Proceedings of the 13th International Conference on Knowledge Engineering and Knowledge Management (EKAW02), 1-4 October 2002 - Sigüenza (Spain)

IKRAFT: Interactive Knowledge Representation and Acquisition From Text

Yolanda Gil and Varun Ratnakar

USC Information Sciences Institute
4676 Admiralty Way
Marina del Rey, CA 90292
gil@isi.edu, varunr@isi.edu

Abstract. We propose a new approach to develop knowledge bases that captures at different levels of formality and specificity how each piece of knowledge in the system was derived from original sources, which are often Web sources. If a knowledge base contains a trace of information about how each piece of knowledge was defined, it will be easier to reuse, extend, and translate the contents of the knowledge base. We are investigating these issues with IKRAFT, an interactive tool to elicit from users the rationale for choices and decisions as they analyze information used in building a knowledge base. Starting from raw information sources, most of them originating on the Web, users are able to specify connections between selected portions of those sources. These connections are initially very high level and informal, and our ultimate goal is to develop a system that will help users to formalize them further.

1 Introduction

Large knowledge bases contain a wealth of information, and yet browsing through them often leaves an uneasy feeling that one has to take the developer's word for why certain things are represented in certain ways, why other things were not represented at all, and where might we find a piece of related information that we know is related under some context. Although the languages that we use are quite expressive (e.g., KIF, MELD), they still force knowledge into a straightjacket: whatever fits the language will be represented and anything else will be left out. Many other things are also left out, but for other reasons such as available time and resources or perhaps lack of detailed understanding of some aspects of the knowledge being specified. The challenges that arise in understanding, reusing, extending, translating, and merging knowledge bases [Burstein et al 00.; Chalupsky 00; McGuinness] may be due in no small part to the impoverished products that we create as knowledge base developers. When the knowledge base needs to be extended or updated, the rationale for their design is lost and needs to be at least partially reconstructed. The knowledge sources are no longer readily available and may need to be accessed. While it is the case that entire knowledge bases can be reused and incorporated into new systems, it is harder to extract only relevant portions of them that are appropriate in the new application. Parts of the knowledge base may be too inaccurate for the new task, or may need to be

modeled in a different way to take into account relevant aspects of the new application.

We believe that knowledge bases should contain the fine-grained, detailed analysis, assumptions, and decisions that their developers made during their design. They should record in enough detail what were the original sources consulted, what pieces seemed contradictory or vague, which were then dismissed, what additional hypotheses were formulated in order to complement the original sources. Knowledge engineers have a sense for what topics or areas within the knowledge base they are more confident about, either because they spent more resources developing them, because they found better sources, or because as knowledge engineers had assessed the end result as more complete and consistent.

This information turns out to be also key to put the answers of a knowledge-based system in context. Consider, for example, our experiences with a system to estimate the duration of carrying out specific engineering tasks, such as repairing a damaged road or leveling uneven terrain. Users invariably wanted us to explain where the answers came from in terms of the sources we consulted and the sources that we chose to pursue when they suggested alternative models. They wanted to know whether common manuals and/or sources of expertise were consulted, which were given more weight, whether practical experience was considered to refine theoretical estimates, and what authoritative sources were consulted to design the content of the knowledge base. In other words, the analysis process that knowledge engineers perform during the implementation phase is part of the rationale of a knowledge base, and needs to be captured in order to justify answers to users.

Our goal is to capture the results of analyzing various information sources consulted by knowledge engineers as they design the detailed contents of a knowledge base. This paper presents our work to date on IKRAFT, a tool that enables knowledge base developers to keep track of the knowledge sources and intermediate knowledge fragments that result in a formalized piece of knowledge. The resulting knowledge base is enhanced with pointers that capture the rationale of its development. There are several other potential benefits to including this rationale within a knowledge base, such as supporting its maintenance, facilitating its integration with other knowledge bases, and transferring (and translating) knowledge among heterogeneous systems.

The paper begins motivating our work with a description of how knowledge bases are built. We then describe the approach we are taking in IKRAFT, and show with several examples how users can keep a trail of sources and intermediate knowledge fragments to support each item in the knowledge base. We finalize with a discussion of the implications of the approach and conclude with our plans for future work.

