Learning Missing Values from Summary Constraints

Xintao Wu
UNC at Charlotte
CS Dept.
Charlotte, NC 28223
xwu@uncc.edu

Daniel Barbará
George Mason University
ISE Dept. MSN 4A4
Fairfax, VA 22030
dbarbara@gmu.edu

ABSTRACT

Real-world data sets often contain errors and inconsistency. Even though this is a very important problem it has received relatively little attention in the research community. In this paper we examine the problem of learning missing values when some summary information is available. We use linear algebra and constraint programming techniques to learn the missing values using apriori-known summary information and that derived from the raw data. We reconstruct the missing values by different methods in three scenarios: ideal-constrained, under-constrained, and over-constrained. Furthermore, for a range query involving missing values, we also give the lower bound and upper bound for the values using constraint programming techniques. We believe that theory of linear algebra and constraint programming constitutes a sound basis for learning missing values when summary information is available.

Keywords

Data cleaning, constraint programming, linear algebra, maximum entropy, cross entropy

1. INTRODUCTION

The presence of missing or incomplete data is commonplace in large real-world databases. Missing values occur for a variety of reasons, e.g., omissions in the data entry process, confusing questions in the data gathering process, sensor malfunction, and so on. Learning missing data is critical for data warehousing, yet little has been done in the research field.

Consider a scenario of data preprocessing in data warehousing. In a datacube [16] (A datacube is a widely used data model for On-Line Analytical Processing (OLAP); a datacube is a multidimensional data abstraction, where aggregated *measures* of the combinations of *dimension* values are kept.) which summarizes sales data for a corporation, with dimensions “time of sale”, “location of sale” and “product of sale”, some sale values at combinations of some products, stores, and periods may be missing. Some of them may be caused by the true absence of sales during a particular period and location, but others may be caused by errors. Hence decisions based on the incomplete data may not be accurate.

In this paper, we assume:

- The missing value occurs only in non-negative *measures* [1].
- All the *dimension* attributes have no missing values.
- The positions of missing cells, i.e., the combination of dimension values, are known apriori.

These assumptions are easy to justify since stores usually know their products lists and send this information to the data warehouse. Figure 1 shows the cases which satisfy or violate the assumptions. In the warehouse the positions of missing cells can be easily dealt with by using bitmaps that indicate their locations.

Apart from the above general assumption, it is often true the participating sources may also have some summary information, S_A, at high levels, apart from the raw records, D, at the finest level (we assume the known raw records are accurate while the summary information may be inaccurate or inconsistent). The summary information can be thought as constraints of underlying raw data. For example, the simplest summation form requires the sum of subset of underlying raw data (which may contain missing cells) be equal to some value. Most “rolled-up” data has this property (i.e., the weekly or monthly sale totals for some particular products). For brevity, we choose to focus on summation constraints for the bulk of this paper. We discuss other constraints in Section 3.

Based on the characteristics of constraint information, we classify the problem into the following sub-scenarios and study some specific techniques from linear algebra, entropy theory, and constraint programming for missing value reconstruction. Figure 2 shows one example for each sub-scenario.

- **Ideal-constrained**: the aggregation values are accurate and sufficient to infer all the missing values. In this case, there is no conflict between the raw data and the known aggregation values. We can get one exact solution for each missing cell by solving the linear constraint equations.
- **Under-constrained**: the aggregation values are accurate but not sufficient to infer all the exact missing

[1] Typically, the measure in data cubes is non-negative and normally is a semicontinuous attribute which has a proportion of values equal to a single value (typically zero), and a positive continuous distribution among the remaining values. For example, the income or expenditures of economic surveys or sale values in superstores is a semicontinuous attribute instead as continuous attribute since these measures can often be zero for some combinations of dimension values.

Table 1: Tuple 3 and 4 violate the assumption for missing occurs in dimensions, ? denotes missing

TUPLE	DIMENSION			MEASURE	Violate Assumption
	STORE	PRODUCT	DATE	SALE	
1	s_1	p_1	d_1	20	N
2	s_1	p_1	d_1	?	N
3	s_2	?	d_2	10	Y
4	?	p_2	d_2	?	Y

values. In this case, there may be more than one solutions to satisfy the constraints. We may pick the optimal one as a representative from the solution set.

- **Over-constrained**: the aggregation values in S_A are not accurate. In this case, no assignment of values to variables satisfies all constraints . The reason for this is that some inconsistent summary data is estimated, rather than known. In this situation, the goal is to find the best compromise.

In both the under-constrained case and the over-constrained case, our goal is to find the best estimates. One simple approach is to reconstruct the missing values by applying linear algebra techniques to solve the linear constraint equations. We may get one representative solution in the under-constrained case or the best compromise solution in the over-constrained case from the overall view of the underlying data. However, the estimates computed from the simple linear algebra method may be inaccurate. For example, in under-constrained case, the estimates may incur big errors due to the freedom of variables when the number of constraints is far less than the number of missing cells. In this paper we investigate entropy techniques to impute the missing values subject to the constraints. In over-constrained case, the estimates may also be inaccurate due to the inconsistence of constraints. One idea is to choose a maximal subset of satisfiable constraints . However, this is known to be an NP-complete problem [5] and a fast algorithm is unlikely to exist.

If our goal is to construct and keep only one estimated copy, the data warehouse can replace the missing cells by the estimates obtained by the above solution. The imputed copy can then be used for analysis or querying. As the reconstructed copy contains the estimation of missing cells, the analysis result or query answer will still be an approximation. This will not be satisfactory if the users expect more accurate results or at least results bounded by a range. In this paper, we also investigate constraint programming techniques to give query answers that come in the form of guaranteed ranges. These answers are obtained by materializing the missing cells on the fly when the query contains missing cells.

The rest of the paper is organized as follows. Section 2 describes the application of linear algebra, entropy, and constraint programming techniques to recover missing values in data warehousing. Section 3 presents extensions to other constraint forms. Section 4 presents the experimental results over a real data set. Section 5 presents the related work. Finally Section 6 presents conclusions and future work.

