PAPER

Uncertain Rule Based Method for Determining Data Currency

Mohan LI[†], **Jianzhong LI**[††], **Siyao CHENG**[††], ***Nonmembers***, ***and*** **Yanbin SUN**[†a)], ***Member***

SUMMARY Currency is one of the important measurements of data quality. The main purpose of the study on data currency is to determine whether a given data item is up-to-date. Though there are already several works on determining data currency, all the proposed methods have limitations. Some works require timestamps of data items that are not always available, and others are based on certain currency rules that can only decide relevant currency and cannot express uncertain semantics. To overcome the limitations of the previous methods, this paper introduces a new approach for determining data currency based on uncertain currency rules. First, a class of uncertain currency rules is provided to infer the possible valid time for a given data item, and then based on the rules, data currency is formally defined. After that, a polynomial time algorithm for evaluating data currency is given based on the uncertain currency rules. Using real-life data sets, the effectiveness and efficiency of the proposed method are experimentally verified.

key words: *data quality, data currency, uncertain rule*

1. Introduction

The correctness and accuracy of data can severely impact the quality of data-driven applications. It is reported that data error rates of enterprises can be as high as 30% [1], and low quality medical data caused up to 98,000 deaths annually in America*. It is believed that the outdated data is one of the most important factors bringing down the data quality. According to statistics, 2% of the records in a common customer database become obsolete in one month because the customers die, divorce, marry and move [2]. This problem is so-called data currency problem [3], and some methods for improving data currency have been proposed. These methods can be categorized into two categories.

(a) Several methods for data currency determination are based on accurate timestamps [4]–[8]. They can achieve good results if timestamps are available for each data item. However, in many real applications, the timestamps of data items are either absent or imprecise [9], which makes these methods not work effectively.

(b) To overcome the shortages of timestamps based methods, a rule based method of data currency has been proposed, and some fundamental problems are studied [3], [10]. Using the rules, the temporal orders of duplicate tuples describing the same entity can be inferred. These methods are effective to the data sets consisting of duplicate tuples, but facing two problems.

First, the methods focus on determining the temporal orders of values and selecting the relatively newest value. However, if all the values are obsolete, the selected value must also be obsolete. For example, assume that there are two tuples representing an employee whose name is Alice. One tuple indicates that Alice's salary is 3000$, and the other indicates that Alice's salary is 2000$. Assuming that there is a currency rule *"An employee's salary does not decrease"*, then we can infer that 3000$ is the current value since 3000 > 2000, but if the salary has been raised to 4000$, and the database has not been updated in time, then 3000$ and 2000$ are both obsolete values. However, if we knows that *"The salary of all the employees have been raised to more than $3500 in the year of 2016"*, we can easily find that the 3000$ is an obsolete value since it is 2018 now. This situation calls for introducing timestamps. It seems that we shall be back to the methods of category (a), but in this case, we only need to add timestamps to a few *rules*, rather than to each data item. This can greatly reduce the workload and makes the timestamping work easily to be done.

Second, the methods assume that all the rules are certain. That is, these rules enforce that the conclusions must be correct if the conditions are satisfied. However, the users often do not have adequate knowledge to provide certain rules. The knowledge of a user are often uncertain, such as *"about 90% of the employees' salaries do not decrease"*. Moreover, the rigorous conditions of a certain rule often lead to limited coverage. That is, in some cases, only a small amount of tuples can satisfy a certain rule's conditions. Thus, although certain rules can provide precise results, they cannot ensure a high recall rate since a large fraction of tuples in a data set maybe cannot satisfy the conditions of any certain rule.

To overcome the problems above, this paper tries to introduce timestamps and uncertainty into currency rules, and to determine data currency based on the rules. The main contributions are as follows.

(1) A class of timestamped uncertain currency rules is proposed for determining data currency. The rules can determine not only the temporal orders, but also the certainty that a data item are not obsolete at a given time point. Furthermore, the soundness of the rules are theoretically analyzed.

(2) Based on the uncertain rules, the currency of data

Manuscript received November 22, 2017.
Manuscript revised June 7, 2018.
Manuscript publicized July 10, 2018.

[†]The authors are with the Guangzhou University, Guangzhou, P.R. China.
[††]The authors are with the Harbin Institute of Technology, Harbin, P.R. China.
a) E-mail: yanbin_hit@foxmail.com (Corresponding author)

DOI: 10.1587/transinf.2017EDP7378

*http://www.medicalnewstoday.com/releases/11856.php

are formally defined. An algorithm for evaluating data currency is proposed. Based on the concepts of domination graph and currency graphs, the algorithm can determine the currency of all the data items in $O(mn^2 \log n)$ time, where n is the number of tuples and m is the number of attributes.

(3) The efficiency and effectiveness of the proposed methods are verified using real-life data sets.

The rest of the paper is organized as follows. Section 2 describes timestamped uncertain rules. Section 3 presents the definition of data currency. Section 4 provides the algorithm for determining data currency. Section 5 illustrates the experimental results. Section 6 discusses the related work and Sect. 7 concludes this paper.

2. Uncertain Currency Rules

2.1 Syntax

Let $R = (A_1, A_2, \cdots, A_m)$ be a relation schema of m attributes, and $dom(A_i)$ denote the domain of attribute A_i. $D = \{e_1, e_2, \cdots, e_n\}$ is a data set with schema R, where $e_i \in dom(A_1) \times \cdots \times dom(A_m)$ is the ith tuple. A *data item* is a certain attribute value of a certain tuple, that is, the data item $e_i[A_j]$ expresses the value of attribute A_j in tuple e_i. We use *column* A_j of D to express the projection of D on attribute A_j.

Each data item $e_i[A_j]$ has a *valid time*, expressed by $VT(e_i[A_j])$. $VT(e_i[A_j]) = \tau$ means that $e_i[A_j]$ is up-to-date only in the time before the time point τ. For example, let e represents employee Alice, and $e[\mathsf{City}] = \mathsf{Beijing}$. $VT(e[\mathsf{City}]) = 2013$ means that Alice lived in Beijing before 2013 and moved to another city after 2013. Let $\prec_A$ represent the currency orders on attribute A. $\prec_A$ is a partial order, and $e_i \prec_{A_k} e_j$ means that $e_i[A_k]$ becomes obsolete *earlier* than $e_j[A_k]$, that is, $VT(e_i[A_k]) < VT(e_j[A_k])$. Similarly, $e_i \prec_{A_k} \tau$ means that $e_i[A_k]$ becomes obsolete earlier than the time point τ, that is, $VT(e_i[A_k]) < \tau$. In other words, $e_i[A_k]$ changes to another value before τ for sure.

It is easy to observe that valid time is the *upper bound* of the time that a value is up-to-date. For some attributes the upper bound is easy to know. Thus these upper bounds can be propagated to other related attributes using the uncertain currency rules. For example, if we know the time that the attribute Zip changes to another value in 2013, i.e. the valid time of the value of Zip is 2013, then we may infer that the valid time of City is also 2013. We use $cer()$ to denote the certainty which is in $[0, 1]$. Consider the example of Zip and City, assume that $e[\mathsf{City}] = \mathsf{Beijing}$, and we get a conclusion that $e \prec_{\mathsf{City}} 2013$ with certainty 1. Since $e \prec_{\mathsf{City}} 2013$ means $VT(e[\mathsf{City}]) < 2013$ (for the value Beijing), we know that Beijing is definitely an obsolete value in the time point 2013. If $cer(e \prec_{\mathsf{City}} 2013) < 1$, then we know that Beijing may be an obsolete value in 2013 with certainty $cer(e \prec_{\mathsf{City}} 2013)$.

Valid time plays a critical role in currency determination, but it is unknown in many applications. The currency rules described in the following can help us to deduce the valid time. We define two types of uncertain currency rules, and try to deduce the possible range of the valid times.

(1) $\forall e_i, e_j (\psi \rightarrow e_i \prec_A e_j, \xi)$, where ξ is the certainty of the rule, and ψ is a conjunction of the predicates of the form (a) $e_i[B]\ op\ e_j[B]$, where $B \in R$ is an attribute, and $op \in \{=, \neq, <, >, \leq, \geq\}$, (b) $e_k[B]\ op\ b$ for $k \in \{i, j\}$, where b is a constant, (c) $e_i \prec_B e_j$, (d) $e_j \prec_B e_i$, (e) $e_k \prec_B \tau'$, where τ' is a time points, or (f) $\tau' \prec_B e_k$.

