Semantic Relations: The key to integrating and query processing in heterogeneous databases[1]

Naphtali Rishe, Rukshan I. Athauda, Jun Yuan, Shu-Ching Chen
High-performance Database Research Center
School of Computer Science
Florida International University
University Park, Miami, FL 33199
{rishen, rathau01, yuanj, chens }@cs.fiu.edu

Abstract

Obtaining useful, complete, accurate information from on-line web data sources has become a challenging issue with multiple heterogeneous data sources on-line. With the increase of structured data sources moving on-line, collective integrated access to such information sources requires resolving semantic heterogeneity using innovative techniques for integration and query processing.

In this paper, we provide a theoretically sound, complete and unambiguous approach to resolving semantic heterogeneity using semantic knowledge. Advantages of using such knowledge in integration and query processing, such as acquiring additional information, complete answers to queries and techniques for intelligent query optimization are outlined.

1. Introduction

With the advent of the Internet, multiple heterogeneous structured data sources available on-line has increased, and thus created a need for access to these heterogeneous data sources in a collective manner. The World-Wide Web (WWW), usually, can be considered as a collection of unstructured documents. However, during the recent past, we have seen an increase number of structured information sources moving on-line. These sources include both free and commercial database on product-information, stock market information, real estate, automobiles, and entertainment. Access to such heterogeneous multiple data sources in collective manner has created a need to investigate the older research issues such as semantic heterogeneity with more vigor, interest and emphasis.

Early research on multi-databases [1], federated databases [11] and heterogeneous database systems [4][5][10] has resulted in different architectures for accessing multiple heterogeneous data sources. A main focus has been dealing with structural heterogeneity between data models and entities of schemas [1][6][7][11]. A plethora of approaches and innovative techniques has been proposed and implemented. However, problem of resolving semantic heterogeneity still evades requiring further research. For resolving semantic heterogeneity, techniques have been proposed to identify semantically related attributes and entities in different schemas using partial and fully automated methods [2][9]. We focus on classifying these semantic related constructs with the intention of utilizing such information to the full-extent in integration and query processing of heterogeneous database schemas.

Our approach to resolving semantic heterogeneity tries to exploit some of the advantages of dealing with structured data sources. Structured data sources, unlike un-structured or semi-structured data sources, consists of a schema with an unambiguous definition, whether explicitly stated or not, and a set of data items (extent) for each construct of the schema. Utilizing this information, we propose a set of relations, named *semantic relations*, which exploits both schema and its extent in integration and query processing. With the use of the semantic relations as the basis in integration, we can easily preserve data quality attributes including completeness and accuracy. An example best illustrates the problem of semantic heterogeneity and answer-completeness.

Example 1. Let us consider accessing two databases (i.e. DB1 and DB2) with the following schema.

[1] This research was supported in part by NASA (under grants NAGW-4080, NAG5-5095, NAS5-97222, and NAG5-6830) and NSF (CDA-9711582, IRI-9409661, HRD-9707076, and ANI-9876409).

DB1: Person(ssn, last_name, first_name, address)
DB2: Researcher(ssn, project_id, position)
Project(pid, project_name, fundingAgency)

The relation *Person* in database DB1 contains tuples describing the currently employed personnel at company A. Database DB2 describes researchers and their projects at lab L of company A since its inception. The relation *Researcher* contains tuples of researchers working or has worked at lab L of company A. The relation *Project* contains tuples of projects that the lab is currently working or already completed. The field *ssn* is the primary key of relation *Person* while fields *ssn* and *project_id* are the primary keys of relation *Researcher*. The field *ssn* in both relations *Person* and *Researcher* represent the social security number of a person. The field *pid* is the primary key of relation *Project* while *project_id* of relation *Researcher* is the foreign-key field referring *pid* field of relation *Project*.

For the above example schemas, let us consider the query, which obtains the last name of researchers who worked or are working at lab L and the names of the projects they worked on or are working on. Assume that the last name of persons working for company A are stored in DB1 as *last_name* field of relation *Person*, while the project names are stored in DB2 as *project_name* field of relation *Project*. We require accessing both databases DB1 and DB2. In the traditional approach to heterogeneous database integration, relation *Person* and relation *Researcher* will be mapped as *equivalent* since they both represent personnel working at company A or mapped as a sub-class/super-class relationship because relation *Researchers* represent a specialized class of all personnel working at company A represented by relation *Person*. Hence, to answer the above-mentioned query, the following SQL statement may be posed by the heterogeneous/multi-database or mediator system:

```
SELECT DB1.Person.last_name,
       DB2.Project.project_name
FROM   DB1.Person, DB2.Researcher,
       DB2.Project
WHERE  (DB1.Person.ssn =
       DB2.Researcher.ssn) AND
       (DB2.Researcher.project_id =
       DB2.Project.pid)
```

Note that the result of this query only suffices to provide a partial answer. Researchers who have worked on a project at lab L but not currently employed in company A are not represented in the query result. This aspect, known as *answer-completeness* of queries [8] becomes a major factor in dealing with multiple databases, especially on-line data sources. Determining answer-completeness is important with multitude of databases since this may determine the need to access additional data sources.

