# FICSR: $F$eedback-based $In$C$on$S$istency$ $R$esolution and Query Processing on Misaligned Data Sources*

Yan Qi
Arizona State University
Tempe, AZ 85287, USA
yan.qi@asu.edu

K. Selçuk Candan
Arizona State University
Tempe, AZ 85287, USA
candan@asu.edu

Maria Luisa Sapino *
Universita' di Torino
10149 Torino, Italy
mlsapino@di.unito.it

## ABSTRACT

A critical reality in data integration is that knowledge from different sources may often be conflicting with each other. Conflict resolution can be costly and, if done without proper context, can be ineffective. In this paper, we propose a novel query-driven and feedback-based approach (*FICSR*[1]) to conflict resolution when integrating data sources. In particular, instead of relying on traditional *model* based definition of consistency, we introduce a ranked interpretation. This not only enables *FICSR* to deal with the complexity of the conflict resolution process, but also helps achieve a more direct match between the users' (subjective) interpretation of the data and the system's (objective) treatment of the available alternatives. Consequently, the ranked interpretation leads to new opportunities for bi-directional ($data \overset{informs}{\longleftrightarrow} user$) feedback cycle for conflict resolution: given a query, (a) a preliminary ranking of candidate results on data can inform the user regarding constraints critical to the query, while (b) user feedback regarding the ranks can be exploited to inform the system about user's relevant domain knowledge. To enable this feedback process, we develop data structures and algorithms for efficient off-line conflict/agreement analysis of the integrated data as well as for on-line query processing, candidate result enumeration, and validity analysis. The results are brought together and evaluated in the *FICSR* system.

**Categories and Subject Descriptors:** H.2.5[Heterogeneous Databases]; H.3.3[Information Search and Retrieval]:Relevance feedback

**General Terms:** Algorithms, experimentation

**Keywords:** Reasoning with misaligned data, conflicts, taxonomy, relevance feedback, query processing

## 1. INTRODUCTION

Integration of data from different sources starts with a *matching/alignment* phase. Matching, which takes two data or schemas as input and produces a mapping (or alignment) between elements, has been investigated in scientific, business, and web data integration [30] contexts. Work on automated matching systems include [9, 11, 21, 23–26]. C-OWL [6] leverages predefined mappings between local and global ontologies to enable contextualized use. [18] proposes a language which allows specification of alternative semantics for mapping tables and shows that a constraint-based treatment of mappings can lead to efficient mechanisms for inferring new mappings. [2] proposes to use DTDs and source-to-target dependencies to eliminate inconsistent data translation from one schema to the other.

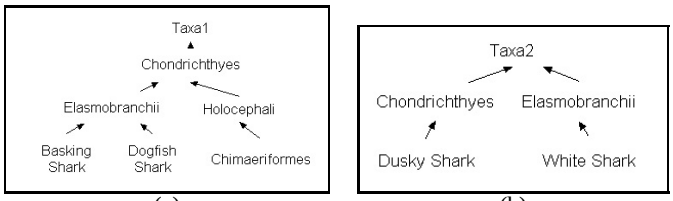



**Figure 1: Two conflicting shark taxonomies**

Despite such advances in mapping technologies, alignments across the data sources are rarely perfect. In this paper, we *focus on the problem of dealing with imperfectly aligned data and present a* F*eedback-based* In*C*on*S*i*stency* R*esolution and query processing (*FICSR*) system which assists the users in query answering when the integrated data contain conflicts*.

### 1.1 Dealing with Misalignments and Conflicts

In many cases, conflict resolution is an ill-defined problem: there may be multiple ways to resolve conflicts and the appropriate conflict resolution strategy may be user- and query context-dependent. Consider a scientist who is trying to work under multiple, conflicting assumptions or hypotheses. Figure 1 shows two alternative shark taxonomies that are available to this scientist, each representing a different view of what the correct categorization of *sharks* should be. This scientist may want to keep both views until she has more understanding on their implications under a particular usage context. In such a case, overly-eager conflict resolution may be detrimental to the effective use of the available knowledge. This is true especially in information mashup scenarios [17], where the ease and speed of integration is as (if not more) important as the completeness and cleanliness.

In this paper, instead of trying to achieve fully-consistent integration in advance of query processing, we rely on query instances to provide contexts in which conflicts should be resolved. Like us, Piazza [15] and HepToX [5] recognize that it is unrealistic to expect an independent data source entering information exchange

*Supported by NSF Grant "*Archaeological Data Integration for the Study of Long-Term Human and Social Dynamics (0624341)*"
*This work was done while the author was visiting ASU
[1]Pronounced as "fixer".


*SIGMOD'07*, June 11–14, 2007, Beijing, China.

to agree to a global mediated schema or to perform heavyweight operations to map its schema to every other schema in the group. Piazza presents a mediation language for mapping both the domain and document structures and focuses on *certain answers* that hold for every consistent instance. HepToX, on the other hand, focuses on automated mapping rule generation, without explicitly considering conflicts. TRIO [3] represents alternatives probabilistically and relies on available lineage information for query processing with alternatives. QUEST presents an assertion-based data model to captures both value-based and structure-based conflicts in data [28,29]. In this paper, we propose a ranked interpretation of constraints, conflicts, and statements on data to enable feedback-based conflict resolution within the context of a query, to support QUEST.

## 1.2 Interpretations of Conflicting Data

Traditionally, a consistent interpretation of the data with conflicts is defined as a maximal, self-consistent subset of the data [4, 22]. This set is referred to as a *model*:

DEFINITION 1.1 (MODEL-BASED INTERPRETATION). *A* model *(or* model-based interpretation*) of a given knowledge base $D$ is a subset $D'$ of the knowledge base ($D' \subseteq D$) such that there exists no other consistent set $D''$, where $D' \subset D'' \subseteq D$.*

Since model-based repair requires selection of a subset of the data, restoration of consistency through a model-based interpretation may lead to loss of information. Furthermore, in many cases, the user may not have enough information (domain knowledge) to select an appropriate model among all the alternatives implied by $D$. Thus, instead of characterizing the user's interpretation as a maximally consistent portion of the data that she commits as being certain, we argue that a more flexible definition of interpretation, *which captures the likelihood that a given asserted statement about the data can be considered as holding*, may be more suitable.

DEFINITION 1.2 (RANKED INTERPRETATION). *Let $D$ be the data and $\mathcal{S}$ be a set of statements (i.e., propositions) on the data. Then, a total ranking of statements in $\mathcal{S}$ is a ranked interpretation of the data $D$.*

The model-based interpretation of the data is a special case of the ranked interpretation, where the rank of all certainly true statements is better than the rank of all certainly false ones.

## 1.3 Objective vs. Subjective Interpretations

In general, data alignment is a subjective process whereby mappings capture the domain expert's interpretation of the data sources and the application requirements. A *subjective* ranked interpretation, $\preceq_{D,U}$, captures the user, $U$'s domain knowledge or preferences, while an *objective* interpretation, $\leq_D$, measures the degree of agreement of the data sources on a given statement.

Query processing over data with conflicts requires any gap between objective (what is represented in the database) and subjective (what the user thinks to hold) interpretations to be bridged. We refer to this as the objective-subjective correspondence:

DESIDERATUM 1 (OBJECTIVE-SUBJECTIVE CORRESP.). *It is preferred that, for all $S_1, S_2 \in \mathcal{S}$, it holds that*

$$(S_1 \preceq_{D,U} S_2) \iff (S_1 \leq_D S_2)\,.$$

This forms the basis of the feedback-based conflict resolution.

## 1.4 Feedback in Information Retrieval

Information retrieval (IR [35]) systems face similar objective-subjective gaps: given an information retrieval request,

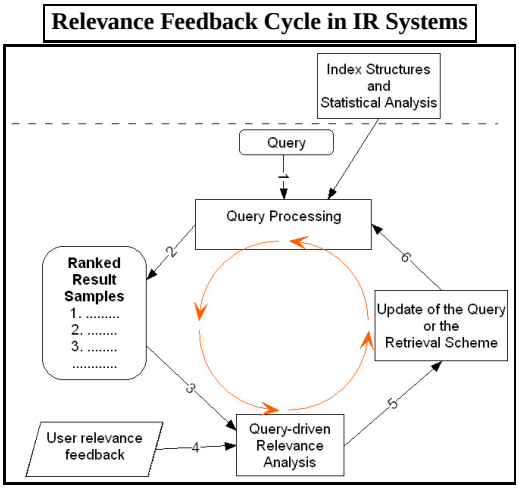



**Figure 2: Overview of the relevance feedback process commonly used in information retrieval when the available data has alternative (user- and query-dependent) interpretations**

- which features of the data are relevant (and how much so) for the user's query may not be known in advance, and
- the number of candidate matches in the database can be large.

In the IR context, these challenges are dealt effectively through relevance feedback. A relevance feedback cycle enables the information retrieval system to learn the user's interests and focus onto a suitable feature set through a query-driven, transparent, and iterative process (Figure 2): (1) Given a query, using the available index structures, the IR system (2) identifies an initial set of candidate results. Since the number of candidates can be large, the system presents a small number of samples to the user. (3) This initial sample and (4) user's relevance/irrelevance inputs are used for (5) learning user's interests (in terms of relevant features) and this information is used for (6) updating the query or the retrieval/ranking scheme. Steps 2-5 are then repeated until the user is satisfied with the ranked result samples returned by the system.

*FICSR* exploits a similar feedback-based approach in the context of query processing in the presence of conflicts and alternative interpretations. The system relies on the *objective-to-subjective* correspondence ($subj \leftarrow obj$) to inform the user about the more likely (i.e., highest source agreement) interpretations for the given query. The *subjective-to-objective* correspondence ($subj \rightarrow obj$), then, informs the system about the user's own interpretations. Since the feedback process can be computationally costly and since the user may have neither the need nor the sufficient domain knowledge to interpret the entire data, instead of considering all possible statements, as in IR systems, it is preferable to focus on only those statements relevant within the context specified by a user query.

