On the Effective Use of Cyc in a Question Answering System

Jon Curtis, Gavin Matthews, David Baxter
Cycorp, Inc. 3721 Executive Center Drive, Suite 100, Austin, TX 78731.
{jonc,gmatthew,baxter}@cyc.com

Topics: flexibility supported by discourse/user model, new types of questions, reasoning with incomplete knowledge, response generation.

Abstract

We describe a commercial question-answering system that uses AI – specifically, the Cyc system – to support passage retrieval and perform deductive QA, to produce results superior to what each question-answering technique could produce alone.

1 Introduction

This paper describes a working prototype of a commercial question-answering system that uses artificial intelligence in conjunction with NLP-driven passage retrieval in a way that integrates the two markedly different approaches to question-answering and produces better results than either approach could yield alone.[1] MySentient[2] Answers 1.0 draws upon the Cyc system, a large knowledge-base, common-sense reasoning system. We describe three areas in which Cyc's knowledge contributes to the system's overall question-answering ability, both directly in deductive question-answering, and indirectly, by supporting passage-retrieval. In particular, we focus on the use of Cyc for:

1. augmentation of NLP-based passage retrieval by generating NL expansions of key concepts mentioned in a question;
2. answering question types that pose a challenge to passage retrieval methods, such as procedural ("How do I …?") and cost/benefit ("Why should I …?"); and
3. paraphrasing the results of deductive question answering as NL strings for display to an end user.

We close with a discussion of the current limitations of the integrated system and a description of anticipated extensions to the use of Cyc in future versions.

2 Cyc

Cyc is a state-of-the-art artificial intelligence program that has been in development since 1984. Drawing upon the world's largest general-purpose knowledge base of over 164,000 concepts and 3,300,000 facts (rules and ground assertions) relating them[3], Cyc is the only AI program in existence today that can reasonably claim to have some degree of common sense. Cyc's knowledge is represented in CycL, a higher-order logical language based on predicate calculus. Every assertion in Cyc is represented in a context, or *microtheory*, which allows the representation of competing theories. Like ordinary concepts, microtheories are explicitly represented as first-class objects in Cyc, giving Cyc a reflective capability to reason about its own representations. Microtheories form a hierarchy that facilitates knowledge re-use (assertions stored in the most general contexts are always available), and inferential focus (given a query posed in a specific microtheory, other knowledge from sibling or more specific microtheories will not come into play). Cyc's inference engine combines general theorem proving (*e.g.* rule chaining) with specialized reasoning (*e.g.* subsumption and transitivity).

Cyc has been used in commercial web-search systems (*e.g.* HotBot) and in question-answering systems, most recently in a purely deductive system for answering AP chemistry questions, developed in collaboration with Vulcan, Inc. [Friedland *et al*., 2004]. Cyc's rôle in the MySentient system heralds its first appearance in a commercial question-answering system. MySentient makes use of Cyc pervasively, as a means to augment NLP-based QA, as the basis for a deductive QA module, and in other capacities, such as clarification and profiling, that will be touched upon here.

3 MySentient Answers 1.0

MySentient Answers 1.0 is a working question-answering system, designed by MySentient Ireland (R&D) Ltd. of Dublin, Ireland, and implemented in collaboration with Cycorp, Inc. of Austin, Texas, and the Center for Natural Language Processing in Syracuse, New York[4]. MySentient Answers has been the subject of extensive demonstration to interested commercial parties and is expected to be available for public access in the near future.

[1] The work described in this paper was made possible by the financial backing of MySentient.

[2] "MySentient" is a registered trademark of MySentient Ireland (I.P.) Limited.

[3] MySentient uses a carefully-chosen subset of the full Cyc knowledge base with 137,000 concepts and 1,700,000 facts.

[4] See http://www.mysentient.com/, http://www.cyc.com/, and http://www.cnlp.org/

3.1 Architecture

MySentient Answers is based on the *S-Core* architecture, which integrates disparate components into a uniform XML-based interface. Components each receive a *storybook* giving the history of the interaction, and their output is appended to the appropriate element. This design gives some of the flexibility of a blackboard architecture, yet allows some powerful simplifying assumptions.

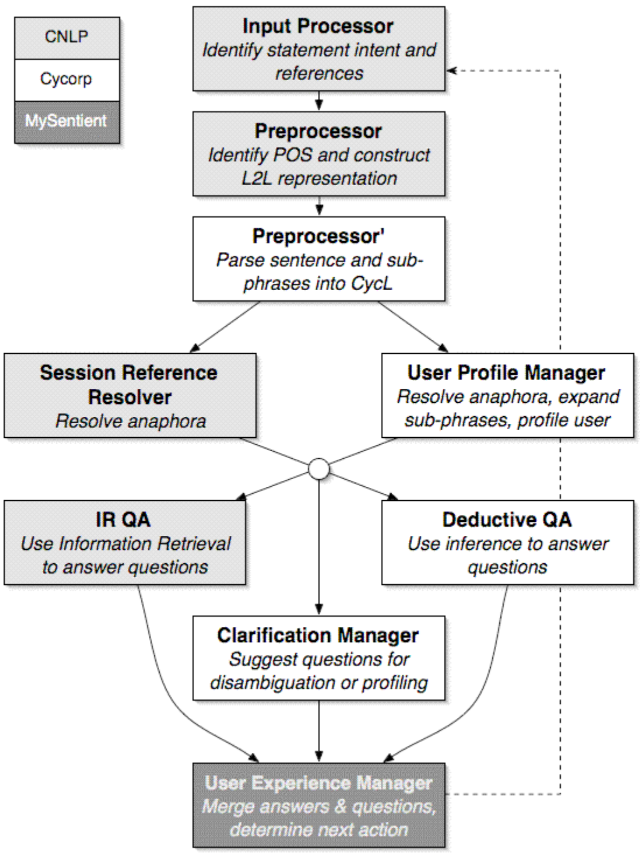

Figure 1: MySentient Answers system architecture, showing selected components.

3.2 Question Answering Modules

The S-Core architecture allows any task to be attempted in parallel (or in sequence) by multiple competing modules. In MySentient Answers, there are several question-answering modules: various NLP passage-retrieval modules developed by CNLP, and a deductive question-answering module based on Cyc. All QA modules return NL answers.

CNLP's question-answering capabilities are grounded in a quasi-logical representation – Language-to-Logic, or "L2L" that has proven successful in recent TREC question answering tracks [Diekema, *et al*., 2000].

