COMPLETION OF THE INCOMPLETE DATA STRUCTURE

Matti Linna
University of Vaasa
P.O.Box 700
Vaasa
Finland

Merja Wanne
University of Vaasa
P.O.Box 700
Vaasa
Finland

Jari Töyli
University of Vaasa
P.O.Box 700
Vaasa
Finland

Abstract

In structured database systems the schema has two purposes. First it describes the structure or type of the data and second it describes some constraints on the system. However, there is data that is not constrained by a schema or the schema is loose. This type of data is called semistructured data. In this paper we consider such data, and present novel ideas of how the degree of the structure of the data can be evaluated. We also give a new definition of semistructured data based on the theory of the Adjacency Relation Systems (ARS) and present some motivating examples.

Key Words

Knowledge Representation, Semistructured Data, Data Structure, Degree of Semistructured Data

1. Introduction

Semistructured data [1,2] has been a target of intensive investigations since the middle of the 90's. During that time the research has focused on developing of data models and query languages for the semistructured data.

In addition to the generic data formats there are two main models proposed for semistructured data. The first one, which is based on object model, is called Lore [3] and the second one, where node labels are absent and the data is carried entirely on the edge labels, is called UnQL [4].

The basic structure of such data is a tree like or a graph like structure that combines data and structure into a simple data format. This schema-less structure arises some problems that must be considered in the data models and query languages developed for semistructured data.

In this paper we consider the same problem area, and give a proposal how the structure of the data can be increased by using the methods presented in [5,6], and by giving an exact definition of the semistructured data based on the degree of the structure of the data.

The key notion in this paper is the concept of adjacency relation system first introduced in [5]. The authors have shown in [7] how relational data can be modelled by the adjacency model and in [8] how semistructured data can be presented using this new model. The new results of this paper are:

- An exact definition of semistructured data.
- A definition of the degree of semistructured data.
- An algorithm for deciding the degree of semistructured data.

The results presented in this paper are novel, and the definitions, concepts and methods give a new angle of view for the research concerning semistructured data models and query languages.

The rest of the paper will be organized as follows. In Section 2 we define the basic concepts necessary for understanding the rest of the paper. Section 3 contains the exact definition of semistructured data and the concept of the degree of the structure of the semistructured data. We give also some clarifying examples concerning the degree of semistructured data. In Section 4 we present an algorithm for determining the degree of semistructured data and formulate a common search problem. In Section 5 we give some examples of how the results can be used to further minimize the "schema" presented with the help of a DataGuide [9]. Finally, Section 6 contains some conclusions.

2. Preliminaries

In this section we present the basic concepts needed in the paper. We start with semistructured data and continue with the Adjacency Relation System (ARS).

2.1 Semistructured Data

The basic structure for presenting of semistructured data is a graph. There are two main variations of the graph model with only minor difference. The first one is called Object Exchange Model (OEM), and it is developed in

TSIMMIS project [10]. OEM is flexible enough to encompass all types of information, yet it is simple enough to facilitate integration. It also includes semantic information of objects [11].

In this model the semistructured database is presented as a rooted, directed labeled graph, where the nodes are the objects, the edges are labeled with attributes, and in which some leaf nodes have an associated atomic value. The graph has a root, i.e. a distinguished object from which all the other objects are accessible. In Figure 1 we can see an OEM model of a movie database.

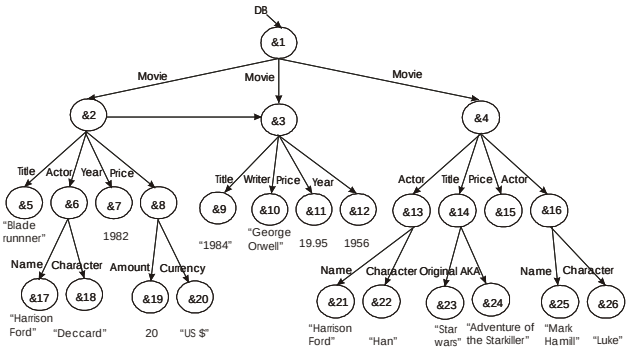

Figure 1. An OEM database [12]

The base types of the graph are numbers (19.95, 20 etc.), strings ("Harrison Ford", "Han", etc.) and labels (Name, Character, Title, etc.). There are also other types, which define textual encodings, e.g. date, time, gif etc. (not present in Figure 1) [13].