2 Creating a Knowledge Base

In order to illustrate why it is important to capture how each individual piece of knowledge in the knowledge base came about, we use a brief example taken from previous work on building a knowledge base from a chapter of a biology textbook. We describe the process of formalizing knowledge with three steps. These steps can be found in typical descriptions of knowledge engineering [Buchanan et al. 83; Stefik 95; Schreiber et al. 00]. Our description sets aside other steps concerning knowledge

base development, such as feasibility studies, integration within the organization, refinement and maintenance. Our focus here is on the steps that involve a single developer set to the task of creating formal definitions given a set of sources that have been previously compiled (through interviews, literature research, or other consultations). We also assume the developer is not following any particular knowledge engineering methodology (such as CommonKADS).

STEP 1: Selecting relevant knowledge fragments

"…The first step a cell takes in reading out part of its genetic instructions is to *copy the required portion of the nucleotide sequence of DNA – the gene – into a nucleotide sequence of RNA. The process is called transcription* because the information, though copied into another chemical form, is still written in essentially the same language – the language of nucleotides. Like DNA, RNA is a linear polymer made of four different types of nucleotides subunits linked together by phosphodiester bonds. It differs from DNA chemically in two respects: (1) the *nucleotides in RNA are ribonucleotides* – that is, they contain the sugar ribose (hence the name ribonucleic acid) rather than deoxyribose; (2) although, like DNA, *RNA* contains the bases adenine (A), guanine (G), and cytosine (C), it *contains uracil (U) instead of the thymine (T)* in DNA. Since U, like T, can base-pair by hydrogen-bonding with A, the base-pairing properties described for DNA also apply to RNA…"

STEP 2: Composing stylized knowledge fragments

- ribose
  - it is a kind of sugar, like deoxyribose
  - it is contained in the nucleotides of RNA
- uracil
  - it is a kind of nucleotide, like adenine and guanine
  - it can base-pair with adenine
- RNA
  - it is a kind of nucleic acid, like DNA
  - it contains uracil instead of thymine
  - it is single-stranded
  - it folds in complex 3-D shapes
  - nucleotides are linked with phospohodiester bonds, like DNA
  - there are many types of RNA
  - RNA is the template for synthesizing protein
- gene : - subsequence of DNA that can be used as a template to create protein
- protein synthesis
  - non-destructive creation process: RNA and protein created from DNA
  - its speed is regulated by the cell
  - substeps: (ordered in sequence)
    1) RNA transcription
       - a DNA fragment (a gene) is copied, just like DNA is copied during DNA synthesis
       - the result is an RNA chain
    2) protein translation
       - RNA is used as a template

STEP 3: Creating knowledge base items

… (defconcept uracil :is-primitive nucleotide :constraints (:the base-pair adenine))
(defconcept RNA :is (:and nucleic-acid (:some contains uracil))) …

Fig. 1. Steps in Creating a Knowledge Base

The steps are illustrated with an example in Figure 1, Here the developer is trying to extract and represent a description of the protein synthesis process through its substeps and the entities that participate in that process.

In the first step, the developer selects original sources (in this case only one is shown, but typically there would be several) and selects from them relevant knowledge fragments. Source text will typically contain the relevant information embedded within details that may be irrelevant to the developer or commentary from the author. In the second step, the developer restates the knowledge fragments in terse English. Typically these new fragments are phrased as unambiguously and briefly as possible. They may be organized in a list of items and sub-items. The developer may combine two or more fragments into one sentence, or break a fragment into several sentences. This step is akin to making a summary when studying for an exam. The developer will often go back to step one to gather more documentation and knowledge fragments as he or she makes sense of the fragments listed. Step two can be done in several iterations, each iteration containing more stylized fragments.

Finally, the third step involves formalizing those fragments into the target language and syntax. Notice that some of the fragments extend existing definitions that are assumed to be already known, and as a result their formalization needs to take into account existing definitions.

Notice that the final formalization of the knowledge does not necessarily contain all the information in the original knowledge fragments selected or in the restated fragments. The developer may decide to formalize only those portions, or perhaps the formal language was not expressive enough to represent certain aspects of the knowledge.

