Knowledge Begets Knowledge: Steps towards Assisted Knowledge Acquisition in Cyc

Michael Witbrock, Cynthia Matuszek, Antoine Brusseau, Robert Kahlert, C. Bruce Fraser, Douglas Lenat

Cycorp, Incorporated
3721 Executive Center Drive
Austin, Texas 78731-1615
{witbrock, cynthia, brusseau, rck, lenat}@cyc.com, bfraser@gmail.com

Abstract

The Cyc project is predicated on the idea that, in order to be effective and flexible, computer software must have an understanding of the context in which its tasks are performed. We believe this context is what is known informally as “common sense.” Over the last twenty years, sufficient common sense knowledge has been entered into Cyc to allow it to more effectively and flexibly support an important task: increasing its own store of world knowledge. In this paper, we describe the Cyc knowledge base and inference system, enumerate the means that it provides for knowledge elicitation, including some means suitable for use by untrained or lightly trained volunteers, review some ways in which we expect to have Cyc assist in verifying and validating collected knowledge, and describe how we expect the knowledge acquisition process to accelerate in the future.

Introduction

In the early 1980s, there was a surge in enthusiasm for Artificial Intelligence research, driven by the success of expert systems like DENDRAL, MYCIN and XCON [Lindsay et al, 1980; Buchanan et al, 1984, Sviokla 1990]. This interest waned, arguably in part because these systems were brittle [Friedland et al 2004] and unable to reason about situations even slightly removed from those originally conceived of by their authors. The purpose of the Cyc project [Lenat 1995] is to circumvent this brittleness by providing computers with a store of formally represented commonsense in which expert knowledge can be embedded and to which programs can refer when reacting to situations partially or wholly outside their intended domain. Over the last twenty years, many human “Cyclists” have been engaged in this painstaking representation process, and have represented over two million facts and rules about more than 200,000 entities and types of entities.

Over the last few years, it has become apparent to us that substantially more knowledge than this will be required to achieve comprehensive common sense, let alone wide-ranging, effective and flexible expert performance; more cost effective knowledge acquisitions must be found.

Fortunately, fulfilling the premise of the Cyc project that *“learning only occurs at the fringes of what you already know,”* the Cyc knowledge base now contains a store of general knowledge sufficiently large that it can be used to aid volunteers and other lightly trained, or untrained, users in its own growth. In this paper, we describe a number of knowledge elicitation techniques, and a framework for using the elicited knowledge effectively; only one of these techniques has been deployed for use by volunteers, in relatively small numbers, but all are designed to act as components in a system that will support eventual use by thousands of untrained volunteer knowledge enterers.

The Cyc Knowledge Base and Inference Engine

As of mid-2004, the Cyc knowledge base (KB) contains more than 2.2 million assertions (facts and rules) about more that 250,000 terms, including nearly 15,000 predicates. Importantly for the purpose of acquiring knowledge from volunteers, a significant component of the knowledge base is a highly developed English lexicon, containing the knowledge about syntax and semantics that allows the system to translate between its formal representations and English when communicating with users; far less complete, but easily extensible, lexical knowledge also exists for other languages. Another important component of the system’s knowledge is its representation of both general and specific knowledge acquisition goals, which enable it to drive knowledge collection.

The knowledge in the KB is made productive by the Cyc Inference Engine, Natural Language (NL) System and Interfaces. The inference engine supports deductive, abductive and inductive inference over a knowledge base the size of Cyc by integrating more than 700 specialized reasoners for commonly occurring classes of sub-problem. The Natural Language system uses both code and rules in the knowledge base to support isolated and, increasingly, discourse-based, translation of English words, phrases and sentences into the CycL formal language, and generation of English from CycL, both in isolation and in the context of a proof. The interfaces provide convenient means for knowledge entry and system query.

