

# Maintaining consistency of vague databases using data dependencies

An Lu <sup>\*</sup>, Wilfred Ng

*Department of Computer Science and Engineering, The Hong Kong University of Science and Technology, Clear Water Bay, Kowloon, Hong Kong, China*

---

## Abstract

Vague information is common in many database applications due to intensive data dissemination arising from different pervasive computing sources, such as the high volume data obtained from sensor networks and mobile communications. In this paper, we utilize *functional dependencies* (FDs) and *inclusion dependencies* (INDs), which are the most fundamental integrity constraints that arise in practice in relational databases, to maintain the consistency of a vague database. First, we tackle the problem, given a vague relation  $r$  and a set of FDs  $F$ , of how to obtain the “best” approximation of  $r$  with respect to  $F$  when taking into account the median membership ( $m$ ) and the imprecision membership ( $i$ ) thresholds. Using these two thresholds of a vague set, we define the notion of *mi*-overlap between vague sets and a merge operation on  $r$ . Second, we consider, given a vague database  $d$  and a set of INDs  $N$ , how to obtain the minimal possible change in value precision for  $d$ . Satisfaction of an FD in  $r$  is defined in terms of values being *mi*-overlapping while satisfaction of an IND in  $d$  is defined in terms of value-precision. We show that Lien’s and Atzeni’s axiom system is sound and complete for FDs being satisfied in vague relations and that Casanova et al.’s axiom system is sound and complete for INDs being satisfied in vague databases. Finally, we study the chase procedure  $VChase(d, F \cup N)$  as a means to maintain consistency of  $d$  with respect to  $F$  and  $N$ . Our main result is that the output of the procedure is the most object-precise approximation of  $r$  with respect to  $F$  and the minimum value-precision change of  $d$  with respect to  $N$ . The complexity of  $VChase(r, F)$  is polynomial time in the sizes of  $r$  and  $F$  whereas the complexity of  $VChase(d, F \cup N)$  is exponential.

*Key words:* Data models, Chase, Data Dependencies, Data Consistency, Uncertainty

---

## 1 Introduction

Fuzzy set theory has long ago been introduced to handle inexact and imprecise data by Zadeh’s seminal paper [1]. In fuzzy set theory, each object  $u \in U$  is assigned a single real value, called the *grade of membership*, between zero and one. (Here  $U$  is a classical set of objects, called the *universe of discourse*.) In [2], Gau et al. pointed out that the drawback of

---

\* Corresponding author. Tel.: +852 2358 8836; fax.: +852 2358 1477.

*Email addresses:* anlu@cse.ust.hk (An Lu), wilfred@cse.ust.hk (Wilfred Ng).

using the single membership value in fuzzy set theory is that the evidence for  $u \in U$  and the evidence against  $u \in U$  are in fact mixed together. Therefore, the “neutral” evidence between “for” and “against” evidence is ignored. In order to tackle this problem, Gau et al. proposed the notion of *Vague Sets* (VSs), which allow using the interval-based membership instead of using the point-based membership as in FSs. By the interval-based membership, “neutral” evidence can be represented in a vague set.

We have also shown in our previous work [3] that the interval-based membership generalization in VSs is more expressive in capturing vague data semantics. In a vague relation, each object with a *vague membership* belongs to a VS. A vague membership (also called a vague value), denoted by  $[\alpha(u), 1 - \beta(u)]$ , is a subinterval of the unit interval  $[0,1]$ , where  $0 \leq \alpha(u) \leq 1 - \beta(u) \leq 1$ . A true (or false) membership function  $\alpha(u)$  (or  $\beta(u)$ ) is a lower bound on the grade of membership of  $u$  derived from the evidence for (or against)  $u$ .

In order to compare two vague values, we define the *median membership*,  $M_m = (\alpha + 1 - \beta)/2$ , which represents the overall evidence contained in a vague value, and the *imprecision membership*,  $M_i = (1 - \beta - \alpha)$ , which represents the overall imprecision of a vague value. With  $M_m$  and  $M_i$ , we have a one-to-one correspondence between a vague value, denoted by  $[\alpha, 1 - \beta]$ , and a *mi-pair* vague value, denoted by  $\langle M_m, M_i \rangle$ , for a given object.

*Functional dependencies* (FDs) [4,5] and *inclusion dependencies* (INDs) [6,7] are known to be the most fundamental integrity constraints in relational databases. In this paper, we study the problem of maintaining consistency of vague databases. We first define the notion of an FD being satisfied in a vague relation, which can be formalized in terms of values being *mi-overlapping* rather than equal. Then, we define the notion of an IND being satisfied in a vague database, which can be formalized in terms of a comparison of values according to value precision rather than set containment.

We show that Lien’s and Atzeni’s axiom system [8,5] is sound and complete for FDs being satisfied in vague relations and Casanova et al.’s axiom system [6] is sound and complete for INDs being satisfied in vague databases. A vague relation (or a vague database) [3] is said to be consistent with respect to a set of FDs  $F$  (or a set of INDs  $N$ ) if it satisfies  $F$  (or  $N$ ). We propose a new chase procedure for a vague database  $d$ , named  $VChase(d, F \cup N)$ , to tackle the database consistency problem with respect to  $F \cup N$ .

Our first main result is that the output of the procedure is the most *object-precise* (or *O-precise* in our notation) approximation of  $r$  with respect to  $F$ . The chase procedure can be further extended to cater for inconsistency arising from a mixed set of  $F$  and  $N$  over vague databases. Another main result is that the complexity of  $VChase(d, F \cup N)$  is EXPTIME-complete (cf. the implication problem of FDs and INDs is undecidable in relation databases [6]) and the chase can be implemented by simply chasing against INDs first and then FDs to obtain the same consistent vague database as output. The above results are fundamental to those applications that involve vague data or fuzzy data as a special case, since maintaining consistency is an important means to guarantee the quality of vague data in practical use.

Here we give a motivating example concerning only FDs over a vague relation for simplicity in illustration. A more detailed but slightly complex example relating to both FDs and INDs over a vague database is presented in Section 7.

Consider a vague relation schema  $R = \{S, T\}$ , where  $S$  stands for the evidence of a sensor ID and  $T$  stands for the temperature monitored by a sensor. Here  $S$  and  $T$  are vague concepts, and their values are all represented by VSs. Suppose the attributes  $S$  and  $T$  share the common universe of discourse,  $U = \{0, 1, \dots, 10\}$ . A vague relation  $r_1$  over  $R$  is shown in Table 1, where the attributes  $S$  and  $T$  are vague. The VS  $\langle 0.8, 0.1 \rangle / 0$  means the evidence for “the sensor ID is 0” is 0.8 and the imprecision for it is 0.1. The median membership threshold  $C$  and the imprecision membership threshold  $I$  are collectively called the *mi-thresholds*. For simplicity, we only show the elements in the values of  $S$  and  $T$  that satisfy the *mi-thresholds*. Intuitively, this means that the elements in the relation all have strong evidence relative to the thresholds. We say that two VSs *mi-overlap* when they have at least one common object which satisfies the *mi-thresholds* (that is,  $0.8 \geq C$  and  $0.1 \leq I$  in this example). We regard two *mi-overlapping* VSs as “similar” to each other and extend the classical FD concept to vague relations based on this notion of similarity. Suppose that the FD  $S \rightarrow_{C,I} T$  is specified as a constraint, meaning that the same sensor  $S$  reads the same temperature  $T$  in a “vague sense”.

We assume a vague relation  $r_1$  over  $R$ , where the current temperature may be obtained from different sensors. Thus, at any given time the information may be inconsistent. It can be verified that  $r_1$  satisfies  $S \rightarrow_{C,I} T$  and is consistent. Suppose later a vague tuple was inserted into  $r_1$ , we have the vague relation  $r_2$  shown in Table 2. It can be verified that  $r_2$  does not satisfy  $S \rightarrow_{C,I} T$  and is inconsistent, since the evidence of  $S$  shows that the two tuples have the common object 0 *mi-overlapped*, but the values of  $T$  do not have a common object and thus do not *mi-overlap*. The vague relation  $r_2$  can be approximated by the less  $O$ -precise relation  $r_3$ , shown in Table 3. It can be verified that  $r_3$  satisfies  $S \rightarrow_{C,I} T$  and is consistent. The vague relation  $r_3$  (one tuple) is in fact the most  $O$ -precise approximation of  $r_2$ . The transformation from  $r_2$  to  $r_3$  is based on the *VChase* procedure introduced later.

Table 1

Sensor relation  $r_1$

S	T
$\langle 0.8, 0.1 \rangle / 0$	$\langle 0.9, 0 \rangle / 0$

Table 2

Sensor relation  $r_2$

S	T
$\langle 0.8, 0.1 \rangle / 0$	$\langle 0.9, 0 \rangle / 0$
$\langle 0.9, 0.2 \rangle / 0$	$\langle 0.8, 0.1 \rangle / 1$

Table 3

Sensor relation  $r_3$

S	T
$\langle 0.9, 0.1 \rangle / 0$	$\langle 0.9, 0 \rangle / 0 + \langle 0.8, 0.1 \rangle / 1$

We define the merge operation which replaces each attribute value in  $r$  by the *mi-union* of all attribute values with respect to the same reflexive and transitive closure under *mi-overlap*. This leads to a partial order on merged vague relations and the notion of a vague relation being less  $O$ -precise than another vague relation. This partial order induces a lattice

on the set of merged vague relations, which we denote by  $MERGE(R)$ , based on *object-equivalence* (*O*-equivalence for short) classes.

The main contributions of this paper are as follows.

- We develop the notions of median membership and imprecision membership in vague sets to capture uncertain information and to maintain consistency of vague data.
- We define a partial order on merged vague relations which induces a lattice based on *O*-equivalence classes. We also define a partial order related to the precision of vague values, which induces a complete semi-lattice within each *O*-equivalence class.
- We extend the satisfaction of an FD in a vague relation in terms of values being *mi*-overlapping rather than equal and show that Lien’s and Atzeni’s axiom system is sound and complete for FDs being satisfied in vague relations.
- We extend the satisfaction of an IND in a vague database in terms of value precision rather than set containment and show that Casanova et al.’s axiom system is sound and complete for INDs being satisfied in vague databases.
- We propose the chase procedure for a vague relation  $r$  over  $R$ , named  $VChase$ , as a means of maintaining consistency of  $r$  with respect to a set of FDs  $F$ . The chase is further extended to the scope of INDs  $N$ . Our main results are that the output  $VChase(r, F)$  is the most *O*-precise approximation of  $r$  with respect to  $F$  and the output  $VChase(d, N)$  is the minimum change of value precision in  $d$  with respect to  $N$ . The complexity of  $VChase(d, F \cup N)$  is exponential but polynomial time if  $N = \emptyset$  (that is, only  $F$ ).

The rest of the paper is organized as follows. Related work is presented in Section 2. Section 3 presents some basic concepts related to *mi*-pair, which are used to enhance vague sets and their operations. In Section 4, we discuss the merge operation, based on the less *O*-precise order. In Section 5, FDs and the  $VChase$  procedure of vague relations are introduced. In Section 6, we give a semantic characterization of the  $VChase$  procedure of a vague relation, which is also consistent with respect to a set of FDs. In Section 7, we extend our  $VChase$  procedure for a mixed set of FDs and INDs. Finally, in Section 8, we offer our concluding remarks.

## 2 Related work

When dealing with uncertain information, many approaches have been applied to extend crisp relational databases, such as probability theory [9–14], fuzzy sets and possibility theory [1,15–17], hedge algebras [18], vague sets [19,3,20–22], and so on.

An early work concerning probabilistic data modeling, as a generalization of relational data model, was proposed in [9], where a relational database is transformed to a probabilistic database by a mapping function, and each tuple in a probabilistic relation is assigned a probability indicating the likelihood that the tuple belongs to the relation. In the probabilistic data model proposed later in [10], probabilities are associated with the values of the attributes, and probabilities of compound events can be obtained by the probabilities of simpler events. Moreover, missing probabilities are supported in this model. In contrast to [10], the probabilistic data model proposed in [11] does not assume that the key values for a relation are deterministic, and it attaches every tuple of a relation with associated probabilities. Within

the model proposed in [11], the concept of stochastic dependency [12], as a generalization of functional dependency, was proposed to extend the scope of normal forms to probabilistic databases. In [13,14], probabilistic interval values are adopted instead of point values in the probabilistic data model, as in many applications (such as image databases, sensor databases, and so on), interval probabilities are employed to handle the widely existent errors inherent to the probability data. There has also been research work related to various prediction models in economics [23], biology [24], traffic flow [25] and so on using Bayesian or statistical methods [26] return confidence intervals that bound the probability. Therefore, the support of interval probabilities is needed in the probabilistic data model.

The fuzzy set (FS) theory [1] was first introduced by Zadeh in 1965. The FS theory plays an important role in modeling uncertain information for many decades. In a nutshell, there are two types of FSs, type-1 (ordinary) FSs and type-2 FSs [27]. The type-2 FSs can be viewed as a generalization of type-1 FSs in the sense that the membership itself can also be fuzzy. In literature, there are several interval-based FSs which can be categorized as type-2 FSs, such as interval-valued fuzzy sets [28], intuitionistic fuzzy sets [29], grey fuzzy sets [30] and vague sets [2]. Although the notations, interpretations and motivations of these type-2 FSs may be different, they are in fact isomorphic from mathematical perspective [31–34]. They all capture the fundamental idea of evaluating membership and non-membership of an object in a given universe of discourse.

In fuzzy control systems, linguistic terms (verbal descriptions of a concept) are often used in representing the dependent relationship between one linguistic variable (a concept) to another. For example, “very hot in temperature” implies “strong cooling in air-conditioning” in a temperature auto-control system. Here “very hot” and “strong cooling” are linguistic terms, and “temperature” and “air-conditioning” are linguistic variables. Each linguistic term contains at least one *primary term*, such as “hot” or “cooling”, and some linguistic term may contain one or more *hedges*, such as “very” or “strong”. As the structure of fuzzy sets does not preserve the ordering structure of linguistic terms determined by their natural meaning, such as the natural orders “very hot > hot” and “very cold < cold”, hedge algebras were proposed in [18], which are defined on the set of linguistic terms of the linguistic variable. In this algebraic approach, primary terms are considered as generators (i.e. constants or zero-argument operations), and hedges as one-argument operations. Hedge algebra has a partially ordered relation on the set of linguistic terms, which is induced by the meaning of hedges, named as *semantically ordering relation*. Thus, hedge algebra is used in linguistic reasoning that handles linguistic terms directly [18].