3.3 Methodology

The Cyc Knowledge Base is essentially open-domain, but deployments of MySentient Answers will be focused on specific domains that reflect the needs and interests of the customer. The final goal of the project is that the customer's domain experts will perform much of this specialization, using a suite of MySentient Authoring Tools. At the current stage of development, these tools are in a rudimentary state; therefore, much of the authoring done in support of the results described in this paper has been simulated by blending prototype tools with the intervention of skilled ontologists. This simulation not only provides a proof-of-concept of the run-time question-answering system, but also permits an informed comparison to be made between the representation capabilities required and the feasible capabilities of the planned authoring tools (see Section 8, "Current Limitations and Future Directions," below). The corpus used for this simulation was provided by the Motley Fool UK, based on its website.[5]

While coverage of a corpus must start with the corpus itself, it is also necessary to concentrate on test queries; Cyc's coverage of the Motley Fool domain therefore had two foci.

The corpus was subjected to automated analysis for noun phrase identification and interpretation, plus extraction of glossary entries. A manual pass of review and correction to ensure broad coverage of the domain followed this.

Known and blind question sets were prepared (by both Cycorp and MySentient) based on the corpus. The known question sets were analyzed by question type (see Question Types of Interest), and strategies for broad coverage of each question type were devised and implemented.

Independently, the quality-assurance team performed daily tests based on the question sets. Results were evaluated on a primitive basis by automatic comparison with a growing set of input/output pairs. Each input/output pair was classified as correct, incorrect, or correct but badly paraphrased. Incorrect results were reviewed with a focus on identifying and resolving the broad class of defect (such as missing or erroneous knowledge) rather than fixing problems specific to particular questions.

The intensive ontological engineering effort for the Motley Fool UK domain was performed over a four-week period, and took 691.25 person-hours. The source corpus was equivalent to about 200 pages of text and a total of 286 test questions were prepared for that domain. The NLP-based components also underwent a training process against the Motley Fool corpus; however, this training was done independently of the simulated authoring/ontological engineering effort done for Cyc. As a limited test of how MySentient Answers benefits from integrating both NLP and deductive approaches to question-answering, MySentient prepared 132 questions that were posed to the system, and for each question, each QA system was scored on whether it produced a satisfactory answer. In borderline cases, a half-point was awarded.

Overall, the multiple CNLP QA modules scored 63% and the Cyc DQA module scored 34%. This asymmetry is to be expected because of the relative maturity of NLP systems in the QA domain. The federation of QA modules (taking the

[5] "The Motley Fool" is a registered trademark of Motley Fool, Inc. See http://www.fool.co.uk/ for the corpus website.

high score for each question) scored 79%, a significant improvement over the individual QA modules. In several cases, both modules gave usefully different answers that, taken together, form a rounded answer to the user's question. Two interesting examples are:

In response to "How do I protect myself from credit card fraud?" the CNLP module returned a passage that described online fraud guarantee and internet delivery protection, whereas the Cyc module returned sentences advocating PIN secrecy, comparing receipts, and reporting credit card loss.

In response to "Should we get married or live together?" the CNLP module returned a passage about the legal rights accorded to married couples, whereas the Cyc module returned sentences describing the economic benefits of cohabitation and the tax benefits of marriage.

It is important to note that this test does not isolate all federation factors. In particular, the Cyc-based DQA uses on upstream CNLP modules for the identification of noun and verb phrases, while the CNLP QA modules make use of Cyc-derived expansions. Nevertheless, the limited test described supports the view that the use of deductive question answering in tandem with a NLP QA system can significantly boost system effectiveness.

4 Discourse Modeling

Cyc's contributions to MySentient Answers are grounded in a discourse model, generated on the fly. This model stores information from a user's session, in CycL, so that Cyc can reason over it. Each discourse model is associated with a microtheory structure that is defined for each user and can be extended across sessions to preserve useful information about the user and his or her interactions with the system.

The most general microtheory in the structure is the user profile, which contains data intended to persist between sessions, and so is available for inference any time the user logs in. Though not currently a mature feature of the system, the user profile gathers information that can be used to improve the quality of subsequent interactions. For example, if the user asks for recommendations for Mazda truck accessories during one session, and later asks for directions to service stations for "my vehicle," the system should be able to use the knowledge, gained during the previous session, that the user has a Japanese vehicle. These features are still prototype technologies, and are therefore described in the "Current Limitations and Future Directions" section of this paper. The ability to profile the user is an exciting distinguishing feature of the MySentient system.

For NL parsing and generation, the user-session model includes a lexical microtheory sub-structure that was originally developed for and used in the DARPA-funded, Cyc-based KRAKEN knowledge-acquisition system [Panton, *et al*., 2002]. The most general microtheory of this sub-structure is a user-specific lexicon, from which the contents of the appropriate general Cyc lexicon are visible. Though British English is the assumed default language for the test domain, the system is can determine the appropriate language on a per-session basis. Once that language is determined, an assertion is added to the Cyc KB linking the user-specific lexicon to the general lexicon for that language. So for British English, Cyc will generate a sub-context link from the user-specific lexicon to #$BritishEnglishMt, making its data about British spellings, common words, etc., available. For more information on the representation and use of lexicons in Cyc, see [Burns and Davis, 1999].

Directly below the user-specific lexicon are two more lexical microtheories, which are used for inference by Cyc's parser and NL generation. By design, the parsing and generation microtheories are siblings; user-specific lexical information, such as information about how the user referred to a concept, is stored in the user-specific lexicon, where it is available for both parsing and generation.

