Input: a set T of FAIs over Y, a fuzzy set $M \in \mathbf{L}^Y$ of attributes,
and a flag $PCLOSED \in \{false, true\}$
Output: $cl_{T^*}(M)$ if $PCLOSED = true$, or $cl_T(M)$ if $PCLOSED = false$

Initialization:

```
1  NEWDEP := M
2  for each A ⇒ B ∈ T:
3    if A = ∅:
4      NEWDEP := NEWDEP ∪ B
5    else:
6      COUNT[A ⇒ B] := |A|
7      CARD[A ⇒ B] := card(A)
8      for each ⟨y, a⟩ ∈ A:
9        add A ⇒ B to LIST[y]
10       DEGREE[y][A ⇒ B] := a
11       SKIP[y][A ⇒ B] := false
12 UPDATE := NEWDEP
13 CARDND := card(NEWDEP)
14 WAITLIST := ()
```

Computation:

```
15 while UPDATE ≠ ∅:
16   choose ⟨y, a⟩ ∈ UPDATE
17   UPDATE := UPDATE − {⟨y, a⟩}
18   for each A ⇒ B ∈ LIST[y] such that
            SKIP[y][A ⇒ B] = false and DEGREE[y][A ⇒ B] ≤ a:
19     SKIP[y][A ⇒ B] = true
20     COUNT[A ⇒ B] := COUNT[A ⇒ B] − 1
21     if COUNT[A ⇒ B] = 0 and
            (PCLOSED = false or CARD[A ⇒ B] < CARDND):
22       ADD := B ⊖ NEWDEP
23       CARDND := CARDND + Σ_{⟨y,a⟩∈ADD} f_L(a) − f_L(NEWDEP(y))
24       NEWDEP := NEWDEP ∪ ADD
25       UPDATE := UPDATE ∪ ADD
26       if PCLOSED = true and ADD ≠ ∅:
27         while WAITLIST ≠ ():
28           choose B ∈ WAITLIST
29           remove B from WAITLIST
30           ADD := B ⊖ NEWDEP
31           CARDND := CARDND + Σ_{⟨y,a⟩∈ADD} f_L(a) − f_L(NEWDEP(y))
32           NEWDEP := NEWDEP ∪ ADD
33           UPDATE := UPDATE ∪ ADD
34     if COUNT[A ⇒ B] = 0 and PCLOSED = true and
            CARD[A ⇒ B] = CARDND:
35       add B to WAITLIST
36 return NEWDEP
```

Fig. 1. Graded LinClosure

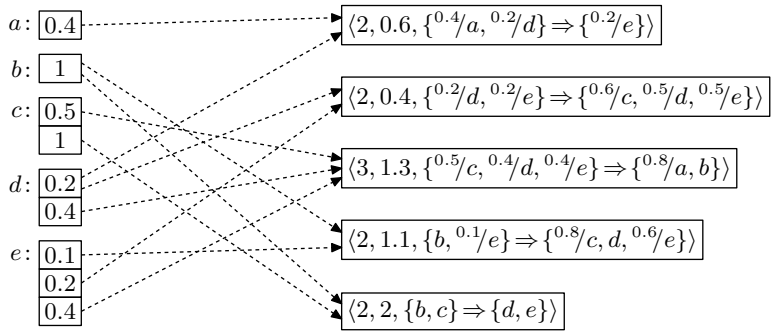

Fig. 2. T-structure encompassing *LIST*, *SKIP*, *DEGREE*, *COUNT*, and *CARD*

and does not depend on the length of the input). In the second part (computation), each graded attribute $\langle y, a\rangle$ is considered at most once for update. Thus, using analogous arguments as in case of the original LinClosure [17], we get that GLinClosure works with asymptotic time complexity $O(n)$. Needless to say, the time complexity of an implementation of GLinClosure depends of our choice of data structures, see Section 4 for further comments.

Remark 4. If $\mathbf{L}$ (our structure of truth degrees) is a two-element Boolean algebra, i.e. if $L = \{0, 1\}$, GLinClosure with *PCLOSED* set to *false* produces the same results as LinClosure [17] (the only difference is that our algorithm allows also for FAIs of the form $\{\} \Rightarrow B$ whereas the original LinClosure does not). From this point of view, GLinClosure is a generalization of LinClosure. GLinClosure is more versatile (even in crisp case): GLinClosure can be used to compute pseudo-intents (and thus a non-redundant basis of data tables with fuzzy attributes) which cannot be done with the original LinClosure (without additional modifications).

4 Implementation Details, Examples, and Remarks

As mentioned before, the efficiency of an implementation of GLinClosure is closely connected with data structures. The information contained in *LIST*, *SKIP*, *DEGREE*, *COUNT*, and *CARD* can be stored in a single efficient data structure. This structure, called a *T-structure*, is a particular attribute-indexed vector of lists of pointers to structures carrying values from *COUNT* and *CARD*. We illustrate the construction of a T-structure by an example. Consider a set T of FAIs which consists of the following fuzzy attribute implications:

φ_1: $\{\} \Rightarrow \{{}^{0.4}/a, {}^{0.1}/d\}$,
φ_2: $\{{}^{0.4}/a, {}^{0.2}/d\} \Rightarrow \{{}^{0.2}/e\}$,
φ_3: $\{{}^{0.2}/d, {}^{0.2}/e\} \Rightarrow \{{}^{0.6}/c, {}^{0.5}/d, {}^{0.5}/e\}$,
φ_4: $\{{}^{0.5}/c, {}^{0.4}/d, {}^{0.4}/e\} \Rightarrow \{{}^{0.8}/a, b\}$,
φ_5: $\{b, {}^{0.1}/e\} \Rightarrow \{{}^{0.8}/c, d, {}^{0.6}/e\}$,
φ_6: $\{b, c\} \Rightarrow \{d, e\}$.

Since φ_1 is of the form $\{\} \Rightarrow B$, its right-hand side is added to *NEWDEP* and the implication itself is not contained in *LIST* and other structures. The

other formulas, i.e. $\varphi_2, \ldots, \varphi_6$, are used to build a new T-structure which is depicted in Fig. 2. The T-structure can be seen as consisting of two main parts. First, a set of records encompassing information about the FAIs, $COUNT$, and $CARD$. For each FAI φ_i, we have a single record, called a T-record, of the form $\langle COUNT[\varphi_i], CARD[\varphi_i], \varphi_i \rangle$, see Fig. 2 (right). Hence, this part of the T-structure also carries information from $LIST$. Second, an attribute-indexed vector of lists containing truth degrees and pointers to T-records, see Fig. 2 (left). A list which is indexed by attribute $y \in Y$ will be called a y-list. The aim of this part of the structure is to keep information about the occurrence of graded attributes that appear in left-hand sides of FAIs from T. In more detail, a y-list contains truth degree $a \in L$ iff there is at lest one $A \Rightarrow B \in T$ such that $0 \neq A(y) = a$. Moreover, if a y-list contains a as its element, then it is connected via pointer to all T-records $\langle m, n, C \Rightarrow D \rangle$ such that $C(y) = a$. Because of the computational efficiency, each y-list is sorted by truth degrees in the ascendant manner. Note that pointers between elements of lists Fig. 2 (left) and T-records Fig. 2 (right) represent information in $SKIP$ ($SKIP[y][A \Rightarrow B] = \mathit{false}$ means that pointer from element $A(y)$ of y-list to T-record of $A \Rightarrow B$ is present). As one can see, a T-structure can be constructed by a sequential updating of the structure with time complexity $O(kn)$. In the following examples, we will use a convenient notation for writing T-structures which correspond in an obvious way with graphs of the from of Fig. 2. For example, instead of Fig. 2, we write:

a: $[(0.4, \langle 2, 0.6, \varphi_2 \rangle)]$
b: $[(1, \langle 2, 2, \varphi_6 \rangle, \langle 2, 1.1, \varphi_5 \rangle)]$
c: $[(0.5, \langle 3, 1.3, \varphi_4 \rangle), (1, \langle 2, 2, \varphi_6 \rangle)]$
d: $[(0.2, \langle 2, 0.4, \varphi_3 \rangle, \langle 2, 0.6, \varphi_2 \rangle), (0.4, \langle 3, 1.3, \varphi_4 \rangle)]$
e: $[(0.1, \langle 2, 1.1, \varphi_5 \rangle), (0.2, \langle 2, 0.4, \varphi_3 \rangle), (0.4, \langle 3, 1.3, \varphi_4 \rangle)]$

Example 1. Consider T which consists of $\varphi_1, \ldots, \varphi_6$ as above in this section. Let $M = \{{}^{0.2}\!/\!d\}$, and $PCLOSED = \mathit{false}$. After the initialization (line 14 of the algorithm), we have $NEWDEP = \{{}^{0.4}\!/\!a, {}^{0.2}\!/\!d\}$ and $UPDATE = (\langle a, 0.4 \rangle, \langle d, 0.2 \rangle)$. Recall that during the update, values of $COUNT$ and $SKIP$ are changed. Namely, values of $COUNT$ may be decremented and values of $SKIP$ are changed to *true*. The latter update is represented by removing pointers from the T-structure. After the update of $\langle a, 0.4 \rangle$ and $\langle d, 0.2 \rangle$, the T-record $\langle 0, 0.6, \varphi_2 = \{{}^{0.4}\!/\!a, {}^{0.2}\!/\!d\} \Rightarrow \{{}^{0.2}\!/\!e\} \rangle$ of φ_2 is processed because we have $COUNT[\varphi_2] = 0$ (see the first item of the T-record). At this point, the algorithm is in the following state:

b: $[(1, \langle 2, 2, \varphi_6 \rangle, \langle 2, 1.1, \varphi_5 \rangle)]$ $\quad ADD = (\langle e, 0.2 \rangle)$
c: $[(0.5, \langle 3, 1.3, \varphi_4 \rangle), (1, \langle 2, 2, \varphi_6 \rangle)]$ $\quad NEWDEP = \{{}^{0.4}\!/\!a, {}^{0.2}\!/\!d, {}^{0.2}\!/\!e\}$
d: $[(0.4, \langle 3, 1.3, \varphi_4 \rangle)]$ $\quad UPDATE = (\langle e, 0.2 \rangle)$
e: $[(0.1, \langle 2, 1.1, \varphi_5 \rangle), (0.2, \langle 1, 0.4, \varphi_3 \rangle), (0.4, \langle 3, 1.3, \varphi_4 \rangle)]$

As a further step of the computation, an update of $\langle e, 0.2 \rangle$ is performed and then the T-record $\langle 0, 0.4, \varphi_3 = \{{}^{0.2}\!/\!d, {}^{0.2}\!/\!e\} \Rightarrow \{{}^{0.6}\!/\!c, {}^{0.5}\!/\!d, {}^{0.5}\!/\!e\} \rangle$ of φ_3 is processed:

b: $[(1, \langle 2, 2, \varphi_6 \rangle, \langle 1, 1.1, \varphi_5 \rangle)]$ $\quad ADD = (\langle c, 0.6 \rangle, \langle d, 0.5 \rangle, \langle e, 0.5 \rangle)$
c: $[(0.5, \langle 3, 1.3, \varphi_4 \rangle), (1, \langle 2, 2, \varphi_6 \rangle)]$ $\quad NEWDEP = \{{}^{0.4}\!/\!a, {}^{0.6}\!/\!c, {}^{0.5}\!/\!d, {}^{0.5}\!/\!e\}$
d: $[(0.4, \langle 3, 1.3, \varphi_4 \rangle)]$ $\quad UPDATE = (\langle c, 0.6 \rangle, \langle d, 0.5 \rangle, \langle e, 0.5 \rangle)$
e: $[(0.4, \langle 3, 1.3, \varphi_4 \rangle)]$

Right after the update of $\langle c, 0.6\rangle$, $\langle d, 0.5\rangle$, and $\langle e, 0.5\rangle$, the algorithm will process the T-record of φ_4. After that, we have the following situation:

b: $[(1, \langle 2, 2, \varphi_6\rangle, \langle 1, 1.1, \varphi_5\rangle)]$
c: $[(1, \langle 2, 2, \varphi_6\rangle)]$

$ADD = (\langle a, 0.8\rangle, \langle b, 1\rangle)$
$NEWDEP = \{{}^{0.8}/a, b, {}^{0.6}/c, {}^{0.5}/d, {}^{0.5}/e\}$
$UPDATE = (\langle a, 0.8\rangle, \langle b, 1\rangle)$

Then, $\langle a, 0.8\rangle$ is updated. Notice that this update has no effect because the T-structure no longer contains attributes of the form $\langle a, x\rangle$ waiting for update (the a-list is empty). After the update of $\langle b, 1\rangle$, the T-record $\langle 0, 1.1, \varphi_5 = \{b, {}^{0.1}/e\} \Rightarrow \{{}^{0.8}/c, d, {}^{0.6}/e\}\rangle$ of φ_5 is processed. We arrive to:

c: $[(1, \langle 1, 2, \varphi_6\rangle)]$

$ADD = (\langle c, 0.8\rangle, \langle d, 1\rangle, \langle e, 0.6\rangle)$
$NEWDEP = \{{}^{0.8}/a, b, {}^{0.8}/c, d, {}^{0.6}/e\}$
$UPDATE = (\langle c, 0.8\rangle, \langle d, 1\rangle, \langle e, 0.6\rangle)$

The algorithm updates $\langle c, 0.8\rangle$, $\langle d, 1\rangle$, $\langle e, 0.6\rangle$ however such updates are all without any effect because the d-list and e-list are already empty, and the c-list contains a single record with $1 \not\leq 0.8$ (see the condition at line 18 of the algorithm). Thus, the T-structure remains unchanged, $UPDATE$ is empty, and the procedure stops returning the value of $NEWDEP$ which is $\{{}^{0.8}/a, b, {}^{0.8}/c, d, {}^{0.6}/e\}$.

Example 2. In this example we demonstrate the role of the $WAITLIST$. Let T be a set of FAIs which consists of

ψ_1: $\{{}^{0.2}/a\} \Rightarrow \{{}^{0.6}/a, {}^{0.3}/c\}$,
ψ_2: $\{{}^{0.3}/c\} \Rightarrow \{{}^{0.2}/b\}$,
ψ_3: $\{{}^{0.6}/a, {}^{0.3}/c\} \Rightarrow \{b\}$,
ψ_4: $\{{}^{0.6}/a, b, {}^{0.3}/c\} \Rightarrow \{d\}$.