(2) $\forall e (\psi \rightarrow e \prec_A \tau, \xi)$ or $\forall e (\psi \rightarrow \tau \prec_A e, \xi)$, where τ is a time point, ξ is the certainty of the rule, and ψ is a conjunction of the predicates of the form (a) $e[B]\ op\ b$, where op is the same as in (1), (b) $e \prec_B \tau'$, where τ' is a time points, or (c) $\tau' \prec_B e_k$.

Given a rule r, let $lhs(r)$ denote the condition of r, i.e. ψ, and $rhs(r)$ denote the conclusion of r, i.e. $e_i \prec_A e_j$, $e \prec_A \tau$ or $\tau \prec_A e$. $rhs(r)$ is uncertain, and $lhs(r)$ may be uncertain since it may contains predicate in $rhs(r')$ of other rule r'. The certainties of $lhs(r)$ and $rhs(r)$ are denoted by $cer(lhs(r))$ and $cer(rhs(r))$. $cer(r)$ is the certainty of r, i.e. ξ which is defined above.

2.2 Semantics

The certainty of a rule r indicates the "strength" (not probability) of the knowledge behind r. We first provide the basic principles of computing certainties, and then discuss the complex situations in Sect. 4. For the right-hand side, $rhs(r)$ is true with certainty $cer(lhs(r)) \times cer(r)$ if $lhs(r)$ is true with certainty $cer(lhs(r))$, where the certainty of $\psi = lhs(r)$ can be calculated as follows.

(a) In case of ψ being $e_i[B]\ op\ e_j[B]$ or $e_k[B]\ op\ b$, $cer(\psi) = 1$ if ψ is true, else $cer(\psi) = 0$.

(b) In case of ψ being $e_i \prec_B e_j$ or $e_k \prec_B \tau'$ and ψ is not $rhs(r')$ of any other rule r', $cer(\psi) = 1$ if ψ is true, else $cer(\psi) = 0$.

(c) In case of ψ being the right-hand side of some other currency rules, $cer(\psi)$ is computed by the method of the certainty of right-hand side.

(d) In case of $\psi = \psi_1 \wedge \psi_2$, where ψ_1 and ψ_2 are predicate formulas, $cer(\psi) = \min\{cer(\psi_1), cer(\psi_2)\}$.

Example 1 *Figure 1 gives an example of a data set D and a currency rule set* Σ*. How to use rules to determine data currency is shown as follows.*

(1) First we apply r_1 *to* e_1 *and* e_2*.* r_1 *means that if the employee* e_i *joined the company earlier than the other employee* e_j *and* e_i*'s rank is lower than* e_j*, then* $e_i[\mathsf{Rank}]$ *becomes out-of-date earlier than* $e_j[\mathsf{Rank}]$*, and the certainty of* r_1 *(i.e.* $cer(r_1)$*) is 0.9. Since* $lhs(r_1)$ *is true on* e_2 *and* e_1*,* $cer(lhs(r_1)) = 1$*, and thus* $cer(e_2 \prec_{\mathsf{Rank}} e_1) = cer(lhs(r_1)) \times cer(r_1) = 0.9$*.*

(2) r_2 *means that if* $e_i[\mathsf{Rank}]$ *becomes out-of-date earlier than* $e_j[\mathsf{Rank}]$*, then* $e_i[\mathsf{Salary}]$ *becomes out-of-date earlier than* $e_j[\mathsf{Salary}]$ *with certainty 0.8. Since* $cer(e_2 \prec_{\mathsf{Rank}} e_1) = 0.9$ *from (1),* $cer(e_2 \prec_{\mathsf{Salary}} e_1) = 0.9 \times 0.8 = 0.72$*.*

EID	Name	Job	City	EntryYear	Rank	Salary
e_1	Alice	R&D	Beijing	2010	5	6000$
e_2	Bob	QA	Shanghai	2009	4	4000$

(a) Data set

r_1	$\forall e_i, e_j(e_i[Rank] < e_j[Rank] \wedge e_i[EntryYear] < e_j[EntryYear] \rightarrow e_i \prec_{Rank} e_j, 0.9)$
r_2	$\forall e_i, e_j(e_i \prec_{Rank} e_j \rightarrow e_i \prec_{Salary} e_j, 0.8)$
r_3	$\forall e(e[Job] = R\&D \wedge e[Rank] = 5 \wedge e[EntryYear] < 2012 \rightarrow e \prec_{Rank} 2013, 1)$
r_4	$\forall e\,(e \prec_{Rank} 2013 \rightarrow e \prec_{Salary} 2013, 0.8)$

(b) Currency rules

Fig. 1 An example of employees' information and the currency rules.

(3) r_3 means that e[Rank] *becomes out-of-date earlier than* 2013 *with certainty 1 if e's rank is 5, e's job is R&D and e entered the company earlier than 2012. Since $lhs(r_3)$ is true on e_1, $cer(lhs(r_3)) = 1$, and $cer(e_1 \prec_{Rank} 2013) = 1 \times 1 = 1$.*

(4) r_4 means that e[Salary] *becomes out-of-date earlier than* 2013 *with certainty 0.8 if e*[Rank] *becomes out-of-date earlier than* 2013. *Since $cer(e_1 \prec_{\text{Rank}} 2013) = 1$ from (3), $cer(e_1 \prec_{\text{Salary}} 2013) = 1 \times 0.8 = 0.8$.*

(5) Assume a data item is considered to be obsolete if the data item becomes out-of-date earlier than 2014. *The currency of data items e_1*[Rank]*, e_2*[Rank]*, e_1*[Salary] *and e_2*[Salary] *can be determined as follows.*

(a) e_1[Rank] *is obsolete with certainty 1 (step (3)).*

(b) e_1[Salary] *is obsolete with certainty 0.8 (step (4)).*

(c) $e_2 \prec_{\text{Rank}} 2013$ is the conjunction of $e_2 \prec_{\text{Rank}} e_1$ and $e_1 \prec_{\text{Rank}} 2013$, thus $cer(e_2 \prec_{\text{Rank}} 2013) = \min\{0.9, 1\} = 0.9$, i.e. e_2[Rank] *is obsolete with certainty 0.9.*

(d) Similarly to (c), e_2[Salary] *is obsolete with certainty 0.72.*

2.3 Discussion of the Certainties

2.3.1 Calculating the Certainties

Please note that the certainty defined in this paper is the *strength* of the knowledge rather than the conditional probability. By using conditional probability, the certainty is calculated by multiplication. However, we need to make a lot of assumptions, such as independencies and conditional independencies of the attributes, but these assumptions may not be true in real world.

For the left-hand side (i.e. conditions) of each rule, we choose the certainty of the "weakest" predicate as the strength of left-hand side. We choose minimum rather than average because we want to use the lower bound of the certainty to do the calculations and thus we don't overestimate the certainty that the data is out of date.

Moreover, the knowledge from the users is not omnipotent, thus there might be different currency rules indicating that a same conclusion has different certainties. For the situation that multiple rules can deduce the same conclusion, the strategy is that we always choose to believe the stronger conclusion which comes from a stronger knowledge. For example, a rule indicates that $e_i[A_j]$ is obsolete with certainty 0.6, but another stronger rule indicates that $e_i[A_j]$ is obsolete with certainty 0.9. Then, the certainty that $e_i[A_j]$ is obsolete is $\max\{0.6, 0.9\} = 0.9$. This is reasonable because different conditions could be the different evidences (i.e. the left-hand side) of different rules, thus stronger rules should lead to a more reliable conclusion. The only thing that should be attention is that we do not allow the same condition deduces the same conclusion with multiple certainties, because we cannot get different certainties of one conclusion from the same evidences.

2.3.2 Setting the Certainties of the Rules

The certainties of each rule can be determined manually by domain experts. However, rules and their certainties can also be obtained based on machine learning. For example, we can learn rules by the following three steps.

(1) Sample from the data set D to be processed and mark the timestamps to get a training set D_{tr}.