## 1.5 Proposed Approach

In this paper, we develop data structures and algorithms to enable feedback-based conflict resolution during query processing on imperfectly aligned data. The overview of the *FICSR* system is presented in Figure 3. The first step is an initial alignment between the input data, obtained through semi-automated techniques, such as [9,11,21,23–26]. The result of the alignment are mapping rules, such as those described in [2, 18]. These rules along with the integrated data are represented in the form of a set of *constraints*.

**Relationship vs. Integrity Constraints:** In this paper, we classify the data constraints into two major classes: (a) *relationship constraints* describe how the individual data objects/entities relate

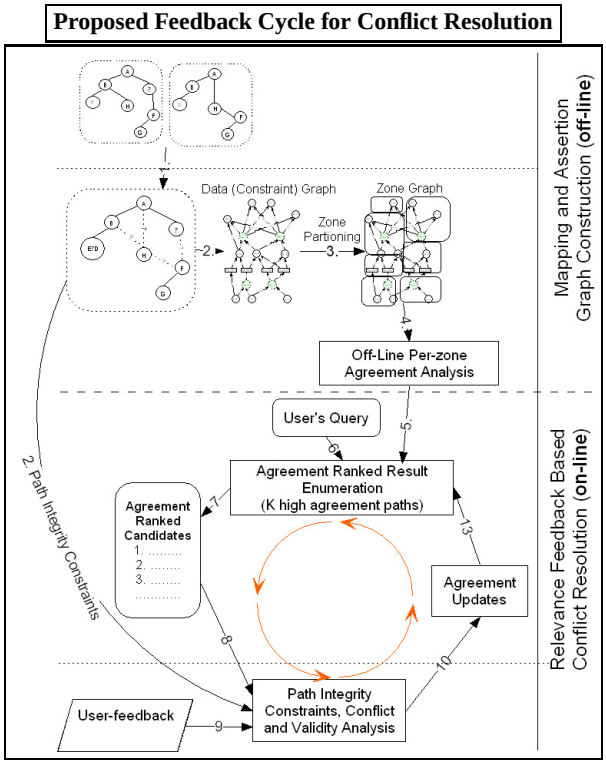



**Figure 3: Query-driven feedback-based conflict resolution**

to each other, while (b) *integrity constraints* describe the general rules these data entities have to obey.

**Source Agreement:** Unless the constraints are conflict free, there will be alternative interpretations of data. Consequently, *FICSR* measures the objective *agreement* of the data sources on a given data fragment. Informing the user regarding the objective ranking involves identifying results with high agreements.

**Query Processing and Feedback-based Conflict Resolution:** The agreement-based ranking task can be computationally expensive if the system would need to enumerate all alternative models (in Section 4.1, we show that the problem is NP-complete even in highly specialized cases). Therefore, a particular challenge in query processing in the presence of conflicts is to postpone the computation of complete solution models until absolutely necessary. In order to deal with the cost of the agreement computation in the presence of conflicts, we divide the task into three stages:

*(Stage 1) Off-line Analysis:* We represent the relationship constraints in the form of a graph which enables off-line *conflict/agreement* analysis. To efficiently compute the agreement values, we partition the graph into small-sized constraint *zones*, each consisting of a mutually-dependent set of relationship constraints. The agreement values are computed for each zone separately.

*(Stage 2) Candidate Enumeration and Ranking:* Given a query, the system identifies and ranks an initial subset of matches on the graph, combining the *zonal* agreement values. Once presented with a ranked set of candidate results, the user can provide her feedback.

*(Stage 3) Path Integrity Constraints and Feedback:* The remaining complex integrity constraints (such as "no-cycles are allowed in data") are used for verifying the *validity* of the ranked interpretation obtained through zonal agreement values. In particular, through the analysis of path integrity constraints within the context provided by the candidate result sets as well as user feedback, candidate results and constraints are assigned *validity values*. Once these validity values are propagated back to the objective agreement values (completing the feedback cycle), the user can be provided with a new subset of *ranked* results.

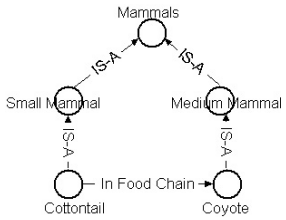



**Figure 4: A basic data graph segment**

## 1.6 Contributions of the Paper

The proposed system brings together various innovative techniques to deal with the computational complexity and the ill-defined nature of the conflict resolution problem:

First of all, we propose a novel, feedback-driven approach to query processing in the presence of conflicts. The feedback process relies on a ranked interpretation of the data. The objective-subjective correspondence of the ranked interpretations enables the user to explore the available data and be informed regarding constraints critical to a given query before providing feedback.

We provide data structures and algorithms for efficient off-line analysis of the data, on-line query processing, candidate result enumeration, and validity analysis. We represent data in the form of relationship and integrity constraints (Sections 2 through 3.2):

- the relationship constraints lend themselves to efficient partitioning into *independent* constraint sets (called zones, Section 3.2). The small sizes of the zones enable efficient off-line agreement analysis (Section 4) and their independent nature enables efficient on-line composition (Section 5).
- the top-$k$ nature of the on-line candidate result enumeration process lets the user focus on high-agreement parts of the data, quickly (also in Section 5).
- the cost of the *conflict* analysis is kept low through the small sizes of the candidate sets that need to be validated as well as through the use of the query context which sets the scope of the conflict analysis (Section 6).
- a constraint programming approach based on the ranked interpretation of data and the fuzzy semantics attached to the validities and conflicts enable *integrity* analysis and feedback without relying on clique-based model enumeration (Sections 7 and 8).

In Section 9, we evaluate the effectiveness and efficiency of the proposed techniques. These techniques are being deployed in a feedback-driven QUEST system to support scientific reasoning with incomplete and conflicting information [28, 29].

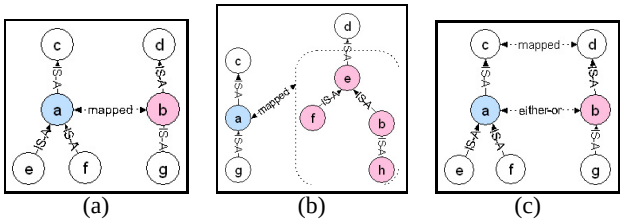
# 2. DATA REPRESENTATION

Since our goal is to maximize the applicability of *FICSR* to diverse application domains, we keep our assumptions from the data low and simply represent the data, $D(G, IC)$, in the form of a graph ($G$) of entities and their relationships and the associated integrity constraints ($IC$). In other words, each $D$ is an instance of a given ER (entity-relationship) schema and associated constraints.

## 2.1 Data Relationship Graph

A basic data (relationship) graph describes the objects/entities in a data source and their relationships.

DEFINITION 2.1 (BASIC DATA GRAPH). *A basic data graph, $G(V, E)$, is a node and edge labeled directed graph, where*




(a) (b) (c)

**Figure 5: Example mappings of two IS-A hierarchies (a) node-to-node, (b) node-to-tree, and (c) either-or mapping**

- *each node, $v \in V$, corresponds to an entity (or data object),*
- *each edge, $e \in E$, corresponds to a relationship between two entities and is labeled with a relationship name.*

*Each relationship name has an arity constraint* `1-1` *(one-to-one),* `1-N` *(one-to-many),* `N-1` *(many-to-one), or* `M-N` *(many-to-many).*

Intuitively, each node in the graph *asserts* the existence of a distinct object and each edge is a constraint which *asserts* a relationship, (such as IS-A, PART-OF, WORKS-AT, ATTRIBUTE/VALUE) between two objects. Figure 4 presents an example.

## 2.2 Data Paths

A *data path* is a sequence of relationship edges on a graph describing how two entities are related. In this paper, we take data paths as the fundamental *statements of interest.* Such data path based treatment of queries is common in object-centric models, such as OODB and XML.

DEFINITION 2.2 (DATA PATH). *A data path, $p$, on the data graph, $G(V, E)$, is a sequence of edges, $p = \langle e_1, e_2, \ldots, e_{length(p)} \rangle$, where*

$$\forall_{i<length(p)} \;\; dest(e_i) = source(e_{i+1}), \;\; \text{and}$$

*where $source(p) = source(e_1)$ and $dest(p) = dest(e_{length(p)})$ are both data nodes.*

The data path, $Cottontail \overset{InFoodChain}{\rightsquigarrow} Coyote \overset{IS-A}{\rightsquigarrow} MediumMammal$ [2], is an example from the graph in Figure 4.

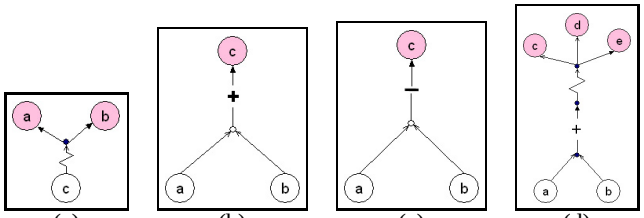
## 2.3 Path Integrity Constraints

The data graph described above captures the data objects and their stated relationships (subject to the associated arity constraints), while it cannot capture more general path integrity constraints to be enforced at the source or in the integrated domain. For example, requirements about the *acyclic* nature of data can only be captured using additional constraints in $IC$. This differentiated treatment is analogous to the *data* vs. *integrity constraints* differentiation common in database management systems [32]. Naturally, whether local (to the source) or global, such path integrity constraints need also to be considered during integration.

# 3. DATA WITH ALTERNATIVES

In this section, we first highlight the need for extending the basic data representation to allow for conflicts when working with imperfectly aligned data. We then propose a data representation extended with alternatives and coordination specifications.

[2]In the rest of the paper, we simply omit the relationship names whenever they are not relevant to the discussion.




(a) (b) (c) (d)

**Figure 6: Example data graphs with (a) choice, (b) positive coordination, (c) negative coordination, and(d) hybrid assertions**

## 3.1 Motivating Examples

In the literature, there is a multitude of data and schema matching algorithms [9, 11, 21, 23–26, 30]. In this paper, we refrain ourselves from assuming any particular matching or alignment strategy. As an example, we consider a mapping scenario, where two concept hierarchies (consisting of concepts and IS-A relationships, with `N-1` arities, among them) are being integrated.

