

Spatial Similarity Queries with Logical Operators

Konstantinos A. Nedas and Max J. Egenhofer

National Center for Geographic Information and Analysis
Department of Spatial Information Science and Engineering
University of Maine, Orono, ME 04469-5711, USA
{kostas, max}@spatial.maine.edu

Abstract. Traditional spatial querying assumes that a user specifies exactly the constraints of valid results, and that the result set contains only those items that fulfill exactly the query constraints. The nature of spatial data, however, makes it difficult for a user to always guess correctly the values stored, while exhaustive enumerations of acceptable alternatives to the ideal target would become a tedious process. Likewise, values that deviate somewhat from the query constraints should be part of a ranked result set as well. This paper develops methods for the retrieval of similar spatial information, in particular when several similarity constraints with logical combinations must be compared and integrated. The first set of methods is concerned with spatial similarity reasoning over null values by denoting the semantics of different types of null values with explicit identifiers that imply different degrees of uncertainty. The second set of methods contributes to a consistent and comprehensive methodology for spatial similarity retrieval in response to complex queries with combinations of logical operators. We provide an exhaustive list of spatial query scenarios with conjunctions, disjunctions, and negation and present justified solutions for each case. Since these computational methods are founded on fundamental psychological knowledge about similarity reasoning, they are particularly well suited for generating similarity results that match with users' intuitions.

1 Introduction

Today's spatial database systems are built on exact matches for spatial information retrieval. Geographic information systems (GISs) and multimedia databases, however, require a different paradigm for spatial information retrieval, emphasizing similarity over equality, mainly for three reasons:

- The data provider-data user gap is wide due to the differences between the stored spatial data and the user's knowledge of these spatial data while querying. People may know only approximately what they are looking for, so that they need to adopt an exploratory way of accessing spatial data. For instance, when querying an archive of satellite imagery based on such properties as color or texture, an exact match is rarely expected.
- The *spatial-intuition gulf* between people who request spatial information and the models in spatial information systems becomes more apparent as spatial

information systems are growing beyond the state of being tools of experts, as a wider and more diversified audience uses them on a daily basis. It is inconceivable that GIS users share a common context and views about reality. The lack of standard, cognitively-plausible formalizations of spatial properties of geographic phenomena makes it even harder to support comprehensive, yet flexible methods for spatial information retrieval.

- The *verbal-visual competition* of requesting spatial information verbally while presenting spatial query results graphically puts an undue cognitive load on users. Thinking spatially is supported only in a very limited way at the query-formulation stage, but alternative visual query modalities, such as sketching, often come closer to a user's mental model of a spatial query than a verbal expression. By their very nature, however, such visual requests for spatial information retrieval are imprecise.

This diversity of background and expertise, combined with ill-defined spatial standards, explains why users' spatial queries often fail to coincide with any stored data.

A successful similarity model for GISs would help eliminate the restrictions imposed by exact matches, thereby providing satisfactory reasoning mechanisms for semantically similar results. Satisfactory results imply a match of methods for spatial similarity retrieval with human perception and cognition. The major obstacle to this goal is the elusiveness and complexity of similarity, which is difficult to describe by formal logical theories or represent with mathematical models; therefore, the focus should be on providing reliable similarity measures that are consistent with people's intuition, rather than conveniently conforming with theories that may have appealing mathematical properties, but contradict human similarity reasoning. This is the main view advocated in this paper. Our approach builds on important findings from psychology, where human similarity has been extensively investigated for decades.

Determining similarity in GISs usually leads to a dichotomy of similarity measures due to the character of spatial entity instances. Objects that represent those instances in a GIS are usually presented as a 3-tuple {thematic attributes, geometric attributes, ID} [1]. Geometric attributes are associated with an object's topology and metric details, whereas thematic attributes capture spatial but non-geometric information. For example, an island as a spatial object may have a name and a population as its thematic attributes, while a shape description provides the value for its geometric attribute. Because of this duality, methods that assess similarity among spatial objects can be divided to separate procedures for geometric similarity assessment and thematic similarity assessment. The overall spatial similarity of two spatial objects is a combination of their geometric and thematic similarity values. Spatial information systems that address objects explicitly have a third dimension of similarity that corresponds to the classes to which the entities belong [2]. Therefore, a formal model for similarity in spatial information systems needs to distinguish three levels of similarity: (1) semantic similarity among entity classes, (2) geometric similarity among entity instances, and (3) thematic similarity among entity instances.

