An Interactive Dialogue System for Knowledge Acquisition in Cyc

Michael Witbrock,
David Baxter, Jon Curtis, Dave Schneider,
Robert Kahlert, Pierluigi Miraglia, Peter Wagner,
Kathy Panton, Gavin Matthews, Amanda Vizedom

Cycorp, Inc., 3721 Executive Center Drive, Austin, Texas 78731
{witbrock, baxter, jonc, daves, rck, miraglia, peter, panton, gmatthew, vizedom}@cyc.com

Content Areas: common sense reasoning, knowledge representation, dialogue systems

Abstract

Cycorp has developed a knowledge acquisition system, based on Cyc, that can engage a user in a natural-language mixed-initiative dialogue. In order to achieve a intelligent dialogue with the user, it employs explicit topic- and user-modeling, a system of prioritized interactions, and a transparent agenda to which either the user or the system can add interactions at any time. Interactions initiated by the Cyc system are directed at closing system-perceived knowledge gaps, at optimizing the inferential utility of existing and added knowledge, and at generally facilitating the user's knowledge-entry task. This is done both deductively, in response to explicit knowledge-elicitation rules, and inductively, from the existing content of the Cyc Knowledge Base.

1 Introduction

Cycorp, and the previous Cyc project at MCC, have, for the last 19 years, been developing a massive repository of common-sense, real world knowledge called Cyc, currently the largest knowledge base in the world, with over 1.6 million largely hand-entered facts and rules that interrelate more than 118,000 concepts. The contents of the Cyc KB are represented in CycL, a highly expressive, logically perspicuous language based on second order logic. Assertions in the KB are stored in microtheories [Lenat, 1998]—explicit representations of context that enable Cyc to represent multiple perspectives, and serve to focus on a concise, relevant body of rules during inference.

Recently, in order to allow a much larger class of people to add knowledge to Cyc, a start has been made on developing natural-language based tools that rely for their operation on information in the Cyc KB, and that shield the knowledge enterer from CycL.

A design principle of the system has been to follow a mixed-initiative dialogue model, with the user engaging the system in a conversation in which Cyc acquires new knowledge, taking the initiative as appropriate to ask clarifying questions or to guide the user toward entering knowledge that would add to the system's inferential abilities. This system relies on the basic support in Cyc for natural language generation and understanding, as described in the remainder of this section.

1.1 The Cyc Lexicon

The lexicon [Burns and Davis, 1999] contains syntactic, semantic, and pragmatic information for approximately 17,000 English root words. The lexicon also contains approximately 36,000 multi-word phrases and 45,000 proper names. Inflectional and derivational morphology are handled by a separate code component. Each root word is represented as a term in the knowledge base, with assertions providing information about the word's part of speech, subcategorization patterns, and semantics. Semantic information in the lexicon involves a mapping between word senses and corresponding KB concepts or formulae.

1.2 Generation

The natural language generation system produces a word-, phrase-, or sentence-level paraphrase of KB concepts, rules, and queries. The NLG system relies on information contained in the lexicon, and is driven by generation templates stored in the KB. These templates are not solely string-based; they contain linguistic data which allows, for example, for correct grammatical agreement to be generated. Semantic information in the templates is used to vary the generation depending on, for example, the semantic types of the arguments. The NLG system is capable of providing paraphrases at varying levels of precision. At one end of the spectrum is the tersest, most natural paraphrase of the CycL input, but which can be ambiguous. At the other end of the spectrum is paraphrase that is much more precise, but is potentially wordy and stilted. The system is able to

raise the degree of precision when necessary to prevent ambiguities that might confuse the user.

