An Effective and Efficient Truth Discovery Framework over Data Streams

Tianyi Li
School of Computer Science and Engineering
Northeastern University
Shenyang, China
litianyi_neu@163.com

Yu Gu
School of Computer Science and Engineering
Northeastern University
Shenyang, China
guyu@mail.neu.edu.com

Xiangmin Zhou
School of Computer Science and Information Technology
RMIT University
Melbourne, Australia
xiangmin.zhou@rmit.edu.au

Qian Ma
School of Computer Science and Engineering
Northeastern University
Shenyang, China
maqian_neu@163.com

Ge Yu
School of Computer Science and Engineering
Northeastern University
Shenyang, China
yuge@mail.neu.edu.com

1. INTRODUCTION

The current big data era has witnessed various sources providing information on the same set of objects or events. The data inconsistency across multiple sources is an important research issue in many applications. The real-world applications like weather situation analysis and health-care require techniques to identify which data sources are more reliable or what information is accurate. For example, when we identify the weather condition of a city, the inconsistent information may be obtained from multiple websites. As another example, different medical records on a patient may be found from different hospitals. Thus, it is highly demanded to automatically identify trustworthy information from conflicting data. For this task, truth discovery has been proposed to model the source quality and derive the truth based on a principle: the information from a reliable source is trustworthy and the source providing trustworthy information is reliable. By leveraging this principle, several mechanisms have been proposed in previous works for both static and dynamic data. Consider a set of conflicting stock information for Apple Inc. at certain time as shown in Figure 1. As the information on the open price is arriving continuously, the truth on it evolves over time. In addition, the value from Insidestocks is closer to the truth at $t_{i-1}$, while that from Stocksmart is closer to the truth at $t_i$. This implies the reliability degrees of these three sources change over time as well. Thus, it is vital to identify the reliability of sources and the truths over continuous data streams, and develop advanced techniques for the truth discovery under dynamic scenario. Existing approaches for truth discovery mainly focus on static data [6, 7, 8, 19, 2, 22, 3, 1, 15, 5, 9, 12, 4, 14, 24], where an iterative process is exploited. The truth discovery process constantly iterates until the source weight converges to an optimal value. Applying the iterative process to the truth discovery at each timestamp over streams, the high accuracy performance can be achieved. However, these approaches suffer from expensive time costs, which is not applicable to high-speed data streams. Recently, some approaches have been proposed to improve the truth discovery efficiency by learning source weights and deriving truths incrementally [11, 23]. However, these methods sacrifice much accuracy,
because they model each source weight as a constant. The reliability of each source estimated by them is converged to a value, while the true source weights in real applications are constantly changing over time [16].

To effectively and efficiently discover truths over streams, we need to well address three issues. First, various iterative methods should be incorporated in a nice way to find the truths and the reliability of sources. This is important, as the optimal truths and source weights at each timestamp can only be derived by iteration strategy. As a result, the accuracy of truth discovery over data streams can be improved. Second, we need to design a set of advanced techniques which adaptively decide the frequency of source weight assessment to minimize the number of iterative operations. As data streams flow in large volume at high speed, it is clearly unacceptable to perform iterations at each timestamp. Finally, we should study the errors caused by not accessing the source weights continually over streams, and control these errors in a certain range.

In this paper, we propose a novel framework for effective and efficient truth discovery over streams. The idea behind it is to incorporate the iterative process in truth discovery for high accuracy and adaptively reduce the frequency of source weight assessment for high efficiency. Specifically, we first define two concepts, Unit error and Cumulative error, to describe the error caused by not changing the source reliability over data streams. Then, we present the relationship between each of these two concepts and the source reliability change based on theoretical analysis, which guarantees the accuracy of our truth discovery framework. For minimizing the source weight assessment frequency, we turn the problem of source weight assessment into an optimization problem and propose a scheme called ASRA to determine this frequency adaptively over data streams. In summary, we make the following contributions:

- We speculate the condition of the source reliability evolution under the constraints of small errors based on theoretical analysis, which guarantees the accuracy of our method. A probabilistic model is constructed to estimate the probability of meeting these conditions.
- We propose an optimization-based scheme ASRA, that minimizes the source reliability assessment frequency by estimating the maximum value of cumulative error smaller than a given threshold in a certain confidence level of probabilities.
- We propose a framework, which adaptively determines the time of source reliability assessment by combining the incoming data. Our framework incorporates various iterative approaches to estimate the reliability of sources, and balances the efficiency and accuracy by tuning the parameters.
- We validate the proposed framework on real datasets, and the results demonstrate the high performance of our proposed framework in terms of effectiveness and efficiency.

The rest of paper is organized as follows. We survey the related work in Section 2, and formulate the research problem in Section 3. Section 4 proves some conclusions of truth discovery over data streams. Section 5 introduces the probability model and proposes our method. Section 6 conducts experiments and analyzes experimental results. Section 7 concludes our paper.
Table 1: Notations

<table>
<thead>
<tr>
<th>Notation</th>
<th>Definition</th>
<th>Defined in (Section)</th>
</tr>
</thead>
<tbody>
<tr>
<td>$v^{(k,e,m)}$</td>
<td>the observation of the $m^{th}$ property for the $e^{th}$ object by the $k^{th}$ source at $t_i$</td>
<td>3</td>
</tr>
<tr>
<td>$V_t$</td>
<td>the observation of all the objects on all the properties from all the sources at $t_i$</td>
<td>3</td>
</tr>
<tr>
<td>$w^t_i$</td>
<td>the weight of the $k^{th}$ source at $t_i$</td>
<td>3</td>
</tr>
<tr>
<td>$W_t$</td>
<td>the source weight collection at $t_i$</td>
<td>3</td>
</tr>
<tr>
<td>$v^{(k,e,m)}_{o,i}$</td>
<td>the truth of the $m^{th}$ property for the $e^{th}$ object at $t_i$</td>
<td>3</td>
</tr>
<tr>
<td>$V_t^o$</td>
<td>the truths of all the objects on all the properties at $t_i$</td>
<td>3</td>
</tr>
<tr>
<td>$\lambda$</td>
<td>the smoothing factor</td>
<td>3.1</td>
</tr>
<tr>
<td>$\Delta w^t_i$</td>
<td>the source weight evolution on $k^{th}$ source at $t_i$</td>
<td>3.2</td>
</tr>
<tr>
<td>$\varepsilon$</td>
<td>the unit error threshold</td>
<td>4</td>
</tr>
<tr>
<td>$\alpha$</td>
<td>the probability threshold</td>
<td>5.2</td>
</tr>
<tr>
<td>$E$</td>
<td>the cumulative error threshold</td>
<td>5.2</td>
</tr>
</tbody>
</table>

can only work over categorical data. In [11], Li et al. proposed an incremental truth discovery method by transforming their optimization-based solution into a probabilistic model. However, the previous truth discovery work has shown that true source weights change over time [16], and this key point has not been considered in the models proposed in [23] and [11]. The source weight learned by these incremental methods converges to a certain value, which is considered as the corresponding true source weight. Although a smoothing factor has been introduced to capture the source’s reliability changes [11], the source weight computed by it also finally converges to a certain value. Thus, these incremental methods suffer from low accuracy compared with optimization-based solutions. To the best of our knowledge, our work is the first attempt ever made to trade off the accuracy and efficiency of truth discovery over data streams flexibly by tuning the parameters [10]. Moreover, with our proposed framework, various iterative truth discovery algorithms can be utilized to improve accuracy with neglectable efficiency losses. The notation used in this paper is listed in Table 1 for easy reference.