Information systems that record spatial properties explicitly store information about shape and spatial relations as qualitative values. In such a setting, spatial similarity information that is typically derived from the geometric representations is determined much like thematic similarity. For example, rather than storing the

geometry of Greek islands and their settlements in the form of a topological data model, spatial relations among islands and between islands and settlements are captured explicitly (e.g., <Crete, disjoint, Thira>, <Thira, disjoint, Karpathos>, <Iraklio, inside Crete>, <Oia, inside, Thira>). Thematic information of such spatial entities is then available within the same framework (e.g., <Crete, populationPerSquareKilometer, 64>, <Thira, populationPerSquareKilometer, 92>) so that a single approach for assessing spatial similarity suffices. The scope of this paper is such an integrated view of spatial similarity.

This paper addresses spatial similarity assessment for queries with multiple constraints. In such scenarios, a query processor must integrate similarity values over Boolean operations, including spatial similarity assessments that involve null values. We develop a set of methods that aim at providing a consistent and comprehensive methodology for spatial similarity retrieval in response to complex queries formed by combinations of logical or relational operators. Logical operators combine separate spatial constraints using such logical connectives as *and*, *or*, and *not*, whereas relational operators refer to such predicates as *greater than* or *less than*. In support of such queries, we also develop a set of methods for spatial similarity reasoning over null values by denoting the semantics of different types of unavailable values with explicit identifiers that imply different degrees of uncertainty. The paper investigates spatial similarity from a conceptual rather than implementation point of view. Issues pertaining to efficiency or optimization of the algorithms as well as to details of lower-level access to the database are beyond the scope of this work.

The remainder of the paper is organized as follows: section 2 provides background information on similarity measures, null values, and complex similarity queries. Section 3 argues for the need of different similarity methods depending on the attributes' types, such as nominal, ordinal, interval, ratio, and cyclic. Problems that arise when null values are part of similarity assessment are addressed in section 4, where we develop a model for null-value similarity. Cognitively plausible methods for similarity assessments with relational and logical operators are addressed in sections 5 and 6, respectively. Conclusions and future work are discussed in section 7.

2 Related Work

2.1 Similarity Measures

Similarity and difference are tightly related concepts [3]. Definitions of *difference* usually coincides with the distance between the representing points of two entity instances in a conceptual space, so that it is equated with a measure of the dissimilarity between the entities. Popular models for similarity assessment distinguish geometric, featural, transformational, and network approaches.

- *Geometric models* [4, 5, 6, 7, 8] view entities as points in an arbitrarily-dimensional space. The distance between the points corresponds to the measure of *dissimilarity*. Similarity is usually derived as some linear, exponential, or Gaussian function of the distance [9, 10]. Geometric models yield a symmetric and transitive similarity relation.

- *Transformation models* are geometric models that rely on a transformational distance that is expressed as the number of operations required to transform one object into the other [11, 12]. Whereas transformational models are especially useful for visual configurations, geometric models apply better to entities that differ along quantitative variables.
- *Featural models* have a qualitative foundation [13]. Rather than estimating similarity as a function of distance, featural models infer directly a similarity measure from common and different features of the entities under comparison. Common features increase similarity, whereas different features decrease it. Since feature-matching is a set-theoretic approach, it is neither dimensional nor metric and, therefore, symmetry and transitivity do not necessarily hold.
- *Network models* provide explicit support for similarity assessments among hierarchically organized concepts. The closer two concepts are in a network, the higher their semantic relation [14]. An intuitive way of evaluating semantic similarity between two concepts in a network is through their semantic distance, which is typically expressed as the shortest path between two nodes in the network and defined as the sum of the number of edges between them [15, 16]

2.2 Null Values

Null values refer to attributes that have no value stored. Database theory recommends the elimination of null values through proper database design and normalization; however, even in the most carefully designed systems null values are often unavoidable due to inability to collect all the required information about an entity, schema restructuring, or tradeoffs between performance and normalization. Null values introduce ambiguities related to the meaning of the missing attribute values [17]. Our concern is to address the implications that derive from null values when such values are encountered during a similarity assessment process.

