Integration of Fuzzy Databases: Problems & Solutions

Awadhesh Kumar Sharma
Department of Computer Sc & Engg
MMM Engg College Gorakhpur
273010, UP, India

A. Goswami
Department of Mathematics
IIT Kharagpur-721302
West Bengal, India

D.K. Gupta
Department of Mathematics
IIT Kharagpur-721302
West Bengal, India

ABSTRACT

In this paper, problems in integration of fuzzy relational databases have been investigated and some solutions have been proposed. In general, database integration consists of two main processes called schema integration and instance integration that results into global schema and global instance respectively. Current work assumes a schema integration process to get a global schema from a collection of existing (local) fuzzy relational databases. Instance integration can be classified into three distinct levels according to the extent to which instance integration is carried out. We have focused only on the level-0 instance integration. There are two major problems in instance integration: entity identification; and attribute value conflict resolution that has been discussed in the context of fuzzy database integration and a solution is proposed.

Keywords

Integration of fuzzy databases, fuzzy schema integration, fuzzy instance integration.

1. INTRODUCTION & BACKGROUND

Databases hold data that represent properties of real-world objects. Ideally, a set of real-world objects can be described by the constructs of a single data model and stored in one and only one database. Nevertheless, in reality, one can usually find two or more databases storing information about the same real-world objects. There are several reasons that result in the overlapping representations. These include:

- Different roles played by the same real-world objects in different applications. For example, a company can be the customer as well as the supplier for a firm. Hence, the company's information can be found in both the customers' database and supplier’s database.
- For performance reasons, a piece of information may be fully or partially duplicated and stored in databases at different geographical locations. For example, the customers' information may be stored in both the branches and headquarter.
- Different ownership of information can also lead to information stored in different databases. For example, the information of a raw material item may be stored in different production databases because each production line wants to own a copy of the information and to exercise control over the information.

When two or more databases represent overlapping sets of real world objects, there is a strong need to integrate these databases in order to support applications of cross- functional information systems. It is, therefore, important to examine strategies for database integration. An important aspect of database integration is the definition of a global schema that captures the description of the combined (or integrated) database. Here, we define schema integration to be the process of merging schemas of databases, and instance integration to be the process of integrating the database instances. Schema integration is a problem well studied by database researchers in [1, 9, 10, 12, 18]. The solution approaches identify the correspondences between schema constructs (e.g. entity types, attributes, etc.) from different databases and resolve their differences. The end result is a global schema which describes the integrated database. In contrast, instance integration focuses on merging the actual values found in instances from different databases. There are two major problems in instance integration:

a) entity identification; and
b) attribute value conflict resolution

The entity identification problem involves matching data instances that represent the same real-world objects. The attribute value conflict resolution problem involves merging the values of matching data instances. These two problems have been studied in [3, 13, 23] and [7, 14, 21] respectively. It is not possible to have attribute value conflicts resolved without entity identification because attribute value conflict resolution can only be done for matching data instances. In defining the integrated database, one has to choose a global data model so that the global schema can be described by the constructs provided by the data model. The queries that can be formulated against the integrated database also depend on the global data model. The selection of global data model depends on a number of factors including the semantic richness of the local databases [19, 20] and the global application requirements. Nevertheless, the impact of instance integration on the global data model has not been well studied so far. In this paper, we study this impact in the context of fuzzy relational data model.

1.1 Database Integration Process

Database integration, involving both schemas and instances of databases, should be performed in database migration/ consolidation, data warehouse, and multidatabase systems. Regardless of the mode of integration, the basic database integration tasks are essentially the same. We view the entire database integration as a set of processes which derives the integrated schema and instances that can be implemented on either multidatabase or data warehouse systems.

Logical steps in which the integrated database is derived from the existing (local) databases does not dictate exactly how and when the steps should be performed. For example, for the actual consolidation of databases, schema integration and instance integration should be performed together. However, if only a virtual integration is required, schema integration will be performed once but the instance integration will be performed whenever queries are evaluated against the integrated database. The actual schema and instance integration techniques adopted will depend on a number of factors such as the global applications' requirements, types of conflicts found among local databases and data quality of local databases.