2. TECHNIQUE

Table 2: Sub-scenarios of missing values and aggregation constraints where x_1, x_2, x_3 are missing cells

sub-scenario	constraints	solutions
ideal-constrained	$x_1 + x_2 = 5$ $x_1 + x_3 = 3$ $x_1 + x_2 + x_3 = 7$	unique solution
under-constrained	$x_1 + x_2 = 5$ $x_1 + x_3 = 3$	many solutions
over-constrained	$x_1 + x_2 = 5$ $x_1 + x_3 = 3$ $x_1 + x_2 + x_3 = 7$ $x_2 + x_3 = 5$	no solution

We use $\mathbf{A}$, $\mathbf{A_k}$, $\mathbf{B}$, $\mathbf{U}$, $\mathbf{V}$, $\mathbf{\Lambda}$ to denote a matrix, T to denote the transpose, $A(i, j)$ to denote the value at (i, j) position of matrix, $\mathbf{x}$, $\mathbf{b}$, $\mathbf{c}$ to denote a vector, x_i to denote the i-th value in the vector $\mathbf{x}$.

2.1 Reconstruction of Missing Values by Linear Algebra

Given raw data D and an aggregation set S_A, assume there are n missing cells and l aggregation records in S_A. The procedure of mapping to linear algebra is straightforward as shown below.

- Identifying the aggregation records which contains the missing cells.
- Computing the aggregation value $\hat{s}_i$ (s_i is the according aggregation value) from the known records in D for each known aggregation record in S_A which contains the missing cells.
- Computing $b_i = s_i - \hat{s}_i$ where b_i contains only missing cells.
- Setting a zero-one matrix $\mathbf{A}$ by scanning the missing value list, such that $A(i, j) = 1$ if and only if the missing value x_i lies in j-th aggregation record.

We get the linear equation as shown in Equation 1, where $\mathbf{A}$ is $m \times n$ matrix of coefficients with $rank(\mathbf{A}) = r$, $\mathbf{x}$ is an unknown $n \times 1$ column vector, and $\mathbf{b}$ is a known $m \times 1$ column vector.

$$\mathbf{A} \cdot \mathbf{x} = \mathbf{b} \qquad (1)$$

Now the task to compute the missing values by using raw data and summary constraints is reduced to solving the vector $\mathbf{x}$ in linear algebra equation 1. The methods for linear algebra include Gauss-Jordan elimination, LU decomposition, and Singular Value Decomposition (SVD). We refer the reader to a linear algebra text such as [13; 29] for further detail.

In this paper, we will apply SVD to solve the linear equation. The reason is twofold. First, the rows in matrix $\mathbf{A}$ may be correlated and one row can be described by the combination of some other rows[2]. Second, the matrix $\mathbf{A}$ is typically large (due to the size of missing values n and the size of constraints m) and sparse (due to only a few missing cells occurring in one given constraint). In general, SVD needs $O(mn)$ space and $O(nm^2)$ (or $O(mn^2)$ dependent on which one is smaller) computation. This may be prohibitive, especially when both n and m are large. In our system, we apply Fast large-sparse-matrix Singular Value Decomposition (SVD) algorithms which have been developed with significant less complexity in [2]. To make the explanation precise, we present SVD Theorem as follows (See [13] for proof).

THEORM 1. *Every $m \times n$ matrix $\mathbf{A}$ can be written as*

$$\mathbf{A} = \mathbf{U}\mathbf{\Lambda}\mathbf{V}^T,$$

where $\mathbf{U}^T\mathbf{U} = \mathbf{V}^T\mathbf{V} = \mathbf{I}_n$ *and* $\mathbf{\Lambda} = diag(\lambda_1, \cdots, \lambda_n)$, $\lambda_i > 0$ *for* $i = 1, \cdots, r$, $\lambda_i = 0$ *for* $i = r+1, \cdots, n$

Some properties of SVD show as following:

- Rank property: $rank(\mathbf{A}) = r$, the *range* space $R(\mathbf{A}) \equiv span\{\mathbf{u}_1, \cdots, \mathbf{u}_r\}$, and the *null* space $N(\mathbf{A}) \equiv span\{\mathbf{v}_{r+1}, \cdots, \mathbf{v}_n\}$, where $\mathbf{U} = [\mathbf{u}_1, \cdots, \mathbf{u}_m]$ and $\mathbf{V} = [\mathbf{v}_1, \cdots, \mathbf{v}_n]$.
- Dyadic decomposition: $\mathbf{A} = \sum_{i=1}^{r} \lambda_i \mathbf{u}_i \mathbf{v}_i^T$.
- Eckart and Young: Assume $\mathbf{A}_k = \sum_{i=1}^{k} \mathbf{u}_i \cdot \lambda_i \cdot \mathbf{v}_i^T$ with $k < r$, then $min_{r(\mathbf{B})=k}||\mathbf{A} - \mathbf{B}||_2 = ||\mathbf{A} - \mathbf{A}_k||_2 = \lambda_{k+1}$.

When we get the SVD form of matrix $\mathbf{A}$, the solution of linear equation 1 is shown as,

$$\mathbf{x} = \mathbf{V} \cdot [diag(1/\lambda_i)] \cdot (\mathbf{U}^T \cdot \mathbf{b}) \tag{2}$$

1. If all $\lambda_i > 0$ for $i = 1, \cdots, n$, in other words, the number of independent constraints equals the number of missing cells, we get one unique solution by Equation 2.

   EXAMPLE 1. *Consider the ideal-constrained example in Table 2,* $\mathbf{A} = \begin{pmatrix} 1 & 1 & 0 \\ 1 & 0 & 1 \\ 1 & 1 & 1 \end{pmatrix}$, $\mathbf{b} = \begin{pmatrix} 5 \\ 3 \\ 7 \end{pmatrix}$, *the unique solution by SVD is* $\mathbf{x} = \begin{pmatrix} 1 \\ 4 \\ 2 \end{pmatrix}$.

2. If the vector $\mathbf{b}$ lies in the range of $\mathbf{A}$, in other words, there are effectively fewer constraints than missing values (under-constrained), then the singular set of equations has more than one solution. We might want to pick the one with the smallest length $||\mathbf{x}||^2$ if we want to single out one particular member of this solution-set of vectors as a representative. Equation 2 gives the solution vector of smallest length by simply replacing $1/\lambda_i$ by zero for those $\lambda_i = 0$.

   EXAMPLE 2. *Consider the under-constrained example in Table 2,* $\mathbf{A} = \begin{pmatrix} 1 & 1 & 0 \\ 1 & 0 & 1 \end{pmatrix}$, $\mathbf{b} = \begin{pmatrix} 5 \\ 3 \end{pmatrix}$, *the representative solution by SVD is* $\mathbf{x} = \begin{pmatrix} 3 \\ 2 \\ 0 \end{pmatrix}$.