Moreover, we consider $M = \{{}^{0.3}/a\}$ and $PCLOSED = true$. After the initialization (line 14), we have $NEWDEP = \{{}^{0.3}/a\}$, $CARDND = 0.3$ ($f_{\mathbf{L}}$ is identity), $UPDATE = (\langle a, 0.3\rangle)$, $WAITLIST = ()$, and the T-structure is the following:

a: $[(0.2, \langle 1, 0.2, \psi_1\rangle), (0.6, \langle 3, 1.9, \psi_4\rangle, \langle 2, 0.9, \psi_3\rangle)]$
b: $[(1, \langle 3, 1.9, \psi_4\rangle)]$
c: $[(0.3, \langle 3, 1.9, \psi_4\rangle, \langle 2, 0.9, \psi_3\rangle, \langle 1, 0.3, \psi_2\rangle)]$

The computation continues with the update of $\langle a, 0.3\rangle$. During that, the T-record $\langle 1, 0.2, \psi_1\rangle$ will be updated to $\langle 0, 0.2, \psi_1\rangle$. Since $CARD[\psi_1] = 0.2 < 0.3 = CARDND$, the left-hand side of ψ_1 is strictly contained in $NEWDEP$, and the algorithm processes $\langle 0, 0.2, \psi_1 = \{{}^{0.2}/a\} \Rightarrow \{{}^{0.6}/a, {}^{0.3}/c\}\rangle$, i.e. we get to

a: $[(0.6, \langle 3, 1.9, \psi_4\rangle, \langle 2, 0.9, \psi_3\rangle)]$
b: $[(1, \langle 3, 1.9, \psi_4\rangle)]$
c: $[(0.3, \langle 3, 1.9, \psi_4\rangle, \langle 2, 0.9, \psi_3\rangle, \langle 1, 0.3, \psi_2\rangle)]$

$ADD = (\langle a, 0.6\rangle, \langle c, 0.3\rangle)$
$NEWDEP = \{{}^{0.6}/a, {}^{0.3}/c\}$
$CARDND = 0.9$
$UPDATE = (\langle a, 0.6\rangle, \langle c, 0.3\rangle)$

After the update of $\langle a, 0.6\rangle$, we have:

b: $[(1, \langle 2, 1.9, \psi_4\rangle)]$
c: $[(0.3, \langle 2, 1.9, \psi_4\rangle, \langle 1, 0.9, \psi_3\rangle, \langle 1, 0.3, \psi_2\rangle)]$

Then, the algorithm continues with updating $\langle c, 0.3\rangle$. The T-record $\langle 2, 1.9, \psi_4\rangle$ is updated to $\langle 1, 1.9, \psi_4\rangle$ and removed from the c-list. In the next step, the T-record $\langle 1, 0.9, \psi_3\rangle$ is updated to $\langle 0, 0.9, \psi_3\rangle$. At this point, we have $CARD[\psi_3] = 0.9 = CARDND$, i.e. we add fuzzy set $\{b\}$ of attributes (the right-hand side of

ψ_3) to the *WAITLIST*. Finally, $\langle 1, 0.3, \psi_2 \rangle$ is updated to $\langle 0, 0.3, \psi_2 \rangle$ which yields the following situation: the T-structure consists of b: $[(1, \langle 1, 1.9, \psi_4 \rangle)]$, $ADD = (\langle b, 0.2 \rangle)$, $NEWDEP = \{^{0.6}/a, ^{0.2}/b, ^{0.3}/c\}$, $CARDND = 1.1$, and $UPDATE = (\langle b, 0.2 \rangle)$. Since *ADD* is nonempty, the algorithm continues with flushing the *WAITLIST* (lines 25–33). After that, the new values are set to $NEWDEP = \{^{0.6}/a, b, ^{0.3}/c\}$, $CARDND = 1.9$, and $UPDATE = (\langle b, 0.2 \rangle, \langle b, 1 \rangle)$. The process continues with updating $\langle b, 0.2 \rangle$ (no effect) and $\langle b, 1 \rangle$. Here again, we are in a situation where $CARD[\psi_4] = 1.9 = CARDND$, i.e. $\{d\}$ is added to the *WAITLIST*, only this time, the computation ends because *UPDATE* is empty, i.e. $\{d\}$ will not be added to *NEWDEP*. Thus, the resulting value being returned is $\{^{0.6}/a, b, ^{0.3}/c\}$.

5 Conclusions

We have shown an extended version of the LinClosure algorithm, so-called Graded LinClosure (GLinClosure). Our algorithm can be used in case of graded as well as binary attributes. Even for binary attributes, GLinClosure is more versatile than the original LinClosure (it can be used to compute systems of pseudo-intents) but it has the same asymptotic complexity $O(n)$. Future research will focus on further algorithms for formal concept analysis of data with fuzzy attributes.

References

1. Bělohlávek R.: *Fuzzy Relational Systems: Foundations and Principles.* Kluwer, Academic/Plenum Publishers, New York, 2002.
2. Bělohlávek R., Chlupová M., Vychodil V.: Implications from data with fuzzy attributes. AISTA 2004 in Cooperation with the IEEE Computer Society Proceedings, 2004, 5 pages, ISBN 2–9599776–8–8.
3. Bělohlávek R., Vychodil V.: Reducing the size of fuzzy concept lattices by hedges. In: FUZZ-IEEE 2005, The IEEE International Conference on Fuzzy Systems, May 22–25, 2005, Reno (Nevada, USA), pp. 663–668 (proceedings on CD), abstract in printed proceedings, p. 44, ISBN 0–7803–9158–6.
4. Bělohlávek R., Vychodil V.: Fuzzy attribute logic: attribute implications, their validity, entailment, and non-redundant basis. In: Liu Y., Chen G., Ying M. (Eds.): *Fuzzy Logic, Soft Computing & Computational Intelligence: Eleventh International Fuzzy Systems Association World Congress* (Vol. I), 2005, pp. 622–627. Tsinghua University Press and Springer, ISBN 7–302–11377–7.
5. Bělohlávek R., Vychodil V.: Attribute implications in a fuzzy setting. In: Missaoui R., Schmid J. (Eds.): *ICFCA 2006*, LNAI **3874**, pp. 45–60, 2006.
6. Bělohlávek R., Vychodil V.: Functional dependencies of data tables over domains with similarity relations. In: *Proc. IICAI 2005*, pp. 2486–2504, ISBN 0–9727412–1–6.
7. Bělohlávek R., Vychodil V.: Data tables with similarity relations: functional dependencies, complete rules and non-redundant bases. In: Lee M. L., Tan K. L., Wuwongse V. (Eds.): *DASFAA 2006*, LNCS **3882**, pp. 644–658, 2006.

