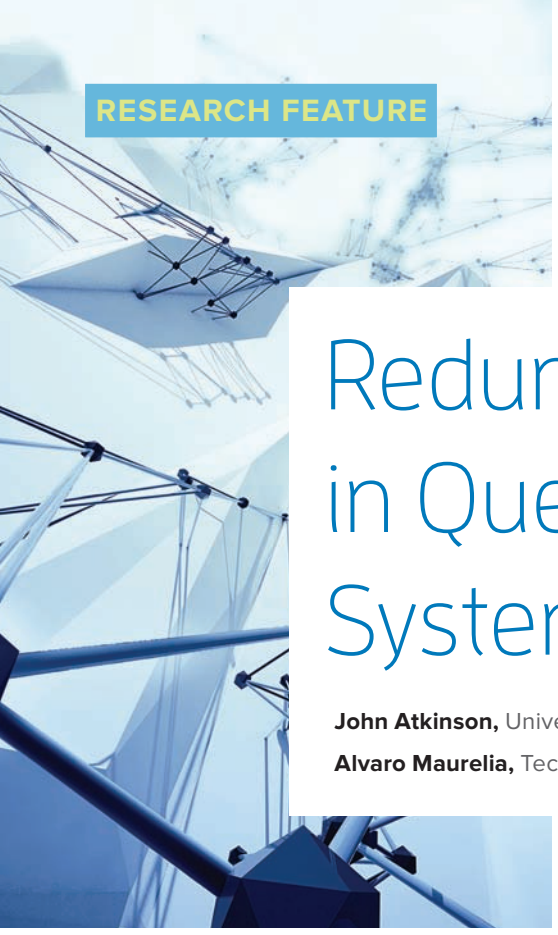

RESEARCH FEATURE

Redundancy-Based Trust in Question-Answering Systems

John Atkinson, Universidad Adolfo Ibañez

Alvaro Maurelia, TecNova

By combining user preferences, redundancy analysis, and trust-network inference, the proposed trust model can augment candidate answers with information about target sources on the basis of connections with other web users and sources. Experiments show that the model is more effective overall than trust analyses based on inference alone.

With the increased number and quality variance of web information, users face a growing dilemma as they attempt to evaluate the results of a query. From the many sources, documents, or answers, users must somehow choose those with the highest quality and relevance in the absence of any guidance or reasoning to refine the list. Users frequently rely on a question-answering (QA) system, which ranks candidate answers according to their relevance to the question. Unlike a search engine that delivers snippets or documents from a query, a QA system uses inference mechanisms to produce the exact answer to a question. For example, a user needing to retrieve information about Donald Trump's birthplace might query, "Where was Donald Trump born?" and receive "Pakistan" as a result. Can the result's source be trusted? If so, why? Can other users cite this source with confidence? At present, the answer is "no" because the result has no evidence to support its validity.

Thus, trust in relevant questions when information sources do not come with any justification for their selection might not be warranted.[1–3] For users to have confidence in the answers they receive, they need trustworthy information along with the answer, such as the origin of the answers and the reasoning processes used to produce them.

Analyzing candidate answers' trustworthiness is a complex task that researchers have just begun to explore. The core issue is how to evaluate the query results' quality with no information or context about how the answers were calculated or ranked. Most existing solutions incorporate an inference mechanism that derives proofs and stores them in a trust network so that information-manipulation tasks can answer additional questions. Although these systems support the ability to track knowledge from the answer to its source, they do not account for the case in which answers are extracted from a single source, such as a webpage, and consequently could have a bias that makes them less trustworthy.

To address this concern, we developed a redundancy-based model that computes trust along many axes: a query's context, the degree to which a user perceives the reputation of users who provide information, user preferences, and the answer's sources. Trust computation is based on key metrics that have proved useful

0018-9162/17/$33.00 © 2017 IEEE

in computing trust in QA systems as well as on reputation metrics that account for the user's degree of belief in sources, search engines, and other users who provide links, for example, through a blog. Our model has broad application, particularly in scenarios in which a user must choose from among several sources or responders. Examples include communication in a crowd-sourced sensor network and network optimizations—both of which require trust in multiple sources and would benefit from finding the most trustworthy paths.