Our approach, based on semantic relations, for integration and query processing of multitude of data sources including structured web data sources is guaranteed to avoid errors such as incomplete answers. A major concern with most web users is obtaining relevant, complete, correct information from a variety of data sources available on the Internet. When dealing with structured on-line data sources, these factors translate to successful integration of on-line data sources and answer-completeness of user's queries. Our approach addresses both these situations successfully. In this paper, we present techniques based on semantic knowledge that is sound, unambiguous and complete to be used for integration and query processing in heterogeneous database environments. The major contributions of this paper include: (i.) a theoretical sound approach to heterogeneous schema integration using semantic relations. In sections 2.1 and 2.2, we define semantic relations, illustrate proofs for its completeness and soundness and also provide inference rules that can be used to automatically generate new semantic information from existing knowledge; (ii.) Classification of *interesting cases* for semantic relations, which yields additional information during integration, that is not explicit otherwise. We discuss and illustrate examples for each case in section 2.3; (iii.) Comparison of existing integration methodology to the proposed technique is illustrated in section 2.4; (iv.) Answer-completeness of queries due to the use of semantic knowledge for integration along with potential means of exploiting semantic knowledge for query processing is illustrated in section 3. Finally, section 4 presents future work and proposes an application area for the use of semantic knowledge.

2. Integration using Semantic Relations

An important feature of structured data sources is the availability of a schema along with data.

Schema is meta-data describing the data/information stored in the database. Hence, each construct in a schema (for instance, relation or field in a relational database schema) contains a set of data values or objects that it represents at any particular database instance, called the extent of the construct. Let us represent the extent of a construct, say *A*, as EXT(*A*). We can define four different types of semantic relations between two constructs of different database schema.

2.1 Semantic Relations

Let *A* be a construct of *Schema1* and *B* be a construct of *Schema2*. We can derive four possible semantic relations between constructs *A* and *B* as follows:

1. Semantically Equivalent (*SEM_EQ*): *A* is semantically equivalent to *B* (represented as, *A SEM_EQ B*) if and only if EXT(*A*) = EXT(*B*) for all database instances at any given time t.
2. Semantically Subset (*SEM_SUB*): *A* is semantically subset of *B* (represented as, *A SEM_SUB B*) if and only if EXT(*A*) $\subseteq$ EXT(*B*) for all database instances at any given time t_1 and EXT(A) $\subset$ EXT(B) for some database instance at time t_2.
3. Semantically Overlap (*SEM_OVER*): *A* is semantically overlapping with *B* (represented as, *A SEM_OVER B*) if and only if EXT(*A*) $\cap$ EXT(*B*) $\neq \phi$ for some database instances at time t_1 and EXT(*A*) $\cap$ EXT(*B*) $\neq$ *A* or EXT(*A*) $\cap$ EXT(*B*) $\neq$ *B* for all database instances at any given time t_2.
4. Semantically Disjoint (*SEM_DIS*): *A* is semantically disjoint with *B* (represented as, *A SEM_DIS B*) if and only if EXT(*A*) $\cap$EXT(*B*) = ϕ for all database instances at any given time t.

Note that the semantic relations are disjoint. That is, if *A* r_1 *B* and *A* r_2 *B* where r_1, $r_2 \in$ {*SEM_EQ*, *SEM_SUB*, *SEM_OVER*, *SEM_DIS*}, then $r_1 = r_2$.

Proof Idea: The completeness and correctness of the above semantic relations can be verified by examining all the possible scenarios of a Venn diagram for the extents of constructs *A* and *B*. This is shown in Figure 1. EXT(*A*) and EXT(*B*) are shaded in the figure. Note the ε represents the {domain of database containing construct A} $\cup$ {domain of database containing construct B}. Figures (a.) – (d.), depict all possible cases for semantic relations between any two database constructs *A* and *B*.

