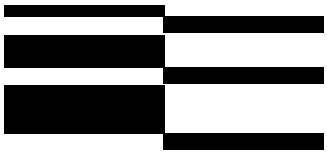
3



Semantic Similarities between Objects in Multiple Databases

Vipul Kashyap
Department of Computer Science
Rutgers University
New Brunswick, NJ 08903

Amit Sheth
LSDIS Lab, 415 GSRC
Department of Computer Science
University of Georgia
Athens, GA

Introduction

Many organizations face the challenge of interoperating among multiple independently developed database systems to perform critical functions. With high interconnectivity and access to many information sources, the primary issue in the future will not be how to efficiently process the data that is known to be relevant, but which data is relevant.

Some of the best known approaches to deal with multiple databases, discussed in Chapter 1 are tightly-coupled federation, loosely-coupled federation, and interdependent data management [Sheth and Larson, 1990; Sheth, 1991]. A critical task in creating a tightly-coupled federation is that of schema integration (e.g., [Dayal and Hwang, 1984]). A critical task in accessing data in a loosely-coupled federation [Litwin and Abdellatif, 1986; Heimbigner and McLeod, 1985] is to define a view over multiple databases or to define a query using a multidatabase language. In performing any of these critical tasks, and hence in any approach to interoperability of database systems, the fundamental question is that of identifying objects in different databases that are semantically related, and then resolve the schematic differences among semantically related objects. Classification taxonomies of *schematic differences* appear in [Dayal and Hwang, 1984; Breitbart *et al.*, 1986; Czejdo *et al.*, 1987; Krishnamurthy *et al.*, 1991; Kim and Seo, 1991] (See [Kashyap and Sheth] for a further discussion). However, purely schematic considerations do not suffice to determine the semantic similarity between objects [Fankhauser *et al.*, 1991; Sheth and Gala, 1989].

Wood [Wood, 1985] defines semantics to be "the scientific study of the relations between signs and symbols and what they denote or mean." We consider these to be aspects of *real world semantics* (RWS) of an object. In this chapter we focus on the techniques and representational constructs used by the various practitioners in the field of multidatabases. We also discuss techniques for representing uncertainty but only as an aspect of the semantic similarity between objects. We do not discuss attempts to represent uncertainty independent of the semantic similarity between objects. We shall also restrict ourselves to attempts to represent semantic similarity in the multidatabase context. We discuss related work in Section 3.8.

Attempts have been made to capture the similarity of objects by using mathematical tools like value mappings between domains and abstractions like generalization, aggregation, etc. However, it is our belief that the RWS of an object cannot be captured using mathematical formalisms. We need

to understand and represent more knowledge in order to capture the semantics of the relationships between the objects. The knowledge should be able to capture and the representation should be able to express the **context** of comparison of the objects, the **abstraction** relating the domains of the two objects and the **uncertainty** in the relationship between the objects.

In Section 3.1, we explore the above perspectives of semantics in detail and illustrate how the various researchers have tackled these issues in their attempts to represent semantic similarity. In Section 3.2 we discuss **semantic proximity** proposed by [Sheth and Kashyap, 1992] In Section 3.3 we discuss the **context building** approach taken by [Ouksel and Naiman, 1993]. In Section 3.4 we discuss the **context interchange** approach taken by [Sciore *et al.*, 1992]. In Section 3.5 we discuss the **common concepts** approach taken by [Yu *et al.*, 1991]. In Section 3.6 we discuss the **semantic abstractions** approach taken by [Garcia-Solaco *et al.*, 1993]. In Section 3.7 we discuss the utilization of **fuzzy terminological knowledge** by [Fankhauser *et al.*, 1991].

3.1 Semantics: Perspectives and Representation

In this section we relate attempts by various researchers to represent semantic similarities to the perspectives on semantics identified in the previous section.

3.1.1 Context: The semantic component

In this section we discuss the representation of context in which the objects are being compared. This in our opinion, provides the *semantic fulcrum* of capturing and representing the object similarities. The importance of context has been realized by researchers in the field of Heterogeneous Databases. Some of the proposals are as follows.

[Sheth and Kashyap, 1992] propose the concept of **semantic proximity** (Section 3.2) to characterize semantic similarity in which the context is the primary vehicle to capture the RWS.

[Ouksel and Naiman, 1993] propose a **dynamic context building** (Section 3.3) approach for meaningful information exchange between various information systems.

[Sciore *et al.*, 1992] propose an approach using **metadata** (Section 3.4) to represent context use metadata for "context mediation".

[Yu *et al.*, 1991] propose **common concepts** (Section 3.5) to characterize similarities between attributes in multiple databases. [Chierchia and McConnell-Ginet, 1990] propose that a concept may be considered to be the image of a function mapping contexts to propositions. Thus a context may be implicitly represented in the functional definitions of the concepts.

3.1.2 Abstractions/Mappings: The structural component

Abstraction here refers to the relation between the domains of the two objects. Mapping between the domains of objects is the mathematical tool used to express the abstractions. However, since abstractions by themselves cannot capture the semantic similarity, they have to be associated either with the context [Kashyap and Sheth] or with extra knowledge in order to capture the RWS. Some of the proposals are as follows.

[Sheth and Kashyap, 1992] define abstractions in terms of value mappings between the domains of objects and associate it with the context as a part of the semantic proximity (Section 3.2).

[Ouksel and Naiman, 1993] define mappings between schema elements which they term as *inter schema correspondence assertions* or ISCAs (Section 3.3). A set of ISCAs under consideration define the context for integration of the schemas.

[Sciore *et al.*, 1992] define mappings which they call *conversion functions* (Section 3.4) which are associated with the meta-attributes which define the context.

[Yu *et al.*, 1991] associate the attributes with "common concepts" (Section 3.5). Thus the mappings (relationship) between the attributes are determined through the extra knowledge associated with the concepts.

[Garcia-Solaco *et al.*, 1993] upgrade the semantic level of the schemas using abstractions which they call *semantic abstractions* (Section 3.6). This is achieved as a result of the knowledge acquisition process described in [Castellanos, 1993].

[Hammer and McLeod] discuss an approach for resolving the representational differences which involves: (a) determining as precisely as possible the relationships between sharable objects; and (b) detect possible conflicts in their structural representations.

3.1.3 Modeling Uncertainty, Inconsistency and Incompleteness

Understanding and representing semantic similarity between objects may involve understanding and modeling **uncertainty**, **inconsistency** and **incompleteness** of the information pertaining to the objects and the relationships between them modeled in the database (both at the intensional and extensional levels). Some proposals are as follows.

[Fankhauser *et al.*, 1991] propose an approach where **fuzzy terminological knowledge** is combined with schema knowledge (Section 3.7) to determine semantic similarity.

[Ouksel and Naiman, 1993] model uncertain information by using the degrees of likelihood of the various intermediate contexts. The Dempster-Shafer (D-S) theory of belief functions is used to model the likelihood of alternative contexts (Section 3.3).

[Sheth and Kashyap, 1992] have used semantic proximity as a basis for representing the uncertainty of the information modeled at the database level (Section 3.2). They propose a framework in which the semantic proximity can be mapped to fuzzy strengths comprising the fuzzy terminological knowledge or to the likelihood of the assertions comprising the context discussed above.

[Yu *et al.*, 1991] represent each attribute as a vector depending on the concepts associated with it. A similarity measure between two attributes is defined as function of the vectors associated with the attributes.

3.2 Semantic Proximity: A model for representing Semantic Similarities

We recognize three basic aspects in the representation of semantics. The first aspect concentrates on pinning down the **real world semantics** of the various entities and the relationships between them. The second aspect is to

represent all known knowledge about the domain to which the various entities and objects belong. The relationships between various objects can then be determined on the basis of the encoded domain knowledge. We consider this knowledge itself as an **implicit context**. The final aspect is when the context is represented **explicitly** and reasoned about as a first class construct.

