Gathering and Managing Facts for Intelligence Analysis

David Schneider, Cynthia Matuszek, Purvesh Shah, Robert Kahlert,
David Baxter, John Cabral, Michael Witbrock, Douglas Lenat
Cycorp, Inc.
3721 Executive Center Dr., Suite 100
Austin, TX, 78731USA
{daves, cyndy, shah, rck, baxter, jcabral, witbrock, lenat}@cyc.com

Keywords: Information Sharing and Collaboration, Information Extraction and Link Analysis, Question Answering

Abstract

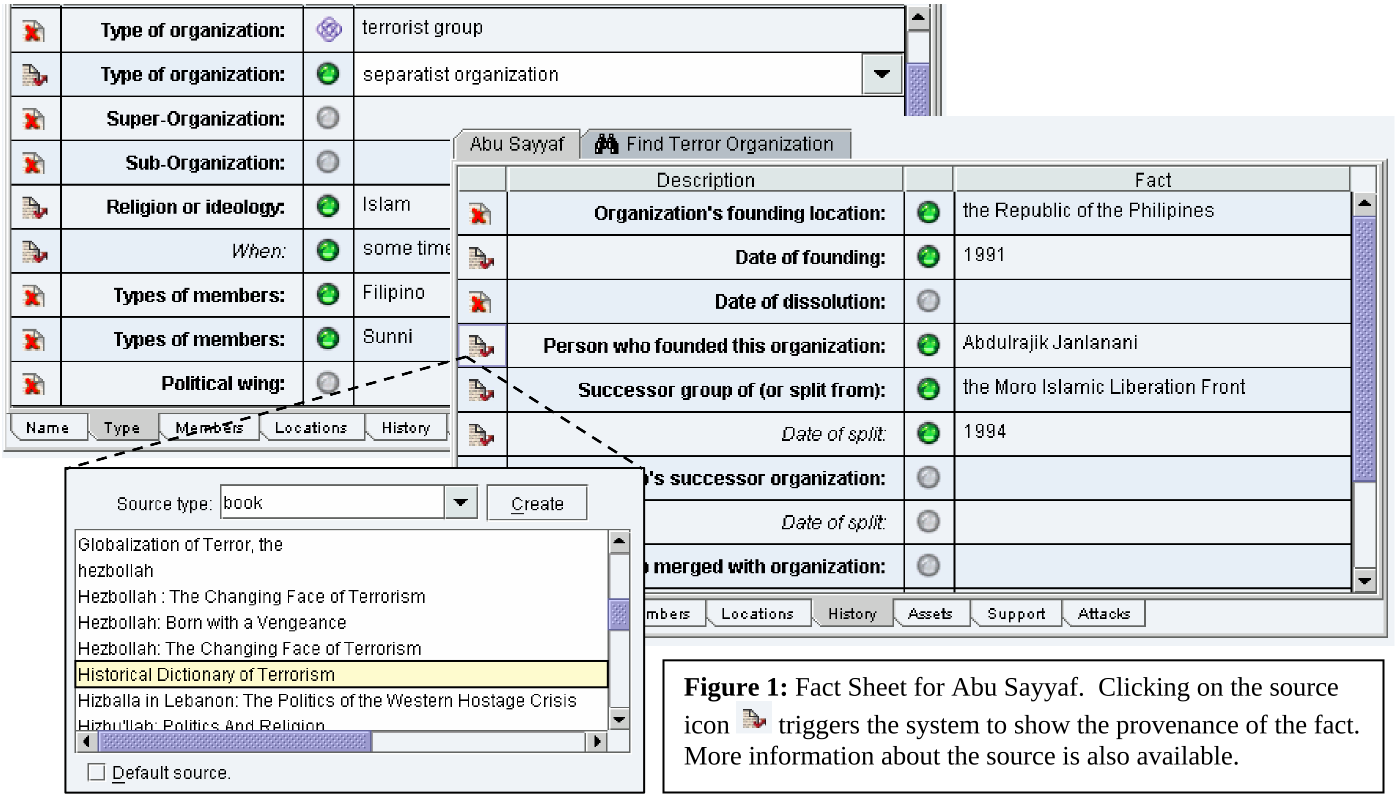
This paper presents a novel method, based on the Cyc Knowledge Base and Inference Engine, of gathering, organizing and sharing information about entities of interest (be they people, organizations, events or some other type of entity). The formal representations used in the Fact Sheets allow users to easily share information with others, run automated queries against the information, and allow the system to attempt to automatically gather and verify information before presenting it to the analyst. The system automatically keeps track of provenance (both which document a fact came from, and who interpreted the document). When gathering information automatically, the system produces a variety of search strings (using all known names for the entity) and then scours its sources for possible answers. Individual analysts can specify what types of information they are interested in for different types of entities, and can also specify additional patterns that can be used for finding that type of information. Once knowledge has been retrieved from the Web (or any other textual corpus) and ingested by the system, it is available to other analysts both for their own queries, and also to fit into Fact Sheets of their own design.