Looking at the figure, it is easy to see where each assertion in the formal definition comes from, what portions of the stylized fragments are formalized, and where in the initial sources the information came from. Unfortunately, the knowledge fragments selected in step one and those composed in step two shown in the figure are never captured in the knowledge base, only the resulting formal definitions included are.

We believe that knowledge bases should include this information, i.e., that the final formalized knowledge items should point back to previous knowledge fragments considered by the developer, and ultimately to the source documents where the knowledge was drawn from. There are many benefits to this approach:

- **Knowledge can be extended more easily.** The formalized, final expressions may not necessarily contain every detail in every knowledge source, but if the need arises the developer is better positioned to track down additional knowledge missing. One could even consider natural language processing tools that would enable the system could use some automated tools to extract that knowledge itself, since it has access to the sources and to stylized fragments that are likely to contain information in the boundaries of the knowledge that was formalized the first time around. Today's knowledge bases are best (more efficiently) extended by their developers.
- **Knowledge can be reused at any level of formality**. Reusing and translating today's knowledge bases means reusing and translating expressions in different formal languages, which can be challenging [McDermott 01; Chalupsky 00]. Here, intermediate knowledge fragments can be reused and formalized in a new language without having to go through the original developer's language as an intermediate step. Moreover, during reuse of intermediate knowledge fragments

can be further detailed and extended incorporating other sources to create different final formalized expressions.

- **Knowledge can be integrated and translated at any level to facilitate interoperability**. Translation is often used to enable integration and interoperation among intelligent tools. Today's translation tools have to deal with the different levels of expressivity and modeling styles of the source and target systems. One can envision developing translators that operate (or are supported by) the stylized knowledge fragments, either automatically or semi-automatically depending on the difficulty of the expressions used by the developer. Also, symbols would be annotated with their intended meaning, which is key when two systems may be using the same term differently. The rationale and meaning of different pieces of knowledge can be available to support translation and interoperation.
- **Intelligent systems will be able to provide better justifications**. We find that many users are reluctant to accept the solutions presented by the systems and ask for explanations not of how the system derived an answer automatically but instead ask for explanations of why the system starts out with a certain fact or belief. When users are shown the reasons for certain assumptions and the fact that certain sources were consulted to make that assumption they are reassured in the competence of the system to provide those answers. Capturing this trail within the knowledge base will enable the system to generate these kinds of justifications and explanations.
- **Content providers will not need to be knowledge engineers**. Although only those trained in the art of designing, modeling, and writing formal expressions can write the final formal knowledge items, anyone can contribute to the initial steps of the process. Many people in diverse disciplines acquire the analytical skills that suffice to organize and digest knowledge sources. The intermediate knowledge fragments shown in Figure 2 were not created by a knowledge engineer, one could argue that they may be more reusable than those shown in Figure 1 that were created by a knowledge engineer. In fact, if the knowledge base is in their area of expertise, they are likely do a much better job at re-expressing knowledge items than knowledge engineers. This would make knowledge base creation a true collaboration between domain experts and knowledge engineers where each is contributing at the stages of the process where they have relevant skills.

Protein : - unique amino acid sequence. - this sequence provides it a unique structure.
DNA : - a sequence of 4 types of nucleotides, linked by phosphodiester bonds.
- stores genetic information. - double stranded helix
RNA : - sequence of 4 types of nucleotides linked by phosphodiester bonds,
- like DNA. - short copies of nucleotide sequence of the DNA.
- passes genetic information.
Transcription : - Part of DNA is copied to the 'RNA
Translation: - Nucleotide sequence of the RNA generates the protein

Fig. 2. Knowledge Fragments that were not created by a knowledge engineer.

3 Overview of IKRAFT

IKRAFT allows users to create new items in the knowledge base from multiple sources, while keeping track of the knowledge sources and intermediate knowledge fragments that were used in deriving the new item.

After collecting enough relevant sources, the user can now make statements to summarize salient parts of the sources with the Statement Editor. This is easily accomplished by highlighting parts of the source (or sources) and summarizing those parts in a statement. Later, when the statement is clicked, the parts in the sources where this information was gathered from are highlighted.

