Does Prior Knowledge Facilitate the Development of Knowledge-based Systems?

Paul Cohen,	Vinay Chaudhri,	Adam Pease,	Robert Schrag
Cohen@cs.umass.edu	chaudhri@ai.sri.com	apease@teknowledge.com	schrag@dc.iet.com
U. of Massachusetts	SRI International	Teknowledge .	IET Inc.

Contact author:
Paul Cohen, Department of Computer Science, Lederle GRC, University of Massachusetts, Amherst MA 01003
Phone: 413 545 3638; fax: 413 545 1249

Abstract: One factor that affects the rate of knowledge base construction is the availability and reuse of *prior knowledge* in ontologies and domain-specific knowledge bases. This paper reports an empirical study of reuse performed in the first year of the High Performance Knowledge Bases (HPKB) initiative. The study shows that some kinds of prior knowledge help more than others, and that several factors affect how much use is made of the knowledge.

Tracking number A261

Content areas: knowledge-representation, design-and-evaluation-of-ontologies

Statement of sole submission: "The author(s) certifies that this paper has not been accepted by and is not currently under review for another AI or AI subarea conference, nor will it be submitted for such during the AAAI-99 review period. The author also certifies that any version of this paper submitted to another non-AI conference or journal contains significant material not included in the AAAI submission, or vice versa."

Does Prior Knowledge Facilitate the Development of Knowledge-based Systems?

Tracking Number: A261

Abstract

One factor that affects the rate of knowledge base construction is the availability and reuse of *prior knowledge* in ontologies and domain-specific knowledge bases. This paper reports an empirical study of reuse performed in the first year of the High Performance Knowledge Bases (HPKB) initiative. The study shows that some kinds of prior knowledge help more than others, and that several factors affect how much use is made of the knowledge.

Introduction

One hypothesis of recent knowledge sharing efforts has been that significant productivity gains can be realized by reusing prior knowledge in ontologies and domain-specific knowledge bases (Patil et al. 1992). Until now, there have been no systematic studies of knowledge reuse. This paper reports an empirical study performed in the first year of the High Performance Knowledge Bases (HPKB) initiative sponsored by the Defense Advanced Research Projects Agency (Cohen et al., 1998)[1]. By comparing the efforts of two HPKB groups under different conditions, we find that prior knowledge in the form of ontologies does help, though many factors affect how much it helps. This work also introduces metrics and methods for evaluating the contribution of prior knowledge to knowledge-based systems.

By *prior knowledge* we mean the knowledge one has available in an ontology or knowledge base prior to developing a knowledge-based system. Several large ontologies have been developed including Cyc (Lenat,1995), Sensus (Knight, 1994), Ontolingua (Farquhar, 1996). All these systems contain hierarchies of knowledge. At the upper levels, one finds knowledge that is general to many applications, such as knowledge about movement, animate agents, space, causality, mental states, and so on. The lower levels contain knowledge specific to domains; for example, rules for inferring the effects of tactical military operations. Bridging general and specific knowledge, one finds middle-level knowledge (Lenat and Guha, 1990); collections of terms and axioms about phenomena such as human physiology, more general than a particular medical expert system but less general than, say, knowledge about physical systems. In addition to hierarchies of terms, all the ontologies cited above contain *axioms* or *rules*, for instance, "if x is an educational institution then x pays no taxes"; and inference methods such as resolution or more specialized forms of theorem-proving. Axioms and rules confer a functional kind of *meaning* on the terms they contain, that is, the meaning of a term is the things one can legitimately say (infer) about it.

One claim of ontologists is that it is easier to build a domain-specific knowledge base **KB** inside an ontology **O**, or informed by **O**, than without **O**. Some of the ways that **O** can help are illustrated in Figure 1. First, a term **p** that you wish to add to **KB** might already exist in **O**, saving you the trouble of adding it. Second, axioms or rules relating to **p** might already exist in **O**, saving you the trouble of thinking them up and encoding them. Third, within **O**, **p** might be a subclass of **v**, so you also have the benefit of axioms about **v** inherited through **p**.

