Adapting a Generic Match Algorithm to Align Ontologies of Human Anatomy

Peter Mork[1] and Philip A. Bernstein
Microsoft Research, Redmond, WA
pmork@cs.washington.edu, philbe@microsoft.com

Abstract

The difficulty inherent in schema matching has led to the development of several generic match algorithms. This paper describes how we adapted general approaches to the specific task of aligning two ontologies of human anatomy, the Foundational Model of Anatomy and the GALEN Common Reference Model. Our approach consists of three phases: lexical, structural and hierarchical, which leverage different aspects of the ontologies as they are represented in a generic meta-model. Lexical matching identifies concepts with similar names. Structural matching identifies concepts whose neighbors are similar. Finally, hierarchical matching identifies concepts with similar descendants. We conclude by reporting on the lessons we learned.

1. Introduction

Schema matching is central to many database problems, such as data integration, data warehouse loading, and XML message mapping. Its difficulty has motivated the development of many algorithms for identifying correspondences between schemata [1]. Most of these schema matching algorithms are intended to be generic, to apply across a variety of domains.

This paper reports on our experiences adapting generic match algorithms to two ontologies of human anatomy: the Foundational Model of Anatomy [2], or FMA, and the GALEN Common Reference Model [3], or CRM. Our goal was to identify corresponding elements between these models. These correspondences can then be used to help identify differences or merge the models [4].

Using schema matching algorithms "off-the-shelf" was not possible for two reasons. The modeling languages used by FMA and CRM are more expressive than ones used by most existing algorithms. And these algorithms expect relatively small schemas with at most hundreds of classes, not tens of thousands. We modified existing strategies to accommodate these issues of expressiveness and scale. Ultimately we found several thousand correspondences between these ontologies.

In the next section we provide background on our modeling framework and the input ontologies. In sections 3–5 we present our three phase match algorithm. Section 6 concludes by discussing lessons learned.

We also reported on this project in [5], which discusses where simple correspondences fail to capture the subtle differences between the ontologies.

2. Background

Central to the practice of medicine and biological research is the field of anatomy, which describes the structural relationships present in living organisms. Once the basic structures have been identified and defined, it becomes possible to describe function (physiology), disease (pathology), clinical intervention, etc.

The FMA [2] is being developed at the University of Washington under the guidance of Dr. Cornelius Rosse using the Protégé-2000 frame system [6]. The FMA attempts to encode all of human anatomy ranging from the macroscopic to macromolecular. The model is intended to support the development of knowledge-base applications and serves as a reference ontology in bioinformatics [2]. The copy of the FMA we used consists of ~59,000 concepts organized in four hierarchies. The model contains more than 100 types of relationships and ~1.6 million instantiated relationships.

The CRM was developed at the University of Manchester by Dr. Alan Rector, et al [3]. It is part of a larger project, GALEN, intended to facilitate knowledge reuse in clinical applications (e.g., an expert system). The CRM has ~24,000 concepts, automatically organized using a description logic frame system [7]. The core concepts are connected by ~913,000 relationships.

[1] Current affiliation: Computer Science & Engineering, University of Washington, Seattle, WA

These ontologies were authored using different modeling languages, but most generic match algorithms require the input models to be expressed in the same language or meta-model. We used an extension of Vanilla [4], which represents models graphically. Nodes in the graph correspond to concepts, and edges to relationships. Core Vanilla supports four types of relationships: is-a (i.e. generalization), contains (i.e. nesting), has (i.e. aggregation) and related-to (note that we use sans serif to indicate concept names or edge labels). These relationships are sufficient to express SQL or XML schemas, but they do not capture the distinction between template and own relationships in frame systems. We therefore extended Vanilla with type-of (to represent instances of a concept), and can-contain and can-have (to express templates). For example, an author can-have a first name. This paper's first author has the name Peter.

Representing the FMA and CRM in Vanilla was relatively straightforward, since as noted in [8], a frame system can be interpreted as a semantic network or edge-labeled graph. The only complication is that Vanilla's relationship types (i.e. edge labels) do not capture all of those used in the FMA and CRM, and it has no mechanism to extend that set of relationship types. To address this mismatch, we reified FMA- and CRM-relationships: Whenever concept *A* references concept *C* using label *B*, we create a new anonymous node *X*. Concept *A* contains node *X*, the type-of *X* is *B*, and *X* is related-to *C*. The only relationship we do not reify is is-a, since it has the same meaning in Vanilla as in the FMA and CRM. Reification allows us to encode the FMA and CRM in Vanilla, but introduces a new problem: we added many anonymous nodes, but lexical matching strategies can only find correspondences between named nodes.