The other variant of the model is called UnQL.

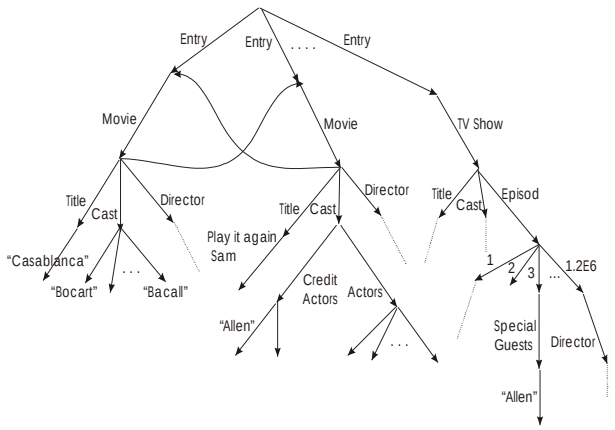

Figure 2. An example movie Database [14]

In this model labels carry all the basic information (Figure 2). Types in this model are the same base types as in OEM (i.e. String, Integer, Real etc.) and also a new type called Symbol. This type corresponds to attribute names in relations. Formally, a rooted, labeled graph is $G = \{V, E, s, t, r, \lambda, \rho\}$, where (V, E, s, t) is a multi-graph ($s : E \to V$ and $t : V \to E$ denote the source and target of each edge), $r \in V$ is the root, and $\lambda : E \to Label \cup \{\varepsilon\}$ is the labeling function. With ρ we can tie the leaf nodes back into the tree, and ε denotes "empty" symbol [14].

The recent research of semistructured data has focused on XML. The reason is the similarities between the structures of XML data and semistructured data [13].

2.2. Adjacency Relation System

In this section we will define our model, called Adjacency Model (AM), for representing semistructured data. Here we give only the basic definitions. In order to get an exact insight of what an adjacency model is, we refer to the definitions introduced in [5] and [6].

An *adjacency relation system* (ARS) is a pair (A, R), where $A = \{A_1, A_2, \cdots, A_n\}$, $n \geq 1$, is a set containing pairwise disjoint finite nonempty sets and $R = \{R_{ij} \mid i, j \in \{1,2,\cdots,n\}\}$ is a set of relations, where each R_{ij} is a relation on $A_i \times A_j$.

If $(x, y_1), (x, y_2), \ldots, (x, y_m) \in R_{ij}$ are all the pairs of relation R_{ij} having x as the first component, then each element y_k $(k = 1,2,\ldots,m)$ is said to be adjacent to the element x. Furthermore, denote by $Ad_j(x)$ the set $\{y_1, y_2, \ldots, y_m\}$.

An adjacency relation system (A, R) is said to be *symmetric* if for each pair $x \in A_i$, $y \in A_j$, it holds that $x \in Ad_i(y)$ if and only if $y \in Ad_j(x)$. In many applications the adjacencies are symmetric.

We assume that the elements of each set A_i, $i = 1,2,\cdots,n$, represent entities of a certain type T_i. The adjacency between elements can also be defined with the help of the so-called adjacency defining sets of types in the following way. Associate with each index pair $i, j \in \{1,2,\cdots,n\}$ a set $K \subseteq \{1,2,\cdots,n\} - \{i, j\}$ of indices and a set of entity types

$$\tilde{T}_{ij} = \{T_k \mid k \in K\}.$$

The set $\tilde{T}_{ij}$ thus gives us the entity types which determine the adjacency between the elements of A_i and A_j.

