Data Integration Using Data Mining Techniques

Karen C. Davis, Krishnamoorthy Janakiraman, Ali Minai
Electrical & Computer Engineering and Computer Science Dept.
University of Cincinnati, Cincinnati OH 45221-0030
{karen.davis, ali.minai}@uc.edu, krish_jana@hotmail.com

Robert B. Davis
Dept. of Mathematics and Statistics
Miami University, Hamilton OH 45011
davisrb@muohio.edu

Abstract

Database integration provides integrated access to multiple data sources. Database integration has two main activities: schema integration (forming a global view of the data contents available in the sources) and data integration (transforming source data into a uniform format). This paper focuses on automating the aspect of data integration known as entity identification using data mining techniques. Once a global database is formed of all the transformed source data, there may be multiple instances of the same entity, with different values for the global attributes, and no global identifier to simplify the process of entity identification. We implement decision trees and k-NN as classification techniques, and we introduce a preprocessing step to cluster the data using conceptual hierarchies. We conduct a performance study using a small testbed and varying parameters such as training set size and number of unique entities to study processing speed and accuracy tradeoffs. We find that clustering is a promising technique for improving processing speed, and that decision trees generally have faster processing time but lower accuracy than k-NN in some cases.

Introduction

Organizations are increasingly experiencing the necessity and benefits of integrated access to multiple data sources. Database integration has two aspects: schema integration and data integration. *Schema integration* arrives at a common schema representing the elements of the source schemas. *Data integration* detects and merges multiple instances of the same real world entities from different databases. *Entity identification* is necessary when there is no common means of identification such as primary keys, and it is usually solved manually. This paper focuses on solving the entity identification problem in an automated way using data mining techniques. We use automated learning techniques to identify characteristics or patterns found in entities and apply this knowledge to detect multiple instances of the same entity.

Ganesh et al. [GSR96] propose an automated framework for solving the entity identification problem. We extend the framework (shown in shaded boxes) in Figure 1 and conduct performance studies to determine the accuracy and processing time resulting from the extensions. The cylinders in Figure 1 represent data, the rectangles represent processes, and the ellipses represent the intermediate results. We assume that a global schema has been created by a schema integration phase, although the data has not yet been merged. The global database may contain different instances of the same real world entity from different sources and without a common unique identifier. Merging instances (database integration) requires entity identification, and that is the focus of our work. In the framework, the module called *Preprocessing* clusters the data from the global database prior to performing entity identification. After preprocessing, training sets are formed as subsets of the global database that are used in the learning module to form integration rules (i.e., to perform entity identification.) The original framework [GSR96] uses a decision tree algorithm as the classification technique in the learning module, and we implement k-NN [FBF77] as an additional technique.

To study the performance of the proposed modifications, we use a small database and vary different parameters such as training set size and number of unique entities in our experiments. Our experiments address the following questions:

1. What is the impact of our preprocessing algorithm on a decision tree implementation of entity identification?
2. What is the impact of using k-NN as the classification technique?

We investigate comparative accuracy and processing speed.

We describe the preprocessing technique next, followed by our experimental setup, and then we offer discussion of the results. Conclusions and related work comprise the final section.

Preprocessing

Our preprocessing module is based on a generalization process using conceptual hierarchies [HCC92]. A *conceptual hierarchy* on an attribute represents a taxonomy of concepts that are successively more general as the hierarchy is ascended, and the leaf level represents values of the attribute domain. Conceptual hierarchies are either provided by a domain expert or can be derived

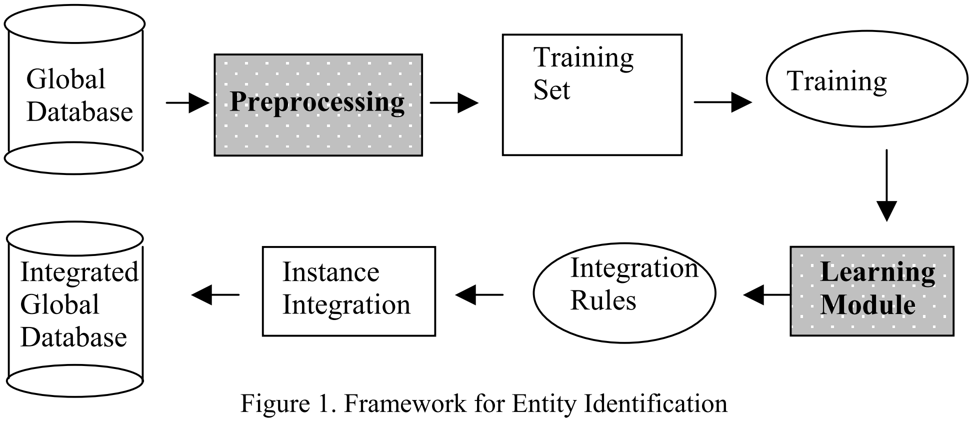

Figure 1. Framework for Entity Identification

automatically for numeric domains [HF94]. An example conceptual hierarchy may generalize GPA values as "low" for values between 0.00 and 1.99 and "high" for values between 2.00 and 4.00.