Each statement is parsed by an NLP tagger, which identifies nouns and verbs in the sentence. The nouns are referred as objects, and verbs as events. These objects and events are checked with those in the database. The user is then shown with a list of objects and events which do not occur in the database. They can be defined by the user with the Object Editor. The user is also shown the list of objects which do occur in the database. If the definitions in the database are not what the user wants, then another definition for the same object can be made by the user.

Also, when the user clicks on an object, the statement where the word came from is highlighted. This is useful in finding the context in which the word is used.

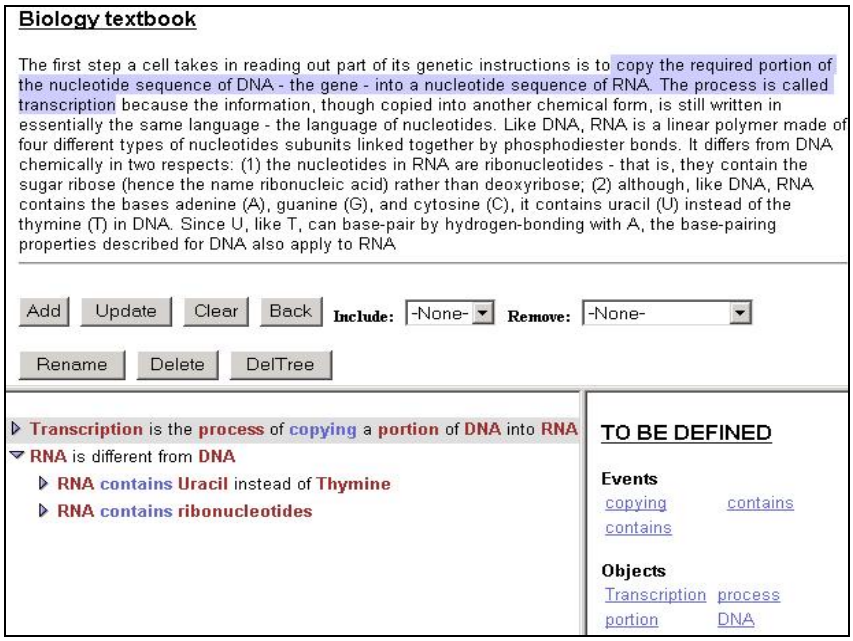

Fig. 3. IKRAFT allows a user to keep track of knowledge sources and intermediate knowledge fragments used in creating a new item in the knowledge base.

Figure 3 shows how the trail of knowledge sources and fragments are captured by IKRAFT as a user represented the scenario described in Figure 1. The knowledge fragments are represented as a collapsible tree structure of statements in the bottom left frame. The objects that have to be defined, and those that are already defined (not visible) are shown in the bottom right frame.

Our current prototype implementation addresses steps 1 and 2 in Fig 1. Supporting the formalization stage (step 3 in Fig 1) will be addressed by future work.

4 Using IKRAFT

In this section, we show an example created using IKRAFT. It is summarized in Figures 4 and 5.

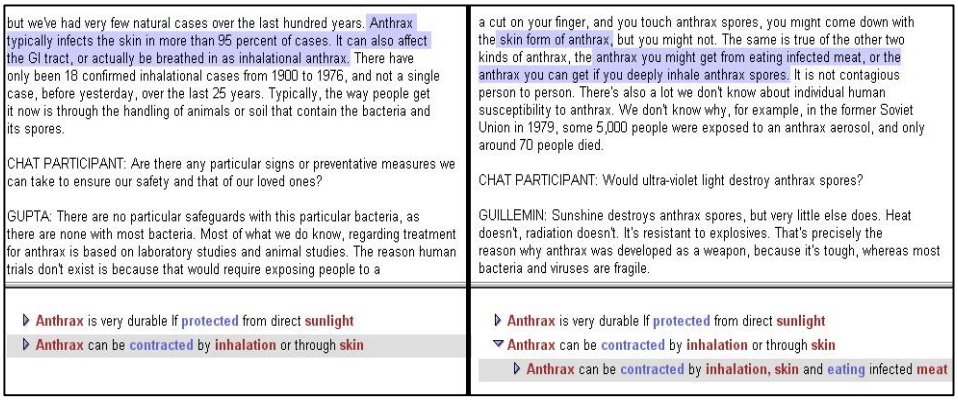

Fig. 4. Demonstrating supplementary information from different sources

This example displays a user trying to find out more about Anthrax. The user has three different sources of information. The first one is from a news article, and the other two are from interviews of experts in the area. In Figure 4, the user summarizes the information from one of the interviews to conclude that Anthrax can be contracted by inhalation and through skin contact. Later, he finds supplementary information to this in the other interview, and concludes that it can also be contracted through ingesting infected meat.

