Artificial Intelligence for Engineering Design, Analysis and Manufacturing (2010), *24*, 127–141.

doi:10.1017/S0890060409000183

Automatic ontology creation using adaptation

VALERIE CROSS[1] AND VISHAL BATHIJA[2]

[1]Computer Science and Systems Analysis Department, Miami University, Oxford, Ohio, USA
[2]Planetsoft Holdings, Burlington, Massachusetts, USA

(RECEIVED July 18, 2008; ACCEPTED January 12, 2009)

Abstract

Ontologies are an emerging means of knowledge representation to improve information organization and management, and they are becoming more prevalent in the domain of engineering design. The task of creating new ontologies manually is not only tedious and cumbersome but also time consuming and expensive. Research aimed at addressing these problems in creating ontologies has investigated methods of automating ontology reuse mainly by extracting smaller application ontologies from larger, more general purpose ontologies. Motivated by the wide variety of existing learning algorithms, this paper describes a new approach focused on the reuse of domain-specific ontologies. The approach integrates existing software tools for natural language processing with new algorithms for pruning concepts not relevant to the new domain and extending the pruned ontology by adding relevant concepts. The approach is assessed experimentally by automatically adapting a design rationale ontology for the software engineering domain to a new one for the related domain of engineering design. The experiment produced an ontology that exhibits comparable quality to previous attempts to automate ontology creation as measured by standard content performance metrics such as coverage, accuracy, precision, and recall. However, further analysis of the ontology suggests that the automated approach should be augmented with recommendations presented to a domain expert who monitors the pruning and extending processes in order to improve the structure of the ontology.

Keywords: Design Rationale; Natural Language Processing Tools; Ontology Evaluation; Ontology Learning; Ontology Reuse

1. INTRODUCTION

An ontology is an explicit specification of a conceptualization (Gruber, 1993) that formalizes the concepts pertaining to a domain, the properties of these concepts, and the relationships that can exist between the concepts. Ontologies are an emerging means of knowledge representation that capture structure and semantics to facilitate interoperability among different applications within a domain. They reduce terminological confusion by representing a common understanding of the domain concepts. Building new ontologies, however, requires substantial time, effort, and cost. Automating the process of ontology reuse would greatly reduce these costs.

Reported ontology reuse has been mostly in the context of the customization and pruning of comprehensive, general purpose ontologies or thesauri to extract relevant portions for a new application (Lonsdale et al., 2002; Volz et al., 2003; Caralt, 2004; Bontas et al., 2005; Ding et al., 2007) or in the context of the adaptation activity of the ontology development process (O'Brien & Abidi, 2006). In contrast, the aim of this research is to investigate how existing learning techniques may be revised to reuse a domain-specific ontology to automate the process of creating a new ontology for a related domain.

This investigation produced a new approach to ontology adaptation (Bathija, 2006) that provides a focused method for reusing domain-specific terminological ontologies. The approach integrates existing software tools for natural language processing with new algorithms developed specifically for the adaptation process. This research examined existing algorithms utilized in a variety of approaches for the reuse of large, generic ontologies, which inspired the development of new algorithms that are specialized to support the adaptation process on domain-specific ontologies.

The new algorithms were developed after first examining a variety of algorithms for general ontology learning and reuse (Maedche, 2002; Navigli, 2002; Navigli & Verlardi, 2004; Buitelaar et al., 2005; Gómez-Pérez & Manzano-Macho, 2005), next determining their usefulness and applicability to the adaptation process, and finally refining them as needed with respect to the aims of the research.

Reprint requests to: Valerie Cross, Computer Science and Systems Analysis Department, Miami University, Oxford, OH 45066, USA. E-mail: crossv@muohio.edu

The new approach is assessed experimentally by automatically adapting an existing ontology in the domain of design rationale for software engineering (Burge, 2005; Burge et al., 2006; Burge & Brown, 2008) to an ontology for the domain of engineering design criteria. The experiment produced an ontology that exhibits comparable quality to previous attempts to automate ontology creation as measured by standard content performance metrics (Spyns & Reinberger, 2005).

The paper is structured as follows. The second section motivates the need for new approaches to ontology creation and presents an overview of our approach that integrates ontology reuse through adaptation and automation. The third section describes the details of our ontology adaptation approach and the components of its software architecture. The fourth section summarizes the performance measures used in the experimental evaluation of the approach, describes the experiment, and analyzes the results of the adaptation process. The fifth section concludes by summarizing the research contributions and presenting future plans for improving the ontology adaptation process.

2. ONTOLOGY REUSE AND ADAPTATION

Developing an ontology can be a costly and tedious task. One approach to reduce costs is to create one ontology from another, that is, to reuse an existing ontology. Ontology reuse can be defined as "the process in which available (ontological) knowledge is used as input to generate new ontologies" (Bontas et al., 2005). Much research on ontology reuse has focused on extracting a subset of concepts from a large and more general ontology to produce an ontology tailored to a specific application (Volz et al., 2003; Noy & Musen, 2004; Stuckenschmidt & Klein, 2004; Ding et al., 2007).

Another approach, which includes a limited form of reuse, is ontology adaptation (O'Brien & Abidi, 2006). Adaptation refers to the localized and specific customization of an existing ontology made to meet the needs of a particular situation. These modifications are based on communal or individual preferences, expertise, user profiles, or application needs and are not necessarily applied back to the existing ontology.

In this paper, ontology adaptation is broader in scope than the aforementioned adaptation activity, but narrower in scope than the reuse of large, general purpose ontologies. The focus is on the reuse of an existing domain-specific ontology to build a new one for a related domain. It assumes that a suitable ontology for reuse has been selected and does not address the process of determining which existing ontology from available candidates is most suitable for reuse.

Our adaptation process consists of two major phases: pruning an existing ontology and then extending the resulting ontology for a new related domain. Pruning eliminates irrelevant concepts from the existing ontology and requires an algorithm that examines each existing concept to determine its significance to the new domain. A decision is made to remove a concept if it is not considered significant. Extending the pruned ontology adds concepts relevant to the new domain and requires an algorithm to determine a concept's relevance. If a concept is considered relevant, then another algorithm determines whether it is considered distinct from existing concepts in the new ontology. If the concept is not distinct, then a decision is made on where to position it within the new ontology.

Both the pruning and extending algorithms were developed after reviewing numerous learning techniques that use text corpora because that has been a predominant approach in the research literature for partially automating the ontology construction process (Kietz et al., 2000; Maedche, 2002; Volz et al., 2003; Navigli & Verlardi, 2004; Buitelaar et al., 2005; Gómez-Pérez & Manzano-Macho, 2005). As stated above, an ontology is an explicit specification of a conceptualization (Gruber, 1993) that formalizes the concepts pertaining to a domain, the properties of these concepts, and the relationships that can exist between the concepts. This definition refers to an intensional ontology that specifies the ontology schema or definition. The extensional ontology consists of the instances of the concepts and relationships defined in the intensional ontology. In the research reported here, the pruning and extending processes are applied to the extensional ontology. No changes are made to the intensional ontology.

The complete details of the ontology adaptation architecture are provided in the next section. Here, the significant contributions of this research are summarized. The pruning algorithm uses the weighted term frequency, which is the term frequency–inverse document frequency measure (TF-IDF), and a domain corpus to generic corpus frequency comparison to determine concept relevance (Volz et al., 2003; Spyns & Reinberger, 2005). However, it uses a bottom-up approach to eliminating concepts from the existing ontology, which differs from the pruning approaches in Navigli (2002) and Volz et al. (2003). It also incorporates another method to determine concept relevance, a statistical Z test that differentiates between concept frequencies for the domain corpus with respect to the general corpus (Spyns & Reinberger, 2005). This new method is added because the results of the evaluations in Volz et al. (2003) showed using absolute term frequency is not significantly different from using weighted term frequency. In addition, the lexical resource JWordNet (http://www.seas.gwu.edu/~simhaweb/software/jwordnet/) is used to determine multiple lexicalizations of the same concept in the corpora to determine concept frequencies.

The extending algorithm draws from the approach in Navigli (2002), which has as its objective the construction of a domain ontology by automatically enriching and reorganizing the WordNet hierarchy. The use of domain concept trees for enriching the WordNet hierarchy suited the objective of extending the pruned ontology with concepts from the domain of the new ontology. OntoLT (Buitelaar et al., 2004) is used to build the domain concept trees. The adaptation software, however, implements a new matching algorithm to determine where to position the domain concept tree within the ontology.

https://doi.org/10.1017/S0890060409000183 Published online by Cambridge University Press

This research has resulted in an ontology adaptation approach and associated software tools whose aim is to start with an existing domain-specific ontology and transform it into a new ontology for a related domain. To evaluate this new approach and software, an experiment in ontology adaptation is carried out to create an engineering design criteria ontology. The domain of engineering design is increasingly employing ontologies (Lin et al., 1996; Eris et al., 1999; Japikse et al., 2003; Fowler et al., 2004; Kitamura et al., 2004) because it is an extremely knowledge-intensive activity and ontologies provide the benefits of knowledge sharing, reuse, and a standard language. A specific example of valuable design knowledge is the rationale behind engineering design decisions.

