Cleaning Missing Data Based on the Bayesian Network

Liang Duan, Kun Yue*, Wenhua Qian, and Weiyi Liu

Department of Computer Science and Engineering,
School of Information Science and Engineering, Yunnan University, 650091, Kunming, China
kyue@ynu.edu.cn

Abstract. To guarantee the data quality, it is necessary to clean the missing data that prevalently exist in real world databases. By incorporating additional information, such as functional dependencies or integrity constraints, the correct value for each missing data item can be derived in many existing data cleaning methods. In this paper, we propose a method for cleaning the missing data item without additional information by adopting Bayesian network (BN) as the framework of the representation and inferences of probability distributions. First, we learn a Bayesian network from the complete part of the given incomplete database, called IBN. Then, we infer the probability distributions of each missing data item based on Gibbs sampling upon the IBN. Consequently, we obtain all possible values with their corresponding probability distributions (i.e., confidence degrees), by which we clean the incomplete databases. Experimental results showed the efficiency, accuracy and precision of our methods.

Keywords: Missing data cleaning, Bayesian network, Probabilistic database, Gibbs sampling, Probabilistic inference.

1 Introduction

Data quality is often affected by data anomalies, e.g., missing data, which prevalently exist in real world databases. It is necessary to carry out data cleaning to guarantee the data quality [1]. Actually, data cleaning is one of the critical mechanisms for companies to realize the full business value of big data in helping meet the quality, performance and scalability goals [2]. Many methods were used to clean missing data upon the additional information. For example, missing data items can be filled in by choosing the values satisfying the given functional dependencies [3]. A correct value for each missing data item can be derived based on the given conditional functional dependencies [4]. The correct values for missing data items could be found given aggregate constraints [5] by deleting the tuples that do not satisfy the constraints.

Actually, it is also necessary to fill in the missing value when the additional information is not available. But the correct value for each missing data item is difficult to be obtained in this situation [1, 6]. In this paper, we address the problem of data cleaning without additional information by providing a set of possible values for each missing data item rather than only one correct value. Particularly, we desire to infer

* Corresponding author.

Y. Gao et al. (Eds.): WAIM 2013 Workshops, LNCS 7901, pp. 348–359, 2013.

the probability distributions of all possible values for the missing data items and then fill in the missing values by these distributions.

In recent years, some methods have been proposed to predict the possible values for missing data based on the probabilistic model. For example, Mayfield [7] presented a framework to infer the missing values by capturing attributes dependencies with a relational dependency network. Stoyanovich [8] provided a framework to infer the probability distributions for missing data by learning a meta-rule semi-lattice (MRSL) model for each attribute from incomplete databases. These methods are efficient, but model templates are required for constructing the structure of the probabilistic model. Moreover, the MRSL model cannot represent the dependencies of all attributes from a global point of view and two inference methods are necessary: one for single attributes and another for multiple attributes separately.

Therefore, in order to clean the missing values for arbitrary attributes, a probabilistic model is necessary to represent the dependencies among all attributes. This means that to fulfill the cleaning of missing data, we will have to address the following two problems: (1) constructing a probabilistic model from the given incomplete databases; (2) providing an efficient inference mechanism to predict the probability distributions of all possible values for the missing data items for both single and multiple attributes.

It is well known Bayesian network (BN) is an effective framework of representing dependencies among random variables [9]. A BN is a directed acyclic graph (DAG) where nodes represent random variables and edges represent dependencies among random variables. Each variable in a BN is associated with a conditional probability table (CPT) to give the probability of each state given parent states. Comparing to the above-mentioned probabilistic model, the global, qualitative and quantitative dependencies among all attributes can be well represented by means of BNs. Furthermore, uncertainties can be inferred effectively by BN inference algorithms [9]. Thus, in this paper, we adopt BN as the underlying framework for representing dependencies among attributes. We learn the BN from the given incomplete database and derive the probability distributions for the missing data item by the BN inference algorithm.