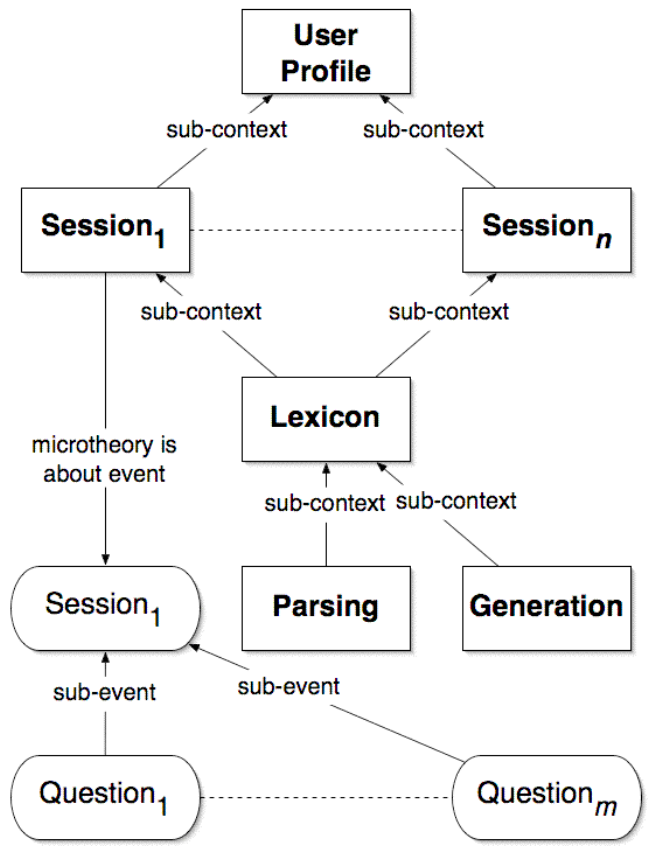

Figure 2: Part of discourse model. Boxes show microtheories; ovals show events. Not shown are user-independent super-contexts and other ontology.

Between the user profile and the user-specific lexicon are one or more session microtheories, containing the vast majority of user-specific assertions. Sessions themselves are explicitly represented as events, allowing their internal, temporal structure to be modeled. Each session revolves around the user asking one or more questions; these question-asking events are also explicitly represented as proper sub-events of the session to which they belong. Information about the user's question, such as the grammatical features of the constituent phrases (e.g., "market indicators" is a bare plural expression, "the stock market" is definite singular), the CycL semantics for the question (when available), and hypotheses about what the question is about, are all related to the user's question-asking event, using special discourse modeling vocabulary that can be leveraged by the Cyc inference engine to support query expansion, deductive question-answering, and natural language generation. Those processes are described in the following sections.

5 Query Expansion

Query expansion is the process of altering an input question, or a (quasi-)formal representation thereof, typically by adding or replacing terms. The modifications a query undergoes during the expansion process is determined by analysis of the query terms. *E.g.* "AIM" might have the *expansion* "Alternative Investment Market." Expansions can be used to focus a query, often by contributing to a set of query words, or the categorization of its answer-type. The most common approach to query expansion seen in the literature is to leverage a dictionary-based program, such as WordNet, to produce syn-sets, hypernyms and hyponyms [Hovy, *et al.*, 2000], or to find appropriate, hard-to-predict part-of-speech variations for noun compounds ("attorneys general," and not "attorney generals" as an expansion of "attorney general") [Bilotti, 2004], or to find stemming information for query-words [Bacchin and Melucci, 2004].

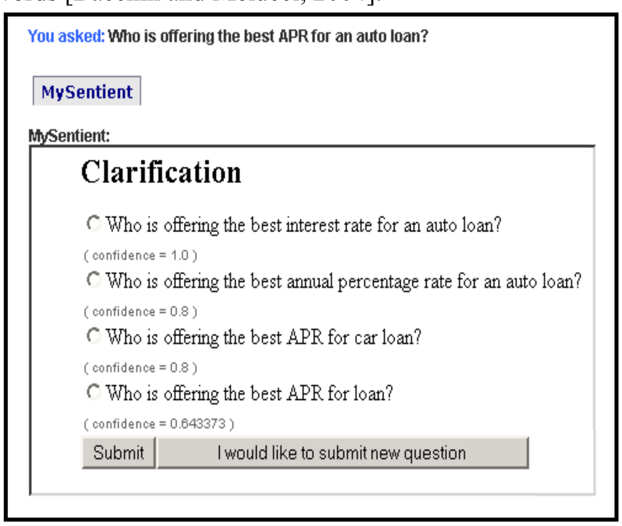

Figure 3: MySentient's Clarification module uses expansions to suggest re-phrasings of the user's question.

A limiting feature of these approaches is a near-complete reliance on lexical methods: Only the relationships between *terms* are considered; the *semantic* relationships between *concepts* are not.[6] MySentient's expansion-generation is a departure in this regard. Key phrases in the user's question are identified, translated into CycL, and placed in the discourse model. The User Profile Manager then reasons over this formal representation of the *meanings* of query-words to identify concepts that form the semantic basis for expansions. Other modules, such as the Deductive Question Answering Module and the Clarification Manager can also use these phrase-level translations.

Several expansion strategies are explicitly represented in the Cyc Knowledge Base, each defining criteria that a concept must meet to be used as the semantic basis for expansions. When a strategy is executed, inference seeks con-

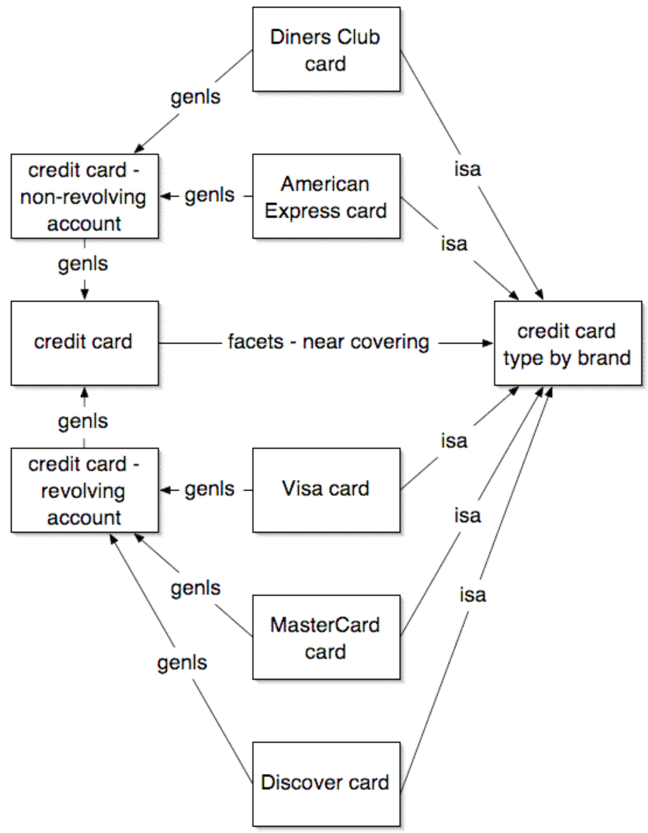

Figure 4: Part of Cyc's credit card ontology with faceting. The rightmost node is a second-order collection; all other nodes are first-order collections.

cepts that meet the relevant criteria. The resultant bindings are then sent to a Natural Language generation function that generates NL for those bindings. The generated strings, along with the strategy used and a confidence, are outputted by the User Profile Manager as proposed expansions for the original input term.