We use conceptual hierarchies to rewrite attribute domain values as more generalized values, and we group records that have the same generalized values into the same clusters. We form clusters in an attempt to reduce the processing time of classification without sacrificing accuracy. Examples of conceptual hierarchies and resulting clusters are given in the next section.

Performance Studies

In this section, we describe our sample global schema and how the corresponding database is populated. A description of the goals and approaches of our experiments follows.

Testbed

The global schema in our study contains student records with string, numeric, and Boolean fields. Some attributes have conceptual hierarchies (either user-provided or generated). There are 16 attributes in our schema, 8 of which have a conceptual hierarchy. The attribute names, types, whether they have a conceptual hierarchy, and size or range of the domain are shown in Table 1. The schema is populated randomly from either the range given in the final column of the table or from a set of values of the given size.

Conceptual hierarchies for string attributes are user-defined. For example, a conceptual hierarchy for *city* generalizes cities to their respective countries, and countries are generalized to continents. Numeric attributes are generalized by applying an algorithm proposed by Han et al. [HF94]. Attributes *age, weight, height,* and *gpa* are generalized into 12, 36, 2, and 2 divisions, respectively. To illustrate the clustering process, an example global database that contains multiple instances of the same entity is shown in Table 2. The entity ID is known for this data set and appears in the second column. The generalized database, where the attributes with conceptual hierarchies have been replaced with more general values, is shown in two clusters in Tables 3 and 4. The data sets used in our experiments are described in the next section.

Experiments

Our experiments are in two major categories: the impact of preprocessing the data into clusters, and the impact of using a different classification strategy than previously proposed in the literature. To address the former, we examine the impact of clustering performance, in terms of speed and accuracy, using only decision trees as the classification technique. Then we compare the performance of decision trees and k-NN; we examine the performance of decision trees both with clustering and without compared to k-NN with clustering. The time required for experiments with unclustered data under k-NN was not feasible according to preliminary investigation. We compute the CPU time used to perform entity identification under various scenarios.

The metric used in the experiments for accuracy considers both positive and negative errors. Positive errors occur when instances that belong to the same entity are classified as instances of different entities. Negative errors occur when instances that belong to different entities are classified as instances of the same entity. The impact of total vectors misclassified on the actual number of records misclassified is difficult to find. To illustrate the difficulty, consider an entity E with instances e1, e2, and e3. Distance vectors are obtained by computing the distance of every instance with every other instance. If the vector <e1, e2> is classified as belonging to the same entity, then instances e1 and e2 are grouped together. If <e1, e3> is classified as different entities, but <e2, e3> is classified as the same entity, then there is a contradiction in determining to which entity the instances e2 and e3 belong. Therefore, we perform analysis on

attribute name	type	CH?	domain
first name	string		5186
last name	string		5186
age	numeric	yes	18..40
street address	string		5537
city	string	yes	1356
phone	string		5000
zipcode	string		5000
gpa	numeric	yes	0.0..4.0
association	string	yes	92
department	string	yes	13
married	boolean		2
campus	boolean		2
scholarship	boolean		2
specialization	string	yes	57
height	numeric	yes	4.0..7.0
weight	numeric	yes	90..220

Table 1. Testbed Schema

recordID	entityID	first name	zipcode	gpa	city
1	1	Krishna	45219	3.5	Madras
2	1	Krishnamoorthy	45220	3.6	Trichy
3	1	Krish	45221	3.6	Trichy
4	2	Rama	38203	1.5	Toronto
5	2	Raman	38211	1.8	Hamilton
6	2	Raman	37213	1.9	Toronto
7	3	Joseph	51234	2.9	Bombay
8	3	Joe	45220	2.8	Pune
9	4	Shiela	45219	1.2	Vancouver
10	4	Sheela	38211	0.9	Victoria
11	5	Robert	37213	3.2	Delhi
12	5	Rob	45220	3.4	Agra

Table 2. Example Global Database

misclassification in distance vectors rather than actual data records. In order to determine the accuracy of the results for clustered data compared to unclustered data, we compute confidence intervals on differences in error rates.