To learn BNs from the incomplete database, called IBN, we extend the classical dependency-analysis based BN learning algorithm [10] by incorporating the inherence of databases with missing values. From the inherence of data cleaning, missing values will always take a very small proportion of the whole database. Thus, learned from the complete part of the given incomplete database, IBN can represent the dependencies or characteristics of the whole database basically, although the IBN does not include the items in the missing values. This makes the IBN be reasonably looked upon as the underlying model of probabilistic inferences for cleaning missing data. Comparing with the MRSL-based inferences, the probability distributions for single and multiple missing attributes can be derived universally by IBN inferences. Many algorithms for BN's exact inferences have been proposed [11], but these methods are of the exponential complexity, which are not efficient and suitable enough with respect to the BN-based inference especially over large scale BNs. Thus, based on Gibbs sampling [11], we proposed an approximate inference algorithm to obtain the probability distribution based on the IBN.

For each missing data item (i.e., incomplete tuple) t in databases, its probability distribution derived by the IBN's inference is a set of all possible combinations of values of the attributes missing in t. The sum of all the probabilities of the filled

values is 1, which means that the distributions can be calculated in a principled fashion. It is known that probabilistic databases have been proposed to manage a probability distribution on a set of possible worlds [12, 13]. Exactly, we store the probability distributions for the missing data items in a probabilistic database.

Generally speaking, our main contributions can be summarized as follows:

- We propose an efficient dependency analysis method to learn the IBN from incomplete databases, as the basis for cleaning missing values.
- We propose an approximate inference method to predict the probability distributions of possible values for the missing data items, and correspondingly give an algorithm to clean the missing data.
- We implement the proposed algorithms and make preliminary experiments to test the feasibility of our method.

The remainder of this paper is organized as follows: In Section 2, we learn a BN from incomplete databases. In Section 3, we infer the probability distributions of missing data and then clean the missing data. In Section 4, we show experimental results. In Section 5, we conclude and discuss future work.

2 Learning Bayesian Network from Incomplete Databases

Learning a BN from databases always has two steps: first constructing the structure of BN and then learning the CPTs [9], where the former is critical and challenging.

A BN is a DAG $G=(V, E)$, where V is the set of nodes and E is the set of edges. If two nodes in a BN are conditionally independent, there will be no edge between them, and information theory based measures can be used to detect conditional independencies of nodes [10]. The mutual information of two nodes X, Y is defined as

$$I(X,Y)=\sum_{i=1}^{I}\sum_{j=1}^{J}P(x_i,y_j)\log_2\frac{P(x_i,y_j)}{P(x_i)P(y_j)} \tag{1}$$

The conditional mutual information is defined as

$$I(X,Y\mid Z)=\sum_{i=1}^{I}\sum_{j=1}^{J}\sum_{k=1}^{K}P(x_i,y_j,z_k)\log_2\frac{P(x_i y_j\mid z_k)}{P(x_i\mid z_k)P(y_j\mid z_k)} \tag{2}$$

That $I(X, Y)$ is smaller than a certain threshold ε means that X and Y are marginally independent. Given condition Z, that $I(X, Y| Z)$ is smaller than ε means that X and Y are conditionally independent.

Cheng [10] proposed a dependency analysis method to construct the structure of BN from complete databases based on the information theory. In this classical algorithm, a node ordering is given to specify a causal or temporal order of the nodes of a BN. Considering the characteristics of the incomplete database, we modify the above distribution for node X (or X and Y), the tuples with missing values on X (or on X and Y) will not be taken into consideration. Following, we illustrate this by an example.

Example 1. Table 1 shows a part of incomplete table containing 4 non-key attributes, where "?" indicates a missing value. For convenience, we denote *tub*, *smo*, *can*, *xray* by *t*, *s*, *c* and *x* respectively. When computing $P(t, s)$, tuple $t5$ should be not taken into consideration, $P(t=absent, s=smoker)=0.5$, $P(t=absent, s=nonsmoker)=0.25$ and $P(t=present, s=nonsmoker)=0.25$. $t5$ should be taken into consideration when computing $P(t, x)$ since the values on t and x exist.

Table 1. A part of an incomplete database table

id	tub	smo	can	xray
t1	absent	nonsmoker	absent	normal
t2	absent	smoker	absent	normal
t3	present	nonsmoker	present	abnormal
t4	absent	smoker	absent	normal
t5	absent	?	?	normal

Algorithm 1 describes the steps of constructing an IBN.

