



ELSEVIER

Decision Support Systems 34 (2002) 19–39

Decision Support  
Systems

www.elsevier.com/locate/dsw

# DIRECT: a system for mining data value conversion rules from disparate data sources

Weiguo Fan<sup>a,\*</sup>, Hongjun Lu<sup>b</sup>, Stuart E. Madnick<sup>c</sup>, David Cheung<sup>d</sup>

<sup>a</sup>Department of Computer and Information Systems, University of Michigan Business School, 701 Tappan, Ann Arbor, MI 48109-1234, USA

<sup>b</sup>Department of Computer Science, Hong Kong University of Science and Technology, Hong Kong, China

<sup>c</sup>Sloan School of Management, MIT, Cambridge, MA 02139, USA

<sup>d</sup>Department of Computer Science, Hong Kong University, Hong Kong, China

Accepted 1 October 2001

## Abstract

The successful integration of data from autonomous and heterogeneous systems calls for the resolution of semantic conflicts that may be present. Such conflicts are often reflected by discrepancies in attribute values of the same data object. In this paper, we describe a recently developed prototype system, *D*iscovering and *R*econciling ConflicTs (**DIRECT**). The system mines data value conversion rules in the process of integrating business data from multiple sources. The system architecture and functional modules are described. The process of discovering conversion rules from sales data of a trading company is presented as an illustrative example. © 2002 Elsevier Science B.V. All rights reserved.

*Keywords:* Data integration; Data mining; Semantic conflicts; Data visualization; Statistical analysis; Data value conversion

## 1. Introduction

Data mining or knowledge discovery from databases (KDD) is defined by Fayyad et al. [11] as the non-trivial process of identifying valid, novel, potentially useful and ultimately understandable hidden structure, patterns, knowledge from data. Organizations are taking advantage of “data mining” techniques to leverage the vast amount of data to make better business decisions [26]. For example, data mining has been used for customer profiling in CRM and customer service support [17], credit card

application approval, fraud detection, telecommunication network monitoring, market-basket analysis [11], healthcare quality assurance [36], and many other decision-making areas [3].

In this paper, we are extending the data-mining paradigm to a new application area: data integration from disparate sources [10,21], in which database administrators often find it very hard to make decisions on the integration of those semantically related attributes [28].

It is known that the successful integration of data from autonomous and heterogeneous systems calls for the resolution of semantic conflicts that may be present. Such conflicts are often reflected by discrepancies in attribute values of the same data object. Traditional approaches to achieving semantic interoperability among heterogeneous and autonomous sys-

\* Corresponding author.

E-mail addresses: wfan@umich.edu (W. Fan), luhj@cs.ust.hk (H. Lu), smadnick@mit.edu (S.E. Madnick), dcheung@cs.hku.hk (D. Cheung).

tems can be identified as either *tightly coupled* or *loosely coupled* [34]. In tightly coupled systems, semantic conflicts are identified and reconciled a priori in one or more shared schema, against which all user queries are formulated. This is typically accomplished by the system integrator or DBA (Database Administrator) who is responsible for the integration project. In a loosely coupled system, conflict detection and resolution is the responsibility of users, who must construct queries that take into account potential conflicts and identify operations needed to circumvent them. In general, conflict detection and reconciliation is known to be a difficult and tedious process since the semantics of data are usually present implicitly and often ambiguous. This problem poses even greater difficulty when the number of sources increases exponentially and when semantics of data in underlying sources changes over time. To provide support for information exchange data among heterogeneous and autonomous systems, one approach is to use context associated with the data [13,4]. The basic idea is to partition *data sources* (databases, web-sites, or data feeds) into groups, each of which shares the same *context*, the latter allowing the semantics of data in each group to be codified in the form of *meta-data*. A *context mediator* takes on the responsibility for detecting and reconciling semantic conflicts that arise when information is exchanged between systems (data sources or receivers). This is accomplished through the comparison of the contexts associated with sources and receivers engaged in data exchange [5].

As attractive as the above approach may seem, it requires data sources and receivers participating in information exchange to provide the required contexts. In the absence of other alternatives, this codification will have to be performed by the system integrator. This, however, is an arduous process, since data semantics is not always explicitly documented, and often evolves with the passage of time. This motivated our work reported in this paper: discovering and reconciling conflicts in disparate information sources by mining the data themselves. The rationale is simple. Most semantic conflicts, such as different data types and formats, units, granularity, synonyms, different coding schemes, etc., exist because data from different sources are stored and manipulated under different context, defined by the systems and applications. Therefore, those conflicts should be uniformly present

in the data. That is, for a real world entity, although the values for the same attribute may be different since they are obtained from different sources, such differences should be present similarly in all entities. It would be reasonable to expect that such discrepancies can be identified and the relationships between conflict attributes can be discovered.

This paper reports the results of our work on discovering and reconciling semantic conflicts using data mining techniques to help DBAs to make better decisions on resolving data value-related conflicts. Our proposed framework is implemented in a prototype system called *DIScovering and REconciling Conflicts (DIRECT)*. Our focus for the moment is on business and financial data, though the framework can be easily extended for other kinds of data. There are a number of reasons to start with business data. First, such data are easily available. Second, the relationships among the attribute values for business data are relatively simple compared to engineering or scientific data. Most of such relationships are linear or product of attributes. On the other hand, business data do have some complex factors. For example, monetary figures are often rounded up to the unit of currencies. Such rounding errors are in fact random noises that are mixed with the value conflicts caused by different context. While the basic techniques implemented in the prototype system are known, our contribution is to integrate various techniques (with necessary modifications and extensions) to provide a solution to a practical problem which has been thought difficult to solve.

### 1.1. Related work

Our work is closely related to work on discovering and resolving data value conflicts for data integration in the context of heterogeneous and autonomous database systems. In an earlier work, Dayal [8] proposed the use of aggregate functions, e.g., average, maximum, minimum, etc., to resolve discrepancies in attribute values. Demichiel [9] proposed to use virtual attributes and partial values to solve the problem of failing to map an attribute value to a definite value. However, no details were given on how to map conflicting values in to the common domain of virtual attributes. Tseng et al. [37] further generalized the concept of partial values into probabilistic partial values to capture the uncertainty in attributes values:

the possible values of an attribute are listed and the probabilities are given to indicate their likelihood. Lim et al. [22] proposed an extended relational model based on *Dempster–Shafer Theory of Evidence* to deal with the situation where uncertain information arises when the database integration process requires information not directly represented in the component databases but can be obtained through some of the data. The extended relation uses evidence sets to represent uncertainty information, which allows probabilities to be attached to subsets of possible domain values. Scheuermann and Chong [32,33] adopted a different view of conflicting attribute values: different values mean different roles the attribute is performing. Therefore, it is not necessary to reconcile the conflicts. What is needed is to be able to use appropriate values for different applications. The work of Agarwal et al. [2] addressed the same problem addressed in this paper: resolving conflicts in non-key attributes. They proposed an extended relational model, flexible relation to handle the conflicting data values from multiple sources. No attempts were made to resolve the conflicts by converting the values.

As evident from the above discussion, most of the work in the existing literature have placed their emphasis on determining the value of an attribute involving semantic conflicts, while considerably less work on discovering and resolving conflicts through the use of conversion functions has been reported. Sciore et al. [35] have proposed the use of conversion functions to solve the semantic heterogeneity problem. However, the details of discovering such functions were not provided.

### 1.2. Our proposed data mining approach

Our approach is distinct from previous efforts in the study of semantic conflicts in various ways. First, we begin with a simpler classification scheme of conflicts presented in data values. We classify such conflicts into two categories: *context dependent* and *context independent*. While context-dependent conflicts are a result of disparate interpretations inherent in different systems and applications, context-independent conflicts are often caused by errors and external factors. As such, context-dependent conflicts constitute the more predictable of the two and can often be rectified by identifying appropriate *data*

*value conversion rules*, or simply *conversion rules*, which we will shortly define. These rules can be used for data exchange or integration directly.

Second, our primary contribution in this paper is the development of a methodology and associated techniques that “mine” data conversion rules from data originating from disparate systems that are to be integrated. The approach requires a training data set consisting of tuples merged from data sources to be integrated or exchanged. Each tuple in the data set should represent a real world entity and its attributes (from multiple data sources) model the properties of the entity. If semantic conflicts exist among the data sources, the values of those attributes that model the same property of the entity will have different values. A mining process first identifies the attribute sets, each of which involves some conflicts. After identifying the relevant attributes, models for possible conversion functions, the core parts of conversion rules are selected. The selected models are then used to analyze the data to generate candidate conversion functions. Finally, a set of the most appropriate functions is selected and used to form the conversion rules for the involved data sources.

The techniques used in each of the above steps can vary depending on the property of data to be integrated. In this paper, we describe a statistics-based prototype system for integrating financial and business data, **DIRECT**. The system uses partial correlation analysis to identify relevant attributes, Bayesian information criterion for candidate model selection, and robust regression for conversion function generation. Conversion function selection is based on the support of rules. Experiment conducted using both synthetic data and a real world data set indicated that the system successfully discovered the conversion rules among data sources containing both context-dependent and -independent conflicts.

The contributions of our work can be summarized as follows. First, our classification scheme for semantic conflicts is more practical than previous proposals. For those context-dependent conflicts, proposed data value conversion rules can effectively represent the quantitative relationships among the conflicts. Such rules, once defined or discovered, can be used in resolving the conflicts during data integration. Second, a general approach for mining the data conversion rules from actual data values is proposed. A

prototype system has been developed to fully automate the whole discovery process. While the aim of the system is to provide automatic discovery, it can also be used in an interactive way to allow the user to fine-tune the system and speed up the whole mining process. Experiment results on both synthetic data and real-world data set demonstrated the promising of our new proposed approach.

