Online Data Fusion

Xuan Liu
National University of Singapore
liuxuan@comp.nus.edu.sg

Xin Luna Dong
AT&T Labs-Research
lunadong@research.att.com

Beng Chin Ooi
National University of Singapore
ooibc@comp.nus.edu.sg

Divesh Srivastava
AT&T Labs-Research
divesh@research.att.com

ABSTRACT

The Web contains a significant volume of structured data in various domains, but a lot of data are dirty and erroneous, and they can be propagated through copying. While data integration techniques allow querying structured data on the Web, they take the union of the answers retrieved from different sources and can thus return conflicting information. Data fusion techniques, on the other hand, aim to find the true values, but are designed for offline data aggregation and can take a long time.

This paper proposes SOLARIS, the first *online* data fusion system. It starts with returning answers from the first probed source, and refreshes the answers as it probes more sources and applies fusion techniques on the retrieved data. For each returned answer, it shows the likelihood that the answer is correct, and stops retrieving data for it after gaining enough confidence that data from the unprocessed sources are unlikely to change the answer. We address key problems in building such a system and show empirically that the system can start returning correct answers quickly and terminate fast without sacrificing the quality of the answers.

1. INTRODUCTION

The Web contains a significant volume of structured data in various domains such as finance, technology, entertainment, and travel; such data exist in deep web databases, HTML tables, HTML lists, and so on. Advances in data integration technologies have made it possible to query such data [4]; for example, a vertical search engine accepts queries on the schema it provides (often through a Web form), retrieves answers from the deep-web sources, and returns the union of the answers. Very often different Web sources provide information for the same data item; however, there is a fair amount of dirty and erroneous information on the Web, so data from different sources can often conflict with each other: from different websites we may find different addresses for the same restaurant, different business hours for the same supermarket at the same location, different closing quotes for the same stock on the same day, and so on. In addition, the Web has made it convenient to copy data between sources, so inaccurate data can be quickly propagated. Integration systems that merely take the union of the answers from various sources can thus return conflicting answers, leaving the difficult decision of which answers are correct to end users.

Proceedings of the VLDB Endowment, Vol. 4, No. 11

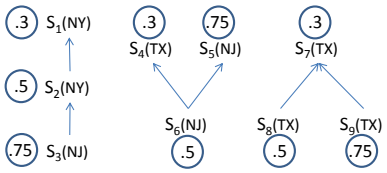


Figure 1: Sources for the motivating example. For each source we show the answer it provides for query "*Where is AT&T Shannon Labs*" in parenthesis and its accuracy in a circle. An arrow from S to S' means that S copies some data from S'.

Table 1: Output at each time point in the motivating example. The time is made up for the purpose of illustration.

Time	Output			Probed source
	Answer	Probability	Probability range	
Sec 1	TX	.4	(0,1)	S_9
Sec 2	TX	.22	(0,1)	S_5
Sec 3	NJ	.94	(0,1)	S_3
Sec 4	NJ	.84	(0,1)	S_4
Sec 5	NJ	.92	(0,1)	S_6
Sec 6	NJ	.97	(.001,1)	S_2
Sec 7	NJ	.97	(.014,1)	S_1
Sec 8	NJ	.98	(.45,1)	S_7

Recently, a variety of data fusion techniques [7] have been proposed to resolve conflicts from different sources and create a consistent and clean set of data. Advanced fusion techniques [2, 6, 8, 14, 16] aim to discover the true values that reflect the real world. To achieve this goal, they not only consider the number of providers for each value, but also reward values from trustworthy sources and discount votes from copiers. Such techniques are designed for offline data aggregation; however, aggregating all information on the Web and applying fusion offline is infeasible because of both the sheer volume and the frequent update of Web data. On the other hand, the whole process can be quite time-consuming and inappropriate for query answering at runtime.

This paper describes SOLARIS, the first *online* data fusion system. Instead of waiting for data fusion to complete and returning all answers in a batch, SOLARIS starts with returning the answers from the first probed source, then refreshes the answers as it probes more sources. For each returned answer, it shows the likelihood that the answer is correct based on the retrieved data and knowledge of the source quality. When the system gains enough confidence that data from the unprocessed sources are unlikely to change the returned answers, it terminates without necessarily probing all sources. Thus, SOLARIS can significantly reduce the latency in query answering, as the next example illustrates.

EXAMPLE 1.1. *Consider answering "*Where is AT&T Shannon Labs?*" on 9 data sources shown in Figure 1. These sources provide three different answers, among which* NJ *is correct. Traditional data integration systems will return all of them to the user.*

SOLARIS *starts with probing* S_9*, returning* TX *with probability .4 (see Table 1; we describe how we order the sources and compute the probability later). It then probes* S_5*, observing a different answer* NJ*; as a result, it lowers the probability for answer* TX *(or switches to* NJ*). Next, it probes* S_3 *and observes* NJ *again, so it refreshes the answer to* NJ *with a probability .94. Probing sources* S_4, S_6, S_2, S_1 *and* S_7 *does not change the answer, and the probability first decreases a little bit but then gradually increases to .98. At this point, the system is confident enough that data from* S_8 *are unlikely to change the answer and terminates. Thus, the user starts to see the correct answer after 3 sources are probed rather than waiting till the system completes probing all 9 sources.* □

There are three challenges in building SOLARIS. First, as we probe new sources and return answers to the users, we wish to *quantify our confidence* for the answers and show that to users. The confidence we return for each answer should consider not only the data we have observed, but also the data we expect to see from the unseen sources considering their accuracy and the copying relationships they may have with the probed sources. Second, online fusion requires both *fast response* and *quick answer refreshing*, so we need to find the answers that are likely to be correct and compute their probabilities quickly. Third, we wish to probe the sources in an order such that we can *return high-quality answers early and terminate fast*. A good source-ordering strategy is essential for quickly converging to the correct answers, computing high probabilities for them, and terminating fast.

This paper takes a first step towards building an online data fusion system and makes the following contributions.

- We propose a framework for online data fusion.
- We define for each returned answer its *expected, maximum*, and *minimum* probability based on our observation of the retrieved data and our knowledge of source quality. We describe efficient algorithms for computing these probabilities.
- We propose source ordering algorithms that can lead to early returning of correct answers and quick convergence.
- We empirically show that our methods can often return correct answers very quickly, terminate fast without sacrificing the quality of the final answers, and are scalable.

While the system we propose assumes a model that probes the sources sequentially, our techniques are still useful if answer retrieval from different sources is allowed to be conducted in parallel. First, even in such a system, data fusion in itself can be time-consuming on a large number of sources and we can apply our techniques on the retrieved answers. Second, querying *all* sources in parallel can require a lot of resources (*e.g.*, bandwidth); our techniques can help choose the *set* of sources we wish to probe first.

Our approach requires knowledge of accuracy of the sources and copying between the sources. We can estimate source accuracy by checking correctness of sampled data, and derive copying probabilities by applying techniques in [5] on sampled data. Details are outside the scope of this paper.

Related work: SOLARIS is inspired by online aggregation [9], which also refreshes answers as more data are processed and outputs confidence of the answers. Our work is different in that (1) we probe data from multiple sources and we describe source ordering techniques that enable early return of the correct answers and quick termination; (2) fusion techniques are very different from computation of aggregates, leading to different ways of computing expected probabilities and probability ranges; and (3) we consider copying between sources, which raises new challenges such as vote counting when a copier is probed before the copied source.

Our work is built upon advanced data-fusion techniques that aim at resolving conflicts and finding true values [2, 6, 8, 14, 16]; however, these techniques all assume the context of offline data fusion. None of our contributions, including source ordering, incremental vote counting when copiers are probed before the copied sources, and computation of expected, maximum, and minimum probabilities, is addressed in the prior work. We point out that although we base our techniques on the methods proposed in [6], the key idea in our solution can be applied for other fusion techniques; for example, results in Section 3 can be applied when we consider only trustworthiness of sources [14, 16].

Finally, there are works on quality-aware query answering (surveyed in [1]). But either they do not fuse relational data [13], or they focus on other quality measures like coverage of sources [12, 10, 11, 15]. To the best of our knowledge, our paper is the first one that considers source accuracy and copying in an online fashion.

Outline: In the rest of the paper, Section 2 reviews fusion techniques and Section 3 proposes the framework of online data fusion. Section 4 considers the copying relationships in online fusion. Section 5 reports experimental results and Section 6 concludes.

2. BACKGROUND FOR DATA FUSION

We start with reviewing existing fusion techniques, based on which we describe our SOLARIS system.

Data sources: Consider integrating data from a set $\mathcal{S}$ of sources, each providing tuples that describe objects in a particular domain (*e.g.*, book, movie, publication). We call an attribute of a particular object instance (*i.e.*, a cell in a table) a *data item* (*e.g.*, title of a book, actor of a movie). We assume that schema mapping techniques have been applied to resolve attribute-label heterogeneity. We also assume that each tuple contains a *key* attribute that can uniquely identify the object the tuple refers to.[1] We consider the case that each non-key data item has a single true value reflecting the real world but the sources may provide wrong values.

We assume knowledge of the following two properties of the data sources, which we rely on in data fusion.

1. *Accuracy:* Different sources may differ in the correctness of their data and we capture this by source *accuracy*. Given a source $S \in \mathcal{S}$, its accuracy, denoted by $\alpha(S)$, is the probability that a value provided by S is correct.
2. *Copying:* A source may copy from others and we capture this by *copying relationship*. A copier can copy all or a part of data from one or multiple sources, and can additionally provide its own data. Given sources $S, S' \in \mathcal{S}, S \neq S'$, the *copying probability*, denoted by $\rho(S \rightarrow S')$, is the probability for each common value that S copies this value from S'. As in [6], we assume there is no mutual copying between a pair of sources; so if $\rho(S \rightarrow S') > 0$, $\rho(S' \rightarrow S) = 0$.[2]

Data fusion: We adopt the fusion techniques proposed in [6], which considers the accuracy of the sources and the copying relationship between the sources in truth finding. In particular, we decide the true value on data item D according to $\mathcal{S}$ in three steps.

[1] In case that such key attributes do not exist, we can apply record linkage techniques to link the records that refer to the same real-world entity.

[2] In case that the copying direction is uncertain, we can choose one direction as described in [6].

Table 2: Vote count of each source in the motivating example.