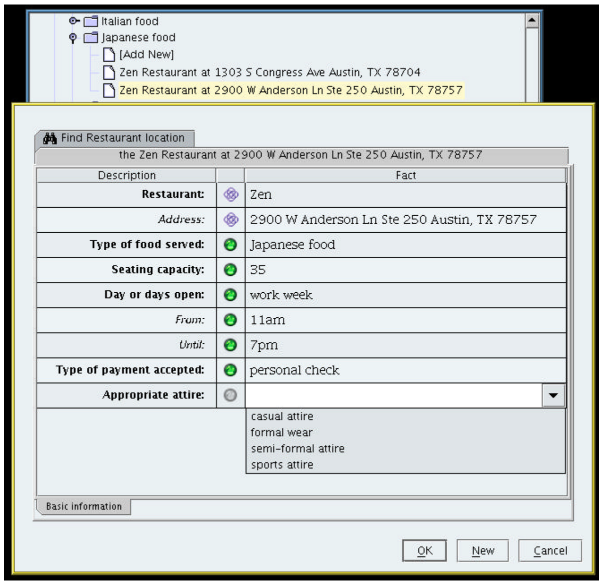
Types of Knowledge

There are two substantially different kinds of knowledge stored in the KB: ground facts and rules. Significant effort has been expended on constructing tools designed to reduce or eliminate the amount of training time required for a user to generate new knowledge of each type. Ground facts are comparatively straightforward to obtain and represent, whether from an untrained user interacting with the system via NL, or from a specialist trained in formal representation. Rules are significantly more difficult; trained Cyclists represent rules comparatively slowly, and interfaces designed to elicit rules from untrained users [Panton et al 2002, Witbrock et al 2003] have for the most part been even slower and fairly ineffective in producing productive, complex output. A compromise approach of representing rule information as simple sentences constructed with *rule macro predicates*[1] falls somewhere between the two in terms of difficulty. Work by other groups in assisted rule authoring has also imposed limitations on the complexity of the rules acquired [Baker et al 2003, Tecuci et al 2002].

We believe that this difficulty in obtaining complex rules is a general problem stemming from the mismatch between the requirements of formal reasoning and the way humans conceptualize their own reasoning processes. For this reason, we are pursuing means by which Cyc can obtain large numbers of ground facts from users, and can use those ground facts to induce and then verify rules to be added to the system[2].

Obtaining Ground Facts

Acquiring Highly Structured Facts in a Domain. One way of obtaining ground facts from users is to provide a semi-interactive form (Figure 1, below) in which users can enter simple statements in NL or NL fragments.[3] Such interfaces are particularly appropriate in cases where a large amount of similar information must be entered. Interfaces that provide a mechanism for populating specific relations, such as typical sizes of objects, are an obvious approach, and such a tool, the "typical size harvester" was, in fact, produced at Cycorp to obtain constraining knowledge for use in an information extraction system.[4]


Italian food
Japanese food
[Add New]
Zen Restaurant at 1303 S Congress Ave Austin, TX 78704
Zen Restaurant at 2900 W Anderson Ln Ste 250 Austin, TX 78757
Find Restaurant location
Basic information
OK
New
Cancel

the Zen Restaurant at 2900 W Anderson Ln Ste 250 Austin, TX 78757

Description		Fact
Restaurant:	❖	Zen
Address:	❖	2900 W Anderson Ln Ste 250 Austin, TX 78757
Type of food served:	🟢	Japanese food
Seating capacity:	🟢	35
Day or days open:	🟢	work week
From:	🟢	11am
Until:	🟢	7pm
Type of payment accepted:	🟢	personal check
Appropriate attire:	○	casual attire / formal wear / semi-formal attire / sports attire

Figure 1: The knowledge-driven Factivore™ interface enables users to enter appropriate information about instances of concepts – a particular restaurant in this case – by filling out simple forms in English. These forms are automatically generated from descriptions in the Knowledge Base, some of which are produced autonomously by inference. Visual feedback assures users that knowledge has been effectively entered.; the system, displays a "green light" and, where appropriate, rephrases the entered information.