Our experiments study the impact of variations in different parameters on the automated entity identification process described here. We choose some parameters to vary where we expect the most important impact to be, and fix other parameter values for the sake of reducing the number of experiments where less impact is expected. Table 5 summarizes the values of parameters used in our experiments.

There are three different data sets with 2500 records each. The number of unique entities varies as 2%, 10%, and 50% of the total number of records. A previous study [GSR96] considers only the number of unique entities as 2% of the total number of records. As the number of unique entities increases, the number of instances of each unique entity decreases, which may impact the entity identification process. Each unique entity has at least one instance and the remaining instances of the data set are distributed randomly among the entities.

In order to simulate realistic errors in the data, we induce errors in some cases for multiple instances of the same entity. Different values for the attributes that do not have a conceptual hierarchy are generated. We fix the non-conceptual hierarchy error rate at 30% of the total number of instances of the data set. For 30% of the data set, 750 instances in our study, a number from 1 to *a* where *a* is the number of attributes that do not have conceptual hierarchies (8 here) is generated, say *e*. Then *e* attributes are randomly chosen for a particular record, and error is induced in these fields. String attributes are corrupted either by replacing the string entirely or adding, deleting, or modifying characters in the string. Boolean values are corrupted by inverting their values.

In order to study the impact of clustering errors on the accuracy of the entity identification process, we perform experiments with and without clustering errors. Clustering errors can only occur if there are multiple instances of the same entity, not across different entities.

recordID	entityI	first name	zip	gpa	city
1	1	Krishna	45219	high	India
2	1	Krishnamoorth	45220	high	India
3	1	Krish	45221	high	India
7	3	Joseph	51234	high	India
8	3	Joe	45220	high	India
11	5	Robert	37213	high	India
12	5	Rob	45220	high	India

Table 3. Generalized Values Resulting in Cluster 1

recordID	entityI	first	zipcode	gpa	city
4	2	Rama	38203	low	Canada
5	2	Raman	38211	low	Canada
6	2	Raman	37213	low	Canada
9	4	Shiela	45219	low	Canada
10	4	Sheela	38211	low	Canada

Table 4. Generalized Values Resulting in Cluster 2

parameter	*fixed*	*varying*
number of records	2500 records	
number of unique entities		2%, 10%, 50% of total records
non-conceptual hierarchy error	30% of total records	
conceptual hierarchy error rate		0%, 10% of total records
classified data set	10% of total records	
test set size	30% of classified data set	
training set size		10%, 80% of the global training set
application of decision tree with fixed training and test set sizes		randomly select the training and test set 3 times
values of k (k-NN) for a fixed training and test set		5 different values in the range 1 <= k <= 2*L-1, where L is the minimum number of positive and negative vectors
application of k-NN for a fixed training and test set size		3 different randomly selected training and test sets * 5 values of k = 15

Table 5. Experiment Foundations

For example, if there are two instances of the same student, both of their GPA values should generalize to the same value. If one instance generalizes to "high" and the other to "low," then this is a clustering error. We vary the clustering errors as 0% (i.e., no clustering errors) and 10% of the number of instances in a data set. For example, if there are 100 instances in a data set with a 10% clustering error rate, then 10 randomly chosen instances have attributes with clustering errors.

Given the variations due to the number of unique entities and clustering error rates, we have 6 different data sets for conducting experiments. Data sets with 50 unique entities and 0% and 10% clustering errors are denoted as 50CE0 and 50CE10, respectively. The other data sets are 250CE0, 250CE10, 1250CE0, and 1250CE10.

In order to study the effect of the training set size on the performance of classification techniques, the training set size is varied. The training set is formed from a set of records that is already classified, i.e., it is known to which entity each instance belongs. The size of the classified data set is fixed as 10% of the data set. Distance vectors are formed by measuring the distance of every record with every other record using standard distance functions [GSR96]. The test set is fixed as 30% of the classified vector set. In order to study the impact of large and small training sets, the training set size is varied as 10% and 80% of the subset of the classified vector set formed by excluding the test set from the classified vector set, denoted as the global training set. Thus, for each of the 6 data sets, there are 2 different variations of the training set sizes.

