Merging Databases under Constraints

Jinxin Lin * Alberto O. Mendelzon

Dept. of Computer Science
University of Toronto
Toronto, CANADA M5S 1A4
{lin,mendel}@db.toronto.edu

June 3, 1996

Abstract

The problem of integrating information from conflicting sources comes up in many current applications, such as cooperative information systems, heterogeneous databases, and multi-agent systems. We model this by the operation of *merging* first-order theories. We propose a formal semantics for this operation and show that it has desirable properties, including abiding by majority rule in case of conflict and syntax independence. We apply our semantics to the special case when the theories to be merged represent databases under integrity constraints. We then present a way of merging databases that have different or conflicting schemas caused by problems such as *synonym*, *homonym* or *type conflicts* in the schema integration literature.

1 Introduction

Being able to share information from multiple sources has become increasingly important. Considerable efforts have been made in both academia and industry to develop global information sharing systems such as federate databases, multidatabases, or interoperable systems (cf. [BAP92, EP90]). Those systems aim at integrating information stored in possibly distributed or heterogeneous component databases. Research in this area focuses on integrating the schemas of local component databases into a global, unified schema (cf. [BL86, MIR94]). Problems treated include name differences, structural differences or format differences. However, little attention has been paid to the problem of conflicting or missing data among component databases. This problem is important since data from different databases often disagree because of, for instance, incomplete updates, system error or valid differences in underlying semantics of the databases [BAP92].

*Current address of this author: Enterprise Integration Laboratory, Department of Industrial Engineering, University of Toronto, 4 Taddle Creek Road, Toronto, CANADA M5S 3G9.

A similar problem exists in many other areas such as cooperative information systems [BJe94, HM92, WL92], multi-agent reasoning systems [HM85, FH88], collaborative computing or groupware [Gru94], computer-supported cooperative work (CSCW), etc. In a system involving multiple agents, knowledge or beliefs held by different agents may be contradictory. However, these systems want to extract additional knowledge that is not locally held by any agent, but collectively by all of them (called *distributed knowledge* in [HM85]). For example, if an agent knows a and another agent knows $a \to b$, then combining their knowledge yields b, even though neither one of them individually knows b.

In this paper, we propose a formal semantics for a generalization of the above problem—merging first-order theories. The approach has a close connection with the work on database updates [FUV83, FKUV86, KM91a] and knowledge base revision [G8̈8, KM91b]. It has a special property of obtaining maximal amount of information from each theory while observing majority rule in case of conflict. We apply the semantics to merge the information among databases, where a database is viewed as a simple form of first-order theory in which facts but no rules are involved. We also present a way of merging databases that have different or conflicting schemas caused by the problems such as *synonym*, *homonym* or *type conflicts* in the schema integration literature. Our approach constitutes one step towards dealing with these problem in multidatabases.

In Section 2 we present a motivating example for merging knowledge of multiple agents. In Section 3 we provide the formal preliminaries for the language used in this paper. We define the semantics for merging multiple theories and study the properties of the semantics in Section 4. In Section 5 we explore the relationship between the merging operator and the AGM theory of revision. We then use the proposed operator in merging databases with uniform schemas, and databases with conflicting schemas, under constraints in Section 6 and 7 respectively. Related work is described in Section 8 and some topics of future research are discussed in Section 9. The proofs of theorems and propositions are left in the Appendix.

2 An Example Scenario

Suppose three doctors forming a committee are in consultation regarding two patients, Jeff and Ed. It may happen that each doctor examines the patients independently and observes different symptoms; and perhaps not every doctor examines every patient.

Suppose there are three symptoms, S_1, S_2, and S_3, and three possible diagnoses, D_1, D_2, and D_3. Let $S_i(x)$ mean "patient x presents symptom S_i" and $D_i(x)$ "patient x has disease D_i". Suppose the doctors' knowledge concerning the two patients is as follows:
$DOC_a = \{S_1(\mathit{Jeff}), S_2(Ed), \forall x.\ S_1(x) \to D_1(x)\}$.
$DOC_b = \{S_1(\mathit{Jeff}), \forall x.\ S_1(x) \to D_2(x), \forall x.\ S_2(x) \to D_3(x)\}$.
$DOC_c = \{S_3(\mathit{Jeff}), \forall x.\ S_3(x) \to D_1(x)\}$.

Then for patient Jeff, both DOC_a and DOC_c diagnose disease D_1 since they observe symptom S_1 and S_3 respectively and their knowledge is that these symptoms lead to disease D_1. In contrast, DOC_b diagnoses Jeff with D_2, as she believes symptom S_1 leads to D_2. For Ed, none of the doctors can draw a conclusion — DOC_a observes symptom S_2 from Ed but she has no knowledge about this symptom, while DOC_b has knowledge about S_2 (i.e., she could conclude D_3 from the symptom S_2) but she (as well as DOC_c) has not examined Ed.

There are several points to note abut this scenario:

- Conflicts can easily arise in situations involving multiple agents, such as the conflict regarding Jeff's diagnosis.
- Although there is a conflict among the doctors on Jeff's disease, a decision must be made. If is it not possible to obtain more information, a common practice is to follow the majority. In this case, as two doctors diagnose $D_1(\mathit{Jeff})$ but just one diagnoses $D_2(\mathit{Jeff})$, the committee would conclude $D_1(\mathit{Jeff})$.
- The committee should be able to determine that Ed has disease D_3. The inconsistency about Jeff's disease should have no influence on the diagnosis of Ed.

3 Preliminaries

We consider a function free first-order language $\mathcal{L}$ constructed in the usual way from a set of relation symbols and variables, equality ($=$), and a set of *domain elements*. The set of domain elements defines the domain over which quantifiers range. Relation symbols take variables and domain elements as their arguments. A *primitive sentence* is any atom of the form $P(a_1, \ldots, a_n)$, where P is an n-ary relation symbol and $a_1, \ldots, a_n$ are domain elements. The set of all primitive sentences is denoted $\mathcal{P}$. The set of all relation symbols in $\mathcal{L}$ is denoted $\mathcal{R}$.

Formulas and sentences of the language $\mathcal{L}$ are formed in the obvious way. If ϕ is a formula where variable x occurs free, $\phi[x/a]$ denotes the formula ϕ with each free occurrence of x substituted by a. A formula is *ground* if it involve no variables. The cardinality of a set S is denoted by $|S|$.

A *possible world* w is a subset of $\mathcal{P}$. w is *finite* if $|w|$ is finite. The recursive definition of when w *satisfies* a sentence ϕ (denoted $w \models \phi$) is as follows:

$w \models \phi$, for a primitive sentence ϕ, iff $\phi \in w$.

$w \models (a_1 = a_2)$ iff a_1 and a_2 are identical domain elements.

$w \models \phi \wedge \psi$ iff $w \models \phi$ and $w \models \psi$.

$w \models \neg\phi$ iff not $w \models \phi$.

