CIHSPS2004 - IEEE International Conference on
Computational Intelligence for Homeland Security and Personal Safety
Venice, Italy, 21-22 July 2004

Database Editing Metrics for Pattern Matching

Enrique H. Ruspini Jerome Thomere Michael Wolverton

Artificial Intelligence Center, SRI International
Menlo Park, California 94025, USA

E-mail: ruspini@ai.sri.com || thomere@ai.sri.com || mjw@ai.sri.com

Abstract – *Pattern-matching techniques are important tools to treat problems in several fields, including bioinformatics, case-based reasoning, information retrieval, and pattern recognition. These procedures are important in homeland security and crime prevention applications because the underlying problems require the discovery, in large databases, of instances of patterns known to be associated with illegal activities.*

While pattern matching may be defined in strict terms as the satisfaction of a logical expression, defining the pattern, by a set of assertions contained in the database, the value of relevant procedures is considerably enhanced by permitting the discovery of approximate matches between database and patterns. The notion of approximate matching is based on the consideration of soft predicates, which may be satisfied to a degree, rather than the conventional crisp predicates of classical logic.

This paper introduces a family of metrics to measure the degree of qualitative match between a database and a pattern, that is, an elastic constraint on database objects and their relations. These metrics provide a formal foundation for the application of graph-editing metrics—measures of the cost associated with graph transformations—to pattern-matching problems.

The degree of matching between database and patterns is determined by means of similarity measures that gauge the resemblance between pairs of objects. In our treatment, these measures have a semantic basis stemming from consideration of knowledge structures, such as ontologies, describing common properties of two objects.

Approximate pattern matching is treated as the process of modifying databases into a transformed database that strictly satisfies the constraints expressed by the pattern. Associated with each transformation is a measure of admissibility derived from the similarity between the original and transformed databases. The degree of matching of database to pattern is defined as the admissibility of the transformation with highest admissibility value.

Keywords – *link analysis, pattern matching, database editing, semantic similarity measures, ontologies, approximate matching*

I. INTRODUCTION

Progress in the development of computational devices and computing techniques combined with advances in the representations of complex real-world objects and systems and the ability to link distributed, heterogeneous data repositories has led to requirements for the identification and retrieval of linked data objects that match constraints expressed through complex *patterns*. From a logical perspective, these patterns may be considered to be axiomatic specifications of a *theory* while retrieved objects, or *pattern instances*, are *models* of that theory.

Informally, the database may be thought of as a collection of objects that are linked by various predefined relations. At a more formal level, facts describing the existence of these objects and their relationships are expressed as the conjunction of the members of a set of instantiated logical predicates such as

`Person`(*Person-1*)
`Event`(*Event-3*)
`Participated`(*Person-1, Event-3*)
`Employment`(*Person-1, Company-2, Position-9*)

Correspondingly, patterns may also be conceived in terms of logic constructs, requiring the existence within the database of certain instances of objects and that of links, or relationships, between them. Typically, a pattern will correspond to a closed first-order-logic expression, such as

$$\exists\, x, y, z, \ldots \quad \texttt{Person}(x) \wedge \texttt{Person}(y) \wedge \texttt{Event}(z) \wedge \ldots \wedge \texttt{Participated}(x, z) \wedge \ldots \wedge \texttt{Participated}(y, z) \wedge \ldots$$

From such a logical perspective, the pattern-matching problem may be regarded as that of finding a correspondence between the variables $x, y, z, \ldots$ and selected database objects (i.e., *variable bindings*) such that the resulting instantiated pattern predicates are, indeed, among the assertions contained in the database. We may also think of the specification of this correspondence between variables and objects as a constructive proof that the database implies the pattern. This perspective is the basis for a number of logical-programming languages and of logical approaches to database representation and manipulation [1].

In many applications of interest in fields as diverse as information retrieval, pattern recognition, case-based reasoning, and crime prevention—the specified patterns are not intended to specify, in a categorical fashion, whether or not linked data objects meet the axiomatic constraints but, rather, as the specification of *prototypical* examples, representing idealized conditions that may be met to different degrees. This type of specification, which is familiar in information-retrieval applications,

permits ranking instances of data structures by their degree of matching with the ideal conditions. For example, a requirement to match the pattern "*Person* P is a *Southern European* who *breeds attack dogs*" might be matched, albeit not perfectly, by an object of the type `Person` who was born in *Central France* (which is close to and overlaps *Southern Europe*) and who *keeps* (but it is unclear whether or not he *breeds*) *wolves*.

In this extended view, patterns do not express strict requirements that are either met or not met. Rather, patterns should be regarded as procedures that rank the adequacy of alternative variable-to-object assignments as potential solutions of a database-retrieval problem. Correspondingly, the values of properties of objects in databases (e.g., *Southern European*) and the nature of the properties themselves (e.g., *breeds*) should be regarded as elastic descriptions that may be met to various degrees. Each possible instantiation matches the pattern to some degree, expressed by a number between 0 and 1 that measures the extent to which such an instance matches the pattern specification. Pattern instances that strictly match, in the logical sense, the pattern specification have a degree of matching equal to 1, while semantically unrelated instantiations—in a sense to be formalized below—have a degree of matching equal to zero.