3. Lexical Match

An obvious way to identify correspondences is to compare concept names. One might expect simple string matching to suffice, especially in anatomy; everybody uses the term heart to describe that organ. But even after removing CaMeL case (e.g., converting ValveInHeart to Valve In Heart), there are only 1834 string matches (ignoring case).

As an ongoing example, consider the concepts ValveInHeart from the CRM and Cardiac valve from the FMA. To match these concepts, we needed lexical tools. Starting with the basic terms, we employed three transformations: 1) Using the SPECIALIST lexicon [9] we normalized each term and 2) converted it to a set of word/usage pairs. 3) To address synonymy, we used the UMLS Metathesaurus [10] to convert words into concept identifiers. After these transformations, every term is a set of concept/usage pairs, which are used to calculate lexical similarity.

The first transformation uses the SPECIALIST lexicon to remove genitives, punctuation, capitalization, stop words (of, and, with, for, to, in, by, on, the), and inflection (plurals and verb conjugations). For example, Valve In Heart becomes valve heart. The lexicon can also indicate the part of speech for each word. This helps us distinguish between adjectives and nouns, information used by Cupid [11] to calculate similarity.

Normalization is a precondition to using the UMLS Metathesaurus. UMLS relates similar words or terms to unique concept identifiers (CUIs). For example, cardiac valve and heart valve are closely related in the Metathesaurus; cardiac valve is more general, since it is related to 3 CUIs as opposed to 2 for heart valve.

Each set of CUI/usage pairs is partitioned into a 'root' component and a 'support' component. The root contains nouns, verbs and terms not found in the Metathesaurus. The support contains everything else. We will denote the root of term *T* as T_R and its support as T_S. Given terms *F* (from the FMA) and *G* (from GALEN), the root similarity (S_R) and support similarity (S_S) between *F* and *G* are:

$$S_R(F,G) = \frac{2|F_R \cap G_R|}{|F_R|+|G_R|} \qquad S_S(F,G) = \frac{2|F_S \cap G_S|}{|F_S|+|G_S|}$$

These scores are combined to produce a combined similarity score (*S*): If both supports are empty, $S=S_R$. Otherwise, $S=S_R \times S_S$. Based on this calculation, the similarity between cardiac valve and heart valve is 0.8. Our choice of multiplication is somewhat arbitrary; it suffices that *S* is large (close to 1) only when S_R and S_S are both large.

Lexical matching works well to match concepts found in both models. But it does not identify many matches between relationship types found in both models (e.g., HasDivision and generic part), which are matched by detecting similarities in how they are used.

4. Structural Match

To exploit structure, we calculate a new value for the similarity of two nodes based on the old similarity score and the similarities of neighbors. Such neighborhood-based back-propagation of similarity is found in a number of generic match algorithms [1]. In the worst case, given models of size *M* and *N*, this strategy can produce $M \times N$ similarity values. If this process is iterated *I* times (as in similarity flooding [12]), the worst-case time complexity becomes $M \times I \times N$.

This approach is feasible for small schemas, but it does not scale to models with tens of thousands of concepts. Scalability is exacerbated by reification, since each instantiated relationship produces a new node. As a result, the FMA contains millions of nodes. We do not want to ignore these nodes. If one model asserts a given relationship between two concepts, we want to know if that relationship is asserted in the other model.

Since we could not compute the entire similarity matrix, we focused our efforts on matching reified nodes, for two reasons: First, these nodes are anonymous and hence cannot be matched using lexical techniques. Second, the similarity of reified nodes can be back-propagated to the relationship types, which were not properly matched in the previous phase.

Even though there are millions of reified nodes, we can compute their similarities overnight by ignoring any pair of reified nodes for which no similar neighbors were identified, since they surely do not match. (Recall that a reified node *X* is created when edge *B* connects node *A* to node *C*, so *X*'s neighbors are *A*, *B* and *C*.)