$w \models (\exists x \phi)$ iff $w \models \phi[x/a]$ for some domain element a.

It can be seen that domain elements in this semantics are pairwise distinct and that all possible worlds are assumed to have the same domain of discourse—the set of all domain elements.

The set of all worlds is denoted $\mathcal{W}$. A *model* of a sentence is a possible world that satisfies the sentence. A model of a set of sentences is a model of every member in the set. The set of all models of ϕ is denoted by $[\![\phi]\!]$. If $[\![\phi]\!]$ is empty, then ϕ is said to be *inconsistent* or *unsatisfiable*; otherwise, it is said to be *consistent* or *satisfiable*.

We say that ϕ *implies* or *supports* ψ (written $\phi \models \psi$) if $[\![\phi]\!] \subseteq [\![\psi]\!]$, and ϕ is logically *equivalent* to ψ (written $\phi \equiv \psi$) if $[\![\phi]\!] = [\![\psi]\!]$. A (first-order) *theory* is a set of sentences of the language $\mathcal{L}$, either finite or infinite. An *(integrity) constraint* is a sentence of $\mathcal{L}$.

4 Semantics of Theory Merging

We now consider how to merge first-order theories under a set of constraints. The constraints may embody world knowledge such as "no person can have more than one birth date", or represent requirements imposed on the result of the merge by the designer, such as in Section 2 "a patient has only one disease". The integrity constraints are requirements that the merged theory must satisfy. Let $T_1, \ldots, T_n$ be the theories to merge and IC be a set of constraints. We denote the result of merging by $Merge(\{T_1, \ldots, T_n\}, IC)$, which is also a theory.

4.1 The Definition

Our merge operator proposed in this section extends the one in [LM94] to the first-order case and to deal with integrity constraints. The idea is that the models of the resulting theory should be those possible worlds that are "closest" to the original theories. We will first define the *distance* between a world and the set of theories $\{T_1, \ldots, T_n\}$, and then define the models of the resulting theory to be those worlds that have *minimal* distances from $\{T_1, \ldots, T_n\}$.

First let the distance between two possible worlds w and w', denoted $dist(w, w')$, to be $|w \oplus w'|$, the cardinality of the symmetric difference of w and w'.[1] Then let the distance between a possible world w and a theory T_i be

$$dist(w, T_i) = \min_{w' \in [\![T_i]\!]} dist(w, w').$$

For the boundary case where $[\![T_i]\!] = \emptyset$, we define $dist(w, T_i) = 0$.

There may be possible worlds that are close to a particular theory but distant from others. We should select the worlds that are closest *overall* to the set of theories. We define the *overall distance* between w and $\{T_1, \ldots, T_n\}$ to be

$$dist(w, \{T_1, \ldots, T_n\}) = \sum_{i=1}^{n} dist(w, T_i).$$

Then *Merge* is defined as:

Definition 4.1 $[\![Merge(\{T_1, \ldots, T_n\}, IC)]\!] = \{w \in [\![IC]\!] \mid dist(w, \{T_1, \ldots, T_n\})$ *is minimum*$\}$.

That is, a possible world is a model of $Merge(\{T_1, \ldots, T_n\}, IC)$ if and only if it is a model of IC and its overall distance to the set of theories is minimum.

[1] $w \oplus w' = (w \setminus w') \cup (w' \setminus w)$

4.2 Some Properties and Examples

The following gives some properties of *Merge*:

Theorem 4.1

M1 $Merge(\{T_1, \ldots, T_n\}, IC) \models IC$.

M2 *$Merge(\{T_1, \ldots, T_n\}, IC)$ is consistent iff IC is consistent.*

M3 *If $T_1 \cup \cdots \cup T_n \cup IC$ is consistent then $Merge(\{T_1, \ldots, T_n\}, IC) \equiv T_1 \cup \cdots \cup T_n \cup IC$.*

M4 *If $T_1 \equiv T_1', \ldots, T_n \equiv T_n'$ and $IC \equiv IC'$ then*

$$Merge(\{T_1, \ldots, T_n\}, IC) \equiv Merge(\{T_1', \ldots, T_n'\}, IC').$$

M5 *If T_i is inconsistent then*

$$Merge(\{T_1, \ldots, T_n\}, IC) \equiv Merge(\{T_1, \ldots, T_{i-1}, T_{i+1}, \ldots, T_n\}, IC).$$

Proof: See Appendix. □

M1 says that the result of merge satisfies IC. M2 guarantees that the result of merge under a consistent IC is always consistent. M3 shows that if there is no conflict among IC and the set of theories then the merged theory is simply the union of the theories and IC. M4 states that *Merge* is independent of the syntax of either IC or $T_1, \ldots, T_n$. M5 shows that if some theories to be merged are inconsistent, they can be discarded without affecting the result of merging. If T_i is inconsistent for every $i \in [1, n]$, then the merged theory is the same as IC since then all models of IC are equally close to $\{T_1, \ldots, T_n\}$ with zero distance. These are intuitively desirable properties for a knowledge merging operator.

Let us call a theory *finitely satisfiable* if it has at least a *finite* model. In many practical cases we need only to consider the constraints and theories that are finitely satisfiable. Theories and constraints that have no finite model contain "axioms of infinity" and they are not suitable for databases (either deductive or relational) purposes. This view seems to be shared by some other authors (e.g., [BM86]). For finitely satisfiable $T_1, \ldots, T_n$ and IC, we have the following two results.

Proposition 4.1 *Let l be a literal of the form $P(a_1, \ldots, a_m)$ or $\neg P(a_1, \ldots, a_m)$, where $a_1, \ldots, a_m$ are domain elements and P is a relation symbol that does not appear in IC. Then if more than half of the theories in $T_1, \ldots, T_n$ support l then*

$$Merge(\{T_1, \ldots, T_n\}, IC) \models l.$$

Proof: See Appendix. □

This result shows that, in case of conflict, *Merge* follows the views of the majority.

In the case that each theory supports one of l and $\neg l$, the simple majority, i.e., there is more support for l than $\neg l$, is sufficient for the result of *Merge* to support l.

