

Auditing the UMLS For Redundant Classifications

Yi Peng, Michael Halper¹, Yehoshua Perl, James Geller

CS Dept., New Jersey Institute of Technology, Newark, NJ 07102

¹Mathematics & Computer Science Dept., Kean University, Union, NJ 07083

The UMLS's Semantic Network (SN) serves as a valuable abstraction for the underlying concept repository called the Metathesaurus (META). Specifically, the SN forms a classification layer for the META, with each of the META's constituent concepts assigned to one or more semantic types in the SN. The rule in the design of the SN is to have concepts explicitly assigned to the lowest possible semantic types in the SN's IS-A hierarchy. Implicit assignment to higher semantic types can be inferred via the IS-A relationships. However, in subsequent versions of the UMLS, unnecessary, simultaneous assignments to descendant and ancestor semantic types have been discovered (e.g., 8,622 in the UMLS 1998 version and 12,657 in the 2001 version). The assignment of concepts to such ancestor semantic types is called redundant classification. There is a need for an automated auditing tool that can identify all these redundant classifications. In this paper, an efficient algorithm for this auditing task is introduced. Details of its application to the current (2001) version of the UMLS are presented and the results are discussed.

INTRODUCTION

The Unified Medical Language System (UMLS) [1,2] is an important consolidation of a number of well established biomedical terminologies. One knowledge source of the UMLS is its concept repository, called the Metathesaurus (META), currently consisting of about 790,000 concepts drawn from 99 constituent terminologies [3]. Another knowledge source is its Semantic Network (SN) [4,5] that serves as an abstraction for the vast content of the META. The SN is composed of 134 semantic types arranged in a hierarchy consisting of two trees, one rooted at the semantic type **Entity**¹ and the other at **Event**. The SN's hierarchy is based on the IS-A relationship, which connects a pair of semantic types: a child to its parent. We will be using additional family terminology such as ancestor and descendant with respect to the SN's IS-A hierarchy. The descendant relationship is the transitive closure of the child relationship.

The relationship between the SN and the META is such that each concept in the META is explicitly assigned one or more semantic types from the SN. These assignments were carried out by domain experts in the respective areas. Due to the way the UMLS was created, it is unavoidable that some erroneous classifications have been introduced. These may have been a result of the integration of many terminological sources, which are not necessarily consistent. Or they may have

been due to the various backgrounds and viewpoints of the many experts contributing to the UMLS and classifying its concepts. It is an ongoing challenge facing the NLM to audit the UMLS and uncover and correct these erroneous classifications. However, due to the huge size and complexity of the META, such a comprehensive audit is an overwhelming task. Thus, there is a need to design auditing techniques that will minimize the effort and maximize the probability of finding these erroneous classifications. (For semantic techniques for auditing, see [6].)

Of particular interest in this paper is the error of *redundant classification*. This error is a violation of an important rule promoted by the SN's designers. Specifically, it was stated that a concept should be explicitly assigned to the lowest (most specialized) possible semantic type in the SN's IS-A hierarchy [7]. Implicit assignment to higher types can then be inferred via the IS-A relationships. For example, a concept should not be assigned to both a child semantic type and its parent. Assignment to the child is sufficient, as assignment to the parent can be inferred.

Admittedly, there are situations where redundant classifications could have been caused by ambiguity with respect to the source terminologies. For example, a concept x in one terminology may have been ascribed a more general semantics than the same concept x in another terminology. In translation to the UMLS, multiple categorizations for x may then have been derived independently from both source terminologies, thus leading to the redundancy. Additionally, it may be the case that an assignment of a concept to an ancestor semantic type is correct while its simultaneous assignment to a descendent semantic type is not. In such a case, a removal of the concept from the more specific type is required.

In previous work [8], we have discovered a number of redundant classifications, which were subsequently reported to the NLM. In this paper, we present an efficient algorithm that can find all redundant classifications. It works automatically, without the need for any domain expert intervention. This algorithm can help to guarantee that no redundant classifications will appear in future releases of the UMLS.

REDUNDANT CLASSIFICATION

In this section, we formally define the notion of redundant classification and present some examples from the 2001 version of the UMLS. We will be using the following.