For this experiment, the selected existing ontology is the design rationale argument ontology for the domain of software engineering utilized in the SEURAT system (Burge, 2005). This ontology is a terminological ontology whose purpose is to support the capture and reuse of design rationale (Lee, 1997) in the evaluation of alternatives for software development. Its intensional ontology consists of an argument entry class that uses a hierarchical structuring relationship. Instances of the argument entry class populate the extensional ontology, as illustrated in Figure 1.

At the top of the hierarchy are the more general or abstract design rationale. As one proceeds down the hierarchy, they are refined. The leaves of the hierarchy represent the most detailed design rationale. The adaptation process is to retain the design task aspects of the existing extensional ontology and adapt the domain from software design to engineering design.

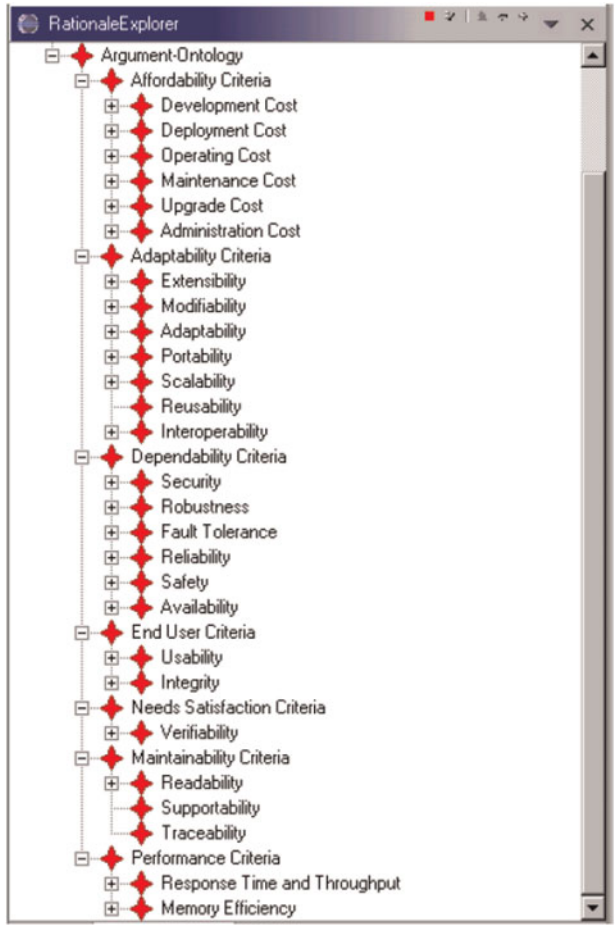

Fig. 1. The top level of the argument ontology. [A color version of this figure can be viewed online at journals.cambridge.org/aie]

3. ONTOLOGY ADAPTATION ARCHITECTURE

The ontology adaptation architecture consists of input resources, existing software tools, and newly developed software tools to implement our adaptation approach. The following sections describe these components and explain the details of the algorithms used in the ontology adaptation process. The overall architecture is shown in Figure 2, with the bottom part of the figure providing an overview of the evaluation process.

3.1. Input resources

Two domain corpora are required: the domain training corpus and the domain test corpus. The domain training corpus is used in determining both the significance of the concepts in the existing ontology during the pruning process and the significance of the new domain concepts during the extending process. The domain test corpus is used to evaluate the performance of the adaptation process.

The training corpus was created by selecting documents from the NASA Technical Reports Server (NTRS; http://ntrs.nasa.gov/), an experimental service that permits users to search for documents such as research reports, journal articles, conference papers, and mission-related operational documents. The first selection criteria used a simple search on the keywords "engineering," "design," and "criteria," If a viewable pdf version existed, then the document's abstract was carefully read to determine if the document contained engineering design information. If so, then it was viewed to see more details and search for references to design criteria. Based on this process, 25 documents were selected with a size of 1.05 MB. The test corpus was created in a manner similar to the training corpus but about 3 months after the training corpus. Updates to the NTRS provided new documents following the same process as used for creating the training corpus. The test corpus consists of 20 documents that are not in the training corpus. The size of the test corpus was 1.30 MB, close to that of the training corpus as recommended in the research literature.

A generic or "neutral" corpus not specific to the domain of interest is used for determining whether a term in the training or test corpus is domain specific. The Reuters-21578 text categorization test collection is used as the generic corpus (http://kdd.ics.uci.edu//databases/reuters21578/reuters21578.html).

https://doi.org/10.1017/S0890060409000183 Published online by Cambridge University Press

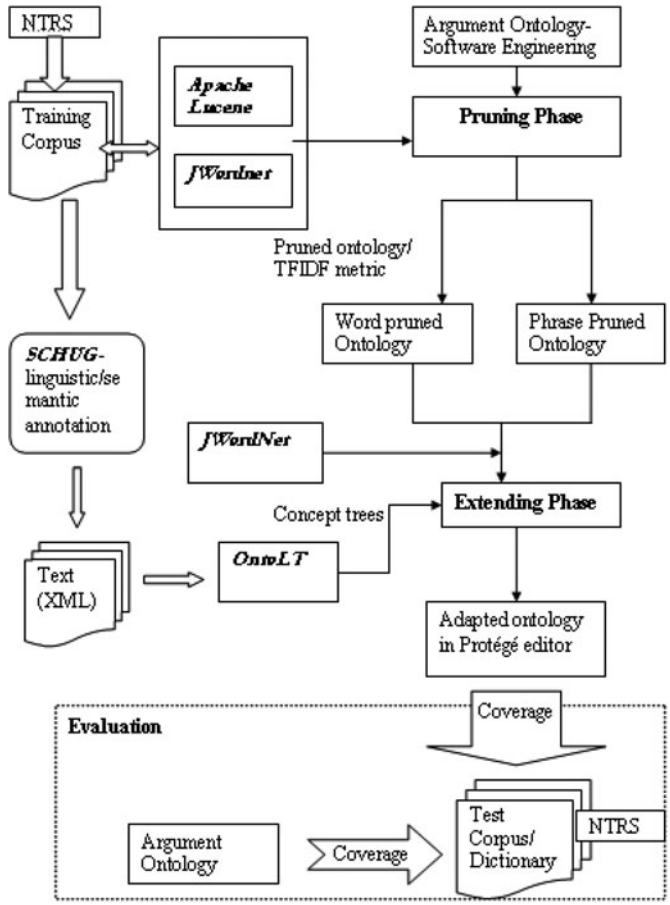

Fig. 2. The ontology adaptation architecture.

3.2. Software tools

The adaptation architecture is composed of several independent software packages integrated with software written for this research to accomplish the pruning and extending tasks of the adaptation process. Additional software was developed for the evaluation process.

Apache's Lucene (http://lucene.apache.org/java/docs/index.html), an open source search engine library, was selected to calculate the term and document frequency of each term in both the training corpus and test corpus. The term frequency $tf(i, j)$ is the number of times that term i appears in the document j. The document frequency is the number of documents in which term i appears, $df(i)$. With N total documents, the TF-IDF value for term i in document j is computed as (Song et al., 2005):

$$\text{TF-IDF}(i, j) = tf(i, j) * \log\left(\frac{\text{N}}{df(i)}\right). \quad (1)$$

JWordnet is used along with Apache Lucene to estimate the term and document frequency of a term and its synonyms appearing in the text corpus. JWordnet is a Java interface to Wordnet (Miller et al., 1990), an English language electronic dictionary accessible from the Internet. WordNet forms a taxonomy or hierarchical ordering of the English language with more general concepts higher in the hierarchy and more specific ones lower. The WordNet synset contains terms that are synonyms for the concept. A total frequency is determined for a concept by adding up the frequencies for all the terms in the synset for the concept. The output from the integration of the Lucene and JWordnet software serves as input to the pruning process.

The extending phase takes as input the domain concept trees that are produced from the sequenced execution of two other software resources SCHUG (Shallow and Chunk-based Unification Grammar tools; Declerck, 2002) and OntoLT (Buitelaar et al., 2004). SCHUG, a rule-based system for English and German, provides annotation of parts of speech, morphological inflections, and phrase and dependency structure for a text file. SCHUG takes plain text documents as input and outputs the annotated file in XML format. The annotated files from SCHUG are input to OntoLT, which maps linguistic entities annotated in the text to ontology class and property candidates.

https://doi.org/10.1017/S0890060409000183 Published online by Cambridge University Press

3.3. Algorithms and software tools for adaptation

This section summarizes the algorithms used to implement the adaptation process. Additional details are found in Bathija (2006). The pruning algorithm requires a method to determine if a concept in the existing ontology is relevant to the new domain. The extending algorithm requires a method to determine if a new domain concept tree is relevant to the new domain. Two approaches are implemented for determining domain relevance: the significant terms list and the term frequency/inverted document frequency measure. The domain relevance approaches are described, followed by a description of the processes of pruning and extending. The effect of these algorithms on introducing inconsistencies into the new ontology is identified as well. Note that these processes are performed on the extensional ontology.

3.3.1. Determining domain relevance

The method used to determine a set of words that differentiate the domain corpora with respect to a general corpus is based on the approaches taken in Kietz et al. (2000), Volz et al. (2003), and Spyns and Reinberger (2005). The assumption with respect to technical texts is that their extended vocabulary constitutes the majority of the distinguishing vocabulary, especially if the general corpus to which this vocabulary is being compared is a collection of broad-spectrum newspaper articles. The process of determining the significant words with respect to a domain involves several steps, first of which is determining a word's relative frequency.

