Editor: James Hendler
University of Maryland
hendler@cs.umd.edu

Question Answering on the Semantic Web

Deborah L. McGuinness, *Stanford University*

Question answering on the Web is moving beyond the stage where users simply type a query and retrieve a ranked ordering of appropriate Web pages. Users and analysts want targeted answers to their questions without extraneous information.[1] These answers might contain information from current and authoritative sources, terms with the same meaning as those used in the query, relevant links such as justifications, follow-up questions fitting the context, and provenance information (author, input date, authoritativeness, source, ranking, and so on). Next-generation question-answering systems might also provide better querying support. This could include identifying whether questions are incoherent and therefore can't be answered, too general and would retrieve too many answers, or over constrained and would retrieve few if any answers. This article is the result of a keynote presentation on this topic at the AAAI Spring Symposium Series on New Directions in Question Answering.[2] This article presents a spectrum of techniques for improving question answering and discusses their potential uses and impact.

Question-answering knowledge

Question-answering improvements can focus on three areas: the content (an information repository of Web documents), the query, and the answer. Figure 1 presents a spectrum of such techniques.

Content manipulation approaches

Authors and implementers can improve content by adding meta information to help question-answering systems find good answers. Provenance is one simple information source.

Header information. Programs or users can store metainformation in the document's header information, distribute it throughout the document in markup, and/or store it in a metadata registry, such as IWBase.[3] Retrieval engines can use this header information, for example, by using recency information in a data header to determine answer order or by using authoritativeness information to choose which documents to place ahead of others.

Hot links. Documents sometimes include tagging using a shared domain vocabulary. For example, document authors can use the Sentius automatic hot-link mechanisms (www.sentius.com) or Microsoft's smart tags (http://msdn.microsoft.com/library/default.asp?url=/library/en-us/modcore/html/deoriworkingwithfactoids.asp) to automatically insert consistent and operational tags. With Sentius' dynamic contextual linking, tools automatically recognize terms from a controlled vocabulary and mark up content with hot links related to those phrases in a consistent manner. Applications can then present users with options appropriate for each phrase. This technique enhances answer presentation and provides consistent, automatic content markup.

Term-meaning markup in text. Another technique adds term-meaning markup in the text. For example, a Stanford researcher's homepage could be marked up so that her name is tagged researcher, her university is tagged employer and university, her email address is tagged emailAddress, and so on. Then, if a user searches for researchers who work for Stanford, the system would retrieve only the portion of the document containing the researcher's name. Users or agents can encode fairly expressive markup using languages such as W3C's OWL[4] or simple markup using XML.

Ontological support for term definitions. If documents are semistructured or structured, the system can derive more meaning and use notions such as domain and range of slots to check consistency. This facilitates connections with reasoners that might use inference to make implicit information explicit and aids information integration. The structured or semistructured information uses a database schema or a knowledge base of terms and their interrelationships, thereby allowing object manipulation. A database schema can be viewed as a set of phrase descriptions—classes, properties, and domain and range information. This might be stored as an ontology in OWL or any language capable of representing class, property, restriction, and individual information.