Elements $x \in A_i$, $y \in A_j$, where $i, j \in \{1,2,\ldots,n\}$ and $x \neq y$, are said to be *adjacent with respect to a set of entity types*

$$\tilde{T}_{ij} = \{T_k \mid k \in K\} \neq \phi$$

if for each $k \in K$ there is an element $z \in A_k$ such that $x \in Ad_i(z)$ and $y \in Ad_j(z)$. The set $\tilde{T}_{ij}$ is called the *adjacency defining set* between the elements of A_i and A_j.

An ARST (A, R, τ) is said to be *unique* if for each pair $i, j \in \{1,2,\ldots,n\}$ of integers such that the corresponding adjacency defining set $\tilde{T}_{ij}$ is nonempty and for all elements $x \in A_i$, $y \in A_j$, x and y are adjacent if and only if they are adjacent with respect to $\tilde{T}_{ij}$.

Consider (Figure 3) a set of relations $R = \{R_{11}, R_{12}, R_{13}, R_{21}, R_{22}, R_{23}, R_{13}, R_{23}, R_{33}\}$, where

$$R_{11} = \{(x_1, x_2), (x_2, x_1)\},$$
$$R_{12} = \{(x_1, y_2), (y_2, x_1), (x_2, y_2), (y_2, x_2)\},$$
$$R_{13} = \{(x_2, z_1), (z_1, x_2), (x_2, z_2), (z_2, x_2)\},$$
$$R_{33} = \{(z_1, z_2), (z_2, z_1)\},$$
$$R_{21} = R_{22} = R_{23} = R_{31} = R_{32} = \phi.$$

and the adjacency defining sets

$$\tilde{T}_{11} = \{T_2\},$$
$$\tilde{T}_{33} = \{T_1\},$$
$$T_{22} = T_{12} = T_{21} = T_{13} = T_{31} = T_{23} = T_{32} = \phi.$$

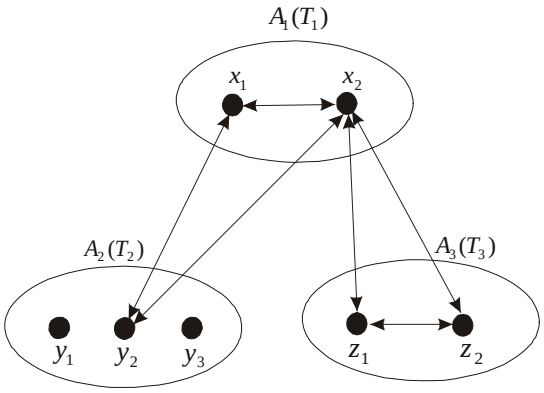

Figure 3: An adjacency defining set.

We can see that elements x_1 and x_2, $x_1, x_2 \in A_1$, are adjacent with each other as well as z_1 and z_2, $z_1, z_2 \in A_3$. The adjacency is defined via the relation R_{11} for x_1 and x_2, and via the relation R_{33} for z_1 and z_2, and on the other hand, also with respect to the adjacency defining sets $\tilde{T}_{11}$ and $\tilde{T}_{33}$. If there are no other non-empty adjacency defining sets, the ARST considered is unique.

3. Definition of Semistructured data

In this chapter we give an accurate definition for semistructured data.

We can use the notation $T_i \to T_j$ for a relation type to indicate that the relations R_{ij} are defined on $A_i \times A_j$. Furthermore, we can consider sets of relation types

$$\{T_i \to T_j \mid (i, j) \in S\},$$

where $S \subseteq \{1,2,\ldots,n\} \times \{1,2,\ldots,n\}$. For an ARST (A, R, τ), denote by $R \mid S$ the restriction of R on S, i.e.

$$R \mid S = \{R_{ij} \in R \mid (i, j) \in S\}.$$

Relation combinations have very remarkable importance for the optimization of the queries on semistructured data because they can be used to restrict the search space, and hereby limit the search time used by the query. Here the concept of valid relation combinations plays a key role.

