Integrating Conflicting Data: The Role of Source Dependence

Xin Luna Dong
AT&T Labs–Research
Florham Park, NJ, USA
lunadong@research.att.com

Laure Berti-Equille*
Université de Rennes 1
Rennes cedex, France
berti@irisa.fr

Divesh Srivastava
AT&T Labs–Research
Florham Park, NJ, USA
divesh@research.att.com

ABSTRACT

Many data management applications, such as setting up Web portals, managing enterprise data, managing community data, and sharing scientific data, require integrating data from multiple sources. Each of these sources provides a set of values and different sources can often provide conflicting values. To present quality data to users, it is critical that data integration systems can resolve conflicts and discover true values. Typically, we expect a true value to be provided by more sources than any particular false one, so we can take the value provided by the majority of the sources as the truth. Unfortunately, a false value can be spread through copying and that makes truth discovery extremely tricky. In this paper, we consider how to find true values from conflicting information when there are a large number of sources, among which some may copy from others.

We present a novel approach that considers *dependence* between data sources in truth discovery. Intuitively, if two data sources provide a large number of common values and many of these values are rarely provided by other sources (*e.g.*, particular false values), it is very likely that one copies from the other. We apply Bayesian analysis to decide dependence between sources and design an algorithm that iteratively detects dependence and discovers truth from conflicting information. We also extend our model by considering *accuracy* of data sources and *similarity* between values. Our experiments on synthetic data as well as real-world data show that our algorithm can significantly improve accuracy of truth discovery and is scalable when there are a large number of data sources.

1. INTRODUCTION

Many data management applications require integrating data from multiple sources, each of which provides a set of values as "facts". However, "facts and truth really don't have much to do with each other" (*by William Faulkner*). Different sources can often provide conflicting values, some being true while some being false. To provide quality data to users, it is critical that data integration systems can resolve conflicts and discover true values. Typically, we expect a true value to be provided by more sources than any particular false one, so we can apply voting and take the value provided by the majority of the sources as the truth. Unfortunately, copying between data sources is common in practice, especially on the web [1]; a value provided by one data source, no matter true or false, can be copied by many other sources. "A lie told often enough becomes the truth" (*by Vladimir Lenin*); telling the truth from conflicting information becomes extremely tricky in such a situation. In this paper, we consider the following problem: from the conflicting values provided by a large number of sources among which some may copy from others, how can one decide which is the true value?

We are mainly motivated by integrating data from the Web. In a variety of domains, such as science, business, politics, art, entertainment, sports, travel, there are a huge number of data sources that seek to provide information and a lot of the provided information overlaps. Whereas some of this information is dynamic, a large portion of the information is about some static aspect of the world, such as authors and publishers of books, directors, actors, and actresses of movies, revenue of a company in past years, presidents of a country in the past, and capitals of countries; the data sources rarely update such information. This paper focuses on such static information and considers a snapshot of data from different sources. Many data sources may copy and paste, crawl, or aggregate data from other sources, and publish the copied data without explicit attribution. In such applications, taking into consideration possible dependence between sources can often lead to more precise truth-discovery results.

Ideally, when applying voting, we would like to ignore copied information; however, this raises at least three challenges. First, in many applications we do not know how each source obtains its data, so we have to discover copiers from a snapshot of data. The discovery is non-trivial as sharing common data does not in itself imply copying. Second, even when we decide that two sources are dependent, with only a snapshot it is not obvious which one is a copier. Third, a copier can also provide some data by itself or verify some of the copied data, so it is inappropriate to ignore all data it provides.

In this paper, we present a novel approach that considers *dependence* between data sources in truth discovery. Our technique considers not only whether two sources share the same values, but also whether the shared values are true or false. Intuitively, for a particular object, there are often multiple distinct false values but usually only one true value. Sharing the same true value does not necessarily imply sources being dependent; however, sharing the same false value is typically a rare event when the sources are fully independent. Thus, if two data sources share a lot of false values, they are more likely to be dependent. We develop Bayes models that compute the probability of two data sources being dependent and take

*Visiting research program supported by the European Commission (Grant FP6-MOIF-CT-2006-041000)

VLDB '09, August 24-28, 2009, Lyon, France

Table 1: The motivating example: five data sources provide information on the affiliations of five researchers. Only S_1 provides all true values.

	S_1	S_2	S_3	S_4	S_5
$Stonebraker$	MIT	Berkeley	MIT	MIT	MS
$Dewitt$	MSR	MSR	UWisc	UWisc	UWisc
$Bernstein$	MSR	MSR	MSR	MSR	MSR
$Carey$	UCI	AT&T	BEA	BEA	BEA
$Halevy$	Google	Google	UW	UW	UW

the result into consideration in truth discovery. Note that detection of dependence between data sources is based on knowledge of true values, whereas correctly deciding true values requires knowledge of source dependence. There is an inter-dependence between them and we solve the problem by iteratively deciding source dependence and discovering truth from conflicting information. To the best of our knowledge, source-dependence analysis has not been investigated for the purpose of truth discovery.

We also consider *accuracy* in voting: we trust an accurate data source more and give values that it provides a higher weight. This method requires identifying not only if a pair of sources are dependent, but also which source is the copier. Indeed, accuracy in itself is a clue of direction of dependence: given two data sources, if the accuracy of their common data is highly different from that of one of the sources, that source is more likely to be a copier. Note that considering accuracy of sources in truth discovery has been explored in [16]. Whereas we share the basic idea, we present a different model for computing source accuracy and extend it to incorporate the notion of source dependence. We present more detailed comparison in Section 4.4.

We now illustrate our main techniques with an example.

EXAMPLE 1.1. *Consider the five data sources in Table 1. They provide information on affiliations of five researchers and only S_1 provides all correct data. However, since the affiliations provided by S_3 are copied by S_4 and S_5 (with certain errors during copying), a naive voting would consider them as the majority and so make wrong decisions for three researchers.*

We solve the problem by considering dependence between data sources. If we knew which values are true and which are false, we would suspect that S_3, S_4 and S_5 are dependent, because they provide the same false values. On the other hand, we would suspect the dependence between S_1 and S_2 much less, as they share only true values. Based on this analysis, we would ignore the values provided by S_4 and S_5 and then decide the correct affiliation for four researchers (except Carey).

Further, we consider accuracy of data sources. Based on the current voting results, S_1 is more accurate than S_2 and S_3. Thus, we would trust S_1 more and decide Carey's affiliation correctly. Note that if we do not consider dependence between S_3, S_4 and S_5, we would consider S_3 and S_4 as the most accurate and that further strengthens the wrong information they provide. □

In summary, our paper makes the following three contributions:

1. First, we formalize the notion of source dependence and present Bayes models to discover dependence from a snapshot of sources.
2. Second, we incorporate the notion of accuracy of sources in the analysis of source dependence, and design an algorithm that considers both dependence and accuracy in truth discovery. We further extend this algorithm by considering similarity between values and distribution of false values.
3. Finally, we tested our algorithms on synthetic data and real-world data sets. The experimental results show that our algorithm can significantly improve accuracy of truth discovery and is scalable when there are a large number of data sources.

We envision our work as a first step towards integrating data from multiple sources where some may copy from others. We expect broad impact on various aspects of data sharing and integration, including resolving conflicts from multiple and potentially dependent sources in data integration, generating reference data for the purpose of Master Data Management [12], detecting and preventing falsification by a group of malicious sources, recommending trustworthy data sources, efficiently answering queries from multiple sources with awareness of copiers, etc.

The rest of this paper is structured as follows. Section 2 formally defines the problem and the notion of dependence between data sources. Section 3 describes the core model that detects copiers and discovers truth accordingly. Section 4 describes an algorithm that considers both dependence and accuracy in truth discovery. Section 5 presents several extensions and Section 6 describes experimental results. Finally, Section 7 discusses related work and Section 8 concludes.

2. OVERVIEW

This section formally describes the problem we solve, defines dependence between data sources, and overviews models we present in this paper.

Problem statement We consider a set of *data sources* $\mathcal{S}$ and a set of *objects* $\mathcal{O}$. An object represents a particular aspect of a real-world entity, such as the director of a movie; in a relational database, an object corresponds to a cell in a table. For each object $O \in \mathcal{O}$, a source $S \in \mathcal{S}$ can (but not necessarily) provide a *value*. Among different values provided for an object, one correctly describes the real world and is *true*, and the rest are *false*. In this paper we solve the following problem: given a snapshot of data sources in $\mathcal{S}$, decide the true value for each object $O \in \mathcal{O}$.

We note that a value provided by a data source can either be atomic, or a set or list of atomic values (*e.g.*, author list of a book). In the latter case, we consider the value as true if the atomic values are correct and the set or list is complete (and order preserved for a list). This setting already fits many real-world applications and the solution is non-trivial.

Dependence between sources We say that there exists a *dependence* between two data sources S and T if they derive the same part of their data directly or transitively from a common source (can be one of S and T). Accordingly, there are two types of data sources: *independent sources* and *copiers*.