Algorithm 1. IBN-Construction

Input: T, an incomplete table; O, a vector of node

Output: $G=(V,E)$, a DAG of the IBN

Variables: L, a list of pairs of nodes and each $l \in L$ is a pair (v_i, v_j), $v_i, v_j \in V$ and $i \neq j$; Z, a cut set

Steps:

$V \leftarrow$ Set-Node-Ordering(O), $E \leftarrow \{\}$, $L \leftarrow \{\}$
if $I(v_i, v_j) > \varepsilon$ then $L \leftarrow L \cup \{(v_i, v_j)\}$ //By Equation (1)
sort L by the decreasing order of $I(v_i, v_j)$
$l \leftarrow L[0]$, $E \leftarrow E \cup \{l\}$, $L \leftarrow L-\{l\}$
for each $l(v_i, v_j)$ in L do
  if no open path[1] between (v_i, v_j) then
    $E \leftarrow E \cup \{l\}$, $L \leftarrow L-\{l\}$
  end if
end for
for each $l(v_i, v_j)$ in L do
  $Z \leftarrow$ Find-Cut-Set(v_i, v_j)
  if $I(v_i, v_j | Z) > \varepsilon$ then $E \leftarrow E \cup \{l\}$ //By Equation (2)
end for
for each e in E do
  if there are paths besides e between v_i and v_j then
   $E \leftarrow E-\{e\}$, $Z \leftarrow$ Find-Cut-Set(v_i, v_j)
   if $I(v_i, v_j | Z) > \varepsilon$ then $E \leftarrow E-\{e\}$
end for
return G

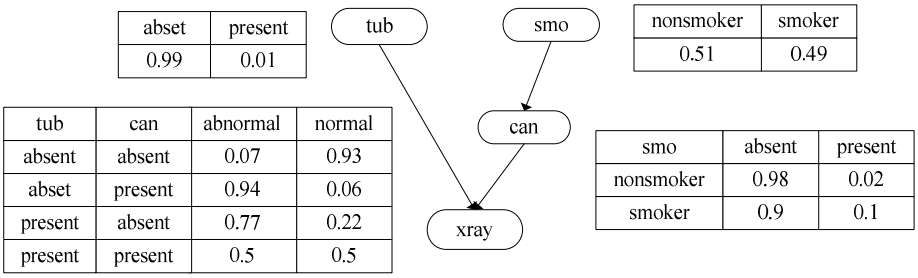
[1] A path that does not include head-to-head nodes is call open path.

For an incomplete database with n attributes, the conditional independency tests will be done for $O(n^2)$ times. Example 2 illustrates the execution of Algorithm 1.

Likelihood estimation [11] is commonly used to estimate the parameters of a statistical method by counting the frequency from tuples. We adopt this method to compute the CPT for each variable easily upon the IBN structure, where the probability is

$$P(x \mid y) = \frac{P(x, y)}{P(y)} \approx \frac{\text{the number of } (x, y)}{\text{the number of } y} \tag{3}$$

Example 2. For the incomplete table shown in Table 1, suppose the node ordering is {*tub*, *smo*, *can*, *xray*}. The mutual information of all pair of nodes obtained from databases are: $I(t, s)$=0.0, $I(t, c)$=0.0001, $I(t, x)$=0.033, $I(s, c)$=0.0278, $I(s, x)$=0.0139 and $I(c, x)$=0.192. Suppose ε is 0.01, so $I(c, x)>I(t, x)>I(s, c)>I(s, x)>\varepsilon$. So, L is {(c, x), (t, x), (s, c), (s, x)} and three edges (c, x), (t, x) and (s, c) can be added into E. Edge (s, x) will not be added since $I(s, x \mid c)= -0.0001$ is smaller than ε, where c is the cut set of s, x. Upon the obtained IBN structure, the CPTs can be computed based on Equation (3). Finally we can obtain the IBN as shown in Fig 1.