We distinguish between the *real world*, and the *model world* which is a representation of the real world (See [Sheth *et al.*, 1993] and [Garcia-Solaco *et al.*] for further discussion). The term object in this chapter refers to an object in a model world (i.e., a representation or intensional definition in the model world, e.g., an object class definition in object-oriented models) as opposed to an entity or a concept in the real world. These objects may model information at any level of representation, viz. *attribute level* or *entity level*.[1]

We introduce the concept of *semantic proximity* to characterize semantic similarities between objects, and use it to provide a classification of semantic similarities between objects. Our approach embodies the explicit context representation approach. Given two objects O_1 and O_2, the *semantic proximity* (Figure 3.1) between them is defined by the 4-tuple given by

semPro(O_1, O_2) = <Context, Abstraction, (D_1, D_2), (S_1, S_2)>
where D_i is domain of O_i and S_i is state of O_i.

The context of an object is the primary vehicle to capture the RWS of the object. Thus, the respective contexts of the objects, and to a lesser extent the abstraction used to map the domains of the objects, help to capture the semantic aspect of the relationship between the two objects.

3.2.1 Context(s) of the two Objects: the semantic component

Each object has its own context. The term context in semPro refers to the context in which a particular semantic similarity holds. This context may be related to or different from the contexts in which the objects were defined. It is possible for two objects to be semantically closer in one context than in another context. Some of the alternatives for representing a context *in an interoperable database system* are as follows.

- In [Ouksel and Naiman, 1993], context is defined as the knowledge that is needed to reason about another system, for the purpose of answering

[1] Objects at the entity level can be denoted by single-place predicates P(x) and attributes can be denoted by two-place predicates Q(x,y) [Sheth and Gala, 1993].

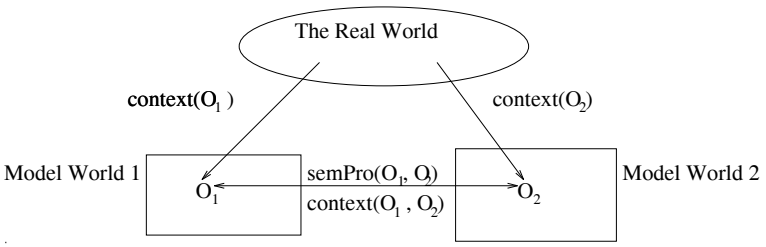


FIGURE 3.1
Semantic Proximity between two Objects

a query and is specified as a set of assertions.

- In [Sciore *et al.*, 1992], context is defined as the meaning, content, organization and properties of data and is modeled using metadata associated with the data.

- A context may also be associated with a **database** or a group of databases (e.g., the object is defined in the context of DB1).

- The **relationship** in which an entity participates may determine the context of the entity.

- From the five-level schema architecture for a federated database system [Sheth and Larson, 1990], a context can be specified in terms of an **export schema** (a context that is closer to a database) or an **external schema** (a context that is closer to an application).

- At a very elementary level, a context can be thought of as a **named collection** of the domains of the Objects.

- When using a well defined ontology, such as Cyc, a well defined partition (called Microtheory) of the ontology can be provided a context [Guha, 1990].

- Sometimes a context can be "hard-coded" into the definition of an object. For example, when we have the two entities EMPLOYEE and TELECOMM-EMPLOYEE, the TELECOMMUNICATIONS context is "hard-coded" in the second entity.

3.2.2 Issues of Representation and Reasoning

Various approaches have been proposed to represent semantic structures similar to context and reason with the help of these representations. We now discuss some of the approaches in the context of Multidatabase Systems.

[Kashyap and Sheth] have used context to provide an intensional descriptions of database objects. They represent contexts as a collection of contextual coordinates and values. A contextual coordinate denotes an aspect of context and models some characteristic of the subject domain and may be obtained from a domain specific ontology. Values can be a set of symbols, objects from a database or concepts from a domain specific ontology. Operations to compare the specificity of two contexts and to compute the greatest lower bound of two contexts are defined. These operations are used to reason with the intensional descriptions.

A similar approach has been adopted by [Sciore *et al.*, 1992] where the represent a context as a collection of meta-attributes and their values (Section 3.4). This representation of context is however at the level of data values and object instances. They are not able to to model constraints at an intensional level, viz., cardinality constraints. Context mediation is used for reasoning and is implemented using rules [Siegel and Madnick, 1991] and predicates in a relational model [Sciore *et al.*, 1992].

[Kashyap and Sheth] have illustrated how contextual descriptions may be expressed using description logic expressions. [Mena and Kashyap] have used CLASSIC [Borgida *et al.*, 1989] to represent the intensional descriptions and use subsumption reasoning to query the intensional descriptions at a *semantic level*.

[Ouksel and Naiman, 1993] represent as a collection of ISCAs (inter-schema correspondence assertions) which are correspondences between different data elements in different databases (Section 3.3). They use Assumption-based Truth Maintenance Systems [DeKleer, 1986] to reason with contexts as multiple sets of contexts can coexist in the absence of complete knowledge.

3.2.3 The Vocabulary Problem

In constructing contexts and intensional descriptions for modeling semantics, the choice of vocabulary is very important. It is important that we choose terms which are specific to the subject domain. Traditional Multidatabase approaches have utilized data dictionaries to enumerate the vocabulary used in the component databases. Researchers in Multidatabases are now using ontologies for building semantic descriptions. An ontology may be

defined as the specification of a representational vocabulary for a shared domain of discourse which may include definitions of classes, relations, functions and other objects [Gruber, 1993].

[Kashyap and Sheth] have illustrated how terminological relationships in ontologies enable representation of *extra information* in the contextual descriptions. [Daruwala *et al.*, 1995] have used an ontology from the financial domain to construct contexts. [Kashyap and Mena] have used ontologies belonging to the bibliographic information domain to construct description logic expressions. Concept hierarchies have also been used in the common concepts [Yu *et al.*, 1991] approach discussed in Section 3.5. Terminological relationships have been represented across ontologies in [Mena and Kashyap] and in [Hammer and McLeod, 1993] to handle cases where contexts or intensional descriptions may be constructed from different ontologies. An approach for resolving representational conflicts using terminological relationships is discussed in [Hammer and McLeod].

3.2.4 The Structural Components

Abstraction used to map the Objects

We use the term abstraction to refer to a mechanism used to map the domains of the objects to each other or to the domain of a common third object. Note that an abstraction by itself cannot capture the semantic similarity. Some of the more useful and well defined abstractions are:

Total 1-1 value mapping For every value in the domain of one object, there exists a value in the domain of the other object and vice versa.

Partial many-one mapping In this case some values in the domain of one of the objects might remain unmapped, or a value in one domain might be associated with many values in another domain.

Generalization/Specialization One domain can generalize/specialize the other, or domains of both the objects can be generalized/specialized to a third domain.

Aggregation One domain can be an aggregation or a collection of other domains.

Functional Dependencies The values of one domain might depend functionally on the other domain.

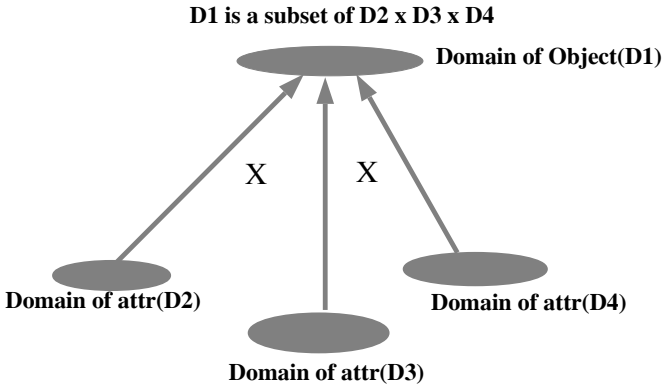


FIGURE 3.2
Domain of an Object and it's Attributes

ANY This is used to denote that any abstraction such as the ones defined above may be used to define a mapping between two objects.

NONE This is used to denote that there is no mapping defined between two semantically related objects.