1. Introduction

This paper presents a novel method, based on the Cyc^{TM} Knowledge Base and Inference Engine, of gathering, organizing and sharing information about entities of interest (be they people, organizations, events or some other type of entity). Fact Sheets are something that an analyst could call up for a particular entity or event to see what is known about it. The Fact Sheet would display the known information, and also note where information is missing. The user could direct the system to try to find the answer, or, if an analyst knows the missing information, they would be able to enter it directly. Once the system's research has been completed, the updated Fact Sheet can be viewed and manipulated by the analyst. The formal representations used in Fact Sheets allow users to arrange and store information about entities and events, which can then be easily shared with others, can be used in running automated queries, and allow the system to attempt to automatically gather and verify information before presenting it to the analyst. The system automatically keeps track of the provenance for facts that it discovers (which document a fact came from, and where) and information entered by users (which user entered it, and what source the user got it from). In order to gather information automatically, the system produces a variety of search strings (using all known names for the entity) and then scours its sources for possible answers. Individual analysts will be able to specify what types of information they are interested in for different types of entities, and will also be able to specify additional patterns that can be used for finding that type of information. Once knowledge has been retrieved from the Web (or any other textual corpus) and ingested by the system, it is available to other analysts both for their own queries, and also to fit into Fact Sheets of their own design.

2. What are Fact Sheets?

Fact Sheets for individual entities show various facts that the system knows about the entity, along with the sources for the facts. The system was originally developed for Cyc's Terrorism Knowledge Base (TKB^{TM}) as a way to allow Subject Matter Experts (SMEs) to enter knowledge into the system quickly, and to query the system for entities or events with certain characteristics. Using this method, SMEs have been able to enter information into the TKB from relevant documents at rates of up to 100 facts per hour. The system is also keeping track of where each fact came from, and can display that information to subsequent users of the facts.



Type of organization:	terrorist group
Type of organization:	separatist organization
Super-Organization:	
Sub-Organization:	
Religion or ideology:	Islam
When:	some time
Types of members:	Filipino
Types of members:	Sunni
Political wing:	

Description	Fact
Organization's founding location:	the Republic of the Philipines
Date of founding:	1991
Date of dissolution:	
Person who founded this organization:	Abdulrajik Janlanani
Successor group of (or split from):	the Moro Islamic Liberation Front
Date of split:	1994
's successor organization:	
Date of split:	
merged with organization:	

Source type: book

Globalization of Terror, the
hezbollah
Hezbollah : The Changing Face of Terrorism
Hezbollah: Born with a Vengeance
Hezbollah: The Changing Face of Terrorism
Historical Dictionary of Terrorism
Hizballa in Lebanon: The Politics of the Western Hostage Crisis
Hizbu'llah: Politics And Religion

□ Default source.

Figure 1: Fact Sheet for Abu Sayyaf. Clicking on the source icon triggers the system to show the provenance of the fact. More information about the source is also available.