Now suppose you want to add a concept **p'** to **KB**, and **p'** is not exactly **p**, but is similar in some respects. For instance, **p** might be part of a microtheory about economics, and **p'** might belong to a microtheory about fluid flows, but both **p** and **p'** represent the concept "source." More generally, suppose the *structure* of the theory of economics in **O** parallels the structure of the theory of fluids that you are trying to build in **KB**. Thus, a fourth way that **O** can help you to build **KB** is to help you structure the theory in **KB**. Designing the structure of microtheories is very time consuming, so this kind of help may be the most important of all.

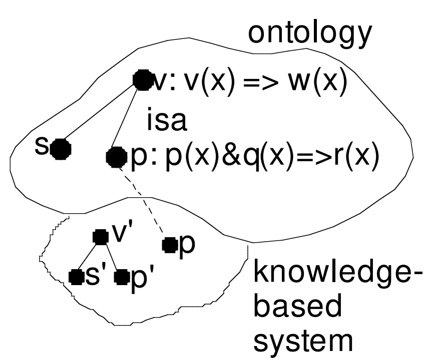

Figure 1. Some ways an ontology O can help one build a knowledge base KB.

Unfortunately it is difficult to assess experimentally how the structure of **O** helps one build **KB**s with similar

[1] See http://www.teknowledge.com/HPKB for a collection of documents about HPKB including the Challenge Problem specifications.

structure, so we focus here on the first three ways that **O** can help one build **KB**.

A Metric

Suppose one wishes to add an axiom, "If x is a state then x maintains an army," to **KB**. This axiom contains three terms, **nation**, **maintains**, and **army**. Suppose the first two terms already exist in **O** but **army** does not. As two thirds of the terms required to add the axiom to **KB** exist in **O,** we say the *support* provided by **O** in this case is 2/3. In general, every item i one wishes to add to **KB** contains n(i) terms, k(i) of which are already in **O,** and support is s(i)=k(i)/n(i). Of course, adding **army** to **O** changes **O**, and the support offered by **O** for future axioms might be higher because **army** was added. Therefore, support is indexed by versions of the ontology: s(i,j)=k(i,j)/n(i) is the support provided by version **O**j of the ontology for concept i.

Experiment Design

We evaluated the support provided by ontologies during a month-long process called the *Crisis Management Challenge Problem* (CMCP). The CMCP was designed by a team led by IET, Inc. and PSR Corp. Two integrated knowledge-based systems were developed to answer questions about international crises, such as, "What will the US response be if Iran closes the Strait of Hormuz?" (Cohen et al., 1998). The systems were developed by teams led by Teknowledge and SAIC. The CMCP had several phases:

1. Some months before any testing began, a *crisis scenario* was released. The scenario bounded the domain and thus the scope of the problems to be solved by the Teknowledge and SAIC systems.
2. Several weeks before testing, a batch of sample questions (SQs) was released.
3. On the first day of the evaluation, a batch of 110 test questions, TQA, was released, and the Teknowledge and SAIC systems were immediately tested. After four days for improvements, the systems were re-tested on TQA.
4. Batch TQB was released at the time of the retest. The purpose of TQB, which contained questions similar to those in TQA, was to check the generality of the improvements made to the systems.
5. After a brief respite, a change was made to the crisis scenario, increasing the scope of the problems that the Teknowledge and SAIC systems would have to solve. Several days were allowed for knowledge entry prior to the release of a new batch of questions, TQC, reflecting the new scope. The systems were tested immediately.
6. Four days were allowed to extend the systems to the new crisis scenario, then the systems were re-tested on TQC. To check the generality of these extensions, the systems were also tested on batch TQD, which was similar to TQC.

One of the methodological innovations of this experiment was to generate all the batches of questions from a *question grammar* – a set of parameterized questions, or PQs – which had been made available to the participants in the experiment several months before testing began. Batches SQ, TQA and TQB were generated by one grammar. The grammar was extended to reflect the change in the crisis scenario and used to generate batches TQC and TQD. Figure 2 shows one of the parameterized questions (PQ53) from the grammar. Many billions of questions could be generated by the question grammar, so it would not have made sense to develop systems to solve particular questions; however, by getting the PQs early, the system developers could limit the scope of their systems to the subjects mentioned in the PQs (e.g., terrorist attacks, EconomicSector, etc.)

