A Non-Repetitive Semantic Approach for Truth Discovery Over the Web

Said Sryheni (saeedsryhini@gmail.com)
Damascus University
Mohammad Saeed AbouTrab
Damascus University

Research Article

Keywords: Truth Discovery, Object-Conflict, Semantic, Search Engines

Posted Date: June 17th, 2022

DOI: https://doi.org/10.21203/rs.3.rs-1750512/v1

A Non-Repetitive Semantic Approach for Truth Discovery Over the Web

Said Sryheni[1] and Mohammad Saeed AbouTrab[1]

[1]Department of Software Engineering and Information Systems, Damascus University.

Contributing authors: saeedsryhini@gmail.com; m.s.aboutrab@damascusuniversity.edu.sy;

Abstract

As the number of web pages on the internet grows, so do the conflicts between data gathered by search engines from different sources, primarily websites. Search engines require a conflict-resolving system that can quantify the truth of obtained data and assess the reliability of the information offered by sources. This paper presents a non-repetitive semantic approach for resolving data conflicts on search engines by including correctness elements into indexed data and estimating the trustworthiness of websites from which data are retrieved. Using the same dataset, our approach is validated in comparison with another approach based on several factors such as execution speed, information veracity efficiency and the error rate of sources' trust. The approach shows promising results and outperforming the other approach. The execution time to process the dataset is about half an hour for our approach against four hours for the other one. The F1 measure of information truth calculations and the area under the ROC curve are around 88 and 94 percent, respectively. The source trustworthiness error is around 0.09 for our approach where is about 0.17 for the other one.

Keywords: Truth Discovery, Object-Conflict, Semantic, Search Engines

1 Introduction

The majority of people nowadays acquire their information via the World Wide Web. Web sites are currently given weight by search engines based on their predicted importance. The page rank algorithm, for example, is used by Google to estimate the significance of a web page based on the value of inbound and outbound links to that page [**?**]. Most websites strive to enrich their content as quickly as possible. It becomes increasingly difficult to discriminate between correct and fraudulent information resulted, in many situations, from copying data from other unreliable websites without verifying its accuracy [**?**]. It's critical that search engines assess the trustworthiness of websites by assigning a truth degree to chunks of provided information prior to prioritizing them in crawling and indexing procedures. To achieve that, the reliability of the information sources should be assessed and analyzed. This is not an easy to achieve in search engines due to the added complexity of the rerunning algorithm calculating the truth factors each time.

Aiming to bridge this gap, this paper develops a new approach for truth discovery over the web. The new approach calculates information truth and estimates the trustworthiness of websites. Minimizing the operation complexity, the developed approach can be deployed within a search engine to improve search results by favoring correct information over fraudulent ones and enhancing the crawler's focus on trustworthy websites.

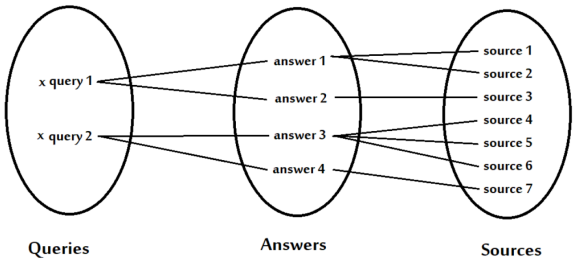

Fig. 1: Representing object conflicts in the philosophical model

The rest of this work is arranged as follows. The related work is presented in section 2. The proposed approach is presented in Section 3 where the theoretical model is introduced and in Section 4 where the practical model including the algorithms for truth finding is discussed. Section 5 sets up the experimental study to evaluate the proposed approach. Finally, section 6 concludes the paper with a summary and future work.

2 Related Work

For resolving object conflicts, a variety of approaches were provided. These approaches concentrate on the issue of object conflicts without addressing how data is retrieved and delivered to the system. They can be classified into many categories based on the algorithms they use. The philosophical approach is the primary one, in which the data is divided into several questions, each with multiple answers. Furthermore, each answer is provided by one or more sources [**?**]. Conflicts are resolved by assigning a truth value to each answer and a level of confidence in the accuracy of data each source provides. The model for the philosophical approach is shown in Figure **??**.

The philosophical approach provides a solid foundation for defining the problem of object conflict. However, it cannot provide an arithmetic method for identifying the correct data. Instead, it goes out into the real world to see if the event associated with each piece of data occurred [**?**]. On the other hand, other approaches use this model to enhance their algorithms further. Since information accuracy is tied to the sources behind it, and source trust is calculated based on its accuracy in providing correct information, these approaches use a repeated process to calculate both values. Either information correctness or source trust is fixed in each cycle, while the other is calculated. This method stops when the values do not change anymore, at which point it is said to have reached convergence [**?**].

Voting Approach: The problem is formulated as a voting procedure. First, information chunks are treated as candidates, with the voters serving as the sources. Each source casts a vote for the information it believes in. As a result, each piece of data is assigned a truth degree based on the sources that voted for it. The process is then inverted, with sources becoming candidates and each piece of information casting a vote for the sources that gave it. After this phase, each source is assigned a trust level based on the accuracy of its data [**?**]. This procedure is repeated until a convergence criterion is met.

This was one of the first approaches to be introduced. Its flaws stem from the fact that it uses the majority voting algorithm, which does not account for the weight of each voter. Instead, the fact offered by most sources is assumed to be correct, regardless of sources' accuracy [**?**]. Furthermore, it does not handle the potential of many valid answers to a question, as the majority voting algorithm only selects one winner[**?**, **?**].