Domains of the Objects

Domains refer to the sets of values from which the objects can take their values. When using an object-oriented model, the domains of objects can be thought of as types, whereas the collections of objects might themselves be thought of as classes. A domain can be either **atomic** (i.e., cannot be decomposed any further) or composed of other atomic or composite domains. The domain of an object can be thought of as a subset of the cross-product of the domains of the properties of the object. Analogously, we can have other combinations of domains, viz. union and intersection of domains.

An important distinction between a context and a domain should be noted. One of the ways to specify a context is as a named collection of the domains of objects, i.e. it is associated with a group of objects. A domain on the other hand is a property of an object and is associated with the description of that object.

States (extensions) of the Objects

The state of an object can be thought of as an extension of an object recorded in a database or databases. However, this extension must not be confused with the actual state of the entity (according to the Real World Semantics) being modeled. Two objects having different extensions can have the same state Real World Semantics (and hence be semantically equivalent).

3.2.5 Modeling Uncertainty: Fuzzy Strengths as a function of Semantic Proximity

In this section we establish the semantic proximity as a basis for the assignment of fuzzy strengths to the terminological relationships between two semantically similar objects. As noted in the previous section, when we assign fuzzy strengths to semantic similarities between schema objects, they should reflect the Real World Semantics. Thus any such assignment of belief measures should depend on and reflect:

- The **context(s)** to which the two schema objects belong to.
- The **mapping(s)** which may exist between the domains of the objects or the domains of the individual attributes of the objects. Here, it may be noted that the mappings between two attributes of the objects might not be independent of each other, but may be dependent on each other.
- The **state(s)** or the extensions of the two objects.

The semantic proximity described in the previous section is able to capture this information which represents the semantic similarity between two objects according to the Real World Semantics. Also the interactions between any two attributes of an object can be captured using the interactions between the mappings of the two attributes, thus avoiding the need for the implicit independence assumption.

We define an **uncertainty function** μ between two objects O_1 and O_2 which maps the semantic proximity to the real interval [0,1]. Thus

μ : semPro(O_1, O_2) $\rightarrow$ [0,1]
i.e., μ(Context, Abstraction, (D_1, D_2), (S_1, S_2)) = X where $0 \leq X \leq 1$.

μ is a **user defined** function such that it accurately reflects the Real World Semantics and may not have specific mathematical properties. It may or may not be a computable function.

3.2.6 A semantic classification of object similarities

Our emphasis is on identifying semantic similarity independent of the representation of the objects. The concept of *semantic proximity* defined earlier provides a qualitative measure to classify the semantic similarities between objects.

The role of Context in Semantic Classification

A partial context specification can be used by humans to decide whether the context for modeling of two objects is the same or different, and whether the comparison of semantic similarity of objects is valid in all possible contexts or specific ones. A more detailed discussion of the nature of context, the operations on contexts, and the relationship of context to semantics is provided in [Kashyap and Sheth].

ALL The semPro between the objects is being defined *wrt* all known and *coherent*[2] comparison contexts. There should be coherence between the definition contexts of the objects being compared and between the definition contexts and the context of comparison.

SOME The semPro between the objects is being defined *wrt* some context. This context may be constructed in the following ways.

- **GLB** The greatest lower bound of the contexts of the two objects. Typically we are interested in the *glb* of the context of comparison and the definition context of the object.
- **LUB** The least upper bound of the contexts of the two objects. Typically, we are interested in the *lub* of the definition contexts of the two objects when there does not exist an abstraction/mapping between their domains in the context of comparison.

SUB-CONTEXTS We might be interested in the **semPro** between two objects in contexts which are more specific or more general *wrt* the context of comparison.

NONE There doesn't exist a context in which a meaningful abstraction or mapping between the domains of the objects may be defined. This is the case when the definition contexts of the objects being compared are *not coherent* with each other.

[2] Intuitively, coherent contexts are such that the constraints used to express them are not inconsistent with each other. The reader may refer to [Kashyap and Sheth] for a formal exposition.

Semantic Equivalence

This is the strongest measure of semantic proximity two objects can have. Two objects are defined to be *semantically equivalent* when they represent the same real world entity or concept. Expressed in our model, it means that given two objects O_1 and O_2, it should be possible to define a total 1-1 value mapping between the domains of these two objects in any known and coherent context. Thus we can write it as:

semPro(O_1, O_2) = <ALL, total 1-1 value mapping, (D_1, D_2), _>[3]

The notion of equivalence described above depends on the definition of the domains of the objects and can be more specifically called *domain semantic equivalence*. We can also define a stronger notion of semantic equivalence between two objects which incorporates the state of the databases to which the two objects belong. This equivalence is called *state semantic equivalence* and is defined as:

semPro(O_1, O_2) = <ALL, M, (D_1, D_2), (S_1, S_2) >
where M is a total 1-1 value mapping between (D_1, S_1) and (D_2, S_2).

Semantic Relationship

This type of semantic similarity is weaker than semantic equivalence. Two objects are said to be *semantically related* when there exists a partial many-one value mapping, or a generalization, or aggregation abstraction between the domains of the two objects. Here we relax the requirement of a 1-1 mapping in a way that given an instance O_1 we can identify an instance of O_2 but not vice versa. The requirement that the mapping be definable in all the known and coherent contexts is not relaxed. Thus we define the *semantic relationship* as:

semPro(O_1, O_2) = <ALL, M, (D_1, D_2), _>
where M may be a partial many-one value mapping, generalization, or aggregation

Semantic Relevance

We consider two objects to be *semantically relevant* if they can be related to each other using some *abstraction* in *some context*. Thus the notion of

[3] We use the "_" sign to denote don't care.

semantic relevance between two objects is context dependent, i.e., two objects may be semantically relevant in one context, but not so in another. Objects can be related to each other using any abstraction.

semPro(O_1, O_2) = <SOME, ANY, (D_1, D_2), _>

Semantic Resemblance

This is the weakest measure of semantic proximity, which might be useful in certain cases. Here, we consider the case where the domains of two objects cannot be related to each other by any abstraction in any context. Hence, the exact nature of semantic proximity between two objects is very difficult to specify. In this case, the user may be presented with extensions of both the objects. In order to express this type of semantic similarity, an aspect of context, called **role** is introduced and semantic resemblance is expressed with its help. A detailed discussion on roles and semantic resemblance is presented in [Sheth and Kashyap.

semPro(O_1, O_2) = <SOME(LUB), NONE, (D_1, D_2), _>
where coherent($C_{def}(O_1)$,$C_{def}(O_2)$) and $\exists$Cntxt$_1$,Cntxt$_2$ exported by DB$_1$,DB$_2$ respectively
and SOME(LUB) denotes a context defined as follows:
context = lub(Cntxt$_1$, Cntxt$_2$) and $D_1 \neq D_2$
and role-of(O_1, context) = role-of(O_2, context)

Semantic Incompatibility

While all the qualitative proximity measures defined above describe semantic similarity, semantic incompatibility asserts semantic dissimilarity. Lack of any semantic similarity does not automatically imply that the objects are semantically incompatible. Establishing semantic incompatibility requires asserting that the definition contexts of the two objects are *incoherent wrt* each other and there do not exist contexts associated with these objects such that they have the same role.

semPro(O_1, O_2) = <NONE, NONE, (D_1, D_2), _>
where $C_{def}(O_1)$ and $C_{def}(O_2)$ are incoherent with each other
and D_1 may or may not be equal to D_2
and $\nexists$ context such that role-of(O_1, context) = role-of(O_2, context)