The relative frequency f_{rel} of a word with respect to the corpus is determined as

$$f_{\text{rel}} = (f/N), \tag{2}$$

where f is the absolute frequency of a word in the corpus and N is the total number of words in the corpus. Then the Z value is calculated using the difference between the word's relative frequency in the domain corpus f_{relD} and in the general corpus f_{relG}:

$$z = (f_{\text{relD}} - f_{\text{relG}})/\sqrt{f_{\text{relD}} * (100 - \text{f}_{\text{relD}})/\text{N}_{\text{D}} + \text{f}_{\text{relG}} * (100 - \text{f}_{\text{relG}})/\text{N}_{\text{G}}}\,. \tag{3}$$

Equation (3) is based on the formula for testing the differences of two proportions. The null hypothesis is that $f_{\text{relD}} = f_{\text{relG}}$. The alternative hypothesis is that $f_{\text{relD}} > f_{\text{relG}}$. A word is consider significant if the value of Z is greater than or equal to 1.64 for 5% significance or 2.32 for 1% significance.

The list of significant words is determined for both the training domain corpus and the test domain corpus. The training domain significant words list is used in the pruning algorithms as described in the following section, and the test domain significant words list is used in the evaluation of the adapted ontology as described in Section 4.

The TF-IDF given in Eq. (1) is another approach used to determine relevant terms. Terms that appear too frequently and too rarely are ranked lower than terms that appear with moderate frequency (Jones, 1972). Either the TF-IDF ratio between the domain and general corpus or the Z value method may be used for determining significance with the threshold value for each method settable by the user. The default for TF-IDF ratio is 5 to 1 and for the Z value, the default threshold is 1.64.

3.3.2. Pruning

The aim of pruning is to remove concepts from the original ontology that are not relevant to the new domain. Concepts that are not domain specific should be less frequent in a domain specific corpus than in general texts (Kietz et al., 2000); therefore, the creation of the domain-specific corpus is important and should be done by domain experts. As previously explained, domain experts were not available for creating the domain specific corpus, but a careful and conscientious approach described in Section 3.1 was used to obtain pertinent documents for the domain-specific corpora.

The approach for pruning is motivated by Volz et al. (2003) and Caralt (2004), but uses a bottom-up approach starting with leaf concepts in the ontology instead of examining the relevance of all concepts within the ontology. It first creates a list of leaf concepts from the original ontology and an initial empty list for the leaf concepts determined to be relevant to the new domain. Only leaf concepts are considered for pruning. Each leaf concept is checked to see if it is relevant to the new domain based on the relevancy approach selected by the user. If the leaf concept is not considered relevant, it is pruned from the ontology and deleted from the list of leaf concepts. After all leaf concepts have been examined for relevancy, the leaf concepts not deleted are transferred to the domain relevant list of leaf concepts. The process begins again with a new list of leaf concepts that have not already been determined relevant to the new domain. A node in an ontology may have multiple children so all the children of a concept must be pruned before the parent itself becomes a leaf concept and eligible for pruning.

Statistical processing is used to identify which leaf concepts are relevant to the domain. The domain corpus text is parsed using Lucene to calculate the term frequencies and document frequencies of all the terms present in the domain training corpus. The frequency used in this approach is actually a total term frequency. For a given term, the frequency of all its synonyms is added to its frequency to determine the total frequency for a document. For example, if an ontology node includes the term "deadlock" and "deadlock" occurs twice and "stalemate" occurs four times in the document, then the term frequency of the concept "deadlock" is six for that document. The same approach is followed while calculating the number of documents that contain the term "deadlock" or any of its synonyms. JWordnet is used to determine

https://doi.org/10.1017/S0890060409000183 Published online by Cambridge University Press

all the synonyms for a particular term. These terms along with their frequencies are stored in a list used to determine the significance of a concept.

The significance of a concept in the original ontology is determined by the presence in the corpus of either the individual words used in the label of the concept or the entire phrase used to label the concept. In the first case, it is determined if all the words or a percentage of words that form the phrase are domain specific. Specificity is determined either using the term frequency-inverted document frequency measure or the *Z* value of the word. In this approach the words could occur anywhere in the corpus, and the phrase would still be considered significant. With this word-based approach, the ontology concept is considered relevant to the domain if all or a user settable percentage of the terms in the concept's label are considered relevant. Stop words such as "a," "the," "and," "or," and "of" are not considered. For example, the concept node "Minimizes Equipment Cost" in the argument ontology has three terms. This concept node is significant if all the three terms or a specified percentage of the number of terms are significant. They need not occur adjacent to one another in the same sentence in the corpus.

There are two methods for the phrase-based approach: match the phrase exactly or utilize a slop value for the phrase. The slop is an edit distance that permits other words within the query. With the phrase-based exact approach, an exact phrase match or synonym variations of the phrase in the corpus must occur. Because it is unclear from just examining the text in the corpus that phrase frequencies need to be estimated, the processing starts by examining the phrases that make up each node in the original ontology and then estimates the phrase frequency and document frequency for each of those phrases. Several Lucene API are used together to construct a query to look for a particular phrase within the corpus and determine the phrase count in each document.

An initial implementation of phrase-base approach attempted to create a cross-product of all the synonyms for the words in a phrase by using the synonym set associated with each term in a node's label. For each term in the label, the synonym set is retrieved and then the cross-product of the synonym sets for each term in the label is used to create all variations for the phrase used to label the node. For example, if the ontology node is labeled "avoids deadlock" and the synonym set for "avoids" consists of {avoids, prevents} and the synonym set for "deadlock" consists of {deadlock, stalemate}, then the following set of phrases would be searched for: "avoids deadlock," "avoids stalemate," "prevents deadlock," and "prevents stalemate." After an initial experiment to create a cross-product of all the synonyms for the words in a phrase by using the synonym set associated with each term in a node's label required too much execution time, the approach used is to select the term in the phrase with the highest significance value and its synonyms are used to create equivalent phrases for the node only using replacement on that most significant term.

3.3.3. *Extending*

The ontology obtained from the pruning phase represents the starting point for the extending phase. Concepts and the taxonomic relations relevant to this domain are added to the pruned ontology using the domain training corpus and the integration of several software resources.

Using domain corpus text files, SCHUG produces annotated XML files that are input to the OntoLT Protégé plugin with the pruned ontology as the opened ontology. The OntoLT software produces concept trees and simply adds these as totally new root concepts within the existing ontology. New software was developed for the adaptation process to allow merging of these concept trees into the pruned ontology. The approach used in the extending process is based on Navigli (2002), which develops domain ontologies by automatic enrichment and reorganization of the WordNet hierarchy. In their approach, domain knowledge in the form of domain concept trees are carefully attached as children to existing WordNet concept nodes.

Given a domain tree *T* and the root of that tree *r*, attaching the domain tree *T* to the WordNet hierarchy disambiguates the root *r* of the domain tree with respect to the whole WordNet ontology. For the domain-specific ontology adaptation process, the algorithm does not disambiguating the sense of the root of the OntoLT domain concept tree as done in Navigli (2002), but instead finds the best significant matching concept currently existing in the new ontology by using a similarity scoring function between the domain concept tree and a concept in the new ontology. It is based on the overlap between terms present in an OntoLT domain concept tree and the terms present in the label of an ontology concept. This overlap is measured using the Jaccard index value, which measures the similarity between two different sets *A* and *B*. The Jaccard coefficient for sets *A* and *B* is defined as the cardinality of their intersection divided by the cardinality of their union (Jaccard, 1908):

$$J(A, B) = \frac{|A \cap B|}{|A \cup B|}. \tag{4}$$

Let *T* be a domain concept tree produced by OntoLT. Each domain concept tree consists of at least a root node and may have multiple descendent nodes. Terms(T) specifies the set of terms used to label each of these nodes in *T*. For instance, consider the domain concept tree *T* with a root labeled "Analysis" that has two children nodes labeled "thermal analysis" and "posttest analysis." Then terms(T) = {"analysis," "thermal," "post," "test"}. Syn-terms(T) specifies terms(T) $\cup\ \forall t \in$ (terms)T synset(t), the union of the original terms used to label the nodes in *T* with the synonym sets of those terms as found using JWordnet. For example, the term thermal has synonyms "heat," "solar," and so forth. These terms are also included in syn-terms(T).

For each node in the OntoLT concept tree, a *Z* value is calculated and an average *Z* value is determined for the concept tree. Each concept tree produced by OntoLT is examined and only those with an average *Z* value meeting the user settable

https://doi.org/10.1017/S0890060409000183 Published online by Cambridge University Press

Z value significance level are considered for merging into the ontology. The attaching of the domain concept trees begins at the leaves of the ontology.

Initially, the examination list contains all the leaf nodes of the pruned ontology. The syn-terms list is created for the leaf node by adding in the synonym sets for each term used to label the leaf. For example, the syn-terms for the leaf "Avoids Deadlock" includes the terms "avoids," "deadlock," and all the synsets of these terms such as "prevents," "stalemate," "standstill," and so forth. These synsets are obtained using JWordNet.