Because Fact Sheets are represented in the KB, it will be possible for users to modify them, for different users to have Fact Sheets with different types of information for the same type of entity, and for the system to suggest additional fields a user might want to see (e.g. if it is not already on the Fact Sheet for Peruvian companies, the system might suggest adding 'annual revenue', based on the fact that it knows the annual revenue of more than 10% of such companies.) Likewise, rules could be created and modified by users to automatically add new fields to the Fact Sheets for certain types of entities or events, but not for others.

Users can use the same interface to search for entities by specifying a set of constraints that needs to be satisfied. For example, when searching for terrorist organizations founded in 1991, six are returned, but narrowing the search to organizations with Islam as the religion or ideology that were founded in 1991 returns only two answers: Abu Sayyaf and Adolat, an Uzbek group.

Many other types of queries are also possible against the knowledge present in the TKB. See Deaton, et al. (2005) for more details.

3. Fact Gathering and Verification

In addition to allowing SMEs and analysts to put information into the system and perform queries against it, the system is also capable of autonomously gathering facts for relevant entities. The fact gathering system uses a multi-step process to gather and verify information from textual sources. The current implementation uses the portion of the web accessible via Google for its corpus and has a small number of possible verification techniques. It should be simple to add new verification methods and target the system at different corpora.

3.1. Fact Gathering

3.1.1. Entity Typing

The first step in gathering facts about an entity is determining what type of entity it is. For many entities (e.g. the 2249 terrorists and 810 terrorist organizations in the TKB), the system can look the name up in the Knowledge Base and determine the types from there. When the system does not already know the entity, a type must be determined. The number of base types (Collections in Cyc) that the system attempts to identify is in the thousands, but because the type system we utilize is recursively combinable, the total number of types the system could theoretically identify is infinite.

The system determines the type of an entity by looking up the name in Google, and gathering sentences from the returned documents that mention the entity. The first step is to run the sentences through a named entity recognizer (NER) to get a coarse typing (we use both Klein, *et al.* 2003, Prager, *et al.* 2000). The system then determines type-strings from these entity mentions. One method for determining an entity's type is to look for particular patterns in parses that are diagnostic of descriptions. The system uses both the Charniak parser (Charniak 2001) and the Link parser (Sleator and Temperley 1993) for this. Charniak is used primarily for post-nominal appositives like "Abu Sayyaf, a Filipino terrorist group", while the Link parser is used primarily to find pre-nominal descriptions, such as "Filipino militant group Abu Sayyaf." The resulting type-strings ("Filipino militant group" and "a Filipino terrorist group") are fed through the system's semantic parser to determine a type. In cases where the system is unable to understand the entire string, a back-off strategy is used to try to parse progressively smaller pieces of the type string, ending with the system trying to understand just the head of the type string. For the examples presented here, the types would include[1]:

Types for Abu Sayyaf:

(SubcollectionOfWithRelationToFn TerroristGroup
  groupMemberType PhillipinesPerson)
"terrorist group composed of Filipinos"

(SubcollectionOfWithRelationToFn TerroristGroup
  hasHeadquartersInRegion Philippines)
"terrorist group headquartered in the Philippines"

In cases where no tighter type can be found, the system uses the types generated by the named entity recognizer. When the types produced by Cyc are consistent with the types returned by the NER, the more specific Cyc types are used. When there is a conflict (e.g. an entity is typed by the NER as an Organization, but Cyc types it as a prime minister), the type produced by the NER is used. We will likely change this behavior in the near future, since it appears that in most of these cases, if Cyc produces a type that conflicts with the NER type, the NER has made a mistake. If the system attempts to fall back on the NER results and the NERs disagree, both types are attempted.

3.1.2. Targeted Fact Gathering

After typing an entity, the system works out what types of facts should be gathered. In the current implementation, the system determines which fields a Fact Sheet for that sort of entity would contain, and looks for those types of facts. There are currently two types of Fact Sheets: one type is manually created, while the other is induced based on facts extant in the knowledge base. The approximately thirty types of manually created Fact Sheets have typically been created for specific well-defined knowledge entry tasks.

