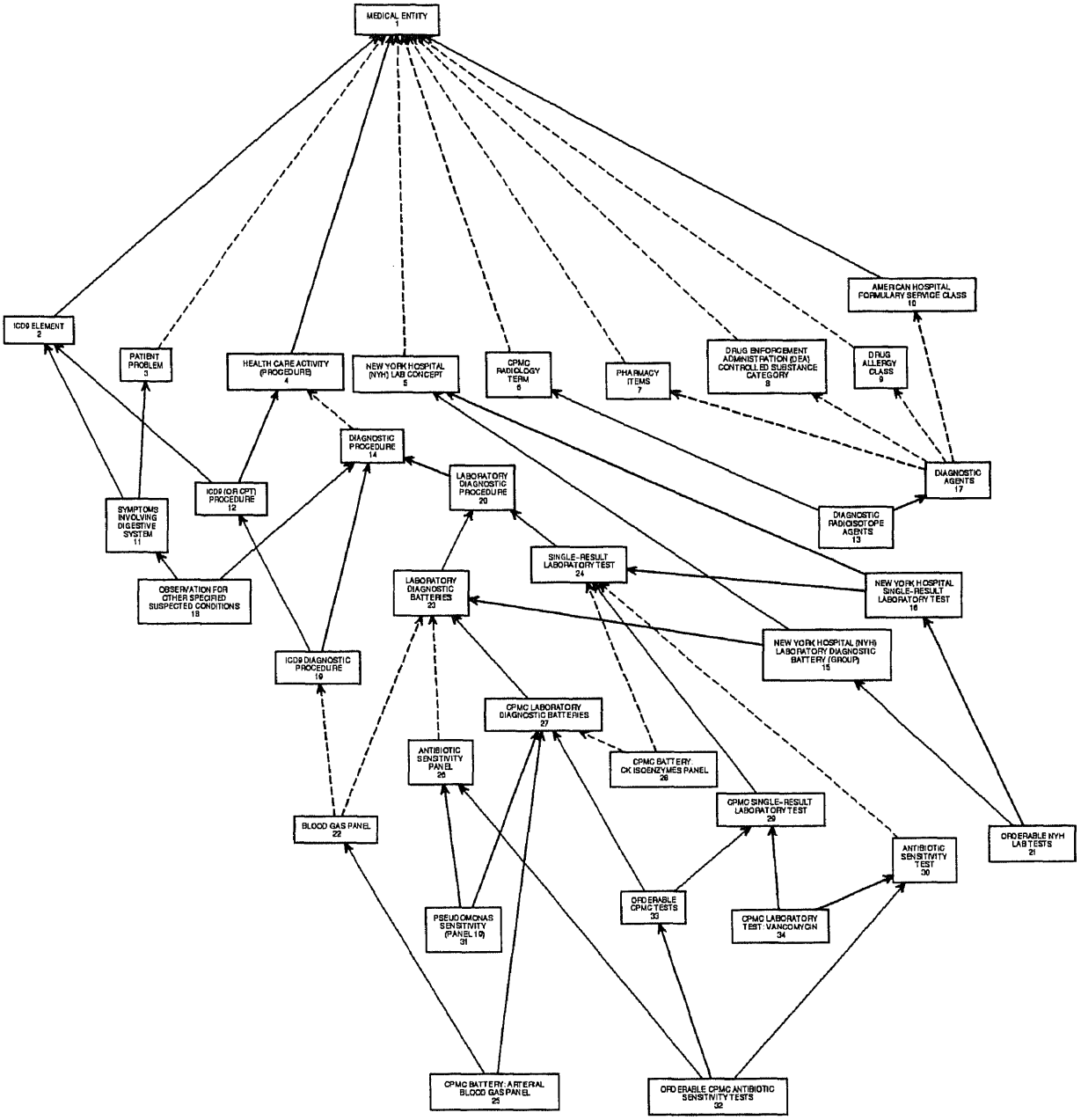

Figure 4.10 The subschema corresponding to Figure 4.9 after **Step 3**

Step 5 of our methodology identifies at most one major superclass for each of those 13 classes found in **Step 4**. Let us present some examples to demonstrate this step.

The class **CPMC BATTERY: PSEUDOMONAS SENSITIVITY (PANEL 19)** (31) has two superclasses **ANTIBIOTIC SENSITIVITY PANEL** (26) and **CPMC LABORATORY DIAGNOSTIC BATTERIES** (27). **CPMC BATTERY: PSEUDOMONAS SENSITIVITY (PANEL 19)** (31) is one of the antibiotic sensitivity panel tests and is used to evaluate the anti-drugs activity of cultured Pseudomonas cosampled from a patient. Its superclass **ANTIBIOTIC SENSITIVITY PANEL** (26) which defines all antibiotic sensitivity panel tests is a definitional superclass of **CPMC BATTERY: PSEUDOMONAS SENSITIVITY (PANEL 19)**. The other superclass **CPMC LABORATORY DIAGNOSTIC BATTERIES** (27) defines CPMC laboratory procedures for diagnostic purposes. According to case 1 of **Step 5**, **ANTIBIOTIC SENSITIVITY PANEL** (26) is the major superclass of **CPMC BATTERY: PSEUDOMONAS SENSITIVITY (PANEL 19)** (31). Thus, **CPMC BATTERY: PSEUDOMONAS SENSITIVITY (PANEL 19)** (31) is a *category-of* **ANTIBIOTIC SENSITIVITY PANEL** (26) and a *role-of* **CPMC LABORATORY DIAGNOSTIC BATTERIES** (27).

Another example is the class **CPMC BATTERY: ARTERIAL BLOOD GAS PANEL** (25) which has two superclasses **BLOOD GAS PANEL** (22) and **CPMC LABORATORY DIAGNOSTIC BATTERIES** (27). Since **BLOOD GAS PANEL** (22) defines various kinds of blood gas tests including the gas tests of arterial blood, it is the definitional superclass of **CPMC BATTERY: ARTERIAL BLOOD GAS PANEL** (25). Thus, **CPMC BATTERY: ARTERIAL BLOOD GAS PANEL** (25) is a *category-of* **BLOOD GAS PANEL** (22) and a *role-of* **CPMC LABORATORY DIAGNOSTIC BATTERIES** (27).

The class **ICD9 DIAGNOSTIC PROCEDURE** (19) has two definitional superclasses **DIAGNOSTIC PROCEDURE** (14) and **ICD9 (OR CPT) PROCEDURE** (12). Since both superclasses contribute with equal importance to the class **ICD9 DIAGNOSTIC PROCEDURE** (19), we cannot prefer one over the other. (For different viewpoints, each one is playing a major role.) Hence, the class **ICD9 DIAGNOSTIC PROCEDURE** (19) is a *role-of* both its superclasses. This is a case of a *role-of/intersection*, denoted r/i in Figure 4.11. Figure 4.11 shows the subschema after identifying the major superclass for each class listed in **Step 4**.

In **Steps 6** and **7**, all diamond structures are identified and checked whether they are contradictory diamond structures. As we described in **Step 7** of our methodology, only diamond structures containing *role-of/intersection* need to be checked. The diamond structure <**ICD9 DIAGNOSTIC PROCEDURE** (19), **HEALTH CARE ACTIVITY** (5)> is the only one which contains *role-of/intersection* (Figure 4.11). However, it is not a contradictory diamond structure.

In **Step 8** the forest hierarchy is obtained by removing all *role-of* relationships from Figure 4.11. Figure 4.12 shows the resulting partition of the subschema in Figure 4.9 into 14 trees. The whole MED schema in Figure 4.8 is partitioned into 51 trees.

4.8 Utilizing the Forest Structure for Schema Comprehension

The partition into contexts results in a high level (macro) view of the application. Each tree in the forest represents a micro view of one context. The relationships of a schema can be divided into two kinds, intra-relationships and inter-relationships, as follows. For every context there is a set of relationships completely contained in the context, called *intra-relationships*. *Role-of* relationships which connect different contexts and user-defined relationships connecting classes of two different contexts are called *inter-relationships*.

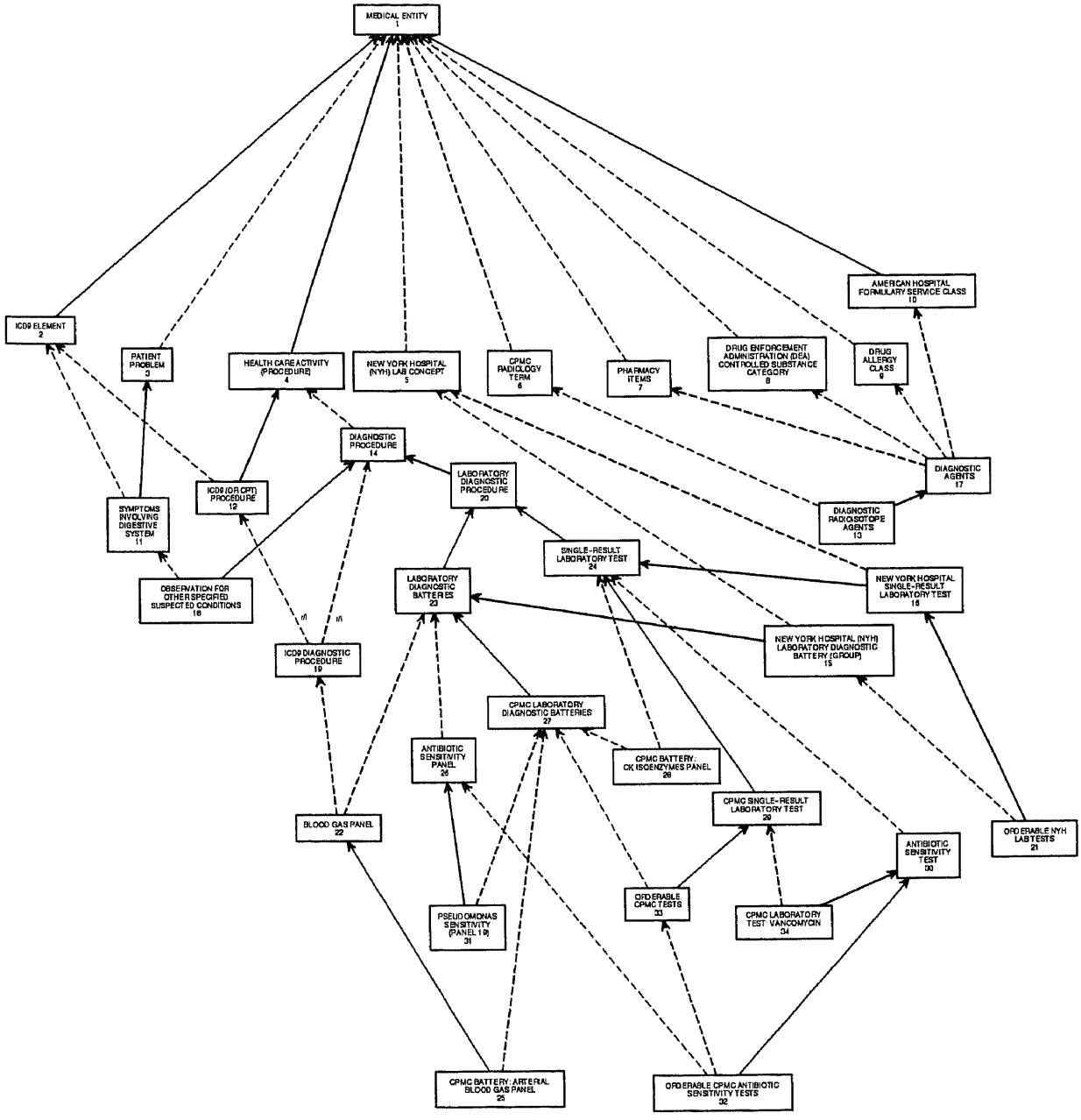

Figure 4.11 The MED subschema in Figure 4.9 after applying **Step 4** of the methodology.

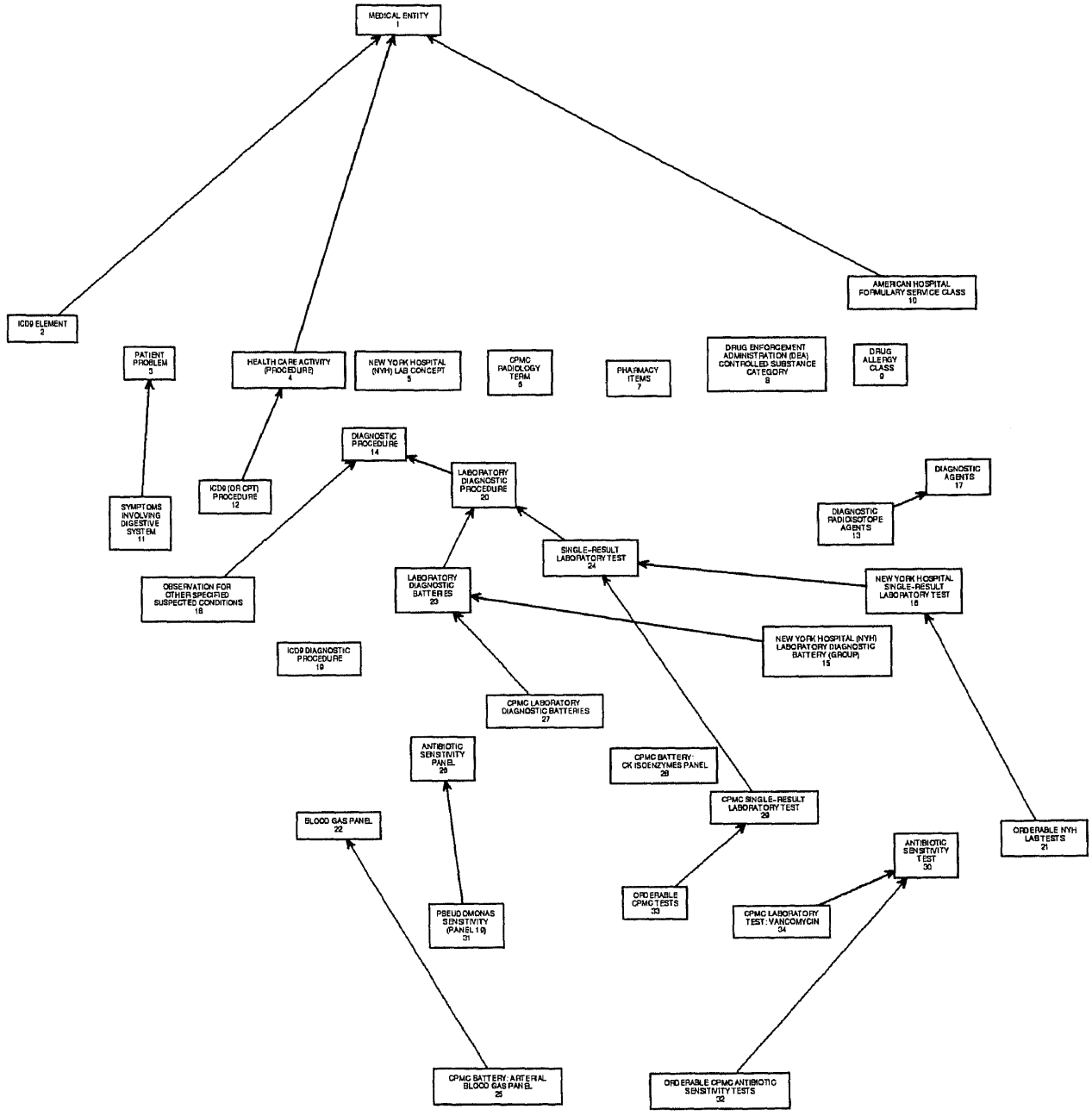

Figure 4.12 A forest structure of the MED subschema in Figure 4.9 after applying the methodology.

To comprehend a complex schema, a user can first pick one context and study the intra-relationships in this context. Then, he can study the inter-relationships between two contexts at a time. In this way, we divide the task of studying all relationships into a number of smaller, well organized and more manageable tasks. This process simplifies the comprehension process, due to the reduction in the complexities of the subschemas versus the complexity of the original schema. Consider, for example, a schema containing k contexts, with n classes in each context. Suppose further that the number of intra-relationships within each context is αn and the number of inter-relationships of each pair of contexts is βn. Then the complexity of the schema is

$$\mathbf{c} = \frac{k\alpha n + \frac{k(k-1)}{2}\beta n}{kn} = \alpha + \frac{k-1}{2}\beta \tag{4.1}$$

On the other hand, when considering only a subschema of two contexts, the complexity is

$$\mathbf{c} = \frac{2\alpha n + \beta n}{2n} = \alpha + \frac{\beta}{2} \tag{4.2}$$

Hence, the complexity of considering only the inter-relationships between two contexts is smaller by a factor of $k-1$, compared to the whole schema. Thus, each comprehension task is less complex than the task of comprehending the schema as a whole.

4.9 Summary

In this chapter, we presented both a theoretical paradigm and a methodology to identify a meaningful forest subschema of a given OODB schema. The extraction of the forest subschema employs two approaches, *informational thinning* and *partitioning*. We developed three rules which express limitations and refinements to the modeling of the schema. Based on these three rules, a new technique for modeling called *disciplined modeling* was introduced and a theorem of the existence of a

forest subschema whose trees represent logical units was proved. A human-computer interactive methodology was developed for finding a forest subschema, based on the theoretical paradigm. Such a forest subschema functions as a skeleton of the original schema and supports comprehension efforts. The methodology was applied to a subschema of a university database and the MED schema. 19 classes of the subschema of a university database were partitioned into five trees. The MED schema with 124 classes was divided into 51 trees.

CHAPTER 5

A METHODOLOGY FOR PARTITIONING A VOCABULARY HIERARCHY INTO TREES

5.1 Introduction

Controlled medical vocabularies ("vocabularies" for short) play an important role in many medical enterprises that employ a large number of disparate information systems (e.g., clinical databases). Often, each such system has its own inherent "language" or terminology. A number of such vocabularies have appeared in the medical field. Of note is the Medical Entities Dictionary (MED) developed and in use at Columbia-Presbyterian Medical Center (CPMC) [36, 37]. Controlled vocabularies have been shown to greatly facilitate the process of integrating medical information systems [38] using different terminologies. They also help to standardize common information handling tasks and reduce the overall cost of data processing.

In previous chapters, we presented using OODB modeling and schema partitioning to enhance comprehension of controlled medical terminologies. An OODB schema captures the complete structure of the vocabulary in a compact form which aids in its comprehension. However, for the much larger vocabularies, the number of instances of a class in the corresponding OODB schema can be large. For example, each class in MED schema summarizes on average 500 concepts. A vocabulary of 500 concepts is still hard to understand. Thus, further partitioning efforts are needed to enhance comprehension.