EXAMPLE 3.1 (NODE TO NODE MAPPING). *Let us assume that two nodes $n_{1,i}$ and $n_{2,j}$ from different IS-A hierarchies are identified as representing the same concept. Naturally, once these two hierarchies are integrated, these two nodes must be represented as a single node, $n'$. In other words, $n'$ needs to preserve all the relationships that $n_{1,i}$ and $n_{2,j}$ have in their respective hierarchies. For example, all the children of these two nodes need to become the children of the combined node. However, preserving the original information after integration (while maintaining the appropriate arity constraints) is not always as easy. To see this, consider Figure 5(a), where $a$ and $b$ are two nodes that are mapped to each other. In this example, (since more than one immediate ancestor is not allowed in an integrated IS-A hierarchy) unless $c$ and $d$ are also identified as representing the same concept during mapping, the integrated hierarchy will contain an inconsistency.*

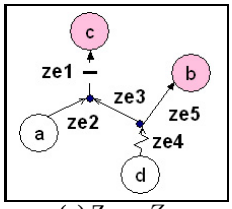
EXAMPLE 3.2 (MAPPING OF GROUPS OF NODES). *Figure 5(b) provides a scenario where a node in one hierarchy is mapped to an entire subtree in the second hierarchy. Thus, in the resulting graph , $h \rightsquigarrow b \rightsquigarrow e \rightsquigarrow \underline{c}$, and $g \rightsquigarrow a \rightsquigarrow \underline{d}$, are independently acceptable paths. Yet, as in the previous example, due to $c$ and $d$ these paths cannot be accepted together.*

EXAMPLE 3.3 (EITHER-OR MAPPINGS). *In many integration scenarios, the user may want to describe* either-or *type of mappings which (positively or negatively) relate the choices for different alternatives. Figure 5(c) provides an example of this type of mapping: in this example, two nodes are mapped with the constraint that the children of the two nodes are not compatible. Unlike the previous examples, this creates a situation where only one of the nodes $a$ and $b$ can belong to a path in the combined hierarchy.*

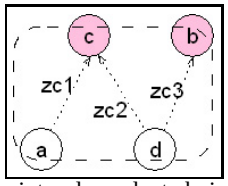
## 3.2 Extending Data with Zones of Choices

As illustrated by the alignment examples, representing data with conflicts requires *asserting* the need for choices and for coordinations among alternatives. In Figure 6, we introduce constructs designed for this purpose, through examples:

- Figure 6(a) presents a data graph with choice semantics. This graph contains a special edge leaving $c$, which can belong to only one path in the data; thus, in this example, either path $c \rightsquigarrow a$ or $c \rightsquigarrow b$ can be interpreted by the user to be true in the data, but not both.
- In a data graph with coordination, alternatives associated with one or more edges may need to be coordinated. In other



(a) Zone $Z_1$



(b) Three inter-dependent choices of $Z_1$

**Figure 7: Zone example: (a) zone $Z_1$ and (b) the three inter-dependent choices of $Z_1$**

words, coordination statements *assert* the need for making the same (or different) choices on involved edges. Figure 6(b) presents a positive coordination (where, if $a \rightsquigarrow c$ is interpreted to hold, then $b \rightsquigarrow c$ must also hold), while

- Figure 6(c) presents a negative coordination requirement (where, $a \rightsquigarrow c$ and $b \rightsquigarrow c$ cannot simultaneously hold).
- The various choice and coordination constraints can be combined to obtain more complex scenarios. Figure 6(d) provides an example with hybrid choice and coordination requirements. This graph asserts that, in the given data, $a$ and $b$ have the same successor, and the shared successor is one of the $c$, $d$, and $e$ data nodes.

In the above graphs, the various edges collectively enforce a set of mutually-dependent relationship constraints. We build the extended data graph on such blocks (or *zones*) of inter-dependent relationship constraints.

DEFINITION 3.1 (ZONE). *A zone (denoting a set of locally inter-dependent constraints) is a directed acyclic graph $Z(Src, Snk, ZV, ZE)$, where*

- *the sources ($Src$) and sinks ($Snk$) are all data nodes (objects, entities),*
- *none of the (internal) vertices ($ZV$) is a data node,*
- *$ZE$ are directed edges which connect sources, sinks, and internal nodes to each other. There are four types of edges:*
  - exclusive edges (marked with \),
  - positive coordination edges (marked with +),
  - negative coordination edges (marked with -), *and*
  - regular edges (unmarked).
- *for any given pair of nodes, $v_i, v_j \in Src \cup Snk \cup ZV$, there exists an undirected path ($v_i \rightsquigarrow_{undir} v_j$) in $Z$ that does not pass through any sources or sinks (i.e., data nodes).*

Figure 7(a) depicts an example zone, $Z_1$. In this example, *source nodes* (lightly shaded) and *sinks* (darkly shaded) are connected through various choice and coordination edges. More specifically, $Z_1(Src_1, Snk_1, ZV_1, ZE_1)$ is such that

- $Src_1 = \{a, d\}$,
- $Snk_1 = \{b, c\}$, and
- $ZE_1 = \{ze_1, ze_2, ze_3, ze_4, ze_5\}$

This zone effectively describes a number of *choices* that the data allows: the paths, $a \rightsquigarrow c$ and $d \rightsquigarrow c$, cannot be in the same data due to the negative coordination edge, $ze_1$, while $d \rightsquigarrow c$ and $d \rightsquigarrow b$ are incompatible due to edge, $ze_4$.

DEFINITION 3.2 (CHOICES OF A ZONE/ ZONAL-CHOICES). *Given a zone, $Z(Src, Snk, ZV, ZE)$, with $k$ sources and $l$ sinks, each path, $zc = i \rightsquigarrow j$, from the $i^{th}$ source to $j^{th}$ sink is said to be an available* choice *for $Z$.*

In the above example, $zc_1 = a \rightsquigarrow c$, $zc_2 = d \rightsquigarrow c$, and $zc_3 = d \rightsquigarrow b$ are three inter-dependent choices (Figure 7(b)).

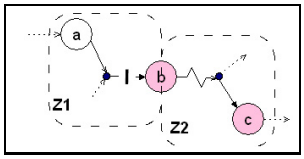


**Figure 8: The data path from node $a$ to node $c$ is passing through two zones; i.e., it can be split into two zonal-choices. We denote this data path as $a \rightsquigarrow b \rightsquigarrow c$**

## 3.3 Zone-Graphs

Since we aim to use zones as the building blocks of the data with conflicts, we consider data graphs that can be partitioned into *zones*. Such graphs are referred to as zone-graphs:

DEFINITION 3.3 (ZONE-GRAPH). *A zone-graph, $G(V, E)$, consists of a set, $\mathcal{Z}$, of zones, where*

- $V = \bigcup_{Z_i \in \mathcal{Z}} (Src_i \cup Snk_i \cup ZV_i)$, *and*
- $E = \bigcup_{Z_i \in \mathcal{Z}} ZE_i$.

*Different zones are allowed to share (and connect through) source and sink data nodes; i.e., $\forall Z_i, Z_j \in \mathcal{Z}$, $Src_i \cap Src_j \supseteq \emptyset$, $Src_i \cap Snk_j \supseteq \emptyset$, $Snk_i \cap Src_j \supseteq \emptyset$, and $Snk_i \cap Snk_j \supseteq \emptyset$. On the other hand, the internal, non-data vertices of the zones or their edges can not be shared; i.e., $\forall Z_i, Z_j \in \mathcal{Z}$, $ZV_i \cap ZV_j = \emptyset$ and $ZE_i \cap ZE_j = \emptyset$. Each zone, $Z \in \mathcal{Z}$, of a zone graph has an associated relationship label, $rel(Z)$, such as IS-A, PART-OF, WORKS-AT, and ATTRIBUTE/VALUE.*

Intuitively, each zone describes the alternative choices and coordination requirements for a mutually-related set of edges (with the same label). The various zones of the graph are separated from each other by their shared data nodes. Conversely, we can also state that the individual zones of a zone-graph are connected to each other through their shared data nodes.

THEOREM 3.1. *Given data-graph $G(V, E)$, its zones can be computed and enumerated efficiently, in $O(E)$ time.*

PROOF SKETCH 3.1. *Due to the undirected connectivity requirement in Definition 3.1, the process of identifying zones can be done in $O(E)$ time, using a connected-components type of algorithm and treating data nodes as* boundaries *of zones.*

Note that in a basic data graph without conflicts (e.g., Figure 4), each edge between two data nodes is a zone with a single source, a single destination, and a single regular edge.

## 3.4 Data Paths on a Zone-Graph

Data paths on a zone-graph are defined similarly to the data paths on a basic data graph (i.e., Definition 2.2). In this more general case, on the other hand, a data path can pass through one or more zones. Thus, we can segment a given data path, $p$, into a sequence of segments, each corresponding to a zonal-choice.

PROPOSITION 3.1 (ZONAL-CHOICES OF A DATA PATH). *A data path, $p = \langle e_1, e_2, \ldots, e_{length(p)} \rangle$, can be segmented into a sequence of zonal-choices, $p = \langle zc_1, zc_2, \ldots, zc_l \rangle$, where each $zc_i = source(zc_i) \rightsquigarrow dest(zc_i)$ is a data path from a source to a sink within the corresponding zone.*

EXAMPLE 3.4. *Figure 8 depicts a data path, $a \rightsquigarrow b \rightsquigarrow c$, from data node $a$ to data node $c$ through data node $b$. In this example, this data path passes through two zones ($Z_1$ and $Z_2$) and, hence, it consists of two zonal choices.*

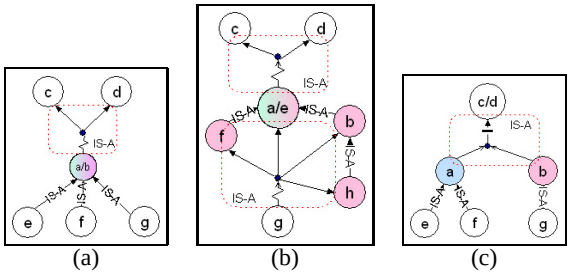



**Figure 9: Zone-graphs obtained through the mapping examples in Section 3.1, Figure 5**

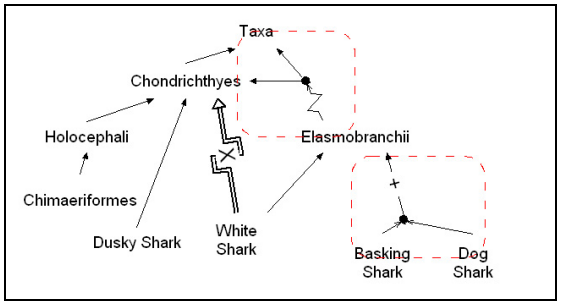



**Figure 10: Integration of the two IS-A hiearchies in Figure 1**

## 3.5 Zonal-Graph Examples

In this subsection, we reconsider the mapping examples (Examples 3.1 through 3.3) earlier in this section and show how zonal representations can capture conflicts that arise during data mapping.

