In Proceedings of 7th International Conference on Extending Database Technology (EDBT 2000), Konstanz, Germany, March 27-31, 2000. LNCS. Springer-Verlag.

A Data Model for Semistructured Data with Partial and Inconsistent Information

Mengchi Liu[1] and Tok Wang Ling[2]

[1] Department of Computer Science, University of Regina
Regina, Saskatchewan, Canada S4S 0A2
mliu@cs.uregina.ca

[2] School of Computing, National University of Singapore
Lower Kent Ridge Road, Singapore 119260
lingtw@comp.nus.edu.sg

Abstract. With the recent popularity of the World Wide Web, an enormous amount of heterogeneous information is now available online. As a result, information about real world objects may spread over different data sources and may be partial and inconsistent. How to manipulate such semistructured data is thus a challenge. Previous work on semistructured data mainly focuses on developing query languages and systems to retrieve semistructured data. Relatively less attention has been paid to the manipulation of such data. In order to manipulate such semistructured data, we need a data model that is more expressive than the existing graph-based and tree-based ones to account for the existence of partial and inconsistent information from different data sources. In this paper, we propose such a data model for semistructured data that allows partial and inconsistent information and discuss how to manipulate such semistructured data.

1 Introduction

With the recent popularity of the World Wide Web, an enormous amount of heterogeneous information is now available online. As a result, information about real world objects may spread over different data sources. Such information may be partial (incomplete) and inconsistent in the data that come from various sources. The need to manipulate such kind of semistructured data has become more and more important. Previous work on semistructured data mainly focuses on developing query languages and systems to retrieve semistructured data, such as W3QS [12], WebSQL [26], WebLog [19], UnQL [9], Lorel [2]. WebOQL [4], and Florid [15]. For surveys, see [1, 14]. How to manipulate semistructured data coming from various sources has also received some attention [3, 6, 11, 10, 21, 28]. Earlier proposals are based on simple graph-based data models such as OEM [29] or tree-based data models such as the one in [8], which fail to account for the existence of partial and inconsistent information. For example, most academic people should have a Bibtex database to keep references. While two or more persons work together on a paper, an immediate problem is how to merge multiple Bibtex databases. Although all Bibtex databases have similar structures, the values in these databases may be partial or inconsistent. The typical case is the *authors* of a paper. Someone likes to list all authors in full names, but others may just indicate the first one or two authors. Even for one author, the order of first name (or initial) and last name may be different. Furthermore, partial data for the same record may be missing or inconsistent. Therefore, how to obtain information as complete as possible from these sources, i.e., their union, how to get the common information from different data sources, i.e., their intersection, or how to find the information that is in the first data source but not in the second one, i.e., their difference, are interesting questions. In other words, we need a set of operations to manipulate such semistructured data.

In the past decade, the similar problem, that is, manipulating data with partial and/or incomplete information, has been investigated in depth in the context of relational, complex object and multiple databases [5, 7, 13, 16–18, 22, 23, 27, 30]. Specific operations such as union, intersection [5] and join [7] are introduced to manipulate such data. However, these works focus on typed data and support only homogeneous sets and tuples. Thus, it is difficult to directly apply them on semistructured data, where although the data may have some structure, the structure is not as rigid, regular, or complete as that required by traditional database systems [1].

In this paper, we propose a novel semistructured data model to reflect the existence of partial and inconsistent information. We also study how to manipulate such semistructured data with a set of operations. In particular, we focus on three operations, *union*, *intersection* and *difference*, and discuss their semantic properties.

The rest of the paper is organized as follows. Section 2 defines our semistructured data model. Section 3 discusses how to manipulate semistructured data and the semantic properties. Section 4 summarizes and points out further research issues.

2 Semistructured Data Models

In this section, we introduce our data model to represent semistructured data. The data model consists of *semistructured data* which are built from *objects*. We assume the existence of a set $\mathcal{M}$ of markers, a set $\mathcal{A}$ of attribute labels, and a set $\mathcal{U}$ of constants such that $\mathcal{M}$ and $\mathcal{A} \cup \mathcal{U}$ are disjoint.

Definition 1. The notion of *objects* is defined as follows:

(1) Constants in $\mathcal{U}$ are objects called *atomic* objects.
(2) Markers in $\mathcal{M}$ are objects called *marker* objects.
(3) There is a special object $\perp$.
(4) If $O_1, ..., O_n, (n > 1)$ are distinct objects, then $O_1|...|O_n$ is an object called *or-value*.
(5) If $O_1, ..., O_n, (n \geq 0)$ are distinct objects, then $\langle O_1, ..., O_n\rangle$ is an object called *partial set*.

(6) If $O_1, ..., O_n, (n \geq 0)$ are distinct objects, then $\{O_1, ..., O_n\}$ is an object called *complete set*.
(7) If $O_1, ..., O_n, (n \geq 0)$ are objects and $A_1, ..., A_n$ are distinct attribute labels, then $O \equiv [A_1 \Rightarrow O_1, ..., A_n \Rightarrow O_n]$ is an object called *tuple*. We denote O_i by $O.A_i$. We also assume that $O.A = \perp$ for an attribute A not in $\{A_1, ..., A_n\}$.

We use $\perp$ for null/unknown object. For example, in a tuple representing a person, if the age of the person is unknown, then we use $[..., age \Rightarrow \perp, ...]$.

As we are dealing with the manipulations of semistructured data from different sources, it is possible that we have conflicting information. In this case, we use an *or-value* to record the conflicting result. For example, the or-value 21|22 in the tuple [..., age $\Rightarrow$ 21|22, ...] implies the age is 21 or 22 as there is a conflict right now and it is not clear that any one of these two values is correct. It is up to the user to solve the conflicts.

The *markers* are used to identify/refer to an object uniquely. They are similar to *object identifier* in OEM (Object Exchange Model) [2], but different in nature. An *object identifier* is attached to each object, even to each constant in OEM. In contrast, *markers* in our data model can be used to identify complex objects. For example, in a Bibtex database, markers correspond to the keys [20]; in a Web page, markers correspond to URLs. See Examples 1 and 2.

Besides null/unknown and inconsistent values, it is quite common that partial rather than complete information is provided for a set. For example, in a Bibtex file, someone may only give partial authorship such as *"Bob and others"* to indicate *"Bob, et al."* are the authors. In this case, the set that contains *"Bob"* is partial and should be represented with $\langle"Bob"\rangle$ to indicate that the set only provides partial authorship. In particular, the empty partial set $\langle\rangle$ indicates that it is a set but we do not know what is in it. It contains more information than $\perp$. On the other hand, if we know the complete authorship such as *"Bob and Tom"*, then the set that contains *"Bob"* and *"Tom"* is complete and should be represented in our data model as $\{"Bob", "Tom"\}$. The empty set $\{\}$ indicates there is nothing in it, which is quite different from $\langle\rangle$. The notions of partial and complete set were first introduced in ROL [24] and later extended in Relationlog [25]. They are used to represent the open and closed world assumption on sets in a database.

