An Environment for Merging and Testing Large Ontologies

Deborah L. McGuinness
Knowledge Systems Laboratory
Computer Science Department
Stanford University
dlm@ksl.stanford.edu

Richard Fikes
Knowledge Systems Laboratory
Computer Science Department
Stanford University
fikes@ksl.stanford.edu

James Rice
CommerceOne
Mountain View, CA
rice@jrice.com

Steve Wilder
Knowledge Systems Laboratory
Computer Science Department
Stanford University
wilder@ksl.stanford.edu

Abstract

Large-scale ontologies are becoming an essential component of many applications including standard search (such as Yahoo and Lycos), e-commerce (such as Amazon and eBay), configuration (such as Dell and PC-Order), and government intelligence (such as DARPA's High Performance Knowledge Base (HPKB) program). The ontologies are becoming so large that it is not uncommon for distributed teams of people with broad ranges of training to be in charge of the ontology development, design, and maintenance. Standard ontologies (such as UNSPSC) are emerging as well which need to be integrated into large application ontologies, sometimes by people who do not have much training in knowledge representation. This process has generated needs for tools that support broad ranges of users in (1) merging of ontological terms from varied sources, (2) diagnosis of coverage and correctness of ontologies, and (3) maintaining ontologies over time. In this paper, we present a new merging and diagnostic ontology environment called Chimaera, which was developed to address these issues in the context of HPKB. We also report on some initial tests of its effectiveness in merging tasks.

1 INTRODUCTION

Ontologies are finding broader demand and acceptance in a wide array of applications. It is now commonplace to see major web sites include taxonomies of topics including thousands of terms organized into five-level or deeper organizations as a browsing and navigation aid. In addition to broader use of ontologies, we also observe larger and more diverse staff creating and maintaining the ontologies. Companies are now hiring "chief ontologists" along with "cybrarian" staffs for designing, developing, and maintaining these ontologies. Many times the team leader may have knowledge representation or library science training, however the staff may not have much or any formal training in knowledge representation. The broader demand for ontologies along with greater diversity of the ontology designers is creating demand for ontology tools.

The average ontology on the web is also changing. Early ontologies, many of which were used for simple browsing and navigation aids such as those in Yahoo and Lycos, were taxonomies of class names. The more sophisticated ontologies were large and multi-parented. More recently, mainstream web ontologies have been gaining more structure. Arguably driven by e-commerce demands, many class terms now also have properties associated with them. Early commerce applications, such as Virtual Vineyards, included a handful of relations, and now many of the consumer electronics shopping sites are including tens or hundreds of slot names, sometimes associated with value-type constraints as well. We now see more complicated ontologies even in applications that are only using ontologies to support smart search applications. Additionally, ontologies are being used more to support reasoning tasks in areas such as configuration and intelligence tasks. A decade ago, there were a modest number of ontology-supported configurators such as PROSE/QUESTAR [McGuinness and Wright, 1998; Wright et. al., 1993], however now, web-based configurators and configurator companies such as Trilogy, Concentra, Calico, etc. are common. There are even spin offs of configurator companies handling special areas of configuration such as PC-Order for PC configuration. Configuration ontologies not only have class, slot, and value-type information, but they typically have cardinality and disjointness information that supports reasoning with contradictions. Thus, we claim that ontologies are becoming more common, the designers come from more diverse backgrounds, and ontologies are becoming larger and more complicated in their representational and reasoning needs.

Simultaneously, there appears to be a stronger emphasis on generating very large and standardized ontologies. Areas such as medicine began this task many years ago with SNOMED [Spackman, et. al., 1997] and UMLS

[McCray and Nelson, 1995]. Recently broader and shallower efforts have emerged like the joint United Nations/Dunn and Bradstreet effort to create an open coding system for classifying goods and services [UNSPSC, 1999]. Another new distributed broad ontology is the DMOZ Open Directory effort [DMOZ, 1999] with over 200,000 categories and over 21,000 registered knowledge editors. The goal of standard ontologies is to provide a highly reusable, extensible, and long-lived structure. Large ontologies in concert with the challenges of multiple ontologies, diverse staffing, and complicated representations strengthens the need for tools.

In this paper, we address two main areas. The first is merging different ontologies that may have been written by different authors for different purposes, with different assumptions, and using different vocabularies. The second is in testing and diagnosing individual or multiple ontologies. In the rest of this paper, we will give two project descriptions that served as motivation for our work on merging and diagnostic tools. We will then describe an ontology environment tool that is aimed at supporting merging and testing ontologies. We will describe the tool's used in our work on DARPA's HPKB program [Cohen, et. al., 1998]. We will also describe an evaluation of the merging capabilities of the tool. Finally, we will present the diagnostic capabilities and discuss future plans.

2 TWO MOTIVATING PROBLEMS

In the last year, some of the authors were involved in each of two major ontology generation and maintenance efforts. We gained insight from the tasks that was used to help shape our resulting ontology tool efforts. Subsequently, we have used [McGuinness, 1999] as well as licensed the tools on other academic and commercial ontology projects. We will describe the tasks briefly and present an abstraction of the problem characteristics and needs with relation to ontology tools.

2.1 MERGING THE HIGH PERFORMANCE KNOWLEDGE BASE ONTOLOGIES

The first problem was in the HPKB program. This program aimed to generate large knowledge bases quickly that would support intelligence experts in making strategic decisions. The KBs had a reasonably broad subject area including terrorist groups, general world knowledge (such as that contained in the CIA World Fact Book [Frank, et. al., 1998]), national interests, events (and their results) in the Middle East, etc. The types of questions that an analyst might ask of a KB may be simple, including straight "look up" questions like finding the leader of an organization or the population of a country. Other questions may be quite complex, including asking about the possible reaction of a terrorist group to a particular action taken by a country. Knowledge bases in this program tended to have a high degree of structure, including many slots associated with classes, value-type constraints on most slots, values on many slots, minimum cardinality constraints on slots, disjoint class information, etc. The knowledge bases were typically designed by people trained in knowledge representation and usually populated by those literate but not expert in artificial intelligence.