8. Bělohlávek R., Vychodil V.: Properties of models of fuzzy attribute implications (to appear in *Proc. SCIS & ISIS 2006,* Sep. 20–24, 2006, Tokyo, Japan).
9. Ganter B.: *Begriffe und Implikationen*, manuscript, 1998.
10. Ganter B.: Algorithmen zur formalen Begriffsanalyse. In: Ganter B., Wille R., Wolff K. E. (Hrsg.): *Beiträge zur Begriffsanalyse.* B. I. Wissenschaftsverlag, Mannheim, 1987, 241–254.
11. Ganter B., Wille R.: *Formal Concept Analysis. Mathematical Foundations.* Springer, Berlin, 1999.
12. Goguen J. A.: The logic of inexact concepts. *Synthese* **18**(1968-9), 325–373.
13. Guigues J.-L., Duquenne V.: Familles minimales d'implications informatives resultant d'un tableau de données binaires. *Math. Sci. Humaines* **95**(1986), 5–18.
14. Hájek P.: *Metamathematics of Fuzzy Logic.* Kluwer, Dordrecht, 1998.
15. Hájek P.: On very true. *Fuzzy Sets and Systems* **124**(2001), 329–333.
16. Klir G. J., Yuan B.: *Fuzzy Sets and Fuzzy Logic. Theory and Applications.* Prentice Hall, 1995.
17. Maier D.: *The Theory of Relational Databases.* Computer Science Press, Rockville, 1983.
18. Pavelka J.: On fuzzy logic I, II, III. *Z. Math. Logik Grundlagen Math.* **25**(1979), 45–52, 119–134, 447–464.
19. Pollandt S.: *Fuzzy Begriffe.* Springer-Verlag, Berlin/Heidelberg, 1997.
20. Takeuti G., Titani S.: Globalization of intuitionistic set theory. *Annals of Pure and Applied Logic* **33**(1987), 195–211.

Yet another approach for completing missing values

L. Ben Othman and S. Ben Yahia

Faculty of Sciences of Tunis
Computer Science Department
Campus Universitaire, 1060 Tunis, Tunisia.
sadok.benyahia@fst.rnu.tn

Abstract. When tackling real-life datasets, it is common to face the existence of scrambled missing values within data. Considered as "dirty data", usually it is removed during a pre-processing step. Starting from the fact that "making up this missing data is better than throwing out it away", we present a new approach trying to complete missing data. The main singularity of the introduced approach is that it sheds light on a fruitful synergy between generic basis of association rules and the topic of missing values handling. In fact, beyond interesting compactness rate, such generic association rules make it possible to get a considerable reduction of conflicts during the completion step. A new metric called *"Robustness"* is also introduced, and aims to select the robust association rule for the completion of a missing value whenever a conflict appears. Carried out experiments on benchmark datasets confirm the soundness of our approach. Thus, it reduces conflict during the completion step while offering a high percentage of correct completion accuracy.
Keywords: Data mining, Formal Concept Analysis, generic association rule bases, missing values completion.

1 Introduction

In recent times, the field of Knowledge Discovery in Databases (KDD) has emerged as a new research discipline, lying at the crossroads of statistics, machine learning, data management, and other areas. The central step within the overall KDD process is Data mining — the application of computational techniques to the task of finding patterns and models in data. Implicitly, such knowledge is supposed to be mined from "high" quality data. However, most real-life datasets encompass missing data, that is commonly considered as withdrawable during the KDD pre-processing step.

Thus, setting up robust mining algorithms handling "dirty" data is a compelling and thriving issue to be addressed towards knowledge quality improvement. In this respect, a review of the dedicated literature pointed out a determined effort from the Statistics community. This is reflected by the wealthy harvest of works addressing the completing missing value issue, *e.g.*, Gibbs sampling [7, 14], the Expectation Maximization [9] and Bound and Collapse [17] —

to cite but a few. Based on the missing information principle [12], *i.e.*, *the value for replacement is one of the existing data*, the use of association rules seemed to be a promising issue [4, 10, 15, 21]. The driving idea is that association rules ideally describe conditional expectation of the missing values according to the observed data catched out by their premise parts. Within based association rule approaches, we shall mention those that present a robust itemset support counting procedure, *i.e.,* without throwing out missing data. They are based on pioneering works of [11, 16] and those that proceed by acquiring knowledge under incompleteness [18, 19]. The main difference between approaches presenting a completion process stands in the way of tackling the conflict problem, *i.e.,* when many values are candidates for the completion of a missing data. In addition, the inherent oversized lists of association rules that can be drawn is a key factor in hampering the efficiency of heuristics used to address the conflict problem.

In this paper, we propose a new approach, called $GBAR_{MVC}$, aiming to complete missing values based on generic basis of association rules. In fact, beyond interesting compactness rate, the use of such generic association rules proved to be fruitful towards efficiently tackling the conflict problem. In addition, a new metric called "*Robustness*" is introduced and aims to select the robust rule for the completion of a missing value whenever a conflict appears. Conducted experiments on benchmark datasets show a high percentage of correct completion accuracy.

The remainder of the paper is organized as follows. Section 2 sketches a thorough study of the related work to the completion of missing values using association rules. In Section 3, we introduce the $GBAR_{MVC}$ approach for completing missing values based on generic basis of association rules. Experimental results showing the soundness of our approach are presented in section 4. Finally, we conclude and outline avenues of future work.

2 Basic definitions and related work

In this section, we present the general framework for the derivation of association rules and the related work dealing with the completion of missing values using the association rule technique.

2.1 Association Rules

Complete - Incomplete context: A table $\mathcal{D}$ is a non-empty finite set of tuples (or transactions), where each tuple T is characterized by a non-empty finite set of attributes, denoted by $\mathcal{I}$. Each attribute X_i is associated to a domain, denoted $dom(X_i)$, which defines the set of possible values for X_i. It may happen that some attribute values for a tuple are missing. A context with missing values is called *incomplete context*, otherwise, it is said to be *complete*. In the sequel, we denote a missing value by "?".

Extraction context: An extraction context is a triplet $\mathcal{K} = (\mathcal{O}, \mathcal{I}, \mathcal{R})$, where $\mathcal{O}$ represents a finite set of transactions, $\mathcal{I}$ is a finite set of items and $\mathcal{R}$ is a binary (incidence) relation (*i.e.*, $\mathcal{R} \subseteq \mathcal{O} \times \mathcal{I}$). Each couple $(o, i) \in \mathcal{R}$ expresses that the transaction $o \in \mathcal{O}$ contains the item $i \in \mathcal{I}$.

Example 1. Let us consider the complete context depicted by Figure 1 (Left). This context is defined by 4 attributes X_1, X_2, X_3 and X_4, such that $dom(X_1) = \{A, B\}$, $dom(X_2) = \{C, D\}$, $dom(X_3) = \{E, F, G\}$ and $dom(X_4) = \{H, I\}$. The associated extraction context is depicted in Figure 1 (Center), where each couple (attribute, value) is mapped to an item. Figure 1 (Right) represents the extraction context in which missing values were randomly introduced. It is important to mention that each missing value indicates the presence of one item among the missing ones.

The formalization of the association rule extraction problem was introduced by Agrawal *et al.* [1]. Association rule derivation is achieved from a set $\mathcal{FI}_{\mathcal{K}}$ of frequent itemsets [2].

Frequent itemset: The support of an itemset I is the percentage of transactions containing I. The support of I, denoted $supp(I)$, is defined as $supp(I) = \frac{|\{o \in \mathcal{O} | I \subseteq o\}|}{|\mathcal{O}|}$. I is said to be frequent if $Supp(I)$ is greater than or equal to a user-specified minimum support, denoted *minsup*.