(2) Find the high frequency patterns from D_{tr}. Use the high frequency patterns to construct the left-hand side of the rules, and enumerate all the $e_i \prec_A e_j$ and $e \prec_A \tau$ to be the right-hand side. Thus, a set of candidate rules can be constructed. For example, two pattern can be in the form of $f_1 = \{A_1\ op_1\ a_1, A_2\ op_2\ a_2\}$ and $f_2 = \{A_3\ op_3\ a_3\}$, where $op_1, op_2, op_3 \in \{=, \neq, <, >, \leq, \geq\}$. Using the two patterns, we can get a set of rules in form of $\forall e_i, e_j(e_i[A_1]\ op_1\ a_1 \wedge e_i[A_2]\ op_2\ a_2 \wedge e_j[A_3]\ op_3\ a_3) \rightarrow e_i \prec_A e_j$, $\forall e(e[A_1]\ op_1\ a_1 \wedge e[A_2]\ op_2\ a_2) \rightarrow e \prec_A \tau$, and $\forall e(e[A_3]\ op_3\ a_3) \rightarrow e \prec_A \tau$ for all the available attribute A and timestamp τ.

(3) Based on the training data set, the frequency at which the rules are established is calculated as certainties of the rules. A threshold can be used to discard the rules with low certainties.

Two problems of the machine learning method that need to be noticed are the problem of overfitting and rule coverage. The first one can be solved by the traditional methods for reducing overfitting, and the second one can be solved by a well-designed D_{tr} sampling method. For example, we can adopt stratified sampling to make sure that D_{tr} can cover as full as possible the various features of the records in D. The discussion of the sampling algorithm is beyond the scope of this article, so it will not be repeated here.

2.4 Soundness

Now we provide an inference system IS for inferring data currency, consisting of the following inference rules IR1 and IR2, where X and Y are valid predicate formulas, ω, ω_i, ω'

are predicates in form of $\alpha \prec_A \beta$, α and β can be either a tuple or a time point.

IR1. If $\alpha \prec_A \beta$ is a predicate of X, then $(X \rightarrow \alpha \prec_A \beta, 1)$.

IR2. If $Y = \omega_1 \wedge, \ldots, \wedge \omega_k$, $(Y \rightarrow \omega', \xi')$, and either $(X \rightarrow \omega_i, \xi_i)$ or $\omega_i \in X$ for all $1 \leq i \leq k$, then $(X \rightarrow \omega', \min_{(X \rightarrow \omega_i, \xi_i)} \{\xi_i\} \times \xi')$.

For example, the Σ shown in Fig. 1 can deduce the following two rules using inferences rules in IS: (a) $\forall e_i, e_j((e_i[\mathsf{Rank}] < e_j[\mathsf{Rank}] \wedge e_i[\mathsf{EntryYear}] < e_j[\mathsf{EntryYear}]) \rightarrow e_i \prec_{\mathsf{Salary}} e_j, 0.72)$; (b) $\forall e((e[\mathsf{Job}] = \mathsf{R\&D} \wedge e[\mathsf{Rank}] = 5 \wedge e[\mathsf{EntryYear}] < 2012) \rightarrow e \prec_{\mathsf{Salary}} 2013, 0.8)$.

Now we briefly analyze the soundness of IS.

Definition 1 *Let D be a data set and r be a rule. D satisfies r, denoted by $D \models r$, if the following conditions are true.*

(1) In case of r being $\forall e_i, e_j(\psi \rightarrow e_i \prec_A e_j, cer(r))$, a conclusion $cer(e_i \prec_A e_j) = cer(\psi) \times cer(r)$ can be deduced if ψ is true on e_i, e_j with certainty $cer(\psi)$.

(2) In case of r being $\forall e(\psi \rightarrow e \prec_A \tau, cer(r))$ (or $\forall e(\psi \rightarrow \tau \prec_A e, cer(r))$), a conclusion that $cer(e \prec_A \tau)$ (or $cer(\tau \prec_A e)$) equals to $cer(\psi) \times cer(r)$ can be deduced if ψ is true on e with certainty $cer(\psi)$.

Definition 2 *Let D be a data set, Σ be a currency rule set, and r be a rule. D satisfies Σ, denoted by $D \models \Sigma$, if $D \models r$ for all $r \in \Sigma$. Σ entails r, denoted by $\Sigma \models r$, if $D \models \Sigma$ implies $D \models r$ for all D.*

Definition 3 *Let* IR *be an inference rule and $\Sigma \vdash_{\mathsf{IR}} r$ represent that a rule r can be deduced by a rule set Σ using* IR. IS *be inference system and $\Sigma \vdash_{\mathsf{IS}} r$ represent that r can be deduced by Σ using the inference rules in* IS. IR *is sound means that $\Sigma \models r$ if $\Sigma \vdash_{\mathsf{IR}} r$ for any Σ and r.* IS *is sound means that $\Sigma \models r$ if $\Sigma \vdash_{\mathsf{IS}} r$ for any Σ and r.*

Theorem 1 IS *is sound.*

Proof 1 *(1)* IR1 *is reflexivity, and its soundness is self-explanatory.*

(2) Let D be a data set, r be the rule $(X \rightarrow \omega', \min\{\xi_1, \ldots, \xi_k\} \times \xi')$. Assume that $D \models \Sigma$ and $\Sigma \vdash_{\mathsf{IR2}} r$. Let S be a subset of D. $S = \{t\} \subseteq D$ if all the predicates in X, Y and ω' are in form of $e \prec_A \tau$, otherwise $S = \{t_i, t_j\} \subseteq D$. If S makes X true with certainty $cer(X)$, then ω_i is true with certainty $cer(X) \times \xi_i$ from the semantic of rule $(X \rightarrow \omega_i, \xi_i)$ (for the j such that $\omega_j \in X$, ξ_j can be considered as 1). Thus, Y is true with certainty $\min_{1 \leq i \leq k}\{cer(X) \times \xi_i\} = cer(X) \times \min_{1 \leq i \leq k}\{\xi_i\}$ since $Y = \omega_1 \wedge, \ldots, \wedge \omega_k$. Hence, ω' is true with certainty $cer(X) \times \min_{1 \leq i \leq k}\{\xi_i\} \times \xi'$ from rule $(Y \rightarrow \omega', \xi')$. According to Definition 1, we have $D \models r$. Therefore, $D \models \Sigma$ implies $D \models r$. According to Definition 2, we have $\Sigma \models r$, that is, $\Sigma \models (X \rightarrow \omega', \min\{\xi_1, \ldots, \xi_k\} \times \xi')$. According to Definition 3, IR2 *is sound.*

In summary, all the inference rules in IS *are sound, thus $\Sigma \models r$ if $\Sigma \vdash_{\mathsf{IS}} r$, that is,* IS *is sound.*

3. Definitions of Data Currency

For any data set D, we assume that D corresponds to a *valid time threshold* θ indicating the lower bound of the valid time of D, that is, all the data items in D must be up-to-date at time point θ if D's currency is perfect, and a data item $e[A]$ should be considered as an *obsolete* data item which negatively influences the currency of D, if $e[A]$'s valid time is *earlier* than θ, i.e. $VT(e[A]) < \theta$. Therefore, the currency of a data item can be defined as follows.

Given an data item $e[A]$ and a time point τ, let $\tau \preceq_A e$ denote that the valid time $VT(e[A])$ is later than or equal to τ, then $e[A]$ is up-to-date if $\theta \preceq_A e$ is true. Thus, we define the *currency of data item* $e[A]$, denoted by $cur(e[A])$, as the certainty that $VT(e[A])$ is later than or equal to τ. Since both $\theta \prec_A e$ and $e \prec_A \theta$ can be possibly deduced from the currency rules, we define $cur(e[A])$ as the higher one of $cer(\theta \prec_A e)$ and $1 - cer(e \prec_A \theta)$ according to the certainty computation method, that is, if $0 < cer(\theta \prec_A e), 1 - cer(e \prec_A \theta) < 1$, we have

$$cur(e[A]) = \max\{cer(\theta \prec_A e), 1 - cer(e \prec_A \theta)\}.$$

However, there are two special circumstances that need attention.