A simple but rudimentary way of dealing with null values is to assign the value *zero* to the similarity measure between two values when one of them is null [18]. This approach, however, misses the different semantics that a null value may carry—for instance, up to 14 different types in ANSI/SPARC [19]. Only a subset of three different interpretations, however, is vital for a formal treatment with respect to their meaning.

Unknown null values were initially investigated by Codd [20]. An unknown value (*unk*) states that a precise value exists, but is currently missing. This model was later extended to include *non-applicable* types of nulls [21]. A non-applicable null (*dne* for does not exist) means that the value is unavailable because the specific attribute is not applicable for an object.

Zaniolo [22] introduced the *no-information* null (*ni*). A no-information value is more generic, subsuming *unk* and *dne* types of nulls. It states that the value is missing either because it exists but is unknown or because it does not apply for that object. The advantage of this approach is that it is conceptually simpler and allows more efficient evaluation of queries. The tradeoff is that it is less informative and hence may result in loss of potentially useful information, since it is inadequate in expressing the full spectrum of semantic interpretations that null values may have.

Vassiliou [23] is concerned with *unk*, *dne*, and *inconsistent* nulls. The domain of the database is extended to include the values “*missing*,” “*inconsistent*,” and “*non-applicable*,” where *non-applicable* is treated as a regular precise value.

Morrissey [24] provided a comprehensive treatment with the distinction of *unk*, *dne*, *p-domains*, and *p-ranges*. *P-domains* specify that a missing value is one out of a list of values, whereas *p-ranges* state that the missing value is within a particular range. A *p-domain* applies better to attributes with an enumerated domain of finite elements, whereas a *p-range* applies is more suited to attributes whose values vary along a continuum. For instance, a *p-range* of, say, (20-50) means that the precise value is between 20 and 50. In response to a query, one set of objects captures the exact matches, while another set captures objects with one or more null values that could possibly be exact matches. While this approach is concerned with the retrieval of possible exact matches, we are interested in finding similar results that, among others, encompass possible exact matches.

The common denominators of these approaches seem to be *unk*, *dne*, and *ni* nulls and this is the set that we will use in our approach.

2.3 Complex Similarity Queries

Conjunctive queries refer to the combination of constraints using the logical operator *and*, for instance, “Find objects where attribute *A* has value *x* *and* attribute *B* has value *y*.” Similar combinations can be obtained with the use of disjunctions and negation. The evaluation of each constraint yields a separate similarity value, so that the key issue becomes how to combine the similarity values. Two popular approaches are:

- The geometric approach which calculates the *mean* as the average of the similarity values. The use of attribute weights that indicate each attribute’s salience within the overall similarity assessment offers a refinement of this process. While appropriate for conjunctions, there are no corresponding operations that would support disjunctions or negations.
- The fuzzy-logic approach which, for conjunctive queries, resorts to selecting the *minimum* of the similarity values. Under a fuzzy set interpretation, the extent to which an object in a database satisfies a query becomes a matter of degree. The core of such a fuzzy set (i.e., elements with a membership value of 1) comprises the set of all exact matches to a query. The fuzzy sets concepts of intersections, unions, and complements correspond to the three fundamental scoring rules for conjunctions (Equation 1a), disjunctions (Equation 1b), and negations (Equation 1c), respectively [25, 26].

$$\mu_{A \wedge B}(x) = \min\{\mu_A(x), \mu_B(x)\} \quad \text{(1a)}$$

$$\mu_{A \vee B}(x) = \max\{\mu_A(x), \mu_B(x)\} \quad \text{(1b)}$$

$$\mu_{\neg A}(x) = 1 - \mu_A(x) \quad \text{(1c)}$$

In this paper we argue, however, that fuzzy logic is inappropriate for expressing similarity measures, especially for conjunctions and negations. We advocate the use of a weighted mean instead, and provide arguments based on widely accepted psychological principles.