abset	present
0.99	0.01

nonsmoker	smoker
0.51	0.49

tub	can	abnormal	normal
absent	absent	0.07	0.93
abset	present	0.94	0.06
present	absent	0.77	0.22
present	present	0.5	0.5

smo	absent	present
nonsmoker	0.98	0.02
smoker	0.9	0.1

Fig. 1. An IBN learned from the incomplete table shown in Table 1

3 Cleaning Missing Data Based on Probabilistic Inferences

For each incomplete tuple t, we are to infer the probability distributions for the possible values of the attribute on which t's value is missing. All the possible values with probability distributions exactly constitute the representation of x-relation for uncertain data tuples, interpreted in [14]. An x-relation consists of one or more x-tuples independent of each other, each of which is a multiset of one or more mutually exclusive tuples, called alternatives. We then store the probability distribution of possible values for each missing data item as an x-tuple with a foreign key linked to the original incomplete table. Finally, we can fill in the missing value using the most probable one in corresponding x-tuples (i.e., the alternative with the largest probability).

It is known that Gibbs sampling is a Markov chain Monte Carlo algorithm and it always generates a Markov chain of samples. Thus, Gibbs sampling is particularly well-adapted to sampling the posterior distributions of a BN [11]. To infer the probability distributions of all possible values, we give Algorithm 2 as an approximate method for IBN inferences. First, we give the basic ideas as follows:

(1) We set a value to each attribute with missing value (i.e., nonevidence variable) and constitute the initial state including the attributes with complete values (i.e., evidence variables).

(2) We sample one of the attributes with missing values randomly and determine the value of the selected attribute from the conditional distribution under the current state. The new state can also be used for the next time of sampling. We repeat the sampling until the given threshold number of samples is reached.

(3) We get a set of samples, containing possible combination of values of the attributes with missing values, and those of the attributes with complete values. Then, the corresponding probability distributions can be achieved.

Algorithm 2. X-Tuple-Deriving

Input: X, missing attributes, $\{X_1, X_2, \ldots, X_n\}$; e, non missing attributes, $\{e_1, e_2, \ldots, e_m\}$; *ibn*, an IBN

Output: an x-tuple

Variables: $T[s]$, a vector of counts over s, initially zero; s, the current state of *ibn*, initially copied from e; x, a vector of values of X in s; $B[]$, a set of probability conditioned on the Markov blanket[2] of X, denoted as $MB(X)$; $s_{(-i)}$, the set $(X_1, X_2, \ldots, X_{i-1}, X_{i+1}, \ldots, X_w, e_1, e_2, \ldots, e_m)$; m, threshold of total number of samples to be generated

Steps:

```
x ← random values of X_i in X, s ← x∪e
for j←1 to m do
  if T[s] contains s then
    T[s] ← T[s]+1
  else
    insert s into T[s], T[s] ← 1
  end if
  select a query variable X_i from X randomly
  B[0] ← P(X=x_1|MB(X_i))   // X_i∈{x_1,x_2,…,x_k}
  for i←1 to k do
    B[i] ← B[i-1]+P(X=x_i|MB(X_i))
  end for
  generate a random value r∈[0,B[k]]
  X_i ← x_j where r <=B[j] //Determine the value of X_i
  s ← (s_(-i),X_i)
end for
T[s] ← T[s]/m
return T[s]
```

[2] A Markov blanket of X is the set of nodes composed of X's parents, its children and its children's other parents.

For an IBN with n nodes, the computations of probabilities conditioned on the Markov blanket are less than $O(nm)$ times. Following, we illustrate the execution of Algorithm 2 by an example.

Example 3. For the incomplete tuple $t5$ in Table 1, we are to obtain the probability distribution $P(s, c|t=absent, x=normal)$. We initialize the state s_0={s=*nonsmoker*, c=*absent*, t=*absent*, x=*normal*} and set 1 as the value of $N[s_0]$. Then, we randomly select s as a query variable and generate a value $r \in [0,1.0]$ randomly. Suppose r=0.67, and then we set s=*smoker* and generate a new state s_1={s=*smoker*, c=*absent*, t=*absent*, x=*normal*}. This procedure will be repeated for m times. Finally, we obtain the estimation (i.e., an x-tuple) of $t5$ shown in the right of Table 2.

Then, we can fill in the missing values by creating a new table including the x-tuple derived by Algorithm 2. For each incomplete tuple t in T, we call the function X-Tuple-Deriving in Algorithm 3 to generate an x-tuple t'. Then, we can select the alternative from T' with the largest probability and then update t in T.