A relation $T_r \to T_s$ is *determined uniquely* by the relation combination $\{T_i \to T_j \mid (i, j) \in S\}$, if for any unique ARST (A, R, τ) there is no other unique ARST (A, R', τ) such that $R \mid S = R' \mid S$ but $R_{rs} \neq R'_{rs}$. Each relation determined uniquely by a given relation combination is said to be *derivable* from the given relation combination. If a relation combination determines uniquely all other relations then it is said to be *valid*.

Example 1. Consider the ARST (A, R, τ), where

$$R_{11} = \{(x_1, x_2), (x_2, x_1)\},$$
$$R_{22} = \phi,$$
$$R_{33} = \{(z_1, z_2), (z_2, z_1)\},$$
$$R_{12} = \{(x_1, y_1), (x_1, y_2), (x_2, y_2), (x_2, y_3)\},$$
$$R_{21} = \{(y_1, x_1), (y_2, x_1), (y_2, x_2), (y_3, x_2)\},$$
$$R_{13} = \{(x_2, z_1), (x_2, z_2)\},$$
$$R_{31} = \{(z_1, x_2), (z_2, x_2)\},$$
$$R_{23} = \{(y_1, z_2), (y_2, z_1), (y_3, z_1)\},$$
$$R_{32} = \{(z_1, y_2), (z_1, y_3), (z_2, y_1)\},$$
$$\tilde{T}_{11} = \{T_2\}, \tilde{T}_{33} = \{T_1\},$$
$$\tilde{T}_{22} = \tilde{T}_{12} = \tilde{T}_{21} = \tilde{T}_{13} = \tilde{T}_{31} = \tilde{T}_{23} = \tilde{T}_{32} = \phi.$$

Let $S = \{(1,2), (2,3)\}$. Then

$$R \mid S = \{R_{ij} \in R \mid (i, j) \in S\} = \{R_{12}, R_{23}\}.$$

Let (A, R', τ) be another ARST, where R' contains the same relations as R except the relation on $A_2 \times A_2$ which is defined by

$$R'_{22} = \{(y_1, y_2), (y_2, y_1)\}.$$

As in Example 1 it can be deduced that the ARST (A, R', τ) is unique. Since $R \mid S = R' \mid S$, the relation combination $\{T_1 \to T_2, T_2 \to T_3\}$ is not valid.

Now, e.g. the relation $T_1 \to T_3$ is not uniquely determined. Namely, if (A, R', τ) is another ARST, where R' contains the same relations as R except the relations on $A_1 \times A_3$ and on $A_3 \times A_1$, which are defined by

$$R'_{13} = \{(x_1, z_1), (x_1, z_2)\} \text{ and } R'_{31} = \{(z_1, x_1), (z_2, x_1)\}.$$

Now we define the notion of the degree of the data structure.

Let C be a subset of all possible relations of the form $T \to T'$, where T and T' are types. Denote by $|C|$ the number of relations in C. Let C' be a relation combination such that $C \cup C'$ is valid and $|C'| = \min\{|C_i| \mid C \cup C_i \text{ is valid}\}$. Denote $\min(C) = |C'|$. In other words, $\min(C)$ is the minimum number of relations, which is needed to complete C to a valid relation combination.

Denote by $D(C)$ the set of all relations, which can be derived uniquely from C. Obviously, $C \subseteq D(C)$.

The *degree of a relation combination* C is defined as the ratio

$$Deg(C) = \frac{|D(C)|}{\textit{the number of all relations}}.$$

According to the definition $0 \le Deg(C) \le 1$ for each relation combination C.

Let C be the relation combination describing the given data structure. The data structure is said to be

(i) *structured*, if $Deg(C) = 1$;
(ii) *unstructured*, if $Deg(C) = 0$;
(iii) *semistructured*, if $0 < Deg(C) < 1$.