3. If $\mathbf{b}$ is not in the range of the singular matrix $\mathbf{A}$, in other words, there are more independent constraints than missing values (over-constrained), then the set of equations has no solution. The reason is not all the known constraints are accurate. Equation 2 can still be used to construct a solution vector of $\mathbf{x}$ which is the closest solution in the least squares sense among all possible vectors $\mathbf{x}$ indicated by SVD property. Note the solution given by Equation 2 will minimize $residule \equiv ||\mathbf{A} \cdot \mathbf{x} - \mathbf{b}||$.

   EXAMPLE 3. *Consider the over-constrained example in Table 2,* $\mathbf{A} = \begin{pmatrix} 1 & 1 & 0 \\ 1 & 0 & 1 \\ 1 & 1 & 1 \\ 0 & 1 & 1 \end{pmatrix}$, $\mathbf{b} = \begin{pmatrix} 5 \\ 3 \\ 7 \\ 5 \end{pmatrix}$, *the representative solution by SVD is* $\mathbf{x} = \begin{pmatrix} 1.57 \\ 3.57 \\ 1.57 \end{pmatrix}$ *which is the closest solution in the least squares sense.*

[2]The reason is the aggregation records may overlap. For example, the sale value for the first quarter can be inferred by the sale value for Jan, Feb, March.

2.2 Maximum Entropy

Another way of approaching the reconstruction of missing values is provided by the application of the Shannon's entropy [37]. Let us slightly re-formulate the problem of missing values by using the formulation of Equation 3.

$$\mathbf{A} \cdot \mathbf{p} = \mathbf{b}' \tag{3}$$

where $\mathbf{p} = \frac{\mathbf{x}}{\mathbf{S}}$, and $S = \sum_{i=1}^{n} x_i$. That is, $\mathbf{p}$ is a vector of normalized missing cells (in which every cell is divided by the summation of them all). Similarly, $\mathbf{b}' = \frac{\mathbf{b}}{\mathbf{S}}$ is a vector of normalized values.

The objective is to recover the vector $\mathbf{p}$. An estimation is provided by applying the entropy concept [37] defined by Shannon. A common method of solving a solution to the problem stated above is known as the RAS algorithm, proposed by Krutihof and discussed in [26; 35]. The RAS algorithm makes use of the entropy concept. The problem can be stated as follows:

PROBLEM 1 (MAXIMUM ENTROPY). *Find* $\mathbf{p} = (p_1, \cdots, p_n)$ *to maximize* $H(\mathbf{p}) = -\sum_{i=1}^{n} p_i \, log(p_i)$ *subject to*

$$\begin{array}{lll} \textit{primary constants} & p_i \geq 0 & \textit{for } i = 1, \cdots, n \\ \textit{additional constraints} & \mathbf{A} \cdot \mathbf{p} = \mathbf{b}' & \\ & \sum_{i=1}^{n} p_i = 1. & \end{array}$$

The objective function $H(\mathbf{p})$ is simply the measure of the entropy of $\mathbf{P}$. Roughly speaking, to maximize the objective function, we try to maximize each missing value's entropy, subject to the additional constraints. The analytical solution to the maximization problem can be obtained using Lagrangian functions [12].

EXAMPLE 4. *Consider the under-constrained example in Table 2,* $\mathbf{A} = \begin{pmatrix} 1 & 1 & 0 \\ 1 & 0 & 1 \end{pmatrix}$, $\mathbf{b} = \begin{pmatrix} 5 \\ 3 \end{pmatrix}$, *the solution by maximum entropy is* $\mathbf{x} = \begin{pmatrix} 0.0015 \\ 4.9985 \\ 2.9985 \end{pmatrix}$.

Notice that the constraints in Equation 3 become non-linear when the sum of the missing values is unknown, since every element on the vector b' is then composed by the inverse of the unknown sum multiplied by a factor (the value of the corresponding component of vector b). So, in those cases, the problem to solve is a maximization of a non-linear function using non-linear constraints. If the sum is known, however, the problem is much simplified: the maximization of a non-linear function over linear constraints.

2.3 Minimum Cross Entropy

In the absence of any other information, the estimates obtained by the RAS algorithm may be poor. Fortunately, with extra information available improved estimates can be obtained. Concretely, if an estimate $\mathbf{p^0}$ of the matrix $\mathbf{p}$ is made available, the cross-entropy method (CE) [23; 15] may be employed. The objective of this method is to minimize the entropy distance between $\mathbf{p}$ and $\mathbf{p^0}$.

PROBLEM 2 (CROSS ENTROPY). *Find* $\mathbf{p} = (p_1, \cdots, p_n)$ *to minimize* $I(\mathbf{p}, \mathbf{p^0}) = \sum_{i=1}^{n} p_i \; log(\frac{p_i}{p_i^0})$ *subject to*

$$\begin{array}{lll} \textit{primary constants} & p_i \geq 0 & \textit{for } i = 1, \cdots, n \\ \textit{additional constraints} & \mathbf{A} \cdot \mathbf{p} = \mathbf{b}' & \\ & \sum_{i=1}^{n} p_i = 1. & \end{array}$$

This problem can also be solved by a Lagrangian function. The estimates provided by the CE method are considerably more accurate than those provided by RAS. However, in terms of our problem of finding missing values, that leaves us with the task of giving a reasonable estimate (and succinct) $\mathbf{p^0}$. That problem can be addressed in many ways. For instance, one could perform a multiple regression or loglinear on the **known** values of the cube (or parts of it), obtaining rough estimates of the missing values. Note the estimates of $\mathbf{p^0}$ is not subject to the constraints.

EXAMPLE 5. *Consider the under-constrained example in Table 2,* $\mathbf{A} = \begin{pmatrix} 1 & 1 & 0 \\ 1 & 0 & 1 \end{pmatrix}$, $\mathbf{b} = \begin{pmatrix} 5 \\ 3 \end{pmatrix}$, *if* $\mathbf{x^0} = \begin{pmatrix} 1 \\ 4 \\ 2 \end{pmatrix}$, *the solution by minimum cross entropy is* $\mathbf{x} = \begin{pmatrix} 1.607 \\ 3.393 \\ 1.393 \end{pmatrix}$; *if* $\mathbf{x^0} = \begin{pmatrix} 2.33 \\ 2.33 \\ 2.33 \end{pmatrix}$, *the solution by minimum cross entropy is* $\mathbf{x} = \begin{pmatrix} 2.287 \\ 2.713 \\ 0.713 \end{pmatrix}$.