Algorithm 3. Missing-Data-Cleaning

Input: T, an incomplete table; *ibn*, an IBN learned from the incomplete table T
Output: T', an x-relation corresponding to T
Variables: X, a set of attributes; e, a set of values of attributes; t', an x-tuple
Steps:

```
create table T'
for each incomplete tuple t∈T do
  //Attributes with missing values as query variables
  X ← Missing-Attributes(t)
  e ← Non-Missing-Attribute-Values(t)
  t'← X-Tuple-Deriving(X, e, ibn)
  insert t' into T'
  update t by the alternative with the largest probabil-
   ity in T' //t is the incomplete tuple in T
end for
return T'
```

Example 4. Revisiting the incomplete tuples in Table 1, we obtain the x-tuple for $t5$ by using Algorithm 2 and store it in Table 2, from which we select tuple $t5.1$ with the largest probability and update $t5$.

Table 2. x-tuple for $t5$ in Table 1

id	tub	smo	can	xray	prob
t5.1	absent	nonsmoker	absent	normal	0.564
t5.2	absent	smoker	absent	normal	0.430
t5.3	absent	smoker	present	normal	0.004
t5.4	absent	nonsmoker	present	normal	0.002

4 Experimental Results

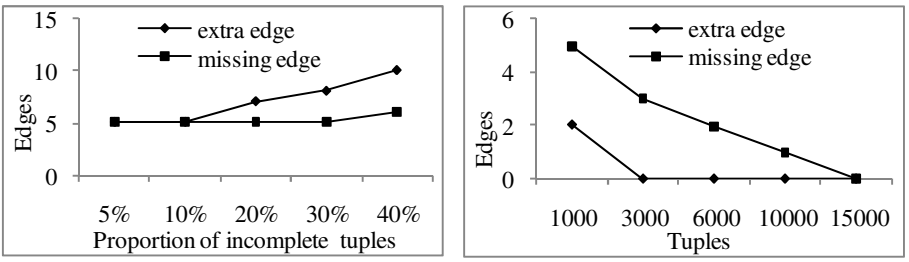
To verify the feasibility of the methods proposed in this paper, we implemented the presented algorithms. We mainly tested the accuracy and efficiency of IBN learning, and tested the convergence of the method of IBN inferences, and finally we tested the precision and efficiency of IBN-based missing data cleaning.

4.1 Experiment Setup

In the experiments, we adopted four classical BNs, Cancer Neapolitan (CN), Chest Clinic (CC), Car Diagnosis2 (CD) and Alarm (AL), widely used benchmarks. For each BN, we generated five original data sets of 1000, 3000, 6000, 10000 and 15000 tuples according to their probability distributions from Norsys [15]. We then generated five test data sets for each original one with 5%, 10%, 20%, 30% and 40% of incomplete tuples respectively, by setting one or more attributes to be NULL randomly. All the data sets were stored in MS SQL Server 2008 and all the codes were written in C#. The machine configurations are as follows: AMD Athlon64 X2 5000+ CPU, 2GB of main memory, running Windows 7 Ultimate 32-bit operating system.

4.2 Accuracy and Efficiency of IBN Learning

The edges which exist in the IBN structure but not exist in the true structure obtained from Norsys [15] are called extra edges. Edges which do not exist in the learned structure but exist in the true structure are called missing edges. We evaluated the accuracy of Algorithm 1 by recording the extra edges and missing edges. The extra edges and missing edges with the increase of incomplete tuples are shown in Fig. 2 (a). It can be seen clearly that the less the incomplete tuples, the more the learned IBN will be close to the true one. Meanwhile, there are few extra edges and missing edges in the IBN when the proportion of the incomplete tuples is about 5%. The extra edges and missing edges with the increase of tuples in the data set are shown in Fig. 2 (b). It is clear that the more the tuples in the given data set, the more accurate the learned IBN will be, which is consistent with the general conclusion for BN learning. Thus,