Cyc is agnostic as to how the expansions should be used; for example an NLP question-answering module might treat them as conjuncts in a Boolean rewrite, or they might be used in answer-type classification. Empirical evaluation of various strategies — based on CNLP's determination of their usefulness for passage-retrieval by their technology, and MySentient's exacting wall-clock performance criteria — led to the decision to include two strategies in MySentient Answers 1.0: A synonym-generation strategy that calls

[6] At least not directly. Some techniques (*e.g.* LSA) approximate semantic "closeness" by measuring co-occurrence in a corpus. Semantic closeness, however, is not a first-order semantic relationship; the authors contend that the semantic relationships that explain semantic closeness are more valuable for expansion.

upon the Cyc Lexicon to simulate traditional NLP-based query expansion[7], and a "classification" strategy[8] that uses definitional, or "type" information to generate expansions useful for categorization. Given an input string, "APR," the User Profile Manager uses the synonym-generation strategy to return the unabbreviated "annual percentage rate," and the classification strategy to return the more general "interest rate." Although the use of expansions by the NLP question answering module is not visible to the end user, the results of expansions are nevertheless discernable in MySentient Answers 1.0: MySentient has implemented a simple Clarification module prototype that substitutes high-

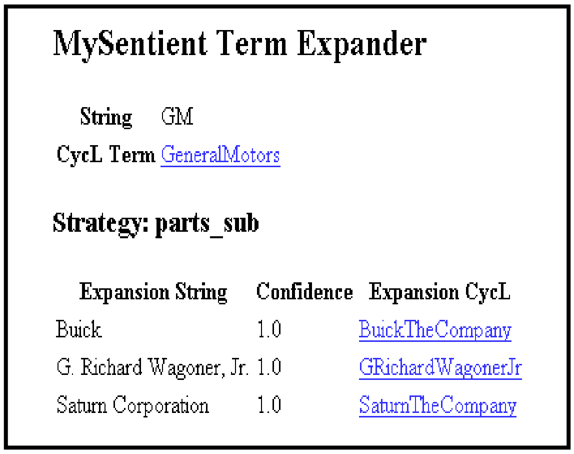

Figure 5: For "GM", we get the CycL term #$GeneralMotors. The "parts" of this term include two sub-divisions and the CEO.

confidence expansions into the original query, and presents them to the user as proposed re-phrasings. For example, if a user asks, "Who is offering the best APR for an auto loan?" the Clarification manager will offer as re-phrasings, "Who is offering the best interest rate for an auto loan?", "Who is offering the best annual percentage rate for an auto loan?" and "Who is offering the best APR for car loan?"[9]

As noted above, other Cyc-based expansion strategies are available, but are turned off by default. Nevertheless, these are worth describing as examples of how the space of possible expansions is extended through a semantic approach. Among these strategies is a "conceptually related" algorithm that finds closely associated concepts, using significant semantic relationships. For example, asked to expand "asthma," the conceptually related strategy will return "lung" because asthma is known to be a specialization of *lung disease*, and lung diseases are ailments that affect the lungs. This same strategy will also return "medical insurance," because medical insurance provides coverage for medical problems, and asthma is a kind of medical problem.

Another strategy is the "specializations" strategy, that, when given a term that maps to a collection, will return salient specializations of that collection. For example, given the input string "credit card," this strategy will return the names of the various brands of credit card, such as "VISA," "MasterCard," "American Express," and "Discover." Cyc draws on the knowledge that the collections representing these cards are all instances of a higher order collection, #$CreditCardTypeByBrand, that facets #$CreditCard by the various brands. By restricting the search to collections that are part of a faceting hierarchy, the strategy is able to avoid returning less helpful specializations of #$CreditCard (e.g., "stolen credit card," "credit card printed at a factory in London,") that the system might know about, but are more or less arbitrary sub-collections, and not part of a more intuitive conceptual hierarchy.

Finally, Cycorp has implemented two parts-based strategies, *parts_super* and *parts_sub*, that use Cyc's knowledge of the structure of types and particular individuals to return expansions that, in the *parts_super* case, reflect that concepts placement in a structure, and in the *parts_sub* case, reflect that concept's internal structure. For example, given the input string "GM" and using the *parts_sub* strategy, Cyc returns "Buick" and "Saturn Corporation," two subdivisions of General Motors, as well as "G. Richard Wagoner, Jr.", the current CEO of GM.

6 Deductive Question Answering

Like the User Profile Manager, the Cyc-based Deductive Question Answering module (DQA) uses explicitly represented strategies. The highest-confidence strategy queries the knowledge base with a CycL representation of the user's question. As such, this strategy depends on the total success of the Natural Language Preprocessor module (based on the parsing technology described in [Panton, *et al*., 2002]), in mapping English to CycL. In cases where syntactic or semantic ambiguity in the user's question results in competing CycL interpretations, simple heuristics (such as preferring the least complex CycL expression) are used to rank the interpretations. The top-ranked interpretation is identified in the discourse model as the default interpretation of the user's question, while the other candidates are recorded as possible interpretations, available for later clarification.

The DQA module retrieves that CycL interpretation and uses it to query the Cyc Knowledge Base. (If there are no sentential interpretations, DQA moves on to the next strategy.) In Cyc, a query consists a CycL formula and several query-properties, including: a *microtheory*, or context, from which to ask the query; the *temporal index and granularity* (do we want bindings that satisfy the formula now, ever, all

[7] The synonym strategy is an exception to the rule that strategies first identify relevant CycL terms and then paraphrase each.; instead, all synonyms are generated from a single CycL term that is the best interpretation of the user's phrase.

[8] The classification strategy differs from other strategies in that it uses a semantic closeness metric to assign confidences to its outputs. Thus as an expansion of "auto loan," "loan," being closer in Cyc's generalization hierarchy, gets a higher confidence than "obligation," a more distant (and abstract) generalization.

[9] The agreement error reflects the simplicity of the Clarification module's current substitution algorithm. That the input phrase "an auto" has an indefinite article is recorded in Cyc's discourse model; the module can be "smartened" to use this information.

the time, *etc.*?); a limit on the number of *transformations*, or inference steps that chain rules; and a *time* limit.

For DQA queries: the microtheory is the user's session; the temporal index and granularity are "any time", allowing the system to find temporally-qualified answers; the number of transformations and the like are determined by the nature of the question (primarily the main predicate); and the time is distributed from a (configurable) 30 second budget.

