Heterogeneous Network-Based Trust Analysis: A Survey

Manish Gupta
University of Illinois at Urbana Champaign
gupta58@illinois.edu

Jiawei Han
University of Illinois at Urbana Champaign
hanj@cs.uiuc.edu

ABSTRACT

Different information sources publish information with different degrees of correctness and originality. False information can often result in considerable damage. Hence, trustworthiness of information is an important issue in this data-driven world economy. Reputation of different agents in a network has been studied earlier in a variety of domains like e-commerce, social sciences, sensor networks, and P2P networks. Recently there has been work in the data mining community on performing trust analysis based on the data provided by multiple information providers for different objects, and such agents and their provided information about data objects form a multi-typed heterogeneous network. The trust analysis under such a framework is considered as *heterogeneous network-based trust analysis*. This paper will survey heterogeneous network-based trust analysis models and their applications. We would conclude with a summary and some thoughts on future research in the area.

Keywords

Trust analysis, truth finder, trust models, source dependency, truth evolution, challenges in trust analysis, provenance, veracity analysis, fact finding

1. INTRODUCTION

The amount of published data every year is growing exponentially. Generating data and publishing it openly have become very easy, thanks to the immense development of websites that promote social interactions (social networks, social sharing sites, discussion forums, blogs, microblogs, content sharing sites). Apart from these sites, there are millions of other websites which publish fresh content everyday and attract a large amount of traffic. Thanks to the wide usage of Internet, this published data reaches out a large number of people very quickly.

The ICCPR (International Covenant on Civil and Political Rights) recognizes the right to freedom of speech as "the right to hold opinions without interference."[1] As a result, there is no punishment for publishing false information. This leniency is often exploited by many information sources who try to manipulate strategic decisions by providing false information.

Need for trust analysis: There is a lot of incorrect data present both on the surface web and in the deep web. For a novice user, it is often not easy to identify and ignore incorrect data. A recent survey report [22] mentions that U.S. consumers have low (8%) trust in online information sources. Many good examples of erroneous information and their propagation on the web can be found in [17]. There are a few websites[2,3] that report latest rumours and hoaxes. Also, there are a few websites that offer for verification of selected political facts like Politifact[4] and FactCheck[5]. Google Maps[6] has recently started showing "Reputation trend" on its "More info" page. But all such fact verification services are limited in scope. Verifying the correctness of claims related to any object made by a surface/deep web source is a challenging task.

Problems	Approaches
Establish reputation of sources (agents)	Survey [36], EigenTrust [29], Advogato [32], StereoTrust [34], PageRank [9], HITS [30]
Establish robustness of information channel	Overview [43], PGP [52]
Establish correctness of information (fact finding) and trustworthiness of sources	See Tables 2, 3 and 4
Data lineage/Provenance (tracing origins and history)	Survey [41], Trio [5]

Table 1: Problems under Trust Analysis

Trust analysis is a broad area covering many problems, some of which we list in Table 1. In this survey, we aim at studying the problem of establishing correctness of information and trustworthiness of sources. We would review very briefly the research to solve other trust problems and then present our roadmap for the problem of establishing correctness of information and trustworthiness of sources.

Other Trust Problems: There has been work in literature on establishing trust in a variety of areas including social sciences, e-commerce, mobile ad-hoc networks, sensor networks, P2P networks, etc. (see [36] for a survey). Previous work in this area has mainly focused on establishing reputation of sources by building a homogeneous network of sources. Edges in homogeneous networks denote relationships between sources (like "agrees", "votes", "recommends", etc.) Popular systems to tackle this problem include EigenTrust [29], Advogato [32], StereoTrust [34],

[1] http://www2.ohchr.org/english/law/ccpr.htm

[2] http://urbanlegends.about.com/od/reference/a/new_uls.htm

[3] http://www.pprune.org/rumours-news-13/

[4] http://politifact.com/

[5] http://www.factcheck.org/

[6] http://maps.google.com/maps/place?um=1&q=kopi+champaign&cid=513128417189510010

PageRank [9], HITS [30]. EigenTrust [29] computed the trustworthiness of each agent based on their past interactions using power iterations. In Advogato [32], users certify each other in a kind of peer review process and use this information to avoid the abuses that plague open community sites. In PageRank [9] and HITS [30], the webpages (sources) "recommend" other webpages, which maps to real-world hyperlinks. Most of these works use iterative algorithms for trust propagation over the network. In StereoTrust [34], a user (graph node), forms stereotypes using her previous transactions with other agents. When facing a stranger, the stereotypes matching stranger's profile are aggregated to derive her expected trust. However, homogeneous networks are based on the simplistic philosophy that sources (or agents) do not interact with any other entities, which is not true. *As a result, homogeneous networks end up having far less information compared to a heterogeneous network which can consist of more entity types like sources, facts, and objects.* Provenance and data lineage are supported by some systems like Trio [5] where data, uncertainty of the data, and data lineage are all first-class citizens (see [41] for a survey). Such probabilistic uncertain database systems assume that the confidence of each of the tuples in database is already known. But we want to focus on works that deal with populating databases with sound probabilistic data. There has been a lot of work to establish the trustworthiness of the channel through which the information has been propagated using public key infrastructure (PKI). In PGP (Pretty Good Privacy) proposed by Zimmermann [52], a user evaluates the trustworthiness of a public-key certificate based on the trust he places on the signers (also called introducers) of the certificate. PGP cannot prevent malicious sources from sending wrong data. In terms of information based approaches to establish trust, there has been some work ([1] and [50]) to determine trust in a wiki's text passages from sequences of revisions but they lack the claim-level granularity and general applicability of fact finders.

Establish correctness of information (fact finding) and trustworthiness of sources: In this work, our aim is to focus on research work in the area of fact finding. Typically the work is based on heterogeneous networks which consist of multiple types of nodes: information sources (providers) and facts (claims) in the simplest models. Sources provide facts about objects. Sometimes, one of the types of entities is folded in as edges between the other entity type, effectively transforming the network to a homogeneous one (network of sources or of facts). Often times, other supporting entities are also added into consideration. We classify the work in the area based on the network design in Table 2. Also, we present the time complexity comparisons of different algorithms in Table 5.

Sub-problems in fact finding: A number of sub-problems have been defined in the area of fact finding. Table 3 lists down some of such problems.

In this work, we would briefly describe each of the above problem settings and discuss various approaches (Sections 2–7) for each of them in detail. For each approach, we would describe the network design chosen, methodology used, datasets used and interesting results discovered.

Fact finding algorithms have been put to use in a variety of applications. Table 4 lists some of the applications. Towards the end of the survey (Section 8), we would discuss these applications briefly.

Table 2: Classification based on Trust Network Design

Network Design	Approaches
Facts only (Section 5.2)	*SSTF* [46]
Sources only (Sections 5.1, 5.3)	*SourceRank* [3; 2; 4], *MembersOnly* [37]
Facts +Sources +Objects (Sections 2, 3, 4, 6)	*TruthFinder* [45], *Sums, Average.Log, Investment, Pooled Investment* [38], *Cosine, 2-Estimates, 3-Estimates* [24], *Basic Cluster-Based Fact Finder, Advanced Cluster-Based Fact Finder* [27], *Common-Sense as Logic* [38], *Solomon* [6; 18; 15; 16], *Truthfulness +completeness +bias* [39]
Facts +Sources +Objects +Time (Section 4.3, 7.1)	*CopyCEF, CopyCEFDelay* [19], *Temporal inconsistency detection* [51]
Rich Facts +Sources +Objects (Section 4.5)	*Attribute-Based Consensus* [8]
Facts +Sources +Objects with Weights (Section 3.2)	*Probabilistic Assertions* [40]
Trust Extended Proof Network (Sources +Premises +Inference Rules +Conclusions) (Section 7.2)	*Trust Using Argumentation* [42]

Outline: In the next section, we would introduce a variety of fact finder models designed to perform trust analysis. In Section 3, we would present four extensions of fact finder algorithms: (i) Incorporating Hardness of Facts, (ii) Uncertainty of Claims, (iii) Cluster-Based Fact Finding, and (iv) Incorporating Common-Sense Reasoning. In Section 4, we would discuss various works based on copycat detection in static and dynamic settings. Also, complex copying relationships like co-copying and transitive copying are discussed. Finally, in the section, we would also discuss detection of copiers using facts with multiple attributes. In Section 5, we would discuss trust analysis models based on homogeneous networks like a providers network and a facts network. We would also discuss a semi-supervised approach to perform trust analysis in this section. Multiple metrics have been suggested in literature to measure trust. We present some of them in Section 6. In Section 7, we would discuss logic-based trust analysis models, one related to temporal logic, the other related to trust using argumentation. Finally, in Section 8, we would discuss in brief about different applications for which fact finder models have been used.

2. BASIC ITERATIVE FACT FINDER MODELS

In this section, we would introduce the basic setting for fact finders and the iterative philosophy used by most of the fact finders. A typical provider-facts network (mainly a bipartite graph) can be represented as shown in Figure 1. Each object is associated with a set of facts. Each information provider can provide facts related to multiple objects, but an information provider can provide only one fact about an object. The network can optionally contain implications between facts. Let p denote a provider and f denote a fact (such as an author list published for a book by a website).

Table 3: Sub-problems in fact finding

Problem Setting	Approaches
Basic Fact Finding (Section 2)	*TruthFinder* [45], *Sums, Average.Log, Investment, Pooled Investment* [38]
Incorporating Hardness of Facts (Section 3.1)	*Cosine, 2-Estimates, 3-Estimates* [24]
Uncertainty of Claims (Section 3.2)	*Probabilistic assertions* [40]
Cluster-Based Fact Finding (Section 3.3)	*BCFF, ACFF* [27]
Incorporating Common-Sense Reasoning (Section 3.4)	*Common-Sense as Logic* [38]
Source Dependency Detection (Section 4)	*Solomon* [6; 18; 15; 16]
Source Copying Detection in Dynamic Settings (Section 4.3)	*CopyCEF, CopyCEFDelay, LifeSpan* [19]
Source Dependency Detection using Multiple Attributes (Section 4.5)	*Attribute-Based Consensus* [8]
Fact Finding in Presence of Disconnected Sources (Section 5.1)	*SourceRank* [3; 2; 4]
Fact Finding in Presence of Labeled Data (Section 5.2)	*Semi-Supervised Truth Finder* [46]
Fact Finding in Presence of Non-Cooperative Sources (Section 5.3)	*MembersOnly* [37]
Fact Finding using Multiple Trust Metrics (Section 6)	*Truthfulness +completeness +bias* [39]
Tracking Facts over Time (Sections 7.1, 4.3)	*Temporal inconsistency detection* [51], *CopyCEF, CopyCEFDelay* [19]
Fact Finding with Proof Networks (Section 7.2)	*Trust Using Argumentation* [42]

Table 4: Fact Finding Applications (Section 8)

Application	References
Provenance-Based Access Control Systems (Section 8.1)	*Integrity Control System* [7; 13; 14]
Ranking Web Results (Section 8.2)	*Corroboration System* [44], [21]
Website Ranking (Section 8.3)	*Popularity & Influence Calculator* (PIC) [25]
Sensor Networks (Section 8.4)	*Apollo* [31]
Quality Inference on User-Generated Content (Section 8.5)	*Score Finder* [33]
Data Fusion (Section 8.6)	Tutorial [20]
News Finding (Section 8.7)	*TWEM, mergeTWEM* [35], *TrustNewsFinder* [49]
Information Credibility on Twitter (Section 8.8)	Supervised classification [12]

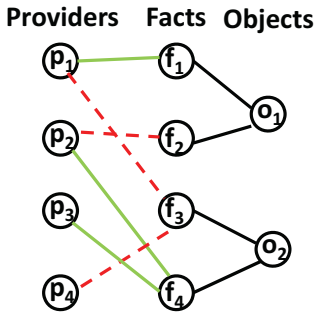

Figure 1: Facts, Objects and Providers Network

DEFINITION 2.1 (TRUSTWORTHINESS OF PROVIDER p). *$t(p)$ denotes the trustworthiness of provider p i.e., the probability that provider p would provide a true fact; e.g., the trust ranking of the website w, compared to the other websites.*

DEFINITION 2.2 (CONFIDENCE OF FACT f). *$s(f)$ denotes confidence of the fact f. Each fact is associated with an object.*

DEFINITION 2.3 (IMPLICATIONS BETWEEN FACTS). *Implication from fact f_1 to fact f_2 ($imp(f_1 \rightarrow f_2)$) denotes influence of fact f_1 on fact f_2. Positive value means f_1 supports fact f_2. Negative value means f_1 opposes fact f_2. Note implication values are defined only between facts related to a single object. The exact function used to compute implication values is domain specific. $-1 \leq imp(f_1, f_2) \leq 1$.*

Let $P(f)$ denote the set of providers that publish fact f and $F(p)$ denote the set of facts provided by provider p.
Most of the fact finder models proposed in the literature are unsupervised. Given a network expressed as a table of (provider, object, fact) tuples, the aim is to be able to predict the best fact for an object. The input database has (provider, object) as the primary key.
Voting: The easiest way of computing the best fact related to an object is to choose the one with maximum number of votes. This method, however, is quite naive and often does not provide good accuracy.
Next, we would discuss different fact finder models proposed in the literature.