Source	Independent vote count	Dependent vote count
S_1	$\ln \frac{49*.3}{1-.3} = 3$	$3*1 = 3$
S_2	$\ln \frac{49*.5}{1-.5} = 4$	$4*(1-.8) = .8$
S_3	$\ln \frac{49*.75}{1-.75} = 5$	$5*1 = 5$
S_4	$\ln \frac{49*.3}{1-.3} = 3$	$3*1 = 3$
S_5	$\ln \frac{49*.75}{1-.75} = 5$	$5*1 = 5$
S_6	$\ln \frac{49*.5}{1-.5} = 4$	$4*(1-.8) = .8$
S_7	$\ln \frac{49*.3}{1-.3} = 3$	$3*1 = 3$
S_8	$\ln \frac{49*.5}{1-.5} = 4$	$4*(1-.8) = .8$
S_9	$\ln \frac{49*.75}{1-.75} = 5$	$5*(1-.8) = 1$

1. For each source $S \in \mathcal{S}$ that provides data on a data item D, we compute its *independent vote count* as $C^{\perp}(S) = \ln \frac{n\alpha(S)}{1-\alpha(S)}$, where n is the number of wrong values in the domain for D; thus, a source with a higher accuracy has a higher independent vote count. Assuming copying relationships between different pairs of sources are independent, we compute the *dependent vote count* of S on D as $C^{\rightarrow}(S) = C^{\perp}(S)\Pi_{S' \in \bar{S}_D(S)}(1-\rho(S \rightarrow S'))$, where $\bar{S}_D(S)$ denotes the set of sources that provide the same value as S on D. Thus, $C^{\rightarrow}(S)$ is a fraction of the independent vote count according to the copying probability and $C^{\rightarrow}(S) \leq C^{\perp}(S)$ (equal when S independently provides the value).
2. For each value v in the domain of D, denoted by $\mathcal{D}(D)$, we compute its vote count as the sum of the dependent vote counts of its providers, denoted by $C(v)$. The value with the highest vote count is considered as the true value.
3. In case we need to compute the probability of a value v being true, we apply equation $Pr(v|\mathcal{S}) = \frac{e^{C(v)}}{\sum_{v_0 \in \mathcal{D}(D)} e^{C(v_0)}}$. This equation is derived from Bayesian analysis, where $e^{C(v)}$ is proportional to the probability of our observed data conditioned on v being true, and we assume the same a-priori probability for each value being true.

EXAMPLE 2.1. *Consider the sources in the motivating example. Assume for each copying relationship from S to S', $\rho(S \rightarrow S') = .8$. To fuse answers from all sources, we first compute the vote count for each source and obtain the results in Table 2 (there are 50 values in the domain). Note that although S_3 is a copier of S_2, it provides a different answer so cannot copy this value from S_2; thus, its dependent vote count is the same as its independent one. Similarly, S_6 cannot copy its value from S_4, so its vote count is $4*.2 = .8$ rather than $4*.2^2 = .16$.*

Thus, the vote count of NJ *is $5+5+.8 = 10.8$; that of* TX *is $3+3+.8+1 = 7.8$; that of* NY *is $3+.8 = 3.8$; and that of the other 47 values is 0. So* NJ *is the correct answer with probability $\frac{e^{10.8}}{e^{10.8}+e^{7.8}+e^{3.8}+e^{0}*47} = .95$. Note that if we apply naive voting or consider only source accuracy, we will return* TX *instead.* □

3. FRAMEWORK OF ONLINE FUSION

We consider select-project queries where the select predicates are posed on the key attribute and the key attribute is in the project list. Such queries are popular in many applications such as vertical search and we discuss other queries (*e.g.*, queries with joins or select predicates on non-key attributes) in Appendix C. For simplicity of understanding, we explain our techniques for the case where all sources have full coverage. We describe extensions for considering coverage in Appendix C. We leave a full-fledged combination of our techniques and those that consider coverage and overlap in integration [3, 12] for future work.

Algorithm 1: FUSIONWACCU($\mathcal{S}$, $\bar{D}$)

Input : $\mathcal{S}$ sources in decreasing order of their accuracy; $\bar{D}$ queried data items
Output : True values for $\bar{D}$; for each returned value, return in addition $expPr(v), minPr(v)$ and $maxPr(v)$.

// Initialization
1 $stop[D \in \bar{D}] \leftarrow$false; $vote[D \in \bar{D}][v \in \mathcal{D}(d)] \leftarrow 0$;
2 $remain \leftarrow \sum_{S \in \mathcal{S}} \alpha(S)$;
3 **while** $\exists D \in \bar{D}$ *s.t.* $stop[D] =$*false* **do**
// Probe the next source in the list
4 $S \leftarrow$ the next source in the list;
5 $remain \leftarrow remain - \alpha(S)$;
6 **foreach** $D \in \bar{D}$ **do**
7 **if** $!stop[D]$ **then**
// 1. Truth finding
8 $vote[D][S(d)] \leftarrow vote[D][S(d)] + C^{\perp}(S)$; *// $S(d)$ is the value provided by S on D*
9 find the value v_1 with the maximum vote count and v_2 with the top-2 vote count;
// 2. Probability computation
10 compute $expPr(v_1), maxPr(v_1), minPr(v_1)$ according to $vote[D]$ and $remain$;
// 3. Termination justification
11 compute $Pr(v_2)$ according to $vote[D]$;
12 **if** $minPr(v_1) > Pr(v_2)$ **then**
13 $stop[D] \leftarrow$ true;
14 Refresh answers in the output;

SOLARIS returns answers as it incrementally probes the sources, and terminates when it believes that data from the rest of the sources are unlikely to change the answers. There are four major components for such a system: *truth finding*, *probability computation*, *termination justification*, and *(offline) source ordering*. Algorithm FUSIONWACCU illustrates how we instantiate these components in case that all sources are independent and we only consider accuracy of the sources in fusion.

Truth finding: As we probe a new source, we find the truth based on the already probed sources, denoted by $\bar{S}$ (Lines 8-9). The key question to ask is "*how to incrementally count the votes such that we can efficiently decide the correct values as we probe each new source*?" In case all sources are independent, incremental vote counting is straightforward: when we probe a new source S, we add $C^{\perp}(S)$ to the vote count of the value it provides.

Probability computation: For each value v that we have determined to be correct, we return the expected probability and the probability range of this value being true (Line 10). To compute these probabilities, we consider all possible worlds that describe the possible values provided by the unseen sources $\mathcal{S} \setminus \bar{S}$, denoted by $\mathbf{W}(\mathcal{S} \setminus \bar{S})$. For each possible world $W \in \mathbf{W}(\mathcal{S} \setminus \bar{S})$, we denote by $Pr(W)$ its probability and by $Pr(v|\bar{S}, W)$ the probability that v is true based on data provided in the possible world. Then, the maximum probability of v is the maximum probability computed among all possible worlds (similarly for minimum probability), and the expected probability of v is the sum of these probabilities weighted by the probabilities of the possible worlds. We formally define them as follows.

DEFINITION 3.1 (EXPECTED/MAX/MIN PROBABILITY). *Let $\mathcal{S}$ be a set of data sources and $\bar{S} \subseteq \mathcal{S}$ be the probed sources. Let v be a value for a particular data item. The* expected probability *of v, denoted by $expPr(v|\bar{S})$, is defined as*

$$expPr(v|\bar{S}) = \sum_{W \in \mathbf{W}(\mathcal{S} \setminus \bar{S})} Pr(W)Pr(v|\bar{S}, W). \quad (1)$$

The maximum probability *of v, denoted by $maxPr(v|\bar{S})$, is defined as (similarly for* minimum probability*)*

$$maxPr(v|\bar{S}) = \max_{W \in \mathbf{W}(\mathcal{S} \setminus \bar{S})} Pr(v|\bar{S}, W). \quad \square \tag{2}$$

The key question to ask is "*how to efficiently compute the expected, maximum, and minimum probabilities based on the counted votes*?" We describe our solution for the independence case shortly.

Termination justification: As we probe the sources, the results often converge before we finish probing all sources. In such situations, we wish to terminate early. We thus check for each data item a termination condition and stop retrieving data for it if the condition is satisfied (Lines 11-13).

To guarantee that probing more sources will not change the returned value v for data item D, we should terminate only if for each $v' \in \mathcal{D}(D), v' \neq v$, we have $minPr(v) > maxPr(v')$. However, satisfying this condition for each returned value is often hard. We can loosen it in two ways: (1) for the value v' with the top-2 vote count, $minPr(v) > Pr(v')$ (or $expPr(v')$); (2) for such v', $Pr(v)$(or $expPr(v)$)$> maxPr(v')$. Our experiments show that these loose conditions lead to much faster termination, while sacrificing the quality of the results only a little, if at all.

Source ordering: The algorithm assumes an ordered list of sources as input and probes the sources in the given order. We wish to order the sources such that 1) we can return the correct answers as early as possible, and 2) we can terminate as soon as possible. To reduce the overhead at runtime, we conduct source ordering offline. The key question to ask is "*how to order the sources such that we can quickly obtain the correct answers and terminate early*?" Intuitively, when the sources are independent, we should order the sources in decreasing order of their accuracy.

3.1 Probability computation for independent sources

Consider a value v and a set $\bar{S} \subseteq \mathcal{S}$ of probed sources. We can compute $Pr(v|\bar{S})$ according to Section 2. In fact, we can prove that the expected probability for v is exactly the same as $Pr(v|\bar{S})$ (proofs of all results in the Appendix). The intuition is that the probability of an unseen source providing v or any other value fully depends on the probability of v being true, which is computed from data in $\bar{S}$; thus, the unseen source does not introduce any new information and so cannot change the expected probability.

THEOREM 3.2. *Let $\mathcal{S}$ be a set of independent sources, $\bar{S} \subseteq \mathcal{S}$ be the sources that we have probed, and v be a value for a particular data item. Then, $expPr(v|\bar{S}) = Pr(v|\bar{S})$.* $\square$

For the maximum probability of value v, it is obvious that we obtain it when all unseen sources provide v.

THEOREM 3.3. *Let $\mathcal{S}$ be a set of independent sources, $\bar{S} \subseteq \mathcal{S}$ be the sources that we have probed, and v be a value for a data item D. Let W be a possible world in which all sources in $\mathcal{S} \setminus \bar{S}$ provide value v on D. Then, $maxPr(v|\bar{S}) = Pr(v|\bar{S}, W)$.* $\square$

Obtaining the minimum probability of value v certainly requires that none of the unseen sources provides v. Among the rest of the values, we can prove that if all unseen sources provide the same value, and the value has the highest probability to be true according to the probed sources, we obtain the minimum probability for v.