1.3 Parsing

Our natural language understanding system parses input strings not simply into syntactic trees, but into fully formed semantic formulas. Design criteria for the parsing system included that it (1) be fast; (2) produce parses of adequate semantic detail; (3) ask the user for clarification only in cases where the system could not itself resolve ambiguities; and (4) support parsing into underspecified formulas, and then rely on other components to determine the best semantic translation. The Text Processor controls the application of the various parsing subcomponents, using a heuristic best-first search mechanism that has information about the individual parsers, their applicability to coarse syntactic categories, cost, expected number of children, and so on. This information is used to perform a syntax-driven search over the parse space, applying relevant parsers to the sub-constituents until all are resolved, or until the parsing options have been exhausted. The parsers at the disposal of the Text Processor are the Template parser, the Noun Compound parser, and the Phrase Structure parser.

The Template parser is essentially a top-down string- and word-matching mechanism driven by a set of templates compiled into an efficient internal format. These templates employ a simple format so that users can add templates as they are entering new knowledge into the system. The template parser is relatively fast, but is of limited flexibility. It tabulates semantic constraints during a parse, but does not attempt to verify them; that task is passed along to the next processing layer.

The Noun Compound parser uses a set of semantic templates combined with a generic bottom-up chart-parsing approach to construct representations for noun compounds such as "anthrax vaccine stockpile". Unlike other parsing components, it makes heavy use of the knowledge base, and can therefore resolve many ambiguities that are impossible to handle on a purely syntactic level (e.g. "Mozart symphonies" vs. "Mozart expert").

The Phrase Structure parser takes a similar bottom-up approach to constructing parses. After completing a syntactic parse, it uses semantic constraints gleaned from the KB to perform pruning and to build the semantic representation using lexical information like which syntactic constituents map to which semantic roles.

1.4 Post-Processing

In order for parsing to be successful in the current application, some decisions about semantic meaning need to be deferred during parsing. In particular, radically vague or underspecified words such as 'is' or 'contains', which can map onto many distinct relations in the KB, introduce ambiguities that are not handled well by producing all possible interpretations in parallel. To deal with such cases, strings are parsed into an intermediate layer (called iCycL) that conflates relevant ambiguities into a single parse, by using very general predicates such as is-Underspecified. The Reformulator, drawing upon background knowledge and iCycL transformation rules, reformulates these intermediate representations into final, more specific CycL representations, often with the user's help.

In addition to handling underspecification, the iCycL layer is also well suited for other types of semantic processing, such as interpretation of quantification and negation. The interpretation of quantifiers, for example, occurs by first compositionally representing the constituent noun phrases of a sentence as syntactic structures that contain CycL terms annotated with syntactic information, such agreement. Quantifier-processing rules are then used to break apart these syntactic structures to produce full-fledged CycL logical forms. Although CycL representations are modeled on first-order logic, the language itself allows the definition of higher-order predicates. We exploit this capability to represent a wide range of NL quantifiers formally as generalized quantifiers, i.e., as higher-order relations between sets of objects.

1.5 Lexical Additions

Because users are allowed to add new terms to the ontology, we have created a special-purpose tool to elicit the information necessary to be able to parse to and generate from new terms in the ontology. The Dictionary Assistant allows the user to specify appropriate syntactic and semantic information through a dialogue in which they do things like verify the part of speech (count noun, adjective, etc.) that the system has assigned, declare it to be any of a large number of types of names, and tell the system whether or not this is the preferred way to refer to the new term. For new relations, the user can add both parsing and generation templates. The interactions of the Dictionary Assistant attempt to minimize the amount of linguistic knowledge required of the user.

2 Infrastructure: The User Interaction Agenda

The infrastructure for the system described in this paper is called the "User Interaction Agenda", or UIA. It is designed in a modular fashion, so that parts of it can be replaced with new versions leaving the rest intact. For instance, we are currently in the process of replacing the HTML-based interface with one written in Java.

The user initiates a UIA session by entering a username and choosing a topic. Based on what the system already knows about the user, a list of languages[1] is displayed for the user to choose from. In the future, elicitation of user preferences based on the KB's persistent user model will be done more extensively; this is just a taste of the wide range of user options that will be available for guiding the conversation, and hence making the knowledge entry session better tailored to the user's needs and individual style, and thus more efficient and pleasant.