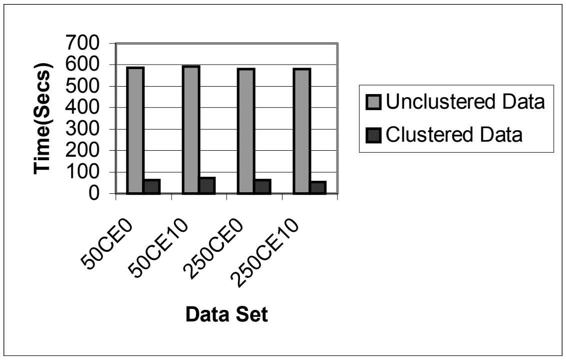

Figure 2. Processing Time for Decision Trees

For unclustered data, 48 experiments are performed. For clustered data, 328 experiments are performed with the decision tree technique and 1312 for k-NN. The entire suite is repeated 3 times for a total of 5064 experiments. Results are presented and discussed in the next section.

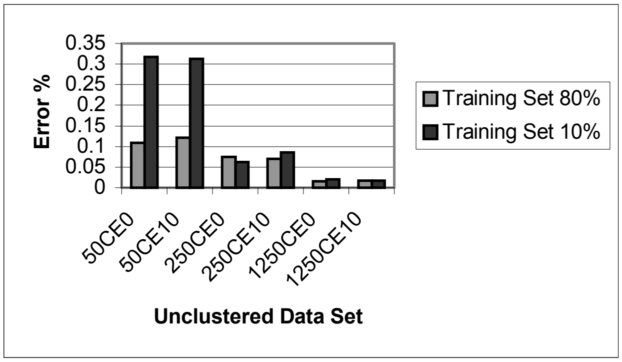

Figure 3. Error Rates for Decision Trees on Unclustered Data

Results

We present results here that investigate the performance impact (time and accuracy) of preprocessing the data as well as the impact of training set size, both large and small, on time and accuracy for both decision trees and k-NN. Representative results are given in the first two sections, followed by summaries of all results in the last two sections.

Preprocessing Impact

In order to assess the impact of preprocessing on performance, we perform experiments with clustered and unclustered data for the decision tree technique. Figure 2 shows processing time in seconds for clustered and unclustered data with a large training set size (80% of the global training set). The times shown for clustered data include the time for preprocessing (not greater than 2.7 seconds for any data set.) The difference in processing time between the large training set (80%) and the small training set size (10%) ranges between 14 and 32 seconds; on average, the processing time for the larger training set is 23 seconds more than the smaller training set size. We observe that there is a savings of approximately 89% for clustered data compared to unclustered data. The data sets not shown (with 1250 unique entities) did not form usable clusters due to an insufficient number of positive vectors. We use only clusters with two or more positive vectors so that the training set and test have at least one each.

In order to determine the impact of preprocessing on accuracy, we examine the total number of misclassified vectors for processing clustered and unclustered data. To compare the error rates, we construct 95% confidence intervals for the difference in error rates observed for clustered and unclustered data, shown in Table 6. We observe that the accuracy for classifying clustered data is always lower than that of unclustered data; however, the clustered accuracy relative to the unclustered accuracy decreases as the number of unique entities increases. For example, in the case of 50 unique entities (50CE0) with the larger training set (80%), the system misclassifies somewhere between 1.20% and 1.49% more records for clustered data than for unclustered data; however, when the number of unique entities is 250, the system misclassifies between 0.60% and 0.83% more records. In other words, the differences in error rates are lower as the number of unique entities increases, possibly because it is equally difficult to distinguish entities in the unclustered data when there are fewer duplicate instances. Although these results are not general since they are drawn over one testbed with a small number of data sets, they are encouraging for further investigation of preprocessing using conceptual hierarchies to speed up entity identification using decision trees.

Error rates for processing unclustered data with decision trees are given in Figure 3.

Data Set	Confidence Interval with 80% Training Set Size		Confidence Interval with 10% Training Set Size	
	Low	High	Low	High
50CE0	1.20	1.49	3.65	4.16
50CE10	1.02	1.29	5.78	6.36
250CE0	0.60	0.83	0.82	1.07
250CE10	0.62	0.87	0.82	1.09

Table 6. Preprocessing Accuracy for Decision Tree Classification

Impact of k-NN as a Classification Technique

In order to study the processing time for k-NN compared to decision trees for performing entity identification, we compare the processing time for large and small training sets using clustered data. For both large and small training sets, decision trees outperform k-NN. On average, for a larger training set size the difference is 201 seconds (ranging from 126 to 278 for k-NN); the difference is smaller for the small training set, on average 34 seconds (ranging from 19 to 52 seconds for k-NN).