The remainder of the paper is organized as follows. We first define data value conflicts conversion rules in Section 2. Section 3 describes the architecture and the functional modules of **DIRECT**. Section 4 describes the techniques implemented in various modules. Section 5 presents two example processes of mining conversion rules from both synthetic data and a real trading company data. Section 5 concludes the paper.

## 2. Data value conflicts and conversion rules

In this section, we define data value conflicts and conversion rules. We will use the relational model for ease of explanation without losing generality. Furthermore, we assume that the entity identification problem, i.e., identifying the records representing the same real world entity from multiple data sources is solved using other methods [38,23]. Therefore, attributes of different sources can be merged into one relation with the attributes modelling the properties of the entity. Each tuple in the relation represents a real-world entity. If semantic conflicts exist among the data sources, the values of those attributes that model the same property of an entity will be different. We call such difference in data attribute values *data value conflict*.

As an illustrative example, Table 1 lists sample tuples from two relations, *stock* and *stk\_rpt*:

*stock* (**stock**, currency, volume, high, low, close);  
*stk\_rpt* (**stock**, price, volume, value);

where relation *stk\_rpt* is produced from *stock* by stock brokers for its clients based on the daily quotation of stock prices.

As mentioned earlier, we assume that the entity identification problem has been largely solved. Therefore, stocks with the same code refer to the same company's stock. For example, tuple with *stock* = 100 in relation *stock* and tuple with *stock* = 100 in relation *stk\_rpt* refer to the same stock. If we know that both *stk\_rpt.price* and *stock.close* are the closing price of a stock of the days, and similarly, both *stock.volume* and *stk\_rpt.volume* are the number of shares traded during the day, it will be reasonable to expect that, for a tuple  $s | s \in stock$  and a tuple  $t | t \in stk\_rpt$ , if  $s.stock = t.stock$ , then  $s.close = t.price$  and  $s.volume = t.volume$ . However, from the sample data, this does not hold. In other words, by analyzing the data, we can get the conclusion that, *data value conflicts* exist between the two data sources. A bit formally, we can define data value conflicts as follows:

*Definition.* Given two data sources  $DS_1$  and  $DS_2$  and two attributes  $A_1$  and  $A_2$  modelling the same property of a real world entity type in  $DS_1$  and  $DS_2$  respectively, if  $t_1 \in DS_1$  and  $t_2 \in DS_2$  represent the same real-world instance of the object but  $t_1.A_1 \neq t_2.A_2$ , then we say that a data value conflict exists between  $DS_1$  and  $DS_2$ .

Table 1  
Example tuples from relation *stock* and *stk\_rpt*

<i>stock</i>						<i>stk_rpt</i>			
Stock	Currency	Volume	High	Low	Close	Stock	Price	Volume	Value
100	1	438	100.50	91.60	93.11	100	65.18	438,000	28,548,840.00
101	3	87	92.74	78.21	91.35	101	164.43	87,000	14,305,410.00
102	4	338	6.22	5.22	5.48	102	31.78	338,000	10,741,640.00
104	1	71	99.94	97.67	99.04	104	69.33	71,000	4,922,430.00
111	0	311	85.99	70.22	77.47	111	30.99	311,000	9,637,890.00
115	2	489	47.02	41.25	41.28	115	41.28	489,000	20,185,920.00
120	3	370	23.89	21.09	22.14	120	39.85	370,000	14,744,500.00
148	3	201	23.04	19.14	21.04				
149	1	113	3.70	3.02	3.32				

To resolve data value conflicts, we need to not only identify such conflicts, but also discover the quantitative relationships among the attribute values involving such conflicts. For ease of explanation, we adopt the notation of Datalog rules for representing such relationship and call them *data value conversion rules* (or simply *conversion rules*) which take the form  $head \leftarrow body$ . The head of the rule is a predicate representing a relation in one data source. The body of a rule is a conjunction of a number of predicates, which can either be extensional relations present in underlying data sources, or built-in predicates representing arithmetic or aggregate functions. Some examples of data conversion rules are described below.

*Example.* For the example given above, we can have the following data value conversion rule:

$$stk\_rpt(stock, price, volume, value) \leftarrow exchange\_rate(currency, rate), stock(stock, currency=volume\_in\_K, high, low, close), price=close*rate, volume=volume\_in\_K*1000, value=price*volume.$$

where *exchange-rate* is another relation in the data source of stock.<sup>1</sup>

*Example.* For conflicts caused by synonyms or different representations, it is always possible to create lookup tables which form part of the conversion rules. For example, to integrate the data from two relations  $D_1.student(sid, sname, major)$  and  $D_2.employee(eid, ename, salary)$ , into a new relation  $D_1.std\_emp(id, name, major, salary)$ , we can have a rule

$$D_1.std\_emp(id, name, major, salary) \leftarrow D_1.student(id, name, major), D_2.employee(eid, name, salary), D_1.same\_person(id, eid).$$

where *same-person* is a relation that defines the correspondence between the student id and employee id. Note that, when the size of the lookup table is small, it can be defined as rules without bodies. One widely cited example of conflicts among student

grade points and scores can be specified by the following set of rules:

$$\begin{aligned} D_1.student(id, name, grade) &\leftarrow D_2.student(id, name, score), score\_grade(score, grade). \\ score\_grade(4, 'A'). \\ score\_grade(3, 'B'). \\ score\_grade(2, 'C'). \\ score\_grade(1, 'D'). \\ score\_grade(0, 'F'). \end{aligned}$$

One important type of conflicts mentioned in literature in business applications is *aggregation conflict* [19]. Aggregation conflicts arise when an aggregation is used in one database to identify a set of entities in another database. For example, in a transactional database, there is a relation  $sales(date, customer, part, amount)$  that stores the sales of parts during the year. In an executive information system, relation  $sales\_summary(part, sales)$  captures total sales for each part. In some sense, we can think *sales* in relation  $sales\_summary$  and *amount* in  $sales$  are attributes with the same semantics (both are the sales amount of a particular part). But there are conflicts as the *sales* in  $sales\_summary$  is the summation of the amount in the  $sales$  relation. To be able to define conversion rules for attributes involving such conflicts, we can extend the traditional Datalog to include aggregate functions. In this example, we can have the following rules

$$sales\_summary(part, sales) \leftarrow sales(date, customer, part, amount), sales = SUM(part, amount).$$

where  $SUM(part, amount)$  is an aggregate function with two arguments. Attribute *part* is the *group-by* attribute and *amount* is the attribute on which the aggregate function applies. In general, an aggregate function can take  $n+1$  attributes where the last attribute is used in the aggregation and others represent the “group-by” attributes. For example, if the  $sales\_summary$  is defined as the relation containing the total sales of different parts to different customers. The conversion rule could be defined as follows:

$$sales\_summary(part, customer, sales) \leftarrow sales(date, customer, part, amount), sales = SUM(part, customer, amount).$$

<sup>1</sup> In fact, the relation of *exchange-rate* can be discovered from the given relation *stock* and *stk-rpt*.

We like to emphasize that it is not our intention to argue about appropriate notations for data value conversion rules and their respective expressive power. For different application domains, the complexity of quantitative relationships among conflicting attributes could vary dramatically and this will require conversion rules having different expressive power (and complexity). The research reported here is primarily driven by problems reported for the financial and business domains; for these types of applications, a simple Datalog-like representation has been shown to provide more than adequate representations. For other applications (e.g., in the realm of scientific data), the form of rules could be extended. Whatever the case may be, the framework set forth here can be used profitably to identify conversion rules that are used for specifying the relationships among attribute values involving conflicts.

### 3. DIRECT: the system architecture

In this section, we briefly describe the architecture of the system, **DIRECT**, developed at NUS with the objective of mining data value conversion rules. As shown in Fig. 1, the system consists of a data visualization module and a set of conversion rule discovery modules. Conversion rule discovery modules of **DIRECT** include *Relevant Attribute Analysis*, *Candidate Model Selection*, *Conversion Function Generation*, *Conversion Function Selection*, *Conversion Rule Formation* and some planned future modules, like

aggregation rule discovery, etc. In case no satisfactory conversion functions are discovered, training data is reorganized and the mining process is re-applied to confirm that there are indeed no conversion functions. The system also tries to learn from the mining process by storing used model and discovered functions in database (the *Model Base*) to assist later mining activities. Fig. 2 depicts the rule discovery process.

- *Relevant attribute analysis* is the first step of the data value conversion rule mining process. For a given attribute in the integrated data set, the relevant attribute analysis module will determine the other attributes in the data set that are semantically equivalent or required for determining the values of semantically equivalent attributes. In the previous *stock* and *stk\_rpt* example, *stock.close* and *stk\_rpt.price* are semantically equivalent attributes because both of them represent the last trading price of the day. If we want to find the attributes that are related with *stock.close*, relevant attribute analysis module will find that it is related with *stk\_rpt.price* and one latent attribute *exchange\_rate*. As shown in Fig. 2, meta data about the data sources may be used to help the relevance analysis. For example, data types of attributes, an easily available type of meta data, can be used to reduce the search space for semantically relevant attributes.