Explicitly setting the topic allows the system to customize certain interactions, and to filter the knowledge

[1] Of which, at present, only one, English, has significant support.

visible by focusing on a restrictive set of microtheories. For example, selecting the topic "Military Course of Action Analysis"[2] customizes the system to organize created knowledge along militarily relevant lines.

An Agenda is then created for the session that keeps an explicit set of parameterized interactions, each with a well-defined type, priority, status, and context.

Both Cyc and the user can add interactions to this Agenda at any time, and the user can select a pending interaction for immediate attention. Once an interaction is complete, it is logged in a journal and the pending interaction on the Agenda with the highest priority becomes active.

In the next sections, we will describe in more detail how dialogue initiative alternates between the user and the system. To illustrate, we will draw examples from an illustrative UIA session in which a user is telling Cyc about Jerry Allison, the drummer in Buddy Holly's rock and roll group, the Crickets.

3 User-Initiated Interactions

As mentioned above, one of the ways the user can take the initiative in the dialogue at any time is by launching a new interaction. The interface allows the user to do this either by selecting an interaction type from a menu, or by typing a natural language utterance into a text box that is always available (known as the "Say This" box).

When an utterance is entered into the Say This box, the system parses it as one of: a question (e.g. "What do you know about the Crickets?"), a command ("Compare the Crickets and the Beatles."), or an assertion ("Jerry Allison is a member of the Crickets.").

3.1 Analogy Development

One way a user can enter knowledge is by selecting a similar concept from the Cyc knowledge base and using it as the basis for an analogy. For example, one could type, in the "Say This" box, [3]

Jerry Allison is similar to Ringo Starr.

prompting Cyc to do two things. First, Cyc will ask the user to assent to or reject the system's suggestions for adding definitional information to Jerry Allison, based on what it knows about Ringo Starr. "Definitional information," in this context, refers to the assertions that place a new term in the overall ontology. For individual entities,[4] such as particular organizations (The Beatles, the World Bank), events (World War II), and living things (Rin Tin Tin, Ringo Starr), this definitional information will be assertions that identify what collection or class an object belongs to. Since Ringo Starr is known to be an instance of each of the classes of British citizens, adult male persons, and drummers, Cyc will ask,

Is Jerry Allison a songwriter?
Is Jerry Allison a man?
Is Jerry Allison a drummer?

If the user responds, "yes" to any of the above, Cyc is able to continue to the next step, using a simple version of the type of analogical reasoning outlined in [Forbus, 2002] to offer non-definitional assertions for the user to reject as inapplicable, accept as is, or to use as a template for constructing similar assertions.

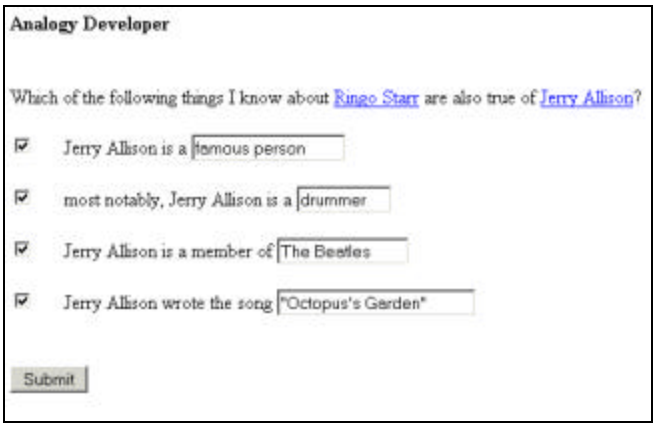

Figure 1: Cyc asks for information about the new term Jerry Allison based on analogy with Ringo Starr.