Association rule: An association rule R is a relation between itemsets of the form $R : X \Rightarrow (Y\text{-}X)$, in which X and Y are frequent itemsets, and $X \subset Y$. Itemsets X and $(Y\text{-}X)$ are called, respectively, *premise* and *conclusion* of the rule R. Valid association rules are those whose confidence measure, $\text{Conf}(R)=\frac{Supp(Y)}{Supp(X)}$, is greater than or equal to a minimal threshold of confidence denoted *minconf*. If $\text{Conf}(R)=1$, then R is called *exact association rule*, otherwise it is called *approximative association rule* [13]. Even though support and confidence metrics are commonly used to assess association rule validity, the lift metric [6] is becoming wider of use. In fact, this statistical metric, presenting a finer assessment of the correlation between the *premise* and the *conclusion* parts, is defined as follows: $\text{Lift(R)}=\frac{Supp(Y)}{Supp(X)\times Supp(Y-X)}$. Nevertheless, in practice, the number of valid association rules is very high. To palliate this problem, several solutions towards a lossless reduction were proposed. They mainly consist in extracting an informative reduced subset of association rules, commonly called *generic basis*.

	X_1	X_2	X_3	X_4
1	A	C	E	H
2	B	C	E	I
3	A	C	E	H
4	A	D	F	I
5	B	C	F	I
6	B	C	F	H
7	A	D	G	I
8	B	D	G	I

	A	B	C	D	E	F	G	H	I
1	×		×		×			×	
2		×	×		×				×
3	×		×		×			×	
4	×			×		×			×
5		×	×			×			×
6		×	×			×		×	
7	×			×			×		×
8		×		×			×		×

	A	B	C	D	E	F	G	H	I
1	×		×		×			×	
2		×	×		×				×
3	×		×		?	?	?	×	
4	×			×		×		?	?
5		×	×			×			×
6	?	?	×			×		×	
7	×			×			×		×
8		×	?	?			×		×

Fig. 1. Left: Extraction complete context $\mathcal{K}$. **Center**: The associated complete transactional mapping. **Right**: Extraction incomplete context.

2.2 Related work

The intuition behind the association rules based approches for completing missing values, is that association rules describe a dependency among data including missing ones. Hence, it should be possible to guess these values by exploiting discovered rules [21]. Interestingly enough, all these approaches can be split into two pools. With respect to Table 1, the first pool approaches begin by discarding missing data. Then, they try to complete missing ones where association rules discovered from only complete data, are of use. However, such approaches may lead to biased results, since such rules were discovered from a misleading data, which considerably affects the efficiency of the completion process [19]. Starting from the fact that *"making up missing data is better than throwing out it away"*, approaches of a second pool were proposed. Such approaches focus on mining knowledge under incompleteness. Unfortunately, these approaches suffer from the handling prohibitive number of rules generated from frequent itemsets. As a result, conflict between rules will be difficult to manage and leads to an inefficient completion process. To palliate such drawback, we propose a new approach based on the use of generic basis of association rules, that aims to complete missing values and reduce conflict during the completion step. In addition, our proposed approach falls within the second pool since it does not discard missing data.

	Pool 1		Pool 2	
	Approach 1 [4]	Approach 2 [10]	Approach 3 [15]	Approach 4 [21]
Knowledge Discovery under incompleteness	No	No	Yes	Yes
Conflict resolution	-	reducing conclusion part's rule	*Score-VM* [15] *J-Measure* [20]	*Score* [21]
Generation of rules based on	relevant maximal rectangles	frequent itemsets	frequent itemsets	frequent itemsets

Table 1. Characteristics of the surveyed approaches dealing with missing values completion.

3 The $GBAR_{MVC}$ approach

The limitations of the above surveyed approaches motivate us to propose a new approach mainly based on the use of generic basis of association rules. The main motivation is that such generic rules consists in a reduced subset of association rules, *i.e.,* fulfills the compactness property. As pointed out in [3], defining generic

association rules relies on the *Closure* operator and the key notion of *minimal generator* [13]. Thus, before introducing the completion approach, we shall show how these key notions are redefined in the case of incomplete context.

3.1 Basic definitions

Certain transaction : A transaction T is said to be *Certain*, with respect to an itemset X, denoted $\mathcal{C}ertain(X)$, if T contains X. The set of *Certain* transactions is defined as follows :

$$\mathcal{C}ertain(X) = \{T \in \mathcal{D} \,|\, \forall\, i \in X \; i \text{ is present in } T\}.$$

Probable transaction : A transaction T is said to be *Probable*, with respect to an item i, denoted $\mathcal{P}robable(i)$, if i is missing in T.
Probably transaction : A transaction T is said to be *Probably* with respect to an itemset (Xi) if T contains X, such that T is $Probable(i)$. The set of *Probably* transactions relatively to an itemset (Xi), denoted $\mathcal{P}robably(X, i)$ is as follows: $\mathcal{P}robably(X, i) = \{T \in \mathcal{D} \,|\, T \in Certain(X) \cap \mathcal{P}robable(i)\}$.

Example 2. With respect to the incomplete context depicted by Figure 1 (Right). Transaction T_3 is considered as $\mathcal{C}ertain(AC)$, since it contains AC and it is $\mathcal{P}robable(E)$ since E is missing. Transaction T_3 is then considered as $\mathcal{P}robably(AC, E)$.

In what follows, we recall the definition of the *Almost-Closure* operator [5].

Definition 1. ***(Almost-Closure)*** *The Almost-Closure operator of an itemset X, denoted $\mathcal{AC}(X)$, is defined as $\mathcal{AC}(X) = X \cup \{i \,|\, i \in \mathcal{I} \wedge supp(X) - supp(Xi) \leq \delta\}$ where $supp(X)$ is the absolute support defined as $supp(X) = |Certain(X)|$ and δ is a positive integer representing the number of exceptions.*

This Definition points out that when an item $i \in \mathcal{AC}(X)$, then that it is to say that this item is present in all transactions containing X with a bounded number of exceptions less than δ.

Example 3. Let us consider the complete context depicted by Figure 1 (Center). With respect to Definition 1, we have $\mathcal{AC}(AC) = ACEH$ with $\delta = 0$, *i.e.,* E and H exist in all transactions containing AC.

It is noteworthy that the *Almost-Closure* operator overlaps with that of *Closure* operator in a complete context for $\delta = 0$ [5]. The *Almost-Closure* was redefined to compute the δ-free sets[1] from an incomplete context [19]. We use this definition to introduce a *minimal generator* in an incomplete context. Then, we prove that with $\delta = 0$, the *Almost-Closure* does no longer correspond to the *Closure* operator like in a complete context. For this reason, in the remainder, we shall employ the *Pseudo-Closure* term to point out this distinction.

Definition 2. ***(Pseudo-Closure)*** *The Pseudo-Closure of an itemset X in an incomplete context, denoted $\mathcal{PC}(X)$, is defined as follows:*

[1] A 0-free-set is also called minimal generator [5].

$$\mathcal{PC}(X) = X \cup \{i \,|\, i \in \mathcal{I} \ \wedge \ supp(X) - supp(Xi) = |Probably(X, i)|\}.$$

The idea of the *Pseudo-Closure* operator is to adopt an optimistic strategy. This involves a consideration of transactions containing X in which i is missing ($Probably(X, i)$). These transactions are considered as transactions containing the item i.