In the first year of the program, many individual knowledge bases were created in order to answer particular "challenge problem" questions. These questions were designed to be typical of those that a government analyst would ask. Two competitive research and integration teams were evaluated on the quality of the answers that their knowledge bases and associated reasoners returned. Many of the challenge problem questions in the first year were answered in particular contexts, i.e., with only a subset of the knowledge bases loaded. In the second year of the program, some teams, including ours, needed to be prepared to answer questions in any portion of the domain. We needed to load all of the knowledge bases simultaneously and potentially reason across all of them. Thus, we needed to load a significant number of KBs (approximately 70) that were not originally intended to be loaded together and were written by many different authors. Our initial loading and diagnosis step was largely manual with a number of ad hoc scripts. This was a result of time pressure in concert with the expectation that this was a one-time task. Some of the findings from the merging and diagnosis task were as follows:

- Large ontology development projects may require extensive systematic support for pervasive tests and changes. Our final ontology contained approximately 100,000 statements (and the version of the ontology after forward chaining rules had been run contained almost a million statements). Even though the individual ontologies all shared a common "upper ontology", there was still extensive renaming that needed to be done to allow all the ontologies to be loaded simultaneously and to be connected together properly. There were also pervasive tests to be run such as checks for comment and source field existence as well as argument order on functions. We discovered, for example, that different authors were using relational arguments in differing orders and thus type constraints were being violated across ontologies. Additionally, if a relation's domain and range constraints were used to conclude additional class membership assertions for arguments of the relation, then those arguments could end up with multiple class memberships that were incorrect. For

example, if relation Leader has a domain of Person and a range of Country, one author states "(Leader Clinton US)", and another states "(Leader US Clinton)", then Clinton could be inferred to be a person AND a country.[1]

- Repairing and merging large ontologies require a tool that focuses the attention of the editor in particular portions of the ontology that are semantically interconnected and in need of repair or further merging. There were many places where taxonomic relationships were missing when multiple ontologies were loaded together. For example, a class denoting nuclear weapons was related to the "weapon" class but not to the "weapon of mass destruction" class, nor to the disjoint partition of classes under "weapon". A tool that showed (just) the relevant portions of the taxonomies and facilitated taxonomy and partition modifications later turned out to be extremely valuable for editing purposes.

- Ontologies may require small, yet pervasive changes in order to allow them to be reused for slightly different purposes. In our HPKB task, we found a number of slots that needed to be added to classes in order to make the classes useful for additional tasks. We found many slots in the same ontology that appeared to be identical yet were unrelated. (We hypothesize that one major cause of this problem was that a term inherited a slot and value-type constraint from a parent class, but the author did not know to look for the slot under its given name, thus the author added a slot to capture the same notion under another name.) Also, we found a large number of slots that were inverses of other slots but were not related by an explicit slot inverse statement. Without the inverse information, the inverse slots were not being populated and thus were not useful for question answering even though the information appeared to be in the knowledge base. Our goal was to support users in finding the connections that needed to be made to make ontologies more useful.

- Ontologies may benefit from partition definitions and extensions. We found many ontologies that contained some disjoint partition information (e.g., "people" are disjoint from "bodies of water"), but in many cases the partition information was under specified. In the previous example with incorrect argument order, if we had information stating that people were disjoint from countries, then the inconsistency could have been detected earlier, most likely at knowledge entry time.

[1] This inference is consistent if there is no information that states that country and person are disjoint.

2.2 CREATING CLASS TAXONOMIES FROM EXISTING WEB ONTOLOGIES

In a noticeably different effort, we used a Web crawler to mine a number of web taxonomies, including Yahoo! Shopping, Lycos, Topica, Amazon, and UN/SPSC, in order to mine their taxonomy information and to build corresponding CLASSIC [Borgida et. al., 1989; Brachman, et. al., 1999] and OKBC (Open Knowledge Base Connectivity) [Chaudhri, et. al, 1998] ontologies. Our goals for this work were (1) to "generate" a number of naturally occurring taxonomies for testing that had some commercial purpose, and (2) to build a larger cohesive ontology from the "best" portions of other ontologies. ("Best" was initially determined by a marketing organization as portions of ontologies that had more usage and visibility.)

Our ontology mining, merging, and diagnosis effort had little emphasis on reasoning, but instead was centered on choosing consistent class names and generating a reasonable and extensible structure that could be used for all of the ontologies. The expected use of the output ontology was for web site organization, browsing support, and search (in a manner similar to that used in FindUR [McGuinness, 1998]).

We found that extensive renaming was required in these ontologies mined from the Web. For example, we found the unique names assumption was systematically violated within individual ontologies so that class names needed their own contexts in order to be useful. Thus, systematic treatment was required to put individual ontology branches into their own name space and to separate terms like steamers under clothing appliances from steamers under kitchen appliances. We also found extensive need for ontological reorganization. Thus, we still required focusing an editor's attention on pieces of the ontology. Additionally, we found need for more diagnostic checks with respect to ontological organization. For example, there were multiple occurrences of cycles within class graphs. So, we introduced checks for cycles into our diagnostics.

There was also no partition information in these ontologies, but there were multiple places where it appeared beneficial to add it. Our initial automatically generated ontologies were obtained from web sites that lacked explicit slot information, thus all of our slot information was systematically generated (and thus less likely to need the same kinds of modifications as those we found from human-generated slot information). Subsequent inspections of other web ontologies containing slot information, however, revealed many of the same issues we observed in our HPKB analysis work.

These two experiences, along with a few decades of staff experience with building knowledge representation and

reasoning systems and applications, led us to design and implement an ontology merging and diagnosis tool that we will describe next.