An *independent source* provides all values independently. It may provide some erroneous values because of incorrect knowledge of the real world, mis-spellings, etc. We thus further distinguish *good* independent sources from *bad* ones: a data source is considered to be good if for each object it is more likely to provide the true value than any *particular* false value; otherwise, it is considered to be bad.

A *copier* copies a part (or all) of data from other sources (independent sources or copiers). It can copy from multiple sources by union, intersection, etc., and as we consider only a snapshot of data, cyclic copying on a particular object is impossible. In addition, a copier may revise some of the copied values or add additional values; though, such revised and added values are considered as independent contributions by the copier.

To make our models tractable, we consider only *direct* copying in copying detection and truth discovery, where we say a source S

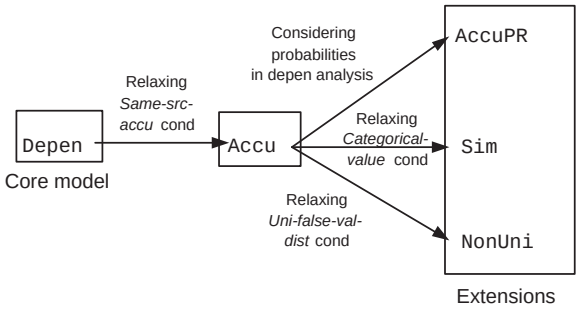

Figure 1: Models for truth discovery.

depends on T if S copies from T. However, as our experiments on synthetic data show, even in presence of transitive copying and co-copying from a hidden source, our algorithms still obtain high accuracy in truth discovery.

Models in this paper We start our discussion from a core case that satisfies the following three conditions, which we relax later:

- *Same source accuracy:* For each object, all independent data sources have the same probability of providing a true value.
- *Uniform false-value distribution:* For each object, there are multiple false values in the underlying domain and an independent source has the same probability of providing each of them.
- *Categorical value:* For each object, values that do not match exactly are considered as completely different.

In this core case, independent sources are *good* under the following condition. For each $O \in \mathcal{O}$, let $\varepsilon(O)$ be the probability that a source provides a false value (*i.e.*, error rate) on O and $n(O)$ be the number of false values on O in the underlying domain. Then, if $1-\varepsilon(O) > \frac{\varepsilon(O)}{n(O)}$ (*i.e.*, $\varepsilon(O) < \frac{n(O)}{n(O)+1}$), independent sources in $\mathcal{S}$ are good. Intuitively, given such a set of independent data sources, we can discover true values by voting. The following proposition, which we prove and generalize in Section 4, formalizes this intuition.

PROPOSITION 2.1 (VOTING). *Let O be an object and $\bar{S}_o$ be a set of independent sources voting for O. In the core case, if $\varepsilon(O) < \frac{n(O)}{n(O)+1}$, among the different values on O provided by $\bar{S}_o$, the one provided by the maximum number of sources has the highest probability to be true.* □

Even for this core case, discovering dependence between data sources and deciding true values are non-trivial; we solve the problem by the DEPEN model (Section 3). Then, we relax the *Same-source-accuracy* condition and present the ACCU model (Section 4). As extensions (Section 5), we present the SIM model relaxing the *Categorical-value* condition, the NONUNI model relaxing the *Uniform-false-value-distribution* condition, and the ACCUPR model considering probabilities of a value being true in dependence discovery (so in effect implicitly considering non-uniform distribution of false values as well). Figure 1 depicts the relationships between the models.

Assumptions To make the computation tractable, our models make the following assumptions.

- *Assumption 1 (Independent values).* The values that are independently provided by a data source on different objects are independent of each other.
- *Assumption 2 (Independent copying).* The dependence between a pair of data sources is independent of the dependence between any other pair of data sources.
- *Assumption 3 (No loop copying).* The dependence relationship between sources is acyclic.

Our experiments on synthetic data that violate the assumptions and on real-world data, which may violate the assumptions, show that performance of our models is not sensitive to these assumptions.

We note that the real world is complex: different sources may represent the same value in different ways, error rates on different data items can be different, errors of certain types may happen more often, copiers can have various copying behaviors, etc. Instead of modeling every possible variant, our models capture the most significant aspects of data providing and copying, so are tractable and can avoid overfitting. Indeed, as our experiments show, the ACCU model already obtains high accuracy on real-world data and synthetic data.

3. DETECTING SOURCE DEPENDENCE

This section describes how we detect copiers and discover truth from conflicting information accordingly.

3.1 Dependence of data sources

Assume $\mathcal{S}$ consists of two types of data sources: good independent sources and copiers. Consider two sources $S_1, S_2 \in \mathcal{S}$. We apply Bayes analysis to compute the probability that S_1 and S_2 are dependent given observation of their data. For this purpose, we need to compute the probability of the observed data, conditioned on the dependence or independence of the sources.

Our computation requires several parameters: n ($n > 1$), the number of false values in the underlying domain for each object; c ($0 < c \leq 1$), the probability that a value provided by a copier is copied; and ε ($0 \leq \varepsilon < \frac{n}{n+1}$), the *error rate*–probability that an independently provided value is false. Note that in practice, we may not know values of these parameters a-priori and the values may vary from object to object and from source to source. We bootstrap our algorithms by setting the parameters to default values initially and iteratively refining them by computing the estimated values according to the truth discovery and dependence detection results (details given at the end of this section).

In our observation, we are interested in three sets of objects: $\bar{O}_t$, denoting the set of objects on which S_1 and S_2 provide the same true value, $\bar{O}_f$, denoting the set of objects on which they provide the same false value, and $\bar{O}_d$, denoting the set of objects on which they provide different values ($\bar{O}_t \cup \bar{O}_f \cup \bar{O}_d \subseteq \mathcal{O}$). Intuitively, two independent sources providing the same false value is a rare event; thus, if we fix $\bar{O}_t \cup \bar{O}_f$ and $\bar{O}_d$, the more common false values that S_1 and S_2 provide, the more likely that they are dependent. On the other hand, if we fix $\bar{O}_t$ and $\bar{O}_f$, the fewer objects on which S_1 and S_2 provide different values, the more likely that they are dependent. We denote by Φ the observation of $\bar{O}_t, \bar{O}_f, \bar{O}_d$ and by k_t, k_f and k_d their sizes respectively. We next describe how we compute conditional probability of Φ based on these intuitions.

We first consider the case where S_1 and S_2 are independent, denoted by $S_1 \perp S_2$. Since there is a single true value, the probability that S_1 and S_2 provide the same true value for object O is

$$Pr(O \in \bar{O}_t | S_1 \perp S_2) = (1-\varepsilon)^2. \quad (1)$$

Under the *Uniform-false-value-distribution* condition, the probability that a data source provides a particular false value for object O is $\frac{\varepsilon}{n}$. Thus, the probability that S_1 and S_2 provide the same false value for O is

$$Pr(O \in \bar{O}_f | S_1 \perp S_2) = n \cdot (\frac{\varepsilon}{n})^2 = \frac{\varepsilon^2}{n}. \quad (2)$$

Then, the probability that S_1 and S_2 provide different values on an object O, denoted by P_d for convenience, is

$$Pr(O \in \bar{O}_d | S_1 \perp S_2) = 1 - (1-\varepsilon)^2 - \frac{\varepsilon^2}{n} = P_d. \quad (3)$$

Following the *Independent-values* assumption, the conditional probability of observing Φ is

$$Pr(\Phi | S_1 \perp S_2) = \frac{(1-\varepsilon)^{2k_t} \varepsilon^{2k_f} P_d^{k_d}}{n^{k_f}}. \quad (4)$$

We next consider the case when S_1 and S_2 are dependent, denoted by $S_1 \sim S_2$. There are two cases where S_1 and S_2 provide the same value v for an object O. First, with probability c, one copies v from the other and so v is true with probability $1-\varepsilon$ and false with probability ε. Second, with probability $1-c$, the two sources provide v independently and so its probability of being true or false is the same as in the case where S_1 and S_2 are independent. Thus, we have

$$Pr(O \in \bar{O}_t | S_1 \sim S_2) = (1-\varepsilon) \cdot c + (1-\varepsilon)^2 \cdot (1-c), \quad (5)$$

$$Pr(O \in \bar{O}_f | S_1 \sim S_2) = \varepsilon \cdot c + \frac{\varepsilon^2}{n} \cdot (1-c). \quad (6)$$

Finally, the probability that S_1 and S_2 provide different values on an object is that of S_1 providing a value independently and the value differs from that provided by S_2:

$$Pr(O \in \bar{O}_d | S_1 \sim S_2) = P_d \cdot (1-c). \quad (7)$$

We compute $Pr(\Phi | S_1 \sim S_2)$ accordingly. Now we can compute the probability of $S_1 \sim S_2$ by applying the Bayes Rule.

$$\begin{aligned} & Pr(S_1 \sim S_2 | \Phi) \\ = & \frac{Pr(\Phi | S_1 \sim S_2) Pr(S_1 \sim S_2)}{Pr(\Phi | S_1 \sim S_2) Pr(S_1 \sim S_2) + Pr(\Phi | S_1 \perp S_2) Pr(S_1 \perp S_2)} \\ = & \left(1 + (\frac{1-\alpha}{\alpha})(\frac{1-\varepsilon}{1-\varepsilon+c\varepsilon})^{k_t} (\frac{\varepsilon}{cn+\varepsilon-c\varepsilon})^{k_f} (\frac{1}{1-c})^{k_d}\right)^{-1} \quad (8) \end{aligned}$$

Here $\alpha = Pr(S_1 \sim S_2) (0 < \alpha < 1)$ is the a-priori probability that two data sources are dependent.