1.1.1 Schema Integration Process

Each local database consists of a schema and a set of data instances. The schema integration process requires knowledge about the local database schemas. The knowledge about database schema can be discovered from the database content. For example, database reverse engineering extracts applications' domain knowledge by analyzing not only the database schema but also database instances of an existing database [5]. However, we always require the database designers or administrators to supply additional knowledge manually. Schema integration produces the global schema as well as the mappings between the global schema elements and the local schema elements. Very often, a local schema can be vastly different from the global schema. This can be caused by different data models or database design decisions adopted by local databases and the integrated database. We may, therefore, have to introduce a view of the local schema, called export schema, such that the local database through the export schema can be seen compatible with the global schema. An export schema also defines the portion or subset of a local database to be integrated. The local database to export database conversion usually involves schema transformation. Efforts in this area are reported in [8, 16, 24].

In this research, we assume that the schema integration process has been carried out to the extent that a global schema has been obtained from a collection of existing (local) fuzzy relational databases. Hence, global users or applications will formulate their queries based on the global schema. Moreover, export relational schemas that are compatible with respect to the global schema have been defined upon the local fuzzy databases. We classify instance integration into three distinct levels according to the extent to which instance integration is carried out:

Level-0: Neither entity identification nor attribute value conflict resolution is performed. Since no instance integration is involved, the integrated database is defined merely by collecting the instances from different local databases into relations specified by the global schema.

Level-1: Entity identification is performed but not attributes value conflict resolution. Hence, local database instances which correspond to the same real-world objects are matched and combined in the global relations. However, the attributes of these matching database instances are not merged.

Level-2: (complete integration). Both entity identification and attribute value conflicts are resolved. In this case, the local database instances are completely integrated.

1.1.2 Instance Integration Process

During instance integration process the entity identification always precedes attribute value conflict resolution since only the conflicting attribute values of matching data instances should be resolved. Throughout the entire instance integration, any detected erroneous integration result (e.g. two data instances from the same existing databases is matched to one single data instance from another database) is forwarded to the schema integration process as a feedback if the error is possibly caused by incorrect schema integration. This can happen when the schema integration makes use of hypothesis obtained by sampling the local databases. However, this hypothesis may not hold for all local database instances.

While considering real world objects another very important consideration that needs to be taken into account is the inherent fuzziness in the data instances. Often the data we have to manage are far from being precise and certain. Indeed, the attribute value of an item may be completely unknown or partially known (a probability distribution is known on the possible values of attribute, for example). Besides an attribute may be irrelevant for some of the considered items; moreover, we may not know whether the values does not exist or is simply unknown. In such circumstances fuzzy relations are incorporated in the database. Integration of fuzziness in database provides means of representing, storing, and manipulating imprecise and uncertain information. Since our knowledge of the real world is often imperfect, one's ability to create databases of integrity poses a great challenge. To maintain the integrity of database in situations where knowledge of the real world is imperfect, one may either restrict the model of database to the portion about which only perfect information is available leading to the loss of valuable information, keeping relevant data unexplored, unanswered queries, unsatisfied user requests and resulting in degraded quality of information delivery. To overcome the aforesaid hazards, formalism has been suggested that allow the representation, storage, retrieval and manipulation of uncertain information. In this research work the term FUZZY is used as a generalized term implying imprecision, uncertainty, partial knowledge, vagueness and ambiguity.

2. SEMANTIC CONFLICTS IN FUZZY RELATIONAL MULTIDATABASES

In a multidatabase with global schema, individual schemas of component databases are merged into a single global conceptual schema for all independent databases by integrating their schemas [2, 6, 17]. In the process of schema integration, a core problem is to identify the same real world object from component databases and then resolve a large number of incompatibilities that exists in different component databases. If the component relations have a common key, the component tuple with the same key values must describe the same real world object, and they can be integrated to produce a single tuple, called target tuple with outer-join [4] or outer-union [15, 22] operation after resolving the conflict. According to semantic relationship between components tuples there may be existing four types of conflicts [7, 22] as follows:

a) Naming Conflict: This type of conflict can be divided into two aspects. One is semantically related with data items being named differently and other is semantically unrelated with data items being named equivalently.

b) Data Type Conflict: This case occurs when semantically related data items are represented in different data types.

c) Data Scaling Conflicts: This case occurs when semantically related data items are represented in different databases using different units of measure.

d) Missing Data: This case occurs when the schemas of component databases have different attribute sets.

