

Corroborating Answers from Multiple Web Sources

Minji Wu
Rutgers University
minji-wu@cs.rutgers.edu

Amélie Marian
Rutgers University
amelie@cs.rutgers.edu

ABSTRACT

The Internet has changed the way people look for information. Users now expect the answers to their questions to be available through a simple web search. Web search engines are increasingly efficient at identifying the best sources for any given keyword query, and are often able to identify the answer within the sources. Unfortunately, many web sources are not trustworthy, because of erroneous, misleading, biased, or outdated information. In many cases, users are not satisfied with—or do not trust—the results from any single source and prefer checking several sources for corroborating evidence. In this paper, we propose methods to aggregate query results from different sources in order to save users the hassle of individually checking query-related web sites to corroborate answers. To return the best aggregated answers to the users, our techniques consider the number, importance, and similarity of the web sources reporting each answer, as well as the importance of the answer within the source. We present an experimental evaluation of our technique on real web queries, comparing the corroborated answers returned with real user clicks.

1. INTRODUCTION

The Internet has changed the way people look for information. Users now expect the answers to their questions to be available through a simple web search. Web search engines are increasingly efficient at identifying the best sources for any given keyword query, and are often able to identify the answer within the sources. Unfortunately, many web sources are not trustworthy, because of erroneous, misleading, biased, or outdated information. With many web sources providing similar information on the Internet, users often have to rummage through a large number of different sites to both retrieve the information in which they are interested, and to ascertain that they are retrieving the correct information. In many cases, users are not satisfied with—or do not trust—the results from any single source, and prefer checking several sources for corroborating evidence,



Figure 1: Results for the query: “Honda Civic 2007 gas mileage” using MSN Search

as illustrated in the following example:

EXAMPLE 1. Consider a user interested in buying a car, and considering a specific brand and make (e.g., Honda Civic). One of the criteria influencing the decision is the gas mileage of the car. However, the gas mileage information available on the Internet differs not only based on the year of the car, but also based on the source from which the information is extracted: the official manufacturer web site (up to 51 mpg in the Honda Civic example) has a different value than some other commercial web sites (40 mpg. Autoweb.com; 30/40mpg, Car.com). None of these sources, all of which appear in the first page of results for the query “Honda Civic 2007 gas mileage” using MSN Search (see Figure 1), has the “perfect” answer for the query, but they all provide valuable information to the user.

In this paper, we propose methods to aggregate query results from different sources in order to save users the hassle of individually checking query-related web sites to corroborate answers. In addition to listing the possible query answers from different web sites, we rank the answers based on the number, importance, and similarity of the web sources reporting each answer, as well as the importance of the answer within the sources. The existence of several sources providing the same information is then viewed as corroborating evidence, increasing the quality of the corresponding information, as measured by a scoring function used to order answers. Our techniques are built on top of a standard web search engine query result. We address the following challenges:

- *Aggregating answers from several web sources.* We first need efficient techniques to extract answers from the web sources. The main challenge of answer corroboration is then the design of a meaningful scoring function. Our proposed approach takes into account the frequency of the answers in the web search engine result, but also the importance and originality of the pages reporting the answers (Section 2).
- *Selecting the web sources from which to retrieve the answers.* Accessing all the pages that match a given web search to retrieve and compare answers would obviously be very inefficient. We choose to consider a prefix of the search engine query result for information extraction, dynamically deciding on the size of this prefix based on the distribution of answers (Section 3).
- *Evaluating the quality of our proposed approach.* We ran preliminary experiments measuring both the time overhead of our techniques, and the quality of the answers returned. To measure quality, we compared our answers with actual MSN user clicks (Section 4).

We report on related work in Section 5 and conclude in Section 6.

2. CORROBORATING ANSWERS

A first step towards aggregating query answers from multiple web sources is to identify potential answers from the list of web pages returned by a search engine query. Once those answers are identified, similar answers are combined and scored based on their importance in the search engine results, as measured by different parameters such as the frequency of the answer in the search result, the prominence of the answer within the search result web pages, the placement of pages containing the answer in the search result, and the duplication of information between pages.