Eq.(8) has several nice properties that conform to the intuitions we discussed early in this section, formalized as follows. (We provide proofs in [7].)

THEOREM 3.1. *Let $\mathcal{S}$ be a set of good independent sources and copiers. Eq.(8) has the following three properties on $\mathcal{S}$.*

1. *Fixing $k_t + k_f$ and k_d, when k_f increases, the probability of dependence increases;*
2. *Fixing $k_t + k_f + k_d$, when $k_t + k_f$ increases and none of k_t and k_f decreases, the probability of dependence increases;*
3. *Fixing k_t and k_f, when k_d decreases, the probability of dependence increases.* □

Note that Eq.(8) does not indicate direction of dependence; Section 4 describes a model that detects directions of dependencies for sources of different accuracy.

Also note that in case of co-copying and transitive copying, Eq.(8) can still compute a high probability of dependence, as a lot of wrong values can be shared in such cases as well.

3.2 Vote count of a value

We have described how we decide if a pair of sources are dependent. However, even if a source copies from another, it is possible that it provides some of the values independently and it would be inappropriate to ignore these values. We next describe how to count the vote for a particular value. We start with ideal vote count and then describe an approximation.

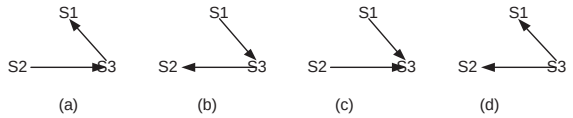

Figure 2: Dependence graphs with a dependence between S_1 and S_3 and one between S_2 and S_3, where $S_1, S_2,$ and S_3 provide the same value on an object.

3.2.1 Ideal vote count

We start from the case where we know deterministically the dependence relationship between sources and discuss probabilistic dependence subsequently. Consider a specific value v for a particular object O and let $\bar{S}_o(v)$ be the set of data sources that provide v on O. We can draw a *dependence graph* G, where for each $S \in \bar{S}_o(v)$, there is a node, and for each $S_1, S_2 \in \bar{S}_o(v)$ where S_1 copies from S_2, there is an edge from S_1 to S_2.

For each $S \in \bar{S}_o(v)$, we denote by $d(S, G)$ the out-degree of S in G, corresponding to the number of data sources from which S copies. If $d(S, G) = 0$, S is independent and its vote count for v is 1. Otherwise, for each source S' that S copies from, S provides a value independently of S' with probability $1-c$. According to the *Independent-copying* assumption, the probability that S provides v independently of any other source is $(1-c)^{d(S,G)}$ and the total vote count of v with respect to G is

$$V(v, G) = \sum_{S \in \bar{S}_o(v)} (1-c)^{d(S,G)}. \quad (9)$$

However, recall that Eq.(8) computes only a probability of dependence and does not indicate its direction. Thus, we have to enumerate all possible dependence graphs and take the sum of the vote count with respect to each of them, weighted by the probability of the graph. Let $\bar{D}_o$ be the set of possible dependencies between sources in $\bar{S}_o(v)$ and we denote the probability of $D \in \bar{D}_o$ by $p(D)$. Consider a subset $\bar{D} \subseteq \bar{D}_o$ of m dependencies. According to the *Independent-copying* assumption, the probability that all and only dependencies in $\bar{D}$ hold is

$$Pr(\bar{D}) = \Pi_{D \in \bar{D}} p(D) \Pi_{D \in \bar{D}_o - \bar{D}} (1 - p(D)). \quad (10)$$

As each dependence can have one of the two directions, there are up to 2^m acyclic dependence graphs with this set of dependencies. Intuitively, the more independent sources in a graph, the less likely that all sources in the graph provide the same value. By applying Bayesian analysis, we can compute the probability of each graph. We skip the equations for space consideration and illustrate the computation of vote count in the following example.

EXAMPLE 3.2. *Consider three data sources S_1, S_2 and S_3 that provide the same value v on an object. Assume $c = .8$ and between each pair of sources the probability of dependence is .4. We can compute v's vote count by enumerating all possible dependence graphs.*

- *There is 1 graph with no dependence. All sources are independent so the vote count is $1 + 1 + 1 = 3$. The probability of this graph is $(1 - .4)^3 = .216$.*
- *There are 6 graphs with only one dependence. The total probability of graphs that contain a particular dependence is $(1 - .4)^2 * .4 = .144$. Each dependence has two directions, so the probability of each such graph is $.144/2 = .072$. No matter which direction the dependence is in, the vote count is $1 + 1 + .2 = 2.2$.*
- *There are 12 graphs with two dependencies. Figure 2 shows the four that contain a dependence between S_1 and S_3, and a dependence between S_2 and S_3. The sum of their probabilities is $(1-.4) * .4^2 = .096$. For each of the first three graphs*

(Figure 2(a)-(c), each with a single independent source), the vote count is $1 + .2 + .2 = 1.4$ *and by applying the Bayes Rule we compute its probability as* $.32 * .096 = .03$*. For the last one (Figure 2(d), with two independent sources), the vote count is* $1 + 1 + .2^2 = 2.04$ *and its probability is* $.04 * .096 = .004$*.*

- *Finally, there are 6 acyclic graphs with three dependencies (details ignored to save space), where each has vote count* $1 + .2 + .2^2 = 1.24$ *and probability* $.4^3/6 = .011$*.*

The total vote count of v*, computed as the weighted sum, is 2.08.* □

3.2.2 Estimating vote count

As there are an exponential number of dependence graphs, computing the vote count by enumerating all of them can be quite expensive. To make the analysis scalable, we shall find a way to estimate the vote count in polynomial time.

We estimate a vote count by considering the data sources one by one. For each source S, we denote by $\overline{Pre}(S)$ the set of sources that have already been considered and by $\overline{Post}(S)$ the set of sources that have not been considered yet. We compute the probability that the value provided by S is independent of any source in $\overline{Pre}(S)$ and take it as the vote count of S. The vote count computed in this way is not precise because if S depends only on sources in $\overline{Post}(S)$ but some of those sources depend on sources in $\overline{Pre}(S)$, our estimation still (incorrectly) counts S's vote. To minimize such error, we wish that the probability that S depends on a source $S' \in \overline{Post}(S)$ and S' depends on a source $S'' \in \overline{Pre}(S)$ be the lowest. Thus, we take a greedy algorithm and consider data sources in such an order: in the first round, we select a data source that is associated with a dependence of the highest probability; in later rounds, each time we select a data source that has the maximal dependence on one of the previously selected sources.

We now consider how to compute the vote count of v once we have decided an order of the data sources. Let S be a data source that votes for v and we denote by $P(S \sim S_0)$ the probability of dependence between sources S and S_0. The probability that S provides v independently of any data source in $\overline{Pre}(S)$, denoted by $I(S)$, is

$$I(S) = \Pi_{S_0 \in \overline{Pre}(S)}(1 - cP(S \sim S_0)). \quad (11)$$

The total vote count of v is $\sum_{S \in \bar{S}_o(v)} I(S)$.

EXAMPLE 3.3. *Continue with Example 3.2. As all dependencies have the same probability, we can consider the data sources in any order. We choose the order of* S_1, S_2, S_3*. The vote count of* S_1 *is 1, that of* S_2 *is* $1 - .4 * .8 = .68$*, and that of* S_3 *is* $.68^2 = .46$*. So the estimated vote count is* $1 + .68 + .46 = 2.14$*, very close to the real one, 2.08.* □

We formalize properties of the vote-count estimation as follows, showing scalability of our estimation algorithm.

THEOREM 3.4. *Our vote-count estimation has the following two properties.*

1. *Let* t_0 *be the ideal vote count of a value and* t *be the estimated vote count. Then,* $t_0 \leq t \leq 1.5t_0$*.*
2. *Let* s *be the number of sources that provide information on an object. We can estimate the vote count of all values of this object in time* $O(s^2 \log s)$*.* □

0: **Input**: $\mathcal{S}, \mathcal{O}$.
Output: The true value for each object in $\mathcal{O}$.
1: $\bar{V} = \emptyset$; *//decided true values*
$\bar{V}_0 = null$; *//true values decided in the last round*
2: **while** ($\bar{V} \neq \bar{V}_0$)
3: $\bar{V}_0 = \bar{V}; \bar{V} = \emptyset$;
4: **for each** ($S_1, S_2 \in \mathcal{S}, S_1 \neq S_2$)
Compute probability of dependence between S_1 and S_2;
5: **for each** ($O \in \mathcal{O}$)
Compute vote count of each value of O;
Select the value with the maximal vote count and add to $\bar{V}$(if there are several values winning the same number of votes, choose the previously selected one if possible and randomly choose one otherwise);

Algorithm 1: VOTE: Discover true values.