EXAMPLE 3.5 (NODE TO NODE MAPPING). *Example 3.1 (Figure 5(a)) illustrated a case where only one of the data nodes, $c$ or $d$, can be a valid immediate ancestor (parent) of a combined node, due to the* N-1 *arity constraint of IS-A hierarchies. This situation can be captured using the proposed zone-graph as shown in Figure 9(a): children of $a$ and $b$ can use the combined node as their parents and the combined node can have either $c$ or $d$ as its immediate ancestor, but not both. On the resulting zone-graph, some sets of paths, such as $\{e \rightsquigarrow b \rightsquigarrow d, f \rightsquigarrow b \rightsquigarrow d, g \rightsquigarrow a \rightsquigarrow d, e \rightsquigarrow a \rightsquigarrow d\}$ are consistent, while others, such as $\{e \rightsquigarrow b \rightsquigarrow c, g \rightsquigarrow b \rightsquigarrow d\}$ or $\{e \rightsquigarrow b \rightsquigarrow c, g \rightsquigarrow a \rightsquigarrow d\}$, are inconsistent.*

EXAMPLE 3.6 (MAPPING OF GROUPS OF NODES). *Example 3.2 (Figure 5(b)) provided a case where, after the integration, a number of paths (including $h \rightsquigarrow b \rightsquigarrow a \rightsquigarrow c$, $g \rightsquigarrow a \rightsquigarrow d$) are independently valid, but mutually incompatible. Figure 9(b) illustrates how these requirements are captured in a zone-graph using exclusive edges.*

EXAMPLE 3.7 (EITHER-OR MAPPINGS). *Example 3.3 (Figure 5(c)) provided an example of an integration scenario, where the user provides explicit* either-or *mappings which negatively relate the choices for different alternatives. Figure 9(c) shows how this is captured using zones and coordination edges. In this example, while both $e \rightsquigarrow a \rightsquigarrow c$ and $g \rightsquigarrow b \rightsquigarrow d$ are independently valid paths, they cannot be valid together due to coordination requirement.*

Finally, let us also reconsider our motivating example of the scientist working under alternative hypotheses, described by two different taxonomies in Figure 1 in the Introduction.

EXAMPLE 3.8 (TAXONOMY INTEGRATION). *Figure 10 presents a data graph describing the combined systems:*

- *The concept 'Elasmobranchii' can be either directly under the root 'Taxa' or under the concept of 'Chondrichthyes', but not both. This leads to a zone with an exclusive edge.*
- *The concepts 'Basking Shark' and 'Dog Shark' are either both under 'Elasmobranchii' (based on 'Taxa1') or are not known to be in the taxa. This leads to a zone with a positive coordination.*
- *The concept 'White Shark' is under 'Elasmobranchii', only when 'Elasmobranchii' is considered independently of 'Chondrichthyes' ('Taxa 2'). Thus, 'Taxa2' introduces a path integrity constraint which requires that there is no path from 'White Shark' to 'Chondrichthyes'.*

## 3.6 Zonal-Graphs and Path Constraints

As described in Section 2.3, zonal-graphs cannot capture all relevant constraints describing the data and their alignments. For example, multi-zonal statements, such as "*there exists no path with a cycle*" or "*there exists no path from 'White Shark' to 'Chondrichthyes'*", require constraints beyond what can be expressed within a zone. Such constraints, which require multi-path or multi-path level evidence to verify or reject, are kept outside of the scope of the zone-graph specifically to ensure the efficacy of the agreement computation and agreement-based ranking processes. They are instead treated as *path integrity constraints* at a post-processing step, along with the user's feedback. However, path constraints themselves may be conflicting and have to be considered during integration [10, 32, 33]. Unlike existing work, where only conflict-free sources can be integrated, in *FICSR*, the path constraints themselves are assessed during the feedback process. We will revisit the *path constraints* and their place in the validity assessment and feedback process in Section 6.

# 4. ANALYSIS OF ZONAL AGREEMENTS

As discussed in Section 1.3, *FICSR* uses a feedback-driven mechanism to deal with alternative interpretations. In particular, the system relies on the *objective-to-subjective* implication ($sub \leftarrow obj$) of the Desideratum 1 to inform the user about the more likely (i.e., highest source agreement) interpretations for the given data. Intuitively, the *agreement* values measure how much different models of the data agree on the stated relationships, with the default assumption that sources agree unless they explicitly conflict on the data or path constraints. In this section, we focus on the off-line analysis of the zone graph for the computation of *agreement* values on the choices of the individual zones.

## 4.1 Agreement Values of Zonal Choices

We define the models of an individual zone in a way parallel to the definition of models of the data with conflicts (Definition 1.1).

DEFINITION 4.1 (ALTERNATIVE MODELS OF A ZONE). *Given a zone, $Z(Src, Snk, ZV, ZE)$, with $k$ sources and $l$ sinks, a set, $\mathcal{C}$, of choices of $Z$ is said to be a model if all the choices in $\mathcal{C}$ are pairwise consistent with the constraints associated with the edges of $Z$ and there is no other consistent set $\mathcal{C}' \supset \mathcal{C}$ of choices of $Z$ (i.e., $\mathcal{C}$ is set-maximal in $Z$).*

Given these models, the zonal agreement of a choice measures the degree of agreement among alternative models on this choice.

DEFINITION 4.2 (ZONAL AGREEMENT ON A CHOICE). *Given a zone, $Z(Src, Snk, ZV, ZE)$, with $k$ sources and $l$ sinks, the zonal agreement value associated with a choice $zc = i \rightsquigarrow j$ is defined in terms of the alternative* models *in which the choice path, $zc$, is valid versus the total number* models *of the zone $Z$:*

$$agr(zc) = \frac{\#\ models\ of\ Z\ in\ which\ zc\ is\ a\ valid\ path}{\#models\ of\ Z} = \frac{nm(zc, Z)}{nm(Z)}.$$

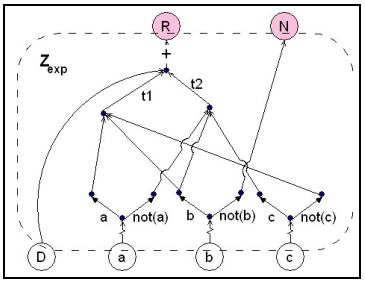



**Figure 11: 3-SAT to zone reduction for the expression** $exp = (a \vee b \vee \neg c) \wedge (\neg a \vee b \vee c)$

EXAMPLE 4.1. *Consider again the zone-graph in Figure 7. In this graph, there are three possible source/sink pairs, i.e., choices* $zc_1 = a \rightsquigarrow c$, $zc_2 = d \rightsquigarrow c$, *and* $zc_3 = d \rightsquigarrow b$. *Among these,* $a \rightsquigarrow c$ *and* $d \rightsquigarrow c$ *are incompatible,* $d \rightsquigarrow c$ *and* $d \rightsquigarrow b$ *are incompatible, while* $a \rightsquigarrow c$ *and* $d \rightsquigarrow b$ *are compatible with each other. Thus, the two alternative models of this zone are* $\{a \rightsquigarrow c, d \rightsquigarrow b\}$ *and* $\{d \rightsquigarrow c\}$ *Another way to look at this is as follows: the choice* $d \rightsquigarrow c$ *is valid in only half of all the possible alternative models. Similarly, the choices* $a \rightsquigarrow c$ *and* $d \rightsquigarrow b$ *are both valid in only half of all the alternative models. Thus, we can conclude that* $agr(a \rightsquigarrow c) = agr(d \rightsquigarrow c) = agr(d \rightsquigarrow b) = 1/2$.

## 4.2 Off-line Agreement Analysis of a Zone

Since the definition of the zonal agreement relies on a model-based interpretation, the computation complexity of this task reflects the cost of computing models of data.

THEOREM 4.1 (PER-CHOICE COMPLEXITY). *Given a zone graph,* $Z$, *and a choice,* $zc$, *the problem of counting the number,* $nm(zc, Z)$, *of models in which* $zc$ *occurs is NP-Complete.*

PROOF SKETCH 4.1. *The proof of NP-completeness of the problem of counting the number of models in which* $zc$ *occurs is through a reduction from the well-known NP-complete 3-SAT problem, which asks the satisfiability of a boolean expression written in conjunctive normal form with 3 variables per clause. While the complete reductive proof is outside of the scope of this paper, we sketch the idea through an example. Consider the expression*

$$exp = (a \vee b \vee \neg c) \wedge (\neg a \vee b \vee c).$$

*The reduction step converts this expression (in polynomial time) into the zone,* $Z_{exp}$ *shown in Figure 11. Given this graph, we restate the 3-SAT problem as follows: Given an expression,* $exp$,

$$satisfiable(exp) \longleftrightarrow (nm(D \rightsquigarrow R, Z_{exp}) \geq 1)\,.$$

*Due to the set-maximality requirement, the models which contain the choice path* $D \rightsquigarrow R$ *will be exactly those models which also use edges* $t_1$ *and* $t_2$*; i.e., satisfying both terms of the conjunctive expressions,* $exp$. *Thus* $exp$ *is satisfied iff there is at least one model of* $Z_{exp}$, *where the path* $D \rightsquigarrow R$ *is used.* □

COROLLARY 4.1 (COMPLEXITY OF ZONE EVALUATION). *Given a zone with* $\mu$ *edges, the complexity of the zonal agreement evaluation is* $O(2^{\mu})$.