3. PROBLEM FORMULATION

In this section, we illustrate our proposed framework for truth discovery over data streams. Before proceeding to the problem formalization, we will introduce several important concepts first, Observation, Source Weight, and Truth.

Definition 1. An observation is the data that describes an object property of a source at a timestamp. We denote the observation of the $m^{th}$ property on the $e^{th}$ object from the $k^{th}$ source at $t_i$ as $v^{(k,e,m)}_{o,i}$, and all observations at $t_i$ as $V_t$.

Definition 2. A source weight is the reliability degree of a source at a timestamp. The source weights at $t_i$ are denoted as $W_t = \{w^t_1, w^t_2, \ldots, w^t_i\}$, where $w^t_i$ is the reliability degree of the $k^{th}$ source at $t_i$.

Definition 3. A truth is an aggregated result derived from truth discovery. We denote the truth of the $m^{th}$ property for the $e^{th}$ object at $t_i$ as $v^{(e,m)}_{o,i}$. Let $v^{(e,m)}_{o,i}$ be the optimal truth satisfying the convergence criterion of a given iterative method at $t_i$, and Dist be a distance function. Given a timestamp $t_i$ for source weight assessment, the truth $v^{(e,m)}_{o,i}$ is a value that holds the condition: $\text{Dist}(v^{(e,m)}_{o,i}, v^{(e,m)}_{o,i}) = 0$. Given a timestamp $t_i$ without source weight assessment, and two thresholds, $\varepsilon$, $\alpha$, the truth $v^{(e,m)}_{o,i}$ is a value that is derived by previous source weights $W_i$ ($i < j$) and holds the condition: the probability of $\text{Dist}(v^{(e,m)}_{o,i}, v^{(e,m)}_{o,i}) \leq \varepsilon (j-i)^2$ is no less than $\alpha$. The truths of all the objects on all the properties at $t_i$ are denoted as $V_t^o$.

Given a set of observations $V_t$, truth discovery over data streams is to automatically infer the truths $V_t^o$ and the source weights $W_i$ at each timestamp $t_i$. In this paper, we propose a novel framework that balances the effectiveness and efficiency of truth discovery over data streams. The idea behind it is to incorporate iterative process in truth discovery for high accuracy and adaptively determine the frequency of source weight assessment for high efficiency.

For this task, we first formalize the truth computation and the source weight evolution to analyze the error caused by not assessing source weights continually over data streams. Then, we define two concepts, unit error and cumulative error, and speculate the relationship between the source weight evolution and the two errors based on theoretical analysis, which guarantees the accuracy of our framework.

Finally, we propose an optimization-based scheme which minimizes the iterative operations, and then propose our method which adaptively decides the source weight assessment frequency by combining the incoming data. We denote the timestamp that our method updates the source weights as update point. Next, we will introduce our basic ideas on truth computation and source weight evolution.

3.1 Truth Computation

Truth computation is to keep the truths close to the claims from reliable sources. Traditional voting or averaging schema assumes all sources are equally reliable, which is generally unreasonable in real applications. To overcome this problem, many truth discovery methods use weighted voting or averaging to obtain the truths [8, 7, 11, 19, 6, 2], which makes the observations from high quality sources more important. In this paper, we infer the truth by exploiting the same weighted averaging strategy considering its advantages:  

$$v^{(e,m)}_{o,i} = \frac{\sum_{k=1}^{K} w^{t_k}_i \cdot v^{(k,e,m)}_{o,i}}{\sum_{k=1}^{K} w^{t_k}_i} \quad (1)$$

According to this weighted combinations, the information from the higher quality sources is more trustworthy, which is consistent with the principle of truth discovery. However, for truth discovery over data streams, the information
usually evolves smoothly. To capture this characteristic, we add one smooth constraint on the aggregated results. As such, the truth \( v_i^{(s,e,m)} \) is computed by:

\[
v_i^{(s,e,m)} = \frac{\sum_{k=1}^{K} w_k^t \cdot v_i^{(k,e,m)} + \lambda \cdot v_i^{(s,e,m)}}{\sum_{k=1}^{K} w_k^t + \lambda}
\]

where \( \lambda \) is the smoothing factor \([11]\). This equation treats the truth \( v_i^{(s,e,m)} \) as the information from a pseudo source and \( \lambda \) as the weight of this source.

Existing iterative truth discovery methods usually assess the truths and source weights by conducting an alternating iterative process \([8, 7, 11, 19, 6, 2]\). In other words, such methods update truths while fixing source weights and then update source weights while fixing truths until convergence. We aim to design a framework which can embed various iterative truth discovery approaches for the accuracy improvement, and infer the truth by exploiting the weighted combinations strategy \(\text{i.e., Formula (1) or (2)}\). Thus, an iterative truth discovery method can be plugged into our framework only in the case that its truth computation is in the form of weighted combinations.

### 3.2 Source Weight Evolution

Based on the principle of truth discovery, the source weight reflects the contribution of a source to the results of weighted combinations. Therefore, a relatively smooth evolution of a source weight implies a small variation on the contribution of this source. Under this situation, neglecting the updating of source weights will cause small errors, while decrease the iterative process. Thus the iterative methods can be applied to dynamic scenarios. The Source Weight Evolution \( \Delta w_k^t \) on \( k^{th} \) source at time \( t \) is computed by:

\[
\Delta w_k^t = \left| w_k^t / \sum_{k=1}^{K} w_k^t - w_k^{t-1} / \sum_{k=1}^{K} w_k^{t-1} \right|
\]

To observe the evolution of source weights, we conduct a set of experiments on two real-world datasets: Stock Dataset and Weather Dataset. These datasets have been used in the evaluation of truth discovery solutions \([9, 3]\), and their ground truths are available. For each dataset, we randomly select two sources, \( S_1 \) and \( S_2 \), for tests. Each source weight is quantified by comparing its observation with the ground truths and measuring the closeness between them. Since data usually contain multiple attributes in real applications, we normalize the deviation from various attribute values. Figure 2 shows the experimental results on source weight evolution over two different real-world datasets. Clearly, the evolution of source weights is quite minor at some moments. Under this scenario, it is natural to utilize previous source weights instead of current ones to obtain truths. For one thing, since the source weight computation is neglected, the iterative process is decreased under dynamic scenario. Thus, the iterative methods are applicable to data streams to improve the accuracy of truth discovery. For another, the deviation between the optimal truth and the approximate one will be small as well. Next, we will analyze this deviation caused by un-assessing source weights.