Probabilistic Approach: These approaches use different variation of probabilistic models [**?**]. The truth finder algorithm introduced the probabilistic approach in 2008 [**?**]. This algorithm used the similar principle of employing a repetitive procedure to calculate information veracity and source trust. On the other hand, it could not handle many accurate answers to a question. Galland et al. presented three object conflict resolution algorithms in 2010 [**?**]. The first, called (Cosine), was based on the well-known cosine formula used in information retrieval to quantify questions' similarity and give relatively similar correctness values for similar responses. The information accuracy and source trust were measured using the 2-Estimates algorithm. The 3-Estimates approach, on the other hand, took into account the problem of measuring the truth degree difficulty for specific questions since 2-Estimates can attain a local optimum when all sources are given a similar trust of 0.5. Both algorithms used a repetitive approach until reaching convergence.

In 2012, two algorithms further used the probabilistic model. Zhao et al. used the Bayesian model to measure information correctness [**?**]. Precision, accuracy, recall, and specificity were all measured for each source. On the other hand, Wang et al. modeled each information correctness and source trust degree as random variables inside the Bayesian model [**?**]. The goal was to achieve the maximum likelihood estimation. Pasternack and Roth published a similar approach [**?**]. However, it used the probabilistic model to infer new truth degrees based on previously labeled data.

After that, the focus shifted from updating the Bayesian model toward dealing with more generic scenarios. PTDCorr, a probabilistic model for truth discovery created by Yang, Bai, and Liu in 2019, evaluates the correlation between questions and their similarity [**?**]. The Gaussian model was utilized in GTFC (Gaussian Truth Finder with Correlations) to divide sources into groups, each having sources that provided similar information [**?**]. As a result, when a group continues to provide incorrect information, its impact on truth degrees is reduced.

The problem with the probabilistic approach is that every time a new piece of information is added, the truth degrees must be recalculated from scratch. As a result of its inability to accommodate new information, the probabilistic model cannot be used for search engines.

Machine Learning Approach: Only a few papers used a machine learning approach to discover the truth. The problem was treated as a graph learning problem in semi-supervised learning, with information pieces and sources represented as nodes and links that model different relationships between these nodes [**?**]. A source that provides a piece of information, for example, is depicted by one edge. However, numerous sources that provide identical information regularly are modeled by linking these sources with another type of edge. The algorithm then would learn the truth values of unknown links based on the labeled ones.

Since acquiring labeled data with known truth degrees is not always possible, Li et al. introduced an unsupervised learning approach that utilizes neural memory networks [**?**]. Information truth and source trust are represented as vectors, differentiating two sorts of data. The first is categorical,

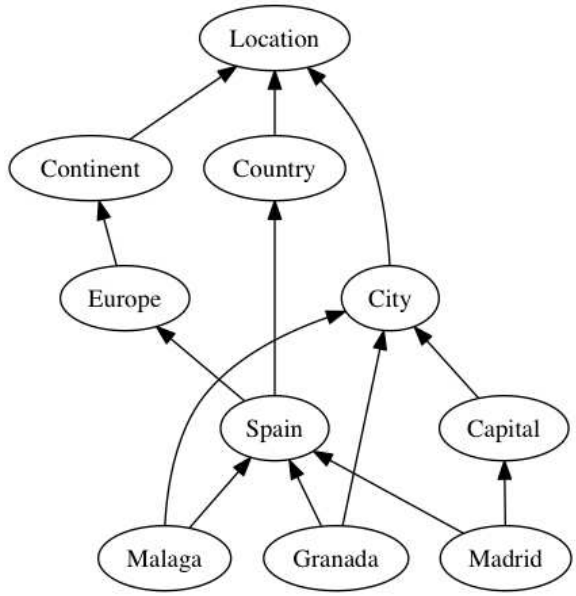

Fig. 2: Partial ordering DAG example

such as determining the correct gate number for a flight, whereas the second analyzes continuous values, such as the flight's departure time.

Semantic Approach: Recently, the semantic approach received more attention. This is because other methods take structured data as input without diving into the complexities of extracting answers, questions, and sources. However, using the semantic approach, the algorithm takes RDF triples as input and discovers data conflicts. RDF triples with the same subject and predicate, but different objects are defined as a conflict because they discuss the same question yet offer different answers [**?**].

Beretta et al. presented TDO, truth discovery with ontology, in 2016, which dealt with the case of multiple valid answers to a question when they represent more generalized knowledge [**?**]. A partial-ordering DAG, shown in Figure **??**[**?**], in which each instance is related to more general ones, can infer generalized information. To evaluate the TDO model, a synthetic dataset was generated. The error rate was calculated by comparing the calculated source trust to the one used to construct the dataset.

Lui et al. published a paper in 2018 that described an algorithm for resolving object conflicts in linked data [**?**]. The case when sources

copy data from each other was studied, with the consequence of reducing their trust. Although they estimated that cases with more than one answer to a question make up around 32% of the data on the web, they left such cases unhandled, preferring to keep it for future research.

Other Approaches: These approaches are designed to address a specific problem concerning object conflicts. Q. Li et al. tackled the problem of long-tail data, which occurs when a few sources supply a large amount of data, but the majority only provide a few. Furthermore [**?**], Ma et al. introduced the concept of sources having varying levels of trust within each domain [**?**]. Y. Li et al. released a paper on resolving object conflicts in situations where data arrives sequentially and evolves over time, such as the temperature of a particular city [**?**]. In addition, in 2014 and 2016, two papers discussed resolving object conflicts in heterogeneous data [**?**, **?**].

Chang et al. published a textual data truth discovery algorithm in 2019 [**?**]. Since the authors assumed that data is constructed as questions and answers, this remained an initial attempt at handling textual data.

Although all of these approaches dealt with various examples of truth discovery, they were all unable to receive new pieces of information over time, making them unsuitable for use on search engines.

3 Theoretical Model

This section will begin by describing the theoretical and mathematical model used to develop the truth discovery component. We outline the overall structure and then go over each sub-component individually.