Integrity constraints ensure that changes made to the database do not result in a loss of data consistency. FDs [4,5] and INDs [6,7] are commonly known as the most fundamental integrity constraints that arise in practice in crisp databases. The problem of maintaining the consistency of databases has been studied for decades [35–38]. In particular, Levene [36] introduced the notion of imprecise relations and FDs being satisfied in imprecise relations. This is in order to cater for the situation when the information is obtained from different sources and therefore may be imprecise. An imprecise set can be reviewed as a special case of an *O*-equivalent VS in our context. By applying the vague set theory, we present in this work a deeper analysis of precision and more general results of maintaining consistency compared with previous work. We apply the interval-based vague memberships, which capture the positive, neutral and negative information of objects, and extend the “equally likely

objects” assumption used in [36].

Repairing data inconsistency is known as an important research issue in the context of crisp databases such as [37,38]. Some research work for handling the data integrity problem in databases having uncertainty has also been proposed in different application areas, such as biological databases [39], sensor databases [40], and so on. Fuzzy FDs [41–43], as a generalization of classical FDs, have been proposed to handle the integrity constraints problem in fuzzy databases. In most fuzzy FDs, a fuzzy resemblance relation is used for comparing domain values instead of the equal relation in crisp databases. In [44], a fuzzy equi-join was proposed based on a fuzzy equality comparison, which allows threshold values to be associated with predicates of the join condition. In order to evaluate the fuzzy equi-join, a sort-merge join algorithm based on a partial order of intervals is presented. In [45], efficient unnesting techniques were proposed to handle nested fuzzy queries in fuzzy databases, and an extended merge-join is adopted to evaluate the unnested fuzzy queries. In [46], fuzzy INs, as a generalization of classical INs, were proposed in fuzzy relational databases with the derived inference rules. An algorithm to discover such fuzzy INs was also introduced. However, to our knowledge, there has not been any work that applies both functional and inclusion dependencies as means to maintain consistency in fuzzy databases.

Within the vague set model, we previously studied the concepts of Vague FD (VFD) [19], Vague SQL (VSQL) [3] and Vague Association Rule (VAR) [20,21]. The preliminary version of the *VChase* procedure was also proposed in [22], in which only FDs are involved and the details of many important results are not included. VFDs [19] are based on the concept of similar equality but not *mi*-overlapping of attribute values in vague relations. VSQL can be employed to formulate a wide range of queries arising from four modes of query and data interaction, that is, crisp or vague data and conventional SQL or VSQL. VARs address a limitation in the traditional AR mining problem, which ignores the hesitation information of items in transactions. For example, in many online shopping applications, the items that are put into the basket but not checked out carry hesitation information. All the above work indicates that the vague set model is an effective means to capture vague information involved in applications.

### 3 *Mi* memberships and vague databases

In this section, we formally define the median membership, the imprecision membership and vague databases, which are fundamental to model vague data. Further related properties of the vague data model can be understood from [19,3,22].

#### 3.1 Vague sets

We give the definition of a vague set [2] as follows. Let  $U$  be a classical set of objects, called the *universe of discourse*, where an element of  $U$  is denoted by  $u$ .

**Definition 3.1 (Vague set)** A vague set  $V$  in a universe of discourse  $U$  is characterized by a true membership function,  $\alpha_V$ , and a false membership function,  $\beta_V$ , as follows:  $\alpha_V : U \rightarrow [0, 1]$ ,  $\beta_V : U \rightarrow [0, 1]$ , and  $0 \leq \alpha_V(u) + \beta_V(u) \leq 1$ , where  $\alpha_V(u)$  is a lower bound on the grade of membership of  $u$  derived from the evidence for  $u$ , and  $\beta_V(u)$  is a lower bound on

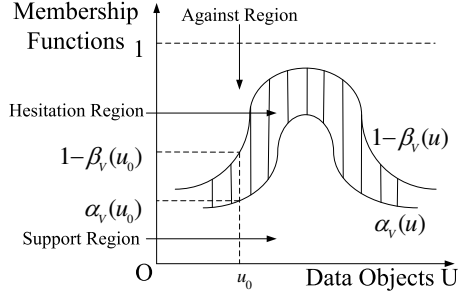


Fig. 1. The true ( $\alpha$ ) and false ( $\beta$ ) membership functions of a vague set

*the grade of membership of the negation of  $u$  derived from the evidence against  $u$ .*

Suppose  $U = \{u_1, u_2, \dots, u_n\}$ . A vague set  $V$  in the universe of discourse  $U$  can be represented by  $V = \sum_{i=1}^n [\alpha(u_i), 1 - \beta(u_i)]/u_i$ , where  $0 \leq \alpha(u_i) \leq 1 - \beta(u_i) \leq 1$  and  $1 \leq i \leq n$ .

This approach bounds the grade of membership of  $u$  to a subinterval  $[\alpha_V(u), 1 - \beta_V(u)]$  of  $[0, 1]$ . In other words, the exact grade of membership  $\mu_V(u)$  of  $u$  may be unknown, but is bounded by  $\alpha_V(u) \leq \mu_V(u) \leq 1 - \beta_V(u)$ , where  $\alpha_V(u) + \beta_V(u) \leq 1$ . We illustrate the idea in Fig. 1. Throughout this paper, we simply use  $\alpha$  and  $\beta$  for  $u$  if no ambiguity of  $V$  arises.

We say that the interval  $[\alpha(u), 1 - \beta(u)]$  is the vague value of the object  $u$ . For example, if  $[\alpha(u), 1 - \beta(u)] = [0.6, 0.9]$ , then we can see that  $\alpha(u) = 0.6$ ,  $1 - \beta(u) = 0.9$  and  $\beta(u) = 0.1$ . This is interpreted as “the degree that object  $u$  belongs to the vague set  $V$  is 0.6, the degree that object  $u$  does not belong to the vague set  $V$  is 0.1.” For example, in a voting process, the vague value  $[0.6, 0.9]$  can be interpreted as “the vote for resolution is 6 in favor, 1 against, and 3 neutral (abstentions).”

The precision of the knowledge about  $u$  is characterized by the difference  $(1 - \beta(u) - \alpha(u))$ . If this value is small, the knowledge about  $u$  is relatively precise. If  $(1 - \beta(u))$  is equal to  $\alpha(u)$ , the knowledge about  $u$  is exact, and the vague set theory reverts to fuzzy set theory. If  $(1 - \beta(u))$  and  $\alpha(u)$  are both equal to 1 or 0, the knowledge about  $u$  is exact (that is,  $[\alpha(u), 1 - \beta(u)]/u \in V$  or  $[\alpha(u), 1 - \beta(u)]/u \notin V$ ) and  $V$  reverts to an ordinary set. Thus, any crisp value can be regarded as a special case of a vague set. For example, a crisp data value  $u$  can be presented by the vague set  $[1, 1]/u$ . In addition, the fuzzy set  $0.8/u$  (the membership of  $u$  is 0.8) can be presented as the vague set  $[0.8, 0.8]/u$ .

### 3.2 Median memberships, imprecision memberships and mi-pair vague sets

In order to compare vague values, we need to introduce two important derived memberships.

The first one is called the *median membership*,  $M_m = (\alpha + 1 - \beta)/2$ , which represents the overall evidence contained in a vague value and is illustrated in Fig. 2.

**Definition 3.2 (Median membership)** *The median membership of an object  $u \in U$  in a vague set  $V$ , denoted by  $M_m^V(u)$ , is defined by  $M_m^V(u) = (\alpha(u) + 1 - \beta(u))/2$ . Whenever  $V$  and  $u$  are understood from the context, we simply write  $M_m$ .*

It can be verified that  $0 \leq M_m \leq 1$ . In addition, the vague value  $[1, 1]$  has the highest  $M_m$ ,

which means the corresponding object totally belongs to  $V$  (that is, a crisp value). The vague value  $[0,0]$  has the lowest  $M_m$ , which informally means that the corresponding object does not “totally” belong to  $V$  (that is, the empty vague value). The higher the  $M_m$  is, the more evidence of existence the vague value represents.

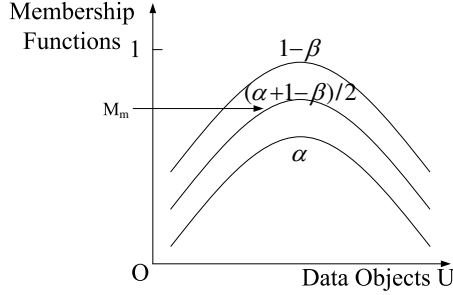


Fig. 2. Median membership of a vague set

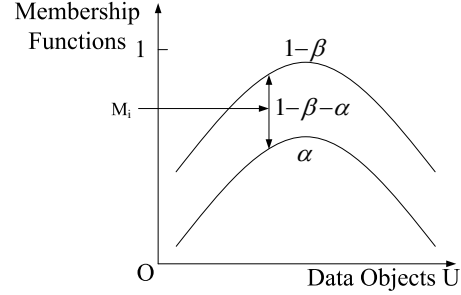


Fig. 3. Imprecision membership of a vague set

The second one is called the *imprecision membership*,  $M_i = (1 - \beta - \alpha)$ , which represents the overall imprecision of a vague value and is illustrated in Fig. 3.

**Definition 3.3 (Imprecision membership)** *The imprecision membership of an object  $u \in U$  in a vague set  $V$ , denoted by  $M_i^V(u)$ , is defined by  $M_i^V(u) = 1 - \beta(u) - \alpha(u)$ . Whenever  $V$  and  $u$  are understood from the context, we simply write  $M_i$ .*

It can be verified that  $0 \leq M_i \leq 1$ . In addition, the vague value  $[a, a]$  ( $a \in [0, 1]$ ) has the lowest  $M_i$  which means that we know exactly the membership of the corresponding object (that is, a vague value is then reduced to a fuzzy value). Furthermore, if  $a = 1$ , the vague value is equivalent to a crisp value as appears in usual databases. The vague value  $[0,1]$  has the highest  $M_i$ , which informally means that we know “nothing” about the precision of the corresponding object. The higher the  $M_i$  is, the more imprecision the vague value represents.

**Proposition 3.1** *The median membership and the imprecision membership of an object satisfy the inequality:  $0 \leq \frac{M_i}{2} \leq M_m \leq (1 - \frac{M_i}{2}) \leq 1$ .*

**Proof.** It follows directly from Definitions 3.2 and 3.3 after simple computations.  $\square$

Proposition 3.1 shows that the median and imprecision memberships are bounded but they relate to each other (cf. the relationship between true and false memberships in Definition 3.1).

**Definition 3.4 (*Mi*-pair vague set)** *A *mi*-pair VS vague set in  $U = \{u_1, u_2, \dots, u_n\}$  is characterized by a median membership function,  $M_m^V$ , and an imprecision membership function,  $M_i^V$ , where  $M_m^V : U \rightarrow [0, 1]$ , and  $M_i^V : U \rightarrow [0, 1]$ .  $V$  is given as follows:  $V = \sum_{i=1}^n \langle M_m^V(u_i), M_i^V(u_i) \rangle / u_i$ .  $\langle M_m^V(u_i), M_i^V(u_i) \rangle / u_i$  is called an element of  $V$  and  $\langle M_m^V(u_i), M_i^V(u_i) \rangle$  is called the (*mi*-pair) vague value of the object  $u_i$ .*

It is worth mentioning that a *mi*-pair vague value  $\langle M_m, M_i \rangle$  not satisfying the inequality in Proposition 3.1 can render the corresponding vague value  $[\alpha, 1 - \beta]$  invalid. For example, the corresponding vague value of  $\langle 0, 1 \rangle$  is  $[-0.5, 0.5]$ , which is not defined.

Using  $M_m$  and  $M_i$ , we have a one-to-one correspondence between a vague value,  $[\alpha, 1 - \beta]$ , and a  $mi$ -pair vague value,  $\langle M_m, M_i \rangle$ . From now on, whenever no confusion arises, a vague set or a vague value refers to a  $mi$ -pair vague set or a  $mi$ -pair vague value, respectively. The definition of a vague database built upon vague sets is presented as follows.

**Definition 3.5 (Vague relation and vague database)** Let  $Dom(A_i)$  be the domain corresponding to the attribute  $A_i$ . We define  $Dom(A_i) = \{V \mid V \text{ is a VS in } U\}$ . A vague tuple  $t = (a_1, a_2, \dots, a_m)$  over a relation scheme,  $R = \{A_1, A_2, \dots, A_m\}$ , is an element in  $D = Dom(A_1) \times Dom(A_2) \times \dots \times Dom(A_m)$ . A vague relation  $r$  over  $R$  is a subset of  $Dom(A_1) \times Dom(A_2) \times \dots \times Dom(A_m)$ . A vague database  $d$  over  $\mathcal{R} = \{R_1, \dots, R_n\}$  is a set of relations  $r_i$  over  $R_i$ .

Unlike classical relations,  $Dom(A_i)$  in vague relations is defined as a set of VSs. As a crisp value can be expressed by a VS, crisp (or classical) relations can be viewed as a special case of vague relations in which each vague set contains only one (crisp) object having membership  $\langle 1, 0 \rangle$ .

Here is an example of a vague relation.

Table 4

A sensor vague relation  $r$

	S	T	L
$t_1$	$\langle 0.7, 0.4 \rangle / 0 + \langle 0.5, 1 \rangle / 3$	$\langle 0.8, 0.3 \rangle / 0 + \langle 0.6, 0.1 \rangle / 1$	$\langle 0.4, 0.3 \rangle / 0 + \langle 0.6, 0.3 \rangle / 1$
$t_2$	$\langle 0.8, 0.1 \rangle / 0 + \langle 0.1, 0.1 \rangle / 1$	$\langle 0.9, 0.1 \rangle / 1 + \langle 0.5, 0.1 \rangle / 2$	$\langle 0.6, 0.6 \rangle / 0 + \langle 0.5, 0.2 \rangle / 2$
$t_3$	$\langle 0.9, 0.2 \rangle / 1 + \langle 0.5, 0.1 \rangle / 2$	$\langle 0.3, 0.2 \rangle / 2$	$\langle 0.2, 0.2 \rangle / 0$
$t_4$	$\langle 0.5, 0.1 \rangle / 3 + \langle 0.8, 0.2 \rangle / 4$	$\langle 0.4, 0.4 \rangle / 3$	$\langle 0.4, 0.2 \rangle / 3$

**Example 1** Let  $R = \{S, T, L\}$  be a vague relation schema, where  $S$  stands for a sensor ID,  $T$  stands for the temperature monitored by a sensor and  $L$  stands for a location area ID. A sensor vague relation  $r$  having 4 tuples  $\{t_1, t_2, t_3, t_4\}$  is shown in Table 4. For those vague elements not listed in the relation, we assume they all have a special vague value  $\langle 0, 1 \rangle$ , which represents the boundary of vague values, since any median membership is greater than or equal to 0 and any imprecision membership is less than or equal to 1.

In order to represent vague elements not listed in a vague relation and develop our  $VChase$  procedure later on in the paper, we formally define a special VS called boundary VSs as follows.

**Definition 3.6 (Boundary VS)** A boundary VS, denoted as  $\chi$ , is a VS in which for all  $u \in U$ ,  $M_m = 0$  and  $M_i = 1$ .