THEOREM 3.4. *Let $\mathcal{S}$ be a set of independent sources, $\bar{S} \subseteq \mathcal{S}$ be the sources that we have probed, v be a value for a data item D, and $v_{max} = argmax_{v' \in \mathcal{D}(D) - \{v\}} Pr(v'|\bar{S})$. Let W be a possible world in which all sources in $\mathcal{S} \setminus \bar{S}$ provide value v_{max} on D. Then, $minPr(v|\bar{S}) = Pr(v|\bar{S}, W)$.* $\square$

4. CONSIDERING COPYING IN ONLINE FUSION

Algorithm FUSIONWACCU falls short in the presence of copying. First, *vote counting* is non-trivial: if we probe a copier before the copied source, we do not know if they provide the same value on a data item and hence, whether we should use the independent or dependent vote count for the copier. Second, *ordering the sources* by accuracy may not lead to fast convergence: if the top-accuracy sources have copying relationships between them, the vote counts can increase slowly as we discount copied values.

This section proposes two solutions for vote counting when a copier is probed earlier than the copied source: the *conservative* approach and the *pragmatic* approach. Each approach can lead to a different source-ordering strategy. Our experiments show that the pragmatic approach always outperforms the conservative one. In our description we call S a *child* of S' and S' a *parent* of S if S copies from S'. We denote by $Pa(S, S')$ the parent of S on the copying path from S to S'.[3]

4.1 Vote counting

We propose two vote-counting approaches, both observing the following *no-over-counting* principle: *for each value, among its providers that could have copying relationships on it, at any time we apply the independent vote count for at most one source*. This principle avoids bias from copied values at any time. We next describe incremental vote counting for each approach.

Conservative approach: The *conservative* approach assumes that for each data item the copier provides the same value as the copied source, so applies its dependent vote count at the beginning, and increases the vote count if it observes a different value from the copied source. Thus, when we probe a new source S, we shall consider its own vote count and the vote counts of its copiers.

1. Suppose S provides value v. Among its parents, S may copy from any one that has not been probed or is observed to also provide v. We denote the set of such parents by $\bar{P}(S)$. Thus, the vote count of S for v is
$$C^{\perp}(S)\Pi_{S_p \in \bar{P}(S)}(1 - \rho(S \to S_p)).$$
2. Suppose S provides a different value from its child S_c. Then, S_c cannot copy from S and we should increase its vote count. Let $v' \neq v$ be the value provided by S_c and $C(S_c)$ be S_c's current vote count. We shall increase the vote count of v' by
$$\frac{C(S_c)}{1 - \rho(S_c \to S)} - C(S_c) = \frac{C(S_c)\rho(S_c \to S)}{1 - \rho(S_c \to S)}.$$

Obviously, this approach guarantees that *the vote count of each value increases monotonically*. However, it may under-estimate the vote count of a value if all the probed providers are copiers.

Pragmatic approach: The *pragmatic* approach assumes that for each data item the copier provides a different value from the copied source, so applies its independent vote count at the beginning, and decreases the vote count when observing the same value from the copied source. We consider both directly and transitively copied sources to avoid violation of the no-over-counting principle. Accordingly, when we probe S, we shall update its own vote count and the vote count of its closest probed descendant (as we will show in Section 4.3, our ordering guarantees that there can be only one such descendant).

[3]We assume a single such parent and can easily extend our techniques when there are multiple such parents.

Table 3: Example 4.1. Vote count of *NY* and *NJ* as we probe $S_1 - S_3$ in the order of S_3, S_2, S_1.

Source	*NY*	*NJ*
S_3	0	1
S_2	.8	5
S_1	3.8	5

(a) Conservative approach.

Source	*NY*	*NJ*
S_3	0	5
S_2	4	5
S_1	3.8	5

(b) Pragmatic approach.

1. Suppose S provides value v. Among its closest probed ancestors, S can copy only from those that also provide v; we denote this set by $\bar{A}$. We compute S's vote count as
$$C^{\perp}(S)\Pi_{S_a \in \bar{A}}(1 - \rho(S \to Pa(S, S_a))).$$
2. Consider the closest probed descendant of S, denoted by S_d. There are two cases. First, if S_d provides the same value as S and none of S's closest probed ancestors is observed to also provide v (*i.e.*, $\bar{A} = \emptyset$), we must have applied the independent vote count of S_d (with respect to S) and need to decrease it. Let $C(S_d)$ be the current vote count of S_d and $S_p = Pa(S_d, S)$. We shall decrease the vote count of v by
$$C(S_d) - C(S_d)(1 - \rho(S_d \to S_p)) = C(S_d)\rho(S_d \to S_p).$$
Second, if S_d provides a different value from S but the same value as one of S's closest probed ancestors, we must have applied the dependent vote count of S_d and need to increase it to the independent vote count. Let $v' \neq v$ be the value provided by S_d, $C(S_d)$ be the current vote count of S_d, and $S_p = Pa(S_d, S)$. We shall increase the vote count of v' by
$$\frac{C(S_d)}{1 - \rho(S_d \to S_p)} - C(S_d) = \frac{C(S_d)\rho(S_d \to S_p)}{1 - \rho(S_d \to S_p)}.$$

This approach does not guarantee monotonicity, but applies the independent vote count to exactly one source among those that have copying relationships, so avoids over-counting *and* under-counting.

EXAMPLE 4.1. *Continue with the motivating example and consider probing sources $S_1 - S_3$ in the order of S_3, S_2, S_1. Table 3 shows the vote counts of* NY *and* NJ *as we probe each source. In the conservative approach, we first add the dependent vote count (*1*) of S_3 for* NJ*, as its parent (S_2) has not been probed. We next add the dependent vote count (*.8*) of S_2 for* NY*; as it provides a different value from S_3, we increase the vote count of* NJ *by $\frac{1}{.2} - 1 = 4$. Finally, we probe S_1 and add its independent vote count* 3 *for* NY*.*

In the pragmatic approach, we first probe S_3 and add 5 *to the vote count of* NJ*. We next probe S_2 and add* 4 *to the vote count of* NY*; since S_2 provides a different value from S_3, we do not change S_3's vote. Last, we probe S_1, adding* 3 *to the vote count of* NY *and reducing the vote count of S_2 by $4 - 4 * .2 = 3.2$. The final vote count for each value is the same in both approaches.* □

4.2 Probability computation

Computing the expected, maximum and minimum probability for a value is much more tricky when we consider copying. The reason, again, is that for a source S, the observation of whether S's parents provide the same value may change its vote count. We describe how we approximate these probabilities efficiently. Note that the estimated maximum and minimum probabilities are looser bounds so still "correct" to show the users, and the estimated expected probability is close to the real one; finally, these estimates will also be used in termination justification and our experiments show that they do not sacrifice the quality of results.

Expected probability: First, we show that when none of the unseen sources is a parent of a probed source, the expected probability of a value is the same as the probability computed according to the probed sources. The intuition is that among unseen data provided for the data item, those that are independently provided will not change the expected probability, for the same reason as discussed in Section 3.1; those that are copied will not be considered in vote counting and so will not affect the expected probability either.

THEOREM 4.2. *Let $\mathcal{S}$ be a set of sources, $\bar{S} \subseteq \mathcal{S}$ be the probed sources, and v be a value. If $\rho(S \to S') = 0$ holds for each $S \in \bar{S}$ and $S' \in \mathcal{S} \setminus \bar{S}$, then, $expPr(v|\bar{S}) = Pr(v|\bar{S})$.* □

However, as the following example shows, if a probed source copies from an unseen source, the theorem does not hold any more.

EXAMPLE 4.3. *Consider a data item D, where $\mathcal{D}(D) = \{0, 1\}$. Consider three sources. Sources S_1 and S_2 are independent and both have accuracy .6; thus, $C^{\perp}(S_1) = C^{\perp}(S_2) = \ln \frac{1*.6}{1-.6} = .4$. Source S_3 is a copier of S_1 with $\rho(S_3 \to S_1) = .8$; it has accuracy .9, so $C^{\perp}(S_3) = \ln \frac{1*.9}{1-.9} = 2.2$. Suppose we have probed S_2, observing value* 0*, and probed S_3, observing* 1*. We next compute the expected probability for value* 1*.*

The conservative approach uses the dependent vote count of S_3 ($2.2 * .2 = .44$). The probability for 1 *is then $\frac{e^{.44}}{e^{.44}+e^{.4}} = .51$, so S_1 has probability $.51 * .6 + .49 * .4 = .5$ to provide* 1*. If S_1 provides* 1*, the probability for* 1 *becomes $\frac{e^{.4+.44}}{e^{.4+.44}+e^{.4}} = .61$. Otherwise, S_3 cannot copy from S_1 so we shall use the independent vote count of S_3; the probability then becomes $\frac{e^{2.2}}{e^{2.2}+e^{.4+.4}} = .8$. The expected probability for* 1 *is thus $.61 * .5 + .8 * .5 = .71 > .51$.*

The pragmatic approach uses the independent vote count of S_3. The probability for 1 *is then $\frac{e^{2.2}}{e^{2.2}+e^{.4}} = .86$, so S_1 has probability $.86 * .6 + .14 * .4 = .57$ to provide* 1*. Similarly, the expected probability for* 1 *is $.61 * .57 + .8 * .43 = .69 < .86$.* □

The discrepancy in this example is because the observation of data from unseen sources will change our belief of whether the copier copies on a particular data item. We next show results that lead to an approximation of the expected probability.

THEOREM 4.4. *Let $\bar{S} \subseteq \mathcal{S}$ be a set of probed sources such that for one and only one $S \in \bar{S}$, there exists $S' \in \mathcal{S} \setminus \bar{S}$ where $\rho(S \to S') > 0$. Let v be a value of a particular data item. Let $Pr^{con}(v|\bar{S})$ (resp. $Pr^{pra}(v|\bar{S})$) denote the probability of v computed in the conservative (resp. pragmatic) approach, and $expPr^{con}(v|\bar{S})$ denote the expected probability in the conservative approach. Then, (similar for the pragmatic approach)*

1. $Pr^{con}(v|\bar{S}) < expPr^{con}(v|\bar{S}) < Pr^{pra}(v|\bar{S})$;
2. $|\frac{Pr^{con}(v|\bar{S}) + Pr^{pra}(v|\bar{S})}{2} - expPr^{con}(v|\bar{S})| < \frac{1}{4}$. □

We estimate the expected probability by $\frac{Pr^{con}(v|\bar{S}) + Pr^{pra}(v|\bar{S})}{2}$. In Example 4.3, the approximation leads to $\frac{.51+.86}{2} = .685$, close to the two expected probabilities that we have computed. Complexity of computing the expected probability remains an open problem.