6.1 Question-types of Interest

The problem of parsing arbitrary English to a formal, logical representation is only partly solved. Thus a deductive question answering system that accepts arbitrary NL input will necessarily be limited both in the expressiveness of the formal language (the vocabulary), and in inherently difficult problems in resolving quantifier scope, negation, implicature, and context-sensitivity. At the same time, IR and passage-retrieval systems are limited by their lack of understanding: Even systems with the ability to classify a question as "about" a type, or as falling into a certain, common class, are fragile in some areas. Such systems are unable to handle questions that require comprehension of the relevant document corpus; though such systems can often return passages that contain an answer to the user's question, many questions do not have answers encapsulated by a particular passage in the text, but are nevertheless answerable from the content contained within the entire document set.

Given these restrictions, the Cyc-based Deductive QA module was optimized for questions that, given the corpus and test queries, appear representative of prevalent question

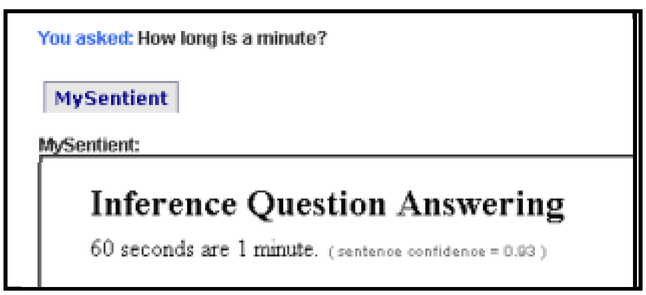

Figure 6: DQA answering a commonsense question not covered in the document corpus.

types. Significantly, the cost of targeting DQA to handle such question types is amortized by its reusability. This is in distinction to NLP-based QA systems, which generally need to be re-trained from scratch against a new corpus.

Commonsense, Off-topic Questions

In general, IR and passage-retrieval systems are limited by the corpora they draw upon in answering questions. While answering (sometimes difficult or technical) questions relevant for the domain defined by the corpora, they can appear quite intelligent or insightful. However, such systems are, by their very nature, easily "gamed" by users who wish to disabuse an otherwise sympathetic audience of the notion that the system is genuinely smart, or capable of understanding what's being asked.

Classic examples include questions that are easily answered by any intelligent agent that can understand the meanings of the words involved, such as "What colour is a blue car?" or simple general knowledge like "What is an amoeba?" and "How long is a minute?" – questions that Cyc has the knowledge to answer. Though Cyc cannot answer every conceivable commonsense question, its ability to apply common sense to both in-domain and out-of-domain problems is expected to give MySentient Answers the general look and feel of intelligence.

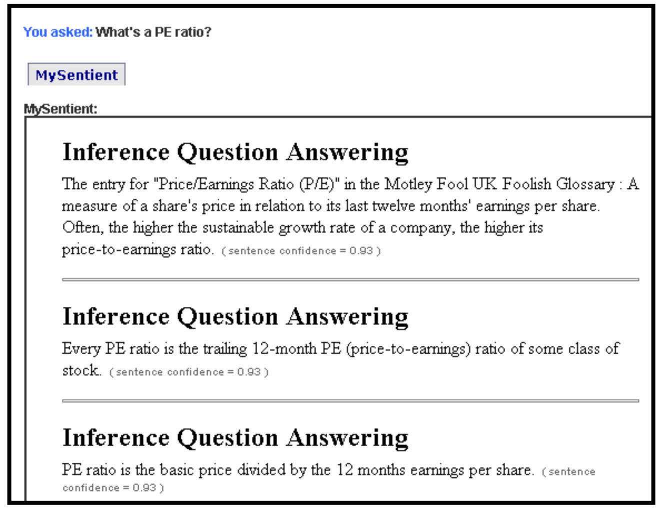

Figure 7: Cyc's DQA module gives answers for a definitional question both from a glossary, and from a formal representation of the concept.

Definitional Questions

Questions in this category are of the familiar, "What is …?", "Who is … ?" "Define …" "What can you tell me about …?" variety. Though many corpora, including the Motley Fool UK corpus used to develop MySentient Answers, include glossaries of important concepts, in general a passage-retrieval system will only succeed in reliably returning glossary entries if one of the two following conditions hold: 1) the glossary entries are formatted so as to contain "tip-off" key-words or phrases (e.g., "APR *is defined as* …") or distinctive formatting (e.g., a bolded entry followed by a colon) that the system is trained to recognize; or 2) the question-answering system is sufficiently permissive in what it will return, that any passage (including the glossary entry) that contains the relevant query-word will be picked up.

Cyc's question-answering strategy for definitional questions is to infer good answers using the predicates #$definitionalDisplaySentence and #$interestingSentence that relate statements to some of the concepts that they are centrally about. The Cyc Knowledge Base contains a handful of general rules that allow Cyc to return sentences constructed from definitional predicates, such as #$isa and #$genls, as well as others. If Cyc has the glossary entry for a term, it will use these rules to construct an "interesting sentence" for that term that includes the glossary entry.

Taxonomic Questions

Taxonomic questions are those that ask the system to identify sub-types, or sub-classes of a focal concept. For example, "what are the different types of bank account?" is answered by producing a list or hierarchy of bank account-types. Such questions are relatively easy for any ontology-based deductive system, yet somewhat difficult for a system that relies solely on IR or passage-retrieval techniques.

Procedural Questions

In many document corpora, "recipes" for achieving a goal are not condensed into a single passage or even article. Often, process knowledge is spread across a corpus, or portions of it are not made explicit, left to the reader as an exercise in small-step inference. Under such circumstances, procedural questions can be difficult for a passage-retrieval system to answer.

The DQA module succeeds in these circumstances, drawing upon Cyc's hierarchy of event types and vocabulary for describing the structure of events. The CycL predicate #$properSubEvents identifies the top-level sub-events of an event, and temporal relations such as #$startsAfterEndingOf and #$startsNoEarlierThanStartingOf describe the order of sub-events. Each sub-event can have its own internal structure in similar fashion, allowing for a recursively constructed, arbitrarily deep representation of an event.

In Cyc, processes and their stages are represented as collections of events, so that process knowledge is represented with rules concluding to #$properSubEvents and the temporal ordering predicates described above. Specialized "rule macro" vocabulary allows for a compact representation of these often complex rules, making efficient reasoning about processes possible.