3.1 Proposed Component Structure

The main goal of this paper is to develop a search engine-friendly component that can resolve object conflicts using a non-repetitive algorithm. We limit our study in this paper to the truth discovery component itself, aside from the search engine as a whole. We will base our algorithm on TDO and use the same partial ordering graph to handle generalization scenarios[**?**]. We treat

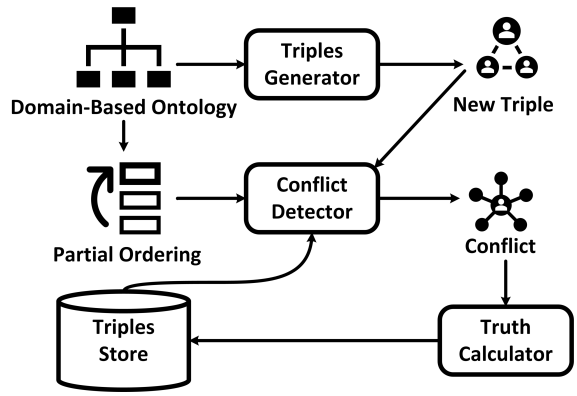

Fig. 3: An Overview of the Proposed Component's Design

RDF triples as the input, and the information truth degree and sources trust as the component's output. Figure **??** shows an overall view of the proposed component design.

The triples generator is treated as an external system that supplies RDF triples in a sequential manner. These triples, however, are subject to two limitations. First, each triple must be associated with its source. Second, the subject, predicate, and object (if it is an instance) must all be linked to the component's ontology; otherwise, the component will not understand or detect conflicts.

The conflict detector pulls all the triples from the triples store that conflict with the newly added one using the partial ordering graph. The partial ordering graph helps to order these triples according to the degree of generality they provide, distinguishing between conflicting facts and those that convey more general information. Finally, the truth calculator adjusts the truth degrees and source trust for the corresponding conflict and saves the results to the triples store.

3.2 Domain-Based Ontology

The truth discovery component uses a domain-based ontology. This is because generalization levels vary depending on the studied domain. Cities, for example, are more specialized than their countries, whereas continents are a more general concept, which only applies to a domain-based ontology of places. On the contrary, the music domain has its own generalization levels.

Any domain-based ontology can be used as long as the partial ordering graph is based on it.

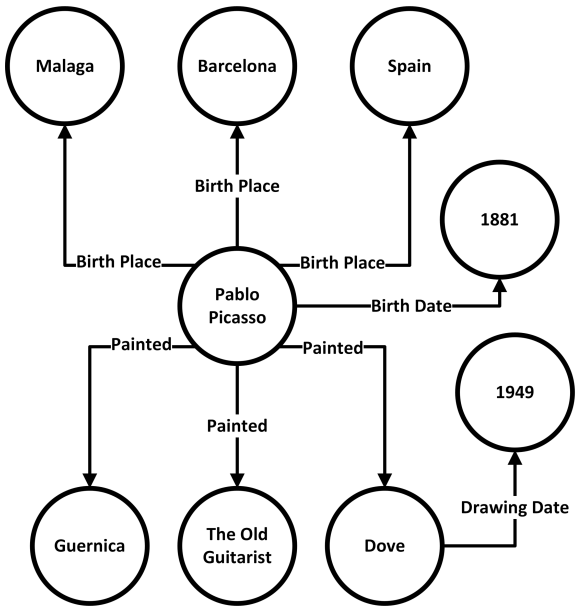

Fig. 4: Example of RDF triples for Picasso

Furthermore, all triples must be mapped to the same ontology. If several domain-based ontologies are to be merged, their partial ordering graphs must be merged as well.

3.3 Conflict Detection

The conflict detector searches for RDF triples conflicting with the new triple. All triples with the same subject and predicate may conflict because they express the same question; however, the conflict is determined by the generality levels of the objects and their nodes inside the partial-ordering graph.

Figure **??** depicts an example of RDF triples related to Picasso, in which his birthplace is linked to Málaga, Barcelona, and Spain, resulting in a data conflict. However, since Picasso painted Guernica, The Old Guitarist, and Dove, these pieces of information are considered a conflict by our definition. As a result, we will need to revise our definition of conflicting information to limit it to predicates with a single correct answer. Handling predicates that accept multiple correct values without being more generalized concepts from each other is left for future research.

Based on the prior discussion, we can formulate the following formula for detecting conflicts between two RDF triples:

$$\begin{gathered}\forall(s \in S), (p \in P), (o_1, o_2 \in O) \\ R_1 = (s, p, o_1) \wedge R2 = (s, p, o_2) \\ \wedge singleValued(p) \\ \Rightarrow conflict(R_1, R2)\end{gathered} \tag{1}$$

Where s and p are the subject and predicate, S and P are the set of all subjects and predicates, o_1 and o_2 are two different objects, and O is the set of all objects. If R_1 and R_2 are two different triples consisting of the previously defined elements and the predicate p accepts a single value, then we define R_1 and R_2 as conflicting triples. Formula **??** can be further expanded to detect conflicts for multiple RDF triples in the following manner:

$$\begin{gathered}\forall(s \in S), (p \in P), (o_1, o_2, ..., o_n \in O) \\ R_1 = (s, p, o_1) \wedge R_2 = (s, p, o_2) \\ \wedge ... \wedge R_n = (s, p, o_n) \\ \wedge singleValued(p) \\ \Rightarrow conflict(R_1, R_2, ..., R_n)\end{gathered} \tag{2}$$

Going back to Figure **??** and using formula **??**, we can rule out the *painted* predicate from being a conflict since it does not require a single correct value.