In this chapter, we are concerned with providing a tool to help users comprehend vocabularies. In particular, we present a methodology to make large and complex vocabularies easier to understand. Our approach is based on the partitioning of a vocabulary into *manageably-sized, meaningful* units. The partitioning assumes the existence of a vocabulary with an IS-A hierarchy and centers around this IS-A (or concept subsumption) hierarchy.

The backbone of many controlled vocabularies is the IS-A hierarchy which relates more specialized concepts (subconcepts) to more generalized concepts (superconcepts) that subsume them. The IS-A hierarchy also serves as the basis for property inheritance. In general, the IS-A hierarchy of a controlled vocabulary will be a directed acyclic graph, permitting multiple superconcepts and multiple inheritance. Our methodology is based on the following two premises: (1) A vocabulary's IS-A hierarchy taken alone is much more comprehensible than the entire vocabulary itself; (2) A "forest" IS-A hierarchy (i.e., a collection of trees in which every link is an IS-A link and where, by definition, no concept has more than one superconcept) is easier to comprehend than a directed acyclic graph containing the same number of concepts.

With these premises in mind, we develop a theoretical framework that reduces an entire vocabulary (typically represented as a large semantic network) into a forest hierarchy composed of small trees, each representing a logical unit whose graphical representation can fit on a computer screen. This reduction in size makes it easier for users and system designers alike to comprehend the contents of the vocabulary in a modular fashion.

Our methodology relies on an interaction between a user (presumably the vocabulary designer or administrator) and the computer. The process requires that a user refines the vocabulary's IS-A hierarchy according to some prescribed principles so that it conforms to what we call the rules of *disciplined modeling*. After the refinement, the computer can automatically reduce the vocabulary to a forest structure. We formally prove that our approach always finds a forest partition as long as the rules of disciplined modeling are adhered to. Let us note that partitioning networks (graphs) according to various criteria has been shown to be NP-complete, i.e., computationally intractable [50].

In Chapter 4, we have employed a similar paradigm to reduce the complexity of large object-oriented database (OODB) subclass hierarchies. In this chapter,

we rework and adapt the approach to the IS-A hierarchy of an extensive, complex vocabulary. To ground our discussion in a real-world application, we will focus on the MED as our test-bed vocabulary. The methodology developed herein will be applied to a complex subnetwork of the MED.

Our approach is closely related to the principle of 'orthogonal taxonomies' as implemented in the GALEN project [120, 121]. There, a taxonomy is organized from the start by requiring that all primitive entities have only one primitive parent. In our methodology, an existing vocabulary is partitioned to achieve a similar effect.

The rest of this chapter is organized as follows. In Section 5.2, we describe the notions of informational thinning and partitioning with respect to vocabularies. Section 5.3 introduces the rules of disciplined modeling and proves that they make it possible to obtain a meaningful forest hierarchy from a directed acyclic graph. In Section 5.4, we describe our methodology for partitioning the vocabulary. In Section 5.5, we apply the methodology to a very complex portion of the MED. Section 5.6 contains the summary.

5.2 Informational Thinning and Partitioning

In this section, we describe two approaches which are used to enhance the comprehension of large and complex vocabularies. If a vocabulary network, containing a vast amount of objects (representing concepts), relationships and attributes, is displayed on a screen, then the user typically has difficulties comprehending and dealing with it. For such an overwhelming display of the InterMED, see [116].

According to our experience, the difficulties of understanding a vocabulary stem more from the number of relationships than from the number of concepts. We define the *complexity* c of a network or vocabulary as the ratio of the number of relationships between objects to the number of objects. As we mentioned, the MED is an example of a large, complex vocabulary. The complexity of the MED is $c =$

$(61000 + 71000)/48000 \approx 2.75$. For two networks with the same number of objects, the more complex network is more difficult to comprehend. Thus, there exists a need to reduce the number of relationships in order to display a simplified comprehensible subnetwork of the vocabulary with a lower complexity. Informational thinning is used to achieve this goal.

Definition 1: *Informational thinning* is a technique for eliminating partial information from the display of a whole network. This is done by prioritizing various *kinds* of properties of the objects in the network and displaying only *kinds* of properties with high priority.

In our graphical OODB schema editor OODINI [66], we support two levels of informational thinning. One level removes all attributes and the other removes attributes *and* non-hierarchical relationships leaving only the IS-A hierarchy displayed. The latter level of informational thinning will be used in the figures of this chapter. The hierarchy of IS-A relationships is the backbone of a vocabulary, which helps users to comprehend it. The use of informational thinning (level 2) permits us to concentrate on the IS-A hierarchy.

To test our theoretical paradigm and methodology we looked for a subnetwork of the MED with a very complex hierarchy. Our reasoning is that for our techniques to be applicable for the whole MED vocabulary, it is necessary, although not sufficient, to be successfully applicable to such a subnetwork. We identified a subnetwork with a very complex hierarchy in the MED as follows. From the 48,000 concepts in the MED, the concept **CPMC Drug: Cortisporin Opthalmic Ointment** has the most ancestors, 39. The subnetwork with a complex hierarchy, which we call cortisporin subnetwork, includes the concept **CPMC Drug: Cortisporin Opthalmic Ointment** and all its ancestors. It contains 821 attributes, 62 IS-A relationships and 157 other relationships. Thus, the *complexity*

of cortisporin subnetwork is $c = (62 + 157)/(39 + 1) \approx 5.5$. Such a complex network with so many properties cannot be displayed on one screen.

In Figure 5.1, we show the hierarchy of IS-A relationships of cortisporin subnetwork. This hierarchy has the same number of concepts as the original network but fewer relationships. The complexity of the IS-A hierarchy of cortisporin subnetwork is $c = 62/40 \approx 1.55$, a much lower complexity than that of cortisporin subnetwork itself. For comparison, the complexity of the IS-A hierarchy of the whole MED is $c = 54547/42744 \approx 1.27$ which is lower than that of cortisporin subnetwork. To help in the forthcoming analysis, we added in Figure 5.1 some concepts which are not the ancestors of the concept **CPMC: Cortisporin Opthalmic Ointment**. The added concepts are (33), (34), (39), (41), (43), and (45).

Obviously, the use of informational thinning makes it easier to understand a vocabulary. But comprehending a large and complex IS-A hierarchy may still be very difficult, although informational thinning was applied, due to multiple inheritance and the large number of objects. Considering the limitations of human comprehension capacity and the size of computer monitors (e.g., a network of less than 20 objects can easily fit on one computer screen), we will provide a set of less complicated and smaller subhierarchies of the original IS-A hierarchy to simplify the comprehension process. To realize this target, another approach, partitioning, is introduced.

Definition 2: *Partitioning* means to divide a complex, large semantic network into disjoint smaller subnetworks which comprise logical units of the original network and jointly constitute the original network.

To partition a graph into logical units, it is necessary to comprehend it first. Thus, the logical partitioning of a network seemingly results in a vicious cycle. Our experience has been that partitioning into logical units tends to minimize the number of relationships between different units. Unfortunately, the problem of partitioning a

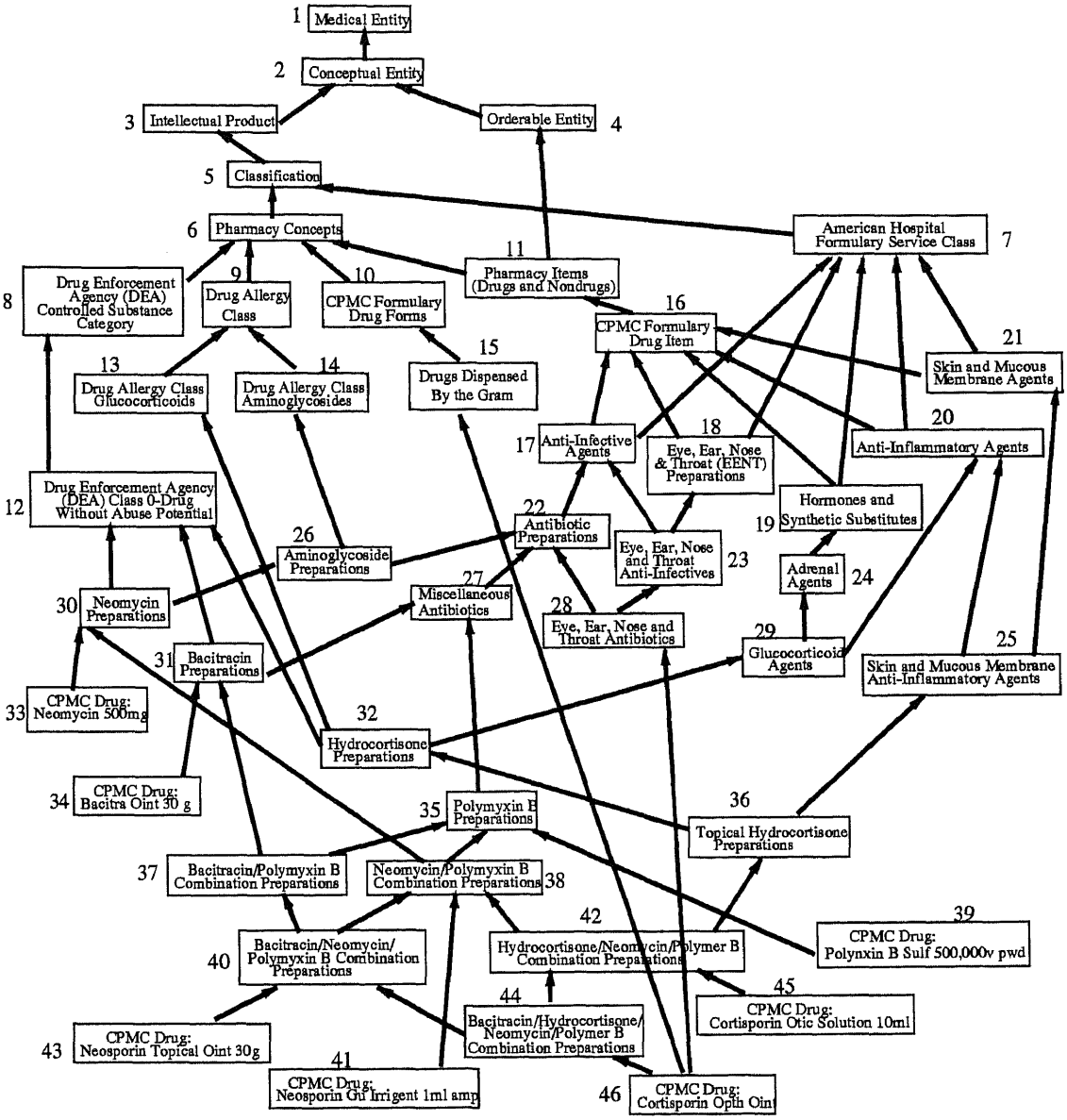

Figure 5.1 A complex subhierarchy in the MED with topological sort order after informational thinning

network according to the above or similar criteria is NP-complete, that is, no efficient algorithm is known for it, and it is conjectured that no such algorithm exists [50]. A possible line of action is to combine informational thinning and partitioning. After an IS-A hierarchy is obtained by applying informational thinning to the original network, the partitioning technique is put to use by partitioning the IS-A hierarchy and then imposing this partition on the original network.

Due to multiple inheritance, the IS-A hierarchy forms a directed acyclic graph, just like our cortisporin subhierarchy shown in Figure 5.1. If the IS-A hierarchy is a directed acyclic graph, its partitioning problem in general is still NP-complete. On the other hand, if it is a tree, then there exist efficient algorithms for various partitioning criteria, e.g., max-min or min-max [3, 9, 10, 11, 81, 92, 118]. In the next section, to make the partitioning possible, we will present a new technique for modeling called disciplined modeling. Based on the rules of *disciplined modeling* we develop a theoretical paradigm and methodology to identify a meaningful forest subhierarchy within the IS-A hierarchy. If the trees in the forest hierarchy are still too large, the above mentioned efficient partitioning algorithm may be applied to them, to yield smaller trees.

5.3 Theoretical Paradigm Using Disciplined Modeling

In order to identify a meaningful forest subhierarchy of an IS-A hierarchy, we shall look into the nature of the specialization IS-A relationship. In previous OODB research [51], we and others [111] have identified two different kinds of SUBCLASS relationships between object classes, called *category-of* and *role-of*. Both are specialization relationships. *Category-of* relates the specialized class to the more general class where both are seen in the same application context. *Role-of* relates the specialized class to the more general class where the two classes are in different contexts of the application.

In [117] we presented a theoretical paradigm for partitioning of an OODB hierarchy schema. However, modifying the theoretical paradigm from the class level [117] to the instance level requires careful examination. One issue is how to interpret *category-of* and *role-of* at the instance level. Naturally, these relationships between classes imply *category-of* and *role-of* relationships between objects which are instances of the corresponding subclass and superclass. Similarly, they can be defined between objects of a semantic network as follows.

Definition 1: *Category-of* is a specialization relationship which relates the specialized object to the more general object where both are seen in the same application context.

Definition 2: *Role-of* is a specialization relationship which relates the specialized object to the more general object where the two objects are in different contexts of the application.

For example, **Aminoglycoside Preparations** is *category-of* **Antibiotic Preparations** and **Neomycin preparations** is *category-of* **Aminoglycoside Preparations**, because all of them are in the same application context "Anti-Infective Agents." On the other hand, **Neomycin preparations** is *role-of* **Drug Enforcement Agency (DEA) Class 0-Drug Without Abuse Potential** in the context of **Drug Enforcement Agency (DEA) Controlled Substance Category** (see Figure 5.6 and 5.7).

A second issue is that in [117] we discussed a relation "represents the same real-world object" between instances of classes. However, in a semantic network-based vocabulary, objects describe general concepts rather than concrete, real-world objects. Therefore, we need to find an alternative for the relation "represents the same real-world object" to be employed in the necessary proof for the vocabulary environment. The impact of this difference on the development of our theoretical

paradigm has to be inspected. An adapted proof technique is presented later in this section.

The decision whether a given IS-A relationship in the hierarchy is either a *category-of* or a *role-of* depends on whether the superobject and object are in the same context or not. An intuitive understanding of the application is required to help make this decision. However, this decision is not always so easy. In spite of extensive research, [22, 23, 60, 74, 94, ?, 128], there is still no widely accepted definition of context. Building a gigantic knowledge base in the CYC project [86] was found doomed to failure if contexts were not introduced as a structuring mechanism. Following the research of [22, 23], others have assumed that a context is a first-class object used to parameterize axiom schemata [60, 94]. However, no clarity about the nature of contexts themselves is gained by this approach. As a workshop on context in Natural Language Processing showed [74], researchers tend to agree that they disagree on what contexts are. Our approach is that we are not trying to define the notion of context. Rather we are making the pretheoretical (axiomatic) assumption that contexts exist in human thinking, and we are requiring the designers and users of an application to identify them explicitly.

In [117], we provided a theoretical paradigm for the existence of such assignments of classes to contexts. This assignment results in a forest subhierarchy of a directed acyclic graph hierarchy, which supports increased comprehension of the OODB schema. The theoretical paradigm is supported by three rules of disciplined modeling which ensure that a forest subhierarchy can be identified. However, while in [117], disciplined modeling was described for a schema of classes, we modify it now for a hierarchy of objects. This modification will provide us with a theoretical paradigm for partitioning a large complex hierarchy of objects into small parts, each of which has a tree structure. For further explanations and motivations on disciplined modeling beyond the material in this section, see [117].

Before we give the rules of disciplined modeling, we define the mathematical relation *equicontext*, or "in the same context," between objects. A pair (a, b) of two objects belongs to the equicontext relation if both objects a and b belong to the same context.

Rule 1: The equicontext relation between objects is an equivalence relation satisfying three conditions of reflexivity, symmetry and transitivity. Thus it partitions all objects of a network into disjoint contexts.

Rule 1 forces the designer into explicit specification of the contexts in his hierarchy and leads him to resolve some ambiguous situations. We do not claim to have a unique way of assigning objects to contexts. As we are dealing with a problem of data modeling, there are usually different ways to model the same real-world environment. We further do not claim that contexts are naturally disjoint. To the contrary, in many applications, initial contexts may overlap. However, disciplined modeling forces the modeler to design disjoint contexts, leading to the desired partitioning.

Rule 2: Two objects which are *category-of* specializations of a superobject cannot have a common *category-of* descendant object, and one cannot be a *category-of* descendant of the other.

According to our definition, the *category-of* relationship is used for refinement in the case where the superobject and the object are in the same context. **Rule 2** guarantees that when we refine a concept represented by an object into several subconcepts in the same context, we achieve a partition into mutually exclusive concepts.

In the next section, we discuss techniques how to specify IS-A relationships as *category-of* or *role-of* in different cases in a way which satisfies **Rule 2**. Examples are provided in Section 5.5.

Rule 3: For each context there exists one object which is the *major* (or defining) object for this context such that every object in this context is a descendant of this object.

This means that each context has only one object which is a "root" for it, i.e., there is a directed path of *category-of* relationships from each object in the context to this root object. Note that we use here the notion of a directed tree where all the directions are towards the root. In graph theory terms, the root is a sink.

We note that sometimes in a semantic network the designer would like to group a subnetwork which has several roots, rather than one, together into one context. In such a case, the designer can add an extra object and make these original roots children of the extra object. The new root will be named to reflect the "meaning" of its context. For example, there are many terms in the MED for procedures which doctors order. Thus, these terms are grouped into one context "procedure." There are also many terms in the MED for tests grouped into a context "tests." Tests are typically ordered as components of procedures. However, in some cases, a component can be ordered by itself and such a test therefore has the properties of a procedure (e.g., an order code, a cost, etc.). Thus, all tests which can not be ordered separately will reside in the "test" context. All other tests which can be ordered separately will be grouped into another context "orderable test," because all of them have properties of tests and procedures at the same time. The context "orderable test" has many roots. It has been helpful to introduce a new object, **Orderable Test**, as the root of this context to keep track of those tests. All those tests become children of **Orderable Test** (see Figure 5.2).

In a previous chapter, we already proved the following theorem: "Using disciplined modeling, a *class* has at most one *category-of* superclass." Now, we need to prove the corresponding theorem for the object level.