- *Candidate model selection* is the second module of the conversion rule mining process. To find a data value conversion rule among semantically related attributes, various models must first be assigned to specify their quantitative relationships. Since the

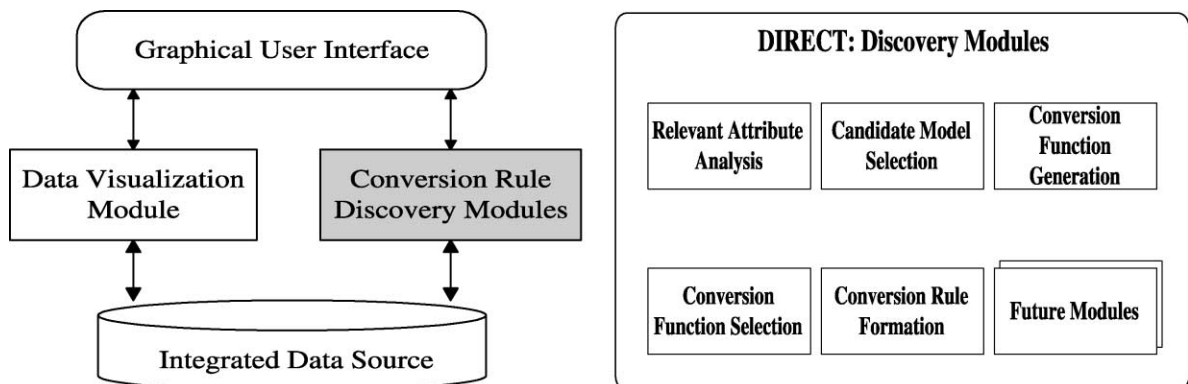


Fig. 1. **DIRECT**: The reference architecture and conversion rule discovery modules.

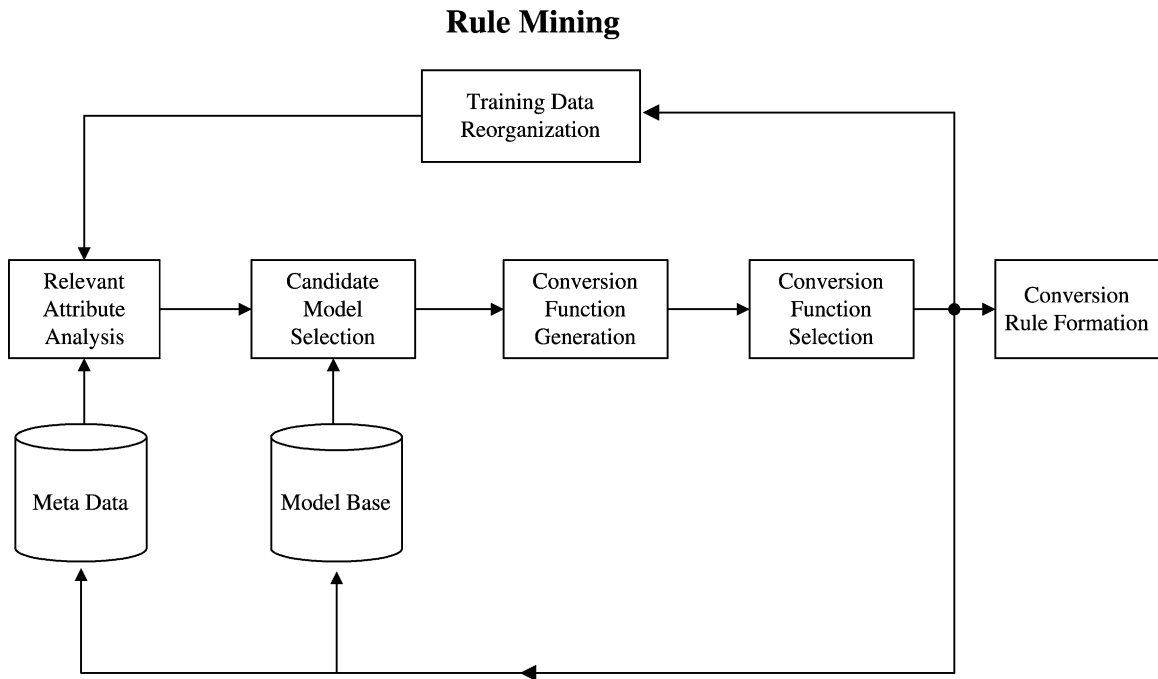


Fig. 2. Discovering data value conversion rules from data.

model space may be of a very large size,<sup>2</sup> it is essential that we can limit the model search space to make our approach viable and practical in real practice. Candidate model selection is such a module to find the best potential models for the data from which the conversion functions will be generated.

- *Conversion function generation* is a module that can efficiently estimate the model parameters and goodness of those candidate models that are generated by the candidate model selection module. Note that the data set may contain a lot of outliers, or error data points, the conversion function generation module must have a very robust property against those outliers. That is, it should still give good estimations of the model parameters even in a contaminated environment.

- *Conversion function selection* and *Conversion rule formation*. It is often the case that the mining process generates more than one conversion functions because it is usually difficult for the system to determine the optimal one. The conversion function

selection module needs to develop some measures or heuristics to select the best conversion functions from the candidates. With the selected functions, some syntactic transformations are applied to form the data conversion rules as specified in Section 2.

#### 4. System implementation

In this section, we give a detailed description of the implementation of our **DIRECT** system. Most of the techniques we used are from statistics field, with modification to meet our special needs of conversion rule mining.

##### 4.1. Data visualization

Data visualization is one of the most important modules of the system. Before the users start the mining process, they can first use the data visualization module to visualize their data and gain some idea about the distribution of their values. As shown in Fig. 3, we provide several statistics-lines on the visualization diagram: minimum, maximum and mean

<sup>2</sup> For a linear model involving 10 independent attributes, the model space will be of the size  $2^{10} = 1024$ .

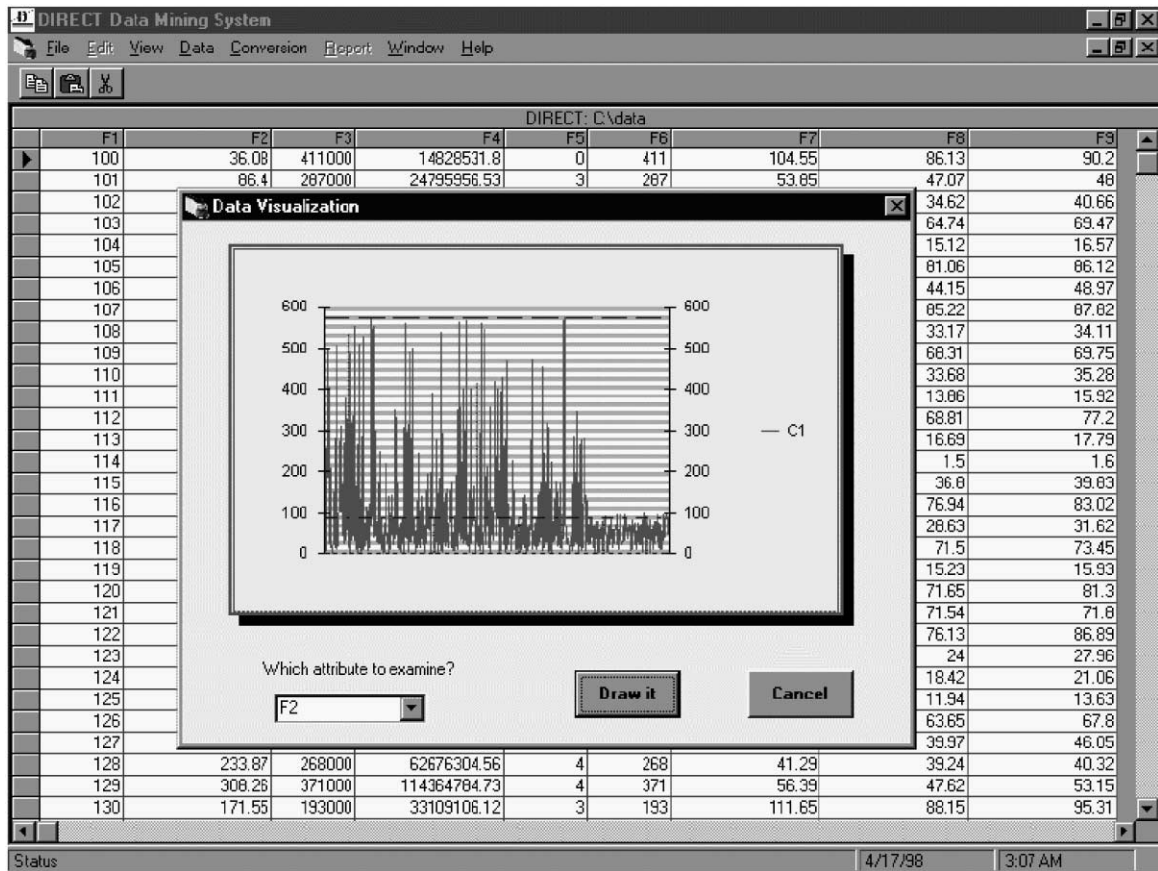


Fig. 3. A snapshot of the output of **DIRECT** visualisation module.

value lines to help users to quickly identify the required information.

Data visualization is complementary to the whole conversion rule discovery modules. A typical example is the relevant attribute analysis, which is described in a separate section. As shown in Fig. 4, scatter plot, which is very powerful in examining the relationship among two attributes, is provided to help users to further determine which attributes should be included in the final model. This can help to restrain the model search space, and thus can greatly improve the system's performance in terms of speed and accuracy.