3 Modeling Thematic Spatial Similarity

To process spatial similarity queries, we map similarity into the quantitative realm, enabling an ordering of the quality of matches. The mappings result in a normalized value of 1 representing an exact match and a value of 0 denoting a complete difference (i.e., no similarity). While the standardized result of the mapping provides a unified framework for high-level, abstract similarity operations, the ways in which the attribute values are mapped onto the similarity interval differ widely. Since similarity is not a unitary concept [27, 28], favoring the use of one model over another depends on the specifics of each attribute, because each model carries different innate assumptions and emphasizes different properties of similarity.

To account for the diversity of attribute types, each attribute needs to carry its similarity operation (i.e., an algorithm that is tailored to the attribute type to assess similarity). Generic classes of algorithms, however, may be developed for attributes that behave the same way. An important set of categories is based on the four *scales of measurement* [29], referring to cognitive and structural commonalities that are typically found in capturing data. The basic types of attributes are those whose values denote nominal, ordinal, interval, ratio, and cyclic measurements.

- A *nominal* attribute type describes values that can be distinguished by equality, for example, the names of Greek Islands, such as Thira, Crete, and Karpathos
- An *ordinal* attribute type captures the sequence of the values, without any reference to how big the differences are. An example of an ordinal attribute type is the landmass of Greek islands, such as Crete > Karpathos > Thira. The values themselves are not enough to establish this ordering, however, because in differing contexts, other orderings could apply (e.g., Karpathos > Crete > Thira as the popularity among tourists who prefer undeveloped islands).
- An *interval* attribute type adds to the ordinal data type the information about how far apart the ranked individuals are. For example, the data type *highestPointOfIsland*—Crete 2,453 meters; Karpathos 1,140 meters; and Thira 567 meters—is of type *interval*.
- A *ratio* data type adds an absolute zero to an interval type. An example is island population—540,000 for Crete; 7,000 for Thira; and 4,600 for Karpathos.
- A *cyclic* attribute type orders ordinal, interval, and ratio types such that the last element in a sequence becomes also the first element of the next round. For example, if a new ferry service always went from Sitia on Crete over Pigadia on Karpathos to Anthinos on Thira, and then back to Sitia to start the next round in the same sequence, then this ferry route would be of type *cyclic*.

A particular problem with comparing spatial configurations for similarity appears when the two items to be compared are of different cardinality. For example, one traveler's route (an ordinal value) was Thira < Karpathos < Crete. Two others traveled such that Crete < Karpathos < Thira and Thira < Crete. While the two trips with three island stops can be compared based on the differences of the islands in the sequence—each of Thira and Crete are two steps apart in the sequences—the route with just two stops does not fit into this pattern as it is questionable how to account quantitatively for the missing piece.

4 Null Values in Spatial Similarity Assessments

To handle efficiently the different semantics of nulls, DBMSs must extend the domain of attributes in the system with the codes *unk*, *dne*, and *ni*, rather than using only the generic code *null*. Similarity between a null value and any other value of an attribute A may be derived from Equation 2, where a and b are the respective minimum and maximum values that define the range of A , x_q is the query value, and $Sim_A(x_q, a)$ and $Sim_A(x_q, b)$ are measures of similarity between x_q and a and x_q and b , respectively.

$$\forall x_q \mid (x_q \neq dne, unk, ni) \quad Sim_A(x_q, null) = \begin{cases} \min(Sim_A(x_q, a), Sim_A(x_q, b)) & \text{if } null = unk \\ 0 & \text{if } null = dne \\ 0 & \text{if } null = ni \end{cases} \quad (2)$$

A comparison between a *dne* and a query value is indeterminate, because *dne* does not exist, whereas the query value exists. We choose to view this existence vs. non-existence of a value as the maximum possible dissimilarity and, therefore, assign a similarity measure of 0 to the pair of *dne* and any query value. Unlike *unk* and *ni*, a *dne* mark should always be treated by the database as a precise value, whether it is encountered in a stored object or specified in a query as the desired value to be retrieved. *Unk* and *ni* nulls, on the other hand, are treated as precise values only when a user queries the system by using them. Such queries are meaningful in the sense that the user may be looking for all missing values in the database in order to update them. In this case a symbolic matching is necessary. In all other cases where *unk* and *ni* values are compared with precise query values, they should be treated as placeholders instead and follow the substitutions (Equation 2); here, the matching is semantic, rather than symbolic [21].