3.2 Clarification Dialogue

When the user has entered a phrase or sentence that could be interpreted in more than one way, the system will temporarily take the initiative and ask one or more questions to clarify what the user meant. For instance, knowing that Jerry Allison is from Hillsboro, Texas, the user might enter:

Jerry Allison is from Hillsboro.

Cyc will then ask the user to clarify which of the interpretation choices in Figure 2 was the intended meaning.

Once the clarification has been made, the system attempts to store the new information.

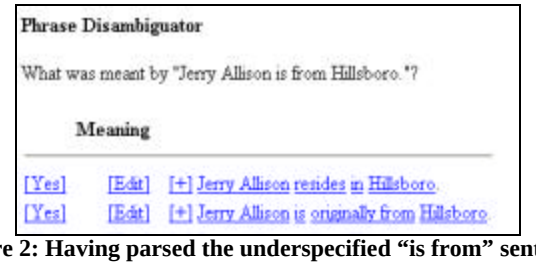

Figure 2: Having parsed the underspecified "is from" sentence, Cyc presents two alternative interpretations so the user can specify which one is intended.

[2] An evaluation topic for a program sponsored by DARPA.

[3] Alternatively, the user may invoke the analogy development tool directly from a menu of tools.

[4] For collections, as opposed to individuals, definitional information includes both which collections the defined collection is an instance of, and which collections it is a specialization of. For example, the collection Musician is an instance of PersonTypeByOccupation, and a specialization of Person.

3.3 Well-formedness Checking and Repair

To understand how well-formedness repair works, it is useful to think of CycL sentences as relational structures. Each CycL sentence is grounded by a predicate that relates the other terms in the sentence to one another. Consider, for example, the CycL translation of "Jerry Allison is a member of The Crickets":

(hasMembers TheCrickets JerryAllison)

This CycL sentence is grounded by the predicate hasMembers, which relates the terms JerryAllison and TheCrickets. As with all CycL predicates, the definitional information for hasMembers includes a set of *argument constraints* – requirements that any term must meet in order to be meaningfully related by the predicate in an assertion. For example, the first argument to hasMembers must be an instance of Organization.

Given this relational structure, it is possible for a user, entering knowledge through natural language parsing or through sentence cloning, to construct a CycL formula that fails to be semantically well-formed, in that one or more of the terms fails to meet the argument constraints of the relevant predicate. If the user enters such an ill-formed CycL formula, Cyc will again take the initiative to point out the problem, suggest an additional assertion that will fix the problem and allow the new fact to be entered.

3.3.1 Semantic Repair: Filling in Knowledge Gaps

One type of ill-formedness arises from a lack of knowledge: If a term *could*, but does *not yet* meet the relevant argument constraints, Cyc will propose to add the requisite definitional information. This corrective behavior extends to situations in which the user introduces a completely new term to the system. For example, suppose the user wishes to use the analogy developer to clone and modify the assertion,

Ringo Starr wrote "Octopus's Garden"

by substituting Jerry Allison for Ringo Starr and Peggy Sue for Octopus's Garden. The CycL 'beneath' the English,

(authorOfSong OctopussGarden-TheSong
RingoStarr),

is grounded in the predicate, authorOfSong, which requires a song as its second argument. If Cyc does not yet have a term representing the song "Peggy Sue," it will respond to the new, cloned input by suggesting that, as part of the novel term's definitional information, "Peggy Sue" must be a song (see Figure 3).