Figure 5 shows the user finding out about the durability of Anthrax spores. The newspaper article provides the information that Anthrax spores are extremely durable as long as there is no sunlight. However it is not clear if the spores are destroyed in sunlight or not. This is made clear by one of the other sources that they are indeed destroyed by sunlight.

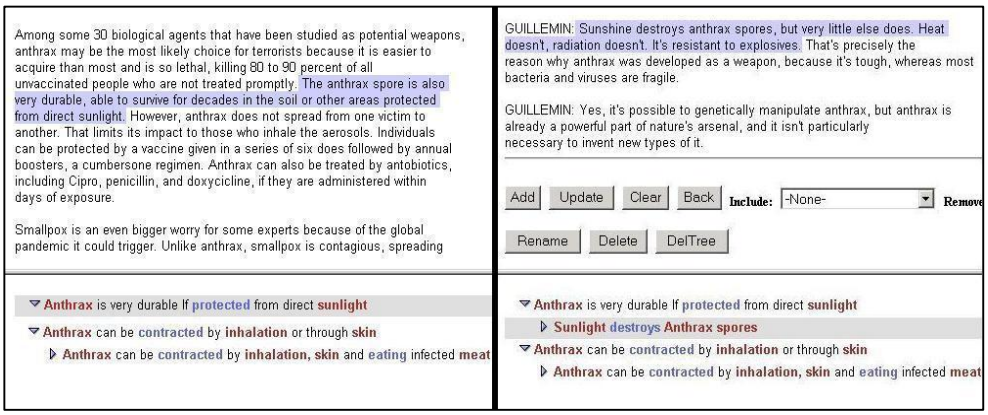

Fig. 5. Demonstrating clarification

5 Discussion

Our current tools can be enhanced with other work on natural language processing and knowledge base analysis. Many existing tools for text extraction (e.g, to extract significant event descriptions from news articles) and discourse analysis (e.g., to segment text into meaningful portions) could be used to support these earlier steps of the analysis [Croft 99; Cowie and Lehnert 96; Radey and McKeown 98]. Existing approaches to derive interdependencies among pieces of knowledge may be used to help users create connections among diverse pieces of knowledge [Kim and Gil]. Other tools can be developed to support transformations at the lower levels (e.g., to turn tables into instances and role values).

The approach presented here has many relations to software engineering methodologies to capture the rationale for knowledge-based development [Schreiber et al 00], and to higher-level languages and frameworks to develop knowledge-based systems [Fensel et al. 98]. However, these methodologies are often aimed at software and knowledge engineers and are not very accessible to other potential knowledge base developers, such as end users and/or domain experts.

The overhead that may be incurred in creating knowledge bases using this approach is, in our view, not as significant compared to the analysis efforts that developers already undergo. It may even save developers time if others can look up the rationale trail instead of asking them directly detailed questions about the portion of the knowledge base they are developing. As in any issue that creates overhead during development, it depends on the motivation and ultimately payoff to the developers in terms of facilitating maintenance and reuse by others.

The Semantic Web [Berners-Lee et al 01] will provide an ideal substrate to ground knowledge bases into their original knowledge sources, and to contain the progressively defined pieces of knowledge and the connections among them. More and more every day, knowledge originates and ends in the Web, and we find ourselves

extracting knowledge from the Web, processing it inside of a knowledge base, then putting the results back on the Web. It only makes sense to integrate knowledge bases (their content and their reasoning) more closely with the Web. Currently, IKRAFT represents the links to the sources and knowledge fragments in its own language. In future work we plan to turn IKRAFT into a Web-based annotation tool, where these pointers would be converted into annotations in a suitable markup language. If the final knowledge base is published as a Web resource, it will be linked to other Web resources that represent the original documents, as well as intermediate knowledge fragments that can be turned into Web resources as well. Using IKRAFT may perhaps allow web users that are not AI experts to contribute to knowledge base development, at least in the initial stages as presented in steps 1 and 2 in Fig 1 in this paper. Others who are more savvy about knowledge representation techniques may take on the subsequent formalization stages (step 3 in Fig 1). Many knowledge bases will finally be open source, and one can only hope they will be adopted, extended, and used by much larger numbers of people than they are today.