(1) $cer(\theta \prec_A e)$ or $1 - cer(e \prec_A \theta)$ will not be used for the calculation of currency if one of $cer(e \prec_A \theta)$ or $cer(e \prec_A \theta)$ cannot be deduced by rules in Σ, that is, $cur(e[A]) = cer(\theta \prec_A e)$ if $cer(e \prec_A \theta)$ is unknown, and $cur(e[A]) = 1 - cer(e \prec_A \theta)$ if $cer(\theta \prec_A e)$ is unknown.

(2) $cur(e[A])$ is unknown if both $cer(\theta \prec_A e)$ and $1 - cer(e \prec_A \theta)$ cannot be deduced from the rules in Σ.

After the currencies of all the data items are computed, a currency threshold δ (e.g. $\delta = 0.5$) can be used to detect the obsolete value. In other words, we can determine whether the currency of a data item $e[A]$ is *unacceptable*, and $e[A]$ should be repaired by the up-to-date value of $e[A]$ if $cur(e[A]) < \delta$.

Based on the currency of data items, the currency of tuples and data set can be easily computed by averaging the currency of data items.

4. Algorithms for Determining Data Currency

4.1 Domination Graph

Certainty of a conclusion can be influenced by other conclusions. For example, a rule $r = \forall e_i, e_j(e_i \prec_{\mathsf{Rank}} e_j \rightarrow e_i \prec_{\mathsf{Salary}} e_j, 0.8)$ indicates $cer(e_i \prec_{\mathsf{Salary}} e_j)$ is influenced by $cer(e_i \prec_{\mathsf{Rank}} e_j)$. Thus, we naturally expect that the calculation of $cer(e_i \prec_{\mathsf{Rank}} e_j)$ can be finished before we start to calculate $cer(e_i \prec_{\mathsf{Salary}} e_j)$. Otherwise, once $cer(e_i \prec_{\mathsf{Rank}} e_j)$ changes, $cer(e_i \prec_{\mathsf{Salary}} e_j)$ needs to be recalculated. Therefore, we need to find a "reasonable order" to ensure that $cer(\alpha \prec_A \beta)$ is computed after that the computations of the conclusions influencing $\alpha \prec_A \beta$.

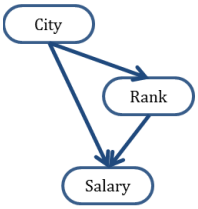


Fig. 2 Domination graph

We assume that $\prec_A$ is transferable. For any conclusions ω and ω', $cer(\omega)$ is influenced by $cer(\omega')$ in two circumstances: (a) there exists a rule r that $\omega = rhs(r)$ and $\omega' \in lhs(r)$, and in this case we say that ω' *directly dominates* ω according to Σ; (b) there is a sequence $\omega_0, \ldots, \omega_k$ ($k \geq 2$), where $\omega_0 = \omega'$, $\omega_k = \omega$, and ω_i directly dominates ω_{i+1} for any $0 \leq i \leq k-1$, and in this case we say that ω' *indirectly dominates* ω according to Σ.

However, the complex domination relationships may make the computation fall into a infinite loop. To avoid this, we simplify the domination relationships, and focus on the domination of the attributes rather than conclusions. We define *Domination Graph* (DG for short) as follows. Each node v_i in DG represent the attribute $A_i \in R$. An arc from v_i to v_j in DG means that there exists $r \in \Sigma$ such that $\prec_{A_i}$ appears in $lhs(r)$ and $\prec_{A_j}$ appears in $rhs(r)$. Arcs in DG represent direct domination relationships between attributes, and paths which contains more than two nodes represent indirect domination relationships.

Example 2 *Consider* Σ *shown by Fig. 1. To illustrate a more general case, we expand* Σ *by adding two rules* $r_5 = \forall e_i, e_j(e_i \prec_{\mathsf{City}} e_j \rightarrow e_i \prec_{\mathsf{Rank}} e_j, 0.8)$ *and* $r_6 = \forall e_i, e_j(e_i \prec_{\mathsf{City}} e_j \rightarrow e_i \prec_{\mathsf{Salary}} e_j, 0.8)$. *The coressponding DG of* Σ *is shown by Fig. 2. According to the DG, we should first compute the certainties of all the conclusions on attribute* City*, then compute the certainties of all the conclusions on* Rank*, and finally compute the certainties of all the conclusions on* Salary.

If a DG is acyclic, ordering attributes can be done by topological sorting DG, that is, we iteratively choose a node whose in-degree is 0, delete the node and its adjacent arcs, till all the nodes are deleted. The order of deleting nodes is the *"reasonable order"* for certainty computation, and we just need to determine currency of the data items column by column according to the order. In other words, we evaluate the currency attribute by attribute according to the topological order.

Please note that the circles in a DG also indicate cyclic rules. In this case, the rule set Σ should be modified to break the circles. That is, we need to delete some arcs of DG. The best way is to complete the removal of the edges by a domain expert. As an aid, we can automatically provide some suggestions for domain experts. Naturally, we want the deleted arcs as few as possible. This problem of breaking circles in a directed graph by removing minimum arcs is called Minimum Feedback Arc Set problem (MFAS). It is proved to be NP-hard [11], and widely studied [12], so we can choose any one of the existed algorithms to solve the problem. A common concern is whether breaking a circle will cause the deletion of important arcs. In order to reduce the negative impact as much as possible, we can approximate the importance of the arc with the maximum strength of the rules involved in the DG arc. This is because we always take the maximum certainty based on the calculation method of certainty.

4.2 Currency Determination of Data Items

Now we discuss how to compute $cer(e \prec_A \theta)$ for all the data items $e[A]$. There are three types of conclusions on attribute A, that is, $e_i \prec_A e_j$, $e_k \prec_A \tau$, and $\tau \prec_A e$. The partial orders of tuples and time points indicated by the three types of conclusions can be represented by a directed graph, named *currency graph*.

Definition 4 *Let* A *be an attribute,* D *be a data set,* Σ *be a set of currency rules,* Γ *be the set of all the possible time points appearing in* Σ*, and* θ *be the valid time threshold of* D*. The currency graph of* A*, denoted by* G_A*, is defined as follows.*

(a) There are two types of nodes, i.e., timestamp nodes and item nodes. An item node corresponds to a tuple in D*, and a timestamp node represents a time point in* $\Gamma \cup \{\theta\}$.

(b) For each $r \in \Sigma$*, if* $\exists \alpha, \beta \in D \cup \Gamma \cup \{\theta\}$ *such that* $cer(lhs(r)) > 0$ *and* $rhs(r) = \alpha \prec_A \beta$*, then we say that* $\alpha \prec_A \beta$ *can be directly deduced by* r*, and an arc* (β, α) *from node* β *to node* α *exists in* G_A.

(c) The weight of the directed edge (β, α)*, denoted by* $weight(\beta, \alpha)$*, is* $\max_{r \in \Sigma}\{cer(lhs(r)) \times cer(r) | \alpha \prec_A \beta$ *can be directly deduced by* $r\}$.

Currency determination of the data items on column A can be divided into two phases. (1) Use rules in Σ to build G_A. (2) Deduce the certainties of $\theta \prec_A e$ and $e \prec_A \theta$ using G_A. Next we will discuss the two phases in details.

4.2.1 Building Currency Graph

The detailed description of using Σ to build G_A on a given column A of a data set D is in Algorithm 1. First, for each tuple in D, a corresponding node is added to the node set of G_A. Then, Σ is scanned once to find whether there exist a unused rule r which can directly deduce $\alpha \prec_A \beta$, if so, use $cer_r(\alpha \prec_A \beta)$ to update arc (β, α).