We now define the notion of semistructured data as follows.

Definition 2. Let $m_1, ..., m_n \in \mathcal{M}$ be markers with $n > 0$, $m \equiv m_1|...|m_n$, and O an object. Then $m : O$ is a *semistructured data*. When $n = 1$ and O does not contain or-values or marked objects, it is called *real*. Otherwise, it is called *virtual*.

Real semistructured data are the ones that can exist in the real world while virtual ones are those generated with our operations, union, intersection, and difference as defined in Section 3. Just like in the object-oriented paradigm, objects with similar properties are grouped into a class.

For example, a Bibtex file can be viewed as a set of real semistructured data while a Web page can be viewed as a single real semistructured data.

Example 1. Consider the following bib file with two cross-reference entries:

```
@InBook{Bob,
  author   = "Bob and others",
  title    = "Oracle",
  crossref = DB}

@Book{DB,
  booktitle = "Database",
  editor    = "John",
  year      = 1999}
```

They can be represented as two semistructured data as follows:

$$Bob\colon [type \Rightarrow "InBook", author \Rightarrow \langle "Bob" \rangle, title \Rightarrow "Oracle", crossref \Rightarrow DB]$$
$$DB\colon [type \Rightarrow "Book", booktitle \Rightarrow "Database", editor \Rightarrow "John", year \Rightarrow 1999]$$

where Bob and DB are markers, and $\langle "Bob" \rangle$ is a partial set which indicates $"Bob"$ is one of the authors.

Example 2. Consider the following simplified Web page:

```
<html>
<head><title>CSDept</title></head>
<body>
<h2>People</h2>
<ul>
<li><a href="faculty.html">      Faculty </a>
<li><a href="staff.html">        Staff   </a>
<li><a href="students.html">     Students</a>
</ul>
<h2><a href="programs.html"> Programs<a></h2>
<h2><a href="research.html"> Research<a></h2>
</body>
</html>
```

Suppose *www.cs.uregina.ca* is the URL of this web page. Then it can be represented as a semistructured data in our data model as follows:

$$\begin{array}{l} www.cs.uregina.ca : [\\ \quad Title \Rightarrow CSDept, \\ \quad People \Rightarrow \{[Faculty \Rightarrow faculty.html], \\ \qquad\qquad [Staff \Rightarrow staff.html], \\ \qquad\qquad [Students \Rightarrow students.html]\}, \\ \quad Programs \Rightarrow programs.html, \\ \quad Research \Rightarrow research.html \\ \quad] \end{array}$$

where *www.cs.uregina.ca, faculty.html, staff.html, students.html, programs.html, research.html* are markers, and the rest are attribute labels.

Note that our semistructured data model can capture more information than the existing semistructured data models such as OEM [2] and labeled tree model [9] since null/unknown, *or-value*, *partial* and *complete* set objects are supported.

3 Manipulation of Semistructured Data

In this section, we discuss how to manipulate semistructured data. Consider the following semistructured data that represent two Bibtex items from two different bib files:

$$B80 : [type \Rightarrow "Article", title \Rightarrow "Oracle", author \Rightarrow "Bob", year \Rightarrow 1980]$$
$$B82 : [type \Rightarrow "Article", title \Rightarrow "Oracle", year \Rightarrow 1980, journal \Rightarrow "IS"]$$

The first tuple has a null value for the attribute $journal$ whereas the second has a null value for the attribute $author$. They have different markers. Let us assume that articles can be identified by their type and title. Then the two semistructured data can be combined, that is, their union, based on the *type* and *title* attributes to obtain more information shown as follows:

$$\begin{aligned} B80|B82 : [&type \Rightarrow "Article", title \Rightarrow "Oracle", author \Rightarrow "Bob",\\ &year \Rightarrow 1980, journal \Rightarrow "IS"] \end{aligned}$$

where $B80|B82$ means that the two Bibtex terms from two different bib files have different markers that refer to the same article.

Similarly, the information common in them, that is, their intersection based on the *type* and *title* attributes, is as follows:

$$\perp : [type \Rightarrow "Article", title \Rightarrow "Oracle", year \Rightarrow 1980]$$

where $\perp$ as a marker indicates that the two Bibtex terms have different markers that refer to the same article but we do not care what they are in terms of their common information.

Finally, the information in the first semistructured data but in not the second except the type and title, that is, the difference between the first and second based on the *type* and *title* attributes, is as follows:

$$B80 : [type \Rightarrow "Article", title \Rightarrow "Oracle", author \Rightarrow "Bob"]$$

In order to formalize these operations on semistructured data, we first introduce the following notions.

Definition 3. An object O_1 is *less informative* than an object O_2, denoted by $O_1 \trianglelefteq O_2$, if and only if one of the following holds:

(1) $O_1 = O_2$

(2) $O_1 = \perp$
(3) $O_1 \equiv O'_1|...|O'_m$ and $O_2 \equiv O'_1|...|O'_n$ with $1 \leq m < n$
(4) O_1 is a partial set and O_2 is a partial or complete set, and for each $O \in O_1 - O_2$, there exists $O' \in O_2 - O_1$ such that $O \trianglelefteq O'$
(5) O_1 and O_2 are both tuples such that for each attribute A in O_1, $O_1.A \trianglelefteq O_2.A$

The less informative relationship is used to express the fact that one object is part of another object. It is used to determine when two objects can be manipulated and to show the properties of the operations. The special treatment of sets in (4) of Definition 3 guarantees that the less informative relationship is a partial order as in [25].

The following are several examples:

$a \trianglelefteq a$	$\{a\} \trianglelefteq \{a\}$	$[A \Rightarrow a] \trianglelefteq [A \Rightarrow a]$	by(1)
$\perp \trianglelefteq a$	$\perp \trianglelefteq \{a\}$	$\perp \trianglelefteq [A \Rightarrow a]$	by(2)
$a_1 \trianglelefteq a_1\|a_2$	$a_1\|a_2 \trianglelefteq a_1\|a_2\|a_3$	$a_1\|a_2\|a_3 \trianglelefteq a_1\|a_2\|a_3$	by (3) or (1)
$\langle a_1\rangle \trianglelefteq \langle a_1, a_2\rangle$	$\langle a_1\rangle \trianglelefteq \{a_1, a_2\}$	$\{a_1, a_2\} \trianglelefteq \{a_1, a_2\}$	by (4) or (1)
$[A \Rightarrow a] \trianglelefteq [A \Rightarrow a, B \Rightarrow b]$	$[A \Rightarrow \langle a_1\rangle] \trianglelefteq [A \Rightarrow \langle a_1, a_2\rangle, B \Rightarrow b]$		by (5)

Definition 4. A semistructured data $m_1 : O_1$ is *less informative* than a semistructured data $m_2 : O_2$, denoted by $m_1 : O_1 \trianglelefteq m_2 : O_2$, if and only if $m_1 \trianglelefteq m_2$ and $O_1 \trianglelefteq O_2$.