Patterns may be regarded, therefore, as tools to measure the *similarity*, of potential solutions of a matching problem to a set of *ideal* or *perfect* matches. This similarity measure, which reflects the semantics of the specific problem being considered, is the basis for the definition of numerical measures of *degree of matching*. This conceptualization of the pattern-matching problem also suggests that it may be treated as a generalized (i.e., multivalued) logical program, that is, as a procedure to search a space of potential solutions and to rank their suitability [2], [3] in a manner that extends logical programming approaches [1].

We will define the degree-of-matching function in terms of the *admissibility*, or adequacy, of the modifications required to transform a database into a modified counterpart that best matches the pattern. Such a best match may be informally described as having the largest admissibility value (i.e., lower transformation cost) among all transformations leading to databases matching the pattern from a classical-logic viewpoint. Database transformations are defined as the composition of a sequence of certain *basic edit operations*. Each edit operation is associated with a numerical value gauging its admissibility. The admissibility of a particular transformation is then defined as a function of the admissibility of its component edits.

In this work, we introduce a family of metrics, based on semantics provided by knowledge structures such as ontologies, to determine the extent to which a database—that is, the representation of a collection of instances of real-world objects linked by instances of specified relations—matches the specifications of a pattern, i.e., a set of constraints on the nature of certain data items and their relationships.

The metrics introduced in this paper are conceptually related to graph-based approaches to the representation of complex objects, databases, and evidence. In these *graph-editing* approaches [4], [5], [6], complex graph modifications are characterized in terms of a sequence of several simple graph-edit operations. The admissibility of such a sequence of graph edits is defined as a function of the admissibility of the individual edits in the sequence. As there may be many such sequences leading from the original graph to an edited version that matches the pattern, these methods seek to determine that having the highest admissibility (i.e., that with the least cost). The admissibility of such a sequence is then defined as the degree of matching between the original and the elastic pattern.

Graph-editing techniques provide a useful framework to describe a variety of complex objects while permitting their comparison in terms of the extent (or cost) of the modifications required to transform a graph-based representation of one object into another. These techniques have considerable generality and may be applied to a wide variety of problems. In each application, however, it is necessary that the functions employed to estimate the admissibility of graph transformations reflect the particular meaning attached to each editing operation. This paper is devoted to the derivation, from the perspective provided by a combination of logical and metric perspectives, of specific admissibility measures, called *database-editing metrics*, applicable to pattern matching in databases.

II. ABSTRACT DATA STRUCTURES

Our approach to the derivation of semantic measures of similarity is based on the representation of databases as a set of assertions about a real-world system. From a logical viewpoint, the database may be interpreted as stating that all such assertions are true, or, equivalently, that their logical conjunction is true (*conjunctive interpretation*).

Instantiated n-ary predicates such as

`Employment`(*Person-1*, *Company-2*, *Position-9*)

may be represented themselves as a set of *triples*, or *binary-relation* instances, that link a set of objects to a relation-instance via their roles in that relation instance, e.g.,

`Employee`(*Employment-1*, *Person-1*)
`Employer`(*Employment-1*, *Company-2*)
`Position`(*Employment-1*, *Position-9*)

where *Employment-1* is an internal identifier of an instance of the relation `Employment`, and where the binary relations `Employee`, `Employer`, and `Position` describe the roles that the objects *Person-1*, *Company-2*, and *Position-9* have in *Employment-1*.

Conventional structured database representations, such as those obtained using the relational [7] or the entity-relationship model [8] may be easily mapped into triple-based representations. These representations are also the bases for approaches to represent and exchange semistructured data such as those

embodied in the XML [9] or RDF [10] representation languages.

In what follows, we will assume that a database is represented by a set of triples containing assertions about a real-world system. The database will be interpreted as the logical conjunction of the relations between object instances that are asserted by the triples.

III. SIMILARITY

The notion of *similarity* or *resemblance* is central to the approach presented in this paper. Similarity measures provide the bases for the determination of the *admissibility* of certain transformations of a database into another that meets the logical conditions expressed by a pattern. The notion of admissibility, which may be thought of as being dual to the concept of cost, is introduced to indicate that certain database modifications are more permissible than others. The basic idea is that transformations resulting in a similar database are more admissible (i.e., less costly) than those resulting in substantial difference between the original and transformed data. The *degree of admissibility* of a database transformation is defined in terms of the numerical measures of similarity, which are themselves the counterpart of the notion of *distance*. Similarity measures, mapping pairs of objects into a numeric value between 0 and 1, provide a desirable foundation for the development of a rational theory of database editing, not only because they are related to notions of cost, utility, and admissibilty but also because they provide the basis to extending classical logical relations of inference to approximate, multivalued, counterparts [11].

Similarity functions are measures of *indistinguishability* that may be thought of as being the dual of the notion of bounded distance (i.e., a distance function taking values between 0 and 1). A similarity measure assigns a value between 0 and 1 to every pair of objects o and o' in some space X. Typically, similarity measures d may be obtained from knowledge of distance functions d by simple relations [12] such as $S = 1 - d$. The similarity of an object o to itself is always equal to 1 (corresponding to a distance equal to zero), while the minimum possible value for a similarity function is 0. The advantage of the use of similarities in lieu of distances lies in their advantages as the foundations of logics of utility [3], which provide the bases to combine on a rational basis, measures defining utility, admissibility, and cost from various perspectives. A detailed discussion of the notion of similarity is beyond the scope of this paper. We limit ourselves to pointing out the important properties of similarity (or generalized *equivalence*) functions:

Reflexivity: $S(x, x) = 1$, for all x in X

Symmetry: $S(x, y) = S(y, x)$, for all $x,\ y$ in X

Transitivity: $S(x, y) \geq S(x, z) \circledast S(y, z)$, for all $x,\ y,\ z$ in X,

where $\circledast$ is a *triangular norm*.