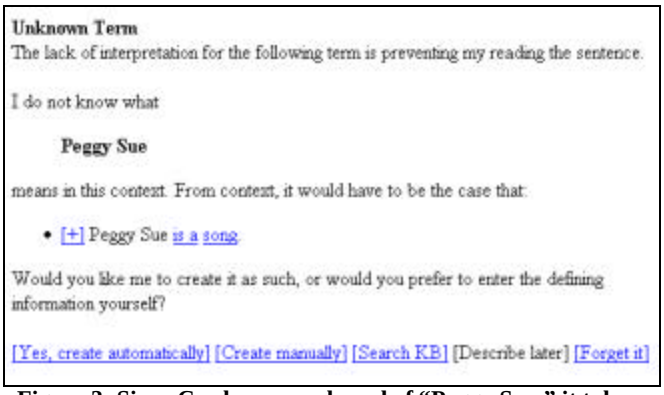

Figure 3: Since Cyc has never heard of "Peggy Sue," it takes the initiative, using information derived from the context, in guiding the user to create this new concept.

The user also has the option to further refine the new term, either immediately or as a later interaction.

3.3.2 Semantic Repair: Instance/Type Mistakes

In another form of semantic constraint repair, Cyc applies its knowledge to certain, recognizable patterns of ill-formed input to simulate something akin to the principle of charity [Davidson, 1986]. This principle is frequently used by humans to ascribe the most sensible interpretation to a dialogue partner's utterances, under the "charitable" assumption that the utterance was intended to be sensible. It is interesting to note that this sort of repair is required by the fact that the UIA interface supports sentence cloning, essentially a process for manipulating CycL terms nearly directly, by manipulating the English in a templatized paraphrase of a CycL formula.

For example, while cloning a sentence during analogy development,

Ringo Starr can play the drums.

a user might be tempted to replace the grammatically correct subject, "Ringo Starr", with an equally grammatically correct subject: "Drummers". However, the CycL for the original sentence,

(skillCapableOf RingoStarr
(PlayingInstrumentFn Drum) performedBy),

requires that whatever currently fills the place of RingoStarr be an instance of the collection Agent; only agentive entities like people and organizations can "perform" intentional activities like drumming. This rules out the CycL term for "drummers" – Drummer – as a valid substitution for RingoStarr, as Drummer is a *collection*, or *class*, (the collection of all drummers, including Ringo Starr), and not a drummer or an agent, itself. Thus, by substituting in well-formed English for well-formed

English, the user can create a situation where the end result is semantically ill-formed CycL[5]

The automatic semantic repair module, however, can correct this situation by introducing quantification, behind-the-scenes, so to speak. Confronted with the ill-formed,

```
(skillCapableOf Drummer
  (PlayingInstrumentFn Drum) performedBy)
```

Cyc checks to see if perhaps this might be more charitably interpreted as an attempt at stating a generalization about drummers, and not as a claim about the capabilities of the abstract collection Drummer. To do this, it checks to see whether Drummer is a subset of the collection of Agents, which proves to be the case. Having verified this, it introduces universal quantification, and interprets the user input as a general (and in this case, correct) rule, that all drummers can play the drums:

```
(implies
  (isa ?X Drummer)
  (skillCapableOf ?X
    (PlayingInstrumentFn Drum) performedBy))
```

3.3 Precision Suggestion

It is often the case that a human knowledge enterer does not formulate knowledge in the "strongest", or most inferentially productive, way possible. Before a user's statement is entered into the Cyc knowledge base using the dialog system, Cyc looks for ways to strengthen the formulation, making it more useful for inference.

For instance, if the user enters the sentence "Jerry Allison is a drummer," Cyc may ask the question in Figure 4.

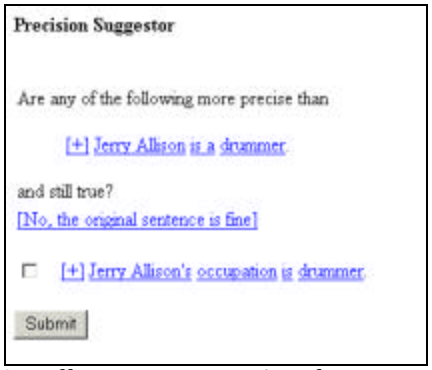

Figure 4: Cyc offers a stronger version of a statement the user has made, i.e. one that allows it to make more or stronger inferences.