Intuitively,  $\chi$  represents the minimal VS as an attribute value in vague relations.

### 3.3 Existence and overlap of vague sets

We next define the concepts of a  $mi$ -existing VS and overlapping VSs. The underlying idea is to check if vague values satisfy the predefined  $mi$ -thresholds:  $C$  as a crisp threshold ( $0 \leq$

$C \leq 1$ , and  $I$  as an imprecision threshold ( $0 \leq I \leq 1$ ).

**Definition 3.7 (Mi-existing)** Given  $V$  and the *mi*-thresholds  $C$  and  $I$ , if  $\exists u \in U$ , such that  $M_m^V(u) \geq C$  and  $M_i^V(u) \leq I$ , then  $u$  is a *mi-existing object*,  $\langle M_m^V(u), M_i^V(u) \rangle / u$  is a *mi-existing element*, and  $V$  is said to be a *mi-existing VS* (or  $V$  is *mi-existing*) under  $C$  and  $I$ .

By Definition 3.7, it follows that  $V$  is not *mi-existing* if all the objects in  $V$  are not *mi-existing* under  $C$  and  $I$ .

**Definition 3.8 (Mi-overlap)** Given two vague sets  $V_1$  and  $V_2$ , if  $\exists u \in U$ , such that  $M_m^{V_1}(u) \geq C$  and  $M_m^{V_2}(u) \geq C$ ,  $M_i^{V_1}(u) \leq I$  and  $M_i^{V_2}(u) \leq I$ , then  $V_1$  and  $V_2$  *mi-overlap* under *mi*-thresholds  $C$  and  $I$ , denoted by  $V_1 \sim_{mi} V_2(C, I)$ .  $u$  is called the *common mi-existing object* of  $V_1$  and  $V_2$  under  $C$  and  $I$ . Otherwise,  $V_1$  and  $V_2$  do not *mi-overlap* under  $C$  and  $I$ , denoted by  $V_1 \not\sim_{mi} V_2(C, I)$ . We may skip the specification of  $C$  and  $I$  if they are known from the context.

By Definition 3.8, it follows that  $V_1$  and  $V_2$  do not *mi-overlap* if there is no common *mi-existing* object of  $V_1$  and  $V_2$  under  $C$  and  $I$ .

**Example 2** Given  $C = 0.2$  and  $I = 0.9$ , it can be verified that  $t_1[L]$  and  $t_2[L]$  in Table 4 *mi-overlap*, that is,  $t_1[L] \sim_{mi} t_2[L](0.2, 0.9)$ . However, if  $C = 0.2$  and  $I = 0.5$ , we find that  $t_1[L]$  and  $t_2[L]$  do not *mi-overlap*, that is,  $t_1[L] \not\sim_{mi} t_2[L](0.2, 0.5)$ .

Using the *mi-existing* objects of VSs, we define *mi-union* and *mi-intersection* of VSs based on maximum and minimum operations on vague values. Intuitively, these two operations are the *mi-pair* counterparts of a usual set union and intersection.

**Definition 3.9 (Mi-union)** Given two VSs  $V_1$  and  $V_2$  under the *mi*-thresholds  $C$  and  $I$ , the *mi-union* of  $V_1$  and  $V_2$  is a VS  $V_3 = V_1 \vee V_2$ , whose median membership and imprecision membership functions are given according to the following cases. Let  $u \in U$ .

- (1) If  $u$  is a *mi-existing object* in both  $V_1$  and  $V_2$ ,  
 $M_m^{V_3}(u) = \max(M_m^{V_1}(u), M_m^{V_2}(u))$ ,  $M_i^{V_3}(u) = \min(M_i^{V_1}(u), M_i^{V_2}(u))$ ;
- (2) If  $u$  is a *mi-existing object* in  $V_1$  but not in  $V_2$ ,  
 $M_m^{V_3}(u) = M_m^{V_1}(u)$ ,  $M_i^{V_3}(u) = M_i^{V_1}(u)$ ;
- (3) If  $u$  is not a *mi-existing object* in both  $V_1$  and  $V_2$ ,  
 $M_m^{V_3}(u) = M_m^{V_1}(u)$ ,  $M_i^{V_3}(u) = M_i^{V_1}(u)$ , if  $M_m^{V_1}(u) \geq M_m^{V_2}(u)$ ;  
 $M_m^{V_3}(u) = M_m^{V_2}(u)$ ,  $M_i^{V_3}(u) = M_i^{V_2}(u)$ , otherwise.

Notably, the third case of Definition 3.9 adopts the vague value from either  $V_1$  or  $V_2$ , depending on which has the higher median membership. This guarantees that the two non-*mi-existing* elements in the union operation cannot be “upgraded” to be *mi-existing* elements. In other words, the union operation always keeps the elements that are under *mi*-thresholds to be non-*mi-existing*.

**Definition 3.10 (Mi-intersection)** Using the same set of notations of Definition 3.9, the *mi-intersection* of VSs  $V_1$  and  $V_2$  is a VS  $V_3 = V_1 \wedge V_2$ , whose median membership and imprecision membership functions are given according to the following cases.

- (1) If  $u$  is a *mi-existing object* in both  $V_1$  and  $V_2$ ,

$$\begin{aligned}
M_m^{V_3}(u) &= \min(M_m^{V_1}(u), M_m^{V_2}(u)), \\
M_i^{V_3}(u) &= \min(\max(M_i^{V_1}(u), M_i^{V_2}(u)), 2M_m^{V_3}(u)); \\
(2) \text{ If } u \text{ is a } mi\text{-existing object in } V_1 \text{ but not in } V_2, \\
M_m^{V_3}(u) &= M_m^{V_2}(u), M_i^{V_3}(u) = M_i^{V_2}(u); \\
(3) \text{ If } u \text{ is not a } mi\text{-existing object in both } V_1 \text{ and } V_2, \\
M_m^{V_3}(u) &= M_m^{V_1}(u), M_i^{V_3}(u) = M_i^{V_1}(u), \text{ if } M_m^{V_1}(u) \leq M_m^{V_2}(u); \\
M_m^{V_3}(u) &= M_m^{V_2}(u), M_i^{V_3}(u) = M_i^{V_2}(u), \text{ otherwise.}
\end{aligned}$$

It can be proved that the term  $2M_m^{V_3}(u)$  in Case 1 of Definition 3.10 ensures that the imprecision membership of the result in  $V_3$  is in the valid range. For example, given two vague values  $\langle 0.2, 0.3 \rangle$  and  $\langle 0.5, 0.8 \rangle$ , we have  $\langle \min(0.2, 0.5), \max(0.3, 0.8) \rangle = \langle 0.2, 0.8 \rangle$ . However, its corresponding interval vague value is  $[-0.2, 0.6]$ , which is out of the valid range  $[0, 1]$ . Therefore, according to the definition, we have  $\langle 0.2, \min(0.8, 2 \times 0.2) \rangle = \langle 0.2, 0.4 \rangle$  as the *mi*-intersection value, whose corresponding interval vague value  $[0, 0.4]$  is valid.

The following proposition indicates that the boundary VS,  $\chi$ , shares the properties of the usual empty set.

**Proposition 3.2** *Given a VS  $V$ ,  $\chi \vee V = V$  and  $\chi \wedge V = \chi$ .*

**Proof.** It follows directly from Definitions 3.6, 3.9 and 3.10.  $\square$

#### 4 Merge operation of vague relations

In this section, we define the merge of a vague relation  $r$  as the operation which replaces each attribute value (represented by a VS) in  $r$  by the *mi*-union of all attribute values with respect to the same reflexive and transitive closure under *mi*-overlap. This leads to the concept of two kinds of partial orders related to precision on merged vague relations.

From now on, we let  $R = \{A_1, A_2, \dots, A_m\}$  be a relation schema and  $r$  be a vague relation over  $R$ . We also assume common notation used in relational databases [5] such as the projection of a tuple  $t[A]$ .

The semantics of a vague set,  $t[A_i]$ , where  $t \in r$  and  $A_i \in R$ , are that an object  $u \in U_i$  has the vague value  $\langle M_m(u), M_i(u) \rangle$  in  $t[A_i]$ . The intuition is that, for those objects which are not *mi*-existing, we regard their memberships as too “weak” to be considered as inconsistent with respect to a set of FDs. This strategy provides simplicity and flexibility for maintaining an inconsistent database, since it deals with those significantly vague elements only.

**Definition 4.1** (*mi-active domain*) *The mi-active domain of  $r$  with respect to  $A_i$  is defined by  $\mathcal{A}(r, A_i) = \{v | v \text{ is } mi\text{-existing and } v \in \bigvee \{t[A_i] | t \in r\}\}$ .*

*The concept of a mi-active domain is naturally extended to relations and databases as follows.  $\mathcal{A}(r) = \bigvee_{A_i \in R} \mathcal{A}(r, A_i)$  and  $\mathcal{A}(d) = \bigvee_{r \in d} \mathcal{A}(r)$ .*

According to Definition 3.9, given any *mi*-existing object in  $r$ , the corresponding vague element in  $\mathcal{A}(d)$  has the highest median membership and the lowest imprecision membership.

We now define the merge operation which replaces each attribute value of a tuple in a vague

relation by the *mi*-union of all attribute values with respect to the same reflexive and transitive closure under *mi*-overlap.

**Definition 4.2 (Merged relation)** Given  $A \in R$  and *mi*-thresholds  $C$  and  $I$ , we construct a directed graph  $G = (V, E)$ , where  $V = \pi_A(r)$ . An edge  $(t_1[A], t_2[A])$  is in  $E$  iff  $t_1[A] \sim_{mi} t_2[A](C, I)$ . Let  $G^+ = (V^+, E^+)$  be the reflexive and transitive closure of  $G$ . The merge of  $r$ , denoted by  $merge(r)$ , is the vague relation resulting from replacing each  $t[A]$  by  $\bigvee \{t[A'] \mid (t[A], t[A']) \in E^+\}$  for all  $A \in R$ .

We let  $MERGE(R)$  be a collection of all merged relations over  $R$  under  $C$  and  $I$ .

**Example 3** Given  $C = 0.2$  and  $I = 0.9$ , the vague relation  $merge(r)$ , is shown in Table 5, where  $r$  is shown in Table 4. For example, since  $t_1[L] \sim_{mi} t_2[L](0.2, 0.9)$  and  $t_2[L] \sim_{mi} t_3[L](0.2, 0.9)$ , we replace  $t_1[L], t_2[L]$  and  $t_3[L]$  by  $\langle 0.6, 0.2 \rangle / 0 + \langle 0.6, 0.3 \rangle / 1 + \langle 0.5, 0.2 \rangle / 2$ . Note that the first two tuples in  $r$  ( $t_1$  and  $t_2$ ) have been merged into a single tuple ( $t'_1$ ) in  $merge(r)$ . With different *mi*-thresholds  $C$  and  $I$ , we may have different merge results. For example, if we set  $C = 0.2$  and  $I = 0.5$ , then  $t_1[L] \not\sim_{mi} t_2[L](0.2, 0.5)$ . In this case, we obtain  $merge(r)$  shown in Table 6. It can be verified that the first two tuples ( $t'_1$  and  $t'_2$ ) are not merged.

Table 5

A relation  $merge(r)$  under  $C = 0.2$  and  $I = 0.9$

	<b>S</b>	<b>T</b>	<b>L</b>
$t'_1$	$\langle 0.8, 0.1 \rangle / 0 + \langle 0.1, 0.1 \rangle / 1 + \langle 0.5, 1 \rangle / 3$	$\langle 0.8, 0.3 \rangle / 0 + \langle 0.9, 0.1 \rangle / 1 + \langle 0.5, 0.1 \rangle / 2$	$\langle 0.6, 0.2 \rangle / 0 + \langle 0.6, 0.3 \rangle / 1 + \langle 0.5, 0.2 \rangle / 2$
$t'_2$	$\langle 0.9, 0.2 \rangle / 1 + \langle 0.5, 0.1 \rangle / 2$	$\langle 0.8, 0.3 \rangle / 0 + \langle 0.9, 0.1 \rangle / 1 + \langle 0.5, 0.1 \rangle / 2$	$\langle 0.6, 0.2 \rangle / 0 + \langle 0.6, 0.3 \rangle / 1 + \langle 0.5, 0.2 \rangle / 2$
$t'_3$	$\langle 0.5, 0.1 \rangle / 3 + \langle 0.8, 0.2 \rangle / 4$	$\langle 0.4, 0.4 \rangle / 3$	$\langle 0.4, 0.2 \rangle / 3$

Table 6

A relation  $merge(r)$  under  $C = 0.2$  and  $I = 0.5$

	<b>S</b>	<b>T</b>	<b>L</b>
$t'_1$	$\langle 0.8, 0.1 \rangle / 0 + \langle 0.1, 0.1 \rangle / 1 + \langle 0.5, 1 \rangle / 3$	$\langle 0.8, 0.3 \rangle / 0 + \langle 0.9, 0.1 \rangle / 1 + \langle 0.5, 0.1 \rangle / 2$	$\langle 0.4, 0.2 \rangle / 0 + \langle 0.6, 0.3 \rangle / 1$
$t'_2$	$\langle 0.8, 0.1 \rangle / 0 + \langle 0.1, 0.1 \rangle / 1 + \langle 0.5, 1 \rangle / 3$	$\langle 0.8, 0.3 \rangle / 0 + \langle 0.9, 0.1 \rangle / 1 + \langle 0.5, 0.1 \rangle / 2$	$\langle 0.6, 0.6 \rangle / 0 + \langle 0.5, 0.2 \rangle / 2$
$t'_3$	$\langle 0.9, 0.2 \rangle / 1 + \langle 0.5, 0.1 \rangle / 2$	$\langle 0.8, 0.3 \rangle / 0 + \langle 0.9, 0.1 \rangle / 1 + \langle 0.5, 0.1 \rangle / 2$	$\langle 0.4, 0.2 \rangle / 0 + \langle 0.6, 0.3 \rangle / 1$
$t'_4$	$\langle 0.5, 0.1 \rangle / 3 + \langle 0.8, 0.2 \rangle / 4$	$\langle 0.4, 0.4 \rangle / 3$	$\langle 0.4, 0.2 \rangle / 3$

There are two levels of precision we consider in vague sets for handling inconsistency. The first is the *object-precision*, which intuitively means the precision according to the cardinality of a set of *mi*-existing objects. The second is that, given the same object in two VSs, the vague values have different *mi* precision, which we term the *value-precision*.

We first define a partial order named *less object-precise* on VSs based on *mi-existing* objects and then extend this partial order to tuples and relations in  $MERGE(R)$ .