The other way of determining what facts might be usefully searched for involves looking at similar entities and the types of information that are available for those entities. A field will be included for an entity type if at least 1% of such entities (and at least 2 entities) have that type of information specified in the KB. For example, because Cyc commonly knows members, leaders, founding dates, and sub-organizations of OrganizationOfOrganizations, these types of information are all deemed relevant when gathering facts about the United Nations. The system currently contains inferred Fact Sheet templates for 1171 different entity types.

After determining what types of facts to search for, the system constructs search-strings that can be sent to an IR engine. The system constructs a variety of search-strings using manually created templates along with lexical information present in the KB for the entities in question. For example, for the query (foundingAgent AbuSayyafGroup ?X), the system constructs 40 different strings, including:

Sample search strings for
(foundingDate AbuSayyafGroup ?X)

Abu Sayyaf was founded in ____
Al Harakat Islamiya, established in ____
ASG was established on ____

The search strings are constructed using the same infrastructure (both representation and code) that is used ubiquitously in Cyc for constructing English representations of CycL assertions. For several reasons, however, we chose not to use the standard generation templates. The first reason is that much of the target information in the corpus is not typically stated as a simple fact, but rather as a qualification or addendum to some other fact, while Cyc generation templates are generally designed to express the single facts as sentences. Thus, for founders, Cyc ordinarily generates "Abu Sayyaf was founded in 1991". Using Cyc's standard English generation for these facts would miss many of the common patterns, such as "Abu Sayyaf, established in 1991, …". Second, because Cyc typically has been optimized to convey all the details of logical representations, English generation can be quite stilted, and therefore be unlikely to be matched in a corpus.

In order to find answers for the queries, the system strips blanks and submits the result as a quoted string to the search engine (*e.g.* "Abu Sayyaf was founded in"). Upon receiving the results from the search engine, the system downloads the documents and searches through them for sentences that contain the search string.

To produce a candidate assertion, the system looks at the portion of the sentence where the blanks would have been. The system then attempts to interpret that string as something that meets the constraints set forth by the predicate. In this case, foundingDate requires that its second argument be a Date. Accordingly, the system attempts to interpret the selected string as a date, expanding the size of the window until it finds a date or reaches a pre-determined maximum length. For example, "the early 1990s" parses to (EarlyPartFn (DecadeFn 199)), while "1991" parses to (YearFn 1991).

3.2. Fact Verification

Before actually asserting anything into the Knowledge Base, the system attempts to verify the 'facts'. The system uses several different methods for verification. The first involves a KB consistency check. For example, the system can reject 'facts' because there can only be one possible answer, and the answer is already known. For this reason, the system would reject an assertion that Dallas, TX is the capital of the U.S., since Cyc already knows that Washington, DC is the capital. Previous

[1] The system currently produces additional types, and additional work will be necessary to weed out the inappropriate answers.

work by Cycorp suggests that a one-step inference is generally enough to find contradictions (Panton, *et al.* 2002). When a fact is already known, the system ceases verification. In the future this procedure will add redundant justifications to already-known facts so that the system will know that there are multiple sources that all convey the same fact.

The first verification mode serves to weed out information that is known to trivially conflict with the Cyc KB, but there are large amounts of information that would be consistent with the Cyc KB, but still wrong. For instance, the sentence (foundingAgent AbuSayyafGroup EdBuckham) doesn't obviously contradict anything in the Cyc KB, but is nevertheless false. To rule out such answers, the system performs an additional verification step by rerunning the search with the string generated for the complete fact. In cases where the search string uses an abbreviation or acronym, an additional disambiguation string is added: the least common word in the expanded acronym, based on Google hits. In this case, "Sayyaf" was be added to "ASG was founded by", thereby eliminating documents that used ASG to refer to "Alexander Strategy Group", which was in fact founded by Ed Buckham.