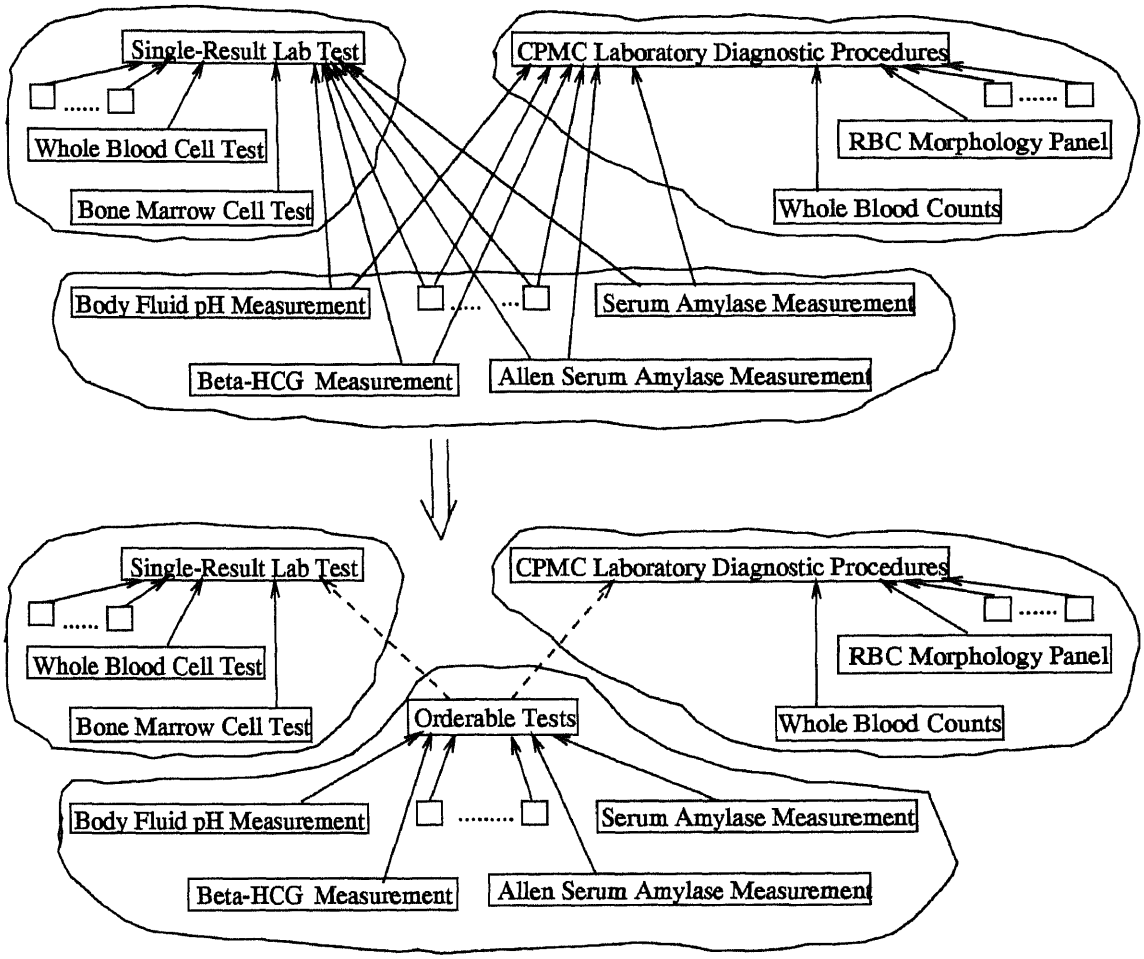

Figure 5.2 Adding new object as the root of a context

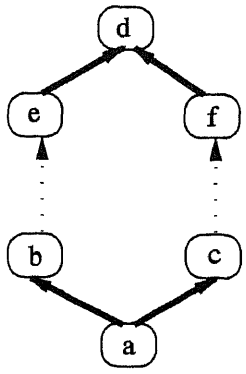

Figure 5.3 The hierarchies demonstrating our proof

Theorem: Using disciplined modeling, an *object* has at most one *category-of* superobject.

Proof: Assume to the contrary that there exists an object a which has two *category-of* superobjects b and c (see Figure 5.3). According to the definition of *category-of*, a and b are in the same context. Similarly, a and c are in the same context. By the transitivity of the equicontext relation (**Rule 1**), b and c are in the same context.

By **Rule 3**, there is a major (root) object d for this context such that the objects b and c are *category-of* descendants of d. This implies that there is a sequence of *category-of* relationships from b (c) up to d. (Note that actually d may be one of the objects b or c. This case does not cause a problem due to the second possibility in **Rule 2**. However, we avoid referring to this option in the rest of the proof to avoid complication of the presentation.) If the paths of *category-of* relationships from b to d and from c to d are not disjoint (i.e. the object d is not the first object which appears in both paths), then denote now by d the first such joint object on these two paths. Let e (f) be a subobject of the object d on a path of *category-of* relationships from b (c) to d. Hence, object a is a *category-of* descendant of object e (f). Thus, both the *category-of* subobjects e and f of the object d have a common *category-of* descendant object a. But by **Rule 2**, such a situation is forbidden, a contradiction.

■

Due to this theorem, we can guarantee that the *category-of* hierarchy has a forest structure which contains one or more trees. This forest structure serves as backbone of the hierarchy and will be critical in the efforts to comprehend the hierarchy and partition it into manageable subhierarchies.

5.4 A Methodology for Context Partitioning of a Hierarchy

We have described a conceptual framework which guarantees that for the price of following the rules of disciplined modeling, there can be found a forest-structure subhierarchy of the given directed acyclic graph hierarchy. This forest structure serves as a skeleton supporting the comprehension of the hierarchy. Furthermore, the trees of the forest represent contexts which are logical subhierarchies concentrating on a specific subject area, further supporting the comprehension of the original hierarchy.

In this section, we will describe a methodology to transform an existing hierarchy which was not designed according to the rules of disciplined modeling. By a methodology we mean a process that involves human-machine cooperation. The human domain expert is called upon to make some judgement decisions based on his understanding of the application while the computer supports the human by providing results of algorithmic procedures for tasks which do not involve complex intuitive decisions but might require many computational steps. By domain expert judgement we refer to:

1. Identifying disjoint contexts in the hierarchy, which correspond to subtrees of *category-of* relationships in the forest structure obtained.

2. Defining some IS-A relationships as *role-of* and others as *category-of*, so that the rules of disciplined modeling are followed.

In the following description of the methodology, we will specify which parts are performed by a computer and which are performed by a human expert. We will differentiate between three kinds of *role-of* relationships. They are *regular role-of*, *role-of/intersection* and *role-of/category-of*. However, for partitioning purposes, they are all just *role-of*. The result of our methodology is a refinement of the IS-A hierarchy. Every IS-A link becomes either a *category-of* or a *role-of*. For the purpose of partitioning, the *category-of* links will form a forest.

Step 1: Topological sort. (Computer)

Arrange the hierarchy in topological sort order.

Step 2: Identify roots of contexts. (Human)

Scan the hierarchy top-down according to the order from **Step 1**. In this scanning, identify objects which should serve as defining objects (roots) for contexts. The choice should be made by the meaning and importance of the object in the application compared to its superobjects' meaning. These chosen objects start new contexts rather than refining the contexts of their superobjects.

After these objects are identified, they are *role-of* their superobjects. This kind of *role-of* relationship is a *regular role-of*, where the relationship models a switch of context, that is, the relationship goes from an object in one context to an object in another context.

Step 3: Multiple superobjects. (Computer)

List all objects with multiple superobjects in bottom-up order. In the discussion following **Step 4** and at the end of this section, we will explain why we are using bottom-up processing at this point.

Step 4: Identify primary parent. (Human)

For each of the objects identified in **Step 3**, the expert needs to identify at most one superobject which is in the same context as the object. The relationship to

this superobject will be defined as a *category-of* relationship while all other superobjects should belong to different contexts than the "chosen" superobject and the relationships to them are defined as *role-of*.

From our experience, for most of the objects with multiple superobjects an expert can easily determine which of the superobjects is the defining one, i.e., which should be in the same context and have a *category-of* relationship directed to it. There is a minority of cases where the decision about a major or definitional superobject of a given object is not easy. In such cases, we try to distinguish which of the several superobjects, if any, should have a *category-of* relationship pointing to it, based on the partial context information we have already accumulated in our bottom-up processing.

We distinguish several cases.

Case 1: One of the superobjects is definitional while the others are functional. For example, drugs can be classified by the chemicals that they contain (definitional) and by their therapeutic uses (functional). Then we look at the context to which the object and its descendants belong. (This is the reason for the bottom-up processing). We try to determine whether the nature of the *category-of* relationships is functional or definitional. If it is definitional, we will prefer the definitional superobject. If it is functional, then we will prefer the functional superobject (or if there are several functional superobjects, we will prefer the one which fits the function appearing in the context of the object). If the object is the only object in its context, we will choose the definitional superobject. In this case, one superobject is chosen as primary superobject. The object is *category-of* this primary superobject and *role-of* the other superobjects. This kind of *role-of* relationship is a *regular role-of* since a switch of context from superobject to object has occurred.

Case 2: All superobjects are definitional with the same importance or indistinguishable importance as each of them contributes to the definition of the object in

an equal or indistinguishable way. In such a situation, the object with multiple superobjects could belong to the context of any of its superobjects. However, by the **Rule 1** it cannot belong to more than one context. Also, we have no reason to prefer one over the other. Each choice of context will disassociate the object from the other contexts. This conflict is resolved by requiring that such an object starts a new context which represents the concept obtained as intersection of the concepts of all its superobjects. Thus, this object is *role-of* all its superobjects. We call this type of *role-of* "*role-of/intersection*" represented as r/i in the figures. By this term, we emphasize that this is not an actual case of a switch of context but an artificial case due to the requirement of the theorem to forbid two *category-of* superobjects. Without the theorem, we could probably leave the intersection concept in the context of its superobjects if all belong to one context.

Case 3: The concept of the object is a combination of the concepts of the multiple superobjects in different contexts, but one of them contributes more to the meaning than the others. Then the *category-of* relationship should point to the preferred superobject, as those two should belong to the same context, while the other relationships should be *role-of* relationships.

Step 5: Identify diamond structures. (Computer)

Scan the hierarchy according to the topological order bottom-up to find all the objects with more than one superobject. For each such object a and for each pair of superobjects s_1 and s_2 of a, find a lowest common ancestor b of both s_1 and s_2. For each pair of such objects a and b, output the diamond or extended diamond structure (represented by $< a, b >$) containing a, b and all the objects which are descendants of b and ancestors of a. The object a is called the source of $< a, b >$, and the object b is called the sink of $< a, b >$.

Step 6: Diamond cutting. (Human)

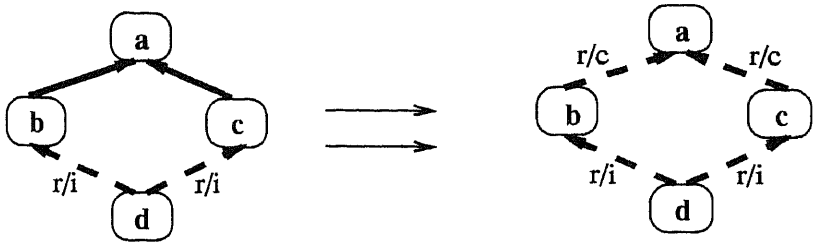

Figure 5.4 The diamond structure

Each diamond or extended diamond structure must contain objects from more than one context in order to fulfill **Rule 2** of disciplined modeling. After executing the first five steps we discussed above, all diamond structures already satisfy **Rule 2.** But there is one case where we must artificially change the *category-of* relationships to *role-of* relationships, to resolve a contradiction.

In this case, which we call *contradictory diamond case*, the source d of the diamond structure $< d, a >$ is a *role-of/intersection* of its superobjects. All other objects in the diamond structure belong to one context (see Figure 5.4). Since the source d is the intersection of two superobjects b and c, they cannot both belong to the same context of their superobject a. Otherwise, because the intersection of a context with itself will result in the original context, the intersection must belong to this common context. Thus, the objects b and c are also defined as separate contexts. The *category-of* relationships are changed, due to **Rule 2**, to *role-of*. However, we want to maintain the distinction between this *role-of* and the two other kinds. Therefore, we denote this kind of *role-of* as "*role-of/category-of*." It is represented by r/c in the figures. This concludes the six steps of our methodology.

Note that the methodology used both top-down processing and bottom-up processing. The determination of the context of objects is performed top-down, as every context root itself has a top-down nature, since the context of the root concept defines the context of its descendants. When scanning the hierarchy top-down, an

expert can identify where an object defines a new context rather than continuing a context of one of its superobjects which has been processed already.

On the other hand, when determining bottom-up to which context an object belongs, choosing from among its superobjects, it is important to know the descendants of the object which belong to the same context. This knowledge will help to determine which of the contexts of the superobjects fits best to the already constructed context.

5.5 Applying the Methodology to a Complex Hierarchy

In order to test the effectiveness of our methodology, we applied it to the previously mentioned cortisporin subnetwork of the MED. First, informational thinning was used to obtain a directed acyclic graph hierarchy out of this subnetwork (Figure 5.1). Then we used the methodology introduced in the previous section to partition the hierarchy into trees. Each tree produced by the partitioning is a logical unit in the forest hierarchy. The root object of a tree defines the unifying context for the objects in that tree.

Step 1, topological sort, is applied to the hierarchy of Figure 5.1. Since there are degrees of freedom in applying topological sort to a directed acyclic graph, the order we used is from left to right and breadth first search [5]. The object numbering from 1 to 46 in Figure 5.1 reflects this order.

In **Step 2**, following the topological sort ordering, the domain expert scans the hierarchy top-down to find all objects which define new contexts. All IS-A relationships from these objects to their superobjects are defined as *role-of*.

Note that by specifying IS-A relationships as *category-of* or *role-of*, the designer is making modeling decisions, which may differ from one designer to another, influencing the emerging contexts. Modeling decisions made by a pharmacist will differ from those made by a surgeon. Thus, each of them can create his own local

partitioning of the vocabulary which represents his view of the vocabulary. In our consideration in this section, we try to take the inclusive approach of a vocabulary administrator (VA).

Because the concept **Medical Entity** (1) is the unique root for all other concepts in the MED, it starts a new context and it does not have any *category-of* or *role-of* superobjects.

Following topological sort order, we can see the concept **Conceptual Entity** (2), which is a straightforward specialization of its superobject **Medical Entity** (1), in the same context as (1). Thus, concept (2) is *category-of* its superobject (1). The concepts **Intellectual Product** (3), **Orderable Entity** (4), **Classification** (5), **Pharmacy Concepts** (6), and **American Hospital Formulary Service Class** (7) are all straightforward specializations of their superobjects. In our intuitive judgement, they are also similar in nature to their superobjects, and thus they are in the same context as their superobjects. Thus, all of them are *category-of* their superobjects.

The concepts **Drug Enforcement Agency (DEA) Controlled Substance Category** (8), **Drug Allergy Class** (9) and **CPMC Formulary Drug Forms** (10) have only one superobject **Pharmacy Concepts** (6) which is a broad term that refers to various ways of grouping drug concepts. But concept (8) defines drug concepts that are controlled by the DEA while concept (9) is a group of drug concepts that have allergic or antiallergic effects and concept (10) refers to the dispensation form (tablet, injectable, etc.) of the drug. Thus, all of these objects represent drug classifications according to various new dimensions and are considered to be defining objects for new contexts. All of them are *regular role-of* children of their superobjects.

The concept **Pharmacy Items** (11) has two superobjects. One is **Pharmacy Concepts** (6) which was analyzed above; the other is **Orderable Entity** (4)

which describes a heterogeneous group of concepts that can be ordered and may be pharmacy or non-pharmacy concepts. Thus, the concept (11) is defined as a root object for a new context, as it is a classification according to a new dimension. It is a *regular role-of* its two superobjects.

The concept **Anti-Infective Agents** (17) has two superobjects which are **CPMC Formulary Drug Item** (16) and **American Hospital Formulary Service Class** (7). Both superobjects define formulas of various pharmacological preparations and contribute their own formulations (one from CPMC and the other from American Hospital Formulary) to the concept. The concept (17) is neither in the same context as (16) nor (7). It is *role-of* both superobjects and starts a new context. The concepts **Eye, Ear, Nose and Throat Preparation** (18), **Hormones and Synthetic Substitutes** (19), **Anti-Inflammatory Agents** (20) and **Skin and Mucous Membrane Agents** (21) all require the same analysis as (17). All are *role-of* their superobjects and are root objects for their contexts.

Consider the concept **Glucocorticoid Agents** (29); it has two superobjects, **Adrenal Agents** (24) and **Anti-Inflammatory Agents** (20). Glucocorticoid Agents are secreted by Adrenal glands and therefore the superobject (24) indicates the physiological source for the Glucocorticoid group of agents. Another child of the concept (24) describes Mineralocorticoids (Aldosterone) (not shown in the figure) which are functionally distinct from Glucocorticoids. (20) describes a heterogeneous set of concepts that includes steroidal anti-inflammatory drugs like Glucocorticoids and non-steroidal anti-inflammatory agents like Aspirin, Ibuprofen, Indomethacin and Phenylbutazone Preparations. Therefore, (29) starts a new context and is *role-of* its two superobjects.

Let us check the concept **Polymyxin B Preparations** (35) which has one superobject **Miscellaneous Antibiotics** (27). The superobject (27) describes a heterogeneous group of antibiotics that belong to chemical families that do not fall

into the major antibiotic families like **Penicillins**, **Cephalosporins**, **Aminoglycosides**, etc. Some of the subobjects of (27) are **Vancomycin** (a glycopeptide), **Bacitracin Preparations** (a polypeptide), and **Clindamycin** (a lincosamide). The concept (35), which is a cyclic polypeptide, is a child of the concept (27). It is in the same context as its superobject (27). It does not start a new context and is therefore *category-of* the superobject (27).

No other objects are determined to start a new context. As a result of this process we have 11 defining objects and contexts. These objects, except for the root concept **Medical Entity** of the whole vocabulary, are *role-of* their superobjects. See Figure 5.5 for the state of the hierarchy at this step of the analysis. We use our graphical notation [66] to display a *category-of* link by a solid arrow and a *role-of* link by a dashed arrow.

In order to improve the clarity of the presentation and to eliminate complicated medical terms, we will sometimes use only the topological sort numbers to represent these medical terms in the balance of this section.

Step 3 of our methodology is to find all the objects which have more than one superobject in bottom-up order (reversing the order of the topological sort in Figure 5.1). These objects are (46), (44), (42), (40), (38), (37), (36), (32), (31), (30), (29), (28), (26), (25), (23), (21), (20), (19), (18), (17), and (11).