Definition (Intersection): An intersection of two or more semantic types is the set of concepts that have

¹A bold font will be used to denote semantic types; italics will be used for concepts.

been *explicitly assigned* to each of these semantic types. \square

We will be using the standard mathematical notation to denote an intersection. E.g., the intersection of the three semantic types **Carbohydrate**, **Pharmacologic Substance**, and **Antibiotic** is denoted **Carbohydrate** \cap **Pharmacologic Substance** \cap **Antibiotic**. This intersection happens to contain a single concept, *Erymax*. If an intersection does not contain any concepts, it is said to be empty and is denoted with \emptyset .

Definition (Redundant Classification): Let concept c be a member of the intersection of two semantic types **B** \cap **A** such that **B** is a descendant of **A** in the SN. Then the classification of c to **A** is called a *redundant classification* since it can be inferred from the classification of c to **B**. \square

As a rule [7], redundant classifications should not appear in the UMLS. Those that do, should be removed.

As an example, the two concepts *Tryplosan* and *Triton* belong to the intersection **Chemical Viewed Functionally** \cap **Pharmacologic Substance**. **Chemical Viewed Functionally** is the parent of **Pharmacologic Substance**. Thus, the two concepts have redundant classifications to **Chemical Viewed Functionally**, and these classifications should be removed.

As another example, the concept *Sacred Bark* belongs to **Pharmacologic Substance** \cap **Substance** \cap **Tissue**. Because **Substance** is an ancestor of **Pharmacologic Substance**, the concept *Sacred Bark* is redundantly classified as **Substance**. Note that *Sacred Bark*'s classification to **Tissue** is not a redundant classification since there is no ancestor/descendant relationship between **Pharmacologic Substance** and **Tissue**. On the other hand, it is an error since *Sacred Bark* is an herb and cannot be a tissue.

METHODS

Our purpose is to provide the NLM with an effective tool to weed out the redundant classifications. In this section, we describe an efficient algorithm for this purpose.

In [8], our technique for finding redundant classifications was based on an exhaustive search of all possible intersections in the SN. In this paper, we present a more efficient procedure for this purpose. Toward that end, we need the following.

Definition (Related): Two semantic types are called *related* if one of them is a descendant of the other. \square

Definition (Independent): Two semantic types that are not related are *independent*. \square

For the purpose of finding redundant classifications, it is not necessary to examine intersections of independent semantic types. By the definition of redundant classification, it is enough to concentrate on related semantic types and check if their intersections

are not empty. E.g., we need to check the intersection **Human-caused Phenomenon or Process** \cap **Environmental Effect of Humans** because the latter is a child of the former. But we do not need to check **Human-caused Phenomenon or Process** \cap **Injury or Poisoning** because these semantic types are independent.

Furthermore, it is sufficient to review only binary intersections of related semantic types in order to uncover all redundant classifications. There is no need to examine ternary intersections, quaternary intersections, etc. To see this, consider, e.g., the concept *4-MBFG*. It belongs to the ternary intersection **Chemical** \cap **Organic Chemical** \cap **Amino Acid, Peptide, or Protein** (see Fig. 1). Because **Amino Acid, Peptide, or Protein** is a descendant of both **Chemical** and **Organic Chemical**, the classifications of *4-MBFG* to **Chemical** and **Organic Chemical** are redundant classifications. These two redundant classifications can be identified by examining the binary intersections **Chemical** \cap **Amino Acid, Peptide, or Protein** and **Organic Chemical** \cap **Amino Acid, Peptide, or Protein**. The fact that we only need to consider binary intersections reduces the complexity of the algorithm that will be presented below.

Note that the redundant classification of *4-MBFG* to **Chemical** is actually found a second time (a *duplicate* redundant classification) when the intersection **Chemical** \cap **Organic Chemical** is considered. Duplicate redundant classifications need to be removed only once. (See Column 3 of Table 1.)

The following definition is based on the SN being a two-tree structure.