The *unk* code represents knowledge that the actual value, although missing, belongs to the set of values that are allowed in the attribute range [31]. Due to uncertainty we assume minimum similarity and, therefore, substitute *unk* with the domain value that maximizes the distance from the query value x_q . In cases of quantitative attributes, this value is logically either the minimum or the maximum as implied by the domain of the attribute or as specified via an explicit constraint. Hence, we only have to evaluate two results. For qualitative attributes the algorithm may perform only when we are dealing with a finite domain of values. In this case, however, all of the values have to be checked in order to choose the one that minimizes similarity. In the case that the user queries specifically for *unk* values, no substitution takes place and *unk* values are the only exact matches, followed by *ni* values.

The *ni* value is a lower level placeholder for either *unk* or *dne* nulls and is the least informative. For the case of a query specifying any value other than *dne* to be retrieved for a particular attribute, we choose the worst-case scenario and let *ni* values be treated as *dne* values and thus assigned zero similarity. However, during output presentation, tuples with *ni* values must be ranked higher than *dne* in terms of similarity, because they leave open the possibility of existence. If the query, however, asks to retrieve specifically the tuples that have a *dne* value for the attribute, then the order is reversed, since *dne* values are exact matches, and *ni* values the next best results, with everything else excluded. In more realistic scenarios that account for the

vast majority of database queries, users will enter precise values, and retrieve similar results, free of nulls.

For example, given the relation in Table 1, for each record, information exists about the type of the accommodation, the category of luxury, the total number of rooms, and the restaurant types that may exist within the establishments. Let the range of possible rooms for accommodations vary from 5 to 70 and explicitly stated so by a constraint. The query for hotels that have 50 rooms and also include a *Greek* restaurant requires similarity assessment with null values. *Dameia Palace* is a good result, because it is a hotel, the value for beds is relatively close to that of the query, and an *Italian* restaurant—also Mediterranean cuisine—exists on its premises. *Caldera Apartments* would be the second best match, followed by *Santorini Palace*. The reason for *Santorini Palace* being ranked so low is because of its *unk* value for rooms. This value will be substituted with number 5, since this is the value in the allowable range for *rooms* that minimizes similarity. If, however, there was a database constraint stating that hotels of category *A* must have between 40 and 70 rooms, then *unk* would be substituted by the number 70, and we would obtain an ordering in which *Santorini Palace* is the most similar result, followed by *Dameia Palace* and then *Caldera Apartments*. *Sun Rocks* is the least similar match, because it is not a hotel and has no restaurants. The similarity between the query value for a *Greek* restaurant and the *dne* value would evaluate to zero.

Table 1. Relation *accomodations* with attributes that include null values

Name	Type	Category	Rooms	RestaurantType
Sun Rocks	Apartments	B	10	<i>dne</i>
Dameia Palace	Hotel	A	70	Italian
Caldera Apartments	Apartments	A	30	Italian
Santorini Palace	Hotel	A	<i>unk</i>	Greek

This approach offers a semantically enhanced and elegant method when dealing with null values, especially when combined with consistency constraints that may be inserted as rules in the database and reduce the uncertainty for certain facts. Specifying the types of null values with different codes allows for more expressive power, both during the modeling of a database, as well as during the retrieval from it. The procedure adopts a pessimistic view when encountering *unk* values, following always a worst-case scenario and substituting *unk* with the most dissimilar value possible. Approaches that are based on probabilities, information content, or entropy [24] do not apply for similarity assessments as they aim at locating probable exact matches. For example, if the values of two tuples in the database are the *p-domains* [Greek, Chinese], [Greek, Italian] and a query asks for a Greek restaurant. Since Italian cuisine is more similar to Greek cuisine than to Chinese, it is logically inferred that the second *p-domain* is always a better similarity match for the query. However, information content or entropy measures would yield equal estimates when assessing the probability of whether these two values are exact matches or not.