Using the examination list, a procedure findBestNodeMatch processes each node looking for the best match for the domain concept tree *T* from OntoLT using the Jaccard value. If a best match leaf node is found that satisfies the minimum specified Jaccard threshold for matching, the tree *T* is added as a child of that particular leaf node in the pruned ontology. It is possible that a particular domain concept tree may have a Jaccard index value that is lower than Jaccard threshold set by the user. In such cases, the leaf node that has the highest Jaccard value is selected and the process of examining its ancestors is begun. This approach is based on the reasoning that its parents being more general concepts than the child might result in a closer match. This proceeding up the pruned ontology stops if the resulting Jaccard value between the concept tree *T* and an ancestor node either exceeds the threshold or decreases in value or if no more ancestors are left. If the threshold is exceeded, then the domain concept tree node is added as a child of that ancestor node; otherwise, it is added to the pruned ontology as a separate tree within the ontology.

It may be possible to find multiple leaf nodes that meet the match criteria for a domain concept tree. In such cases, an association of the domain concept tree *T* with the most specific leaf is desired. The specificity of the leaf is determined by its depth. The leaf with the greater depth is selected. If the leaves happen to have the same specificity then the first leaf found is arbitrarily selected.

A possible modification of this matching algorithm could be to rename the node in the pruned ontology. For instance, if the best match for the concept tree "control team" is "provides access control" then instead of making "control team" the child of "provides access control" we could rename the original node as "provides access control|control team" ("provides access control or control team"). Another improvement could be to investigate switching the parent child relationship, that is, "control team" might become the parent of the node "provides access control." These two modifications require a technique to determine which of the two concepts is more general and to assign that as the parent. This research did not investigate such modification.

Two approaches in generating the examination list of the pruned argument ontology can be used. In the static approach, only the leaves of the original pruned ontology are used for finding a match with a domain concept tree *T*. In the dynamic approach, when a domain concept tree is added to the pruned ontology, its leaf nodes are then considered for the matching process and are added to the examination list. A dynamic approach was also used in the experiments because it might produce a better adapted ontology.

3.3.4. Adaptation process and inconsistencies

A variety of approaches for handling the introduction of inconsistencies as a result of making modifications to an existing ontology have been proposed (Haase et al., 2005). Most approaches specifically address description logics-based ontologies and focus on logical inconsistencies. For example, the DINO system (Novacek et al., 2007) has an ontology reasoning wrapper that is used when a learned ontology is being merged with a master ontology. It handles three inconsistencies: cycles in the subclass hierarchy, disjointness–subsumption conflicts, and disjointness–instantiation conflicts. A disjointness–subsumption conflict occurs in the intensional ontology when two classes are said to be disjoint but one is a subclass of the other or they share a common subclass. A disjointness–instantiation conflict occurs in the extensional ontology when an instance belongs to two disjoint classes. The solution taken is to remove the addition that is causing the inconsistency.

The disjointness–subsumption conflict is also addressed by looking at two causes for the conflict: polysemy of concept names and overgeneralizations (Ovchinnikova & Kühnberger, 2007). For the polysemy cause, the inconsistency is reported to the user because distinguishing between word senses without the help of an ontology engineer or additional external information about the usage context of the concept is not possible. For the overgeneralizations cause, an algorithm repairs multiple overgeneralizations by replacing the conflicting definitions with their least common subsumer.

Because the adaptation process described in this research only modifies the extensional ontology and the only class defined in the intensional ontology is the argument entry class, the three previously described logical inconsistencies cannot be technically introduced. However, if one views the extensional ontology as a hierarchically organized domain vocabulary, semantic inconsistencies might be introduced by the adaptation process.

The pruning process should not introduce semantic inconsistencies because it starts with the leaf concepts in the argument extensional ontology. A higher level concept is not eligible for pruning until all its children have been pruned. However, it would be useful for a domain expert to review the concept entries that are considered to be irrelevant before they are pruned. The automatic determination of relevance by the adaptation software eliminates tedious work, but making the process semiautomatic by presenting the domain expert with a list of suggestions for pruning would serve as a safeguard to reduce the chance that truly relevant entries are pruned by the system.

The extending phase may introduce semantic inconsistencies in the generalization hierarchy of the vocabulary. For example, an OntoLT domain concept tree might be erroneously added as a new root entry instead of positioned correctly as a child subtree of an already existing entry or vice versa. In addition, the extending phase does not determine if the root of an OntoLT concept tree matches sufficiently well to an existing entry in

https://doi.org/10.1017/S0890060409000183 Published online by Cambridge University Press

the ontology so that only its children need to be added under the existing entry in the ontology. This shortcoming produces an extra level in the generalization hierarchy. Again, a semiautomatic process that allows the domain expert to first review and then select the concept trees to be integrated and next visually inspect the resulting extended ontology for editing corrections could help to eliminate such semantic inconsistencies. In addition, heuristic processing needs to be added to address such inconsistency problems. For example, the algorithm for addressing multiple overgeneralizations (Ovchinnikova & Kühnberger, 2007) might be revised for the extending process to better position the domain concept trees, eliminate extra levels in the hierarchy, and even evaluate the consistency of the OntoLT-produced domain concept trees.

4. EXPERIMENTAL RESULTS

The performance of the ontology adaptation process was evaluated through numerous experiments. The following sections describe the metrics for performance evaluation, the experimental method with the parameters used in the experiments, and an analysis of the experimental outcomes.

4.1. Performance evaluation metrics

Approaches to evaluate an ontology are broadly classified as (Brank et al., 2005): evaluation by domain experts, comparison to a gold standard ontology, evaluation by using an ontology in an application, and comparison of an ontology with a document collection relevant to the domain. Because no gold standard ontology was available and conscientious attempts to get assistance from NASA domain experts failed, the last approach was used. The evaluation metrics to assess the resulting adapted ontology are coverage, accuracy, precision, and recall (Spyns & Reinberger, 2005).

Coverage represents the degree to which the concepts in an ontology represent the domain (Spyns & Reinberger, 2005). The words are grouped into frequency classes, that is, the absolute frequency of the word in the entire corpus. For example if the term "system" occur 30 times in the corpus and another term "performance" also occurs 30 times in the entire corpus, then the frequency class 30 contains the words "system" and "performance." A modified approach is used because not many frequency classes exist for words used in ontology concept labels. The coverage is estimated by finding the overlap between the words in a frequency class from the test domain corpus, $word\_corpus\_freq\_class_i$, with all the words labeling the concepts in the ontology, *words_concept_labels*. This calculation is done for each frequency class and averaged over the number of frequency classes for the test domain corpus, n as

$$\frac{\sum_{i=1}^{n} \frac{|word\_concept\_labels \cap word\_corpus\_freq\_class_i|}{|word\_corpus\_freq\_class_i|}}{n}. \quad (5)$$

Accuracy is related to coverage but is based on Zipf's law (Zipf, 1949). According to this law, the more frequently a word is used, the lesser meaning it carries. Hence, the high-frequency classes mainly contain words not carrying much meaning. A set of relevant frequency classes is determined for the domain test corpus. The accuracy is computed as coverage using only the relevant frequency classes:

$$\frac{\sum_{i=1}^{n} \frac{|word\_concept\_labels \cap word\_corpus\_rel\_freq\_class_i|}{|word\_corpus\_rel\_freq\_class_i|}}{n}. \quad (6)$$

where $word\_corpus\_rel\_frequency\_class_i$ designates a relevant frequency class. Relevant frequency classes are determined using the Z value statistic described in Section 3.3.1. A list of significant words is created for the domain test corpus using the Z statistic. A frequency class is considered relevant, as suggested in Spyns and Reinberger (2005), if 60% of the terms in the frequency class are present in the significance list of words. All the frequency classes satisfying the criteria are the relevant classes used in Eq. (6). The required percentage to determine relevance is a user settable parameter.

Precision determines if the concepts in the ontology are relevant to the domain. It is difficult to determine if the concepts in an ontology are correct without the help of human evaluation or a gold standard ontology. An artificial "gold standard" is developed as a list of statistically significant words in order to determine correctness (Spyns & Reinberger, 2005) using the Z value described in Section 3.3.1. Precision is then determined as the ratio of number words in the ontology concept labels overlapping with words in the statistically significant words list, *statistically_relevant_words* to the number of words in the ontology concept labels

$$\frac{|word\_concept\_labels \cap statistically\_relevant\_words|}{|word\_concept\_labels|}. \quad (7)$$

Recall is used to determine if all the relevant concepts in the domain have been retrieved. This measure is also determined using the statistically significant words list. Recall is defined as the ratio of number of words in the ontology that overlap with words in the statistically significant list to the number of statistically significant words. It represents how much of the artificially created "gold standard" is contained within the learned ontology and is given as

$$\frac{|word\_concept\_labels \cap statistically\_relevant\_words|}{|statistically\_relevant\_words|}. \quad (8)$$

4.2. Format and parameters for experiments

The design rationale argument ontology, the domain training corpus, and the general corpus are input to the ontology adaptation software for each experiment. The general corpus and the domain test corpus are used in determining the performance measures coverage, accuracy, precision, and recall.

The original design rationale argument ontology was evaluated with respect to general corpus and the domain test

https://doi.org/10.1017/S0890060409000183 Published online by Cambridge University Press

corpus using these four measures as the base case. These results for the original ontology in the row labeled Argument Ontology of Table 1 can be compared to those for each experiment of the adaptation process.