Let *X* (*A*à *B*à *C*) denote a reified relationship from the FMA and let *Y* (*D*à *E*à *F*) denote a reified relationship from the CRM. Given similarity scores for *AD*, *BE* and *CF*, the similarity score for *XY* is the average of these three values. In practice, lexical match produced many high *AD* and *CF* scores, but few non-zero *BE* scores.

To detect similarities in the usage of relationship types, we back-propagated similarity from *XY* to its neighbors. We iterate over the non-zero *XY* similarity scores and update the similarity score for *AD* to:

$$AD_{new} = AD_{old} + 0.2(1 - AD_{old})(XY)$$

The similarity scores for *BE* and *CF* are updated using the same formula, which has several interesting properties. The new similarity score cannot be greater than one, nor can it be lower than the previous similarity score. A single *XY* match has little effect, but five or more such matches begin to have a noticeable effect. (The constant 0.2 was chosen arbitrarily.)

Back-propagation improved the similarity between cardiac valve and heart valve from 0.8 to 0.92. The effect was more pronounced for relationship types. The similarity between is branch of and branch of increased from 0.286 to 0.98 ('is' is related to five CUIs and 'of' is a stop-word).

The branch example illustrates an interesting tradeoff. In retrospect, we could have added 'is' to SPECIALIST as a stop-word. This would help match a handful of relationship types (but may have introduced additional spurious matches). It is unlikely, though, that a single approach will work perfectly, so it is important to incorporate multiple strategies.

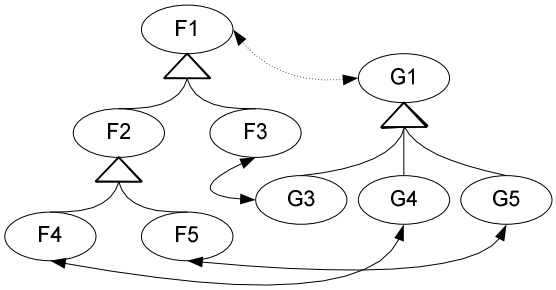

Figure 1: Example of intermediate concept

5. Hierarchical Match

It is tempting to use back-propagation in the inheritance hierarchy, as we did for structure matching. That is, if two concepts have similar specializations, the concepts are probably related. Some kind of inheritance-based matching is needed, since more-general concepts were not aligned using lexical match due to lack of agreement in the correct names for these abstractions. We will therefore leveraged consensus concerning the names of leaf concepts (like heart and lung) and iteratively derived new correspondences higher in the inheritance hierarchy.

Given two concepts, we estimate their similarity by comparing the similarities of their descendants. Before we present our algorithm three observations are relevant. First, since the FMA is much larger than the CRM, we would expect that many FMA-concepts will not match any CRM-concepts. Thus two concepts are similar if every child of the CRM-concept matches an FMA-concept, or more generally, if all children of the concept with fewer specializations have matches. Second, the models are too large to consider all descendants. Third, we must consider more than just children.

To see why children are not enough, consider the example in figure 1. The solid arrows indicate known correspondences (e.g., *F3–G3*); the triangular arrows indicate is-a relationships. We would like to conclude (as indicated with a dotted line) that *F1* and *G1* also match. But there is an intermediary generalization between *F1* and, for example, *F4*. So if we consider only direct children, the best match will be *F2–G1*.

We struck a balance by considering each concept's children and grandchildren (denoted C_{Ch}). Given *F* and *G* (from the FMA and GALEN, respectively), we iteratively evaluate their similarity:

$$S(F,G) = Max(1, \frac{\sum_{f \in F_{Ch}, g \in G_{Ch}} S(f,g)}{Min(|F_{Ch}|, |G_{Ch}|)})$$

In the previous example, *F1* now matches *G1*. The cross-similarity (numerator) of $F1_{Ch}$ and $G1_{Ch}$ is 3 (based on the leaf matches). *G1* has fewer children, so

$|G1_{Ch}|$ becomes the denominator. This ratio is exactly one, a perfect match.

As we reported in [5], this final match phase produced a disappointing number of results. We expected that anatomists would agree on key generalizations, but this was not the case. In retrospect, this is perhaps not surprising given the differences in context (structural vs. clinical). The impact of context was one of the lessons learned.