The maximum entropy is a special case of the minimum cross entropy principle. If we do not have knowledge of the apriori estimates, we will choose p_i^0 as $1/n$ since it has the most uncertainty. Therefore minimizing cross entropy $\mathbf{I}(\mathbf{p}, \mathbf{p^0})$ is equivalent to maximizing entropy $\mathbf{H}(\mathbf{p})$ as shown by,

$$\begin{aligned} \mathbf{I}(\mathbf{p}, \mathbf{p^0}) &= \sum_{i=1}^{n} p_i log(\frac{p_i}{p_i^0}) \\ &= \sum_{i=1}^{n} p_i log(\frac{p_i}{1/n}) \\ &= logn - (-\sum_{i=1}^{n} p_i log(p_i)) \\ &= logn - \mathbf{H}(\mathbf{p}) \end{aligned} \tag{4}$$

One point is worth pointing out that a solution to Problem 1 and 2 exists and is unique if and only if the constraints do not contradict each other [8].

2.4 Discussion

For a given missing value reconstruction problem, it generally involves two issues:

- whether using constraints will improve the accuracy of estimates.
- which method (SVD, maximum entropy, and cross entropy) we need choose.

There is no definitely answer to the above two issues. However, some observations are as follows:

- If the number of independent constraints m is far less than the number of missing cells n, the representative solution by SVD will deteriorate due to too much flexibility. We may turn to cross entropy method if we have relatively accurate estimates of missing cells or we may directly apply the modeling techniques such as loglinear and logistic models [39] without using constraints.
- On the other hand, the SVD solution in over-constrained case is the best compromise if the inconsistencies among the constraints are not large.
- Another theoretical framework for over-constrained problems is the Maximal Constraint Satisfaction Problem (Max-CSP) where the goal is to minimize the number of constraint violations. In [7] an algorithm for automatically localizing an infeasibility to a minimal set of causative constraints is developed. The idea is to gradually eliminate constraints from the original constraints set until the remaining constitute a minimal infeasible set. The minimal infeasible set contains the important information about which constraints are mutually inconsistent.
- If we know the priorities of the constraints (some aggregation values are more accurate than the others), we can even apply constraint hierarchies [20]. Intuitively, optimal solutions of constraint hierarchies are determined so that they will satisfy as many strong constraints as possible, leaving weaker inconsistent constraints unsatisfied.

2.5 Approximate Answers by Applying Constraint Programming

The missing values $\mathbf{x}$ computed by SVD or entropy is the closest from the overall view. If our goal is to construct and keep only one estimated copy of data, the solution given by SVD would be the most appropriate one. The data warehouse can materialize all the summary information at high levels based on this copy and answer the users' query. The approximate query answer based on one imputed copy will not satisfy users' needs if the user expects the exact answer or at least an approximate answer bounded by a range. We propose to materialize and recover the missing cells on the fly by applying linear programming techniques.

Linear Programming (LP), deals with the problem of maximizing or minimizing a linear objective function over some linear constraints as shown,

PROBLEM 3 (SIMPLE LP). *Find* $\mathbf{x} = (x_1, \cdots, x_n)$ *to minimize or maximize* $V(\mathbf{x}) = \sum_{i=1}^{n} c_i x_i$ *subject to*

primary constants	$x_i \geq 0$	*for* $i = 1, \cdots, n$
additional constraints	$\mathbf{Ax} = \mathbf{b}$	$\mathbf{A}$ *is* $m \times n$, $\mathbf{b}$ *is* $m \times 1$

Given raw data D and aggregation set S_A, we follow the same procedure to get additional constraints $\mathbf{Ax} = \mathbf{b}$. For a range query q, assume the missing values involved in q is $\mathbf{x}^q = (x_{i_1}, \cdots, x_{i_{n'}})$, where $\mathbf{x}^q \subseteq \mathbf{x}$, the objective function to minimize or maximize $V(\mathbf{x}^q) = \sum_{x_i \in \mathbf{x}^q} x_i$ can be mapped to Simple Linear Problem in the general form $V(\mathbf{x}) = \sum_{i=1}^{n} c_i x_i$, where $c_i = 1$ if and only if the missing value $x_i \in \mathbf{x}^q$, otherwise $c_i = 0$. Therefore, the according minimum and maximum value subject to the primary constraints and additional constraints shown in Liner Problem 3 will give the lower bound and upper bound of query q where the true unknown value lies.

One most prominent algorithm for linear programming is *simplex* method. Although the simplex method is not a polynomial-time method in the worst case (some artificial examples show exponential running time), in practice, the simplex method is very fast and in theory the average number of steps can be shown to be linear. [29] gave some routine implementation in C code.

However, the cost to solve the linear programming on the fly is not negligible, especially when the matrix $\mathbf{A}$ is large. One approach is to precompute and save the lower bound and upper bound for each missing cell. This bound can be used to check whether the estimated value computed from other models lies in this range. We may also use the bound for each missing cell to give the coarser bound of range query which contains more than one missing cells. For example, if we have $s_1^L \leq x_1 \leq s_1^U$, and $s_2^L \leq x_2 \leq s_2^U$ ..., then $s_1^L + s_2^L \leq x_1 + x_2 \leq s_2^U + s_2^U$.

3. OTHER CONSTRAINT FORMS

The specific constraints in S_A may take many different forms. For example, some aggregation values may be given an upper bound while some may be given a lower bound. All the distributive and algebraic aggregation [16] such as sum, average, minimum, and maximum can be described by linear algebra in the very similar way [3]. For these constraint forms, the Simple Linear Problem may be generalized as follows,

[3]The binary inequalities ($x_i - x_j \leq c_{ij}$) and holistic functions such as median can not be treated this way.

PROBLEM 4 (GENERAL LP). *Find* $\mathbf{x} = (x_1, \cdots, x_n)$ *to minimize or maximize* $V(\mathbf{x}) = \sum_{i=1}^{n} c_i x_i$ *subject to*

primary constants	$x_i \geq 0$	*for* $i = 1, \cdots, n$
additional constraints	$\mathbf{A}_1\mathbf{x} = \mathbf{b}_1$	
	$\mathbf{A}_2\mathbf{x} \leq \mathbf{b}_2$	
	$\mathbf{A}_3\mathbf{x} \geq \mathbf{b}_3$	$\mathbf{A}_i$ *is* $m_i \times n$, $\mathbf{b}_i$ *is* $m_i \times 1$

Note all of the constraints described above are fundamentally similar in nature and can be transformed from one form to the other in a straightforward manner. It should be pointed out that all the following three problems are polynomially solvable [36].