Maximum or minimum probability: We first show that computing maximum or minimum probability of a value is tractable.

THEOREM 4.5. *Given $\bar{S} \subseteq \mathcal{S}$ and value v, computing $maxPr(v|\bar{S})$ and $minPr(v|\bar{S})$ is in PTIME.* □

Although we can compute a tight bound of value probability in polynomial time, the algorithm is still quite costly and not suitable for an online process. We next describe how we compute a loose (but still fairly tight) bound for minimum probability and we can compute the maximum probability similarly.

To minimize the probability of value v, we shall minimize $C(v)$ and maximize $C(v')$ for each $v' \neq v$. Our algorithm, MINPR (details in Appendix B) does so in four steps.

1. To minimize $C(v)$, for each of v's probed provider $S \in \bar{S}$ that 1) has an unseen parent S_p, and 2) satisfies $C^{\perp}(S) > C^{\perp}(S_a) + C^{\rightarrow}(S)$ for the ancestor S_a that leads to the minimum $C^{\perp}(S_a) + C^{\rightarrow}(S)$, we use $C^{\perp}(S_a) + C^{\rightarrow}(S)$ as its vote count.
2. To maximize $C(v'), v' \neq v$, from the probed sources, for each $S \in \bar{S}$ that does not provide v and has an unseen parent, we use its independent vote count $C^{\perp}(S)$.
3. Let v_{max} be the value with the highest vote count among all values other than v after Step 2. To maximize $C(v_{max})$ from unseen sources, we assume they all provide v_{max} independently and use their independent vote count.
4. Compute the probability of v accordingly.

THEOREM 4.6. *Let $\mathcal{S}$ be a set of sources, $\bar{S} \subseteq \mathcal{S}$ be the probed ones, v be a value, and f_a be the maximum number of ancestors a source has. Algorithm* MINPR *finishes in time $O(f_a|\bar{S}|)$ and its result M satisfies $M \leq minPr(v|\bar{S})$.* □

EXAMPLE 4.7. *Consider sources in Figure 1 and assume we have probed all sources except S_8. Consider the minimum probability of* NJ. *Step 1 and Step 2 will not change vote count of any probed source. Step 3 assumes S_8 also provides* TX *and uses its independent vote count* 4*. Thus, $minPr(NJ|\mathcal{S}/\{S_8\}) = \frac{e^{10.8}}{e^{10.8}+e^{7+4}+e^{3.8}+e^{0}*47} = .45$. At this time we compute a probability of .02 for the top-2 value* TX. *If we use the termination condition $minPr(NJ) > Pr(TX)$, we can terminate then.* □

The full online algorithm, FUSIONWCOPY (Appendix B), summarizes the techniques in Section 4.1-4.2. Each round takes time $O(f_a|\bar{S}||\bar{D}|)$. Note however that in early rounds we have not probed many sources and $|\bar{S}|$ is small, and in late rounds the number of remaining data items is often much less than $|\bar{D}|$.

4.3 Source ordering

Finally, we discuss ordering of sources according to the two vote-counting approaches. A source can have different vote counts for different values under consideration of copying. We order the sources by their *minimum* vote counts, which can be obtained in the extreme case where all sources provide the same value. (The *maximum* vote count is the independent vote count.)

Conservative approach: When all sources provide the same value, in the conservative approach the vote count of a source is fixed as its dependent vote count. Let S be a source and $\overline{Pa}(S)$ be its parents. Then, this *fixed vote count* of S is computed by

$$C(S) = C^{\perp}(S)\Pi_{S_p \in \overline{Pa}(S)}(1 - \rho(S \rightarrow S_p)). \quad (3)$$

We order the sources in decreasing order of their fixed vote count. As a result, we often order an independent source before its copier, even if the copier has a higher accuracy.

Pragmatic approach: When all sources provide the same value, as we probe a new source in the pragmatic approach, we may need to decrease the vote count of its probed descendants. We thus order the sources iteratively, each time choosing the one that increases the total vote count most. In particular, given a source S and a set of probed sources $\bar{S}$, we compute the *conditional vote count* of S as

$$C(S|\bar{S}) = C(\bar{S} \cup \{S\}) - C(\bar{S}), \quad (4)$$

where $C(\bar{S})$ denotes the total vote count of $\bar{S}$ if all sources in $\bar{S}$ provide the same value. In other words, $C(\bar{S} \cup \{S\})$ serves as an *invariant* for deciding $C(S|\bar{S})$. We compute $C(\bar{S})$ as follows: for each $S \in \bar{S}$, if S has an ancestor in $\bar{S}$, we consider S may (directly or transitively) copy from its ancestor and take its dependent vote count; otherwise, we consider S provides the value independently and take its independent vote count.

Table 4: Example 4.10: Vote counts computed in source ordering. The maximum vote count in each round of the pragmatic approach is in bold font.

Method	Rnd	S_1	S_2	S_3	S_4	S_5	S_6	S_7	S_8	S_9
Fixed		3	.8	1	3	5	.16	3	.8	1
Cond	1	3	4	**5**	3	**5**	4	3	4	**5**
	4	-1	0	-	**3**	-	.8	-1	4	-
	5	-1	0	-	-	-	**.16**	-1	4	-
	6	-1	**0**	-	-	-	-	-1	4	-
	7	**-.2**	-	-	-	-	-	-1	4	-
	8	-	-	-	-	-	-	**-1**	4	-

EXAMPLE 4.8. *Consider the sources in Figure 1. First consider S_2 and $\bar{S} = \{S_3\}$. We have $C(\bar{S}) = 5$, $C(\bar{S} \cup \{S_2\}) = 4 + 5 * .2 = 5$, so $C(S_2|\bar{S}) = 5 - 5 = 0$. Now consider S_1 and $\bar{S}' = \{S_2, S_3\}$ (transitive copying). We have $C(\bar{S}') = 5$, $C(\bar{S}' \cup \{S_1\}) = 3 + 4 * .2 + 5 * .2 = 4.8$, so $C(S_1|\bar{S}') = 4.8 - 5 = -.2$. Next consider S_4 and $\bar{S}'' = \{S_5, S_6\}$ (multi-source copying). We have $C(\bar{S}'') = 5 + 4 * .2 = 5.8$, $C(\bar{S}'' \cup \{S_4\}) = 3 + 5 + 4 * .2 * .2 = 8.16$, so $C(S_4|\bar{S}'') = 8.16 - 5.8 = 2.36$.* □

As shown in Example 4.8, when S is a parent of a probed copier S_c and has lower accuracy than S_c, $C(S|\bar{S})$ can be negative. This is due to the assumption that the accuracy of the copied data is the same as that of the copied source [6], which can be much lower than that of the copier. This negative vote count can put S to the end of the ordered list, which may be actually desired because of its low accuracy and its dependence with S_c.

Note however that such a vote counting strategy can fall short in the presence of co-copying. In Figure 1, if we probe S_8 and S_9 before S_7, their total vote count is $4 + 5 = 9$, violating the no-over-counting principle. We should apply the independent vote count only for S_8 *or* S_9, but different choices can lead to different results. Our solution is to guarantee that we never probe two co-copiers if none of their common ancestors is probed. We formalize the condition as follows.

DEFINITION 4.9 (CO-COPIER CONDITION). *Let S and S' be two sources where neither one is the ancestor of the other. For each of their closest ancestor S_a, we shall probe S_a or one of its ancestors before we probe both S and S'.* □

Accordingly, our source ordering algorithm proceeds in four steps.

1. Initialize $\bar{S} = \emptyset$ and set $C(S|\bar{S}) = C^{\perp}(S)$ for each S.
2. Among the sources that satisfy the co-copier condition, select the one with the highest vote count and add it to $\bar{S}$.
3. Adjust the conditional vote count for unselected sources.
4. Go to Step 2, until all sources are selected (*i.e.*, $\bar{S} = \mathcal{S}$).

The full algorithm, shown in Appendix B, takes time $O(f^2|\mathcal{S}|^2)$, where f is the maximum number of ancestors and descendants a source has. This is reasonable given that source ordering is offline, and the number of sources for a particular domain is rarely huge.

Pragmatic ordering has the advantage of often ordering high-accuracy sources early and meanwhile taking copying into consideration. We next illustrate its benefit using an example.

EXAMPLE 4.10. *Consider ordering the sources in Figure 1. Table 4 shows the fixed vote counts and the conditional vote counts we compute in each round, and we order the sources accordingly (pragmatic ordering randomly chooses S_5 first among the sources that have a tie). Note that in the pragmatic approach, although*

S_8 has a higher conditional vote count than S_7, it is ranked later because otherwise the co-copier condition would be violated.

The pragmatic order is better as it ranks the most accurate sources earlier even if some of them are copiers. Indeed, if we follow the conservative order, the result will not converge to NJ *until we have probed 6 sources (3 for the pragmatic order).* □

5. EXPERIMENTAL RESULTS

This section presents an experimental study of our online fusion system on a real-world data set. We show that (1) our system can quickly return the correct values for most of the queried items; (2) our system scales well; and (3) in presence of copying, the pragmatic approach is the most effective among various approaches.

5.1 Experiment setup

Data: We experimented on the AbeBooks data set, which were extracted in 2007 from *AbeBooks.com* by searching computer-science books.[4] In the data set there are 894 bookstores (data sources), 1263 books, and 24364 listings, each provided by a bookstore and containing attributes ISBN, name, and authors. We normalized the author lists to a standard format.

We applied techniques in [5, 6] for computing source accuracy and detecting copying; we found copying between 1758 pairs of sources. Note that the computed quality measures may not be exactly the same as the real ones. Most experiments are conducted over 100 sources with the largest coverage on a set of 100 books that we describe shortly. Their coverage ranges from 0.02 to 0.87; their accuracy ranges from 0.005 to 0.74; and we found copying between 774 pairs of sources for this subset of sources (see Figure 11 in the appendix for distribution of coverage vs. accuracy). If S copies from S' through a source outside the subset, we treat S and S' as independent because we seldom observe big overlap of data in case of transitive copying (same for co-copying). On this subset of data, each book on average has 18.6 listings and the number ranges from 2 to 49.

Query and measure: We used a golden standard that contains 100 randomly selected books and the list of authors found on the cover of each book. We considered queries that ask for authors of a subset of these books. We measured *precision* of the results by the percentage of correctly returned author lists.