2.1 TruthFinder

TruthFinder is the first unsupervised trust analysis method proposed by Yin et al. [45]. Algorithm 1 shows the *TruthFinder* algorithm which computes trustworthiness of all providers and confidence of all facts iteratively by expressing them in terms of each other. Confidence of a fact f is computed as $1-p$ where p is the probability that each of the providers that mention that fact are not trustworthy (Step 5). Further, in Step 8, the algorithm considers the influence of other related facts as an additional factor while computing the confidence of fact f. However, facts for different objects do not influence one another. Finally, trustworthiness of a provider p is computed as the average confidence of the facts published by that provider (Step 14). Alternate computation of trustworthiness and confidence is repeated till convergence. Yin et al. [45] use these parameters: δ (=0.001%) for convergence, ρ (=0.5) which decides the significance of contribution from related facts for the computation of the confidence of a fact, γ (=0.3) which acts as a compensation factor for source dependency behaviour (we would discuss this in detail in Section 4) among providers.

Algorithm 1 *TruthFinder* Algorithm

Input: 1. Facts f provided by different providers related to objects $o \in O$. 2. Implications matrix imp.
Initialize $t(p)$ to a value v $\forall p$, where $0 \leq v \leq 1$.
while $|normalize(t^t) - normalize(t^{t-1})| \geq \delta$ **do**
  for every fact f **do**
    $\sigma(f) = log(\prod_{p \in P(f)}(1 - t(p)))$
  end for
  for every fact f **do**
    $\sigma^*(f) = \sigma(f) + \rho \sum_{o(f')=o(f)} \sigma(f') imp(f' \rightarrow f)$
  end for
  for every fact f **do**
    $s(f) = \frac{1}{1+e^{\gamma \sigma^*(f)}}$
  end for
  for every provider p **do**
    $t(p) = \frac{\sum_{f \in F(p)} s(f)}{|F(p)|}$
  end for
end while
return $t(p)$ and $s(f)$ for every f and p

There are two flows of information in this iterative process: (1) Trust information flow across objects via the providers. (2) Implication flow between facts related to the same object.

This method makes the following assumptions.

- Usually there is only one true fact for a property of an object.
- This true fact appears to be the same or similar on different websites.
- The false facts on different websites are less likely to be the same or similar.
- In a certain domain, a website that provides mostly true facts for many objects will likely provide true facts for other objects.

Apart from the above assumptions, this basic fact finding methodology has the following drawbacks.

- **Non-Cooperative sources:** Sources are assumed to be cooperative. Which sources provide which facts is known beforehand. It is assumed that we perfectly know what information is provided by which source. Deep web sources are not considered.
- **Consensus learning:** It is assumed that an object is associated with only one type of fact. Multiple facts related to an object can be handled but consensus learning from multiple facts for an object would be interesting.
- **Clusters:** It is assumed that a provider either provides good facts for every object or bad facts for every object. There is no concept of clustering. But in real world, providers excel in particular areas.
- **Multi-valued truth:** It is assumed that there is only one true value for an object. But often, multiple values (set) could be true sometimes with different degrees of truth. Also, sometimes none of the facts provided for the object may be true.
- **Bad world:** It is assumed that number of providers providing the true fact are much more than the number of providers providing the bad fact.
- **Generalizing facts:** One fact can talk about only one object. While in real scenarios, often facts relate to multiple objects. There is no concept of record linkage except using an object identifier.
- **Too many parameters:** It has a large number of parameters. Hence, it is difficult to tune.
- **No convergence guarantees:** The algorithm may not converge. In general, a fixed number of iterations are performed. Accuracy is observed to fluctuate as iterations progress.

To overcome the problems with the basic fact finder model, other fact finder models and extensions have been proposed. We would look at other fact finder models in the next subsection and extensions to handle specific aspects in the next section.

2.2 Other Fact Finder Models

Pasternack et al. [38] introduced a few more fact finders which are based on the same framework as above, but differ in the way the confidence and trust values are computed. We describe some of the fact finders below.

Sums (Hubs and Authorities): Sources are hubs with 0 authority and facts are authorities with 0 hub score.

$s^i(f) = \sum_{p \in P(f)} t^{i-1}(p)$

$t^i(p) = \sum_{f \in F(p)} s^i(f)$

Both s^i and t^i are normalized by dividing by the maximum values in the respective vectors to prevent from them growing in an unbounded way. This is also done for the Average.Log and the Investment algorithms.

Average.Log: Computing $t^i(p)$ as an average of belief in its claims overestimates the trustworthiness of a source with relatively few claims; certainly a source with 90% accuracy over a hundred examples is more trustworthy than a source with 90% accuracy over ten. The *Average.Log* fact finder tries to achieve this by incorporating a log factor.

$s^i(f) = \sum_{p \in P(f)} t^{i-1}(p)$

$t^i(p) = log|F(p)|.\frac{\sum_{f \in F(p)} s^i(f)}{|F(p)|}$

Investment: In the *Investment* algorithm, sources "invest" their trustworthiness uniformly among their claims. The belief in each claim then grows according to a non-linear function G, and a source's trustworthiness is calculated as the sum of the beliefs in their claims, weighted by the proportion of trust previously contributed to each (relative to the other investors). Since claims with higher trust sources get higher belief, these claims become relatively more believed and their sources become more trusted.

$s^i(f) = G(\sum_{p \in P(f)} \frac{t^{i-1}(p)}{|F(p)|})$ where $G(x) = x^g$ with $g = 1.2$.

$t^i(p) = \sum_{f \in F(p)} s^i(f).\frac{t^{i-1}(p)}{|F(p)|.\sum_{p' \in P(f)} \frac{t^{i-1}(p')}{|F(p')|}}$

Pooled Investment: Let F be the set of all facts. For each fact f, let M_f be the mutual exclusion set ($M_f \subseteq F$) i.e. a set of facts (including f) that are mutually exclusive with one another. It is assumed that there is only one true fact per exclusion set. Like *Investment*, sources uniformly invest their trustworthiness in claims and obtain corresponding returns, so $t^i(p)$ remains the same, but now after the belief in the claims of mutual exclusion set M have grown according to G, they are linearly scaled such that the total belief of the claims in M remains the same as it was before applying $G(x) = x^g$.

Given, $H^i(f) = \sum_{p \in P(f)} \frac{t^{i-1}(p)}{|F(p)|}$,

$s^i(f) = H^i(f).\frac{G(H^i(f))}{\sum_{d \in M_f} G(H^i(d))}$; $G(x) = x^g$ with $g = 1.4$.

3. EXTENSIONS TO BASIC FACT FINDERS

The basic fact finder models are based on very simple representations. There has been some work to deal with more complicated settings. In this section, we would discuss four such settings. In the first one, iterative algorithms are used to incorporate hardness of fact i.e. what is the average probability of making a mistake on that fact. The second work deals with probabilistic assertions which considers the issue of sources providing facts with some uncertainty. The third work deals with trust analysis using clusters. The fourth work focuses on using common-sense reasoning in fact finding.

3.1 Incorporating Hardness of Facts

Intuitively, an answer to an easy question should earn less trust than to a hard one. Galland et al. [24] propose three iterative algorithms: *Cosine*, *2-Estimates* and *3-Estimates*, that estimate the truth values of facts and the trust in sources using this intuition. In their model, facts can take two values T or F. *Cosine* is based on the cosine similarity measure between the values for the facts mentioned by a source and given true values. For the estimate of the truth value of facts given the trustworthiness of sources, a simple averaging is used. *2-Estimates* uses two estimators for the truth of facts and the error of sources. It iteratively computes the confidence of the T value of the fact and the error probability of the source with appropriate normalization. Confidence in the T value of a fact is computed as the ratio of sum of 'positiveViews' and 'negativeViews' to 'totalViews' for the fact. 'positiveViews' is the sum of the trustworthiness (1-error) values of sources that support the T value for the fact. 'negativeViews' is the sum of the error probability of sources that support the F value for the fact. Similarly, error of the source is computed as the ratio of sum of 'posFacts' and 'negFacts' to number of total facts provided by the source. 'posFacts' is the sum of the confidence of the F value of the fact for facts for which this source provides a T value. 'negFacts' is the sum of the confidence of the T value of the fact for facts for which this source provides a F value. *3-Estimates* refines *2-Estimates* by also estimating how hard each fact is, i.e. the propensity of sources to be wrong on this fact. It estimates three factors iteratively: hardness of facts, confidence in the T value of the fact and the error (1-trustworthiness) of the sources. They perform experiments on synthetic datasets. They show that *2-Estimates* yields good results but is quite unstable. *Cosine* is most of the time significantly better than most baselines. *3-Estimates* consistently yields better results than *Cosine*. They also experiment with a question answer dataset from Hubdub[7] with 357 questions each having 1 to 20 expressed answers provided by 473 users, but only one correct answer. They also experimented with a general knowledge quiz dataset . On two other real-world datasets (a sixth-grade biology test, and results from Web search engines), they report that their algorithms do not perform better than the baselines.

3.2 Probabilistic Assertions

The sources may provide facts with some uncertainty. Pasternack et al. [40] present a framework which allows to elegantly incorporate knowledge such as the confidence of the information extractor and the attributes of the information sources. Traditional fact finders do not consider the situation in which sources provide claims with uncertainty or there is some uncertainty in the information extraction process. Rather than considering the provider-fact network as an unweighted bipartite graph of sources and claims, Pasternack et al. consider it as a k-partite weighted graph. In their lifted model, source $s \in S$ asserts claim $c \in C$ with weight $w(s,c) \in [0,1]$. $w(s,c)$ is calculated as $w_u(s,c) \times w_p(s,c) + w_\sigma(s,c) + w_g(s,c)$. $w_u(s,c)$ is the probability that s asserted c according to the information extractor (due to inherent ambiguity, OCR error, etc.), while $w_p(s,c)$ is the certainty expressed by the source in the claim; their product can be viewed as the expected probability of c according to s. Additionally, $w_\sigma(s,c)$ encodes similarity among claims (a source objects less to a claim similar to the one it asserts) and $w_g(s,c)$ provides an alternate method of encoding source groups and attributes. They rewrite the formulas used by the basic fact finders like *Sums*, *Average.Log*, *Investment* and *TruthFinder* so as to incorporate the weight of the edges when computing the trustworthiness of the sources and the confidence of the facts.

As a more general model, they propose a layered model where one can add new "layers" of groups and attributes to the existing two layers of sources and claims. The first additional layer connects directly to the sources and higher layers model meta-groups and meta-attributes, making the graph k-partite.