A. Semantic distance, similarity, and ontologies

A simple scheme for numerical assessment of the admissibility of edit operations is based on the similarity measures between leaves of an ontological directed acyclical graph (DAG) that measure the extent to which leaf nodes share ancestors in the ontology [3]. This scheme, first proposed by Ruspini and Lowrance in the context of the development of SRI's SEAS [13], is also similar, in spirit, to approaches to defining semantic distance between concepts on the basis of the knowledge provided by a generalized thesaurus [14].

These ontology-based measures of similarity permit only, gauging of the resemblance between different types of unlinked objects. Pattern-matching problems, however, require the consideration of similarity between complex linked structures where the similarity between two such structures depends on the nature of the links and that of the attributes of the related objects. To address this problem, we discuss mechanisms for the derivation of complex similarity measures in terms of simpler constructs.

In our discussion, we will, mainly for the sake of simplicity, assume that every node in an ontology is relevant to the measurement of similarity between classes of objects. In general, however, the measurement of similarity between types is made on the basis of a connected subgraph of the ontology as many properties might be irrelevant to certain types of comparison (e.g., certain items of furniture and cows both have four legs but this shared property is generally irrelevant to the characterization of similarity between these rather different objects).

The relations of set inclusion between nodes in an ontology may be represented in a number of ways by vectors describing whether a node of some type is the ancestor of another node. Ruspini and Lowrance [13] suggested a representation of the j-th node in terms of a vector having the length n of the cardinality of the ontology with the k-th component of that vector, representing whether or not the k-th node is an ancestor of the j-th node, that is,

$$v_k(N_j) = \begin{cases} 1, \text{ if node } k \text{ is an ancestor of node } j, \\ 0, \text{ otherwise.} \end{cases}$$

On the basis of this representation the degree of similarity between the leaf nodes N_i and N_j may be defined by

$$\hat{S}(\, N_i, N_j \,) = \frac{< v(N_i), v(N_j) > -1}{\sqrt{(||v(N_i)||^2 - 1)\,(||v(N_j)||^2 - 1)}}\,,$$

where $v(N_i)$ and $v(N_j)$ are the just-introduced binary vector representations of the nodes N_i and N_j. This similarity measure is a symmetric function taking values between 0 and 1, such that the similarity of a leaf node to itself is always equal to 1.

B. Complex similarity measures

While ontologies permit the construction of semantic-based similarity functions between basic objects (e.g., `Countries` by Type of Economy) and between values of attributes (e.g., `Age`), it is often the case that many of the structures found in a typical pattern-matching problem, involving complex links between primitive objects, may not be amenable to this type of treatment.

Consider, for example, the set of linked objects characterizing an illegal transaction such as `Money Laundering`. Unless this type of object has been subject to some form of ontological characterization, it should be clear that any measure of similarity between objects of this type needs to be based on the similarity of attributes of corresponding objects in each structure.

These complex structures may be characterized, however, through logical expressions, such as

$$\begin{aligned}\texttt{Person}(x) \wedge \texttt{Person}(y) \wedge \texttt{Money-Transfer}(z) \wedge \\ \wedge\ \texttt{Depositor}(x,z) \wedge \texttt{Payee}(x,z) \wedge \ldots \\ \Rightarrow \texttt{Money-Laundering}(x,y,z\ldots.)\,,\end{aligned}$$

which can be employed as the basis for the definition of a similarity measure between money-laundering events as a function of the similarities between the various basic ground predicates and the logical operators that interconnect them [15].

In general, the computation of these complex measures is straightforward. In some cases, however, as when trying to measure similarity between say, people, by the type of company they keep, application of the above approach results in a definition that depends on other values of the measure being defined, as the similarities between associates depend on the nature of the very people being compared (since human association is a symmetric transitive, relation and the people being compared are themselves associates of their associates). While the required measure may be usually derived by solution of the resulting fixed-point problem, it is important to reduce the complexity of the definition as the underlying computational problem may be intractable. These cases, common when considering transitive relations, may require limiting the extent of the structures being compared to reduce such computations.

C. Generalized similarity measures

To determine, on a sound formal basis, the admissibility of each editing operation in terms of the nature of the objects and links involved, it is necessary to derive a general metric characterizing the acceptability of the outcome of an editing operation from the viewpoint of the reference pattern being matched.

Ruspini [11], [15] proposed the measurement of the similarity between two sets by extension to a logical framework of concepts from the theory of metric spaces. The connection between these concepts and utilitarian interpretations of certain possibilistic constructs has been studied recently to a considerable extent [3]. The basic construct of this theory is the function

$$\mathbf{I}(A \mid B) = \min_{o \in B} \max_{o' \in A} S(o, o')$$

defined also over pairs of subsets of X, which measures the *degree of inclusion* of B in A with respect to S, that is, the extent of the minimal metric neighborhood of B that encloses A (in the sense of set inclusion).