Note. Obviously, $Deg(C) = 1$ if and only if C is valid.

Example 2. Consider a special case of an ARST associated with planar graphs. We have the entities V (vertices), E (edges) and F (faces). Denote e.g. $T_1 = V$, $T_2 = E$ and $T_3 = F$. By the usual interpretation, two vertices are adjacent if they are adjacent with respect to E. Thus we may denote $\tilde{T}_{11} = \tilde{T}_{VV} = \{E\}$. Similarly, we may define $\tilde{T}_{22} = \tilde{T}_{EE} = \{V\}$ and $\tilde{T}_{33} = \tilde{T}_{FF} = \{E\}$. Moreover, $\tilde{T}_{12} = \tilde{T}_{VE} = \phi$, $\tilde{T}_{13} = \tilde{T}_{VF} = \{E\}$ and $\tilde{T}_{23} = \tilde{T}_{EF} = \phi$.

In this case we have altogether nine different types of relations (Figure 4).

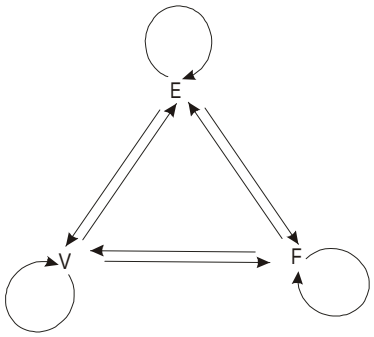

Figure 4. Three entities and nine relations

Example 3. It can be shown [5] that, if we know the relations $F \to V$ and $V \to E$, i.e. $C = \{F \to V, V \to E\}$, then we can always derive the relations $V \to F$, $E \to V$ and $V \to V$. This means that

$$D(C) = \{F \to V, V \to E, V \to F, E \to V, V \to V\}.$$

So the degree of C is

$$Deg(C) = 5/9 = 0.56.$$

If e.g. $C = \{V \to E, E \to F\}$, it can be shown that $D(C)$ contains all the nine relations and so

$$Deg(C) = 9/9 = 1$$

and the relation combination C is valid.

If on the other hand $C = \{V \to E, F \to V, E \to E\}$, it can be shown that

$$D(C) = \{V \to E, F \to V, E \to E, E \to V, V \to F, V \to V\}.$$

So the degree of C in this case is

$$Deg(C) = 6/9 = 0.67.$$

4. Determining of the Degree

We first give an algorithm for determining the degree of a given data structure C, given as a set of relations. We end the section by proposing the definition for a general query.

Let entity types $T_1,\dots,T_n$, a set $S \subseteq \{1,2,\dots,n\}\times\{1,2,\dots,n\}$ and adjacency defining sets τ be given. Consider the relation combination $C = \{T_i \to T_j \mid (i,j)\in S\}$. By the symmetry, we may assume that for each pair $(i,j)\in S$ also $(j,i)\in S$.

Degree Testing Algorithm.

Step 1. Search for each pair $(i,j)\notin S$, such that there exists an integer $k \neq i,j$ for which $(i,k)\in S$ and $(j,k)\in S$. If such pair and integer are found, goto Step 2. Otherwise goto Step 3.

Step 2. Find all integers $k_1,\dots k_r$, $r \geq 1$ such that $(i,k_s),(j,k_s)\in S$ for $1\leq s\leq r$. If for at least one s, $1\leq s\leq r$, we have $T_{k_s}\in \tilde{T}_{ij}$ and $\tilde{T}_{ij}\subseteq\{T_{k_1},T_{k_2},\dots,T_{k_r}\}$, then include the relations $T_i \to T_j$ and $T_j \to T_i$ in the relation combination, i.e. add the pairs (i,j) and (j,i) to the set S and goto Step 2. If there is no such integer s, then goto Step 3.