To assess our model's effectiveness, we implemented a prototype using the natural-language toolkit in NLTK 3.0 and conducted experiments in a variety of settings to evaluate our model's ability to precisely infer users' trust values for difference sources. Results show that our model outperformed other trust methods in terms of relevance ranking with only slightly less precision.

EVALUATING TRUST IN QUESTION-ANSWERING SYSTEMS

False information can often result in considerable damage. Hence, trustworthiness of information is an important issue in this data-driven world economy. The reputation of different agents in a network has been studied earlier in a variety of domains like e-commerce, social sciences, sensor networks, QA systems, and peer-to-peer (P2P) networks.[4,5] Recent work in the data mining and QA systems communities has proposed the use of agents that analyze the trustworthiness of various data objects given information from multiple providers. Such agents and their information about data objects form a heterogeneous network.

Trust metrics

For a posed natural-language question q, a QA system retrieves a set of answers $A = (a_1, a_2, \ldots a_k)$ from different knowledge sources $(s_1, s_2, \ldots s_j)$. In a standard QA system, a preprocessed query is sent to a search engine, which retrieves candidate passages that might contain correct answers.[3]

Because myriad retrieved answers are possible, filters are employed to restrict the number of answers according to relevance criteria, which are also the basis for ranking answers. Despite the popularity of this approach, users struggle to trust the information given if they do not know how it was obtained, even if an answer is highly relevant to the question.[3] Hence other QA models consider the user's u preferences to determine the set of proofs $N(A)$ for a set of answers.

To compute the trust for candidate retrieved answers for a target user $(T(u,N(A)))$, trust-based QA models often use one or more of four metrics:

- *Relevance*—an answer's closeness to a target question as determined by a search engine's assessment, which might be based on semantic relevance or keywords that answers and questions share.
- *Confidence*—trust in the sources from which answers are extracted, which involves calculating some judgment related to the information's quality.
- *Proof*—explanation of how a set of answers were obtained by evaluating users' preferences, usually expressed as $N(A)$ for a set of answers.
- *Reputation*—the trust that a user community has in a particular member of that community.

Proof mechanisms

Inference Web Trust (IWTrust), a popular trust-based QA method, is a good example of proof mechanism use. IWTrust provides proofs for retrieved answers that are based on user preferences, which it derives from assessing a user's trust in other users, sources, and search engines.[3] The method includes a trust net that represents associations between users and trust values with respect to an information repository (IWBase) that contains a set of sources and search engines. Mechanisms traverse the trust net, inferring proofs for each answer.[6] IWTrust obtains trust values between users and between a user and a source or search engine by multiplying all the values of the net's

ANALYZING CANDIDATE ANSWERS' TRUSTWORTHINESS IS A COMPLEX TASK THAT RESEARCHERS HAVE JUST BEGUN TO EXPLORE.

edges that must be traversed to reach other users with reliable information.

Once a trust net is built, a Proof Markup Language[7] is used to apply the proof mechanisms to certain answers. The PML views a node sequence as the application of inference steps to reach a conclusion.

Trust analysis between users and sources depends heavily on how often answers are retrieved. To determine this frequency, some researchers have proposed probabilistic methods to estimate trust values in the edges between nodes.[8]

Redundancy analysis

The main advantage of methods like IWTrust is that they consider user preferences in determining an extracted answer's reliability. Thus, the same answer might have different trust values, depending on its sources. On the downside, these methods do not address the occurrence of multiple similar answers, which means that no specific answers emerge as more trustworthy than others. In this case, redundancy analysis is an effective addition because trust can then be computed according to how often independent sources produce the answer. From a quality view, the same answer retrieved several times from different sources makes the answer more believable and trustworthy for some user.