Example 4. Let us consider the incomplete context depicted by Figure 1 (Right). We have $supp(AC) - supp(ACH) = 0$ which is equal to $|Probably(AC, H)|$. Moreover, we have $supp(AC) - supp(ACE) = 1$ which represents the number of the transactions $\mathcal{P}robably(AC, E)$. Hence, $\mathcal{PC}(AC) = ACEH$.

Definition 3. ***(Minimal generator in an incomplete context)*** *An itemset g is said to be minimal generator in an incomplete context if it is not included in the Pseudo-Closure of any of its subsets of size $|g| - 1$.*

Proposition 1. *The Pseudo-closure in an incomplete context is not a Closure operator.*

Proof. By fulfilling the extensivity property, the *Closure* operator induces that each *minimal generator* and its associated *Pseudo-closed* itemset have the same support value. However, the *Pseudo-closure* adopts an optimistic strategy as presented in [19]. When computing the *Pseudo-closure* of an itemset X, if an item is missing, then it is considered as present. Thus, the *minimal generator* and its *Pseudo-closed* itemset do not necessarily have the same support value. Consequently, the *Pseudo-closure* in an incomplete context is not a *Closure* operator. ■

In what follows, we adapt the definition of the generic basis of exact association rules introduced in [3] to an incomplete context. Such rules allow the selection of a generic subset of all association rules. Thus, the minimal set of rules is used for completing missing values, since it reduces conflict between rules during the completion step.

Definition 4. ***(Generic basis of pseudo-exact association rules)*** *Let $\mathcal{F}PC$ be the set of frequent Pseudo-closed itemsets extracted from an incomplete context. For each frequent Pseudo-closed itemset $c \in \mathcal{F}PC$, let $\mathcal{M}G_c$ be the set of its minimal generators. The generic basis of pseudo-exact association rules $\mathcal{G}B$ is defined as:*

$$\mathcal{G}B = \{R : g \Rightarrow (c - g) \mid c \in \mathcal{F}PC \text{ and } g \in \mathcal{M}G_c \text{ and } g \neq c^{(2)}\}.$$

For the completion of the missing values, we use generic rules of the form $premise \Rightarrow (X_i, v_i)$, where *premise* is a conjunction of elements of the form (X_j, v_j), $i \neq j$ where (X_j, v_j) is considered as an item.

[2] The condition $g \neq c$ ensures discarding rules of the form $g \Rightarrow \emptyset$.

3.2 The missing values completion $GBAR_{MVC}$

In the remainder, we present a missing value completion approach called $GBAR_{MVC}$[3]. This approach is based on the one hand, extracting the generic basis of pseudo-exact association rules from an incomplete context. On the other hand, we provide a new metric called *Robustness* that aims to select the robust rule for the completion of a missing value whenever a conflict appears. This new metric evaluates the degree of correlation between the premise and the conclusion of a rule materialized through the *Lift* measure [6] and it introduces the degree of assessment of the incomplete transaction. This assessment is materialized through the *Matching* measure. Below, we recall the notion of *consistently interpreting* a transaction by a rule [21] and we provide the definitions of the *Matching* and *Robustness* metrics.

Consistently interpreting [21]: A rule $R : premise \Rightarrow (X_i, v_i)$ is said to be *consistently interpreting* a transaction T presenting a missing value in the attribute X_i, if there is no element (X_j, v_j) in the premise of R that differs from the existing value of X_j in T.

Definition 5. *The* ***Matching*** *measure of a rule $R : premise \Rightarrow (X_i, v_i)$ with an incomplete transaction T is defined as follows :*

$$Matching(R,t) = \begin{cases} 0 & \text{if } R \text{ is not consistently interpreting } T \\ \frac{\sum matched(X_j,v_j)}{number\ of\ attributes} & \text{otherwise.} \end{cases}$$

where

$$matched(X_j, v_j) = \begin{cases} 0 \text{ if } X_i \text{ presents a missing value in } T \\ 1 \text{ otherwise.} \end{cases}$$

Example 5. Let us consider transaction T_6: $(X_1, ?)(X_2, C)(X_3, F)(X_4, H)$. Rule $R_1 : (X_2, D)(X_3, F) \Rightarrow (X_1, A)$ does not consistently interpret T_6, since the value of the attribute X_2 for T_6 is C, which is different from the value D related to attribute X_2 in the rule R_1. Thus, $Matching(R_1, T_6) = 0$. However, if we consider the example of $R_2 : (X_2, C)(X_3, F) \Rightarrow (X_1, B)$, we can affirm that $Matching(R_2, T_6) = \frac{1}{2}$ since (X_2, C) and (X_3, F) are present in T_6.

The main idea of our proposed approach is to select a rule that maximizes both the *Lift* and the *Matching* values. The *Lift* measure of a rule A $\Rightarrow$ B is interesting for the completion issue since it describes the strength of the correlation between A and B, *i.e.,* the presence of the item A indicates an increase of the item B. The purpose of the *Matching* measure is to select the rule that corresponds best to the incomplete transaction. For example, if the hair color of a person is missing and we are faced by a conflict between these two rules: Bleu eyes $\Rightarrow$ Blond hair and redheaded person $\wedge$ clear skin $\Rightarrow$ Red hair. Then, we tend to use the second rule since it presents a maximum matching. This is performed through the *Robustness* metric defined as follows:

[3] The acronym $GBAR_{MVC}$ stands for Generic Basis of Association Rules based approach for Missing Values Completion.

Definition 6. *The Robustness of an associative rule R for completing a missing transaction T is defined as follows:*

$$Robustness(R, T) = Matching(R, T) \times Lift(R).$$

In the remainder, we present the $GBAR_{MVC}$ algorithm, whose pseudo-code is given by Algorithm 1. The main steps of $GBAR_{MVC}$ algorithm are sketched by the following sequence :

- For each missing attribute X_i of an incomplete transaction T, select rules concluding on X_i and consistently interpreting T. We denote such rule set by $R_{probables}(T, X_i)$ (lines 3-7).
- If the set $R_{probables}(T, X_i)$ is empty, then there is no rules permitting the completion of X_i (lines 8-9).
- If all rules in $R_{probables}(T, X_i)$ conclude on the same value v, then v is used to complete the missing attribute value (lines 11-12).
- Otherwise, *i.e.,* $R_{probables}(T, X_i)$ leads to a conflict. Hence, we compute the *Robustness* value for all rules belonging to $R_{probables}(T, X_i)$ (lines 14-18).
- The rule presenting the highest *Robustness* value is used to complete the missing value on X_i (line 19).

4 Experimental results

It was worth the effort to experience in practice the potential benefits of the proposed approach. Thus, we have implemented both $GBAR_{MVC}$ and AR_{MVC} [21] approaches in the C++ language using gcc version 3.3.1. Experiments were conducted on a Pentium IV PC with a 2.4 GHz and 512 MB of main memory, running Red Hat Linux. The set of *minimal generators* and their associated *Pseudo-closed* itemsets were extracted thanks to *MVminer* kindly provided by F. Rioult. For these experiments, we consider a complete database to act as a reference database, and we randomly introduce missing values per attribute with the following different rates : 5%, 10%, 15% and 20%. Benchmark datasets used for this experiments are from the UCI Machine Learning Repository[4]. Characteristics of these datasets are depicted by Table 2. During these experiments, we compared statistics yielded by $GBAR_{MVC}$ vs those of AR_{MVC}, by stressing on the following metrics :

- The percentage of missing values that an approach permits to complete.
- The *accuracy* : the percentage of correctly completed missing values.