2.1 Identifying Answers

Given a list of web pages pertinent to a user keyword query, as returned by a web search engine, our first challenge is to efficiently retrieve from these web pages the data that qualifies as an answer to the query. For instance, in our example we need to retrieve gas mileage values that correspond to a 2007 Honda Civic from the search engine web page results.

Information extraction, the process of extracting relevant information from documents using text and structure, is a

Queries Containing Keywords	Average Number of Clicks for all Queries (including Queries with 0 Click)	Average Number of Clicks for Queries with 1 Click or more
“Length”	1.21	1.9
“Distance”	0.89	1.54
“Mileage”	0.98	1.5
“Width”	1.03	1.61
“Temperature”	1.07	1.68
All	0.78	1.32

Table 1: Average Number of Clicks per Query

complex problem that has been the focus of many work in the Information Retrieval community [4]. Since the focus of our work is on the aggregation and scoring of answers, we decided to use existing techniques for information extraction. In addition, as a first step we only focus on queries whose answers are numerical values. We restricted ourselves to such queries for two reasons:

- It is easier to identify numerical answers within web pages. For instance, mileage information will typically be of the form “ x mpg,” or “mpg of x ,” where x is a numerical value.
- Our preliminary analysis of about 15 million MSN query logs shows that keyword queries for which numerical query answers are expected (e.g., queries containing “mileage”, “length”) tend to result in more search engine user clicks than other queries. Table 1 shows the average number of user clicks for all queries, and for queries containing the keywords “Length,” “Distance,” “Mileage,” “Width,” and “Temperature.” These numerical queries, whose answers are usually numerical values, result in on average more user clicks than general queries. This result suggests that users are not satisfied with the answer to numerical queries found in a single web page and tend to check several sources to corroborate answers.

While our techniques can be used to answer queries for which a single “correct” answer is expected (e.g., “what is the length of the Brooklyn Bridge”), we focus more specifically on those queries for which there is no agreed upon correct answer (e.g., our gas mileage query example). For these queries, our approach aims at finding the “best” answer, i.e., the answer for which there is the most promising evidence within the search engine result pages.

2.2 Scoring Answers

Once answers are extracted from the web pages, we need to assign them individual scores. We then aggregate the score of similar answers to identify the best corroborative answers. We divide our scoring function discussion into three parts. In section 2.2.1, we first define the relevance score of a web page, we then define the score of each answer within a given page in section 2.2.2. The final overall score of an answer is defined in section 2.2.3

2.2.1 Relevance score of web pages

The score of an answer depends on the relevance of the web page from which the answer is extracted. To measure

the relevance of a web page, we consider the following two factors:

- The *importance* I of a web page p containing the answer, as measured by the search engine. We consider that a page ranked highly by the search engine provides better information than a page with a lower rank. Therefore an answer found within the higher ranked page will be scored higher than one found within the lower ranked page. We introduce a parameter α so that the importance of the $(i+1)$ th page is the product of importance of i th page and $(1 - \alpha)$. This parameter α represents the speed at which the importance of web page drops as we traverse the search engine result list and quantifies the decrease in importance of lower ranked pages. For page p , $r(p)$ is the rank of the page as returned from the search engine, The *importance* of the page p is as follows:

$$I(p) = (1 - \alpha)^{r(p)-1} \quad (1)$$

- The *duplication* D between pages. Our scoring aggregation technique takes source similarity into account to reduce the effect of similar information coming from sources in the same domain, or sources that seem to be mirror (or copy-paste) information from each other. Since we are looking into different web pages and trying to corroborate answers, duplicated pages which tend to have similar content should not have as much weight in the corroboration as pages that contain original content. However, they should still be taken into account as corroborative evidence. Our solution is to halve the score of a page each time a duplication is detected. For example, the first page from **Car.com** may have a score of 1, subsequent page coming from **Car.com** will have a score of $1/2$, $1/4$, ... etc. Therefore, if for a given page, there exist d_m pages sharing the same domain and d_c pages generated from copy/paste information, then we have:

$$D(p) = \frac{1}{2^{d_m+d_c}} \quad (2)$$

Combining the importance and duplication factors, the relevant score of a page p would be

$$s(p) = I(p) \cdot D(p) = (1 - \alpha)^{r(p)-1} \cdot \frac{1}{2^{d_m+d_c}} \quad (3)$$