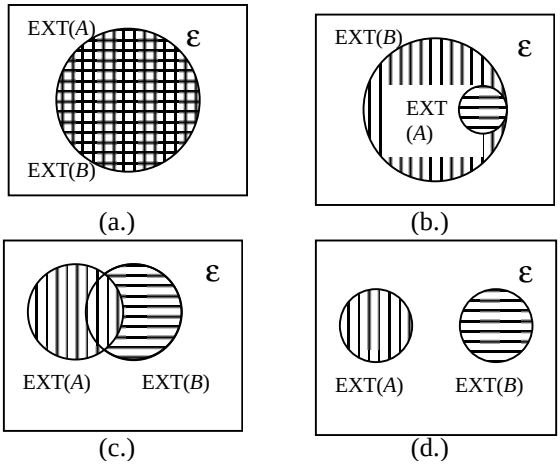

Figure 1. All possible scenarios for EXT(A) and EXT(B): (a.) EXT(A) = EXT(B); (b.)EXT(A) $\subseteq$ EXT(B); (c.) EXT(A) $\cap$ EXT(B) $\neq \phi$; (d.) EXT(A) $\cap$ EXT(B) = ϕ;

Some commutative rules for semantic relations include: *A SEM_EQ B* $\equiv$ *B SEM_EQ A*; *A SEM_DIS B* $\equiv$ *B SEM_DIS A*; *A SEM_OVER B* $\equiv$ *B SEM_OVER A*; where *A*, *B* are constructs of database schema.

The following example illustrates each semantic relation.

Example 2. Let us consider five constructs of different database schema in a university application.

Database	**Construct**	**Extent**
Registrar	*Employee*	contains all current employees of university A
Registrar	*Student*	contains all currently enrolled students of university A
Registrar	*Department*	contains all the department of university A
Payroll	*Faculty*	contains all current faculty of university A
Payroll	*Emp*	contains all current employees of university A

By considering the extents, we can assume the following:
Registrar.*Employee* SEM_EQ Payroll.*Emp* (because both constructs represents the current employees of university A)
Payroll.*Faculty* SEM_SUB Registrar.*Employee* (because *Faculty* construct contain the current faculty members of university A who are also employees of university A)
Payroll.*Faculty* SEM_OVER Registrar.*Student* (assuming that the faculty member can also be registered to courses as students in university A)
Registrar.*Department* SEM_DIS Payroll.*Emp* (since departments cannot be employees for obvious reasons)

2.2 Inference Rules

By examining the semantic relations in example 2, we can figure out for instance that Registrar.*Employee* SEM_DIS Payroll.*Emp* due to the same reason that Registrar.*Department* SEM_DIS Payroll.*Emp*. Thus, in order to derive new semantic relations from existing semantic relations, we have identified the following inference rules:

Assume that *A*, *B* and *C* are constructs of different database schema.
Rule 1: If *A* SEM_EQ *B* and *B* SEM_EQ *C* then for *A* SEM_EQ *C*
Rule 2: If *A* SEM_EQ *B* and *B* SEM_SUB *C* then *A* SEM_SUB *C*
Rule 3: If *A* SEM_EQ *B* and *B* SEM_OVER *C* then *A* SEM_OVER *C*
Rule 4: If *A* SEM_SUB *B* and *B* SEM_SUB *C* then *A* SEM_SUB *C*
Rule 5: If *A* SEM_SUB *B* and *B* SEM_DIS *C* then *A* SEM_DIS *C*
The inference rules 1-5 can be proved using set theory principles, however not shown here due to space limitations.

The above-mentioned inference rules derive the following set of semantic relations from the already identified semantic relations in example 2.
Payroll.*Faculty* SEM_SUB Registrar.*Emp*
Registrar.*Department* SEM_DIS Payroll.*Employee*
Payroll.*Faculty* SEM_DIS Registrar.*Department*

2.3 Interesting cases

The semantic relations, defined in section 2.1, have been enumerated previously in literature [3]. However, its powerful expressiveness and full potential have not been realized. If constructs *A* and *B* are related by either *SEM_EQ*, *SEM_SUB* or *SEM_OVER*, we classify the constructs as *interesting* for integration and query processing in a heterogeneous database environment. Two important aspects arise with these *interesting* cases:
(i.) Object Equivalence: Identification of common objects in EXT(*A*) and EXT(*B*) when *A SEM_EQ B* or *A SEM_SUB B* or *A SEM_OVER B*.
(ii.) Boundary Conditions: Specification of boundary conditions for constructs *A* and *B* when *A SEM_SUB B* and *A SEM_OVER B*.