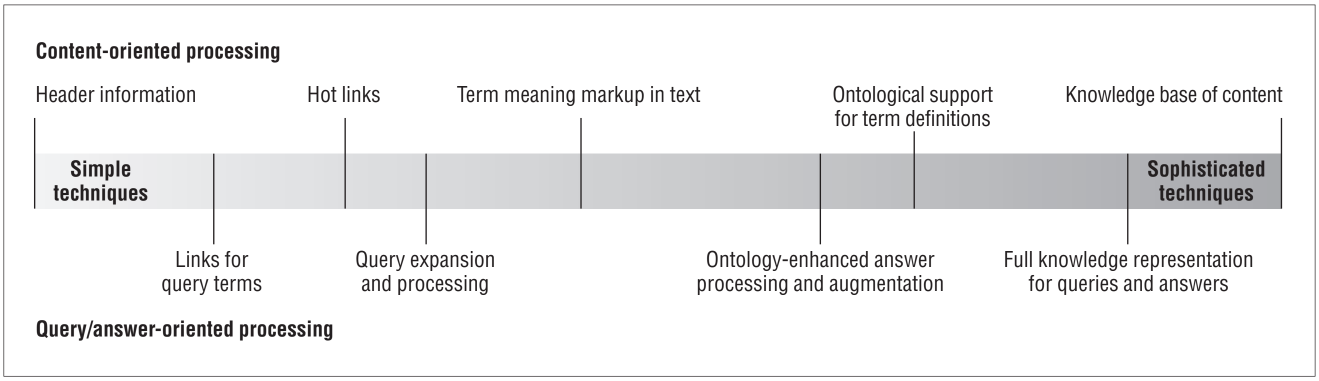
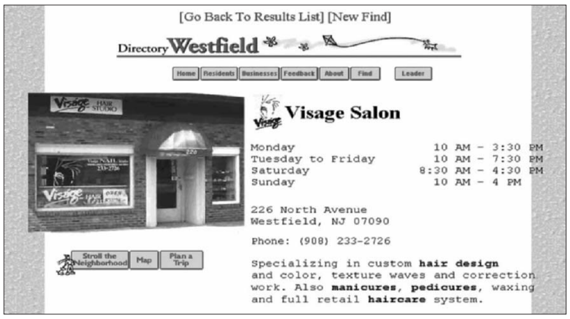

Content-oriented processing
Header information
Hot links
Term meaning markup in text
Ontological support for term definitions
Knowledge base of content
Simple techniques
Sophisticated techniques
Links for query terms
Query expansion and processing
Ontology-enhanced answer processing and augmentation
Full knowledge representation for queries and answers
Query/answer-oriented processing

Figure 1. Knowledge representation options for question answering.



Visage Salon

```
Monday                        10 AM - 3:30 PM
Tuesday to Friday             10 AM - 7:30 PM
Saturday                    8:30 AM - 4:30 PM
Sunday                        10 AM - 4 PM

226 North Avenue
Westfield, NJ 07090

Phone: (908) 233-2726

Specializing in custom hair design
and color, texture waves and correction
work. Also manicures, pedicures, waxing
and full retail haircare system.
```

Figure 2. FindUR ontology-enhanced search for electronic yellow pages.

Knowledge base of content. For every phrase recognized in the source information as a phrase in the ontology or knowledge base, authors or programs can add markup about the term's type. If the programs reference a full background knowledge base of content, instance information with extensive structure can be maintained as well. Thus, the source could have additional markup from the knowledge base. For example, text containing a mention of a particular person could have additional markup including the person's phone number, address, and so on. Many systems don't add the markup to the source information. They just maintain the link to the knowledge base containing the term (or a direct link to the term), so the question-answering system can use the information when necessary by accessing the knowledge base and the content does not require markup.

Query and answer manipulation approaches

The previous techniques manipulate information asserted and derived as source information and then use the enhanced content for retrieval and presentation. This section (and more of the author's work) addresses query and answer manipulation.

Links for query terms. This simple approach lets implementers predetermine links of preferred answers for expected query terms. This approach depends on the implementer's knowledge of typical query terms and desired answers for queries containing those terms.

Query expansion and processing. Systems can enhance a query by expanding it to include more terms than initially input to help find related answers. For example, a system can expand a query for "car" to search for the synonym "auto" and a specialization "roadster." FindUR[5] and TAP's Semantic Search[6] (http://tap.stanford.edu/tap/ss.html) use background ontologies to improve recall. FindUR, for example, improved the recall of an electronic yellow-pages application more than 300 percent while minimally decreasing precision. FindUR used a standard information retrieval system (Verity) for accessing Web pages annotated with markup information and a background ontology stored in a description logic system. FindUR dumped its knowledge base into topic sets for deployment but used an ontological-evolution environment to build and maintain the background ontologies. People literate in only the domain information maintained the knowledge base content. FindUR did not require maintainers skilled in knowledge representation techniques because it had an evolution environment built to help laypeople analyze usage and update simple ontologies. It was also successfully deployed for a long period because of its interface to standard commercial off-the-shelf technology, such as Verity. This is just one example of how a knowledge-representation-based approach could enhance a Web service question-answering offering and be used and maintained by a diverse workforce. The FindUR family of "smart search" applications was deployed at AT&T (under the name Smart Search) in a wide range of applications, from simple retrieval for newspapers and electronic yellow pages to more complex medical information system retrievals for doctors[7] and competitive research sites for market analysts with very structured queries and answers.