2.2.2 Score of answers within one page

Given a web page, we consider the following two factors to compute the score of each answer extracted from this page:

- The number of different answers within a page p . The score of each answer is computed as the score of the page $s(p)$ divided by the number of answers extracted $N(p)$. In other words, as the number of answers extracted within one page grows, the scores assigned to each answer decreases, assuming that answers within the same page are equally relevant. Therefore the score of each answer extracted will be as follows:

$$s(x, p) = \frac{s(p)}{N(p)} \quad (4)$$

- The *prominence* of the answers within the page containing them. One problem of the above method is that all the answers extracted from the same page are rarely equally helpful in answering queries. In fact, there may exist some incorrect answers extracted from the page that are actually irrelevant to the query. Consider the following text from which we can extract two answers from a single web page for our running example:

EXAMPLE 2. *The 2007 Honda Civic has a gas mileage of up to 51 mpg, compared to a 33 mpg of Toyota Camry 2006.*

Obviously, the first numerical value is the one in which we are interested, whereas the second one is not. Unfortunately, due to the limitation of the information extraction techniques, it is difficult to figure out whether each numerical value is a correct answer to the query. Our solution is to consider the position of each answer with respect to the position of the query subject. This approach is based on the assumption that in most cases, relevant answers will be close to the query subject in the web page. First we pick the answer which appears closest to the query subject, at a distance of d_{min} characters. For every other answer x found in the same page, at a distance of d_x characters from the query subject, its score within page p is adjusted as:

$$s(x, p) = \frac{s(p)}{N(p)} \cdot \frac{d_{min}}{d_x} \quad (5)$$

Finally, some web pages may provide several answers; for example the **Car.com** web site in Figure 1 gives two answers for our query: 30 mpg (city), and 40 mpg (highway). Our current implementation weights down the scores of answers that are not unique in their sources, using a function of the number of conflicting answers. In some cases, as in the city/highway example, multiple answers in a given web page may be due to different context. We plan to include context in our scoring in the future, possibly enabling our techniques to output different answers for different contexts.

Taking into account both the core of an individual answer within a web page, and the score of the page, and substituting Equation 3 in equation 5, we can assign a score for an answer x from a page p as:

$$s(x, p) = \frac{1}{N(p)} \cdot \frac{d_{min}}{d_x} \cdot (1 - \alpha)^{r(p)-1} \cdot \frac{1}{2^{d_m+d_c}} \quad (6)$$

2.2.3 Corroborative scoring of answers

The *frequency* of the answers in the set of pages is considered the main evidence in the corroboration. Intuitively, an answer that appears in 10 pages is more likely to be the correct answer than an answer that appears in one page. Formally, if $score(x, p_i)$ is the score of answer x extracted from page p_i , the corroborative score of answer x is given by:

$$score(x) = \sum_{i=0}^n s(x, p_i) \quad (7)$$

where n is the number of pages we consider from the search engine query result.

Sources Domain	Answer	$s(x,p)$	$I(p)$	$D(p)$	score(x)
1. honda.com	51 mpg	1.0	1.0	1.0	1.0
2. honda.com	33 mpg	0.5	0.8	0.5	0.2
	38 mpg	0.5	0.8	0.5	0.2
3. autoweb.com	40 mpg	1.0	0.64	1.0	0.64
4. autoweb.com	30 mpg	0.5	0.512	0.5	0.128
	38 mpg	0.5	0.512	0.5	0.128

Table 2: Answers extraction results ($\alpha = 0.2$)

2.3 Scoring Example

We illustrate our scoring process with the following example. Consider the answers extracted from the first four pages of our example from Figure 1 and listed in Table 2. (We ignored the answer prominence score in the example and considered every answer within a page to be equally important. However, our implementation does consider answer prominence.)