5 Similarity Assessments for Queries with Relational Operators

Relational operators extend the concept of an exact match to that of a range match. Besides the equality operator, relational operators determine whether one value is greater or less than another. Specifying queries with relational operators is meaningful only on terms that have a natural order on a scale; therefore, their usage applies to ratio, interval, ordinal, and—in some cases—cyclic attributes.

In a relational query users refer to a range of values, whereas in a query with an equality operator they refer to a single value. Therefore, we are dealing with a query range R_q instead of a query value x_q . The range may be a closed or an open interval. We denote the endpoints of the range with r_1 and r_2 . For instance, in a query of $x \geq 100$, r_1 is the number 100, and r_2 is plus infinity. Similarity between the range R_q specified by the user and any database value x_i of an attribute A is derived by Equation 3, where $Sim_A(r_1, x_i)$ and $Sim_A(r_2, x_i)$ are measures of similarity between r_1 and x_i and r_2 and x_i , respectively. The justification for Equation 3 is that if an attribute value x_i is contained in the range R_q defined by the relational operator, then it is an exact match and, therefore, receives a similarity measure of 1. If x_i is outside of the range, then its similarity is determined by the algorithm chosen for the attribute. Relational operators are typically pertinent only to quantitative attributes where similarity is derived as a function of distance. In order to estimate the distance, we choose from the range of values that constitute exact matches the one that is closer to x_i . This value will logically be either the minimum or the maximum value of the range R_q (i.e., either r_1 or r_2).

$$Sim_A(R_q, x_i) = \begin{cases} \max(Sim_A(r_1, x_i), Sim_A(r_2, x_i)) & \text{if } x_i \notin R_q \\ 1 & \text{if } x_i \in R_q \end{cases} \quad (3)$$

For example, for a query for all buildings in downtown Iraklion, Crete that occupy an area between 40,000 and 60,000 square feet (i.e., $r_1 = 40,000$ and $r_2 = 60,000$), every building whose area is within the specified interval is an exact match. For buildings with an area value x_i outside of the interval, similarity is a function of the distance from x_i to r_1 if x_i is less than 40,000, or from the distance of x_i to r_2 if x_i is greater than 60,000 (Figure 1).

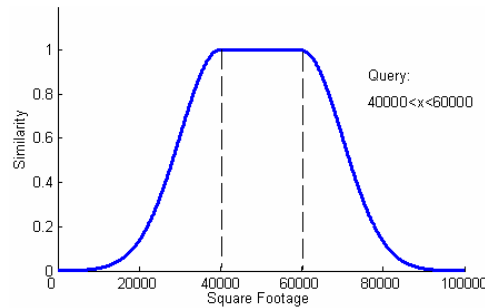


Fig. 1. Similar results to a query involving relational operators

6 Similarity in Spatial Queries Expressed with Logical Operators

Querying a system with logical operators is based on concepts from Boolean algebra. The fundamental logical or Boolean operators include the connectives *and*, *or*, and *not*. This section investigates retrieval of similar results to Boolean queries.

6.1 Queries with *and* on Different Attributes

The use of *and* requires that all the values that it connects be present in the results. Terms joined by the *and*-operator are called conjuncts. In a typical *and*-query the conjuncts are values of two or more different attributes; therefore, the operator *and* is used to allow queries that simultaneously engage multiple attributes of an object. For instance, if A_1 and A_2 are two attributes of a class, the expression $A_1(x)$ *and* $A_2(y)$ means that the user wants to retrieve those objects for which the attribute A_1 has the value x and the attribute A_2 has the value y . For each attribute, similarity between the query value and the stored values is calculated with the algorithm that has been assigned to that attribute. The overall similarity measure between the reference object O_q , characterized by the query values of the user, and any other object (i.e., record) O_i in the database is a weighted sum of the similarity measures obtained for each of the attributes. This measure is captured in Equation 4, where A_1, \dots, A_n are the n attributes connected by the *and*-operator, x_q and x_i are the pair of query and database values for an arbitrary attribute A_k , and Sim_{A_k} is the similarity measure obtained for that pair. The coefficient ω_k represents the weight assigned to attribute A_k .