Figure 2 shows one answer to a query for "beauty parlors" on the Directory Westfield Web site. The small town's electronic yellow pages listed 22 beauty parlors, including limited information for each (parlor name, business hours, and a short description). Although "beauty" was supposed to be a standard industry coding term for haircutting establishments, in practice, only one haircutting listing had its content enhanced with the metatag "beauty," and only one other related site mentioned the term "beauty" in the short description. Thus, before the ontology enhancement, a query for "beauty parlors" missed most of the listings. The figure shows one beauty parlor description and highlights the terms used to find this answer. FindUR used a background ontology to expand the query before it used the Verity search engine to find matching listings. Figure 2 shows the full answer including highlighted terms that matched the expanded query. Because the listing matched on five terms or phrases ("salon," "hair design," "manicures," "pedicures," and "haircare"), it was viewed as a good match. While this is a simple and early application of question answering on the Web, it shows how a simple approach with limited background knowledge can make a significant difference in a system's ability to find appropriate answers.

Applications can use ontologies to help analyze and refine queries. Queries with terms from a background ontology can be analyzed to determine their interrelationships (as in the Interactive Market Analysis and Classification System[8]). For example, the system can determine that a query for "sports cars" is more specific than a query for "autos," assuming a background ontology containing the synonyms "auto" and "car" and the subclass relationship of "sports car" and "car." If the user enters a query that's too general, such as "auto" on a car Web site, the system can suggest subclasses of "car" to help refine a query. Similarly, an analysis of an over constrained query, such as "sports cars that cost less than $3,000," can produce generalizations such as "cars that cost less than $3,000" or "sports cars that cost less than *x*," where *x* is greater than $3,000. The system can also remember answers to queries. So, when the same or more general queries come in, the previous answers can provide a quick partial answer. The system uses the background ontology to determine which queries are more general and thus can automatically classify the queries. In addition to helping refine and classify queries, ontologies can also be used for such functions as selecting portions of answers or documents to return and helping to crawl free text to generate semistructured text.[5,7,9]

Ontology-enhanced processing and augmentation. When using this method, applications can enhance answers to include identified objects satisfying a query. Document-retrieval systems typically do not identify the portion of the document (that is, the document objects) that contains the answer. For example, today's agents or human users who ask for universities in Santa Clara County want more than simply documents including the phrases "Stanford University" or "University of California, Santa Cruz" or the university name and address. They want the identified object representing Stanford University or UCSC, along with the option of accessing object properties, such as address, county, information source, relationships between the object and other objects, such as the university's student population, and so on.

Applications can also enhance answers with optional justifications,[10] including information such as why the query retrieved Stanford, where the data came from, whether the system made inferences, how they were deduced, and so on. Inference Web (www.ksl.stanford.edu/software/iw), for example, defines and uses a portable proof markup language,[12] allowing information sources and reasoners to provide optional justifications for any answers they return in a portable, machine-understandable, and distributed "proof."[11] Then the Inference Web browser lets other programs access those proofs, also allowing humans to use a browser to view the justifications. Thus, humans or agents can access information such as author, deductive processes used, and recency to help decide if they should trust the information and how they should use it. This can be particularly useful when agents or humans receive conflicting answers and want to know which answers to trust. If humans are trying to reuse past answers, they might want to examine prior assumptions and sources before relying on the information again. The capabilities the Inference Web supports for explanations, summaries, and general content-use exposition (called "meta-information" or "provenance") are critical to both human and agent trust. They are also critical to users' and agents' ability to judge when to use and reuse answers from Web systems. This technology represents the next significant growth area for question answering on the Web.

Even simple question-answering approaches that use limited background knowledge can significantly improve a system's ability to find appropriate answers.