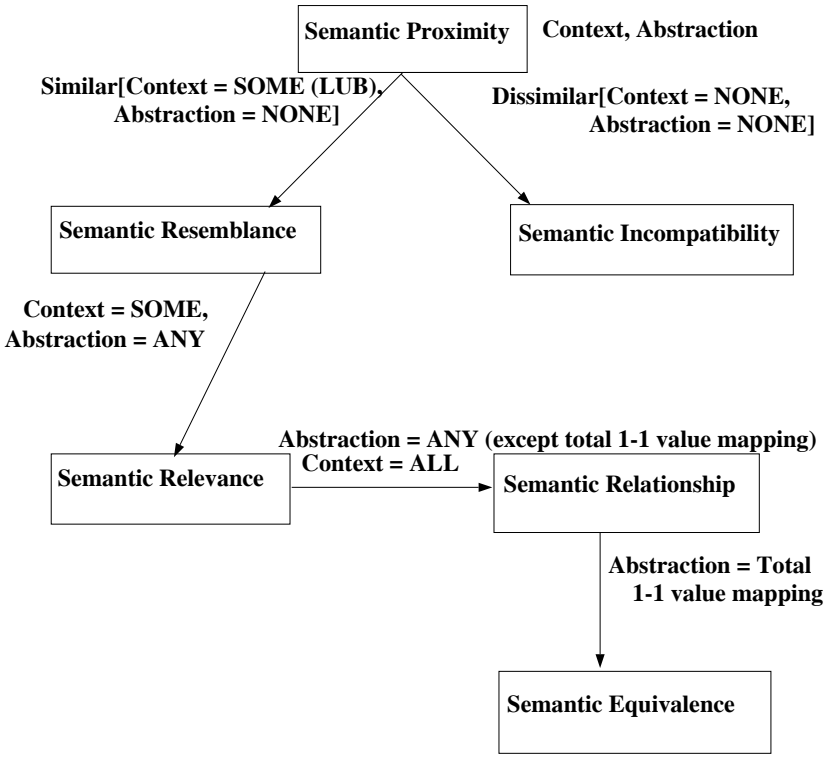


FIGURE 3.3
Semantic Classification of Object Similarities

3.3 The Context Building Approach

The approach followed by [Ouksel and Naiman, 1993] is centered on the construction and maintenance of the **context**, within which meaningful information between heterogeneous information systems is proposed to be exchanged. The context is defined as a set of *inter schema correspondence assertions* (ISCAs). Each ISCA consists of three dimensions, viz., *Naming*, *Abstraction* and *Level of Heterogeneity*.

The mathematical formalism used to model the structural similarity is expressed as the last two dimensions of the ISCA, viz., Abstraction and Level of Heterogeneity. The association between the abstraction and the context is achieved through the ISCAs comprising the appropriate context. A classification of semantic conflicts is used as a basis on which to build and refine the context, by discovering the ISCAs between corresponding elements of the component systems. A truth management system is used to manage the multiple intermediate contexts. Uncertainty in modeling the information is present in the degrees of likelihood of the various intermediate contexts. The Dempster-Shafer (D-S) theory of belief functions is used to model the likelihood of alternative contexts.

Context is described here as the knowledge that is needed to reason about another system, for the purpose of answering a specific query. The context must provide an easily understood representation of how much is known and what is still needed in order to answer the query. The following (partial) schemas and the following query is used to elaborate on the issues of context building.

Example:

Database: A database of a cold cereal manufacturer
Data model: Relational
Schema: COMPETITOR(**Name**, Location, Mkt_Sgmt, Mkt_Shr, Qtly_Income)

Database: Financial information of companies in the European Community
Data model: Relational
Schema: COMPANY(**Rank**, Co-Name, Location, Industry)

Query: "What is the performance of my competitors in Europe ?"

3.3.1 Context Dependent Interpretation

Different contexts can lend different interpretations to a schematic conflict. The processes of detection of conflicts and integration help dynamically build a context. The query itself may be important in providing a context for the meaningful interpretation of schematic elements. They consider examples based on the example schema and the query shown above which illustrate the above ideas.

Example(s):

Hypothesis: COMPETITOR.Name and PRODUCT.Name appear to be synonyms.
Context: Suppose COMPETITOR and PRODUCT are unrelated, say homonyms.
Interpretation: We may most likely conclude with a high probability that COMPETITOR.Name and PRODUCT.Name are homonyms.

Hypothesis: COMPETITOR.Name and COMPANY.Co-Name appear to be related.
Context: Suppose the term "competitor" is the specialization of the term "company". The query specifies "competitor" and provides a context. Interpretation: We may conclude that COMPETITOR.Name and COMPANY.Co-Name are synonyms.

Organization of Context by levels of Heterogeneity

The object level is considered to be a coarser or higher level of heterogeneity than the attribute level, which is coarser than the instance level. In addition to structural schematic levels of heterogeneity, there are *metadata* levels of heterogeneity. These include the differences that require knowledge describing the objects and the attributes of the schema (commonly called *descriptive metadata*), knowledge of the semantics inherent, implicit and explicit in data models, and general or domain-specific knowledge available about the database itself. Concepts at the metadata level could be objects, attributes or instances.

Example(s):

The *Inter Schema Correspondence Assertion* (ISCA) that COMPETITOR.Name and COMPANY.Co-Name are corresponding attributes, *triggers* an attempt

to place them in the context of the respective objects they describe. This may result in the inference of an ISCA that COMPETITOR and COMPANY are corresponding objects. This is called *upward propagation* through the levels of heterogeneity.

If there is enough evidence to refute the synonymy of COMPETITOR and COMPANY this would trigger a *downward propagation* to reclassify the relationship of COMPETITOR.Name and COMPANY.Co-Name as homonyms.

The semantic knowledge of another system, organized by levels of heterogeneity, provides a context for interpreting the view of the system.

Classification and Representation of Semantic Conflicts

The classification and representation of semantic conflicts is the fundamental building block of dynamic context building. Conflicts are classified along three dimensions of naming, abstraction and level of heterogeneity. The semantic relationship between two elements of different databases is represented as an inter-schema correspondence assertion as follows:

Assert[x,y](naming, abstraction, heterogeneity)

The first two dimensions (naming and abstraction) include what they view as fundamental relationships between semantic concepts. The third dimension (level of heterogeneity) is needed in order to place the semantic relationship in the appropriate schematic context.

Naming *Naming Conflicts* refer to the relationship of the object, attribute or instance names. These conflicts include *synonyms* and *homonyms*. If a conflict cannot be categorized as either a synonym or a homonym, it is classified as *unrelated*.

Abstraction A conflict can involve objects that refer to the same *class* of objects, objects that represent similar semantic concepts at different levels of abstraction (*generalization/specialization*), an object that maps to a group of objects in another database (*aggregation/part-of*), or an incompatibility that occurs when one object can be mapped to another through a *computed or derived function*.

Levels of Heterogeneity Naming and abstraction conflicts can exist at the object, attribute and instance levels of the schema.

Example:

The conflict between COMPANY.Co-Name and COMPETITOR.Name can be classified as:

Assert[COMPANY.Co-Name, COMPETITOR.Name](synonyms, generalization,
(attribute, attribute))

Assertions that have been inferred through upward propagation require reconciliation to fill the slots of the classification.

Assert[COMPANY, COMPETITOR](?, ?, (object, object))

3.3.2 Context Management using a Truth Maintenance System

Let us compare the attributes COMPETITOR.Location and COMPANY.Location. It can be reasonably proposed that the attributes are synonyms or that they are homonyms. These two assertions are contradictory to each other, yet they are temporarily both compatible with the evidence (or the lack of it). A truth maintenance system is used for:

- Handling the effect of retracting assumptions when they are invalidated by the evidence and to keep track of the multiple plausible sets of assertions which can coexist in the absence of complete knowledge. The Assumption-based Truth Maintenance System (ATMS) [DeKleer, 1986] is used.
- Providing a symbolic mechanism for identifying the set of assumptions needed to assemble the desired proofs, so when probabilities are assigned to these assumptions, the system can be used as symbolic engines for computing the degrees of belief sought by the D-S theory [Dempster, 1968; Shafer, 1976].

Multiple Context Management

At any given moment, in systems based on this approach, an assertion is either believed to be true (*Confirmed*), believed to be false (*Retracted*), or not believed to be either (*Undetermined*). A value *Undetermined* is assigned when, based on the current evidence, A and not A coexist in the set of assertions.