They present experiments on a *Population* dataset which consists of 44761 claims of city populations (extracted from *Wikipedia infoboxes*) from 171171 editors, with 308 true claims identified from census data as an evaluation set. They split Wikipedia editors into three groups: administrators, blocked users, and everyone else. As administrators are elected by the community, they can be expected to have rather high trustworthiness; conversely, blocked users can be expected to be rather untrustworthy. They incorporate these groups as an additional layer, creating a tripartite fact finding graph. They observe that their generic lifted fact finders with knowledge of the sources' groups produced significantly better results than the basic fact finders.

3.3 Cluster-Based Fact Finding

Consider the set of author lists for different books available on book selling websites. Some websites get accurate lists from the publishers themselves, while some of them use different automated methods to extract this information. Some others manually collect such information and can be prone to human errors. So, first there is a need to establish the trustworthiness of each of the websites across all the different books and then there is a need to cluster the books according to the similarity between the sets of "good" websites for each of the books. This clustering would be based on trustworthiness and may be quite different from natural clustering on a single dimension. This aspect of trust analysis is discussed in [27]. The authors perform clustering of objects based on provider trustworthiness profiles ($t_o(p)$) personalized to the particular object. Also, restrictive flow of trust information across objects, using clusters, can improve ranking accuracy of facts and providers.

If some natural clustering $C = \{c_k\}_{k=1}^{K}$ is known, which partitions objects into K clusters, one can run basic fact finder model for each of the clusters of objects separately. But, one would not use the information about providers related to objects in other clusters. Also, this method needs some input clustering. Clusters are fixed and depend on a particular dimension only.

[7] http://www.hubdub.com

Alternatively, one can compute provider trust per object and then cluster the objects using their object-conditional trust vectors. (Object-conditional trust, $t_o(p)$, is computed as the confidence of the fact provided by provider p for the object o. It denotes the trust of information provider p conditioned with respect to the object o.) This computation is much similar to the basic fact finder algorithm, except that trustworthiness of a provider conditional with respect to an object is computed as the confidence of the fact provided by the provider for that object. Finally, object conditional trust vectors are clustered using KMeans. This approach is referred to as *Basic Cluster-Based Fact Finder (BCFF)*.

Clustering in the trustworthiness space is a novel form of clustering and may provide quite different clusters compared to clustering on natural dimensions. Also, no training data or data related to any other dimensions of the objects are needed to perform this clustering. But, note that there is no trustworthiness information sharing between objects in this computation. An integrated clustering with trust analysis algorithm should start with an initial clustering, perform trust analysis using this clustering and then refine the clusters using the analysis obtained on the previous set of clusters. Cluster-conditional trust, $t_{c_k}(p)$, is defined as the trust of the provider p considering the facts published by the provider p related to objects in cluster c_k. Their *Advanced Cluster-Based Fact Finder (ACFF)* algorithm performs alternate clustering and cluster-conditional trust analysis iterations. In each iteration, they perform cluster-conditional trust analysis, compute object-conditional trust vectors using confidence values obtained from trust analysis and then perform reclustering of objects using the t_o vectors. The clustering steps bring similar objects together, while the trust analysis steps compute better cluster-conditional trust rankings and better fact confidence values. While performing trust analysis, the flow of trustworthiness information for a provider is restricted to be within the current cluster. Modifications in fact confidence values (in analysis steps) leads to better object-conditional trust vectors which are then re-clustered to get modified clusters using KMeans clustering. Finally, they smooth the cluster-based fact confidence scores with the global fact confidence scores.

They experimented with multiple real datasets like *Abebooks.com Books* Dataset, *Wikipedia Biography Infobox* Datasets and *Population* dataset. Cluster-Based Fact Finders show some gains over the basic fact finders. For *Population* dataset, with five clusters, they observed: Wikipedia contributors (providers) are clustered into (IL, CA, NY), (PA, VT, MI), (IL, AL, AR, CA), (IN, IA, GA), (MN, OH, NY, TX).

3.4 Using Common-Sense Reasoning

The basic fact finders are often a dramatic improvement over voting, but are incapable of taking advantage of the user's prior (common-sense) knowledge. Pasternack et al. [38] introduce a framework for incorporating prior knowledge into any fact finding algorithm, expressing both general "common-sense" reasoning and specific facts already known to the user as first-order logic and translating this into a tractable linear program. This linear program constrains the claim beliefs produced by a fact finder, ensuring that the belief state is consistent with both common-sense ("cities usually grow") and known facts ("Los Angeles is more populous than Wichita"). The framework iteratively performs the following: compute trustworthiness values using confidence of facts, update confidence of facts based on trustworthiness of providers and then correct confidence of facts using the linear program.

To create the constraints, first they propositionalize the first-order formulae into propositional logic. Each claim c is represented by a proposition, and a $[0, 1]$ variable in the linear program (LP) corresponding to the confidence of claim ($s(c)$). Then propositional formula are converted into conjunctive normal form. For each disjunctive clause consisting of a set P of positive literals (claims) and a set N of negations of literals, they add the constraint $\sum_{c \in P} c_v + \sum_{c \in N} (1 - c_v)$ where c_v denotes the $[0, 1]$ variable corresponding to each c. The left-hand side is the union bound of at least one of the claims being true (or false, in the case of negated literals). For mutually exclusive claims, this is a tight bound. After every iteration of the algorithm, confidence values (beliefs) of each fact (claim) are computed using the trustworthiness of providers. From these belief values, one can compute number of "votes" for each claim. Let CF be a cost function which denotes the distance between the new distribution of beliefs satisfying the constraints and the original distribution produced by fact finder. They use a cost function such that the cost for increasing the belief in a claim is proportional to the number of votes against it, and the cost for decreasing belief is proportional to the number of votes for it. Thus, the belief distribution found by the LP will be the one that satisfies the constraints while simultaneously minimizing the number of votes frustrated by the change from the original distribution. For recomputing the votes, they suggest two strategies: vote conservation and vote loss. Vote conservation reallocates votes such that the total number of votes in each mutual exclusion set remains the same after the redistribution. Under Vote Loss, a claim can only lose votes, ensuring that if other claims in the same exclusion set become less believable, c does not itself become more believable relative to claims in other mutual exclusion sets. Experiments were conducted over three domains (*city population*, *basic biographies*, and *American vs. British spelling*) with four datasets. For the *Population* dataset, the common-sense knowledge used was "Population grows over time" and "Larger cities are more populous". For *Biographies* dataset, they used the following common-sense rules: "nobody dies before they are born, people are infertile before the age of 7, nobody lives past 125, all spouses have overlapping lifetimes, no child is born more than a year after a parent's (father's) death, nobody has more than two parents, and nobody is born or dies after 2008 (dataset collection year). For *Spellings* dataset, they use the rule that if a spelling a is correct and of length ≥ 4, then if a is a substring of b, b is also correct. Their results show that this approach scales well to even large problems, both reducing error and allowing the system to determine truth respective to the user rather than the majority.

4. SOURCE COPYING DETECTION

Almost all algorithms discussed till now, are based on some kind of weighted bidirectional voting. But such an approach may fail if sources copy from each other. Often times, multiple sources copy information from each other. This is more prominent when the information being provided is very critical e.g., news rumours. In such scenarios, voting based mechanisms would incorrectly choose false facts (copied by large number of providers) over true facts. Hence, it is essential to be able to detect source dependencies between sources in a scalable and an efficient way. The Data Management Department at AT&T Labs presented a set of research problems, that deal with detecting source dependency issues, at CIDR [6] and then followed up with detailed problem de-

scriptions and solutions for both static and dynamic settings in [18; 15; 19; 16]. A follow-up work from Blanco et al. [8] focuses on copy detection using knowledge from multiple attributes. We would discuss all of these briefly in this section.

4.1 Challenges in Source Dependency Detection

Dependence between sources can arise when a source copies values from another source (similarity dependence), or when a source chooses to provide values that conflict with those provided by another source (dissimilarity dependence). Identifying dependence between sources can be invaluable when discovering truth, identifying unbiased consensus values, or characterizing benevolent versus malicious sources. Knowledge of the copying relationships between the providers and copying directions can help in a variety of technologies (such as data fusion, record linkage, query answering, web search, news aggregation, source recommendation systems, etc.) based on information supplied by these providers.

Intuitively, the higher the similarity between the data sources, the more is the likelihood of similarity dependence. Also, the higher the dissimilarity between the data sources, the more is the likelihood of dissimilarity dependence. But this is a very simplistic view which faces the following challenges.

- Accurate Sources: Accurate sources that independently provide true values would be determined as having a high similarity, which might lead to the erroneous conclusion that they are dependent.
- Different Coverage and Expertise: Some sources are specialist sources providing correct information in a particular domain, while some sources are generalist sources providing correct information for many domains. Specialist sources may copy information from others in areas outside their expertise.
- Lazy Copiers and Slow Providers: When an original data source updates its data over time, its copier may be lazy in updating (e.g., the updated data may be copied only in a batch mode), or it may not copy certain kinds of updates (e.g., deleting affiliations), etc. In such cases, the similarity between sources might again be low, leading to the erroneous conclusion that the sources are likely to be independent. On the other hand, an independent source may be slow and often behind other sources in updating values, and so appears to be a copier.
- Partial Dependence: A source may copy only a subset of information, reformat some of the copied information and provide other information independently. A dissimilarity dependent source may also do some partially. Hence, similarity (or dissimilarity) between the sources might not always be high, leading to the erroneous conclusion that the sources are likely to be independent.
- Correlated Information: A high similarity between the information provided by sources may be just because of a popular opinion (or correlation between items for which information is provided). Such a similarity could be wrongly considered as source dependence.
- Incomplete Observations: Sources may be non-cooperative. Some data sources may hide some of their data for security concerns. Incomplete observations can again lead to incorrect conclusions about the dependence between sources.

Probability of two independent events is the product of the probabilities of each individual event; any pair of events that violate this equation are dependent. This intuition could be used to solve the problem of discovering source dependence as follows. Let D_1 and D_2 be the data provided by sources S_1 and S_2. Then sources S_1 and S_2 are likely to be dependent if $P(D_1, D_2) \neq P(D_1) \times P(D_2)$. Next, one needs to determine the direction of source dependence. We can consider the data source whose different subsets of data show different properties (e.g., accuracy, average rating) as more likely to be dependent on the other.

Source dependence can be studied in two different settings: static and dynamic. The former setting deals with identifying source dependencies in a single network snapshot. The latter setting deals with the case when the data sources include temporal information as well.

4.2 Source Dependency Detection for Snapshots

For the static setting (presented by Dong et al. in [18]), data sources that share common false values are much more likely to be dependent than data sources that share common true values, as the probability that two independent sources provide the same false value for a set of identifiers is typically low. Direction of copying (or partial copying) can be determined as follows. If the accuracy of a data source (e.g., fraction of true values) on the subset of information it shares in common with another data source is significantly different from its accuracy on the remaining information, the data source is more likely to be a partial copier than an independent data source. To apply these intuitions to determine dependence, it appears that one would require knowledge of what is true and what is false. If, however, one wishes to determine the truth from amongst conflicting values provided by multiple sources, knowledge of the dependence between sources is critical. A solution strategy can be devised using Bayesian analysis by iteratively determining true values, computing accuracy of sources, and discovering dependence between sources.

A dependency exists between two sources S_1 and S_2 if they derive the same part of their data directly or transitively from a common source (can be one of S_1 and S_2). Accordingly, there are two types of data sources: independent sources and copiers. Further, 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.; cyclic copying is not possible on a single snapshot. The authors present a probability-based model to detect dependency of data sources (*Depen*). A basic algorithm called *Vote* uses the *Depen* model to iteratively compute the probability of dependence between each pair of data sources and also the vote count for each value of the object, considering the current dependencies between sources. The value with maximum votes is considered as the true value. In this model, accuracy of each source is assumed to be the same. This assumption is removed in the *Accu* model. The *AccuVote* algorithm iteratively computes the dependencies between every pair of sources, confidence of each value for each object and the accuracy of each source. Further, extensions of the *Accu* model are proposed: *Sim* which incorporates

similarity between values when computing the confidence of a value, *NonUni* which removes the assumption of "false values of an object being uniformly distributed", *AccuPR* which incorporates the accuracy of a source when computing the dependency between pair of sources.