Since fuzzy relational databases exist in multiple relational databases and crisp relational databases are essentially the special form of fuzzy relational databases, there are new types of conflicts that should be resolved in schema integration together with the conflicts identified above. Let r and s be component fuzzy relations under fuzzy relation schemes R and S, and t_r, t_s be their tuples, called component tuples, respectively. Let R and S have a common key such that $dom(Key\_Attribute)$ is a crisp set. Let ($Key\_Value(t_r) = Key\_Value(t_s)$).

2.1 Membership Degree Conflict:

It occurs at the level of tuples and can be classified into two classes as follows:

a) Missing Membership Degree: When one of two tuples belongs to a crisp relation.

b) Inconsistent Membership Degree: This occurs when $\mu_R(t_r) = \mu_S(t_s)$.

Example: Consider the following three relations *Student*, *Sincere_Student* and *Smart_Student* given in table 1. There exist a conflict of missing membership between tuples of relation *Student* and *Sincere_Student* as well as *Student* and *Smart_Student* where as a conflict of inconsistent membership exists between the tuples of relations *Sincere_Student* and *Smart_Student.*

Table 1: Fuzzy relations with membership degree conflicts

Student	
Roll	**Name**
33	Soma

Sincere_Student		
Roll	**Name**	**Mu**
33	Soma	0.6

Smart_Student		
Roll	**Name**	**Mu**
33	Soma	0.9

2.2 Attribute Value Conflicts in Identical Attribute Domains:

Let $dom(A_i) = dom(A_j)$ where $A_i \in R$ and $A_j \in S$, A_i and A_j are semantically related.

(a) Inconsistent Crisp Attribute Values: When $dom(A_i) = dom(A_j)$ is a crisp set but $t_r(A_i) \neq t_s(A_j)$. Example: Age of "Roma" in Relation r is "22", but in relation s it is"25".

(b) Missing Fuzzy Attribute Values: When $dom(A_i) = dom(A_j)$ is a fuzzy set but $t_r(A_i)$ is crisp and $t_s(A_j)$ is fuzzy. Example: Age of "Roma" in Relation r is "22", but in relation s it is "0.8/22".

(c) Inconsistent Fuzzy Attribute Values: When $dom(A_i) = dom(A_j)$ is a fuzzy set but $t_r(A_i) \neq t_s(A_j)$. Example: Age of "Roma" in Relation r is "0.7/22" but in relation s it is"8/22".

2.3 Missing Attributes:

It occurs when an attribute in relation scheme R is semantically not related with any of the attributes in relation scheme S i.e. R and S have different set of attributes.

2.4 Attribute Name Conflict:

This conflict has two aspects:

a) Semantically related attributes are named differently, i.e. Synonyms.

b) Semantically unrelated attributes are named equivalently, i.e. Homonyms.

It is important to note that one is not concerned with the conflicts of missing attributes and attribute names if component relations are fuzzy.

2.5 Attribute Domain Conflict:

Let $dom(A_i) \neq dom(A_j)$ where $A_i \in R$ and $A_j \in S$, A_i and A_j are semantically related.

a) **Data Format Conflicts:** Although A_i and A_j have same data type and data unit, they have different expressive formats: For example-$t_r(A_i)$ and $t_s(A_j)$ both represent an attribute value say the date

November 30, 2009, but $t_r(A_i) = "30.11.2009"$ and $t_s(A_j) = "11.30.2009"$.

b) **Data Unit Conflict:** Attribute A_i and A_j have same data type but their unit of measure are different: For example- $t_r(A_i)$ and $t_s(A_j)$ are all real data but $t_r(A_i) = "22kg"$ and $t_s(A_j) = "22lb"$.

c) **Data Type Conflict:** Attribute A_i and A_j have different data types. For example- we may have $t_r(A_i) = "23"$ and $t_s(A_j) = "22.5"$ which are integer and real respectively.

3. RESOLUTION OF SEMANTIC CONFLICTS IN FUZZY RELATIONAL MULTIDATABASES

Among the conflicts mentioned in section above some of them including missing attributes, attribute name conflicts, inconsistent crisp attributes values on identical attribute domains and inconsistent crisp attribute values on different attribute domains, have been investigated and resolved [7, 11]. In this paper we focus on some new types of conflict in connection to fuzzy databases. We integrate t_r and t_r to form a tuple t_g. The values of component attribute of t_g other than that of key attribute are formed after resolving the conflicts between semantically related attribute values. Here we assume that there is no attribute name conflict in t_r and t_s because they can be resolved beforehand.