A context is a set of consistent assertions. Therefore, these assertions must all be *Confirmed*. An *Undetermined* may be part of a context so long as its contradiction is not. Several contexts may co-exist at any given point in time. The sets of assertions in the various contexts may be mutually exclusive or overlapping.

Each set of assertions is managed by an ATMS which provides a mechanism for keeping track of a context of a context of noncontradictory plausible assertions and the evidence developed during an inference session. An ATMS is able to retract some of the plausible assumptions in light of new information. All conclusions that were derived from these assumptions have to be retracted as well. The degrees of belief of the assertions associated with a context determine whether it is chosen for further refinement.

Assigning degrees of belief to assertions

The D-S approach considers a set of assertions and assigns to each of them, an interval [B, Pl] in which the degree of belief must lie. B measures the strength of evidence in favor of a set of assertions. Plausibility measures the probability that an assertion is compatible with the evidence, i.e., the probability that it cannot be disproved and is therefore possible. Initially, the exhaustive universe of initially exclusive assertions Ω is considered. This is referred to as the *frame of discernment*. However, not all evidence is directly supportive of individual elements. Generally it supports subsets of the frame of discernment.

The D-S theory defines a probability density function, which is denoted by d. The function d is defined for all subsets of Ω. The quantity d(q) measures the amount of belief that is currently assigned to exactly the set q of assertions. The values of d are assigned so that the sum of all degrees of belief of all subsets of Ω is 1.

Typically the context with the highest degree of belief is selected for further refinement. A context is essentially a set of assertions which would be some subset of Ω. Typically, in a dynamically evolving system, there would be several pieces of evidence for a set of assertions q associated with the context. The D-S theory provides a simple mechanism for computing the degrees of belief of all intersections of current frames of discernment. This can be used to combine the various pieces of evidence in support of the context.

Discussion

The context defined in this approach embodies both the semantic and the structural components. As defined before, the context is a set of ISCAs which

are used to direct the process of dynamic schema integration. The semantic component in the ISCA is the naming dimension which corresponds to the *real world semantics aspect* of representation of semantics. The other components viz., abstraction and level of heterogeneity are structural in nature. Thus the context is a partly semantic and partly structural entity. This is in contrast to the semantic proximity approach where the context is a purely semantic entity and has structure associated with it as abstractions.

3.4 The Context Interchange Approach

[Sciore *et al.*, 1992] discuss the problem of transfer of data from one environment to another and its manipulation in another environment. They use the **context** to deal with the meaning, content, organization and properties of data and to model the environment to which the data belongs. The context of the data is essentially a collection of meta-attributes which model the **metadata** associated with the data. The mathematical formalism used to model the structural similarity is expressed using **conversion functions**. These conversion functions are used to translate data from one context to another, thus achieving **context interchange**. The association between the mappings (conversion functions) and the context is achieved by defining a conversion function for each meta-attribute which may be a part of the definition of some context.

3.4.1 Context and Metadata

The definition of context in [Sciore *et al.*, 1992] is guided by the requirements for effective data interchange among information systems. A data source's metadata defines the *export context*, and consists of stored information or rules describing the data provided by the source. A data receiver's metadata is called its *import context*, and consists of predicates describing the properties of the data expected by the receiver.

Example:
A data source provides trade prices for stocks on the NYSE. The context of a trade price value might be the latest trade price in US dollars. Data from this environment may be moved to the environment of a Japanese company where the context might be the latest trade price in Yen.

During this data interchange, the export context of the source is compared to the import context of the receiver to determine if the data is meaningful to the receiver. Thus it is necessary to represent context using context specifications and comparing those specifications to achieve *context mediation*. The approach adopted in [Sciore *et al.*, 1992] is to represent all context from the data environment at the attribute level, using meta-attributes. Meta-attributes are attributes which have a special relationship to the attributes whose context they define. For the financial data source discussed in the preceding example, *Currency* and *PriceStatus* would be meta-attributes of the *TradePrice* attribute.

Example
Consider the following schema for the financial data source:
TRADES(InstrumentType, CompanyName, Exchange,
TradePrice(PriceStatus, Currency))
An example tuple in the above schema can be given by:
<'equity', 'IBM', 'NYSE', 89.25('latestTradePrice', 'USDollars')>

3.4.2 Data Conversion

Conversion Functions

A *conversion function* changes values of a tuple's attributes and meta-attributes, in a way that leaves the "meaning" of the information unchanged. Each meta-attribute has a conversion function defined for it; by default, the conversion function for M is called cvtM. For example, suppose that a tuple in the TRADES relation has a TradePrice value of 3 and a TradePrice.Currency value of 'USDollars'. The conversion function call *cvtTradePrice*(3, 'USDollars, 'Yen') returns the value 330 (assuming 1 USDollar = 110 Yen). The function *cvtTradePrice* is an example of a *total* and *lossless* conversion function. A total function is one which is defined for all arguments. A lossless function is one for which every conversion has an inverse. Not all functions may have these features.

Context Mediation

The *context mediator* component examines each incoming tuple and attempts to build a new conversion function that will produce a tuple that conforms to the receiver's assumptions. The receiver's assumptions about the incoming tuple are described by the receiver's *import context* - a predicate on the tuple's attribute values. The job of the context mediator is to re-

turn a conversion function that maps the input tuples to the receiver's import context. This function is used to define a relational operator called *cvtContext*.

Example:
Suppose the TRADES table contains values for the attribute TradePrice with TradePrice.Currency = 'USDollars'
Suppose the receiver context is TradeCurrency = 'Yen'
The conversion function used will be:
cvtTradePrice(X, 'USDollars', 'Yen')
The relational operator can be defined using the above conversion function:
cvtContext(TRADES, TradePrice.Currency = 'Yen')

Discussion

The context defined in this approach is essentially a semantic entity on the lines of the semantic proximity approach. Here an attempt is made to give an explicit (partial) representation of context by using metadata. A context is defined as a collection of meta-attributes and values. This corresponds to the *explicit context* aspect of the representation of semantics. There is also the association of the structural component (conversion functions) with the semantic component (context) similar to the semantic proximity approach.

3.5 Common Concepts: An approach to determine attribute similarities

[Yu *et al.*, 1991] seek to capture the denotation or meaning of the attributes to some extent by using *common concepts, concept hierarchies*, and *aggregate concept hierarchies*. It is our opinion based on Chierchia and McConnell-Ginet's definition of a linguistic concept [Chierchia and McConnell-Ginet, 1990], that the **context** is considered to be implicitly represented in the functional definition of the concepts. The attributes are associated with the common concepts. Thus the mappings (relationships) between attributes are determined through their association with the extra knowledge or the implicit context embodied in the common concepts. Uncertainty in the relationship between two attributes is modeled using **similarity values**. Each attribute is defined as a vector depending on the concepts associated with it. The similarity value between two attributes is a function of the vectors associated with the attributes.

3.5.1 Representation of Attribute Semantics by Common Concepts

Each attribute can be characterized by a set of *common concepts*. Each concept represents a certain characteristic or a property that may be possessed by many objects (physical and/or abstract objects). These concepts are generic i.e., they are not application dependent[4].

Example:
The common concepts *horizontal_position* and *vertical_position* represent the horizontal and vertical position of a point respectively.

The common concept *identification* can be associated with the attributes **sensor**, **mission** and **platform** in the NASA databases as these attributes have the property of identifying object.

Concepts may have hierarchical relationships with other concepts. Two relationships of particular significance are: *aggregate concept hierarchies* and *IS-A concept hierarchies*.

Aggregate Concept Hierarchies

The example of the concept *time_interval* is considered, which specifies a duration of time, could be modeled as an aggregate concept of the concepts *begin_time* and *end_time*.

time_interval = aggregate(*begin_time*, *end_time*)

The attributes **tpstart** and **tpstop** can be characterized by *time_interval.begin_time* and *time_interval.end_time*, respectively. The two attributes are recognized to be an aggregate attribute which are characterized by the aggregate concept *time_interval*.