- Given a matrix $\mathbf{A}$ and a column vector $\mathbf{b}$, test if $\mathbf{Ax} \leq \mathbf{b}$ has a solution, and if so, find one.
- Given a matrix $\mathbf{A}$ and a column vector $\mathbf{b}$, test if $\mathbf{Ax} = \mathbf{b}$ has a nonnegative solution, and if so, find one.
- Given a matrix $\mathbf{A}$ and a column vector $\mathbf{b}$, and a row vector $\mathbf{c}$, test if $max\{\mathbf{cx} \| \mathbf{Ax} \leq \mathbf{b}\}$ is feasible, finite, or unbounded. If it is finite, find an optimal solution.

4. EXPERIMENTAL RESULTS

In this section we show the results of experimenting with census data (http://www.census.gov/main/www/access.html). The data set contains population broken by state and year with domain size of 51 and 5 respectively. There is a total of 255 cells.

4.1 Quality Measures

Since the imputation value is approximate, we need to define ways to compare the results computed by different methods. In this subsection we describe the measures we use for that purpose.

The measure has to do with the error incurred in the approximation methods. Equation 5 and 6 show the relative sum error (RErr) and squared sum error (SSE) for the missing values respectively, where x_j is the true value for missing cell and $\hat{x}_j$ is the estimated value computed from a model. These two measures can tell us roughly how good a missing value imputation algorithm is.

$$RErr = \frac{\sum_j \frac{\|x_j - \hat{x}_j\|}{x_j}}{n} \tag{5}$$

$$SSE = \frac{\sum_j (x_j - \hat{x}_j)^2}{n} \tag{6}$$

4.2 Under-constrained

From the census data set, we randomly choose 10 cells as missing cells and randomly generate some constraints. Each constraint contains a random number of missing cells. Note in under-constrained case, the aggregation value for each constraint is accurate.

Figure 1 and Figure 2 show the $RErr$ and SSE for the different imputation methods. In each figure, we vary the number of independent constraints (0,2,4,6,8,10) and show the accuracy measure for each method, mean substitution, loglinear model, SVD, maximum entropy, and cross entropy (the apriori estimated value from loglinear model).

Some observations from the graph are worth pointing out:

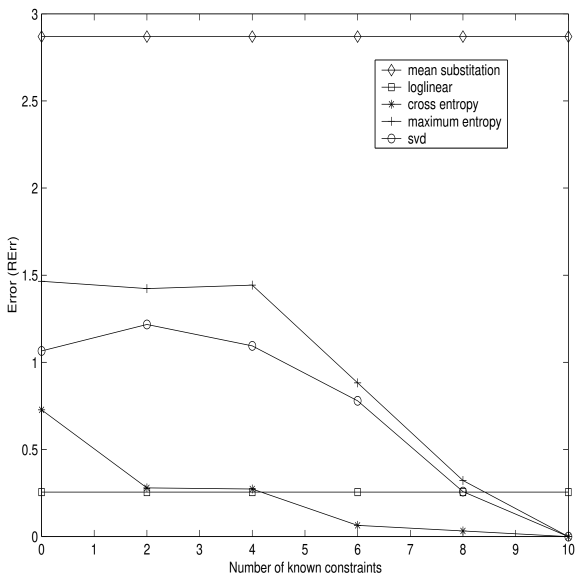

Figure 1: Accuracy (RErr) of different methods over census data set which contains 10 missing cells

- As expected, as the number of independent constraints grows, the results get more accurate for all the methods that do use the constraints (cross entropy, maximum entropy, and SVD.) In the ideal constrained case (10 independent constraints are given), we can restore the missing values losslessly by all the three methods.

- The cross entropy method works best for almost all cases we tried. The exception being cases when there are few (or zero) independent constraints, where loglinear estimation works better. However, the accuracy of cross entropy method depends on the initial estimate $\mathbf{p}^0$ $(\mathbf{x}^0)$. Figure 3 and 4 show the accuracy comparison of cross entropy with loglinear estimate and cross entropy with simple mean substitution. We can see the better estimates from loglinear model give better accuracy result.

- Maximum entropy tends to be outperformed by cross entropy. The reason is that maximum entropy tends to split the probabilities as equally as possible while respecting the constraints.

- SVD will be a good choice when the size of known constraints (compared with the size of missing cells) is large. In that case, more missing values are given as exact values.

- Mean substitution is generally a bad choice, even though it requires very little processing.

4.3 Over-constrained

In this experiment, we fix the size of missing cells as 10 and vary the number of independent constraints (10, 15, 25). For each combination, we test 10 cases by varying the perturbation error of constraints from 1% to 10%. Note here the aggregation values for constraints are not accurate any more. In this case, as we discussed before, no assignment of

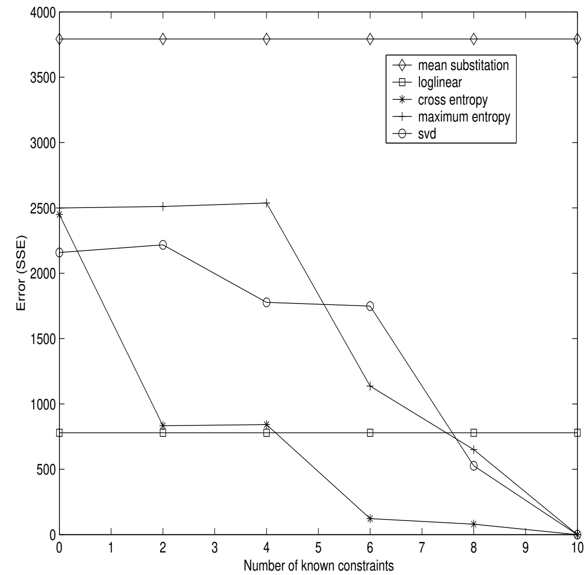

Figure 2: Accuracy (SSE) of different methods over census data set which contains 10 missing cells