### 4. THEORETICAL ANALYSIS

In this section, we prove the condition of the source weight evolution under the constraints of small errors caused by un-assessing source weights. We first define the error in the form of mathematical formula. The unit error \( \Phi_j \) \((i < j)\) is given by:

\[
\Phi_j = \frac{v_{ij}^{(s,e,m)} - v_{ij}^{(k,e,m)}}{v_{ij}^{(max,e,m)}}
\]

where \( v_{ij}^{(s,e,m)} \) \((i < j)\) is the approximate truth computed based on the previous source weight \( W_i \), and \( v_{ij}^{(max,e,m)} \) is the absolute maximum value of \( v_{ij}^{(k,e,m)} \) \((1 \leq k \leq K)\). We use \( v_{ij}^{(max,e,m)} \) to normalize the distance between the optimal truth \( v_{ij}^{(s,e,m)} \) and the approximate one \( v_{ij}^{(s,e,m)} \) at \( t_j \). Here, \( v_{ij}^{(s,e,m)} \) refers to \( v_{ij}^{(max,e,m)} \) in Definition 3.3. Specifically, let \( \Phi \) represent \( \Phi_{i-1} \). The relationship between the unit error \( \Phi \) and the source weight evolution is given by Theorem 1.

**THEOREM 1.** Given a unit error threshold \( \varepsilon \), let \( K \) be the size of source collection. If for all \( k, 1 \leq k \leq K \), the source weight evolution holds: \( \Delta w_k^t \leq \sqrt{\varepsilon / K} \), then the unit error \( \Phi \leq \varepsilon \) is satisfied.

**PROOF.** According to Formulas (1) and (4), we derive the following:

\[
\sqrt{\Phi} = \frac{\sum_{k=1}^{K} (w_k^t / \sum_{k=1}^{K} w_k^{t-1} - w_k^{t-1} / \sum_{k=1}^{K} w_k^{t-1}) \cdot v_i^{(k,e,m)}}{v_i^{(max,e,m)}}
\]

Then, we can infer

\[
\sqrt{\Phi} \leq \sum_{k=1}^{K} \left| \frac{(w_k^t / \sum_{k=1}^{K} w_k^{t-1} - w_k^{t-1} / \sum_{k=1}^{K} w_k^{t-1}) \cdot v_i^{(k,e,m)}}{v_i^{(max,e,m)}} \right|
\]

Since \( \left| v_i^{(max,e,m)} \right| \geq \left| v_i^{(k,e,m)} \right| \((1 \leq k \leq K)\), we have

\[
\sqrt{\Phi} \leq \sum_{k=1}^{K} \left| \frac{w_k^t / \sum_{k=1}^{K} w_k^{t-1} - w_k^{t-1} / \sum_{k=1}^{K} w_k^{t-1}}{v_i^{(max,e,m)}} \right|
\]

Further,

\[
\sqrt{\Phi} \leq K \cdot \sqrt{\varepsilon / K} = \sqrt{\varepsilon}
\]

So far, we prove that \( \Phi \leq \varepsilon \) holds. \( \Box \)
Theorem 1 demonstrates the relationship between the source weight evolution and the unit error, i.e., the unit error $\Phi$ should be no more than $\varepsilon$ if the formula (5) is satisfied,

$$\Delta w_k^j \leq \sqrt{\varepsilon}/K \quad (1 \leq k \leq K) \quad (5)$$

Under this scenario, we can use $W_{t-1}$ to approximate $W_t$ and ensure that the deviation between the optimal truth and the approximate one will be constrained by a threshold $\varepsilon$. Since we un-assess all source weights at $t_i$, the time complexity of truth discovery is linear. For further improving the efficiency, we aim to assess source weights over time as few as possible. Therefore, it is essential to further analyze the relationship between the source weight evolution and the errors accumulated in a time period, i.e., the cumulative error, which is computed by Formula (6),

$$\Phi_t^j = \sum_{h=1}^{j} \Phi_h^i$$

Combining with Formula (4), we can see that the cumulative error is defined as the sum of unit errors in a time period. Then, we give the maximum value of the cumulative error under the condition that Formula (5) holds in a time period.

**THEOREM 2.** Given a unit error threshold $\varepsilon$, let $K$ be the size of source collection. If for all $k$, $1 \leq k \leq K$, $i < h \leq j$, the source weight evolution holds: $\Delta w_i^k \leq \sqrt{\varepsilon}/K$, then the cumulative error $\Phi_t^j$ meets the condition $\Phi_t^j \leq \Delta T(\Delta T + 1)/(2\Delta T + 1)\varepsilon/6$, where $\Delta T = j - i$.

**PROOF.** According to Formulas (1) and (4), we derive the following:

$$\sqrt{\Phi_t^j} = \sqrt{\sum_{h=1}^{K} w_h^k / \sum_{k=1}^{K} w_h^k - w_i^k / \sum_{k=1}^{K} w_i^k \cdot \Phi_h^{(k,e,m)}}$$

Then similar to Theorem 1, we have

$$\sqrt{\Phi_t^j} \leq \sum_{k=1}^{K} \left| w_h^k / \sum_{k=1}^{K} w_h^k - w_i^k / \sum_{k=1}^{K} w_i^k \right|$$

According to $\Delta w_i^k \leq \sqrt{\varepsilon}/K$, for any $h$ ($i < h \leq j$), it is easy to derive the following:

$$\sqrt{\Phi_t^j} \leq (h - i) \cdot \sqrt{\varepsilon}$$

Further,

$$\sum_{h=1}^{j} \Phi_h^i \leq \sum_{h=1}^{j} (h - i)^2 \varepsilon$$

Then,

$$\sum_{h=1}^{j} \Phi_h^i \leq (j - i)(j - i + 1)(2(j - i) + 1)\varepsilon/6$$

Since $\Psi_j^i = \sum_{h=1}^{j} \Phi_h^i$, we have

$$\Psi_j^i \leq (j - i)(j - i + 1)(2(j - i) + 1)\varepsilon/6$$

Let $\Delta T = j - i$, we prove that $\Psi_j^i \leq \Delta T(\Delta T + 1)/(2\Delta T + 1)\varepsilon/6$ holds. $\square$

According to Theorem 2, we can get that the relationship between the unit error and the maximum value of cumulative error under the condition of $\Delta w_i^k \leq \sqrt{\varepsilon}/K$ ($i < h \leq j$, $1 \leq k \leq K$):

$$\max(\Psi_j^i) = \Delta T(\Delta T + 1)/(2\Delta T + 1)\varepsilon/6 \quad (7)$$

where $\Delta T = j - i$. Let the size of source collection $K$ be 3 and the unit error threshold $\varepsilon$ be 0.03. Suppose that we update the source weights at $t_1$ and the source weight evolutions satisfy Formula (5), from $t_2$ to $t_3$, i.e., $\Delta w_i^k \leq \varepsilon/5(1 \leq k \leq 3, 1 < i \leq 5)$. The cumulative error $\Psi_j^i$ will be no more than $4 \times (4+1) \times (2 \times 4+1) \times 0.03/6 = 0.9.$

**Theorem 2** ensures that, under dynamic scenario, we can incorporate iterative methods to improve the accuracy of truth discovery without sacrificing much efficiency. The reason is that we neglect the iterative estimation of source weights $W_i$ when the source weight evolutions $\Delta w_i^k$ ($1 \leq k \leq K$) satisfy Formula (5), i.e., the iterative truth discovery methods are utilized over data streams only at certain timestamps. In addition, as the cumulative error is constrained by $\varepsilon$ and $\Delta T$, we can ensure the accuracy of truth discovery even if the iterative process is reduced. Although we do not update the source weights at each timestamp, the accuracy of our method is still much higher than the existing incremental methods (as shown in Section 6).