This stronger version allows Cyc to conclude new knowledge about Jerry Allison from general knowledge about occupations. For example, it can now conclude that he is paid for drumming.

4 System Initiative: The Salient Descriptor

The most compelling example of Cyc taking the initiative in dialogue is the Salient Description process. As the user enters knowledge about a concept, a background process uses the term representing that concept as a seed to come up with questions to ask about it, based on what Cyc knows about the seed term and on the state of the knowledge base in general. The user can at any time inspect the list of pending questions Cyc has available for concepts mentioned in the dialog and select one to answer, and can instruct the system to begin generating questions for any term that Cyc knows about.

When the user chooses to answer a question, the interface not only displays the question and various means by which the user might answer it, but also allows the user to view Cyc's explanation for why it judged the question appropriate,[6] and, when possible, a small set of examples that the user can either select an answer from or pattern an answer after.

4.1 Predefined Question Types

Ontologists at Cycorp have authored numerous knowledge entry (KE) facilitation rules that describe the kinds of knowledge it is useful for Cyc to have regarding terms with certain properties. For instance, the Cyc knowledge base contains the following "KE facilitation" rule:

For every *X*, if *X* is a musician, then it is strongly expected that *X* plays some type of musical instrument in some musical group.

So, once Cyc knows that Jerry Allison is a drummer, and therefore a musician, it knows to ask the question in Figure 5.

Figure 5: Cyc uses a predefined question for musicians to ask the user for a new piece of information about a particular musician the user has defined.

The user can then conveniently add the information that Jerry Allison plays the drums in the Crickets.

4.2 Rules that Almost Apply to a Term

As well as using handcrafted rules to drive the interview process, the system uses its knowledge to autonomously derive and propose new interview questions. In one such

[5] This illustrates one of many difficulties in covering a system, like Cyc, based on predicate calculus, with a natural language like English, with a vastly different syntax. Small modifications to a CycL sentence can cause vast changes in the corresponding English, and vice versa.

[6] The user may disagree; the system also provides an option for explaining why the question may be inappropriate, so that the basis for its having been asked can be reviewed by ontologists.

process, the Salient Descriptor inspects Cyc's inventory of rules to find ones that would apply to the seed term if it knew just one or two more facts, and then asks the user to confirm (or deny) those facts.

This approach allows Cyc to guide knowledge entry in a way that is designed to make it more inferentially powerful, often eliciting knowledge that the user might not otherwise think to enter.

For instance, once the user tells Cyc that Jerry Allison is married to Peggy Sue, Cyc realizes that Jerry Allison may have some nieces or nephews by marriage, and asks the question in Figure 6.[7]

Figure 6: Cyc finds a rule that almost applies to Jerry Allison, and asks the user if the unknown part of the rule's antecedent is true.

4.3 Induced Interview Questions

When the user is entering a new instance of a class of terms about which Cyc already has some knowledge, the system can inspect the existing instances for patterns that suggest knowledge the user may be able to provide about the new term.

For instance, many of the people Cyc knows about have their nationalities represented. If the user enters a new term representing a person, and neglects to enter the person's nationality, Cyc can notice this and prompt the user to enter it, giving previously entered nationalities of people as examples.

[7] The fact that this question may be somewhat off-topic is an issue with the system, and is discussed further in section 5.1.

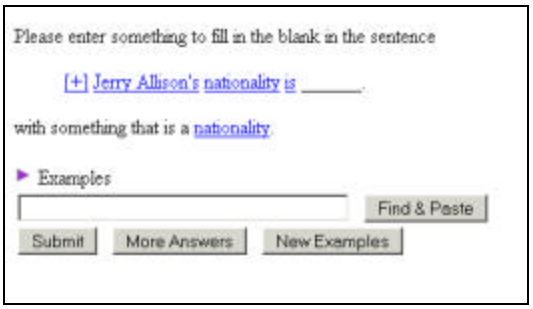

Figure 7: Cyc asks for Jerry Allison's nationality, based on analogy to similar terms for which it already knows nationality information.