We extend the notion to sets of semistructured data as follows.

Definition 5. Let S_1 and S_2 be sets of semistructured data. Then S_1 is *less informative* than S_2, denoted by $S_1 \trianglelefteq S_2$ if and only if for each $D_1 \in S_1 - S_2$ there exists a $D_2 \in S_2 - S_1$ such that $D_1 \trianglelefteq D_2$.

It turns out that the less informative relationship has the following property.

Proposition 1. The less informative relationship is a partial order.

Definition 6. Let O_1 and O_2 be two objects and $K = \{A_1, ..., A_m\}$ a set of attributes. Then O_1 and O_2 are *compatible* with respect to K if and only if one of the following holds:

(1) both are constants and are equal
(2) both are markers and are equal
(3) both are or-values that do not contain $\perp$ and are equal set-wise.
(4) both are complete sets and are equal
(5) both are tuples and are equal or $O_1.A_i$ and $O_2.A_i$ are compatible with respect to K for $1 \leq i \leq m$

In other words, two objects are compatible with respect to K if they are other than $\perp$ and are equal, or they have compatible K value when they are tuples.

In our data model, the set K of attributes in the above definition is similar to the notion of the key in the relational data model, but can be non-atomic. It is used to identify objects. If two different objects are compatible, then we treat them as the different aspects of the same object so that we can manipulate them using operations to be introduced shortly.

The following pairs of objects are compatible with respect to $K = \{A, B\}$:

$$\begin{array}{lll}
a & \text{and } a & \text{by (1)} \\
m & \text{and } m & \text{by (2)} \\
a|b & \text{and } b|a & \text{by (3)} \\
\{a_1, a_2\} & \text{and } \{a_1, a_2\} & \text{by (4)} \\
[A \Rightarrow a_1, B \Rightarrow \{b_1, b_2\}, C \Rightarrow c_1] & \text{and } [A \Rightarrow a_1, B \Rightarrow \{b_1, b_2\}, C \Rightarrow c_2] & \text{by (5)}
\end{array}$$

The following pairs of objects are not compatible with respect to $K = \{A, B\}$:

$$\begin{array}{lll}
\perp & \text{and} & \perp \\
a & \text{and} & \perp \\
a_1 & \text{and} & a_2 \\
a_1|a_2 & \text{and} & a_1|a_2|a_3 \\
\langle a_1 \rangle & \text{and} & \langle a_1, a_2 \rangle \\
\langle a_1 \rangle & \text{and} & \{a_1, a_2\} \\
\langle a_1 \rangle & \text{and} & \{a_2, a_3\}
\end{array}$$

$[A \Rightarrow a_1, B \Rightarrow \perp, C \Rightarrow \{c_1\}]$ and $[A \Rightarrow a_1, B \Rightarrow \perp, C \Rightarrow \{c_1\}]$
$[A \Rightarrow \perp, B \Rightarrow b_1, C \Rightarrow \{c_1\}]$ and $[A \Rightarrow \perp, B \Rightarrow b_2, C \Rightarrow \{c_1\}]$

Two $\perp$ are not compatible because two different occurrences may not denote the same real-world entity; a and $\perp$ are not compatible because $\perp$ may not mean a; $a_1|a_2$ and $a_1|a_2|a_3$ are not compatible because $a_1|a_2$ means the value is either a_1 or a_2 while $a_1|a_2|a_3$ means the value is either a_1, a_2 or a_3; $\langle a_1 \rangle$ and $\langle a_1, a_2 \rangle$ are not compatible because our knowledge about the two partial sets is different: $\langle a_1 \rangle$ means the set contains a_1 while $\langle a_1, a_2 \rangle$ means the set contains a_1 and a_2.

As shown in the above examples, two identical objects may not be compatible if they involve $\perp$.

Definition 7. Let $S_1 \equiv m_1 : O_1$ and $S_2 \equiv m_2 : O_2$ be two semistructured data and $K = \{A_1, ..., A_k\}$, $k \geq 1$. Then S_1 and S_2 are *compatible* with respect to K if and only if O_1 and O_2 are compatible with respect to K.

Consider the two semistructured data again:

$$B80 : [type \Rightarrow \text{"}Article\text{"}, title \Rightarrow \text{"}Oracle\text{"}, author \Rightarrow \text{"}Bob\text{"}, year \Rightarrow 1980]$$
$$B82 : [type \Rightarrow \text{"}Article\text{"}, title \Rightarrow \text{"}Oracle\text{"}, year \Rightarrow 1980, journal \Rightarrow \text{"}IS\text{"}]$$

They are compatible with respect to $K = \{type, title\}$, but not compatible with respect to $K' = \{type, title, author\}$ or $K'' = \{type, title, author, year\}$.

The main goal of this paper is to define how to manipulate semistructured data when given a set of attributes. Now we introduce the *union* operation which is used to get more information from two objects representing the same real-world entity. It is similar to the union operation in [5], the join operation in [7], and the grouping operation in [25], but it handles partial and inconsistent information.

Definition 8. Let $K = \{A_1, ..., A_m\}$ be a set of attributes, and O_1 and O_2 two objects. Then their *union* based on K, denoted by $O_1 \cup_K O_2$, is defined as follows:

(1) $O_1 \cup_K O_1 = O_1$, $O_1 \cup_K \perp = O_1$
(2) If O_1 and O_2 are distinct partial sets, then
$$O_1 \cup_K O_2 = \langle O \mid O \in O_1, \nexists O' \in O_2,\ O \text{ and } O' \text{ are compatible wrt } K, \text{ or } O \in O_2, \nexists O' \in O_1,\ O \text{ and } O' \text{ are compatible wrt } K, \text{ or } \exists O' \in O_1, \exists O'' \in O_2,\ O' \text{ and } O'' \text{ are compatible wrt } K,\ O{=}O' \cup_K O'' \rangle$$
(3) If O_1 is a partial set and O_2 is a complete set such that $O_1 \trianglelefteq O_2$, then $O_1 \cup_K O_2 = O_2$
(4) If O_1 and O_2 are distinct tuples that are compatible with respect to $A_1, ..., A_m$ and $A_1, ..., A_n$ are all attributes in them, then
$$O_1 \cup_K O_2 = [A_1 \Rightarrow O_1.A_1 \cup_K O_2.A_1, ..., A_n \Rightarrow O_1.A_n \cup_K O_2.A_n]$$
(5) In all other cases, $O_1 \cup_K O_2 = O_1|O_2$