General augmentation of a KB with Cyc's scope would require an enormous number of such specific tools to populate all the predicates used across its many domains; a better approach has proven to be use of a general-purpose template-based knowledge entry tool, the *Factivore*™. The Factivore requires that new templates be written for any given domain, but still requires far less per-domain expense than building more focused tools. Because the template specifies the expected underlying CycL representation,

[1] A rule macro predicate is a single predicate with a pre-defined expansion into an implication. The simplest example is #$genls, or "generalizes to," which applies to collections of constants:
(genls ?A ?B) = (implies (isa ?X ?A) (isa ?X ?B))

[2] In fact, we don't believe that the situation for lightly trained users is hopeless; in the latter stages of the DARPA RKF project, we developed techniques for automatically transforming descriptions of problem solving *procedures* into rules. These "analysis diagrams" will be described in other publications.

[3] This effort resembles Chklovski's LEARNER [Chklovski 2003] and the Open Mind project [Singh et al 2002]. The most significant methodological difference is that, because the Cyc effort is using an existing ontology, new facts are canonically represented and are immediately available for use in inference, requiring no post-processing.

[4] Using the size harvester tool, 10 participants were able to populate three concepts at the rate of 3000 assertions/month.

users of the tool need only fill in clearly defined blanks to produce valid formal representations. These templates are flexible enough to allow relevant knowledge to be properly contextualized (e.g., specifying the specific interval when Bill Clinton was president of the U.S.).

The Factivore has been successfully used on tasks as diverse as project management (Cycorp project managers have populated the KB with project staffing and task information) and historical terrorism data (lightly trained SMEs, some of them volunteers, entering information about terrorist attacks around the world, have produced half a million assertions [Belasco et al 2004] at an average fact-entry rate of 96 assertions/hour).

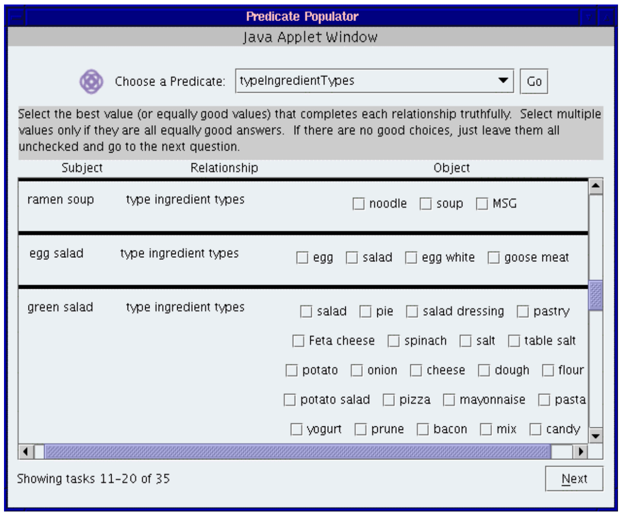
Acquiring Commonsense Type-Level Knowledge. A class of knowledge that is difficult for untrained users to describe readily in natural language is *type-level* knowledge, such as *"Croissants contain flour."* or *"Cafés sell coffee."* Although these statements certainly represent commonsense knowledge, most SMEs have had difficulty producing them without implicitly confusing the type and instance levels[5]. This is representative of the difficulty volunteers have describing rules, even rules as simple as: *"If there is a café, that café sells coffee."* Most such simple deductive knowledge is captured by rule macro predicates or type-level predicates:

```
(relationAllExists ingredients Croissant Flour)
   =
(implies
   (isa ?X Croissant)
   (ingredients ?X Flour))


(agentTypeSellsProductType Cafe-Org Coffee-Hot)
   =
(thereExists ?X
   (implies
      (isa ?Y Cafe-Org)
      (sellsProduct ?Y Coffee-Hot)))
```

The difficulties encountered by volunteers in populating this kind of knowledge (and the increased sophistication of the underlying representation) suggests that doing so qualitatively differs from the process of entering simple ground facts. In this case, a template-based tool that offers the opportunity to enter a single piece of knowledge is less useful than a tool with lower cognitive load that offers the user the ability to select among plausible choices. Efforts to populate type-level predicates can benefit from automated mechanisms providing a set of options from which the user may select.