The experiments were done in two steps: determining the best results from the pruning process and evaluating the extending process on those results. In the first step, experiments were run using the various parameters defined for the pruning phase. These results provided a set of pruned ontologies for performance comparison with respect to the four measures. The approach taken for selecting the pruned ontologies for the next phase of extending was based on selecting the pruned ontology with the highest coverage, the pruned ontology with highest accuracy, the pruned ontology with the highest precision and the pruned ontology with the highest recall. Each selected pruned ontology was then used as input to the extending process to produce its corresponding resulting adapted ontology.

For pruning, the first selection parameter is either word pruning (WP) or phrase pruning. For WP, the parameters are domain relevance method (TF-IDF or Z), domain relevance cutoff value (2:1 or 5:1 ratio for TF-IDF, 1.64 or 2.32 for Z), and percentage of terms in the ontology concept label that must meet the relevance cutoff value (all or 60%). There are eight possible parameter settings for WP. For phrase pruning, the parameters are TF-IDF ratio value (2:1 or 5:1) and synonym substitution for significant term (Yes or No). There are four possible setting for phrase pruning.

After initial experiments with phrase pruning, it was determined that the phrase pruning algorithm produced extremely low values for all the measures except for precision because it basically pruned almost the complete argument ontology leaving only around 20 concepts in the best case. As noted previously, an improvement to use slop values and the cross-product of synonyms for terms in the ontology concept labels was implemented and tested but required too much computation time. Because of the poor results with phrase pruning, only WP ontologies were considered as input to the extending process.

For extending by merging, the first parameter is the Z value relevance cutoff for the domain concept tree ($Z = 1.64$ or $X = 2.32$), the failing match action (ignore or separate) and the merging environment (static or dynamic). The failing match action determines whether a domain concept tree is ignored or added as a separate tree to the ontology when it does not meet the Jaccard values (JVs) threshold for merging into the ontology. The Jaccard threshold value is also settable, but after experimenting and collecting data about the JVs produced during initial testing, it was determine that the average JV was 0.058. The Jaccard threshold value was set at JV = 0.05, and used in the final testing of the extending process of the ontology adaptation software.

4.3. Analysis of experimental results

A subset of the WP ontologies was selected for the extending process and subsequent analysis. For WP, the Z value relevance produced better results for all performance measures than the TF-IDF relevance. The results of WP with the two Z values (1.64 or 2.32) and the two percentages of words in concept labels (60% or All) showed that the test case $Z = 1.64$, 60% has the highest number of concepts and is tied for highest performance values for coverage, accuracy, and recall. The test case $Z = 2.32$, All has the least number of resulting concepts 113 and the best performance for precision. An extra test case with $Z = 1.64$, All has a resulting 159 concepts. This number of concepts falls between the other two described cases. These three pruned ontologies were selected for continuation to the extending phase.

Table 1 shows the final results after the extending process on the three pruned ontologies. Initially, the plan was to do merging by ordering the OntoLT produced concept trees on their Z values. This feature was not implemented so that all extending steps were done using unordered (UO) concept trees.

Table 1. *Performance results after extending the three pruned ontologies*

		Coverage	Accuracy	Precision	Recall	No. Concepts
Argument ontology	1	16.24	16.41	30.74	12.90	280
WP $Z = 1.64$, all, merge $Z = 1.64$ (UO), ignore, JV = 0.05, stat	2	22.34	22.56	31.62	14.91	350
WP $Z = 1.64$, all, merge $Z = 2.32$ (UO), ignore, JV = 0.05, stat	3	22.34	22.56	32.54	12.61	290
WP $Z = 1.64$, all, merge $Z = 2.32$, (UO), separate, JV = 0.05, stat	4	23.86	24.10	33.83	14.36	357
WP $Z = 1.64$, all, merge $Z = 1.64$ (UO), separate, JV = 0.05, stat	5	25.88	26.15	32.98	17.66	447
WP $Z = 1.64$, 60%, merge $Z = 1.64$ (UO), ignore, JV = 0.05, stat	6	24.36	24.61	29.65	18.00	486
WP $Z = 1.64$, 60%, merge $Z = 2.32$ (UO), ignore, JV = 0.05, stat	7	24.36	24.61	30.30	15.81	414
WP $Z = 1.64$, 60%, merge $Z = 2.32$ (UO), separate, JV = 0.05, stat	8	25.88	26.15	31.21	17.23	474
WP $Z = 1.64$, 60%, merge $Z = 1.64$ (UO), separate, JV = 0.05, stat	9	27.41	27.69	30.69	20.22	570
WP $Z = 2.32$, all, merge $Z = 1.64$ (UO), ignore, JV = 0.05, stat	10	21.82	22.05	33.53	12.73	287
WP $Z = 2.32$, all, merge $Z = 2.32$ (UO), ignore, JV = 0.05, stat	11	21.82	22.05	34.50	10.58	234
WP $Z = 2.32$, all, merge $Z = 2.32$ (UO), separate, JV = 0.05, stat	12	23.35	23.59	35.82	12.86	314
WP $Z = 2.32$, all, merge $Z = 1.64$ (UO), separate, JV = 0.05, stat	13	25.88	26.15	34.96	16.27	404

https://doi.org/10.1017/S0890060409000183 Published online by Cambridge University Press

An examination of the results produces an initial reassuring result. For coverage and accuracy, all the test cases have improved values over those taken on the original argument ontology. For precision, most test cases have higher values than that of the original design rationale argument ontology except for points 6, 7, and 9. In each of these cases, the WP significance is set at the lower value 1.64 and only requires 60% of the words in an ontology concept label to be found in the significant words list. These parameter values cause the denominator in the precision formula to increase because more ontology concepts are not pruned; therefore, there are more leaves for the extending process to use for adaptation. Note that these cases include the highest and second highest value for the recall measure.

Similarly, for recall, most test cases have higher values than that of the original design rationale argument ontology except for points 10, 11, and 12. These cases have the fewest number of concepts added to the ontology. All of these have severe pruning with a *Z* value of 2.32 and require ALL of the words in the ontology concept label to be found in the significant words list. Note that these test cases include the highest precision value and other very high precision values.

The following scatter plots reflect the data contained in Table 1. The point labels on the plots correspond to the point label in the second column of the table.

Figure 3 shows graph 1, which reveals the close correspondence between the coverage and accuracy values with the accuracy performance measure always higher, although not by much. For accuracy, the average should be over fewer frequency classes and the calculated ratios should be higher, because over a majority of the words in the frequency class are consider domain significant. Precision and recall are close as seen in graph 2 in Figure 4. Table 1 also shows this effect since for all cases; when the merge *Z* value increases from 1.64 to 2.32 and the rest of the parameter are identical, the precision improves but the recall worsens. This result occurs because the domain concept tree requires a higher average *Z* value before it is merged into the pruned ontology.

A visual examination of graph 3 in Figure 5 appears to uncover two separate lines, each showing as precision increases, the number of concepts decreases. The groups of points are group 1 consisting of 6, 7, 2, 3, 10, and 11, and group 2 consisting of 9, 8, 5, 13, 4, and 12. In Table 1, group 1 points are all cases were merging ignores domain concept trees that do not meet the JV match threshold. Group 2 points are all cases were merging adds domain concept trees as separate roots if they do not meet the JV match threshold. There appears to be some interaction between the other parameters and the ignore/separate parameter that causes lower precision for the group 1 (ignore) points compared to group 2 (separate) points with similar number of concepts.

An examination of graph 4 in Figure 6 shows a tendency of increasing recall with increasing number of concepts. There are pairs that represent very close recall values and number of concepts where there is a small reversal: 2 and 4, 5 and 8, and 7 and 13. For the pair 2 and 4, more merging occurs with point 2 because it has a lower *Z* value of 1.64 but point 4 allows separate domain concept trees to be added as roots. These two parameters appear to be balancing each other. Likewise, for both the other pairs, there is a balancing. For test points 5 and 13, there is more pruning with a stricter ALL words meeting significance test for point 5 and a higher *Z* value requirement for point 13. This, however, is balanced