To capture the temporal relations among truths by adding smoothing factor as in Formula (2), we only need to redefine $\Psi_j^{(max,e,m)}$ in Formula (4) as the absolute maximum value of $\Psi(j,e,m), \Psi(j,e,m), ... , \Psi(K,e,m)$, and slightly modify Formula (5) by changing $K$ into $K + 1$. The reason is that we treat the smoothing factor as the weight of the $(K + 1)\text{th}$ source and $\Psi^{(e,m)}$ as the information from this source. Since we still compute truths by exploiting weighted combinations, the smoothing factor will not affect our conclusions. Moreover, we introduce the smoothing factor for truth computation only when the data changing is smooth, thus it is reasonable to utilize the $\Psi_j^{(max,e,m)}$ to normalize the unit error.

As shown in Theorems 1 and 2, a relative smooth source weight evolution leads to a lower unit error comparing with a big “jump” (the peaks in Figure 2) of source weight evolution. However, since the evolution of source weight is unknown over data streams, it is hard to make sure whether Formula (5) is satisfied. For solving this issue, we propose a probabilistic model to dynamically estimate the probability of Formula (5) holding over data streams.

### 5. ASRA-BASED TRUTH DISCOVERY

In this section, we propose an adaptive source reliability assessment scheme (ASRA) for truth discovery over data streams. The basic idea behind this scheme is to dynamically determine the time for source weight assessment. Then the truth with a predetermined accuracy is identified. Specifically, we first derive a probabilistic model to estimate the probability of the source weight evolution which meets the condition in Formula (5). By integrating the conclusions in section 4, we achieve the maximal period of source weight assessment under the condition that the maximum value of cumulative error is smaller than a given threshold in a certain confidence level. This will transform the source weights assessment into an optimization problem. Based on this optimization problem, we then propose our ASRA scheme that adaptively assesses source weights over streams.

#### 5.1 Probability Forecasting Model

As proved in Theorem 1, the source weight evolution has great influence on unit error. If all the source weight evolutions meet the conditions in Formula (5), the unit error will be less than $\varepsilon$. Otherwise, it can not be controlled within the $\varepsilon$ constraint. However, in real-world applications, even if
the variation trend of the information from various sources can be obtained, the evolution of each source weight is still not available. Considering this, we propose a probability model based on the Bernoulli distribution to estimate the probability of Formula (5) holding over data streams. Given a timestamp \( t_i \), we can consider \( \Delta w_i^k \leq \sqrt{\varepsilon/K} \) (\( 1 \leq k \leq K \)) as an independent and random event. Here, the probability of the event occurrence is a random variable which follows Bernoulli distribution, i.e., \( \xi \sim B(1, p) \). Based on the probability theory, the probability \( p \) can be estimated by sampling as explained by Example 1.

**Example 1.** Given a unit error threshold \( \varepsilon \) and a source collection, assume that \( t_1 \sim t_1 \) is the initial period of time. We assess the source weight at each timestamp. Let \( N \) be the times of all source weight evolutions satisfying Formula (5) during this period. The total times of counting all source weight evolutions is \( M = l - 1 \). Thus the probability \( p \) can be estimated as \( N/M \).

As the time increases, both the source weight evolution and the probability \( p \) are likely to change. Thus, a dynamic estimation makes the probability \( p \) more accurate. This is also the basis of ASRA scheme. We will illustrate the time for the update of probability \( p \) while introducing our scheme.

### 5.2 ASRA Scheme

This section presents our ASRA scheme in details. The ASRA scheme includes two parts: (1) adaptive update point prediction; and (2) ASRA-based truth discovery algorithm. We first transform the update point prediction issue into an optimization problem which minimizes the frequency of source weight assessment. Then, an ASRA-based algorithm is proposed with the support of this optimization strategy. ASRA assesses source weights with changeable frequencies while finding the truth with a certain level of accuracy given by users. Accordingly, we can achieve high efficiency by reducing the frequency of assessing source weights and high accuracy by incorporating the iterative process. Given a current update point \( t_i \), ASRA predicts the next update point \( t_j \) by solving the following optimization problem:

\[
\begin{align*}
\max \quad & j = i + \Delta T \\
\text{s.t.} \quad & (\Delta T - 1)(\Delta T - 2)(2\Delta T - 3)\varepsilon/6 \leq E \quad (8)
\end{align*}
\]

where \( \Delta T \) is considered as the maximum period of assessing source weights. There are two constraint functions regarding this optimization problem as listed below:

- \( p^{\Delta T - 2} \geq \alpha \): This is equivalent to \( p^{\Delta w_i^k} \leq \sqrt{\varepsilon/K} \geq \alpha (1 \leq k \leq K, i + 1 < h < j) \), where \( \alpha \) is the probability threshold given by users. We do not need to estimate the source weight evolutions at \( t_{i+1} \) and \( t_j \). For one thing, we assess the source weights \( W_i \) since \( t_i \) is an update point. Considering that we should compute the source weight evolutions for dynamically updating \( p \), the source weights \( W_{i+1} \) is also assessed to obtain the evolution of all source weights, i.e., \( \Delta w_{i+1}^1, \ldots, \Delta w_{i+1}^K \). Then, we utilize \( W_{i+1} \) instead of \( W_{i+2} \) to compute the truth at \( t_{i+2} \). For another, we assess the source weights \( W_j \) since \( t_j \) is also an update point. Thus, it is unnecessary to estimate the probability of \( \Delta w_{i+1}^k \leq \sqrt{\varepsilon/K} \), \( \Delta w_j^k \leq \sqrt{\varepsilon/K} \) (\( 1 \leq k \leq K \)).
- \( (\Delta T - 1)(\Delta T - 2)(2\Delta T - 3)\varepsilon/6 \leq E \): Based on Theorem 2, when \( p^{\Delta w_i^h} \leq \sqrt{\varepsilon/K} \geq \alpha (i + 1 < h < j) \), the probability of max(\( \Psi_{i+1}^j \)) = \( (\Delta T - 1)(\Delta T - 2)(2\Delta T - 3)\varepsilon/6 \) is no smaller than \( \alpha \). Though we expect \( \Delta T \) to be large for high efficiency, max(\( \Psi_{i+1}^j \)) will become large with \( \Delta T \) increasing. Thus, we also need to make sure that max(\( \Psi_{i+1}^j \)) is no more than \( E \), where \( E \) is the cumulative error threshold given by users. By this way, the cumulative error between any two update points is constrained.

Formula (8) implicates that our ASRA scheme tries to search for the maximum period of assessing source weights. When the unit error threshold \( \varepsilon \) is fixed, only two tuned parameters, \( \alpha \) and \( E \), need to be set. A large \( \alpha \) may lead to a small \( \Delta T \), while a small \( \Delta T \) will also result in a small \( \Delta T \). However, the performance trend of \( \varepsilon \) is actually uncertain. We will show in Section 6 that the effects of the probability threshold \( \alpha \), cumulative threshold \( E \) and unit error threshold \( \varepsilon \) in our framework, and the performance of our framework can be flexibly changed by tuning these parameters.