6. Lessons Learned

Initially, we had hoped to apply an off-the-shelf generic schema matching algorithm. In the end, we used many ideas from such algorithms, but we needed to do a lot of customization to get a satisfactory result. The lexical and structural steps were tailored to our problem and the hierarchical step is new, as far as we know. Although our matching problem is an extreme case in terms of size and complexity, we still suspect that this kind of customization is a necessary ingredient for future robust, generic schema matching systems.

We naïvely expected medical terminology would be more uniform than it turned out to be. Moreover, the authors responsible for constructing the FMA and CRM made very different choices concerning how to represent phenomena, which led to more differences than we expected. Some of these choices were induced by their choice of modeling language; Protégé-2000 and GRAIL differ slightly in their basic constructs. Other differences were caused by differences in philosophy and context. Examples appear in [5].

Reification was an important tool for addressing these differences. Because the relationship types and instances were first-class objects in Vanilla, we could identify more subtle correspondences or differences. For example, the lung appears in both models, but the information about the lung differs between the models. This was easy to see because the reified relationships relevant to the lung were not matched.

A more practical lesson is that size matters. Most generic match algorithms expect relatively small inputs. We quickly realized that the size of the FMA and CRM would be a problem, if we wanted the match algorithm to run in less than a day. So many of our design choices were based on time and space-complexity consideration. One might think the FMA and CRM are an extreme case. However, as systems become more complex (e.g., an integrated data network with hundreds of sources), the schemas involved can become quite large and complex. For example, large ERP applications have thousands of table definitions. Our work is a reminder that time- and space-complexity are important.

Much of our work was done using a relational database. In general, we were quite satisfied with this choice. For example, in many cases, we need to find all non-zero similarity scores between every possible pair of nodes. The fact that we can ignore similarity scores of zero meant we could, in most cases, leverage relational JOINs.

Our anatomist colleagues are satisfied with this first-cut match. Moreover, we are encouraged by the fact that the number of matches we found is similar to that of an independent study [13]. But its true quality can only be determined by a time-consuming element-at-a-time analysis of the correspondences by anatomists; work that we hope will be done at least for parts of the match result.

7. References

[1] E. Rahm and P. A. Bernstein, "A Survey of Approaches to Automatic Schema Matching," *The VDLB Journal*, vol. 10, pp. 334–350, 2001.

[2] C. Rosse and J. L. Mejino, "A Reference Ontology for Biomedical Informatics: the Foundational Model of Anatomy," *Journal of Biomedical Informatics*, pp. in press.

[3] A. L. Rector, A. Gangemi, E. Galeazzi, A. J. Glowinski, and A. Rossi-Mori, "The GALEN CORE Model Schemata for Anatomy: Towards a Re-usable Application-Independent Model of Medical Concepts," MIE 1994.

[4] R. A. Pottinger and P. A. Bernstein, "Merging Models Based on Given Correspondences," VLDB 2003.

[5] P. Mork, R. A. Pottinger, and P. A. Bernstein, "Challenges in Precisely Aligning Models of Human Anatomy Using Generic Schema Matching," Microsoft Research, Redmond, WA MSR-TR-2003-76 (http://research.microsoft.com/research/pubs/view.aspx?tr_id=689), 2003.

[6] M. Musen, M. Crubézy, R. Fergerson, N. F. Noy, S. Tu, and J. Vendetti, "Protégé-2000," http://protege.stanford.edu/, Stanford Medical Informatics.

[7] P. E. Zanstra, E. J. van der Haring, F. Flier, J. E. Rogers, and W. D. Solomon, "Using the GRAIL language for Classification Management," MIE 1997.

[8] S. Russell and P. Norvig, *Artifical Intelligence A Modern Approach*. Prentice Hall, 1995.

[9] "SPECIALIST Lexicon," http://www.nlm.nih.gov/pubs/factsheets/umlslex.html, National Library of Medicine (NLM).

[10] "Unified Medical Language System (UMLS)," http://www.nlm.nih.gov/research/umls/, National Library of Medicine (NLM).

[11] J. Madhavan, P. A. Bernstein, and E. Rahm, "Generic Schema Matching Using Cupid," VLDB 2001.

[12] S. Melnik, H. Garcia-Molina, and E. Rahm, "Similarity Flooding: A Versatile Graph Matching Algorithm," ICDE 2002.

[13] S. Zhang and O. Bodenreider, "Aligning Representations of Anatomy using Lexical and Structural Methods," AMIA 2003.