Example 3. The following are several examples where $K = \{A, B\}$:

$a \cup_K a = a$	by (1)
$\{a\} \cup_K \{a\} = \{a\}$	by (1)
$[C \Rightarrow c] \cup_K [C \Rightarrow c] = [C \Rightarrow c]$	by (1)
$a \cup_K \perp = a$	by (1)
$\langle a\rangle \cup_K \langle b\rangle = \langle a, b\rangle$	by (2)
$\langle a_1, a_2\rangle \cup_K \{a_1, a_2, a_3\} = \{a_1, a_2, a_3\}$	by (3)
$[A \Rightarrow a_1, B \Rightarrow b_1, C \Rightarrow \langle c_1\rangle] \cup_K [A \Rightarrow a_1, B \Rightarrow b_1, C \Rightarrow \{c_1, c_2\}]$	
$= [A \Rightarrow a_1, B \Rightarrow b_1, C \Rightarrow \{c_1, c_2\}]$	by (4)
$a_1 \cup_K a_2 = a_1|a_2$	by (5)
$a_1 \cup_K \{a_1\} = a_1|\{a_1\}$	by (5)
$a_1 \cup_K [A \Rightarrow a_1] = a_1 \mid [A \Rightarrow a_1]$	by (5)
$a_1 \cup_K a_2|a_3 = a_1|a_2|a_3$	by (5)
$\{a_1, a_2\} \cup_K \{a_1, a_2, a_3\} = \{a_1, a_2\}|\{a_1, a_2, a_3\}$	by (5)

Note that the union of two partial sets is still a partial set as we still do not know if the result is complete. The union of two distinct complete sets however generates an or-value as shown in the last case above. Assuming that these two sets represent authors of the same paper, then their union indicates there is a conflict in them. Using the union of the traditional set theory cannot detect such a conflict.

To find common information in objects that represent the same real-world entity, we need to use the *intersection* operation which is defined as follows.

Definition 9. Let $K = \{A_1, ..., A_m\}$ be a set of attributes, and O_1 and O_2 two objects. Then their *intersection* based on K, denoted by $O_1 \cap_K O_2$, is defined as follows:

(1) $O_1 \cap_K O_1 = O_1$
(2) If O_1 and O_2 are distinct or-values such that $O'_1, ..., O'_n$ are common in them with $n \geq 1$, then $O_1 \cap_K O_2 = O'_1|...|O'_n$
(3) If O_1 and O_2 are distinct partial sets or one of them is complete, then
$O_1 \cap_K O_2 = \langle O \mid \exists O' \in O_1, \exists O'' \in O_2,\ O'$ and O'' are compatible wrt K, $O = O' \cap_K O''\rangle$
(4) If O_1 and O_2 are distinct complete sets, then
$O_1 \cap_K O_2 = \{O \mid \exists O' \in O_1, \exists O'' \in O_2,\ O'$ and O'' are compatible wrt K, $O = O' \cap_K O''\}$
(5) If O_1 and O_2 are distinct tuples that are compatible with respect to K and $A_1, ..., A_n$ are all attributes in them, then
$O_1 \cap_K O_2 = [A_1 \Rightarrow O_1.A_1 \cap_K O_2.A_1, ..., A_n \Rightarrow O_1.A_n \cap_K O_2.A_n]$
(6) In all other cases, $O_1 \cap_K O_2 = \perp$

Note that the intersection of two partial sets or a partial set and a complete set is a partial set as we are not sure if the result is complete. However, the intersection of complete sets is a complete set as we know exactly what are the common elements in two complete sets.

Example 4. The following are several examples where $K = \{A, B\}$:

$a \cap_K a = a$ by (1)
$\{a\} \cap_K \{a\} = \{a\}$ by (1)
$[C \Rightarrow c] \cap_K [C \Rightarrow c] = [C \Rightarrow c]$ by (1)
$a_1 \cap_K a_1|a_2 = a_1$ by (2)
$\langle a_1, a_2\rangle \cap_K \langle a_1, a_2, a_3\rangle = \langle a_1, a_2\rangle$ by (3)
$\langle a_1, a_2\rangle \cap_K \{a_1, a_2, a_3\} = \langle a_1, a_2\rangle$ by (3)
$\langle a_1, a_2\rangle \cap_K \{a_3\} = \langle\rangle$ by (3)
$\{a_1, a_2\} \cap_K \{a_1, a_2, a_3\} = \{a_1, a_2\}$ by (4)
$\{a_1, a_2\} \cap_K \{a_3\} = \{\}$ by (4)
$[A \Rightarrow a_1, B \Rightarrow b_1, C \Rightarrow \langle c_1\rangle] \cap_K [A \Rightarrow a_1, B \Rightarrow b_1, C \Rightarrow \{c_1, c_2\}]$
$= [A \Rightarrow a_1, B \Rightarrow b_1, C \Rightarrow \langle c_1\rangle]$ by (5)
$a_1 \cap_K \perp = \perp$ by (6)
$a_1 \cap_K a_2 = \perp$ by (6)
$a_1 \cap_K [A \Rightarrow a_1] = \perp$ by (6)
$[A \Rightarrow a_1, B \Rightarrow b_1, C \Rightarrow c_1] \cap_K [A \Rightarrow a_2, B \Rightarrow b_2, C \Rightarrow c_2] = \perp$ by (6)

Proposition 2. The union and intersection operations are commutative.

To obtain information in the first object but not in the second, we can use the *difference* operation defined as follows.