Algorithm 1 presents the whole procedure of ASRA-based truth discovery. Let \( t_i \) denote the current timestamp and \( t_j \) denote the update point. Algorithm 1 performs in three steps. In the first step (lines 3-4), we update the source weights. Given the update points \( t_i \) and \( t_j+1 \) (line 3), we call the existing truth discovery method to assess the source weights \( W_j, W_{i+1} \). In the second step (lines 5-13), we update the probability \( p \) of satisfying Formula (5) by re-estimating \( p \) according to \( \Delta w_{i+1}^k (1 \leq k \leq K) \). In the last step (lines 14-18), we predict the next update point. By utilizing the probability \( p \) computed in the second step, we predict the next update point \( t_j \) according to Formula (8). If \( \Delta T \) computed by Formula (8) is less than 2, we set \( \Delta T = 2 \) (lines 16-17).

In Algorithm 1, line 4 suggests that various methods for source weight computation can be plugged into our scheme only if the truth computation of the plugged method is in the form of weighted combinations. We set a window size \( M \) for more accurately estimating the probability \( p \) without the influence of out-of-date data. Note that we can introduce the smoothing factor by slightly modifying our algorithm. As mentioned above, we treat the smoothing factor \( \lambda \) as the weight of \((K + 1)th\) source and the previous truths as the information from this source. As \( \lambda \) is a constant [11], only the source weight evolution and the size of source collection will be changed when the smoothing factor \( \lambda \) is introduced. Accordingly, for capturing the temporal relationship over streaming data, we only need to change \( K \) into \( K + 1 \) in line 6, and change “Formula (1)” into “Formula (2)” in line 21. For the existing truth discovery methods plugged into our scheme (line 4), we also simply change its truth computation from “Formula (1)” into “Formula (2)”. Obviously, the complexity of the algorithm is determined by the corresponding iterative truth discovery methods at an update point. Otherwise, its complexity is \( O(|V|) \) at \( t_i \).

For probability \( p \), there are two points to remark: (1) the cumulative error is usually constrained to a small value in real world applications. According to Formula (8), \( \Delta T \) will not be a large value. Thus we can assume that \( p \) is a constant in a small time window \( \Delta T \); and (2) \( p \) is defined as the probability of all the source weight evolutions satisfying Formula (5) at each timestamp, i.e., a small \( p \) implies the source weight evolution is generally large over data streams.
Algorithm 1: ASRA-based truth discovery
Input : Observation collection $V_i$, threshold $\alpha$, $E$;
Output: Truth collection $V^*_i$;
1 $j \leftarrow 1$, $m \leftarrow 1$, $N[1...M] \leftarrow 0$, $p \leftarrow 0$;
2 for $i = 1 \rightarrow \infty$ do
3   if $i = j$ then
4     Update $V_i^*$, $W_i$ according to existing iterative
5   if $i = j$ then
6     if $\Delta w_i^k (1 \leq k \leq K)$ satisfy Formula (5);
7     then
8     $N[m] = 1$;
9     if $m <= M$ then
10    $p = (\sum_{m-1}^M N[n])/m$;
11   else
12    Slide the window forward and keep array $N$
13    always contains $M$ elements;
14    $p = (\sum_{m-1}^M N[n])/M$;
15    $i = i - 1$;
16   Update $j$ by Formula (8);
17   if $j = i < 2$ then
18     $j = i + 2$;
19    $i = i + 1$;
20 else
21 $W_i \leftarrow W_{i-1}$;
22 Set $V_i^*$ by Formula (1);
23 Return $V_i^*$;

Note that the exact timestamp with a large source weight
evolution is still unknown if we do not compute the source
weights. Therefore, the algorithm may also neglect source
weight computation when the source weight evolution does
not satisfy Formula (5). However, according to Formula (8),
a small $p$ will lead to more frequent source weight estimation,
thus the high performance of our framework can be
ensured (as shown in Section 6).

6. EXPERIMENTS
In this section, we experimentally validate the proposed
approach for truth discovery over data streams.

6.1 Experimental Setup
We evaluate our framework on three real-world dataset-
s: Sensor Dataset$^1$, Stock Dataset$^2$ and Weather Dataset$^2$. The Sensor Dataset contains data from 54 sensors deployed in the Intel Berkeley Research lab between Feb. 28, 2004 and Apr. 5, 2004. Each sensor collected the time-stamped topology values once per 30 seconds. The temperature and humidity properties are adopted for evaluation. The Stock Dataset contains data for 1000 stocks that are collected from 55 sources over the weekdays of July 2011. We adopt three properties: change %, change value and last trade price. The ground truths are given. The Weather Dataset contains 18 sources that record weather data for 30 cities of United States from Jan. 28, 2010 to Feb. 4, 2010. We adopt the temperature and humidity properties, and consider the information collected from Accuweather.com as the ground truths.

Since the ground truths of Stock Dataset and Weather Dataset are known, each source weight can be quantified by measuring the distance between its observations and the ground truths. Accordingly, the true source weight of Stock Dataset and Weather Dataset are also available. Moreover, although Stock Dataset and Weather Dataset have been used in [11], the experimental results can be different because we choose various types of properties to conduct the experiments while only one type of property was used in [11].

6.2 Evaluation Methodology
We have conducted extensive experiments to evaluate the
effectiveness and efficiency of the proposed method by four
steps: (1) validate the effectiveness of the probabilistic model estimating source weight evolution; (2) analyze the effects of three parameters, probability threshold $\alpha$, cumulative error threshold $E$ and unit error threshold $\varepsilon$ in our framework; (3) evaluate the effectiveness and efficiency of our approach by comparing with state-of-art competitors; and (4) further confirm the accuracy of source weight computation of our proposed approach. Eleven methods, including seven state-of-the-art competitors and four proposed alternatives, are used in the experiments.

Baseline Methods. The following state-of-the-art methods for truth discovery over continuous data are implemented. The parameters of each baseline method are set according to the original paper.

- GTM: Using Bayesian probabilistic model for resolving conflicts on continuous data [21].
- CRH: Working with heterogeneous data by incorporating into various loss functions [8].
- DynaTD+smoothing: Adding the smoothing factor based on DynaTD [11].
- DynaTD+decay: Adding the decay factor based on DynaTD [11].
- DynaTD+all: Adding both the smoothing factor and the decay factor based on DynaTD [11].

Proposed Alternatives. We plug different existing truth
discovery methods into our framework. All these methods iteratively conduct the updates of source weights and truths until convergence. For the truth update, all these methods exploit weighted combinations strategy (i.e., Formula (1) or (2)) [8, 11] and can be plugged into our framework. The details on the source weight update for each method are as follows:

- ASRA(CRH): We incorporate CRH into our framework and choose the normalized squared loss function to measure the deviation from the truths to the observations. The source weight $w_i^k$ is derived as the following formula:

$$w_i^k = -\log\left(\frac{\sum_{k'=1}^K l_{i}^{k'}}{l_{i}^{k}}\right)$$  (9)
where \( l^k_i \) refers to the normalized squared loss function of the \( k^{th} \) source at \( t_i \), i.e.,
\[
l^k_i = \sum_{e=1}^{E} \sum_{m=1}^{M} \frac{(v^{(k,e,m)}_i - v^{(e,m)}_i)^2}{\text{std}(v^{(1,e,m)}, \ldots, v^{(K,e,m)})}
\] (10)

- ASRA(CRH-smoothing): We further introduce the smoothing factor \( \lambda \) to ASRA(CRH) for capturing the temporal relations over streams. Under this scenario, we consider \( v^{(e,m)}_{i-1} \) as the information from the \((K+1)^{st}\) source (\( v^{(e,m)}_{i+1} = v^{(K+1,e,m)}_i \)) and \( \lambda \) is the weight of this source. Therefore, only the number of sources in Formula (10) and Formula (9) need to be changed for computing loss functions and source weights.