The first such tool developed was the Predicate Populator. This tool (shown in Figure 2, below) looks at web pages



Predicate Populator

Java Applet Window

Choose a Predicate: typeIngredientTypes Go

Select the best value (or equally good values) that completes each relationship truthfully. Select multiple values only if they are all equally good answers. If there are no good choices, just leave them all unchecked and go to the next question.

Subject	Relationship	Object
ramen soup	type ingredient types	☐ noodle ☐ soup ☐ MSG
egg salad	type ingredient types	☐ egg ☐ salad ☐ egg white ☐ goose meat
green salad	type ingredient types	☐ salad ☐ pie ☐ salad dressing ☐ pastry ☐ Feta cheese ☐ spinach ☐ salt ☐ table salt ☐ potato ☐ onion ☐ cheese ☐ dough ☐ flour ☐ potato salad ☐ pizza ☐ mayonnaise ☐ pasta ☐ yogurt ☐ prune ☐ bacon ☐ mix ☐ candy

Showing tasks 11-20 of 35

Next

Figure 2: The Predicate Populator is populated from a combination of predicates and collections found in the Knowledge Base and information gathered from web pages about those collections.

about collections, such as *"Café,"* and makes a list of all proximate nouns that are known to correspond to Cyc collections matching the argument constraints for the predicate in question.

Without the knowledge base's preexisting base of Natural Language lexical assertions, the system could not automatically match terms in the knowledge base to those found in natural language corpora to select appropriate choices. The rate at which positive facts were asserted into the KB by the Predicate Suggestor was approximately 400 facts an hour, or 7 per minute, while the rate of suggestions that were deemed "inappropriate" (i.e., that should have been rejected before reaching human review) as opposed to simply factually incorrect was quite low[6]:

	Total	True	Skipped	Inappropriate
Reviews/ Hour	609	414	167	28

[5] In RKF Years 2 and 3, we discovered that untrained SMEs generally could not distinguish generic singulars from instances without coaching.

[6] While no metrics were gathered, it is worth noting that the interest level of participants working on the Predicate Populator tasks was much higher than that of participants asked to use the Typical Size Harvester. In general, users asked to enter the typical size of various objects found the task dull, while users asked to select typical type information (such as what products are sold where or what ingredients are found in foods) found the task enjoyable. This may relate to the relative cognitive complexity of the tasks, the difficulty of filling in blanks versus selecting from presented options, the size of the trials conducted, or other factors.

A disadvantage of both the template-based approach and the more specific type-level tool approach is that they can require significant preparation by ontologists.[7] Although each has produced significant improvement in the rate of knowledge entry, large numbers of ground facts would ideally be obtained from text corpora, or from volunteers or untrained enthusiasts, and that rules be concluded from that data. Satisfying such a goal cost-effectively precludes significant OE involvement in preparation.

Validating Non-Expert Knowledge. Although the initial work on providing interfaces that non-experts can use is promising in terms of time per assertion, it does not address the question of validation. While the large-scale effort to gather knowledge about terrorists and terrorist actions has resulted in tens of thousands of useful assertions, it has also resulted in many incorrect assertions. The system can take advantage of the knowledge already in the Cyc KB to identify some incorrect entries. Anything that is in direct violation of existing knowledge can be automatically detected and rejected. Acceptance of (isa Witbrock LawnFurniture) ≡ *"Michael Witbrock is a piece of lawn furniture"* would be blocked by a rule-macro (disjointWith Animal ManufacturedProduct).[8] When something is entered into the Factivore that it does not understand, a functional term describing that entry is created, which incorporates the name that was given to it by the user and any type information that can be extracted from the template, *e.g.*:

```
(InstanceNamedFn "Roger Dodds" Person)
(InstanceNamedFn "Federal Republic of Germany"
   GeopoliticalRegion)
```

When a terrorist act is described as happening in a country that Cyc has no knowledge of, it is probable that a knowledge entry error has occurred, either in parsing or at the user end. The majority of existing countries are represented in Cyc (which claim is in turn described with the predicate completeExtentAsserted.) However, when the functional term refers to a person, it is probable that it represents a legitimately novel individual.