Table 3 sketches the variation of the completion percentage and the *Accuracy* metric vs the percentage of the missing values variation of $GBAR_{MVC}$. From the reported statistics, we remark that the variation of incrusted missing values does not really affect the percentage of the completion. However, the higher the percentage of the missing values is, the lower the obtained *accuracy*. This decrease in of the percentage of the correctly completed missing values seems to be legitimate and quite expectable. This result can be explained by the following:

[4] http://www.ics.uci.edu/∼mlearn/MLRepository.html.

```
1  Algorithm : GBAR_MVC
   Data: - K_MV : Incomplete context
         - GB : Generic basis of pseudo-exact association rules
   Results: K_VM completed
2  Begin
3    Foreach incomplete transaction T in K_MV do
4      Foreach attribute X_i in T with a missing value do
5        Foreach rule R in GB such that X_i appears in the conclusion
         do
6          If R consistently interpreting T then
7            R_probables(T, X_i) = R_probables(T, X_i) ∪ R;
8        If |R_probables(T, X_i)| = 0 then
9          V_completion = ∅;
10       Else
11         If R_probables(T, X_i) concludes on the same value v then
12           V_completion = v;
13         Else
14           max=0;
15           Foreach rule r in R_probables(T, X_i) do
16             r.Robustness=r.Matching× r.Lift;
17             If r.Robustness>max then
18               V_completion=r.conclusion;
19       T.X_i = V_completion;
20   return (K_VM completed);
21 End
```

Algorithm 1: $GBAR_{MVC}$ algorithm

the higher the incrusted number of missing values is, the worse the extracted rule quality. This fact considerably affects the *Accuracy* metric. Table 4 sketches the variation of the completion percentage and the *Accuracy* metric vs the variation of the *minsup* value. From Table 4, we can remak, as far as the *minsup* value increases, the percentage of the completion diminishes. On the contrary, in most cases by increasing the *minsup* value the accuracy value increases. In fact, rules with a higher *minsup* permit an accurate completion since they describe a more frequent expectation of the missing values according to the observed data. Table 5 sketches the statistics for the completion percentage and those of the *Accuracy* values obtained by $GBAR_{MVC}$ *vs* those pointed out by AR_{MVC} for a *minsup* value equal to 10%. For both approaches, as far as we lower the percentage of missing values, the number of rules considered during the completion step increases. However, those used by $GBAR_{MVC}$ is by far less than the rules used by AR_{MVC}. A careful scrutinize of these statistics permits to shed light on the following:

Dataset	Number of transactions	Number of items	Number of attributes
Mushroom	8124	128	23
Zoo	101	56	28
Tic-tac-toe	958	58	29
House-votes	435	36	18
Monks2	432	38	19

Table 2. Dataset characteristics.

- **Mushroom - House-Votes:** For these datasets, the *percentage of completion* of AR_{MVC} is better than $GBAR_{MVC}$. This result is not explained by the reduced number of rules presented by $GBAR_{MVC}$. This is can be justified by the *Score* metric used by AR_{MVC}. This metric allows the use of rules on which all items in the *premise* part are missing. Such rules are not used by $GBAR_{MVC}$. We considered them as non reliable for the completion.
- **Zoo - Tic-tac-toe - Monks2 :** In the contrary of the previous datasets, we remark that $GBAR_{MVC}$ has permitted a high *percentage of completion* as well as AR_{MVC}. This statement is observed even with the reduced number of rules of $GBAR_{MVC}$ in comparison with rules of AR_{MVC}. This fact represents the advantage of $GBAR_{MVC}$, *i.e.,* rules of $GBAR_{MVC}$ are not redundant.
- For all datasets, $GBAR_{MVC}$ has permitted a better *Accuracy*. This better *Accuracy* result can be justified as follows:
  1. Rules produced by $GBAR_{MVC}$ are more reliable in presence of missing values. This is materialized thorough the *Pseudo-Closure* definition.
  2. It was shown in [21] that the *Accuracy* depends on the number of the extracted rules. However, AR_{MVC} generates a large number of rules, which affects considerably the completion *Accuracy*.

Finally, according to these experimental results, it should be mentioned that $GBAR_{MVC}$ presents a more accurate completion process. Moreover, this completion process is less affected by the rate of the introduced missing values than AR_{MVC}. This efficiency can be explained by the strategy adopted during the completion step. In fact, based on generic bases of association rules, it permitted a considerable reduction of conflicts, leading to high rate of correct completion accuracy.

5 Conclusion and future work

In this paper, we proposed a new approach called $GBAR_{MVC}$, permitting the completion of the missing values. The main particularity of our proposed approach is that is based on the generic basis of association rules and a new metric called *Robustness*. Carried out experiments on benchmark datasets confirmed that $GBAR_{MVC}$ approach turns out to be very beneficial for resolving the challenge of completing missing values, specially at the pre-processing KDD step. In fact, $GBAR_{MVC}$ approach offers a high rate of correct completion accuracy

Dataset	Number of missing values(%)	Number of rules	Percentage of completion	Accuracy
Mushroom	05	028293	74%	99%
	10	027988	76%	99%
	15	027988	78%	97%
	20	024410	80%	97%
Zoo	05	824650	100%	97%
	10	756741	098%	89%
	15	626390	100%	88%
	20	547075	099%	88%
Tic-tac-toe	05	315094	100%	91%
	10	296222	100%	89%
	15	279915	100%	87%
	20	266022	100%	60%
House-votes	05	125909	91%	95%
	10	102310	93%	90%
	15	094246	92%	87%
	20	081162	92%	82%
Monks2	05	028325	100%	83%
	10	025402	100%	65%
	15	021790	100%	71%
	20	019741	100%	63%

Table 3. Variation of the *percentage of completion* and the *Accuracy* metric of $GBAR_{MVC}$ *vs* the percentage of missing values variation for *minsup* value equal to 10%.

Dataset	minsup	Percentage of completion	Accuracy
Mushroom	10%	80%	97%
	15%	75%	98%
	20%	71%	98%
	25%	55%	73%
	30%	53%	57%
Zoo	10%	99%	88%
	15%	96%	78%
	20%	92%	88%
	25%	89%	92%
	30%	88%	93%
Tic-tac-toe	10%	100%	060%
	15%	100%	070%
	20%	089%	085%
	25%	086%	096%
	30%	065%	100%
House-votes	10%	92%	82%
	15%	45%	90%
	20%	76%	79%
	25%	70%	74%
	30%	33%	50%
Monks2	10%	100%	63%
	15%	100%	63%
	20%	100%	75%
	25%	100%	83%
	30%	083%	92%

Table 4. Variation of the *percentage of completion* and the *Accuracy* metric of $GBAR_{MVC}$ *vs* the variation of the *minsup* value for a number of missing values equal to 20%.