3.4 Partial Ordering

When RDF triples are connected to the ontology, they become connected to the partial-ordering graph since it was generated from that ontology. Thus, we can accurately differentiate conflicting information from those that support each other. Figure **??** shows the same example of RDF triples for Picasso from Figure **??** after linking the instances to the partial-ordering graph. Although Málaga and Barcelona remain conflicting information, Spain is now considered supporting information.

Hence, we update our previous definition in formula **??** to use partial-ordering relations as follows:

$$\begin{gathered}\forall(s \in S), (p \in P), (o_1, o_2 \in O) \\ R_1 = (s, p, o_1) \wedge R2 = (s, p, o_2) \\ \wedge o_1 \not\preceq o_2 \wedge o_2 \not\preceq o_1 \\ \wedge singleValued(p) \\ \Rightarrow conflict(R_1, R2)\end{gathered} \tag{3}$$

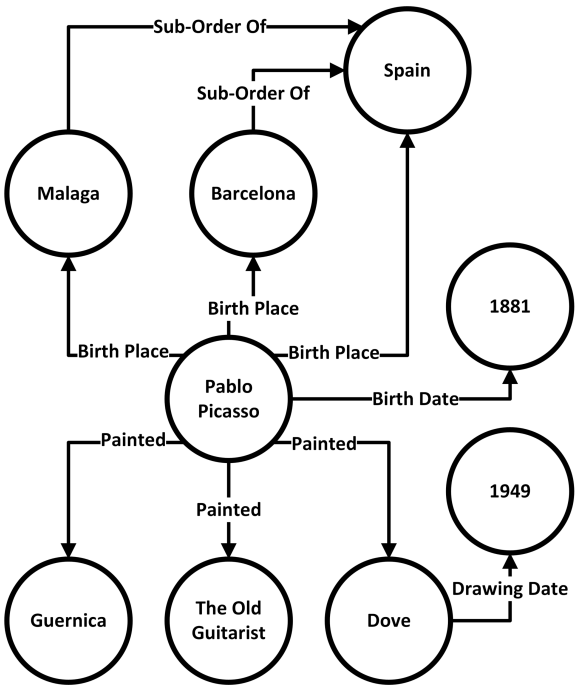

Fig. 5: Example of RDF triples for Picasso after linking the instances to the partial-ordering graph

$x \preceq y$ indicates that y is a more general instance than x, whereas $x \npreceq y$ indicate that such generalization order does not exist. The new definition in formula **??** declares two RDF triples as conflicting under the additional condition that none of their objects is an ancestor of the other; thus, not a more generalized instance. Therefore, the only conflicting RDF triples based on the formula **??** definitions are Picasso's birthplace being in Málaga or Barcelona.

On the other hand, when a source provides an RDF triple, we can infer that it also supports all the generalized triples. This can be demonstrated through the following definition:

$$(s, p, o) \Rightarrow (s, p, o_i);\ o \preceq o_i \tag{4}$$

From the definition in formula **??**, we conclude that not all triples that form a conflict are conflicting with each other. In fact, the root node of the partial-ordering graph, which is named *thing*, is the correct information for all conflicts because it represents the highest level of generalization.

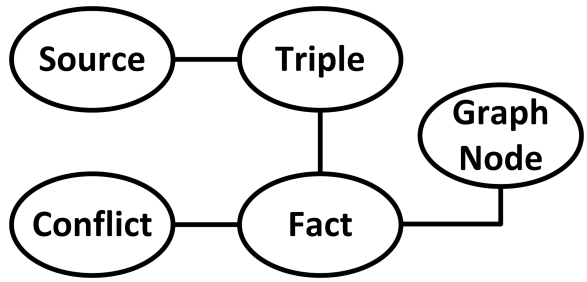

Fig. 6: Theoretical model for representing the object-conflict problem

3.5 Truth Calculation

Truth calculator takes the conflicting RDF triples as input and updates the truth degree and source trust before saving them back to the triples store. The theoretical model used to represent the object conflicts problem is shown in Figure **??**.

Each triple is connected with the source that provided it and a fact. The fact contains all the triples with the same subject, predicate, and object but different sources. On the other hand, each fact is connected with the conflict, which is formed from the subject and predicate, representing the question in the philosophical model. Also, the object of each fact is connected with the partial-ordering graph node to help determine its generalization level.

Using the theoretical model in Figure **??**, we develop three rules for updating information truth degree and source trust. First, we define the public version concept, which starts from zero, and increases with every newly added triple. Then, we define the three rules for updating truth values as follows:

1. **Updating an expired triple:** Each triple has its own version, representing the public version when it was last updated. When triple's version becomes outdated, we update the truth degree and source trust related to that triple. In addition, we define the maximum number of iterations for a triple to be updated. This can be formulated using the following formula:

$$\begin{gathered}\forall R(t,s,p,o): \\ expired(R) \wedge \neg maxIterations(R) \\ \Rightarrow update(t), update(s,p,o), \\ update(version(R)), \\ increment(iterations(R))\end{gathered} \tag{5}$$

Where t, s, p, and o are the source that provided the triple, the subject, the predicate, and the object, respectively. In addition, (s,p,o) forms the fact that triple R supports.

2. **Updating an expired conflict:** Same as triples, each conflict is given a version representing the number of triples belonging to that conflict when it was last updated. Once the conflict version becomes outdated, when many new triples related to that conflict are added, we recalculate the truth degree values for all facts inside that triple. This is shown in the following formula:

$$\begin{gathered}\forall (s,p): expired(s,p) \\ \Rightarrow truth(s,p,o) = \\ \frac{\sum_{R_i(t_i,s,p,o_i);\, o_i \preceq o} trust(t_i)}{max_{s_j \in S, p_j \in P, o_j \in O}(truth(s_j,p_j,o_j))}\end{gathered} \tag{6}$$

The truth degree of each fact inside the expired conflict (s,p) equals the sum of all sources trust that provided the fact or a fact whose object is a more general instance. The result is then divided by the maximum truth degree to normalize it.