A very straightforward approach addresses the validation problem with voting audiences, where several volunteers must agree on the answer to a question before it is taken as fact. This approach presupposes that arbitrary volunteer time is available. Approaches such as automatically validating the information given by performing web searches and attempting to find verification in other corpora [*e.g.* Ji, 2000] have the potential to be less wasteful of human time, but represent a more difficult problem.

Answering Questions and Validating Hypotheses Automatically. Another approach to gathering knowledge in Cyc is to hypothesize plausible sentences, based on the knowledge already in Cyc, whose truth is not known. Such sentences can draw on the breadth of the KB to produce a much higher rate of non-arbitrary sentences. Given that someone is an artificial intelligence researcher, and lives in Austin, for example, it is plausible, based on abductive application of rules already in the Cyc KB, to propose that perhaps that individual is an employee of Cycorp[9].

In theory, such sentences can be validated automatically against other corpora, as with volunteer-entered knowledge. In practice, providing the means for doing so is still a very open research problem[10]. Fortunately, evaluating the truth of a hypothesized sentence is a task that is highly appropriate for interested volunteers. Since many of the abduced suggestions are amusing, it is reasonable to hope for sustained volunteer interest.

In the course of implementing and testing the suggested-sentence review tool, we found several other axes along which a sentence can vary. Ultimately, we found that everything the reviewers wanted to express about a single sentence could be captured by five characteristics:

- **Comprehensibility:** is a generated sentence easily understood by a human knowledge worker? In most cases, when a sentence is not clear, either the natural-language generation is faulty or the sentence makes use of a predicate that is intended to express a non-natural structural or internal concept.
- **Appropriateness:** is a sentence something that should have been found to be invalid during the type-checking phase of generation? For example, it would be inappropriate, if mildly entertaining, to suggest that *"Osama bin Laden is a triangle"* or that *"Al-Qaida is affiliated with inference parameters."*
- **Truth:** is the information content of the sentence sound? Many of the generated sentences, while reasonable hypotheses, are simply false.

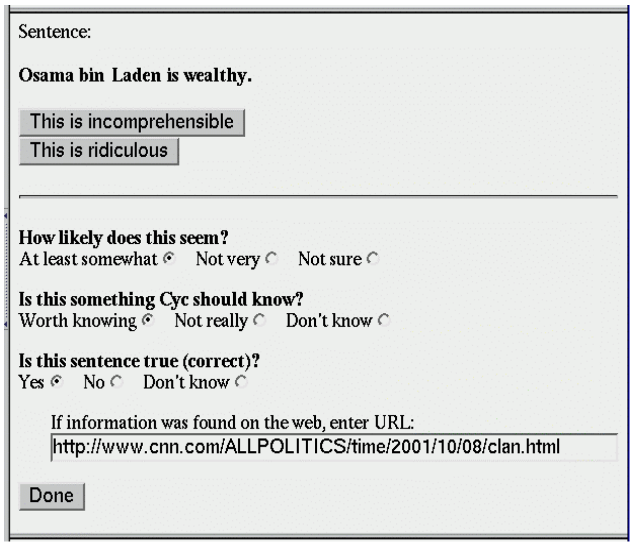
[7] While Factivore forms can be produced autonomously by induction over useful case and NL generation of field names, such forms are less polished. The Predicate Populator requires ontologist involvement in predicate selection. We expect to improve the automation of the predicate selection process, and some work on that goal has been done; for example, candidate predicates can be selected based on their "inferential fecundity," or utility in inference, which is determined recursively by their appearance in rule antecedents.

[8] Which, admittedly, may not be eternally true.

[9] Hypothesized sentences of this form have been generated in Cyc by adding abductive reasoning capabilities to the Inference Engine. Abduction takes advantage of rules in the knowledge base to hypothesize facts that would result in known assertions.

[10] To which we are seeking solutions in our work on Question Answering systems.