3.1 Membership Degree Conflict

Consider two relations r and s under relation schemes $R(K,C)$ and $S(K,C,Mu)$ respectively, where K the key attribute is, C is the set of attributes that are common to both the relations and Mu is the membership degree attribute. Let $dom(K)$ is crisp and $t_r(K) = t_s(K)$. Then t_r and t_s denote the same real world object. Hence it is resolved that the integrated tuple t_g would be under relation scheme $G(K,C,Mu)$ such that $t_g[K] = t_r[K] = t_s[K]$; $t_g[C] = t_r[C] = t_s[C]$; $t_g[Mu] = \max(1, t_s[Mu])$ =1 .

Now consider two relations r and s under relation schemes $R(K,C,Mu)$ and $S(K,C,Mu)$ respectively, where the attributes K,C,Mu are the same as defined above. Let $dom(K)$ is crisp and $t_r[K] = t_s[K]$. Then t_r and t_s denote the same real world object. Let $t_r[Mu] = t_s[Mu]$. Hence it is resolved that the integrated tuple t_g would be under relation scheme $G(K,C,Mu)$ such that $t_g[K] = t_r[K] = t_s[K]$, $t_g[C] = t_r[C] = t_s[C]$, $t_g[Mu] = \max(t_r[Mu], t_s[Mu])$

3.2 Attribute Value Conflicts in Identical Attribute Domains

Now consider two relations r and s under relation schemes $R(K,C,Mu)$ and $S(K,C,Mu)$ respectively, where the attributes K,C,Mu are the same as defined above. Here the attribute indicating membership degree is omitted for the sake of simplicity of discussion. If included, the potential conflicts can be resolved by applying the above methods. Let $dom(K)$ is crisp and $t_r[K] = t_s[K]$. Then t_r and t_s denote the same real world object. Thus $t_g[K] = t_r[K] = t_s[K]$. Now let $A \in C$ then we have following resolutions:

a) When $t_r[A]$ and $t_s[A]$ are crisp and $t_r[A] \neq t_s[A]$, the conflict of inconsistent attribute values occurs and $t_g[A] = [t_r[A], t_s[A]]$, being a partial value (DeMichiel,1989). If $t_r[A] = t_s[A]$ then $t_g[A] = t_r[A] = t_s[A]$.

b) When $t_r[A]$ and $t_s[A]$ are crisp and fuzzy respectively, the conflict of missing fuzzy attribute value occurs. Then $t_g[A] = t_r[A]$.

c) When $t_r[A]$ and $t_s[A]$ are crisp and fuzzy respectively and $t_r[A] \neq t_s[A]$, the conflict of inconsistent fuzzy attribute values occurs and $t_g[A] = t_r[A] \overset{f}{\cup} t_s[A]$ where $\overset{f}{\cup}$ represents fuzzy union.

3.3 Attribute Value Conflicts in Inconsistent Attribute Domains

In order to resolve it, firstly the conflicts of attribute domains should be resolved. For this purpose component relations are converted into other relations, called virtual component relations. The attributes in virtual component relations are called virtual attributes [7, 22]. There is no attribute domain conflict in virtual component relations because they are resolved by mapping an attribute concerned with domain conflict in an original component relation to the corresponding virtual attribute. It is clear that such mappings must also have been done between a tuple in original component relation and the corresponding tuple in virtual component relation, called virtual tuple, or more precisely between an attribute value and a value of the corresponding virtual attribute. According to different types of attribute domain conflicts, the above mentioned mappings can be classified into one-to-one, many-to-one or one-to-many mapping. Instead of integrating original component relations, their virtual component relations are integrated to form the target relation.

4. CONCLUSION

In this paper, the impact of instance integration on the global fuzzy relational data model has been studied and some resolutions of semantic conflicts in fuzzy relational multidatabases are proposed.

5. REFERENCES

[1] Batini, C., Lenzerini, M., Navade, S.B. (1986). A coperative analysis of methodlogies for database schema integration", *ACM Computing Surveys* 18(4), pp 323-364.Ding, W. and Marchionini, G. 1997 A Study on Video Browsing Strategies. Technical Report. University of Maryland at College Park.

[2] Breitbart, Y., Olson, P.L., Thompson, G.R. (1986). Database Integration in a Distributed Heterogeneous Database System", *In: Proceedings of IEEE International Conference on Data Engineering*, pp 301-310.