PROOF SKETCH 4.2. *This is the worst case complexity based on exponential evaluation. Since, the zonal value agreement computation is NP-complete per Theorem 4.1, we do not expect to find a polynomial solution for this task.* □

It is important to note that the initial zone-partitioning of the data graph[3] and the modular (per-zone) nature of the agreement-analysis prevent this off-line process from becoming costly.

COROLLARY 4.2 (COMPLEXITY OF ZONE-GRAPH ANALYSIS). *The complexity of the zonal agreement evaluation for a complete data graph* $G$, *with* $|\mathcal{Z}|$ *zones, where the largest zone has* $\mu$ *edges is* $O(|\mathcal{Z}| \times 2^{\mu})$.

Thus, zone partitioning significantly reduces the cost of this off-line process: $O(|\mathcal{Z}| \times 2^{\mu}) \ll O(2^{|\mathcal{Z}| \times \mu})$, which would have been the cost of agreement computation without zone partitioning. The zone-graph analysis is an off-line pre-processing process which is performed on the integrated data graph once, at the bootstrap phase.

# 5. QUERY PROCESSING AND RANKED CANDIDATE ENUMERATION

Since the goal is to help the user identify the best matches to her query, at the first stage of the query processing and conflict resolution, *FICSR* provides the user with a set of high-agreement candidate matches. These candidates help the user also in seeing the critical conflicts that affect these most-agreed upon (i.e., objectively most likely) results to the query. This, we refer to as the *objective-to-subjective* ($sub \leftarrow obj$) flow of feedback (Section 1.4).

Definition 1.2 of ranked interpretations calls for a set, $\mathcal{S}$, of statements to be ranked. This set of statements can include any general statement about the data. In this paper, we focus on *data paths* as the fundamental *statements of interest.* Thus, given a source, $n_{src}$, and destination, $n_{dst}$, nodes, the data paths from $n_{src}$ to $n_{dst}$, are ranked and a small ($\leq k$) subset of candidate data paths from $n_{src}$ to $n_{dst}$ are chosen to be presented to the user based on their agreement values (Figure 3 in the Introduction section).

## 5.1 Computation of Data Path Agreements

Given a data path, $p = \langle zc_1, zc_2, \ldots, zc_l \rangle$, its overall *(multi-zonal) agreement value* can be defined in terms of the agreements of the choices involved in it. Since

- the agreement value of a *choice* in a given zone is the ratio of the number of models of the zone in which the choice is valid to the number of all possible models of the zone, and
- each zonal alignment choice is independent from the choices of the other zones (modulo the path constraints that will be enforced at a later stage, in Section 6),

we can treat the agreement ratios as independent selection probabilities (Figure 12).

DEFINITION 5.1 (AGREEMENT OF A DATA PATH). *Given a path in its zonal choice representation,* $p = \langle zc_1, zc_2, \ldots, zc_l \rangle$, *we define the* agreement value *of* $p$ *as*

$$agr(p) = \prod_{1 \leq i \leq l} agr(zc_i).$$

In Section 6, we will relax the independence assumption and consider the affects of multi-zonal path constraints that tie choices in a given zone to the choices in the other zones.

## 5.2 Agreement-Ranked Enumeration of Candidates

In order to provide the user with the most likely paths based on the available zonal conflict/agreement analysis, the system needs

[3]Normally, each zone represent a single or closely related few relations, while the data graph can be composed of many relationships.

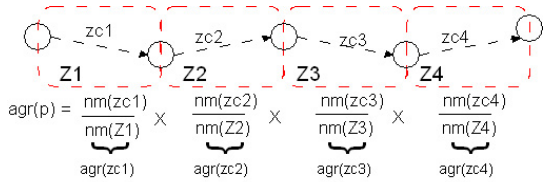



**Figure 12: A path which passes through four zones involves four independently made alignment choices along the way**

**Algorithm** $K - HighAgreementPaths(G, n_{src}, n_{dst}, k)$
Given a zone-graph $G(V, E)$, a source node, $n_{src}$, and a destination node, $n_{dst}$, and a positive integer $k$, do

1. Let $V' = \emptyset$ and $E' = \emptyset$ */* Construct a dual graph*/*
2. Let $\mathcal{Z}$ be the set of all zones of $G$,
3. for all $Z_i \in \mathcal{Z}$ do
   (a) $V' = V' \cup Src_i \cup Snk_i$;
   (b) for all $zc = src(zc) \rightsquigarrow dest(zc)$, where $zc$ is a choice path in $Z_i$;
      i. Let $e_{zc}$ be a new edge with length $length(e_{zc}) = -log(agr(zc))$;
      ii. $E' = E' \cup \{e_{zc}\}$
4. $resultPaths = YenKShortestPath(V', E', length, n_{src}, n_{dst}, k)$
5. return $resultPaths$; */* Return $k$-shortest paths of the dual*/*

**Figure 13: Algorithm for enumerating highest agreement paths**

to identify the highest-agreement data paths. We formally pose this task in the form of a *$k$ highest-agreement data paths problem.*

DEFINITION 5.2 ($k$ HIGHEST-AGREEMENT DATA PATHS). *Given a zone-graph $G(V, E)$, a source node, $n_{src}$, and a destination node, $n_{dst}$, identify the $k$* highest-agreement *data paths from the source node, $n_{src}$, to the destination node, $n_{dst}$.*

Since the data schema (i.e., relationships between entities) themselves are subject to interpretation, the enumeration of the data paths cannot rely on *(ranked)* join techniques used in fuzzy data management systems where the imprecision is associated with the values and objects, but the schema is fixed and known advance [13]. Instead, as shown in Figure 13, we solve the $k$ high-agreement data paths problem by translating it into the *$k$-shortest simple paths*, a classical problem in graph theory. Due to its applications in various domains, such as networking, this problem has been studied extensively. Among the alternative solutions, Yen's algorithm for $k$-shortest paths is preferred due to its general and optimal nature [27, 34]. This algorithm searches shortest paths in a "pseudo"-tree containing $k$ shortest loopless paths. First the shortest path is obtained; then, this path is deviated to obtain the $2^{nd}$ shortest path. This process is repeated, one new shortest path at a time, until $k$ shortest paths are found. We use a version of Yen's algorithm [27].

THEOREM 5.1 (CORRECTNESS). *The $k$-highest agreement algorithm presented in Figure 13 is correct.*

PROOF. Given a graph $G(V, E)$, the algorithm in Figure 13 constructs a dual graph, $G'(V', E')$, where the choices in each zone are replaced with explicit edges between the corresponding source/sink pairs. Thus, it is trivial to show that there is a one-to-one mapping between any data path on $G(V, E)$ and a path on $G'(V', E')$.

In Step 3(b)i of the algorithm, given a choice $zc$ in $G$, the length of the corresponding edge in $G'$ is set to

$$length(e_{zc}) = -log(agr(zc)),$$

which is never negative. Thus, the $k$-shortest path algorithm ran on $G'$ (Step 4) returns $k$ simple paths, such that the term

$$\sum_{zc \in path} -log(agr(zc)) = log\left(\prod_{zc \in path} \frac{1}{agr(zc)}\right),$$

is minimized. Since the $log$ function is monotonic, this corresponds to the minimization of the term $\prod_{zc \in path} \frac{1}{agr(zc)}$ or, equivalently, the maximization of the term

$$\prod_{zc \in path} agr(zc).$$

By Definition 5.1, this term is equal to the agreement of the data path on the original zone graph $G$. Hence, these enumerated paths are also the $k$ *highest-agreement* data paths in $G$. □

THEOREM 5.2 (COMPLEXITY). *The worst case execution time of the $k$* highest-agreement *data paths algorithm in Figure 13 is $O(kn(m^2 + nlogn))$.*

PROOF. Given a dual graph, $G'$, with $n'$ nodes and $m'$ edges, Yen's algorithm would identify the $k$-shortest simple paths in $G'$ with the worst case time of $O(kn'(m' + n'logn'))$ [27, 34].

Given a graph, $G$, with $n$ nodes and $m$ edges, the algorithm creates a dual graph, $G'$, with at most $n' = n$ nodes and $m' = m^2$ edges. The worst case occurs when a zone in $G$ with $u$ edges leads to $O(u^2)$ individual choices, each with a corresponding edge in $G'$.

In the first phase of the algorithm (Steps 1 through 3), the dual graph creation costs $O(m^2)$ time. In the second phase (Step 4), given the dual graph, $G'$, with $m' = O(m^2)$ and $n' = n$ nodes, Yen's algorithm costs $O(kn(m^2 + nlogn))$ time. Thus, the worst case of the algorithm in Figure 13 is $O(kn(m^2 + nlogn))$. □

Note that, in general, not all nodes and edges on the graph, $G$, need to be involved in the enumeration process. In fact, the only nodes that are relevant for path enumeration are those nodes that are *reachable* from the source node, such that the destination node is *reachable* from them. A reachability graph (generated off-line) is used to limit the nodes and edges that are involved in the generation of $G'$, reducing the cost of the candidate path enumeration.

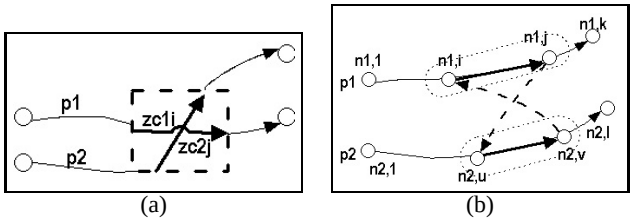
# 6. MULTI-ZONAL PATH CONSTRAINTS

After the enumeration of the $k$ highest-agreement paths based on the zone-graph (i.e., following the initial assumption that the zonal choices are independent from each other), *FICSR* considers multi-zonal path constraints (such as *acyclicity*) to assess the *validities* of the candidate paths. However, *FICSR* recognizes that, while they are used to assess the candidate paths, path constraints from different sources themselves may require assessment. When the highest-agreement paths are conflicting with the path constraints, this may result in (a) lowering of the zone and path agreement values computed in the earlier processing stages or (b) reducing of the validities of the path constraints themselves.