2.3.1 Object Equivalence

When two constructs, say *A* and *B*, are known to be semantically related by either *SEM_EQ*, *SEM_SUB* or *SEM_OVER*, it is possible for EXT(*A*) and EXT(*B*) to have the same real-world objects represented (i.e. this is the set of objects in EXT(*A*) $\cap$ EXT(*B*)). The identification of equivalent objects in different constructs is especially advantageous if the constructs are entities. This allows extraction of extra information during integration of different schemas. This factor is illustrated in example 3.

Example 3. Let us consider two relational databases DB1 and DB2 consisting of students at university A:
DB1: *Pupil*(ssn, address)
DB2: *Student*(social_sec, gpa, phone)
For simplicity, let us assume that *Pupil* SEM_EQ *Student* and fields, ssn and social_sec, represent social security number in the same format and they are the primary keys of relations *Pupil* and *Student* respectively. Hence, if DB1.*Pupil*.ssn match with DB2.*Student*.social_sec, implies that objects are equivalent (i.e. the same student).

Since *Pupil* SEM_EQ *Student*, every object in *Pupil* has a matching object in *Student* and vice-versa. Now it is possible to obtain a relation, say STD in global schema, which contains attributes: *social_security*, *address*, *gpa*, and *phone* for every student object at university A. This information cannot be obtained by accessing DB1 or DB2 individually. That is, it was possible to obtain additional information (i.e. address, gpa, phone, collectively) for every student in

university A using an integrated access to DB1 and DB2. This example illustrates a simple scenario; this concept can be generalized for complex schemas.

2.3.2 Boundary Conditions

When either semantic relations, SEM_SUB or SUM_OVER relates two constructs, it is important to consider the boundary conditions on which the two constructs intersect. Considering these boundary conditions provides useful knowledge similar to object equivalence which otherwise is not explicit.

Example 4. Let us consider the scenario presented in example 1. Since relation *Person* contains all the employees currently working for company A and relation *Researcher* contains all the persons who worked or are working in lab L of company A, we can infer that *Person* SEM_OVER *Researcher*. The persons currently working at lab L who are also employees of company A consists of EXT(*Person*) ∩ EXT(*Researcher*). Current employees of company A not working in lab L are in {EXT(*Person*) – {EXT(*Person*) ∩ EXT(*Researcher*)}}. Researchers who used to work at lab L, but are not presently employees of company A are in {EXT(*Researcher*) – {EXT(*Person*) ∩ EXT(*Researcher*)}}. This knowledge is significant in query processing as shown below.

For instance, we can now answer the query that asks for social security numbers of researchers who worked in lab L but have left company A (not currently working for company A) as follows:

```
SELECT  DISTINCT DB2. Researcher.ssn
FROM    DB2.Researcher
WHERE  DB2.Researcher.ssn NOT IN
        (SELECT         DB1.Person.ssn
        FROM            DB1.Person)
```

This information could not be obtained by accessing the databases individually. This example illustrates a simple case, but can be generalized for complex schemas.

2.4 Integration

We use semantic relations as the basis for integration in a heterogeneous database environment. This is advantageous as it is complete, unambiguous and sound because they are based on the extent, which is same as the original DBA's view when he/she initially creates the constructs of the schema. The following example illustrates the difference between integration with semantic rules taken into consideration and without the use of semantic rules.

Example 5. Let us consider two relations *Pupil* and *Student* from DB1 and DB2 respectively, such that *Pupil* contains the currently enrolled students at university A and *Student* contains the currently enrolled student at university B. If we assume that a student cannot be enrolled in both universities A and B simultaneously, then relations *Student* and *Pupil* will be related by semantic relation SEM_DIS (i.e. DB1.*Student* SEM_DIS DB2.*Pupil*) according to the definition. This methodology is in contrast to existing semantic heterogeneity techniques, which relates *Student* and *Pupil* as "similar" or "equivalent". The argument is that *Student* and *Pupil* represent similar real-word concepts. However, if we define them as related (say by relation "equivalent") during query processing, it will lead to incomplete answers such as in example 1. Thus, the method of integration is ambiguous. It is true that there is a high-probability to find interesting cases of semantic relations by looking for similar concepts, however, finding similar concepts does not necessarily mean they are semantically related if they are presented in different contexts (such as the case where relation *Student*'s context is university A while relation *Pupil*'s context is university B). Such kinds of ambiguity do not occur with semantic relations.