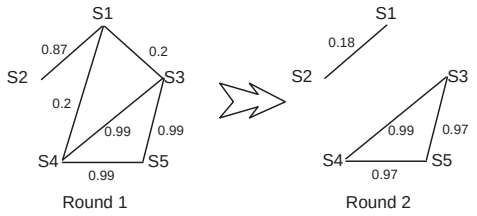

Figure 3: Probabilities of dependencies computed by DEPEN **on the motivating example. We only show dependencies with a probability over .1.**

3.3 Finding the true values

Once we know the vote count of each value, we can decide the true value by voting. However, computing vote counts requires knowing probabilities of dependencies between data sources, whereas computing the probabilities of dependencies requires knowing the true values. There is an inter-dependence between them and we solve the problem by computing them iteratively.

Algorithm VOTE describes how to discover true values from conflicting information provided by multiple data sources. VOTE iteratively computes the probability of dependence between each pair of data sources and the vote count of each value, and then for each object takes the value with the maximal vote count as the true value. This process repeats until the voting results converge.

Note that it is critical to consider the dependence between sources from the beginning; otherwise, a data source that has been duplicated many times can dominate the vote results in the first round and make it hard to detect the dependence between it and its copiers (as they share only "true" values). Our initial decision on dependence is similar to Eq.(8) except considering both possibilities of a value being true and being false and we skip details here.

We can prove that when there are a finite number of objects in $\mathcal{O}$, Algorithm VOTE cannot change the decision for an object O back and forth between two different values forever; thus, the algorithm converges. In practice, our experiments show that the algorithm typically converges in only a few rounds.

THEOREM 3.5 (CONVERGENCE OF VOTE). *Let* $\mathcal{S}$ *be a set of good independent sources and copiers that provide information on objects in* $\mathcal{O}$*. Let* l *be the number of objects in* $\mathcal{O}$ *and* n_0 *be the maximum number of values provided for an object by* $\mathcal{S}$*. The* VOTE *algorithm converges in at most* $2ln_0$ *rounds on* $\mathcal{S}$ *and* $\mathcal{O}$*.* □

We illustrate the algorithm on the motivating example.

EXAMPLE 3.6. *We run Algorithm* VOTE *on data sources in Example 1.1. Figure 3 shows the probabilities of dependencies com-*

Table 2: Vote counts of affiliations for *Carey* and *Halevy* in the motivating example.

	Carey			*Halevy*	
	UCI	AT&T	BEA	Google	UW
Round 1	1	1	1.24	1.3	1.24
Round 2	1	1	1.25	1.85	1.25

puted in each round and Table 2 shows the vote count of affiliations for Carey *and* Halevy.

Initially, we compute the probability of dependence between S_1 and S_2 (sharing three values) as .87 and those between S_3, S_4, S_5 (sharing four or five values) as .99. Accordingly we decide that the affiliations are MIT, MSR, MSR, BEA, Google respectively.

In the second round, we refine the dependence probabilities according to the selected true values. The probability between S_1 and S_2 (sharing only true values) is reduced to .18 and those between S_3, S_4, S_5 (sharing two or three false values) remain high; thus, the refined probabilities more closely reflect the reality. The new probabilities do not further change our voting results. The voting converges and finds four correct affiliations. □

Setting parameters: We set parameters in our model initially according to our a-priori knowledge of the data or by guessing a default value. During the voting process, in each iteration we can refine α, ε and c based on the computed dependence probabilities and the decided true values, and use the new parameters in the next iteration. Our experimental results show that different initial parameter settings lead to similar voting results (Section 6), providing evidence of robustness.

4. CONSIDERING SOURCE ACCURACY

This section describes the ACCU model, which considers *accuracy* of data sources. We first discuss how the accuracy of sources can affect our belief of dependence between sources, and then describe how we compute accuracy and take it into consideration when we count votes.

Our ACCU model indeed computes a probabilistic distribution of various values in the underlying domain for a particular object. We can either choose the value with the highest probability as the true value, or store all possible values with their probabilities using a probabilistic database.

4.1 Dependence w.r.t. accuracy of sources

In this section, we consider different directions of dependence, denoting S_1 depending on S_2 by $S_1 \to S_2$ and S_2 depending on S_1 by $S_2 \to S_1$. Intuitively, if between two data sources S_1 and S_2, the accuracy of the common values is closer to the overall accuracy of S_1, then it is more likely that S_2 copies from S_1. We incorporate this intuition by considering accuracy of sources when we compute the probability of dependencies.

Let S be a data source. We denote by $A(S)$ the *accuracy* of S and by $\varepsilon(S)$ the *error rate* of S; $\varepsilon(S) = 1 - A(S)$. We describe how to compute $A(S)$ shortly. A similar analysis as in Section 3 leads to the following sets of equations. When S_1 and S_2 are independent, we have

$$Pr(O \in \bar{O}_t | S_1 \perp S_2) = (1-\varepsilon(S_1))(1-\varepsilon(S_2)) = P_t, \quad (12)$$

$$Pr(O \in \bar{O}_f | S_1 \perp S_2) = \frac{\varepsilon(S_1)\varepsilon(S_2)}{n} = P_f, \quad (13)$$

$$Pr(O \in \bar{O}_d | S_1 \perp S_2) = 1 - P_t - P_f. \quad (14)$$

When S_2 copies from S_1 (similar for S_1 copying from S_2), we have

$$Pr(O \in \bar{O}_t | S_2 \to S_1) = (1-\varepsilon(S_1)) \cdot c + P_t \cdot (1-c), \quad (15)$$

$$Pr(O \in \bar{O}_f | S_2 \to S_1) = \varepsilon(S_1) \cdot c + P_f \cdot (1-c), \quad (16)$$

$$Pr(O \in \bar{O}_d | S_2 \to S_1) = (1 - P_t - P_f) \cdot (1-c). \quad (17)$$

Note that the probability of S_1 and S_2 providing the same true or false value is different with different directions of dependence. By applying the Bayes Rule, we can compute the probabilities of $S_1 \perp S_2$, $S_1 \to S_2$ and $S_2 \to S_1$. If we consider $Pr(S_1 \to S_2) + Pr(S_2 \to S_1)$ as the probability of dependence between S_1 and S_2, the three properties in Theorem 3.1 still hold.

4.2 Accuracy of a data source

We next consider how one can compute the accuracy of a data source. A naive way is to compute the fraction of true values provided by the source. However, we do not know for sure which are the true values, especially among values that are voted by similar number of sources. We instead compute the accuracy of a source as the average probability of its values being true.

Formally, let $\bar{V}(S)$ be the values provided by S and let m be the size of $\bar{V}(S)$. For each $v \in \bar{V}(S)$, we denote by $P(v)$ the probability that v is true. We compute $A(S)$ as follows.

$$A(S) = \frac{\Sigma_{v \in \bar{V}(S)} P(v)}{m}. \quad (18)$$

Now we need a way to compute the probability that a value is true. Intuitively, the computation should consider both how many sources provide the value and accuracy of those sources. We apply a Bayes analysis again.

We start with the case where all data sources are independent. Consider an object $O \in \mathcal{O}$. Let $\mathcal{V}(O)$ be the domain of O, including one true value and n false values. Let $\bar{S}_o$ be the sources that provide information on O. For each $v \in \mathcal{V}(O)$, we denote by $\bar{S}_o(v) \subseteq \bar{S}_o$ the set of sources that vote for v ($\bar{S}_o(v)$ can be empty). We denote by $\Psi(O)$ the observation of which value each $S \in \bar{S}_o$ votes for.

To compute $P(v)$ for $v \in \mathcal{V}(O)$, we need to first be able to compute the probability of $\Psi(O)$ conditioned on v being true. This probability should be that of sources in $\bar{S}_o(v)$ each providing the true value and other sources each providing a particular false value.

$$Pr(\Psi(O) | v \text{ true}) = \Pi_{S \in \bar{S}_o(v)} A(S) \cdot \Pi_{S \in \bar{S}_o - \bar{S}_o(v)} \frac{1-A(S)}{n} \quad (19)$$

Among the values in $\mathcal{V}(O)$, there is one and only one true value. Assume our a-priori belief of each value being true is the same, denoted by β. We then have

$$Pr(\Psi(O)) = \sum_{v \in \mathcal{V}(O)} \left(\beta \cdot \Pi_{S \in \bar{S}_o(v)} A(S) \cdot \Pi_{S \in \bar{S}_o - \bar{S}_o(v)} \frac{1-A(S)}{n} \right). \quad (20)$$

Applying the Bayes Rule leads us to

$$P(v) = Pr(v \text{ true} | \Psi(O)) = \frac{\Pi_{S \in \bar{S}_o(v)} \frac{nA(S)}{1-A(S)}}{\sum_{v_0 \in \mathcal{V}(O)} \Pi_{S \in \bar{S}_o(v_0)} \frac{nA(S)}{1-A(S)}}. \quad (21)$$

To simplify the computation, we define the *confidence* of v[1], denoted by $C(v)$, as

$$C(v) = \sum_{S \in \bar{S}_o(v)} \ln \frac{nA(S)}{1-A(S)}. \quad (22)$$

[1]Note that the confidence of a value is derived from, but not equivalent to, the probability of the value.