They tested their algorithms on synthetic data and real-world data sets. The experimental results show that their algorithms can significantly improve accuracy of truth discovery and are scalable when there are a large number of data sources. Though they did not model transitive copying or co-copying in their dependency detection model, they show that their model works well even in presence of such complex copying scenarios. On *Books* dataset consisting of 1263 books, they showed that some online book stores could have as many as 17 copiers.

4.3 Time Varying Truth

For the temporal setting (presented by Dong et al. in [19]), true values can evolve over time. Also copying relationships can evolve over time (a source can stop copying and become independent, can change sources from which it copies, and can copy at some times and provide data independently at other times). Data sources that share common false values are much more likely to be dependent than data sources that share common recent or outdated true values. Data sources that perform the same updates in close enough time frame are more likely to be dependent, especially if the same update trace is rarely observed from other sources. If the accuracy of source S_1 on the subset of information it shares in common with, but provides earlier than, source S_2 is significantly different from its accuracy on the subset of information it shares in common with, but provides later than, S_2 , then S_1 and S_2 are likely to be similarity dependent. As in the static setting, a solution strategy can be devised based on an iterative scheme using Bayesian analysis, which would also need to address the following considerations: (i) Discover dependence patterns of a data source over time. For example, a copier is more likely to remain as a copier; it can even choose to copy periodically and to copy from the same data sources. (ii) Distinguish between the update pattern of a copied value and that of an independently provided value. For example, if a source copies the value of a particular identifier, it may keep updating it from the original source. On the other hand, if a source provides the value of an identifier independently, it may not overwrite it in later copying.

Quality of sources over time is modeled by their coverage (how many values in the history a source covers), exactness (how many updates conform to reality) and freshness (how quickly a source captures a new value). First, a Hidden Markon Model (HMM) is developed that decides whether a source is a copier of another source and identifies specific moments at which it copies. They propose a basic HMM for copying discovery and then extend it for periodical copying (timespan HMM). The basic HMM corresponding to the copying relationship between sources S_1 and S_2 consists of five hidden states:
$I, C1_c, C1_{\neg c}, C2_c, C2_{\neg c}$. State I represents independence of S_1 and S_2. States $C1_c$ and $C1_{\neg c}$ represent that S_1 is a copier of S_2; the former represents that S_1 is actively copying from S_2 while the latter represents S_1 not copying at that moment. States $C2_c$ and $C2_{\neg c}$ represent that S_2 is a copier of S_1. Now, once a source remains as a copier, it copies sooner or later. Typically, the probability of copying by a source would be higher if the source has not yet been in copying mode. It is also possible that a copier copies periodically; it makes independent updates for a period of time and then copies recent updates for a period of time and then copies the recent updates by the original source. To incorporate such intuitions, they extended the basic HMM to include more hidden states. The state $C1_{\neg c}$ is divided into a set of states $C1^1_{\neg c}, C1^2_{\neg c}, \ldots C1^q_{\neg c}$ where q is the number of observations within which a copier typically will conduct at least one copying. Similarly, state $C2_{\neg c}$ is divided into multiple states. Transitions between states are constrained to incorporate the intuitions appropriately. The HMMs use the intuition that although the copying relationship between a pair of sources can evolve over time, frequent radical change is less likely i.e., a copier is more likely to remain as a copier. Second, a Bayesian model is developed that aggregates information from the sources (considering source quality and data copying) to decide the true value for a data item, and the evolution of true values over time. This model also computes the life span of the objects. In each round of their iterative algorithm, they first compute the CEF (coverage, exactness and freshness) measure of each source (depending only on life spans), then compute the probability of copying between sources, and finally (re-)decide the life span of each object, until the discovered life spans do not change. In the first round, life spans are unknown and so CEF measure of each source is initialized to the same default value except setting the coverage as the percentage of objects being covered by the source. In addition, they also consider different publish patterns such as instant publishing or batch-mode publishing.

They perform experiments on synthetic data which consists of 100 objects and 102 different values for each object with 20 different snapshots. They show that their algorithms are robust to the quality of the sources being copied and to publish delay. Also, their HMMs are shown to be good at detecting transformations of copiers. The basic HMM is accurate in detecting copying; the timespan HMM improves the results only slightly, but with higher cost.

4.4 Copying in Complex Scenarios

Till now, we discussed about simple copying relationships between sources in static and dynamic settings. All these techniques discovered relationships in a local way (by considering a pair of sources at a time). So, they cannot distinguish between complex copying relationships like co-copying, transitive copying and copying from multiple sources. Also there could be correlations in copying of data items (e.g. a source that copies the name of a book tends to also copy the author list). All techniques mentioned above view common mistakes as important evidence of copying but neglect other kinds of evidence such as whether the data are formatted in the same way, and whether two sources provide similar sets of real-world objects.

Dong et al. [15] present techniques that discover global copying relationships between a set of structured sources for static data. Though they consider many complex copying scenarios, some situations like mutual copying (S_1 copies from S_2 and S_2 copies from S_1 on different objects) are ignored. Copying detection proceeds in two steps.

The first step locally decides possibility of copying and copying direction between each pair of sources. For making more accurate decisions on the copying direction, critical for global detection, the previous model is enhanced by gleaning more evidence such as completeness and formatting of data, and considering correlated copying on data items. A data item is profiled using completeness, accuracy, and formatting style. Similarly, every source is also profiled with re-

spect to these three measures. The key in deciding whether S_1 copies from S_2 ($S_1 \rightarrow S_2$) is to decide if the probability of S_1 providing the observed data conditioned on it being independent of S_2 is much lower than that conditioned on it being a copier of S_2. Intuitively, the former probability will be much lower than the latter in two cases: first, when the two sources share low-completeness items, low-probability values, or low-popularity formats; second, when there is a big difference between the profile of the overlapping data and that of S_1's self-provided data. Correlated copying takes into account the fact that a copier may do per-object copying (copy a subset of objects on a subset of attributes) or do per-attribute copying (copies on a subset of attributes for a set of independently provided objects). The difference is whether the copier also copies the key values or not.

The second step globally identifies co-copying and transitive copying and these copying relationships are distinguished from a source indeed copying from multiple sources. The key intuition is that since co-copying and transitive copying can often be inferred from direct copying, they first find a set of copying relationships R that significantly influence the rest of the relationships, and then adjust the rest accordingly and decide if each is a direct or indirect copying (the results are denoted by $P(S_1 \rightarrow S_2|R)$, $(S_1, S_2) \notin R$). Two problems to solve are: (1) how to select the set R; and (2) how to compute $P(S_1 \rightarrow S_2|R)$. Given a set R, the second problem is easier to solve. Ideally, R should be chosen such that $\sum_{(S_1,S_2)\notin R}(P(S_1 \rightarrow S_2) - P(S_1 \rightarrow S_2|R))$ is maximized. This is an NP hard problem and they provide a greedy solution. Using these solutions, they finally propose a global detection algorithm.

They perform experiments on both synthetic and real datasets. On *Books* dataset of 1263 CS books and 877 online book stores, they found that a source can copy from up to 17 sources and can be copied by up to 9 sources; and there are transitive paths of length up to 9. They found a total of 1553 pairs of sources with copying of which only 465 pairs are found to be direct copying relationships, others being co-copying or transitive copying. They also experiment with weather data for 30 cities with 28 distinct attributes taken every 45 minutes from 18 websites. The *Global* algorithm presents an F1 of 0.79 on weather data. The *Solomon* system[8] demonstrates the effectiveness of their copying detection mechanisms.

4.5 Attribute-Based Consensus for Copying Detection

In most of the previous works, it has been assumed that objects are described by just one attribute, e.g. the price of a stock quote. On the contrary, data sources usually provide complex data, i.e. collections of tuples with many attributes. For example, sources that publish stock quotes always deliver values for price, volume, max and min values, and many other attributes. Different attributes, by their very nature, may exhibit drastically different properties and evidence of dependence. This statement is validated by the observation that state-of-the-art algorithms, when executed on real datasets lead to different conclusions if applied on different attributes published by the same web sources.

Blanco et al. [8] extend the models previously proposed in the literature by working on several attributes at a time to better leverage all the available evidence. Their aims are to compute (i) the probability that the observed properties of an object assume certain values, given a set of observations that refer to that object from a collection of providers; (ii) the accuracy of a provider with respect to each observed property, that is, the probability that a provider provides the correct values of each observed property for a set of objects. They present two models. The first model is developed considering only one property at a time, as they assume that a provider can exhibit different accuracies for different properties.

For the first model, they assume that the values provided by a provider for an object are independent of the values provided for the other objects. Also, they assume that the value provided by a provider for a property of an object is independent of the values provided for the other properties of the same object, and that the accuracy of a provider for a property is independent of the property values. For simplicity, they consider that the values for a property are uniformly distributed among the set of different values provided by sources for that property. They iteratively compute the probability distribution for the values of one property of an object (given the observations of several providers, and the accuracies of the providers with respect to that property) and accuracy of the providers with respect to one property (given the current probability distributions of the observed object properties). The confidence in a particular value of the property of an object is defined as the sum of the accuracy scores of providers that provide that value. Accuracy score of a provider w is defined as $\frac{n.A_w}{1-A_w}$ where $n+1$ is the total number of possible values for the property and A_w is the accuracy of the provider w. Finally the confidence values for the property of the object are normalized such that confidence of all values for a property of an object add up to 1. Further, the accuracy of a source (with respect to a property) which provides values for a property over m objects is computed as the confidence of the value the source provides for the object, averaged across all the objects.

More properties at a time are taken into account in the second model, which considers the copying dependencies too. In this model, the confidence in a particular value of the property of an object is defined as the sum of the accuracy scores of providers that provide that value, weighted by the probability of independent opinion (I_w) of the corresponding provider. This weight factor I_w is computed as $\prod_{w' \neq w}(1 - cP(w \rightarrow w'))$. $P(w \rightarrow w')$ is computed in a way similar to [18] except that they consider values for multiple properties together to infer copying probabilities. Intuitively, the dependence between two providers w_1 and w_2 can be detected by analyzing for which objects they provide the same values, and the overall consensus on those values. Indeed, whenever two providers provide the same value for an object and the provided value is false, this is an evidence that the two providers are copying each other. Much less evidence arises when the two have a common true value for that object: those values could be shared just because both providers are accurate, as well as independent.

They report the results of several experiments on both synthetic and real-world *NASDAQ stock quotes* data to show the effectiveness of the proposed approach. Accuracy obtained using multiple attributes is always better than the accuracy obtained using one attribute because of the effective detection of copiers in the multi-attribute case. In case of data about stock prices, the attributes considered are open price, last price, 52 week high, 52 week low and volume. The NASDAQ[9] is considered as the ground truth. They tried dif-

[8] http://www2.research.att.com/~yifanhu/SourceCopying/

[9] http://www.nasdaq.com

ferent configurations of three attributes at a time and found that some configurations were better than the others. All of them performed better than the 1-attribute configuration. The 5-attribute configuration did not perform as good as some of the 3-attribute configurations. They support this observation by remarking that the quality of the data exposed by an attribute can be more or less useful in the computation of the dependencies: for example, an attribute does not provide information to identify copiers if either all the sources provide the correct values or all the sources provide different values. They report that the execution times increase linearly with the number of attributes involved in the computation.

5. TRUST ANALYSIS USING HOMOGENEOUS NETWORK MODELS

Recently there have been some research where the provider-facts network has been transformed to a homogeneous one before performing trust analysis. Generally, in such setings, one entity type is folded in as edges between the entities of the other type. To build a homogeneous network of providers, one can compute weight between two providers using the facts commonly provided by the two providers. We discuss three such studies in this section. The first one, SourceRank, performs trust analysis on a network of providers. The second one, SSTF, is a semi-supervised trust analyzer which performs analysis on a network of facts. The third one deals with delay tolerant networks and performs analysis on a network of non-cooperative providers.