Definition (Depth): The depth of a semantic type **A**, denoted $d(\mathbf{A})$, is the number of **A**'s ancestor semantic types in the SN. \square

For example, $d(\mathbf{Organic\ Chemical}) = 5$ (see Fig. 1). To examine all binary intersections involving related semantic types, it is sufficient to examine for each semantic type **A**, its binary intersections with its $d(\mathbf{A})$ ancestors. The reason is that the intersection **A** \cap **B** with a descendant **B** of **A** will be considered when processing focuses on **B** with all its ancestors, including **A**. Hence, the number of binary intersections (involving related semantic types) to be examined is:

$$\sum_{\mathbf{T} \in \text{SN}} d(\mathbf{T}),$$

where **T** runs over all the semantic types in the SN.

The following is the pseudocode for the algorithm that uncovers and removes all redundant classifications from the UMLS.

```
for(all T  $\in$  SN (excluding Entity and Event))
{
  A = T;
  repeat
  {
    // Set A to be its parent.
    A = IS-A(A);
    // Are there any redundant classifications
```

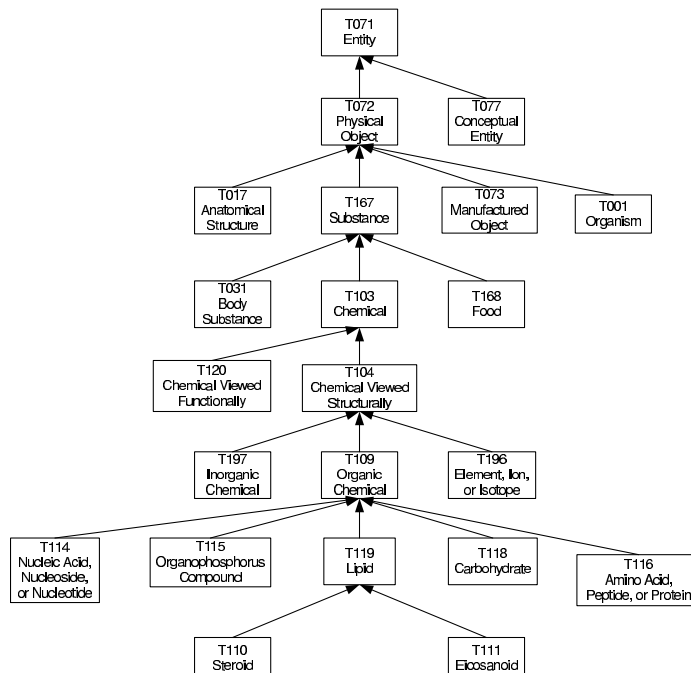


Figure 1: Portion of the **Entity** tree of the SN

```
// associated with the intersection  $A \cap T$ ?
if( $A \cap T \neq \emptyset$ )
{
  // Yes; remove the redundant
  // classifications to  $A$  from the UMLS.
  for(all concepts  $c \in A \cap T$ )
  {
    Remove the explicit classification of
    concept  $c$  to semantic type  $A$  from
    the UMLS.
  }
}
} until( $A = \text{Entity}$  OR  $A = \text{Event}$ )
} // for  $T$ 
```

The statement $A = \text{IS-A}(A)$ means that A is assigned to be its parent. This assignment to the parent can be done in an implementation using the “tree number” of A provided as part of the SN [9]. Note that this assignment is unique since each semantic type (except for **Entity** and **Event**) has exactly one parent in the SN.

Let us demonstrate the operation of the algorithm. Suppose T is currently **Steroid** (see Fig. 1). According to the algorithm, in order to find redundant classifications involving **Steroid**, we need to examine all the binary intersections of **Steroid** and its ancestors. In the first iteration of the loop, A is made the parent of **Steroid** which is **Lipid** (Fig. 1). It is found that $\text{Steroid} \cap \text{Lipid} \neq \emptyset$. In fact, 53 concepts reside in this intersection (see Table 1, Column 4). Thus, the redundant classifications of all these 53 concepts to **Lipid** are removed.

In the second iteration, A becomes **Organic Chemical**, the parent of **Lipid**. The intersection $\text{Organic Chemical} \cap \text{Steroid}$ has 771 concepts in it (Table 1). These 771 redundant classifications of concepts to **Organic Chemical** are removed.