Proposition 4.2 *Let l be a literal of the form $P(a_1,\ldots,a_m)$ or $\neg P(a_1,\ldots,a_m)$, where $a_1,\ldots,a_m$ are domain elements and P is a relation symbol that does not appear in IC. Suppose each T_i $(i=1,\ldots,n)$ supports either l or $\neg l$, and more T_i's support l than supporting $\neg l$. Then*

$$Merge(\{T_1,\ldots,T_n\}, IC) \models l.$$

Proof: In a way similar to that of Proposition 4.1. □

Example 4.1 (DOC_a,DOC_b,DOC_c are given in Section 2.) Let IC contains the sentence: $\forall x.\ D_1(x) \rightarrow \neg D_2(x)$ "no patient has diseases D_1 and D_2 at the same time".

$$\begin{aligned} Merge(\{DOC_a, DOC_b, DOC_c\}, IC) \ \equiv\ & IC \wedge \{S_1(Jeff), S_3(Jeff), S_2(Ed), \\ & \forall x.\ S_1(x) \rightarrow D_1(x) \vee D_2(x), \\ & \forall x.\ S_3(x) \rightarrow D_1(x), \\ & \forall x.\ S_2(x) \rightarrow D_3(x)\}. \end{aligned}$$

Then from $S_3(Jeff)$ and the rule $\forall x.\ S_3(x) \rightarrow D_1(x)$, we can conclude $D_1(Jeff)$. And from $S_2(Ed)$ and the rule $\forall x.\ S_2(x) \rightarrow D_3(x)$, we conclude $D_3(Ed)$, as desired. The conflict between DOC_a and DOC_b about what to infer from symptom S_1 is resolved by taking the rule $\forall x.\ S_1(x) \rightarrow D_1(x) \vee D_2(x)$. The merged theory contains the sentences $\forall x.\ S_2(x) \rightarrow D_3(x)$ and $\forall x.\ S_3(x) \rightarrow D_1(x)$, as they are not in conflict among the doctors. From this we can see that the conflict on Jeff's diagnosis has no influence on Ed's and that the doctors cooperate on the diagnosis of Ed's disease. In addition, the majority view is followed to resolve the conflict on Jeff's disease.

5 Relationship with AGM Postulates

We now consider the special case when $n = 1$, i.e., there is only a single theory T_1 to be merged with IC. Then if we view IC as new knowledge and T_1 as an old theory to be revised, $Merge(\{T_1\}, IC)$ can be interpreted as a belief revision operation. Let us define a revision operator $\circ_M$ that incorporates a new sentence μ into an existing knowledge base ψ as follows:

Definition 5.1 $\psi \circ_M \mu = Merge(\{\psi\}, \mu)$.

Gärdenfors and his colleagues [G$\ddot{8}$8] propose a set of *rationality postulates*, colloquially known as the *AGM postulates*, that they argue to be desirable for revision. Katsuno and Mendelzon [KM91b] rephrase those postulates so that they are applicable for finite knowledge bases. The rephrased postulates can be described as follows, where ψ is the old knowledge base, μ is new information and $\circ$ is a revision operator.

(R1) $\psi \circ \mu$ implies μ.

(R2) If $\psi \wedge \mu$ is consistent then $\psi \circ \mu \equiv \psi \wedge \mu$.

(R3) If μ is consistent then $\psi \circ \mu$ is also consistent.

(R4) If $\psi_1 \equiv \psi_2$ and $\mu_1 \equiv \mu_2$ then $\psi_1 \circ \mu_1 \equiv \psi_2 \circ \mu_2$.

(R5) $(\psi \circ \mu) \wedge \phi$ implies $\psi \circ (\mu \wedge \phi)$.

(R6) If $(\psi \circ \mu) \wedge \phi$ is consistent then $\psi \circ (\mu \wedge \phi)$ implies $(\psi \circ \mu) \wedge \phi$.

The intuition behind these postulates is described in [G$\ddot{8}$8, KM91b].

We now show that $\circ_M$ satisfies the above postulates for revision. This shows that in single theory case, our merge operator $Merge(\{\psi\}, \mu)$ can act as a revision operator with all the desirable properties.

Theorem 5.1 $\circ_M$ *satisfies R1-R6.*

Proof: See Appendix. □

6 Merging Databases under Constraints

Some time ago, several graduate students in the database group of the University of Toronto tried to share their own bibliographic databases, which were maintained individually by the students and used for their LaTeX purposes. They soon found out that their data are inconsistent. For example, in one student's database, the paper of Reiter "On closed world databases" is recorded for the publication year 1988, while in the others' it is recorded for the year 1978. (In fact, 1978 is the correct publication year for the paper). Tracing the problem, they discovered that one student mistyped 1978 with 1988. Such mistakes also exist in various other places of the databases. To create a unified bibliographic database containing correct information, the students needed to discover and correct every detail of the inconsistencies, by comparing each other's data manually. However, the problem is the large size of the databases—each student maintained at least several hundreds of bibliographic records. No one in the group was willing to undertake this task, and the idea of sharing bibliographic databases remains unrealized.

This scenario shows that a computer solution for integrating inconsistent databases is called for. It presents a typical example of the problem of merging databases under constraints, where each database contains relational records such as $Bibitem(Author, Title, Year, \ldots)$, and integrity constraints are functional dependencies (fds) such as:

$$Author, Title \longrightarrow Year \quad \text{("every publication has a unique publication year").}$$

A simple way of merging a set of databases is to take their union, but this may violate the given constraints. For instance, suppose a database contains $Bibitem(a, t, y_1)$ while another database contains $Bibitem(a, t, y_2)$. Then the union of the databases violates the mentioned functional dependency. We would like the result of merge to satisfy the constraints and retain as much information as possible from each component database. We shall use the proposed merging operator to deal with the problem of merging databases under constraints.

We define a *database* to be a finite set of primitive sentences in $\mathcal{P}$. We adopt the *open-world assumption* as opposed to the *closed-world assumption* (both assumptions are described in [Rei78]). The reason is that for databases such as $db_1 = \{Bibitem(a_1, t_1, y_1)\}$ and $db_2 = \{Bibitem(a_2, t_2, y_2)\}$, we do not want the two databases to be considered in conflict, and we want the result of merging to be simply $\{Bibitem(a_1, t_1, y_1), Bibitem(a_2, t_2, y_2)\}$. There may be situations where a closed-world view is appropriate; it is possible to treat this case along similar lines, which we do not pursue here.

To talk about the *schema* of a database, we associate the language $\mathcal{L}$ with a set of *attribute names* such as $Author$, $Title$, $Year$, etc. We associate each n-ary relation symbol $R \in \mathcal{R}$ with a *schema* denoted by $R(A_1, \ldots, A_n)$, where $A_1, \ldots, A_n$ are attribute names. Then the *schema* of a database db, denoted by $\sigma(db)$, is the set of schemas of all relational symbols appearing in db. As usual, a database is often naturally viewed in its tabular format. In the schema integration literature, schema conflicts in different databases, such as *synonym* and *homonym* problems, are studied (see Section 7 for examples of these problems). However, in this section we assume that there are no conflicts among the schemas of the databases to be merged. We will relax this assumption and study merging databases with conflicting schemas in Section 7.

As before, we consider IC to be a set of first-order sentences and the choice of IC is the task of the merge designer. Those are the constraints that the designer wishes to enforce during merging, e.g., the fd: $Author, Title \longrightarrow Year$ (expressed as a first-order sentence). It is well known that many static integrity constraints can be expressed as first-order sentences. For example, the aforementioned fd is expressed as follows:

$$\forall x \forall y \forall z_1 \forall z_2.\ Bibitem(x, y, z_1) \wedge Bibitem(x, y, z_2) \rightarrow z_1 = z_2. \tag{1}$$

6.1 Computing the Result of Merging

We now present a syntactic way of computing the result of merging databases under constraints. For a set of databases $db_1, \ldots, db_n$ and a set of constraints IC, $Merge(\{db_1, \ldots, db_n\}, IC)$ is a single database or a disjunction of databases, plus the IC.