Once additional facts have been found by the system, analysts will be able to view those facts in the Fact Sheet, along with the sources for them. While we would like to claim that it will only produce true facts, that is not the case. Luckily, this is not an insurmountable problem; recall that the system-gathered facts will be viewed using an interface that is also used for manual knowledge entry. This will allow users to view the provenance of the assertions and remove those that are false. For example, when gathering facts about countries, the system came up with the following 'fact':

```
(sellsProductType Germany
 (SubcollectionOfWithRelationFromFn Soul-Spiritual
  possessiveRelation
  (PronounFn ThirdPerson-NLAttr Singular-NLAttr
    Neuter-NLAttr PossessivePronoun-Pre)))
```

"Germany sells its soul"

Obviously, despite its verbatim appearance in the corpus, this is not the sort of 'fact' that would withstand scrutiny, and an analyst could easily mark this fact as false, thereby removing it and ensuring that it doesn't reappear.

4. Result Analysis

For the work reported here, we limited the system to looking at no more than the first 20 hits for any search string. This was done in large part because of limitations on the number of permitted queries against the Google API.

One restriction in the current system is that, except for dates, it will only return answers that are already reified concepts in the Cyc KB. This has the effect of drastically limiting the possible results for some types of queries. For example, Cyc only knows a relatively small number of companies. As such the system will reject the employers for the vast majority of people.

In a test of twelve US presidents chosen randomly (famous people were chosen because we were fairly certain that relevant data would be available on the web), the system was able to correctly determine the date of death for seven of them, was only able to find dates without years for three more, and failed to find any death-dates for the remaining two. For birth dates, the system found the correct answer for ten, found no answer for one, and found only a date without a year for one.

There were several errors in this task, including a death date for Franklin Pierce of October 8, 1809 in addition to his actual death on October 8, 1869. This is a result of the garbage-in garbage-out principle—one web page erroneously stated that Pierce died in 1809. One solution to this problem lies in weighting the answers according to the number of times each answer was found. Even though the system looked at fewer than twenty pages, it found three that included the correct death date, and just one with the incorrect date. A search through more documents would doubtless reveal that many more documents state that he died in 1869.

We expected ambiguity to be a serious problem, but it turned out to be less of a problem than we expected. The system found an incorrect death date for only one of the presidents because of a naming ambiguity (Benjamin Harrison, who died in 1901, was also alleged to have died in 1925). The nature of the search results that Google returns, along with the fact that we tested the system using famous people, probably reduced ambiguity by biasing the system towards only seeing results that were for the presidents.

The system also spent substantial time looking for information that turned out to be very difficult to find in texts. For example, for all people, the system attempts to find out what languages the person speaks, what areas they were educated in, who they work for, what their email address is, and a variety of other things that proved difficult to find. Although the corpus and type of entities searched for were helpful in reducing ambiguity, we believe that they worked against the system when trying to find these types of information. Language spoken, for example, is not commonly discussed for US presidents in the English documents that Google returned.

Table 1 shows results for 24 FamousHumans, including the 12 presidents just discussed. A sentence was judged plausible if it was a reasonable reflection of the content of the document, and implausible if it conveyed content not clearly in the document. Many of the sentences found were already known by Cyc, but even more were novel, and the number of sentences asserted when they should not have been was relatively low (only 27% of the total unique sentences the system considered). Even though the system typically only looked through 10-20 documents for each fact (predicate/person pair), a substantial number of duplicates were encountered.

	Unique Sentences	**Total**
Plausible	96	161
Already Known	30	79
Novel	66	82
Implausible	41	67
Provably Incorrect	4	9
Wrongly Asserted	37	58
% incorrectly asserted	27% (37/137)	25% (58/228)

Table 1: Results of fact-finding for information about 17 predicates and 24 arbitrarily chosen instances of FamousHuman.

Several types of information that we initially tried to gather using specific search strings turned out to be poorly suited to this mechanism, but are well suited to other methods, including the entity typing methods discussed earlier. For example, to find the ethnicity of Abdulrajik Janiani, the system searches for strings including "____ leader Abdurajak Janjalani", which yields 19 documents, none of which yield an ethnicity using this technique.[2] On the other hand, using the entity-typing methods, the search string would be just "Abdurajak Janjalani", the 18th document contains an appositive description of Janjalani as "a Philippine Muslim", which would be enough for the system to determine not only his ethnicity but also his religion.