During the third iteration, A is set to **Chemical Viewed Structurally**, the parent of **Organic Chemical**. In this case, as well, the intersection $\text{Chemical Viewed Structurally} \cap \text{Steroid} \neq \emptyset$. It has 32 concepts. The resolution of these redundant classifications requires that the assignments of the 32 concepts to **Chemical Viewed Structurally** be removed.

As the loop proceeds through **Chemical**, **Substance**, etc. toward the root **Entity**, no more redundant classifications involving **Steroid** are found. Hence, all redundant classifications involving **Steroid** have been resolved. The algorithm then goes on to the next semantic type in the SN.

RESULTS

In [8], while reviewing all intersections of semantic types in the SN of the 1998 version of the UMLS, we discovered that 8,622 concepts had redundant classifications. This group of redundant classifications was reported to the NLM so they could be omitted in subsequent releases. Recently, a follow-up audit was performed on the 2001 UMLS to determine the status of these 8,622 concepts. It was found that a portion of the redundant classifications were properly removed. However, a number of them were still present. E.g., the concept *Z-L(3)VS* belongs to the intersection $\text{Organic Chemical} \cap \text{Amino Acid, Peptide, or Protein} \cap \text{Pharmacologic Substance}$, both in the 1998 and 2001 versions. This is a redundant classification to **Organic Chemical** because **Amino Acid, Peptide, or Protein IS-A Organic Chemical**.

A third portion of the redundant classifications were partially treated. For instance, an existing redundant

Ancestor Semantic Type (AST)	Number of RCs to AST	Number of Non-Duplicate RCs to AST	Descendant (Number of concepts in intersection of AST and Descendant)	
Biologic Function	1	1	Organism Function (1)	
Entity	2	2	Biologically Active Substance (1)	
			Intellectual Product (1)	
Chemical	458	278	Neuroreactive Substance or Biogenic Amine (1)	
			Indicator, Reagent, or Diagnostic Acid (1)	
			Biologically Active Substance (2)	
			Hazardous or Poisonous Substance (2)	
			Lipid (3)	
			Organic Chemical (40)	
			Nucleic Acid, Nucleoside, or Nucleotide (83)	
			Pharmacologic Substance (137)	
Amino Acid, Peptide, or Protein (189)				
Chemical Viewed Structurally	1,407	1,360	Eicosanoid (10)	
			Organophosphorus Compound (14)	
			Steroid (32)	
			Element, Ion, or Isotope (35)	
			Lipid (60)	
			Inorganic Chemical (61)	
			Carbohydrate (87)	
			Nucleic Acid, Nucleoside, or Nucleotide (133)	
			Amino Acid, Peptide, or Protein (157)	
Organic Chemical (818)				
Organic Chemical	10,396	10,185	Eicosanoid (38)	
			Organophosphorus Compound (645)	
			Steroid (771)	
			Nucleic Acid, Nucleoside, or Nucleotide (1,120)	
			Lipid (1,186)	
			Carbohydrate (2,403)	
Amino Acid, Peptide, or Protein (4,233)				
Lipid	64	64	Eicosanoid (11)	
			Steroid (53)	
Chemical Viewed Functionally	18	17	Biomedical or Dental Material (3)	
			Pharmacologic Substance (15)	
Pharmacologic Substance	163	163	Antibiotic (163)	
Biologically Active Substance	147	146	Neuroreactive Substance or Biogenic Amine (1)	
			Vitamin (2)	
			Hormone (5)	
			Enzyme (23)	
Immunologic Factor (116)				
Substance	1	1	Pharmacologic Substance (1)	
Total:	10	12,657	12,217	40

Table 1: Distribution of redundant classifications (“RCs”) involving two semantic types

classification was removed, and a new assignment to another semantic type was added instead, only to create a new redundancy. An example of this involves the concept *Ferroplex* that was originally classified as both **Chemical Viewed Structurally** and **Organic Chemical** in UMLS 1998 (see Fig. 1). In the 2001 version, it is classified as **Chemical Viewed Structurally** and **Carbohydrate**. In both cases, we have redundant classifications to **Chemical Viewed Structurally**. Additionally, there were cases of multiple assignments causing one redundant classification, and only one of these assignments was deleted.