PQ53 [During/After <TimeInterval>,] what {risks, rewards} would <*InternationalAgent*> face in <*InternationalActionType*>?
<*InternationalActionType*> =
{[exposure of its] {supporting, sponsoring} <InternationalAgentType in <InternationalAgent2>, successful terrorist attacks against <InternationalAgent2>'s <EconomicSector>, <InternationalActionType>, taking hostage citizens of <InternationalAgent2>, attacking targets <SpatialRelationship> <InternationalAgent2> with <Force>}
<*InternationalAgentType*> =
{terrorist group, dissident group, political party, humanitarian organization}

Figure 3. A parameterized question suitable for generating sample questions and test questions.

In the following section we analyze how prior ontology – what was available before SQs, TQA and TQC were released – supported the development of the Teknowledge and SAIC systems. The former system was based on Cyc, and much of its development was done at Cycorp, so we call it Cyc/Tek here. The SAIC system was a collection of component systems, none of which answered all the questions in any test batch. The one we analyze here, developed by SRI International, answered roughly 40 of the 110 questions in each batch; we lack data for the other components of the SAIC system. To compare the Cyc/Tek and SRI systems properly we will report two sets of results for Cyc/Tek, one for all the test questions and another for the subset of questions answered by the SRI system.

The Cyc/Tek and SRI systems also differed in the prior ontologies available to them. Long before testing began, Cycorp, the developers of Cyc, released their *upper ontology* (UO), which contains very general class names; subclass relationships; instance-type relationships; relation names and their argument types; function names, their argument types, and the types of value they return; as well as English documentation of every class, function and relation; and a mapping to terms in the Sensus ontology developed by ISI. Whereas the SRI team had access to the UO, only, Cyc/Tek had access to all of Cyc.

Results

The performance of the Teknowledge and SAIC integrated systems is analyzed thoroughly in (Cohen et al., 1998). Performance is not the focus of this paper – support provided by ontologies is – but two performance results set some context for the following discussion of support and reuse: Both systems performed better on the sample questions (SQs) than on TQA, and both performed better when re-tested on TQA and TQC than on the original tests performed four days earlier. In the four days between test and retest, significant improvements were made to the systems. The question is, how much did the prior ontologies help in making these improvements?

We present results for two kinds of knowledge development. One is the development of knowledge sufficient to encode in a formal language the test questions in each batch, the other is the development of knowledge to answer the test questions. Results for the former are summarized in Table 1. The columns of the table represent the SRI system, which was tested on roughly 40 questions in each batch of 110; the Cyc/Tek system tested on the same questions as the SRI system; and the Cyc/Tek system tested on all 110 questions in each batch. Three numbers are reported for each system: n is the number of terms needed to encode all the questions attempted (i.e., roughly 40 or 110); k is the number of terms available in a prior ontology; and s is the ratio of k to n. The rows of Table 1 represent the batches of questions and the help provided by different prior ontologies. For example, the notation SQ | UO means "the help provided by the upper ontology (UO) in encoding the sample questions (SQ)." One can see in this row that SRI needed 104 terms to encode roughly 40 of the sample questions, and 22 of these terms were found in the UO, so the help provided by the UO is 22/104 =.21. Encoding the questions in SQ required a number of terms to be added to the ontologies, and these terms were available to help encode questions in TQA and TQC. The notation TQA | UO denotes the help provided by the UO *only*, whereas TQA | SQ denotes the help provided by *everything encoded up through* SQ. Similarly, TQC | TQA denotes the help provided in encoding the questions in TQC by the terms in the ontology including those defined for SQ and TQA. For the Cyc/Tek system, our data support only a simpler distinction, between UO terms and non-UO terms, the latter category including the entire Cyc ontology and all terms defined while encoding the test questions. The category of non-UO terms is reported in rows labeled "Cyc" in Table 1. For instance, 418 terms were required by Cyc/Tek to encode the 110 questions in TQA, 402 of them were available in Cyc, including some defined when the sample questions SQ were added. Note that SRI used only the public release of the upper ontology, so all rows in which questions were encoded with the help of Cyc are marked n/a for SRI.