But first, a data structure known as *multiset* is needed. A *multiset* is similar to a *set* except that a multiset allows duplicate elements. We express a multiset like a set, e.g., a multiset $C = \{a, b, a\}$. We define the operation *combination* ($\sqcup$) between two sets A and B as: $A \sqcup B$ results in a multiset containing all the elements of A and all the elements of B. *Combination* is similar to the operation *union* of sets, except that during the operation of combination duplicate elements are not removed. The predicates $\in$, $\subseteq$ and $\subset$ are as in set theory.

Theorem 6.1 *(Syntactic equivalence of merging dbs)*

$$Merge(\{db_1, \ldots, db_n\}, IC) \equiv IC \wedge (db'_1 \vee \cdots \vee db'_m),$$

where db'_i ($i \in [1, m]$) is a set resulting from removing duplicate elements in a maximum sub-multiset C_i of $db_1 \sqcup \cdots \sqcup db_n$ such that $db'_i \wedge IC$ is consistent.

Proof: See Appendix. □

To compute the result of merging a set of databases under constraints IC, we first compute the maximum sub-multisets of the combination of the databases that are consistent with IC. If there is only one maximum sub-multiset, then the result is a single database obtained from the sub-multiset; otherwise, the result is a disjunction of databases, each transformed from one of the maximum sub-multisets. Note that in this syntactic equivalence, *maximum sub-multisets*, instead of *maximal subsets*, are used. In a multiset the number of same elements make different in calculating cardinality. This is the key to achieving majority principle in the merging process. The result of a disjunction of databases is when there is no majority for a unique selection, i.e., each of the selection supported by a database is equally plausible.

It is clear that $db'_1, \cdots, db'_m$ are *databases* (i.e. finite subsets of $\mathcal{P}$). Since IC is always preserved during the merge operation, in computing the result of merge we only need to focus on $db'_1 \vee \cdots \vee db'_m$. Let us denote $db'_1 \vee \cdots \vee db'_m$ by $\mathcal{DB}_{ic}(db_1, \ldots, db_n)$. That is,

Definition 6.1 *$Merge(\{db_1, \ldots, db_n\}, IC) \equiv IC \wedge \mathcal{DB}_{ic}(db_1, \ldots, db_n)$.*

6.2 Tabular Representation

We have seen from the last section that the result of merge $\mathcal{DB}_{ic}(db_1, \ldots, db_n)$ may be a disjunction of a databases; the situation is encountered when there is no majority to resolve the conflict. Then the question is how to represent the result in a usual tabular format of database. This question is essentially the same as how to representing disjunctive information in databases. The topic of representing disjunctive information has been studied by many researchers, and a number of proposals have been put forward in the literature, e.g. [KW85, AG85, IV89].

If the disjunctive occurs over a single attribute in a relation, then a simple form, called *OR-table* as in [IV89], can be employed.

Example 6.1 Suppose we have the ground formula

$$R(a, b, c) \wedge (R(e, f, g_1) \vee R(e, f, g_2) \vee R(e, f, g_3)).$$

The representation is:

R

a	b	c
e	f	$\{g_1, g_2, g_3\}$

In the following we shall describe a general method for representing the result of merging databases. The method can represent any conjunctive normal form of a ground formula; we will discuss shortly how this method is useful in the merge framework. We choose to describe our approach through examples instead of formal definitions since it will be easier.

Example 6.2 Suppose we have the ground formula

$$R(a) \wedge (R(b) \vee R(c) \vee R(d)) \wedge (R(b) \vee R(c) \vee R(e)).$$

The representation is:

R

Attribute	*disj*
a	
b	1,2
c	1,2
d	1
e	2

This means that the relation R contains two pieces of disjunctive information, one (noted by the number 1 in the attribute $disj$) involving b, c and d, the other (noted by the number 2 in the attribute $disj$) involving b, c and e. The tuple a is not in any disjunction, and hence its column $disj$ is empty.

The following example shows a case where disjunctive information spans over several relations.

Example 6.3 Suppose we have the ground formula

$$R(a) \wedge (R(b) \vee Q(c)) \wedge Q(d).$$

The representation is:

R

(Attribute)	*disj*
a	
b	1

Q

(Attribute)	*disj*
c	1
d	

In this case the disjunction spans over two relations R and Q. In merging databases, we rarely encounter such a case, since the constraints concerned are mainly data dependencies, which involve a single relation and disjunctive information occurs only within the relation.

It can be seen that this method can represent any ground formula in *conjunctive normal form*, without duplicating tuple information (i.e. each tuple appears in the table only once). From Theorem 6.1 we know that the result of merging relational databases is a disjunction of conjunctions of ground atoms (together with the constraints for the merging). The constraints stay as they are (as first-order formulas) and the disjunction can be transformed into a ground formula in conjunctive normal form. Hence the result of merging databases under constraints can be represented by the above method.

6.3 Examples of Merging Databases

Example 6.4 Suppose db_1, db_2 and db_3, each containing a single table, are given as follows:

db_1:

Bibitem

Author	Title	Year
a	b	c_1
e	f	g

db_2:

Bibitem

Author	Title	Year
a	b	c_2
h	i	j

db_3:

Bibitem

Author	Title	Year
a	b	c_1
l	m	n

And IC is as (1). Then

$$\mathcal{DB}_{ic}(db_1, db_2, db_3) \equiv$$

Bibitem

Author	Title	Year
a	b	c_1
e	f	g
h	i	j
l	m	n

The databases are in conflict on the year of the publication authored a and entitled b. Since two databases support c_1 and only one database supports c_2, the result of merge supports the publication year c_1. The result of merge also contains the information that is irrelevant to the conflict—it contains the tuples $(e, f, g), (h, i, j), (l, m, n)$ in the relation *Bibitem*.

Example 6.5 Suppose IC, db_1 and db_2 are as in Example 6.4, and db_3 is given as follows:

Bibitem

Author	Title	Year
a	b	c_3
l	m	n

Then

$$\mathcal{DB}_{ic}(db_1, db_2, db_3) \equiv$$

Bibitem

Author	Title	Year
a	b	$\{c_1, c_2, c_3\}$
e	f	g
h	i	j
l	m	n

Now each database supports a different year for the publication authored a and entitled b. The result of merge says that the year for this publication is either c_1 or c_2 or c_3, but it does not know which one. And as in Example 6.4, the resulting database contains other information that is irrelevant to the conflict, i.e., the tuples $(e, f, g), (h, i, j), (l, m, n)$ in the relation *Bibitem*.