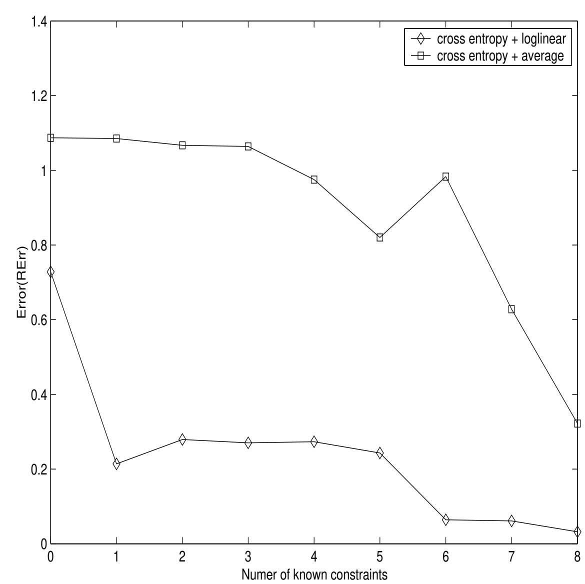

Figure 3: Accuracy (RErr) of cross entropy with loglinear initialization vs. cross entropy with mean initialization over census data set which contains 10 missing cells

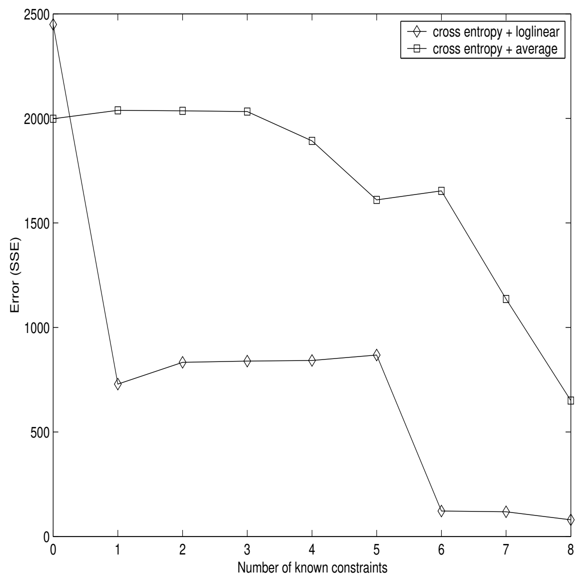

Figure 4: Accuracy (SSE) of cross entropy with loglinear initialization vs. cross entropy with mean initialization over census data set which contains 10 missing cells

values to missing cells satisfies all given constraints. Figure 5 and 6 shows the RErr and SSE for SVD method with different size of constraints.
Some observations from the graph are as follows:

- As expected, as the pertubation error of constraints grows, the results get less accurate for all cases. If there is no perturbation error and the number of constraints is equal to or greater than the missing cells, SVD can restore the missing cells losslessly.
- SVD with the largest number of constraints (25) works best, then SVD with the smallest number of constraints (10), and SVD with medium number of constraints (15) works worst. The explaination has two parts. First, when there exists some conflict constraints (the number of constraints is slightly greater than the number of missing cells), the simple approach (To randomly choose some constraints or to choose those strong constraints if we know the priorities of the constraints) works better than SVD which tries to satisfy all the inconsistent constraints. Second, when the number of constraints is much larger than the number of missing cells, SVD which tries to satisfy all the constraints will improve the performance as the effect of conflicts diminishes along the increase of constraints.

4.4 Discussion of Scalablity

The above experiment only shows the accuracy performance over a small data set. The scalablity is another important issue when we apply those methods to real world problems. Figure 7 gives the number of iterations cross entropy and maximum entropy need to converge. For each iteration, we need one scan of matrix A (bounded by the number of missing values n and the number of constraints m). Both CPU cost and I/O cost may be very high due to the largeness of A.

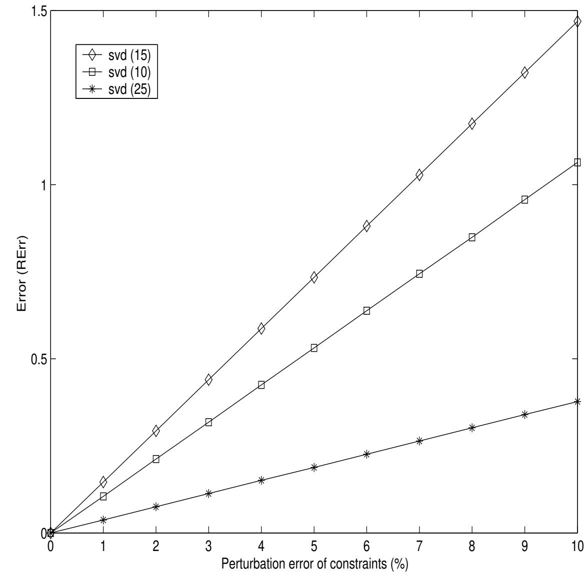

Figure 5: Accuracy (RErr) comparison of SVD with different constraints over census data set which contains 10 missing cells

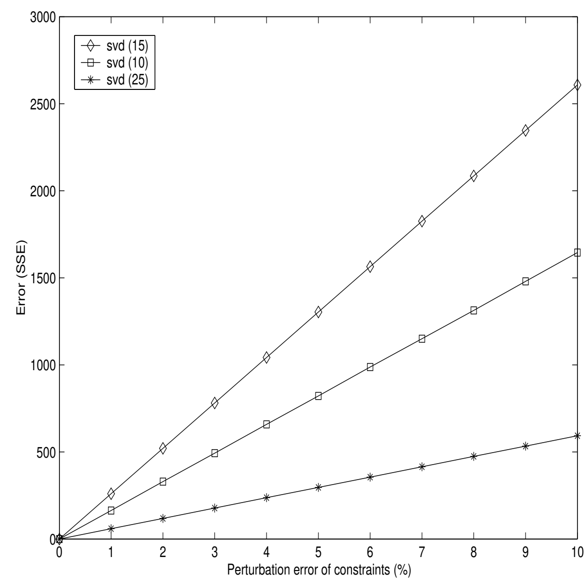

Figure 6: Accuracy (SSE) comparison of SVD with different constraints over census data set which contains 10 missing cells

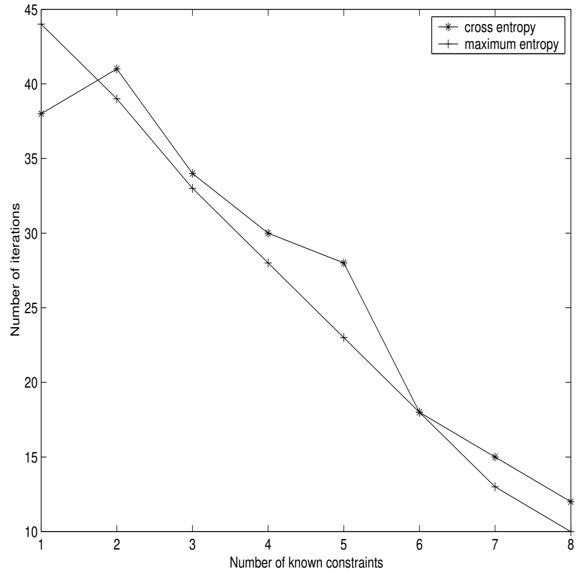

Figure 7: Iterations of different methods

There has been work for large scale linear programming techniques [2; 38] in mathematics field. [2] present the SVD techniques over large scale and sparse matrix ($O(10^6)$ rows or columns). The traditional methods [3; 14] for computing the SVD of dense matrices are not optimal for large sparse matrices such as missing imputation application. [38] presents the heuristic techniques to solve linear programming problems with tens of millions of constraints in a few hundred variables.