$$Sim_{A_1, \dots, A_n}(O_q, O_i) = \sum_{k=1}^n \omega_k Sim_{A_k}(x_q, x_i) \quad (4)$$

Consider for instance a user who wants to retrieve islands with a population of 10,000 habitants and an area of 70km². The logical expression for this query is Population(10,000) *and* Area(70). Similarity between the value 10,000 and the values of the stored database objects for the attribute Population is derived from an algorithm appropriate for ratio values that has been assigned to the attribute. The same occurs for the other conjunct about the area. The combined similarity for each database record to the reference query object is a weighted sum for each of the attributes that are linked by the *and*-operator (Table 2). Both islands have the same degree of similarity with respect to their population, but Mykonos is more similar than Santorini to the query value with respect to the area. Hence, we expect that Mykonos will be returned as the most similar result. By using the mean, the combined similarity measure for Santorini is 0.5, whereas for Mykonos it is 0.7.

Table 2. Similar results to a logical and query involving two attributes, both equally weighted

Island	Population	Area (km ²)	Overall Similarity
Mykonos	7,000 (0.5)	75 (0.9)	0.7
Santorini	7,000 (0.5)	90 (0.5)	0.5

The validity of this result is verified by an important finding from psychology, [32, 27, 8, 33] according to which the perceived distance between two stimuli varying along a number of separable dimensions is given by their “city-block” distance metric in the space defined by the dimensions (Equation 5). The term *city-block* means that one travels along the dimensions to get from one point (i.e., stimuli) to the other instead of taking the shortest path (i.e., Euclidean distance). Separable dimensions are those that correspond to highly analyzable and non-related attributes such as length and weight, (or population and area in Table 2). Most databases use attributes that correspond to separable dimensions. The coefficient w_i corresponds to the weight assigned to dimension i . The weights must sum up to one.

$$d_c(x, y) = \sum_{i=1}^n w_i \cdot |x_i - y_i| \quad (5)$$

Equation 4 computes perceived similarity, whereas Equation 5 computes perceived dissimilarity. The two measures add up to 1. Hence, the weighted mean implies that the overall similarity is the inverse of the overall perceived distance according to a measure that has proved to yield consistent results with human reasoning. One may argue that non-linear functions such as Shepard’s [10] exponential formula or Nosofsky’s [9] Gaussian function will give more realistic results. These monotonic functions are indeed preferred during the similarity assessment within individual attributes. At this point of the assessment, however, where similarity measures for each attribute have already been derived and only their integration remains, such functions will just change the similarity scores for each tuple in the set of retrieved similar results, but the ordering from most to less similar object will be preserved. Hence, if the system has a threshold of retrieving 20 similar results, it will retrieve the same set and in the same order, regardless of whether we use the weighted mean or a non-linear monotonic function.

The justification provided clarifies the shortcomings of approaches based on fuzzy logic. The minimum operator that is used in such work [34, 35, 36] is too restrictive, because it only takes into consideration the similarity of one attribute (Equation 1a). For instance, Santorini and Mykonos would both receive an overall similarity value of 0.5 (Table 2). This seems counter-intuitive. Even if one kept adding extra attributes in the database, for all of which the similarity was 0.9 for Mykonos and 0.5 for Santorini, the two islands would still be ranked as equally good results for the query. We do not make the claim here that fuzzy logic is flawed, but rather that the choice of minimum as a fuzzy aggregation operator when reasoning for similarity is erroneous and counter-intuitive. In fact, most of the researchers seem to be somewhat troubled by their results. Santini [37] reports problems between judgments of similarity with his model and others that were experimentally obtained, and admits that the minimum is too restrictive for conjunction. The same is admitted by Ortega *et al.* [26], as well as Fagin [38] who justifies the use of minimum because it has attractive properties that are quite useful in optimizing the algorithms for faster access to the database.