Trust values among users are important because trust increases if the answer comes from a source provided by a user with a solid reputation among those in the same community. For example, users are more likely to trust answers from ny.com than from foxnews.com because writers for the *New York Post* have an established reputation. Adding redundancy analysis to a trust network lets an agent infer trust from reputation because incorporating redundant answers means fewer nodes and edges with a trust network, which in turn increases the trust computation's accuracy.[9]

Redundancy analysis aims to determine whether two questions use different linguistic styles to ask essentially the same thing. Although answers might be phrased differently, they can still follow some answer pattern, and if it is possible to define a pattern for a question, the answer type can be inferred. For example, for the question "Where is the Hudson River?," an answer is likely to contain "river." After identifying the question pattern and expected answer type, redundancy analysis uses a named entity–recognition (NER) technique to match that information with candidate answers.[10] The more instantiated answers it finds, the higher the trust values of those answers become.

Methods that build trust nets

As this brief description implies, redundancy analysis is intended to identify similar answers, not to compute trust, and it does not analyze trust in the sources of redundant information. To order retrieved answers according to quality, methods such as PageRank (PR) are useful. These methods base ranking on the number and quality of a retrieved answer's links from and to other answers.[11] The more links and the higher the quality of those links, the more important a source is likely to be. The underlying assumption is that more important sources are likely to receive more links from other sources.

Techniques such as TrustNet use social media data to build their own trust nets where users interact.[1,7] When no relationships can be found between a user pair, trust cannot be directly determined, and trust computing is then based on the trust values of friends of the target user, such as friend-of-a-friend (FOAF).

Fusion methods

One recent approach[12] investigated the truthfulness of web data in different domains, which is important to people looking for information in general, not necessarily answers to a particular question. Experiments using this approach revealed many data inconsistencies and low-quality sources. To find the truth in conflicting data such as untrusted and trusted sources for the same question and to generally understand web data's accuracy, the authors of the approach applied fusion methods to target datasets. These methods showed good potential; different data fractions from the same source can have different quality, so fusion results might be a promising way to distinguish attribute quality. On the other hand, some issues remain open. For example, trustworthiness computed in this way might not be precise; indeed, knowing precise trustworthiness can fix nearly half the mistakes in the best fusion results.

REDUNDANCY ANALYSIS IN A TRUST NETWORK

In our model, a trust network is composed of nodes that represent users or sources and edges that contain values reflecting the degree of trust in those users. The network

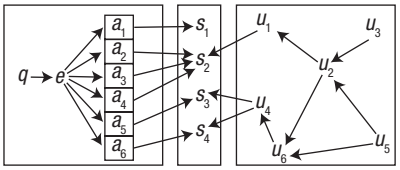

FIGURE 1. Trust model extended with redundancy analysis, which considers the user's preferences. In the example shown, q represents the initial question to an entity, e, which generates six candidate answers from four sources. User u_1 is likely to find source s_2 trustworthy and by extension answers a_2, a_3, and a_4.

uses transitivity links to infer relationships between nodes: *A* trusts *B* and *B* trusts *C*, so *A* trusts *C*. In addition, the model assumes that the shortest path between nodes has the highest trust value. When it finds more than one shortest path, it averages their values.

Overall, highly trusted paths tend to be used more frequently. For example, a user who trusts Google will use it regularly. In contrast, users do not trust Yahoo to the same degree are unlikely to use it because they see the lack of other users' interactions as a sign that Yahoo sources might be untrustworthy.

When the path between nodes is too long, inference becomes less accurate. For that reason, our model uses reputation to compute trust between distant nodes. Computing includes three major tasks:

- It classifies the question and expected answer and selects the best strategy to group candidate answers, for example, by location.
- It retrieves candidate answers from a target corpus, such as Wikipedia, WordNet, or a specific-purpose knowledge base using a search engine and singles out answers with a definiendum-matching technique. (In phrasing ontology, a definiendum is a word, phrase, or symbol that is the subject of a definition.)
- It computes trust for and ranks selected answers, using redundancy analysis to group similar information.