3. **Updating an expired source:** The source's version is the number of RDF triples it provided when it was last updated. Once that version becomes outdated, the trust of that source is recalculated using the following formula:

$$\begin{gathered}\forall (t): expired(t) \\ \Rightarrow trust(t) = \\ \frac{\sum_{R_i(t,s_i,p_i,o_i)} truth(s_i,p_i,o_i)}{max_{t_j \in T}(trust(t_j))}\end{gathered} \tag{7}$$

The trust of each source t equals the sum of all facts (s_i,p_i,o_i) that this source supports. Similarly to formula **??** The result is divided by the maximum source trust to normalize it. Note that, in formula **??** we don't add more general facts to the sum because we don't want to overrate the source trust [**?**].

4 Practical Model

This section will describe the practical model and the algorithms used to build the truth discovery component. We begin by explaining the steps to construct the partial-ordering graph, then move to the algorithms to add a new triple. Finally, we go over finding the correct facts within a conflict.

4.1 Building Partial-Ordering DAG

We will use the same steps for constructing the partial-ordering DAG as in [**?**]. Note that the partial-ordering DAG is a graph without cycles. In other words, starting from a node, we cannot get back to the same node by following the graph's edges. This is a mandatory restriction for the generalization hierarchy.

4.2 Adding a New Triple

We start from formulas **??**, **??**, and **??** to build the algorithm for adding a new triple. To do that, we define four constants that regulate the frequency of updates inside the truth discovery component:

1. *TRIPLE_PERIOD*: The triple store period indicates the maximum difference in the public version number between the triple store and an RDF triple before updating that triple's truth degree and source trust.
2. *MAX_ITERS*: This constant is related to *TRIPLE_PERIOD* because it specifies the maximum number of times a triple's truth degree and source trust are updated. The associated triple is no longer updated when the number of updates surpasses this threshold.
3. *CONFLICT_PERIOD*: It represents the number of newly added triples that belong to a particular conflict, after which the truth degrees of all facts in that conflict are updated. The number of newly added triples is calculated from the last time that conflict was updated.
4. *SOURCE_PERIOD*: It represents the number of newly added triples from a particular source, after which the trust of that source is updated. The number of newly added triples is calculated from the last time that source was updated.

Now, we can develop algorithm **??** which is used to add a new triple to the truth component and update truth degrees and source trust.

Algorithm 1 Adding a New Triple

Require: *graph*: partial-ordering graph
store: triples store
t: the source of the new triple
s: the subject of the new triple
p: the predicate of the new triple
o: the object of the new triple

$store.version \leftarrow store.version + 1$
$source \leftarrow store.getSource(t)$
$fact \leftarrow store.getFact(s, p, o)$
$node \leftarrow graph.getNode(o)$
$ancestorNodes \leftarrow graph.getAncestors(node)$
$ancestorFacts \leftarrow \{\}$
for each $node \in ancestorNodes$ **do**
  $ancestorFact \leftarrow store.getFact(s, p, node)$
  $ancestorFacts.add(ancestorFact)$
end for
$triple = store.save(t, s, p, o)$
$store.tripleQueue.addLast(triple)$
$store.updateValues(triple)$
for each $ancFact \in ancestorFacts$ **do**
  $ancTriple\ store.save(source, ancFact)$
  $store.tripleQueue.addLast(ancTriple)$
  $store.updateValues(ancestorTriple)$
end for
$store.updateExpiredTriples()$
$store.updateConflict(fact.conflict)$
$store.updateSource(source)$

Algorithm **??** calls three functions at the end, *updateExpiredTriple*, *updateConflict*, and *updateSource*, which correspond to formulas **??**, **??**, and **??**, respectively. We implement these three functions as follows:

1. *updateExpiredTriples*: Updates expired triples whose version is behind triple store version by at least *TRIPLE_PERIOD*, without exceeding *MAX_ITERS* iterations for each triple.

   The *normalize* function normalizes the fact truth by dividing it by the maximum value among all facts truths calculated so far. The same thing applies to the function that normalizes the source trust; however, it normalizes the source trust by the maximum value among all sources trust calculated so far.

2. *updateConflict*: Updates a conflict if it is expired based on the value of *CONFLICT_PERIOD*. It takes the conflict to update as an input.

Algorithm 2 *updateExpiredTriples*

while $\neg tripleQueue.empty()$ **do**
  $triple \leftarrow tripleQueue.first()$
  $diff \leftarrow store.version() - triple.version()$
  if $diff < TRIPLE\_PERIOD$ **then**
    return
  end if
  $tripleQueue.removeFirst()$
  $triple.fact.truth \leftarrow triple.fact.truth$
    $+triple.source.trust$
    $-triple.source.oldTrust$
  $triple.source.trust \leftarrow triple.source.trust$
    $+triple.fact.truth$
    $-triple.fact.oldTruth$
  $normalize(triple.fact.truth)$
  $normalize(triple.source.trust)$
  $triple.iters \leftarrow triple.iters + 1$
  if $triple.iters < MAX\_ITERS$ **then**
    $tripleQueue.addLast(triple)$
  end if
end while

Algorithm 3 *updateConflict*

Require: $conflict$: the conflict to update

$oldV \leftarrow conflict.lastUpdateVersion$
$curV \leftarrow conflict.currentVersion$
if $curV - oldV < CONFLICT\_PERIOD$ **then**
  return
end if
for each $fact \in conflict.facts$ **do**
  $fact.truth \leftarrow 0$
  for each $source \in fact.sources$ **do**
    $fact.truth \leftarrow fact.truth + source.trust$
  end for
  $normalize(fact.truth)$
end for
$conflict.version \leftarrow conflict.version + 1$

3. *updateSource*: Updates a source if it is expired based on the value of $SOURCE\_PERIOD$. It takes the source to update as an input.