	SRI			Cyc/Tek(40)			Cyc/Tek(110)		
	n	k	s	n	k	s	n	k	s
SQ | UO	104	22	.21	201	75	.37	377	126	.33
SQ | Cyc	n/a	n/a	n/a	201	153	.76	377	280	.74
TQA | UO	104	20	.19	168	67	.4	418	126	.30
TQA | SQ	104	81	.78	n/a	n/a	n/a	n/a	n/a	n/a
TQA | Cyc	n/a	n/a	n/a	168	168	1.0	418	402	.96
TQC | UO	106	16	.15	277	81	.29	402	131	.33
TQC | TQA	106	82	.77	n/a	n/a	n/a	n/a	n/a	n/a
TQC | Cyc	n/a	n/a	n/a	277	270	.97	402	395	.98

Table 1. Support (s) provided by ontologies for the development of problem solving systems to answer batches of test questions.

The six reuse rates from Table 1, are presented graphically in Figure 4. Reuse from the UO on all test question batches clusters in the lower half of the graph. The highest levels of reuse from the UO are achieved by Cyc/Tek on the roughly 40 test questions encoded by SRI. The upper half of the graph represents reuse from the UO *and* all of Cyc in the Cyc/Tek conditions; and reuse of terms defined for earlier test question batches, in the SRI condition.

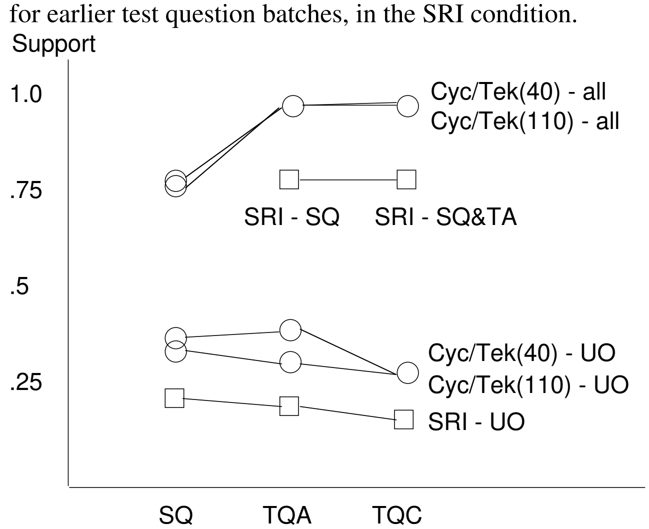

Figure 4. Support rates for SRI and Cyc/Tek. Lines denoted "UO" represent reuse of terms from the upper ontology. SRI-SQ denotes SRI's reuse of terms from the UO and the SQ-encoding effort; SRI-SQ&TA adds in terms defined during the TA-encoding effort. Cyc/Tek(40)-all and Cyc/Tek(110)-all denote reuse of terms from all of Cyc.

Cyc/Tek had higher support numbers in all conditions than SRI, meaning they found more terms in their prior ontologies than SRI did. However, we have broken the data into support provided to Cyc/Tek by *all* of Cyc vs.

support provided by just the upper ontology, which SRI had. For example, the first row of Table 1 shows that to encode roughly 40 sample questions, SRI required 104 terms of which it found 22 in the UO; whereas Cyc/Tek required 201 terms to encode the *same* questions, and found 75 in the UO. Similarly, Cyc/Tek required 377 terms to encode all 110 sample questions, and found 126 in the UO.

Cyc/Tek required more terms to encode test questions (3.62 terms/question) than SRI (2.61 terms/question, and got more support from prior ontologies. For example, for Cyc/Tek to encode the roughly 40 questions in the TQA batch that SRI encoded, they required 168 terms, all of which existed in the Cyc ontology.