Because the number of primary superobjects for each object with multiple parents is at most one, the domain expert needs to identify at most one primary parent for each object listed above, in **Step 4** of the methodology. For example, the concept **CPMC Drug: Cortisporin Opthalmic Ointment** (46) has three superobjects, **Bacitracin/Hydrocortisone/Neomycin/Polymyxin B Combination Preparations** (44), **Drug Dispensed by Gram** (15) and **Eye, Ear, Nose and Throat Antibiotics** (28). The superobject (44) defines the chemicals that form the Cortisporin Opthalmic Ointment. They uniquely define the structural components

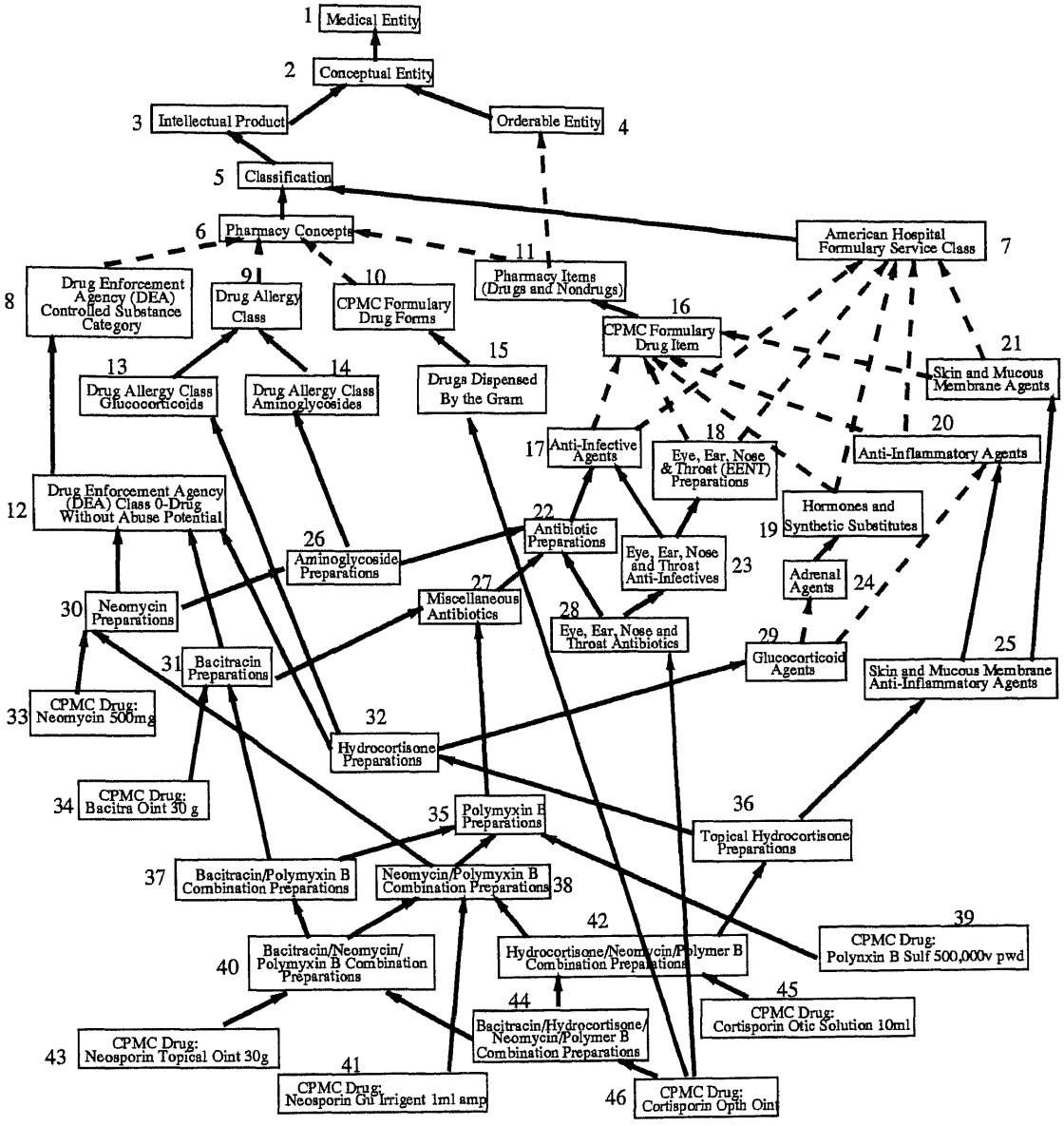

Figure 5.5 The subhierarchy in Figure 5.1 after executing methodology **Step 2**

of the ointment, and therefore by **Case 1** of **Step 4** (44) is the primary superobject of (46). The superobject (15) specifies the mode of dispensation and the superobject (28) specifies the site and action, and therefore both do not define the context of the concept. Thus, according to **Case 1** of our methodology, (46) is *category-of* (44), *role-of* (15) and *role-of* (28).

Let us check another object which has more than one superobject. The concept **Bacitracin/Hydrocortisone/Neomycin/Polymyxin B Combination Preparations** (44) has two superobjects, **Bacitracin/Neomycin/Polymyxin B Combination Preparations** (40) and **Hydrocortisone/Neomycin/Polymyxin B Combination Preparations** (42). Both superobjects contribute two chemicals common to both concepts (Neomycin and Polymyxin B) to the concept. In addition, (40) contributes Bacitracin and (42) contributes Hydrocortisone. All these chemicals together define the concept (44). According to **Case 2** of **Step 4**, it is not possible to identify the primary superobject. Hence it is *role-of* its superobjects. As we defined before, this kind of *role-of* is *role-of/intersection*. In Figure 5.6, we marked this *role-of* as "r/i" to distinguish it from a regular *role-of*. A similar analysis can be applied to all concepts that are roots of drug combinations like (40), (42), (38), and (37).

Another example is the concept **Bacitracin Preparations** (31) which has two superobjects, **Miscellaneous Antibiotics** (27) and **Drug Enforcement Agency (DEA) Class 0-Drug Without Abuse potential** (12). We already analyzed the superobject (27) in **Step 2**. (31), which is a polypeptide, is a child of (27). It is in the same context with its superobject (27). The other superobject (12) simply indicates a classification for the DEA according to a drug's abuse potential and is not a definitional superobject for the concept. Thus, (31) is *category-of* (27) and *role-of* (12).

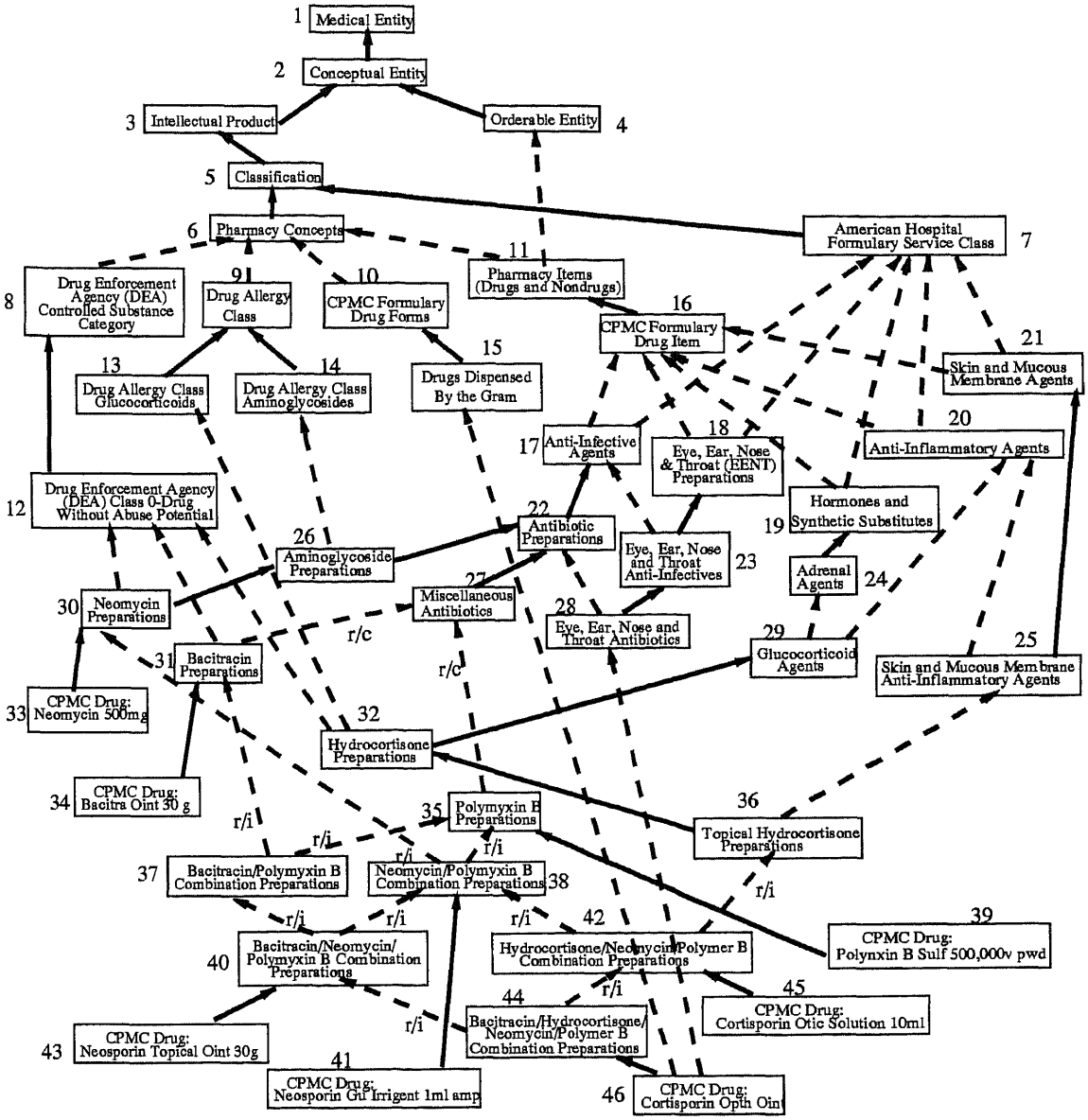

Figure 5.6 The subhierarchy in Figure 5.1 with *category-of* and *role-of*

After **Step 4** is completed, none of the objects with multiple superobjects in Figure 5.6 has more than one primary parent. That means that each object is *category-of* at most one superobject.

Now we need to identify the diamonds or extended diamonds structures in bottom-up order. As discussed above, we use a pair <A, B> to denote a diamond structure with A as source and B as sink. The (extended) diamond structures in Figure 5.6 are <(46), (6)>, <(46), (22)>, <(44), (35)>, <(42), (7)>, <(42), (12)>, <(42), (16)>, <(40), (35)>, <(38), (22)>, <(37), (27)>, <(36), (20)>, <(32), (6)>, <(31), (6)>, <(30), (6)>, <(29), (7)>, <(29), (16)>, <(28), (17)>, <(26), (6)>, <(25), (16)>, <(23), (16)>, <(21), (5)>, <(20), (5)>, <(19), (5)>, <(18), (5)>, <(17), (5)>, and <(11), (2)>.

After we identify all the (extended) diamond structures in Figure 5.6, we need to check whether any <A, B> is a contradictory diamond case as described in **Step 6**. If such cases exist, we need to change the appropriate *category-of* relationships to *role-of* relationships.

One of the extended diamond structures is <(30), (6)>. It is already divided into three contexts. It is not a contradictory diamond case, thus, we do not need to do anything about it.

Now, let us examine <(37), (27)>. As a result of **Step 2** and **Step 4**, both concepts (31) and (35) would be *category-of* concept (27). The source concept (37) is *role-of/intersection* of two superobjects (31) and (35) according to the result of **Step 4**. This diamond structure is a contradictory diamond case as described in **Step 6**. Thus, at least one of the superobjects (31) and (35) must be made *role-of/category-of* the superobject (27). Since both concepts (31) and (35) are in the same configuration that we encountered before, we cannot choose only one to be *role-of/category-of* their superobject. Thus, both of them now are *role-of/category-of* their superobject. In Figure 5.6, the *role-of/category-of* relationship is represented as r/c. This is the only

diamond structure in Figure 5.1, for which the contradictory diamond case of **Step 6** holds. In this way, we represent the knowledge that both concepts (31) and (35) were separated from their parent's context just to fulfill the requirements of **Rule 2**. But for other purposes, they and their *category-of* descendants may be considered part of the context to which the concept (27) belongs.

After all IS-A relationships in Figure 5.1 have been changed to *category-of* or *role-of*, the forest subhierarchy of the original subnetwork is obtained by removing all the *role-of* relationships. Figure 5.7 shows all contexts as trees in the forest. The relationship between objects of different contexts (trees) is *role-of*.

The hierarchy in Figure 5.1 is partitioned into 18 contexts, many of which are very small and seem to be too detailed. But note that this is not a typical subnetwork of the MED. By choosing a subnetwork with a very complex hierarchy we ended up with a network with many interrelated subjects. Furthermore, even the contexts shown in Figure 5.7 are not complete since some terms which belong to these contexts are not shown as they are not ancestors of (46). To demonstrate this, we added in Figure 5.7 some of those extra concepts A, B, and C representing **CPMC Drug: Polysporin Opthalmic Ointment 3.5 Gm**, **CPMC Drug: Polysporin Topical Ointment 30 Gm**, and **CPMC Drug: UD Polysporin Ointment**.

We applied our methodology to the InterMED (an offshoot of the MED) containing about 3,000 concepts. It was partitioned into 545 contexts, 394 of them consisting of single concepts due to the InterMED's incompleteness. (I.e., if more concepts of the MED would be added to the InterMED, then some of these singleton concepts would get descendants and turn into actual contexts.) Thus, the InterMED is practically partitioned into 151 actual contexts with an average size of 16. This partition of the InterMED achieves our original goal of partitioning the vocabulary into screen-sized, logical units reasonably comprehensible to a user.

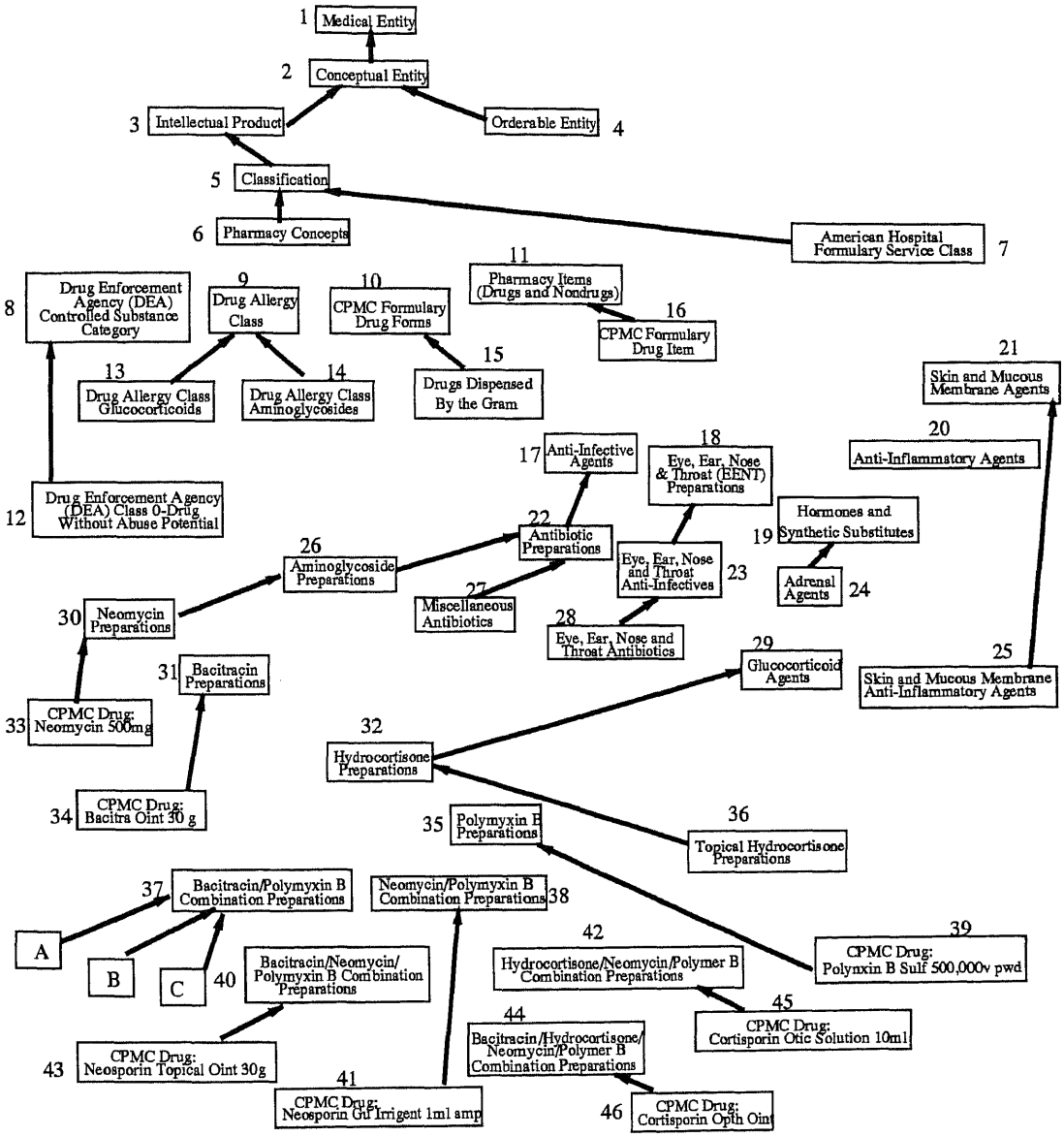

Figure 5.7 The forest subhierarchy of Figure 5.1

5.6 Summary

Vocabularies promise to be important tools for many medical information processing tasks. They can help overcome differences in terminology between different databases and information systems and different categories of users. Unfortunately, the job of understanding and maintaining the vocabulary itself is difficult and time-consuming. A graphical representation can help in the process of understanding most vocabularies. However, if the vocabulary is very large, the graphical representation rapidly loses its intuitive appeal. In this chapter, we have presented a methodology for partitioning a (graphical) vocabulary representation into meaningful units.

Disciplined modeling assumes a vocabulary that is structured around a directed acyclic graph of IS-A relationships. It defines three simple rules that, if followed, guarantee that a forest, i.e. a collection of trees, can be identified, which partition the vocabulary into meaningful units called contexts. Based on this formal result, we presented a methodology for partitioning an existing vocabulary into contexts. As computers cannot (yet) judge "meaning" well, our methodology relies on the close interaction between human and computer. The result of the partitioning process can be used to study a single context at a time and the interaction between pairs of contexts. This presents a major improvement over studying "the part of the vocabulary that is just now displayed in the window."

Two experiments with the methodology were presented. The first one used a very complex subnetwork of the MED vocabulary, which poses a challenge due to its complexity. The second one used the InterMED, a medium sized vocabulary. Both experiments demonstrated the effectiveness of the methodology.