#### 4.2. Relevant attribute analysis

The basic techniques used to measure the (linear) association among numerical variables in statistics is

*correlation analysis* and *regression*. Given two variables,  $X$  and  $Y$  and their measurements  $(x_i, y_i)$ ;  $i = 1, 2, \dots, n$ , the strength of association between them can be measured by *correlation coefficient*  $r_{x,y}$ .

$$r_{x,y} = \frac{\sum(x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum(x_i - \bar{x})^2} \sqrt{\sum(y_i - \bar{y})^2}} \quad (1)$$

where  $\bar{x}$  and  $\bar{y}$  are the mean of  $X$  and  $Y$ , respectively. The value of  $r$  is between  $-1$  and  $+1$  with  $r=0$  indicating the absence of any linear association between  $X$  and  $Y$ . Intuitively, larger values of  $r$  indicate a stronger association between the variables being examined. A value of  $r$  equal to  $-1$  or  $1$  implies a perfect linear relation. While correlation analysis can reveal the strength of linear association, it is based on an assumption that  $X$  and  $Y$  are the only

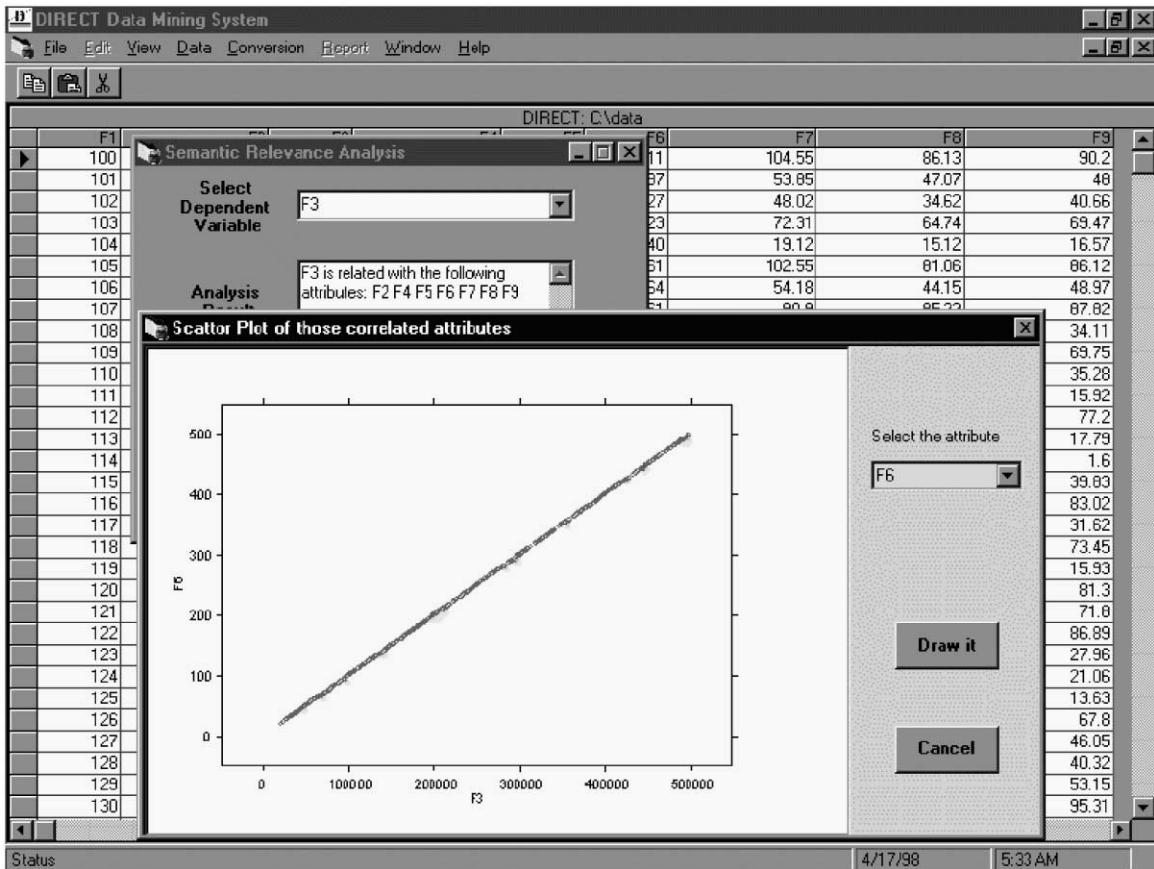


Fig. 4. Output of relevant attribute analysis and their scatter plot.

two variables under the study. If there are more than two variables, other variables may have some effects on the association between  $X$  and  $Y$ . Partial correlation analysis (PCA) is a technique that provides us with a single measure of linear association between two variables while adjusting for the linear effects of one or more additional variables. Properly used, partial correlation analysis can uncover spurious relationships, identify intervening variables, and detect hidden relationships that are present in a data set.

It is true that correlation and semantic equivalence are two different concepts. A person's height and weight may be highly correlated, but it is obvious that height and weight are different in their semantics. However, since semantic conflicts arise from differences in the representation schemes and these differences are uniformly applied to each entity, we expect a high correlation to exist among the values of

semantically equivalent attributes. Partial correlation analysis can at least isolate the attributes that are likely to be related to one another for further analysis.

The limitation of correlation analysis is that it can only reveal linear associations among numerical data. For non-linear associations or categorical data, other measures of correlation are available and will be integrated into the system in the near future.

A snap-shot of the output of relevant attribute analysis is shown in Fig. 4.

#### 4.3. Candidate model selection

As discussed above, a model is required before performing regression. Given a set of relevant attributes, a model specifies the number of attributes that should actually be included and the basic relationship among them.

A number of techniques have been developed to automatically select and rank models from a set of attributes, like *t*-test, step-wise regression. All of these methods are available in the popular statistical packages, like SAS, SPSS, S-PLUS [24]. However, these model selection methods tend to favor those models that involve more attributes, which make them very vulnerable in the real modelling process [27]. We adopt the approach proposed by Raftery [27]. To assess a model, Raftery [27] introduced the Bayesian Information Criterion (BIC), an approximate to Bayes factors used to compare two models based on Bayes's theorem. For different regression functions, the BIC takes different forms. In the case of linear regression, BIC<sub>*k*</sub> of a model *M<sub>k</sub>* can be computed as

$$\text{BIC} = N \log(1 - R_k^2) + p_k \log N \quad (2)$$

where *N* is the number of data points, *R<sub>k</sub><sup>2</sup>* is the value of adjusted *R<sup>2</sup>* for model *M<sub>k</sub>* and *p<sub>k</sub>* is the number of independent attributes. Using the BIC values of models, we can rank the models. The smaller the BIC is, the better the model is to fit the data. One important principle in comparing two nested models *M<sub>k</sub>* and *M<sub>k+1</sub>*, where *k* is the number of independent attributes, is the so-called Occam's Window. The essential idea is that, if *M<sub>k</sub>* is better than *M<sub>k+1</sub>*, model *M<sub>k+1</sub>* is removed. However, if model *M<sub>k+1</sub>* is better, it requires a certain "difference" between two models to cause *M<sub>k</sub>* to be removed. That is, there is an area, the Occam's Window, where *M<sub>k+1</sub>* is better than *M<sub>k</sub>* but not better enough to cause *M<sub>k</sub>* being removed. The size of Occam's Window can be adjusted. Smaller Occam's Window size will cause more models to be removed. An illustrative example on synthetic data in Section 5 is provided to show its impact on the model selection process. Please refer to Ref. [27] for more detailed discussion on BIC criterion and its computation.

#### 4.4. Conversion function generation

With a given set of relevant attributes,  $\{A_1, \dots, A_k, B_1, \dots, B_l\}$  with *A<sub>1</sub>* and *B<sub>1</sub>* being semantically equivalent attributes, the conversion function to be discovered

$$A_1 = f(B_1, \dots, B_l, A_2, \dots, A_k) \quad (3)$$

should hold for all tuples in the training data set. The conversion function should also hold for other unseen data from the same sources. This problem is essentially the same as the problem of learning quantitative laws from the given data set.

Various approaches have been proposed in the literature. They can be classified into two categories: parametric approach and non-parametric approach [12]. Even though non-parametric approach has the merit that it does not require many statistical assumptions, it suffers from the fact that it is very hard to explain the relationship it discovered. Moreover, chances exist it can approximate spurious relationship even though there are no explicit function relationships existing at all [15]. Since the purpose of our data mining process is to discover the underlying conversion function that can be used for later data integration, only functions of explicit forms like linear or linear with several orders of interaction, etc., should be of our primary concern. So in the later implementation of our system, we will concentrate on using only the parametric methods, especially, parametric regression methods for the conversion function discovery purpose.

*Regression analysis* is one of the most well-studied method for discovering such quantitative relationships among variables from a data set [1,7]. If we view the database attributes as variables, the conversion functions we are looking for among the attributes are nothing more than regression functions. For example, the equation *stk\_rpt.price* = *stock.close*\**rate* can be viewed as a regression function, where *stk\_rpt.price* is the response (dependent) variable, *rate* and *stock.close* are predictor (independent) variables. In general, there are two steps in using regression analysis. The first step is to define a model, which is a prototype of the function to be discovered. For example, a linear model of *p* independent variables can be expressed as follows:

$$Y = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \dots + \beta_p X_p + \mu \quad (4)$$

where  $\beta_0, \beta_1, \dots, \beta_p$  are model parameters, or *regression coefficients*, and  $\mu$  is an unknown random variable that measures the departure of *Y* from exact dependence on the *p* predictor variables.

For a given model, the model parameters  $\beta_1, \dots, \beta_p$  for the regression model can be easily estimated using a variety of statistical techniques. The essence of these

techniques is to fit all data points to the regression function by determining the coefficients. The goodness of the discovered function with respect to the given data is usually measured by the *coefficient of determination*, or  $R^2$ . The value  $R^2$  ranges between 0 and 1. A value of 1 implies a perfect fit of all data points with the function. We argued earlier that the conversion function for context-dependent conflicts is uniformly applied to all tuples (data points) in a data set, hence, all the data points should fit to the discovered function ( $R^2=1$ ). However, this will be true only if there are no noises or errors, or *outliers*, in the data. In real world, this condition can hardly be true. One common type of noise is caused by rounding of numerical numbers. This leads to two key issues in conversion function generation using regression:

- A robust regression technique that can produce the functions even when noises and errors exist in data; and
- The acceptance criteria of the regression functions generated.

We will address the first issue in this subsection. The second issue will be discussed in the next subsection.