		Mushroom			
	Number of missing values (%)	05	10	15	20
AR_{MVC}	Percentage of completion	50%	92%	99%	80%
	Accuracy	42%	58%	64%	66%
	Number of rules	79461	77161	76830	68168
$GBAR_{MVC}$	Percentage of completion	74%	76%	78%	80%
	Accuracy	99 %	99%	97%	97%
	Number of rules	28893	27988	27988	24410
		Zoo			
	Number of missing values(%)	05	10	15	20
AR_{MVC}	Percentage of completion	100%	100%	100%	100%
	Accuracy	55%	57%	55%	66%
	Number of rules	3898169	3842627	3761081	3293571
$GBAR_{MVC}$	Percentage of completion	100%	098%	100%	099%
	Accuracy	97%	89%	88%	88%
	Number of rules	0824650	0756741	0626390	0547075
		Tic-tac-toe			
	Number of missing values(%)	05	10	15	20
AR_{MVC}	Percentage of completion	100%	100%	100%	100%
	Accuracy	86%	73%	76%	71%
	Number of rules	632826	592115	554530	528343
$GBAR_{MVC}$	Percentage of completion	100%	100%	100%	100%
	Accuracy	91%	89%	87%	60%
	Number of rules	315094	296222	279915	266022
		House-votes			
	Number of missing values (%)	05	10	15	20
AR_{MVC}	Percentage of completion	95%	96%	97%	98%
	Accuracy	87%	77%	73%	71%
	Number of rules	387342	369180	335639	309617
$GBAR_{MVC}$	Percentage of completion	91%	93%	92%	92%
	Accuracy	95%	90%	87%	82%
	Number of rules	125909	102310	094246	081162
		Monks			
	Number of missing values (%)	05	10	15	20
AR_{MVC}	Percentage of completion	100%	100%	100%	100%
	Accuracy	76%	67%	65%	60%
	Number of rules	52660	45249	33815	34490
$GBAR_{MVC}$	Percentage of completion	100%	100%	100%	100%
	Accuracy	83%	65%	71%	63%
	Number of rules	28325	25402	21790	19741

Table 5. Evaluation of the *percentage of completion* and the *Accuracy* metric of AR_{MVC} *vs.* $GBAR_{MVC}$ for a *minsup* value equal to 10%.

and outperforms the approach proposed in [21]. The preliminary obtained results offer exciting additional alternatives avenues of future work. In fact, we are interested first, in tackling the "silence problem", *i.e.,* improving the percentage of completion. Second, it will be interesting to complete missing values by using the concept of disjunction-free-sets [8]. These sets allow the extraction of generalized rules with negative terms which could be interesting for the completion of missing values. Finally, our future work includes a further evaluation of the *Robustness* metric.

References

1. R. Agrawal, T. Imielinski, and A. Swami. Mining association rules between sets of items in large databases. In *Proceedings of the ACM-SIGMOD Intl. Conference on Management of Data, Washington D. C., USA*, pages 207–216, May 1993.
2. R. Agrawal and R. Srikant. Fast algorithms for mining association rules. In J. B. Bocca, M. Jarke, and C. Zaniolo, editors, *Proceedings of the 20th Intl. Conference on Very Large Databases, Santiago, Chile*, pages 478–499, 1994.
3. Y. Bastide, N. Pasquier, R. Taouil, L. Lakhal, and G. Stumme. Mining minimal non-redundant association rules using frequent closed itemsets. In *Proceedings of the International Conference DOOD'2000, LNAI, volume 1861, Springer-Verlag, London, UK*, July 2000.
4. S. BenYahia, K. Arour, and A. Jaoua. Completing missing values in databases using discovered association rules. *In Proceedings of the International Conference on Artificial and Computational Intelligence for Decision, Control and Automation in Engineering and Industrial Applications. Monastir, Tunisia*, pages 138–143, March 22-24 2000.
5. J.-F. Boulicaut, A. Bykowski, and C. Rigotti. Approximation of frequency queries by means of free-sets. *In Proceedings of the International Conference in Principles and Practice of Data Mining and Knowledge Discovery in Databases (PKDD'2000), Lyon, France*, pages 75–85.
6. S. Brin, R. Motwani, and C. Silverstein. Beyond market baskets: Generalizing association rules to correlations. In Joan Peckham, editor, *In Proceedings of the International Conference on Management of Data (ACM SIGMOD), May 13-15, 1997, Tucson, Arizona, USA*, pages 265–276. ACM Press, 1997.
7. W.L. Buntine. Operations for learning with graphical models. *Journal of Artificial Intelligence*, 2:159–225, 1994.
8. A. Bykowski and C. Rigotti. A condensed representation to find frequent patterns. *In Proceeding of the ACM SIGMOD-SIGACT-SIGART symposium of Principles of Database Systems, Santa Barbara, USA*, pages 267–273.
9. A.P. Dempster, N.M. Laird, and D.B. Rubin. Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society*, 39(1):1–38, 1977.
10. S. Jami, T. Jen, D. Laurent, G. Loizou, and O. Sy. Extraction de règles d'association pour la prédiction de valeurs manquantes. *ARIMA journal*, pages 103–124, November 2005.
11. M. Kryszkiewicz. Probabilistic approach to association rules in incomplete databases. *In: Proc. of Web-Age Information Management Conference (WAIM), Shanghai, China, 2000. Lecture Notes in Computer Science, Vol. 1846. Springer-Verlag (2000).*

12. R. J.A. Little and D.B. Rubin. *Statistical analysis with missing data*. Wiley, New York, 2002.
13. N. Pasquier, Y. Bastide, R. Touil, and L. Lakhal. Discovering frequent closed itemsets. In C. Beeri and P. Buneman, editors, *Proceedings of 7th International Conference on Database Theory (ICDT'99), LNCS, volume 1540, Springer-Verlag, Jerusalem, Israel*, pages 398–416, 1999.
14. P.Giudici and R. Castelo. Improving Markov Chain Monte Carlo model search for data mining. *Machine Learning*, 50(1–2):127–158, 2003.
15. A. Ragel. *Exploration des bases incomplètes : Application à l'aide au pré-traitement des valeurs manquantes*. PhD Thesis, Université de Caen, Basse Normandie, December 1999.
16. A. Ragel and B. Crémilleux. Treatment of missing values for association rules. *In Proceedings of the International Conference Pacific-Asia Conference on Knowlege Discovery and Data Mining (PAKDD'98), Melbourne, Australia, Lecture Notes in Computer Science, Springer-Verlag*, pages 258–270, April 15-17 1998.
17. M. Ramoni and P. Sebastiani. Bayesian inference with missing data using bound and collapse. *Journal of Computational and Graphical Statistics*, 9(4):779–800, 2000.
18. F. Rioult. *Knowledge discovery in databases containing missing values or a very large number of attributes*. PhD Thesis, Université de Caen, Basse Normandie, November 2005.
19. F. Rioult and B. Crémilleux. Condensed representations in presence of missing values. *In Proceedings of the International symposium on Intelligent Data Analysis, Berlin, Germany*, pages 578–588, 2003.
20. P. Smyth and R.M. Goodman. An information theoretic approach to rule induction from databases. IEEE Trans. *On Knowledge And Data Engineering*, 4:301–316, 1992.
21. C. Wu, C. Wun, and H. Chou. Using association rules for completing missing data. In *Proceedings of 4th International Conference on Hybrid Intelligent Systems, (HIS'04), Kitakyushu, Japan, IEEE Computer Society Press*, pages 236–241, 5-8 December 2004.