(a) Edges of Alarm BN with the increase of incomplete tuples

(b) Edges of Alarm BN with the increase of total tuples

Fig. 2. Accuracy of Algorithm 1 for constructing IBNs

Table 3. Accuracy of Algorithm 1 on various BNs

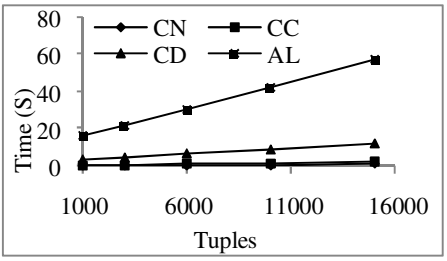
	extra edges	missing edges	correct edges
Caner Neapolitan	0	0	5
Chest Clinic	0	1	7
Car Diagnosis2	0	0	20
Alarm	0	3	43

we can conclude that Algorithm 1 is accurate for IBN learning. Further, for 15000 tuples with 10% incomplete tuples, we recorded the extra, missing and correct edges all the benchmark BNs, shown in Table 3.

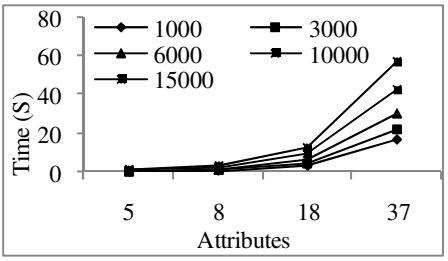
Then, Fig. 3 presents the execution time of IBN learning, including the time of DAG constructing and that of CPT computation. It can be seen that execution time is increased linearly with the increase of data tuples and nearly quadratically with the increase of attributes. This means that the execution time is not sensitive to the scale of the data set. In particular, we can also see that the cost of IBN learning is mainly dependent of the number attributes of the data set, as the node numbers of IBN, instead of that of tuples. Thus, our method for IBN leaning is efficient.

4.3 Convergence of the Inference Algorithm

It is pointed out that the posterior probabilities predicted by an approximate algorithm for BN's inferences are correct only if the sampling results are converged to a



(a) Execution time of IBN learning with the increase of tuples



(b) Execution time of IBN learning with the increase of attributes

Fig. 3. Execution time of learning IBNs with 10% incomplete tuples

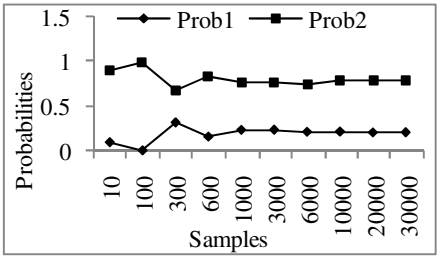


Fig. 4. Convergence of Algorithm 2

certain probabilistic value [11]. Thus, we tested the convergence of Algorithm 2 by recording the results upon the Cancer Neapolitan IBN under *Serum_Calcium= increased*, *Brain_Tumor=absent*, *Coma=present* and *Severe_Headaches=present*. Prob1 and Prob 2 in Fig. 4 is the probability of *Metastatic_Cancer = absent* and that of *Metastatic_Cancer=present* under the above evidences. It can be seen that Prob1 and Prob2 are stable around 0.2 and 0.8 respectively with the increase of the generated samples. The results show that the probabilities returned by Algorithm 2 converge to a certain value efficiently with just about 1000 samples, which guarantees the efficiency and correctness of Algorithm 2.

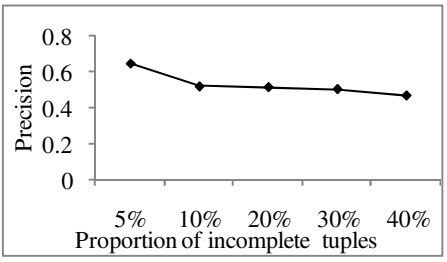
4.4 Precision and Efficiency of Data Cleaning

First, we compared the most possible value predicted by Algorithm 3 with the true value in the original data set. We used 1 (and 0) to denote the case that the predicted value is (not) the same with the true one. We defined the average precision by the mean of 0 or 1 for all incomplete tuples. Fig. 5 (a) shows the average precision of the most possible values obtained by Algorithm 3 on 1000 tuples, from which we can see that the precision will be decreased slowly with the increase of the proportion of incomplete tuples.