Definition 10. Let $K = \{A_1, ..., A_m\}$ be a set of attributes, and O_1 and O_2 two objects. Then their *difference* based on K, denoted by $O_1 -_K O_2$, is defined as follows:

(1) If O_1 is an object other than a partial or complete set, then $O_1 -_K O_1 = \perp$
(2) If O_1 and O_2 are distinct or-values such that $O'_1, ..., O'_n$ are all objects in O_1 but not in O_2 with $n \geq 1$, then $O_1 -_K O_2 = O'_1|...|O'_n$
(3) If O_1 is a partial set and O_2 is a partial or complete set, then
$$O_1 -_K O_2 = \langle O \mid O \in O_1, \nexists O' \in O_2,\ O \text{ and } O' \text{ are compatible wrt } K, \text{ or } \exists O' \in O_1, \exists O'' \in O_2,\ O' \text{ and } O'' \text{ are compatible wrt } K,\ O{=}O' -_K O'' \rangle$$
(4) If O_1 is a complete set and O_2 is a partial or complete set, then
$$O_1 -_K O_2 = \{O \mid O \in O_1, \nexists O' \in O_2,\ O \text{ and } O' \text{ are compatible wrt } K, \text{ or } \exists O' \in O_1, \exists O'' \in O_2,\ O' \text{ and } O'' \text{ are compatible wrt } K,\ O{=}O' -_K O'' \}$$
(5) If O_1 and O_2 are distinct tuples that are compatible with respect to K and $B_1, ..., B_n$ all attributes in O_1 other than $A_1, ..., A_m$, then
$$O_1 -_K O_2 = [A_1 \Rightarrow O_1.A_1, ..., A_m \Rightarrow O_1.A_m, B_1 \Rightarrow O_1.B_1 -_K O_2.B_1, ...B_n \Rightarrow O_1.B_n -_K O_2.B_n]$$
(6) In all other cases, $O_1 -_K O_2 = O_1$

Note that we keep the value of K in the result as it provides the identity for the result.

Example 5. The following are several examples where $K = \{A, B\}$:

$a -_K a = \perp$ by (1)
$a -_K \perp = a$ by (6)
$a_1|a_2 -_K a_1 = a_2$ by (2)
$\langle a_1, a_2\rangle -_K \langle a_2, a_3\rangle = \langle a_1\rangle$ by (3)
$\langle a_1, a_2\rangle -_K \{a_1, a_2\} = \langle\rangle$ by (3)
$\{a_1, a_2\} -_K \langle a_3\rangle = \{a_1, a_2\}$ by (4)
$\{a_1, a_2\} -_K \{a_1, a_2\} = \{\}$ by (4)
$[A \Rightarrow a_1, B \Rightarrow b_1, C \Rightarrow c_1|c_2, D \Rightarrow \{d_1, d_2\}] -_K [A \Rightarrow a_1, B \Rightarrow b_1, C \Rightarrow c_2]$
$= [A \Rightarrow a_1, B \Rightarrow b_1, C \Rightarrow c_1, D \Rightarrow \{d_1, d_2\}]$ by (5)
$[A \Rightarrow a_1, B \Rightarrow \langle b_1\rangle] -_K [A \Rightarrow a_2, B \Rightarrow \langle b_2\rangle, C \Rightarrow c_2]$
$= [A \Rightarrow a_1, B \Rightarrow \langle b_1\rangle]$ by (6)

Definition 11. Let $K = \{A_1, ..., A_m\}$ be a set of attributes, $D_1 = m_1 : O_1$ and $D_2 = m_2 : O_2$ two semistructured data. Then their *union*, *intersection* and *difference* based on K, denoted by $D_1 \cup_K D_2$, $D_1 \cap_K D_2$, and $D_1 -_K D_2$, are defined as follows:

$$D_1 \cup_K D_2 = m_1 \cup_K m_2 : O_1 \cup_K O_2$$
$$D_1 \cap_K D_2 = m_1 \cap_K m_2 : O_1 \cap_K O_2$$
$$D_1 -_K D_2 = m_1 -_K m_2 : O_1 -_K O_2$$

Definition 12. Let $K = \{A_1, ..., A_m\}$ be a set of attributes, S_1 and S_2 two sets of semistructured data. Then their *union*, *intersection* and *difference* based on K, denoted by $S_1 \cup_K S_2$, $S_1 \cap_K S_2$, and $S_1 -_K S_2$ are defined as follows:

$$
\begin{aligned}
S_1 \cup_K S_2 = & \{D_1 \in S_1 \mid \nexists D_2 \in S_2 \text{ such that } D_1 \text{ and } D_2 \text{ are compatible wrt } K\} \\
& \cup \{D_2 \in S_2 \mid \nexists D_1 \in S_1 \text{ such that } D_1 \text{ and } D_2 \text{ are compatible wrt } K\} \\
& \cup \{D_1 \cap_K D_2 \mid D_1 \in S_1, D_2 \in S_2 \text{ such that } D_1 \text{ and } D_2 \text{ are compatible wrt } K\} \\
S_1 \cap_K S_2 = & \{D_1 \cap_K D_2 \mid D_1 \in S_1, D_2 \in S_2 \text{ such that } D_1 \text{ and } D_2 \text{ are compatible wrt } K\} \\
S_1 -_K S_2 = & \{D_1 \in S_1 \mid \nexists D_2 \in S_2 \text{ such that } D_1 \text{ and } D_2 \text{ are compatible wrt } K\} \\
& \cup \{D_1 -_K D_2 \mid D_1 \in S_1, D_2 \in S_2 \text{ such that } D_1 \text{ and } D_2 \text{ are compatible wrt } K\}
\end{aligned}
$$

Example 6. Consider the following two sets of semistructured data which are essentially two Bibtex files, one of which (S_1) contains pure journal papers and the other (S_2) contains both journal and conference papers.

$$
\begin{aligned}
S_1 = \{ & B80: [type \Rightarrow \text{"}Article\text{"}, title \Rightarrow \text{"}Oracle\text{"}, auth \Rightarrow \text{"}Bob\text{"}, year \Rightarrow 1980], \\
& S78: [type \Rightarrow \text{"}Article\text{"}, title \Rightarrow \text{"}Ingres\text{"}, auth \Rightarrow \text{"}Sam\text{"}, jnl \Rightarrow \text{"}TODS\text{"}], \\
& A78: [type \Rightarrow \text{"}Article\text{"}, title \Rightarrow \text{"}Datalog\text{"}, auth \Rightarrow \text{"}Ann\text{"}, year \Rightarrow 1978], \\
& J88: [type \Rightarrow \text{"}Article\text{"}, title \Rightarrow \text{"}DOOD\text{"}, auth \Rightarrow \text{"}Joe\text{"}, jnl \Rightarrow \text{"}JLP\text{"}]\} \\
S_2 = \{ & B82: [type \Rightarrow \text{"}Article\text{"}, title \Rightarrow \text{"}Oracle\text{"}, auth \Rightarrow \text{"}Bob\text{"}, year \Rightarrow 1980], \\
& A78: [type \Rightarrow \text{"}Article\text{"}, title \Rightarrow \text{"}Datalog\text{"}, auth \Rightarrow \text{"}Tom\text{"}, year \Rightarrow 1978], \\
& P90: [type \Rightarrow \text{"}Article\text{"}, title \Rightarrow \text{"}DOOD\text{"}, auth \Rightarrow \text{"}Pam\text{"}, jnl \Rightarrow \text{"}JLP\text{"}], \\
& S85: [type \Rightarrow \text{"}Article\text{"}, title \Rightarrow \text{"}NF2\text{"}, auth \Rightarrow \text{"}Sam\text{"}, year \Rightarrow 1985], \\
& T79: [type \Rightarrow \text{"}InProc\text{"}, title \Rightarrow \text{"}RDB\text{"}, auth \Rightarrow \text{"}Tom\text{"}, conf \Rightarrow \text{"}PODS\text{"}], \\
& A75: [type \Rightarrow \text{"}InProc\text{"}, title \Rightarrow \text{"}NF2\text{"}, auth \Rightarrow \text{"}Ann\text{"}, year \Rightarrow 1975], \\
& S76: [type \Rightarrow \text{"}InProc\text{"}, title \Rightarrow \text{"}Ingres\text{"}, auth \Rightarrow \text{"}Sam\text{"}, conf \Rightarrow \text{"}EDBT\text{"}]\}
\end{aligned}
$$