In one respect, the SRI and Cyc/Tek results are very similar. The reuse rate of terms *not* in the upper ontology – terms in Cyc or terms developed for earlier batches of test questions – was 60%-65% for both SRI and Cyc/Tek, across question batches TQA and TQC. This result is shown in Table 2. The columns in this table represent the number of terms needed to encode a test batch, N; the number found in the upper ontology, K(UO); the number found elsewhere, K(other); and the ratios of K(UO) and K(other) to N. That is, the support provided by terms in the upper ontology is s(UO)=K(UO)/N, while the support provided by other prior ontology is s(other)=K(other)/N. Note that s(other) ranges from .59 to .68 for test batches TQA and TQC. In fact, the overall reuse of non-UO terms for Cyc/Tek and SRI was .66 and .60, respectively; whereas the overall reuse of UO terms for Cyc/Tek and SRI was .32 and .17, respectively. Thus, much of the difference in reuse statistics between SRI and Cyc/Tek is due to their exploitation of the upper ontology. Said differently, 22% of the terms SRI reused came from the upper ontology while the figure was 33% for Cyc/Tek.

	N	K(UO)	K(other)	S(UO)	S(other)
SRI TQA	104	20	61	.19	.59
SRI TQC	106	16	66	.15	.62
Cyc/Tek TQA(40)	168	67	101	.40	.6
Cyc/Tek TQC(40)	277	81	189	.29	.68
Cyc/Tek TQA(110)	418	126	276	.30	.66
Cyc/Tek TQA(110)	402	131	264	.33	.66

Table 2. Support provided by terms in UO and terms from other prior knowledge bases and ontologies.

In addition to encoding test questions, Cyc/Tek and SRI developed knowledge to answer the questions. This knowledge, called *axioms* generically, is composed of terms, so we can ask how prior ontologies helped the development of axioms. As before the relevant metric is s(i,j)=k(i,j)/n(i), only here, n(i) denotes the number of terms required to encode the ith axiom.

SRI provided data on how ontologies supported writing axioms. The rows of Table 3 represent the phases of the experiment and the source of prior ontology. The first row, SQ | UO shows that 1703 axioms were encoded to solve the sample questions SQ, and these axioms required 461 terms, of which 51 were in the upper ontology, UO, for a support value of 0.11. The second row shows that in the four days between the test and retest on batch TQA, 123 axioms were encoded, requiring 195 terms. 30 of these terms were found in the UO. The third row shows that 109 of the 195 terms were found in *all* the ontology developed prior to the test on TQA, namely UO *and* SQ. A comparison of the second and third rows shows that 109–30=79 reused terms came from SQ. The same pattern repeats in the two remaining phases of the experiment: After the scenario modification but before TQC, 1485 axioms were added to the SRI system. These required 583 terms of which 40 existed in the UO and 254 were found in the UO, SQ, and TQA prior ontologies. Similarly, between the test and retest on TQC, 215 terms were required for 304 axioms; only 24 of these existed in the UO, and 100 more were found in the ontologies developed after the UO.

It is unclear why prior ontologies provided significantly less support for encoding axioms than for encoding test questions. In both cases the support came in the form of terms, but why are the terms required to define axioms less likely to be in a prior ontology than the terms needed for test questions? One possibility is that test questions include fewer terms that represent *individuals* (e.g., #$HassiMessaoud-Refinery) than do axioms, so terms in test questions are less specific and more likely to exist in a prior ontology than terms in axioms. We will be looking at our data more closely to see whether this is the case.

	SRI			
	Axioms	n	k	s
SQ I UO	1703	461	51	.11
From TQA to TQA retest I UO	123	195	30	.15
From TQA to TQA retest I SQ	123	195	109	.56
From TQA retest to TQC I UO	1485	583	40	.09
From TQA retest to TQC I TQA	1485	583	254	.44
From TQC to TQC retest I UO	304	215	24	.11
From TQC to TQC retest I TQC	304	215	124	.58

Table 3: SRI measured the number of terms required to add problem-solving axioms to their system, and the reuse of terms from the UO and subsequent ontology efforts.