Another enhancement method is pruning an answer for presentation if it's too long or complicated. Users and programs can use *pruning* or *matching languages* to ask for answers when they must satisfy many constraints.[10,13–15] With this method, the language lets a user present a pattern for matching.[13] This pattern designates the structured object's portions that must be matched and the "interesting" portion of that object to return as part of the answer. Thus, agents and end users can use the language to specify both the criteria for matching and the criteria for presentation. For example, although something is relevant to the object, such as a person's address, it might not be considered something that should be presented with the answer to the user.

Implementing this approach in the CLASSIC knowledge representation system allowed users to store filter patterns with classes.[10,14] Therefore, although classes might be known to have many properties that might have values, the system can store only a small set as "interesting" and worthy of presentation in the default presentation interface. The query pattern language allows for variables in the query. Bindings for those variables are returned subject to pruning specification information stored in the ontology. Additionally, the person asking a query or a system implementer can specify instructions to override the default presentation strategy, since interesting properties can depend on things such as context and user.

To enhance answers, applications can also use background information. This information can be definitional, such as simple ontology descriptions or previously constructed Web pages for common information on the term (along the lines of the dynamic contextual linking mentioned previously). For example, if a user searches for a term known to be a performing artist, the system can supply links for the artist's preferred page as well as optional links for upcoming and local concerts, if known. Commercial search engines, such as Google, use this approach when they include selected (typically sponsored) links with answers, such as the artist's albums available for purchase. TAP's Activity Based Search (http://tap.stanford.edu/tap/ss.html) takes this a step further, adding links and content on the right side of the screen from a knowledge base containing structured information on activities the term participates in. For example, performers' activities include concert schedules, albums, posters, and biographies, and all can be provided as optional portions for hyperlinking with answers. The TAP search determines links based on activities associated with the term rather than on terms purchased by advertisers.

Full knowledge representation for questions and answers. When queries move from unstructured form to structured form, applications and end users can use full knowledge representation techniques for questions and answers. Some more sophisticated information retrieval systems expose a form of this to users. Verity's query language is one example, allowing users to look for terms in a particular field or a word or phrase within a certain distance of another. Query languages based on structural patterns, such as OWL-QL,[15] also embody this approach. You can view them as an extension of a pattern-matching method, such as the one mentioned for pruning, where a query provides a pattern used to match the content.[13, 14] One requirement for this scenario is that the content is expected to be structured—for example, encoded in OWL. So, a query could be something like "Chardonnays from Napa Valley." This query might be more precisely mapped to wines whose varietal is chardonnay and whose region is in Napa Valley. Then, a question-answering system that knows Howell Mountain is in Napa Valley will return chardonnays from Forman Vineyards, if it also knows Forman Vineyards is in Howell Mountain.

One demonstration system that integrates many techniques in the spectrum is the KSL Wine Agent (www.ksl.stanford.edu/people/dlm/webont/wineAgent). This Web service uses structured content information (written in OWL with source information taken from the OWL Guide[16]), a structured query language (OWL-QL) to pose queries to the knowledge base, and a reasoner (JTP) to check if the content matches the query. It also leverages an explanation facility to justify any answer and to provide summary information about the source information used (Inference Web). Additionally, the service connects to Web information sources to gather recent product availability information. A typical question for the KSL Wine Agent is, "What kind of wine should I serve with a meal whose main course is pasta with spicy red sauce?" The answer includes a list of wines matching the course (for example, Marietta Zinfandel) and a description of the wine's properties (for example, dry, red, medium body, moderately flavored). The explanation includes the source of the content used (the KSL wines ontology). If the user requests a detailed explanation, it also includes what rules the service used to choose the wine's properties. Pricing information is also provided from online sources, such as wine.com. This application goes well beyond simply returning Web pages that might include foods and wines and instead provides information about specific wine suggestions as well as the description of the wine, and its explanation for its recommendation.