Algorithm 2 shows how to update (β, α) using weight c. In step 1 to 2, the algorithm checks whether there are nodes corresponding to β and α. If not, new nodes are added to G_A. When invoking *addNode*(), if the node v_i to be added to V represents a time point τ_i, a set of new arcs with weight 1 should be added to G_A. More precisely, for each timstamp node v_j which corresponds to a time point τ_j, add (v_i, v_j) to

Algorithm 1 Build the currency graph of A

Input: D, Σ, A
Output: $G_A = (V, E)$
1: $V = \emptyset, E = \emptyset$
2: **for each** $e_i \in D$ **do**
3: add a new node v_i to V
4: **for each** $e_i \in D$ **do**
5: **if** $\exists r$ and τ s.t. $e_i \prec_A \tau$ (or $\tau \prec_A e_i$) can be directly deduced by r **then**
6: $UpdArc(\ G_A, \tau, e_i, cer(r) \times cer(lhs(r))\ )$ (or $UpdArc(\ G_A, e_i, \tau, cer(r) \times cer(lhs(r))\ )$)
7: **for each** $e_i, e_j \in D$ **do**
8: **if** $\exists\ r$ s.t. $e_i \prec_A e_j$ can be directly deduced by r **then**
9: $UpdArc(\ G_A, e_j, e_i, cer(r) \times cer(lhs(r))\ )$

Algorithm 2 $UpdArc(\ G_A, \beta, \alpha, c\ )$

1: **if** $\alpha \notin V$ **then** $addNode(v_\alpha)$
2: **if** $\beta \notin V$ **then** $addNode(v_\beta)$
3: **if** $(\beta, \alpha) \notin E$ **then** add (β, α) to E, $weight(\beta, \alpha) = c$,
4: **else** $weight(\beta, \alpha) = \max\{weight(\beta, \alpha), c\}$

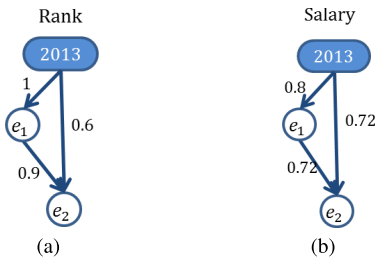


Fig. 3 The currency graph of Rank and Salary

G_A if τ_i is later than τ_j, otherwise add (v_j, v_i) to G_A. In step 3 to step 4, if (β, α) is not in G_A, add (β, α) with weight c to G_A, otherwise, change $weight(\beta, \alpha)$ to $\max\{weight(\beta, \alpha), c\}$.

The time complexity of Algorithm 1 is $O(|V|^2|\Sigma|)$, where $|V|$ is the number of nodes in G_A and $|V| \leq |D| + |\Gamma| + 1$. If $|\Gamma| \ll |D|$, then the time complexity is $O(|D|^2|\Sigma|)$.

Example 3 *Consider* Σ *shown by Fig. 1. To illustrate a more general case, We expand* Σ *by adding a rule* $r_7 = \forall e(e[\mathsf{City}] = \mathsf{Beijing} \rightarrow e \prec_{\mathsf{Rank}} 2013, 0.6)$ *to* Σ. *By applying expanded* Σ *to* D *shown by Fig. 1 (a), the currency graphs of attribute* Rank *and* Salary, *i.e.* G_{Rank} *and* G_{Salary}, *are shown by Fig. 3. Both of the graphs have one timestamp node representing 2013 and two item nodes representing the data items corresponding to tuple* e_1 *and* e_2. *We use* (e_1, e_2) *in* G_{Salary} *as an example to illustrate how to compute the weight of an arc. Using* $r_2 \in \Sigma$, *we can directly deduce* $e_2 \prec_{\mathsf{Salary}} e_1$, *thus* $weight(e_1, e_2)$ *in* G_{Salary} *is* $cer(e_1 \prec_{\mathsf{Rank}} e_2) \times cer(r_2) = 0.9 \times 0.8 = 0.72$.

Since conclusions may influence each other, $cer(\alpha \prec_A \beta)$ does not necessarily equal to $weight(\alpha, \beta)$. We still need to do some inference on the currency graphs to get the final currency of each data item.

4.2.2 Inference on Currency Graph

We build G_As for each attribute A in turn in the reasonable order. After that, the certainty of each conclusion can be calculated. $\alpha \prec_A \beta$ can be obtained as a conclusion through two ways, that is, (a) directly deduced from a rule, or (b) indirectly deduced by the rules obtained by Σ using IS. The conclusions which can be directly deduced by rules have already been represented by the arcs in G_A, so in this phase, we just need to use these conclusions to do the remnant deductions through IS. Theorem 2 explains the basis of how to do the deductions.

Theorem 2 *In the currency graphs built in the reasonable order, there must be at least one path from* β *to* α *in* G_A *if* $cer(\alpha \prec_A \beta) \neq 0$.

Proof 2 *If* $\alpha \prec_A \beta$ *is directly deduced, then there must be an arc from* β *to* α. *If* $\alpha \prec_A \beta$ *is indirectly deduced, there must be a sequence* $\alpha \prec_A \gamma_1, \gamma_1 \prec_A \gamma_2, \cdots, \gamma_k \prec_A \beta$ *that* $\alpha \prec_A \beta$ *can be deduced according to the transitivity of* $\prec_A$. *Thus, there must be at least one path from* β *to* α.

Consequently, to compute $cer(\alpha \prec_A \beta)$, we just need to find out the paths from β to α. The method is as follows.

(a) First we deal with the case that there is only one path p from β to α in G_A. Without loss of generality, let $p = (\beta, \gamma_k, \cdots, \gamma_1, \alpha)$. p represents a sequence $\alpha \prec_A \gamma_1, \gamma_1 \prec_A \gamma_2, \cdots, \gamma_k \prec_A \beta$, where (β, γ_k), (γ_{i-1}, γ_i), (γ_1, α) are arcs in G_A. According to the transitivity of $\prec_A$, we have $(\bigwedge_{1 \leq i \leq k} (\gamma_i \prec_A \gamma_j) \wedge (\alpha \prec_A \gamma_1) \wedge (\gamma_k \prec_A \beta) \rightarrow \alpha \prec_A \beta, 1)$. Because p is the unique path from β to α, we have $cer(\gamma_i \prec_A \gamma_{i+1}) = weight(\gamma_i, \gamma_{i+1})$ and $cer(\bigwedge_{1 \leq i \leq k} (\gamma_i \prec_A \gamma_j) \wedge (\alpha \prec_A \gamma_1) \wedge (\gamma_k \prec_A \beta)) = \min_{\varsigma \in p} weight(\varsigma)$, where ς represents an arc in p. Thus,

$$cer(\alpha \prec_A \beta) = 1 \times \max\{0, \min_{\varsigma \in p} weight(\varsigma)\}. \quad (1)$$

(b) Then we deal with the case that there are more than one path from β to α. Let $cer_p(\alpha \prec_A \beta)$ be the certainty of $\alpha \prec_A \beta$ computed by path p using Formula (1). According to Sect. 2.1, $cer(\alpha \prec_A \beta) = \max_{p \in Path(\beta, \alpha)} \{cer_p(\alpha \prec_A \beta)\}$, where $Path(\beta, \alpha)$ consists of all the paths from β to α. More precisely,

$$\begin{aligned} cer(\alpha \prec_A \beta) &= \max_{p \in Path(\beta, \alpha)} \{cer_p(\alpha \prec_A \beta)\} \\ &= \max_{p \in Path(\beta, \alpha)} \{\max\{0, \min_{\varsigma \in p} weight(\varsigma)\}\}. \end{aligned}$$

Definition 5 *(Length of the Path) Given a currency graph* G_A *and a path* p *in* G_A, *the length of* p, *denoted by* $len(p)$, *is defined as the minimum arc weight in* p, *that is,* $len(p) = \min_{\varsigma \in p} weight(\varsigma)$.

According to Definition 5, the computation of $cer(\alpha \prec_A$

β) can be reformulated as the problem of finding longest path in G_A. For example, in Fig. 3 (a), there are two paths from 2013 to e_2 in G_{Rank}, that is, $(2013, e_2)$ and $(2013, e_1, e_2)$. The length of the two paths are 0.6 and 0.9 respectively. Thus, $cer(e_2 \prec_{\mathsf{Rank}} 2013) = 0.9$.

4.2.3 Calculating Currency of Data Items

As defined in Sect. 3, $cur(e[A])$ is $\max\{cer(\theta \prec_A e), 1 - cer(e \prec_A \theta)\}$. $cer(\theta \prec_A e)$ equals to the lengths of longest path from e to θ, and similar to $cer(e \prec_A \theta)$. By slightly modifying Dijkstra's algorithm, the longest path's length can be computed in $O(|V| \log |V|)$ time, where $|V|$ is the node number of G_A. If the node representing θ is not in G_A, then we invoke $addNode(\theta)$ to add a new node θ to the currency graph, thus $|V|$ equals to $|D| + |\Gamma \cup \{\theta\}|$. If $|\Gamma| \ll |D|$, then the time complexity of determine the currency of all the data items on attribute A is $O(|D|^2 \log |D|)$.