**Definition 4.3 (Less object-precise and object-equivalence)** We define a partial order, *less object-precise* (or *less O-precise* for simplicity) between two vague sets  $V_1$  and  $V_2$  as follows:

$V_1 \sqsubseteq_O V_2$  if the set of *mi-existing* objects in  $V_1$  is a superset of the set of those in  $V_2$ . We say that  $V_1$  is *less O-precise* than  $V_2$ .

We extend  $\sqsubseteq_O$  in  $r$  as follows. Let  $t_1, t_2 \in r$ .  $t_1 \sqsubseteq_O t_2$  if  $\forall A_i \in R, t_1[A_i] \sqsubseteq_O t_2[A_i]$ . We say that  $t_1$  is *less O-precise* than  $t_2$ .

Finally, we extend  $\sqsubseteq_O$  in  $MERGE(R)$  as follows: Let  $r_1, r_2 \in MERGE(R)$ .  $r_1 \sqsubseteq_O r_2$  if  $\forall t_2 \in r_2, \exists t_1 \in r_1$  such that  $t_1 \sqsubseteq_O t_2$ . We say that  $r_1$  is *less O-precise* than  $r_2$ .

We define an *object-equivalence* between  $V_1$  and  $V_2$ , denoted as  $V_1 \doteq_O V_2$ , iff  $V_1 \sqsubseteq_O V_2$  and  $V_2 \sqsubseteq_O V_1$ . Similar definitions of *object-equivalence* are extended to tuples and relations.

We denote by  $\emptyset^O$  a special class of *object-equivalent* relations that contain no *mi-existing* objects. A vague relation in  $\emptyset^O$  can be regarded as an empty set of *mi-existing* tuples.

Thus, an *object-equivalence* relation on  $MERGE(R)$  induces a partition of  $MERGE(R)$ , which means all *object-equivalent* vague relations are put into one *O-equivalence class*. Given any two vague relations in an *O-equivalence class* of  $MERGE(R)$ , each tuple in one vague relation has a one-to-one correspondence in another vague relation.

Now, we define a partial order named *less value-precise* of VSs based on the vague values of *mi-existing* objects. We also extend this partial order to tuples and relations.

**Definition 4.4 (Less value-precise and value-equivalence)** We define a partial order, *less value-precise* (or *less V-precise* for simplicity), between  $V_1$  and  $V_2$  as follows:

Let  $a = \langle M_m^{V_1}, M_i^{V_1} \rangle$  and  $b = \langle M_m^{V_2}, M_i^{V_2} \rangle$  be the respective vague values of an object  $u$  in  $V_1$  and  $V_2$ . If  $M_m^{V_1} \leq M_m^{V_2}$  and  $M_i^{V_1} \geq M_i^{V_2}$  (that is,  $a$  represents less evidence of existence and more imprecision than  $b$  does), then we say  $a$  is *less V-precise* than  $b$  and denote this fact by  $a \sqsubseteq_V b$ .

Given *mi-thresholds*  $C$  and  $I$ ,  $V_1 \sqsubseteq_V V_2(C, I)$  if the vague value of each *mi-existing* object under  $C$  and  $I$  in  $V_1$  is *less V-precise* than that of the same object in  $V_2$ . We simply say that  $V_1$  is *less V-precise* than  $V_2$  whenever  $C$  and  $I$  are understood.

For any two tuples  $t_1$  and  $t_2$  in vague relations  $r$  and  $s$  over  $R = \{A_1, \dots, A_m\}$  and  $S = \{B_1, \dots, B_m\}$  respectively. Then  $t_1 \sqsubseteq_V t_2$  if  $\forall A_i \in R$  and  $B_i \in S, t_1[A_i] \sqsubseteq_V t_2[B_i]$ . The *less value-precise* order can be naturally extended to the projected tuples over  $R$  and  $S$ .

We further extend  $\sqsubseteq_V$  between  $r$  and  $s$  (or their projected relations) as follows.  $r \sqsubseteq_V s$  if  $\forall t_1 \in r, \exists t_2 \in s$  such that  $t_1 \sqsubseteq_V t_2$ . We say that  $r$  is *less V-precise* than  $s$  and denote this fact by  $r \sqsubseteq_V s$ .

We define a value-equivalence, denoted as  $V_1 \doteq_V V_2$  iff  $V_1 \sqsubseteq_V V_2$  and  $V_2 \sqsubseteq_V V_1$ . Similar definitions are extended to tuples and relations.

Within an  $O$ -equivalence class of  $MERGE(R)$ ,  $V$ -precision can be viewed as a second level precision with respect to the  $mi$ -existing objects. In other words, we ignore those non- $mi$ -existing objects in the  $V$ -precision comparison.

**Proposition 4.1** *The vague value of each  $mi$ -existing object in  $merge(r)$  is no less  $V$ -precise than the corresponding vague value of the same object in  $r$ .*

**Proof.** It follows directly from Definitions 3.9, 4.2 and 4.4.  $\square$

For example, the vague value  $\langle 0.8, 0.1 \rangle$  of the  $mi$ -existing object 0 in  $t_1[S]$  in Table 5 is more  $V$ -precise than the corresponding vague value  $\langle 0.7, 0.4 \rangle$  of the same object 0 in  $t_1[S]$  in Table 4.

According to Definition 4.4, we define  $V$ -join,  $\cup$ , and  $V$ -meet,  $\cap$ , under  $\sqsubseteq_V$  of vague values of a given object, that is,  $\langle M_m^x, M_i^x \rangle \cup \langle M_m^y, M_i^y \rangle = \langle \max\{M_m^x, M_m^y\}, \min\{M_i^x, M_i^y\} \rangle$  and  $\langle M_m^x, M_i^x \rangle \cap \langle M_m^y, M_i^y \rangle = \langle \min\{M_m^x, M_m^y\}, \max\{M_i^x, M_i^y\} \rangle$ . It is easy to check that the less  $V$ -precise order  $\sqsubseteq_V$  induces a complete semi-lattice by using  $\cup$  and  $\cap$  as shown in Fig. 4.

It can be checked that  $\langle 1, 0 \rangle$  is the top element according to the less  $V$ -precise order. Note that for some  $mi$ -pair vague values,  $V$ -meet may cause the corresponding vague value  $[\alpha(u), 1 - \beta(u)]$  falling beyond the valid range  $[0, 1]$ . From now on, we restrict our discussion to the  $V$ -meet that gives rise to valid vague values as a result.

Given any  $mi$ -thresholds  $C$  and  $I$ , if  $\langle C, I \rangle$  is a valid vague value, then we can use  $\langle C, I \rangle$  as a cut-off boundary to construct a complete lattice, rather than the original complete semi-lattice as illustrated in Fig. 4, induced by the less  $V$ -precise order  $\sqsubseteq_V$ . For example, given  $\langle C, I \rangle = \langle 0.5, 0.5 \rangle$  (or  $\langle 0.6, 0.4 \rangle$ ), which is a valid vague value, in the dotted-line region in Fig. 4, all vague values form a complete lattice, since given any two values in the enclosed region, we have their greatest lower bound and lowest upper bound. However, if  $\langle C, I \rangle$  is not a valid vague value, then we have a complete semi-lattice, since some values in the enclosed region constructed by  $\langle C, I \rangle$  do not have their greatest lower bound. For instance, in the dotted-line region with respect to an invalid vague value  $\langle 0.1, 0.3 \rangle$ , all vague values form a complete semi-lattice, since for  $\langle 0.1, 0.2 \rangle$  and  $\langle 0.2, 0.3 \rangle$ , we do not have their greatest lower bound.

From Definition 4.3, we can deduce that  $MERGE(R)$  is a lattice based on  $O$ -equivalence classes with respect to  $\sqsubseteq_O$ . In this lattice, each node is an  $O$ -equivalence class, in which all vague relations are  $O$ -equivalent. The top node is the  $O$ -equivalence class of  $\emptyset^O$ , that is, the set of vague relations with no  $mi$ -existing objects. The bottom node is the  $O$ -equivalence class, in which all vague relations have only one tuple and all  $mi$ -existing objects in vague relations form the universe of discourse.

**Example 4** *For simplicity, we assume  $U = \{0, 1\}$  and  $R = \{A\}$  and construct the lattice for  $MERGE(R)$  under  $C = 0.5$  and  $I = 0.5$  according to  $O$ -equivalence classes. As shown in Fig. 5, all  $O$ -equivalence classes (the nodes represented by circles) form a lattice based*

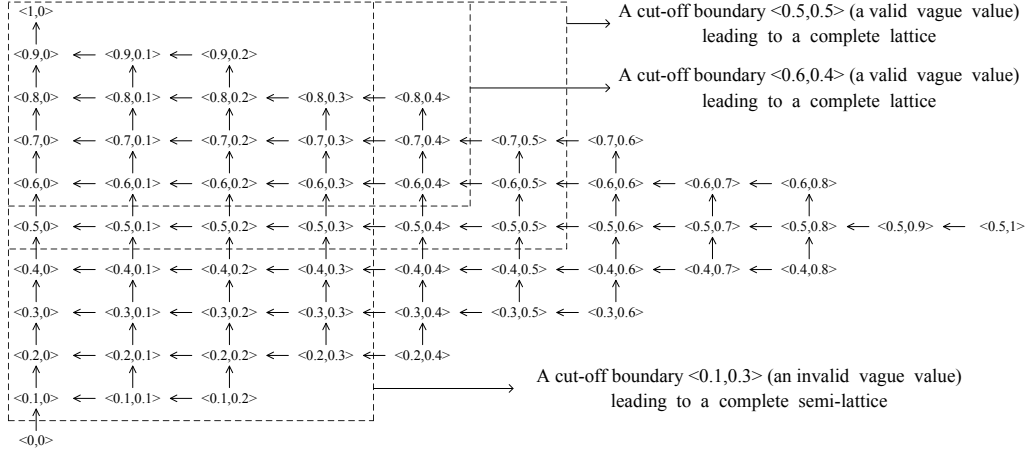


Fig. 4. A complete semi-lattice or lattice of vague values of an object  $u$  according to the cut-off

on  $\sqsubseteq_O$ . Each node in the lattice is actually the set of all vague relations (represented by tables with single attribute) which are  $O$ -equivalent to each other. For instance,  $r_1$  and  $r_2$  are two vague relations with two tuples such that  $r_1 \doteq_O r_2$ . Similarly, we have  $r_3 \doteq_O r_4$ , where  $r_3$  and  $r_4$  are two vague relations with only one tuple. Inside each node, based on  $\sqsubseteq_V$  in Definition 4.4, all vague relations in the node form a complete lattice (when the cut-off boundary is a valid vague value) or a complete semi-lattice (when the cut-off boundary is not a valid vague value). In the complete (semi-)lattice, the top element is the vague relation in which all vague values of objects are  $\langle 1,0 \rangle$ , and if the cut-off boundary  $\langle C, I \rangle$  is a valid vague value, then the bottom element is the vague relation in which all vague values of objects are  $\langle C, I \rangle$ . For example, in the lattice shown in Fig. 7, which is the bottom node of the lattice in Fig. 5, each table represents a single attribute vague relation. The top is the single attribute vague relation  $r_3$  with one tuple  $(\langle 1,0 \rangle / 0 + \langle 1,0 \rangle / 1)$ . The bottom is the vague relation  $r_4$  with single tuple  $(\langle 0.5,0.5 \rangle / 0 + \langle 0.5,0.5 \rangle / 1)$ , and the vague value of each object is  $\langle C, I \rangle$ .

Given different  $mi$ -thresholds, a lattice induced by  $\sqsubseteq_V$  exists inside each node. For instance, we have the lattice of  $MERGE(R)$  under  $C = 0.5$  and  $I = 0.4$  as shown in Fig. 6. The bottom elements in each node are different from those in Fig. 5.

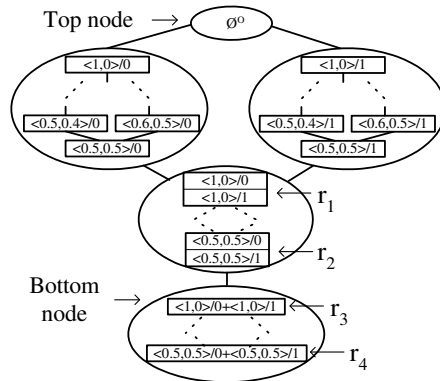


Fig. 5. A lattice of  $MERGE(R)$  under  $C = 0.5$  and  $I = 0.5$

Now, we extend the concept of  $mi$ -existing VSs given in Definition 3.7 to tuples as follows:  $t[X]$  is  $mi$ -existing, if  $\forall A \in X, t[A]$  is  $mi$ -existing, where  $X \subseteq R$ . We also extend the concept of  $mi$ -overlap given in Definition 3.8 to tuples  $t_1, t_2 \in r$  under  $mi$ -thresholds  $C$  and

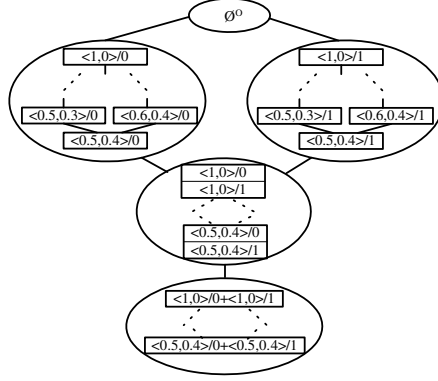


Fig. 6. A lattice of  $MERGE(R)$  under  $C = 0.5$  and  $I = 0.4$

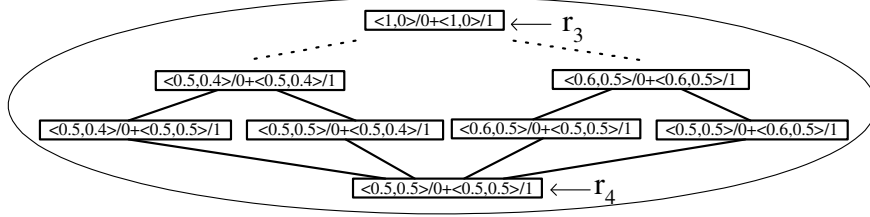


Fig. 7. A lattice within the bottom node of the lattice of  $MERGE(R)$  of Fig. 5

$I$  as follows:  $t_1[X] \sim_{mi} t_2[X](C, I)$ , if  $\forall A \in X, t_1[A] \sim_{mi} t_2[A](C, I)$  where  $X \subseteq R$ .

**Example 5** We can verify that  $t_1 \sim_{mi} t_2(0.2, 0.9)$  but  $t_1 \not\sim_{mi} t_2(0.9, 0.2)$  in the relation shown in Table 4.

## 5 Functional dependencies and vague chase

In this section, we formalize the notion of an FD being satisfied in a vague relation  $r$  and present the  $VChase$  procedure for  $r$  as a means of maintaining consistency of  $r$  with respect to a given set of FDs.