Let $K = \{type, title\}$. Then their union, intersection and difference based on K are as follows:

$$
\begin{aligned}
& S_1 \cup_K S_2 \\
& = \{S78: [type \Rightarrow \text{"}Article\text{"}, title \Rightarrow \text{"}Ingres\text{"}, auth \Rightarrow \text{"}Sam\text{"}, jnl \Rightarrow \text{"}TODS\text{"}]\} \\
& \cup \{S85: [type \Rightarrow \text{"}Article\text{"}, title \Rightarrow \text{"}NF2\text{"}, auth \Rightarrow \text{"}Sam\text{"}, year \Rightarrow 1985], \\
& \quad T79: [type \Rightarrow \text{"}InProc\text{"}, title \Rightarrow \text{"}RDB\text{"}, auth \Rightarrow \text{"}Tom\text{"}, conf \Rightarrow \text{"}PODS\text{"}], \\
& \quad A75: [type \Rightarrow \text{"}InProc\text{"}, title \Rightarrow \text{"}NF2\text{"}, auth \Rightarrow \text{"}Ann\text{"}, year \Rightarrow 1975], \\
& \quad S76: [type \Rightarrow \text{"}InProc\text{"}, title \Rightarrow \text{"}Ingres\text{"}, auth \Rightarrow \text{"}Sam\text{"}, conf \Rightarrow \text{"}EDBT\text{"}]\} \\
& \cup \{B80: [type \Rightarrow \text{"}Article\text{"}, title \Rightarrow \text{"}Oracle\text{"}, auth \Rightarrow \text{"}Bob\text{"}, year \Rightarrow 1980] \cup_K \\
& \quad B82: [type \Rightarrow \text{"}Article\text{"}, title \Rightarrow \text{"}Oracle\text{"}, auth \Rightarrow \text{"}Bob\text{"}, year \Rightarrow 1980], \\
& \quad A78: [type \Rightarrow \text{"}Article\text{"}, title \Rightarrow \text{"}Datalog\text{"}, auth \Rightarrow \text{"}Ann\text{"}, year \Rightarrow 1978] \cup_K \\
& \quad A78: [type \Rightarrow \text{"}Article\text{"}, title \Rightarrow \text{"}Datalog\text{"}, auth \Rightarrow \text{"}Tom\text{"}, year \Rightarrow 1978], \\
& \quad J88: [type \Rightarrow \text{"}Article\text{"}, title \Rightarrow \text{"}DOOD\text{"}, auth \Rightarrow \text{"}Joe\text{"}, jnl \Rightarrow \text{"}JLP\text{"}] \cup_K \\
& \quad P90: [type \Rightarrow \text{"}Article\text{"}, title \Rightarrow \text{"}DOOD\text{"}, auth \Rightarrow \text{"}Pam\text{"}, jnl \Rightarrow \text{"}JLP\text{"}]\} \\
& = \{S78: [type \Rightarrow \text{"}Article\text{"}, title \Rightarrow \text{"}Ingres\text{"}, auth \Rightarrow \text{"}Sam\text{"}, jnl \Rightarrow \text{"}TODS\text{"}], \\
& \quad S85: [type \Rightarrow \text{"}Article\text{"}, title \Rightarrow \text{"}NF2\text{"}, auth \Rightarrow \text{"}Sam\text{"}, year \Rightarrow 1985], \\
& \quad T79: [type \Rightarrow \text{"}InProc\text{"}, title \Rightarrow \text{"}RDB\text{"}, auth \Rightarrow \text{"}Tom\text{"}, conf \Rightarrow \text{"}PODS\text{"}], \\
& \quad A75: [type \Rightarrow \text{"}InProc\text{"}, title \Rightarrow \text{"}NF2\text{"}, auth \Rightarrow \text{"}Ann\text{"}, year \Rightarrow 1975], \\
& \quad S76: [type \Rightarrow \text{"}InProc\text{"}, title \Rightarrow \text{"}Ingres\text{"}, auth \Rightarrow \text{"}Sam\text{"}, conf \Rightarrow \text{"}EDBT\text{"}], \\
& \quad B80|B82: [type \Rightarrow \text{"}Article\text{"}, title \Rightarrow \text{"}Oracle\text{"}, auth \Rightarrow \text{"}Bob\text{"}, year \Rightarrow 1980], \\
& \quad A78: [type \Rightarrow \text{"}Article\text{"}, title \Rightarrow \text{"}Datalog\text{"}, auth \Rightarrow \text{"}Ann\text{"}|\text{"}Tom\text{"}, yr \Rightarrow 1978], \\
& \quad J88|P90: [type \Rightarrow \text{"}Article\text{"}, title \Rightarrow \text{"}DOOD\text{"}, auth \Rightarrow \text{"}Joe\text{"}|\text{"}Pam\text{"}, jnl \Rightarrow JLP]\}
\end{aligned}
$$