To date, we have only anecdotal evidence that the partitioned vocabulary is easier to use than the original source vocabulary. We are planning a human-factors evaluation of the results of our methodology using students in the Biomedical Informatics program at the University of Medicine and Dentistry of New Jersey. We

expect that such a study will show that students with access to our partitioned vocabulary will solve a given problem faster and more accurately than students in a control group.

CHAPTER 6

USING A SIMILARITY MEASUREMENT TO PARTITION A VOCABULARY OF MEDICAL CONCEPTS

6.1 Introduction

In order to improve the quality of medical communications, the medical community has created a number of standardized vocabularies. These vocabularies help healthcare providers, insurance companies, pharmacies, and other members of the medical community avoid misunderstandings and define unique encodings for drugs, diagnoses, diseases, procedures, etc. Some of these vocabularies have been computerized, such as the MED [36, 37] and the InterMED [112, 129] which are modeled as semantic networks [53, 133]. However, the extensive size of most vocabularies often makes it difficult for users to gain an understanding of their contents.

The MED is a very large semantic network with over 48,000 concepts, 61,000 IS-A links and 71,000 other links in its year 1996 version. While the MED is extremely useful, it is very difficult for its users to grasp the wealth of knowledge contained in it. Intuitively, a graphical representation of the MED should be helpful. However, one can estimate that a complete picture of the MED would comprise an area of at least 300 square feet, which is by far too large for any comprehension purposes. For the InterMED (a partial revised version of the MED), we were able to perform a measurement of the size of a display. This display required over sixty square feet for a layout of about 3,000 concepts and all the IS-A links connecting those concepts.

In order to deal with this problem, we have investigated the problem of partitioning a semantic network into smaller "contexts." The basic goal has been to break the large semantic network of a vocabulary into smaller disjoint units, called contexts, that fulfill three conditions: (1) Every group of concepts by itself should be meaningful in the eyes of an expert user; (2) Every group of concepts by itself

should fit onto one screen of a modern workstation; and (3) The union of all disjoint contexts should be identical to the original semantic network.

Semantic networks are in their essence graphs that are constructed around backbones of IS-A links. These IS-A links form directed acyclic graphs (DAGS). One would think that the extensive literature on graphs and graph partitioning [3, 10, 11, 81] would supply the theoretical means for achieving the kind of partitioning that we described above. Unfortunately, this expectation fails for two reasons: (1) Partitioning problems tend to be NP-complete [50], i.e., probably not solvable by polynomial algorithms; (2) The requirement of "meaningful groups" is beyond the scope of graph algorithms.

To overcome these problems, we have previously combined human expert judgment with algorithmic tools. In Chapter 5, we have introduced a technique called "disciplined modeling" that results in a partitioning of a semantic network into a set of trees, if three simple modeling rules are obeyed. However, in this approach, human expert judgment is an important ingredient. In this chapter, we present an approach which attempts to completely avoid the involvement of a human expert in the partitioning process. Rather, it relies on structural features of the semantic network.

To understand the basic idea of our partitioning approach, we need to give some background regarding the MED and InterMED. From this point on, we will only refer to the InterMED, but all ideas apply equally to the MED. In the InterMED semantic network, each attribute and relationship is introduced at a unique concept and is inherited by all concepts below the point of introduction, via the IS-A links. Sometimes several attributes and/or relationships are introduced at the same place.

Attributes describe information local to a concept, while relationships are links between concepts. In Figure 1.1, we show a small subnetwork of the InterMED. The fact that an attribute or relationship is introduced at a certain point, however, does

not guarantee that a value is also introduced for it at that point. Attribute values are *not* inherited in the InterMED, but relationship targets usually are. In some cases, relationships are overridden at lower levels in the hierarchy.

It turns out that sometimes concepts in the InterMED are very similar to their parent concepts, while at other times they are very different. To be specific, let us consider the features introduced above: A child concept might not introduce any new attributes and relationships. In addition, it might inherit all relationship targets without any overriding. Clearly, such a child concept is very similar to its parent concept. On the other hand, a child concept might introduce several new attributes and relationships and override the targets of all its inherited relationships. In this case, we would say that the child concept is very different from its parent.

The underlying idea of the partitioning is to examine every concept in the InterMED and find out how similar it is to its parent(s). Obviously, if a concept N is very different from (all) its parent(s), we may suspect that a new meaningful group of concepts starts just below it. Furthermore, if we find that all the children of N are very similar to it, our assumption is justified.

The similarity measure we propose can be expressed concisely by a single number. By using either the introduction points of attributes and relationships alone, or by combining introduction points with target inheritance behavior, we can gain some fine control over the computation of these similarity numbers. In this approach, no human expert is required at all.

The rest of this chapter is organized as follows. In Section 6.2, we will describe the partitioning approach that we have used to partition the InterMED. The results of partitioning the InterMED are discussed. Section 6.3 compares these results with the results obtained by our previous approach, which made use of a human expert. In Section 6.4, we will briefly review some related literature. Finally, in Section 6.5, we will conclude with a discussion of the value of these results and future work.

6.2 Partitioning Approach

Our partitioning approach is based on the similarities of the property sets of child/parent concept pairs in the semantic network. The property similarity of a child/parent pair gives a quantitative measure of the similarity of the children and parents. It is obtained from comparisons of the respective properties of each. We define the property similarity of the child/parent pair σ as a number between 0 and 1, where 0 means the lowest similarity and 1 represents the highest similarity (identical).

6.2.1 Similarity Based on Property Introduction

In the InterMED, each property is first introduced at a unique concept which we will call *property-introducing concept* for the property [90]. A concept may serve as the property-introducing concept for many properties. A property is inherited by all the children and descendants of the specific property-introducing concept. Thus, there are only two cases for all child/parent pairs. One is that the child concept has the same properties as its parent concept. This means that the child concept only inherits the properties from its parent concept instead of introducing any new properties of its own. This child concept is obviously similar to its parent concept. For this case, we assign this pair the similarity 1. The other case is that the child concept has more properties than its parent. Either the child concept introduces at least one new property, or it inherits properties from multiple parents with different properties. In this case, the two concepts are not similar. This pair will be given the similarity 0.

According to the property similarities of all child/parent pairs, we can partition the original network into several subnetworks by removing all child/parent pair links which have similarities 0. Thus, each subnetwork will contain the concepts with the

same properties. One could argue for a more sophisticated numeric evaluation. For instance, we could use the following formula to get finer distinctions.

$$\sigma = \frac{\text{number of properties at parent}}{\text{number of properties at child}} \tag{6.1}$$

However, the InterMED contains only 58 distinct properties and some concepts introduce more than one property. A typical concept in the InterMED just inherits its properties from its parent concept. There are only 38 child/parent pairs in which the child concepts have more properties than their parent concepts. Thus, we obviously want to cut all child/parent links where the child and parent are at all different. Therefore, assigning 0 to such pairs of concepts is a good choice. In a network that has a higher degree of property introduction, we would use the above formula. After we remove all links with similarities 0, the InterMED with 2,820 concepts is divided into 38 subnetworks. Unfortunately, one of the subnetworks contains more than 1,000 concepts. This result is not what we expect. We need a better formula to compute the property similarity.

6.2.2 Similarity Based on Relationship Overriding

As we saw, considering only the numbers of properties for computing the similarity of each child/parent pair does not give a satisfactory partitioning result. The property similarities need to be calculated more accurately.

Before giving the formulas which we used to compute the property similarity, we need to recall the two kinds of properties in the InterMED. One is attributes which have literal values. The other is relationships which point to other concepts. Both attributes and relationships may be set-valued. That means that one relationship may have several target concepts. The inheritance in the InterMED may take place in different ways:

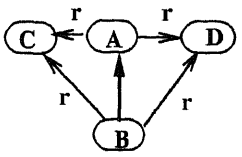

Figure 6.1 Full inheritance

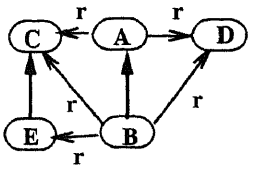

Figure 6.2 Additive inheritance

- **Attributes:** A child concept only inherits the definitions of attributes from its parent, but not the values of the attributes.

- **Relationships:** A child concept inherits the definitions of relationships from its parents. For each relationship's value, one of the following four possibilities holds.

  - *Full inheritance:* A child concept inherits all values (target concepts) of the relationship. In Figure 6.1, A has the relationship targets C and D. B inherits the relationship r and, therefore, has the same targets C and D.

  - *Additive inheritance:* A child concept inherits all values of the relationship and adds more values for the relationship. In Figure 6.2, A has the relationship targets C and D. B inherits the relationship r and has the same targets C and D. In addition, B has the additional target E, which is a child of the target C.

  - *Partial overriding:* A child concept inherits some values and overrides (refines) other values of the relationship. In Figure 6.3, B inherits the target D from A. However, the target C is overridden by the new target E, which is a child of C.

  - *Full overriding:* A child concept overrides (refines) all values of the relationship. In Figure 6.4, all targets of the relationship r of A are

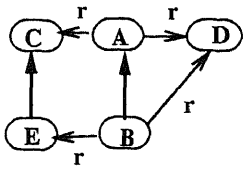

Figure 6.3 Partial overriding

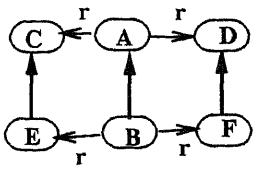

Figure 6.4 Full overriding

overridden. B refers to E and F instead of their respective parents, C and D.

We will now introduce a similarity formula that takes the distinct inheritance possibilities listed above into account. Consider a child concept having m relationships and its parent concept having n relationships. Suppose further that the number of fully overridden relationships is f, the number of partially overridden relationships is p, and the number of additive inheritance relationships is d. Then the similarity of the child/parent pair is defined to be:

$$\sigma = \frac{n - f - p - d}{m} \tag{6.2}$$

When the child has the same number of relationships as its parent ($m = n$) and no relationships are overridden, the similarity of the child/parent pair will be 1. On the other hand, if there is no full inheritance for any relationship, the similarity for the child/parent pair will be 0.

Using formula 6.2, we can compute the property similarities for all child/parent pairs. Based on these property similarities, we can remove IS-A links between child concepts and their parent concepts for pairs with low similarities. Ideally, the original network will be partitioned into several smaller sub-networks, within each of which all concepts are quite similar. This is the result that we want.

Unfortunately, due to multiple inheritance, this is not always true. As we mentioned above, the InterMED is a DAG. Cutting "random edges" in a DAG gives us little control over partitioning into components. For example, if all removed

links happen to be links of child concepts with multiple parents, and, after cutting, each child is still connected to at least one parent, the original network will not be partitioned at all. If we can first reduce the graph to a tree, the network will be partitioned significantly. Removing links from the tree will result in a forest. There are additional advantages to working with a tree. It is generally considered easier to comprehend a tree than a DAG consisting of the same concepts, because in a forest upward paths are not branching. For these reasons, we reduce the DAG of IS-A links to a tree as follows. We call this step *tree identification.*

Assume that a child concept C has q parent concepts ($q > 1$). The property similarities of child/parent pairs are $\sigma_1, \sigma_2, \ldots, \sigma_q$ respectively. If there is only one maximum number, call it σ_{max}, among $\sigma_1, \sigma_2, \ldots, \sigma_q$, then all links with similarities σ_i ($i \neq max$) will be removed. If there are two or more maximum numbers among $\sigma_1, \sigma_2, \ldots, \sigma_q$, one IS-A link with maximum similarity σ_{max} will be retained randomly, and all others will be removed.

Because the *tree identification* step just described will result in a tree, removing any links which have low similarities in the tree will result in a forest comprising more than one tree. All links inside a given tree will have high similarities.

Now the question is: Which links of the tree should be removed to create a useful partitioning result? Because the purpose of partitioning is to help users comprehend the concept network, each subtree produced by the partitioning scheme should be meaningful and have a manageable size. After we compute the similarities for all child/parent pairs, the distribution of property similarities can be calculated (see Table 6.1). According to Table 6.1, there are n_1 child/parent pairs that have property similarities between 0 and k_1, etc. Table 6.1 gives us the similarity distributions for the whole network and helps us decide which child concepts and parent concepts should reside in the same context. We can choose a numeric parameter K and remove all the IS-A connections between child concepts and their parent concepts

Table 6.1 Distribution of property similarities

Property Similarities	Number of child/parent pairs
0	n_0
$(0,k_1)$	n_1
$[k_1, k_2)$	n_2
$\vdots$	$\vdots$
$[k_i, 1)$	n_{i+1}
1	n_{i+2}

$$0 < k_1 < k_2 < ... < k_i < 1$$

for which the similarity $\sigma < K$. The result will be several trees. All child/parent pairs in each tree have similarities greater than K. By varying K, we have some control over the number of links that are cut. With that, we get an indirect and non-homogeneous control over the size of the contexts that are generated.

An interesting question is whether we gain anything if we combine similarity by property introduction with similarity by relationship overriding. The answer is that the contribution of relationship introduction is already contained in the formula 6.2: Not only is the number of a child's relationships in there, but so is the parent's relationships. For example, if the child has more relationships than its parent due to relationship introduction, the link will not be assigned a similarity of 1, even if the child has full inheritance for all relationships. The contribution of attribute introduction is negligible because there are only 12 attributes in a vocabulary of 2,820 concepts. Most of them are introduced at the root of the vocabulary.

6.2.3 Partitioning the InterMED

Now let us use the approach described above to partition the InterMED. At present, the InterMED contains 2,820 medical concepts. The number of child/parent pairs is 4,687. For concepts with multiple parents, *tree identification* will be applied to

Table 6.2 Distribution of similarities of the InterMED

Property Similarities	Number of child – parent pairs
0	855
(0,0.1)	0
[0.1, 0.2)	0
[0.2, 0.3)	1
[0.3, 0.4)	17
[0.4, 0.5)	0
[0.5, 0.6)	72
[0.6, 0.7)	775
[0.7, 0.8)	22
[0.8, 0.9)	212
[0.9, 1.0)	1
1	864

create a tree. After that, the distribution of property similarities can be computed (see Table 6.2). Based on this distribution table, we can choose a K to partition the InterMED by removing all links with similarities less than K. We can vary K to obtain alternative partitioning results.

First, let us choose $K = 1.0$. This means that each subtree resulting from the partitioning will contain concepts which are maximally similar, i.e. have similarities 1. The number of subtrees created by this partitioning is 1,955; the biggest subtree contains 519 concepts; there are 1,868 subtrees with only one concept. Figure 6.5 shows a plot of the number of trees compared to the size of the trees. It can be seen that the partitioning results in a large numbers of trees with less than 10 concepts.

Table 6.2 shows the results when $K = 0.7$. The partitioning gives us 1,720 subtrees, containing the child/parent pairs with similarities greater than 0.7. The biggest tree consists of 519 concepts. The number of trees consisting of a single concept is 1,646 (see Figure 6.6). Compared with $K = 1.0$, this result is better because even though the biggest trees are still of the same size, the number of small

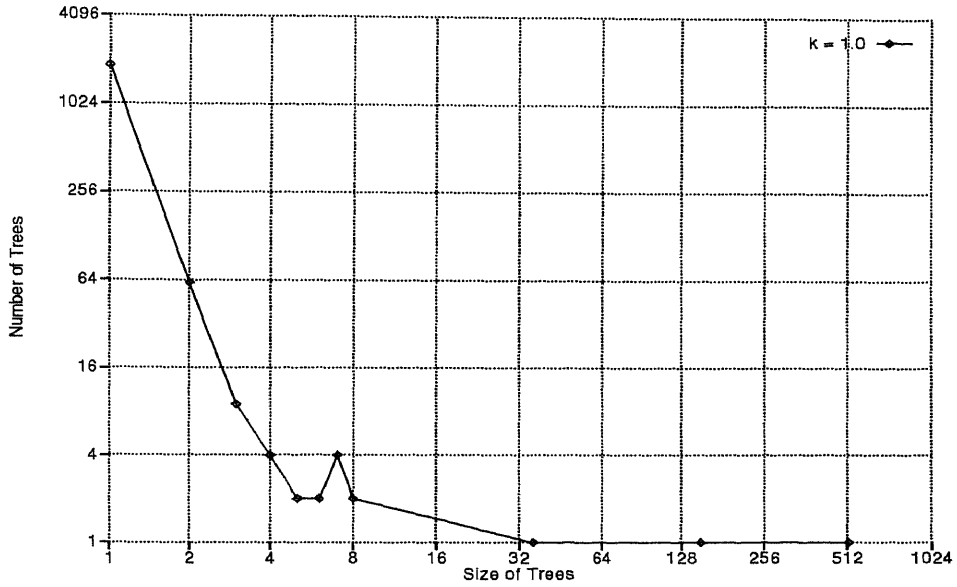

Figure 6.5 Distribution of the number of trees according to their sizes for $k = 1.0$

trees is reduced. However, there are still too many trees with small numbers of concepts. Thus, K needs to be adjusted further.

This time, let $K = 0$. The partitioning will produce a forest, with trees containing child/parent pairs with similarities greater than 0. The number of trees is 855. The largest contains 758 concepts. There are 795 trees which consist of only one concept (see Figure 6.7).

As we have seen, different values of K can be chosen to partition the InterMED differently. Unfortunately, we do not get ideal partitioning results. Some of trees contain large numbers of concepts, which cannot be displayed neatly on one screen. There are also many trees consisting of only a single concept or very few concepts; such trees do not capture much meaning.