Since candidate paths are weighted with *agreement* values quantifying their likelihood based on the zone-graph, the conflicts between available candidate paths and the constraints must also reflect the *likelihood* of their inconsistencies. This leads to the concept of *degree of conflict* between a set of paths and constraints.

DEFINITION 6.1 (DEGREE OF CONFLICT). *Given a set of paths $P$ (on a data-graph $G$) and constraint $\Theta$, the degree of conflict between $P$ and $\Theta$ (denoted as $conflict_G(P, \Theta)$) is defined as*

$$\frac{|\{M | (M \supseteq P \; is \; a \; model \; of \; G) \wedge (M \cup \Theta \; is \; inconsistent)|)}{|\{M | M \supseteq P \; is \; a \; model \; of \; G)|}$$



(a) (b)

**Figure 14: (a) A zone-conflict between paths $p_1$ and $p_2$; (b) a cycle due the subpaths of $p_1$ and $p_2$ and other paths in graph $G$**

Thus, the degrees of conflict can be treated as agreement values (of the integrated data, on the conflict statement, within the context of the given set of paths). Since the problem of counting the models is NP-complete (Section 4.2), quantification of the degrees of conflict needs to avoid explicit enumeration. Next, we classify the conflicts into two major types and discuss how to compute corresponding degrees of conflict efficiently.

DEFINITION 6.2 (ZONAL CONFLICTS). *Given two paths in their zonal choice representations, $p_1 = \langle zc_{1,1}, zc_{1,2}, \ldots, zc_{1,u}\rangle$ and $p_2 = \langle zc_{2,1}, zc_{2,2}, \ldots, zc_{2,v}\rangle$, there is a zonal conflict due to zonal choices, $zc_{1,i}$ and $zc_{2,j}$, iff the two zonal choices are in the same zone, $Z$, but they are not compatible according to the zonal constraints. This is illustrated in Figure 14(a). Given two paths $p_1$ and $p_2$, and incompatible choices $zc_{1,i}$ and $zc_{2,j}$ on them, the zonal conflict is certain:*

$$conflict_G(\{p_1, p_2\}, \texttt{ZONE}(zc_{1,i}, zc_{2,j})) \equiv 1.0.$$

*The degree of zonal conflict is* 0 *otherwise.*

In other words, the degree of zonal conflicts can be computed in quadratic worst case time in the size of the result paths.

DEFINITION 6.3 (PATH INTEGRITY CONFLICTS). *Given a path $p$, let $\mathcal{SP}_p$ be the set of all subpaths in $p$. Then, given a set, $P = \{p_1, \ldots, p_m\}$, of paths in $G$ and an $m$-ary constraint predicate, $\Theta : \mathcal{SP}_{p_1} \times \ldots \times \mathcal{SP}_{p_m} \rightarrow \{true, false\}$, a constraint violation occurs iff there exists an $m$-tuple $\vec{\texttt{st}} \in \mathcal{SP}_{p_1} \times \ldots \times \mathcal{SP}_{p_m}$ of subpaths, such that $\Theta(\vec{\texttt{st}})$ evaluates to* false.

*The degree of the path conflict is measured using the likelihood of the sub-paths in $\vec{\texttt{st}}$ and the likelihood of any further subgraph $G' \subseteq G$ contributing to the conflict:*

$$conflict_G(P, \Theta(\vec{\texttt{st}})) \equiv \left(\prod_{1 \leq i \leq m} agr(\vec{\texttt{st}}[i])\right) \times agr(G').$$

Path integrity constraints include limits on path lengths, reachability constraints between objects, overlap and intersection constraints on the data paths, and acyclicity constraints (Figure 14 (b)).

EXAMPLE 6.1 (ACYCLICITY CONFLICTS). *Let us be given two paths, $p_1 = n_{1,1} \rightsquigarrow \ldots \rightsquigarrow n_{1,k}$ and $p_2 = n_{2,1} \rightsquigarrow \ldots \rightsquigarrow n_{2,l}$. Let also, $sp_1 = n_{1,i} \rightsquigarrow \ldots \rightsquigarrow n_{1,j}$ and $sp_2 = n_{2,u} \rightsquigarrow \ldots \rightsquigarrow n_{2,v}$ be two subpaths on $p_1$ and $p_2$. As depicted in Figure 14 (b), there exists a cycle involving the subpaths $sp_1$ and $sp_2$ iff*

$$reachable(n_{1,j}, n_{2,u}) \wedge reachable(n_{2,v}, n_{1,i}).$$

*Based on this, we can measure the degree of any acyclicity violation involving subpaths $sp_1$ and $sp_2$ using the* agreement *values of the involved paths as follows:*

$$conflict_G(\{p_1, p_2\}, \texttt{NOCYC}(sp_1, sp_2)) \equiv agr(sp_1) \times agr(sp_2) \times agr(reachable(n_{1,j}, n_{2,u})) \times agr(reachable(n_{2,v}, n_{1,i})).$$

*The agreement values on the statements regarding the* reachability *of one node in the zone graph from the other can be computed by treating the graph as a Bayesian network (or a Markov Chain) and applying probabilistic reachability analysis techniques [7].*

THEOREM 6.1. *Let $\Theta$ be an $m$-ary path constraint. Let $P$ be a set of $m$ paths, such that the $i^{th}$ argument of $\Theta$ is selected from the set, $\mathcal{SP}_{p_i}$, of subpaths of path $p_i \in P$. Then, the number of conflicts that need to be assessed in the worst case is $O(\prod_{1 \leq i \leq m} length(p_i)^2)$*

The proof of the theorem follows trivially from the quadratic number of subpaths of a given path. This result shows that focusing on the context provided by the candidates paths (as opposed to counting all models) renders the conflict degree computation process tractable. This complexity can be further reduced by pruning conflicts described by *stabbed subpath-tuples.*

DEFINITION 6.4 (STABBED SUBPATH-TUPLES). *Let $\mathcal{ST}$ be a set of $m$-tuples of subpaths. Then, $\vec{\texttt{st}}_x \in \mathcal{ST}$ is a* stabbed subpath-tuple *iff $\exists \vec{\texttt{st}}_y \neq \vec{\texttt{st}}_x$ s.t. $\forall_{1 \leq i \leq m}\ \vec{\texttt{st}}_y[i] \subseteq \vec{\texttt{st}}_x[i]$.*

Intuitively, if $\vec{\texttt{st}}_y$ stabs $\vec{\texttt{st}}_x$, then $\vec{\texttt{st}}_y$'s subpaths (hence the conflicts these paths are involved in) are more specific. In Section 9, we will evaluate the impact of pruning stabbed $m$-tuples.

# 7. SYSTEM FEEDBACK GENERATION

The concepts of *degrees of conflict* introduced above, together with the concepts of *zones*, *choices*, and *zonal agreements* introduced in Sections 3 and 4, enable us to articulate the basic postulates that any statement needs to obey to be accepted *valid*.

## 7.1 Postulates of Validity

The following postulates characterize the dependencies between validities of zonal-choices, paths, and constraints:

POSTULATE 1 (ZONAL CONFLICTS AND VALIDITY). *If two valid paths are conflicting within a given zone, then the corresponding zonal choices cannot be simultaneously* valid*: i.e., it holds that for all pairs of paths, $p_1$ and $p_2$*

$$conflict_G(\{p_1, p_2\}, \texttt{ZONE}(zc_1, zc_2)) \rightarrow \neg valid(zc_1) \vee \neg valid(zc_2) \vee \neg valid(\texttt{ZONE}(zc_1, zc_2)).$$

POSTULATE 2 (PATH CONSTRAINTS AND VALIDITY). *If a given set of paths and a path constraint are in conflict, then it cannot be true that all paths and the constraint are simultaneously valid: i.e., it holds that for a given set, $P$, of paths and a path integrity constraint, $c$, violated by the subpath tuple $\vec{\texttt{st}}$ from $P$,*

$$conflict_G(P, c(\vec{\texttt{st}})) \rightarrow \neg valid(c) \vee \left(\bigvee_{\vec{\texttt{st}}[i] \in \vec{\texttt{st}}} \neg valid(\vec{\texttt{st}}[i])\right)$$

These postulates imply that whenever a conflict occurs, the validities of some of the (sub)paths and/or the constraints involved in the conflicts need to be correspondingly reduced.

## 7.2 Validity vs. Agreement

Note that both agreement and validity values assess the preferred order of statements. *Agreement* describes the objective ranked interpretation based on the zonal graph, while the *validity* reflects the path constraints as well as user feedback to capture the subjective ranking. Let us define the *normalized value* $norm(v)$, of the agreement value, $v$, of a (path/constraint/conflict) statement $S$ as $\left(1 - \frac{log(v)}{log(v_{min})}\right)$, where $v_{min} > 0$ is the smallest agreement value

in the input. Note that for the statement with the smallest agreement, $v_{min}$, the normalized agreement value is equal to 0. On the other hand, for any statement, $S$, with objective agreement value $v = 1$, the value of $norm(v)$ is also equal to 1. This implies that for all input statements, their normalized values are between 0 and 1 and increase monotonically with their agreement values. Given these normalized agreement values, the subjective to objective correspondence (Desideratum 1) implies that subjective validity values should be related to the objective agreement values.

DESIDERATUM 2. *For any path (candidate path or subpath), p, it should be that*

$$valid(p) \sim norm(agr(p))$$

$$\forall_{zc \in zonalchoice(p)} \; valid(zc) \sim norm(agr(zc)).$$

This desideratum implies a correspondence between the validity of a given path and the validities of the choices involved.

PROPOSITION 7.1 (ZONAL CHOICES AND PATH VALIDITIES). *A path's* validity *is related to the validity of its zonal choices; in particular, it holds that for any path, p,*

$$valid(p) = 1 + \sum_{zc \in zonechoice(p)} (valid(zc) - 1).$$

Intuitively, the use of $\sum$ to relate validity of a path to the validities of its zonal choices is similarly motivated to the use of addition in the computation of $k$ highest agreement paths in Section 5.2.