The nonsymmetric metric $\mathbf{I}$ will be useful in the definition of costs of basic editing operations as it measures the extent by which the concept represented by the class A needs to be extended (or "stretched") to encompass that described by the class B. The metric $\mathbf{I}$ is a measure of set inclusion since a value of $\mathbf{I}(A \mid B)$ equal to one indicates that every member of B is also a member of A, that is, that B is included in A.

The function $\mathbf{I}$ has a number of useful properties that are the foundation of our approach to the estimation of the degree of matching between a pattern and a database in terms of the $\circledast$-aggregation of the admissibilities of a sequence of simpler editing operations. The following properties are of particular importance to pattern matching.

If A, B, and C are subsets of a domain $\mathbf{X}$, then

1. If $B \subseteq A$, then $\mathbf{I}(A \mid B) = 1$.
2. $\mathbf{I}(A \mid B) \geq \mathbf{I}(A \mid C) \circledast \mathbf{I}(C \mid B)$.
3. $\mathbf{I}(A \mid B) = \max_{C} [\mathbf{I}(A \mid C) \circledast \mathbf{I}(C \mid B)]$.

In what follows, we will need to consider situations where single objects are either added to or deleted from to an existing set A to produce a transformed set A'. In such cases, we will need to compute the degree of implication $\mathbf{I}(A \mid A')$ as the measure of the admissibility of the simple operation changing A into A'. In those cases, it is straightforward to see that

1. If a member o of A is deleted to produce A', that is, if $A = A' \cup \{o\}$, then $\mathbf{I}(A \mid A') = 1$.
2. If a member o of $\mathbf{X}$ is added to A to produce A', that is, if $A' = A \cup \{o\}$, then $\mathbf{I}(A \mid A') = \max_{o' \in A} S(o, o')$, where S is the similarity function underlying $\mathbf{I}$.

IV. DEGREE OF MATCHING

We present now the basic elements that permit the estimation of the degree of matching between a pattern and a database. Our discussion starts with a description of the basic constructs required to characterize a database as a collection of assertions about binary relations between objects or between objects and their properties. On the basis of that characterization, we present a formal definition of a *Database* as a set of instances of binary predicates. The database may be interpreted, in logical terms, as the conjunction of that set of predicate instances.

We proceed then to sketch the general characteristics of our editing approach by means of definitions of the degree of admissibility of sequences of basic database-editing operations and that of degree of matching. Finally, we present results characterizing the degree of admissibility of basic database-editing operations in terms of related similarity functions.

We have previously remarked, in Section II, that databases may be represented by sets of triples describing links between various instances of database objects. In the rest of this paper, we will distinguish between triples that specify the value of an attribute of an object instance (e.g., to represent that "the age of Person-1 is 25 years")

`(predicate, object, value)`

and those that relate two object-instances (e.g., to represent that "the father of Person-83 is Person-44")

`(predicate, object`$_1$`, object`$_2$`)`.

Following OpenCyc [16] nomenclature, we will refer to these classes of structures as being either *binary attribute predicates* or *binary object predicates*, respectively (or *attribute predicates* and *object predicates*, for short). In what follows, and in deference to terminology employed in the graph-theoretical and database literature, we will also refer to specific instances of object and attribute predicates as *links*.

V. DATABASES

Having introduced the required conceptual structures, we are now in a position to provide a formal definition of a database:

Definition 1: A database is a 4-tuple

$$\mathcal{D} = (\,\mathsf{Obj}, \mathsf{Vals}, \mathsf{Pred}, \mathsf{Data})\,,$$

where

(i) *Objects*: Obj is a nonempty set, called the set of *objects*. It is often the case that objects are structured by means of knowledge structures, such as ontologies, describing set-inclusion and subsumption relations between objects as well as the conditions upon object properties that make possible the differentiation of subclasses (e.g., the property of `Animals` that makes `Vertebrates` different from `Invertebrates`).

(ii) *Values*: Values are nonempty sets in a collection of basic primitive types Vals, such as numbers, strings, or members of predefined discrete sets such as
$\texttt{BalkanCountries} = \{\texttt{Albania}, \texttt{Bosnia}, \ldots\}$
that permit specification of the values of properties and attributes of objects.

(iii) *Predicates*: Pred is a nonempty set, called the set of *predicates*, or the set of *links*.
Links may also be thought of as being relations defined between objects in some domain and the set of possible values of some *property* or *attribute*. Members (in the set-theoretic sense) of such a relation are predicate, or link, *instances*, being sometimes also called *relationships*.
Since predicates are typically members of some domain that is structured by knowledge constructs such as ontologies, we will assume that, in general, there exists a similarity function $\mathsf{Sim}^L_{\mathcal{P}}$ defined between pairs of predicates.[1]

(iv) *Predicate Classes and Instances*: As discussed earlier, we will need to consider two classes of predicates, called *object predicates* and *attribute predicates*, respectively.
We will assume that the sets $\mathcal{P}_O$ and $\mathcal{P}_A$ of object- and attribute-predicate instances, respectively, have a metric structure defined by means of similarity functions $\mathsf{Sim}^O_{\mathcal{P}}$ and $\mathsf{Sim}^A_{\mathcal{P}}$, respectively. We discuss below an approach to the definition of such similarity measures.