5. RELATED WORK

Data cleansing deals with detecting and removing errors and inconsistence from data in order to improve the quality of data. The anomalies involved in data cleansing include: incorrect attribute values, duplicate records, missing values, schema differences among multiple sources etc. While there has been work on schema translation and schema integration [22; 1], on duplicate identification and elimination [19], on general data cleansing framework such as AJAX [11], Potter's Wheel [33], and IntelliClean [24], missing values have received very little attention in the research community.

In statistics, the methods of handling missing data can be grouped into two broad classes. One is data deletion, i.e., ignoring the records with missing values during the analysis. Sometimes, ignoring a certain part of the data may result in the data being biased. Another way of treating missing values is substitution. The substitution methods are well studied in statistics field and some popular approaches such as Mean substitution, Regression methods, Hot deck imputation, Expectation Maximization (EM) approach, Multiple imputation etc. are well developed in statistical literature [25; 18]. However those approaches deal with almost exclusively with parameter estimation rather than missing value prediction [25]. For example, multiple imputation represents the missing values as a random sample and computes statistical inferences which properly reflect the uncertainty due to the missing values.

In data mining and data warehousing field, techniques for imputation of missing values range from stand alone techniques such as using the mean or median of related values (e.g., cells whose coordinates differ from the missing value in just one of the dimension values) to using regression [39] or probabilistic inference such as Markov Chain Monte Carlo methods (MCMC) [28]. The first method [39] involves using a two-part mixture model, which combines the logistic model and loglinear model together, to predict and impute the missing values. The logistic model is applied to predict positions of non-zero cells in the missing cell list while the loglinear model is applied to compute the estimation. The last method [28] involves using a probabilistic model such as a belief network, combined with sampling to avoid the prohibitive price of estimating the probability distributions. The samples are used to obtain *Monte Carlo* estimates for the expectations of functions of the variables involved.

Conditional tables were proposed in [21] to handle incomplete information in relational databases. The conditional table is an extension of a table with one more column containing logical formulas attached with the tuples of the relation. The new table is a relation with constant and variables in which no variable occurs twice. Materializing the conditional table usually generates many tables. For the missing continuous attribute values, it is unknown how to apply conditional table techniques to estimation. The rule induction techniques such as decision tree [31; 32], decision tables [17; 18] are studied to predict the value of the missing attribute based on the values of the other attributes in that tuple. However, techniques based on rule induction models can only handle missing data for categorical attributes with low cardinality domains (few values). Recently, in [30], Prodromidis and Stolfo propose the prediction of attributes in a database schema using an auxiliary classifier, or more accurately, a prediction model. Such prediction model can be built for example using regression methods (e.g., Cart [4; 27; 10]). This is done in [30] to "bridge" the differences between different schemas in the context of integrating classifier agents.

Some work has investigated the inconsistency between summary information and low level raw data. Chin and Kossowski, in [6], presented an efficient inference control (called auditing) for *one* dimensional range sum queries on statistical databases. Based on all previous answered queries, they developed an algorithm (O(n) time and storage) to decide whether a new sum range query could lead to compromise. The auditing problem can be thought as an inverse one to our problem. In this paper, we aim to discover the missing values from all available summary data by applying linear algebra, entropy, or constraint programming technique and the summary data here is associated with multidimensional range. Faloutsos et al., in [9] focused on recovering detailed information from summary data (under-constrained) by maximizing entropy where all detailed data are unavailable.

6. CONCLUSIONS

In this paper, we describe our efforts to recover the missing values by utilizing the summary constraints. We investigate different techniques (SVD, maximum entropy, cross entropy, and constraint programming) and bring to bear on problems in recovering missing values when some summary information are available. We classify the missing values

problem into three sub-scenarios: ideal-constrained, under-constrained, and over-constrained. The simple method SVD from linear algebra is applicable to reconstruct missing values by solving linear constraint equations. This method gives more accurate estimation than the traditional methods (without using constraints) especially when the number of independent constraints are close to the number of missing cells in under-constrained case or larger than the number of missing cells in over-constrained case.
We also investigate maximum entropy and cross entropy techniques to reconstruct missing values. The cross entropy technique even improves the estimates if the relatively accurate estimates of the missing values are available as input parameters. The experiment results done over the census data set support the claim that by utilizing the constraints we do improve the accuracy of estimates. Furthermore, we investigate the constraint programming technique to answer users' query on the fly with a tighter bound.
There are some aspects of this work that merit futher research. Among them,

- We are trying to investigate the scalablity issues. As all the methods (SVD, maximum entropy, cross entropy and constraint programming) are iterative in essential, it is prohibitive to apply those methods directly . One idea we are experimenting is to divide the data set into chunks and each chunk (fitted in memory) is associated with its own missing values and constraints. Of course, we will lose some accuracy as some cross-chunk constraints have to be discarded.
- We will investigate how to combine the traditional modeling techniques with summary constraints especially for under-constrained case.
- We will also investigate techniques when the dimension attributes have some missing values as shown by Tuple 3 and 4 in Figure 1. We will study the pattern of missingness among continuous dimensions and their relationships to the categorical ones. One specific technique for joint modeling and imputation of incomplete categorical and continuous attributes is the general location model [34] which combines a) a loglinear model for describing relationships among categorical dimensions with b) a multivariate linear regression for describing the correlations among continuous dimensions and their relationships to the categorical ones.

7. REFERENCES

[1] *Bulletin of the Technical Committee on Data Engineering*, 23(4), 2000.

[2] M. Berry. Large scale sparse singular value computations. *The International Journal of Supercomputer Applications*, 6(1):13–49, 1992.

[3] M. Berry and A. Sameh. An overview of parallel algorithms for the singular value and dense symmetric eigenvalue problems. *Journal of Computational and Applied Mathematics*, 27:191–213, 1989.