5.1 SourceRank (on Network of Providers)

SourceRank [3; 2] is a technique to provide a global measure of trust and importance for deep web sources[10], rather than just considering their relevance. Existing methods for source selection in databases use query-similarity-based relevance assessment ,that is, given a query what is the relevance of the results returned by the source. First, relevance may not imply trustworthiness. For example, for the query *The Godfather* many databases in Google Base return copies of the book with unrealistically low prices to attract the user attention. When the user proceeds towards the checkout, these low priced items would turn out to be out of stock, and many times a different item with the same title and cover (e.g., solution manual of the text book). Second, query based relevance does not consider the importance of the results. E.g., if one searches for Libya (in Feb 2011), a source that provides geographical statistics about Libya may not be as important as the one which presents a timeline of the recent protests taking place in the country.

Since a number of sources provide answers to the same query, they conjecture that the agreements between these answers are likely to be helpful in assessing the importance and the trustworthiness of the sources. They build an agreement graph (directed weighted graph) where sources are nodes and edge weight between two sources denotes the degree to which the sources agree. They estimate the agreement between the sources as the agreement of the answers returned by the sources for a set of sample partial-queries. Agreement computation is challenging because first, different web databases represent the same entity syntactically differently. To solve this problem, they combine record linkage models with entity matching techniques for accurate and speedy agreement computation. Further, attribute matchings are weighted against the computed attribute importance. Secondly, most web databases are non-cooperative – i.e., they do not allow access to full data or source statistics. Instead, access is limited to retrieving a set of top-k answers to a simple keyword query. To overcome this, they adapted query based sampling methods used for text databases. Agreement between two sources is computed at three levels: attribute level similarity (using softTF-IDF[11] and Jaro-Winkler[12]), tuple level similarity and agreement between answer sets. While computing the agreement, they measure and try to adjust for any collusion between the sources. They detect the source dependence based on answers to the large answer queries. A large answer query is a very general keyword like *DVD* or *director* with a large set of possible answers. If two sources always return the same answers to these type of queries, they are likely to be dependent (colluding). Quality score of a source (SourceRank), is calculated as the stationary visit probability of a random walk on the agreement graph.

They evaluate SourceRank over two sets of web sources:(i) sets of books and movie databases in *TEL-8 repository* and (ii) books and movie sources from *Google Base*. Their relevance evaluations show that SourceRank improves precision by 22–60% over the *Google Base* and the existing methods. They compare their methodology with three different techniques used for ranking sources for deep web: (i) Coverage based ranking used in relational databases, (ii) CORI [10] ranking (query specific selection based on relevance) used in text databases, and (iii) Google Product search on Google Base. Also, their experiments show that the SourceRank diminishes almost linearly with the source corruption. Factal [4][13] is a system for integrating deep web sources based on SourceRank.

5.2 Semi-Supervised Learning (on Network of Facts)

Traditionally fact finders have been unsupervised learning methods. They assumed that the values provided by more sources are more accurate. However, because false values can be widespread through copying among different sources and out-of-date data often overwhelm up-to-date data, such bootstrapping methods are often ineffective. Including some level of supervision can help guide the iterative fact finding algorithms in the right direction.

Yin et al. [47] propose a semi-supervised approach (Semi-Supervised Truth Finder (SSTF)) that finds true values with the help of ground truth data. Such ground truth data, even in very small amount, can greatly help us identify trustworthy data sources. The approach is based on three principles: (i) facts provided by the same data source should have similar confidence scores, (ii) similar (and therefore mutually supportive) facts should have similar confidence scores and (iii) if two facts are conflicting, they cannot be both true. These three principles are encoded into a graph with facts as nodes. Relationships between facts are encoded as graph edges. If facts f_i and f_j are provided by the same data source, then edge weight w_{ij} is set to a positive value

[10] for surface web, TrustRank [28] is an authority propagation method initialized by a few good seeds selected based on inverse PageRank or high PageRank

[11] SoftTF-IDF measure is similar to the normal TF-IDF measure. But instead of considering only exact same words in two documents to calculate similarity, softTF-IDF also considers occurrences of similar words.

[12] http://en.wikipedia.org/wiki/Jaro%E2%80%93Winkler_distance

[13] http://factal.eas.asu.edu

$(0 < \alpha < 1)$ because if f_i has a high (or low) confidence score, f_j should probably have that as well. If f_i and f_j are on the same subject, then $w_{ij} = sim(f_i, f_j)$. Otherwise $w_{ij} = 0$. Here, $sim(f_i, f_j)$ is a symmetric similarity function which indicates the consistency or conflict between the facts f_i and f_j. $-1 \leq sim(f_i, f_j) \leq 1$.

The aim is to learn a confidence value between -1 and 1 for each of the facts. Truth discovery is equivalent to solving an optimization problem that aims to assign scores to graph nodes that are consistent with the relationships indicated by the graph edges. Specifically, the objective function to be minimized is $E(c) = \frac{1}{2}\sum_{i,j}|w_{ij}|(c_i - s_{ij}c_j)^2$ where s_{ij} is 1, if $w_{ij} \geq 0$ and -1, if $w_{ij} < 0$. They first provide an analytical solution to this optimization problem, although it is very expensive to compute for large scale problems as it involves inversion of a huge matrix. Then they provide an iterative algorithm that converges to the optimal solution. Further, to make the computation more efficient they describe a way to decompose the matrices involved in the computation. If there are n facts, m data sources, l total number of cases of a data source providing a fact, q is the average number of facts on the same subject, and t is the number of iterations, then the algorithm is $O((nq + l)t)$.

Experiments on five real-world datasets (*author lists from AbeBooks.com books*, *directors of American films*, *developers (i.e., studios) of video games*, *governors of U.S. states*, and *presidents of universities*) show that their method achieves higher accuracy than existing approaches, and it can be applied on very huge data sets when implemented with MapReduce. They test their approach on a very large, diverse, and noisy data set (749M facts, 65.7M objects, 89M data sources) containing attribute-values for all kinds of entities extracted from HTML tables of the whole web, with data from Wikipedia serving as ground truth. *SSTF* converges with a steady pace and is not sensitive to changes in parameter values. They vary the size of the training set and conclude that *SSTF* can significantly improve accuracy with a small training set and is significantly faster than many other approaches. They also tested the scalability of their approach on synthetic datasets and observe close to linear increase in time required and memory consumed as the number of facts are increased.

5.3 Fact Finding in Presence of Non-Cooperative Sources

The enhancement of Delay Tolerant Networks (DTNs) communication through the use of group information requires that nodes throughout the network be aware of the membership lists for all groups. Group membership management in DTNs can be broken into four components: group creation, group information propagation, group information consolidation, and group assisted routing. Since group membership is propagated through the network, nodes collect multiple, potentially conflicting, pieces of information about each group. The nodes must consolidate this information into a single local view of each group's membership list. All of the fact finders mentioned above work well in cooperative static scenarios, and assume that all of the providers are always available and can be readily contacted, and hence gather all facts before running fact finding algorithms. Also, they are not designed to omit any information from the final result. But in case of DTNs, a malicious node should not be appended to any membership list.

In [37], the authors present an online algorithm *MembersOnly* that collects group membership information from each node it meets and consolidates it on-the-fly. The algorithm continually refines its decisions as more information becomes available, quickly converging to very accurate decisions about group membership. Nodes propagate group membership lists only for groups of which they are members, to every contact that node makes. Duplication is expected in DTNs, and hence nodes may receive multiple membership lists from the same node. Since membership lists are constantly evolving, the newest list should be used. There can be at most $n \times g$ membership lists stored at a node, where n is the number of nodes in the network and g is the number of groups. During consolidation, *MembersOnly* looks at all recommendations about membership for each potential member of a group and computes a confidence score about that member. This score is based on sigmoid transformation of the positive evidence extracted from all membership lists that claim that the node is part of the group and sigmoid transformation of the negative evidence extracted from all membership lists that do not have that node listed as part of the group. To reconcile these differences of opinion, *MembersOnly* calculates the difference between the strength of the positive evidence and the strength of the negative evidence. If the result is greater than a threshold, the node is placed on the high-confidence list for that group, else not.

Further they analyze two kind of attacks on the system by malicious nodes: addition attack and deletion attack and set parameters of the system to handle them effectively. The goal of the addition attack is to convince as many nodes as possible that the attacking node is part of some or all groups in the network. The goal of the deletion attack is to convince as many nodes as possible that valid members of a group are in fact not members. Their experiments show that *MembersOnly* enables fast propagation and is extremely resistant to attacks, even in the presence of multiple malicious nodes.

6. TRUST METRICS

Most of the fact finders consider truthfulness as the only factor to determine the trustworthiness of information sources. Castelfranchi et al. [11] discuss different factors that comprise trust for multi-agent systems (MAS). They mention different types of beliefs and goals of establishing trust. They deal with different kinds and degrees of trust and relate them with delegation, morality and reputation.

In [23], Fogg et al. define key terms relating to computer credibility, synthesize the literature in this domain, and propose three new conceptual frameworks for better understanding the elements of computer credibility. These frameworks deal with (1) the four types of credibility (presumed credibility, reputed credibility, surface credibility and experienced credibility), (2) the two credibility evaluation errors (gullibility error and incredulity error), and (3) the three strategies for evaluating credibility (binary evaluation, threshold evaluation and spectral evaluation). They also offer two perspectives on what computer users evaluate when assessing credibility. They finally present a set of credibility-related terms.

6.1 Factors Determining Trust

Gil et al. [26] present a large number of factors that affect how users determine trust in content provided by Web information sources. These include (i) Topic: Resources about certain topics may be trusted more than others. (ii) Context and criticality: If the need for information is critical and a true fact needs to be found with high precision, the amount of effort placed in comparing, contrasting, ranking, and dis-

Table 5: Time Complexity Comparison

Algorithm	Time Complexity
TruthFinder [45] (Section 2.1)	$O(IkL)$ where I = number of iterations, L = total number of links between all websites and facts, k = average number of facts about each object.
Sums, Average.Log, Investment, Pooled Investment [38] (Section 2.2)	$O(IL)$ (when implications between facts are not considered) where I = number of iterations, L = total number of links between all websites and facts.
Cosine, 2-Estimates, 3-Estimates [24] (Section 3.1)	$O(IL)$ (when implications between facts are not considered) where I = number of iterations, L = total number of links between all websites and facts.
Probabilistic Assertions [40] (Section 3.2)	$O(IL)$ where I = number of iterations, L = total number of links between all websites and facts.
Basic Cluster-Based Fact Finder, Advanced Cluster-Based Fact Finder [27] (Section 3.3)	$O(C(IkL + I'c^2\|S\|))$ where I = number of iterations, L = total number of links between all websites and facts, k = average number of facts about each object, I' = number of KMeans iterations, C = number of clustering rounds, c = number of clusters, $\|S\|$ = number of sources.
Common-Sense as Logic [38] (Section 3.4)	$O(I(L+T))$ where I = number of iterations, L = total number of links between all websites and facts, T = Time for solving a linear program. Complexity of linear program (LP) solvers depends on n and B where n = number of variables and B is input bits to encode the linear program. Exact complexity depends on the LP solver used.
Source Dependency Detection for Snapshots: *AccuVote* [18] (Section 4.2)	$O(\|O\|.\|S\|^2 Log(\|S\|))$ where $\|O\|$ = number of objects, $\|S\|$ = number of sources.
Time Varying Truth: *LifeSpan* [19] (Section 4.3)	$O(\|O\|\|S\|^2\|T\| + m\|O\|\|S\|\|T\|^2 + u\|S\|)$ where $\|O\|$ = number of objects, $\|S\|$ = number of sources, m = number of values in the domain of an object, u = total number of updates by sources in S.
Copying in Complex Scenarios: *GlobalDetection* [15] (Section 4.4)	$O(m^2\|S\|\|O\| + r\|S\|^2\|O\|)$ where $\|O\|$ = number of objects, $\|S\|$ = number of sources, m = maximum size of the $R(s)$ for any source $s \in S$, r = maximum number of sources related to a source in R, R is the set of copying relationships inferred from direct copying.
SourceRank [3; 2; 4] (Section 5.1)	$O(n^2k^2)$ (for agreement computation)+ time for random walk computation over graph of n nodes, where n = number of sources and top-k result set from each source is used for calculating the agreement graph.
SSTF [46] (Section 5.2)	$O((nk + L)I)$ where I = number of iterations, L = total number of links between all websites and facts, k = average number of facts about each object, n = number of facts.
MembersOnly [37] (Section 5.3)	$O(Tm)$ where m = average number of members publicized for a group, T = number of groups in the network.