In the table, the score of an answer within a page is listed in the column $s(x,p)$; the relevance score of web pages is divided between *importance* in column $I(p)$ and *duplication* in column $D(p)$. We set the value of the *importance* parameter α to be 0.2 in this example. The final score of each answer, which is the product of $s(x,p)$, $I(p)$ and $D(p)$, is listed in the rightmost column.

According to Equation 4, the score of each answer within a web page depends on the number of answers extracted. We extracted one answer from the pages 1 and 3, and two answers from pages 2 and 4; answers from pages 1 and 3 are then given a score of 1, while answers from pages 2 and 4 are given a score of 0.5.

The *importance* $I(p)$ of the pages decreases exponentially according to the value of α as we traverse the search engine results, as shown in the $I(p)$ column. The duplication $D(p)$ takes into account the fact that we have answers coming from pages sharing the same domain, as page 2 is sharing the same domain ([honda.com](#)) as page 1 and page 4 is sharing the same domain ([autoweb.com](#)) as page 3. Therefore pages 1 and 3 have a duplication score of 1 while page 2 and 4 have a duplication score of 0.5.

Answer	Score	Sources
51 mpg	1.0	honda.com
40 mpg	0.64	autoweb.com
30 mpg	0.128	autoweb.com
38 mpg	0.328	honda.com autoweb.com
33 mpg	0.2	honda.com

Table 3: Answer scores for our running example

We then aggregate the score of similar answers, as described in Section 2.2.3, resulting in scores listed in Table 3.

Answer Range	Score	Normalized Score	Sources
(50,55] mpg	1.0	0.4355	honda.com
(35,40] mpg	0.968	0.4216	honda.com autoweb.com
(30,35] mpg	0.2	0.0871	honda.com
(25,30] mpg	0.128	0.0557	autoweb.com

Table 4: Scores of different answer groups for our running example

In addition, we can opt to group answers that are close in

values, by setting answer intervals. Table 4 shows the resulting answer intervals, when we consider values in intervals of 5 mpg. Our current implementation allows the user to set predefined intervals. We are working on developing dynamic interval creation based on answer distribution. Finally, as shown in Table 4, we present scores normalized between 0 and 1 to the users, to make the output more easily readable.

3. RANKING ANSWERS

Finding and scoring answers is not the only challenge we face when corroborating answers from search engine query results; another challenge is to select the set of result pages from which we extract information. Specifically, we have to decide how deep we should go in the search engine query result. Accessing all the pages that match a given web search to retrieve and compare answers would obviously be impractical and inefficient. Work on top-k query processing algorithms has focused on adaptively reducing the amount of processing done by query evaluation techniques by ignoring data that would not be useful to identify the best answers to a query [6, 11]. We adapt ideas from top-k query processing work to the answer aggregation problem by adaptively selecting a subset of search results from which to extract answers, based on the current scores of the retrieved answers. Succinctly, we process web pages in the search result order, and stop retrieving new pages when the score of newly discovered answer would not be high enough to impact the overall corroborative answer score ranking.

We consider our page importance metric (Section 2.2.2) to decide when to stop retrieving new pages from the search engine result. As we traverse the search engine result list, the importance score of new pages we encounter decreases. We stop retrieving new pages when the importance of the remaining pages is too small for any answer extracted from new pages to cause any significant change to the corroborative answer list.