6.4 Some Properties of Database Merging

Clearly, $Merge(\{db_1, \ldots, db_n\}, IC)$ enjoys all the properties of first-order theory merging. In addition, it has some properties specific to relational database merging.

The first proposition says that if no constraints need to be enforced, then the result of the merge is simply the union of the databases. This is due to the simple form of the theories that represent databases under the open world assumption.

Proposition 6.1 *$Merge(\{db_1, \ldots, db_n\}, \emptyset) \equiv db_1 \cup \ldots \cup db_n$.*

The next result is also due to the fact of databases being a simple form of theory.

Proposition 6.2 *Let $P(a_1, \ldots, a_m)$ be an element in some db_i ($i \in [1, n]$). Then if IC contains no constraint on the m-ary predicate P then:*

$$Merge(\{db_1, \ldots, db_n\}, IC) \models P(a_1, \ldots, a_m).$$

Proof: (Sketch) Proof by contradiction. Suppose $Merge(\{db_1, \ldots, db_n\}, IC) \not\models P(a_1, \ldots, a_m)$. Then there is $w \in [\![Merge(\{db_1, \ldots, db_n\}, IC)]\!]$ such that $P(a_1, \ldots, a_m) \notin w$. Let $w' = w \cup \{P(a_1, \ldots, a_m)\}$. We can then prove that $\sum_{i=1}^{n} dist(w', db_i) < \sum_{i=1}^{n} dist(w, db_i)$, thus deriving a contradiction. □

Note that data dependencies deal with one predicate only. For this particular form, we have the following property for *Merge*.

Proposition 6.3 *Suppose $db_1, \ldots, db_n$ share no common predicate (of the same name and same arity), and IC is a set of data dependencies. Then:*

$$Merge(\{db_1, \ldots, db_n\}, IC) \equiv Merge(\{db_1\}, IC) \wedge \cdots \wedge Merge(\{db_n\}, IC).$$

Proof: (Sketch) Proceed by proving that for any w in $[\![IC]\!]$ such that $\sum_{i=1}^{n} dist(w, db_i)$ is minimum we have that $dist(w, db_i)$ is minimum, for all $i \in [1, n]$. And conversely, for any w in $[\![IC]\!]$ such that $dist(w, db_i)$ is minimum, for all $i \in [1, n]$, we have that $\sum_{i=1}^{n} dist(w, db_i)$ is minimum. This can be easily proved using the property that each data dependency in IC deals with one predicate only and $db_1, \ldots, db_n$ share no common predicate symbol. □

Proposition 6.3 results in the following observation. Since a data dependency deals with only one predicate and the databases share no common predicate, merging the database with IC is equivalent to merging each database with the IC and then combine the results. In this particular case we can localize the enforcement of the data dependencies on the databases.

For tuple-generating dependencies, we have the following result.

Proposition 6.4 *If IC is a set of tuple-generating dependencies, then*

$$Merge(\{db_1, \ldots, db_n\}, IC) \equiv db_1 \cup \ldots \cup db_n \cup IC.$$

Proof: Immediately from the property of tuple-generating dependencies. □

7 Merging Databases of Conflicting Schemas

In last section we assume the databases to be merged have no conflicting schemas—the schemas are all part of the common schema $\mathcal{R}$. We consider merging databases with conflicting schemas in this section.

In many applications, it is common that different databases are developed independently with different requirements in mind. Hence there may exist discrepancies among the representations of the databases. For example, the head of a department is referred to as "manager" in one database, but as "supervisor" in another—a different name but exactly the same concept. The opposite problem—the same name is used for different concepts in different databases—may also exist. For example, the $Size$ of furniture in db_1 means the height of the furniture while it means the width of the furniture in db_2. Different databases may also have different implementations of the same relation. For example, student and course information in one database is stored in one relation $Student(Name, Gender, Birthdate, Course)$ but stored separately in two relations $Student(Name, Gender, Birthdate)$ and $Enrolled(Name, Course)$ in another. The differences among the schemas of databases are well known in schema integration (see [BL86] for a survey, also [MIR94]), e.g., this first problem is known as "synonym" problem and the second "homonym" problem. The differences and relationships among the schemas of databases must be recognized and dealt with when merging the information of the databases.

In dealing with these problems, the designer must decide on a global representation for each relation, called global schema, through which users interact with the integrated database. The global schema must ensure that all the information in the original databases can be expressed. This issue, however, is not our concern here. We can use, for example, one of the many methodologies proposed in [BL86, SL90, MIR94] to design the global schema. In this section, we assume the global schema has been constructed. We use rules to transform data from local schemas to the global schema. We then use the same merge operator to merge databases of different schemas, treating the schema transformation rules as part of constraints that must be enforced during the merging.

As the same name may be used for different concepts in different databases, we index each relational symbol by the name of the database it is associated with. For example, the $Size$ relation of db_i is denoted by $Size_{db_i}$.

7.1 Examples

We use the same semantics of *Merge* for merging databases of conflicting schemas. In this case, we have a set of databases $db_1^I, \ldots, db_n^I$ whose relation symbols are indexed by the names of the databases, a set of constraints IC that apply to the indexed or un-indexed relations, and a set of schema transformation rules S which mention indexed or un-indexed relation symbols. (See below for some examples of schema transformation rules in S.) Then, the result of merging is $Merge(\{db_1^I, \ldots, db_n^I\}, IC \cup S)$. We can see S is treated in the same status as IC. Hence both IC and S will remain true in the result of merge. Let us denote the portion of the result of merge other than $IC \cup S$ by $\mathcal{DB}_{ic}(db_1^I, \ldots, db_n^I)$.

Example 7.1 (Synonym)
Suppose db_1, db_2 and db_3, each containing a single table, are given as follows:

db_1:

Manager

Name	Gender	Age	Dept
a	m	g	d_1

db_2:

Manager

Name	Gender	Age	Dept
a	m	g	d_2

db_3:

Supervisor

Name	Age	Sex	Dept
a	g	m	d_1

Suppose it has been identified that $Manager$ and $Supervisor$, and $Gender$ and Sex are synonyms. And the global schema adopts $Manager(Name, Gender, Age, Dept)$. Then the set of schema transformation rules S consists of:

$$\forall(xyzu).\ Manager_{db_1}(x,y,z,u) \equiv Manager(x,y,z,u).$$
$$\forall(xyzu).\ Manager_{db_2}(x,y,z,u) \equiv Manager(x,y,z,u).$$
$$\forall(xyzu).\ Supervisor_{db_3}(x,y,z,u) \equiv Manager(x,z,y,u).$$

Suppose we have an integrity constraint stating that each manager manages only one department; hence IC consists of the following sentence:

$$\forall(xyzu_1u_2).\ Manager(x,y,z,u_1) \wedge Manager(x,y,z,u_2) \rightarrow u_1 = u_2.$$

Then

$$\mathcal{DB}_{ic}(db_1^I, db_2^I, db_3^I) \quad \equiv$$

Manager

Name	Gender	Age	Dept
a	m	g	d_1

The result implies $Manager(a, m, g, d_1)$, as desired.