When asked a procedural "How do I …?" question, the NL-to-CycL parser identifies the collection of events (the process) referred to. The DQA module then asks the Cyc KB for all top-level stages in temporal order. The result is a

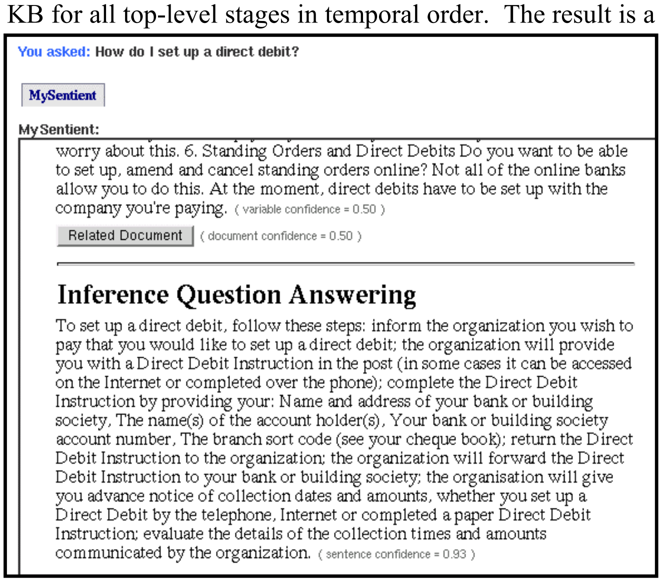

Figure 8: DQA answers a "How do I ...?" question with a step-by-step procedure.

fully bound CycL sentence that relates a process to this list, which is then paraphrased appropriately in NL. The output is a step-by-step description of the process.

Cost/Benefit Questions

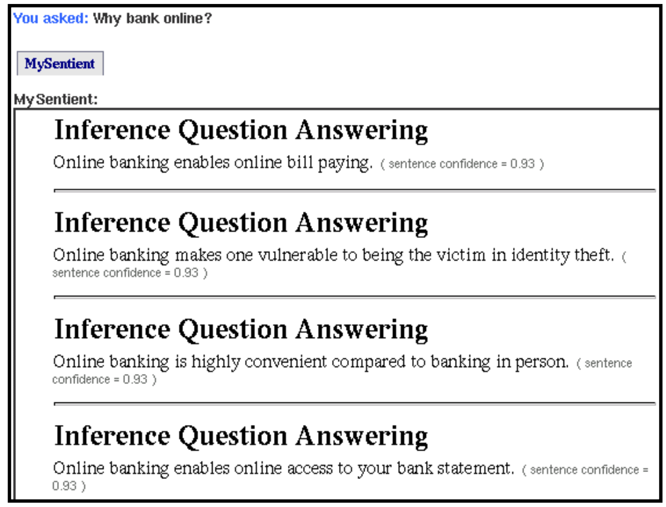

Figure 9: In answer to a "Why" question, the DQA module lists possible costs and benefits.

Questions in this category often take forms such as, "Why should I …?" or "What's the best reason to …?". Again, a passage retrieval system will do well on such questions insofar as the corpus contains explicit FAQ pages or articles with helpful headings or titles such as "Why should I …?" Even then, either the question-answering system must have received just the right input (e.g., a "Should I X or Y?" input to get back a passage entitled "Should we X or Y?") or else contain an internal table of equivalent phrasings (e.g., a question of the form, "Should I X or Y" is answerable by any passage that contains, "Why would I X over Y"). In either case, unless a passage contains some sort of explicit header or clue as to its relevance to this area, it will be passed over by a passage retrieval system.

In the DQA module, such questions are handled by asking Cyc for CycL sentences that are salient for consideration in a cost/benefit analysis of a given action type. Using the higher-order features of CycL, these sentences are inferred from assertions that identify certain *predicates* as relevant for cost/benefit analysis. For example, the assertion:

(#$costBenefitPredForSitType #$typePromotesRisk
#$Event #$doneBy 1)

tells the inference engine that the relation #$typePromotesRisk is relevant in a cost/benefit analysis of "doing" any type of event. (The "1" is the argument position of the event-type in the #$typePromotesRisk sentence.) Thus, if the user asks, "Should I bank online?", Cyc tries to prove a sentence of the form:

(#$typePromotesRisk #$OnlineBanking …)

For example:

(#$typePromotesRisk #$OnlineBanking
#$performedBy #$IdentityTheft #$victim)

which means that "performing" an online banking event increases ones vulnerability to being the victim of identity theft. Upon proving such a sentence, it is returned as a binding for the original query, and paraphrased into English.

6.2 Alternative DQA Strategies

As noted above, the general problem of parsing arbitrary English into inference-friendly CycL has not been fully solved. As such, the discourse model will not always contain a CycL translation of the user's question; indeed, for unfamiliar or complicated question-types, this will frequently be the case. Also, even when a CycL interpretation is available, there is no guarantee that the knowledge needed to answer the question is in the Cyc Knowledge Base. Thus, in order to be as effective as possible, the DQA module has been designed to reason from incomplete knowledge: it can invoke a number of lower-confidence question-answering strategies that do not depend on the total success of NL-to-CycL parsing. Because these strategies operate from a state of less information than the primary question-answering strategy, they are necessarily more brittle – specifically, more prone to returning inappropriate (but factually correct) answers. Nevertheless, these strategies provide some level of robustness against parse-failure or unanticipated discourse modeling problems, and have indeed resulted in a general increase in coverage over the target question-set. These strategies are described in order of the level of discourse model information required for them to apply, from most to least:

Topic-based Responsiveness

This strategy uses partial parse information, based on the translation into CycL of identified key phrases from the user's query. Where two or more phrases have been successfully assigned Cyc semantics, this strategy searches for interesting or informative links between them. For example, if asked a question from which only "LSE" and "AIM" are understood, Cyc would return a sentence about their relationship, such as "The Alternative Investment Market is a junior market to the London Stock Exchange."

Interesting Sentences about Terms

Like the Topic-based Responsiveness strategy, this strategy also works by reasoning over the CycL semantics of phrases from the query. This strategy, however, is more broadly applicable (and so of lower confidence), using individual query phrases and #$interestingSentence reasoning to return summary or definitional information for each phrase.

Glossary-driven Sub-string Matching

Unlike the other backup strategies, Glossary-driven Sub-string Matching does not require that any part of the question be parsed. If a query matches the title of a slurped glossary entry, then that glossary entry is returned as the answer. This strategy thus guards against knowledge gaps in the Cyc KB (*e.g.*, "Free Float" is not represented in the Knowledge Base, but the Motley Fool UK glossary entry for that term is), as well as unanticipated parser failure for definitional questions that would otherwise return glossary entries using standard question-answering methods.