$$
\begin{array}{l}
S_1 \cap_K S_2 \\
= \{B80: [type \Rightarrow \text{"}Article\text{"}, title \Rightarrow \text{"}Oracle\text{"},\ auth \Rightarrow \text{"}Bob\text{"},\ year \Rightarrow 1980\ ] \cap_K \\
\quad B82: [type \Rightarrow \text{"}Article\text{"}, title \Rightarrow \text{"}Oracle\text{"},\ auth \Rightarrow \text{"}Bob\text{"},\ year \Rightarrow 1980\ ], \\
\quad A78: [type \Rightarrow \text{"}Article\text{"}, title \Rightarrow \text{"}Datalog\text{"}, auth \Rightarrow \text{"}Ann\text{"}, year \Rightarrow 1978\ ] \cap_K \\
\quad A78: [type \Rightarrow \text{"}Article\text{"}, title \Rightarrow \text{"}Datalog\text{"}, auth \Rightarrow \text{"}Tom\text{"}, year \Rightarrow 1978\ ], \\
\quad J88: [type \Rightarrow \text{"}Article\text{"}, title \Rightarrow \text{"}DOOD\text{"},\ auth \Rightarrow \text{"}Joe\text{"},\ jnl \Rightarrow \text{"}JLP\text{"}] \cap_K \\
\quad P90: [type \Rightarrow \text{"}Article\text{"}, title \Rightarrow \text{"}DOOD\text{"},\ auth \Rightarrow \text{"}Pam\text{"}, jnl \Rightarrow \text{"}JLP\text{"}]\} \\
= \{\ \bot: [type \Rightarrow \text{"}Article\text{"}, title \Rightarrow \text{"}Oracle\text{"},\ auth \Rightarrow \text{"}Bob\text{"}, year \Rightarrow 1980], \\
\quad A78: [type \Rightarrow \text{"}Article\text{"}, title \Rightarrow \text{"}Datalog\text{"}, year \Rightarrow 1978\ ], \\
\quad \bot: [type \Rightarrow \text{"}Article\text{"}, title \Rightarrow \text{"}DOOD\text{"},\ jnl \Rightarrow \text{"}JLP\text{"}\ ]\}
\end{array}
$$

$$
\begin{array}{l}
S_1 -_K S_2 \\
= \{S78: [type \Rightarrow \text{"}Article\text{"}, title \Rightarrow \text{"}Ingres\text{"},\ auth \Rightarrow \text{"}Sam\text{"}, jnl \Rightarrow \text{"}TODS\text{"}]\} \\
\cup \{B80: [type \Rightarrow \text{"}Article\text{"}, title \Rightarrow \text{"}Oracle\text{"},\ auth \Rightarrow \text{"}Bob\text{"},\ year \Rightarrow 1980\ ] -_K \\
\quad B82: [type \Rightarrow \text{"}Article\text{"}, title \Rightarrow \text{"}Oracle\text{"},\ auth \Rightarrow \text{"}Bob\text{"},\ year \Rightarrow 1980\ ], \\
\quad A78: [type \Rightarrow \text{"}Article\text{"}, title \Rightarrow \text{"}Datalog\text{"}, auth \Rightarrow \text{"}Ann\text{"}, year \Rightarrow 1978\ ] -_K \\
\quad A78: [type \Rightarrow \text{"}Article\text{"}, title \Rightarrow \text{"}Datalog\text{"}, auth \Rightarrow \text{"}Tom\text{"}, year \Rightarrow 1978\ ], \\
\quad J88: [type \Rightarrow \text{"}Article\text{"}, title \Rightarrow \text{"}DOOD\text{"},\ auth \Rightarrow \text{"}Joe\text{"},\ jnl \Rightarrow \text{"}JLP\text{"}\ ] -_K \\
\quad P90: [type \Rightarrow \text{"}Article\text{"}, title \Rightarrow \text{"}DOOD\text{"},\ auth \Rightarrow \text{"}Pam\text{"}, jnl \Rightarrow \text{"}JLP\text{"}\ ]\} \\
= \{S78: [type \Rightarrow \text{"}Article\text{"}, title \Rightarrow \text{"}Ingres\text{"},\ auth \Rightarrow \text{"}Sam\text{"}, jnl \Rightarrow \text{"}TODS\text{"}], \\
\quad B80: [type \Rightarrow \text{"}Article\text{"}, title \Rightarrow \text{"}Oracle\text{"}\ ], \\
\quad \bot\ : [type \Rightarrow \text{"}Article\text{"}, title \Rightarrow \text{"}Datalog\text{"}, auth \Rightarrow \text{"}Ann\text{"}\ ], \\
\quad J88: [type \Rightarrow \text{"}Article\text{"}, title \Rightarrow \text{"}DOOD\text{"},\ auth \Rightarrow \text{"}Joe\text{"},\ ]\}
\end{array}
$$

Note that the two semistructured data with title *"Ingres"* (also *"NF2"*) are not compatible as one has type *"Article"* and the other has type *"InProc"*.

As the above example shows, the union operation combines sets of semistructured data and records inconsistency in the meantime, the intersection operation finds common information in sets of semistructured data and indicates inconsistency in the meantime, while the difference operation finds the information in the first set of semistructured data but in not the second set.

The user can then solve the inconsistency based on the results.

The *union*, *intersection*, and *difference* operations have the following properties.

Proposition 3. Let S_1 and S_2 be sets of semistructured data and K a non-empty set of attributes. Then

(1) $S_1 \trianglelefteq S_1 \cup_K S_2, \quad S_2 \trianglelefteq S_1 \cup_K S_2$
(2) $S_1 \cap_K S_2 \trianglelefteq S_1, \quad S_1 \cap_K S_2 \trianglelefteq S_2$
(3) $S_1 -_K S_2 \trianglelefteq S_1$
(4) $S_1 = (S_1 -_K S_2) \cup_K (S_1 \cap_K S_2)$

Proposition 4. Let S_1, S_2 be sets of semistructured data and K_1 and K_2 non-empty sets of attributes. Then $K_1 \subseteq K_2$ implies the following:

(1) $S_1 \cup_{K_2} S_2 \trianglelefteq S_1 \cup_{K_1} S_2$
(2) $S_1 \cap_{K_1} S_2 \trianglelefteq S_1 \cap_{K_2} S_2$

(3) $S_1 -_{K_1} S_2 \ \trianglelefteq\ S_1 -_{K_2} S_2$

For example, let $K_1 = \{type, title\}$ and $K_2 = \{type, title, author\}$. Then for the two sets of semistructured data in Example 6, $S_1 \cup_{K_2} S_2 \ \trianglelefteq\ S_1 \cup_{K_1} S_2$, $S_1 \cap_{K_1} S_2 \ \trianglelefteq\ S_1 \cap_{K_2} S_2$, and $S_1 -_{K_1} S_2 \ \trianglelefteq\ S_1 -_{K_2} S_2$.

4 Conclusion

The need for manipulating semistructured data naturally arises in real-world applications. In this paper, we present a novel approach for representing and manipulating semistructured data with partial and inconsistent information. Three powerful operations: *union*, *intersection*, and *difference*, are defined and their semantic properties are discussed in detail. This work provides a firm foundation in discussing the semantics of semistructured data in which heterogeneous data may come from various data sources with partial and inconsistent information.