5. Experimental Study

5.1 Experiment Settings

We conduct the experiments on two real-life data sets. The codes are written in C++ and run on a machine with 3.10GHz Intel CPU and 4GB of RAM.

Experimental Data. We adopt two real-life data sets.

NBA[†]. These tables consist of the history information of players, teams and associated arenas of the teams. It consists 19197 tuples with schema (EId, Player, Age, League, Game, Points, TeamName, TeamAbbreviation, TeamCoaches, TeamYears, TeamGame, TeamWin, TeamLoss, TeamChamp, Arena, Location, Capacity). The valid time threshold is 2014. We manually derive 73 currency rules. Some examples of the rules can be found in Table 1. To generate the golden standard of the obsoleteness of data items, we also manually collect the valid time for each data item.

Camera[††]. It consists 1038 tuples and 14 attributes. The schema is (Brand, Series, Model, MaxResolution, LowResolution, EffectivePixels, ZoomWide, ZoomTele, NormalFocus, MacroFocus, Storage, Weight, Dimensions, Price). The valid time threshold is 2005. There are 15 manually written currency rules. Some examples of the rules can be found in Table 2.

The characteristics of NBA and Camera are different. The regularity of NBA is strong so that it is easier to write currency rules with both high certainty and wide coverage. However, the regularity of Camera is much weaker, and the rules on Camera are either with low certainty or limited coverage. For example, about 63% of the tuples in NBA can satisfy the lhs of $\forall e(e[\text{TeamYears}] > 30 \rightarrow 2014 \prec_{\mathsf{TeamName}} e, 0.8)$, and the rule has a high certainty of 0.8. By contrast, the rule $\forall e(e[\text{MacroFocus}] < 6 \rightarrow 2005 \prec_{\mathsf{Model}} e, 0.5)$ on Camera has a coverage of 54.4% but the certainty is only 0.5. A certain rule $\forall e(e[\text{Brand}] = Agfa \rightarrow e \prec_{\mathsf{Brand}} 2001, 1)$, which is written according to the knowledge "Agfa stopped its digital camera business in September 2001", only covers less than 0.7% of the tuples in Camera. The weak regularity makes Camera's currency determination very hard and mostly relay on uncertain rules.

[†]The tables are from http://www.basketball-reference.com and http://en.wikipedia.org/wiki/List_of_National_Basketball_Association_arenas.

[††]The tables are from www.aviz.fr/Teaching2012/Datasets and Wikipedia.

5.2 Accuracy

We measure the accuracy of our method in two aspects.

(1) The absolute error (AbsE for short). AbsE is defined as $|cur_{\mathsf{real}}(D) - cur_{\mathsf{esti}}(D)|$, that is, the magnitude of the difference between the exact currency of dataset and the currency value computed by our algorithms. The value of AbsE is in $[0, 1]$, and the smaller value is better.

(2) F-measure, recall, and precision. We set the currency threshold δ to be 0.5. Since the range of currency is $[0, 1]$, $\delta = 0.5$ is coinciding with the intuition of humans, and it means that we do not have any assumptions about the distribution of obsolete data. Thus, the data items whose currency is less than 0.5 is considered to be obsolete. According to the computed currency, the data items can be classified into three groups, that is, the data items whose currency are greater than or equal to δ (EC), less than δ (EO), unknown (Unk). A data item $e[A]$ belongs to the class of Unk if no rules in Σ can be used to determine the currency of $e[A]$. Meanwhile, according to the golden standard of data sets, the data items can be classified into two groups: current i.e. up-to-date, data items (AC), and obsolete data items (AO). Recall, precision, and F-measure are defined as follows.

- Positive recall (pos-recall) is the fraction of current data items whose currency scores are greater than or equal to δ, that is, $\frac{|AC \cap EC|}{|AC|}$. Similarly, negative recall (neg-recall) is $\frac{|AO \cap EO|}{|AO|}$.
- Positive precision (pos-precision) is the fraction of the current data items in the data items with currency scores greater or equal to δ, i.e. $\frac{|AC \cap EC|}{|EC|}$. Similarly, Negative precision (neg-precision) is $\frac{|AO \cap EO|}{|EO|}$.
- F-measure is the harmonic mean of precision and recall, that is, pos-F-measure is defined as $\frac{2 \times \text{pos-recall} \times \text{pos-precision}}{\text{pos-recall} + \text{pos-precision}}$. Similarly, neg-F-measure is defined as $\frac{2 \times \text{neg-recall} \times \text{neg-precision}}{\text{neg-recall} + \text{neg-precision}}$.

The influence of the number of attributes. The number of attributes may also affect the accuracy of currency determination. For accuracy, the increase in the number of attributes mainly affects the number of required rules. The greater the number of attributes, the more rules are needed to determine the timeliness of the data set, because we have to increase the rules to cover these newly emerging attributes.

Table 1 Some examples of the currency rules for NBA.

NBA
$\forall e(e[\text{League}] = BAA \rightarrow e \prec_{\text{TeamName}} 2014, 1)$
$\forall e(e[\text{TeamYears}] > 30 \rightarrow e \prec_{\text{TeamName}} 2014, 0.8)$
$\forall e(e[\text{TeamCoaches}] = J.\ Vaughn \rightarrow e \prec_{\text{TeamCoaches}} 2014, 0.5)$
$\forall e_i, e_j(e_i[\text{TeamName}] = St.Louis\ Hawks \wedge e_j[\text{TeamName}] = Atlanta\ Hawks \rightarrow e_i \prec_{\text{TeamName}} e_j, 1)$
$\forall e_i, e_j(e_i[\text{TeamName}] = Dallas\ Chaparrals \wedge e_j[\text{TeamName}] = Texas\ Chaparrals \rightarrow e_i \prec_{\text{TeamName}} e_j, 0.6)$
$\forall e_i, e_j(e_i[\text{TeamAbbreviation}] = SAA \wedge e_j[\text{TeamAbbreviation}] = SAS\ Chaparrals \rightarrow e_i \prec_{\text{TeamAbbreviation}} e_j, 1)$

Table 2 Some examples of the currency rules for Camera.

Camera
$\forall e(e[\text{Brand}] = Agfa \rightarrow e \prec_{\text{Model}} 2001, 1)$
$\forall e(e[\text{Weight}] < 200 \rightarrow 2005 \prec_{\text{Model}} e, 0.64)$
$\forall e(e[\text{MaxResolution}] < 2500 \rightarrow e \prec_{\text{Model}} 2005, 0.9)$
$\forall e(e[\text{Brand}] = Casio \wedge e[\text{Series}] = Exilim \rightarrow 2005 \prec_{\text{Model}} e, 0.45)$
$\forall e_i, e_j(e_i[\text{Brand}] = e_j[\text{Brand}] \wedge e_i[\text{MaxResolution}] < e_j[\text{MaxResolution}] \rightarrow e_i \prec_{\text{Model}} e_j, 0.2)$
$\forall e_i, e_j(e_i[\text{Brand}] > e_j[\text{Brand}] \wedge e_i[\text{Price}] < e_j[\text{Price}] \wedge e_i[\text{MaxResolution}] < e_j[\text{MaxResolution}] \rightarrow e_i \prec_{\text{Model}} e_j, 0.8)$

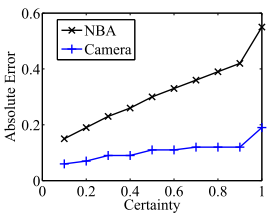


Fig. 4 AbsE under different certainties

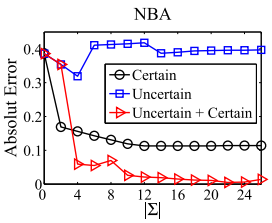


Fig. 5 AbsE on NBA

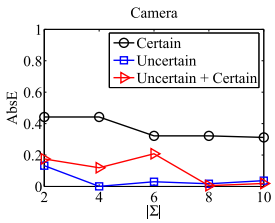


Fig. 6 AbsE on Camera

Since the number of attributes affects the number of required rules to a greater extent, we only tested the effect of the rules on the experimental results in the accuracy experiments. On the other hand, the number of attributes affects efficiency, because we need to build more currency graphs as the attributes increasing. Related experiments of efficiency are given in Sect. 5.3.