Computing trust is determined by building a trust network from candidate answers and their trust values and validating and ranking answers with the highest values. To group answers through redundancy analysis, the model first uses NER techniques to extract entities from candidate answers and then it attempts to match them with the question. To illustrate, let q be the question posed by user u_1, and $(a_1, a_2, \ldots a_5)$ be the extracted candidate answers. When q = Where is target$_1$?, candidate answers could have these compositions of words (w) and entities (e):

- $a_1 = w_1, w_2, w_3, e_1$
- $a_2 = w_1, w_4, e_1, w_5, w_6, e_2$
- $a_3 = w_2, e_1, w_7, w_8$
- $a_4 = w_1, e_2, e_3$
- $a_5 = w_7, w_8, w_9, e_3$

By analyzing the candidate answers, the model extracts three named entities (e_1, e_2, and e_3), and each answer contains one or more of these. e_1 has been retrieved three times (by a_1, a_2, and a_3); e_2, twice (by a_2 and a_4); and e_3, once (by a_5). Hence the most trusted answer is e_1 because it was retrieved the most often.

However, suppose the user does not trust the sources from which e_1 and by extension does not trust a_1, a_2, and a_3, but does trust the source for a_5. Then, e_3 is more trustworthy for that user, even though it was retrieved from fewer sources than e_1. This is an example of why user preference must be considered and why our model extends the trust model with redundancy analysis, as Figure 1 shows.

Analyzing answers in this manner is the start of extracting a set of candidate answers and their sources. Entities in those answers are then identified through NER techniques, similar answers and sources are grouped, and the sources' trust values are summed for each identified entity.

Trust computation

Our model considers user preferences in computing trust in redundant answers[3] and calculates trust in sources associated with the entities in an answer through an extended trust network that includes edges to the answers' sources.

Unlike other trust network approaches which determine trust values for a unique node,[3] our model computes trust values for every node from which the extracted answers originated. For example, when e_1 is extracted from sources s_1, s_2, and s_3, the model calculates and sums trust values for each source.

To calculate the trust value of a user in a previously unknown node, the extended network employs the preferences of that user's neighbor who knows about the node—an FOAF link—which results in several likely paths. For example, to determine trust of u_1 in u_4, the path might be u_1–u_2–u_4 or u_1–u_2–u_3–u_4. As Figure 2 shows, each path has different trust values, and it is possible that no path between them exists, such as between u_1 and u_3 or between users and an unreachable source. The latter is important because trust among users depends on the trusted sources, which is determined by how often users rely on them.

Another example is an unreachable source. In Figure 2, for example, u_1 has no direct path to s_1. Calculating trust in

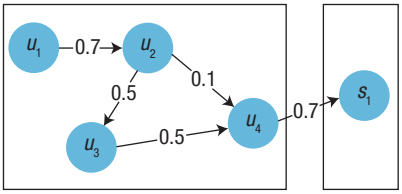

FIGURE 2. Trust values in a user community. Establishing trust between users involves choosing the path between those users that has the highest sum of trust values. Trust between users u_1 and u_4, for example, starts with the high trust that u_1 has in user u_2. The path could go from u_2 to u_4, but the trust value on that segment is low. In contrast, the segment from user u_3 to u_4 has a higher trust value, so that path would be more trustworthy for u_1.

unreachable sources is based on an analysis of reputation—the trust value assigned by the whole group to the source.

Our model uses a trust network that extends the traditional trust network—one created directly from user preferences with no inference. The expanded trust network adds edges inferred from FOAF links. The links in turn can be on different levels, where a level represents the length of a path used by FOAF to infer the trust value minus 1. For example, If u_1 knows u_2, u_2 knows u_3, and u_3 knows u_4, then two edges are created with expansion level 2: one from u_1 up to u_4 and the other from u_1 up to u_3.[3]

With the expanded network, it is easy to determine that, when no direct edge exists between nodes u_1 and u_2, u_2 is unreachable from u_1. In this case, we use reputation analysis to estimate u_1's trust value by summing the direct, indirect, and net trust factors. The *direct* factor is obtained whenever a user has a direct edge within the expanded network up to a source. The trust value becomes the weight of such an edge, as in u_4's trust in s_1 in Figure 2. The *indirect* factor is obtained whenever a user does not have an edge to a source, which requires using the source's reputation. Finally, the *net trust* factor is the sum of the direct and indirect factors for each source containing a redundant answer.