Example 7.2 (Different structure—type conflict)
As part of an example in [BL86], suppose

$$\sigma(db_1) = \{Publisher(Name, Address)\};$$
$$\sigma(db_2) = \{Publication(Title, Code, Publisher)\}.$$

In db_1, $Publisher$ is represented as an entity but as an attribute in db_2. This difference is called *type conflict* in [BL86]. Suppose the global schema is

$$\{Publication(Title, Code), Publisher(Name, Address), PublishBy(Title, Name)\}.$$

Then S contains:

$$\forall(xy).\ Publisher_{db_1}(x, y) \equiv Publisher(x, y),$$
$$\forall(xyz).\ Publication_{db_2}(x, y, z) \equiv Publication(x, y) \wedge PublishBy(x, z).$$

Now suppose $db_1 = \{Publisher(a, d)\}$, $db_2 = \{Publication(t, c, a)\}$ and $IC = \emptyset$. Then

$$\mathcal{DB}_{ic}(db_1^I, db_2^I) \quad \equiv \quad \{Publisher(a, d), Publication(t, c), PublishBy(t, a)\}.$$

Then we can impose the query such as "retrieve the address of the publisher who publishes t". The query can be expressed as "finding the y such that $PublishBy(t, x) \wedge Publisher(x, y)$ is true". (Note now in the global schema $Publisher$ and $PublishBy$ have the same attribute $Name$.)

The answer is d, as desired.

8 Related Work

The topic of merging databases is closely related to the early work on database update and belief revision. There is a huge literature in these two areas. Pioneer work includes Fagin et al.'s theory of updating logical databases [FUV83, FKUV86], Gärdenfors's theory of revision [G̈88], Borgida's mechanism of exception handling [Bor85], Abiteboul and Grahne's update semantics [AG85], etc. There is also a great deal of work on multidatabases and schema integration. Bright et al. [BAP92] and Batini and Lenzerini [BL86] put forward a survey on the issues in multidatabase systems and schema integration respectively.

Baral *et al.* [BKMS92] present several algorithms for combining logic theories by manipulating the maximally consistent subsets of the union of the theories. Fagin et al. [FUV83] take a similar approach, not to theory merging but to belief revision. Their approaches are largely syntactic, whereas our method has a model-theoretical semantic definition. Subrahmanian [Sub94] uses annotated logic to interpret the inconsistency among multiple knowledge bases and proposes a framework for amalgamating knowledge bases. His framework is based on a non-classical logic, i.e. annotated logic, while ours is developed on top of the classical logic. There are different properties of the two approaches as well, e.g. ours has the majority property, and his method also allows the individual knowledge bases to contain inconsistencies, uncertainties and nonmonotonic modes of negation.

Halpern and Moses [HM85] defined the notion of *distributed knowledge* held by a group of agents, but their formulation of distributed knowledge applies only to agents who do not have conflicting views.

9 Concluding Remarks

We have presented a new approach for integrating conflicting information from different databases. The main properties of the approach is that it obtains maximal amount of information from each database while observing majority rule in case of conflict.

This research can be continued along a number of directions: considering the databases that contain incomplete information such as null values or disjunctions, considering priority information over the databases, or combining our formalism with Reiter's closed world assumption.

Acknowledgments

We thank Anthony Bonner, Hector Levesque, Fangzhen Lin and Ray Reiter for helpful discussions on the subject. We are grateful to the Institute for Robotics and Intelligent Systems of Canada for providing funding to this research.

Appendix: Proofs of Theorems and Propositions

Theorem 4.1

M1 $Merge(\{T_1, \ldots, T_n\}, IC) \models IC$.

M2 $Merge(\{T_1, \ldots, T_n\}, IC)$ is consistent iff IC is consistent.

M3 If $T_1 \cup \cdots \cup T_n \cup IC$ is consistent then $Merge(\{T_1, \ldots, T_n\}, IC) \equiv T_1 \cup \cdots \cup T_n \cup IC$.

M4 If $T_1 \equiv T'_1, \ldots, T_n \equiv T'_n$ and $IC \equiv IC'$ then

$$Merge(\{T_1, \ldots, T_n\}, IC) \equiv Merge(\{T'_1, \ldots, T'_n\}, IC').$$

M5 If T_i is inconsistent then

$$Merge(\{T_1, \ldots, T_n\}, IC) \equiv Merge(\{T_1, \ldots, T_{i-1}, T_{i+1}, \ldots, T_n\}, IC).$$

Proof: **M1:** Follows from the fact that $[\![Merge(\{T_1, \ldots, T_n\}, IC)]\!]$ is a subset of $[\![IC]\!]$.

M2: Equivalently, we prove that $[\![Merge(\{T_1, \ldots, T_n\}, IC)]\!] \neq \emptyset$ iff $[\![IC]\!] \neq \emptyset$.

(If part): $[\![IC]\!]$ is not empty. If there is a $w \in [\![IC]\!]$ such that $\sum_{i=1}^{n} dist(w, T_i)$ is a finite number, then there must be some $w' \in [\![IC]\!]$ (w' may be w) such that $\sum_{i=1}^{n} dist(w', T_i)$ is minimum and hence $[\![Merge(\{T_1, \ldots, T_n\}, IC)]\!]$ is not empty. Otherwise, for every $w \in [\![IC]\!]$, $\sum_{i=1}^{n} dist(w, T_i) = \infty$. It follows that $[\![Merge(\{T_1, \ldots, T_n\}, IC)]\!] = [\![IC]\!] \neq \emptyset$.

(Only-if part): Follows from the fact that $[\![Merge(\{T_1, \ldots, T_n\}, IC)]\!]$ is a subset of $[\![IC]\!]$.

M3: Easily follows from the fact that $\sum_{i=1}^{n} dist(w, T_i) = 0$, for all $w \in [\![T_1 \cup \cdots \cup T_n \cup IC]\!]$.

M4: From the semantic definition of *Merge*.

M5: From the definition that if T_i is inconsistent then $dist(w, T_i) = 0$, for all world w. □

Proposition 4.1 ($T_1, \ldots, T_n$ and IC are finitely satisfiable theories) Let l be a literal of the form $P(a_1, \ldots, a_m)$ or $\neg P(a_1, \ldots, a_m)$, where $a_1, \ldots, a_m$ are domain elements and P is a relation symbol that does not appear in IC. Then if more than half of the theories in $T_1, \ldots, T_n$ support l then

$$Merge(\{T_1, \ldots, T_n\}, IC) \models l.$$

Proof: Let $u_i \in [\![T_i]\!]$ $(i = 1, \ldots, n)$ and $w_{ic} \in [\![IC]\!]$ that are finite. Then for $i = 1, \ldots, n$, $dist(w_{ic}, u_i)$ is finite, and therefore $dist(w_{ic}, T_i)$ is finite. Then $dist(w_{ic}, \{T_1, \ldots, T_n\})$ is a finite number. It follows that for all $w \in [\![Merge(\{T_1, \ldots, T_n\}, IC)]\!]$, $dist(w, \{T_1, \ldots, T_n\})$ is a finite number.