Consider the possible attribute similarity between (**start_date**, **length**) and (**tpstart**, **tpstop**). There are two levels of possible similarity:

Individual similarity: There is a similarity between **start_date** and **tpstart** as they are associated with the concept *time_interval.begin_time*.

Aggregate similarity: There is a similarity between (**start_date**, **length**) and (**tpstart**, **tpstop**) as they are associated with the concept *time_interval*.

[4] Discussion at the end of this section gives our view on this assumption.

It should be noted that a similarity on the basis of aggregate attributes does not necessarily imply that there exists a similarity among individual attributes involved in the aggregate attributes. The attribute **start_date** is associated with *time_interval.begin_time*, but the attribute **length** is not associated with *time_interval.end_time*. For each aggregate concept in a hierarchy, there is a component concept labeled "others" which illustrates the incompleteness of any pre-existing hierarchy and enables the user to specify other component concepts.

IS-A Concept Hierarchies

The IS-A concept hierarchy is in fact the generalization/specialization used in most semantic data models and object-oriented data models. If objects characterized by concept X is a subset of objects characterized by concept Y, then we call Y the generalization of X (equivalently X is the specialization of Y). X is also called a *subconcept* of Y, and Y is called the *superconcept* of X.

Example:
The concept *square* is a subconcept of *rectangle*, which in turn is a subconcept of *quadrangle*.

3.5.2 Establishing Attribute Relationships

Let C be a n-component vector, where n is the total number of concepts in the system, and C(i), $1 \leq i \leq n$, denote its i^{th} component of C, namely the i^{th} concept. The association of a set of concepts with an attribute X, can be represented by an n-component vector C_X, where $C_X(i) = 0$ if concept C(i) is not associated with X; otherwise $C_X = 1$.

Aggregation between Attributes

A set of attributes forms an aggregate attribute if:

1. Their vectors agree on all the concepts in the hierarchy except one, and
2. In that hierarchy, they correspond to all the component concepts of an aggregate concept, with the possible exception of one component.

Example:
The attribute **tpstart** is associated with the concepts *time_interval* and *time_interval.begin_time*, and the attribute **tpstop** is associated with the concepts *time_interval* and *time_interval.end_time*. Thus, they disagree in the component concepts *time_interval.begin_time* and *time_interval.end_time*, while

these two component concepts are all the components of the aggregate concept *time_interval*.

Attribute Similarities

Given two vectors, C_X and C_Y, of two (aggregate or individual) attributes X and Y, a similarity function can be defined as follows:

$$sim(C_X, C_Y) = \frac{C_X \bullet C_Y}{\sqrt{|C_X| \times |C_Y|}}$$

where $\bullet$ is the dot product of two vectors and |Z| is the number of 1 components in the vector Z. The above similarity function between two vectors is similar to that used in information retrieval [Salton, 1989]. The intuition of the dot product is that if more components of the two vectors agree then the more similar the two attributes are. Clearly if the two vectors are identical ($sim = 1.0$), then the attributes are said to be equivalent.

One of the advantages of using a similarity function is that some attributes that are related but not equivalent, i.e., "similar", may also be revealed. Suppose the attributes **income** and **salary** are not equivalent but closely related. It can be expected that the similarity between these two vectors would be very high.

Discussion

The common concepts defined in this approach may be application independent to some extent, but cannot be context independent. In some cases applications might belong to the same domain and hence it might be possible to define concept hierarchies in application independent manner. However, we believe that in the case where the concepts span more than one domain of discourse the concepts encode some form of the domain context in their functional definitions. This illustrates the *implicit context* aspect of representations of semantics in contrast to the *explicit context* aspect adopted in the semantic proximity approach.

3.6 The Semantic Abstractions Approach

The integration of database schemas into a federated one involves determining similarities between the classes of different databases in order to determine their semantic relationships. [Garcia-Solaco *et al.*, 1993] adopt an

approach of upgrading the semantic level of the local schemas is adopted. The abstractions (called semantic abstractions) used to model the structural similarities play a key role in this approach. Context is not represented or used explicitly in this approach. However these abstractions are the result of the knowledge acquisition process described in [Castellanos, 1993] and have extra knowledge associated with them. The search of the comparison process is guided by the structure of the generalization/specialization semi-lattices and aggregation graphs of the resulting rich schemas.

3.6.1 The Semantic Enrichment Phase

The choice of the canonical model must be rich enough to model the semantics already expressed in the local schemas, as well as the semantics obtained from a semantic enrichment process to upgrade the semantic level of the schemas. This would help to detect easily the similarities between classes and specify interdatabase semantic relationships. The canonical model chosen is BLOOM [Castellanos *et al.*, 1991] and it can represent the following abstractions which allows it to capture a rich set of semantic relationships:

- *Classification/Instantiation*
- *Generalization/Specialization*
  - *Disjoint Specialization* Each object of the superclass belongs to at most one subclass.
  - *Complementary Specialization* Each object of the superclass belongs at least to one subclass.
  - *Alternative Specialization* Each object belongs to exactly one subclass.
  - *General Specialization* No restrictions.
- *Aggregation*
  - *Simple Aggregation* The attributes are used to express the properties of the object.
  - *Collection Aggregation* The collection of a given class of objects gives rise to a new complex object.
  - *Association Aggregation* The component objects are not simply properties of the aggregate but it is their association which gives rise to it.

- *Existence and Interest Dependencies*

The Knowledge Acquisition Phase

In this stage the relational databases are analyzed both in extension and intension in order to discover hidden semantics in the form of restrictions like keys and different types of dependencies. This is the extra knowledge associated with and used to define the abstractions discussed later in this section. This is done through a series of steps explained in [Castellanos, 1993].

1. *Key Inference*
2. *Functional Dependency Inference*
3. *Normalization*
4. *Determining Identifier Type*
5. *Inclusion Dependency Inference*
6. *Inferring Exclusion and Complementariness Dependencies*

The Schema Conversion Phase

Once the relational model has been augmented with the knowledge extracted in the previous stage, they are converted to rich BLOOM schemas through a series of steps enumerated in [Garcia-Solaco *et al.*, 1993; Castellanos, 1993].

1. *Analysis of Inclusion Dependencies*
   (a) *Detection of Missing Entities*
   (b) *Identification of Semantic Abstractions*
2. *Processing of Remaining Attributes*
3. *Creation of Classes*

3.6.2 The Detection Phase

The similarities that exist among the classes of the component schemas are discovered in order to determine their semantic relationships. The strategy proposed to find out resemblance between classes has two components:

- *Which* pairs of classes to compare at each moment.
- *Criterion* to determine how similar the classes are.

The authors take advantage of the fact that the canonical model clearly distinguishes not only the generalization and the aggregation dimensions, but even different kinds of abstractions in each dimension.

The strategy is guided, at a coarse level by the generalization/specialization semi-lattices of the BLOOM schemas. Specialization are analyzed from the more restrictive to the more permissive. If no similar classes are found, they can be *relaxed* to a less restrictive kind (alternative, disjoint, complementary and general, in that order).

At a finer level, the aggregation dimension makes possible to establish an *order of comparison of classes* according to their aggregation abstractions. Thus, the comparison of the classes of two groups begins with those that involve the same kind of relevant abstraction (first collections and then associations). Only if no similarities are found, or if there is no class with the same relevant abstraction in the other group, then relaxation is applied on the kind of aggregation to continue the search for a similar class. With this approach, the more promising comparisons are identified first.

The criteria for similarity between a pair of classes is provided by the aggregation dimension. On one side it indicates at every moment which *aspects* of the classes are to be compared, and on the other side it indicates *how similar* the classes are.

The process of detection cannot be completely automated. Even in the case a strong similarity is found, the human integrator has to confirm it as it is based on structural similarity. This approach is based on classes and not on attributes as classes have a clearer meaning than attributes and there are much fewer classes than attributes.