For this, assume we divide the web pages returned from the search engine into two groups, T^+ and T^- , where T^+ includes those pages that have been retrieved from the search engine result, and T^- those that have not. Initially, T^+ is set to empty set and T^- is the full list of pages T returned from the search engine. We define the importance of pages within T^- as follows:

$$I(T^-) = \sum_{p \in T^-} I(p) \quad (8)$$

As we retrieve and process pages from T for our corroboration, we remove pages from T^- and add them into T^+ . The combined importance $I(T^-)$ of pages in T^- is therefore constantly decreasing. During corroboration processing, we maintain a threshold variable D , which represents the score value required to cause significant change to the current result list. Our current implementation considers D to be the current difference in score between the top-1 answer (or answer group) and the top-2 answer. That is, we stop retrieving new pages when the current top-1 answer cannot change. Formally, we stop retrieving pages from T^- when the following condition happens:

$$I(T^-) \leq D \quad (9)$$

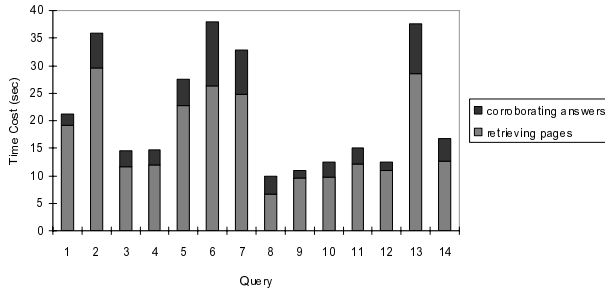


Figure 2: Average Time Cost

4. EXPERIMENTAL EVALUATION

We describe our experimental setup (Section 4.1), present our preliminary results (Section 4.2), and present a snapshot of our system (Section 4.3).

4.1 Experiment Setup

We implemented our system on top of the MSN Search SDK¹. For each answer, we dynamically retrieve web pages from which to extract answers. We set the default value of α to be 0.05. In Section 4.2.3, we show the impact of varying the value of α . We picked 14 real-user numerical queries from the MSN query log. (For privacy reasons, we are not able to provide the actual query keywords in the paper). We report on the following measures in our experiments:

- **Time Cost:** We report on the time needed to return the corroborated answer list. The time cost is divided into the time for retrieving the pages from the web, and the time used for the corroboration, including answer extraction and scoring.
- **Similarity of User Clicks and Results:** We compare the corroborated answers with the answers we extracted from pages on which real users clicked.
- **Number of Pages Retrieved:** As discussed in Section 3, we do not retrieve every page from the search result. The number of pages retrieved depends on the actual data, and on the α parameter.

4.2 Experimental Results

We now report on our experimental evaluation results.

4.2.1 Time Cost

Figure 2 shows the time cost for each of our 14 queries. We evaluated each query ten times and report on the average cost. While the overall cost can be high (around 35 seconds for the longest query), most of the evaluation time is spend retrieving pages from the web. Our current implementation is not optimized for speed, and we expect to be able to significantly reduce query evaluation time in the future, for instance by retrieving web pages in parallel. In addition, current query response time are reasonable, and users may find it acceptable to wait for corroborated answers if it saves them the hassle of manually checking the pages themselves.

4.2.2 Comparison of User Clicks and Corroboration Answers

Since we are focusing on queries for which there is no “correct” answer, it is difficult to evaluate the quality of our