Algorithm 4 *updateSource*

Require: *source*: the source to update

```
1: oldV ← source.lastUpdateVersion
2: curV ← source.currentVersion
3: if curV − oldV < SOURCE_PERIOD then
4:     return
5: end if
6: source.trust ← 0
7: for each fact ∈ source.facts do
8:     source.trust ← source.trust + fact.truth
9: end fornormalize(source.trust)
10: source.version ← source.version + 1
```

4.3 Finding Correct Information

The straightforward approach for determining the correct fact inside a conflict is to pick the fact with the highest truth degree. However, in our case, that is incorrect because the most generic fact associated with the node "thing" will always have the highest truth degree in a conflict. Thus, we must identify the most specific fact that is correct. Finally, we consider facts that are more generic than the correct one as correct facts.

We use a similar approach to the one used in [**?**] with some optimizations to determine the most specific correct fact. The algorithm used in [**?**] started from the most generic node "thing" in the partial-ordering graph and worked its way down to the children with the highest truth degree. Since the majority of nodes are connected to "thing", that algorithm had a very high complexity. Instead, we start from the nodes that have a fact associated with them. Thus, we prevent the algorithm from visiting nodes that are not needed.

In addition, we prevent the algorithm from visiting facts whose object is connected to an ontology node. The idea here is that ontology nodes contain concepts, such as "city" or "state". These facts are unlikely to be the most specific because sources do not claim a generic fact like "some is born in a city". By avoiding visiting these facts, we reduce the algorithm complexity.

Algorithm **??** demonstrates this strategy.

Algorithm 5 Find Correct Facts For a Conflict

```
1: correctFacts ← {}
2: maxVal ← 0
3: for each fact ∈ conflict.facts do
4:     if fact.graphNode.isOntologyNode then
5:         continue
6:     end if
7:     if fact.truth > maxVal then
8:         maxVal ← fact.truth
9:         correctFacts.clear()
10:        correctFacts.add(fact)
11:    else if fact.truth = maxVal then
12:        correctFacts.add(fact)
13:    end if
14: end for
15: while ¬correctFacts.empty() do
16:    maxChildren ← {}
17:    maxVal ← 0
18:    for each fact ∈ correctFacts do
19:        node ← fact.graphNode
20:        for each child ∈ node.children do
21:            if child.isOntologyNode then
22:                continue
23:            end if
24:            childFact ← getFact(child)
25:            childTruth ← childFact.truth
26:            if childTruth > maxVal then
27:                maxVal ← childFact.truth
28:                maxChildren.clear()
29:                maxChildren.add(childFact)
30:            else if childTruth = maxVal then
31:                maxChildren.add(childFact)
32:            end if
33:        end for
34:    end for
35:    if maxChildren.empty() then
36:        break
37:    end if
38:    correctFacts ← maxChildren
39: end while
40: maxChildren ← correctFacts
41: correctFacts ← {}
42: for each fact ∈ maxChildren do
43:    node ← fact.graphNode
44:    ancNodes ← graph.getAncestors(node)
45:    for each node ∈ ancNodes do
46:        fact ← getFact(node)
47:        correctFacts.add(fact)
48:    end for
49: end for
50: return correctFacts
```

5 Experimental Study

The experimental study, as well as the dataset used to evaluate our approach, will be presented in this section. We will also provide the acquired results and a discussion to grasp the main benefits of the developed truth component.

5.1 Dataset Description

In order to evaluate the developed truth component and compare our results to the ones provided in [**?**], we use their dataset that lists the birthplace of famous people retrieved from DBPedia linked data. As a result, we also use the same partial-ordering graph for places.

The used partial-ordering graph consists of 682,658 nodes. Among them, 663,373 are leaves, which do not have more specific instances. The graph also contains a single root node, the "dbpedia-owl:Thing" node.

On the other hand, the dataset is synthetic and is based on the assumption that the information in DBPedia is correct. Hence, synthetic sources are generated and assigned a fixed trust for each. Each source is simulated to provide facts based on their predetermined trust. Thus, three types of files are generated. The EXP files tend to give specific facts, whereas the UNI files give more generic information. On the other hand, the LOW_E files represent the intermediate case. Table **??** lists statistics about the three types of files.

Table 1: Dataset Statistics

Number of	EXP	UNI	LOW_E
Files	20	20	20
Triples[1]	3,109,990	3,116,184	3,114,642
Facts[1]	423,393	424,973	422,941
Conflicts[1]	9,998	9,997	9,997
Sources[1]	1,000	1,000	1,000
Total Triples[1 2]	21,538,207	19,107,670	19,541,867

[1]Average number on each file

[2]Number of triples that are provided by sources or generated as generic ones when adding new triples.

From table **??** we can see that files are similar in size. However, EXP files generate more generic triples than UNI and LOW_E because source give more specific facts in them.

5.2 Evaluation

We evaluated the truth component using the following three criteria:

1. **Speed:** We measured the time taken by the truth component to finish processing the complete dataset. The test was run on a PC with an 8th generation corei7 processor, 16 GB of RAM, and an SSD hard drive.
2. **Correct Information Accuracy:** We calculated the confusion matrix to evaluate the truth component accuracy in detecting correct information. Table **??** shows how values are distributed inside the confusion matrix.

Table 2: Confusion Matrix

	Predicted Negative	Predicted Positive
Actual Negative	TN	FP
Actual Positive	FN	TP

In table **??** we see four values:

- TN: The number of facts that were correctly predicted as incorrect.
- FN: The number of correct facts that the component predicted to be incorrect.
- FP: The number of incorrect facts that the component predicted to be correct.
- TP: The number of facts that were correctly predicted as correct.