The system also searched for marital status, attempting to find a term that would map directly to Married or Divorced. While this technique might work well for Divorced, there are much better tips that a person is married. In particular, any mention of "X's wife" or "X's husband" is sufficient to determine that X is married, regardless of whether the spouse is mentioned by name. Clearly, a technique capable of using these types of cues would yield better results.

In an initial experiment to gauge the promise of the system, 17 instances of FamousHuman (different from those used for fact-finding above) were run through the system. The NERs produced incorrect types for two of the 17. The system found more detailed type information for nine of the 17. Accurate information was found for all nine, although two of the nine also yielded incorrect information (e.g. a prime minister was understood to be a secretary-general).

Analysis of the work done by the entity-typing system also turned up several interesting issues. Because the system uses multiple sentences mentioning the same entity, the NERs produced conflicting types for two of the 17 entities. This is clearly undesirable, and a simple voting scheme should be more than sufficient to deal with this problem. In both of these cases, there was one false typing and numerous correct typings for the entity.

Another issue arises with entities that change type over time. For example, for political figures it is quite common that they will, over the course of their career, hold several different positions. The system typed Shimon Peres as a prime minister, deputy prime minister, and foreign minister. By asserting these facts into contexts that are agnostic about time, it is possible to assert that Shimon Peres is all of these things, and this is sufficient for many uses of this information.

5. Future Work & Conclusion

Looking forward, we would like to extend this system in a number of ways. In terms of the Fact Sheet user interface, we already have the ability for users to modify the knowledge about the entities, but do not yet have the ability for analysts to modify the format of the Fact Sheets. Future interface work will focus on allowing analysts to manipulate the order and type of fields in Fact Sheets, adding additional modalities to the Fact Sheets (e.g. pictures), and dynamic updating of Fact Sheets, with notification when new information is found about an entity.

On the fact-gathering side, we are planning several extensions. One extension is the ability to gather facts from documents in Chinese, and display the results (in English) as part of a Fact Sheet. Initially, we will do this by exploiting existing Chinese-English lexica, combined with our existing English-CycL lexicon to produce a prototype Chinese-CycL lexicon. With the additional semantic constraints that we will bring to bear from the logical representation in the Cyc KB, we believe that we will have a reasonable chance of achieving a measurable level of success using this simple method, though we expect to get substantially poorer results than we get with English texts.

As mentioned earlier, the system currently accepts only known entities as possible fillers. In an intelligence analysis domain where discovering new players is important, this is clearly too strong a condition for acceptance. Instead, we will run the process described here recursively. Once a possible entity has been identified, the system will attempt to determine the type of that possible entity. If a type can be determined that is consistent with the requirements of the field it will fill, we will create a record for that entity and allow its use henceforth.

It is worth noting that entities in the Cyc KB can have multiple names (or aliases) associated with them. For example, if the system has three different entities named "Abdul", it is not obvious which of them a particular extracted fact should be associated with. In future versions, we will allow facts that might map to several entities to be asserted for all of the possible entities, along with an assertion stating that this fact is really only true of one of the entities. If any of these facts is ever used in a justification, the analyst will be alerted and encouraged to look at the source document to confirm or deny that the fact is asserted on the correct entity.

[2] This search does, however, yield many hits that contain the information that he is the leader of Abu Sayyaf.