If we define the *accuracy score* of a data source S as

$$A'(S) = \ln \frac{nA(S)}{1 - A(S)} \tag{23}$$

we have

$$C(v) = \sum_{S \in \bar{S}_o(v)} A'(S). \tag{24}$$

So we can compute the confidence of a value by summing up the accuracy scores of its providers. Finally, we can compute $P(v) = \frac{e^{C(v)}}{\omega}$, where, $\omega = \sum_{v_0 \in \mathcal{D}(O)} e^{C(v_0)}$, and compute accuracy of each source accordingly.

A value with a higher confidence has a higher probability to be true; thus, rather than comparing vote counts, we compare confidence of values. The following theorem shows three properties of Eq.(24).

THEOREM 4.1. *Eq.(24) has the following properties:*

1. *If all data sources have the same accuracy, when the size of $\bar{S}_o(v)$ increases, $C(v)$ increases;*
2. *Fixing all sources in $\bar{S}_o(v)$ except S, when $A(S)$ increases for S, $C(v)$ increases.*
3. *If there exists $S \in \bar{S}_o(v)$ such that $A(S) = 1$ and no $S' \in \bar{S}_o(v)$ such that $A(S') = 0$, $C(v) = +\infty$; if there exists $S \in \bar{S}_o(v)$ such that $A(S) = 0$ and no $S' \in \bar{S}_o(v)$ such that $A(S') = 1$, $C(v) = -\infty$.* □

Note that the first property is actually a justification of the voting strategy (Proposition 2.1). The third property shows that we should be careful not to assign very high or very low accuracy to a data source, which has been avoided by defining the accuracy of a source as the average probability of its provided values.

Finally, if a data source S copies a value v from other sources, we should ignore S when computing the confidence of v. Following the same analysis, we have

$$C(v) = \sum_{S \in \bar{S}_o(v)} A'(S) I(S). \tag{25}$$

In the equation, $I(S)$ is computed by Eq.(11), except that we sort the sources differently: if the probability of $S_1 \to S_2$ is much higher than that of $S_2 \to S_1$, we consider S_1 as a copier and order S_2 before S_1; otherwise, we consider both directions as equally possible and sort the sources following the same rule as for the basic model.

4.3 Combining accuracy and dependence

We now extend the VOTE algorithm to incorporate analysis of accuracy. We need to compute three measures: accuracy of sources, dependence between sources, and confidence of values. Accuracy of a source depends on confidence of values; dependence between sources depends on accuracy of sources and the true values selected according to the confidence of values; and confidence of values depends on both accuracy of and dependence between data sources.

We conduct analysis of both accuracy and dependence in each round. Specifically, Algorithm ACCUVOTE starts by setting the probability of each value as one minus the overall error rate, iteratively (1) computes accuracy and dependence based on the confidence of values computed in the previous round, and (2) updates confidence of values accordingly, and stops when the accuracy of the sources becomes stable. Note that ACCUVOTE may not converge: when we select different values as the true values, the direction of the dependence between two sources can change and in turn suggest different true values. We stop the process after we detect oscillation of decided true values. Our experiments show that when the number of objects is much more than the number of sources, our algorithm typically converges soon; though, the precise condition for convergence remains an open problem. Finally, we note that the complexity of each round is $O(|\mathcal{O}||\mathcal{S}|^2 \log |\mathcal{S}|)$.

0: **Input**: $\mathcal{S}, \mathcal{O}$.
Output: The true value for each object in $\mathcal{O}$.
1: Set the accuracy of each source as $1 - \epsilon$;
2: **while** (accuracy of sources changes && no oscillation of decided true values)
3: Compute probability of dependence between each pair of sources;
4: Sort sources according to the dependencies;
5: Compute confidence of each value for each object;
6: Compute accuracy of each source;
7: **for each** ($O \in \mathcal{O}$)
Among all values of O, select the one with the highest confidence as the true value;

Algorithm 2: ACCUVOTE: Discover true values by considering accuracy and dependence of data sources.

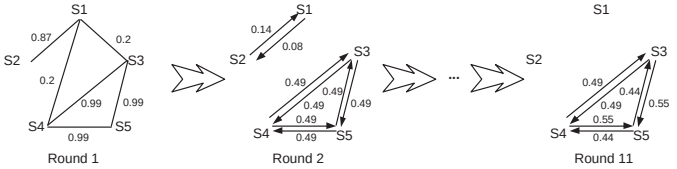

Figure 4: Probabilities of dependencies computed by ACCU on the motivating example. We only show dependencies where the sum of the probabilities on both directions is over .1.

EXAMPLE 4.2. *Continue with the motivating example. Figure 4 shows the probability of dependence, Table 3 shows the computed accuracy of each data source, and Table 4 shows the confidence of affiliations computed for* Carey *and* Halevy.

Initially, Ln.1 of Algorithm ACCUVOTE *sets the accuracy of each source to .8. Accordingly, Ln.3 computes the probability of dependence between sources as shown on the left of Figure 4. Taking the dependence into consideration, Ln.5 computes confidence of the values; for example, for* Carey *it computes 1.61 as the confidence of value* UCI *and* AT&T*, and 2.0 as the confidence of value* BEA*. Then, Ln.6 updates the accuracy of each source to .52, .42, .53, .53, .53 respectively according to the computed value confidence; the updated accuracy is used in the next round.*

Starting from the 2nd round, S_1 is considered more accurate and its values are given higher weight. In later rounds, ACCU *gradually increases the accuracy of S_1 and decreases that of S_3, S_4 and S_5. At the 4th round,* ACCU *decides that* UCI *is the correct affiliation for* Carey *and finds the right affiliations for all researchers. Finally,* ACCU *terminates at the 11th round and the source accuracy it computes converges close to the expected ones.* □

4.4 Comparison with TRUTHFINDER

Yin et al. [16] proposed TRUTHFINDER, which considers accuracy of sources in truth discovery. Whereas we both consider accuracy of sources, our model differs from theirs.

The most important difference is that we consider the dependence between sources. TRUTHFINDER uses a *dampening factor* to address the possible dependence between sources; however, this approach is not necessarily effective and for our motivating example, TRUTHFINDER incorrectly decides that all values provided by S_3 are true. Our model considers dependence in a principled fashion. By examining the probability that a pair of data sources are dependent and its effect on voting, we are able to apply dampening

Table 3: Accuracy of data sources computed by ACCU on the motivating example.

	S_1	S_2	S_3	S_4	S_5
Round 1	.52	.42	.53	.53	.53
Round 2	.63	.46	.55	.55	.41
Round 3	.71	.52	.53	.53	.37
Round 4	.79	.57	.48	.48	.31
...	...	...	...	...	...
Round 11	.97	.61	.40	.40	.21

Table 4: Confidence of affiliations computed for *Carey* and *Halevy* in the motivating example.

	Carey			*Halevy*	
	UCI	AT&T	BEA	Google	UW
Round 1	1.61	1.61	2.0	2.1	2.0
Round 2	1.68	1.3	2.12	2.74	2.12
Round 3	2.12	1.47	2.24	3.59	2.24
Round 4	2.51	1.68	2.14	4.01	2.14
...	...	...	...	...	...
Round 11	4.73	2.08	1.47	6.67	1.47

only when appropriate and apply different "dampening factors" for different data sources.

Another major difference is that we compute the probability of a value being true in a different way. TRUTHFINDER computes it as the probability that at least one of its providers provides the true value and ignores sources that vote for other values. As they pointed out, they have the problem of "overly high confidence" if they do not apply the dampening factor. Our computation (Eq.(21)) considers all data sources and considers both the possibility that the value is true and the possibility that the value is false.

Section 6 presents an experimental comparison between the two approaches.

5. EXTENSIONS

This section describes several extensions of the ACCU model by relaxing the *Categorical-value* condition and the *Uniform-false-value-distribution* condition, and by considering probabilities of a value being true in dependence discovery. The extensions we present are complementary to each other and can be easily combined for a full model.

SIM: We consider similarity between values. Let v and v' be two values that are similar. Intuitively, the sources that vote for v' also implicitly vote for v and should be considered when counting votes for v. For example, a source that claims *UW* as the affiliation may actually mean *UWisc* and should be considered as an implicit voter of *UWisc*.

We extend ACCU by incorporating the similarity model in [16]. Formally, we denote by $sim(v, v') \in [0, 1]$ the similarity between v and v', which can be computed by edit distance of strings, difference between numerical values, etc. After computing the confidence of each value of object O, we adjust them according to the similarities between them as follows:

$$C^*(v) = C(v) + \rho \cdot \sum_{v' \neq v} C(v') \cdot sim(v, v'), \tag{26}$$

where $\rho \in [0, 1]$ is a parameter controlling the influence of similar values. We then use the adjusted confidence in computation in later rounds.