7 Natural Language Generation

For a system that uses a formalized representation of the input question and performs deduction against a knowledge base whose content is also represented formally, the problem of presenting the results of a query to the end user in a readable and useful way is especially difficult.

Many systems that perform deduction will typically reduce the problem to that of generating natural-looking NL from the bindings that the inference engine returns in response to an open query. Others will attempt to augment this process by providing additional "context" – terse passages of relevant text, links to web-pages relevant to some of the entities returned as bindings, or general information in the knowledge base about those entities [Vargas-Vera and Motta, 2004; Breck *et al*., 1999].

The generation of English answer-text from the results of deductive question answering is handled in MySentient Answers in a very different way. The answer-text generator has access not only to the variable bindings, but also to the proof tree produced by the Inference Engine, and hence to the supporting assertions in the KB. This allows for more informative and nuanced presentation of inference answers.

For instance, given the query "Who are the officers of Martha Stewart Omnimedia?" the Inference Engine finds two bindings: Susan Lyne and Martha Stewart.

Rather than somewhat misleadingly presenting these two bindings without qualification, the answer-text generator inspects the inference datastructures to find that the "Susan Lyne" answer is supported by the following line of reasoning:

1. Susan Lyne is asserted to hold the position of Chief Executive Officer in Martha Stewart Omnimedia. This assertion is in a microtheory whose contents are temporally qualified to hold during the time period from November 11, 2004 through the present.
2. *CEO* is a specialization of *Officer in Organization*.
3. If someone holds a specialized version of some position in an organization, the person may be concluded to hold the more general position.

Of these supports, (2) and (3) do not mention the binding "Susan Lyne," so Cyc chooses to present (1), and passes it to the CycL-to-NL paraphrase module. This module is independent of the explanation-generation module, and is simply tasked with rendering a CycL sentence into English. It uses the Cyc Lexicon (part of the Cyc KB), which contains mappings from atomic concepts onto names and lexical entries, and phrase-generation templates for functors. These templates are recipes for the compositional construction of natural language phrases (not strings) that have syntactic and semantic information. This information permits grammatical manipulation, such as tense, agreement or sentential force (*e.g.* question or statement.). Once the phrase is built, a string is generated from it, and returned.

For the Susan Lyne assertion, this module uses the temporal qualification on the assertion's microtheory to generate the adverbial phrase "since November 11, 2004" and to assign present perfect tense to the head verb, producing this sentence:

Since November 11, 2004, Susan Lyne has held the position of chief executive officer in Martha Stewart Living Omnimedia.

Using a similar approach, the following answer text for the "Martha Stewart" binding is produced:

From 1998 to March 15, 2004, Martha Stewart held the position of corporate president in Martha Stewart Living Omnimedia.

Thus the answer text includes not only the bindings found, but also the temporal qualification for each and the specific position held, while omitting more general, less pertinent facts and rules used to reach the conclusions. Furthermore, it does so using general principles for determining the best support to show, and existing KB assertions and paraphrase functionality.

8 Current Limitations and Future Directions

MySentient Answers 1.0 is a fully functional question-answering system, but certain areas require further development before full integration. These include clarification, anaphora resolution, external knowledge sources, and authoring tools.

8.1 Clarification

As described above, the MySentient Answers system includes a module to generate clarification questions that suggest replacement questions from the expansions generated by the User Profile Manager. Use of the Cyc KB and inference engine will allow the system to not only generate more sophisticated questions for the user, but also solicit useful information about the user. As presently envisioned, this falls into the following types:

Term-Level Disambiguation: This type of question seeks to disambiguate a term (typically a Noun Phrase) from within a question.

What did you mean by "IRA"? ...

Sentence-Level Disambiguation: This type of question seeks to resolve ambiguity at the sentence level by suggesting replacements for the entire question.

What did you mean by 'Can I get a mortgage and rent the house out?'? ...

Precisification: This sounds very similar to Sentence-Level Disambiguation but is subtly different in both implementation and effect. This attempts to take a (possibly answerable) question, and suggest more precise forms for it. The new questions can not only help QA systems find answers, but will allow them to filter out irrelevant ones.

How big is Afghanistan? →
Which of the following questions did you mean to ask?
What is Afghanistan's population?
What is Afghanistan's gross domestic product?
What is Afghanistan's land area?

Topic Redirects: This suggests potentially relevant information sources. It is intended that the author can suggest key topics, and relevant resources (with associated URLs).

Are you interested in sellers of mortgages?

Interview Questions: These use Cyc's knowledge base to determine what sorts of information about discourse entities is commonly available and important to know in order to induce relevant questions:

What breed is your dog?

Such interview questions are intended not only to make it easier for QA modules to answer the user's question, but also to gather profile information about the user. This technology is based on the Salient Descriptor first developed for the KRAKEN system as part of DARPA's Rapid Knowledge Formation (RKF) programme [Witbrock, *et al*., 2003].

8.2 Anaphora Resolution

Cyc's discourse modeling enables the system to make significant headway into the problem of resolving anaphoric pronouns and noun phrases, which has been recognized as a difficult and important problem in question answering [Vicedo and Ferrandez, 2000].

The anaphora resolution implemented for MySentient Answers makes the simplifying assumptions that 1) definite NPs can be the antecedents of anaphoric NPs and pronouns, and 2) such NPs refer to instances of the relevant type (so "the shark" is interpreted as referring to a particular fish, though there are contexts where it does not, *e.g.* "The shark is a ferocious predator."). Cyc's anaphora resolution proceeds by searching backward through the discourse model for the most recent possible antecedent, eliminating candidates by applying both linguistic and semantic knowledge. On the linguistic side, for example, "he," being singular, would not resolve to "us," which is plural. On the semantic side, the referent of "he," presumably a male animal, cannot be identical to the referent of "my mother," who is represented in the discourse model as a woman.

8.3 External Knowledge Sources

Cyc's inference engine has the capability of drawing information, not only from its knowledge base, but also from external knowledge sources such as databases, and structured websites [Masters and Güngördü, 2003]. Information from multiple external sources (and the KB) can be combined in one inference.

There are two main difficulties associated with use of this technology in a system such as MySentient Answers: the task of authoring the formal semantics of an external knowledge source is still time-consuming and requires extensive training; and the use of external sources, especially websites, generally makes it difficult to ensure that the system is fast enough to be responsive to the user.