- ASRA(Dy-OP): We incorporate the basic optimization function of DynaTD [11], denoted as Dy-OP, into our framework. The source weight \( w^k_i \) is derived as the following formula:
\[
w^k_i = \frac{q^k_i}{\eta \cdot l^k_i}
\] (11)
where \( q^k_i \) refers to the number of observations provided by the \( k^{th} \) source at \( t_i \) and \( \eta \) is a trade-off parameter of Dy-OP [11]. In addition, the normalized squared loss functions \( l^k_i \) (\( 1 \leq k \leq K \)) in Formula (11) are computed by Formula (10).

- ASRA(Dy-OP+smoothing): The smoothing factor \( \lambda \) is also introduced to ASRA(Dy-OP) for capturing the temporal relations over streaming data. As mentioned, only the number of sources need to be changed for computing source weights and loss functions.

So far, for each method plugged into our framework, we have presented the formula for its source update step. The details of Formulas (9) and (11) are listed in Appendix. For truth computation, we only need to utilize Formula (2) to capture the temporal relations over streaming data.

**Performance Metrics.** To evaluate the efficiency of our framework, we report the running time of each method. To assess the accuracy of the MAEs of each method by comparing their outputs with ground truths. For both metrics, lower values indicate better performance. All the algorithms were performed on a PC with Windows OS, Intel Core i7 processor.

### 6.3 Probabilistic Model Validation

This part validates the effectiveness of the probabilistic model for estimating the source weight evolution over data streams. Obviously, if the probabilistic model can capture the large source weight evolution (Formula (5) cannot be satisfied), our proposed model is effective. Thus, we validate the effectiveness of our probabilistic model by counting all probable scenarios including:

1. Formula (5) does not hold and our framework updates the source weights at the same time (denoted as TP);
2. Formula (5) holds and our framework keeps the source weights at the same time (denoted as TN);
3. Formula (5) does not hold and our framework keeps the source weights at the same time (denoted as FN);
4. Formula (5) holds and our framework updates the source weights at the same time (denoted as FP).

Both scenario (1) and (2) show that our probabilistic model captures the source weight evolution successfully. Thus, the effectiveness of our probabilistic model can be transformed into Capture Rate (CR) formulated as:
\[
CR = \frac{TP}{TP + TN}
\] (12)

The experiments are conducted over Stock Dataset and Weather Dataset. We vary two parameters, \( \alpha \) and \( \varepsilon \), to observe the effectiveness of our probabilistic model with different parameter settings. The cumulative threshold \( E \) is given to constrain the maximum of \( \Delta T \).

The experimental results are reported in Table 2. As we can see, \( CR \) is always more than 0.6 on both two datasets and can achieve more than 0.9 in some cases. Note that our framework assigns the first two timestamps to update points, which may lead to a higher \( FP \) and a lower \( CR \).

Therefore, our probabilistic model can capture the source weight evolution in most situations, which further proves the effectiveness of our framework.

### 6.4 Evaluation on Parameters

To analyze the effects of the probability threshold \( \alpha \), cumulative error threshold \( E \) and unit error threshold \( \varepsilon \) in our framework, we test the performance of our method over the Sensor Dataset and Weather Dataset by changing the value of one parameter while fixing the others. To discover truths, we incorporate Dy-OP into our framework, i.e., ASRA(Dy-OP). Three metrics: running time, MAE and assess times, are used to observe the influence of three parameters to our framework. Here, assess times is defined as the average times of assessing source weight over streaming data. Ob-

---

**Table 2: Probabilistic Model Validation**

<table>
<thead>
<tr>
<th>Parameter Setting</th>
<th>Experimental Results</th>
</tr>
</thead>
<tbody>
<tr>
<td>( \varepsilon )</td>
<td>( \alpha )</td>
</tr>
<tr>
<td>( 5 \times 10^{-4} )</td>
<td>0.45</td>
</tr>
<tr>
<td>( 1 \times 10^{-3} )</td>
<td>0.45</td>
</tr>
<tr>
<td>( 5 \times 10^{-3} )</td>
<td>0.45</td>
</tr>
<tr>
<td>( 5 \times 10^{-4} )</td>
<td>0.55</td>
</tr>
<tr>
<td>( 1 \times 10^{-3} )</td>
<td>0.55</td>
</tr>
<tr>
<td>( 5 \times 10^{-3} )</td>
<td>0.55</td>
</tr>
<tr>
<td>( 5 \times 10^{-4} )</td>
<td>0.65</td>
</tr>
<tr>
<td>( 1 \times 10^{-3} )</td>
<td>0.65</td>
</tr>
<tr>
<td>( 5 \times 10^{-3} )</td>
<td>0.65</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Parameter Setting</th>
<th>Experimental Results</th>
</tr>
</thead>
<tbody>
<tr>
<td>( \varepsilon )</td>
<td>( \alpha )</td>
</tr>
<tr>
<td>( 5 \times 10^{-2} )</td>
<td>0.45</td>
</tr>
<tr>
<td>( 1 \times 10^{-1} )</td>
<td>0.45</td>
</tr>
<tr>
<td>( 5 \times 10^{-1} )</td>
<td>0.45</td>
</tr>
<tr>
<td>( 5 \times 10^{-2} )</td>
<td>0.55</td>
</tr>
<tr>
<td>( 1 \times 10^{-1} )</td>
<td>0.55</td>
</tr>
<tr>
<td>( 5 \times 10^{-1} )</td>
<td>0.55</td>
</tr>
<tr>
<td>( 5 \times 10^{-2} )</td>
<td>0.65</td>
</tr>
<tr>
<td>( 1 \times 10^{-1} )</td>
<td>0.65</td>
</tr>
<tr>
<td>( 5 \times 10^{-2} )</td>
<td>0.65</td>
</tr>
</tbody>
</table>
6.4.1 Effect of $\alpha$

In this test, we evaluate the effect of the probability threshold $\alpha$ on the accuracy and efficiency of our framework. For Sensor Dataset, we fix $\varepsilon$ to $10^{-5}$ and $E$ to 1, and vary the value of $\alpha$ from 0.65 to 0.85. For Weather Dataset, we fix $\varepsilon$ to 0.1 and $E$ to 1, and vary the value of $\alpha$ from 0.15 to 0.45. The results are shown in Figures 3(a)-(e).

As we can see, with the increasing of $\alpha$, running time and assess times increase while MAE decreases. This result is caused by the following reason. The probability threshold $\alpha$ controls the holding probability of Formula (5) during the period of keeping source weights. Therefore, a relatively large $\alpha$ means Formula (5) should be more likely hold, and a smaller $\alpha$ will relax this constraint while leading to a relatively large $\Delta T$. In other words, a larger $\alpha$ achieves a higher accuracy while suffering from much sacrifice on efficiency.