NonUni: In reality, false values of an object may not be uniformly distributed; for example, an out-of-date value or a value similar to the true value can occur more often than others. We extend ACCU for this situation as follows.

We define $f(d), d \in [0, 1]$, as the percentage of false values whose distribution probability is d; thus, $\int_0^1 f(d) = 1$. Then, the probability that two false-value providers provide the same value is $\int_0^1 d^2 f(d)$ instead of $(\frac{1}{n})^2 \cdot n = \frac{1}{n}$. Accordingly, we revise Eq.(13) as

$$Pr(O \in \bar{O}_f | S_1 \perp S_2) = \varepsilon(S_1)\varepsilon(S_2) \int_0^1 d^2 f(d) = P_f. \tag{27}$$

Similarly, we need to revise Eq.(19) as follows. Let $E = e^{\int_0^1 \ln df(d)(|\bar{S}_o| - |\bar{S}_o(v)|)}$.

$$Pr(\Psi(O)|v \text{ true}) = \Pi_{S \in \bar{S}_{o(v)}} A(S) \cdot \Pi_{S \in \bar{S}_o - \bar{S}_{o(v)}} (1 - A(S)) \cdot E. \tag{28}$$

AccuPR: As the decision of a value being true or false is rather probabilistic, we can use the probability in our dependence analysis. In particular, we denote by $Pr(S, v)$ the probability that source S independently provides value v; then

$$Pr(S, v) = P(v) \cdot A(S) + (1 - P(v)) \cdot \frac{1 - A(S)}{n}. \tag{29}$$

Accordingly, we compute probability of two sources providing a particular pair of values v_1 and v_2 respectively, denoted by $Pr(v_1, v_2)$. Then,

$$Pr(v, v | S_1 \perp S_2) = P(S_1, v) \cdot P(S_2, v) = P_c, \tag{30}$$
$$Pr(v_1, v_2 | S_1 \perp S_2) = P(S_1, v_1) \cdot P(S_2, v_2) = P_d, \tag{31}$$
$$Pr(v, v | S_2 \to S_1) = cP(S_1, v) + (1 - c)P_c, \tag{32}$$
$$Pr(v_1, v_2 | S_2 \to S_1) = (1 - c)P_d. \tag{33}$$

We can then apply the Bayes Rule similarly. We note that ACCUPR implicitly captures the intuition that a frequent-occurring false value is not a strong indicator of dependence: a frequent false value tends to have a high probability of being true and thus lowers the probability of dependence between its providers.

6. EXPERIMENTAL RESULTS

We applied our truth-discovery algorithms on a real-world data set. For the purpose of systematically testing our models in various conditions, many not easily found in real data, we also tested our algorithms on synthetic data. In addition, we examined if our algorithm can prevent falsification of true values.

6.1 Experimental settings

We first describe the synthetic data sets we generated and leave the description of the real-world data set to Section 6.5. We consider three types of *universes* of data sources, where each source provides information for all objects.

1. *Indep-source universe* contains 10 independent sources;
2. *Copier universe* contains 10 independent sources and 9 copiers that copy from the same independent source and provide 20% of the values independently;
3. *Pareto universe* contains 25 to 100 sources, of which 20% are independent and 80% are copiers. Among the independent sources, 20% have an error rate of .2 and 80% have an error rate of .5. Among the copiers, 80% copy from one of the more accurate independent sources and 20% copy from one of the less accurate independent sources. Also, among the copiers, 50% provide 10% of the values independently with an error rate of .1, 25% provide 10% of the values by randomly picking a value in the domain, and 25% only copy[2].

[2]We call it *Pareto universe* as it observes the *Pareto Rule (80/20 Rule)* in many aspects.

For the first two types of universes, we consider three cases: *no-authority*, where every independent source has the same error rate; *copy-from-non-authority*, where there is an *authority* source that provides the true value for each object, but the copiers copy from a non-authority source; and *copy-from-authority*, where there is an authority source and the copiers copy from it. Note that in the *Indep-source universe*, the latter two cases are the same since we have no copier.

For each type of universe and each case, we randomly generated the set of data sources according to ε_u, the error rate, n_u, the number of incorrect values, and o_u, the number of objects. The values range from 0 to n_u, where we consider 0 as the true value and the others as false. We varied ε_u from .1 to .9, n_u from 5 to 100, and o_u from 5 to 100. For *Pareto universe*, we in addition randomly decided from which source a copier copies according to the distribution. For each set of parameters, we randomly generated the data set 100 times, applied our algorithms to decide the true values, and reported the average precision of the results. We define *precision* of the results as the fraction of objects on which we select the true values (as the number of true values we return and the real number of true values are both the same as the number of objects, the *recall* of the results is the same as the precision). Note that this definition is different from that of accuracy of sources.

We implemented models DEPEN, ACCU, ACCUPR and SIM as described in this paper. We also implemented the following methods for comparison:

- NAIVE conducts naive voting;
- NAIVESIM conducts naive voting but considers similarity between values;
- ACCUNODEP considers accuracy of sources as we described in Section 4, but assumes all sources are independent;
- TF applies the model presented in [16].
- TFNOSIM is the same as TF except that it does not consider similarity between values.
- TFNODAM is the same as TFNOSIM except that it does not apply the dampening factor (0.3).

For all methods, when applicable we (1) set $\alpha = .2$ and $c = .8$, (2) set ε and n to the value used in generating the data sources, (3) set $\varepsilon = .25$ for the *Pareto universe*, and (4) set $\rho = 1$ for SIM. We implemented the algorithms in Java and conducted our experiments on a WindowsXP machine with AMD Athlon(tm) 64 2GHz CPU and 960MB memory.

6.2 Comparing dependence-detection models

We first compare DEPEN and ACCU with NAIVE on the first two types of universes. We report our results for $n_u = 100$ and $o_u = 100$ and briefly discuss the results for other parameter settings at the end of this section. Figure 5 and 6 plot the precision of the results.

In the *Indep-source universe*, DEPEN obtains the same precision as NAIVE, and ACCU obtains a precision of 1.0 when an authority source exists. Being able to obtain the same results as simple voting in absence of copiers is important: it shows that we do not generate false dependence that can change the voting results.

When there are copiers, NAIVE is biased by the copiers and performs badly. Our algorithms consider dependence between sources so obtain much higher precision. In particular, DEPEN successfully detects copiers and in general obtains similar results as if the copiers did not exist. ACCU obtains a precision of 1.0 when there exists an authority source and a similar precision to DEPEN otherwise. The only exception is when $\varepsilon_u = .9$ in the *no-authority* case, where the independent sources often do not agree with each other and so the values that are copied 9 times (even though ACCU detects the copying) have a slightly higher confidence. The copiers thus are considered to be more accurate and gradually dominate the results. This problem disappears when more independent sources are present.

Effects of parameters: We also conducted experiments with different parameter settings.

First, we varied parameters in the random generation of data sources. We observed that when there are fewer objects or fewer false values in the domain, there is less statistical evidence and so the precision of the results can be lower and less stable; when there are more sources, true values are provided by more sources and the precision can be higher.

Second, we varied voting parameters, including α, c, ε and n. We observed that ranging α and ε, the a-priori probabilities, from .1 to .9 does not change precision of the results. This observation is common in Bayes analysis. We also observed that ranging n from 10 to 100 does not change the precision and setting it to 1000 or 10000 even increases the precision. However, ranging c from 0 to 1 can significantly change the precision. As shown in Figure 7, in the *Copier universe* ($c_u = .8$) and *no-authority* case, when we set c to c_u, ACCU obtains a precision of .99 when $\varepsilon_u = .5$ and .66 when $\varepsilon_u = .8$. However, when we vary c, in case of $\varepsilon_u = .5$, the precision drops significantly when c is set to .2; in case of $\varepsilon_u = .8$, it drops significantly when c is set to .6. Recomputing c in each iteration in voting can effectively solve the problem: the precision remains stable when $\varepsilon_u = .5$ and drops at $c = .2$ when $\varepsilon_u = .8$. Thus, our models are not sensitive to initial setting of parameters in a reasonable range.

6.3 Comparing truth-discovery algorithms

To examine the effect of each algorithm in a more complex universe, we experimented on the *Pareto universe*. Figure 8 shows the precision for a universe with 100 values. We observe that ACCUPR, ACCU and DEPEN obtain the highest precision, showing that considering dependence between sources significantly improve results of truth discovery, and when more accurate sources are copied more often, considering accuracy of sources does not necessarily help. ACCUNODEP, TF and TFNODAM obtain even lower precision than NAIVE, showing that considering accuracy of sources while being unaware of dependence can become more vulnerable in presence of duplications. ACCUNODEP and TFNODAM both extend NAIVE with only analysis of source accuracy but do so in different ways; between them ACCUNODEP obtains better results.