### 5.1 Functional dependencies in vague relations

Functional dependencies (FDs) being satisfied in a vague relation  $r$  can be formalized in terms of values being  $mi$ -overlapping rather than equal. Lien's and Atzeni's axiom system is sound and complete for FDs being satisfied in vague relations.

**Definition 5.1 (Functional dependency)** Given  $mi$ -thresholds  $C$  and  $I$ , a functional dependency over  $R$  (or simply an FD) is a statement of the form  $X \rightarrow_{C,I} Y$ , where  $X, Y \subseteq R$ . We may simply write  $X \rightarrow Y$  if  $C$  and  $I$  are known from the context. An FD  $X \rightarrow Y$  is satisfied in a relation  $r$ , denoted by  $r \models X \rightarrow Y$ , if  $\forall t_1, t_2 \in r, t_1[X] \sim_{mi} t_2[X](C, I)$ , then  $t_1[Y] \sim_{mi} t_2[Y](C, I)$ , or  $t_1[Y]$  or  $t_2[Y]$  are not  $mi$ -existing.

A set of FDs  $F$  over  $R$  is satisfied in  $r$ , denoted by  $r \models F$ , if  $\forall X \rightarrow Y \in F, r \models X \rightarrow Y$ . If  $r \models F$  we say that  $r$  is consistent with respect to  $F$  (or simply  $r$  is consistent if  $F$  is understood from context); otherwise if  $r \not\models F$  then we say that  $r$  is inconsistent with respect to  $F$  (or simply  $r$  is inconsistent). We let  $SAT(F)$  denote the finite set  $\{r \in MERGE(R) \mid$

$r \models F$ }. We remark that it is necessary to restrict *mi*-thresholds  $C > 0$  or  $I < 1$  in defining FDs. Otherwise (that is,  $C = 0$  and  $I = 1$ ), any given FD is trivially satisfied in a relation.

**Example 6** Let  $F = \{S \rightarrow_{0.2,0.9} TL, L \rightarrow_{0.2,0.9} S\}$  be a set of FDs over  $R$ , where  $R$  is the relation schema whose semantics are given in Example 1. We can verify that  $r \models S \rightarrow_{0.2,0.9} TL$  but that  $r \not\models L \rightarrow_{0.2,0.9} S$ , where  $r$  is the relation shown in Table 4. Thus  $r \in \text{SAT}(\{S \rightarrow_{0.2,0.9} TL\})$  but  $r \notin \text{SAT}(F)$ . Consider also  $\text{merge}(r)$  shown in Table 5, we have  $\text{merge}(r) \in \text{SAT}(\{S \rightarrow_{0.2,0.9} TL\})$  but  $\text{merge}(r) \notin \text{SAT}(F)$ . If we change the *mi*-thresholds from 0.2 and 0.9 to 0.2 and 0.5, the result is different. Let  $F = \{S \rightarrow_{0.2,0.5} TL, L \rightarrow_{0.2,0.5} S\}$  be a set of FDs over  $R$ . We can verify that  $r \not\models S \rightarrow_{0.2,0.5} TL$  and that  $r \not\models L \rightarrow_{0.2,0.5} S$ . Thus  $r \notin \text{SAT}(\{S \rightarrow_{0.2,0.5} TL\})$  and  $r \notin \text{SAT}(F)$ . Consider also  $\text{merge}(r)$  shown in Table 6, we have  $\text{merge}(r) \notin \text{SAT}(\{S \rightarrow_{0.2,0.5} TL\})$  and  $\text{merge}(r) \notin \text{SAT}(F)$ .

The next proposition is immediate from Definition 5.1. The first part shows that the semantics of FDs can be characterized in terms of object equivalence. The second part indicates that the consistency in  $r$  and  $\text{merge}(r)$  with respect to an FD is equivalent.

**Proposition 5.1** *The following statements are true:*

- (1) If  $r \in \text{MERGE}(R)$ , then  $r \models X \rightarrow Y$ , if and only if,  $\forall t_1, t_2 \in r$ , if  $t_1[X] \doteq_O t_2[X]$  then  $t_1[Y] \doteq_O t_2[Y]$  or  $t_1[Y]$  or  $t_2[Y]$  are not *mi*-existing.
- (2)  $r \models X \rightarrow Y$  if and only if  $\text{merge}(r) \models X \rightarrow Y$ .

**Proof.** Part 1. It follows directly from Definitions 4.3 and 5.1.

Part 2. By Definitions 4.2 and 4.3, for each  $A \in R$ , the family of  $O$ -equivalent VSs over  $R$  with respect to  $r$  is equal to the family of  $O$ -equivalent VSs over  $\pi_A(\text{merge}(r))$ . Then by Definition 5.1, the result follows.  $\square$

We say that  $F$  logically implies an FD  $X \rightarrow Y$ , denoted by  $F \models X \rightarrow Y$ , such that  $\forall r$  over  $R$ , if  $r \models F$  holds then  $r \models X \rightarrow Y$  also holds.

Let  $X, Y, Z \subseteq R$  and  $XY = X \cup Y$ . Here we state the well known Lien's and Atzeni's axiom system [8,5] for incomplete relations  $r$  as follows:

- (1) *Reflexivity*: If  $r \models Y \subseteq X$ , then  $r \models X \rightarrow Y$  holds.
- (2) *Augmentation*: If  $r \models X \rightarrow Y$  holds, then  $r \models XZ \rightarrow YZ$  also holds.
- (3) *Union*: If  $r \models X \rightarrow Y$  and  $r \models X \rightarrow Z$  hold, then  $r \models X \rightarrow YZ$  holds.
- (4) *Decomposition*: If  $r \models X \rightarrow YZ$  holds, then  $r \models X \rightarrow Y$  and  $r \models X \rightarrow Z$  hold.

**Definition 5.2 (Soundness and completeness of axiom system)** *Whenever an FD  $X \rightarrow Y$  can be proven from  $F$  using a finite number of inference rules from Lien's and Atzeni's axiom system [5], we write  $F \vdash X \rightarrow Y$ .*

*Lien's and Atzeni's axiom system is sound if  $F \vdash X \rightarrow Y$  implies  $F \models X \rightarrow Y$ . Correspondingly, Lien's and Atzeni's axiom system is complete if  $F \models X \rightarrow Y$  implies  $F \vdash X \rightarrow Y$ .*

The proof of the following theorem is standard [5], in which we establish a counter example relation to show that  $F \not\models X \rightarrow Y$  but  $F \vdash X \rightarrow Y$ .

**Theorem 5.2** *Lien's and Atzeni's axiom system is sound and complete for FDs being satisfied in vague relations.*

Table 7

The counter example vague relation

$X$	$X^+ - X$	$R - X^+$
$\langle 1,0 \rangle / 0 \cdots \langle 1,0 \rangle / 0$	$\langle 0,1 \rangle / 0 \cdots \langle 0,1 \rangle / 0$	$\langle 1,0 \rangle / 1 \cdots \langle 1,0 \rangle / 1$
$\langle 1,0 \rangle / 0 \cdots \langle 1,0 \rangle / 0$	$\langle 1,0 \rangle / 0 \cdots \langle 1,0 \rangle / 0$	$\langle 1,0 \rangle / 0 \cdots \langle 1,0 \rangle / 0$

**Proof.** The soundness of Lien’s and Atzeni’s axiom system can be easily verified. We prove completeness by showing that  $F \not\vdash X \rightarrow Y$ , then  $F \not\vdash X \rightarrow Y$ , where  $F$  is a set of FDs over schema  $R$ . Assume  $F \not\vdash X$ . Equivalently for the latter, it is sufficient to exhibit a vague relation  $r$  over  $R$ , such that  $\forall W \rightarrow Z \in F, r \models W \rightarrow Z$  but  $r \not\vdash X \rightarrow Y$ . Let  $X^+ = \cup\{Y \mid F \vdash X \rightarrow Y\}$ . Given  $r$  shown in Table 7,  $C > 0$  or  $I < 1$ , we first need to show that  $\forall W \rightarrow Z \in F, r \models W \rightarrow Z$ . Suppose there exists an FD  $W \rightarrow Z \in F$ , such that  $r \not\vdash W \rightarrow Z$ . From  $r$ , we see that  $W \subseteq X$  and  $\exists A \in Z \cap (R - X^+)$ , which implies that  $A \notin X^+$ . By augmentation we have  $F \vdash X \rightarrow ZX$ , and by decomposition we have  $F \vdash X \rightarrow A$ , which follows that  $A \in X^+$ . It thus leads to a contradiction. Next, we show that  $r \not\vdash X \rightarrow Y$ . Suppose  $r \models X \rightarrow Y$ . Since  $Y \subseteq X^+$ . By the definition of  $X^+$ , we have  $F \vdash X \rightarrow X^+$ . By decomposition,  $\forall A \in Y, F \vdash X \rightarrow A$  holds and by union,  $F \vdash X \rightarrow Y$  holds, which is contradictory to our assumption.  $\square$

## 5.2 Vague chase

We now define the chase procedure for maintaining consistency in vague relations. Assuming that a vague relation  $r$  is updated with information obtained from several different sources, at any given time the vague relation  $r$  may be inconsistent with respect to a set of FDs  $F$ . Thus we input  $r$  and  $F$  into the  $VChase$  procedure and its output, denoted by  $VChase(r, F)$ , is a consistent relation over  $R$  with respect to  $F$ . The pseudo-code for the algorithm  $VChase(r, F)$  is presented in Algorithm 1. For convenience, we may also use  $VChase(r, F)$  to represent the output relation.

**Algorithm 1.**  $VChase(r, F)$

- 1: Result :=  $r$ ;
- 2: Tmp :=  $\emptyset$ ;
- 3: **while** Tmp  $\neq$  Result **do**
- 4:   Tmp := Result;
- 5:   **if**  $X \rightarrow_{C,I} Y \in F, \exists t_1, t_2 \in$  Result **such that**  $t_1[X] \sim_{mi} t_2[X](C, I), t_1[Y]$  **and**  $t_2[Y]$  **are** *mi*-existing **but**  $t_1[Y] \not\sim_{mi} t_2[Y](C, I)$  **then**
- 6:      $\forall A \in (Y - X), t_1[A], t_2[A] := t_1[A] \vee t_2[A]$ ;
- 7:   **end if**
- 8: **end while**
- 9: **return** merge(Result);

We call an execution of Line 6 in Algorithm 1 a  $VChase$  step, and say that the  $VChase$  step applies the FD  $X \rightarrow Y$  to the current state of  $VChase(r, F)$ .

**Example 7** The vague relation  $VChase(r, F)$  is shown in Table 8, where  $r$  is shown in Table 4 and  $F = \{S \rightarrow_{0.2,0.9} TL, L \rightarrow_{0.2,0.9} S\}$  is the set of FDs over  $R$ . We can verify that  $VChase(r, F) \models F$ , that is,  $VChase(r, F)$  is consistent, and that  $VChase(r, F) = VChase(merge(r), F)$ , where  $merge(r)$  is shown in Table 5. If  $F = \{S \rightarrow_{0.2,0.5} TL,$

Table 8

Vague relation  $VChase(r, F)$  under  $C = 0.2$  and  $I = 0.9$ 

<b>S</b>	<b>T</b>	<b>L</b>
$\langle 0.8, 0.1 \rangle / 0 + \langle 0.9, 0.2 \rangle / 1$ + $\langle 0.5, 0.1 \rangle / 2 + \langle 0.5, 1 \rangle / 3$	$\langle 0.8, 0.3 \rangle / 0 + \langle 0.9, 0.1 \rangle / 1$ + $\langle 0.5, 0.1 \rangle / 2$	$\langle 0.6, 0.2 \rangle / 0 + \langle 0.6, 0.3 \rangle / 1$ + $\langle 0.5, 0.2 \rangle / 2$
$\langle 0.5, 0.1 \rangle / 3 + \langle 0.8, 0.2 \rangle / 4$	$\langle 0.4, 0.4 \rangle / 3$	$\langle 0.4, 0.2 \rangle / 3$

Table 9

Vague relation  $VChase(r, F)$  under  $C = 0.2$  and  $I = 0.5$ 

<b>S</b>	<b>T</b>	<b>L</b>
$\langle 0.8, 0.1 \rangle / 0 + \langle 0.9, 0.2 \rangle / 1$ + $\langle 0.5, 0.1 \rangle / 2 + \langle 0.5, 1 \rangle / 3$	$\langle 0.8, 0.3 \rangle / 0 + \langle 0.9, 0.1 \rangle / 1$ + $\langle 0.5, 0.1 \rangle / 2$	$\langle 0.4, 0.2 \rangle / 0 + \langle 0.6, 0.3 \rangle / 1$ + $\langle 0.5, 0.2 \rangle / 2$
$\langle 0.5, 0.1 \rangle / 3 + \langle 0.8, 0.2 \rangle / 4$	$\langle 0.4, 0.4 \rangle / 3$	$\langle 0.4, 0.2 \rangle / 3$

$L \rightarrow_{0.2, 0.5} S\}$  is the set of FDs over  $R$ , we can also verify that  $VChase(r, F) \models F$ , which is shown in Table 9, and that  $VChase(r, F) = VChase(merge(r), F)$ , where  $merge(r)$  is shown in Table 6.

From Tables 8 and 9, we see that different mi-thresholds  $C$  and  $I$  may give rise to different  $VChase$  results (the corresponding values of  $L$  in the first tuple).

The next lemma shows that  $VChase(r, F)$  is less  $O$ -precise than  $merge(r)$  and unique. Its complexity is polynomial time in the sizes of  $r$  and  $F$ .

**Lemma 5.3** *The following statements are true:*

- (1)  $VChase(r, F) \sqsubseteq_O merge(r)$ .
- (2)  $VChase(r, F)$  is unique.
- (3)  $VChase(r, F)$  terminates in polynomial time in the sizes of  $r$  and  $F$ .

**Proof.** Part 1. We use an induction on the minimal number,  $k$ , of  $VChase$  steps needed to compute a current state  $s$  of  $VChase(r, F)$ , asserting that  $merge(s) \sqsubseteq_O merge(r)$  after  $k$   $VChase$  steps.

(Basis): If  $k = 0$ , then the result is immediate by Line 9 in Algorithm 1, since  $merge(s) \doteq_O VChase(r, F) \doteq_O merge(r)$ .

(Induction): Assume the result holds when the minimal number of  $VChase$  steps needed to compute  $s$  is  $k$ , with  $k \geq 0$ ; we then prove that the result holds when the minimal number of  $VChase$  steps to compute  $s$  is  $k + 1$ . Let  $t_1, t_2$  be the tuples in Line 6 of Algorithm 1 just after the application of the  $(k + 1)$ th  $VChase$  step. Also, let  $r'$  be the current state of  $VChase(r, F)$  and  $t_1, t_2 \in r'$  be the current states of  $t_1, t_2$ , respectively, prior to the application of the  $(k + 1)$ th  $VChase$  step. From Line 6 we have for  $i = 1, 2$ ,  $t_i \sqsubseteq_O t'_i$  and thus also  $merge(\{t_i\}) \sqsubseteq_O merge(\{t'_i\})$ . Therefore,  $VChase(r, F) \sqsubseteq_O merge(r')$  holds. Furthermore, by inductive hypothesis  $merge(r') \sqsubseteq_O merge(r)$ . The result that  $VChase(r, F) \sqsubseteq_O merge(r)$  follows as required by the transitivity of  $\sqsubseteq_O$ .