Although the knowledge formation dialogue uses Cyc's knowledge extensively throughout its operation, this effective use of straightforward induction to drive knowledge acquisition offers support for the Cyc hypothesis: that by hand crafting a large knowledge base, an effective inductive bias for automatic knowledge acquisition can be provided.

5 Future Directions

The system described here has been developed using the Cyc KB and inference system and was able in initial experiments to support a two-fold increase in knowledge entry rates. From observation of Subject Matter Expert (SME) behaviour during challenge problems, of the sort described in the papers surveyed by [Kim and Gil, 2002], and from internal studies, Cycorp has been able to identify some key opportunities for the future development of the system, to enhance the sense of intelligent dialogue, while increasing both flexibility and robustness. These span areas of both system and user initiative, and include:

5.1 Conversational Goals and Focus

In advancing the system's capabilities towards the level of a human teacher-student interaction, the most important improvement must be at the highest level. The system must be able to perform meta-reasoning about the dialogue at multiple levels.

At a short-term level, a human conversation has identifiable focal topics and concepts. Succeeding contributions will both build upon this context, and change it. In order to permit both efficient knowledge entry, and effective pedagogical output, the system will track the current focus, promoting and demoting concepts as appropriate, recognizing changes in the discourse context and computing their implications.

The specific focus will change over the course of a conversation. For example, a conversation about Jerry Allison might focus on his band, his musical specialty, his wife, and a particular song. Indeed, every user response will change the discourse context in some respect.

Conversations, however, typically have a general topic. Thus a conversation about Jerry Allison might transition from his wife to involve his nieces and nephews, but should not pursue that thread further without a specific indication from the user that they are topical.

Further, in addition to having a general topic, conversations, especially in a pedagogical context, typically have goals. A subject-matter expert using the system will either have a specific body of knowledge to enter, or will have a target question-answering capability to add to the system.

In a reversal of the pedagogical roles, the user takes on the role of student, and wants to learn particular facts, or comprehend some concept. The system's ability to communicate about the objective of the conversation will enable it to act agentively in pursuing the shared goal.

In terms of specific implementation, the system will take the existing salient descriptor technology described above, and use the discourse model to drive the generation of interview questions towards the goals, filter them by current focus, and structure them by future focus. This will enable the system to mimic the characteristics of intelligent conversation that make it an efficient medium of information exchange.

5.2 Ambiguity Resolution

Beyond a mundane understanding of vocabulary and syntax, the outstanding problems relate to the ambiguity introduced by anaphora and polysemy.

The system must be able to take anaphoric references and correctly resolve the referent in the context of the discourse model. For example in the paragraph:

> Jerry Allison wrote Peggy Sue.
> He used his wife's name for the title.

the second sentence includes two pronominal references to Jerry Allison ("he" and "his), whereas "the title" is both a definite description and a bridging reference, because the song's title has not been explicitly referred to before.

In the paragraph:

> Jerry Allison married Peggy Sue.
> It was a big day.

the pronoun "it" is an event anaphor, referring to the wedding implied by the marriage. This bears on the general problem that natural language utterances rarely reify the events, whereas formal representations typically map this into Davidsonian form.

Similarly, it is not in general possible to maintain a one-to-one mapping between natural language words and the concepts in a formal representation. In the examples above, "wrote" refers to song-writing, but it could equally refer to book-authorship, or even the act of writing a letter; "Peggy Sue" could now be either the song or his wife. At a higher level, words can act as different parts of speech, as in the World War II headline "British push bottles up Germans".