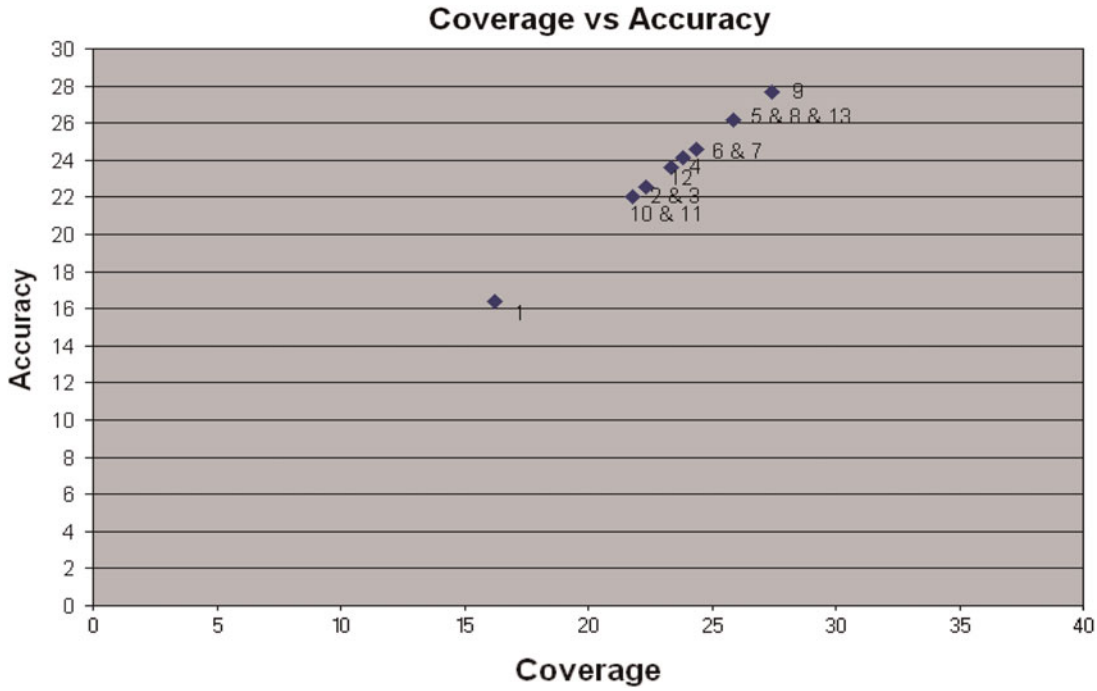

Fig. 3. Graph 1: coverage versus accuracy. [A color version of this figure can be viewed online at journals.cambridge.org/aie]

https://doi.org/10.1017/S0890060409000183 Published online by Cambridge University Press

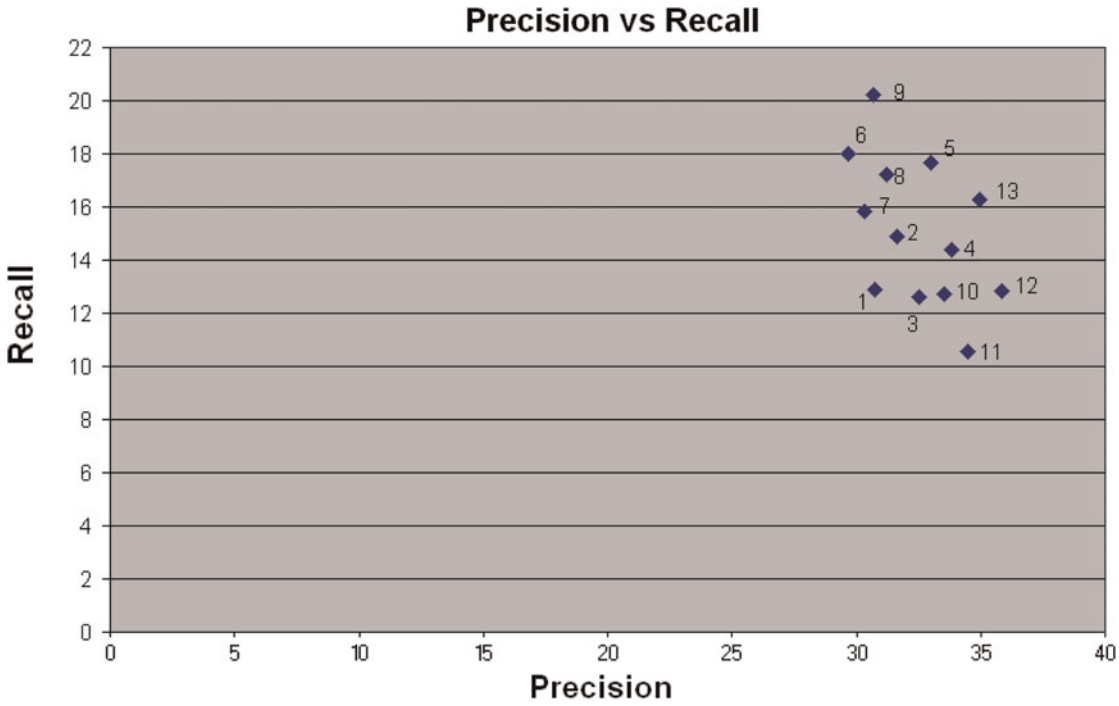

Fig. 4. Graph 2: precision versus recall. [A color version of this figure can be viewed online at journals.cambridge.org/aie]

with more merging occurring for both because a lower Z value of 1.64 is used.

Graph 5 in Figure 7 also reveals a pattern of increasing accuracy with increasing number of concepts if the initial argument ontology point 1 is ignored. If a vertical line is drawn at the 25 marker for accuracy, the points are separated into two groups: those to the left with less than 350 concepts and those to the right with more than 350 concepts. A linear pattern exists except for a reversal of points 6 and 13. Point 13 test case results in 404 concepts and point 6 test case has 486 concepts. The point 13 test case has stricter pruning parameters than the point 6 test case but permits adding separate domain concept trees when the JV match threshold is not matched for merging.

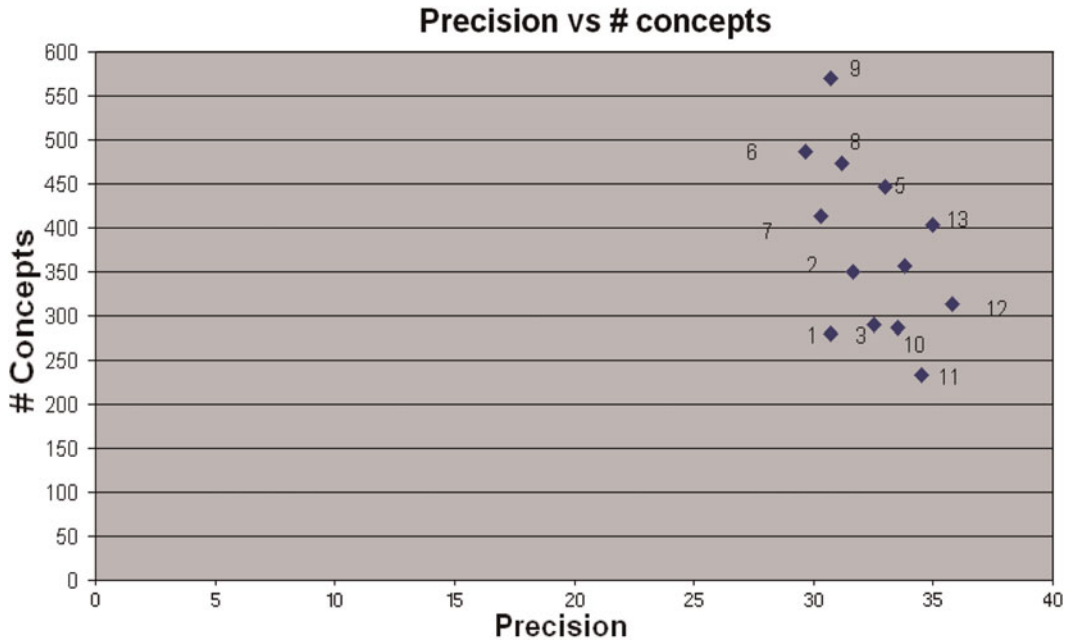

Fig. 5. Graph 3: precision versus the number of concepts. [A color version of this figure can be viewed online at journals.cambridge.org/aie]

https://doi.org/10.1017/S0890060409000183 Published online by Cambridge University Press

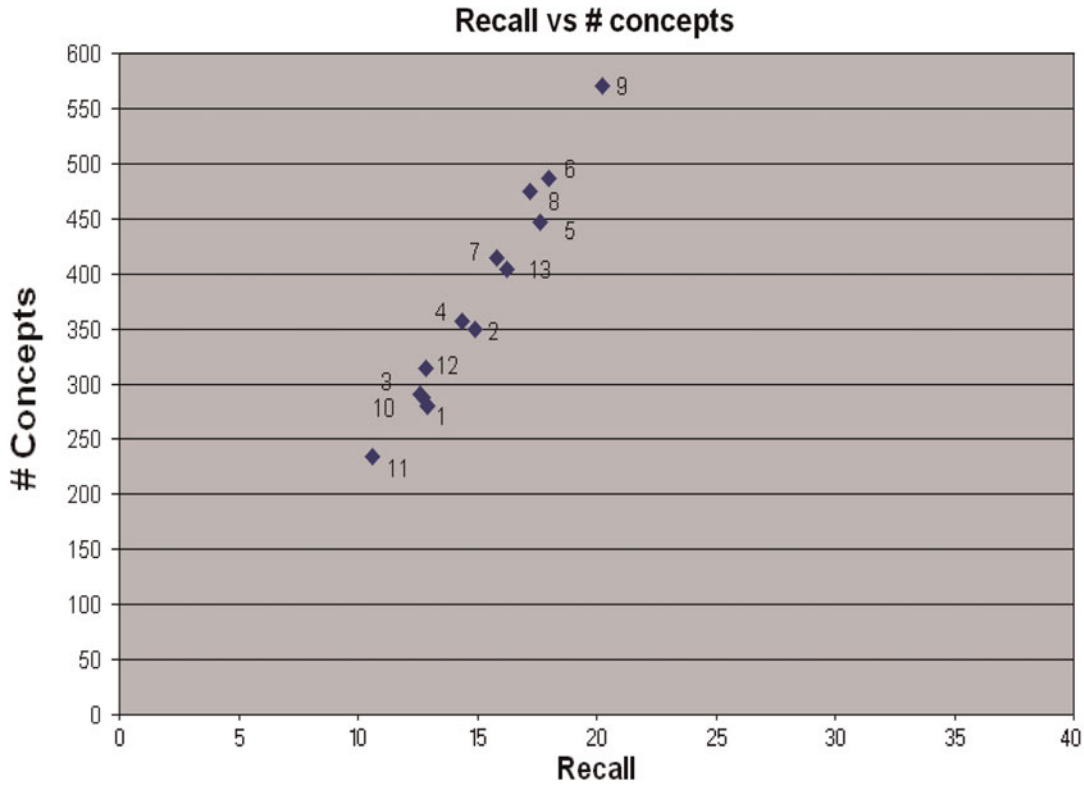

Fig. 6. Graph 4: recall versus the number of concepts. [A color version of this figure can be viewed online at journals.cambridge.org/aie]