The net trust factor is insurance that the model can be independent of a user's preferences when that user does not know the sources. In that case, reputation becomes a single value. For example, in Figure 3, u_2 can reach u_4 through two paths. Thus, when u_1 poses a question and gets a redundant answer from two sources (s_1 and s_2), the net trust value (NTV) can be determined by computing the trust value to s_1, which is the value of the edge from u_1 to s_1 (direct factor). The value for s_2 must then be calculated, as u_1 has no edge to s_2. The calculation of s_2's value is then computed from the reputation value for u_4 (indirect factor).

To compute a source reputation, we used a modified PR algorithm,[11,13] because the original PR algorithm assumes that pages (sources) are interconnected only though links and thus does not consider the probability of users trusting those sources. Our modified PR (MPR) algorithm accounts for this probability:

$$\mathrm{MPR}(a) = (1-d) + d \times \sum_{i=1}^{n} \mathrm{MPR}_i / C(i) \ , \qquad (1)$$

where a is a candidate source answer, MPR(a) is the MPR for a, MPR_i is the MPR for u_i nodes that point to a, $C(i)$ is the number of outbound links of those nodes; d is the possibility that a user will open a source, such as a webpage, instead of typing its address.

As an example, assume that a user community is composed of u_1, u_2, and u_3 and that u_1 does not trust any user; u_2 trusts u_1 and u_3; and u_3 trusts u_1. Computing the MPR for every node starts with the default value of 1 and a value of 0.85 for d, as suggested in other work.[11,13] From equation 1, $\mathrm{MPR}(u_1) = 1$, $\mathrm{MPR}(u_2) = 0.575$, and $\mathrm{MPR}(u_3) = 1.425$.

Once NTVs are obtained for each candidate answer source, threshold values (for example, 0.5) can be used to determine whether the NTVs are valid:

$$\mathrm{valid}(A) = \begin{cases} \text{no} & \text{if NTV} = 0.5\ (\text{unknown } A) \\ \text{yes (trust)} & \text{if NTV} > 0.5\ (\text{can trust } A) \\ \text{yes (untrust)} & \text{if NTV} < 0.5\ (\text{cannot trust } A) \end{cases}$$

As a last step, our model ranks valid answers along with their groups of similar sources.

EXPERIMENTS

In our experiments, we were interested in evaluating both the relevance and precision of retrieved answers. We computed precision as $1 - |\mathrm{RTV} - \mathrm{ITV}|$, where RTV is the *real trust value* calculated for a user and source that are directly connected by an edge and ITV is the *inferred trust value* calculated by inference.

Overall precision is acceptable as long as difference between RTV and ITV is sufficiently small. For example, if u_1 has an RTV = t_1 to source s_1 (where t_x represents some trust value), then u_1 has a direct edge to that source. Whenever the edge is removed, our model must use a different path to compute trust values, such as from u_2 to s_1, which has a direct edge and whose ITV = $t_2 \times t_3$. Precision is high whenever t_1 is close to $t_2 \times t_3$.

Although precision is a useful metric for assessing and comparing trust models, we also wanted to examine relevance metrics to assess how well our model ranks answers.

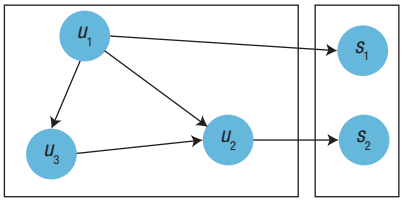

FIGURE 3. An expanded network that extends the trust network created directly from user preferences with edges inferred from friend of a friend (FOAF) links. The expanded network ensures that trust can be computed independent of a user's preferences when that user does not know the sources, as is the case with user u_1 and source s_2 in the diagram.