6.4.2 Effect of $E$

In this test, we evaluate the effect of the cumulative error threshold $E$ on the performance of our framework. For Sensor Dataset, we set $E$ to 0.02 and 0.1 respectively, and fix $\varepsilon$ to $10^{-5}$ and $\alpha$ to 0.75. For Weather Dataset, we set $E$ to 0.2 and 1 respectively, and fix $\varepsilon$ to 0.1 and $\alpha$ to 0.2. The results are shown in Figures 3(d)-(f).

Obviously, with the decreasing of cumulative threshold $E$, running time and assess times increase while MAE decreases. According to Formula (8), a relatively large $E$ means our framework is allowed to make more errors between any two update points. Therefore, a large $E$ will lead to a large period of assessing source weights and improve the efficiency. However, it suffers from much sacrifice on accuracy.

6.4.3 Effect of $\varepsilon$

We test the effect of the unit error threshold, $\varepsilon$, on three metrics. For Sensor Dataset, we fix $\alpha$ to 0.6 and $E$ to 0.01, and set $\varepsilon$ to $5 \times 10^{-5}$, $10^{-4}$, and $5 \times 10^{-4}$ respectively. For Weather Dataset, we fix $\alpha$ to 0.95 and $E$ to 1, and set $\varepsilon$ to 0.2 and 0.5 respectively. The results are shown in Figures 3(g)-(i).

As we can observe, with the increasing of $\varepsilon$, running time and assess times decrease while MAE increases. However, the performance trend of unit error threshold is actually uncertain. Based on the first constraint function of Formula (8), a relatively small $\varepsilon$ may result in a larger $\Delta T$. At the same time, the second constraint of Formula (8) implicates that a relatively small $\varepsilon$ can also result in a smaller $\Delta T$. Since we set a relatively large cumulative error threshold $E$ ($E = 1$) in our experiments, the optimal $\Delta T$ is mainly restricted by the second constraint function of Formula (8). Thus a larger $\varepsilon$ achieves a better efficiency and suffers from much sacrifice on accuracy.

For the same parameter setting, with the time increasing, MAE decreases while both running time and assess time increase over two datasets. This is because the source weight evolutions of these two datasets become large as the time increases. Thus our framework automatically improve the frequency of assessing source weights and achieve the high accuracy of the truth discovery.

To summarize, all the experimental results (Figures 3(a)-(i)) show that these three parameters of our framework can tune the performance of truth discovery flexibly.

6.5 Evaluation on Performance

We first compare our proposed approach with the state-of-the-art competitors in terms of effectiveness and efficiency. Then, we further study the effectiveness of our approach under the optimal efficiency, and its efficiency under the best accuracy.

6.5.1 Comparison with Existing Approaches

In this test, we evaluate our proposed approach by comparing with the existing competitors: DynaTD, DynaTD+smoothing, DynaTD+decay, DynaTD+all, Dy-OP, CRH and GTM. For Stock Dataset, we set $\varepsilon$ to $10^{-3}$, $\alpha$ to 0.75 and $E$ to 1. For Weather Dataset, we set $\varepsilon$ to 0.1, $\alpha$ to 0.8 and $E$ to 1. For Sensor Dataset, we set $\varepsilon$ to $5 \times 10^{-6}$, $\alpha$ to 0.85 and $E$ to 0.01. Table 3 shows the experimental results for all the methods on the three datasets. Since the ground truths of Sensor Dataset are unknown, we only report the accuracy (MAE) on two datasets with ground truths, i.e., Stock Dataset and Weather Dataset.

**Efficiency.** In terms of efficiency, the proposed method performs nearly as well as DynaTD, DynaTD+smoothing,
Table 3: Comparison with Existing Approaches

<table>
<thead>
<tr>
<th>Method</th>
<th>Stock Dataset</th>
<th>Weather Dataset</th>
<th>Sensor Dataset</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>MAE (Time(ms))</td>
<td>MAE (Time(ms))</td>
<td>MAE (Time(ms))</td>
</tr>
<tr>
<td>ASRA(Dy-OP)</td>
<td>1.3941 (99)</td>
<td>0.4974 (419)</td>
<td>1.658 (658)</td>
</tr>
<tr>
<td>ASRA(CRH)</td>
<td>1.4007 (104)</td>
<td>0.5029 (424)</td>
<td>1.549 (674)</td>
</tr>
<tr>
<td>ASRA(Dy-OP+smoothing)</td>
<td>1.0142 (103)</td>
<td>0.4474 (417)</td>
<td>1.638 (658)</td>
</tr>
<tr>
<td>ASRA(CRH+smoothing)</td>
<td>1.9781 (117)</td>
<td>0.5096 (427)</td>
<td>1.549 (674)</td>
</tr>
<tr>
<td>DynaTD</td>
<td>1.5462 (99)</td>
<td>1.0593 (510)</td>
<td>1.496 (549)</td>
</tr>
<tr>
<td>DynaTD+smoothing</td>
<td>1.5064 (98)</td>
<td>0.9261 (306)</td>
<td>1.332 (595)</td>
</tr>
<tr>
<td>DynaTD+decay</td>
<td>1.4956 (98)</td>
<td>0.9300 (340)</td>
<td>1.332 (595)</td>
</tr>
<tr>
<td>DynaTD+all</td>
<td>1.4455 (93)</td>
<td>0.9265 (307)</td>
<td>1.332 (595)</td>
</tr>
<tr>
<td>Dy-OP</td>
<td>1.3328 (305)</td>
<td>0.4425 (1680)</td>
<td>1.394 (549)</td>
</tr>
<tr>
<td>CRH</td>
<td>1.3994 (325)</td>
<td>0.5028 (1782)</td>
<td>1.411 (692)</td>
</tr>
<tr>
<td>GTM</td>
<td>1.4112 (430)</td>
<td>0.6011 (1718)</td>
<td>2.133 (2133)</td>
</tr>
</tbody>
</table>

Figure 4: Efficiency Study

Figure 5: Accuracy Study

DynaTD+decay and DynaTD+all. As all these methods work in an incremental way, they can be viewed as the low bound of the iterative methods. Therefore, the results shown in Table 3 implicate our proposed framework can achieve high efficiency. Meanwhile, ASRA(Dy-OP) can run as fast as DynaTD on Stock Dataset. The reason is that the proposed framework only performs iterations at certain timestamps. Moreover, our proposed framework is more efficient compared with other iteration-based truth discovery methods. Specifically, our framework outperforms the iterative method GTM in terms of both accuracy and efficiency. The reason is that the basic methods plugged into our framework (CRH, Dy-OP) achieve better performance than GTM.

**Accuracy.** In terms of effectiveness, the proposed method is better than existing competitors, DynaTD, DynaTD+smoothing, DynaTD+decay and DynaTD+all. The reason is that these competitors exploit incremental computation, updating the source weights according to the new arrival data until each source weight converges to a certain value. However, the true source weights in real applications are constantly changing. Thus, the source weights computed by the incremental methods deviate from the true ones, leading to big errors. In addition, CRH and Dy-OP are more accurate than our methods (ASRA(CRH), ASRA(Dy-OP)), as they solve the truth discovery task by an iterative process that iteratively computes the truths and source weights at each timestamp. In this way, each source weight converges to its optimal one. However, without computing the source weights at each timestamp, the accuracy of ASRA(Dy-OP) and that of ASRA(CRH) are still similar to the corresponding basic methods Dy-OP and CRH. The reason is that our proposed framework updates the source weights frequently when the source weight evolutions are generally large. Based on Theorems 1 and 2, we can constrain the cumulative error and ensure the accuracy of our framework. When a smoothing factor is introduced, our methods, ASRA(Dy-OP+smoothing) and ASRA(CRH+smoothing), achieve the best accuracy among all the methods on Stock Dataset. It can also be observed that ASRA(Dy-OP) achieves better accuracy than ASRA(CRH), while Dy-OP performs better than CRH on both two datasets. Obviously, the accuracy of our framework is consistent with the basic method plugged into it.