[3] Chatterjee, A., Segev, A.(1991). Data manipulation in heterogeneous databases", *SIGMOD Record*, 20(4), pp 64-68.

[4] Chen, A.L.P.(1990). Outerjoin Optimization in Multidatabase Systems", In: Proccedings of IEEE International Symposium on Databases in Parallel and Distributed Systems, pp 211-218.

[5] Chiang, R.H.L. , Barron, T.M., Storey, V.C .(1994). Reverse engineering of relational databases: Extraction of an EER model from a relational database". *Data andKnowledge Engineering*, 12(2), pp 107-142.

[6] Deen, S.M., Amin, R.R., Taylor, M.C. (1987), Data Integration in Distributed Databases", *IEEE Transactions on Software Engineering*, 13, pp 860-864.

[7] DeMichiel, L.G. (1989). Resolving database incompatibility: an approach to performing relational operations over mismatched domains. IEEE Trans. on Knowledge and Data Engineering 1(4), pp 485-493.

[8] Fahrner C., Vossen G.,(1995). A servey of database design transformations based on entity relationship model", *Data and Knowledge Engineering*, 15(3), pp 213-250.

[9] Hayne, S., Ram, S. (1990). Multi-user view integration system (MUVIS): An expert system for view integration", In: *Proc. Intl. Conf. on Data Engineering*, pp 402-409.

[10] Kaul, M., Drosten, K., Neuhold, E.J. (1990). Integrating heterogeneous information bases by object-oriented views", In: *Proc. Intl. Conf. on Data Engineering*, pp 2-10.

[11] Kim, W., Choi, I., Gala, S., Scheevel, M.(1995). On Resolving Schema Heterogeneity in Multidatabase Systems", In: Kim, W., (eds.): *Modern Database Systems: the Object Model, Interoperability, and Beyond. Addison-Wesley ACM Press*, pp 521-550.

[12] Larson, J.A., Navade, S.B., Elmasari, R. (1989), A theory of attribute equivalence in database with application to schema integration", *IEEE Trans. on Software Engineering*, 15(4), pp 449-463.

[13] Lim E.P., Srivastava J., Prabhakar S., Richardson J. (1993). Entity identification problem in database integration". In: *Proc, Intl. Conf. on Data Engineering*, 294-301.

[14] Lim, E.P., Srivastava, J., Shekhar, S. (1994), Resolving attribute incompatibility in database integration: An evidential reasoning approach". In: *Proc, Intl. Conf. on Data Engineering*, pp 154-163.

[15] Ma, Z.M., Zhang, W., Ma, W. (1999). View Relation for Schema Integration of Multiple Databases and Data Dependencies". In: *Proceedings of 9th International Database Conference on Hetergeneous and Internet Databases*, pp 278-289.

[16] Meier A. et al.(1994). Hierarchical to relational database migration", *IEEE Software*, pp 21-27.

[17] Motro, A. (1987). Super-views: Virtual Integration of Multiple Databases", *IEEE Transcations on Software Engineering*, 13, pp 785-798.

[18] Spaccapietra, S., Parent, C., Dupont, Y., (1992). Model independent assertions for integration of heterogeneous schemas", *Very Large Database Journal*, l(l), pp 81-126.

[19] Seth, A.P., Larson, J.A. (1990). Federated database systems for managing distributed heterogeneous and autonomous databases", *ACM Computing Surveys*, 22(3), pp 183-236.

[20] Saltor, F., Castellanos, M., Garcia-Solaco, M. (1991). Suitability of data models as canonical models for federated databases", *SIGMOD Record*, 20(4), pp 44-48.

[21] Tasi, P.S.M., Chen, A.L.P. (1993), Querying uncertain data in heterogeneous databases", In: *Proc. RIDE-IMS Conf.*, pp 161-168.

[22] Tseng, F.S.C., Chen, A.L.P., Yang, W.P. (1993). Answerin Heterogeneous Database Queries with Degrees of Uncertianity", *Distributed and Parallel Databases: An International Journal*, 1, 281-302.

[23] Wang, Y.R., Madnick, S.E. (1989), The inter-database instance identifcation problem in integrating autonomous systems", In: *Proc. Intl. Conf. on Data Engineering*, pp 46-55.

[24] Zaniolo C. (1979). Design of relational views over network schemas", In: *Proc. ACM SIGMOD Intl. Conf. on Management of Data*, pp 179-190.