Implementation: We implemented four online algorithms: NAIVE probes *all* sources in a random order and repeatedly applies prior fusion techniques (Section 2) on probed sources; ACCU applies FUSIONWACCU; CONSERVATIVE applies FUSIONWCOPY with conservative ordering and vote counting; and PRAGMATIC applies FUSIONWCOPY with pragmatic ordering and vote counting. The latter three methods apply termination condition $minPr(v_1) > Pr(v_2)$, where v_1 and v_2 are the top-1 and top-2 values. As NAIVE uses random order, we ran it 5 times and reported the average.

We used Java and experimented on a Windows7 machine with 2.33GHz Intel CPU and 4GB of RAM.

5.2 Experimental results

Query answering behavior: We first considered the query that asks for authors of all 100 books and reported our observation on query answering by PRAGMATIC. Figure 2-3 plot as we probe each source, (1) the total number of returned books, (2) the number of correctly returned values, (3) the number of books on which the returned values do not change any more, (4) the number of books on which we stopped retrieving data, and (5) the average expected, minimum, and maximum probabilities for the returned answers.

[4]We thank the authors of [16] for providing us the data.

We have several observations. First, a large fraction of answers quickly get stable: after probing 14 sources, the answers on 73 books get stable; then, this number increases gradually as we probe more sources and reaches 100 at the 97th source. Second, the number of terminated books climbs much more slowly, showing that we typically require more evidence before we decide to stop. Third, the number of correctly returned answers also quickly increases at the beginning and then flattens out, but decreases as we probe the last 32 sources. These sources have very low accuracy (as low as 0.005) and provide a lot of wrong values; even though each of them has a low vote count, accumulatively they can still bias the decision. Fourth, the average expected probability increases gradually as we probe more sources, while the average maximum probability remains 1 till the 96th source, and the average minimum probability remains less than .001 till the 91st source. Finally, we observe big jumps for all numbers at several sources (source 14, 20, etc.), as these sources have high coverage and are independent.

Result precision: We next compared precision of the results by various methods. Figure 4 plots for each method the number of answers that have *stabilized* at the *correct* value as we probe each source. PRAGMATIC has the best performance. (1) Compared with ACCU, PRAGMATIC at the beginning returns more correct values (on average 12 more from source 14 to 32), then returns fewer correct values (on average 2 fewer from source 33 to 46) as it considers the copying relationship and discounts votes from the copied correct values, and eventually returns more correct answers (starting from the 50th source). (2) PRAGMATIC dominates CONSERVATIVE: starting from the 13th source it on average returns 15.2 more correct values. (3) PRAGMATIC at the beginning returns fewer correct values than NAIVE as it first probes sources with high accuracy but maybe low coverage; starting from source 14, PRAGMATIC significantly outperforms NAIVE and on average returns 25 more correct values, as it probes the sources in a better order.

Figure 5 plots the precision of the results as we increase the number of queried books, where we start from more popular books. As we probe more unpopular books, the precision obtained by all methods decreases because of less abundance of information. We observe that PRAGMATIC always obtains the highest accuracy, as it starts from more accurate sources and ignores copied data; in most cases it even beats NAIVE, which probes all data from all sources so can be more affected by the inaccurate sources that are ranked later. ACCU also often beats NAIVE, but it does not perform as well as PRAGMATIC since it ignores copying and can be biased by copied data. Finally, although CONSERVATIVE considers copying, it has the lowest precision because it can terminate on a value after probing a few less accurate sources.

In addition, we reported comparison of various ordering, vote counting, and termination strategies in Appendix D.

Efficiency and scalability: We did two sets of experiments for scalability study. First, we started with the 10 sources with the largest coverage, and gradually added sources until reaching 100 sources. Figure 6 plots the CPU time for fusion for all 100 books on each data set. Among different methods, NAIVE took the longest time (2 orders of magnitude more than PRAGMATIC) and the CPU time increases quadratically, as it retrieved data for all 100 books on all sources and counted votes from scratch as it probes each new source; note that linear increase of CPU time for fusion would require applying incremental vote counting strategies as we described in Section 4.1. The CPU time for the rest of the methods increases linearly. ACCU took the shortest time as its vote-counting process is very simple. PRAGMATIC and CONSERVATIVE are in the middle. Note that although PRAGMATIC spent longer time than ACCU, it obtains a higher precision and outputs correct values faster.

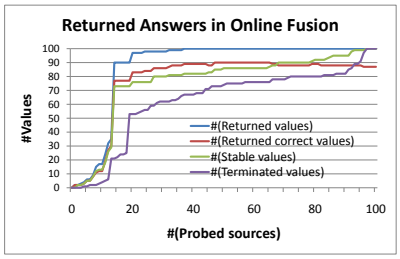


Figure 2: Observations of output values by PRAGMATIC.

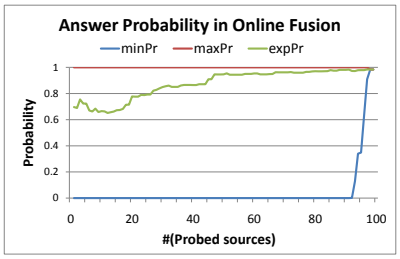


Figure 3: Observations of output probabilities by PRAGMATIC.

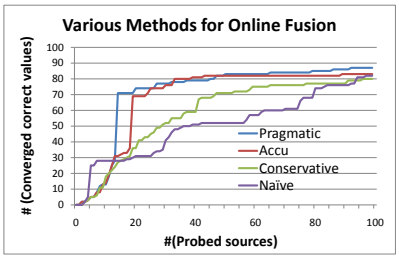


Figure 4: Stable correct values of different methods.

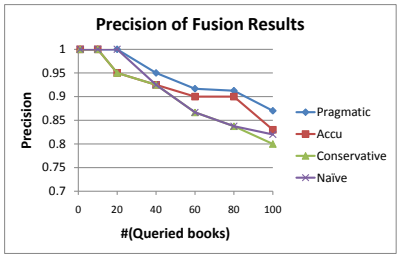


Figure 5: Precision of various methods.

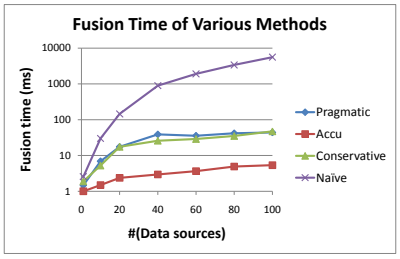


Figure 6: Fusion CPU time.

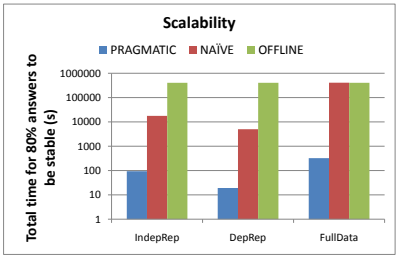


Figure 7: Method scalability.

Second, we started with the 100 sources and added more sources in three ways: I. *Independent replica* replicates each source 9 times and assumes independence between the replicas and the original sources; II. *Dependent replica* replicates each source 9 times but assumes each replica copies from the original source with probability .99; III. *Full AbeBooks data* adds the rest of the 794 sources in the AbeBooks data set in decreasing order of source coverage. We query the 100 books on data I and II and the 500 most popular books on data III. Figure 7 reports the total time (including connection and data transmission) for 80% of the answers to be stable (time for other percentages reported in Appendix D); here we tried 10 bookstore websites and used the mean of connection setup time (758 ms) and transmission time for one record (.3 ms). We compared PRAGMATIC, NAIVE, and offline fusion. (1) PRAGMATIC is the fastest and spent at most a few minutes on each data set; NAIVE is 2-3 orders of magnitude slower than PRAGMATIC and the offline method, which has to probe all sources before returning any answer, is 3-4 orders of magnitude slower. (2) Typically connection setup is the bottle-neck; the more sources required to converge, the longer the execution time. However, NAIVE also took a long CPU time for fusion on data III; although 80% values got stable after probing 382 sources, much fewer than for offline fusion (all 894 sources), it spent even longer time overall because the CPU time was high. (3) PRAGMATIC spent the longest time on data III, as most sources in this data set have very low coverage so convergence is slow; it spent the shortest time on data II, as the duplicates are considered as copiers and ranked later than the original sources, so most values get stable before probing the duplicates.

Finally, we observed quadratic growth of source-ordering time in the number of sources. PRAGMATICSOURCEORDERING took .2 second for 100 sources and 20.7 minutes for 894 sources; this is acceptable given that source ordering is a one-time offline process.

6. CONCLUSIONS

This paper describes SOLARIS, the first online data fusion system. It addresses several challenges in building such a system, including incrementally maintaining vote counts for each value, computing expected, maximum, and minimum probabilities of a value being true, deciding when to terminate fusion on particular data items, and ordering sources for early termination and early output of the correct answers. Future work includes combining our techniques with those that consider coverage of sources and overlap between sources for online fusion, and exploring other quality measures such as freshness of data.

7. REFERENCES

[1] L. Berti-Equille. *Quality Awareness for Managing and Mining Data*. PhD thesis, Universite de Rennes 1, 2007.

[2] L. Blanco, V. Crescenzi, P. Merialdo, and P. Papotti. Probabilistic models to reconcile complex data from inaccurate data sources. In *CAiSE*, 83–97, 2010.

[3] J. Bleiholder, S. Khuller, F. Naumann, L. Raschid, and Y. Wu. Query planning in the presence of overlapping sources. In *EDBT*, 811–828, 2006.

[4] M. J. Cafarella, A. Y. Halevy, and J. Madhavan. Structured data on the web. *Commun. ACM*, 54(2):72–79, 2011.

[5] X. Dong, L. Berti-Equille, Y. Hu, and D. Srivastava. Global detection of complex copying relationships between sources. *PVLDB*, 3(1):1358–1369, 2010.

[6] X. L. Dong, L. Berti-Equille, and D. Srivastava. Integrating conflicting data: The role of source dependence. *PVLDB*, 2(1):550–561, 2009.

[7] X. L. Dong and F. Naumann. Data fusion - resolving data conflicts for integration. *PVLDB*, 2(2):1654–1655, 2009.

[8] A. Galland, S. Abiteboul, A. Marian, and P. Senellart. Corroborating information from disagreeing views. In *WSDM*, 131–140, 2010.

[9] J. M. Hellerstein, P. J. Haas, and H. J. Wang. Online aggregation. In *SIGMOD*, 171–182, 1997.