The use of semantic relations do not restrict in defining a new relation, say *S*, in the global schema which has its extent as, EXT(S) = EXT(*Student*) ∪ EXT(*Person*), which contains both enrolled student at university A and university B. Object-oriented data models can represent relation *S* as a super class of relations *Student* and *Pupil*. Hence, use of semantic relations do not restrict the expressiveness in any way, on the contrary, provides an unambiguous definition of semantic relations between entities of different database for schema integration and interoperability in a heterogeneous database environment.

3. Query processing using semantic knowledge

Extraction of semantic knowledge in terms of semantic relations, object equivalences and boundary conditions during integration/ reconciliation process can be exploited for efficient, correct and intelligent query processing and optimization techniques. Detailed specification of query processing techniques exploiting the semantic knowledge is out-of-scope for this paper. However, we will briefly introduce two important techniques.

Complete answers: Using semantic knowledge, the query processor is able to provide complete answers to queries (see example 6 below).

Example 6. Let us consider the query in example 1. Semantic relation *SEM_OVER* relates the constructs *Person* and *Researcher* (see example 4). Thus, when trying to answer the query, it is apparent that there exist objects in *Researcher*, which are not in *Person* (see definition of *SEM_OVER*). Hence, the query processor will either look for a data source that will provide the missing values or place NULL (not known) values for the query result, thus providing a complete answer.

Intelligent query distribution: The query processor can utilize the semantic relations and boundary conditions to intelligently distribute the queries. For instance, if two constructs in different databases are related by *SEM_EQ* it is spurious to query both databases, since they contain the same information, rather choosing to query the less expensive and easily accessible data source. These types of optimizations techniques may result in significant performance gains, especially in web environment, where accessing certain web data sources may be extremely expensive. The use semantic knowledge for query processing and optimizing has significant potential and these issues will be investigated in detail in our future work.

4. Future Work

A significant challenge and the success of utilizing of semantic knowledge in a variety of application domains will depend on techniques that are developed for efficiently and accurately identifying semantic knowledge. We will focus on this issue in our future work. Semantic knowledge has potential to be used in a variety of applications, involving integration, interoperability of multiple data sources, such as mediator based web information integration systems. With the on-going efforts in XML-based web data sources, extracting schema/meta-data information is feasible for semi-structured data. Hence, concepts presented in this paper can be applied for integrating and querying such data sources in future.

5. References

[1] Batini C., Lenzerini M. and Navathe S.B., "A comparative analysis of methodologies for database schema integration". In ACM Computing Surveys, Vol.18, No.4, pp. 323-364, 1986.

[2] Bright M.W., Hurson A.R. and Pakzad S., “Automated Resolution of Semantic Heterogeneity in Multidatabases”. In ACM Transactions on Database Systems, Vol. 19, No. 2, pp. 212-253, 1994.

[3] Castano S. and Antonellis V., “Semantic Dictionary Design for Database Interoperability”. In Proceedings of 13th International Conference on Data Engineering, pp. 43-54, 1997.

[4] Collet C., Huhns M.N. and Shen W.M., “Resource Integration Using a Large Knowledge Base in Carnot”. In IEEE Computer, Vol. 24, No. 12, pp. 55-62, 1991.

[5] Kelley W., Gala S., Kim W., Reyes T. and Graham B., “Schema Architecture of the UniSQL/M Multidatabase System”. In Modern Database Systems: The Object Model, Interoperability, and Beyond, editor Kim W., ACM Press, pp. 621-648, 1995.

[6] Kim W. and Seu J., “Classifying Schematic and Data Heterogeneity in Multidatabase Systems”. In IEEE Computer, Vol. 24, No. 12, pp. 12-18, 1991.

[7] Kim W., Choi I., Gala S.K. and Scheevel M., “On Resolving Schematic Heterogeneity in Multidatabase Systems”. In Distributed and Parallel Databases, Vol.1, No. 3, pp. 251-279, 1993.

[8] Levy A., “Obtaining Complete Answers from Incomplete Databases”. In Proceedings of 22nd International Conference on Very Large Data Bases, pp. 402-412, 1996.

[9] Li W.S. and Clifton C, “Semantic Integration in Heterogeneous Databases Using Neural Networks”. In Proceedings of 20th International Conference on Very Large Data Bases, pp. 1-12, 1994.

[10] Rafi A., Smedt P.D., Du W., Kent W., Ketabchi M.A., Litwin W.A., Rafii A., and Shan M.C., “The Pegasus Multidatabase System”. In IEEE Computer, Vol. 24, No. 12, pp. 19-27, 1991.

[11] Shet A., and Larson J.A., "Federated database systems for managing distributed, homogeneous, and autonomous databases". In ACM Computing Surveys, Vol.22, No. 3, pp. 183-236, 1990.