Finally, we calculated the following four metrics to evaluate the component's accuracy in detecting correct facts.

- Precision:

$$P = \frac{TP}{TP + FP} \tag{8}$$

- Recall:

$$R = \frac{TP}{TP + FN} \tag{9}$$

- F1 Measure:

$$F1 = \frac{2 \times P \times R}{P + R} \tag{10}$$

- Area Under ROC Curve:

$$AUC = \frac{TP}{2 \times (TP + FN)} + \frac{TN}{2 \times (FP + TN)} \quad (11)$$

3. **Source Trust Error:** We calculate the error in absolute value between the predetermined source trust and the calculated trust for that source. The calculated trust equals the number of facts the source claimed that the component predicted to be correct. Finally, we calculate the average source trust error over all sources.

5.3 Results and Comparison

We ran the tests for each file individually. Since our approach uses four constants to control the number of updates, we defined the value for these constants experimentally, such that increasing them after this point did not improve the results. Table **??** lists the constant values used in the experiment.

Table 3: Constant Values

Constant	Value
TRIPLE_PERIOD	1,000,000
MAX_ITERS	1
CONFLICT_PERIOD	1,000
SOURCE_PERIOD	2,000

The results were as follows:

1. **Speed:** The developed truth component took around half an hour to process all 60 files. Since TDO did not provide any information on their speed, we ran their code to the same dataset and recorded the time taken. TDO took approximately four hours to finish processing all 60 files.
2. **Correct Information Accuracy:** We calculated the correct information accuracy for the three types of files individually. For each of them, we used the average values per file to calculate the metrics. Table **??** shows the confusion matrix values for each of the three file types.

   Based on table **??**, we calculate the metrics for each of the three file types in table **??**.

Table 4: Confusion Matrix Calculated Values

Value	EXP	UNI	LOW_E
TN	330,417.7	321,995.9	323,583.5
FN	1,699.8	8,244.15	5,618.75
FP	10,617.8	21,051.45	17,259.5
TP	80,770.3	73,964.35	76,685.45

Table 5: Metrics for Each File Type

Metric	EXP	UNI	LOW_E
Precision	88.38%	77.84%	81.63%
Recall	97.93%	89.97%	93.17%
F1	92.91%	83.47%	87.02%
AUC	97.41%	91.92%	94.05%

   Unfortunately, since TDO authors did not provide any information on their accuracy to determine the correct information, we could not compare our results to theirs.
3. **Source Trust Error:** We calculated the average source trust error for each file type. Also, we compared our results to the ones provided in [**?**]. Table **??** shows our results with the comparison.

Table 6: Source Trust Errors

File Type	Our component	TDO
EXP	0.0734	0.171
UNI	0.1126	0.173
LOW_E	0.0963	0.172

5.4 Results Discussion

We compare our speed and source trust error results to the ones obtained by TDO in [**?**]:

1. **Speed:** Our component was able to finish processing the dataset eight times faster than TDO. These results are justified because TDO makes 20 iterations to reach its results [**?**]. Our component makes a farther less number of iterations which can be roughly calculated by checking the dataset statistics in table **??** and the used constant values in table **??**.

Since the value of *TRIPLE_PERIOD* equals one, each triple is updated twice; when it is first added and then again when pulled from the triples queue. Also, comparing the *CONFLICT_PERIOD* (1,000) with the average number of conflicts in each file (300), we conclude that most triples did not conduct iterations based on their conflict, except some triples whose conflict contains a large number of triples. Finally, since the *SOURCE_PERIOD* equals 2,000, and the average number of triples for each source is 3,000, we estimate that most triples perform one or two iterations based on their source's period.

Overall, most triples perform three or four iterations, which, compared to 20 iterations in TDO, explains the speed improvement of about eight times.

2. **Source Trust Error:** Although we significantly lowered the number of iterations, we notice a considerable improvement in reducing the source trust error when adding triples one by one compared to the error when computing the values for all triples at once.

   We explain this by examining the truth component's life cycle. The first added triples might require many iterations to obtain good values because the source's trust is not accurately evaluated yet. However, sources reach a realistic estimation of their trust over time. Thus, the later added triples require fewer iterations to calculate their truth degree because they rely on accurate trust values. Furthermore, starting from a realistic truth degree for these facts allows calculating the sources trust more accurately using fewer iterations.

6 Conclusion and Perspectives

This paper presented a non-repetitive semantic truth discovery approach that calculates information truth degree and source trust, taking into account the generalization of information.

The developed component can be further enhanced as well as its algorithm. Enhancing the model itself includes adding support to handle predicates that accept more than one correct value. On the sources side, the relationship between sources can be studied to detect the ones that copy information from each other. Furthermore, each source can have a trust related to a specific domain, such that sources have different trusts in different domains. One more enhancement can include studying the relationship between different predicates. For example, the age of a particular person is related to their birth date.

On the other hand, the algorithm for adding a new triple can also be enhanced. First, we can change the constants to change dynamically. The algorithm can start with values that make a large number of updates and update these values to reduce the number of updates when sources start to gain realistic trust. Finally, if used in a search engine, the algorithm can take feedback from query processors related to the most searched information. This information can have higher priority for being updated than stale information.

References

[1] Valentina Beretta, Sébastien Harispe, Sylvie Ranwez, and Isabelle Mougenot. How can ontologies give you clue for truth-discovery? an exploratory study. In *Proceedings of the 6th International Conference on Web Intelligence, Mining and Semantics*, pages 1–12, 2016.

[2] Laure Berti-Équille. *Truth Discovery*, pages 1–8. Springer International Publishing, Cham, 2018.