¹<http://msdn.microsoft.com/live/msnsearch/default.aspx>

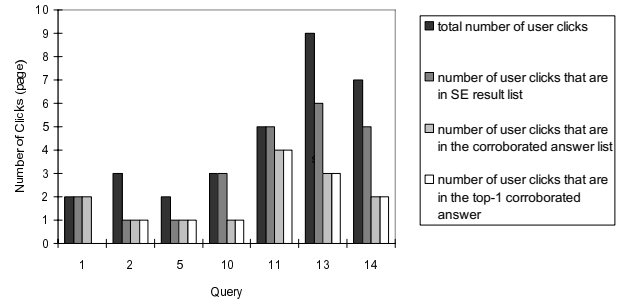


Figure 3: Comparison of user clicks and corroboration answers

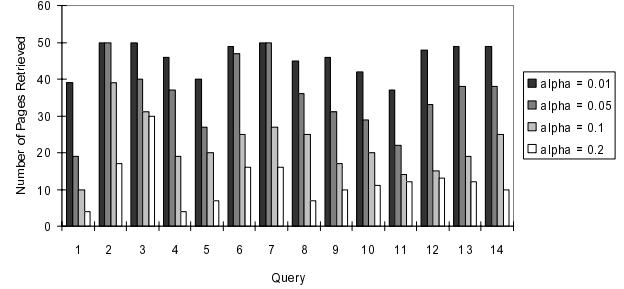


Figure 4: Number of pages retrieved

techniques. To provide some quality evaluation, we compared our corroborative answers with the user clicks from the MSN Live Search dataset. Our comparison was performed almost a year after the original queries were issued, and in some cases the pages users clicked on were not available in the search engine (SE) query result anymore. Figure 3 shows the comparison results for the seven queries for which at least one page on which the user clicked was present in the current result. We compare user clicks both with the set of pages used to generate corroborated answers, and the set of pages used to generate the top-1 answer. (In some cases, some pages on which the user clicked do not appear in the corroborated answer list; we believe this is due to the limitation of our answer extraction mechanism, which is not yet complex enough to extract every answer from the web pages.) The results show that the corroboration approach is good at identifying answers that are of interest to the user, as many user clicks correlate with top-1 corroborated answers.

4.2.3 Number of Pages Retrieved

Finally, we report on the number of pages retrieved from the search engine query result to corroborate answers. Figure 4 shows the number of pages retrieved for different values of α . As we discussed in Section 2.2.1, α represents the speed at which the importance of web page drops as we traverse the search engine result list, therefore, when α is high, fewer pages are retrieved. (In our experiments, we capped the number of retrieved pages at 50.) Figure 4 shows that our dynamic approach allows to limit the number of pages retrieved from the web for answer corroboration based on the actual answer scores.

4.3 The Corroboration System

We present an overview of our corroboration-based system in Figure 5. This figure is a snapshot of our system that shows the corroborated results for our running exam-

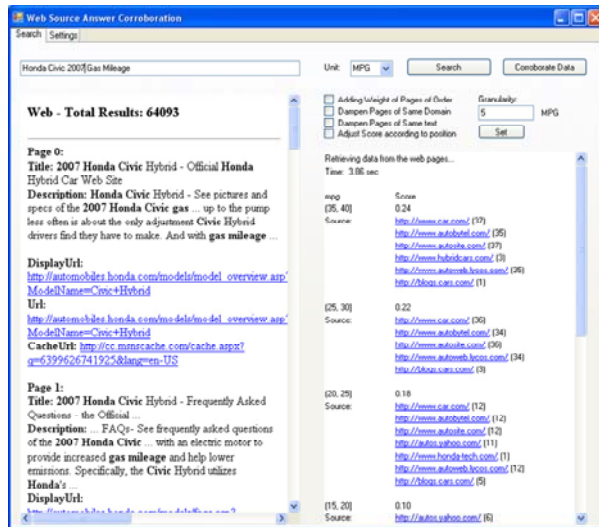


Figure 5: Snapshot of our corroboration system.

ple. The result of the answer corroboration step appears in the right result window with the set of sources that were used to corroborate each answer. Users can click on individual sources to see the corresponding answer in its context, and compare the corroborative answers with the answers extracted from the search engine top results.

5. RELATED WORK

Several work have addressed the issue of extracting answers from web pages [8, 1, 10, 2] for question answering. However, most of these question answering systems consider extracted answers separately and do not perform any answer corroboration. An exception is the Mulder system [10], which uses frequency of answers to increase answer scores. To the best of our knowledge, our techniques are the first to consider various measures of the quality of the answers and of pages as well as the frequency of these answers within the search engine query result to corroborate information from web query result pages. Using corroboration as a measure of answer quality has recently been suggested in non-web scenarios [9] where corroborating information is used to identify good answers across multiple databases in the presence of low quality data.