(v) *Data*: The data Data is a tuple $(\mathsf{Data}_{\mathcal{O}}, \mathsf{Data}_{\mathcal{A}})$, where $\mathsf{Data}_{\mathcal{O}}$ is a set of object-predicate instances and $\mathsf{Data}_{\mathcal{A}}$ is a *nonempty* set of attribute-predicate instances.

This formal definition simply says that the database is a collection of triples relating objects and a nonempty collection of triples specifying the values of properties and attributes of objects. We require the latter collection to be nonempty to assure that the information represented in the database is grounded on objects that are described, to some extent, by their properties.

[1] This function defined between predicates should not be confused with similar metrics, discussed below, between pairs of predicate *instances*.

VI. SIMILARITIES BETWEEN PREDICATE INSTANCES

The definition of a metric structure in the sets $\mathcal{P}_O$ and $\mathcal{P}_A$ of predicate instances is an essential element of our approach since it permits definition, on a rational semantic basis, of the admissibility functions $\mathsf{Ad}^O_{\mathcal{P}}$ and $\mathsf{Ad}^A_{\mathcal{P}}$ that measure the adequacy of basic database editing transformations.

We have shown that knowledge structures such as ontologies permit the definition of similarity measures in various domains. Our approach will not place any constraints on the nature of the similarity measures $\mathsf{Sim}^O_{\mathcal{P}}$ and $\mathsf{Sim}^A_{\mathcal{P}}$ between predicate instances other than assuming that there are similarity functions defined within object instances (on the basis of their relationships and values of their properties) and between specific property values.

Among the many possibilities open for such definitions, however, there is a simple choice that defines similarity between triples as the composition of the similarities between each of the triple components. A straightforward definition of the similarity between triples in terms of simpler measures defined over their three components, however, is not possible as, usually, these measures depend on the nature of the predicate being considered. A similarity function defined over the set of integer numbers may be employed to compare person ages (measured in years), while a different metric may be required to compare their weights (expressed in kilograms). Although both similarity functions compare elements of the same primitive set (i.e., integers), it is clear that their definition is strongly dependent on the nature of the predicates involved (i.e., `hasAge` and `hasWeight`, respectively).

To arrive at a simple suitable formulation that properly addresses this problem, consider first the problem of defining the similarity of two attribute-predicate instances (l, o, v) and (l, o', v') having the same first component, that is, the same attribute predicate.

Assuming that, for every attribute predicate l, there exists

1. a similarity function $\mathsf{Sim}^l_{\mathcal{O}}$ defined between pairs of objects in Obj (i.e., a way to compare possible replacements of the second component of the tuple)
2. a similarity function $\mathsf{Sim}^l_{\mathcal{A}}$ defined between pairs of attributes each lying in some (usually, but not necessarily, the same) primitive value set contained in the collection Vals (i.e., a way to compare possible replacements of the third component of the tuple)

we can now define a similarity function in the set $\mathcal{P}^l_A = \{(l, o, v)\colon o \text{ is an object, } v \text{ is a value}\}$

$$\mathsf{Sim}^A_{\mathcal{P}}\big(\,(l, o, v),\,(l, o', v')\big) = \mathsf{Sim}^l_{\mathcal{O}}(o, o') \circledast \mathsf{Sim}^l_{\mathcal{A}}(v, v')\,,$$

where $\circledast$ is a T-norm such that $\mathsf{Sim}^l_{\mathcal{O}}$ and $\mathsf{Sim}^l_{\mathcal{A}}$ are $\circledast$-transitive for all links l in Pred.

Trying to extend this definition to the case where it is necessary to measure the resemblance between attribute-predicate instances (l, o, v) and $(l'o', v')$, involving different predicates l and l', we resort to the $\circledast$-transitive similarity function $\mathsf{Sim}^L_{\mathcal{P}}$, introduced earlier, which is defined between pairs of attribute predicates in Pred. Although this function provides the bases for the required extension, the dependence of metrics measuring resemblance between objects and between attribute values on the nature of the attribute predicate l demands that, when comparing (l, o, v) and $(l'o', v')$, we define whether we employ metrics that measure such similitudes either from the viewpoint of l or from that of l'.

While several choices are possible, our preferred approach is to require that, in order for (l, o, v) to be similar to $(l'o', v')$, the following conditions must be met:

1. The predicates l and l' should be similar.
2. The objects o and o' should be similar from the viewpoint of the predicate l *and* from the viewpoint of the predicate l'.
3. The attribute values v and v' should be similar from the viewpoint of the predicate l *and* from the viewpoint of the predicate l'.