In its current incarnation, the system identifies the possible parses for a sentence, and asks the user to choose between them in a process known as interactive clarification dialogue. In future versions, the system will be designed to endure such ambiguity, using information both within a sentence, and in other sentences past and future to narrow the field. Thus this ambiguity tolerance will provide the user with two key benefits:

1. The dialogue is more fluent, because it never becomes blocked either by lengthy computations or by modal interactions; and
2. The deferred resolution allows the system to resolve more references by itself, and the user can better concentrate on the domain of discourse.

5.3 Automatic Generation of Test Questions

Test questions are routinely used to check the correctness of the existing KB knowledge itself. These test questions are organized into topic-specific tests suites.

The rule-based induction methods used to ask the user for information (section 4.2, above) can be extended easily to produce a suite of inference tests that both serve to validate the SME-entered knowledge, and protect against system degradation. Keeping with our example from Section 4.2, the user is asked whether Peggy Sue Allison is an aunt, as part of Cyc's attempt to gain information about Jerry Allison. That question is prompted by the existence of the following rule:

> If *FEMALE-PERSON* is *PERSON*'s aunt,
> and *FEMALE-PERSON* is the wife of *MALE-PERSON*,
> then *MALE-PERSON* is *PERSON*'s uncle.

By answering 'yes' to the question that this rule has prompted, or by going a step further and supplying the identity of Peggy Sue's niece or nephew, the SME has given the system the ability to prove that Jerry Allison is an uncle. The system thus can be prompted to add the question, "Is Jerry Allison an uncle?" or, in CycL,

```
(thereExists ?PERSON
      (uncles ?PERSON JerryAllison))
```

to a suite of tests, customized for the knowledge that the SME has entered. The benefits to extending rule-based induction technology in this way are threefold:

1) The SME has the ability to verify that the knowledge that he or she has entered is useful to—in fact used by—the system. Rather than merely seeing that his or her knowledge has been added to the system, there is concrete evidence that the knowledge has "taken," and that the dialogue has thus borne fruit.

2) All questions added to the test suite in this manner *should answer correctly*; failure is indicative of an error in the system, either in the state of knowledge, or in the inference engine itself. A dialogue-created test suite thus represents an important first step towards self-diagnosis by the system, itself an

important goal in achieving system-initiated intelligent behavior.

3) Some of the test questions will undoubtedly be inappropriate for the topic of the dialogue, as intended by the user. The inappropriateness of such questions thus speaks to the inappropriateness of the rules for that domain; the SME can thus help improve the system's understanding of topical relevance by tagging such questions (and, therefore, indirectly, such rules) as belonging outside the domain of the relevant discussion.

6 Conclusion

The system is currently capable of engaging the user in a mixed-initiative dialogue in which the user teaches the system new information. This system has enabled Subject Matter Experts to enter information (in English) at a faster rate than trained ontologists can when working directly in CycL. Future work will focus on making the system better able to track the topic and be more discerning in deciding when to take the initiative. This will have the effect of allowing a SME to enter information within a narrow domain without undue digression. To close the knowledge entry loop, the system will induce its own test questions, allowing both the user and the system to know that the knowledge has been conveyed effectively.

References

[Burns and Davis, 1999] K.J. Burns and A.R. Davis. Building and maintaining a semantically adequate lexicon using Cyc. In Viegas, E. ed. *Breadth and depth of semantic lexicons*. Dordrecht: Kiuwer, 1999.

[Davidson, 1986] Donald Davidson *Inquiries into Truth and Interpretation*, 1986.

[Forbus, 2002] Ken Forbus, T. Mostek, and R. Ferguson. An analogy ontology for integrating analogical processing and first-principles reasoning.
Proceedings of IAAI-02, July 2002.

[Kim and Gil, 2002] Jihie Kim and Yolanda Gil. Deriving Acquisition Principles from Tutoring Principles. In *Proceedings of the Intelligent Tutoring Systems conference (ITS-2002)*, Biarritz (FR), June 5-7, 2002.

[Lenat, 1998] Douglas B. Lenat. The Dimensions of Context Space. Cycorp technical report.