Part 2. In order to show that  $VChase(r, F)$  is unique we adopt the technique used in Theorem 4.1 of [5]. We define two  $A$ -values  $t_1[A]$  and  $t_2[A]$ , with  $t_1, t_2 \in r$ , to be  $A$ -equatable

in  $r$  (or simply equatable if  $A$  and  $r$  are understood from context) if one of the following conditions holds:

- (1)  $t_1[A] \sim_{mi} t_2[A]$ , or
- (2) for some FD  $X \rightarrow Y \in F$ , with  $A \in (Y - X)$ ,  $t_1[X] \sim_{mi} t_2[X]$ , or
- (3) for some  $A$ -value,  $t_3[A]$ , with  $t_3 \in r$ , both  $t_1[A]$  and  $t_3[A]$  are equatable and also  $t_2[A]$  and  $t_3[A]$  are equatable, or
- (4) for some FD  $X \rightarrow Y \in F$ , we have that  $\forall A \in X$ ,  $t_1[A]$  and  $t_2[A]$  are equatable.

From the above conditions it follows that equatability is an equivalence relation on the  $A$ -values of the tuples in  $r$ . Moreover, it can be verified that equatability is invariant with respect to  $VChase$  steps for given  $C$  and  $I$ . Therefore,  $t_1[A]$  and  $t_2[A]$  are equatable in  $r$  if and only if either  $t_1[A] \sim_{mi} t_2[A]$  or after some  $VChase$  step  $s_1[A] \sim_{mi} s_2[A]$ , where  $s_1$  and  $s_2$  are the current states of  $t_1$  and  $t_2$ , respectively. Let  $\{c_1, c_2, \dots, c_n\}$  be an equivalence class with respect to  $A$ -equatability in  $r$ . It follows that  $\bigcup_{i \in I} c_i$  is a  $mi$ -overlap set over  $A$  with respect to  $VChase(r, F)$ . Let  $t \in r$  and assume that  $t[A] \sim_{mi} c_i$ , for some  $i \in I$ . The result follows, since by Line 9 in Algorithm 1 which merges the result, we can deduce that, irrespective of the order of  $VChase$  steps,  $s[A] = \bigcup_{i \in I} c_i$ , where  $s$  is the current state of  $t$  in  $VChase(r, F)$ .

Part 3.  $VChase(r, F)$  terminates due to the following argument. Prior to each chase step we have that  $\forall A \in (Y - X)$ ,  $t_1[A], t_2[A] \sqsubset_O t_1[A] \vee t_2[A]$ , since  $t_1[A]$  and  $t_2[A]$  are disjoint. It follows that the number of  $mi$ -existing objects in  $A$ -values of tuples in  $r$  is strictly monotonically increasing with respect to  $\sqsubset_O$ . It follows that  $VChase(r, F)$  terminates, since  $\forall A \in R$ ,  $\mathcal{A}(r, A)$  is finite.

We now show that  $VChase(r, F)$  can be computed in polynomial time in the sizes of  $r$  and  $F$ . By Definition 4.2, testing whether  $t_1[X] \sim_{mi} t_2[X]$  and whether  $t_1[Y] \sim_{mi} t_2[Y]$  can be done in polynomial time in the size of  $r$ , since the reflexive and transitive closure of  $G$  can be computed in polynomial time in the cardinality of  $V$  and the size of  $r$  (both  $\pi_A(r)$  and intersection of sets are polynomial time operations). So, each execution of the while loop beginning at Line 3 and ending at Line 8 can be computed in polynomial time in the sizes of  $r$  and  $F$ . It remains to show that this while loop is executed a polynomial number of times in the sizes of  $r$  and  $F$ , recalling that by Definition 4.2  $merge(Result)$  can be computed in polynomial time in the size of  $r$ .

Let  $m$  be the maximum cardinality, over all  $A \in R$ , of a  $mi$ -overlap set over  $A$  with respect to  $r$ ;  $m$  is less than the size of  $r$ , since  $m \leq |U_A|$ . The result of the first part now follows, since the while loop cannot be executed more than  $m|r||R|$  times due to the fact that the number of  $mi$ -existing objects in  $A$ -values of  $r$  are strictly monotonically increasing after each  $VChase$  step.  $\square$

The next theorem shows that the  $VChase$  procedure outputs a consistent relation and that it commutes with the merge operation. These results justify the chase and the merge operation.

**Theorem 5.4** *The following two statements are true:*

- (1)  $VChase(r, F) \models F$ . (that is,  $VChase(r, F)$  is consistent.)
- (2)  $VChase(r, F) = VChase(merge(r), F)$ .

**Proof.** Part 1. Let  $s = VChase(r, F)$ . Assume that for some  $X \rightarrow Y \in F$ ,  $s \not\models X \rightarrow Y$ .

Then for some  $t_1, t_2 \in s$ ,  $t_1[X] \sim_{mi} t_2[X]$  but  $t_1[Y] \not\sim_{mi} t_2[Y]$ . This leads to a contradiction, since we can deduce that  $s \neq VChase(r, F)$ , due to the fact that the condition of the “if” statement of Line 5 in Algorithm 1 is true.

Part 2. We can view that the merge of  $r$  induces a mapping  $\rho$  from  $r$  to  $merge(r)$  such that  $\rho(t) = t'$ , with  $t \in r$  and  $t' \in merge(r)$ , if and only if  $t \sim_{mi} t'$  in  $s$ , with  $s = \{t, t'\}$ . The result now follows by Lemma 5.3 on using the notion of equatability of  $A$ -values, since two  $A$ -values  $t_1[A]$  and  $t_2[A]$ , with  $t_1, t_2 \in r$ , are  $A$ -equatable in  $r$  if and only if  $(\rho(t_1))[A]$  and  $(\rho(t_2))[A]$  are  $A$ -equatable in  $merge(r)$ .  $\square$

From Lemma 5.3 and Theorem 5.4, we conclude that the  $VChase$  procedure solves the consistency problem in polynomial time in the sizes of  $r$  and  $F$ .

## 6 The most $O$ -precise approximation of a vague relation

In this section, we show that the  $VChase(r, F)$  procedure can be regarded as the most  $O$ -precise approximation of  $r$ , which is also consistent to  $F$ .

We first define the *join* of vague relations, which corresponds to the least upper bound of these relations in the lattice  $MERGE(R)$  based on  $O$ -equivalence classes. (Recall the lattices shown in Figures 5 and 6.) Next, we define the *most  $O$ -precise approximation* of  $r$  with respect to  $F$  to be the *join* of all the consistent and merged relations which are less  $O$ -precise than  $r$ . Our main result in this section is that  $VChase(r, F)$  is the most  $O$ -precise approximation of  $r$  with respect to  $F$ .

We now define the join operation on relations in the lattice of  $MERGE(R)$  based on  $O$ -equivalence classes.

**Definition 6.1 (Join operation)** *The join of two vague relations,  $r_1, r_2 \in MERGE(R)$ , denoted by  $r_1 \sqcup r_2$ , is given by*

$$r_1 \sqcup r_2 = \{t \mid \exists t_1 \in r_1, \exists t_2 \in r_2 \text{ such that } \forall A \in R, t_1[A] \sim_{mi} t_2[A](C, I), t[A] = t_1[A] \wedge t_2[A]\}.$$

It can be verified that the  $O$ -equivalence class that consists of  $r_1 \sqcup r_2$  is the least upper bound with respect to the  $O$ -equivalence classes of  $r_1$  and  $r_2$  in  $MERGE(R)$ . From now on we assume that  $r_1, r_2 \in MERGE(R)$ .

The next theorem shows that if two relations are consistent then their join is also consistent.

**Theorem 6.1** *Let  $r_1, r_2 \in SAT(F)$ . Then  $r_1 \sqcup r_2 \in SAT(F)$ .*

**Proof.** Let  $r = r_1 \sqcup r_2$  and suppose that  $r \notin SAT(F)$ . It follows that for some FD,  $X \rightarrow Y \in F$  and for some attribute  $A \in Y$ ,  $r \not\vdash X \rightarrow A$ . Thus,  $\exists u_1, u_2 \in r$  such that  $u_1[X] \sim_{mi} u_2[X]$  but  $u_1[A] \not\sim_{mi} u_2[A]$ . Moreover,  $u_1[X] \doteq_O u_2[X]$  and  $u_1[A] \wedge u_2[A] \doteq_O \emptyset^O$ . It follows by Definition 6.1 that  $\exists t_1, t'_1 \in r_1, t_2, t'_2 \in r_2$  such that  $\{u_1\} = \{t_1\} \sqcup \{t_2\}$  and  $\{u_2\} = \{t'_1\} \sqcup \{t'_2\}$ . Thus,  $u_1[X] \sqsubseteq_O t_1[X], t_2[X]$  and  $u_2[X] \sqsubseteq_O t'_1[X], t'_2[X]$ . It follows that  $t_1[X] \wedge t'_1[X] \neq_O \emptyset^O$  and also  $t_2[X] \wedge t'_2[X] \neq_O \emptyset^O$ . Therefore, since  $r_1, r_2 \in MERGE(R)$  we have that  $t_1[X] \doteq_O t'_1[X]$  and  $t_2[X] \doteq_O t'_2[X]$ . Thus,  $t_1[A] \doteq_O t'_1[A]$  and  $t_2[A] \doteq_O t'_2[A]$ ,

since  $r_1, r_2 \in SAT(F)$ . It follows that  $(t_1[A] \wedge t_2[A]) \doteq_O (t'_1[A] \wedge t'_2[A])$ . This leads to  $u_1[A] \doteq_O u_2[A]$ , which contradicts to our assumption.  $\square$

The most  $O$ -precise approximation of a vague relation  $r$  over  $R$  with respect to  $F$  is the join of all consistent relations  $s$  such that  $s$  is a merged relation that is less  $O$ -precise than  $r$ .

**Definition 6.2 (Most  $O$ -precise approximation)** *The most  $O$ -precise approximation of a vague relation  $r$  with respect to  $F$ , denoted by  $approx(r, F)$ , is given by  $\sqcup\{s \mid s \sqsubseteq_O merge(r) \text{ and } s \in SAT(F)\}$ .*

The next lemma shows some desirable properties of the approximation.

**Lemma 6.2** *The following statements are true:*

- (1)  $approx(r, F)$  is consistent.
- (2)  $approx(r, F) \sqsubseteq_O merge(r)$ .
- (3)  $approx(r, F) \doteq_O merge(r)$  iff  $r$  is consistent.

**Proof.** Part 1. The result follows by Theorem 6.1, since by Definition 6.2  $approx(r, F)$  is the join of consistent and merged relations.

Part 2. The result follows by noting that the join of a collection of relations in  $approx(r, F)$  corresponds to their least upper bound due to the commutativity and associativity of the join.

Part 3. The *if part* of the statement follows by Definition 6.2, since if  $r$  is consistent then  $r \in SAT(F)$ . The *only if part* of the statement follows by Theorem 6.1 and Proposition 5.1 Part 2, since by Definition 6.2  $approx(r, F)$  is the join of consistent relations.  $\square$

The next theorem shows that output of the  $VChase$  procedure is equal to the corresponding most  $O$ -precise approximation. Thus, the vague relation  $VChase(r, F)$ , which is shown in Table 8, is the most  $O$ -precise approximation of  $r$  with respect to  $F$ , where  $r$  is the relation shown in Table 4 and  $F$  is the set of FDs specified in Example 6.

**Theorem 6.3**  $VChase(r, F) \doteq_O approx(r, F)$ .

**Proof.** We establish the proof by showing  $VChase(r, F) \sqsubseteq_O approx(r, F)$  and  $approx(r, F) \sqsubseteq_O VChase(r, F)$ .

$(VChase(r, F) \sqsubseteq_O approx(r, F))$ : By Lemma 5.3 Part 2,  $VChase(r, F) \sqsubseteq_O merge(r)$ . By Theorem 5.4 Part 1 and Line 9 in Algorithm 1,  $VChase(r, F) \in SAT(F)$ . Thus, by Definition 6.2,  $VChase(r, F) \sqsubseteq_O approx(r, F)$ , since  $F$  corresponds to the least upper bound operation in  $MERGE(R)$ .

$(approx(r, F) \sqsubseteq_O VChase(r, F))$ : We use an induction on the minimal number,  $k$ , of  $VChase$  steps needed to compute a current state  $s$  of  $VChase(r, F)$ , asserting that  $approx(r, F) \sqsubseteq_O merge(s)$  after  $k$   $VChase$  steps.

(Basis): If  $k = 0$ , then the result follows immediately by Part 3 of Lemma 6.2, since  $s = r$  and by Line 9 in Algorithm 1,  $VChase(r, F) \doteq_O merge(r)$ .

(Induction): Assume the result holds when the minimal number of  $VChase$  steps needed to compute  $s$  is  $k$ , with  $k \geq 0$ . We then need to prove that the result holds when the minimal

number of  $VChase$  steps to compute  $s$  is  $(k+1)$ . Let  $r'$  be the current state of  $VChase(r, F)$  prior to the application of the  $(k+1)$ th  $VChase$  step.

Thus, by inductive hypothesis,  $approx(r, F) \sqsubseteq_O merge(r')$ . By inspection of Line 6 in Algorithm 1 we can deduce that  $VChase(r, F)$  is the maximal consistent and merged relation with respect to  $v$ , which is less precise than  $merge(r')$ . That is,  $VChase(r, F) \sqsubseteq_O merge(r')$  and if for some  $s' \sqsubseteq_O SAT(F)$ , we have that  $s' \sqsubseteq_O merge(r')$ , then it is also true that  $s' \sqsubseteq_O VChase(r, F)$ . The result follows, since by Lemma 6.2 Part 2,  $approx(r, F) \in SAT(F)$ .  $\square$

## 7 Inclusion dependencies and vague chase

In this section we consider inclusion dependencies (INDs) [6,7] being satisfied in a vague database, since INDs are also the fundamental data dependencies that arise in practice [4,5]. Satisfaction of an IND can be formalized in terms of less  $V$ -precise order,  $\sqsubseteq_V$  as given in Definition 4.4.

We assume  $C > 0$  or  $I < 1$  and define INDs in a vague database as follows.