In an effort to get information that cannot be easily gathered using the string-based fact-gathering system, we also hope to include more of the information that we can gather with our existing sentence-level parsers. See below for an example parse of a sentence retrieved from Google for Abu Sayyaf:

Correct semantic parse for
"In May 2001, Abu Sayyaf kidnapped 20 people, including three Americans"

```
 (thereExistAtLeast 20 ?PEOPLE18
   (and
    (isa ?PEOPLE18 Person)
    (thereExists ?KIDNAPPED7
     (and
       (agentCaptured ?KIDNAPPED7 ?PEOPLE18)
       (isa ?KIDNAPPED7 KidnappingSomeone)
       (perpetrator ?KIDNAPPED7 AbuSayyafGroup)
       (in-UnderspecifiedContainer ?KIDNAPPED7
         (MonthFn May (YearFn 2001)))))))
```

"20 people were kidnapped (in at least one kidnapping) in May 2001 by Abu Sayyaf."

Once facts like these have been gathered, the users will be able to expand upon the knowledge automatically gathered to include other information present in the sources that the system was unable to understand.

A natural extension to the fact gathering system described here is the integration of an existing relational information extraction system. While relational IE systems do not attempt to provide answers to all the different types of facts that our system looks for, they typically do a good job at finding some of these types of information, and we are actively looking for opportunities to integrate with such systems. The additional semantic interpretation that this system performs has the ability to eliminate some of the false positives that IE systems typically generate. Similarly, the Fact Sheet interface will allow users to provide additional information (e.g. other names for an entity) that can be used to merge some of the entities found by IE systems, resulting in better information than either type of system could achieve on its own.

One open question is how well this system will function with a different corpus, such as that which might be found in a classified environment. We do not believe that different methods of accessing the corpus (e.g. not being able to rely on page-rank to order the documents) will substantially hinder the system. In fact, because news articles tend to be very infrequent in Google's top results, working on a system without page rank may result in better results, as news articles tend to be packed with the sorts of information that our system gathers fairly well. Future tests of the system will include looking at news corpora in addition to the information returned by standard web-search engines.

We believe that this system shows substantial promise as a tool that analysts can use to gather, store, and arrange information. Because all the information is stored in a formal representation, others can readily reuse it, and any updates to that information that an analyst decides to make will be automatically disseminated to other users. The fact-gathering ability of the system, already substantial, will only increase as more methods are implemented and existing methods are extended and refined. Preliminary results reported elsewhere (Matuszek, *et al*., submitted) indicate that the verification steps described here can reduce the number of false positives by nearly 90%.

6. Acknowledgements

Production of this paper was supported by ARDA's AQUAINT program.

References

Charniak, E. 2001. A Maximum-Entropy-Inspired Parser. In *Proceedings of the 1st Conference of the North American chapter of the Association for Computational Linguistics*. Seattle, Washington, 132-139.

Deaton, C, B. Shepard, C. Klein, C. Mayans, B. Summers, A. Brusseau, M. Witbrock, D. Lenat. 2005. The Comprehensive Terrorism Knowledge Base in Cyc. Poster presented at 2005 International Conference on Intelligence Analysis.

Klein, D, J. Smarr, H Nguyen, C. Manning. 2003. Named Entity Recognition with Character-Level models. *Proceeedings of the Seventh Conference on Natural Language Learning*, 180-183.

Masters, James and Z. Güngördü. 2003. Structured Knowledge Source Integration: A Progress Report. In *Integration of Knowledge Intensive Multiagent Systems*, Cambridge, Massachusetts, USA, 2003.

Matuszek, C., M. Witbrock, R. Kahlert, J. Cabral, D. Schneider, P. Shah, D. Lenat. 2005 Searching for Common Sense: Populating Cyc from the Web. Submitted to IJCAI 2005.

Panton, K., P. Miraglia, N. Salay, R. Kahlert, D. Baxter, and R. Reagan. 2002. Knowledge Formation and Dialogue Using the KRAKEN Toolset. In *Proceedings of the 18th International Joint Conference on Artificial Intelligence*. Edmonton, Canada, 900-905.

Prager, J., E. Brown, A. Coden, D. Radev. 2000 Question Answering by Predictive Annotation. In *Proceedings of the 23rd SIGIR Conference, 184-191.*

Sleator, D. and D. Temperly. 1993. Parsing English with a Link grammar. Third International Workshop on Parsing Technologies. August 10-13, 1993, Tilburg, Germany.