6 Conclusions

Knowledge base developers may consult many sources presenting contradictory or complementary information, analyze the different implications of each alternative belief, and decide what and how to model the knowledge. In essence, developers often capture in the knowledge base only their final beliefs about some body of knowledge. The rationale for modeling the knowledge the way it appears in the knowledge base is not captured declaratively. Only consistent and complete information is captured. No indication of inconsistent but possible statements is added to the knowledge base.

In ongoing work, we are developing a knowledge base using IKRAFT to represent terms in the geosciences domain. We plan to measure whether the rationale capture by IKRAFT is useful by measuring how users access the rationale from the application as they try to find out how and why terms were defined as they appear in the final formal representation. We also plan to use IKRAFT to develop a knowledge base using two kinds of users: a user that is an experienced domain expert and that performs the first two steps outlined above, and a user that is a knowledge engineer and formalizes the statements input by the first user. An interesting possibility would be to explore how our framework would support collaborative knowledge base development by larger groups of users, both experts and knowledge engineers.

Intelligent systems should be able to access the roots and rationale of the knowledge they contain. This is the approach that we have taken in developing IKRAFT, a tool to allow users to link knowledge bases to their original sources and other knowledge fragments that result from the analysis of the knowledge base developer. This approach would create a new generation of knowledge bases that will be more amenable to updates, reuse, migration, and interoperation.

References

1. Berners-Lee, T.; Hendler, J.; and Lassila, O. 2001. The Semantic Web. Scientific American 78(3):20–88.
2. Burstein, M., McDermott, D., Smith, D. R., Westfold, S. 2000. "Derivation of Glue Code for Agent Interoperation". Proceedings of the International Conference on Autonomous Agents 2000. Barcelona, Spain.
3. Chalupsky, H. 2000. "OntoMorph: A Translation System for Symbolic Knowledge". Proceedings of the International Conference on Knowledge Representation and Reasoning, KR-2000, Breckenridge, CO.
4. Cooke, N. J. 1994. “Varieties of Knowledge Elicitation Techniques”, International Journal of Human-Computer Studies, Vol. 41.
5. Cowie, J. and Lehnert, W. 1996. "Information Extraction". Communications of the ACM, 39(1):80--91, Jan 1996.
6. Croft, W.B. 1999. "Combining Approaches to Information Retrieval," in Advances in Information Retrieval: Recent Research from the CIIR, W. Bruce Croft, Ed. Kluwer.
7. Fensel, D., Angele, J., and Studer, R. 1998. "The Knowledge Acquisition and Representation Language KARL", Knowledge and Data Engineering, 10 (4).
8. Kim J., and Gil, Y. 2000. "Acquiring Problem-Solving Knowledge from End Users: Putting Interdependency Models to the Test." Proceedings of the Fifteenth National Conference on Artificial Intelligence (AAAI-2000), Austin, TX.
9. McGuinness, D. L., Fikes, R., Rice, J., and Wilder, S. 2000. "An Environment for Merging and Testing Large Ontologies". Proceedings of KR-2000, Breckenridge, CO.
10. Radev, D. and McKeown, K. 1998. "Generating natural language summaries from multiple online sources". Computational Linguistics, 1998.
11. Schreiber, G, Akkermans, H., Anjewierden, A., de Hoog, R., Shadbolt, N., Van de Velde, W., and Wielinga, B. 2000. "Knowledge Engineering and Management: The CommonKADS Methodology". MIT Press.
12. Shum, S.B. 1996. Design Argumentation as Design Rationale. Encyclop- edia of Computer Science and Technology (M.Dekker Inc: NY).
13. Stefik, M., 1995. "Introduction to Knowledge Systems". Morgan Kaufmann.
14. Swan, R. and Jensen, D. 2000. “TimeMines: Constructing Timelines with Statistical Models of World Usage”, Proceedings of KDD-2000