Step 3. Let C_1 be the set of all new relations derived by Step 2. The degree of the given data structure C is

$$Deg(C) = \frac{|C|+|C_1|}{n^2}$$

and the set of all obtained relations is $C\cup C'$.

The proof of the correctness of the algorithm is omitted here.

The Degree Testing Algorithm enables the definition of a general query based on the given data structure, i.e. on the given relation combination C.

General query: Given ARST (A,R,τ), a relation combination C and an element x of type X, find the set Y_x of all elements of type Y such that

(i) the relation $X \to Y \in D(C)$ and
(ii) $(x,y)\in R_{XY}$ for all $y\in Y_x$.

Note that in (i) the set $D(C)$ of relations can be determined by the preceding algorithm.

It can be shown that this general definition of a query includes e.g. all the special queries introduced in [15], namely the queries such as direct query, inverse query, self queries, direct transitive query, inverse transitive query, and indirect transitive queries.

5. Examples

In Lore [3] the structure of the database is presented by a Data Guide, which is a structural summary of a semistructured database. It contains all the paths of a data graph, and there can be more than one data guide of which one is minimal. A minimal data guide means that it is the smallest data guide in terms of total number of nodes [16].

A data guide is concise and accurate, i.e. it contains all the paths of the data graph, and every path in the data guide occurs only once. However, it is not always obvious that we need all those relations or paths that are represented in a data guide. If we consider a data guide in the light of the theory of the Adjacency Relation Systems, we can reduce the number of edges between some of the nodes (or types as they are called in the ARS). It is fully possible, because the adjacency between the nodes can be represented with the help of the so-called adjacency defining types.

For example in Figure 5, the data guide can be reduced by eliminating the edges between the types *Boss* and *Regular*, because the adjacency is defined with respect to the type *Employee*.

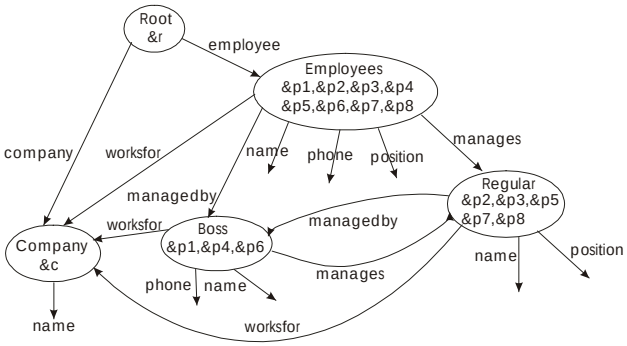

Figure 5. A data guide of an OEM database [13].

Additionally, we can eliminate the edges between the types *Employee* and *Company*, and between the types *Company* and *Regular*. In the first case the reason is that the adjacency is defined with respect to the type *Root* and in the second case the adjacency is defined transitively with respect to the path *Regular-Employee-Boss-Company*.

We now consider a schema graph (Figure 6) of a database, which actually is also a data guide. This schema graph can be further reduced by eliminating an edge called *worksfor*, because of the same reason as before, i.e. the adjacency between the type *Emp* and *Comp* can be defined with respect to the type *Root*.

As we can see from these two examples the data guides of Figures 5 and 6 are not minimal from the point of view of the theory of the adjacency relation systems. This fact gives new emphasis for the definition of semistructured data and its degree presented in Section 3.

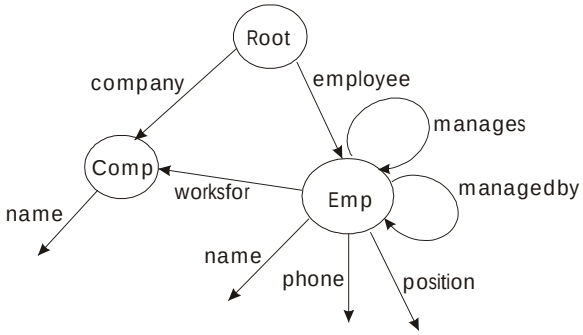

Figure 6. A schema graph of an OEM database [13].