- **Interest value:** is the sentence saying something that is worth representing in the system for further reasoning, as opposed to an irrelevance, such as *"George W. Bush is sitting down"*?
- **Plausibility:** is the sentence something that seems likely to be true, based on common-sense evaluation rather than domain expertise? An example of an implausible sentence might be, *"Mel Gibson is affiliated with Al-Qaida"* – this *might* be true, but seems improbable. Evaluating the plausibility of a sentence is both subjective and contextual; it remains to be seen, therefore, whether evaluating the plausibility of sentences is useful.



Sentence:

Osama bin Laden is wealthy.

This is incomprehensible

This is ridiculous

How likely does this seem?
[✓ At least somewhat] Not very Not sure

Is this something Cyc should know?
[✓ Worth knowing] Not really Don't know

Is this sentence true (correct)?
[✓ Yes] No Don't know

If information was found on the web, enter URL:
http://www.cnn.com/ALLPOLITICS/time/2001/10/08/clan.html

Done

Figure 3: The Suggested Sentence Review Tool offers abductively proposed hypotheses for review.

Using this tool, lightly trained reviewers evaluated 900 sentences. The rate of knowledge entry over that time depended largely on whether reviewers were asked to determine the truth of sentences (*e.g.,* using Google) when they did not know. If they were asked to skip such sentences, the rate of review was approximately four sentences per minute. On an unfamiliar corpus about terrorist organization policies where reviewers were asked to determine the truth of the sentences, the rate dropped to one sentence per minute.

Obtaining Higher-Level Knowledge

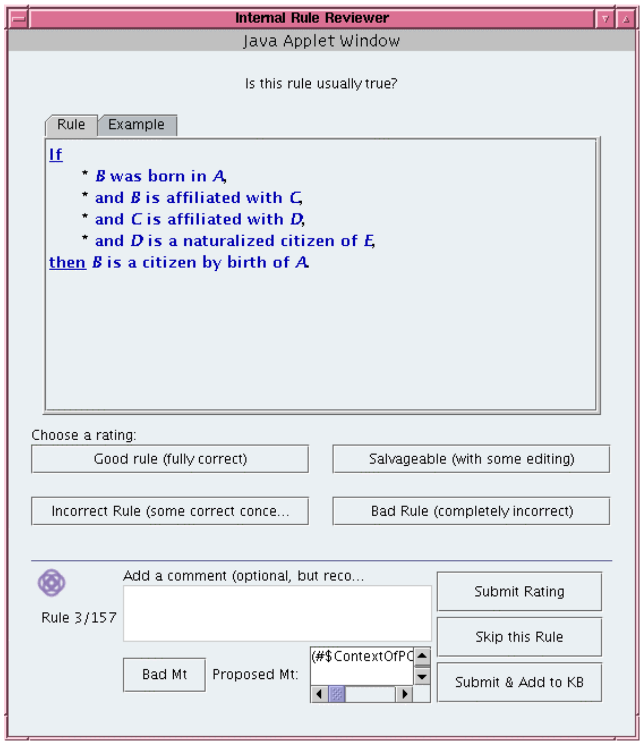
Obtaining Rules Via Induction. While gathering ground facts is inherently worthwhile, ground facts combined with rules provide the largest part of Cyc's ability to reason about novel queries. Constructing correct new rules in a formal representation system is a challenging task for specialists; efforts to utilize lightly trained Subject Matter Experts in rule generation, even with tools designed to simplify the process, have not been notably successful [Panton et al 2002, Witbrock et al 2003]. However, rules can be automatically generated applying machine learning techniques (e.g. the Inductive Logic Programming (ILP) system FOIL [Quinlan and Cameron-Jones, 1995]) to ground facts. The abductive sentence suggestor provides an unusually large body of plausible but negative assertions, which improves the performance of ILP over the Knowledge Base enormously; applying FOIL to a set of 10 predicates drawn from the Knowledge Base generates a set of approximately 300 rules.