proving information is much higher. (iii) Popularity: A more popular source providing a popular fact is more trustworthy. (iv) Perceived Authority: If a source is considered to be authoritative, it is more trustworthy. (v) Direct experience: If a source has been found trustworthy in the past, it can be considered as trustworthy for future transactions too. (vi) Recommendation: If many trustworthy sources recommend an item or a source, it is more trustworthy. (vii) Related Resources: If the related resources are trustworthy, so is the resource under consideration. (viii) Provenance: Trust in the entities responsible for generating a unit of content may transfer trust to the content itself. (ix) User expertise: A user with expertise in the information sought may be able to make better judgements regarding a resource's content, and conclude whether or not it is to be trusted. (x) Bias: A source can be believed to provide correct objective information if it has no bias towards or against the matter being discussed. (xi) Incentive: Information may be more believable if there is motivation for a resource or its associations to provide accurate information. (xii) Limited resources: The absence of alternate resources may result in placing trust in imprecise information. (xiii) Agreement: Even if a resource does not engender much trust in principle, a user may end up trusting it if several other resources concur with its content. (xiv) Specificity: Precise and specific content tends to engender more trust than abstract content that is consistent with true facts. (xv) Likelihood: The probability of content being correct, in light of everything known to the user, may be determined with an understanding of the laws and limits of the domain. (xvi) Age: The time of creation or lifespan of time-dependent information indicates when it is valid. (xvii) Appearance: The design and layout of a site and the grammar and spelling of the content may both be used to judge content accuracy, and whether it should be trusted. (xviii) Deception: Some resources may have deceptive intentions. (xix) Recency: Content, associations, and trust change with time. For example, a resource that had a very bad reputation a few months ago, may improve its behavior and have earned a better reputation.

They also determine (1) what information can be captured in practice from users regarding content trust decisions as they perform Web searches, (2) how a user's information can be complemented by automatically extracted information, (3) how is all the information related to the factors outlined above, and (4) how to use this information to derive content trust. However they do not detail much about how to obtain trust values with respect to each of these factors mentioned above, in a real system.

6.2 Computing Different Trust Factors

Pasternack et al. [39] propose that trustworthiness should not be just based on truthfulness but should instead be computed as a triple of truthfulness, completeness, and bias scores, and argue that these must be calculated relative to the user to be meaningful. Let $P(c)$ be the belief in a claim c. Then, they compute truthfulness of a collection of claims C (e.g. a document) as $T(C) = \frac{\sum_{c \in C} P(c).I(c,P(c))}{\sum_{c \in C} I(c,P(c))}$ where $I(c, P(c))$ is the subjective importance of a claim given its truth.

Next, if a collection C purports to cover a topic t, and A is the collection of all claims in the corpus, completeness with respect to t can be computed as $C(C) = \frac{\sum_{c \in C} P(c).I(c,P(c)).R(c,t)}{\sum_{c \in A} P(c).I(c,P(c)).R(c,t)}$ where $R(c, t)$ is the $[0, 1]$ relevance of a given claim c to the topic t.

Bias results from supporting a favored position (e.g. pro- or anti-) with either untruthful statements or a targeted incompleteness. Let $S(c, s)$ be the $[0, 1]$ degree to which a claim c supports a position $s \in \mathbf{S}$. Let the user's support for each position be $S(s)$ where $\sum_{s \in \mathbf{S}} S(s) = 1$. Then bias of a claim c and collection of claims C can be computed as:

$B(c) = \sum_{s \in \mathbf{S}} |S(s) - S(c, s)|$

$B(C) = \frac{\sum_{s \in \mathbf{S}} \sum_{c \in C} P(c).I(c,P(c)).|S(s)-S(c,s)|}{\sum_{c \in C} P(c).I(c,P(c)).\sum_{s \in \mathbf{S}} S(c,s)}$

Finally, the truthfulness (or completeness or bias) of a provider can be computed as the weighted average of the truthfulness (or completeness or bias) of each of the collections provided by the provider.

They conducted a survey using a news article with the help of nine participants. The survey consisted of one position (bias-based) question with two possible positions, six based on user's assessment of article trustworthiness and 57 questions based on 19 claims made by the article. Users provided ratings from 0 to 10 as response to each question. Finally they conclude based on the small user study (survey) that predicting consistent, semantically well-defined components of trust is a dramatic improvement; an existing trust system might have assigned the newspaper article a perfect trustworthiness score based upon its factual accuracy, completely ignoring the incompleteness and bias that proved obvious to human readers.

7. TRUST ANALYSIS USING LOGIC

Logic propositions can be used to represent claims in databases. Logical deduction can be used to detect inconsistencies. In this section, we will discuss how to represent and detect temporal inconsistencies using temporal logic. Also, logic-based argumentation can be combined with weighted trust networks between providers, and trust analysis can be performed on such an augmented network. We will also discuss how to generate such a network and use it for trust analysis in this section.

7.1 Expressing Temporal Inconsistency Using Logic

Given a temporal database (knowledge base), one often needs to identify points in time when the database is temporally inconsistent. For example, consider an airline database. Airlines are often engaged in a practice where more tickets are sold for a flight than there are seats on the flight, thus resulting in an inconsistency with the safety regulations. This is based on the belief by the airlines that there might be enough no shows by the boarding time to the point that the inconsistency will resolve itself. In the event that this overbooking inconsistency does not resolve itself during the boarding time, it should trigger a number of possible airline actions ranging from upgrading yet-to-be-seated passengers, arranging for a later flight with rewards, to making alternative travel. Thus, inconsistency detection is important.

Temporal inconsistency detection can be done using temporal logic. Once conflicting literals are identified in a knowledge base (KB), we need to ascertain whether their time intervals are coinciding to determine if they constitute temporal inconsistency. If two literals (or a literal and its related logic expression) are temporally inconsistent, we want to identify the coinciding or conflict interval between the two. Once we obtain the conflict intervals for all temporal inconsistent cases in a KB, we are in a position to characterize the temporal properties for the entire KB. To this end, Zhang et al. [51] propose a systematic approach to identify conflicting intervals for temporally inconsistent propositions. They focus on detecting time points when data is inconsistent and do not suggest any conflict resolution strategies.

To investigate the temporal properties of inconsistent information, they resort to a temporal logic to formally establish temporal relationships between time periods of propositions and to reason about the presence or absence of temporal inconsistency. They adopt James Allen's interval temporal logic, which is based on characterizing actions and events in terms of interval relationships. Consider two time intervals i and j. They define 13 interval relationships: Meets (i, j), Before (i, j), Overlaps (i, j), Equals (i, j), Starts (i, j), During (i, j), Finishes (i, j), MetBy (j, i), After (j, i), OverlappedBy (j, i), StartedBy (j, i), Contains (j, i), and FinishedBy (j, i).

A predicate in the temporal logic consists of an atomic formula $P(t_1, \ldots, t_n)$, a time point T and a duration D. Given two literals L_i and L_j, or a literal L_i and its related logical expression L, temporal inconsistency between L_i and L_j occurs when L_i and L_j or L_i and L are logically conflicting and their time intervals are temporally coinciding. There are three different types of inconsistencies between L_i and L_j, depending on the relationship between the intervals corresponding to them. For the congruent case, temporal inconsistency for both literals will be persistent. For the subsuming case, the literal (or the expression) whose interval is subsumed by that of the other will have a persistent temporal inconsistency. In the overlapping case, there is a chance for both literals (or the literal and its related expression) to be still temporally consistent. They provide an algorithm for temporal inconsistency detection between two literals in temporal logic, using the interval relationship definitions.

7.2 Trust Using Argumentation

Argumentation tracks the origin of data used in reasoning. Tang et al. [42] develop a general system of argumentation that can represent trust information, and be used in combination with a trust network. This system makes it possible for an agent to construct arguments where belief in the conclusions is related to the degree of trust in the agents who supplied the information used as premises in the arguments. A trust network is a graph of agents where a directed link denotes a trust relationship from one agent to another. An agent-centric trust graph is a trust network with a single root. Basically, it is a subgraph of the original trust network, which contains all edges that originate at an agent a and nodes on which these edges are incident. An agent a has a knowledge base ($\Sigma = P \cup \Delta$) which consists of a set of premises (P) and a set of inference rules (Δ). The inference rules link some set of premises to a conclusion using a process called inference. The inference process can be depicted by a rule (or proof) network. An argument A from a knowledge base $\Sigma = P \cup \Delta$ is a pair $\langle h, H \rangle$ where H is a proof network for h, and h is the only leaf (conclusion node) of H. They further define different ways in which two arguments can attack (conflict with) each other: rebut (conclusions conflict), premise-undercut (premise of one conflicts with conclusion of the other), intermediate-undercut (an intermediate conclusion of one conflicts with the conclusion of the other), or inference-undercut (conclusion of one conflicts with an inference rule of the other). A trust-extended proof network is a combination of the trust network and the rule networks. It consists of agents, premises, inference rules and conclusion nodes. It relates the premises of an argument to their source. Such a network therefore captures the reasoning of the agent at the root of the trust network, including which pieces of information it has used from the agents it trusts. In the same way that an argument is relative to a knowledge base, so a trust-extended argument is relative to a set of agents, and, in particular, to the set of knowledge bases of those agents. Given a trust-extended argument, the only agent that is sanctioned to infer the conclusion of the argument is the one at the root of the trust graph. Finally, a trust-extended argument graph is defined as a set of trust-extended arguments with the attack relationships between the arguments denoted by labeled links.

An example of such a graph is shown in Figure 2 taken from [42]. John trusts Mary who in turn trusts Dave and

Jane. John wants to decide whether to watch a film or not. John can reach two conflicting conclusions depending on whom he believes (Jane or Dave). One can use the weights on different edges on the network from 'John' node to the different conclusion nodes and check which argument wins.

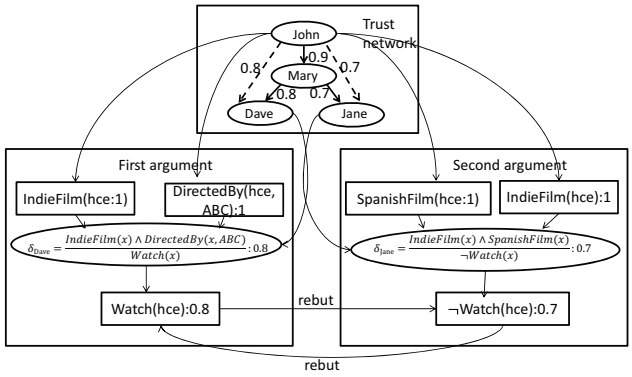

Figure 2: A trust-extended argumentation graph capturing John's view of the film

8. APPLICATIONS OF TRUST ANALYSIS MODELS

Trust analysis models have been used for a variety of real-world applications. The iterative philosophy of establishing trustworthiness of providers and confidence of facts has been especially exploited to rank different kinds of entities. Trust analyzer is a major part of Provenance-based systems. Trust analysis can be used to rank importance of websites and the confidence of values returned in the top search engine results. One can use the same philosophy to establish popularity of articles on a website and the influence of the website itself. One can use trust analysis to rank most trustworthy news publishers for news stories of particular categories. In sensor networks, trust can be used to establish credibility of sensor data and sensors themselves. Trust analysis has also been used in data fusion and to ensure quality of systems that promote user generated content. We discuss such applications in this section.