6. Conclusions

In this paper a new definition of semistructured data based on the theory of the Adjacency Relation Systems is introduced. Especially, in this model the degree of semistructured data is defined accurately. It is also shown that the degree is a decidable property. Based on this result the paper contains also a formulation for a general query. The next step will be the developing of different kinds of query algorithms based on the model as well as applying them to frequently occurring applications.

References

[1] S. Abiteboul. Querying Semi-Structured Data. *Proceedings of International Conference on Database Theory*, Delphi, Greece, 1997, 1-18.

[2] P. Buneman. Semistructured Data. *Proceedings of the sixteenth ACM SIGACT-SIGMOD-SIGART Sympiosium on Principles of Database System*, Tucson, Arizona, 1997, 117-121.

[3] J. McHugh, J., S. Abiteboul, R. Goldman, D. Quass and J. Widom. Lore: A Database Management System for Semistructured Data. *SIGMOD Record*, *26* (3), 1997, 54-66.

[4] P. Buneman, M. Fernandez and D. Suciu. UnQL : A Query Language and Algebra for Semistructured Data Based on Structural Recursion. *Very Large Data Bases, VLDB Journal, vol. 9*, no. 1, 2000, 76-110.

[5] M. Wanne. Adjacency Relation Systems. *Acta Wasaensia No. 60. Computer Science 1*. Universitas Wasaensis, Vaasa, 1998.

[6] M. Wanne, and M. Linna. A General Model for Adjacency. *Fundamenta Informaticae, 38*, Numbers 1-2, 1999, 39-50.

[7] J. Töyli, M. Linna, and M. Wanne. Modeling Relational Data by the Adjacency Model. *Proceedings of the Fourth International Conference on Enterprise Information Systems*, Ciudad Real, Spain, 2002, 296-301.

[8] J. Töyli, M. Linna and M. Wanne. Modeling Semistructured Data by the Adjacency Model. *Proceedings of the Fifth Joint Conference on Knowledge-Based Software Engineering*, Maribor, Slovenia, 2002, 282-290.

[9] R. Goldman and J. Widom. DataGuides: Enabling Query Formulation and Optimization in Semistructured Databases. *Proceedings of the Twenty-Third International Conference on Very Large Data Bases*, Athens, Greece, 1997, 436-445.

[10] S. Chawathe, H. Garcia-Molina, J. Hammer, K. Ireland, Y. Papakonstantinou, J. Ullman, and J. Widom. The TSIMMIS Project: Integration of Heterogeneous Information Sources. *Proceedings of the 100th Anniversary Meeting of the Information Processing Society of Japan*, Tokyo, Japan, 1994, 7-18.

[11] Y. Papakonstantinou. *Query Processing in Heterogeneous Information Sources.* Thesis. Stanford University, 1997.

[12] J. McHugh, J. Widom, S. Abiteboul, Q. Luo, and A. Rajaraman. *Indexing Semistructured Data*. Technical Report, the Stanford Database Group, 1998.

[13] S. Abiteboul, P. Buneman & D. Suciu. D*ata on the Web from Relations to Semistructured Data and XML* (San Francisco, CA, Morgan Kaufman Publishers, 2000).

[14] P. Buneman, S. Davidson, G. Hillebrand & D. Suciu. A Query Language and Optimization Techniques for Unstructured Data. *Proceedings of ACM-SIGMOD International Conference on Management of Data*, Montreal, Canada, 1996, 505-516.

[15] X. Ni, S. Bloor. Performance Evaluation of Boundary Data Structures. *IEEE Computer Graphics and Applications*, *14*:6, 1994, 66-77.

[16] R. Goldman. Integrated Query and Search of Databases, XML, and the Web. Thesis. Stanford University, Department of Computer Science, 2000.