These rules are not guaranteed to be correct. However, while rule authoring may be very difficult for untrained users of the system, rule *review* is feasible (although still difficult). In early trials, a reviewer could evaluate an average of twenty rules per hour. Of those rules, 7.5% were found to be correct, and 35% were found to need only minor editing to be assertible. For comparison purposes, experimental work in RKF Year 2 and Year 3 showed that human experts produce rules at the rate of approximately three per hour.

Some of the difficulties users encountered in performing this ranking task could be addressed by the design of the review tool, a preliminary version of which is shown in Figure 4 (following page). Defining the concept of rule quality presented an unexpected source of difficulty. Human ontologists produce and evaluate rules regularly as part of manually building the knowledge base; it might seem, therefore, that a definition of a "good" rule would have evolved naturally over time. Standards for evaluating the quality of a rule do in fact exist, but are largely implicit; in order to obtain reasonable, consistent results, a better definition of rule quality was formalized.[11] A second challenge was determining how many ranking categories could be presented without either constraining the user or forcing excessively fine-grained choices.

Rules are, furthermore, difficult to comprehend in the abstract. Providing reviewers with concrete examples helped them perform consistently, and when the option to show an example was provided they invariably chose to make use of it. Despite the promise of initial attempts at rule review, it may never be a task that can be performed well by amateur volunteers. An alternative approach would be to abductively generate large numbers of cases in which the rule would be valid, validate the hypothesized ground facts in the antecedent of the rule, and, for those rule exemplars whose antecedents are judged true, solicit

[11] There are two characteristics relevant to determining the goodness of a rule. The first is whether it is written at the correct level of generality; the second is correctness, which can be defined as *predictive power* over novel data in the domain.


Internal Rule Reviewer

Java Applet Window

Is this rule usually true?

If
* B was born in A,
* and B is affiliated with C,
* and C is affiliated with D,
* and D is a naturalized citizen of E,

then B is a citizen by birth of A.

Rule
Example
Choose a rating:
Good rule (fully correct)
Salvageable (with some editing)
Incorrect Rule (some correct conce...
Bad Rule (completely incorrect)
Add a comment (optional, but reco...
Rule 3/157
Submit Rating
Skip this Rule
Bad Mt
Proposed Mt:
(#$ContextOfPC
Submit & Add to KB

Figure 4: A preliminary version of a system for rule review, designed for internal use at Cycorp. The structure of existing KB content is used to drive the selection of clauses for rules induced from ground facts.

judgments over the validity of the corresponding consequent.

Once rules have been tentatively validated, it is our intention to make them part of the Cyc KB, offering users the ability to give negative feedback when such a rule is used in the proof of a conclusion with which the user disagrees.

Conclusions and Future Work

Although Cycorp has not yet deployed the knowledge acquisition tools described here for use by large numbers of volunteers, it intends to do so. We currently hypothesize that the most productive activity for such volunteers will be entering and validating ground facts in areas of common sense knowledge that are partially but not completely represented. The system's ability to infer knowledge acquisition goals from the state of KB content, and use those goals to drive interface presentation, makes it an active participant in its own construction. Having gathered large numbers of ground facts, we intend to apply ILP techniques to the production of inferentially productive rules, and to perform both experimental (by construction of new cases) and *in situ* (by allowing their use in proofs to be disputed) validation of the rules. We are currently extending existing ILP work for use in knowledge bases, like Cyc, with an enormous term and predicate vocabulary, and a representation grounded in natural concepts.

In future work, we hope to relieve even volunteers of the task of adding ground facts to the KB, instead using Cyc's increasing ability to interpret written texts into formal CycL to automatically add facts in areas where experience shows that the knowledge obtained is consonant with that entered using human volunteers.

Having spent twenty years manually constructing a knowledge infrastructure that provides an inductive bias for knowledge acquisition, we are increasingly applying our efforts to reaping the benefits of that work to greatly accelerate the rate and utility with which knowledge can be acquired. We invite other researchers to make use of the ResearchCyc platform to drive this effort forward.