Let us note that there is a reason for the appearance of a large number of single-concept trees or trees with very few concepts. It is due to the incompleteness of the current version of the InterMED; among the 2,820 concepts of the InterMED, there are 2,186 concepts which are leaves. Most of the single-concept trees are derived from the leaves of the original network. If more concepts were added to the InterMED,

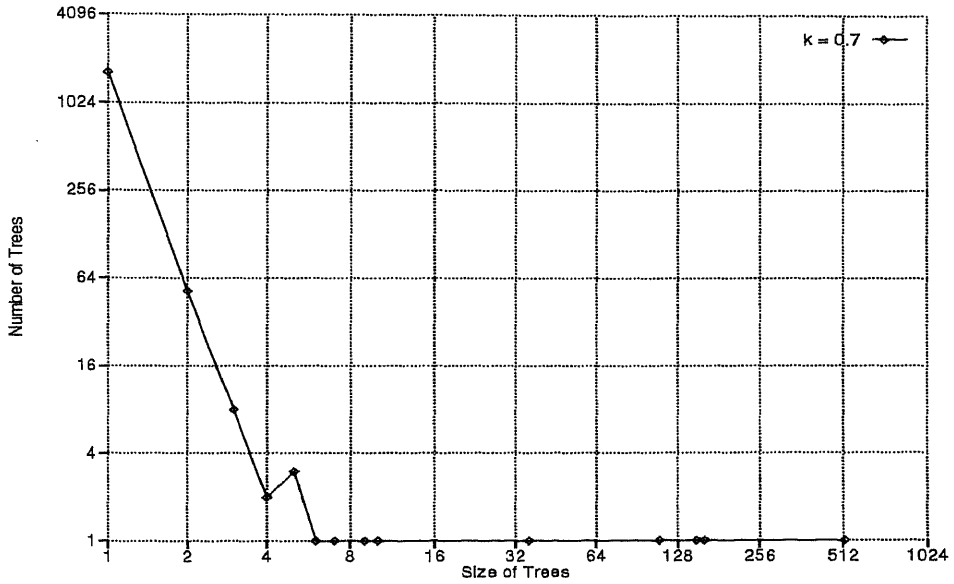

Figure 6.6 Distribution of the number of trees for k =0.7

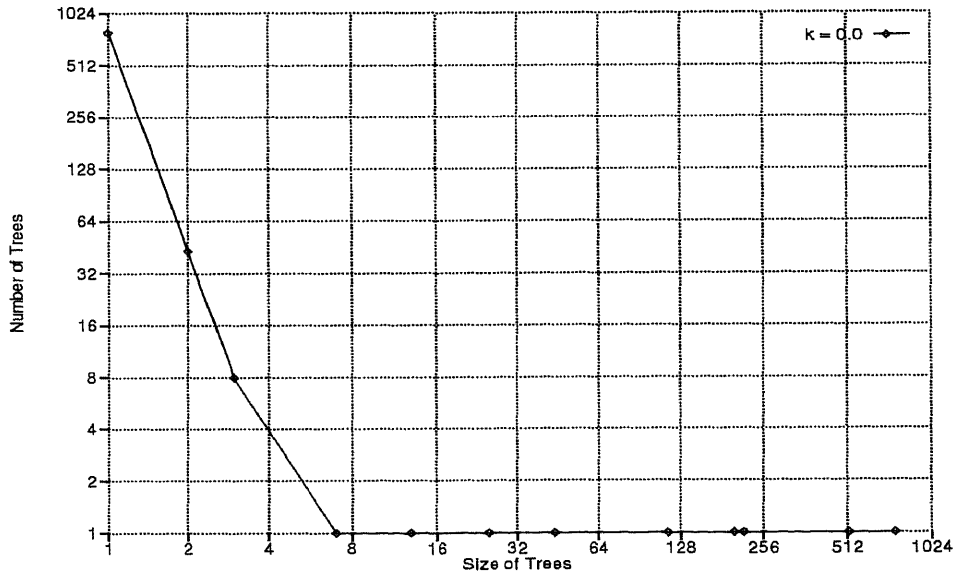

Figure 6.7 Distribution of the number of trees for $k = 0$

some of these leaves would become parents and the trees would turn into actual contexts with significant numbers of concepts.

Concerning the remaining large trees, they would need to be partitioned by using human input. Our automated partitioning algorithm still improves the situation, because there are few large trees left. In addition, even the largest of those remaining trees is considerably smaller than the original vocabulary.

6.3 Structural Partitioning vs. Semantic Partitioning

In this section, we will compare our partitioning results with the results obtained by semantic partitioning [59]. We apply these two methods to the most complex subnetwork of the MED. The results turn out to be quite similar, as will be discussed now.

In the MED, the concept **Cortisporin Opthalmic Ointment** has the most ancestors: 39. We will focus on the subnetwork containing this concept and all its ancestors. The subnetwork contains 62 IS-A relationships and 157 other relationships. In Figure 6.8, we show this subnetwork with only its IS-A relationships.

For 62 child/parent pairs in Figure 6.8, we use the formula 6.2 given in the previous section to compute their similarities. After applying the *tree identification* procedure, we compute the similarity distribution of the tree (see Table 6.3). According to the table, if we want only child/parent pairs with maximum similarity to reside in the same tree, we can choose $K = 1$. After removing all IS-A links with similarities less than 1, we obtain a forest with 13 trees (see Figure 6.9).

In [59], we described a methodology to partition a network into several trees. There, a domain expert is required to make a judgment about whether a child concept and its parent concept are similar, based on his previously acquired domain

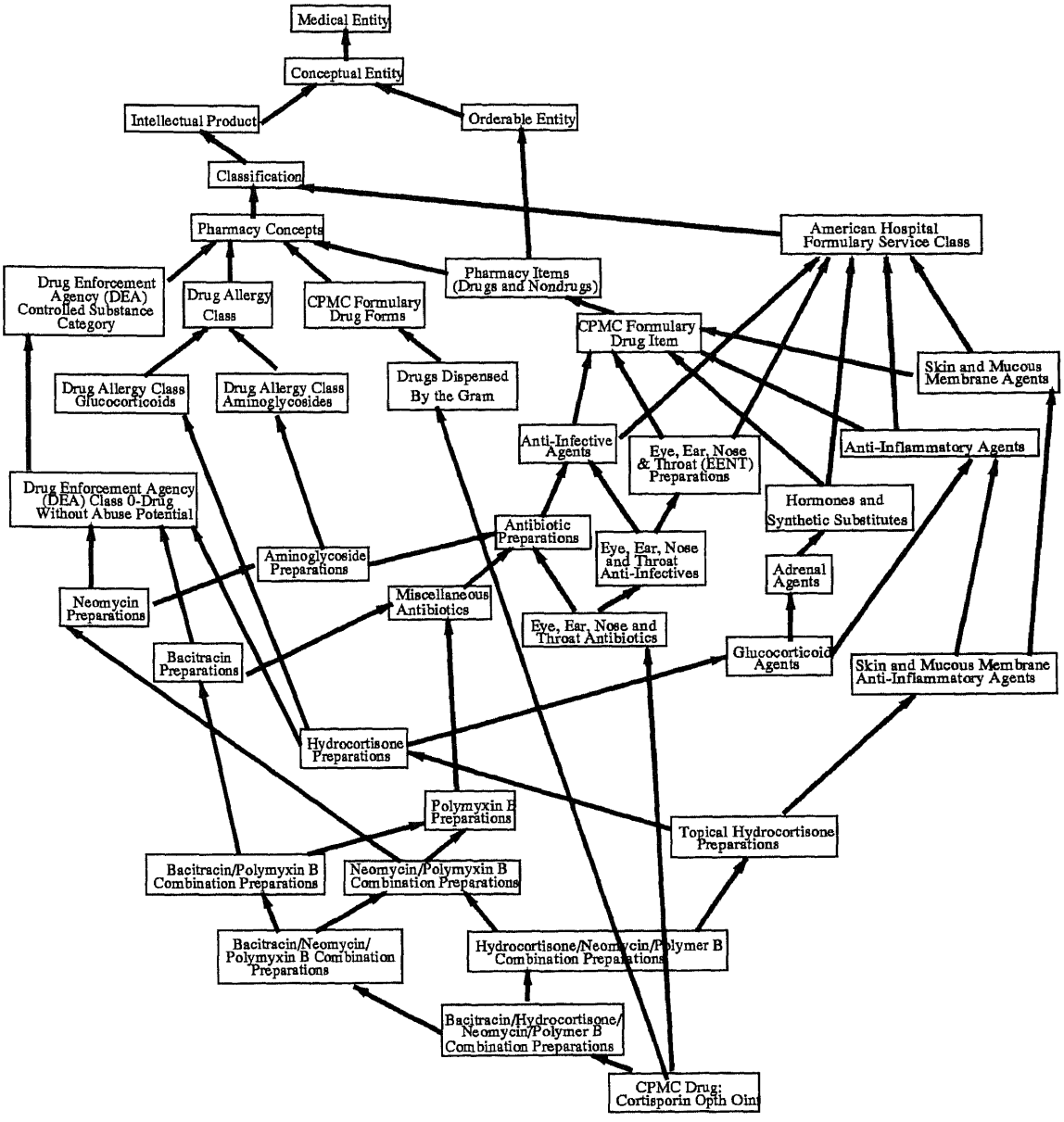

Figure 6.8 Complex subnetwork of the MED

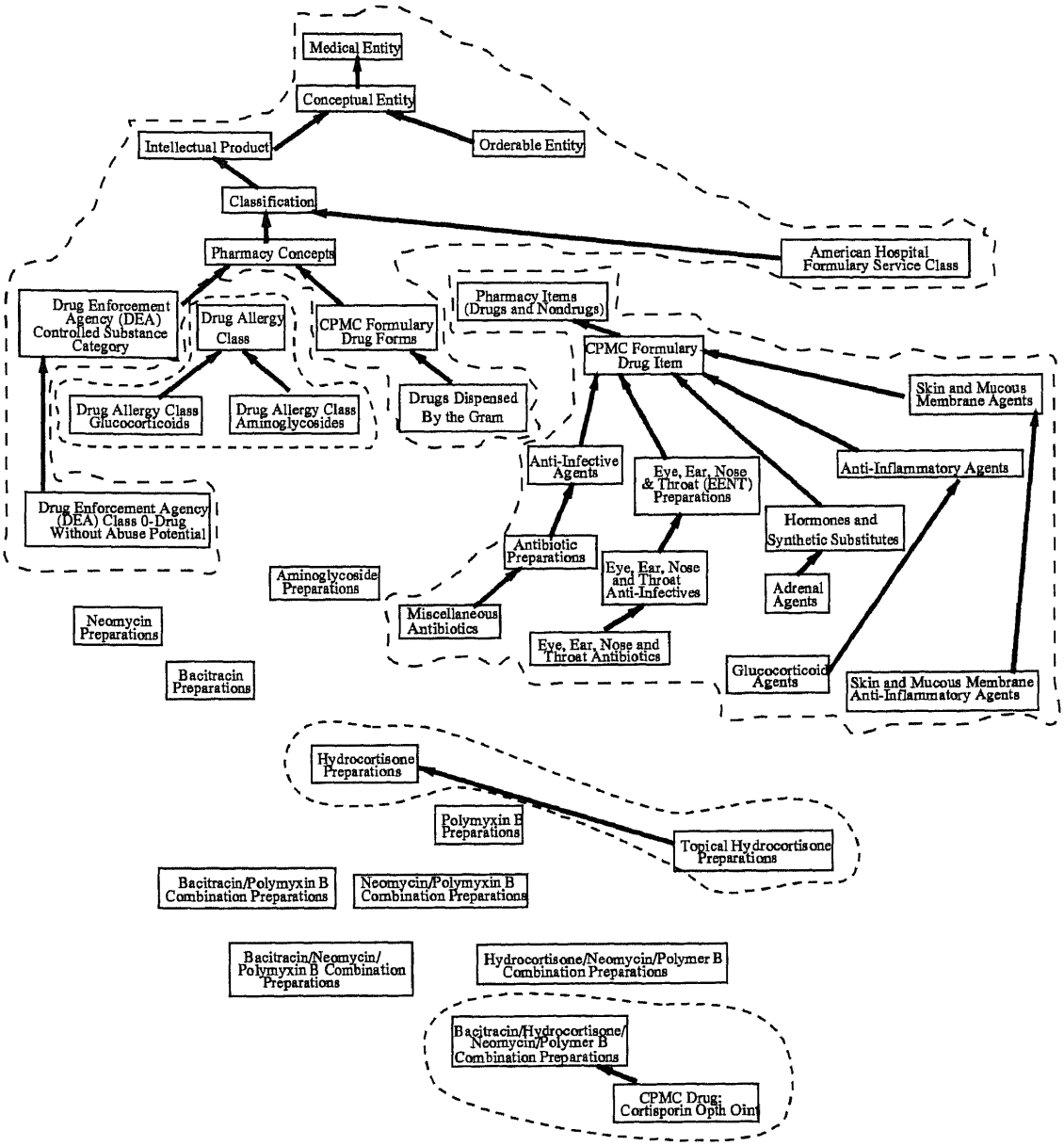

Figure 6.9 Partitioning result based on structural partitioning

Table 6.3 Distribution of similarities of complex subnetwork of MED

Property Similarities	Number of child/parent pairs
0	0
(0,0.1)	0
[0.1, 0.2)	0
[0.2, 0.3)	0
[0.3, 0.4)	0
[0.4, 0.5)	1
[0.5, 0.6)	2
[0.6, 0.7)	4
[0.7, 0.8)	2
[0.8, 0.9)	3
[0.9, 1.0)	0
1	26

knowledge. Using that approach, the complex subnetwork was partitioned into 18 trees (see Figure 6.10).

Comparing the results obtained from the two approaches (Figure 6.9 and Figure 6.10), we find that our "structural" approach gives us results that are similar to the results of the semantic approach in [59]. The results of the structural approach also appear semantically plausible. There are 10 tree roots out of 13 that are the same as for the semantic partitioning obtained from a domain expert's knowledge.

Both partitioning results contain many trees which are very small and too detailed. But this is not a typical subnetwork of the MED. It contains many inter-related subjects. On the other hand, certain contexts are not complete since some of their members which are not ancestors of **Cortisporin Opthalmic Ointment** are not shown in the figure. If we add more concepts, some contexts would become larger.

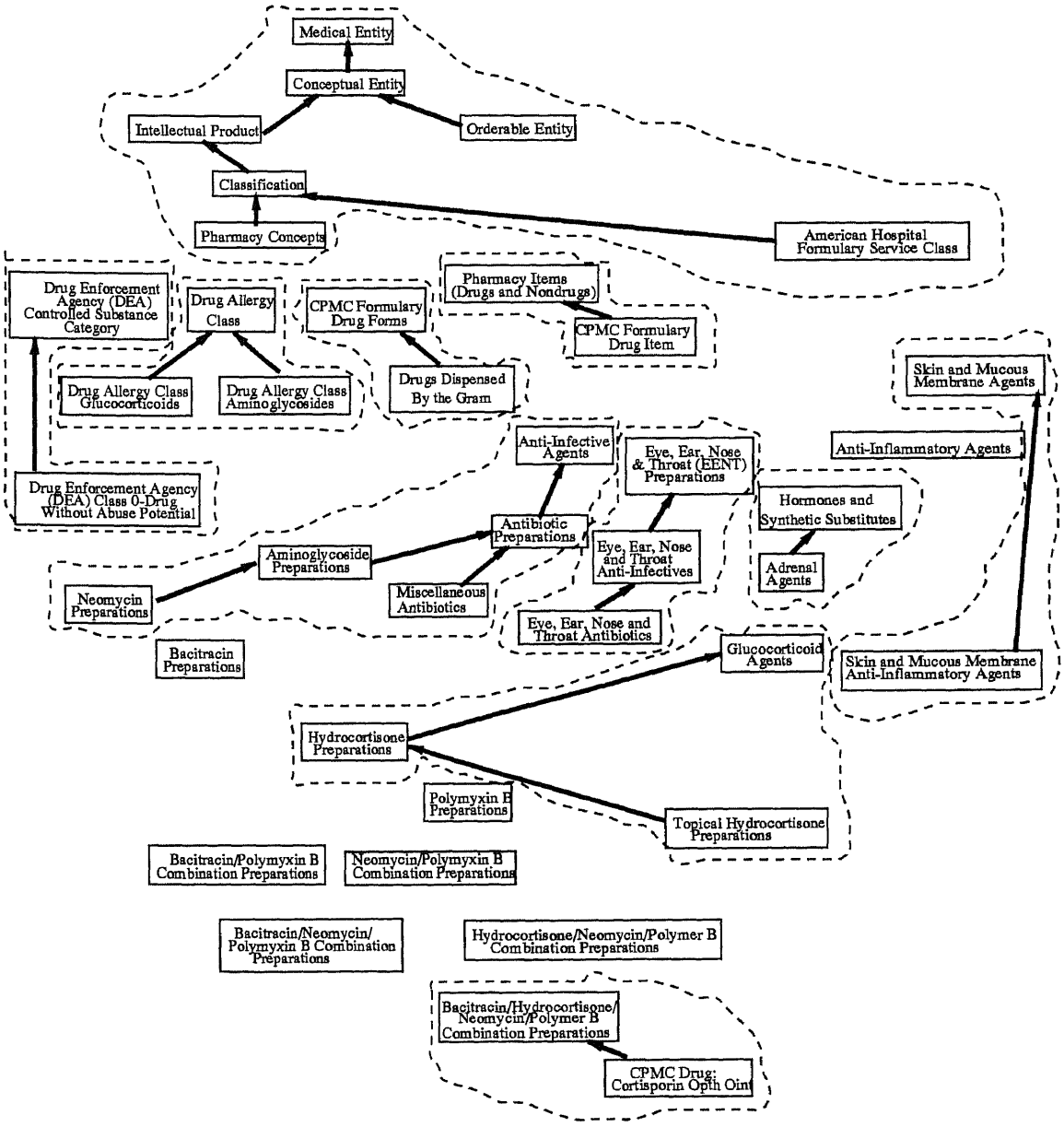

Figure 6.10 Partitioning result based on semantic partitioning

6.4 Related Literature

The issue of grouping concepts together in a "reasonable" manner has long been known as "conceptual clustering" in AI. "Clustering is usually viewed as a process of grouping physical or abstract objects into classes of similar objects. One needs to define a measure of similarity between the objects and then apply it to determine classes" [102]. A "goodness measure" is usually defined for the overall partitioning of objects [32]. Note that these are not classes in the sense of object-oriented programming, but classes in the sense of conceptual categories. On the other hand, in statistical clustering [44] and numerical taxonomy [130], most similarities are defined between pairs of objects. Our formula considers child/parent pair similarity and is applied to all concepts in order to partition an entire network into a collection of trees.

In [68, 69], we find one of the oldest AI approaches to this problem, which partitions networks into "net spaces." These net spaces delimit the scopes of quantified variables. The partitioning into net spaces is done by experts and it cannot be carried out by checking the similarity among concepts. Different from our vocabulary networks, the concepts in SNePS [127] are pre-classified into four types: base concepts, variable concepts, molecular concepts, and pattern concepts. SNePS [127] treats the entire knowledge base as a single network. The problem of partitioning a network into trees is not relevant in SNePS because IS-A relationships are treated in the same way as all other relations. In [88], Levinson presents the principle of pattern-associativity by indexing objects into multi-levels. This approach allows one to organize conceptual graphs into a multi-level partial order by subgraph-isomorphism. (See [88] for more details.)

In [146], Woods describes the taxonomies of structured conceptual descriptions following work of the KL-ONE family [21]. Such taxonomies were generated by the subsumption relationship that relates each pair of concepts [53]. Our formula

provides an approach to break the IS-A hierarchy in a reasonable way to generate a collection of trees.