8.1 Provenance-Based Access Control Systems

It is essential to ensure data integrity so that effective decisions can be made based on available data. A group from Purdue University [7; 13; 14] presents an architecture for a comprehensive integrity control system based on data validation and metadata management. An integrity controller consists of these components: (i) Integrity policy supplier (provides information related to integrity policies and metadata values to integrity policy repository and integrity metadata repository), (ii) Integrity policy repository and integrity metadata repository (maintain and manage various information about policies, users, and data for the integrity validator), and (iii) Integrity validator (carries out integrity validation functions for access requests and data objects in database). To specify integrity control policies, they design metadata specification language and integrity policy specification language. A data validation policy specified using the integrity policy specification language specifies what events can trigger which validation procedures on different items and what are the actions to be taken depending on the outcome of the validation procedures.

For trust evaluation of data provenance, they present a model consisting of source nodes, intermediate nodes and end users. Credibility of data then depends on multiple factors: Where did the data come from? How trustworthy is the original data source? Who handled the data? Are the data managers trustworthy? Based on the confidence scores, data users can determine whether they want to directly use the received information or need to further verify the information. Credibility of a data item depends on: (i) Data similarity: Confidence score of an item tends to be higher when an item is in a larger cluster, i.e., there are more similar items. (ii) Data conflict: Data conflict has a negative impact on the credibility of items. To determine if two data items conflict, the users need to mention some prior knowledge. Data conflict between two items can be computed in terms of the data conflict between their cluster representatives. (iii) Path similarity: If the path similarity among a group of similar items is low, confidence scores of corresponding items are increased; otherwise, a negative effect is added to the scores. Given two data items, one can obtain two paths P_1 and P_2 each denoting a list of sources and intermediate agents through which the respective data items have been propagated. Path similarity between paths P_1 and P_2 is calculated as $\frac{max\{|P_1|,|P_2|\}-Idist}{max\{|P_1|,|P_2|\}}$ where $Idist$ is the edit distance between the sequences representing the two paths and $max\{|P_1|, |P_2|\}$ is the maximum of the lengths of the two paths. (iv) Data deduction: Trustworthiness of the deduced data depends on the confidence of input data and of the intermediate agents that deduce (merge or process) the data. Initially, each source provider and intermediate agent is assigned an initial trust score by querying the information that the end users already knew. The initial confidence of each data item and knowledge item is then set to the trustworthiness of its source providers and intermediate agent. At each iteration, the confidence of the data is computed based on the combined effects of the aforementioned four aspects, and the trustworthiness of the data provider is recomputed by using the confidence scores of the data it provides. When a stable stage is reached, that is, when the changes of trust scores are negligible, the trust computation process stops.

Further, they observe that accurate data is not always necessary. For example, if a user just wants to compute some statistical summary of the data, data with low confidence may be sufficient since providing data with high confidence is usually very expensive. On the other hand, if the user has to make a critical decision, data with high confidence may be required. Therefore, they propose a data provenance-aware access control policy, referred to as confidence policy. As a complement to the integrity control mechanism that applies to the database before any operation, the confidence policy restricts the access to the query results based on (i) provenance confidence levels of the query results, (ii) minimum confidence level expected by the user, and (iii) purpose for which access is required. Users can specify a minimum percentage m of results that they want to receive. In such a case, the system estimates cost of verifying the fact and increasing the confidence of the facts such that $m\%$ facts can be returned. If user agrees to bear the cost, the system returns the desired number of query results.

8.2 Ranking Web Results

Wu et al. [44] proposed methods to aggregate query results (for a query q) from different sources in order to save users the hassle of individually checking query-related web sites

to corroborate answers. To return the best aggregated answers to the users, their techniques consider the number, importance, and similarity of the web sources reporting each answer, as well as the importance of the answer within the source. Once the answers are identified (by information extraction from top search results), similar answers are combined and scored based on their importance in the search engine results, as measured by different parameters such as the frequency of the answer in the search result (more answers on a single result web page decreases the score of an individual answer, but the same answer on multiple web pages increases the score of the individual answer), the prominence of the answer within the search result web pages (position of answer in web page relative to the position of the query, the closer the better), the placement of pages containing the answer in the search result, and the duplication of information between pages. To retrieve answers, they process only the top-k pages from search results. They process web pages in the search result order, and stop retrieving new pages when the score of newly discovered answer would not be high enough to impact the overall corroborative answer score ranking. Note that the importance of the page (as they define) decreases very quickly as its rank in the search results increases, hence k is typically small. On a few queries, they experiment with results from MSN Live Search[14] and show that their corroboration approach is good at identifying answers that are of interest to the user, as many user clicks correlate with top-1 corroborated answers.

A similar method to find answers from web search results is proposed in [21]. Intuitively, if a value is provided by many accurate web sites, it is very likely that the value is correct. They use the iterative fact finder methodology to compute accuracy of web sites and the confidence of values. Their model handles the top-k page summaries returned by search engines, identifies the related values, analyzes the values and finds the correct answer automatically. The model considers the accuracy of web site, the duplication of web site, the similarity of values; counts the vote for each value and gets the most likely correct answers by iterative computation.

8.3 Website Ranking

Popularity&InfluenceCalculator (PIC) [25] is an algorithm to get the most popular web pages and influent websites under certain keywords. If a certain piece of content appears on more web pages, it will have higher importance. If a web page contains more important contents, it will have more popularity. The popularity of a page can be divided into two parts, influence of the websites it is on and the effects of other pages. The influence of a website can be divided into two parts, its own significance and the popularity of its pages. The own significance of a website can be calculated through several factors, such as the daily visit number, its sensibility to specific domain of news, and so on.

In *PIC* algorithm, the influence of websites is initialized as their own significance. Then, those values are used to calculate the popularity of web pages. Meanwhile, the influences among all the pages is also taken into consideration. Afterwards, the websites' influence is computed again through their own significance and their pages. The website which has more popular pages will have more influence, while the page which exists on more influent websites will have more popularity. The calculation continues until the convergence of the importance of both pages and websites. Their empirical results show that the *PIC* algorithm can rank the pages in famous websites and pages with descriptive facts higher.

[14] now, http://www.bing.com/

8.4 Sensor Networks

Participatory sensing, where users proactively document and share their observations, has received significant attention in recent years as a paradigm for crowd-sourcing observation tasks. *Apollo* [31] is a fact finder for sensor network data. It is the first fact finder designed and implemented specifically for participatory sensing. *Apollo* uses Twitter as the underlying engine for sharing participatory sensing data. Data-type-dependent modules first convert source data (a set of (source, observation) tuples) into a common representation of sources and claims. Clustering is performed on input tuples (tweets and related metadata) first, by similarity of their observations, to generate a smaller number of claims for scalability. Their credibility is then assessed in an iterative fashion (using a fact finder algorithm), together with the credibility of each source.

8.5 Quality Inference on User-Generated Content

User-generated content has been a great source of information (Delicious, Flickr, Twitter, Wikipedia) on the World Wide Web. Conventional, centralized confirmation of credibility is infeasible at the Internet scale and so making use of the annotators themselves to evaluate each item is essential. However, users usually differ in opinions about the same item, and the existence of bias, variance and maliciousness makes the problem of aggregating opinions more difficult. Addressing this problem, Liao et al. [33] propose the use of an Annotator-Article model with an iterative algorithm, called *ScoreFinder*, for inferring credibility (intrinsic quality) by ranking shared items. The model is a bipartite weighted graph of annotators and articles. An annotation is a link from an annotator to an article and consists of a numeric score and a confidence measurement of annotator in giving such a score. Each article is associated with one or more topics, independently, to a varying degree. In order to reduce the influence from a variety of error sources, they identify reliable users (using the accuracy of their scores) on each topic, and adaptively aggregate scores from them. An iterative algorithm refines both the evaluation of the intrinsic quality of articles and the estimates of expertise of annotators over several iterations. Moreover, they transform the users' input to remove errors/anomalies, by identifying patterns of misbehaviour learned from a real data set.

8.6 Data Fusion

Modern data management applications, such as setting up Web portals, managing enterprise data, managing community data, and sharing scientific data, often require integrating available data sources and providing a uniform interface for users to access data from different sources. Different sources can provide conflicting data. Conflicts can arise because of incomplete data, erroneous data, and out-of-date data. Dong et al. [20] present a classification about various conflict handling strategies. They categorize conflict handling strategies into conflict ignoring strategies (perform no resolution), conflict avoiding strategies (aware of conflicts, but perform no individual resolution for conflicts), and conflict resolving strategies (perform individual fusion decisions for each conflict). Then they present a set of advanced techniques for conflict resolution that consider accuracy of sources like [45; 18], freshness of sources like [19] and de-

pendencies between sources like [6; 18]. Finally they survey data fusion techniques in existing data integration systems.

8.7 News Finding

Given a collection of news articles, News Website Evaluation is primarily concerned with evaluating the importance of the websites with respect to specific news topics. For a news topic, there are a set of news articles published by different websites. Miao et al [35] propose *Topic-oriented Website Evaluation Model (TWEM)*. *TWEM* considers interdependency between websites and news articles, as well as mutual support among news articles. Articles are considered as authorities and websites as hubs. The inherent popularity of websites (based on Alexa Rank[15]) is also considered when they infer their final importance scores. Then, in an attempt to handle copycats, they incorporate an article merging operation into *TWEM* and present the *merge-TWEM* model. If the similarity between two articles exceeds a predefined threshold, they merge them into a single super-article. Influence of one super-article on another is computed as an average similarity between articles belonging to the two super-articles relative to its similarity with other super-articles.

In [49], the authors propose Corroboration Trust (i.e. the problem of finding credible news events by seeking more than one source to verify information on a given topic). They design an evidence-based corroboration trust algorithm called *TrustNewsFinder*, which utilizes the relationships between news articles and related evidence information (person, location, time and keywords about the news). A news article is trustworthy if it provides many pieces of trustworthy evidence, and a piece of evidence is likely to be true if it is provided by many trustworthy news articles. The evidence extraction is done using named entity recognition and articles about the same news story are grouped together using topic detection and tracking (TDT) techniques. Paraphrasing is detected using dependency tree analysis over the set of discovered named entities. Once the evidence has been extracted and articles have been clustered properly, an iterative fact finder algorithm computes trustworthiness of a news article and confidence of the evidence by expressing them in terms of each other. An article's trustworthiness is computed using different metrics like currency, availability, information-to-noise ratio, authority, popularity and cohesiveness.

8.8 Information Credibility on Twitter

Most of the messages posted on Twitter are truthful, but the service is also used to spread misinformation and false rumors, often unintentionally. Castillo et al. [12] analyze the information credibility of news propagated through Twitter. They propose automatic methods for assessing the credibility of a given set of tweets. Specifically, they analyze microblog postings related to "trending" topics (obtained using Twitter Monitor), and classify them as credible or not credible, based on features extracted from them. They use features from users' posting and re-posting ("re-tweeting") behavior, from the text of the posts, and from citations to external sources. Their main hypothesis is that the level of credibility of information disseminated through social media can be estimated automatically. This can be done using following factors: the reactions that certain topics generate and the emotion conveyed by users discussing the topic, the level of certainty of users propagating the information, the external sources cited, and characteristics of the users that propagate the information. They propose four types of features depending on their scope: message-based features, user-based features, topic-based features, and propagation-based features.

Sentiment based features are very relevant for the credibility prediction task. In general tweets which exhibit sentiment terms are more related to non-credible information. In particular this is very related to the fraction of tweets with positive sentiments, as opposed to negative sentiments, which tend to be more related to credible information. Tweets which exhibit question marks or smiling emoticons tend also to be more related to non-credible information. Something similar occurs when a significant fraction of tweets mention a user. On the other hand, tweets having many re-tweets on one level of the propagation tree, are considered more credible. Presence of URLs, tweet sentiment, presence of exclamation marks, number of tweets by user, number of friends of user, maximum level size of an RT tree were found to be the best features.

They evaluate their methods using a significant number of human assessments (Mechanical Turk[16]) about the credibility of items on a recent sample of Twitter postings. A supervised classifier is trained to determine if a set of tweets describes a newsworthy event. Their results shows that there are measurable differences in the way messages propagate, that can be used to classify them automatically as credible or not credible, with precision and recall in the range of 70% to 80%.