References

Barker. K., Blythe, J., Borchardt, G., Chaudhri, V.K., Clark, P.E., Cohen, P., Fitzgerald, J., Forbus, K., Gil, Y., Katz, B., Kim, J., King, G., Mishra, S., Murray, K., Otstott, C., Porter, B., Schrag, R.C., Uribe, T., Usher, J. and Yeh P.Z., 2003, "A Knowledge Acquisition Tool for Course of Action Analysis," In *Proceedings of the Fifteenth Innovative Applications of Artificial Intelligence Conference* (IAAI-03), Acapulco, August 12-14 2003.

Belasco, A., Curtis, J., Kahlert, R. C., Klein, Charles, Mayans, C., and Reagan, P., 2004, "Representing Knowledge Gaps Effectively," in *Proceedings of the Fourth International Conference on Practical Aspects of Knowledge Management (PAKM 2004).* Springer-Verlag, 2004. (in Press)

Buchanan, B. G. and Shortliffe, E. H., 1984, *Rule-Based Expert Systems: The MYCIN Experiments of the Stanford Heuristic Programming Project.* Reading, MA: Addison-Wesley, 1984.

Chklovski, T., 2003 "LEARNER: A System for Acquiring Commonsense Knowledge by Analogy," in *Proceedings of Second International Conference on Knowledge Capture (K-CAP 2003).*

Friedland, N.S, Allen, P.G., Witbrock, M., Angele, J., Staab, S., Israel, D., Chaudhri, V., Porter, B., Barker, K., Clark, P., 2004, "Towards a Quantitative, Platform-Independent Analysis of Knowledge Systems," in *Proceedings of the Ninth International Conference on the Principles of Knowledge Representation and Reasoning (KR 2004).* Whistler, 507-515.

Ji, J., 2000, Semi-automatic Ontology-based Knowledge Extraction and Verification from Unstructured Document[s], M.Sc. Dissertation, University of Florida.

Lenat, D. B., 1995, “Cyc: A Large-Scale Investment in Knowledge Infrastructure, *Communications of the ACM* 38, no. 11.

Lindsay, R. K., Buchanan B. G., Feigenbaum, E. A. and Lederberg, J., 1980, *Application of Artificial Intelligence for Chemistry: The DENDRAL Project.* New York: McGraw-Hill, 1980.

Panton, K., Miraglia, P., Salay, N., Kahlert, R.C., Baxter, D., and Reagan, R., 2002, “Knowledge Formation and Dialogue using the KRAKEN Toolset,” in *Proceedings of the Fourteenth Innovative Applications of Artificial Intelligence Conference* (IAAI-2002), pp 900-905.

Singh, P., Lin, T., Mueller, E., Lim, G., Perkins, T. Zhu, W.L., 2002, “Open Mind Common Sense: Knowledge Acquisition from the General Public.” *Proceedings of the First International Conference on Ontologies, Databases, and Applications of Semantics for Large Scale Information Systems*. Irvine, CA.

Quinlan and Cameron-Jones, 1995, Induction of Logic Programs: FOIL and Related Systems, *New Generation Computing* 13, pp 287-312.

Sviokla, J.J., 1990, “An examination of the impact of expert systems on the firm: the case of XCON,” in *MIS Quarterly*. V. 14 no. 2, pp 127-140. June 1990.

Tecuci, G., Boicu, M., Marcu, D., Stanescu, B., Boicu, C., Comello, J., Lopez, A., Bonion, J. and Cleckner, W., 2002, “Development and Deployment of a Disciple Agent for Center of Gravity Analysis,” in *Proceedings of the Eighteenth National Conference on Artificial Intelligence*, 2002, pp 853-860, Edmonton.

Witbrock, M. J., Baxter, D., Curtis, J., Schneider, D., Kahlert, R., Miraglia, P., Wagner, P., Panton, K., Matthews, G., Vizedom, A., “An Interactive Dialogue System for Knowledge Acquisition in Cyc,” in *Proceedings of the IJCAI-03 Workshop on Mixed-Initiative Intelligent Systems*, Acapulco, Mexico, 2003. pp 138-145.