Grand Challenges Conference Application

Beyond Information to Answers

David Goldberg
Palo Alto Research Center

The web is making more and more *information* publicly available, but having a vast store of information doesn't mean it is easy to get an *answer* to a question. I believe that building a system that can provide answers and not just information is both important and challenging.

Getting an answer quickly from the web is difficult for many reasons. Search engines use syntactic not semantic analysis and so can't do sophisticated queries. Even if they could reliably find appropriate documents, the answer is often only implicit and can require expertise that the user doesn't have. And even if the user can ferret out the information, it might require more time than the user has available (e.g. in an emergency situation).

The following examples illustrate the importance of timely access to an answer:

- A poison control center gets a call concerning a patient who has ingested an unusual combination of chemicals. They need to contact a medically knowledgeable chemist immediately.
- A physician outside the major medical centers sees a patient who has symptoms that may or may not indicate an extremely serious but rare condition. The physician needs to access a specialist while the patient is still at the doctor's office.
- A traveler at an airport has unusual vials in his suitcase, and claims to be a famous biochemist on his way to an important conference. Security would like to be able to quickly verify this so that if he really is legitimate, he doesn't miss his plane.
- A 911 operator gets a call late at night from someone in a very large building who only says they are trapped in the "photonics lab" before passing out. The building security guard doesn't have this listed on his directory. The operator would like to quickly find someone who knows where the photonics lab is.

When you really need to know something, it's much better to be able to ask an expert than to be in the middle of a vast library. And for the foreseeable future, human experts will significantly outperform AI-based software experts. So I believe the key to building a system that can handle examples like the ones above is the proper tradeoff between a completely machine-based system and a completely human-based one, just as traditional systems were concerned with the tradeoff between hardware and software. For example in machine translation (from say Russian to English), it is not a good tradeoff to have the whole task done by computers -- it is better for computers to do a rough translation and for humans to do the polishing.

The pieces are all available for a reliable, easy-to-use answer system: portable phones for contacting experts, technology for assembling databases of experts, technology for

determining in real-time which experts are actually available and "on call." However, assembling a workable system is challenging. Some of the problems are:

(1) Easy ways to find experts. Finding an expert needs to be much easier than finding the answer to the question itself.

(2) Reliability guides to experts -- how much trust can you put in their expertise, and exactly what do they know about

(3) Compensation: how do experts get compensated (and users pay for expertise)?

(4) Load-balancing among experts: if there are 20 medically knowledgeable chemists who have volunteered to be on call to poison centers, they should be contacted equally often. But how do you weight a 3 am call against one that occurs at a more convenient time?

(5) How to rapidly find an expert who is available? Simply phoning one after the other might be too slow. Phoning many of them simultaneously and then saying "never mind" to all but one will be annoying and a disincentive to participating as an expert. Technology can help. For example experts whose cell phones are turned off could be known to the system so it doesn't waste time trying to contact them. And if experts are willing to have their approximate location tracked, they could have location-linked availability profiles such as "After 9pm I'm available if at work but not if I'm at home".

This system could also be used for less serious pursuits. For example, your CD player just broke, the local stores are closed, and you want an expert to help walk you through an attempt to repair it. Or you are planning a scuba diving vacation in the Caribbean and want to talk to someone who has experience with diving in this region.

Let me conclude by examining related systems that exist right now. If you have a problem with a specific product, you can phone the manufacturer's hot line with a question. I envisage something that is much more general: it works for a wide range of topics, and connects to people who aren't (necessarily) full-time telephone support people. Instead, the system connects to true experts who are fielding calls as a part-time sideline. [Often those so-called experts on the hotline know less than you do anyway.]

A second related system is topic-specific chat rooms. This differs in three major ways from what I imagine. Chat rooms have fixed categories, but I imagine something more flexible -- for example I can find combinations of expertise (e.g. both scuba diving and traveling in the Caribbean). Second, I imagine that a large part of the system design is tailored to making real-time connections convenient. As a simple example, I might register times when I am usually available. However, when the phone rings and I am busy there will be an easy way to refuse the call and reschedule my availability times. And third, I imagine that the system would have ratings and payment schemes. I might volunteer as an expert on topic A, and thus be granted free use of the system for my own queries as long as my ratings hold up. Or I might chose to pay cash in order to use the system.

As computer systems become more sophisticated I think systems research will need to focus more on higher-level layers of the system. Designing the proper tradeoff between

human and computer expertise will be important in many such systems – the information answer system I've sketched out is just one example.

Brief Biographical statement for David Goldberg

I have a PhD in mathematics from Princeton, and started my career as a math professor. I then went to Silicon Valley and worked at Sun Microsystems for 5 years before moving on to Xerox PARC. At Sun, I was a member of the team that designed and built NSF, the still ubiquitous network file system. At PARC, I have worked on a diverse series of projects including

- Information Filtering ("Using Collaborative Filtering to Weave an Information Tapestry" from Dec 1992 CACM).This paper coined the term 'collaborative filtering.'

- Pen Input for computers ("Touch Typing with a Stylus" from the INTERCHI '93 Conference in Amsterdam). This paper outlined the pen-input technique that is the basis for Graffiti on the Palm Pilot.

- Computer arithmetic (Appendix A of Patterson & Hennessy's "Computer Architecture" book, 1996).

- Image Processing of Scanned Text Documents ("Scanner-Model-Based Document Image Improvement", from Sep 2000 ICIP). This work represents an attempt to apply CS theory to a problem highly relevant to Xerox. It presents a way to improve the quality of fax images using a super-resolution algorithm.

- Digital Authentication of Paper Documents ("Secure Notarization of Paper Text Documents" from the Jan 2001 ACM/SIAM SODA). Another paper in the 'relevant to Xerox' series. This presents a technique that achieves the security of digital authentication for paper documents, and that doesn't require saving the document (or any information about it) online.

I have recently begun working in bioinformatics.