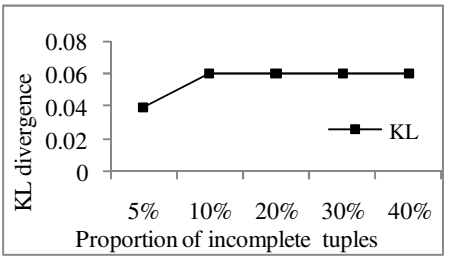
Meanwhile, we know that the measure of Kullback-Leibler (KL) divergence [16] is close to zero when the distributions are close to the results of Enumeration algorithm, as the exact BN inference algorithm [11]. The KL divergence of distribution Q from distribution P is computed by

$$D_{KL}(P \| Q) = \sum_i \ln(\frac{P(i)}{Q(i)})P(i) \tag{4}$$

Then, we recorded the KL divergence values for different proportions of incomplete tuples in the data set, shown in Fig. 5 (b). It can be seen that less the incomplete tuples, the smaller the KL divergence, i.e. the closer the two probability distributions, will be. This means that derived probability distributions are quite close to those obtained by the exact inference algorithm.



(a) Precision of Algorithm 3 with the increase of incomplete tuples

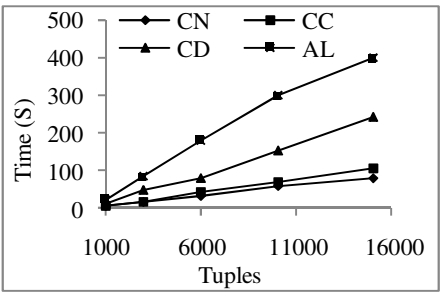


(b) KL divergence of Algorithm 3 with the increase of incomplete tuples

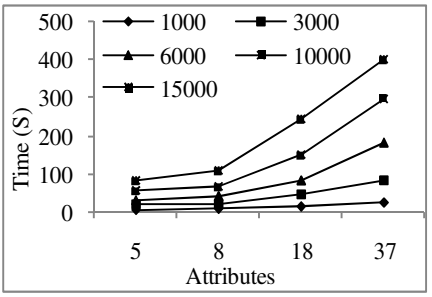
Fig. 5. Precision of Algorithm 3 under 1000 tuples

Table 4. Precision of Algorithm 3 on various numbers of tuples

tuples	CN		CC		CD		AL	
	precision	KL	precision	KL	precision	KL	precision	KL
1000	0.82	0.07	0.65	0.04	0.82	0.06	0.68	0.16
3000	0.74	0.11	0.55	0.05	0.75	0.06	0.66	0.21
6000	0.75	0.11	0.52	0.06	0.71	0.12	0.80	0.15
10000	0.76	0.10	0.49	0.06	0.72	0.10	0.80	0.14
15000	0.72	0.11	0.52	0.07	0.69	0.14	0.83	0.14



(a) Execution time as a function of the number of tuples.



(b) Execution time as a function of the number of attributes.

Fig. 6. Execution time of data cleaning with 5% incomplete tuples

Further, we recorded the results of cleaning on various numbers of tuples with 5% incomplete parts, shown as Table 4. Thus, the accuracy of Algorithm 3 for missing data cleaning is mainly determined by the IBN with a certain portion of incomplete tuples. From the perspective of real applications of data cleaning on an original data table with a small proportion of incomplete tuples, our method can work effectively.

Following, we recorded the execution time of Algorithm 3 shown in Fig. 6 for four BNs, each of which corresponds to five test data sets with 5% incomplete tuples. It can be seen that the execution time is increased linearly with the increase of tuples and attributes of the data sets, which verifies the efficiency of Algorithm 3.

5 Conclusion and Future Work

In this paper, we proposed the BN-based method for cleaning missing data. Focusing on the associations between attributes, we gave the methods for BN learning and inferences taking the given incomplete database as input. Theoretical and experimental analysis results verify the feasibility of our method.

To test our method further, we will make experiments on arbitrarily distributed data sets. As well, more efficient method for model learning and inferences will be considered by incorporating some optimization strategies. To extending our method to the realistic big data sets by incorporating the techniques of uncertain databases. Based on the methods proposed in this paper, we can explore the BN-based cleaning for redundant or wrong values. These are exactly our future work.