The most frequently used regression method is least squares (LS) regression. However, it is only efficient and effective for the case where there are no outliers in the data set. To deal with outliers, various robust/resistant regression methods were proposed [31], including least median squares (LMS) regression [29], least quartile squares (LQS) regression and least trimmed squares (LTS) regression [30]. We implemented the LTS regression method. LTS regression, unlike the traditional LS regression that tries to minimize the sum of squared residuals of all data points, minimizes a subset of the ordered squared residuals, and leaves out those large squared residuals, thereby allowing the fit to stay away from those outliers and leverage points. Compared with other robust techniques, like LMS, LTS also has a very high breakdown point: 50%. That is, it can still give a good estimation of model parameters even half of the data points are contaminated. More importantly, a faster algorithm exists to estimate the parameters [6].

#### 4.5. Conversion function selection

As mentioned in the earlier subsection, we need to define a criteria that can be used to select the best conversion function. To address this issue, we defined a new measure, *support*, to evaluate the goodness of a discovered regression function (conversion function) based on recent work of Hoeting et al. [16]. A function generated from the regression analysis is accepted as a conversion function only if its support is greater than a user specified threshold  $\gamma$ . The *support* of a regression function is defined as

$$\text{support} = \frac{\text{Count}\{y_i' \mid (\text{round}(|y_i - y_i'|)/\text{scale}) \leq \varepsilon\}}{N} \quad (5)$$

where  $i \in (1, N)$ ,  $y_i$  is the actual value of the dependent attributes for the  $i$ th point,  $y_i'$  is the predicted value of the dependent value using the function for the  $i$ th point,  $\varepsilon$  is a user-defined parameter that represents the required prediction accuracy,  $N$  is the number of the total points in the data set and *Count* is a function that returns the number of data points satisfying the conditions. *scale* in Eq. (5) is a robust estimate of the residuals from the regression function, defined as follows:

$$\text{scale} = 1.4826 * \text{median}(\text{abs}((Y - Y') - \text{median}(Y - Y'))) \quad (6)$$

where *median()* is a function that returns the median value of its vector.  $Y$  and  $Y'$  are the vectors of the actual value and the predicted value using the regression function for the dependent attribute, respectively.

The *support* indicates the percentage of data points whose residuals are within a certain range of the scale estimate,  $\varepsilon * \text{scale}$ . In other words, those points with residuals greater than  $\varepsilon * \text{scale}$  are identified as outliers. To determine whether a regression function should be accepted, user specifies a minimum support,  $\gamma$ . The system only generates those functions with *support*  $>$   $\gamma$ . Two parameters,  $\varepsilon$  and  $\gamma$  have different meanings.  $\varepsilon$  *in fact specifies the requirement of the conversion precision required* in our case. Depending on whether the detected outliers are genuine outliers or data points still with acceptable accuracy, the value of  $\varepsilon$  and  $\gamma$  can be adjusted. The default values for  $\gamma$  and  $\varepsilon$  are 0.80 and 4, respectively.

#### 4.6. Conversion rule formation

We have briefly discussed the major techniques currently implemented in **DIRECT**. The last step, conversion rule formation, is merely a syntactic transformation and the details are omitted here. A snap-shot of the conversion rule formation is shown in Fig. 5. If there are any potential outliers having been identified, they will also be reported by this module accordingly.

#### 4.7. Training data reorganization

It is possible that no satisfactory conversion function is discovered with a given relevant attribute set because no such conversion function exists. However, there is another possibility from our observations: the conflict cannot be reconciled using a single function,

but is reconcilable using a suitable collection of different functions. To accommodate for this possibility, our approach includes a training data reorganization module. The training data set is reorganized whenever a single suitable conversion function cannot be found. The reorganization process usually partitions the data into a number of partitions. The mining process is then applied in attempting to identify the appropriate functions for each of the partition taken one at a time. This partitioning can be done in a number of different ways by using simple heuristics or complex clustering techniques, e.g., partition based on the categorical attributes, or partition based on the distribution of the dependent attributes, or even more complex, using the clustering algorithm to cluster the data set into different groups [18]. The partition should be done by the system in an automatic way, or can be offered to users

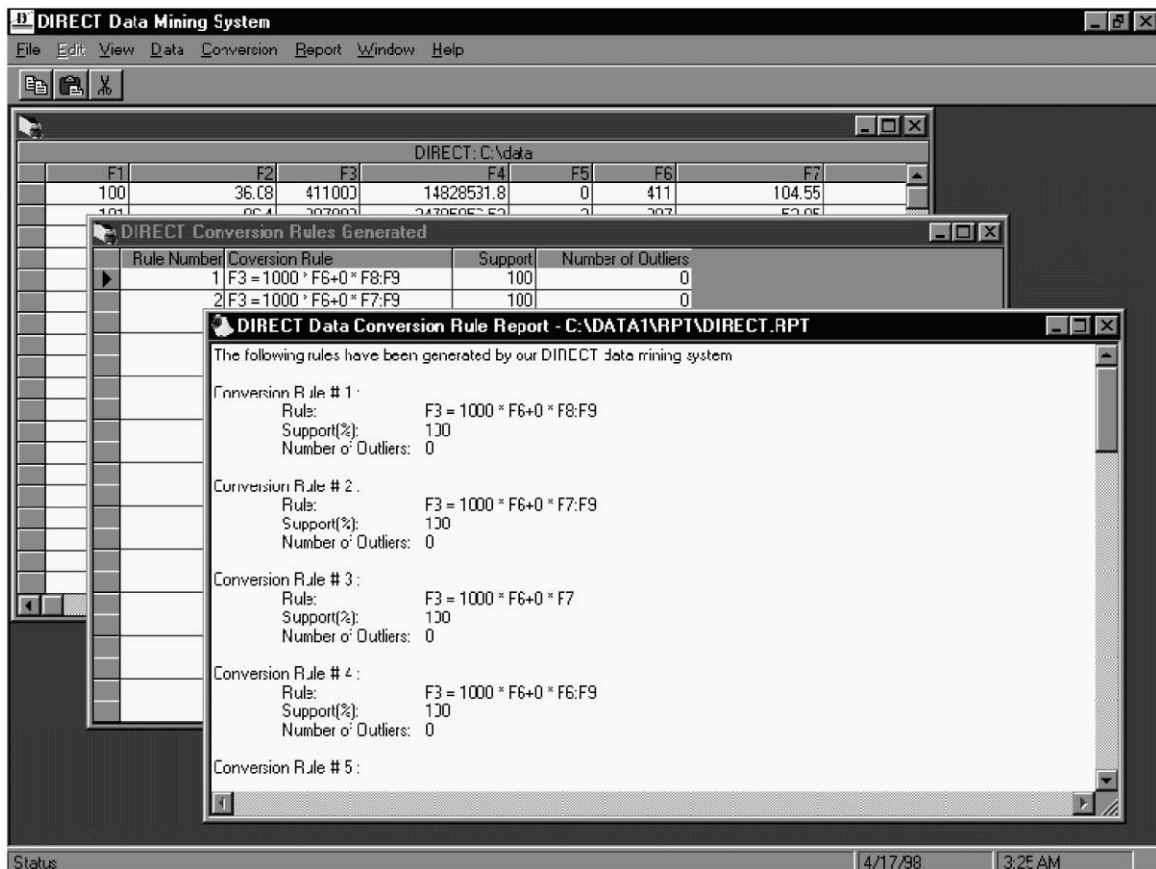


Fig. 5. Conversion rule generation result.

as options to allow them to select. Currently, we have implemented a simple heuristic that partitions the data set based on categorical attributes present in the data set. The rationale is, if multiple functions exist for different partitions of data, data records in the same partition must have some common property, which may be reflected by values assumed by some attributes within the data being investigated.

## 5. Experiments

To validate the efficacy of our proposed approach, two experiments were run using different data sets. The first data set is a synthetic data set which contains no outliers, while the second may contain outliers that we do not know a priori. Both of these two experiments are done fully automatically using our **DIRECT** system without any user intervention. Some of the intermediary results are also presented for illustration purpose.

### 5.1. An illustrative example using synthetic data

In this subsection, we use a synthetic data set as an example to illustrate the workings of the prototype system. We use the example schema *stock* and *stk\_rpt* mentioned at the beginning of the paper to generate the training data set.

*stock* (**scode**, currency, volume, high, low, close);  
*stk\_rpt* (**scode**, price, volume, value);

Table 2  
Example relations

Attribute no.	Relation	Name	Value range
A1	<i>stock</i> , <i>stk_rpt</i>	scode	[1, 500]
A2	<i>stk_rpt</i>	price	$stock.close * exchange\_rate$ [ <i>stock.currency</i> ]
A3	<i>stk_rpt</i>	volume	$stock.volume * 1000$
A4	<i>stk_rpt</i>	value	$stk\_rpt.price * stk\_rpt.volume$
A5	<i>stock</i>	currency	1, 2, 3, 4, 5, random
A6	<i>stock</i>	volume	20–500, uniform distribution
A7	<i>stock</i>	high	[ <i>stock.close</i> , $1.2 * stock.close$ ]
A8	<i>stock</i>	low	[ $0.85 * stock.close$ , <i>stock.close</i> ]
A9	<i>stock</i>	close	[0.50, 100], uniform distribution

Table 3  
Correlation coefficients from the zero-order correlation analysis

	A6	A7	A8	A9
A2	0.0291	0.3933	0.4006	0.4050
A3	1.0000	−0.0403	−0.0438	−0.0434
A4	0.3691	0.2946	0.2994	0.3051

Tuples in relation *stk\_rpt* are in fact derivable from relation *stock*. This is used to simulate the data integration process: the broker receives stock information and integrates it into his/her own stock report. For this experiment, we created a data set of 6000 tuples. The domains of the attributes are listed in Table 2.

#### 5.1.1. Relevant attribute analysis

The zero-order correlation analysis was first applied to the training data. The results are shown in Table 3.