[3] Chen Chang, Jianjun Cao, Qin Feng, Nianfeng Weng, and Yuling Shang. Truth discovery of multi-source text data. *IEICE Transactions on Information and Systems*, 102(11):2249–2252, 2019.

[4] Xin Luna Dong, Laure Berti-Equille, and Divesh Srivastava. Integrating conflicting data: the role of source dependence. *Proceedings of the VLDB Endowment*, 2(1):550–561, 2009.

[5] Luciano Floridi. Semantic information and the correctness theory of truth. *Erkenntnis*, 74(2):147–175, 2011.

[6] Alban Galland, Serge Abiteboul, Amélie Marian, and Pierre Senellart. Corroborating information from disagreeing views. In *Proceedings of the third ACM international conference on Web search and data mining*, pages 131–140, 2010.

[7] Luyang Li, Bing Qin, Wenjing Ren, and Ting Liu. Truth discovery with memory network.

Tsinghua Science and Technology, 22(6):609–618, 2017.

[8] Qi Li, Yaliang Li, Jing Gao, Lu Su, Bo Zhao, Murat Demirbas, Wei Fan, and Jiawei Han. A confidence-aware approach for truth discovery on long-tail data. *Proceedings of the VLDB Endowment*, 8(4):425–436, 2014.

[9] Qi Li, Yaliang Li, Jing Gao, Bo Zhao, Wei Fan, and Jiawei Han. Resolving conflicts in heterogeneous data by truth discovery and source reliability estimation. In *Proceedings of the 2014 ACM SIGMOD international conference on Management of data*, pages 1187–1198, 2014.

[10] Xian Li, Xin Luna Dong, Kenneth Lyons, Weiyi Meng, and Divesh Srivastava. Truth finding on the deep web: Is the problem solved? *arXiv preprint arXiv:1503.00303*, 2015.

[11] Yaliang Li, Jing Gao, Chuishi Meng, Qi Li, Lu Su, Bo Zhao, Wei Fan, and Jiawei Han. A survey on truth discovery. 17(2):1–16, feb 2016.

[12] Yaliang Li, Qi Li, Jing Gao, Lu Su, Bo Zhao, Wei Fan, and Jiawei Han. On the discovery of evolving truth. In *Proceedings of the 21th acm sigkdd international conference on knowledge discovery and data mining*, pages 675–684, 2015.

[13] Yaliang Li, Qi Li, Jing Gao, Lu Su, Bo Zhao, Wei Fan, and Jiawei Han. Conflicts to harmony: A framework for resolving conflicts in heterogeneous data by truth discovery. *IEEE Transactions on Knowledge and Data Engineering*, 28(8):1986–1999, 2016.

[14] Wenqiang Liu. Truth discovery to resolve object conflicts in linked data. *arXiv preprint arXiv:1509.00104*, 2015.

[15] Wenqiang Liu, Jun Liu, Bifan Wei, Haimeng Duan, and Wei Hu. A new truth discovery method for resolving object conflicts over linked data with scale-free property. *Knowledge and Information Systems*, 59(2):465–495, 2019.

[16] Fenglong Ma, Yaliang Li, Qi Li, Minghui Qiu, Jing Gao, Shi Zhi, Lu Su, Bo Zhao, Heng Ji, and Jiawei Han. Faitcrowd: Fine grained truth discovery for crowdsourced data aggregation. In *Proceedings of the 21th ACM SIGKDD international conference on knowledge discovery and data mining*, pages 745–754, 2015.

[17] Lawrence Page, Sergey Brin, Rajeev Motwani, and Terry Winograd. The pagerank citation ranking: Bringing order to the web. Technical Report 1999-66, Stanford InfoLab, November 1999. Previous number = SIDL-WP-1999-0120.

[18] Jeff Pasternack and Dan Roth. Latent credibility analysis. In *Proceedings of the 22nd international conference on World Wide Web*, pages 1009–1020, 2013.

[19] Mr P Bastin Thiyagaraj and A Aloysius. A survey on truth discovery methods for big data. *International Journal of Computational Intelligence Research*, 13(7):1799–1810, 2017.

[20] Dalia Attia Waguih and Laure Berti-Equille. Truth discovery algorithms: An experimental evaluation. *arXiv preprint arXiv:1409.6428*, 2014.

[21] Dong Wang, Lance Kaplan, Hieu Le, and Tarek Abdelzaher. On truth discovery in social sensing: A maximum likelihood estimation approach. In *Proceedings of the 11th international conference on Information Processing in Sensor Networks*, pages 233–244, 2012.

[22] Yi Yang, Quan Bai, and Qing Liu. A probabilistic model for truth discovery with object correlations. *Knowledge-Based Systems*, 165:360–373, 2019.

[23] Xiaoxin Yin, Jiawei Han, and S Yu Philip. Truth discovery with multiple conflicting information providers on the web. *IEEE transactions on knowledge and data engineering*, 20(6):796–808, 2008.

[24] Xiaoxin Yin and Wenzhao Tan. Semi-supervised truth discovery. In *Proceedings of the 20th international conference on World wide web*, pages 217–226, 2011.

[25] Bo Zhao and Jiawei Han. A probabilistic model for estimating real-valued truth from conflicting sources. *Proc. of QDB*, 1817, 2012.

[26] Bo Zhao, Benjamin IP Rubinstein, Jim Gemmell, and Jiawei Han. A bayesian approach to discovering truth from conflicting sources for data integration. *arXiv preprint arXiv:1203.0058*, 2012.

[27] Yan Zheng, Meijuan Yin, Junyong Luo, and

Gongzhen He. Truth discovery on multi-dimensional properties of data sources. In *Proceedings of the ACM Turing Celebration Conference-China*, pages 1–8, 2019.