Assume to the contrary that $Merge(\{T_1, \ldots, T_n\}, IC) \not\models l$. Then there exists $w \in [\![Merge(\{T_1, \ldots, T_n\}, IC)]\!]$ such that $w \not\models l$. Let w_l^- be the possible world that agrees with w on everything except the truth value of l, in which case $w_l^- \models l$. Since IC does not mention P and $w \in [\![IC]\!]$, we have $w_l^- \in [\![IC]\!]$.

In the next we will prove that $\sum_{i=1}^n dist(w_l^-, T_i) < \sum_{i=1}^n dist(w, T_i)$.

Without loss of generality, let $T_1, \ldots, T_m$ $(m \leq n)$ be the knowledge bases in $\{T_1, \ldots, T_n\}$ that support l. Then, by hypothesis we have $m > n/2$.

For $i \in [1, m]$, we prove $dist(w_l^-, T_i) \leq dist(w, T_i) - 1$. Note that $[\![T_i]\!]$ is not empty since T_i is consistent. Let $w_i \in [\![T_i]\!]$ such that $dist(w, T_i) = dist(w, w_i)$. Since $w_i \in [\![T_i]\!]$ and $T_i \models l$, $w_i \models l$. Then $dist(w_l^-, w_i) = dist(w, w_i) - 1$ since $w_l^- \models l$ while $w \not\models l$ and w_l^- agrees with w on everything except the truth value of l. Then $dist(w_l^-, T_i) \leq dist(w, w_i) - 1 = dist(w, T_i) - 1$.

For $i \in [m+1, n]$, we prove $dist(w_l^-, T_i) \leq dist(w, T_i) + 1$. Let $w_i \in [\![T_i]\!]$ such that $dist(w, w_i) = dist(w, T_i)$. Then $dist(w_l^-, w_i) \leq dist(w, w_i) + 1$, since w_l^- agrees with w on everything except the truth value of l. As $dist(w_l^-, T_i) \leq dist(w_l^-, w_i)$, we have $dist(w_l^-, T_i) \leq dist(w, w_i) + 1 = dist(w, T_i) + 1$.

Then

$$\begin{aligned}
\sum_{i=1}^{n} dist(w_l^-, T_i) &= \sum_{i=1}^{m} dist(w_l^-, T_i) + \sum_{i=m+1}^{n} dist(w_l^-, T_i) \\
&\leq \sum_{i=1}^{n} dist(w, T_i) - m + (n - m) \\
&< \sum_{i=1}^{n} dist(w, T_i).
\end{aligned}$$

Since $w_l^- \in [\![IC]\!]$, this contradicts the fact that $w \in [\![Merge(\{T_1, \ldots, T_n\}, IC)]\!]$. $\Box$

Theorem 5.1 $\circ_M$ satisfies R1-R6.

Proof: R1–R4 can be easily derived from Theorem 4.1.

R5: If μ or ϕ is inconsistent, R5 can be verified trivially. We now prove the case where μ and ϕ are consistent. Let M be a world such that $M \models (\psi \circ_M \mu) \wedge \phi$. Then $M \models \phi$ and $M \models \psi \circ_M \mu$. Hence $M \in [\![\mu \wedge \phi]\!]$. Assume $M \not\models \psi \circ_M (\mu \wedge \phi)$. Then by the definition of $\circ_M$ and $Merge$, there must be some $M' \in [\![\mu \wedge \phi]\!]$ such that $dist(M', \psi) < dist(M, \psi)$. But since $M' \in [\![\mu]\!]$, this contradicts the fact that $M \models \psi \circ_M \mu$.

R6: Let M be a world such that $M \models \psi \circ_M (\mu \wedge \phi)$. Then $M \models \mu \wedge \phi$. Assume $M \not\models (\psi \circ_M \mu) \wedge \phi$. Then since $M \models \phi$, $M \not\models \psi \circ_M \mu$. Note that since $(\psi \circ_M \mu) \wedge \phi$ is consistent, there must exist at least some M' such that $M' \models (\psi \circ_M \mu) \wedge \phi$. Then $M' \models \psi \circ_M \mu$ and $M' \models \phi$. From $M' \models \psi \circ_M \mu$ we have $M' \models \mu$. It follows that $M' \in [\![\mu \wedge \phi]\!]$. From $M' \models \psi \circ_M \mu$ and $M \not\models \psi \circ_M \mu$ we have $dist(M', \psi) < dist(M, \psi)$. But since $M' \in [\![\mu \wedge \phi]\!]$, this contradicts the fact that $M \models \psi \circ_M (\mu \wedge \phi)$. $\Box$

Theorem 6.1 (Syntactic equivalence of merging dbs)

$$Merge(\{db_1, \ldots, db_n\}, IC) \equiv IC \wedge (db'_1 \vee \cdots \vee db'_m),$$

where db'_i ($i \in [1, m]$) is a set resulting from removing duplicate elements in a maximum sub-multiset C_i of $db_1 \sqcup \cdots \sqcup db_n$ such that $db'_i \wedge IC$ is consistent.

Proof: From the definition of $dist$ it is easy to see that for any $w \in \mathcal{W}$ and db_i ($i \in [1, n]$)

$$dist(w, db_i) = |\{e \mid e \in db_i, e \notin w\}| = |db_i - w \cap db_i| = |db_i| - |w \cap db_i|. \tag{2}$$

Let $U = db_1 \sqcup \cdots \sqcup db_n$.

Let $w \models RHS$ and assume $w \not\models LHS$. By $w \models RHS$, we have $w \models IC$ and $w \models C_k$ for some $k \in [1, m]$. By $w \not\models LHS$, there exists $w' \in [\![IC]\!]$ such that $\sum_{i=1}^n dist(w', db_i) < \sum_{i=1}^n dist(w, db_i)$. Then using equation 2, we obtain

$$\sum_{i=1}^n |db_i| - \sum_{i=1}^n |w' \cap db_i| < \sum_{i=1}^n |db_i| - \sum_{i=1}^n |w \cap db_i|.$$

This means $\sum_{i=1}^n |w' \cap db_i| > \sum_{i=1}^n |w \cap db_i|$, from which it follows that

$$|\bigsqcup_{i=1}^n (w' \cap db_i)| > |\bigsqcup_{i=1}^n (w \cap db_i)|. \tag{3}$$

We now prove that $C_k \subseteq \bigsqcup_{i=1}^n (w \cap db_i)$. Let $e \in C_k$. Then $e \in db_j$ for some $j \in [1, n]$ since $C_k \subseteq U$. And $e \in w$ as $w \models C_k$. Hence $e \in w \cap db_j$. Therefore $C_k \subseteq \bigsqcup_{i=1}^n (w \cap db_i)$. Then by invoking (3), we obtain $|\bigsqcup_{i=1}^n (w' \cap db_i)| > |C_k|$. Note that $\bigsqcup_{i=1}^n (w' \cap db_i)$ is a sub-multiset of U and is consistent with IC. This contradicts the fact that C_k is a maximum sub-multiset of U consistent with IC.