Discussion

The critical component in enriching the local schema is the knowledge acquisition phase. In the knowledge acquisition phase the local schema is analyzed to infer knowledge about the database, viz. keys, functional dependencies, inclusion and exclusion dependencies, etc. This may be viewed as an

attempt to infer and represent the knowledge about the domain to which the entities in the local database belongs to. This knowledge can be considered as the *implicit context* wrt which the semantic abstractions are defined in the enriched schema. This is in contrast to the *explicit context* aspect adopted in the semantic proximity approach.

3.7 Semantic Similarity based on Fuzzy Knowledge

The approach adopted by [Fankhauser *et al.*, 1991] to determine the similarity of classes utilizes *fuzzy* and *incomplete* terminological knowledge together with schema knowledge. There is a clear distinction between *semantic* similarity determining the degree of resemblance according to real world semantics, and *structural* correspondence explaining *how* classes can actually be interrelated. They do not represent or use context explicitly in this approach. However, the structural schema knowledge is combined with fuzzy terminological knowledge to determine *semantic similarity*. The notion of *semantic relevance* is introduced and fuzzy set theory is applied to reason both about terminological knowledge and schema knowledge.

3.7.1 Terminological Knowledge

It is assumes that some classes or some of their attributes and/or relationships are assigned meaningful names in a preintegration phase. Thus the knowledge about the *terminological relationship* between the names can indicate the real world correspondence between the classes. The terms are related by three kinds of binary relationships which are represented in an associative network.

generalization/specialization: A term a is related by generalization to term b if a comprises b in a taxonomic sense (e.g., *Person, Employee*) or partitive sense (e.g., *Name, FirstName*).

negative association: Terms are related by negative association if they are complementary (e.g., *Man, Woman*), incompatible (e.g., *isEmployed, counsel*) or antonyms (e.g., *small, big*).

positive association: This is the most general relationship. It relates terms, which are synonyms in some context (e.g., *Title, Heading*), and terms that are typically used in the *same* context (e.g., *Letter, Address*).

Since the real world semantics of terms can vary, the relationships are *fuzzy* [Zadeh, 1978] i.e., they have a strength out of [0,1] assigned. It is not possible to associate all terms with each other. Relationships that are not explicitly specified are inferred by traversing the associative networks. The paths traversed in the network are investigated in the order of their strength.

The *kind* of a path depends on the component relationships in the path. Positive association, generalization and specialization are *transitive*, thus their composition forms the same kind of relationship again. *Anti-transitive* negative associations are not composed at all. Specialization composed with (its *inverse*) generalization results in a (least specific) positive association. The kind of a positive (negative) association composed with a generalization (specialization) depends on the existence of other (possibly weaker) paths between two terms.

Example:
Consider the two paths in an associative network:
1. *Instructor* p *Student* g *Person*
2. *Instructor* g *Employee* g *Person*

The weaker of the two paths are chosen (2) and hence the relationship between *Instructor* and *Person* is that of generalization.

The *strength* of a path depends on the degree of dependence between its relationships. The strengths of the relationships are combined using different triangular norms (t-norms). The authors restrict themselves to the following t-norms:

$\tau_1(\alpha, \beta) = max(0, \alpha + \beta - 1)$
$\tau_2(\alpha, \beta) = \alpha\beta$
$\tau_3(\alpha, \beta) = min(\alpha, \beta)$
where $\tau_1(\alpha, \beta) \leq \tau_2(\alpha, \beta) \leq \tau_3(\alpha, \beta)$ for all $0 \leq \alpha, \beta \leq 1$

The most pessimistic t-norm τ_1 is used to compose specialization with generalization or vice versa. The neutral t-norm τ_2 is used to compose positive (negative) associations with specialization and generalization. The optimistic τ_3 norm is used to compose the transitive relationships generalization and specialization, respectively, with themselves.

3.7.2 Semantic Similarity of Classes

The authors propose to consider semantic knowledge to achieve a more intuitive notion of similarity. The main consideration for this purpose is the *semantic relevance* of an attribute to a class. The attribute of an object may not be semantically relevant to the object. The semantic relevance can be expressed by a positive association or by a generalization. The semantically relevant parts of the structure of classes has also to be considered. This can be achieved by matching classes only via attributes that are relevant to at least one of the classes. We consider an example discussed in [Fankhauser *et al.*, 1991] and the strategy to determine semantic similarity.

Example:
Consider the following schema:
CarOwner: [Name: string, Address: string, Cars: string]
Letter: [From: Person, To: Person, Text: string]
Person: [Name: string, Address: string, Send: Letter, Receive: Letter]

Consider the following terminological knowledge base:
Send g Communicate, strength = 0.8
Receive g Communicate, strength = 0.6
Communicate p Person, strength = 0.6
Address p Letter, strength = 0.8

The strategy consists of the following three steps:

1. *Collect the Semantic Aspects:*
   An attribute is relevant to C, either if there exists a (derived) terminological relationship between the attribute and C directly, or if the attribute is semantically relevant to a class C' which can be reached via a (schema) relationship that is relevant to C. In the latter case, the relevance of the attribute to C' is composed with the relevance of the (schema) relationship between C and C' by the neutral t-norm τ_2.

   $\text{CarOwner}_{sem} = \{\}$
   $\text{Person}_{sem} = \{(0.48, \text{Send}), (0.48, \text{Receive}), (0.38, \text{Address})\}$

   Send g Communicate g Person, Strength = 0.6*0.8 = 0.48
   Receive g Communicate g Person, Strength = 0.6*0.8 = 0.48
   Address p Letter, Letter is related to Person via Send and Receive

Strength = max(0.48, 0.48)*0.8 = 0.38

2. *Collect the Structural Aspects:*
   C_{struct} is defined as the set of the immediate, but not semantically relevant attributes of a class:

   $CarOwner_{struct}$ = {Name, Address, Cars}
   $Person_{struct}$ = {Name}

3. *Determine Semantic Similarity:*
   For two classes A and B, the set of terminological relationships between all pairs of elements of (A_{sem}, B_{sem}), (A_{sem}, B_{struct}) and (A_{struct}, B_{sem}). The strength of each terminological relationship found in this step is multiplied with the relevance of the attribute in C_{sem}.

   The only non-null set is the set of all pairs in $Person_{sem}$ and $CarOwner_{struct}$ which is given by {(0.38, Address)}
   Thus the overall similarity of CarOwner and Person is 0.38.

Discussion

This approach attempts to combine the semantic and the structural (schematic) knowledge available. The semantic knowledge employed is represented in the form of a fuzzy terminological knowledge base with relationships like synonymy, generalization associated with fuzzy strengths. This may be viewed as an attempt to capture and represent the *real world semantics* aspect of representation of semantics. This is in contrast to the semantic proximity approach where the *explicit context* aspect is modeled and used as a framework for assigning the fuzzy strengths.

3.8 Related Work

[Collet *et al.*, 1991] discuss an approach where resource integration is performed by mapping the individual schemas to concepts in the Cyc knowledge base using a set of articulation axioms. In addition to the structural description of the local schema, schema knowledge, resource knowledge and organization knowledge are used. [Sheth *et al.*, 1993] discuss an approach where the attributes are mapped to an attribute hierarchy based on the RWS, which is then used in schema integration. [Sheth, 1991*] lays a framework for

semantic relationships between objects. [Kent, 1991] discusses how multidatabases violate implicit assumptions about how databases model reality giving rise to semantic conflicts and uncertainty. [DeMichiel, 1989] has used partial values and maybe tuples using the framework of 3-valued logic to model uncertainty. [Tseng *et al.*, 1993] have used discrete probability distributions to model uncertainty in relational databases.

Bibliography

[Borgida *et al.*, 1989] A. Borgida, R. Brachman, D. McGuinness, and L. Resnick. CLASSIC: A structural data model for objects. In *Proceedings of ACM SIGMOD-89*, 1989.

[Breitbart *et al.*, 1986] Y. Breitbart, P. Olson, and G. Thompson. Database Integration in a Distributed Heterogeneous Database System. In *Proceedings of the 2nd IEEE Conference on Data Engineering*, February 1986.