[10] G. A. Mihaila, L. Raschid, and M.-E. Vidal. Using quality of data metadata for source selection and ranking. In *WebDB*, 93–98, 2000.

[11] F. Naumann. *Quality-Driven Query Answering for Integrated Information Systems*. Springer, 2002.

[12] A. D. Sarma, X. L. Dong, and A. Y. Halevy. Data integration with dependent sources. In *EDBT*, 401–412, 2011.

[13] M. A. Suryanto, E.-P. Lim, A. Sun, and R. H. L. Chiang. Quality-aware collaborative question answering: methods and evaluation. In *WSDM*, 142–151, 2009.

[14] M. Wu and A. Marian. A framework for corroborating answers from multiple web sources. *Inf. Syst.*, 36(2):431–449, 2011.

[15] N. K. Yeganeh, S. Sadiq, K. Deng, and X. Zhou. Data quality aware queries in collaborative information systems. *Lecture Notes in Computer Science*, 5446:39–50, 2009.

[16] X. Yin, J. Han, and P. S. Yu. Truth discovery with multiple conflicting information providers on the web. *IEEE Trans. Knowl. Data Eng.*, 20:796–808, 2008.

APPENDIX

A. PROOFS FOR SECTION 3

PROOF THEOREM 3.2 We compute the probability of each possible world according to the probabilities of the values being true, which are in turn computed based on observations on $\bar{S}$. Thus, we have

$$Pr(W) = \sum_{v \in \mathcal{D}(D)} Pr(W|v)Pr(v|\bar{S}) = Pr(W|\bar{S}).$$

Obviously, W and v are independent conditioned on $\bar{S}$.

According to the definition of expected probability, we have

$$\begin{aligned} expPr(v|\bar{S}) &= \sum_{W \in \mathbf{W}(\mathcal{S}-\bar{S})} Pr(W)Pr(v|\bar{S},W) \\ &= \sum_{W \in \mathbf{W}(\mathcal{S}-\bar{S})} Pr(W|\bar{S}) \cdot \frac{Pr(v,\bar{S},W)}{Pr(\bar{S},W)} \\ &= \sum_{W \in \mathbf{W}(\mathcal{S}-\bar{S})} Pr(W|\bar{S}) \cdot \frac{Pr(\bar{S})Pr(v|\bar{S})Pr(W|\bar{S})}{Pr(\bar{S})Pr(W|\bar{S})} \\ &= \sum_{W \in \mathbf{W}(\mathcal{S}-\bar{S})} Pr(W|\bar{S})Pr(v|\bar{S}) = Pr(v|\bar{S}) \quad \square \end{aligned}$$

PROOF THEOREM 3.3 Consider another possible world W_0 where some unseen sources do not provide v. We denote by $W_0(S)$ the value provided by S in W_0. We can transform W to W_0 step by step, where in each step for a source that does not provide v in W_0, we change its value from v to $W_0(S)$. Assuming the transformation steps are $W, W_n, \ldots, W_1, W_0$, we can prove easily

$$Pr(v|\bar{S},W) > Pr(v|\bar{S},W_n) > \cdots > Pr(v|\bar{S},W_1) > Pr(v|\bar{S},W_0). \square$$

PROOF THEOREM 3.4 Consider another possible world W_0 where some unseen sources do not provide v_{max}. We denote by $W_0(S)$ the value provided by S in W_0. We can transform W to W_0 step by step, where in each step for a value $v_0 \neq v_{max}$ and the set of unseen providers of v_0, denoted by $\bar{S}(v_0)$, we change their value from v_{max} to v_0. Assuming the transformation steps are $W, W_n, \ldots, W_1, W_0$, we can prove easily

$$Pr(v|\bar{S},W) < Pr(v|\bar{S},W_n) < \cdots < Pr(v|\bar{S},W_1) < Pr(v|\bar{S},W_0). \square$$

B. DETAILS FOR SECTION 4

Algorithms 2-6 give pseudo-code of key algorithms in Section 4. We next prove the theorems.

PROOF THEOREM 4.2 For each possible world W, we consider all its *sub-worlds*, each corresponding to a possible combination of the unseen copiers copy or do not copy. For each *sub-world*, we denote it by W' and only need to consider the sources that independently provide the values in probability computation. Then, for each W', it still holds that $Pr(v|W',\bar{S}) = Pr(v|\bar{S})$. Thus,

$$\begin{aligned} expPr(v|\bar{S}) &= \sum_{W \in \mathbf{W}(\mathcal{S}-\bar{S})} Pr(v|W,\bar{S})Pr(W) \\ &= \sum_{W \in \mathbf{W}(\mathcal{S}-\bar{S})} Pr(v|\bar{S})Pr(W) = Pr(v|\bar{S}). \square \end{aligned}$$

PROOF THEOREM 4.4 We consider single-source copying and can prove for multi-source copying similarly. Assume $S \in \bar{S}$ copies from $S_p \notin \bar{S}$. Consider probing S next. We denote by $Pr^{\rightarrow}(v|\bar{S},S)$ the probability of v based on the observation from $\bar{S}$ and S, where we use the dependent vote count of S no matter whether S provides the same value as S_p. We denote by $Pr^{con}(S,v)$ the probability that S provides v, $\omega = \alpha(S) - \frac{1-\alpha(S)}{n}$, and $\xi = \frac{1-\alpha(S)}{n}$.

$$\begin{aligned} Pr^{con}(S,v) &= \alpha(S)Pr^{con}(v|\bar{S}) + \frac{1-\alpha(S)}{n}(1 - Pr^{con}(v|\bar{S})) \\ &= \omega Pr^{con}(v|\bar{S}) + \xi. \end{aligned}$$

Algorithm 2: FUSIONWCOPY($\mathcal{S}$, $\bar{D}$)

Input : $\mathcal{S}$ ordered sources; $\bar{D}$ queried data items.
Output : True values for $\bar{D}$; for each returned value, return in addition $expPr(v)$, $minPr(v)$ and $maxPr(v)$.

```
1  stop[D ∈ D̄] ←false;
2  cVote[D ∈ D̄][v ∈ D(D)] ← 0;
3  sVote[D ∈ D̄][v ∈ D(D)] ← 0;
4  sV[S ∈ S][D ∈ D̄] ← C⊥(S);
5  remain ← Σ_{S∈S} α(S);
6  while ∃D ∈ D̄ s.t. stop[D] =false do
7      remain ← remain − α(S);
8      C̄ ← probed children of S;
9      P̄ ← probed parents of S;
10     Ā ← closest probed ancestors of S;
11     S_d ← closest probed descendant of S;// There is only one such
         descendant according to the co-copier condition.
12     cV[S][D] ← C⊥(S) · Π_{S_p is unseen parent of S}(1 − ρ(S → S_p));
13     foreach D ∈ D̄ do
14         if !stop[D] then
               // 1. Truth finding
15             ConservativeVoteCount(S, D, cV, cVote);
16             PragmaticVoteCount(S, D, sV, sVote);
17             find the value v_1 with the maximum vote count and v_2
                 with the top-2 vote count; // Either conservative or
                 pragmatic
               // 2. Probability computation
18             expPr(v_1) ←
                 Avg(e^{cV[D][v_1]} / Σ_{v∈D(D)} e^{cV[D][v]}, e^{sV[D][v_1]} / Σ_{v∈D(D)} e^{sV[D][v]});
                 Pr(v_2) ← e^{sV[D][v_2]} / Σ_{v∈D(D)} e^{sV[D][v]};
19             minPr(v_1) ← MinPr(v_1, d, S̄, S, sV, remain);
20             maxPr(v_1) ← MaxPr(v_1, d, S̄, S, sV, remain);
               // 3. Termination justification
21             if minPr(v_1) > expPr(v_2) then
22                 stop[D] ← true;
23     Refresh answers in the output;
```

According to Theorem 3.2,

$$Pr^{con}(v|\bar{S}) = \sum_{v_0 \in \mathcal{D}(D)} Pr^{con}(S,v_0)Pr^{\rightarrow}(v_0|\bar{S},S). \quad (5)$$

Similarly, we denote by $Pr^{\perp}(v|\bar{S},S)$ the probability of v based on the observation from $\bar{S}$ and S, where we use the independent vote count of S no matter whether S provides the same value as S_p. We denote by $Pr^{pra}(S,v)$ similarly and have

$$Pr^{pra}(v|\bar{S}) = \sum_{v_0 \in \mathcal{D}(D)} Pr^{pra}(S,v_0)Pr^{\perp}(v_0|\bar{S},S). \quad (6)$$

Obviously, for any $v_0 \in \mathcal{D}(D)$, $Pr^{\rightarrow}(v|\bar{S},S) < Pr^{\perp}(v|\bar{S},S)$. We denote the difference by $\Delta(v)$. In addition, $Pr^{con}(S,v) < Pr^{pra}(S,v)$ and $Pr^{con}(S,v') > Pr^{pra}(S,v')$ for any $v' \neq v$; we denote the difference by $\Delta'(v)$. Finally,

$$\begin{aligned} expPr^{con}(v|\bar{S}) &= Pr^{con}(S,v)Pr^{\rightarrow}(v|\bar{S},S) \\ &\quad + \sum_{v_0 \in \mathcal{D}(D), v_0 \neq v} Pr^{con}(S,v_0)Pr^{\perp}(v_0|\bar{S},S). \quad (7) \end{aligned}$$

(1) According to Eq.(5) and (7), $expPr^{con}(v|\bar{S}) > Pr^{con}(v|\bar{S})$. We now prove $expPr^{con}(v|\bar{S}) < Pr^{pra}(v|\bar{S})$. Without losing generality, assume v_0 has the highest vote count among values excluding v.