The total vectors misclassified as a percentage of the total vectors is given in Figure 4 for decision trees with large and small training set sizes (Decision80 and Decision10, respectively) and for k-NN with large and small training set sizes (KNN80 and KNN10, respectively.) We observe that k-NN with small and large training sets has better accuracy results than decision trees as the number of unique entities increases. Additional studies are needed to determine whether the trend continues and whether the gain in accuracy is worthwhile

Processing Speed Results

The impact of preprocessing on entity identification processing speed under different scenarios is summarized in Table 7.

Accuracy Results

Table 8 summarizes the accuracy results of our study. When we cluster the data and apply entity identification on each cluster, the search region of the classification system is reduced when compared to non-clustering, making the classification system a local expert on that cluster. On the other hand, the number of training records in each cluster is fewer than the total training records without clustering, which might reduce the accuracy of entity identification when clustering is done. In the experiments, we find that clustering accuracy is always lower than non-clustering accuracy.

We expect that all the techniques perform better with large training sets than with small training sets because as the training set size increases, the classification system is trained better. Our expectation matches the experiment results for decision trees. When k-NN is applied, for the data set with a small number of unique entities, we obtain the expected results, but for the data set with a larger number of unique entities, the accuracy is slightly better with the smaller training set size. Since this is unexpected, further study of the impact of the number of unique entities on accuracy could be investigated.

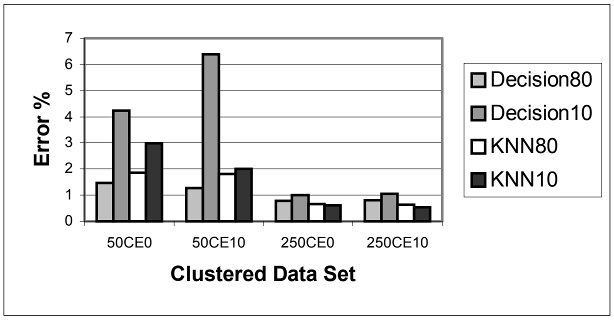

Figure 4. Error Rates for Clustered Data

Discussion and Conclusions

Automated approaches to database integration are reviewed here, followed by a discussion of our contributions and areas for future work. Schema integration involves semantic integration, the process of determining which attributes are equivalent between databases. Li et al. [LC94] employ neural networks to perform semantic integration in an automated manner. They extract metadata (attribute names and descriptions, for example) from databases automatically using neural networks. Dao et al. [DP95] use the relationships between and among the entire sets of attributes to perform semantic integration. They use data mining techniques devised by Han et al. [HCC92] to find the relationships between attributes. Lu et al. [LFG97] have employed data mining techniques to solve the semantic conflicts that arise during schema integration. All of these approaches employ data mining techniques to automate schema integration, the first phase of database integration. In our work, we assume that schema integration has already taken place and focus on the application of data mining techniques to automate data integration.

Scheuermann et al. [SC94] propose an approach using role-sets for dynamic database integration in a multidatabase environment. The role-set approach is based on the observation that many conflicting data values for the same real world entity are not inconsistencies, but values that correspond to the same entity appearing in multiple roles. Their approach performs dynamic data integration (on demand with no global schema) using a multidatabase key (global ID), either provided in the data or determined using an automated technique [LC4, SLC98]. Our approach to data integration assumes a global schema is known but no global ID is present.

We apply clustering in a framework for entity identification and study its performance; we implement k-NN for entity identification and study its performance. Our conclusions are that a tremendous savings in processing time is indicated when clustering is done, however, the accuracy of techniques over non-clustered data is always better than over clustered data. Clustering errors impact the results at the same rate and thus do not introduce any additional error other than the original clustering error. For clustered data, we observe that the processing time is better for decision trees, but k-NN has better accuracy in all cases except when there is a small number of unique entities and a large training set size. This means there are a large number of copies of the same entity, and decision trees with a large training set gives higher accuracy than k-NN.

In our testbed, we vary the number of unique entities as a percentage of the total number of records in the data set. When the number of unique entities is equal to half the number of total records in the data set and when the data set is clustered, there are very few positive training vectors in some clusters and they are not sufficient to perform entity identification on the clusters. In future work, training sets from other clusters may be used to classify the unknown vectors in these clusters and the impact of this approach can be studied.