5.2.1 The Effect of Different Uncertainty Values

First, we explore the effect of different uncertainty values on the accuracy of the algorithm for a given set of uncertain rules. For NBA and Camera, we varied the certainties of all the rules. That is, for NBA, the certainties of all 73 rules is set to the same value, from 0.1 to 1, and for Camera, the certainties of all 15 rules is set to the same value, from 0.1 to 1. We compared the results of the algorithms under different certainty conditions. The results of AbsE is shown by Fig. 4. From the experimental results, we can see that in the two data sets, AbsE slightly increases with the increase of deterministic value. However, when the uncertainty value increases from 0.9 to 1, AbsE rises to a greater extent. This is because when all the rules are certain (i.e. the certainty equal to 1), the conflict between rules will significantly increase, resulting in a large amount of data items being classified as Unk. In other words, for most of the data, we cannot use rules to determine if it is out of date.

We carefully examined the reasoning results under different certainties and found the following facts. For uncertain rules, although the certainties of all rules is the same, the certainties of the conclusion may still be different in the process of reasoning. Therefore, when there are contradictory conclusions, we may still compare its certainty to judge which conclusion is stronger. However, for the determination of the rules, since the entire reasoning process is definitive, many contradictory conclusions will be introduced, and the degree of certainty of these contradictory conclusions is all 1, we cannot know which conclusion is more reliable.

From the experimental results, it can be seen that introducing uncertainty is good for effectively determining whether the data is outdated, but the experiment also reflects another fact, that is, the certainties of the rules can not be roughly set as a unified value. This is because that the certainties essentially reflects the strength of the knowledge behind the rules, and the strength of different knowledge is also different. Therefore, when using these uncertain rules, we need to be careful to set the certainties for each rule. A method worth further exploration is set the certainties of the rules with some statistics, such as the frequency of a rule being true in the training data set (mentioned in Sect. 2.3.2). Therefore, in subsequent experiments, the certainties of the rules are set to be different. This can also be seen from the examples shown in Table 1 and Table 2.

5.2.2 Absolute Error

Figures 5 and 6 show the AbsE as Σ increases. In the experiments shown by Fig. 5, we examine 1000 data items on the attribute Points, where 386 of them is up-to-date and 614 is obsolete according to the golden standard. In the experi-

ments of "Certain" (black line), Σ consists of 26 random selected *certain rules* with a certainty equal to 1. It can be seen that AbsE decreases as |Σ| increases, and stabilizes around 0.1. In the experiments of "Uncertain" (blue line), Σ consists of all the 26 uncertain rules whose certainty is less than 1. We sort these uncertain rules according to their certainty, from highest to lowest, and add them into Σ in the sorted order. Since uncertain rules can introduce wrong judgements on currency determination, the absolute error is large, and fluctuates around 0.4. However, if we use **both** certain and uncertain rules, then the result will be much better. The result is shown by the red line. In this set of experiments, we add a certain rule and an uncertain rule to Σ each time in the same order with the experiments "Certain" and "Uncertain". We found that when certain and uncertain rules are taken together, the AbsE decreases quickly. If |Σ| is larger than 10, it stabilizes at a much lower value (around 0.002).

Figure 6 show the result on Camera. There are 1038 data items, where 579 of them are up-to-date and 459 are obsolete. As discussed above, we can hardly write certain rules with high coverage. Actually, the 6 certain rules can only cover 57 tuples totally. We tried to use the automated learning method in Sect. 2.3.2 to find rules of Camera, and found that most rules cover no more than two tuples. Therefore, it is very hard to only use certain rules to do the currency determination. In Fig. 6, when we only use the certain rules to evaluate the currency of the data, AbsE always stays around 0.4. Since the real currency of Camera is 0.46, the highest AbsE is 0.54. It can be seen that when all the certain rules are used for determination, most of the data cannot be covered by the determination rule, so that the determination effect is not good. However, when using both certain and uncertain rules to evaluate data currency, most tuples are not actually evaluated at the beginning, but as the number of uncertain rules increase, more and more tuple can be covered by rules, the accuracy increases very fast.

5.2.3 F-Measure

Figures 7 and 8 show the F-measure as we add rules. The experimental parameters are same with the experiments shown by Figs. 5 and 6. Since the increase trends of neg-F-measure is similar to pos-F-measure, we only illustrate the result of pos-F-measure. It can be observed that F-measure increases as |Σ| increases. When evaluating NBA (Fig. 7), taking certain and uncertain rules together got the highest F-measure, which stabilizes around 0.94. When using only uncertain rules, the F-measure is around 0.58, and when using only certain rules, the F-measure is around 0.45.

In the results shown by Fig. 8, when |Σ| is small, using only uncertain rules got a higher F-measure, but as |Σ| increases, taking certain and uncertain rules together got a more accurate result, with a F-measure around 0.85. However, if we use only certain rules, F-measure is no more than 0.18. Low F-measure is caused by the low recall of the certain rules, which will be reflected in the experiment shown in Fig. 10.

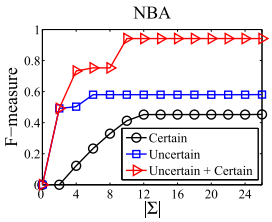


Fig. 7 F-measure on NBA

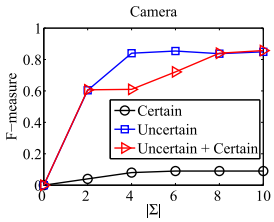


Fig. 8 F-measure on Camera

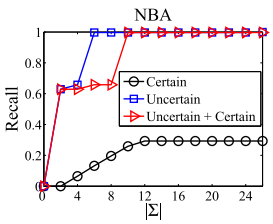


Fig. 9 Recall on NBA

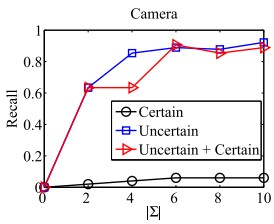


Fig. 10 Recall on Camera

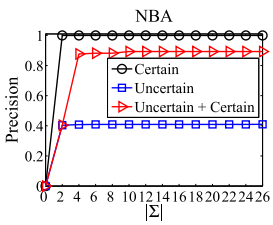


Fig. 11 Precision on NBA

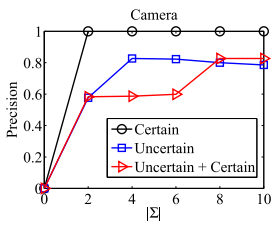


Fig. 12 Precision on Camera

5.2.4 Recall

Figures 9 and 10 show the results on recall as |Σ| increases. Similar to F-measure, we only show the result of pos-recall. The experimental parameters are same with Figs. 5 and 6. Figure 9 shows the result on NBA. Recall is closely related to the coverage of rules, so that the recall of "uncertain" increases fastest, and the recall of "certain" increases slowest. When taking uncertain rules, the final recall is around 0.99, that is, almost all the current data items are identified. When taking certain rules only, the number of data items classified to *Unk* is quite large, therefore, the final recall is much lower.

Figure 10 illustrates the result on Camera. Recall of "uncertain" is a little higher than "certain + uncertain", and the final recall is around 0.9. However, if we only use the certain rules set to evaluate the currency of Camera, the recall is only 0.05. As we have discussed above, this is because that the coverage of rules are very low and it is very hard to only use certain rules to do the currency determination. This verifies the conclusion that the uncertain rules can increase the recall.