If the correlation efficient between two attributes is greater than the threshold, which was set to 0.1 in the experiment, they were considered relevant. Therefore, three relevant attribute sets are formed as follows:

Set	Attribute	Correlated attributes
1	A3	A6
2	A4	A6, A7, A8, A9
3	A2	A7, A8, A9

Partial correlation analysis was conducted by controlling A6, A7, A8 and A9 in turn to see whether there are more attributes that should be included in the obtained relevant attribute sets. In this example, there were no such attributes; and the relevant attribute sets remained the same.

#### 5.1.2. Candidate model selection and conversion function generation

Each of the relevant attribute sets was used to generate conversion functions in turn. For convenience, we explain the two processes together.

Set 1: {A3, A6}

Set 1 contains only two variables. There is only one possible model ( $A3 = A6$ ). Only the following function was obtained with 100% support:

$$A3 = 1000 * A6. \quad (7)$$

Set 2: {A4, A6, A7, A8, A9}

Table 4  
Results of model selection for attribute Set 2

Window size	Model selected
20	$A4 = A6 * A9$
30	$A4 = A6 * A9$
40	$A4 = A6 * A9$ ; $A4 = A7 + A6 * A9$ ; $A4 = A8 + A6 * A9$ ; $A4 = A9 + A6 * A9$ ;
60	$A4 = A6 * A9$ ; $A4 = A7 + A6 * A9$ ; $A4 = A8 + A6 * A9$ ; $A4 = A9 + A6 * A9$ ; $A4 = A6 * A9 + A7 * A8$ ; $A4 = A6 * A9 + A8 * A9$

$A4$  is correlated to four other attributes. Without any prior knowledge, and considering only linear model with first-order and second-order terms (an acceptable practice for most financial and business data), the initial model includes all the attributes.

$$A4 = A6 + A7 + A8 + A9 + A6 * A7 + A6 * A8 + A6 * A9 + A7 * A8 + A7 * A9 + A8 * A9 \quad (8)$$

Table 4 lists the output of the model selection module for different Occam's Window sizes. It can be seen that with larger window size, the number of candidate models selected increases and more attributes were included in the model. When the window sizes are 20 and 30, only one model with attributes  $A6$  and  $A9$  was selected. When the window size increased to 40, four models were selected and attributes  $A8$  and  $A7$  appeared in some of the models.

In the conversion function generation process, the selected model was used to generate conversion functions. For the models in Table 4, the system in fact did not report any functions. By further examin-

Table 5  
Functions discovered for selected models

Model	Function generated	Support (%)
$A4 = A6 * A9$	$A4 = 708.04 * A6 * A9$	81.73
$A4 = A7 + A6 * A9$	$A4 = 9980 * A7$ $+ 678.05 * A6 * A9$	81.22
$A4 = A8 + A6 * A9$	$A4 = 10875.13 * A8$ $+ 685.14 * A6 * A9$	81.08
$A4 = A9 + A6 * A9$	$A4 = 9970.88 * A9$ $+ 677.84 * A6 * A9$	81.31
$A4 = A6 * A9$ $+ A7 * A8$	$A4 = 685.90 * A6 * A9$ $+ 102.15 * A7 * A8$	81.54
$A4 = A6 * A9$ $+ A8 * A9$	$A4 = 681.94 * A6 * A9$ $+ 127.34 * A8 * A9$	81.58

Table 6  
Models selected for partitioned data

Window size	Model selected
20	$A4 = A6 * A9$
40	$A4 = A6 * A9$
60	$A4 = A6 * A9$ ; $A4 = A6 + A6 * A9$ ; $A4 = A7 + A6 * A9$ ; $A4 = A8 + A6 * A9$ ; $A4 = A9 + A6 * A9$ ; $A4 = A6 * A7 + A6 * A9$ ; $A4 = A6 * A8 + A6 * A9$ ; $A4 = A6 * A9 + A7 * A8$ ; $A4 = A6 * A9 + A7 * A9$ ; $A4 = A6 * A9 + A8 * A9$

ing the process, we found that all the models resulted in functions with support lower than the specified threshold (Table 5).

As mentioned earlier, one possibility that a selected model does not generate any conversion function is that there exist multiple functions for the data set. To discover such multiple functions, the data set should be partitioned. In our implementation, a simple heuristic is used, that is, to partition the data using categorical attributes in the data set. After the data was partitioned based on a categorical attribute  $A5$ , the model selected is shown in Table 6.

The conversion function generation module estimates the coefficients for each of the models selected for each partition. There is only one conversion function reported for each partition, since all the coefficients for the terms other than  $A6 * A9$  were zero. The results are summarized in Table 7.

Set 3:  $\{A2, A7, A8, A9\}$

The process for relevant attribute Set 3 is similar to what was described for Set 2. The initial model used is

$$A2 = A7 + A8 + A9 + A7 * A8 + A7 * A9 + A8 * A9 \quad (9)$$

and the model selection module selected 10 models without producing functions with enough support. Using the data sets partitioned using  $A5$ , the conversion functions listed in Table 8 were obtained.

Table 7  
The conversion functions generated for Set 2

$A5$	Conversion function	Support (%)
0	$A4 = 400 * A6 * A9$	100
1	$A4 = 700 * A6 * A9$	100
2	$A4 = 1000 * A6 * A9$	100
3	$A4 = 1800 * A6 * A9$	100
4	$A4 = 5800 * A6 * A9$	100

Table 8  
The conversion functions generated for Set 3

A5	Conversion function	Support (%)
0	$A2 = 0.4 * A9$	100
1	$A2 = 0.7 * A9$	100
2	$A2 = 1.0 * A9$	100
3	$A2 = 1.8 * A9$	100
4	$A2 = 5.8 * A9$	100

### 5.1.3. Conversion function selection and data conversion rule formation

Since there is only one set of candidate functions obtained for each set of relevant attribute set with 100% support. The functions generated were selected to form the data conversion rules. By some syntactic transformation, we can obtain the following data conversion rules for our example:

$stk\_rpt(stock, price, rpt\_volume, value) \leftarrow stock$   
 $(stock, currency, stk\_volume, high, low, close,$   
 $price = rate * close, rpt\_volume = 1000 * stk\_vol-$   
 $ume, value = rate * 1000 * stk\_volume * close, ex-$   
 $change\_rate(currency, rate).$   
 $exchange\_rate(0, 0.4).$   
 $exchange\_rate(1, 0.7).$   
 $exchange\_rate(2, 1.0).$   
 $exchange\_rate(3, 1.8).$   
 $exchange\_rate(4, 5.8).$

It is obvious that the discovered rule can be used to integrate data from *stock* to *stk\_rpt*.

Table 9  
Attributes of sales data sample

Attribute	Data Source	Name	Description
A1	T(ransaction)	<i>month</i>	financial period
A2	T(ransaction)	<i>invoice_no</i>	the unique invoice number
A3	T(ransaction)	<i>amount</i>	total amount invoiced in the original currency
A4	T(ransaction)	<i>sales_type</i>	type of sales that determines tax
A5	T(ransaction)	<i>GST_rate</i>	goods and service tax rate
A6	T(ransaction)	<i>currency</i>	currency code
A7	T(ransaction)	<i>exchange_rate</i>	exchange rate for the invoice
A8	T(ransaction)	<i>GST_amount</i>	goods and service tax in the original currency
A9	C(ost/profit)	<i>amount</i>	amount for the sales account, i.e., before-tax amount
A10	A(ccounts)	<i>amount</i>	amount of the invoice in the local currency

### 5.2. An example of mining process using a real-world data set

In this section, we describe an example mining process using **DIRECT** in which a set of real-world data collected from a small trading company is used.

To allow us to focus on mining data value conversion rules, a program was written to extract data from the database into a single data sets with the 10 attributes shown in Table 9. Attributes A1 to A8 are from a transaction database, *T*, which records the details of each invoice billed, hence, all the amounts involved are in the original currency. Attribute A9 is from a system for cost/profit analysis, *C*, which captures the sales figure in local currency exclusive of tax. Attribute A10 is from the accounting department, *A*, where all the monetary figures are in the local currency. Therefore, although attributes A3, A9 and A10 have the same semantics, i.e., all refer to the amount billed in an invoice, their values are different. In the experiment, we tried to discover the conversion functions among those conflicting attributes.

From the context of the business, the following relationship among the attributes should exist:

$$A.amount = T.amount * T.exchange\_rate. \quad (10)$$

$$C.amount = (T.amount - T.GST\_amount) * T.exchange\_rate \quad (11)$$

$$C.amount = A.amount - T.exchange\_rate * T.GST\_amount. \quad (12)$$

The goods and service tax (GST), is computed based on the amount charged for the sales of goods or

services. As  $C.amount$  is in the local currency and all transaction data are in the original currency, we have the following relationship:

$$T.GST\_amount = C.amount / T.exchange\_rate * T.GST\_rate \quad (13)$$

where GST rate depends on the nature of business and clients. For example, exports are exempted from GST (tax rate is 0%) and domestic sales are taxed in a fixed rate of 3% in our case.

The data set collected contains 7898 tuples corresponding to the invoices issued during the first 6 months of a financial year. The discovering process and the results are summarized in the following subsections.

### 5.3. Transaction data versus analysis data

**Relevant attribute analysis:** As our current system only works for numeric data, the categorical attributes are not included in analysis. That is, among eight attributes from data source  $T$ , only attributes  $A3$ ,  $A5$ ,  $A7$  and  $A8$  are taken into the relevant attribute analysis. The results of the partial correlation analysis are shown in Table 10.

Note that the correlation coefficients between  $A10$  and  $A5$ ,  $A10$  and  $A7$  are rather small in the zero-order test. However, this alone is not sufficient to conclude that  $A10$  is not related to  $A5$  and  $A7$ . By the PCA test with controlling  $A3$ , the correlation between  $A10$  and  $A7$  became rather obvious and the correlation coefficient between  $A10$  and  $A5$  also increased. With one more PCA test that controls  $A8$ , the correlation between  $A10$  and  $A5$  became more obvious. Thus, all the four attributes are included into the quantitative relationship analysis.