**Definition 7.1 (Inclusion dependency)** *Given  $mi$ -thresholds  $C$  and  $I$ , an inclusion dependency (or simply an IND), is a statement of the form  $R[X] \sqsubseteq_{C,I} S[Y]$ . An IND  $R[X] \sqsubseteq_{C,I} S[Y]$  is satisfied in a vague database  $d$ , if  $r[X] \sqsubseteq_V s[Y](C, I)$ , where  $r, s \in d$  are vague relations over  $R$  and  $S$ , respectively. We may simply write  $R[X] \subseteq S[Y]$  if  $C$  and  $I$  are known from the context.*

Table 10

Sensor relation  $r_4$

	$S_f$	<b>T</b>	<b>L</b>
$t_4$	$\langle 0.8, 0.1 \rangle / 0$	$\langle 0.9, 0 \rangle / 0 + \langle 0.8, 0.2 \rangle / 1$	$\langle 0.7, 0.2 \rangle / 1 + \langle 0.6, 0.3 \rangle / 2$

Table 11

Sensor relation  $r_5$

	$S_b$	<b>T</b>	<b>L</b>
$t_5$	$\langle 0.9, 0.1 \rangle / 0$	$\langle 0.8, 0.1 \rangle / 1 + \langle 0.7, 0.3 \rangle / 2$	$\langle 0.6, 0.1 \rangle / 1$
$t_6$	$\langle 0.7, 0.2 \rangle / 0$	$\langle 0.9, 0.2 \rangle / 0 + \langle 0.7, 0.1 \rangle / 2$	$\langle 0.7, 0.1 \rangle / 1 + \langle 0.8, 0.2 \rangle / 2$

Table 12

Sensor relation  $r_6$

	$S_b$	<b>T</b>	<b>L</b>
$t_5$	$\langle 0.9, 0.1 \rangle / 0$	$\langle 0.8, 0.1 \rangle / 1 + \langle 0.7, 0.3 \rangle / 2$	$\langle 0.6, 0.1 \rangle / 1$
$t_6$	$\langle 0.7, 0.2 \rangle / 0$	$\langle 0.9, 0.2 \rangle / 0 + \langle 0.7, 0.1 \rangle / 2$	$\langle 0.7, 0.1 \rangle / 1 + \langle 0.8, 0.2 \rangle / 2$
$t_7$	$\langle 0.8, 0.1 \rangle / 0$	$\langle 0.9, 0 \rangle / 0 + \langle 0.8, 0.2 \rangle / 1$	$\chi$

**Example 8** *Here we give a simplified example of inconsistency arising from IND in a vague database to illustrate the semantics.*

Suppose a vague database  $d$  contains two vague relations,  $r_4$  over  $R_4 = \langle S_f, T, L \rangle$  and  $r_5$  over  $R_5 = \langle S_b, T, L \rangle$ , as shown in Tables 10 and 11, respectively. Here  $S_f$  represents

the sensors of a particular floor of a building and  $S_b$  represents the sensors of the whole building as the central control. Let the  $mi$ -thresholds be  $C = 0.5$  and  $I = 0.5$ .

It can be verified that  $t_4[S_f, T] \not\sqsubseteq_V t_5[S_b, T]$  and  $t_4[S_f, T] \not\sqsubseteq_V t_6[S_b, T]$ . (It can be checked that  $t_4[S_f] \sqsubseteq_V t_5[S_b]$ , but  $t_4[T] \not\sqsubseteq_V t_5[T]$  and  $t_4[S_f] \not\sqsubseteq_V t_6[S_b]$ .) Thus,  $d$  is inconsistent, since it does not satisfy the IND,  $R_4[S_f, T] \subseteq R_5[S_b, T]$ .

Suppose later a vague tuple  $t_7$  is added into  $r_5$ , we have the vague relation  $r_6$  shown in Table 12. It can be verified that the database containing  $r_4$  and  $r_6$  satisfies  $R_4[S_f, T] \subseteq R_5[S_b, T]$  and  $d$  is now consistent, since  $t_4[S_f, T] \sqsubseteq_V t_7[S_b, T]$ . Recall that  $\chi$  is the boundary VS in which all objects have the membership  $<0, 1>$  as defined in Definition 3.6. Intuitively, this is the least change in vague value precision in  $r_5[L]$  in order to form  $t_7$ .

Given a database schema  $\mathcal{R}$  such that  $R_1, R_2, R_3 \in \mathcal{R}$ ,  $N$  is a set of INDs over  $\mathcal{R}$ . Here we state the well known Casanova et al.'s axiom system [6,7] for crisp database  $d$  as follows.

- (1) *Reflexivity*: If  $X \subseteq \text{schema}(R)$ , then  $d \models R[X] \subseteq R[X]$  holds.
- (2) *Projection and permutation*: If  $d \models R_1[X] \subseteq R_2[Y]$  holds, where  $X = \langle A_1, \dots, A_m \rangle \subseteq R_1$ ,  $Y = \langle B_1, \dots, B_m \rangle \subseteq R_2$  and  $\langle i_1, \dots, i_k \rangle$  is a sequence of distinct integers from  $\{1, \dots, m\}$ , then  $d \models R_1[A_{i_1}, \dots, A_{i_k}] \subseteq R_2[B_{i_1}, \dots, B_{i_k}]$  holds.
- (3) *Transitivity*: If  $d \models R_1[X] \subseteq R_2[Y]$  and  $d \models R_2[Y] \subseteq R_3[Z]$  hold, then  $d \models R_1[X] \subseteq R_3[Z]$  holds.

We assume a similar notation and the usual concepts of inference rules (cf. Definition 5.2), which are employed to establish the soundness and completeness of INDs in vague databases.

**Theorem 7.1** *Casanova et al.'s axiom system is sound and complete for INDs in vague databases.*

**Proof.** Let  $I_1$  be an IND over  $\mathcal{R}$ . Let  $CRISPDB(\mathcal{R})$  be the set of all possible crisp databases over  $\mathcal{R}$ , and  $VDB(\mathcal{R})$  be the set of all possible vague databases over  $\mathcal{R}$ . Clearly,  $CRISPDB(\mathcal{R}) \subseteq VDB(\mathcal{R})$ , since a crisp database can be expressed by a vague database by transforming a crisp data value  $u$  to a vague element  $\langle 1, 0 \rangle / u$ . The soundness of the axiom system can easily be verified. We prove completeness by showing that if  $N \not\models I_1$ , then  $N \not\models I_1$ . Equivalently, we need to show that there exists a database  $d \in VDB(\mathcal{R})$  such that

$$\forall I_2 \in N, d \models I_2 \text{ but } d \not\models I_1. \quad (1)$$

When only crisp databases are involved, by the completeness of the axiom system for INDs (cf. [7]),  $\exists d \in CRISPDB(\mathcal{R})$  satisfying Equation (1). Since  $CRISPDB(\mathcal{R}) \subseteq VDB(\mathcal{R})$ , the result follows.  $\square$

For convenience in discussion, we now define the FD chase rule and the IND chase rule in vague databases. The FD chase rule is the same as in Line 5 of Algorithm 1.

**FD chase rule:** if  $X \rightarrow_{C,I} Y \in F$ ,  $\exists t_1, t_2 \in \text{Result}$  such that  $t_1[X] \sim_{mi} t_2[X](C, I)$ ,  $t_1[Y]$  and  $t_2[Y]$  are  $mi$ -existing but  $t_1[Y] \not\sim_{mi} t_2[Y](C, I)$ , then  $\forall A \in (Y - X)$ , change  $t_1[A]$  and  $t_2[A]$  to  $t_1[A] \vee t_2[A]$ .

**IND chase rule:** if  $R[X] \subseteq_{C,I} S[Y] \in N$  and  $\exists t_1 \in r$  such that  $\nexists t_2 \in s$  with  $t_1[X] \sqsubseteq_V t_2[Y](C, I)$ , then add a tuple  $t_2$  over  $S$  to  $s$ , satisfying  $t_2[Y] = t_1[X]$  and  $\forall A \in (S - Y)$ ,  $t_2[A] = \chi$ .

Now, we present the extended chase algorithm,  $VChase(d, \Sigma)$ , where  $\Sigma = F \cup N$ , in Algorithm 2. We let  $merge(d) = \{merge(r) \mid r \in d\}$ .

**Algorithm 2.**  $VChase(d, \Sigma)$

- 1: Result :=  $d$ ;
- 2: Tmp :=  $\emptyset$ ;
- 3: **while** Tmp  $\neq$  Result **do**
- 4:   Ttmp := Result;
- 5:   Apply the FD chase rule or the IND chase rule to Result;
- 6: **end while**
- 7: **return** merge(Result);

$VChase(d, \Sigma)$  terminates since the number of objects in a vague database  $d$  is finite and the vague values are monotonically increasing but bounded. However, we observe that the restriction of the application order of the chase rules in Line 5 of Algorithm 2 actually affects the output of the consistent database. For example, it may be possible that  $VChase(d, \Sigma) \neq VChase(VChase(d, F), N)$ . To explain this problem we now present an example to illustrate that, in general, we cannot apply all FD chase rules first and then IND chase rules to obtain a consistent output. Fortunately, the converse can always hold: we can apply all chase steps for INDs first and then all chase steps for FDs.

Table 13

Sensor relation  $r_8$

S	$T_1$	$T_2$
$\langle 0.8, 0.1 \rangle / 0$	$\langle 0.9, 0 \rangle / 0$	$\langle 0.7, 0.2 \rangle / 1$

Table 15

Sensor relation  $r_{10}$

S	T
$\langle 0.8, 0.1 \rangle / 0$	$\langle 0.9, 0 \rangle / 0$
$\langle 0.8, 0.1 \rangle / 0$	$\langle 0.7, 0.2 \rangle / 1$

Table 14

Sensor relation  $r_9$

S	T
$\langle 0.8, 0.1 \rangle / 0$	$\langle 0.9, 0 \rangle / 0$

Table 16

Sensor relation  $r_{11}$

S	T
$\langle 0.8, 0.1 \rangle / 0$	$\langle 0.9, 0 \rangle / 0 + \langle 0.7, 0.2 \rangle / 1$

**Example 9** Suppose  $d$  containing two vague relations  $r_8$  over  $R_8$  and  $r_9$  over  $R_9$ , respectively, as shown in Tables 13 and 14. Let  $C = 0.5$  and  $I = 0.5$ ,  $F = \{R_9 : S \rightarrow T\}$  and  $N = \{R_8[S, T_1] \subseteq R_9[S, T], R_8[S, T_2] \subseteq R_9[S, T]\}$ . It can be verified that if we apply the FD chase rule first and then the IND chase rule,  $r_9$  is changed to  $r_{10}$  as shown in Table 15 by the IND chase rule (Since  $r_9$  satisfies  $F$ , no FD chase rule needs to be applied.). After the final merge operation in  $VChase$ ,  $r_{10}$  is unchanged. However,  $r_{10}$  does not satisfy  $F$ . Therefore, the database containing  $r_8$  and  $r_{10}$  is inconsistent. That is,  $VChase(VChase(d, F), N)$  is inconsistent when the restriction of the rule application order is imposed. If we apply the IND chase rule first and then the FD chase rule,  $r_9$  is changed to  $r_{10}$  by the IND chase rule first and then  $r_{11}$  by FD rules and the final merge operation, as shown in Table 16. It can be checked that the database containing  $r_8$  and  $r_{11}$  is consistent. That is,  $VChase(VChase(d, N), F)$  is consistent.

We now formalize our observation of applying the chase rules by the following theorem. The significance of this result is that it guarantees that we have a definite way to apply the chase rules in order to maintain the consistency with respect to a mixed set of FDs and INDs.

**Theorem 7.2** Given a mixed set of FDs and INDs  $\Sigma = F \cup N$  and a vague database  $d$ , the following statements are true:

- (1)  $VChase(d, \Sigma) \models \Sigma$ .
- (2)  $VChase(d, \Sigma) = VChase(VChase(d, N), F)$ .

**Proof.** Part 1. Due to the two chase rules we have the consistent  $VChase(r, \Sigma)$ , which is similar to the argument in Part 1 of Theorem 5.4.

Part 2. The result is followed by the definition of the FD rule, since due to the consistency of  $VChase(d, \Sigma)$  from Part 1, by applying the FD rule to any state  $e$  of the database  $d$  during the execution of Algorithm 2, which unions non- $O$ -equivalent vague values. If some modifications by the IND rule change the  $O$ -equivalence, then by Step 7, the union can also be done by  $merge(r)$ .  $\square$

The above theorem implies that the chase can be applied in a deterministic way as follows: we apply the IND rule for as long as possible and then the FD rule for as long as possible.

We now present the main result of the complexity of  $VChase$  with respect to  $\Sigma$ . In contrast to the well known result of undecidable complexity for the implication problem for FDs and INDs in crisp databases [7], the complexity of  $VChase$  for maintaining consistency with respect to  $\Sigma$  is exponential in time complexity.

**Theorem 7.3** The time complexity of  $VChase(d, \Sigma)$  is exponential in the sizes of  $d$  and  $\Sigma$ .

**Proof.** Assume without loss of generality, we let  $\mathcal{R} = \{R_1, \dots, R_n\}$  such that  $\forall i \in \{1, \dots, n\}$ ,  $R_i = \{A_1, \dots, A_n\}$ . We let  $d = \langle r_1, \dots, r_n \rangle$  (a list of  $n$  relations), where  $r_1 = \{(v_1, \dots, v_n)\}$  (one tuple), with  $v_1, \dots, v_n$  being  $n$  distinct  $mi$ -existing VSs such that  $\forall i \in \{2, \dots, n\}$ ,  $r_i \in \emptyset^O$ . We consider the chase arising from this particular tuple in  $r_1$ .

Let  $|VChase(d, N)| = \sum_{i=1}^n |r_i|$ , where  $r_i \in VChase(d, N)$  is the relation over  $R_i \in \mathcal{R}$ . We need to show a set of INDs  $N$  over  $\mathcal{R}$  such that  $|N| = 2(n-1)$  and  $|VChase(d, N)| = 2^n - 1$ . Let  $\alpha_i$  denote the IND,  $R_i[A_1, \dots, A_n] \subseteq R_{i+1}[A_1, \dots, A_n]$ , and  $\beta_i$  denote the IND,  $R_i[A_1, \dots, A_{i+1}, A_i, \dots, A_n] \subseteq R_{i+1}[A_1, \dots, A_{i+1}, A_i, \dots, A_n]$ , where  $i \in \{1, \dots, (n-1)\}$ . Intuitively,  $\alpha_i$  inserts the identical set of tuples from  $R_i$  into  $R_{i+1}$  and  $\beta_i$  swaps the  $i$ th attribute with the  $(i+1)$ th attribute in  $R_i$ . We let  $N = \cup_{i \in \{1, \dots, (n-1)\}} I_i$ , where  $I_i = \{\alpha_i\} \cup \{\beta_i\}$ .

We obtain the result by induction on  $i \in \{1, \dots, n\}$ . For all  $i$ , the cardinality of the relation  $r_i \in VChase(d, N)$  over  $R_i \in \mathcal{R}$  is  $2^{i-1}$ . We need to establish the following claim:

Claim(\*):  $|VChase(d, N)| = 2^n - 1$ .