To prove the other direction, let $w \models LHS$ and assume $w \not\models RHS$. Then $w \models IC$. Note that $\bigsqcup_{i=1}^n (w \cap db_i)$ is a sub-multiset of U and is consistent with IC. In addition, $w \models \bigsqcup_{i=1}^n (w \cap db_i)$. Since $w \not\models RHS$, there must exist some C_k ($k \in [1, m]$) such that $|\bigsqcup_{i=1}^n (w \cap db_i)| < |C_k|$. Since $C_k \wedge IC$ is consistent, there exists $w' \models C_k \wedge IC$. Since $w' \models C_k$, by the proof of the other direction, we have $C_k \subseteq \bigsqcup_{i=1}^n (w' \cap db_i)$. It follows that $|\bigsqcup_{i=1}^n (w \cap db_i)| < |\bigsqcup_{i=1}^n (w' \cap db_i)|$. Then using equation 2 and some trivial manipulations, we obtain $\sum_{i=1}^n dist(w', db_i) < \sum_{i=1}^n dist(w, db_i)$. This contradicts the fact that $w \models LHS$. $\Box$

References

[AG85] S. Abiteboul and G. Grahne. Update semantics for incomplete databases. In *Proc. of 11th International Conference on Very Large Data Bases*, pages 1–12, 1985.

[BAP92] M.W. Bright, A.R.Hurson, and S.H. Pakzad. A taxonomy and current issues in multidatabase systems. *Computer*, 25(3):50–59, 1992.

[BJe94] M. Brodie, M. Jarke, and M. Papazoglou (eds.). Proc. of the 2nd intenational conference on cooperative information systems. Toronto, Canada, May, 1994.

[BKMS92] C. Baral, S. Kraus, J. Minker, and V. S. Subrahmanian. Combining knowledge bases consisting of first-order theories. *Computational Intelligence*, 8:45–71, 1992.

[BL86] C. Batini and M. Lenzerini. A comparative analysis of methodologies for database schema integration. *ACM Computing Surveys*, 18(4):323–364, Dec. 1986.

[BM86] F. Bry and R. Manthey. Checking consistency of database constraints: a logical basis. In *Proceedings of the 12th International Conference on Very Large Data Bases*, pages 13–20, 1986.

[Bor85] A. Borgida. Language features for flexible handling of exceptions in information systems. *ACM Trans. on Database Systems*, 10:563–603, 1985.

[BV84] C. Beeri and M. Vardi. Formal systems for tuple and equality generating dependencies. *SIAM J. Comput.*, 13(1):76–98, Feb. 1984.

[Dal88] Mukesh Dalal. Investigations into a theory of knowledge base revision: preliminary report. In *Proceedings of the 7th National Conference of the American Association for Artificial Intelligence*, pages 475–479, 1988.

[EP90] A. Elmagarmid and C. Pu. *ACM Computing Surveys* special issue on heterogeneous databases. *ACM Computing Surveys*, 22(3), Sept. 1990.

[FH88] R. Fagin and J. Halpern. Belief, awareness, and limited reasoning. *Artificial Intelligence*, 34:39–76, 1988.

[FKUV86] R. Fagin, G. M. Kuper, J. D. Ullman, and M. Vardi. Updating logical databases. *Advances in Computing Research*, 3:1–18, 1986.

[FUV83] R. Fagin, J. D. Ullman, and M. Vardi. On the semantics of updates in databases. In *2nd ACM SIGACT-SIGMOD Symposium on Principles of Database Systems*, pages 352–365, 1983.

[G8̈8] Peter Gärdenfors. *Knowledge in Flux - Modeling the Dynamics of Epistemic States*. MIT press, 1988.

[Gru94] J. Grudin. Groupware and social dynamics: Eight challenges for developers. *Communications of the ACM*, 37(1):92–105, Jan. 1994.

[HM85] J. Halpern and Y. Moses. A guide to the modal logics of knowledge and belief. In *Proceedings IJCAI-85*, pages 480–490, 1985. A complete version appears in *Artificial Intelligence*, 54(3):319-379, 1992, under the title "A guide to completeness and complexity for modal logics of knowledge and belief".

[HM92] K. Huff and J. Mylopoulos. Development of intelligent information systems: A workshop summary. *International Journal of Intelligent & Cooperative Information Systems*, 1(2):233–248, 1992.

[IV89] T. Imielinski and K. Vadaparty. Complexity of query processing in databases with or-objects. In *Proc. PODS'89*, 1989.

[KM91a] H. Katsuno and A. O. Mendelzon. On the difference between updating a knowledge base and revising it. In *Proceedings of the 2nd International Conference on Principles of Knowledge Representation and Reasoning*, pages 387–394, 1991.

[KM91b] H. Katsuno and A. O. Mendelzon. Propositional knowledgebase revision and minimal change. *Artificial Intelligence*, 52:263–294, 1991.

[KW85] A. Keller and M. Wilkins. On the use of an extended relational model to handle changing incomplete information. *IEEE Transactions on Software Engineering*, SE-11:620–633, 1985.

[Lin96] Jinxin Lin. Integration of weighted knowledge bases. *Artificial Intelligence*, 83/2:363–378, 1996.

[LM94] Jinxin Lin and Alberto O. Mendelzon. Knowledge base merging by majority. KRR Technical Report, Dept. of Computer Science, Univ. of Toronto, Toronto M5S 1A4, 1994.

[MIR94] R. J. Miller, Y. E. Ioannidis, and R. Ramakrishnan. Schema equivalence in heterogeneous systems: bridging theory and practice. In *Proc. of the Int'l Conference on Extending Database Technology*, 1994.

[Rei78] R. Reiter. On closed world databases. In H. Gallaire and J. Minker, editors, *Logic and Data Bases*, pages 119–140. Plenum, New York, 1978.

[SL90] A. P. Sheth and J. A. Larson. Federated database systems for managing distributed, heterogeneous, and autonomous databases. *ACM Computing Surveys*, 22(3):183–235, Sept. 1990.

[Sub94] V. S. Subrahmanian. Amalgamating knowledge bases. *ACM Transactions on Database Systems*, 19(2):291–331, 1994.

[WL92] C. C. Woo and F. H. Lochovsky. Knowledge communication in intelligent information systems. *Intenational Journal of Intelligent & Cooperative Information Systems*, 1:203–228, 1992.