In conclusion, from the performance comparison results, it can be seen that our framework always outperforms the iterative methods with respect to efficiency and performs better than the incremental methods in terms of accuracy.
Since our framework can contain different plugged truth discovery methods, it also outperforms some baselines in terms of both accuracy and efficiency (such as GTM).

### 6.5.2 Further Study

To further confirm the performance of our framework, we evaluate its efficiency while achieving the optimal accuracy, and its accuracy while the efficiency is optimal. In this test, we conduct experiments on Stock Dataset and Weather Dataset. Since our framework can flexibly tune the efficiency and accuracy of truth discovery over streaming data, both accuracy and efficiency can be optimized by tuning the parameters. Also, we change the number of properties in this part, and denote the experiments conducted on a single property as Single-Property (“Sin” in Figures (4)-(5)), and the ones on multiple properties as Multiple-Property (“Mul” in Figures (4)-(5)). For evaluation on Single-Property, we choose the last trade price property for Stock Dataset, and the humidity property for Weather Dataset.

**Efficiency.** From Table 3, we can see that Dy-OP achieves the best accuracy comparing with all the baselines. Thus, the accuracy of Dy-OP can be considered as the optimal accuracy. We achieve the same accuracy with Dy-OP by tuning the parameters ($\varepsilon = 10^{-3}$, $\alpha = 0.85$, $E = 0.1$ for Stock Dataset and $\varepsilon = 10^{-3}$, $\alpha = 0.85$, $E = 1$ for Weather Dataset). Under this scenario, we evaluate the efficiency of our framework by comparing with Dy-OP.

From Figures 4(a)-(d), we can see that our framework achieves much higher efficiency performance than Dy-OP for both Single-Property and Multi-Property. The reason is that our framework does not assess the source weights continually. In addition, the gap between our framework and Dy-OP on Multi-Property is larger than the one on Single-Property, which illustrates our method is more suitable for addressing different types of properties.

**Accuracy.** To the best of our knowledge, DynaTD is the most effective incremental truth discovery method for continuous data, and also the basis of DynaTD+smoothing, DynaTD+decay, DynaTD+all [11]. Thus, we consider the efficiency of DynaTD as the optimal efficiency. Then we achieve the same efficiency with DynaTD by tuning the parameters ($\varepsilon = 10^{-3}$, $\alpha = 0.75$, $E = 1$ for Stock Dataset and $\varepsilon = 0.1$, $\alpha = 0.65$, $E = 1$ for Weather Dataset). Under this scenario, we evaluate the accuracy of our framework by comparing with DynaTD.

From Figures 5(a)-(d), we can see that, for both Single-Property and Multi-Property, the accuracy of our proposed framework is much higher than the incremental method. For one thing, we use the iterative method to assess source weights, which makes source weights converge to the optimal values at each timestamp. For another, both Theorems 1 and 2 ensure the accuracy of our framework. Although we do not assess the source weights continually, our framework achieves much higher accuracy comparing with the existing incremental methods. Moreover, Figure 5(a) shows that, at the initial time, the truths computed by our framework is nearly equal to the ground truths, which also implicates the high accuracy of our framework.

To summarize, by tuning the parameters of our framework, we can balance the efficiency and accuracy of the truth discovery task, and achieve better performance than the state-of-the-art competitors as well.

### 6.6 Evaluation on Source Weight

As aforementioned, the estimation of source weights plays a vital role in the truth discovery task. Thus, we design a set of experiments to evaluate the accuracy of source weight computation using our proposed framework. In this test, we choose Weather Dataset as the experimental dataset. We randomly select two sources (denoted as $S_1$, $S_2$ respectively) for experiments. Dy-OP method is plugged into our framework, i.e., ASRA(Dy-OP). For comparison purpose, we also compute the source weights using the existing incremental methods, DynaTD and DynaTD+decay. Moreover, for controlling the source weights in a same range, we utilize $L^1$-norm to regularize the source weights computed by all the methods.

Figures 6(a)-(b) show the experimental results. Clearly, each true source weight changes constantly over time, and the source weights computed by our framework are usually more closer to the true values. Conversely, a source weight computed by DynaTD and DynaTD+decay can converge to a certain value quickly, which is inconsistent with the real source weight change. In conclusion, these results prove the accuracy of our approach in terms of source weight computation.

### 7. CONCLUSION

In this paper, we study the truth discovery problem over data streams. We propose a framework for truth discovery which adaptively determines the frequency of assessing source weights for high efficiency and incorporates various iterative truth discovery methods for high accuracy. We first define and study the unit error and the cumulative error of truth discovery. Then we transform the prediction of the cumulative error into an optimization problem, and propose our ASRA scheme. Tuning parameters of our framework supports a trade-off between accuracy and efficiency in truth discovery. Moreover, by a series of theoretical analysis, the accuracy of our framework is guaranteed while the iterative processes are reduced. Extensive experiments on real-world datasets have been conducted to evaluate the effectiveness and efficiency of our approach, and the experimental results have proved the high performance of our truth discovery framework.

### 8. ACKNOWLEDGMENTS

This work is supported by the National Natural Science Foundation of China (61472071 and 61433008) and the National Basic Research Program of China (973 Program) under Grant No. 2012CB316201. Yu Gu is the corresponding author.

### 9. REFERENCES

Integrating conflicting data: the role of source trustworthiness.

**Proof of Formula (9)**

According to [8], for each timestamp $t_i$, the source weights $W_i$ are conducted as the following:

$$W_i \leftarrow \arg \min_{W_{i}} \sum_{k=1}^{K} w_i^k t_i^k \quad s.t. \quad \sum_{k=1}^{K} \exp(-w_i^k) = 1$$

Then the derivation of Formula (9) is the same as the derivation of source weights in [8].

**Proof of Formula (11)**

According to [11], as we model that each source weight changes over time, the source weights $W_i$ can be conducted as the following:

$$W_i \leftarrow \arg \min_{W_{i}} \eta \sum_{k=1}^{K} w_i^k t_i^k - \sum_{k=1}^{K} q_i^k \log(w_i^k)$$

where $q_i^k$ denotes the number of observations provided by $k$th source at $t_i$, and $\eta$ is given to support the trade-off between the two terms in Formula (14) [11]. Moreover, the initial loss function in [11] is un-normalized. However, in this paper, we choose the normalized squared loss function for addressing different types of attributes (Formula (10)). Since the standard deviation of the observations at each timestamp can be considered as a constant, the conclusions will not be affected. We take the partial derivative of $W_i$ in Formula (14) with respect to $w_i^k$, and set the partial derivative equal to zero. Then we obtain the source weight expression as shown in Formula (11).