In order to answer queries using different knowledge representations at different abstraction levels, Chu [31, 32, 101] proposed the "Type Abstraction Hierarchy" which characterizes the instance values differently at different knowledge levels. The hierarchy is defined by "induce rules" [101]. However, the instances have neither subsumption relationships nor attributes among them. For the purpose of generating a concept hierarchy, the method cannot be applied to a complicated network with numerous attributes and subsumption relationships among the concepts.

In OODB approaches, objects with common properties are grouped into classes and a class name is assigned as an abstract designation. The class hierarchy is also used to capture generalization and specialization information [15]. However, the similarity among instance values is not considered in the creation of the class hierarchy.

6.5 Summary and Future Work

In this chapter, we have presented a technique for "structurally" partitioning a large network of medical concepts. The technique has as its basis a similarity measure which is assigned to child/parent pairs in the vocabulary. We have shown the results of applying the approach to a large vocabulary, breaking it into meaningful subnets for the purpose of graphical display and comprehension. We have obtained an important result, namely, that the outcome of a human expert-based partitioning (e.g., a semantic partition) of a large vocabulary is quite similar in effect to the partitioning which is the result of applying our purely structural and statistical method.

In future work, we will further refine our similarity measures by in turn computing the similarity of every single partially overridden relationship. We expect

that this will improve both our control of subtree sizes and the agreement of our results with the human expert.

Another interesting idea is to combine the structural and semantic approaches. Vocabularies are slowly changing and important enough to justify an investment in their partitioning. On the other hand, the task of partitioning a vocabulary of thousands or ten-thousands of concepts into manageable groups can be overwhelming and by far too time consuming. As an expert is needed for the semantic partitioning, this can be very expensive! In short, the semantic approach by itself is not realistic for large vocabularies, and large vocabularies are the only interesting ones.

We can use our structural approach to help a domain expert focus more clearly on the problem. The results of the structural partitioning method described in this chapter can be given to the domain expert who can then work on partitioning the remaining large trees into smaller subtrees. The expert may also check on the validity of the structural partitioning and make improvements when deemed necessary.

CHAPTER 7
CONCLUSIONS

Controlled medical terminologies serve as excellent tools for the management of diverse terminologies within healthcare. They can help overcome differences in terminologies between different databases and information systems and different categories of users. Unfortunately, the job of understanding a large controlled medical vocabulary (e.g., the MED of CPMC) is itself difficult and time-consuming. The difficulty arises from the need to comprehend the extensive network of medical concepts and semantic links that forms the vocabulary.

Five papers, describing different approaches to enhance comprehensibility of CMTs, were presented in this dissertation. In the published journal paper [56] (Chapter 2), our experience with OODB modeling for the purpose of comprehending a controlled medical terminology has been reported. In particular, we have described a technique for mapping an existing semantic network-based controlled medical vocabulary into an equivalent OODB-based vocabulary by partitioning the vocabulary into sets of concepts with the same sets of properties (Section 2.3). We called the resulting OODB terminology the "OOHTR." The schema of the OOHTR, which captures the structure of the vocabulary, turns out to be a very compact representation relative to the size and scope of the original vocabulary. Because of this, it offers insights into the overall structure of the vocabulary and greatly aids in its comprehension. In fact, in Section 2.4, we have described how the schema was utilized for updating and redesigning operations as well as uncovering and correcting some errors and inconsistencies that had existed in the original vocabulary. An example of an error discovered through the use of the schema was the *Pancreatin_Area* intersection area class which was classified as a chemical and a medication. The OOHTR has been implemented as an ONTOS database and is currently up and running.

In order to enhance comprehension and improve navigation of the UMLS Metathesaurus, we also represented the UMLS as an OODB in Chapter 3 (a journal paper [58] is being submitted presently). The UMLS OODB modeling is based on the UMLS Semantic Network. Every semantic type has been represented as a semantic type class in the schema that we developed. A new kind of class, intersection class, has been introduced in this research in Section 3.2. As a result, the average number of concepts per semantic type class has been reduced from 5,000 to about 2,700. The average number of concepts in each intersection class is 100. These smaller extents make comprehension of the UMLS easier. To properly define superclasses of intersection classes, we defined *maximal subset* and *minimal superclass* in Section 3.3. Based on those two concepts, a rule to determine superclasses for all intersection classes has been developed. The resulting UMLS OODB schema is deeper and more refined than the original Semantic Network. The UMLS Metathesaurus has thus been classified into a large number of disjoint, uniform sets of concepts, which again helps with comprehension. Examples of how the intersection classes helped expose omissions of concepts, highlighted errors of semantic type classification, and uncovered ambiguities of concepts in the UMLS were presented in Section 3.4. For example, 100 intersection classes with only one instance have been checked. For only 11 intersection classes of these 100, we found the classification of concepts to be correct. For 55 of these intersection classes, the multiple classifications were wrong. For 32 intersection classes, the classified concepts indicated non-uniform classifications. Furthermore, we found two intersection classes which are redundant classification cases. Information on these problems has been forward to the National Library of Medicine for inclusion in future release of the UMLS.

In order to understand a large OODB schema, a new technique for modeling called *disciplined modeling* was described in Chapter 4, a journal paper [57] to be submitted shortly. (A preliminary version was published in [117].) The technique is

based on three rules which express limitations and refinements to the modeling of the schema. The three rules presented in Section 4.3 are (1) The equicontext relation between classes is an equivalence relation, that is, it satisfies the three conditions of an equivalence relation: reflexivity, symmetry and transitivity; (2) Two classes which are *category-of* specializations of the same superclass cannot both contain an instance representing the same real world object; (3) For each context there exists one class R which is the *major* (or definitive) class for this context such that every class in this context is a descendent of R. In Section 4.4, we proved the existence of a meaningful forest subhierarchy of a given specialization hierarchy if those rules are followed. Thus, based on our theoretical paradigm, a large OODB schema can be partitioned into meaningful units called contexts, which are subschemas of smaller sizes and lower complexities. This kind of partition of a schema aids the understanding of the original schema. In order to partition a large OODB schema into several meaningful subschemas, we needed to identify a forest subhierarchy of the original schema first. To find a meaningful forest hierarchy from a DAG is not a straightforward job. Extensive analysis was needed for decisions whether two classes are in the same context or not. In Section 4.5, we have presented an interactive methodology based on our theoretical framework to recognize a meaningful forest sub-hierarchy of an OODB schema (DAG). Our methodology relied on the interaction between a user (presumably the CMT designer or administrator) and a computer. The process required that a user refines the subclass relationships of an OODB schema so that it conforms to the above rules of *disciplined modeling*. After the refinement, the computer can automatically reduce the CMT to a forest structure. The methodology has been applied to the subschema of a university database in Section 4.6. 19 classes of the subschema of the university database were partitioned into five contexts. We also applied the methodology developed in this dissertation research to the MED

OODB schema in Section 4.7. The MED schema with 124 classes was divided into 51 subschemas.

As we mentioned above, an OODB schema representation of a controlled medical vocabulary helps to comprehend the original vocabulary. However, classes in an OODB schema may contain many instances. For example, each class in the MED schema contains on average about 500 concepts. Therefore, further comprehension efforts are needed. In Chapter 5 (a published journal paper [59]), the theoretical partitioning framework and the methodology developed for the schema level has been adapted to the object level. Based on the adapted theoretical partitioning framework, we reduced an entire vocabulary into a forest hierarchy composed of small trees, each representing a logical unit whose graphical representation can fit on a computer screen. Partitioning a vocabulary into small size units makes it easier for users and system designers alike to comprehend the contents of a vocabulary in a modular fashion. In Section 5.5, we have demonstrated our methodology by applying it to a complex subnetwork of the MED and InterMED (a partial revised version of the MED) [112, 129]. The complex subnetwork of the MED consisting of 39 concepts was partitioned into 18 contexts and the InterMED containing about 2,800 concepts was partitioned into 545 contexts.

As human expert judgement is an important ingredient of the above methodology and expert time is very expensive, the above methodology is quite costly for large vocabularies, which are the most interesting ones. Therefore, in this dissertation, we developed another approach that avoids the involvement of a human expert in the partitioning process. This material was presented in Chapter 6 (a paper [55] submitted to a major conference). Our approach is based on defining the similarity of a pair consisting of a child node and its parent node in the semantic network (Section 6.2). A distribution over these similarities for all pairs in the semantic network was computed. Based on this distribution, the semantic network has been

partitioned into small pieces, which are easier to understand and display. We have demonstrated this approach by partitioning the InterMED and a complex subnet of the MED in Section 6.3. The complex subnet of the MED containing 39 concepts was partitioned into 13 trees. The result of applying this approach is quite similar to result of a human expert-based partitioning approach described in Chapter 5.

REFERENCES

1. TOVE ontologies. URL: http://www.ie.utoronto.ca/EIL/tove/toveont.html (Jan. 9, 1998).

2. The WordNet. URL: http://www.cogsci.princeton.edu/wn/ (Oct. 1, 1998).

3. E. Agasi, R. I. Becker, and Y. Perl. A shifting algorithm for constrained min-max partition on trees. *Discrete Applied Mathematics*, 45:1–28, 1993.

4. R. Agrawal and N. H. Gehani. ODE (Object Database and Environment): The language and data model. In *Proc. 1989 ACM SIGMOD Int'l Conference on Management of Data*, pages 36–45, Portland, OR, May 1989. ACM.

5. A. V. Aho, J. E. Hopcroft, and J. D. Ullman. *Data Structures and Algorithms.* Addison-Wesley Publishing Company, Reading, MA, 1983.

6. American Medical Association, Chicago, IL. *Physicians' Current Procedural Terminology: CPT. 4th ed.*, 1998.

7. F. Baader and B. Hollunder. KRIS: Knowledge representation and inference system. *SIGART Bulletin*, 2(3):8–14, 1991.

8. S. Bayer and M. Vilain. The relation-based knowledge representation of King Kong. *SIGART Bulletin*, 2(3):15–21, 1991.

9. R. I. Becker and Y. Perl. Shifting algorithms for tree partitioning with general weighting functions. *J. Algorithms*, 4:101–120, 1983.

10. R. I. Becker and Y. Perl. The shifting algorithm technique for the partitioning of trees. *Discrete Applied Mathematics*, 62:15–34, 1995.

11. R. I. Becker, Y. Perl, and S. Schach. A shifting algorithm for min-max tree-partitioning. *J. ACM*, 29:56–67, 1982.

12. R. I. Becker and S. Schach. A bottom-up algorithm for weight- and height-bounded minimal partition of trees. *International J. Computer Math.*, 16:211–228, 1984.

13. T. J. M. Bench-Capon and P. Visser. Ontologies in legal information systems: the need for explicit specifications of domain conceptualisations. In *Proc. of the sixth International Conference on Artificial Intelligence and Law*, page 132, 1997.

14. F. W. Bergmann and J. J. Quantz. Parallel propagation in the description-logic system flex. In J. Geller, H. Kitano, and C. B. Suttner, editors, *Parallel Processing for Artificial Intelligence 3*, pages 181–207. North-Holland, New York, 1997.

15. E. Bertino and L. Martino. *Object-Oriented Database Systems, Concepts and Architectures.* Addison-Wesley Publishing Company, Reading, MA, 1993.

16. G. Booch. *Object-Oriented Analysis and Design with application.* The Benjamin/Cummings Publishing Company, Inc., Redwood City, CA, 1994.

17. A. Borgida, R. J. Brachman, D. L. McGuinness, and L. A. Resnick. CLASSIC: A structural data model for objects. *Proceedings of the 1989 ACM SIGMOD International Conference on the Management of Data, appeared as SIGMOD*, 18:58–67, 1989.

18. R. J. Brachman. On the epistemological status of semantic networks. In N. Findler, editor, *Associative Networks*, pages 3–50. Academic Press, New York, NY, 1979.

19. R. J. Brachman, R. E. Fikes, and H. J. Levesque. KRYPTON: A functional approach to knowledge representation. *IEEE Computer*, 16(10):67–73, 1983.

20. R. J. Brachman and H. J. Levesque. The tractability of subsumption in frame based description languages. In *Proceedings of AAAI-84*, pages 34–37, Austin, TX, 1984.

21. R. J. Brachman and J. Schmolze. An overview of the KL-ONE knowledge representation system. *Cognitive Science*, 9(2):171–216, 1985.

22. S. Buvač and R. Fikes. A declarative formalization of knowledge translation. In *CIKM'95*, pages 340–347, Baltimore, MD, 1995.

23. S. Buvač and I. M. Mason. Propositional logic of context. In *Proceedings of the 11th National Conference on Artificial Intelligence (AAAI-93)*, pages 412–419, Washington, DC, 1993.

24. K. E. Campbell. *Distributed development of a logical-based controlled medical terminology.* PhD thesis, Stanford University, CA, 1997. CS-TR-97-1596.

25. K. E. Campbell, S. P. Cohn, C. G. Chute, G. Rennels, and E. H. Shortliffe. Gálapagos: Computer-based support for evolution of a convergent medical terminology. In J. Cimino, editor, *Proc. 1996 AMIA Annual Fall Symposium*, pages 269–273, Washington, DC, October 1996.

26. K. E. Campbell, D. Oliver, K. A. Spackman, and E. H. Shortliffe. Representing thoughts, words, and things in the Unified Medical Language System. *Journal of the American Medical Informatics Association*, 5(5):421–431, 1998.

27. K. E. Campbell, D. E. Oliver, and E. H. Shortliffe. The Unified Medical Language System: Toward a collaborative approach for solving terminologic problems. *Journal of the American Medical Informatics Association*, 5(1):12–16, 1998.

28. G. Carenini and J. D. Moore. Using the UMLS semantic network as a basis for constructing a terminological knowledge base: a preliminary report. In *Proceedings of the Seventeenth Annual SCAMC*, pages 725–729, 1993.

29. W. Ceusters, J. Rogers, F. Consorti, and A. Rossi-Mori. Syntactic-semantic tagging as a mediator between linguistic representations and formal models: an exercise in linking SNOMED to GALEN. *Artificial Intelligence in Medicine*, 15(1):5–24, 1999.

30. P. P.-S. Chen. The Entity-Relationship Model: Toward a unified view of data. *TODS*, 1(1):9–36, 1976.

31. W. W. Chu, Q. Chen, and R. Lee. Cooperative query answering via type abstraction hierarchy. In *Proc. International Working Conference on Cooperating Knowledge Based Systems*, pages 271–290, Univerity of Keele, UK, October 1990.

32. W. W. Chu and K. Chiang. Abstraction of high level concepts from numerical values in databases. In *Proc. AAAI Workshop on Knowledge Discovery in Databases*, pages 133–144, Seattle, Washington, July 1994.

33. J. J. Cimino. Vocabulary and health care information technology: State of the art. *Journal of the American Society for Information Science*, 46(10):777–782, 1995.

34. J. J. Cimino. Review paper: Coding systems in health care. *Methods of Information in Medicine*, 35:273–284, 1996.

35. J. J. Cimino. Personal communication, 1997. CIS Department, NJIT, Newark, NJ.

36. J. J. Cimino and G. O. Barnett. Automated translation between medical terminologies using semantic definitions. *MD Comput.*, 7:104–109, 1990.

37. J. J. Cimino, P. D. Clayton, G. Hripcsak, and S. Johnson. Knowledge-based approaches to the maintenance of a large controlled medical terminology. *JAMIA*, 1(1):35–50, 1994.

38. J. J. Cimino, G. Hripcsak, S. Johnson, and P. D. Clayton. Designing an introspective, controlled medical vocabulary. In *Kingsland LC, ed. Proceedings of the Thirteenth Annual SCAMC*, pages 513–518, Washington, DC, 1989. IEEE Computer Society Press.

39. J. J. Cimino, G. Hripcsak, S. Johnson, C. Friedman, D. Fink, and P. D. Clayton. UMLS as knowledge base - a rule-based expert system approach to controlled medical vocabulary management. In *Proceedings of the Fourteenth Annual Symposium on Computer Applications in Medical Care*, pages 175–179, CA, 1990. IEEE Computer Society Press.

40. J. J. Cimino, G. Hripcsak, S. B. Johnson, and P. D. Clayton. Designing an introspective, multipurpose, controlled medical vocabulary. In *Proc. Thirteenth Annual Symposium on Computer Applications in Medical Care*, pages 513–517, Washington, DC, November 1989.

41. College of American Pathologists, Chicago, IL. *SNOP: The Systematized Nomenclature of Pathology*, 1965.

42. College of American Pathologists, Skokie, IL. *The Systematized Nomenclature of Medicine*, 1982.

43. College of American Pathologists, Northfield, IL. *The Systematized Nomenclature of Human and Veterinary Medicine: SNOMED International*, 1996.

44. B. Everitt. *Cluster Analysis.* Heinemann Educational Books, London, UK, 1980.

45. D. H. Fischer. Consistency rules and triggers for thesauri. *Int. Classif.*, 18(4):212–225, 1991.

46. D. H. Fischer. Consistency rules and triggers for multilingual terminology. In *Proc. TKE93, Terminology and Knowledge Engineering*, pages 333–342, 1993.

47. M. S. Fox, J. F. Chionglo, and F. G. Fadel. A common sense model of the enterprise. In *Proceedings of the 2nd Industrial Engineering Research Conference*, pages 425–429, 1993.

48. M. S. Fox and M. Gruninger. An organisation ontology for enterprise modelling: Preliminary concepts for linking structure and behaviour. *Computers in Industry*, 29:123–134, 1996.

49. M. S. Fox and M. Gruninger. Enterprise modelling. *AI Magazine*, 19:109–121, 1998.

50. M. R. Gary and D. S. Johnson. *Computers and Intractability.* Freeman, New York, 1979.

51. J. Geller, Y. Perl, and E. Neuhold. Structure and semantics in OODB class specifications. *SIGMOD Record*, 20(4):40–43, 1991.

52. R. M. Gregor. A deductive pattern matcher. In *Seventh National Conference on Artificial Intelligence*, pages 403–408. Morgan Kaufmann, San Mateo, CA, 1988.

53. R. M. Gregor. The evolving technology of classification-based knowledge representation systems. In F. Lehman, editor, *Semantic Networks in Artificial Intelligence*, pages 385–400. Pergamon Press, Oxford, UK, 1992.

54. H. Gu, J. J. Cimino, M. Halper, J. Geller, and Y. Perl. Utilizing OODB schema modeling for vocabulary management. In J. J. Cimino, editor, *Proc. '96 AMIA Annual Fall Symposium*, pages 274–278, Washington,DC, October 1996.

55. H. Gu, J. Geller, M. Halper, and L. Liu. Using a similarity measurement to partition a vocabulary of medical concepts. submitted for conference publication, 1999.

56. H. Gu, M. Halper, J. Geller, and Y. Perl. Benefits of an OODB representation for controlled medical terminologies. To appear in *JAMIA*, 1999.