Acknowledgement. This paper was supported by the National Natural Science Foundation of China (61063009, 61163003, 61232002), the Ph. D Programs Foundation of Ministry of Education of China (20105301120001), the Yunnan Provincial Foundation for Leaders of Disciplines in Science and Technology (2012HB004), the Natural Science Foundation of Yunnan Province (2011FZ013), and the Foundation for Key Program of Department of Education of Yunnan Province (2011Z015).

References

1. Muller, H., Freytag, J.C.: Problems, Methods, and Challenges in Comprehensive Data Cleansing. Technical report, Humboldt-Universitat zu Berlin (2003)
2. Arasu, A., Chaudhuri, S., Chen, Z., Ganjam, K., et al.: Experiences with using Data Cleaning Technology for Bing Services. IEEE Data Engineering Bulletin, 14–23 (2012)
3. Beskales, G., Ilyas, I.F., Golab, L.: Sampling the repairs of functional dependency violations under hard constraints. PVLDB 3(1), 197–207 (2010)
4. Bohannon, P., Fan, W., Geerts, F., Jia, X., Kementsietsidis, A.: Conditional Functional Dependencies for Data Cleaning. In: Chirkova, R., Dogac, A., Ozsu, M.T., Sellis, T.K. (eds.) Proc. of ICDE 2007, Istanbul, Turkey, pp. 746–755. IEEE Computer Society (2007)
5. Chen, H., Ku, W.S., Wang, H.: Cleansing Uncertain Databases Leveraging Aggregate Constraints. In: Workshops Proc. of ICDE 2010, California, USA, pp. 128–135. IEEE Computer Society (2010)
6. Srivastava, D.: Analyzing Data Quality Using Data Auditor. In: Chen, L., Tang, C., Yang, J., Gao, Y. (eds.) WAIM 2010. LNCS, vol. 6184, pp. 1–1. Springer, Heidelberg (2010)
7. Mayfield, C., Neville, J., Prabhakar, S.: ERACER: A Database Approach for Statistical Inference and Data Cleaning. In: Elmagarmid, A.K., Agrawal, D. (eds.) Proc. of SIGMOD 2010, Indiana, USA, pp. 75–86. ACM (2010)
8. Stoyanovich, J., Davidson, S., Milo, T., Tannen, V.: Deriving Probabilistic Databases with Inference Ensembles. In: Abiteboul, S., Bohm, K., Koch, C., Tan, K.L. (eds.) Proc. of ICDE 2011, Hannover, Germany, pp. 303–314. IEEE Computer Society (2011)
9. Darwiche, A.: Modeling and Reasoning with Bayesian Networks. Cambridge University Press (2009)
10. Cheng, J., Greiner, R., Bell, D., Liu, W.: Learning Bayesian Networks from Data: An Efficient Approach Based on Information Theory. Artificial Intelligence 137(1-2), 43–90 (2002)
11. Russell, S., Norvig, P.: Artificial Intelligence: A Modern Approach, 3rd edn. Prentice Hall (2009)
12. Cavallo, R., Pittarelli, M.: The Theory of Probabilistic Databases. In: Stocker, P.M., Kent, W., Hammersley, P. (eds.) Proc. of VLDB 1987, Brighton, England, pp. 71–81. Morgan Kaufmann (1987)
13. Huang, J., Antova, L., Koch, C., Olteanu, D.: MayBMS: A Probabilistic Databases Management System. In: Cetintemel, U., Zdonik, S.B., Kossmann, D., Tatbul, N. (eds.) Proc. of SIGMOD 2009, Rhode Island, USA, pp. 1071–1074. ACM (2009)
14. Benjelloun, O., Sarma, A., Halevy, A., Widom, J.: ULDBs: Databases with Uncertainty and Lineage. In: Dayal, U., Whang, K.Y., Lomet, D.B., Alonso, G.A., Lohman, G.M., Kersten, M.L., Cha, S.K., Kim, Y.K. (eds.) Proc. of VLDB 2006, Seoul, Korea, pp. 953–964. Morgan Kaufmann (2006)
15. Norsys Software Corporation, `http://www.norsys.com/`
16. Cover, T., Thomas, J.: Elements of Information Theory. Wiley and Sons (2006)