To meet the desired requirement that the objects o and o' be similar from the viewpoint of the predicates l and l' (similarly with the attribute values v and v'), we need to define a function that combines the numeric distinctions made by both $\mathsf{Sim}^l_{\mathcal{O}}$ and $\mathsf{Sim}^{l'}_{\mathcal{O}}$ (similarly, $\mathsf{Sim}^l_{\mathcal{A}}$ and $\mathsf{Sim}^{l'}_{\mathcal{A}}$). The following result, stated without proof, permits us to define the required similarity function:

Theorem: Let S_1 and S_2 be $\circledast$-transitive similarity functions between objects in a domain **X**. The function $S_{12} = \min(S_1, S_2)$ is the largest $\circledast$-transitive similarity function such that $S_{12} \leq S_1$ and $S_{12} \leq S_2$

This result allows us to define a similarity function that measures the required distinctions from the criteria imposed by both attribute predicates l and l':

$$\mathsf{Sim}^{(l,l')}_{\mathcal{A}} = \min\left(\mathsf{Sim}^l_{\mathcal{A}}, \mathsf{Sim}^{l'}_{\mathcal{A}}\right),$$
$$\mathsf{Sim}^{(l,l')}_{\mathcal{O}} = \min\left(\mathsf{Sim}^l_{\mathcal{O}}, \mathsf{Sim}^{l'}_{\mathcal{O}}\right).$$

From these definitions we can now define a metric between pairs of attribute-predicate instances as

$$\mathsf{Sim}^{\mathcal{A}}_{\mathcal{P}}\big((l, o, v),\ (l', o', v')\big) = \\ = \mathsf{Sim}^L_{\mathcal{P}}(l, l') \circledast \mathsf{Sim}^{(l,l')}_{\mathcal{O}}(o, o') \circledast \mathsf{Sim}^{(l,l')}_{\mathcal{A}}(v, v').$$

The definition of a similarity function between object-predicate instances (l, o_1, o_2) and (l', o'_1, o'_2) on the basis of

1. a $\circledast$-similarity function $\mathsf{Sim}^L_{\mathcal{P}}$ defined between pairs of object predicates in Pred
2. a $\circledast$- similarity function $\mathsf{Sim}^l_{\mathcal{O}}$ defined, for every object predicate l in Pred, between pairs of objects in Obj (i.e., a way to compare possible replacements of the second component of the triple)
3. a $\circledast$- similarity function $\widehat{\mathsf{Sim}}^l_{\mathcal{O}}$ defined, for every object predicate l in Pred, between pairs of objects in Obj (i.e., a way to compare possible replacements of the third component of the triple)

is given by

$$\mathsf{Sim}^O_{\mathcal{P}}\big((l, o_1, o_2),\ (l', o'_1, o'_2)\big) = \\ = \mathsf{Sim}^L_{\mathcal{P}}(l, l') \circledast \mathsf{Sim}^{(l,l')}_{\mathcal{O}}(o_1, o'_1) \circledast \widehat{\mathsf{Sim}}^{(l,l')}_{\mathcal{O}}(o_2, o'_2),$$

where

$$\mathsf{Sim}^{(l,l')}_{\mathcal{O}} = \min\left(\mathsf{Sim}^l_{\mathcal{O}}, \mathsf{Sim}^{l'}_{\mathcal{O}}\right),$$
$$\widehat{\mathsf{Sim}}^{(l,l')}_{\mathcal{O}} = \min\left(\widehat{\mathsf{Sim}}^l_{\mathcal{O}}, \widehat{\mathsf{Sim}}^{l'}_{\mathcal{O}}\right).$$

To complete our definition of similarity between predicate instances, we will assume that, having usually rather different meanings, the similarity between any object-predicate instance and any attribute-predicate instance is zero.

VII. DATABASE EDITING

We can now propose a database-editing methodology to compute the degree of matching between a database and an instantiation of a pattern. Each of the three basic database-editing operations

1. Deletion of binary-predicate instances
2. Addition of binary-predicate instances
3. Modification of binary-predicate instances

transforms a database $\mathcal{D}$ into a modified database $\mathcal{D}'$. If, as discussed earlier, databases are thought of as sets of predicate instances, it is reasonable to measure the degree of adequacy of any transformation employing a measure, based on the underlying metric structures, that gauges the extent to which the knowledge expressed by $\mathcal{D}'$ is consistent with that expressed by $\mathcal{D}$. Such a measure is provided by the degree of implication **I**:

Definition 2: The degree of admissibility of a basic transformation changing a database $\mathcal{D}$ into a database $\mathcal{D}'$ is the degree of inclusion of $\mathcal{D}'$ in $\mathcal{D}$, that is, $\mathbf{I}(\mathcal{D}|\mathcal{D}')$.

We will consider sequences of transformations of the triples in a database that progressively transform the database into a modified, edited, database that matches the pattern. The degree of admissibility of a sequence of basic database-editing transformations

$$T = (E_1, E_2, \ldots, E_n)$$

is defined as the composition of the degrees of admissibilities of the component editing transformations, that is,

$$\mathsf{Ad}(T) = \mathsf{Ad}(E_1) \circledast \mathsf{Ad}(E_2) \circledast \ldots \circledast \mathsf{Ad}(E_n).$$

The validity of this aggregation is assured by the transitivity of the degree of inclusion $\mathbf{I}$ [11].

Several transformations, or sequences of database-editing operations, may result in the transformation of a database $\mathcal{D}$ into the same transformed database $\mathcal{D}'$. We may think of each such sequence as a path in a database space between the original and the transformed database. Each path accomplishing the same transformation has an associated admissibility measure that is a function of the admissibility of individual edits. From this perspective, it makes sense to measure the admissibility of a transformation in terms of the path having maximum admissibility.