There are two topics of future investigation concerned with making changes in the testbed. The first one is to study the performance of the classification techniques by varying the number of attributes in the schema. A second area varies the fixed parameter values. We fix the values of some parameters in our experiments for the sake of reducing the number of experiments. Varying the previously fixed parameters (or expanding the values

Techniques	Processing Time
Decision Trees: Clustered and Unclustered	When clustered, 89.2% savings with larger and 89.1% savings with smaller training sets.
Decision Trees on Unclustered Data Sets: Training Set Size	Small savings with smaller training sets
Decision Trees on Clustered Data Sets: Training Set Size	No significant savings with smaller training sets
k-NN on Clustered Data Sets: Training Set Size	Considerable savings with smaller training sets
Decision Trees and k-NN: Clustered	Decision trees: 76% savings with larger and 35.77% savings with smaller training set compared to k-NN
Decision Trees Unclustered and k-NN Clustered	k-NN: 55% savings with larger and 83% savings with smaller training sets compared to decision trees

Table 7. Processing Speed Results

Techniques	Effect of Training Set Size	Effect of Clustering Errors	Effect of Increase in Unique Entities
Decision Trees on Clustered and Unclustered Data Sets	Better with larger training set	No significant difference	Clustering accuracy less than non- clustering accuracy, difference decreases with increase in unique entities
Decision Trees on Unclustered Data Sets	Better with larger training set	No significant difference	Accuracy increases with increase in unique entities
Decision Trees on Clustered Data Sets	Better with larger training set	No significant difference	Accuracy increases with increase in unique entities
k-NN on Clustered Data Sets	Better with larger training set for small unique entities, same for large unique entities	No significant difference	Accuracy increases with increase in unique entities
Decision Trees and k-NN on Clustered Data Sets	k-NN better than decision trees with smaller training set, With large training set, decision trees give better accuracy than k-NN for small unique entities	No significant difference	k-NN accuracy better than decision trees with increase in unique entities
Decision Trees Clustered and k-NN Unclustered Data Sets	Better with large training set	No significant difference	Clustering accuracy less than non- clustering accuracy, difference decreases with increase in unique entities

Table 8. Accuracy Results

considered for other parameters) to study the impact on the performance of the classification techniques could examine the impact of the size of the testbed, the training set size and the conceptual and non-conceptual hierarchy error rates, for example. Future investigation with both more repetitions of the experiments and real databases could be done. In addition, other data mining techniques could be investigated in the learning module to perform entity identification.

References

[DP95] Dao, S., and B. Perry, "Applying a Data Miner to Heterogeneous Schema Integration," *Proceedings of the First International Conference on Knowledge Discovery in Databases,* Montreal, Canada, Aug. 1995, pp. 63-68.

[FBF77] Friedman, J.H., Bentley, J.L., and R.A. Finkel, "An Algorithm for Finding Best Matches in Logarithmic Expected Time," *ACM Transactions on Mathematical Software*, Vol. 3 (3), 1977, pp. 209-226.

[GSR96] Ganesh, M., Srivastava, J., and T. Richardson, "Mining Entity-Identification Rules for Database Integration," *Proceedings of the Second International Conference on Knowledge Discovery in Databases*, Portland, OR, Aug. 1996, pp. 291-294.

[HCC92] Han, J., Cai, Y., and N. Cercone, "Knowledge Discovery in Databases: An Attribute-Oriented Approach," *Proceedings of the 18th VLDB Conference,* Vancouver, British Columbia, Canada, 1992, pp. 547-559.

[HF94] Han, J., and Y. Fu, "Dynamic Generation and Refinement of Concept Hierarchies for Knowledge Discovery in Databases," *Workshop on Knowledge Discovery in Databases*, Seattle, July 1994, pp. 157-168.

[LC94] Li, W., and C. Clifton, "Semantic Integration in Heterogeneous Databases Using Neural Networks," *Proceedings of the 20th VLDB Conference,* Santiago, Chile, Sep. 1994, pp. 1-12.

[LFG97] Lu, H., Fan, W., and C.H. Goh, "Discovering and Reconciling Semantic Conflicts: A Data Mining Perspective," *Seventh Conference on Database Semantics*, Leysin, Switzerland, October 1997, pp. 409-427.

[SLC98] Scheuermann, P., Li, W., and C. Clifton, "Multidatabase Query Processing with Uncertainty in Global Keys and Attribute Values," *Journal of the American Society for Information Science,* Vol. 49 (3), 1998, pp. 283-301.

[SC94] Scheuermann, P., and E.I. Chong, "Role-based Query Processing in Multidatabase Systems," *Proceedings of the International Conference on Extending Database Technology*, Mar. 1994, pp. 95-108.