2.3 Needs Analysis

The two previous efforts motivate the following needs in a merging and diagnostic tool:

- Name searching support (across multiple ontologies)
- Support for changing names in a systematic manner
- Support for merging multiple terms into a single term
- Focus of attention support for term merging based on term names
- Focus of attention support for term merging based on the semantics of term descriptions
- Browsing support for class and slot taxonomies
- Support for modifying subsumption relationships in class and slot taxonomies
- Partition modification support
- Support for recognizing logical inconsistencies introduced by merges and edits.
- Diagnostic support for verifying, validating, and critiquing ontologies

3 AN ONTOLOGY MERGING AND DIAGNOSIS TOOL

Chimaera is a new ontology merging and diagnosis tool developed by the Stanford University Knowledge Systems Laboratory (KSL). Its initial design goal was to provide substantial assistance with the task of merging KBs produced by multiple authors in multiple settings. It later took on another goal of supporting testing and diagnosing ontologies as well. Finally, inherent in the goals of supporting merging and diagnosis are requirements for ontology browsing and editing. We will define the tasks of ontology merging and diagnosis as used in our work, and then we will introduce the tool.

We consider the task of merging two ontologies to be one of combining two or more ontologies that may use different vocabularies and may have overlapping content. The major two tasks are to (1) to coalesce two semantically identical terms from different ontologies so that they are referred to by the same name in the resulting ontology, and (2) to identify terms that should be related by subsumption, disjointness, or instance relationships and provide support for introducing those relationships. There are many auxiliary tasks inherent in these tasks, such as identifying the locations for editing, performing the edits, identifying when two terms could be identical if they had small modifications such as a further specialization on a value-type constraint, etc. We will focus on the two main tasks for our discussion.

The general task of merging can be arbitrarily difficult, requiring extensive (human) author negotiation. However, we claim that merging tools can significantly reduce both labor costs and error rates. We support that claim with the results from some initial tool evaluation tests.

We addressed the task of diagnosing single or multiple ontologies by producing a test suite that evaluates (partial) correctness and completeness of the ontologies. The major tasks involve finding and reporting provable inconsistencies, possible inconsistencies, and areas of incomplete coverage. As with merging, diagnosis can be arbitrarily complex, requiring extensive human analysis to identify all problems and present them in an order appropriate to the problem importance. Tools built to provide the first level of analysis, however, can greatly reduce human labor cost as well as improve the consistency of the analysis. In our diagnostic test suite, we do not attempt to find all problems; we just choose a subset that is computationally viable and motivated by usefulness of the reports.

3.1 CHIMAERA

Chimæra is a browser-based editing, merging, and diagnosis tool. Its design and implementation is based on our experience developing other UIs for knowledge applications such as the Ontolingua ontology development environment [Farquhar, et al, 1997], the Stanford CML editor [Iwasaki, et al, 1997], the Stanford JAVA Ontology Tool (JOT), the Intraspect knowledge server [Intraspect 1999], a few web interfaces for CLASSIC [McGuinness, et. al., 1995; Welty, 1996], and a collaborative environment for building ontologies for FindUR [McGuinness, 1998]. Chimaera has a web-based UI that is optimized for Netscape and MSIE browsers. It is written in HTML, augmented with Javascript where necessary to support niceties like spring-loaded menus and drag and drop editing.

Our goal was to make it a standards-based generic editing, merging, and diagnosis tool. When Ontolingua's editor was first developed, there was no standard API for knowledge-based systems. Since then, the OKBC API has been developed by KSL and SRI International's AI Lab. OKBC allows us to develop tools that can merge KBs in any OKBC-compliant representation system either on the same machine or over the network. Chimæra was designed from the ground up to be a pure OKBC application. Our typical editing environment is Ontolingua, but this is not a requirement. For example,