5.2.5 Precision

Figures 11 and 12 show the results on precision as we add

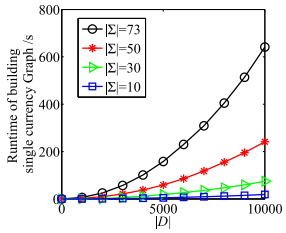


Fig. 13 Building single currency graph

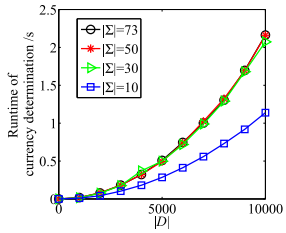


Fig. 14 Evaluating data currency

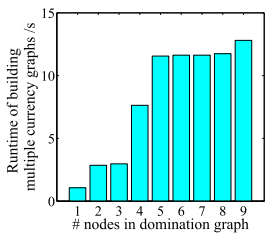


Fig. 15 Runtime of building multiple currency graphs

rules. Similar to Figs. 7 and 8, we only show the pos-precision. In the experiments of "certain", the precision is always 1. In the experiments of "uncertain", some errors are introduced, and the precision is only about 0.4. However, when certain and uncertain rules are taken together, the precision reaches higher than 0.87 after the fourth rule added to Σ. As more rules are added, precision increases slightly.

In the experiments on Camera (Fig. 12), when taking uncertain rules only, the precision increases before the fourth rule, but as more uncertain rules are added, it decreases. If we only use certain rules, the precision is always 1, because the certain rules do not misjudge. When certain and uncertain rules are taken together, the precision always increases, and finally reaches 0.827.

5.3 Efficiency

The time of building currency graph and evaluating data currency on single attribute is shown by Figs. 13 and 14. For each rule r, attribute in $rhs(r)$ is changed into Points to make sure every rule would be used. We observe quite smooth quadratic execution time for currency graph building. It is consistent with the analysis in Sect. 4.2.

The time of currency graph building on multiple attributes is shown by Fig. 15. For each attribute, we use 10 rules to evaluate 1000 data items. For example, if domination graph contains 9 attributes, then we use 90 rules to evaluate 9,000 data items. We construct a matrix $M(A)$ for each attribute A, where $M(A)_{ij} = cer(e_i \prec_A e_j)$. $M(A)$ is constructed immediately after G_A has been built, so that $cer(e_i \prec_A e_j)$ can be read in $O(1)$ time when needed. When attribute number is 9, it takes 12.8 seconds to build all the currency graphs. The building time of the posterior graph is almost unrelated to the size of previous graphs. The time of building domination graph is negligible for currency determination, since only one domination graph needs to be built, and the time complexity is $O(|\Sigma|)$. When $|\Sigma| = 100$, it took only 2 ms.

6. Related Work

Some works determine the data currency based on the data age and shelf life defined by timestamps [4], [5]. The data age is the time interval between "now" and the data's latest update. The shelf life is the maximum length of valid time interval of the data. These works determine the currency of data using the probability of the data's shelf life being longer than its age. These works assume that all data items have the exact timestamps, ages and shelf lives. The work are different from ours since we have no the assumptions. There are also some works aiming to answer queries with the most current data [6]–[8]. These works focus on how to find current data according to the global timestamps and the data updating model. These works are different from ours since our work does not focusing on query processing and has no requirement of updating model.

A more related branch of works are to determine the data currency based on rules [3], [10] without the assumption of timestamps being available. Our work is different from these works. First, those methods only deduce time orders of data, but cannot deduce the possible range of valid time of the data. Our method can deduce the possible range of valid time of the data. Second, the rules proposed by [3] and [10] cannot express uncertain semantics, which is deferent from our uncertain rules.

Another related area is temporal data models [13] and temporal knowledge discovering [14], [15]. However, their main purposes are not data repairing. Our work also related to the works of data quality evaluation [16], [17]. The methods in [17] also consider how to find the true value of a given data item when there are multiple data sources. However, since they are not aimed at obsolete data and our work are not require multiple data sources, our work is different from these works.

7. Conclusions

We proposed a class of currency rules representing the uncertain semantics of currency rules, and tried to overcome the shortages of previous work. Based on the rules, we formally defined the data currency, and gave the method of currency determination. The experimental results on real-life data showed that our algorithms are effective and efficient. In future work, we will study the following problems: (1) fixing the obsolete values to up-to-date ones, (2) determining currency in dynamic databases, (3) automatically learning currency rules effectively.

Acknowledgements

The work of this paper is supported by the National Basic Research Program of China under Grant No. 2012CB316200, the National Natural Science Foundation

of China (No. 61702220, No. 61702223, No. 61133002), the Fundamental Research Funds for the Central Universities (No. 11617303, 11617302) and Zhuhai Top Discipline Information Security.

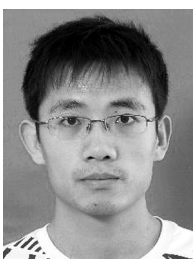
References

[1] E. Rahm and H.H. Do, "Data cleaning: Problems and current approaches," IEEE Data Engineering Bulletin, 2000.
[2] W. Eckerson, "Data warehousing special report: Data quality and the bottom line," Applications Development Trends, p.1, 2002.
[3] W. Fan, F. Geerts, and J. Wijsen, "Determining the currency of data," ACM Transactions on Database Systems (TODS), vol.37, no.4, p.25, 2012.
[4] B. Heinrich and M. Klier, "Assessing data currency — a probabilistic approach," Journal of Information Science, vol.37, no.1, pp.86–100, 2010.
[5] B. Heinrich, M. Klier, and M. Kaiser, "A procedure to develop metrics for currency and its application in crm," Journal of Data and Information Quality, vol.1, no.1, 2009.
[6] B. Yan, D. Qian, and Y. Huang, "Data currency in replicated distributed storage system," Networking, Architecture, and Storage, 2009, NAS 2009, IEEE International Conference on, New York, NY, USA, pp.57–63, IEEE, 2009.
[7] R. Akbarinia, E. Pacitti, and P. Valduriez, "Data currency in replicated dhts," Proceedings of the 2007 ACM SIGMOD international conference on Management of data, New York, NY, USA, pp.211–222, ACM, 2007.
[8] C.H.C. Leung and K. Wolfenden, "Analysis and optimisation of data currency and consistency in replicated distributed databases," The Computer Journal, vol.28, no.5, pp.518–523, 1985.
[9] H. Zhang, Y. Diao, and N. Immerman, "Recognizing patterns in streams with imprecise timestamps," Information Systems, vol.38, no.8, pp.1187–1211, 2013.
[10] W. Fan, F. Geerts, N. Tang, and W. Yu, "Conflict resolution with data currency and consistency," Journal of Data and Information Quality (JDIQ), vol.5, no.1-2, p.6, 2014.
[11] R.M. Karp, Reducibility among Combinatorial Problems, Springer, Berlin, Heidelberg, Germany, 1972.
[12] G. Even, J.S. Naor, B. Schieber, and M. Sudan, "Approximating minimum feedback sets and multicuts in directed graphs," Algorithmica, vol.20, no.2, pp.151–174, 1998.
[13] G. Ozsoyoglu and R.T. Snodgrass, "Temporal and real-time databases: A survey," IEEE Trans. Knowl. Data Eng., vol.7, no.4, pp.513–532, 1995.
[14] J.F. Roddick and M. Spiliopoulou, "A survey of temporal knowledge discovery paradigms and methods," IEEE Trans. Knowl. Data Eng., vol.14, no.4, pp.750–767, 2002.
[15] S. Laxman and P.S. Sastry, "A survey of temporal data mining," Sadhana, vol.31, no.2, pp.173–198, 2006.
[16] L. Li, J. Li, and H. Gao, "Evaluating entity-description conflict on duplicated data," Journal of Combinatorial Optimization, vol.31, no.2, pp.918–941, 2016.
[17] X.L. Dong, L. Berti-Equille, and D. Srivastava, "Integrating conflicting data: the role of source dependence," Proceedings of the VLDB Endowment, vol.2, no.1, pp.550–561, 2009.

Mohan Li received her B.S., M.S. and Ph.D degree in Computer Science from Harbin Institute of Technology (HIT), Harbin, China. She is currently a assistant professor in Guangzhou University, China. Her research interests include data quality and data cleaning.

Jianzhong Li is a professor at Harbin Institute of Technology. His research interests include data management, sensor networks, and data intensive computing.

Siyao Cheng received the PhD degree in computer science from Harbin Institute of Technology, China, in 2012. Her research interests include data management and wireless communications.



Yanbin Sun received the B.S., M.S. and Ph.D degree in Computer Science from Harbin Institute of Technology (HIT), Harbin, China. He is currently a assistant professor in Guangzhou University, China. His research interests include Information-centric networking and scalable routing.