Effects of assumptions and indirect copying: To examine effects of assumptions in Section 2 and indirect copying (transitive copying and co-copying from a hidden source) on our model, we change the *Pareto universe* as follows: (1) *dependent values*: the last 20 values provided by each independent source are the same as the first 20 values correspondingly, and the 81st to 90th values provided by each copier is the same as the first 10 values correspondingly; (2) *transitive copying*: a copier can copy from any independent source or other copier as far as no loop copying; (3) *co-copying from hidden sources*: high-accuracy independent sources are removed; (4) *loop copying*: each low-accuracy independent source S has a peer S', such that S provides the first half of values independently and S' provides the second half independently, and they each copy the rest of the data from each other; (5) *dependent copying*: copiers are divided into pairs, and sources of each pair copy from the same source and copy the same values. We observe exactly the same precision of ACCU with different combinations of these changes when there are at least 50 sources. When there are

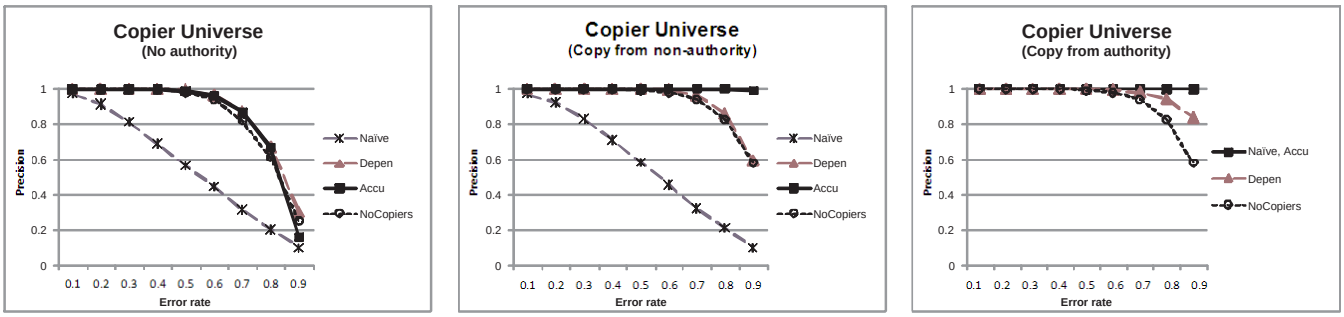

Figure 5: Precision in the *Copier universe*. For comparison, we also plotted the precision of NAIVE when there is no copier.

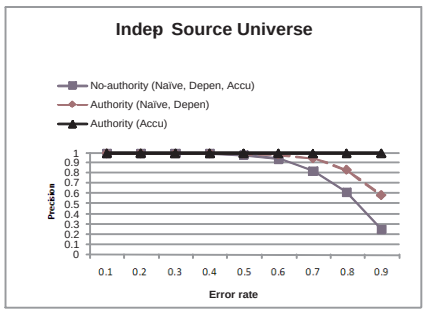

Figure 6: Precision in the *Indep-source universe*.

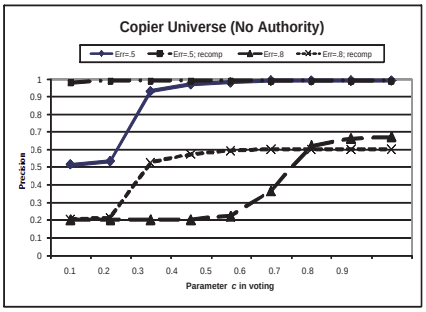

Figure 7: Precision of results by ACCU when parameter c varies.

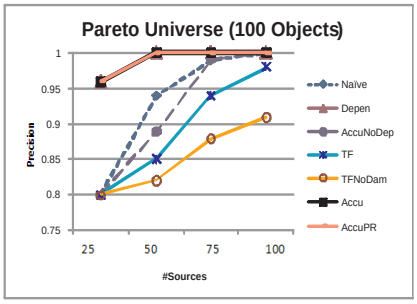

Figure 8: Precision in the *Pareto universe*.

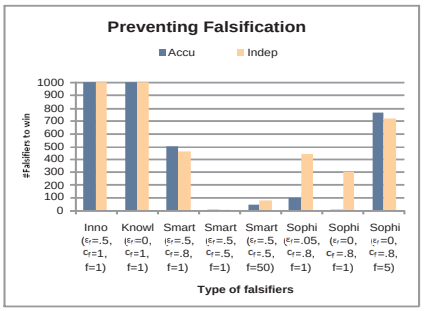

Figure 9: Number of falsifiers required to falsify a set of true values. Our algorithms can effectively prevent innocent and knowledgeable falsifiers from falsifying one true value and prevent smart and sophisticated falsifiers from falsifying multiple true values.

only 25 sources (5 independent), no combination of the first three changes the precision, change (4) increases the precision by .01, and change (5) decreases the precision by .02, showing robustness of ACCU.

6.4 Preventing falsification

We next studied whether our algorithms can prevent falsification. We consider a *Pareto universe* with 25 data sources (so 5 independent sources) and a set of falsifiers who intend to falsify the true values on a set of f objects. Among the falsifiers, one is a bad independent source and the others are copiers. For each object that the falsifiers intend to falsify, all falsifiers provide value -1, which is not provided by any other source in the universe. For the rest of the objects, the independent falsifier provides values observing error rate ε_f, and the copiers either copy from the independent falsifier with probability c_f, or randomly pick a value from the domain.

We classify falsifiers into the following four categories.

- *Innocent:* $\varepsilon_f = .5$ and $c_f = 1$;
- *Knowledgeable:* $\varepsilon_f = 0$ and $c_f = 1$;
- *Smart:* $\varepsilon_f = .5$ and $c_f < 1$;
- *Sophisticated:* ε_f is very low and $c_f < 1$.

We want to find out how many falsifiers are required to falsify the true values for f objects. For each category of falsifiers, we started from one randomly generated independent falsifier and gradually added copier falsifiers. If -1 is selected as the true value for each object of falsification three times consecutively, we stopped and reported the number of falsifiers. If after reaching 1000 falsifiers the falsifiers still cannot succeed, we stopped and reported 1000. Figure 9 shows the results for different types of falsifiers. We have several observations.

First, it is very hard for innocent and knowledgeable falsifiers to falsify the true values: even 1000 falsifiers cannot falsify one true value against 25 non-falsifiers.

Second, when $c_f = .8$, it requires around 500 smart falsifiers to falsify one true value; but when $c_f = .5$, the falsifiers look like independent sources and only 6 falsifiers are required under the ACCU algorithm. From another perspective, this result indicates that even if a wrong value is provided by a set of more accurate sources, a small number of independent sources *do* have the chance to fix it.

Third, under the ACCU model it is easy for sophisticated falsifiers with $\varepsilon_f = 0$ and $c_f = .8$ to win: only 4 falsifiers are required to falsify one value. From another perspective, this indicates that a value provided by an authority data source is more likely to be considered as true, even if a different value is provided by several less accurate sources.

Fourth, it is hard even for sophisticated falsifiers to falsify multiple true values. Falsifying 5 true values requires more than 700 sophisticated falsifiers with $\varepsilon_f = 0$ and $c_f = .8$. However, it is still easy for smart falsifiers to falsify a set of true values: no more

Table 5: Different types of errors by naive voting.

Missing authors	Additional authors	Mis-ordering	Mis-spelling	Incomplete names
23	4	3	2	2

Table 6: Results on the book data set. For each method, we report the precision of the results, the run time, and the number of rounds for convergence. ACCU **and** DEPEN **obtain a high precision.**

Model	Precision	Rounds	Time(s)
NAIVE	.71	1	.2
NAIVESIM	.74	1	.2
ACCUNODEP	.79	23	1.1
DEPEN	.83	3	28.3
ACCU	.87	22	185.8
SIM	.89	18	197.5
TFNOSIM	.71	10	.5
TF[5]	.83	8	11.6

than 100 sources are required to falsify 50 true values, reflecting one direction for improvement.

6.5 Experiments on real-world data

We experimented on a real-world data set also used in [16][3] (we removed duplicates). The data set was extracted by searching computer-science books on *AbeBooks.com*. For each book, *AbeBooks.com* returns information provided by a set of online bookstores. Our goal is to find the list of authors for each book. In the data set there are 877 bookstores, 1263 books, and 24364 listings (each listing contains a list of authors on a book provided by a bookstore).

We did a pre-cleaning of authors' names and generated a normalized form that preserves the order of the authors and the first name and last name (ignoring the middle name) of each author. On average, each book has 19 listings; the number of different author lists after cleaning varies from 1 to 23 and is 4 on average. We applied various models on this data set and set $\alpha = .2, c = .8, \varepsilon = .2$ and $n = 100$ when applicable. Though, we observed that ranging α from .05 to .5, ranging c from .5 to .95, and ranging ε from .05 to .3 did not change the results much. We compared similarity of two author lists using 2-gram Jaccard distance.

We used a golden standard that contains 100 randomly selected books and the list of authors found on the cover of each book (the same as used in [16]). We compared the results of each method with the golden standard and reported the precision. We consider missing or additional authors, mis-ordering, misspelling, and missing first name or last name as errors; though, we do not report missing or misspelled middle names[4]. Table 5 shows the number of errors of different types on the selected books if we apply a naive voting (note that the result author lists on some books may contain multiple types of errors).