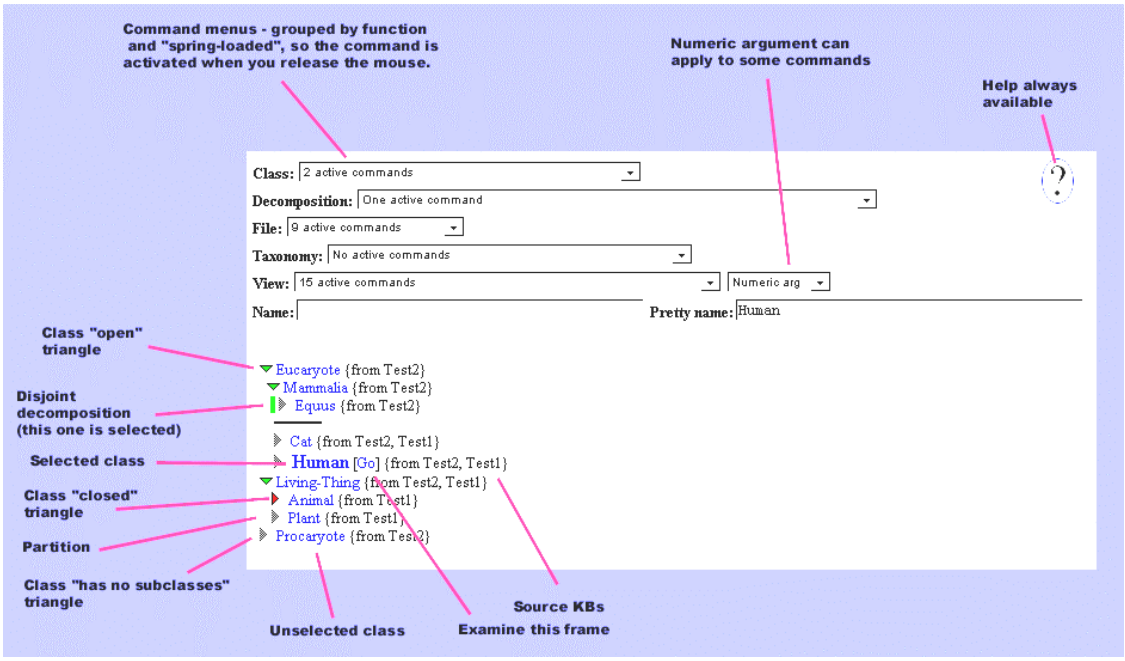

Figure 1: A view of Chimæra's user interface

one could edit in Ontosaurus {Swartout, et. al, 1996] or OntoWeb [Domingue, 1998] to produce the ontology. If the ontology editor produces OKBC-compliant files, then they can be loaded directly into Chimaera. Otherwise, indented list format, tuple format, or a few other standard forms may be used. In general, OKBC wrappers can be developed for a wide range of knowledge representation systems. For example, in one of our e-commerce ontology projects, we generated CLASSIC ontologies and developed an OKBC wrapper for CLASSIC that was used to load OKBC-compliant input into Chimaera. Translation systems such as OntoMorph [Chalupsky, 2000] could also be used to support multiple languages.

The UI for the current version of Chimæra is shown in Figure 1. The interface consists of a set of commands on spring-loaded menus (the command activates as soon as the menu selection is made). Like most GUIs, the user selects operands by clicking on them, and selection is shown by the selected operands being displayed in boldface. Applicable commands are then available on the menus, and inapplicable commands are also displayed showing the reason why they are inapplicable. The UI contains amongst its seventy odd commands a rather full-featured taxonomy and slot editor as well as commands more obviously associated with the ontology merging task, e.g., the "Merge Classes" command. It also contains 17 diagnostic commands along with options for their modification. The current UI is not a general-purpose editing environment for ontologies. It only addresses classes and slots; non-slot individuals and facets are not displayed. Similarly, there is no support for the editing of axioms. We plan to extend the functionality of the tool in later versions to include all object types. In contrast to two other merging efforts [Fridman Noy and Musen, 1999; and Chalupsky, et. al., 1997], our environment also supports creating and editing disjoint partition information and includes an extensive repertoire of diagnostic commands.

The restricted nature of the UI allows us to present a view of the KB to the user that is not cluttered by extraneous commands, widgets, or KB content. This is very important to the design of the UI, since focus of attention is vital in the KB merging task. The user may never be able to make merging decisions if the classes to be considered are many screens apart. There are, therefore (currently) no fewer than 29 different commands in the View menu that affect the way the KB is displayed to the user, and the system uses sophisticated techniques to automatically select default settings for those commands that are appropriate in most cases.

Chimaera currently addresses only a portion of the overall ontology merging and diagnosis tasks. Even though it may be viewed as an early design in terms of a complete merging and diagnostic tool, we have found significant value in it to date. We now describe some experiments designed to evaluate its usefulness in merging.

The experiments we have run only make use of those features in Chimaera designed to support the merging of class-subclass taxonomies. Chimaera includes support for

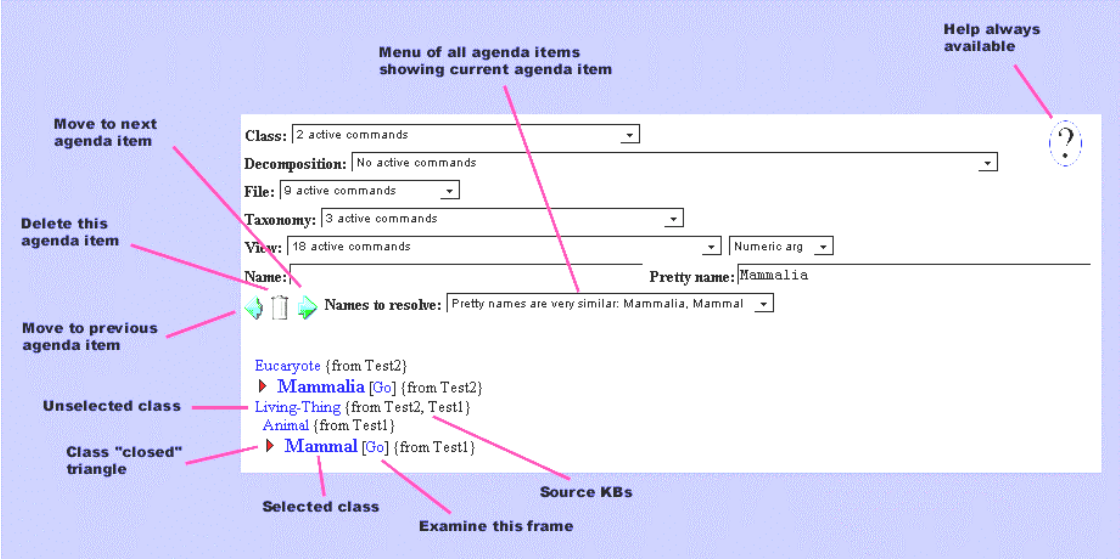

Figure 2: Chimæra in name resolution mode suggesting a merge of Mammal and Mammalia

merging slots and in the future, will support merging of facets, relations, functions, individuals, and arbitrary axioms. Similarly, the diagnosis functions only include domain independent tests that showed value in our experiments to date. These tests allow limited user input for modifications to the tests. In our future environment, we expect to include a diagnostic testing language that allows users to dynamically add new tests to the test suite, and thus support more domain-dependent diagnostics as well

3.2 MERGING AND EVALUATION

Chimaera generates name resolution lists that help the user in the merging task by suggesting terms each of which is from a different ontology that are candidates to be merged or to have taxonomic relationships not yet included in the merged ontology. For example, figure 2 shows a suggestion for merging Mammalia from ontology "Test2" with Mammal from ontology "Test1" based on the similarity of the names. The suggested candidates may be names of classes or slots. The module that puts candidates on the list is controlled by a user-settable "vigor" metric that activates a progressively more extensive search for candidate sets of terms. It considers term names, presentation names (called "pretty names" in Ontolingua), term definitions, possible acronym and expanded forms, names that appear as suffixes of other names, etc.