PROOF SKETCH 7.1. *Desideratum 2 and the fact that* $agr(p) = \prod_{zc \in zonalchoices(p)} agr(zc)$ *together imply that*

$$\underbrace{1 - \frac{log(agr(p))}{log(v_{min})}}_{valid(p)} = 1 - \left( \sum_{zc \in zonalchoices(p)} \frac{log(agr(zc))}{log(v_{min})} \right)$$

*The left hand side of the equation can be rewritten as*

$$1 + \sum_{zc \in zonalchoices(p)} \left( \underbrace{\left(1 - \frac{log(agr(zc))}{log(v_{min})}\right)}_{valid(zc)} - 1 \right).$$

*Thus,* $valid(p) = 1 + \sum_{zc \in zonechoice(p)} (valid(zc) - 1)$. □

## 7.3 Measuring Validity

Since the conflict and agreement values associated with the paths and constraints are non-boolean, we cannot rely on boolean, model-based schemes to assess validity. Thus, instead of relying on a costly and boolean, maximal clique-based formulations (as in [22]), *FICSR* computes validity values by translating the postulates and user-feedback into a fuzzy constraint program. A fuzzy set, $F$, with domain $D$ is defined using a membership function, $F : D \rightarrow [0, 1]$. A fuzzy predicate then corresponds to a fuzzy set; instead of returning *true(1)* or *false(0)* values for propositional functions, fuzzy predicates return the corresponding membership values. The meaning of the constraint program depends on the fuzzy semantics chosen for the logical operators $\wedge$ and $\vee$. It is well established that the only fuzzy semantics which preserves logical equivalence of statements (involving conjunction and disjunction) and is also monotonic is the `min` semantics, where $a \wedge b = min(a, b)$, $a \vee b = max(a, b)$, and $\neg a = 1 - a$. Therefore, *FICSR* uses fuzzy `min` semantics to translate the postulates into a constraint program.

An example constraint program is presented in Figure 15. Here, *penalty* variables represent the degrees of conflicts observed in the data within the context of the query. Given such a constraint program, *FICSR* searches for a solution with maximal constraint validity (in other words,

- *validities of path constraints need to be maximized* and
- *validities of zonal constraints need to satisfy Desideratum 2*),

and minimal deviation from Postulates 1 and 2 (that is,

- *zonal and path constraint penalties must match the (normalized) degrees of conflict*).

Thus, the **objective** function for the constraint program is

$$\texttt{MAX} \quad \underbrace{\omega_1 \sum_{ic \in IC} valid(ic)}_{Path\ consts.} - \underbrace{\omega_2 \sum_{zc \in ZC} |valid(zc) - norm(agr(zc))|}_{Desideratum\ 2\ (zonal\ constraints)}$$
$$\underbrace{- \omega_3 \sum_{\chi \in ZoneConf} | \, penalty_{zone}(\chi) - norm(conflict_G(\chi)) \, |}_{Postulate\ 1\ (zonal\ conflicts)}$$
$$\underbrace{- \omega_4 \sum_{\xi \in PathConf} |penalty_{path}(\xi) - norm(conflict_G(\xi))|}_{Postulate\ 2\ (path\ integrity\ conflicts)}$$

Here, $\omega_1$ through $\omega_4$ are the preference weights associated with different desiderata and postulates. To enable the user to explore the alternatives and provide more informed decisions regarding the validities of the paths and constraints, the preference of the path integrity constraints ($\omega_1$) is left *flexible (or less important)* in the initial cycles of the feedback process and rendered incrementally *stronger* through user assessment and feedback. This is similar to the way IR systems approach to the assessment of features whose relevance are not known in advance [35].

## 7.4 System Feedback

Once the validity values are obtained, the next step in the feedback process is the recomputation of the agreement values of choices and paths based on their validity assessments. Given a computed validity value $\nu$ and the $v_{min}$ (used earlier for normalization), *FICSR* computes the corresponding agreement value, $agr_\nu$ (based on Desideratum 2) as follows:

$$agr_\nu = norm^{-1}(\nu) = e^{(1-\nu) \times (log(v_{min}))}.$$

At this stage, for every zonal choice and candidate result the user is presented with two agreement values: one obtained through the zonal agreements and the other obtained by incorporating path constraints. Furthermore, the user is also presented with validities associated with constraints as well as degrees of individual conflicts. This is referred to as the objective-to-subjective information flow ($sub \leftarrow obj$), or the *system feedback*.

# 8. USER FEEDBACK

The postulates presented above enable a fuzzy framework in which the validities of the candidate paths as well as the path integrity constraints can be assessed. However, the postulates (and the resulting fuzzy constraint program) can be satisfied in multiple ways and, thus, the user feedback is necessary to inform the system on the appropriate adjustments [29]. This is the subjective-to-objective information flow ($sub \rightarrow obj$), or the *user feedback*.

Although, as discussed in the Introduction, the motivation is similar, it is important to note a significant difference between the treatment of feedback in *FICSR* and in traditional IR systems. Since in the information retrieval context, the major challenge is to identify user's interest indirectly, from examples, a significant part of the effort goes into the learning task. In *FICSR*, on the other hand, the query as well as conflicts are known, but the user's ranking is

Given candidate paths, $P$, conflicting subpaths, $SP$, zonal choices, $ZC$, and path integrity constraints, $IC$,

$\forall c \in ZC \cup IC \quad 0 \leq valid(c) \leq 1;$

$\forall p \in P \cup SP \quad 0 \leq valid(p) \leq 1;$

$\forall zc_1, zc_2 \in ZC \quad 0 \leq valid(\texttt{ZONE}(zc_1, zc_2)\}) \leq 1;$ /* *Validities of zonal constraints*/*

$\forall sp_1, sp_2 \in SP \quad 0 \leq valid(\texttt{NOCYC}(sp_1, sp_2)\}) \leq 1;$ /* *Validities of acylicity constraints*/*

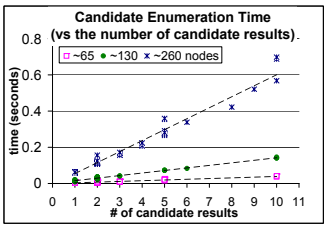
$\forall p \in P \cup SP \quad valid(p) = 1 + \sum_{zc \in zonalchoices(p)}(valid(zc) - 1)$ /* *Proposition 7.1*/*

/* *Postulates 1 and 2*/*

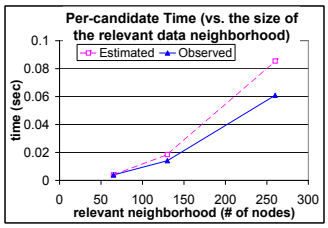
$\forall zc_1, zc_2 \in ZC \quad 0 \leq penalty_{zone}(zc_1, zc_2) \leq max\{1 - valid(zc_1), 1 - valid(zc_2), 1 - valid(\texttt{ZONE}(zc_1, zc_2)\})$

$\forall sp_1, sp_2 \in SP \quad 0 \leq penalty_{cyc}(sp_1, sp_2) \leq max\{1 - valid(sp_1), 1 - valid(sp_2), 1 - valid(\texttt{NOCYC}(sp_1, sp_2))\}$

**Figure 15: Fuzzy constraint program capturing the validity postulates**

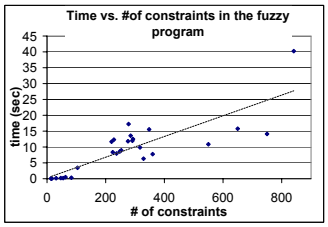



(a) time vs. # candidate results

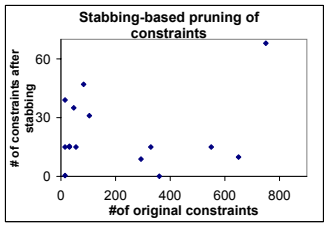



(b) time vs. neighb. size

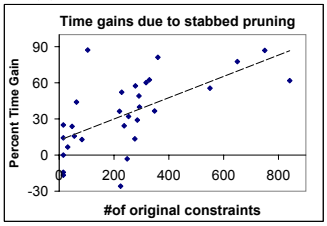
**Figure 16: Candidate enumeration complexity**



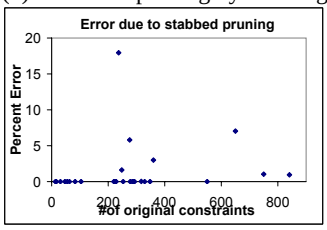

(a) time vs.# of constraints




(b) constraint pruning by stabbing




(c) gains in time due to stabbing




(d) quality of stabbing

**Figure 17: Scatter-plots showing validity analysis complexity and stabbing effectiveness (each point represents a single case)**

not. In fact, unlike in IR systems where the object features used for ranking are many times hidden from the user, in *FICSR*, given the system feedback (Section 7.4), the user can observe individual conflicts and provide concrete adjustments, without the system having to *learn* from examples. Thus, given the initial set of candidates and system feedback (agreement values and degrees of conflict), the user's feedback is primarily in the form of preferred validity rankings and assessments (e.g., $valid(p_i) > valid(p_j)$, $valid(c_k) > max\{valid(p_i)|p_i \in candidates\}$, or $valid(c_k) = 1.0$). These describe users' ranked interpretation of the statements regarding the candidate paths, constraints, and conflicts.

Through the QUEST interface [29], *FICSR* enables two types of feedback: *hypothetical* and *committed*. Hypothetical (*soft*) feedback is captured similarly to the path constraints: mismatches between the feedback-constraint and the data are implicitly minimized within the objective function. The user can observe the effect of the feedback-constraint by varying its weight in the optimization function (for example changing its importance as opposed to a certain conflicting path constraint) and observing its various impacts. The committed (*hard*) feedback is inserted into the program as constraints to be enforced, overriding any conflict from the sources.

# 9. EVALUATION

In order to evaluate the proposed algorithms, we used input taxonomies of varying sizes and with varying degrees of misalignments. In the results presented in Figures 16 through 17, we merged 2 to 8 taxonomies with high (upto $\sim$ 15%) pairwise misalignments. The queries on the integrated taxonomies involved fragments (reachability neighborhoods) of upto $\sim$ 260 data nodes. Experiments ran on a 2.8GHz Pentium with 2GB main memory.