(Basis). If  $i = 1$ ,  $|r_1| = 1$ . The result holds.

(Induction). Assume that the result holds for  $i \geq 1$ . We need to prove that the result holds for  $i + 1$ . By inductive hypothesis  $|r_i| = 2^i - 1$ . By the definition of  $I_i$ ,  $\pi_{A_{i+1}}(r_i) = \{(v_{i+1})\}$  and  $(\pi_{A_i}(r_i) \wedge \pi_{A_{i+1}}(r_i)) \in \emptyset^O$ . Thus  $r_{i+1}$  has twice as many tuples as  $r_i$  due to  $I_{i+1}$ . It follows that Claim(\*) is established.

We let  $q = |\mathcal{A}(d)|$  and  $k$  denote the maximal cardinality for all  $R_i \in \mathcal{R}$ . Then the maximal cardinality of  $VChase(d, \Sigma)$  is bounded by  $q^k$ . Thus, using Lemma 5.3 Part 3 and Theorem 7.2, it follows that the time complexity of computing  $VChase(d, \Sigma)$  is exponential.  $\square$

Now, we evaluate the quality of  $VChase(d, N)$  by showing that the value precision difference between  $d$  and  $VChase(d, N)$  is minimum.

When applying the IND chase rule, we actually add a new tuple  $t$  into  $r \in d$  and obtain an updated (new) relation  $r' \in d'$ . We now define the notions of  $\Delta_V(r, r')$  and  $\Delta_V(d, d')$  which measure the *value precision change* between the relations  $r$  and  $r'$ , and between the databases  $d$  and  $d'$ , respectively. The value precision difference between them is defined in a progressive manner as follows.

- (1) Given two VSs  $V_1$  and  $V_2$ ,  $\Delta(V_1, V_2) = \frac{(\Delta m + \Delta i)}{2}$ , where  $\Delta m = \frac{(|\Delta m_1| + \dots + |\Delta m_n|)}{n}$  and  $\Delta i = \frac{(|\Delta i_1| + \dots + |\Delta i_n|)}{n}$  are the average of the difference between the median and imprecision memberships in  $V_1$  and  $V_2$  for all  $\langle m_k, i_k \rangle / u_k$  where  $k = 1, \dots, n$ .
- (2)  $\Delta_V(r, r') = \sum_{A_i \in R} \Delta(\mathcal{A}(r, A_i), \mathcal{A}(r', A_i))$ .
- (3)  $\Delta_V(d, d') = \sum_{r_i \in d} \Delta_V(r_i, r'_i)$ .

Note that using the arithmetic mean to define  $\Delta m$  and  $\Delta i$  is just one possible choice. Other monotonically increasing functions, such as the geometric mean, can also make the following theorem hold. The significance of formulating the value precision difference here is that the IND chase rule guarantees the minimum possible change in the median and imprecision memberships.

**Theorem 7.4** *Given  $d$  and  $N$ ,  $\Delta(d, VChase(d, N))$  is minimum.*

**Proof.** Let  $R[X] \subseteq S[Y]$  be the IND involved in the last step of the  $VChase$  procedure, where  $X = \langle A_1, \dots, A_m \rangle$  and  $Y = \langle B_1, \dots, B_m \rangle$ . There are two scenarios for applying the IND chase rule arising from  $t \in r$ .

- (1) There is no *mi*-existing object that is the same as a *mi*-existing object of  $r[A_i]$  in  $s[B_i]$ , or some *mi*-existing object that is the same as a *mi*-existing object of  $r[A_i]$  in  $s[B_i]$  but has smaller value precision. Then obviously the IND chase rule adds new elements of  $r[A_i]$  into  $s[B_i]$ , which results in the minimum change in value precision of  $s$  and  $d$ , since the memberships of the corresponding elements in  $s[B_i]$  are just equal to those in  $r[A_i]$ , which is the minimum value precision needed to fix the inconsistency according to Definition 7.1.
- (2) There are some *mi*-existing objects that are the same as a *mi*-existing object of  $r[A_i]$  in  $s[B_i]$  but has larger value precision. That is, if the memberships of the corresponding objects in  $r[A_i]$  are less  $V$ -precise than those in  $s[B_i]$ , then the chase rule adds the corresponding elements of  $r[A_i]$  in  $s[B_i]$  which results in no change in value precision of  $s$  and  $d$ .

It follows that  $\Delta(d, VChase(d, N))$  is minimum according to the two scenarios mentioned above.  $\square$

According to Theorems 6.3, 7.2 and 7.4, we can conclude that the  $VChase$  procedure returns the answer that has the most  $O$ -precise approximation and the minimum change in value precision.

## 8 Conclusions

In this paper, we extend FDs and INDs to be satisfied in a vague database. We define the concept of *mi*-overlap between vague sets and the merge operation of a vague relation  $r$ .

The merge operation replaces each attribute value in  $r$  by the  $mi$ -union of all attribute values with respect to the same reflexive and transitive closures under  $mi$ -overlap. We also define a partial order on merged vague relations which induces a lattice on the set of merged vague relations based on  $O$ -equivalence classes. Inside each  $O$ -equivalence class, we define another partial order based on the precision of vague values of a  $mi$ -existing object which induces a complete semi-lattice.

We study the semantics of FDs and INDs in vague databases. First, satisfaction of an FD in a vague relation is defined in terms of values being  $mi$ -overlapping rather than equality. Second, satisfaction of an IND in a vague database is defined in terms of the value precision order of  $mi$ -existing objects rather than usual set containment of objects. Lien's and Atzeni's axiom system is sound and complete for FDs being satisfied in vague relations whereas Casanova et al.'s axiom system is sound and complete for INDs being satisfied in vague databases. We define the chase procedure  $VChase$  as a means of maintaining consistency of  $d$  with respect to  $F$  and  $N$ .

Our main results relate to maintaining the consistency of vague databases by using  $VChase$ . The chase procedure outputs the most  $O$ -precise approximation of  $r$  with respect to  $F$  and can be computed in polynomial time in the sizes of  $r$  and  $F$ . On the other hand,  $VChase$  still outputs a consistent database having the minimum change in value precision in exponential time when INDs are taken into account. Our result suggests a mechanized way that maintains the consistency of vague data with respect to fundamental data dependencies. It is both interesting and challenging to use the chase result to facilitate more effective and efficient evaluation of SQL over vague relations [3] as future work.

## References

- [1] L. A. Zadeh, Fuzzy sets, *Information and Control* 8 (3) (1965) 338–353.
- [2] W.-L. Gau, D. J. Buehrer, Vague sets, *IEEE Trans. Syst., Man, Cybern.* 23 (2) (1993) 610–614.
- [3] A. Lu, W. Ng, Vague sets or intuitionistic fuzzy sets for handling vague data: Which one is better?, in: *ER*, 2005, pp. 401–416.
- [4] J. D. Ullman, *Principles of Database and Knowledge-Base Systems, Volume I*, Computer Science Press, 1988.
- [5] P. Atzeni, V. D. Antonellis, *Relational Database Theory*, Benjamin/Cummings, 1993.
- [6] J. C. Mitchell, The implication problem for functional and inclusion dependencies, *Information and Control* 56 (3) (1983) 154–173.
- [7] M. A. Casanova, R. Fagin, C. H. Papadimitriou, Inclusion dependencies and their interaction with functional dependencies, *J. Comput. Syst. Sci.* 28 (1) (1984) 29–59.
- [8] Y. E. Lien, On the equivalence of database models, *J. ACM* 29 (2) (1982) 333–362.
- [9] R. Cavallo, M. Pittarelli, The theory of probabilistic databases, in: P. M. Stocker, W. Kent, P. Hammersley (Eds.), *VLDB*, Morgan Kaufmann, 1987, pp. 71–81.

- [10] D. Barbará, H. Garcia-Molina, D. Porter, The management of probabilistic data, *IEEE Trans. Knowl. Data Eng.* 4 (5) (1992) 487–502.
- [11] D. Dey, S. Sarkar, A probabilistic relational model and algebra, *ACM Trans. Database Syst.* 21 (3) (1996) 339–369.
- [12] D. Dey, S. Sarkar, Generalized normal forms for probabilistic relational data, *IEEE Trans. Knowl. Data Eng.* 14 (3) (2002) 485–497.
- [13] L. V. S. Lakshmanan, N. Leone, R. Ross, V. S. Subrahmanian, Proview: A flexible probabilistic database system, *ACM Trans. Database Syst.* 22 (3) (1997) 419–469.
- [14] R. Ross, V. S. Subrahmanian, J. Grant, Aggregate operators in probabilistic databases, *J. ACM* 52 (1) (2005) 54–101.
- [15] L. A. Zadeh, Fuzzy sets as a basis for a theory of possibility, *Fuzzy Sets and Systems* 1 (1) (1978) 3–28.
- [16] P. Bosc, H. Prade, An introduction to the fuzzy set and possibility theory-based treatment of flexible queries and uncertain or imprecise databases., in: A. Motro, P. Smets (Eds.), *Uncertainty Management in Information Systems: From Needs to Solutions*, Kluwer Academic Publ., 1996, pp. 285–324.
- [17] D. Dubois, H. Prade, *Possibility Theory: An Approach to Computerized Processing of Uncertainty*, Plenum Press, New York, 1988.
- [18] N. C. Ho, W. Wechler, Extended hedge algebras and their application to fuzzy logic, *Fuzzy Sets and Systems* 52 (3) (1992) 259–281.
- [19] A. Lu, W. Ng, Managing merged data by vague functional dependencies, in: *ER*, 2004, pp. 259–272.
- [20] A. Lu, Y. Ke, J. Cheng, W. Ng, Mining vague association rules, in: *DASFAA*, 2007, pp. 891–897.
- [21] A. Lu, W. Ng, Mining hesitation information by vague association rules, in: *ER*, 2007, pp. 39–55.
- [22] A. Lu, W. Ng, Handling inconsistency of vague relations with functional dependencies, in: *ER*, 2007, pp. 229–244.
- [23] S. An, F. Schorfheide, Bayesian analysis of DSGE models, *Econometric Reviews* 26 (6) (2007) 113–172.
- [24] R. Jansen, H. Yu, D. Greenbaum, Y. Kluger, N. J. Krogan, S. Chung, A. Emili, M. Snyder, J. F. Greenblatt, M. Gerstein, A Bayesian networks approach for predicting protein-protein interactions from genomic data, *Science* 302 (5644) (2003) 449–453.
- [25] S. Sun, C. Zhang, G. Yu, A Bayesian network approach to traffic flow forecasting, *IEEE Trans. Intelligent Transportation Syst.* 7 (1) (2006) 124–132.
- [26] J. Aitchison, I. R. Dunsmore, *Statistical Prediction Analysis*, Cambridge University Press, 1975.
- [27] J. M. Mendel, R. I. B. John, Type-2 fuzzy sets made simple, *IEEE Trans. Fuzzy Syst.* 10 (2) (2002) 117–127.
- [28] R. Sambuc, *Fonctions  $\phi$ -floues. Application a l’aide au diagnostic en pathologie thyroïdienne*, Ph.D. thesis, Univ. Marseille, France (1975).

- [29] K. T. Atanassov, Intuitionistic fuzzy sets, Seventh Scientific Session of ITKR, Sofia(Dep. in CINTI, Nd 1697/84).
- [30] J.-L. Deng, Control problems of grey systems, *Systems & Control Letters* 1 (5) (1982) 288–294.
- [31] H. Bustince, P. Burillo, Vague sets are intuitionistic fuzzy sets, *Fuzzy Sets and Systems* 79 (3) (1996) 403–405.
- [32] D. Dubois, S. Gottwald, P. Hájek, J. Kacprzyk, H. Prade, Terminological difficulties in fuzzy set theory - the case of “intuitionistic fuzzy sets”, *Fuzzy Sets and Systems* 156 (3) (2005) 485–491.
- [33] C. Cornelis, K. T. Atanassov, E. E. Kerre, Intuitionistic fuzzy sets and interval-valued fuzzy sets: a critical comparison, in: *Proc. Third European Conf. on Fuzzy Logic and Technol. (Eusflat’03)*, Zittau, Germany, 2003, pp. 159–163.
- [34] G. Deschrijver, E. E. Kerre, On the relationship between some extensions of fuzzy set theory, *Fuzzy Sets and Systems* 133 (2) (2003) 227–235.
- [35] D. Bitton, J. Millman, S. Torgersen, A feasibility and performance study of dependency inference, in: *ICDE*, IEEE Computer Society, 1989, pp. 635–641.
- [36] M. Levene, Maintaining consistency of imprecise relations, *Comput. J.* 39 (2) (1996) 114–123.
- [37] J. Wijsen, Condensed representation of database repairs for consistent query answering, in: *ICDT*, 2003, pp. 378–393.
- [38] J. Wijsen, Database repairing using updates, *ACM Trans. Database Syst.* 30 (3) (2005) 722–768.
- [39] T. Topaloglou, S. B. Davidson, H. V. Jagadish, V. M. Markowitz, E. W. Steeg, M. Tyers, Biological data management: Research, practice and opportunities, in: *VLDB*, 2004, pp. 1233–1236.
- [40] N. Khoussainova, M. Balazinska, D. Suciú, Towards correcting input data errors probabilistically using integrity constraints, in: *MobiDE*, 2006, pp. 43–50.
- [41] K. V. S. V. N. Raju, A. K. Majumdar, Fuzzy functional dependencies and lossless join decomposition of fuzzy relational database systems, *ACM Trans. Database Syst.* 13 (2) (1988) 129–166.
- [42] P. Bosc, D. Dubois, H. Prade, Fuzzy functional dependencies-an overview and a critical discussion, in: *Proceedings of the Third IEEE Conference on Fuzzy Systems, 1994. IEEE World Congress on Computational Intelligence.*, Vol. 1, 1994, pp. 325–330.
- [43] P. Bosc, D. Dubois, H. Prade, Fuzzy functional dependencies and redundancy elimination, *Journal of the American Society for Information Science* 49 (3) (1998) 217–235.
- [44] W. Zhang, K. Wang, An efficient evaluation of a fuzzy equi-join using fuzzy equality indicators, *IEEE Trans. Knowl. Data Eng.* 12 (2) (2000) 225–237.
- [45] Q. Yang, W. Zhang, C. Liu, J. Wu, C. T. Yu, H. Nakajima, N. Risse, Efficient processing of nested fuzzy SQL queries in a fuzzy database, *IEEE Trans. Knowl. Data Eng.* 13 (6) (2001) 884–901.
- [46] A. K. Sharma, A. Goswami, D. K. Gupta, Discovery of fuzzy inclusion dependencies in fuzzy relational databases, in: *ISCC*, IEEE Computer Society, 2004, pp. 128–133.