Chimaera also generates a taxonomy resolution list where it suggests taxonomy areas that are candidates for reorganization. It uses a number of heuristic strategies for finding such edit points for taxonomies. One looks for classes that have direct subclasses from more than one ontology (since such subclasses are likely to need some reorganization have additional intended relationships among them that are not yet in the merged ontology).

We ran four experiments aimed at evaluating Chimaera's merging effectiveness. They were focused on (1) coalescing ontology names, (2) performing taxonomic edits, (3) identifying ontology edit points, and (4) testing overall effectiveness in a substantial merging task. Because of space constraints here, we describe our high level findings and only describe one of the experiments in detail.

A long version of the merging experiment write-up is available from http://www.ksl.stanford.edu/yearindex.html.

4 EXPERIMENTAL FINDINGS

We conducted a set of experiments scoped to be within our resource budget that were designed to produce a measure of the performance of Chimæra. We also compared those results to the performance of other tools that a KB developer might reasonably use to do such a merge, absent the KB merging tool itself. At each stage in the experiment, our goal was to control for as many factors as possible and to assure that the experimental settings correspond closely to the settings in which the tool would actually be used.

KB merging is a non-trivial cognitive task, and our tools are also non-trivial, so it is not at all surprising that it should be difficult to design experiments to measure the utility of such tools. The overriding principle we used was that whenever a judgement call had to be made in the

experiment design, we tried to make sure that any bias introduced in that judgement worked *against* showing Chimæra in a good light.

We began by conducting a set of three calibration experiments in which we determined the number of steps and time required to do specific types of operations that would be performed while doing a merging task using Chimæra, a representative KB editing tool (Ontolingua), and a representative text editing tool (Emacs). These studies were designed to provide quantitative "rate" comparisons in that they indicated which steps in the merging task Chimæra speeds up and by how much, and to provide qualitative indications of the steps for which Chimæra provides substantial improvements in reliability. Using the results of these calibration experiments, we then performed a larger merge task using only Chimæra. The calibration experiments were then used to estimate the comparative utility of Chimæra over this larger task.

The primary results of these experiments are the following:

- Merging two or more substantial ontologies was essentially not doable in a time effective manner using a text-editing tool, primarily because of the difficulty of examining the taxonomy of any non-trivial ontology using only a text editor.
- Chimaera is approximately 3.46 times faster than an ontology editing tool (Ontolingua) for merging substantial taxonomies. Moreover, for the portion of the taxonomy merging task for which Chimaera's name resolution heuristics apply, Chimaera is approximately 14 times faster than an ontology editing tool (Ontolingua).
- Almost all of the operations performed during a taxonomy merge would be more error-prone if they were performed using an ontology editing tool (Ontolingua), and the critical "merge class" operations would be extremely error-prone if performed using a KB editing tool.

The ontology merging task is only an interesting problem when one tries to merge large ontologies. Chimaera has proved to provide considerable utility in non-trivial merging tasks. The other tool options tried were so poor at this task that it became impractical to perform a head-to-head experiment against other tools because the other tools simply were not able to merge reasonably large ontologies in a practical amount of time. We conclude, therefore, that Chimæra, even though it addresses only a portion of the overall merging task, makes a significant qualitative difference in one's ability to build large ontologies using fragments derived from a number of sources.

4.1 EXPERIMENT 3: FINDING EDIT POINTS

The time taken to execute an editing operation is only a minor part of the ontology merging process; a major task for the user is finding the places in the input ontologies that are to be edited. Our third experiment, which focused on that task, may be the most instructive and important; thus, we describe it in detail here.

In this experiment, we attempted to determine the relative performance of Emacs, the Ontolingua editor, and Chimæra in the edit-point finding activity. When we attempted to build scripts for these activities using Emacs, it became apparent that the task of finding good edit points is so difficult in Emacs that one simply could never realistically use Emacs for such a task. The core problem is that most of these activities involve the user being able to see the ontology's taxonomy in order to make rational decisions. It is so difficult to examine the taxonomy of any non-trivial ontology using Emacs that the user would be forced, in effect, to reinvent some sort of representation system using either shell scripts or keyboard macros in order to have a chance of knowing what to do. We decided that this was sufficiently unrealistic that we eliminated Emacs from Experiment 3.

The idea behind Experiment 3, therefore, was to try to measure the time taken by a user to find candidate edit points using the Ontolingua editor and Chimæra. Our goal in designing the experiment was to factor out the time actually taken to perform the suggested edits, since that editing time was considered in Experiment 2.

Our goal was to measure the performance of users performing the edit-point-finding task in as unbiased manner as possible. In the best of all possible worlds, we would have a large pool of input ontologies and of test subjects with the necessary skills so that we could get an overall idea of the performance of the tools. This was not practical, so we selected a pair of test subjects who were as closely matched as we could find in knowledge representation skill as well as skill in the use of the tool. We used one subject with the Ontolingua editor who had considerable experience using the tool as a browser, though little as an editor. The test subject who was to use Chimæra had a small amount of experience using Chimæra as a browser, but no experience using any of the editing features. In accordance with our overall strategy of bias reduction, the bias in the test subject selection clearly favored the Ontolingua editor over Chimæra.

The test subject using Chimæra was given a guided tour of its editing operations. Both users had about two hours to practice using their respective tools specifically on the ontology merging task. For the practice session, they were each provided with some small sample ontologies that had no overlap with the ontology content of the test ontologies.

We instrumented Chimæra so that we could identify the commands being invoked, the times at which the commands were executed, and the number of arguments used by the commands. The goal was for the test subject to use Chimæra not only to find the edit points, but also to perform the edits so that we could learn as much as possible from the process. Having performed the timed merge, we would then subtract out the time taken to perform the edits to make the results more comparable to the use of the Ontolingua editor.