Definition 3: The **degree of matching** between two databases $\mathcal{D}$ and $\mathcal{D}'$ is the admissibility of the sequence of transformations T mapping $\mathcal{D}$ into $\mathcal{D}'$ having maximum admissibility.

It is important to note that, unlike classical similarity and distance metrics, the degree-of-matching function defined above will not be, in general, a symmetric function of its two arguments. The major reason for this lack of symmetry lies in the different cost associated with editing operations that are the inverse of each other (e.g., the cost of adding a predicate instance to a database $\mathcal{D}$ is not the same as that of deleting that link from $\mathcal{D}'$).

VIII. ADMISSIBILITY OF BASIC EDIT OPERATIONS

In our formulation, the value of admissibility measures for basic database-editing operations depends on the nature of predicate instances being modified. Whenever needed to introduce new triples, however, new unlinked objects will be added to the database at no cost (i.e., addition of new, unrelated, internal object representations does not entail introduction of unavailable knowledge).

A. Addition of binary-predicate instances

The addition of triples of the form

$$\left(\texttt{predicate}, \texttt{object}_1, \texttt{object}_2\right)$$

results in the replacement of a database $\mathcal{D}$ by the modified database $\mathcal{D}' = \mathcal{D} \cup \{t\}$. As we have already seen, the degree of admissibility of this editing transformation E is given by

$$\mathsf{Ad}(E) = \mathbf{I}(\mathcal{D}|\mathcal{D} \cup \{t\}) = \max_{t' \in \mathcal{D}} \mathsf{Sim}^{O}_{\mathcal{P}}(t, t') \,.$$

A similar argument leads to the definition of the admissibility $\mathsf{Ad}^2_{\mathcal{P}}(T)$ of the transformation that adds an attribute-predicate instance t to the database $\mathcal{D}$ as

$$\mathsf{Ad}(E) = \mathbf{I}(\mathcal{D}|\mathcal{D} \cup \{t\}) = \max_{t' \in \mathcal{D}} \mathsf{Sim}^{A}_{\mathcal{P}}(t, t') \,.$$

B. Deletion of binary-predicate instances

The deletion of a predicate instance from a database $\mathcal{D}$ results in a database $\mathcal{D}'$ that is a subset of the transformed database.

These transformations are fully admissible, that is,

$$\mathsf{Ad}(E) = \mathbf{I}(\mathcal{D}' \cup \{t\}|\mathcal{D}') = 1 \,,$$

since there should not be any cost associated with disregarding information that facilitates the matching between database and pattern. On the other hand, the inverse operation—adding a predicate instance to the database—entails, as we have seen above, the introduction of information that is not supported by evidence, and its admissibility is measured by the extent to which the assumed information resembles that in the database.

C. Replacement of binary-predicate instances

It is straightforward to prove that the admissibility of a basic editing transformation E of replacing an object-predicate instance t by another object-predicate instance t' is given by the expression

$$\mathsf{Ad}(E) = \max_{''\in \mathcal{D}} \mathsf{Sim}^{O}_{\mathcal{P}}(t', '') \,.$$

This result, which is consistent with our previous estimates of the admissibility of addition and deletion of triples, shows that the cost associated with the replacement of a triple t by a triple t' is equivalent to the cost of adding t' composed with the cost of deleting t. Since, as we have seen, there is no cost associated with deletions, the cost of replacement is, therefore, simply that of adding the new triple.

A similar result holds for replacement of attribute-predicate instances, that is,

$$\mathsf{Ad}(E) = \max_{t'' \in \mathcal{D}} \mathsf{Sim}^{A}_{\mathcal{P}}(t', t'') \,.$$

IX. IMPRECISION, UNCERTAINTY, VAGUENESS

Our discussion has focused, so far, on the nature of databases that are conventional in the sense that, whenever the value of the attribute of an object is specified, such a value is a unique element of the range of possible values of the attribute. Similarly, if two objects are linked, there is no ambiguity as to their identity. In many real-world applications, however, knowledge about the world may not exhibit such desirable characteristics.

The assertions contained in the database may be, for example, imprecise in the sense that they do not permit identification of the value of an attribute or that of the object instances related by a link. Imprecise information permits, however, identification of a set of possible attribute values or a set of objects where the actual attribute value or the related object lies, respectively. The following assertions exemplify imprecise knowledge:

"The `age` of Person `P-3` is at least `20 years`"
"The `father` of Person `P-99` is either Person `P-100` or person `P-102`."

The key feature of imprecision is the inability to specify actual values of attributes or to permit unique identification of objects, allowing, rather, identification of a subset of the range of a particular attribute or relation.

In some applications, the nature of the links that define the property being described or the nature of the relation between two objects may be imprecise. It may only be known, for example, that

"Person `P-46` is a `Relative` of Person `P-3`"

while better knowledge may reveal that `P-46` is the `Father` of `P-3`, that is, a more precise characterization of a link between the object instances representing those persons.