Table 10  
Partial correlation analysis between  $A10$  and  $\{A3, A5, A7, A8\}$

	Correlation coefficient of $A10$ with		
	Zero-order	Controlling $A3$	Controlling $A8$
$A3$	0.91778708	–	0.8727613
$A5$	0.02049382	0.1453855	–0.2968708
$A7$	0.02492715	0.4562510	–0.006394451
$A8$	0.71552943	0.5122901	–

Table 11  
Partial correlation analysis between  $A9$  and  $\{A3, A5, A7, A8\}$

	Correlation coefficient of $A9$ with		
	Zero-order	Controlling $A3$	Controlling $A8$
$A3$	0.91953641	–	0.8739174
$A5$	0.01350388	0.1292704	–0.2976056
$A7$	0.02355110	0.4581783	–0.007597317
$A8$	0.70452050	0.4791232	–

**Candidate model selection and conversion function generation:** The candidate model selection and conversion function generation module produced the following function with 100% support.

$$A10 = A3 * A7 \quad (14)$$

### 5.4. Transaction data versus accounting data

Similarly, tests were conducted among attributes  $A9$  and  $A3$ ,  $A5$ ,  $A7$  and  $A8$  to discover the relationship between  $A.amount$  and  $T.amount$ . Table 11 listed the correlation coefficients obtained from the partial correlation analysis.

Again, the following function was obtained with 100% support.

$$A9 = A3 * A7 - A7 * A8 \quad (15)$$

### 5.5. Tests with partitioned data

In the above results, we did not find any relationship involving attribute  $A5$  which shows high correlation with both  $A9$  and  $A10$ . Since the analysis on the whole data set did not generate any functions, we partitioned the data according to categorical attributes. One categorical attribute,  $A4$ , is named as  $sales\_type$ , and is used to partition the data set first. There are two values for  $sales\_type$ : 1, 2. The results obtained from the partitioned data are listed in Table 12.

The functions with higher support in each group, i.e., functions (1) and (3), are selected as the conversion functions. We can see that they in fact represent the same function. Using the attribute numbers instead of the name, we can rewrite Eq. (12) as

$$A9 = A3 * A7 - A8 * A7$$

Table 12  
Results after partitioning data using *sales\_type*

A4	No.	Conversion functions	Support	Number of outliers
1	1	$A9 = 0.9708738 * A3 * A7$	99.94	2
	2	$A9 = 33.33333 * A7 * A8$	99.36	21
2	3	$A9 = A3 * A7$	100.00	0

The tax rate,  $A5$ , is stored as the percentage, Eq. (13) can be rewritten as

$$A8 = \frac{A9}{A7} * 0.01 * A5$$

Thus, we have

$$\begin{aligned} A9 &= A3 * A7 - \frac{A9}{A7} * 0.01 * A5 * A7 \\ &= A3 * A7 - 0.01 * A9 * A5 \end{aligned}$$

Therefore,

$$A9 = \frac{A3 * A7}{1 + 0.01 * A5} \quad (16)$$

From the data, we have

$$A5 = \begin{cases} 3 & \text{for } A4 = 1 \\ 0 & \text{for } A4 = 2 \end{cases}$$

Substituting this into Eq. (16), we have

$$A9 = \begin{cases} \frac{A3 * A7}{1.03} = 0.9708738 * A3 * A7 & \text{for } A4 = 1 \\ A3 * A7 & \text{for } A4 = 2 \end{cases}$$

which are the functions (1) and (3). In other words, although the two functions have different forms, they do represent the original relationship among the data values specified by Eq. (16).

What seems unnatural is that the support of function (1) is not 100% as it should be. We found the following two tuples listed in Table 13 that were

identified as the outliers. They are indeed the tuples with errors contained in the original data: both tuples have incorrect value of GST amount.<sup>3</sup>

## 5.6. Discussions

By going through the mining process of two sample data sets using DIRECT system, we demonstrated that we could not only successfully discover the underlying conversion functions and resolve the semantic conflicts, but also address the data quality issue by successful identification of those erroneous records (outliers) existing in the databases.

There are several points we want to emphasize here on the possible usage of, or extensions to our proposed framework.

- Robust regression is used as the primary methodology to uncover the true relationships among those semantically related attributes. By “robust”, we mean that it does NOT suffer from the situation where the error terms in the regression model do not follow a normal distribution as required in traditional least-square linear regression analysis. When there are many outliers in the data set, traditional LS regression requires users to check the normality assumptions and identify the outliers by going through a series of diagnostics steps before running the regression analysis. But even if they are following these procedures, there are still some potential outliers in the data set that cannot be uncovered a priori (refer to pp. 5–8 in Ref. [29] for detailed examples). This is no longer the case in LTS robust regression since LTS can automatically detect and neglect those outliers in the regression analysis. Besides, it has been shown that LTS robust regression is a type of non-parametric regression method which does not require very rigid linear independence and normality assumptions [29].

- At the moment, **DIRECT** supports the mining of conversion functions of a linear form with multiple order interaction terms, which is very common these days because a lot of current data warehouses and heterogeneous database systems store a large amount

<sup>3</sup> We realized that the system used in the company allows data entry clerk to modify GST values calculated and prompted by the system. It is obvious that for these two invoices, incorrect tax values were entered.

Table 13

Two erroneous tuple discovered

Month	Invoice number	Amount	GST type	GST rate	Currency	Exchange rate	GST	C.amount	A.amount
3	8237	45.50	1	3	1	1.00	0.00	45.50	45.50
6	12,991	311.03	1	3	1	1.00	6.53	304.50	311.03

of business and financial transaction data. The purpose of the mining process is to mine the “original” or “meaningful” functions involving numerical attributes in the databases as we mentioned in Section 2, and to resolve the semantic conflicts among these attributes. There are many other data mining techniques proposed in the literature [14], e.g., statistical approximation using non-parametric methods like neural networks, kernel methods, etc., that could be used for this purpose. But since their results are harder to interpret and encode when we perform the data integration task, they are not implemented in our system. These techniques, however, are very useful for situations when we do not care very much about the final functional forms or when the functional form is of a highly complex and nonlinear form and is very hard to detect using conventional regression methods. The discussion of these techniques to these situations is beyond the scope of this paper.

- Data mining is an interactive process. Both of the two examples shown above are retrospective studies to validate the efficacy of our approach. The interestingness of the conversion rules we learned can be easily interpreted and verified. In real situation, the data set could be very large. In this case, the entire data set could be sampled and split into two sub data sets: training data and test data, following machine learning experimental design approach [25]. The training data is used to discover the possible candidate models which are presented to the user for further selection. The test data is used for final verification and selection of those candidate models. These features will be incorporated into future releases of our **DIRECT** system. Currently, the rules learned by **DIRECT** after the retrospective study are sent to the DBAs for final verification and selection to help them make better decisions.

- It is known that that there are also cases where the relationship among attributes are not linear in functional form. We are currently working on extending

our system **DIRECT** to make nonlinear conversion function discovery modules using domain knowledge possible. Similar to Ref. [26], it can support the domain knowledge by knowledge representation using AI techniques. It can also support the novel function discovery using more advanced inductive discovery technique like Genetic Programming [20]. Because of the uniqueness of the nonlinear function discovery due to its very high complexity and computational intensity, the results will be reported in another paper.

## 6. Conclusions

In this paper, we addressed the problem of discovering and resolving data value conflicts for data integration. We first proposed a simple classification scheme for data value conflicts based on how they can be reconciled. We argue that those conflicts caused by genuine semantic heterogeneity can be reconciled systematically using data value conversion rules. A general approach for discovering data conversion rules from data automatically was proposed. A prototype system **DIRECT** was developed and tested using both synthetic and real-world data set. The result is very promising. It cannot only be used to discover the underlying data value conversion rules interactively or automatically, but also can be used as a tool to improve the quality of the data because of its capability to identify outliers presented in the data set.

The various modules implemented in our system are based on the approaches developed in the statistics literature. Some of the individual modules are also implemented in the commercial statistical packages, like SPSS, SAS, S-PLUS, etc. Our system, however, is different from these systems in the following ways.

- **DIRECT** is used specifically for mining the data conversion rules. The modules are fully optimized and

integrated into one system that can be run both automatically and interactively. While most of the statistical packages cannot.

- Our system does not require the user to have any statistical background to understand and use our system. It has a very nice and intuitive graphical user interface (GUI). All the procedures can be run by simply clicking one button. Almost all of the current statistical packages require that you have at least basic statistics knowledge to be able to operate their systems. And you need to learn on how to program using their own programming or scripting languages.

- The model selection module using BIC measure, as well as robust regression module and outlier detection module using genetic algorithm, is not available in most of the statistical packages.

- Our system supports the SQL syntax that can be used to extract the data directly from different databases. The final conversion rules can also be stored directly as meta data information into the databases.

It is not our intention to say that the methods we adopted in our implementation in each of the discovery modules are the best ones. New learning techniques keep coming up in statistics, mathematics, pattern recognition, and machine learning. However, the general discovery framework proposed in this paper, should be applicable to all the other function discovery processes. Future developments of **DIRECT** are the following.

- Design and implement a module that can discover nonlinear conversion functions as well as automatic power/log transformations of data.

- Design and implement a module that can discover conversion rules involving aggregate functions (like, Sum, Avg., Max., etc.) that are often presented in business and financial data.

- Currently, data repartition module only supports heuristic based on categorical attributes, i.e., partition the data using single categorical attribute or their combinations. More advanced techniques, like data classification using decision tree, neural networks, or data clustering as discussed in Ref. [25], could be extended and implemented to support this non-trivial task.

- Data value conversion rules are now defined among data sources. It has the drawback that a large

number of such rules have to be defined if we are to integrate data from a large number of sources. We are exploring some other useful schemes to improve the scalability of our system.