For the experiment with the Ontolingua editor, we timed the test subject with a stopwatch. When the test subject identified an edit point, the clock was stopped, and the test subject turned away from the screen. The desired edits were then performed, and the clock restarted. It is essential to perform the edits suggested because performing the proposed edits changes the structure of the ontology and the way that it appears on the display. This often results in terms that were previously distant on the display appearing close together, thereby making other candidate edit operations more obvious. Ironically, as we saw in our previous calibration experiments, Chimæra is so much more efficient at performing these edits than the Ontolingua editor that when the experiment referee stepped in to perform the requested edits during the experiment, he used Chimæra to perform them.

We decided that before conducting the experiment it would be a good idea to calibrate the experiment by getting an idea of the upper bound of the possible performance using Chimæra. We therefore had one of the developers of Chimæra - and experienced user - use Chimæra to perform the merge of the two proposed input ontologies a number of times. This was to give us an idea both of the number of edit operations that could reasonably be found in the ontologies, and the maximum speed with which a user could perform the merge if all of the thinking time necessary to decide what to merge was reduced as close to zero as possible. We anticipated that were we to graph the edit operations against time we would see a clear knee in the curve at the point at which the "low-hanging fruit" had all been plucked. Given this point, we intended to run the real experiment for a time not exceeding the time for the knee in the curve. This would, we thought, give us some confidence that we were likely to be in the low-hanging fruit operating region of both tools, since we anticipated that Chimæra would be faster than the Ontolingua editor. If we were to stop the experiment after an arbitrary time without performing this calibration, we might bias the result in favor of the slower tool if the faster tool had been fast enough to get outside its low-hanging fruit region in the time allotted, but the slower tool had not.

Figure 3 shows the results from this calibration experiment. The anticipated knee in the curve did not actually appear, though there is a knee at 622 seconds, where the user finished his first pass through the Name Resolution agenda. After about an hour, the user stopped, reporting that all of the edits that he was performing seemed by that point to be concentrated in cleaning up one of the input ontologies, rather than actually merging the ontologies. Given the results of this calibration run, we decided to give the two test subjects 55 minutes in which to perform their edit-point finding.

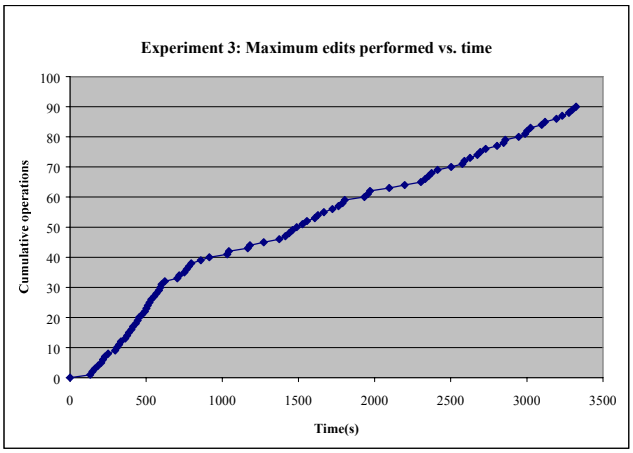

Figure 3: The cumulative number of edit points found and edits performed using Chimæra plotted against time in seconds. The dense set of points early on was characterized by a large number of merge operations driven from the Name-Resolution agenda. Subsequent edit points were found using the Taxonomy-Traversal agenda and other Chimæra browsing features.

We wanted to try to control as much as possible for the different representational decisions that two test subjects might make, so during the experiment, the developer who performed the calibration experiment was on-hand to answer any representational questions whenever such questions arose. This oracle was not allowed to volunteer any edit suggestions, but would, if prompted, say whether any pair of classes should be merged or have an additional relationship asserted.

Our goal was to define the idea of "finding an edit point" to mean as closely as possible the same thing for the two tools. There were, however, some differences because of the nature of the tools. For Chimæra, the time taken to find an edit point was the time taken to identify the place to edit and to select the arguments for the editing operation. These times were captured by Chimæra's instrumentation. For the Ontolingua editor, we measured the time taken between the clock being started and the user calling out for the clock to be stopped upon finding a candidate edit and calling out the arguments to be used in the edit. We did this because we knew that we would not

be using the Ontolingua editor actually to perform the edits.

We had to decide whether to allow edits within input ontologies as well as across ontology boundaries during the experiment. We anticipate that the ontology merging process may typically involve some clean up to the input ontologies, but we wanted to focus our evaluation on the merging process rather than on intra-ontology editing, so we decided to disallow intra-ontology edits for this experiment. Cycorp provided to us for use in this experiment the part of their IKB relating to Agents that they had built for the HPKB program. With this input ontology, which we had not seen before, we could be confident that we had received a clean and well-documented ontology, and that there would be a reasonable amount of overlap with our own Agents representation similarly developed for the HPKB program. Restricting scoring of candidate edit points to include only proposed edits that had at least one argument from each ontology was easy in Chimæra, since they are color-coded. In the case of the Ontolingua editor, we renamed all of the terms in one ontology to have a common suffix indicating the ontology of origin. We provided this substantial help to the subject using Ontolingua in order to improve the comparability of the experimental results. Interestingly, we used a command in Chimæra to do this systematic renaming.

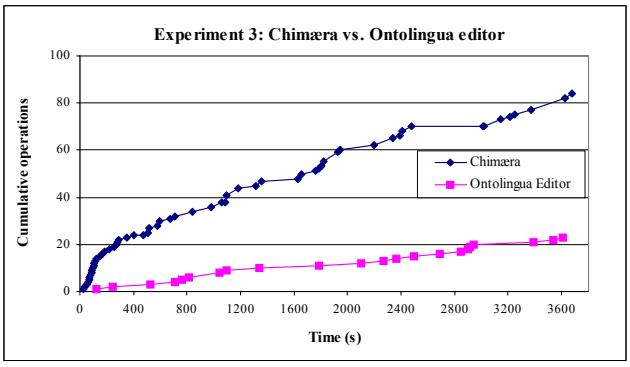

Figure 4: Chimæra proved to be superior to the Ontolingua editor at finding candidate edit points

Figure 4 shows the results from Experiment 3. There are a number of ways to interpret at these results depending on what we hope to learn. The fact that the curve for Chimæra is always well above the curve for the Ontolingua editor clearly shows the significant superiority of Chimæra for this task. Overall, the Chimæra test subject was able to identify 3.7 times as many edit points, and was also able to perform the edits.