9. CONCLUSIONS

In this survey paper, we reviewed the work in the data mining community on performing heterogeneous network-based trust analysis based on the data provided by multiple information sources for different objects. We briefly discussed the recent efforts in the area of fact finding and trustworthiness-based ranking of information sources. We presented a classification of the approaches based on the network design used and the sub-problems solved. We discuss various aspects of trust including the basic fact finder models, their extensions, source dependency models, logic based models, homogeneous trust network models, and semi-supervised learning models. For each of the fact finding problems, we introduced the problem, described the methodology in short and also discussed interesting insights from experiments. Finally, we presented some applications where trust analysis has played a major role.

The area of trust analysis has recently gained attention of the data mining community. We believe that a lot of work needs to be done for trust analysis algorithms to be applicable to real world situations. Future work in the area could focus on multiple research directions. (1) More Complex Models: Currently most of the work deals with simple situations where there are information providers and claims provided by information providers. But in real world situations, there are a lot of entities which interact with each other. More complex models may be designed which take into consideration a larger set of entities and their interactions. (2) Time and Space Complexity: Current algorithms are heavy on time and memory and are well suited for one time computations. But, to handle stream of facts, more work needs to be done. (3) Improvements for Source Dependency Detection Mechanisms: Source dependency detection

[15] http://www.alexa.com/

[16] http://www.mturk.com

methods consider just a few factors. There could be other factors like "I would like to copy from a provider I consider more trustworthy". There are a lot of such intuitions which advanced source dependency detection algorithms can exploit. (4) Metrics: Trust may not be limited to just truthfulness. In practice, one uses a lot of factors, when believing a piece of news. More complicated fact finders may include more factors, some of which may be domain dependent or user-specified. (5) Novel Applications of Fact Finders: Some applications have certainly been built that exploit fact finding algorithms. But, there does not seem a direct way of applying any fact finding algorithm to verify facts on Twitter. Recently, a system *Facto* [48] has been introduced which builds a clean database using structured data from tables on the web. (6) Handling Stream Data: Most of the social media appears as a text stream. Fact finding algorithms need to be developed to be able to deal with stream data. (7) Models for Supervised or Semi-Supervised Fact Finding: Most of the work in the field has been done in an unsupervised setting. In presence of some labeled data in the form of a combination of source dependencies and true claims, source dependency detection algorithms may need to be tweaked to achieve better accuracy.

10. ACKNOWLEDGEMENTS

The work was supported in part by NSF IIS-09-05215, U.S. Air Force Office of Scientific Research MURI award FA9550-08-1-0265 and the U.S. Army Research Laboratory under Cooperative Agreement Number W911NF-09-2-0053 (NS-CTA). The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of the Army Research Laboratory or the U.S. Government. The U.S. Government is authorized to reproduce and distribute reprints for Government purposes notwithstanding any copyright notation here on.
Thanks to the anonymous reviewers for their insightful comments.

11. REFERENCES

[1] B. Thomas Adler and Luca de Alfaro. A content-driven reputation system for the wikipedia. In *WWW*, pages 261–270, 2007.

[2] Raju Balakrishnan. Source rank: Relevance and trust assessment for deep web sources based on inter-source agreement. In *WWW*, 2011.

[3] Raju Balakrishnan and Subbarao Kambhampati. Sourcerank: relevance and trust assessment for deep web sources based on inter-source agreement. In *WWW*, pages 1055–1056, 2010.

[4] Raju Balakrishnan and Subbarao Kambhampati. Factal: Integrating deep web based on trust and relevance. In *WWW*, 2011.

[5] Omar Benjelloun, Anish Das Sarma, Alon Halevy, Martin Theobald, and Jennifer Widom. Databases with uncertainty and lineage. *The VLDB Journal*, 17:243–264, Mar 2008.

[6] Laure Berti-Equille, Anish Das Sarma, Xin Dong, Amélie Marian, and Divesh Srivastava. Sailing the information ocean with awareness of currents: Discovery and application of source dependence. In *CIDR*, 2009.

[7] Elisa Bertino, Chenyun Dai, Hyo-Sang Lim, and Dan Lin. High-assurance integrity techniques for databases. In *BNCOD*, pages 244–256, 2008.

[8] Lorenzo Blanco, Valter Crescenzi, Paolo Merialdo, and Paolo Papotti. Probabilistic models to reconcile complex data from inaccurate data sources. In *CAiSE*, pages 83–97, 2010.

[9] Sergey Brin and Lawrence Page. The anatomy of a large-scale hypertextual web search engine. *Computer Networks and ISDN Systems*, 30:107–117, Apr 1998.

[10] Jamie Callan and Margaret Connell. Query-based sampling of text databases. *ACM Transactions on Information Systems*, 19:97–130, Apr 2001.

[11] C. Castelfranchi and R. Falcone. Principles of trust for mas: Cognitive anatomy, social importance, and quantification. In *International Conference on Multi Agent Systems*, pages 72–79, 1998.

[12] Carlos Castillo, Marcelo Mendoza, and Barbara Poblete. Information credibility on twitter. In *WWW*, pages 675–684, 2011.

[13] Chenyun Dai, Dan Lin, Elisa Bertino, and Murat Kantarcioglu. An approach to evaluate data trustworthiness based on data provenance. In *Secure Data Management*, pages 82–98, 2008.

[14] Chenyun Dai, Dan Lin, Elisa Bertino, and Murat Kantarcioglu. Trust evaluation of data provenance. Technical Report CERIAS TR 2001-147, Purdue University, 2008.

[15] Xin Dong, Laure Berti-Equille, Yifan Hu, and Divesh Srivastava. Global detection of complex copying relationships between sources. *PVLDB*, 3(1):1358–1369, 2010.

[16] Xin Dong, Laure Berti-Equille, Yifan Hu, and Divesh Srivastava. Solomon: Seeking the truth via copying detection. *PVLDB*, 3(2):1617–1620, 2010.

[17] Xin Luna Dong. Presentation for [6], sailing the information ocean with awareness of currents, 2009.

[18] Xin Luna Dong, Laure Berti-Equille, and Divesh Srivastava. Integrating conflicting data: The role of source dependence. *PVLDB*, 2(1):550–561, 2009.

[19] Xin Luna Dong, Laure Berti-Equille, and Divesh Srivastava. Truth discovery and copying detection in a dynamic world. *PVLDB*, 2(1):562–573, 2009.

[20] Xin Luna Dong and Felix Naumann. Data fusion - resolving data conflicts for integration. *PVLDB*, 2(2):1654–1655, 2009.

[21] Huitao Dou, Qingzhong Li, and Yongxin Zhang. Find answers from web search results. *Web Information Systems and Applications Conference*, 0:95–98, 2010.

[22] Allison Enright. Consumers trust information found online less than offline messages. *Internet Retailer*, Aug 2010.

[23] B. J. Fogg and Hsiang Tseng. The elements of computer credibility. In *Proceedings of the SIGCHI conference on Human factors in computing systems: the CHI is the limit*, pages 80–87, 1999.

[24] Alban Galland, Serge Abiteboul, Amélie Marian, and Pierre Senellart. Corroborating information from disagreeing views. In *WSDM*, pages 131–140, 2010.

[25] Song Gao, Yajie Miao, Liu Yang, and Chunping Li. Topic-based computing model for web page popularity and website influence. In *Australasian Conference on Artificial Intelligence*, pages 210–219, 2009.

[26] Yolanda Gil and Donovan Artz. Towards content trust of web resources. *Web Semantics*, 5:227–239, Dec 2007.

[27] Manish Gupta, Yizhou Sun, and Jiawei Han. Trust analysis with clustering. In *WWW*, 2011.

[28] Zoltán Gyöngyi, Hector Garcia-Molina, and Jan Pedersen. Combating web spam with trustrank. In *International Conference on Very large data bases (VLDB)*, pages 576–587, 2004.

[29] Sepandar D. Kamvar, Mario T. Schlosser, and Hector Garcia-Molina. The eigentrust algorithm for reputation management in p2p networks. In *International Conference on World Wide Web*, pages 640–651, 2003.

[30] Jon M. Kleinberg. Authoritative sources in a hyperlinked environment. *J. ACM*, 46:604–632, Sep 1999.

[31] H. Khac Le, J. Pasternack, H. Ahmadi, M. Gupta, Y. Sun, T. Abdelzaher, J. Han, D. Roth, B. Szymanski, and S. Adali. Apollo: Towards factfinding in participatory sensing. In *ISPN*, 2011.

[32] Raph Levien. Attack resistant trust metrics. *Draft of Ph.D thesis, U. C. Berkeley*, 2003.

[33] Yang Liao, Aaron Harwood, and Kotagiri Ramamohanarao. Scorefinder: A method for collaborative quality inference on user-generated content. In *ICDE*, pages 345–348, 2010.

[34] Xin Liu, Anwitaman Datta, Krzysztof Rzadca, and Ee-Peng Lim. Stereotrust: a group based personalized trust model. In *Proceeding of the 18th ACM conference on Information and knowledge management (CIKM)*, pages 7–16, 2009.

[35] Yajie Miao, Chunping Li, Liu Yang, Lili Zhao, and Ming Gu. Evaluating importance of websites on news topics. In *PRICAI*, pages 182–193, 2010.

[36] Mohammad Momani and Subhash Challa. Survey of trust models in different network domains. *CoRR*, abs/1010.0168, 2010.

[37] Samuel C. Nelson and Robin Kravets. For members only: local and robust group management in dtns. In *ACM workshop on Challenged networks (CHANTS)*, pages 5–12, 2010.

[38] J. Pasternack and D. Roth. Knowing what to believe (when you already know something). In *International Conference on Computational Linguistics (COLING)*, Aug 2010.

[39] Jeff Pasternack and Dan Roth. Comprehensive trust metrics for information networks. In *Army Science Conference*, Dec 2010.

[40] Jeff Pasternack and Dan Roth. Generalized fact-finding. In *WWW*, 2011.

[41] Yogesh L. Simmhan, Beth Plale, and Dennis Gannon. A survey of data provenance in e-science. *SIGMOD Rec.*, 34:31–36, Sep 2005.

[42] Yuqing Tang, Kai Cai, Elizabeth Sklar, Peter McBurney, and Simon Parsons. A system of argumentation for reasoning about trust. In *EUMAS*, 2010.

[43] Jose Weise. Public key infrastructure overview. *SunPS Global Security Practice*, Aug 2001.

[44] Minji Wu and Amélie Marian. Corroborating answers from multiple web sources. In *WebDB*, 2007.

[45] Xiaoxin Yin, Jiawei Han, and Philip S. Yu. Truth discovery with multiple conflicting information providers on the web. *IEEE Trans. Knowl. Data Eng.*, 20(6):796–808, 2008.

[46] Xiaoxin Yin and Wenzhao Tan. Semi-supervised truth discovery. In *WWW*, 2011.

[47] Xiaoxin Yin, Wenzhao Tan, Xiao Li, and Yi-Chin Tu. Automatic extraction of clickable structured web contents for name entity queries. In *WWW*, pages 991–1000, 2010.

[48] Xiaoxin Yin, Wenzhao Tan, and Chao Liu. Facto: a fact lookup engine based on web tables. In *WWW*, pages 507–516, 2011.

[49] Guosun Zeng and Wei Wang. An evidence-based iterative content trust algorithm for the credibility of online news. *Concurrency and Computation: Practice and Experience*, 21(15):1857–1881, 2009.

[50] Honglei Zeng, Maher A. Alhossaini, Li Ding, Richard Fikes, and Deborah L. McGuinness. Computing trust from revision history. In *PST*, page 8, 2006.

[51] Du Zhang. On temporal properties of knowledge base inconsistency. *Transactions on Computational Science*, 5:20–37, 2009.

[52] Philip R. Zimmermann. *The official PGP user's guide*. MIT Press, Cambridge, MA, USA, 1995.