For that reason, we employed traditional QA metrics in our experiments, such as mean reciprocal rank (MRR). MRR assesses a list of candidate answers to a sample question set Q, and ranks answers by correctness probability. An answer's reciprocal rank is the multiplicative inverse of the first correct answer's rank, so MRR is the average of the reciprocal ranks for Q:

$$\text{MRR} = \frac{1}{|Q|} \sum_{\forall_q \in Q} \frac{1}{\text{rank}_q}. \quad (2)$$

To compute trust values in the extended trust network, we needed to create the trust network itself, construct a reputation table, and extract a set of questions and answers.

Creating the trust network

To create the trust network, we implemented a simple web service that used a custom search engine to enable Google-based searching. From the search engine, we constructed a group of 117 users who posed location questions to the search engine. We used these questions to infer trusted answers (those users clicked on)—on average, 110 answers for each question—according to a set of 50 answer sources. We then asked users to open the trusted pages, which were sources appearing next to the retrieved answers.

The service tracked and recorded all user actions into logs, from which we manually crafted simple rules for computing trust. For example, users were to open only trusted links and to start with links that had the highest trustworthiness. Building a more complex network would require extensions such as techniques to create user profiles, theme-based or self-maintained trust nets, which automatically generate trust values so that different paths provide the same values.

Constructing a reputation table

The reputation table kept a list of sources and their reputation values, which we used to calculate the indirect factor by applying the MPR algorithm. For example, if u_1 has trust value t in source s_1, the indirect factors might have values such as

- if $t = 0.1$, then u_1 does not trust s_1;
- if $t = 0.4$, then u_1 does not fully trust s_1; or
- if $t = 0.9$, then u_1 fully trusts s_1.

To generate the reputation table, our model first normalizes indirect factors to less than 1 by dividing all MPR values into the maximum MPR value.

Extracting questions and answers

Table 1 shows a sample of the 110 questions we extracted using the custom search engine. We grouped questions according to their nature, such as monument or location. Each question returned 100 answers. We used the trust network and reputation table to analyze answers, extract their entities, and group answers according to similar sources. Table 2 presents the MRR for ranking candidate answers, showing the first, second, and third correct answer.

Results

Our model achieved an average precision greater than 67 percent with 167 nodes (117 users and 50 sources), which is only slightly lower than the 73 percent achievable with a traditional approach such as TrustNet.[7] with the same number of nodes. The difference might be due to our use of simple rules in creating the extended trust network, which might not have been sufficient to create a user profile.

On the other hand, maximum precision was as high as 98.47 percent, which is evidence that our model can infer trust accurately. In ranking the answers extracted with the Google search engine, the average MRR was 0.90, which compares favorably with the 0.86 MRR achieved with Google's QAHON approach.[9] The lowest MRR of 0.80 for the Actions group (see Table 2) could be because this group has questions with no unique correct answer.

Our experimental results show the promise of combining redundancy analysis and an extended trust network in improving the precision and trustworthiness of answers that a QA system retrieves. The model's MRR outperformed competitive approaches such as QAHON, and its precision was only slightly lower—which could be due to the simplicity of the trust rules we used.

TABLE 1. Precision with 110 sample answers.

Group	Sample questions	No. of answers	Precision (%) (min)	Precision (%) (max)	Precision (%) (avg)
Monuments	Where is the library located? Where is the Bio-Bio river? Where is Peru Square?	15	47.42	98.31	68.22
Cities	Where is Futaleuf? Where is Puerto Montt? Where is Concepcion?	40	50.36	94.86	70.12
Actions	Where can I buy videogames? Where can I buy books? Where should I study engineering?	15	51.17	96.38	69.83
Events	When was Albert Einstein born? When did World War II take place? When did Leonardo da Vinci die?	40	44.71	98.47	61.29

TABLE 2. Mean ranking rate (MRR) with 110 samples.

Group	MRR	Correct answers: First	Correct answers: Second	Correct answers: Third	Correct answers: Total
Monuments	0.90	8	2	0	10
Cities	0.93	9	0	1	10
Actions	0.80	7	1	0	10
Events	0.95	9	1	0	10

A possible research direction is to improve data collection and rules for inferring users' profiles based on close answer groups by providing a recording system through websites. Alternatively, a theme-based extended network could be generated to provide assurance that users can trust sources from a particular theme. In this approach, edges within a trust network would contain both trust values and associated themes.