Our work can be extended by adding other important operations to make it complete for query and manipulate semistructured data. One of them is the *expand* operation that can be used to expand the markers to semistructured data for further manipulation. We also intend to investigate how to implement the semistructured data model and use it for various practical semistructured data applications. Finally, we would like to develop rule-based languages for such semistructured data model based on Complex Object Calculus [5], Datalog extensions such as Relationlog [25] and deductive object-oriented database languages such as ROL [24].

References

1. S. Abiteboul. Querying Semistructured Data. In *Proceedings of the International Conference on Data Base Theory*, pages 1–18. Springer-Verlag LNCS 1186, 1997.
2. S. Abiteboul, D. Quass, J. McHugh, J. Widom, and J. L. Wiener. The Lorel Query Language for Semistructured Data. *Intl. Journal of Digital Libraries*, 1(1):68–88, 1997.
3. J. L. Ambite, N. Ashish, G. Barish, G.A. Knoblock, S. Minton, P.J. Modi, I. Muslea, A. Philpot, and S. Tejada. ARIADNE: A system for constructing mediators for internet sources. In *Proceedings of the ACM SIGMOD International Conference on Management of Data*, 1998.
4. G. Arocena and A. Mendelzon. WebOQL: Restructuring Documents, Databases and Webs. In *Proceedings of the International Conference on Data Engineering*, pages 24–33. IEEE Computer Society, 1998.
5. F. Bancilhon and S. Khoshafian. A Calculus for Complex Objects. *J. Computer and System Sciences*, 38(2):326–340, 1989.
6. C. Beeri, G. Elber, T. Milo, Y. Sagiv, O. Shmueli, N. Tishby, Y. Kogan, D. Konopnicki, P. Mogilevski, and N. Slonim. Websuite – A tool suite for harnessing web data. In *Proceedings of the International Workshop on the Web and Databases*, 1998.
7. O. P. Buneman, S. B. Davidson, and A. Watters. A Semantics for Complex Objects and Approximate Answers. *J. Computer and System Sciences*, 43(1):170–218, 1991.

8. P. Buneman, S. Davidson, M. Fernandez, and D. Suciu. Adding Structure to Unstructured Data. In *Proceedings of the International Conference on Data Base Theory*, pages 336–350. Springer-Verlag LNCS 1186, 1997.
9. P. Buneman, S. Davidson, G. Hilebrand, and D. Suciu. A Query Language and Optimization Techniques for Unstructured Data. In *Proceedings of the ACM SIGMOD International Conference on Management of Data*, pages 505–516, 1996.
10. S. S. Chawathe, H. Garcia-Molina, J. Hammer, K. Ireland, Y. Papakonstantinou, J. D. Ullman, and J. Widom. The TSIMMIS Project: Integration of Heterogeneous Information Sources. In *Proceedings of the 10th Meeting of the Information Processing Society of Japan*, pages 7–18, 1994.
11. W. W. Cohen. Integration of Heterogeneous Databases without Common Domains Using Queries Based on Textual Similarity. In *Proceedings of the ACM SIGMOD International Conference on Management of Data*, pages 201–212, 1998.
12. O. Shmueli D. Konopnicki. W3QS: A Query System for the World-Wide Web. In *Proceedings of the International Conference on Very Large Data Bases*, pages 54–65, Zurich,Switzerland, 1995. Morgan Kaufmann Publishers, Inc.
13. L.G. Demichiel. Resolving Database Incompatibility: An Approach to Performing Relational Operations over Mismatched Domains. *IEEE Transactions on Knowledge and Data Engineering*, 1(4):485–493, 1989.
14. D. Florescu, A. Levy, and A. Mendelzon. Database Techniques for the World-Wide Web: A Survey. *SIGMOD Record*, 26(3), 1997.
15. R. Himmeroder, G. Lausen, B. Ludascher, and C. Schlepphorst. On a declarative semantics for web queries. In *Proceedings of the International Conference on Deductive and Object-Oriented Databases*, pages 386–398, Switzerland, 1997. Springer-Verlag LNCS.
16. R. Hull and G. Zhou. A Framework for Supporting Data Integration Using the Materialized and Virtual Approaches. In *Proceedings of the ACM SIGMOD International Conference on Management of Data*, pages 481–492, 1996.
17. T. Imielinski and W. L. Jr. Incomplete Information in Relational Databases. *Journal of ACM*, 31(4):761–791, 1984.
18. W. L. Jr. On Databases with Incomplete Information. *Journal of ACM*, 28(1):41–70, 1981.
19. L. V. S. Lakshmanan, F. Sadri, and I. N. Subramanian. A Declarative Language for Querying and Restructuring the Web. In *Proceedings of the 6th International Workshop on Research Issues in Data Engineering*, 1996.
20. L. Lamport. *Latex User Guide and Reference Manual*. Addison Wesley, 2 edition, 1994.
21. A. Y. Levy, A. Rajaraman, and J. J. Ordille. Querying heterogeneous information sources using source descriptions. In *Proceedings of the International Conference on Very Large Data Bases*, pages 251–262. Morgan Kaufmann Publishers, Inc., 1996.
22. L. Libkin. A Relational Algebra for Complex Objects based on Partial Information. In *Proceedings of the Conference on Mathematical Foundations of Programming Semantics*, pages 26–41, Rostock, Germany, 1991. Springer-Verlag LNCS 495.
23. L. Libkin. Normalizing Incomplete Databases. In *Proceedings of the ACM Symposium on Principles of Database Systems*, pages 219–230, San Jose, California, 1995.
24. M. Liu. ROL: A Deductive Object Base Language. *Information Systems*, 21(5):431 – 457, 1996.
25. M. Liu. Relationlog: A Typed Extension to Datalog with Sets and Tuples. *Journal of Logic Programming*, 36(3):271–299, 1998.

26. A. Mendelzon, G. Mihaila, and T. Milo. Querying the World Wide Web. In *Proceedings of the First International Conference on Parellel and Distributed Information System*, pages 80–91, 1996.
27. A. Motro and I. Rakov. Estimating the Quality of Data in Relational Databases. In *Proceedings of the 1996 Conference on Information Quality*, pages 94–106, 1996.
28. K. Munakata. Integration of Semistructured Data Using Outer Joins. In *Proceedings of the Workshop on Management of Semistructured Data*, 1997.
29. Y. Papakonstantinou, H. Garcia-Molina, and J. Widom. Object Exchange across Heterogeneous Information. In *Proceedings of the International Conference on Data Engineering*, pages 251–260. IEEE Computer Society, 1995.
30. F. S. C. Tseng, A. L. P. Chen, and W. P. Yang. Answering Heterogeneous Databases Queries with Degrees of Uncertainty. *Distributed and Parallel Databases*, 1(3):281–302, 1993.