Algorithm 3: CONSERVATIVEVOTECOUNT($S, D, cV, cVote$)

$cV[S][D] \leftarrow cV[S][D] * \Pi_{S_p \in \bar{P}, S_p(D)=S(D)}$ (1-$\rho(S \to S_p)$); // $S(D)$ *is the value provided by* S *on* D.
$cVote[D][S(D)] \leftarrow cVote[D][S(D)] + cV[S][D]$;
foreach $S_c \in \bar{C}$ **do**
  if $S_c(D) \neq S(D)$ **then**
    $\Delta \leftarrow \frac{cV[S_c][D]\rho(S_d \to S)}{1-\rho(S_d \to S)}$;
    $cV[S_c][D] \leftarrow cV[S_c][D] + \Delta$;
    $cVote[D][S_c(D)] \leftarrow cVote[D][S_c(D)] + \Delta$;

Algorithm 4: PRAGMATICVOTECOUNT($S, D, sV, sVote$)

$sV[S][D] \leftarrow V_i(S)\Pi_{S_a \in \bar{A}, S_a(D)=S(D)}$ (1-$\rho(S \to Pa(S, S_a))$); // $Pa(S, S_a)$ *is the parent of* S *on the copying path to* S_a
$sVote[D][S(D)] \leftarrow sVote[S_d][S(D)] + sV[S][D]$;
if $\forall S_a \in \bar{A}, S_d(D) \neq S_a(D) \&\& S_d(D) = S(D)$ **then**
  $\Delta \leftarrow sV[S_d][D]\rho(S_d \to Pa(S_d, S))$;
  $sV[S_d][D] \leftarrow sV[S_d][D] - \Delta$;
  $sVote[D][S_d(D)] \leftarrow sVote[D][S_d(D)] - \Delta$;
else if $\exists S_a \in \bar{A}, S_d(D) = S_a(D) \&\& S_d(D) \neq S(D)$ **then**
  $\Delta \leftarrow \frac{sV[S_d][D]\rho(S_d \to Pa(S_d,S))}{1-\rho(S_d \to Pa(S_d,S))}$;
  $sV[S_d][D] \leftarrow sV[S_d][D] + \Delta$;
  $sVote[D][S_d(D)] \leftarrow sVote[D][S_d(D)] + \Delta$;

$$
\begin{aligned}
& Pr^{pra}(v|\bar{S}) - expPr^{con}(v|\bar{S}) \\
= \; & (Pr^{pra}(S,v)Pr^{\perp}(v|\bar{S},S) - Pr^{con}(S,v)Pr^{\to}(v|\bar{S},S)) \\
& + \sum_{v_0 \in \mathcal{D}(D), v_0 \neq v} Pr^{\perp}(v|\bar{S},S)(Pr^{pra}(S,v) - Pr^{con}(S,v)) \\
> \; & Pr^{\perp}(v|\bar{S},S)(\omega Pr^{pra}(v|\bar{S}) + \xi) \\
& - Pr^{\to}(v|\bar{S},S)(\omega Pr^{con}(v|\bar{S}) + \xi) - Pr^{\perp}(v_0|\bar{S},S)\omega\Delta(v) \\
= \; & \omega(Pr^{\perp}(v|\bar{S},S)Pr^{pra}(v|\bar{S}) - Pr^{\to}(v|\bar{S},S)Pr^{con}(v|\bar{S})) \\
& - \omega Pr^{\perp}(v_1|\bar{S},S)\Delta'(v) + \xi\Delta(v) \\
> \; & \omega(\Delta'(v) - Pr^{\perp}(v_1|\bar{S},S)\Delta'(v)) + \xi\Delta(v) > 0
\end{aligned}
$$

(2) We prove as follows.

$$
\begin{aligned}
& \frac{1}{2}\left(Pr^{con}(v|\bar{S}) + Pr^{pra}(v|\bar{S})\right) - expPr^{con}(v|\bar{S}) \\
= \; & \frac{1}{2}\left(Pr^{pra}(S,v)Pr^{\perp}(v|\bar{S},S) - Pr^{con}(S,v)Pr^{\to}(v|\bar{S},S)\right) \\
& + \frac{1}{2}\sum_{v_0 \in \mathcal{D}(D), v_0 \neq v} (Pr^{pra}(S,v_0)Pr^{\perp}(v_0|\bar{S},S) \\
& \quad + Pr^{con}(S,v)Pr^{\to}(v_0|\bar{S},S)) \\
& - \frac{1}{2}\sum_{v_0 \in \mathcal{D}(D), v_0 \neq v} (2Pr^{pra}(S,v)Pr^{\to}(v_0|\bar{S},S)) \\
< \; & \frac{1}{2}\left(Pr^{pra}(S,v)Pr^{\perp}(v|\bar{S},S) - Pr^{con}(S,v)Pr^{\to}(v|\bar{S},S)\right) \\
& + \frac{1}{2}\sum_{v_0 \in \mathcal{D}(D), v_0 \neq v} (Pr^{pra}(S,v_0)Pr^{\perp}(v_0|\bar{S},S) \\
& \quad - Pr^{pra}(S,v)Pr^{\to}(v_0|\bar{S},S)) \\
= \; & \frac{1}{2}(\omega(Pr^{\perp}(v|\bar{S},S)Pr^{pra}(v|\bar{S}) - Pr^{\to}(v|\bar{S},S)Pr^{con}(v|\bar{S})) \\
& \quad + \xi\Delta'(v)) \\
\leq \; & \frac{\omega\Delta'(v) + \xi\Delta'(v)}{2} \leq \frac{\alpha(S)}{2} \cdot \frac{1}{2} = \frac{1}{4} \quad \square
\end{aligned}
$$

Algorithm 5: MINPR($v, D, \bar{S}, \mathcal{S}, sV, remain$)

Input : v the value; D the data item;
  $\bar{S}$ probed sources; S all sources;
  sV pragmatic vote counts of each source,
  $remain$ sum of vote counts of the unseen sources.
Output : $minPr(v)$.
$vote[v \in \mathcal{D}(D)] \leftarrow 0$;
foreach $S \in \bar{S}$ **do**
  if *S's parents are all probed* **then**
    $vote[S(D)] \leftarrow vote[S(D)] + sV[S][D]$;
  else if $S(D) = v$ **then**
    $temp = \min_{S_a \text{ is an ancestor of } S} V_i(S_a) + sV[S][D](1 - \rho(S \to Pa(S, S_a)))$;
    if $temp > sV[S][D]$ **then**
      $vote[v] \leftarrow vote[v] + sV[S][D]$;
    else
      $vote[v] \leftarrow vote[v] + temp$;
  else
    $vote[S(D)] \leftarrow vote[S(D)] +$
    $\mathrm{C}^{\perp}(S)\Pi_{S_p(D)=S(D)}(1 - \rho(S \to S_p))$;
find the value $v_{max} \neq v$ with the maximum vote count;
$vote[v_{max}] \leftarrow vote[v_{max}] + remain$;
return $\frac{e^{cV[D][v]}}{\sum_{v_0 \in \mathcal{D}(D)} e^{cV[D][v_0]}}$;

Algorithm 6: PRAGMATICSOURCEORDERING($\mathcal{S}$)

Input : $\mathcal{S}$ set of sources.
Output : Ordering of the sources.
$\bar{S} \leftarrow \emptyset$;
$vote[S \in \mathcal{S}] \leftarrow C^{\perp}(S)$;
while $\bar{S} \neq \mathcal{S}$ **do**
  $max \leftarrow -\infty$; $S_0 \leftarrow$ null;
  foreach $S \in \mathcal{S}$ **do**
    if $vote[S] > max$ && S *satisfies the* co-copier condition **then**
      $max \leftarrow vote[S]$; $S_0 \leftarrow S$;
  Add S_0 to the end of $\bar{S}$;
  foreach $S \in \mathcal{S} - \bar{S}$ && S *is an ancestor or descendant of* S_0 **do**
    $Vote[S] \leftarrow C(\bar{S} \cup \{S\}) - C(\bar{S})$;
return $\bar{S}$;

PROOF THEOREM 4.5 Because the vote count of a source depends only on the values its direct parents provide, we can compute the minimum probability using a dynamic programming algorithm. The algorithm proceeds as follows.

1. Sort the sources such that for each S, S' where $\rho(S \to S') > 0$, S' is ranked earlier than S. Assume the order of the unseen sources is $S_0, S_1, \ldots, S_m$.
2. For the first source S_0 (it cannot be a copier of any other unseen source), compute its dependent vote count for each value $v_0 \in \mathcal{D}(D)$, and denote the corresponding probability for v_0 by $minPr(v|S, v_0)$.
3. For each latter source S_i, assume $\bar{P}$ is the set of its unseen parents. Then, for each $v_0 \in \mathcal{D}(D)$, for different value combination from $\bar{P}$, compute the probability of v based on the recorded vote counts from $\bar{P}$. Choose the minimum one, set $minPr(v|S_i, v_0)$ accordingly, and record the vote counts.
4. The minimum value for v is $\min_{v_0 \in \mathcal{D}(D)} minPr(v|S_m, v_0)$.

Let constant f_p be the maximum number of parents of a source. Then, the algorithm takes time $O(|\mathcal{S}||\mathcal{D}(D)|^{f_p})$. $\square$

PROOF THEOREM 4.6. For each provider S of value v, in case S is a copier, the lowest vote count for v is S's dependent vote

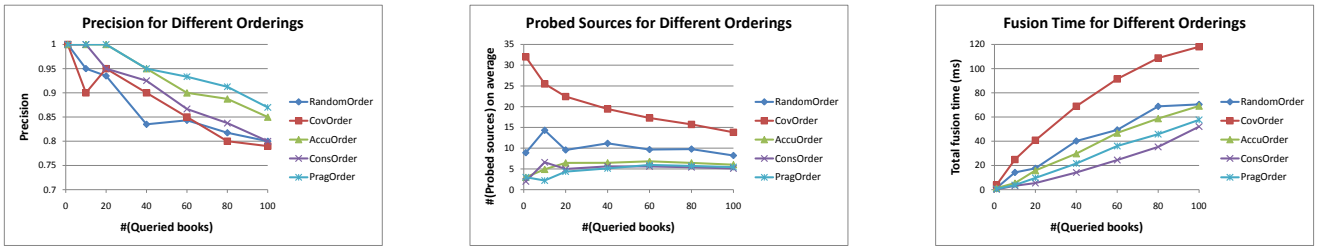


Figure 8: Comparison of different source ordering strategies.

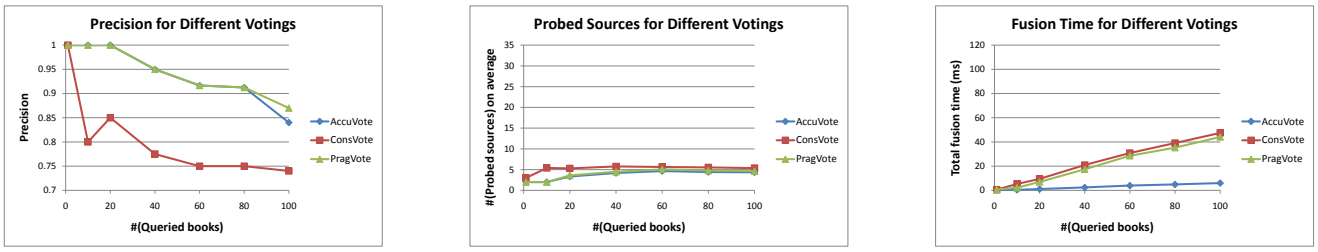


Figure 9: Comparison of different vote counting strategies.

count plus the minimum vote count of its unprobed ancestors. For each provider S' of another value, S''s independent (with respect to unprobed parents) vote count is the highest vote count S' can contribute. Finally, according to the proof of Theorem 3.4, we obtain the lowest vote count when unseen sources provide the value other than v with the highest vote count. Thus, Algorithm MINPR obtains the lowest probability for v.