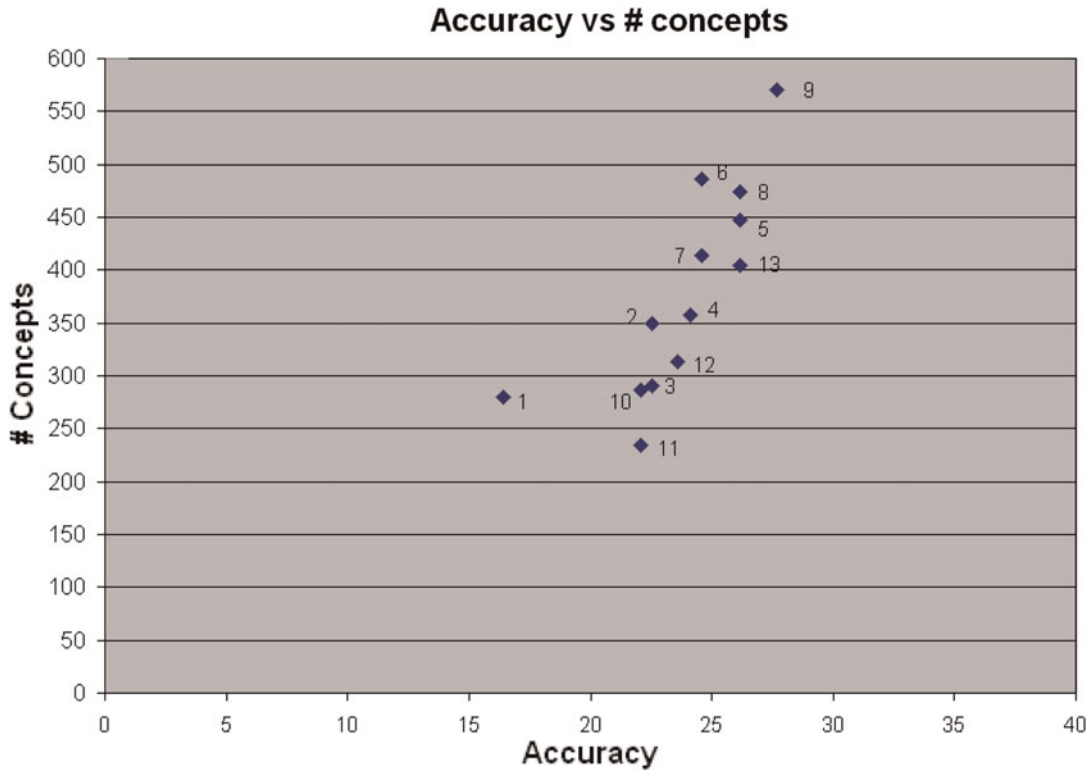

Fig. 7. Graph 5: accuracy versus the number of concepts. [A color version of this figure can be viewed online at journals.cambridge.org/aie]

https://doi.org/10.1017/S0890060409000183 Published online by Cambridge University Press

Because of the computational complexity of doing dynamic merging, only the best results from the above static cases were dynamically merged. The two cases selected were the WP $Z = 1.64$, 60%, merge $Z = 1.64$ (UO), separate, JV = 0.05 case because it had the highest coverage (27.411%), accuracy (27.69%), and recall (20.37%), and the WP $Z = 2.32$, all, merge $Z = 2.32$ (UO), separate, JV = 0.05, because it had the highest precision (35.685%). The results from dynamic merging resulted in no significant differences for the performance measures.

Table 2 shows the best improvement for each performance measure based on the two best test cases, WP $Z = 1.64$, 60%, merge $Z = 1.64$ (UO), separate, JV = 0.05 case with the highest coverage, accuracy and recall, and the WP $Z = 2.32$, all, merge $Z = 2.32$ (UO), separate, JV = 0.05 with the highest precision. The results show a substantial percentage increase in coverage, accuracy, and recall, and more than double the number of concepts when compared to the original design rationale argument ontology.

Although an improvement is made over the initial ontology, the resulting best performance measures are still low. These results, however, if compared to those reported in Spyns and Reinberger (2005) as 39.68% for coverage, 52.1% for accuracy, 58.78% for precision, and 9.84% for recall appear to be at a reasonable starting place for work to proceed in making improvements to the ontology adaptation software. Some ideas for future improvements are presented in the following section.

All the comparisons reported above are based on these four performance measures that deal only with the content of ontologies. In addition, one would like to visually examine the resulting adapted argument ontologies for human evaluation. Detailed examples of visual inspection of the resulting adapted ontology with the highest coverage, accuracy, and recall, the point 9 test case, are explained in Bathija (2006). From the manual analysis on this test case, a needed improvement for the extending process is a method to determine when the root concept of the OntoLT domain concept tree should be eliminated and only its children added as children to an existing ontology concept. For example, the whole domain concept tree with root "communication" produced by OntoLT is currently placed as a subtree under the "Minimizes Communication Cost" concept of the pruned ontology. The "communication" root introduces another level that is not needed. Only its subconcepts ("communication-effective") should be added as a subconcept of "Minimizes Communication Cost."

Table 2. *Changes in performance measures before and after adaptation*

	Coverage	Accuracy	Precision	Recall	No. Concepts
Best measures from test cases	27.41	27.69	35.82	20.22	570.00
Argument ontology	16.24	16.41	30.74	12.90	280.00
Difference	11.17	11.28	5.08	7.32	290.00
Increase (%)	68.76	68.74	16.54	56.74	103.57

4. CONCLUSIONS AND FUTURE DIRECTIONS

One approach to reducing the cost of creating ontologies is to reuse an existing ontology. Typically, a large generic ontology is used to extract a smaller more domain specific ontology. The ontology adaptation approach described here focuses on starting with a domain-specific ontology that is to be adapted to a new ontology in a related domain. The approach performs the process without domain expert intervention to experiment with how well general ontology learning methods could be modified to achieve domain-related adaptation.

After surveying various approaches to ontology learning and reviewing the available software tools for use in ontology learning, a new adaptation architecture was developed. Major algorithms developed for this adaptation process are those for bottom-up pruning and for matching a domain concept tree to an ontology concept in the extending phase. The pruning algorithm incorporates techniques used for ontology evaluation because the objective was to prune concepts from the original domain that are not relevant to the new domain. Existing pruning techniques are typically used after the learning process has created an initial ontology for purposes of removing irrelevant concepts that have been included in the learned ontology. The use of the Z value statistic suggested for ontology evaluation (Spyns & Reinberger, 2005) is a new application for determining whether a concept is related to the ontology domain in the pruning process of the ontology adaptation software.

The objective of the extending process is to position each OntoLT domain concept tree in the pruned ontology. The algorithm for extending the pruned ontology is inspired by the approach taken by Navigli's (2002) as described in Section 3.3.3; however, it uses a different matching approach that is based on measuring the overlap of the synonym sets for the terms labeling the OntoLT domain concept tree and the synonym sets for the terms labeling a concept in the ontology. The algorithm for extending the pruned ontology also incorporates a new use of the Z value statistic to evaluate the significance of the domain concept trees and to exclude those that do not meet the specified Z cutoff value. This use of the Z value is a unique application of a parameter used in ontology evaluation for the purpose of ontology adaptation.

An experiment was performed to adapt the design rationale argument extensional ontology for the software engineering domain to the domain of engineering design. The results of the ontology adaptation approach as measured by standard content performance measures are comparable to that of other attempts to automatically produce ontologies. Although this adaptation process was completely automated for research purposes, a semiautomatic approach with a domain expert to help direct the process is recommended. The domain expert can monitor the intermediate results of the adaptation phases. For example, in the pruning phase, a list ordered by the significance

https://doi.org/10.1017/S0890060409000183 Published online by Cambridge University Press

value of all the concept nodes identified for removal from the ontology should be provided to the domain expert. The domain expert could then review the list and uncheck the removal box for each concept the expert determines is indeed relevant to the domain of the new ontology. Likewise, the extending process could display the *Z* ordered (highest to lowest) list of domain concept trees to be merged into the ontology. The domain expert could then select the ones most related to the domain for merging. The system could merge those and then visually display the results. The user could then decide whether to undo an extension and/or reposition the new concept tree differently within the ontology. Further research on the extending process should investigate applying algorithms that address the multiple overgeneralizations problem (Ovchinnikova & Kühnberger, 2007) to better position the domain concept trees, eliminate extra levels in the hierarchy, and even evaluate the consistency of the domain concept trees produced by OntoLT.