REFERENCES

1. T. Knap and I. Mlynkova, "Quality Assessment Social Networks: A Novel Approach for Assessing the Quality of Information on the Web," *Proc. 8th ACM Int'l Workshop Quality in Databases* (QDB 10), in *Proc. 36th ACM Int'l Conf. Very Large Data Bases* (VLDB 10), 2010, pp. 1–10.
2. T. Knap, A. Freitas, and I. Mlynkova, "Web Quality Assessment Model," *Proc. 8th ACM Int'l Conf. Ubiquitous Intelligence and Computing* (UIC 11), 2011, pp. 252–266.
3. I. Zaihrayeu, P. Pinheiro da Silva, and D. McGuinness, "IWTrust: Improving User Trust in Answers from the Web," *Proc. Trust Management Conf.* (ITrust 05), 2005, pp. 384–392.
4. G. Manish, S. Yizhou, and H. Jiawei, "Trust Analysis with Clustering," *Proc. ACM Int'l WorldWide Web Conf.* (WWW 11), 2011, pp. 53–54.
5. G. Manish and H. Jiawei, "Heterogeneous Network-Based Trust Analysis: A Survey," *ACM SIGKDD Explorations Newsletter*, 2011, pp. 54–71.
6. P. Pinheiro da Silva, D. McGuinness, and R. Fikes, "A Proof Markup Language for Semantic Web Services," *J. Information Systems*, vol. 31, no. 4–5, 2006, pp. 381–395.
7. Y. Wang and J. Vassileva, "Trust-Based Community Formation in Peer-to-Peer File Sharing Networks, *Proc. IEEE Int'l Conf. Web Intelligence* (WI 04), 2004, pp. 341–348.
8. C. Hang, Z. Zhang, and M. Singh, "Shin: Generalized Trust Propagation with Limited Evidence," *Computer*, vol. 46, no. 3, 2013, pp. 78–85.
9. S. Cruchet et al., "What about Trust in the Question Answering World?" *Proc. Ann. Symp. American Medical Informatics Assoc.* (AMIA 09), 2009; www.hon.ch/Conf/Docs/submittedpaper_QA_HON_NLM.pdf.
10. J. Guo, G. Xu, and X. Cheng, "Named Entity Recognition in

ABOUT THE AUTHORS

JOHN ATKINSON is a full professor in the Faculty of Engineering and Sciences, Universidad Adolfo Ibañez. While conducting the research reported in this article, he was a professor in the Department of Computer Sciences at the Universidad de Concepcion. His research interests include natural-language processing, text mining, and AI. Atkinson received a PhD in AI from the University of Edinburgh. He is a member of the AAAI and IEEE, and a Senior Member of ACM. Contact him at john.atkinson@uai.cl.

ALVARO MAURELIA is a software engineer at TecNova. While conducting the research reported in this article, he was an undergraduate student in the Department of Computer Sciences at the Universidad de Concepcion. His research interests include databases, question-answering systems, and applied AI. Maurelia received a BEng in computer sciences from the Universidad de Concepcion. Contact him at seba4to@gmail.com.

Query," *Proc. 32nd ACM Int'l Conf. Research and Development in Information Retrieval* (SIGIR 09), 2009, pp. 267–274.

11. C. Kohlschutter, P. Chirita, and W. Nejdl, "Efficient Parallel Computation of PageRank," *Proc. 29th ACM SIGIR European Conf. Information Retrieval* (ECIR 06), 2006, pp. 241–252.
12. L. Xian et al., "Truth Finding on the Deep Web: Is the Problem Solved?," *Proc. 39th ACM Int'l Conf. Very Large Data Bases* (VLDB 13), 2013, pp. 97–108.
13. P. Constantine and D. Gleich, "Using Polynomial Chaos to Compute the Influence of Multiple Random Surfers in the PageRank Model," *Proc. 5th Int'l Workshop Algorithms and Models for the Web-Graph* (WAW 07), 2007, pp. 82–95.