**Candidate enumeration.** Figure 16 presents the execution times for candidate enumeration phase. The scatter-plot in Figure 16(a) depicts the time against the candidate results returned for different integration scenarios. The enumeration time is (as expected) linearly correlated with the number of candidate matches. The figure also shows that the fragment of the data involved in enumeration affects the complexity. To further observe this effect, in Figure 16(b), we compare the observed per-candidate enumeration time with the estimated worst case time complexity according to Theorem 5.2. The results show that the measured time is indeed upperbounded by the predicted values. It is important to note that, in the experiments, the per-candidate enumeration time was less than 100ms, even for highly-conflicting, highly-dense data neighborhoods.

**Constraint analysis and feedback generation.** In the second phase, we employed a constraint solver (LINGO) to observe the time required to solve the optimization problem described in Section 7. Figure 17 presents the results. In Figure 17(a), we plot execution time against the number of (zonal and path integrity) constraints required. The observe times were monotonically correlated with the number of constraints involved. Although they reached 45 seconds in one extreme case with $> 800$ constraints, the times were mostly under 20 seconds. However, using the stabbing strategy presented in Section 6, constraints were pruned significantly to less than 70 for all cases (Figure 17(b)). As a consequence, the cost of this step was reduced in time upto 90%, especially for cases with high number of constraints (Figure 17(c)). For instance, for the extreme 45 seconds run, the stabbed version was $\sim$ 60% faster. Figure 17(d) shows that, although stabbing may cause reduction in the value of the objective function (i.e., may miss some constraints), except very few cases, the error is negligible.

# 10. RELATED WORK

In 80s, the problem of dealing with incomplete and/or inconsistent information led to multiple model semantics [14], non monotonic reasoning systems [12], and the theory of *belief revision* [1]. [36] defines a choice semantics for logic programs with negation. In 90s, uncertain information at the data value level has been modeled through different types of *nulls* [8, 16]. The null-value treatment has also been extended to missing and partial data in semistructured domains in [20]. More recently, in *TRIO* [3] system, the existence of alternative database instances is captured through the notion of uncertainty associated to the available data values. *Probability* values capture uncertainty, while *lineage* captures data items' derivation. As in *FICSR*, in *TRIO* the probabilities

associated to attribute values capture the likelihood in the alternative scenarios. Unlike *FICSR*, in *TRIO* the probability values are taken to be known at the storage time (or derivation time for derived items), and structural uncertainties (i.e., schema/constraint-level uncertainty and conflicts) or coordination requirements are not captured. *FICSR* instead provides a uniform *object centered* model that can express uncertainty both in data and in constraints governing structural relationships. In *TRIO* lineage guarantees that derivations are coherent. In *FICSR*, however, the user can specifically choose to trust and combine results that are not coherent; i.e., the imperfect alignment constraints dictated by the initial mapping strategy can be overruled by users' data interpretations.

*Orchestra* [31] focuses on managing disagreement (at both schema and instance levels) among different collaborating members and addresses the problem of propagating source updates. Updates are propagated among participants through a series of *acceptance rules*; i.e., acceptance of tuples from other sources is an *option* encoded within these acceptance rules. No global consistency requirement is imposed. Unlike the user feedback mechanisms in *FICSR* which enables domain knowledge in conflict resolution, in *Orchestra* acceptance rules are simply predicated on the fixed trust values and priorities associated to the sources.

[4] investigates formal characterizations of the notion of consistent data in a possibly inconsistent database, and discusses the complexities of consistent query answering. Similarly, [19] addresses modeling and query answering complexity issues in data integration systems. A global architecture (mediator) provides a reconciled, integrated, virtual view over the real data sources. Unlike *FICSR*, integrity constraints are taken as inherently valid and, thus, the focus is on returning answers while guaranteeing consistency with respect to the given constraints. Instead, in *FICSR*, we do not take integrity constraints (which can be source specific) to be inherently true. The domain expert has the final say in the validity of the integration constraints, along with the data.

# 11. CONCLUSION

In this paper, we presented innovative techniques to deal with the computational complexity and the ill-defined nature of the conflict resolution problem. In particular, we presented a novel, feedback-driven approach to query processing in the presence of conflicts. The novel feedback process relies on a ranked interpretation of the data, as opposed to more traditional model-based interpretations. The objective-to-subjective correspondence of the ranked interpretations enables the user to explore the available data within the context of a query and be informed regarding data and relationship-constraints critical to a given query before providing feedback. In a similar fashion, the subjective-to-objective correspondence of the ranked interpretations inform the system regarding user's preferences and domain knowledge within the context of a query. We provided data structures and algorithms that enable an efficient and effective implementation of the proposed feedback cycle.

# Acknowledgment

We thank Prof. Kintigh at the ASU School of Human Evol. & Social Change for his feedback.

# 12. REFERENCES

[1] C. Alchourron, P. Gardenfors, and D. Makinson. On the logic of theory change: Partial meet contraction and revision functions", In J. Symbolic Logic, 1985

[2] M. Arenas and L. Libkin. XML data exchange: consistency and query answering. In *PODS*, 2005.

[3] O. Banjelloun, A. D. Sarma, A. Halevy, and J. Widom. ULDBs: Databases with uncertainty and lineage. In *VLDB*, 2006.

[4] L. Bertossi. Consistent query answering in databases. *VLDB*,2006.

[5] A. Bonifati, E. Chang, and L. Lakshmanan. Heptox: Marrying XML and heterogeneity in your P2P databases. In *VLDB*, 2005.

[6] P. Bouquet, F. Giunchiglia, F. van Harmelen, L. Serafini, and H. Stuckenschmidt. COWL: Contextualizing Ontologies. *ISWC'03*.

[7] C. Boutilier, R. I. Brafman, and C. Geib. Structured Reachability Analysis for Markov Decision Processes. UAI'98.

[8] K.S. Candan, J. Grant, and V. Subrahmanian. A unified treatment of null values using constraints. *Information Systems J.*, 98(1–4), 1997.

[9] K.S. Candan, J.W. Kim, H. Liu, R. Suvarna. Discovering mappings in hierarchical data from multiple sources using the inherent structure. J. of KAIS, 10, 2, 185-210, Aug. 2006.

[10] S. Conrad, M. Höding, G. Saake, I. Schmitt, and C. Türker. Schema Integration with Integrity Constraints. In *BNCOD*, 1997.

[11] A. Doan, P. Domingos, and A. Y. Levy. Learning source description for data integration. In *WebDB*, 2000.

[12] J. Doyle. A truth maintenance system. *J. of Artificial Intelligence*, 1979.

[13] R. Fagin. Combining fuzzy information from multiple systems. In *PODS*, 1996.

[14] M. Gelfond and V. Lifschitz. The stable model semantics for logic programming. In *Proc. ICSLP*, 1988.

[15] A. Halevy, Z. G. Ives, D. Suciu, and I. Tatarinov. Schema mediation in peer data managemen. In *ICDE*, 2003.

[16] T. Imielinski and W. Lipski. Incomplete information in relational databases. *JACM*, 31(4):761–791, 1984.

[17] A. Jhingran. Enterprise information mashups: Integrating information, simply. In *VLDB*, 2006.

[18] A. Kementsietsidis, M. Arenas, and R. Miller. Mapping data in peer-to-peer systems: Semantics and algorithmic issues. *SIGMOD'03*.

[19] M. Lenzerini. Data integration: a theoretical perspective. *PODS'02*.

[20] M. Liu and T. W. Ling. A data model for semistructured data with partial and inconsistent information. In *LNCS 1777*, 2000.

[21] J. Madhavan, P. A. Bernstein, and E. Rahm. Generic schema matching with cupid. In *VLDB*, 2001.

[22] R.Mercer and V.Risch. Properties of maximal cliques of a pair-wise compatibility graph for three nonmonotonic reasoning system. In *Answer Set Programming*, 2003.

[23] R. Miller, L. Haas, and M. Hernandez. Schema mapping as query discovery. In *VLDB*, 2000.

[24] T. Milo and S. Zohar. Using schema matching to simplify heterogeneous data translation. In *VLDB*, 1998.

[25] P.Mitra, G.Wiederhold, and M.Kersten. A graph oriented model for articulation of ontology interdependencies. In *EDBT*, 2000.

[26] L. Palopoli, D. Sacca, and D. Ursino. An automatic technique for detecting type conflicts in database schemes. In *CIKM*, 1998.

[27] M. Pascoal and E. Martins. A new implementation of Yen's ranking loopless paths algorithm. *4OR – Quarterly Journal of the Belgian, French and Italian Operations Research Societies*, 2003.

[28] Y. Qi, K. S. Candan, M. L. Sapino, and K. Kintigh. Using QUEST for integrating taxonomies in the presence of misalignments and conflicts. In *SIGMOD Demos*, 2007.

[29] Y. Qi, K. S. Candan, M. L. Sapino, and K. Kintigh. QUEST: QUery-driven Exploration of Semistructured Data with ConflicTs and partial knowledge, CleanDB, 2006.

[30] E. Rahm and P. A. Bernstein. A survey of approaches to automatic schema matching. *VLDB Journal*,2001.

[31] N. E. Taylor and Z. G. Ives. Reconciling while tolerating disagreement in collaborative data sharing. In *SIGMOD*, 2006.

[32] C. Turker and G. Saake. Deriving relationships between integrity constraints for schema comparison. In *Advances in Databases and Information Systems*, 1998.

[33] M. W. W. Vermeer and P. M. G. Apers. The role of integrity constraints in database interoperation. In *VLDBJ*, 1996.

[34] J. Y. Yen. Finding the $k$ shortest loopless paths in a network. *Management Science*, 1971.

[35] C. T. Yu, W. S. Luk, and T. Y. Cheung. A statistical model for relevance feedback in information retrieval. *J. ACM*, 23(2), 1976.

[36] C. Zaniolo. A unified semantics for active and deductive databases. In *Proc. First Int'l Workshop Rules in Database Systems* 1994.