Precision and Efficiency Table 6 lists the precision of each algorithm. SIM obtained the best results and improved over NAIVE by 25.4%. NAIVESIM, ACCUNODEP and DEPEN each extends NAIVE on a different aspect; while all of them increased the precision, DEPEN increased it the most. We also observed that considering similarity between author lists increased the precision of ACCU only slightly (by 2.3%), but increased the precision of TFNOSIM significantly (by 16.9%); indeed, TFNOSIM obtained the same precision as NAIVE.

To further understand how considering dependence and precision of sources can affect our results, we looked at the books on which ACCU and NAIVE generated different results and manually found the correct authors. There are 143 such books, among which ACCU and NAIVE gave correct authors for 119 and 15 books respectively, and both gave incorrect authors for 9 books.

Finally, DEPEN was quite efficient and finished in 28.3 seconds. It took ACCU and SIM longer time to converge (3.1, 3.3 minutes respectively); though, truth discovery is often a one-time process and so taking a few minutes is reasonable.

Table 7: Bookstores that are likely to be copied by more than 10 other bookstores. For each bookstore we show the number of books it lists and its accuracy computed by SIM**.**

Bookstore	#Copiers	#Books	Accu
Caiman	17.5	1024	.55
MildredsBooks	14.5	123	.88
COBU GmbH & Co. KG	13.5	131	.91
THESAINTBOOKSTORE	13.5	321	.84
Limelight Bookshop	12	921	.54
Revaluation Books	12	1091	.76
Players Quest	11.5	212	.82
AshleyJohnson	11.5	77	.79
Powell's Books	11	547	.55
AlphaCraze.com	10.5	157	.85
Avg	12.8	460	.75

Table 8: Difference between accuracy of sources computed by our algorithms and the sampled accuracy on the golden standard. The accuracy computed by ACCU **is the closest to the sampled accuracy.**

	Sampled	SIM	ACCU	ACCU NODEP	TF NOSIM
Avg src accu	.542	.607	.614	.623	.908
Avg diff	-	.082	.087	.096	.366

Dependence and source accuracy: Out of the 385,000 pairs of bookstores, 2916 pairs provide information on at least the same 10 books and among them SIM found 508 pairs that are likely to be dependent. Among each such pair S_1 and S_2, if the probability of S_1 depending on S_2 is over 2/3 of the probability of S_1 and S_2 being dependent, we consider S_1 as a *copier* of S_2; otherwise, we consider S_1 and S_2 each has .5 probability to be a *copier*. Table 7 shows the bookstores whose information is likely to be copied by more than 10 bookstores. On average each of them provides information on 460 books and has accuracy .75. Note that among all bookstores, on average each provides information on 28 books, conforming to the intuition that small bookstores are more likely to copy data from large ones. Interestingly, when we applied NAIVE on only the information provided by bookstores in Table 7, we obtained a precision of only .58, showing that bookstores that are large and referenced often actually can make a lot of mistakes.

Finally, we compare the source accuracy computed by our algorithms with that sampled on the 100 books in the golden standard. Specifically, there were 46 bookstores that provide information on more than 10 books in the golden standard. For each of them we computed the *sampled accuracy* as the fraction of the books on which the bookstore provides the same author list as the golden standard. Then, for each bookstore we computed the difference between its accuracy computed by one of our algorithms and the sampled accuracy (Table 8). The source accuracy computed by

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

[4]Note that the precision we reported is not comparable with that reported in [16], as their partially correct results are each given a partial score between 0 and 1, mis-ordering of authors is not penalized, but incorrect or missing middle name is penalized.

[5][16] reports that correct authors were provided for 85 books; however, they did not count mis-ordering of authors as incorrect.

SIM is the closest to the sampled accuracy, indicating the effectiveness of our model on computing source accuracy and showing that considering dependence between sources helps obtain more precise source accuracy. The source accuracy computed by TFNOSIM is too high, consistent with the observation of *overly high confidence* made in [16].

6.6 Summary

Our experiments on real-world and synthetic data show the following features of our models.

- In presence of source dependence, our models significantly improve truth-discovery results by considering dependence between sources; in absence of dependence, our models do not generate false dependence that can change the voting results, thus obtain similar results as not considering dependence.
- Though our algorithms do not tend to capture every variant of the real world, they are robust with respect to different settings of parameters, violations of assumptions, and indirect copying, and they apply well even if the real data do not conform to our models.
- Our models can effectively prevent falsification in most cases.
- As a by-product, the source accuracy our models compute is similar to the percentage of true values over all values provided by the source.

7. RELATED WORK

We are not aware of any existing work on detecting dependence between data sources. *Data provenance* [5] is a hot research topic but it focuses on managing provenance already provided by users or applications. *Opinion pooling*, which combines probability distribution from multiple experts and arrives at a single probability distribution to represent the consensus behavior, has been studied in the context of dependent experts in [6, 8, 11]; however, these works did not study how to discover such dependence. Moss [13] detects plagiarism of programs by comparing *fingerprints* (k-grams) of the programs; our method is different in that we consider values provided for different objects in databases.

There has been many works studying how to assess trustworthiness of data sources. Among them, PageRank [4], Authority-hub analysis [10], etc., decide authority based on link analysis [3]. EigenTrust [9] and TrustMe [14] assign a global trust rating to each data source based on its behavior in a P2P network. The strategy that is closest to ours is TruthFinder [16], with which we have compared in detail in Section 4.4 and in experiments.

Finally, a lot of research has been done on combining conflicting data from multiple sources. Bleiholder and Naumann [2] surveyed existing strategies for resolving inconsistency in structured or semi-structured data. Wu and Marian [15] proposed aggregating query results from different web sources by considering importance and similarity of the sources. Our algorithm differs from theirs in that we developed formal models to discover dependence between data sources and accuracy of sources, based on which we decide truth from conflicting information.

8. CONCLUSIONS

In this paper we studied how to improve truth discovery by detecting dependence between sources and analyzing accuracy of sources. We considered a snapshot of data and developed Bayesian models that discover copiers by analyzing values shared between sources. The results of our models can be considered as a probabilistic database, where each object is associated with a probability distribution of various values in the underlying domain. Experimental results show that our algorithms can significantly improve accuracy of truth discovery and are scalable when there are a large number of data sources.

Our work is a first step towards integrating data among sources where some can copy from others. There are many future topics under this umbrella. First, we are extending our current models by considering evolution of true values and evolution of data in a dynamic world. Second, we plan to combine techniques of record linkage and truth discovery to enhance both of them. Third, we plan to leverage knowledge of dependence between sources to answer queries more efficiently in a data integration system.

9. REFERENCES

[1] L. Berti-Equille, A. D. Sarma, X. L. Dong, A. Marian, and D. Srivastava. Sailing the information ocean with awareness of currents: Discovery and application of source dependence. In *CIDR*, 2009.

[2] J. Bleiholder and F. Naumann. Conflict handling strategies in an integrated information system. In *WWW*, 2006.

[3] A. Borodin, G. Roberts, J. Rosenthal, and P. Tsaparas. Link analysis ranking: algorithms, theory, and experiments. *ACM TOIT*, 5:231–297, 2005.

[4] S. Brin and L. Page. The anatomy of a large-scale hypertextual Web search engine. *Computer Networks and ISDN Systems*, 30(1–7):107–117, 1998.

[5] P. Buneman, J. Cheney, and S. V. Wang-Chiew Tan. Curated databases. In *Proc. of PODS*, 2008.

[6] K. Chang. *Combination of opinions: the expert problem and the group consensus problem*. PhD thesis, University of California, Berkeley, 1985.

[7] X. L. Dong, L. Berti-Equille, and D. Srivastava. Integrating conflicting data: the role of source dependence. http://www.research.att.com/∼lunadong/publication/indep_techReport.pdf.

[8] S. French. Updating of belief in the light of someone else's opinion. *Jour. of Roy. Statist. Soc. Ser. A*, 143:43–48, 1980.

[9] S. Kamvar, M. Schlosser, and H. Garcia-Molina. The Eigentrust algorithm for reputation management in P2P networks. In *Proc. of WWW*, 2003.

[10] J. M. Kleinberg. Authoritative sources in a hyperlinked environment. In *SODA*, 1998.

[11] D. Lindley. Reconciliation of probability distributions. *Oper. Res.*, 31:866–880, 1983.

[12] Master data management. http://en.wikipedia.org/wiki/Master_Data_Management.

[13] S. Schleimer, D. S. Wilkerson, and A. Aiken. Winnowing: Local algorithms for document fingerprinting. In *Proc. of SIGMOD*, 2003.

[14] A. Singh and L. Liu. TrustMe: anonymous management of trust relationships in decentralized P2P systems. In *IEEE Intl. Conf. on Peer-to-Peer Computing*, 2003.

[15] M. Wu and A. Marian. Corroborating answers from multiple web sources. In *Proc. of WebDB*, 2007.

[16] X. Yin, J. Han, and P. S. Yu. Truth discovery with multiple conflicting information providers on the Web. In *Proc. of SIGKDD*, 2007.