8.4 Authoring Tools

As described above, the prototype system was specialized for the Motley Fool UK domain by a combination of prototype authoring tools and manual ontological engineering. It is anticipated that, eventually, the customer will perform almost all authoring. A number of authoring tools are planned to support both NLP and Cyc-based modules. Those that most directly support Cyc's rôle are:

Concept Extractor: This component processes domain-relevant documents to identify concepts (primarily noun phrases), relate them to existing Cyc terms, and conjecture type information for novel terms. This uses Cycorp's Noun Learner, developed under Phase I of the AQUAINT project.

Coverage Checker: This component uses Cyc's Knowledge Base to ensure that the terms identified by the Concept Extractor are adequately represented, and identify knowledge gaps in the form of questions. Like the Interview clarification strategy described in section 8.1, the coverage checker is based on the Salient Descriptor.

Term Lexifier: This component, allows an author to relate Cyc terms (both denotational and sentential) to their natural language representations. These mappings can be used for both parsing and paraphrase generation. This is based on emerging Cycorp technology, and early RKF experiments.

Ontology Editor: This component projects a stratified digraph onto a relevant subset of the Cyc ontology, permitting a GUI to display the graph to enable browsing of and modification to the ontology. This component ties together the foregoing components, by visualizing of the results of the Concept Extractor, allowing the user to answer the Concept Extractor's questions, and providing access the Term Lexifier. This is novel technology developed for this project.

Acknowledgments

The authors would like to thank MySentient, especially Mike Mendoza, John Kranz, Dave Wade-Stein, Scott Gosling, Sean O'Connor, Mike Krell, and Rob Halsted. Also, the authors thank CNLP, in particular, Elizabeth Liddy, Eileen Allen, Ozgur Yilmazel, Nancy McCracken, and Niranjan Balasubramanian. Finally, the authors acknowledge the other contributors to the MySentient project at Cycorp: Robert C. Kahlert, Karen Pittman, Dave Schneider, Ben Gottesman, Peter Wagner, Linda Aramil, Jennifer Sullivan, Larry Lefkowitz, Michael Witbrock, Steve Reed, Matt Watson, Jim Zaiss, Blake Shepard, Chris Deaton, Casey McGinnis, Brett Summers, Kevin Knight, Pace Reagan, Keith Goolsbey, Kathy Panton, Chester John, and Amanda Vizedom.

References

[Friedland *et al.*, 2004] Noah S. Friedland, Paul G. Allen, Gavin Matthews, Michael Witbrock, David Baxter, Jon Curtis, Blake Shepard, Pierluigi Miraglia, Jurgen Angele, Steffen Staab, Eddie Moench, Henrik Oppermann, Dirk Wenke, David Israel, Vinay Chaudhri, Bruce Porter, Ken Barker, James Fan, Shaw Yi Chaw, Peter Yeh, Dan Tecuci, Peter Clark. Project Halo: Towards a Digital Aristotle. *AI Magazine*, 25(4): 29-48, Winter 2004.

[Diekema, *et al.*, 2000] Diekema, A. Liu, X., Chen, J., Wang, H., McCracken, N., Yilmazel, O., and Liddy,E.D. Question Answering: CNLP at the TREC-9 Question Answering Track. In *Proceedings of the 9th Text REtrieval Conference*, pages 501–510, Gaithersburg, MD, USA, November 2000.

[Panton, *et al.*, 2002] Kathy Panton, Pierluigi Miraglia, Nancy Salay, Robert C. Kahlert, David Baxter, Roland Reagan. Knowledge Formation and Dialogue Using the KRAKEN Toolset. In *Proceedings of the Eighteenth National Conference on Artificial Intelligence and Fourteenth Conference on Innovative Applications of Artificial Intelligence*, pages 900–905, Edmonton, Canada, July 28–August 1, 2002.

[Burns and Davis, 1999] K.J. Burns and A.R. Davis. Building and Maintaining a Semantically Adequate Lexicon Using Cyc. In *Breadth and Depth of Semantic Lexicons*, 2(3):397–425, June 1992.

[Hovy, *et al.*, 2000] Eduard Hovy, Laurie Gerber, Ulf Hermjakob, Michael Junk, and Chin-Yew Lin. Question Answering in Webclopedia. In *Proceedings of the 9th Text REtrieval Conference*, pages 655–664, Gaithersburg, MD, USA, November 2000.

[Bilotti, 2004] Matthew W. Bilotti, Query Expansion Techniques for Question Answering. Masters Thesis, Massachusetts Institute of Technology, 2004.

[Bacchin and Melucci, 2004] Michela Bacchin and Massimo Melucci, Expanding Queries using Stems and Symbols. In *Proceedings of the 13th Text REtrieval Conference*, Gaithersburg, MD, USA, November 2004.

[Vargas-Vera and Motta, 2004] Maria Vargas-Vera and Enrico Motta. AQUA – Ontology-based Question Answering System. In *Proceedings of the Third Mexican International Conference on Artificial Intelligence*, pages 468–477, Mexico City, Mexico, April 2004.

[Breck *et al.*, 1999] Eric Breck, John Burger, Lisa Ferro, David House, Marc Light, Inderjeet Mandi. A Sys called Qanda. In *Proceedings of the Eighth Text REtrieval Conference*, pages 499–506, Gaithersburg, MD, USA, November 1999.

[Witbrock *et al.*, 2003] Michael Witbrock, David Baxter, Jon Curtis, Dave Schneider, Robert Kahlert, Pierluigi Miraglia, Peter Wagner, Kathy Panton, Gavin Matthews, Amanda Vizedom. An Interactive Dialogue System for Knowledge Acquisition in Cyc. In *Proceedings of the Workshop on Mixed-Initiative Intelligent Systems*, pages 138–145, Acapulco, Mexico, August 2003.

[Vicedo and Ferrandez, 2000] Jose L. Vicedo and Antonio Ferrandez. Importance of Pronominal Anaphora Resolution in Question Answering Systems. In *Proceedings of the 38th Annual Meeting of the Association for Computational Linguistics*, pages 555–562, Hong Kong, China, October 2000.

[Masters and Güngördü, 2003] Chip Masters and Zelal Güngördü, Structured Knowledge Source Integration: A Progress Report. In *Proceedings of the International Conference on Integration of Knowledge Intensive Multi-Agent Systems (KIMAS 03)*. Pages 562-566, Piscataway, New Jersey, 2003.