[Castellanos, 1993] M. Castellanos. Semantic enrichment of Interoperable Databases. In *Proceedings of RIDE-IMS*, April 1993.

[Collet *et al.*, 1991] C. Collet, M. Huhns, and W. Shen. Resource Integration using a Large Knowledge Base in Carnot. *IEEE Computer*, December 1991.

[Chierchia and McConnell-Ginet, 1990] G. Chierchia and S. McConnell-Ginet. *Meaning and Grammar: An Introduction to Semantics*, chapter 6. MIT Press Cambridge MA, 1990.

[Czejdo *et al.*, 1987] B. Czejdo, M. Rusinkiewicz, and D. Embley. An approach to Schema Integration and Query Formulation in Federated Database Systems. In *Proceedings of the 3rd IEEE Conference on Data Engineering*, February 1987.

[Castellanos *et al.*, 1991] M. Castellanos, F. Saltor, and M. Garcia-Solaco. The development of Semantic Concepts in the BLOOM Model using an Object Metamodel. Technical Report LSI-91-22, UPC, 1991.

[DeKleer, 1986] J. DeKleer. An assumption-based truth maintenance system. *Artificial Intelligence*, 28, 1986.

[Dempster, 1968]A. Dempster. A generalization of the bayesian inference. *Journal of the Royal Statistical Society, Series B*, 30, 1968.

[DeMichiel, 1989] L. DeMichiel. Resolving Database Incompatibility: An approach to performing Relational Operations over Mismatched Domains. *IEEE Transactions on Knowledge and Data Engineering*, 1(4), 1989.

[Daruwala *et al.*, 1995] A. Daruwala, C. Goh, et al. The Context Interchange Network. In *Proceedings of the IFIP WG2.6 Conference on Database Semantics, DS-6*, May 1995.

[Dayal and Hwang, 1984] U. Dayal and H. Hwang. View definition and Generalization for Database Integration of a Multidatabase System. *IEEE Transactions on Software Engineering*, 10(6), November 1984.

[Fankhauser *et al.*, 1991] P. Fankhauser, M. Kracker, and E. Neuhold. Semantic vs. Structural resemblance of Classes. *SIGMOD Record, special issue on Semantic Issues in Multidatabases*, A. Sheth, ed., 20(4), December 1991.

[Gruber, 1993] T. Gruber. A translation approach to portable ontology specifications. *Knowledge Acquisition, An International Journal of Knowledge Acquisition for Knowledge-Based Systems*, 5(2), June 1993.

[Garcia-Solaco *et al.*] M. Garcia-Solaco, M. Castellanos, and F. Saltor. Semantic Heterogeneity in Multidatabase Systems.

[Garcia-Solaco *et al.*, 1993] M. Garcia-Solaco, M. Castellanos, and F. Saltor. Discovering Interdatabase Resemblance of Classes for Interoperable Databases. In *Proceedings of RIDE-IMS*, April 1993.

[Guha, 1990] R. V. Guha. Micro-theories and Contexts in Cyc Part I: Basic Issues. Technical Report ACT-CYC-129-90, Microelectronics and Computer Technology Corporation, Austin TX, June 1990.

[Hammer and McLeod] J. Hammer and D. McLeod. On the Resolution of Representational Diversity in Multidatabase Systems. *Chapter 4* of this book.

[Heimbigner and McLeod, 1985] D. Heimbigner and D. McLeod. A Federated Architecture for Information Systems. *ACM Transactions on Office Information Systems*, 3(3), 1985.

[Hammer and Mcleod, 1993] J. Hammer and D. McLeod. An approach to resolving Semantic Heterogeneity in a Federation of Autonomous, Heterogeneous, Database Systems. *International Journal of Intelligent and Cooperative Information Systems.*, March 1993.

[Kent, 1991] W. Kent. The breakdown of the Information Model in Multidatabase Systems. *SIGMOD Record, special issue on Semantic Issues in Multidatabases*, A. Sheth, ed., 20(4), December 1991.

[Krishnamurthy *et al.*, 1991] R. Krishnamurthy, W. Litwin, and W. Kent. Language features for Interoperability of Databases with Schematic Discrepancies. In *Proceedings of 1991 ACM SIGMOD*, May 1991.

[Kashyap and Sheth] V. Kashyap and A. Sheth. Semantic and Schematic Similarities between Database Objects: A Context-based approach. *The VLDB Journal.* To appear; http://www.cs.uga.edu/LSDIS/~amit/66b-VLDB.ps.

[Kim and Seo, 1991] W. Kim and J. Seo. Classifying Schematic and Data Heterogeneity in Multidatabase Systems. *IEEE Computer*, 24(12), December 1991.

[Litwin and Abdellatif, 1986] W. Litwin and A. Abdellatif. Multidatabase Interoperability. *IEEE Computer*, 19(12), December 1986.

[Mena and Kashyap] E. Mena and V. Kashyap. OBSERVER: An Approach for Query Processing in Global Information Systems based on Interoperation across Pre-existing Ontologies. working paper, http://www.cs.uga.edu/LSDIS/infoquilt.

[Ouksel and Naiman, 1993] A. Ouksel and C. Naiman. Coordinating Context Building in Heterogeneous Information Systems. *Journal of Intelligent Information Systems*, 1993.

[Salton, 1989] G. Salton. *Automatic text processing.* Addison-Wesley, 1989.

[Sheth and Gala, 1989] A. Sheth and S. Gala. Attribute relationships: An impediment in automating Schema Integration. In *Proceedings of the NSF Workshop on Heterogeneous Databases*, December 1989.

[Sheth *et al.*, 1993] A. Sheth, S. Gala, and S. Navathe. On automatic Reasoning for Schema Integration. *International Journal on Intelligent and Cooperative Information Systems*, 2(1), March 1993.

[Shafer, 1976] G. Shafer. *A Mathematical Theory of Evidence.* Princeton University Press, 1976.

[Sheth, 1991] A. Sheth. Federated Database Systems for managing Distributed, Heterogeneous, and Autonomous Databases. *Tutorial Notes - the 17th VLDB Conference*, September 1991.

[Sheth, 1991*] A. Sheth. Semantic issues in Multidatabase Systems. *SIGMOD Record, special issue on Semantic Issues in Multidatabases*, A. Sheth, ed., 20(4), December 1991.

[Sheth and Kashyap, 1992] A. Sheth and V. Kashyap. So Far (Schematically), yet So Near (Semantically). *Invited paper in Proceedings of the IFIP TC2/WG2.6 Conference on Semantics of Interoperable Database Systems, DS-5*, November 1992. In IFIP Transactions A-25, North Holland, 1993.

[Sheth and Larson, 1990] A. Sheth and J. Larson. Federated Database Systems for managing Distributed Heterogeneous and Autonomous Databases. *ACM Computing Surveys*, 22(3), September 1990.

[Siegel and Madnick, 1991] M. Siegel and S. Madnick. A Metadata Approach to resolving Semantic Conflicts. In *Proceedings of the 17th VLDB Conference*, September 1991.

[Sciore *et al.*, 1992] E. Sciore, M. Siegel, and A. Rosenthal. Context Interchange using Meta-Attributes. In *Proceedings of the CIKM*, 1992.

[Tseng *et al.*, 1993] F. Tseng, A. Chen, and W. Yang. Answering Heterogeneous Database Queries with Degrees of Uncertainty. *Distributed and Parallel Databases, An International Journal*, 1(3), July 1993.

[Wood, 1985] J. Wood. What's in a link ? In *Readings in Knowledge Representation.* Morgan Kaufmann, 1985.

[Yu *et al.*, 1991] C. Yu, W. Sun, S. Dao, and D. Keirsey. Determining relationships among attributes for Interoperability of Multidatabase Systems. In *Proceedings of the 1st International Workshop on Interoperability in Multidatabase Systems*, April 1991.

[Zadeh, 1978] L. Zadeh. Fuzzy Sets as a basis for a Theory of Possibility. *Fuzzy Sets and Systems*, 1(1), 1978.