The above examples show that redundant classification has been a persistent problem in the UMLS. Utilizing our algorithm while preparing new releases of the UMLS will prevent the recurrence of such problems. To see the efficiency of our algorithm, note that in the current version of the SN:

$$\sum_{\mathbf{T} \in \text{SN}} d(\mathbf{T}) = 496.$$

This number is much smaller than the total number of pairs of semantic types in the SN which is $(134 \times 133)/2 = 8,911$. Without providing the details of the analysis, our algorithm must process $O(N \cdot h)$ pairs of semantic types, where N is the number of semantic types and h is the height of the SN. Since $h \approx \log_2 N$, the number of pairs is $O(N \log_2 N)$. In contrast, the naive algorithm must process $O(N^2)$ pairs of semantic types.

Table 1 summarizes all the redundant classifications in the 2001 version of the UMLS detected by our algorithm. The second column shows the total number of redundant classifications (including duplicate redundant classifications) to the respective ancestor semantic type in Column 1. The total number of redundant classifications after duplicates were omitted appears in Column 3. Column 4 lists the descendant semantic types, the classifications to which produced the redundant classifications. The number of concepts in the intersection of the descendant and the respective ancestor is written in parentheses alongside the descendant. Altogether, the algorithm discovered 40 binary intersections (of related semantic types) that were non-empty. A total of 12,657 redundant classifications were found, including 496 that were duplicates. This represents a 46.8% increase in the number of redundant classifications between the 1998 version of the UMLS and the 2001 version. A total of $12,657 - 496 = 12,161$ explicit classifications of concepts to semantic types should be removed from the UMLS.

CONCLUSION

The UMLS is a large and complex terminological repository containing about 790,000 concepts. During its yearly revisions, following the addition and removal of various concepts, the introduction of errors is nearly unavoidable. The evolution of biological knowledge

also contributes to this situation. Therefore, the NLM needs to perform various auditing tasks to ensure a high quality product. This quality assurance process can be facilitated with the use of automated tools.

The examples presented herein show that redundant classification has been a persistent problem in the UMLS. In such a situation, a concept is explicitly assigned simultaneously to a semantic type and one of its ancestors in the Semantic Network's IS-A hierarchy. Importantly, it is a kind of error that does *not* require the intervention of a domain expert to identify. In fact, by its definition, all cases can be detected algorithmically.

In this paper, we have presented an efficient algorithm that focuses on the error of redundant classification. With the use of our algorithm, such errors can be automatically uncovered and fixed in a cost-effective way. Thus, all redundant classifications can easily be expurgated from future versions of the UMLS before their release.

References

1. K. E. Campbell, D. E. Oliver, and E. H. Shortliffe. The Unified Medical Language System: Toward a collaborative approach for solving terminologic problems. *JAMIA*, 5(1):12–16, 1998.
2. B. L. Humphreys, D. A. B. Lindberg, H. M. Schoolman, and G. O. Barnett. The Unified Medical Language System: An informatics research collaboration. *JAMIA*, 5(1):1–11, 1998.
3. 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.
4. A. T. McCray and W. T. Hole. The scope and structure of the first version of the UMLS Semantic Network. In *Proc. Fourteenth Annual SCAMC*, pages 126–130, Los Alamitos, CA, November 1990.
5. A. T. McCray. Representing biomedical knowledge in the UMLS Semantic Network. In N. C. Broering, editor, *Proc. High-Performance Medical Libraries: Advances in Information Management for the Virtual Era*, pages 45–55, Westport, CT, 1993.
6. J. J. Cimino. Auditing the Unified Medical Language System with semantic methods. *JAMIA*, 5(1):41–51, 1998.
7. A. T. McCray and S. J. Nelson. The representation of meaning in the UMLS. *Methods of Information in Medicine*, 34:193–201, 1995.
8. H. Gu, Y. Perl, J. Geller, M. Halper, L. Liu, and J. J. Cimino. Representing the UMLS as an OODB: Modeling issues and advantages. *JAMIA*, 7(1):66–80, Jan/Feb 2000.
9. U. S. Dept. of Health and Human Services, National Institutes of Health, National Library of Medicine. *Unified Medical Language System*, 2001.