Discussion

Does prior knowledge in ontologies and domain-specific knowledge bases facilitate the development of knowledge-based systems? Our results suggest that the answer depends on the kind of prior knowledge, who is using it, and what it is used for. The HPKB upper ontology, 3000 very general concepts, was less useful than other ontologies, including Cyc and ontologies developed specifically for the crisis management domain. This said, Cyc/Tek made more effective use of the upper ontology: 33% of the terms it reused came from there whereas 22% of the terms SRI reused came from the upper ontology. Why is this? One reason is probably that Cycorp developed the upper ontology and was more familiar with it than SRI. Knowledge engineers tend to define terms for themselves if they cannot quickly find the terms in an available ontology. Once this happens – once a term is defined anew instead of reused – the knowledge base starts to diverge from the available ontology, because the new definition will rarely be identical with the prior one. Another reason for disparity in reuse of the upper ontology is that SRI preferred their own definitions of concepts to the available ones.

As to the uses of prior knowledge, our data hint at the possibility that prior knowledge is less useful for encoding axioms than it is for encoding test questions.

Whereas reuse of the upper ontology depends on who is using it, other ontologies seem to account for a roughly constant (60% – 66%) rate of reuse, irrespective of who developed these ontologies. For SRI, these ontologies were just those developed for batches of questions SQ, TQA, TQB, TQC and TQD. To be concrete, the 60% of the terms required for TQC were defined while encoding SQ, TQA and TQB. The picture is a bit cloudier for Cyc/Tek because they had the Cyc ontology throughout, and we have not yet analyzed whether the 66% non-UO reuse came from terms defined for previous batches or from Cyc.

Despite this ambiguity we speculate that in the process of building a domain-specific knowledge-based system, the rate of reuse of terms defined earlier in the process is 60%-70%. Although the rate of reuse of terms from very general ontologies may be significantly lower (e.g., 15%–30%), the real advantage of these ontologies probably comes from helping knowledge engineers organize their knowledge bases along sound ontological lines. It is essential for the ontology community to collect data on this use of general ontologies.

Conclusion

Although the idea of knowledge sharing has been in the literature for many years (e.g., Patil et al. 1992), the current paper presents the first empirical results quantifying ontology reuse. Many questions remain. Our data are crude summaries of reuse of terms, they do not tell us much about the work that knowledge engineers do when they build domain-specific knowledge bases. How long will a knowledge engineer hunt for a relevant term or axiom in a prior ontology? How rapidly do knowledge bases diverge from available ontologies if knowledge engineers don't find the terms they need in the ontologies? By what process does a knowledge engineer reuse not an individual term but a larger fragment of an ontology, including axioms? How does a very general ontology inform the design of knowledge bases, and what factors affect whether knowledge engineers take advantage of the ontology? Why do prior ontologies apparently provide less support for encoding axioms than for encoding test questions? Finally, will the results we report here generalize to domains other than crisis management and research groups other than SRI and Cyc/Tek? We expect to answer some of these questions retrospectively by analyzing other data from the first year of the HPKB program and prospectively by designing experiments for the second year.

References

Paul Cohen, Robert Schrag, Eric Jones, Adam Pease, Albert Lin, Barbara Starr, David Gunning, and Murray Burke. The DARPA High Performance Knowledge Bases Project. AI Magazine, Winter, 1998. pp. 25-49

Farquhar, A. Fikes, R., Rice, J. 1997. A collaborative tool for ontology contruction. International Journal of Human Computer Studies. Vol. 46. pp 707 - 727

Knight, K., and Luk, S. 1994. Building a large-scale knowledge base for machine translation. AAAI-94.

Lenat D. 1995. Artificial Intelligence. Scientific American. September.

Lenat D. and Guha R. 1990. Building Large Knowledge based system. Reading MA Addison Wesley.

Patil, R., Fikes, R., Patel-Schneider, P., Mackay, D., Finin, T., Gruber, T., and Neches, R. 1992. The DARPA Knowledge Sharing Effort: Progress Report. In KRR-92. pp 777 – 788.