The techniques described in this article range from simple enhancements that need limited background knowledge to more sophisticated techniques that leverage more extensive knowledge of content, query language, reasoners, and explainers. All of them rely on some way to encode meaning. As more ontologies become available, and as more structured sources of content become available on the Web, question-answering services will have more opportunities to use these techniques, and thus will be able to provide more targeted, understandable, and useful answers to end users. ■

References

1. P.R. Hagen, H. Manning, and Y. Paul, "Must Search Stink?" Forrester Research Tech-Strategy Briefing, June 2000.
2. D. Day et al., *AAAI Spring Symp. Series New Directions in Question Answering Workshop*, March 2003.
3. D.L. McGuinness and P. Pinheiro da Silva, "Registry-Based Support for Information Integration," *Proc. 18th Int'l Joint Conf. Artificial Intelligence 2003: Workshop on Information Integration on the Web* (IIWeb-03), S. Kambhampati and C. Knoblock, eds., AAAI Press, 2003, pp. 117–122.
4. D.L. McGuinness and F. van Harmelen, eds., "OWL Web Ontology Language Overview," World Wide Web Consortium (W3C) recommendation, 15 Dec. 2003; www.w3.org/TR/owl-features.
5. D.L. McGuinness, "Ontological Issues for Knowledge-Enhanced Search," *Proc. Formal Ontology in Information Systems*, June 1998, IOS Press. Also in *Frontiers in Artificial Intelligence and Applications*, IOS Press, 1998.
6. R. McCool, R. Fikes, and D.L. McGuinness, *Semantic Web Tools for Enhanced Authoring*, tech. report, Knowledge Systems Laboratory, Stanford Univ., May 2003.
7. D.L. McGuinness, "Ontology-Enhanced Search for Primary Care Medical Literature," *Proc. Int'l Medical Informatics Assoc. Working Group 6—Medical Concept Representation and Natural Language Processing Conf.*, Dec. 1999.
8. R.J. Brachman et al., "Integrated Support for Data Archaeology," *Int'l J. Intelligent and Cooperative Information Systems*, vol. 2, no. 2, 1993, pp. 159–185.
9. D.L. McGuinness, "Ontologies Come of Age," *Spinning the Semantic Web: Bringing the World Wide Web to Its Full Potential*, D. Fensel et al., eds., MIT Press, 2002, pp. 171–194, www.ksl.stanford.edu/people/dlm/papers/ontologies-come-of-age-abstract.html.
10. D.L. McGuinness, *Explaining Reasoning in Description Logics*, doctoral thesis, tech. report LCSR-TR-277, Rutgers Univ., 1996.
11. D.L. McGuinness and P. Pinheiro da Silva, "Infrastructure for Web Explanations," *The Semantic Web – ISWC 2003*, LNCS 2870, Springer-Verlag, 2003, pp. 113–129.
12. P. Pinheiro da Silva, D.L. McGuiness, and R. Fikes, "A Proof Markup Language for Sematic Web Services," tech. report KSL-04-01, Knowledge Systems Laboratory, Stanford Univ., 2004.
13. F. Baader et al., "Matching in Description Logics," *J. Logic and Computation*, vol. 9, no. 3, 1999, pp. 411–447.
14. A. Borgida and D.L. McGuinness, "Asking Queries about Frames," *Proc. 5th Int'l Conf. Principles of Knowledge Representation and Reasoning*, Morgan Kaufmann, 1996.
15. R. Fikes, P. Hayes, and I. Horrocks, eds., *OWL-QL: A Language for Deductive Query Answering on the Semantic Web*, tech. report 03-14, Knowledge Systems Laboratory, Stanford Univ., 2003.
16. M.K. Smith, C. Welty, and D.L. McGuinness, eds., "OWL Web Ontology Language Guide," World Wide Web Consortium (W3C) recommendation, 15 Dec. 2003, www.w3.org/TR/owl-guide.

Deborah L. McGuinness is the associate director and senior research scientist at Stanford University's Knowledge Systems Laboratory. She also runs a consulting business dealing with ontologies and artificial intelligence for business applications. Contact her at the Knowledge Systems Laboratory, Stanford Univ., Stanford, CA 94305; dlm@ksl.stanford.edu; www.ksl.stanford.edu/people/dlm.