It is difficult to come up with a simple numeric rate for the number of edit points found per minute because we need to have reason to believe that there is an underlying linear model before such a rate number has any meaning or predictive value. Luckily, we do have reason to believe that there is a linear model underlying at least part of Experiment 3 (there may be linear models underlying other parts, but we do not know this with certainty). The Name Resolution menu in Chimæra presents the user with a simple list of candidate edit points. The user, in general, iterates through this list until all elements of the list have been processed. This is what our user was doing during the first 185 seconds of the experiment. In this linear region, merges were being found, considered, and accepted or rejected at the rate of about one every nine seconds. The correlation to a linear fit in this region is R^2=0.94, supporting our reasoning that the underlying model is linear. This result is consistent with our experience on other ontologies. We expect that this linear model should be broadly independent of ontology size, since the time taken to construct the agenda itself is factored out. The algorithm that constructs the name resolution menu is $O(n^2)$. Within the known linear region, Chimæra outperformed the Ontolingua editor by a factor of fourteen.

The test subject who was using the Ontolingua editor also, we believe, exhibited a linear strategy. This happened because during the practice session the test subject developed a systematic method for finding candidate edit points that involved a systematic traversal of the ontology. The strategy involved looking in turn at each class and then considering all of its siblings and the siblings of each of its superclasses up to the roots to see whether a merge or taxonomic edit was appropriate. The number of examination steps necessary for any given class is a function of the depth of the taxonomy and branching factor of the superclasses. The time taken to perform any given iteration in this strategy is therefore reasonably constant, though influenced by the number of subclasses that must be inspected once a candidate edit has been selected. We believe that the complexity of this strategy is bounded by $O(n.\log_b(n))$ and $O(n^2)$, where b is the average subclass branching factor and n is the size of the ontology. For an ontology such as the one we used as input, however, the model should be roughly linear. The results for a linear fit throughout the experiment for the Ontolingua editor give us a rate of 157 seconds per edit point found, with a correlation of 0.98.

5 DIAGNOSTICS TESTS

Chimaera also has a set of diagnostics that can be run selectively or in their entirety. Routinely when we obtain knowledge bases now, we run the diagnostics and invariably find issues with our incoming KBs. The current list of diagnostics was derived as a retrospective analysis of the most useful domain independent tests that we needed to run on the HPKB and on the crawled web ontologies. They group into four areas:

1) Simple checks for incompleteness (missing argument names, missing documentation strings, missing sources, missing type constraints, missing term definitions);

2) Syntactic analysis (incidence of words (or sub-strings), possible acronym expansion);

3) Taxonomic analysis (redundant super classes, redundant types, trivial instances or subclasses of THING, definition extensions from included ontologies), and

4) Semantic evaluation (slot value/type mismatch, class definition cycle, domain/range mismatch).

This is obviously not everything that could be checked. The current diagnostic suite does not connect to the full theorem prover so there is only limited consistency checking. The current testing environment also does not give users the power to add their own, potentially domain-specific, checks. Even with the limited power of the diagnostics set though, we successfully used it to provide initial correctness and completeness checks of all incoming HPKB knowledge bases for our final team evaluation. Possibly more importantly, its output was usable by people with little training in knowledge representation, and we found that with no training they could make effective and correct improvements to the knowledge bases guided by the diagnostic output. Also, the tool takes multiple input formats, thus we were able to allow people to use it who had no familiarity with OKBC or Ontolingua. We had some SNARK and KIF-literate users load in their ontologies in the input format they were familiar with, run diagnostics, and debug their knowledge bases with little intervention from us. We also used this toolset to check for problems in our semi-automatically generated ontologies from web crawls. The tests found a surprising number of things that would have been tedious or difficult for us to find ourselves, such as class cycles and inconsistency in naming in Amazon's ontology. Finally, we used the merging tool with ontologies generated by naïve users with no training, and they were able to immediately merge independent ontologies and use the tool effectively to focus their attention on the problem areas in the ontologies. They also used the diagnostics effectively with no training.

6 CONCLUSION

We have presented an ontology editing, merging, and diagnostic environment developed to meet the emerging needs of representation and reasoning tasks on the Web and of ontology creation and maintenance tasks. We have briefly overviewed the merging and diagnostics components and presented some evaluation results on the merging side and some anecdotal reports on the diagnostics side. While our tool is in its early stages, these evaluations of the tool, our own personal use of the tool, and demand for the tool from both the commercial and academic sectors provide notable evidence that it makes significant improvements in productivity and quality of ontology development and maintenance. We are continuing to develop the tool, focusing in particular on extending its reasoning capabilities, semantic analysis, its extensibility, and usability by non-experts.

Acknowledgements

The authors are indebted to DARPA for their support of this research under contract N66001-97-C-8554, *Large-Scale Repositories of Highly Expressive Knowledge*. We would also like to thank Cycorp for kindly providing knowledge base fragments for some of the merging experiments and Bob Schrag at IET for his support in the design and conducting of the experiments.

Bibliography

Alex Borgida, Ronald J. Brachman, Deborah L. McGuinness, and Lori Alperin Resnick; *CLASSIC: A Structural Data Model for Objects*, Proceedings of the 1989 ACM SIGMOD International Conference on Management of Data, Portland, Oregon, June, 1989, pp. 59--67.