57. H. Gu, Y. Perl, and J. Geller. Partitioning an OODB schema into contexts by identifying a forest hierarchy. In preparation for journal paper, 1999.

58. H. Gu, Y. Perl, J. Geller, M. Halper, L. Liu, and J. J. Cimino. Representing the UMLS as an OODB: Modeling issues and advantages. Submitted for Journal publication, 1999.

59. H. Gu, Y. Perl, J. Geller, M. Halper, and M. Singh. A methodology for partitioning a vocabulary hierarchy into trees. *Artificial Intelligence in Medicine*, 15(1):77–98, 1999.

60. R. V. Guha. *Contexts: A Formalization and Some Applications.* PhD thesis, Computer Science Department, Stanford University, CA, 1991.

61. R. V. Guha and D. B. Lenat. Enabling agents to work together. *Communications of the ACM*, 37(6):127–142, 1994.

62. T. R. Gurber. ONTOLINGUA: A mechanism to support portable ontologies. Technical Report KSL-91-68, Knowledge Systems Laboratory, Stanford University, CA, 1991.

63. T. R. Gurber. A translation approach to portable ontology specifications. *Knowledge Acquisition*, 5:199–220, 1993.

64. T. R. Gurber. Toward principles for the design of ontologies used for knowledge sharing. *International Journal of Human-Computer Studies*, 43:907–928, 1995.

65. U. Hahn, M. Romacker, and S. Schulz. How knowledge drives understanding–matching medical ontologies with the needs of medical language processing. *Artificial Intelligence in Medicine*, 15(1):25–52, 1999.

66. M. Halper, J. Geller, Y. Perl, and E. J. Neuhold. A graphical schema representation for object-oriented database. In R. Cooper, editor, *Workshop on Interfaces in Database Systems (IDS-92)*, pages 282–307. Springer Verlag, London, 1993.

67. Health Care Financing Administration, Washington, DC. *International Classification of Diseases: 9th Revision, Clinical Modification: ICD-9-CM. 4th ed.*, 1991.

68. G. G. Hendrix. Expanding the utility of semantic networks through partitioning. In *Proc. IJCAI-75*, pages 115–121, Tabilisi, Georgia, USSR, 1975.

69. G. G. Hendrix. Encoding knowledge in partitioned networks. In N. V. Findler, editor, *Associative Networks: Representation and Use of Knowledge by Computers*, pages 51–92. Academic Press, Inc., New York, NY, 1979.

70. G. Hripcsak, B. Allen, J. J. Cimino, and R. Lee. Access to data: comparing AccessMed to Query by review. *JAMIA*, 3(4):288–299, 1996.

71. B. L. Humphreys and D. A. B. Lindberg. Building the Unified Medical Language System. In *Proc. Thirteenth Annual Symposium on Computer Applications in Medical Care*, pages 475–480, Washington, DC, November 1989.

72. B. L. Humphreys and D. A. B. Lindberg. The Unified Medical Language System project: a distributed experiment in improving access to biomedical information. *Methods of Information in Medicine*, 7(2):1496–1500, 1992.

73. B. L. Humphreys, D. A. B. Lindberg, H. M. Schoolman, and G. O. Barnett. The Unified Medical Language System: An informatics research collaboration. *Journal of the American Medical Informatics Association*, 5(1):1–11, 1998.

74. L. Iwanska. Context in natural language processing. In *Working Notes of Workshop W13, IJCAI.* Montreal, Canada, 1995.

75. M. Joubert, M. Fieschi, and J. J. Robert. A conceptual model for information retrieval with UMLS. In *Proceedings of the Seventeenth Annual Symposium on Computer Applications in Medical Care*, pages 715–719, 1993.

76. M. Joubert, M. Fieschi, J. J. Robert, and A. Tafazzoli. Users conceptual views on medical information databases. *International Journal of Biomed Comput*, 37(2):93–104, 1994.

77. M. Joubert, J. J. Robert, and M. Fieschi. The project ARIANE: conceptual queries to information databases. In *Proc. '96 AMIA Annual Fall Symposium*, pages 378–382, Washington, DC, 1996.

78. J. L. Kannry, L. Wright, M. Shifman, S. Silverstein, and P. L. Miller. Portability issues for a structured clinical vocabulary: mapping from yale to the columbia medical entities dictionary. *JAMIA*, 3:66–78, 1996.

79. W. Kim and F. H. Lochovsky, editors. *Object-Oriented Concepts, Databases, and Applications.* ACM Press, New York, NY, 1989.

80. A. Kobsa. First experience with the SB-ONE knowledge representation workbench in natural-language applications. *SIGART Bulletin*, 2(3):70–76, 1991.

81. S. Kundu and J. Misra. A linear tree-partitioning algorithm. *SIAM J. Comput.*, 6:131–134, 1977.

82. C. Lamb, G. Landis, J. Orenstein, and D. Weinreb. The ObjectStore database system. *Communications of the ACM*, 34(10):50–63, 1991.

83. F. Lehmann. Semantic networks. In F. Lehmann, editor, *Semantic Networks in Artificial Intelligence*, pages 1–50. Pergamon Press, Tarrytown, NY, 1992.

84. F. Lehmann, editor. *Semantic Networks in Artificial Intelligence.* Pergamon Press, Tarrytown, NY, 1992.

85. D. B. Lenat. CYC: toward programs with common sense. *Communications of the ACM*, 33(8):30–49, 1990.

86. D. B. Lenat. CYC: A large-scale investment in knowledge infrastructure. *Communications of the ACM*, 38(11):33–38, 1995.

87. D. B. Lenat and R. V. Guha. *Building Large Knowledge-Based Systems: Representation and Inference in the CYC project.* Addison-Wesley, Reading, MA, 1990.

88. R. Levinson. Pattern associativity and the retrieval of semantic networks. In F. Lehmann, editor, *Semantic Networks in Artificial Intelligence*, pages 573–600. Pergamon Press, Tarrytown, NY, 1992.

89. D. A. B. Lindberg, B. L. Humphreys, and A. T. McCray. The Unified Medical Language System. *Methods of Information in Medicine*, 32:281–291, 1993.

90. L. Liu, M. Halper, J. Geller, and Y. Perl. Controlled vocabularies in OODBs: Modeling issues and implementation. To appear in *Distributed and Parallel Databases*, 1999.

91. L. Liu, M. Halper, H. Gu, J. Geller, and Y. Perl. Modeling a vocabulary in an object-oriented database. In *CIKM'96*, pages 179–188, Rockville, Maryland, 1996.

92. M. Lucertini, Y. Perl, and B. Simeone. Most uniform path partitioning and its use in image processing. *Discrete Applied Mathematics*, 42:227–256, 1993.

93. E. Mays, R. Weida, R. Dionne, M. Laker, B. White, C. Liang, and F. J. Olse. Scalable and expressive medical terminologies. In J. Cimino, editor, *Proc. 1996 AMIA Annual Fall Symposium*, pages 259–263, Washington, DC, October 1996.

94. J. McCarthy. Notes on formalizing context. In *13th International Joint Conference on Artificial Intelligence*, pages 555–560, Chambery, France, 1993.

95. A. T. McCray. UMLS semantic network. In *Proceedings of the Thirteenth Annual SCAMC*, pages 503–507, 1989.

96. A. T. McCray. Representing biomedical knowledge in the UMLS semantic network. In N. C. Broering, editor, *High-performance medical libraries: advances in information management for the virtual era.*, pages 45–55. Meckler, Westport, CT, 1993.

97. A. T. McCray and W. T. Hole. The scope and structure of the first version of the UMLS semantic network. In *Proceedings of the Fourteenth Annual SCAMC*, pages 126–130, 1990.

98. A. T. McCray and S. J. Nelson. The representation of meaning in the UMLS. *Methods of Information in Medicine*, 34:193–201, 1995.

99. A. Mehta, J. Geller, Y. Perl, and P. Fankhauser. Computing access relevance for path-method generation in OODBs and IM-OODB. *Journal of Intelligent Information System*, 7(1):75–100, 1996.

100. A. Mehta, J. Geller, Y. Perl, and Erich Neuhold. The OODB Path–Method Generator (PMG) using access weights and precomputed access relevance. *VLDB Journal*, 7(1):25–47, 1998.

101. M. Merzbacher and W. W. Chu. Pattern-based clustering for database attribute values. In *Proc. AAAI Workshop on Knowledge Discovery in Databases*, pages 291–298, Washington, DC, July 1993.

102. R. S. Michalski and R. E. Stepp. Clustering. In Stuart C. Shapiro, editor, *The Encyclopedia of Artificial Intelligence*. John Wiley & Sons, New York, NY, second edition, 1992.

103. G. A. Miller. WordNet: A lexical database for English. *Communications of the ACM*, 38(11):39–41, 1995.

104. W. Möhr and L. Rostek. TEDI: An object-oriented terminology editor. In *Proc. TKE'93, Terminology and Knowledge Engineering*, pages 363–374, 1993.

105. National Library of Medicine, Bethesda, MD. *Medical Subject Headings*, 1997. Updated annually.

106. B. Nebel and K. von Luck. Issues of integration and balancing in hybrid knowledge. In K. Morik, editor, *GWAI-87*, pages 114–123. Springer Verlag, Berlin, Germany, 1987.

107. S. J. Nelson, L. F. Fuller, M. S. Erlbaum, M. S. Tuttle, D. D. Sherertz, and N. E. Olson. The semantic structure of the UMLS metathesaurus. In *Proceedings of the Sixteenth Annual SCAMC*, pages 649–653, 1992.

108. S. J. Nelson, M. S. Tuttle, W. G. Cole, D. D. Sherertz, W. D. Sperzel, M. S. Erlbaum, L. F. Fuller, and N. E. Olson. From meaning to term: semantic locality in the UMLS metathesaurus. In *Proceedings of the Fifteenth Annual SCAMC*, pages 209–213, 1991.

109. E. J. Neuhold, J. Geller, Y. Perl, and V. Turau. Separating structural and semantic elements in object-oriented knowledge bases. In *Advanced Database System Symposium*, pages 67–74, Kyoto, Japan, 1989.

110. E. J. Neuhold, J. Geller, Y. Perl, and V. Turau. A theoretical underlying Dual Model for knowledge-based systems. In *Proceedings of the First International Conference on Systems Integration*, pages 96–103, Morristown, NJ, 1990.

111. E. J. Neuhold and M. Schrefl. Dynamic derivation of personalized views. In *VLDB'88*, pages 183–194, Long Beach, CA, 1988.

112. D. Oliver and E. Shortliffe. Collaborative model development for vocabulary and guidelines. In J. J. Cimino, editor, *Proc. '96 AMIA Annual Fall Symposium*, page 826, Washington,DC, 1996.

113. D. E. Oliver and Y. Shahar. Development of a change model for a controlled medical vocabulary. In D. R. Masys, editor, *Proc. '97 AMIA Annual Fall Symposium*, pages 605–609, Nashville, TN, 1997.

114. D. E. Oliver, Y. Shahar, E. H. Shortliffe, and M. A. Musen. Representation of change in controlled medical terminologies. *Artificial Intelligence in Medicine*, 15(1):53–76, 1999.

115. P. Patel-Schneider, D. L. McGuiness, R. J. Brachman, L. A. Resnick, and A. Borgida. The CLASSIC knowledge representation system: guiding principles and implementation rationale. *SIGART Bulletin*, 2(3):108–113, 1991.

116. Y. Perl and J. Geller. Using object-oriented databases to make medical vocabularies comprehensible. *NJIT Research*, 5, 1997.

117. Y. Perl, J. Geller, and H. Gu. Identifying a forest hierarchy in an OODB specialization hierarchy satisfying disciplined modeling. In *Proc. CoopIS'96*, pages 182–195, Brussels, Belgium, 1996.

118. Y. Perl and S. Schach. Max-min tree-partitioning. *J. ACM*, 28:5–15, 1981.

119. M. R. Quillian. Semantic memory. In M. L. Minsky, editor, *Semantic Information Processing*, pages 227–270. The MIT Press, Cambridge, MA, 1968.

120. A. Rector. Coordinating taxonomies: Key to re-usable concept representations. In Mario Stefanelli Pedro Barahona and Jeremy Wyatt, editors, *Artificial Intelligence in Medicine*, pages 17–28. Springer, Berlin, Germany, 1995.

121. A. Rector, S. Bechhofer, C. Goble, I. Horrocks, W. Nowlan, and W. Solomon. The GRAIL concept modelling language for medical terminology. *Artificial Intelligence in Medicine*, 9:139–171, 1997.

122. D. Robinson, D. Comp, E. Schulz, P. Brown, and C. Price. Updating the read codes: User-interactive maintenance of a dynamic clinical vocabulary. *Journal of the American Medical Informatics Association*, 4(6):465–472, 1997.

123. R. A. Roocha, S. M. Huff, P. J. Haug, and H. R. Warner. Designing a controlled medical vocabulary server: The voser project. *Computers and Biomedical Research*, 27:472–507, 1994.

124. J. Rumbaugh, M. Blaha, W. Premerlani, F. Eddy, and W. Lorensen. *Object-Oriented Modeling and Design.* Prentice-Hall Inc., Englewood Cliffs, NJ 07632, 1991.

125. P. L. Schuyler, W. T. Hole, M. S. Tuttle, and D. D. Sherertz. The UMLS metathesaurus: representing different views of biomedical concepts. *Bull Med Libr Assoc*, 81(2):217–222, 1993.

126. S. C. Shapiro and W. J. Rapaport. SNePS considered as a fully intensional propositional semantic network. In N. Cercone and G. McCalla, editors, *The Knowledge Frontier*, pages 262–315. Springer Verlag, New York, NY, 1987.

127. S. C. Shapiro and W. J. Rapaport. The SNePS family. In Fritz Lehmann, editor, *Semantic Networks in Artificial Intelligence*, pages 243–275. Pergamon Press, Tarrytown, NY, 1992.

128. Y. Shoham. *Varieties of Context Artificial Intelligence and Mathematical Theories of Computation.* Academic Press, London, 1991.

129. E. Shortliffe, G. Barnett, J. J. Cimino, R. Greenes, S. Huff, and V. Patel. Collaborative medical informatics research using the Internet and the World Wide Web. In *Proc. '96 AMIA Annual Fall Symposium*, pages 125–129, Washington, DC, 1996.

130. P. H. A. Sneath and R. R. Sokal. *Numerical Taxonomy: The Principles and Practice of Numerical Classification.* W. H. Freeman and Company, San Francisco, CA, 1973.

131. V. Soloviev. An overview of three commercial object-oriented database management systems: ONTOS, ObjectStore, and O_2. *SIGMOD Record*, 21(1):93–104, 1992.

132. J. F. Sowa, editor. *Conceptual Structures, Information Processing in Mind and Machine.* Addison-Wesley Publishing Co., Inc., Reading, MA, 1984.

133. J. F. Sowa, editor. *Principles of Semantic Networks.* Morgan Kaufmann Publishers, Inc., San Mateo, CA, 1991.

134. K. A. Spackman, K. E. Campbell, and R. A. Côté. SNOMED RT: A reference terminology for health care. In D. R. Masys, editor, *Proc. 1997 AMIA Annual Fall Symposium*, pages 640–644, Nashville, TN, 1997.

135. O. N. Suarez-Munist, M. S. Tuttle, N. E. Olson, M. S. Erlbaum, D. D. Sherertz, S. S. Lipow, and et al. MEME II supports the cooperative management of terminology. In J. Cimino, editor, *Proc. 1996 AMIA Annual Fall Symposium*, pages 84–88, Washington, DC, October 1996.

136. M. S. Tuttle and S. J. Nelson. The role of the UMLS in 'storing' and 'sharing' across systems. *International Journal of Bio-Medical Computing*, 34:207–237, 1994.

137. M. S. Tuttle, S. J. Nelson, L. F. Fuller, D. D. Sherertz, M. S. Erlbaum, W. D. Sperzel, N. E. Olson, and O. N. Suarez-Munist. The semantic foundations of the UMLS metathesaurus. *Medinfo*, 7(2):1506–1511, 1992.

138. M. S. Tuttle, D. D. Sherertz, M. S. Erlbaum, et al. Adding your terms and relationships to the UMLS metathesaurus. In P. D. Clayton, editor, *Proceedings of the Fifteenth Annual SCAMC*, pages 219–223, Washington, D.C., 1991.

139. M. S. Tuttle, D. D. Sherertz, N. E. Olson, M. S. Erlbaum, W. D. Sperzel, L. F. Fuller, and S. J. Nelson. Using meta-1 the first version of the UMLS metathesaurus. In *Proceedings of the Fourteenth Annual SCAMC*, pages 131–135, 1990.

140. US Dept. of Health and Human Services, NIH, National Library of Medicine. *Unified Medical Language System*, 1998.

141. M. Vilain. The restricted language architecture of a hybrid representation system. In *Proceedings of the Ninth International Joint Conference on Artificial Intelligence*, pages 547–551. Morgan Kaufmann, San Mateo, CA, 1985.

142. P. H. Winston. Learning structural descriptions from examples. In R. J. Brachman and H. J. Levesque, editors, *Readings in Knowledge Representation*, pages 141–168. Morgan Kaufmann, Los Altos, California, 1985.

143. W. A. Woods. What's in a link: Foundations for semantic networks. In D. G. Bobrow and A. M. Collins, editors, *Representation and Understanding*, pages 35–82. Academic Press, New York, NY, 1975.

144. W. A. Woods. What's in a link: Foundations for semantic networks. In R. J. Brachman and Hector J. Levesque, editors, *Readings in Knowledge Representation*, pages 218–241. Morgan Kaufmann Publishers, Inc., San Mateo, CA, 1985.

145. W. A. Woods. Knowledge representation: What's important about it? In N. Cercone and G. McCalla, editors, *The Knowledge Frontier*, pages 44–79. Springer Verlag, New York, NY, 1987.

146. W. A. Woods. Understanding subsumption and taxonomy: A framework for progress. In J. F. Sowa, editor, *Principles of Semantic Networks*, pages 45–94. Morgan Kaufmann Publishers, Inc., San Mateo, CA, 1991.

147. W. A. Woods and J. G. Schmolze. The kl-one family. In F. Lehmann, editor, *Semantic Networks in Artificial Intelligence*, pages 133–177. Pergamon Press, Oxford, UK, 1992.

148. S. B. Zdonik and D. Maier, editors. *Readings in Object-Oriented Database Systems*. Morgan Kaufmann Publishers, Inc., San Mateo, CA, 1990.

149. J. Zhang. Application of OODB and SGML techniques in text database: An electronic dictionary system. *SIGMOD Record*, 24(1):3–8, March 1995.