Sometimes the available information is *uncertain*, providing only probabilistic knowledge about the value of an attribute or about the identity of linked objects, as exemplified by

"The probability that the `rainfall` will be `50in` is 50%,"
"The probability that the `father` of `P-90` is `P-3` is 90%,"

defining elements of a probability distribution in the range of a property or relation (i.e., the conditional probability that a property has a value, or the conditional probability that an object is related to another object, *given available evidence*). The basic difference between imprecise and uncertain knowledge is that the former simply specifies a subset of possible values or related objects, while the latter fully specifies a probability distribution over the range of the property or relation.

Our previous developments were based in the representation of a database as a set of assertions, which, in some cases, may be incomplete in the sense that the value of an attribute may be completely unknown (e.g., the `Age` of Person `P-3` is unknown), or there is no information about links between object instances that actually exist in their modeled, real-world, counterparts (e.g., the person modeled in the database as `P-3` is, in the real world, the father of the person represented by `P-90` but this fact is unknown and, therefore, the two object instances are not linked in the database).

The extension of our editing scheme, beyond this form of incompleteness, to imprecise information having a desirable level of generality, requires the introduction of several enhanced data structures including

1. Structures to define sets of objects by enumeration of their members, i.e.,
   $\texttt{Object-Set} = \{ \texttt{Object}_1, \ldots, \texttt{Object}_n \}$
2. Structures that permit the direct representation of sets of possible values, i.e.,
   $\texttt{Value-Set} = \{ \texttt{Value}_1, \ldots, \texttt{Value}_n \}$
   or taxonomical structures defined in value sets permitting the specification of special subsets of values (e.g., tt European Countries)
3. Structures to define imprecise predicates and to organize them along subsumption relations (i.e., ontologies of relations)
4. Generalized predicate-instances permitting the representation of ignorance about the nature of linked objects or the value of attributes, respectively,
   `(Predicate, obj-instance, Object-Set)`
   `(Predicate, obj-instance, Value-Set)`

On the bases of these generalized structures it is possible to extend our database-editing approach to general imprecise databases [17]. A full description of this generalization, including the definition of the admissibility of editing operations that modify these generalized triples, is outside of the scope of this paper.

ACKNOWLEDGMENTS

This work was supported by the United States Air Force, Air Force Research Laboratory, under Contract No. F30602-01-C-0193. The authors benefitted from insightful exchanges and conversations with Pauline Berry, David Israel, Ian Harrison, John Lowrance, David Morley, Andrés Rodriguez, and Lotfi Zadeh. To all of them, many thanks.

REFERENCES

[1] A. Colmerauer, "An introduction to Prolog-III," *Communications of the ACM*, vol. 330, pp. 69–90, 1990.
[2] D. Dubois and H. Prade, "An introduction to possibilistic and fuzzy logics," in *Non-Standard Logics for Automated Reasoning*, D. Dubois, H. Prade, and P. Smets, Eds. Academic Press, 1988.
[3] E. H. Ruspini and F. Esteva, "Interpretations of fuzzy sets," in *The Handbook of Fuzzy Computation*, E. H. Ruspini, P. P. Bonissone, and W. Pedrycz, Eds. Institute of Physics, 1998.
[4] L. Shapiro and R. Haralick, "Structural descriptions and inexact matching," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 3, pp. 504–519, 1981.
[5] H. Bunke, "On a relation between graph edit distance and maximum common subgraph," *Pattern Recognition Letters*, vol. 18, pp. 689–694, 1997.
[6] M. Wolverton, "Retrieving semantically distant analogies," Ph.D. dissertation, Stanford University, 1994.
[7] C. J. Date, *Introduction to Database Systems*, 6th ed. Addison-Wesley, 1995.
[8] P.-S. Chen, "The entity-relationship model – toward a unified view of data," *IACM Transactions on Database Systems*, pp. 9–36, 1976.
[9] J. Cowan, Ed., *Extensible Markup Language (XML) 1.1, Candidate Recommendation 15 October 2002*, 2002. [Online]. Available: http://www.w3.org/TR/xml11/
[10] F. Manola and E. Miller, Eds., *RDF Primer*, 2004. [Online]. Available: http://www.w3.org/TR/rdf-primer/
[11] E. H. Ruspini, "On the semantics of fuzzy logic," *Int. J. of Approximate Reasoning*, vol. 5, pp. 45–88, 1991.
[12] E. H. Ruspini, P. P. Bonissone, and W. Pedrycz, Eds., *The Handbook of Fuzzy Computation*. Institute of Physics, 1998.
[13] E. H. Ruspini and J. Lowrance, "Semantic indexing and relevance measures in SEAS," 2002, working Paper (DARPA GENOA Project).
[14] A. Budanitsky and G. Hirst, "Semantic distance in WordNet: An experimental, application-oriented evaluation of five measures," in *Workshop on WordNet and Other Lexical Resources, Second meeting of the North American Chapter of the Association for Computational Linguistics*, Pittsburgh, PA, 2001.
[15] E. H. Ruspini, "On truth and utility," in *Symbolic and Quantitative Approaches for Uncertainty: Proceedings of the European Conference ECSQAU, Marseille, France, October 1991*, R. Kruse and P. Siegel, Eds. Berlin: Springer-Verlag, 1991, pp. 297–304.
[16] Cycorp, "Opencyc documentation." [Online]. Available: http://www.opencyc.org/doc/
[17] E. H. Ruspini, J. Thomere, and M. J. Wolverton, "Pattern-matching metrics for imprecise databases," 2004.