For each source in $\mathcal{S}$, the algorithm takes linear time $O(f_a)$ to set vote count, where f_a is the maximum number of ancestors of a source. The complexity of the algorithm is $O(f_a|\mathcal{S}|)$. □

C. EXTENSIONS

We discuss two extensions of our techniques, one is in the direction of considering sources that do not have full coverage, and one is in the direction of considering more complex queries.

Coverage in online fusion: We assume the sources have fairly large but not necessarily full coverages, and extend our techniques in two ways. First, when we order the sources in the pragmatic approach, we take into consideration the overlap between sources. Consider source $S_0, \ldots, S_l$, where S_i copies from $S_{i+1}, i \in [0, l-1]$. Even when we have decided to probe S_l, it may not be appropriate to use the dependent vote count for S_0 in ordering, because S_0 and S_l can actually overlap in only a few data items. If we denote by $\iota(S_i, S_{i+1})$ the overlap between S_i and S_{i+1} (*i.e.*, $\iota(S_i, S_{i+1}) = \frac{|S_i \cap S_{i+1}|}{|S_{i+1}|}$) and assume independence of copying between each pair of sources, the probability that S_0 transitively copies from S_l on a particular item is $\rho(S_0 \to S_1)\Pi_{i=0}^{l-1}\iota(S_i, S_{i+1})$. In our experiments, we simplified and considered only direct copying (between a child and a parent) in source ordering.

Second, when we compute the minimum probability of a value (similar for maximum probability), we need to consider the possibility that an unseen source does not provide data on a particular data item at all. Thus, instead of using its independent vote count, we down-weight it by its coverage; in other words, we assume an unseen source S contributes vote count $C^{\perp}(S)\gamma(S)$ to value v_{max}, where $\gamma(S)$ denotes the coverage of S.

More complex queries: We next briefly discuss how we apply our techniques for queries that contain predicates on non-key attributes and queries that contain joins. For the former, the values on the predicates may be wrong as well and from a source we may miss some results or retrieve some additional results. As we probe new sources, we apply fusion techniques also on the predicate attribute of the returned tuples and decide if the value satisfies the predicate. If we decide the value actually does not satisfy the predicate, we remove it from the answer. For join queries, we assume the join-column values are accurate and apply fusion only on the projected attributes.

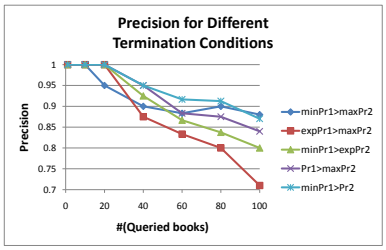
D. DETAILED EXPERIMENTAL RESULTS

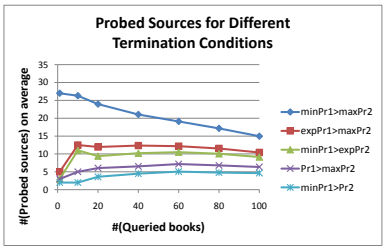
We compared variants of PRAGMATIC in terms of source ordering, vote counting, and termination condition. The experimental results show that PRAGMATIC with termination condition $minPr(v_1) > Pr(v_2)$ obtains the best results.

Comparing different orderings: We ordered the sources in 5 ways:

- RANDOMORDER probes the sources in a random order;
- COVORDER orders the sources by coverage;
- ACCUORDER orders the sources by accuracy;
- CONSORDER orders the sources in decreasing order of their (fixed) dependent vote count (the conservative approach);
- PRAGORDER orders the sources by applying algorithm PRAGMATICSOURCEORDERING.

We applied pragmatic vote counting for each ordering. Figure 8 shows the precision of the results, the average number of probed sources for each book, and the CPU time for fusion as we increase the number of queried books. We make the following observations. (1) PRAGORDER allows fast convergence and high precision of the results: it probes the least number of sources for each book (on average 4.5 sources) and obtains the highest precision. (2) ACCUORDER probes slightly more sources for each book than PRAGORDER (on average 5.7 sources), but obtains slightly lower precision. Although it starts with probing accurate sources, it ignores the copying relationship and can probe at an early time copiers whose vote counts are discounted. (3) CONSORDER probes similar number of sources for each book as PRAGORDER, but obtains a much lower precision. This is because it favors independent sources more than accurate sources, and can thus be biased by data from those independent but inaccurate sources. (4) COVORDER spent longest time (even 96% longer than RANDOMORDER on average) while obtaining nearly the lowest precision (the average is similar to that of RANDOMORDER). This is because in this data





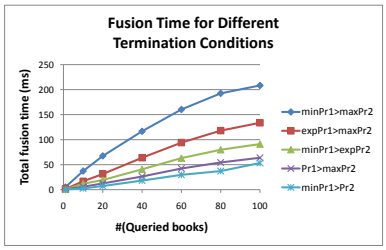


Figure 10: Comparison of different termination conditions.

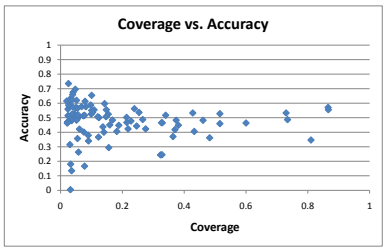


Figure 11: Coverage vs. accuracy.

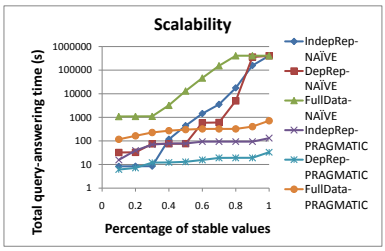


Figure 12: Query-answering time.

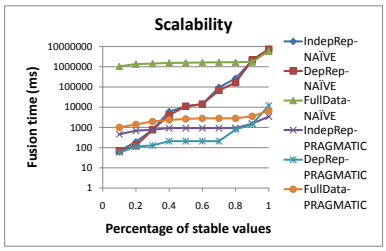


Figure 13: Fusion time.

set, we observe that many high-coverage sources have only moderate accuracy (see Figure 11). Finally, we note that as the number of books increases, (1) the average number of probed sources for a book can decrease, since the unpopular books have fewer providers; and (2) the fusion time increases, as different books are provided by different sets of sources and we need to probe more sources for convergence on all books.

Comparing different vote counting strategies: We next examine various vote counting strategies:

- ACCUVOTE computes the vote count as the sum of the independent vote counts of its providers;
- CONSVOTE applies the conservative voting approach;
- PRAGVOTE applies the pragmatic voting approach.

We applied the pragmatic order for each vote-counting strategy. Figure 9 shows the precision of the results, the average number of probed sources for each book, and the CPU time for fusion as we increase the number of queried books. We observe that PRAGVOTE obtains the highest precision. In comparison, ACCUVOTE probes similar number of sources on average, spent much less time as vote counting in ACCUVOTE is very simple, and obtains comparable precision. This is because although ACCUVOTE ignores copying in vote counting, it is less likely to make mistakes as the pragmatic order typically puts upfront only one source among the correlated sources. Even so, PRAGVOTE is more stable than ACCUVOTE; for example, its precision is 3.4% higher with 100 books. Its longer fusion time is acceptable because (1) query answering time is often dominated by data retrieval time, which is proportional to the number of probed sources, in which PRAGVOTE is similar to ACCUVOTE, and (2) our online fusion techniques actually return answers long before finishing fusion. Finally, CONSVOTE always has lower precision than PRAGVOTE (on average 15% lower) even though it probes slightly more sources, as it can often under-estimate vote counts when copiers are ranked earlier.

Comparing different termination conditions: Finally, we tried many different termination conditions, namely, (1) $minPr(v_1) > maxPr(v_2)$, (2) $expPr(v_1) > maxPr(v_2)$, (3) $minPr(v_1) > expPr(v_2)$, (4) $Pr(v_1) > maxPr(v_2)$, and (5) $minPr(v_1) > Pr(v_2)$ (the default) on PRAGMATIC, where v_1 is the top-1 value and v_2 is the top-2 value. Figure 10 shows the precision of the results, the average number of probed sources for each book, and the CPU time for fusion as we increase the number of queried books.

We have 3 observations. First, as expected, Condition (1) requires the longest time to converge as it is the strictest condition; however, it obtains a better result than Condition (5) only when there are 100 books (the precision is only 1% higher), as it can be biased by the additionally probed low-accuracy sources. Second, using the expected probability in the termination condition (Condition (2)(3)) can lead to long execution time and low precision. This is because the expected probability of the top-1 value is often lower than the probability computed in the pragmatic approach, so Condition (2) is harder to satisfy than Condition (4), and the expected probability of an unseen value is higher than the computed probability, so Condition (3) is harder to satisfy than Condition (5). Third, using the maximum probability in the termination condition (Condition (2)(4)) can lead to long execution time and low precision. This is because when there are a lot of unseen sources, the maximum probability of a value can easily reach as high as 1 and so the condition is hard to satisfy. Accordingly, Condition (5) leads to fastest termination. It also obtains the highest precision in most cases because it probes the least number of sources and so is least affected by low-accuracy data.

Details of experiments for scalability: Finally, we give more details for experiments on scalability. Figure 12-13 shows the query-answering time and the fusion time for a particular percentage of answers to be stable. In addition to the observations as we discussed for Figure 7, we also observe that (1) in general PRAGMATIC spent much less time than NAIVE both in query answering and in fusion; (2) it can take much longer time to get all stable answers than to get 80% stable answers (1.78 times longer for PRAGMATIC and 35.15 times longer for NAIVE); (3) typically the trends for query-answering time and for fusion time are comparable, except that first, the CPU time increased significantly for getting stable on the last 30% answers for PRAGMATIC on *Dependent replica* because of the complex copying relationships to handle in fusion, and the CPU time increased only slightly for NAIVE on *Full AbeBooks data* because most of the sources have very low coverage.