Another aspect for future investigation is more experimentation with OntoLT user-specified rules. Default mapping rules included with the OntoLT plug-in were used. More experimentation with user-defined mapping rules might result in relevant and specialized domain concept trees for merging into an ontology. In addition, research (Sabou, 2005) suggests that a domain sublanguage might assist in better parsing of the text collections looking for key words of the domain sublanguage. A domain expert could provide the sublanguage and help develop user-defined mapping rules using the sublanguage.

REFERENCES

Bathija, V. (2006). *An adaptation methodolgy for reusing ontologies.* Masters thesis. Miami University, Oxford, OH.

Bontas, E.P., Mochol, M., & Tolksdorf, R. (2005). Case studies on ontology reuse. *Proc. 5th Int. Conf. Knowledge Management (I'Know '05)*, Graz, Austria.

Brank, J., Grobelnik, M., & Mladenic, D. (2005). A survey of ontology evaluation techniques. *Proc. 8th Int. Multiconf. Information Society IS-2005.*

Buitelaar, P., Cimiano, P., & Magnini, B., Eds. (2005). Ontology learning from text: methods, evaluation and applications. In *Frontiers in Artificial Intelligence and Applications*, Vol. 123. Amsterdam: IOS Press.

Buitelaar, P., Olejnik, D., & Sintek, M. (2004). Protégé plug-in for ontology extraction from text based on linguistic analysis. *Proc. 1st European Semantic Web Symp., The Semantic Web: Research and Applications*, pp. 31–44. Heraklion, Crete.

Burge, J., & Brown, D.C. (2008). Software engineering using RATionale. *Journal of Systems and Software 81(3)*, 395–413.

Burge, J., Cross, V., Kiper, J., Maynard-Zhang, P., & Cornford, S. (2006). Enhanced design checking involving constraints, collaboration, and assumptions. *Proc. Design Computing and Cognition '06*, pp. 655–674. Eindhoven, The Netherlands.

Burge, J.E. (2005). *Software engineering using design RATionale*, PhD dissertation. Worcester Polytechnic Institute.

Caralt, J.C. (2004). Ontology-driven information systems: pruning and refactoring of ontologies. *Proc. 7th Int. Conf. Unified Modeling Language*, Lisbon, Portugal.

Declerck, T. (2002). A set of tools for integrating linguistic and non-linguistic information. *Proc. SAAKM Workshop*, ECAI, Lyon.

Ding, Y., Lonsdale, D.W., Embley, D.W., Hepp, M., & Xu, L. (2007). Generating ontologies via language components and ontology reuse. *Proc. 12th Int. Conf. Applications of Natural Language to Information Systems, NLDB 2007*, pp. 131–142.

Eris, O., Hansen, P.H.K., Mabogunje, A., & Leifer, L. (1999). Toward a pragmatic ontology for product development projects in small teams. *Proc. Int. Conf. Engineering Design*, pp. 1645–1950. Munich: Technische Universität München.

Fowler, D.W., Sleeman, D., Wills, G., Lyon, T., & Knott, D. (2004). The Designers' Workbench: using ontologies and constraints for configuration. *Proc. 24th SGAI Int. Conf. Innovative Techniques and Applications of AI*, pp. 209–221.

Gómez-Pérez, A., & Manzano-Macho, D. (2005). An overview of methods and tools for ontology learning from texts. *Knowledge Engineering Review 19*, 187–212.

Gruber, T.R. (1993). A translation approach to portable ontologies. *Knowledge Acquisition 5*, 199–220.

Haase, P., van Harmelen, F., Huang, Z., Stuckenschmidt, H., & Sure, Y. (2005). A framework for handling inconsistencies in changing ontologies. *Proc. Int. Semantic Web Conf., ISWC 2005*, pp. 353–367.

Jaccard, P. (1908). Nouvelles recherches sur la distribution florale. *Bulletin de la Societe de Vaud des Sciences Naturelles 44*, 223.

Japikse, R., Langdon, P.M., & Wallace, K.M. (2003). Structuring engineering design information using a model of how engineers' intuitively structure design information, *Proc. 14th Int. Conf. Engineering Design, ICED '03*, pp. 433–434.

Jones, K.S. (1972). A statistical interpretation of terms specificity and its application retrieval. *Journal of Documentation 28*, 11–20.

Kietz, J.U., Maedche, A., & Volz, R. (2000). A method for semi-automatic ontology acquisition from a corporate intranet. *EKAW '00 Workshop on Ontologies and Texts, CEUR Workshop Proc.*, pp. 51:4.1–4.14. Accessed at http://CEUR-WS.org/Vol-51/

Kitamura, Y., Kashiwase, M., Fuse, M., & Mizoguchi, R. (2004). Deployment of an ontological framework of functional design knowledge. *Advanced Engineering Informatics 18(2)*, 115–127.

Lee, J. (1997). Design rationale systems: understanding the issues. *IEEE Expert 12(3)*, 78–85.

Lin, J., Fox, M.S., & Bilgic, T.A. (1996). Requirement ontology for engineering design. *Concurrent Engineering: Research and Applications 4(3)*, 279–291.

Lonsdale, D., Ding, Y., Embley, D.W., & Melby, A. (2002). Peppering knowledge sources with SALT; boosting conceptual content for ontology generation. *Proc. AAAI Workshop on Semantic Web Meets Language Resources*, Edmonton, Alberta, Canada.

Maedche, A. (2002). *Ontology Learning for the Semantic Web*. Boston: Kluwer Academic.

Miller, G., Beckwith, R., Fellbaum, C., Gross, D., & Miller, K.J. (1990). Introduction to Wordnet: an on-line lexical database. *International Journal of Lexicography 3(4)*, 235–244.

Navigli, R. (2002). Automatically extending, pruning and trimming general purpose ontologies. *Proc. Int. Conf. Systems, Man and Cybernetics*, pp. 631–635.

Navigli, R., & Velardi, P. (2004). Learning domain ontologies from document warehouses and dedicated Web sites. *Computational Linguistics 30(2)*, 151–179.

Novacek, V., Laera, L., & Handschuh, S. (2007). Semi-automatic integration of learned ontologies into a collaborative framework. *Proc. IWOD/European Semantic Web Conf.*, pp. 13–26, Innsbruck.

Noy, N., & Musen, M.A. (2004). Specifying ontology views by traversal. *Proc. Int. Semantic Web Conf.*, pp. 713–725.

O'Brien, P., & Abidi, S.S.R. (2006). Modeling intelligent ontology evolution using biological evolutionary processes. *IEEE Int. Conf. Engineering of Intelligent Systems*, Islamabad.

Ovchinnikova, E., & Kühnberger, K.-U. (2007). Automatic ontology extension: resolving inconsistencies. In *Ontologies and Text Technology: Approaches to Extract Semantic Knowledge From Syntactic Information* (Mönnich, U., & Kühnberger, K.-U., Eds.), Vol. 2007–1, pp. 93–98. Accessed at http://www.cogsci.uni-osnabrueck.de/cogsci/de/m.ikwPublications.php on March 6, 2009.

Sabou, M. (2005). Learning Web service ontologies: an automatic extraction method and its evaluation. In *Ontology Learning and Population* (Buitelaar, P., Cimiano, P., & Magnini, B., Eds.). Amsterdam: IOS Press.

Song, M.H., Lim, S.Y., Park, S.B., Kang, D.J., & Lee, S.J. (2005). Ontology-based automatic classification of Web pages. *International Journal of Lateral Computing 1(1)*, 57–62.

Spyns, P., & Reinberger, M. (2005). Lexically evaluating ontology triples generated automatically from texts. *Proc. 2nd European Semantic Web Conf., Semantic Web: Research and Applications*, pp. 563–577, Heraklion, Crete, May 29–June 1.

Stuckenschmidt, H., & Klein, M. (2004). Structure-based partitioning of large concept hierarchies. *Proc. 3rd Int. Semantic Web Conf.*, pp. 289–303.
Volz, R., Studer, R., Maedche, A., & Lauser, B. (2003). Pruning-based identification of domain ontologies. *Journal of Universal Computer Science 9*, 520–529.
Zipf, G.K. (1949). *Human Behaviors and the Principle of Least Effort*. Cambridge, MA: Addison–Wesley.

Valerie Cross is an Associate Professor in the Computer Science and Systems Analysis Department at Miami University in Oxford, OH. She earned a BS in computer science and a BS in statistics from West Virginia University, an MS in computer science at the University of Colorado, Boulder, and a PhD in computer science from Wright State University. Dr. Cross is currently Secretary of the North American Fuzzy Information Processing Society and a member of the Women in Computational Intelligence Committee of the IEEE Computational Intelligence Society. Her research interests include fuzzy set theory, ontology learning, evaluation, and visualization.

Vishal Bathija is a Business Analyst at Planetsoft Holdings, which provides software solutions to streamline the operations of insurance carriers. He is responsible for gathering business requirements and translating them into functional product specifications, evaluating market needs, and forming the product roadmap. Before joining Planetsoft in 2008, Vishal worked for a diverse group of organizations including Zeo Inc., Cardinal Health, and TATA Consultancy Services. For his master's thesis in computer science at Miami University, he designed an adaptation architecture for reusing ontologies through Protégé, an open source ontology editor.

https://doi.org/10.1017/S0890060409000183 Published online by Cambridge University Press