Ronald J. Brachman, Alex Borgida, Deborah L. McGuinness, and Peter F. Patel-Schneider; *"Reducing" CLASSIC to Practice: Knowledge Representation Theory Meets Reality*; In the Artificial Intelligence Journal, 114(1-2) pages 203-237, October, 1999.

Hans Chalupsky, Eduard Hovy, Tom Russ; *Progress on an Automatic Ontology Alignment Methodology*; Proceedings of the ANSI ad hoc group on ontology standards, 1997. http://ksl-web.stanford.edu/onto-std/ .

Hans Chalupsky. OntoMorph: A Translation System for Symbolic Knowledge, in A. G. Cohn, F. Giunchiglia, and B. Selman, editors, Principles of Knowledge Representation and Reasoning: Proceedings of the Seventh International Conference (KR2000), Morgan Kaufmann Publishers, San Francisco, CA., 2000.

Vinay Chaudhri, Adam Farquhar, Richard Fikes, Peter Karp, and James Rice; *OKBC: A Programmatic Foundation for Knowledge Base Interoperability*; Proceedings of the Fifteenth National Conference on Artificial Intelligence (AAAI-98); AAAI Press. (http://www.ksl.stanford.edu/KSL_Abstracts/KSL-98-08.html)

Paul R. Cohen, Robert Schrag, Eric Jones, Adam Pease, Albert Lin, Barbara Starr, David Easter, David Gunning, and Murray Burke. The DARPA High Performance

Knowledge Bases Project. In Artificial Intelligence Magazine. Vol. 19, No. 4, pp.25-49, 1998.

DMOZ – Open Directory Project. http://www.dmoz.org .

John Domingue. *Tadzebao and WebOnto: Discussing, Browsing, and Editing Ontologies on the Web*. 11th Knowledge Acquisition for Knowledge-Based Systems Workshop, April 18th-23rd. Banff, Canada, 1998.

Adam Farquhar, Richard Fikes, and James Rice; *The Ontolingua Server: a Tool for Collaborative Ontology Construction*; International Journal of Human-Computer Studies **46**, 707-727, 1997. (http://www.ksl.stanford.edu/KSL_Abstracts/KSL-96-26.html)

Gleb Frank, Adam Farquhar, and Richard Fikes; *Building a Large Knowledge Base from a Structured Source: The CIA World Fact Book*; IEEE Intelligent Systems, Vol. 14, No. 1, January/February 1999. (http://www.ksl.stanford.edu/KSL_Abstracts/KSL-98-16.html)

Natalya Fridman Noy and Mark A. Musen. *An Algorithm for Merging and Aligning Ontologies: Automation and Tool Support* in AAAI-99 Workshop on Ontology Management. Also, SMI Technical Report SMI-99-0799.

Natalya Fridman Noy and Mark A. Musen; *SMART: Automated Support for Ontology Merging and Alignment*; Twelfth Banff Workshop on Knowledge Acquisition, Modeling, and Management; Banff, Alberta, Canada; 1999. Also, SMI Technical Report SMI-1999-0813

Intraspect Knowledge Server, Intraspect Corporation, 1999. (http://www.intraspect.com/product_info_solution.htm)

Y. Iwasaki, A. Farquhar, R. Fikes, & J. Rice; *A Web-based Compositional Modeling System for Sharing of Physical Knowledge*. Morgan Kaufmann, Nagoya, Japan, 1997. (http://www.ksl.stanford.edu/KSL_Abstracts/KSL-98-17.html).

Kevin Knight and Steve Luk; *Building a Large-Scale Knowledge Base for Machine Translation*; Proceedings of the National Conference on Artificial Intelligence (AAAI), 1994.

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

Deborah L. McGuinness; *Ontologies for Electronic Commerce*, Proceedings of the AAAI Artificial Intelligence for Electronic Commerce Workshop, Orlando, Florida, July, 1999.

Deborah L. McGuinness; *Ontological Issues for Knowledge-Enhanced Search*; Proceedings of Formal Ontology in Information Systems, June 1998. Also in Frontiers in Artificial Intelligence and Applications, IOS-Press, Washington, DC, 1998.

Deborah L. McGuinness and Peter Patel-Schneider; *Usability Issues in Knowledge Representation Systems*; Proceedings of the Fifteenth National Conference on Artificial Intelligence, Madison, Wisconsin, July, 1998. This is an updated version of *Usability Issues in Description Logic Systems* published in Proceedings of International Workshop on Description Logics, Gif sur Yvette, (Paris) France, September, 1997.

Deborah L. McGuinness, Lori Alperin Resnick, and Charles Isbell; *Description Logic in Practice: A CLASSIC: Application*; Proceedings of the 14th International Joint Conference on Artificial Intelligence, Montreal, Canada; August, 1995.

Deborah L. McGuinness and Jon Wright; *An Industrial Strength Description Logic-based Configurator Platform*; IEEE Intelligent Systems, Vol. 13, No. 4, July/August 1998, pp. 69-77.

Deborah L. McGuinness and Jon Wright; *Conceptual Modeling for Configuration: A Description Logic-based Approach*; Artificial Intelligence for Engineering Design, Analysis, and Manufacturing Journal - special issue on Configuration, 1998.

K.A. Spackman, K.E. Campbell, and R.A. Cote; *SNOMED-RT: a reference terminology for health care*; Proceedings of the 1997 AMIA Annual Fall Symposium; 1997; 640-644.

William Swartout, Ramesh Patil, Kevin Knight, and Tom Russ; *Toward Distributed Use of Large-Scale Ontologies*, Proceedings of the 10th Banff Knowledge Acquisition Workshop; Banff, Alberta, Canada; November 9-14, 1996.

United Nations Standard Product and Services Classification (UNSPSC) Code organization. http://www.unspsc.org/home.htm

Chris Welty; *An HTML Interface for CLASSIC*; Proceedings of the 1996 International Workshop on Description Logics; AAAI Press; November, 1996.