We use existing Information Extraction [4] techniques for extracting answers from web pages. Our information extraction implementation is basic, considering only plain text content of the page. We plan to improve it by using structural information within the page using techniques similar to those described in [1, 3, 5].

Work on top-k query processing algorithms has focused on adaptively reducing the amount of processing done by query evaluation techniques by ignoring data that would not be useful to identify the best answers to a query [6, 11]. We use similar ideas in Section 3 to limit the number of pages we are retrieving from the search engine. However, we should note that standard top-k query processing techniques cannot be applied directly to our scenario as they consider each answer individually, and therefore can always easily identify the maximum possible score of an answer. In contrast, when allowing for corroboration, the maximum score of an answer can always increase.

Finally, recent work have focused on improving user experience when querying the web. For instance the NAGA system [7] provides semantic knowledge to improve web search engine results.

6. CONCLUSION

We presented corroboration-based scoring techniques to improve web query results. Our techniques are built on top of a search engine query result and use information extraction techniques to identify relevant answers. Answers are then corroborated and scored based on their frequency, their importance within the web pages and the importance of the web pages in which they appear. We presented preliminary experimental results comparing our corroborated answers with real search engine user clicks and showed that our corroboration scoring can adequately identify good answers.

7. REFERENCES

- [1] Eugene Agichtein and Luis Gravano. Snowball: Extracting relations from large plain-text collections. In *Proc. of the Fifth ACM International Conference on Digital Libraries (DL '00)*, 2000.
- [2] Eric Brill, Susan Dumais, and Michele Banko. An analysis of the AskMSR question-answering system. In *Proc. of the ACL-02 Conference on Empirical Methods in Natural Language Processing (EMNLP '02)*, 2002.
- [3] Michael J. Cafarella, Christopher Re, Dan Suci, and Oren Etzioni. Structured querying of web text data: A technical challenge. In *Proc. of the Third Biennial Conference on Innovative Data Systems Research (CIDR'07)*, 2007.
- [4] William Cohen and Andrew McCallum. Information extraction and integration: An overview. In *Proc. of the 9th ACM International Conference on Knowledge Discovery and Data Mining (KDD'03)*, 2003.
- [5] Oren Etzioni, Michael Cafarella, Doug Downey, Stanley Kok, Ana-Maria Popescu, Tal Shaked, Stephen Soderland, Daniel S. Weld, and Alexander Yates. Web-scale information extraction in KnowItAll (preliminary results). In *13th International World Wide Web conference (WWW '04)*, 2004.
- [6] Ronald Fagin, Amnon Lotem, and Moni Naor. Optimal aggregation algorithms for middleware. *Journal of Computer and System Sciences*, 66(4), 2003.
- [7] Gjergji Kasneci, Fabian Suchanek, Maya Ramanath, and Gerhard Weikum. How NAGA uncoils: Searching with entities and relations. In *16th International World Wide Web conference (WWW '07)*, 2007.
- [8] B. Katz. Annotating the world wide web using natural language. In *In Proc. of Conference on Computer Assisted Information Searching on the Internet (RIAO '97), Montreal, Canada, 1997.*, 1997.
- [9] Yannis Kotidis, Amélie Marian, and Divesh Srivastava. Circumventing data quality problems using multiple join paths. In *Proc. of the First International VLDB Workshop on Clean Databases (CleanDB'06)*, 2006.
- [10] Cody C. T. Kwok, Oren Etzioni, and Daniel S. Weld. Scaling question answering to the web. In *Proc. of the 10th International World Wide Web Conference (WWW '01)*, 2001.
- [11] Amélie Marian, Nicolas Bruno, and Luis Gravano. Evaluating top-k queries over web-accessible databases. *ACM Transactions on Database Systems*, 29(2), 2004.