[4] L. Breiman, J. Friedman, R. Olshen, and C. Stone. *Classification and regression trees*. Wadsworth, Belmont, CA, 1984.

[5] N. Chakravarti. Some results concerning post-infeasibility analysis. *European Journal of Operation Research*, 73:139–143, 1994.

[6] F. Chin and P. Kossowski. Efficient inference control for range sum queries on statistical data bases. In *Proceedings of the First LBL Workshop on Statistical Database Management*, Dec 1981.

[7] J. W. Chinneck and E. W. Dravnieks. Locating minimal infeasible constraint sets in linear programs. *ORSA Journal on Computing*, 3(2):157–168, 1991.

[8] J. Darroch and D. Ratcliff. Generalized iterative scaling for log-linear models. *Annals of Mathematical Statistics*, 43:1470–1480, 1972.

[9] C. Faloutsos, H. Jagadish, and N. Sidiropoulos. Recovering information from summary data. In *Proceedings of the 23rd VLDB Conference*, 1997.

[10] J. Friedman. Multivariate adaptive regression splines. *The Annals of Statistics*, 19(1):1–141, 1991.

[11] H. Galhardas, D. Florescu, D. Shasha, E. Simon, and C. Saita. Declarative data cleaning: language, model, and algorithms. In *Proceedings of the 27th VLDB Conference*, Sept 2001.

[12] A. Golan, G. Judge, and D. Miller. *Maximum entropy econometrics: robust estimation with limited data*. John Wiley, New York, 1996.

[13] C. Golub and C. V. Loan. *Matrix computations*. Johns Hopkins University Press, 1989.

[14] G. Golub and W. Kahan. Calculating the singular values and pseudoinverse of a matrix. *SIAM Journal of Numerical Analysis*, 2(3):205–224, 1965.

[15] I. Good. Maximum entropy for hypothesis formulation, especially for multidimensional contingency tables. *Annals of Mathematical Statistics*, 34:911–934, 1963.

[16] J. Gray, A. Bosworth, A. Layman, and H. Pirahesh. Data cube: A relational aggregation operator generalizing group-by, cross-tab, and sub-totals. In *Proceedings of the International Conference on Data Engineering*, 1996.

[17] J. Grzymala-Busse. On the unknown attribute values in functional dependencies. In *Proceedings of Methodologies for Intelligent Systems, Lecture Notes in AI*, pages 368–377, 1991.

[18] J. Grzymala-Busse and M. Hu. A comparison of several approaches to missing attribute values in data mining. In *Proceedings of the second International Conference on Rough Sets and Current Trends in Computing*, 2000.

[19] M. A. Hernandez and S. J. Stolfo. The merge/purge problem for large databases. In *Proceedings of the ACM SIGMOD Conference*, 1995.

[20] H. Hosobe. A scalabe linear constraint solver for user interface construction. In *Proceedings of Constraint Programming*, pages 218–233, 2000.

[21] T. Imielinski and W. Lipski. Incomplete information in relational databases. *Journal of ACM*, 31(4):761–791, 1984.

[22] V. Kashyp and A. P. Sheth. Semantic and schematic similarities between database objects: a context-based approach. *VLDB Journal*, 5(4):276–304, 1996.

[23] J. Kullback. *Information theory and statistics.* John Wiley, New York, 1959.

[24] M. Lee, T. W. Ling, and W. L. Low. Intelliclean: a knowledge-based intelligent data cleaner. In *Proceedings of ACM Knowledge Discovery and Data Mining*, Aug 2000.

[25] R. Little and D. B. Rubin. *Statistical analysis with missing data.* New York, John Wiley and Sons, 1987.

[26] R. Lynch. An assessment of the RAS method for updating input-output Tables. In I. Sohn, editor, *Readings in Input-Output Analysis*, pages 271–284. Oxford University Press, 1986.

[27] R. Myers. *Classical and modern regression with applications.* Duxbury, Boston, MA, 1986.

[28] R. Neal. Probabilistic inference using markov chain monte carlo methods. Technical Report CRG-TR-93-1, Dept. of Computer Science, University of Toronto, 1993.

[29] W. H. Press, B. P. Flannery, S. A. Teukolsky, and W. T. Vetterling. *Numerical recipes in C, the art of scientific computing.* Cambridge University Press, 1988.

[30] A. L. Prodromidis and S. Stolfo. Mining databases with different schemas: integrating incompatible classifiers. In *Proceedings of the International Conference on Knowledge Discovery and Data Mining*, 1998.

[31] J. Quinlan. Induction of decision trees. *Machine Learning*, 1:81–106, 1986.

[32] J. Quinlan. Unknown attribute values in induction. In *Proceedings of the Sixth International Machine Learning Workshop*, pages 164–168, 1989.

[33] V. Raman and J. M. Hellerstein. Potter's wheel: an interactive data cleaning system. In *Proceedings of the 27th VLDB Conference*, Sept 2001.

[34] J. L. Schafer. *Analysis of incomplete multivariate data.* Chapman and Hall, London, 1997.

[35] M. Schneider and S. Zenios. A comparative study of algorithms for matrix balancing. *Operations Research*, 38:439–455, 1990.

[36] A. Schrijver. *Theory of linear and integer programming.* John Wiley and Sons, 1986.

[37] C. Shannon. A mathematical theory of communication. *Bell System Technical Journal*, 27:379–423, 1948.

[38] M. Wagner, J. Meller, and R. Elber. Large-scale linear programming techniques for he design of protein folding potentials. *Technical Report 2002-02, Old Dominion University*, 2002.

[39] X. Wu and D. Barbará. Modeling and imputation of large incomplete multidimensional datasets. In *Proceedings of the Fourth International Conference on Data Warehousing and Knowledge Discovery*, Sept. 2002.

About the Authors

Xintao Wu received his BS degree in Information Science from University of Science and Technology of China in 1994, a MS degree in Computer Science from Chinese Academy of Space Technology in 1997, and a Ph.D. in Information Technology from George Mason University in 2001. He is currently an Assistant Professor in the Computer Science Department at University of North Carolina at Charlotte. His major research interests include data mining and knowledge mining, data cleansing and data warehousing.

Daneil Barbará is an Associate Professor in the Information and Software Engineering Department at George Mason University. His current research interests are Data Mining and Data Warehousing. Previously, he has worked in Bell Communication Research and Panasonic Laboratories. He earned a Ph.D. from the Computer Science Department of Princeton University in 1985.