## Acknowledgements

The authors wish to thank the editor-in-chief, the associate editor and anonymous reviewers for their helpful and constructive suggestions for improvement of this paper. The second author's work is partially supported by a grant from the Research Grant Council of the Hong Kong Special Administrative Region, China (HKUST6092/99E), and a grant from the National 973 project of China (No. G1998030414).

## References

- [1] A. Afifi, V. Clark, *Computer-Aided Multivariate Analysis*, 3rd edn., Chapman & Hall, New York, 1996.
- [2] S. Agarwal, A.M. Keller, G. Wiederhold, K. Saraswat, Flexible relation: an approach for integrating data from multiple, possibly inconsistent databases, Proc. IEEE Intl. Conf. on Data Engineering, Taipei, Taiwan (March 1995).
- [3] R. Brachman, T. Khabaza, W. Kloesgen, G. Piatetsky-Shapiro, E. Simoudis, Mining business databases, Communications of ACM 39 (11) (1996) 42–48.
- [4] S. Bressan, K. Fynn, C. Goh, M. Jakobisiak, K. Hussein, H. Kon, T. Lee, S. Madnick, T. Pena, J. Qu, A. Shum, M. Siegel, The CContext INterchange mediator prototype, Proc. ACM SIGMOD/PODS Joint Conference, Tucson, AZ (1997).
- [5] S. Bressan, C. Goh, S. Madnick, M. Siegel, A procedure for context mediation of queries to disparate sources, Proceedings of the International Logic Programming Symposium (October 12–17) 1997.
- [6] P. Burns, A genetic algorithm for robust regression estimation, StatScience Technical Note (1992).
- [7] S. Chatterjee, B. Price, *Regression Analysis by Example*, 2nd edn., Wiley, New York, 1991.
- [8] U. Dayal, Processing queries with over generalized hierarchies in a multidatabase system, Proceedings of VLDB Conference, 1983, pp. 342–353.
- [9] L.G. Demichiel, Resolving database incompatibility: an approach to performing relational operations over mismatched domains, IEEE Trans. on Knowledge and Data Engineering 1 (4) (1989) 485–493.
- [10] W. Fan, H. Lu, S. Madnick, D. Cheung, Discovering and reconciling value conflicts for numerical data integration, Information Systems 26 (8) (2001) 635–656.
- [11] U. Fayyad, G. Piatetsky-Shapiro, P. Smyth, From data mining to knowledge discovery: an overview, in: U. Fayyad, G. Piatetsky-Shapiro, P. Smyth, R. Uthurusamy (Eds.), *Advances in*

- Knowledge Discovery and Data Mining, AAAI/MIT Press, Cambridge, MA, 1996, pp. 1–36.
- [12] J. Friedman, Multivariate adaptive regression splines, *The Annals of Statistics* 19 (1) (1991) 1–141.
- [13] C. Goh, S. Bressan, S. Madnick, M. Siegel, Context interchange: representing and reasoning about data semantics in heterogeneous systems, Sloan School Working Paper #3928, Sloan School of Management, MIT, 50 Memorial Drive, Cambridge, MA 02139 (Oct. 1996).
- [14] J. Han, M. Kamber, *Data Mining: Concepts and Techniques*, Morgan Kaufmann Publishers, San Francisco, 2000.
- [15] T. Hastie, R. Tibshirani, *Nonparametric regression and classification, From Statistics to Neural Networks: Theory and Pattern Recognition Applications*, Springer, Berlin-New York, 1994, pp. 70–82.
- [16] J. Hoeting, A. Raftery, D. Madigan, A method for simultaneous variable selection and outlier identification, Technical Report 9502, Department of Statistics, Colorado State University (1995).
- [17] S. Hui, G. Jha, Data mining for customer service support, *Information and Management* 38 (1) (2000) 1–14.
- [18] A.K. Jain, R.C. Dubes, *Algorithms for Clustering Data*, Prentice-Hall, New Jersey, 1988.
- [19] V. Kashyap, A. Sheth, Schematic and semantic similarities between database objects: a context-based approach, *VLDB Journal* 5 (4) (October 1996) 276–304.
- [20] J.R. Koza, *Genetic Programming: On the Programming of Computers by Means of Natural Selection*, MIT Press, Cambridge, MA, USA, 1992.
- [21] E.-P. Lim, R. Chiang, The integration of relationship instances from heterogeneous databases, *Decision Support Systems* 29 (2000) 153–167.
- [22] E.-P. Lim, J. Srivastava, S. Shekhar, An evidential reasoning approach to attribute value conflict resolution in database integration, *IEEE Transactions on Knowledge and Data Engineering* 8 (5) (Oct. 1996) 707–723.
- [23] E.-P. Lim, J. Srivastava, S. Shekhar, J. Richardson, Entity identification problem in database integration, *Proceedings of the 9th IEEE Data Engineering Conference*, 1993, pp. 294–301.
- [24] A. Miller, *Subset Selection in Regression*, Chapman & Hall, New York, 1990.
- [25] T.M. Mitchell, *Machine Learning*, McGraw-Hill, New York, 1997.
- [26] B. Padmanabhan, A. Tuzhilin, Unexpectedness as a measure of interestingness in knowledge discovery, *Decision Support Systems* 27 (1999) 303–318.
- [27] A. Raftery, Bayesian model selection in social research, *Sociological Methodology*, 1995, pp. 111–196.
- [28] P. Robertson, Integrating legacy systems with modern corporate applications, *Communications of ACM* 40 (5) (1999) 39–46.
- [29] P. Rousseeuw, Least median of squares regression, *Journal of the American Statistical Association* 79 (1984) 871–880.
- [30] P. Rousseeuw, M. Hubert, Recent developments in progress, L1—Statistical Procedures and Related Topics, Institute of Mathematical Statistics Lecture Notes-Monograph Series, vol. 31, Hayward, California, 1997, pp. 201–214.
- [31] P. Rousseeuw, A. Leroy, *Robust Regression and Outlier Detection*, Wiley, New York, 1987.
- [32] P. Scheuermann, E.I. Chong, Role-based query processing in multidatabase systems, *Proceedings of International Conference on Extending Database Technology*, 1994, pp. 95–108.
- [33] P. Scheuermann, W.-S. Li, C. Clifton, Dynamic integration and query processing with ranked role sets, *Proc. First International Conference on Interoperable and Cooperative Systems (CoopIS'96)*, Brussels, Belgium, Jun. 1996, pp. 157–166.
- [34] P. Scheuermann, C. Yu, A. Elmagarmid, H. Garcia-Molina, F. Manola, D. McLeod, A. Rosenthal, M. Templeton, Report on the workshop on heterogeneous database systems, *ACM SIGMOD Record* 4 (19) (Dec. 1990) 23–31, Held at Northwestern University, Evanston, IL, Dec. 11–13, 1989, Sponsored by NSF.
- [35] E. Sciore, M. Siegel, A. Rosenthal, Using semantic values to facilitate interoperability among heterogeneous information systems, *ACM Transactions on Database Systems* 19 (2) (Jun. 1994) 254–290.
- [36] M. Tschansky, N. Pliskin, G. Rabinowitz, A. Porath, Mining relational patterns from multiple relational tables, *Decision Support Systems* 27 (1999) 177–195.
- [37] F.S. Tseng, A.L. Chen, W.-P. Yang, Answering heterogeneous database queries with degrees of uncertainty, *Distributed and Parallel Databases: An International Journal* 1 (3) (1993) 281–302.
- [38] Y. Wang, S. Madnick, The inter-database instance identification problem in integrating autonomous systems, *Proceedings of the Sixth International Conference on Data Engineering*, 1989.

**Weiguo Fan** is currently a Ph.D Candidate at the University of Michigan Business School. He will join Virginia Tech University as an assistant professor in Information System and Computer Science in August, 2002. He received a B.E. in Information Science and Engineering in 1995 from Xi'an Jiaotong University, Xi'an, P.R. China, and a M.Sc degree in Computer Science in 1997 from the National University of Singapore, Singapore. His research interests include data mining, text(web) mining, information retrieval on the WWW, and information integration. He has published various papers in leading information system and database conferences and journals.

**Hongjun Lu** is a full professor at the Hong Kong University of Science and Technology. He received his Ph.D degree in Computer Science from the University of Wisconsin, Madison. His research interests include query processing and optimization, parallel and distributed database systems, and knowledge discovery and data mining. He has published more than 80 papers in various database conferences and journals. He is currently a member of the ACM SIGMOD Advisory Committee and a trustee of the VLDB Endowment. He has served as a programme committee member for many leading database conferences like SIGMOD, VLDB, ICDE and as a reviewer for major database journals and conferences.

**Stuart E. Madnick** is the John Norris Maguire Professor of Information Technology and Leaders for Manufacturing Professor of Management Science at the MIT Sloan School of Management. He is also an affiliate member of the MIT Laboratory for Computer Science and a member of the Executive Committee of the MIT Center for Information Systems Research. His current research interests include connectivity among disparate distributed information systems, database technology, and software project management. He is the author or co-author of over 200 books, articles, or reports on these subjects, including the classic textbook, *Operating Systems* (McGraw-Hill), and the book, *The Dynamics of Software Development* (Prentice-Hall). He has been active in industry, making significant contributions as one of the key designers and developers of projects such as IBM's VM/370 operating system and Lockheed's DIALOG information retrieval system.

**David W. Cheung** received the M.Sc and Ph.D degrees in Computer Science from Simon Fraser University, Canada, in 1985 and 1989, respectively. He also received a BSc degree in mathematics from the Chinese University of Hong Kong. From 1989 to 1993, he was with Bell Northern Research, Canada, where he was a member of the scientific staff. Currently, he is an associate professor of the Department of Computer Science at the University of Hong Kong. His research interests include distributed databases, spatial databases, data mining and data warehousing.