Accuracy and *correctness* of a computer-produced similarity measure and the suitability of an algorithm are only reflected in their fidelity to human behavior, perceptions, and intuition. It makes no sense to succumb to the niceties of a well-defined theory or model when it does not comply with human reasoning, which is the primary objective of all efforts on semantically similar information retrieval.

6.2 Queries with *and* on the Same Attribute

An alternative but rather unorthodox use of *and* occurs when the conjunction is used to connect values of the same attribute. If A is an attribute for a class of objects, the expression $A(x)$ *and* $A(y)$ means that the user wants to retrieve those objects for which the attribute A simultaneously has the values x and y . We are not dealing here with fuzzy variables to which an object may belong simultaneously with different degrees of membership, but rather with multi-valued attributes (i.e., attributes that have a set of values for an entity). Comparing a multi-valued property of two objects requires a different logic than comparing a single-valued property. The similarity measure in this case relates two sets of values, rather than two individual values, and describes how similar one set is to the other.

One approach to finding similarity among multi-valued attributes is to arrange the values of the sets in pairs, and then separately compute the similarity for each pair. The overall similarity of the sets is the maximum possible sum of the similarity measures for each pair, divided by the number of formed pairs. This sum may also be weighted, meaning that the user may specify different weights on different values of the set. The first step of this approach consists of constructing a matrix in which the rows represent the values of one set and the columns the values of the other. The coefficient in a cell of the matrix is the similarity measure for the pair represented in that cell. The second step is to examine the different permutations of pairs that may be constructed and choose the one that gives the maximum sum. For a matrix of order n , the number of possible permutations is $n!$.

An example is the multi-valued property *Color* for buildings. Assume a semantic similarity matrix (Table 3) that stores the similarity coefficients between various colors and create the instances $Building_1_Color(blue, red)$ and $Building_2_Color(blue, black)$. If a user's query is to find a building that is *blue and red*, then $Building_1$ is an exact match. In order to find the similarity of $Building_2$ to $Building_1$ with respect to their colors, we examine the possible combinations of their values (Figure 2). The columns represent the colors of $Building_1$ and the rows the colors of $Building_2$. Each cell stores a similarity coefficient for a pair of colors, as specified in the similarity matrix of Table 3. There are two possible ways to combine the colors of the two buildings. One is to take the sum of the similarity measures, denoted by *italic* numbers, and the other is to take the sum of the similarity measures, denoted by **bold** numbers. In the first case the overall similarity of the sets is the sum of 0.7 and 0.3 divided by the number of formed pairs, which is 2. The result is 0.5. In the latter case the overall similarity is the sum of 1 and 0.2 divided by 2, which yields 0.6. This is the maximum possible sum, therefore, the measure of similarity between the two sets.

Table 3. Similarity matrix for a color attribute

	yellow	red	blue	black
yellow	1	0.5	0.2	0.1
red	0.5	1	0.3	0.2
blue	0.2	0.3	1	0.7
black	0.1	0.2	0.7	1

Our approach to multi-valued attributes captures indirectly a combination of featural and geometric similarity models. Pairs of values that are common in two sets have a similarity coefficient of 1 assigned to them, and thus are likely to be included in the combination of pairs, which yields the maximum sum; therefore, common values are counted as common features and contribute significantly to the overall similarity of the sets compared. For the rest of the pairs, which comprise of different values in each set, we estimate their similarity, rather than adopting the binary logic of featural models that treats them only as distinctive features.

		Building 1	
		blue	red
Building 2 {	blue	1	0.3
	black	0.7	0.2

Fig. 2. Similarity between sets of values for a multi-valued attribute

A problem arises when the sets have different cardinalities. For instance, consider the entity instances $Building_1_Color(yellow, red)$ and $Building_2_Color(yellow, red, blue)$. If we follow the same process, then the combination of pairs that yields the maximum sum is given by the two highlighted cells (Figure 3a) and the similarity between two sets is computed from Equation 6.

$$Sim_{color}(B_1, B_2) = \frac{Sim(yellow, yellow) + Sim(red, red)}{2} = 1 \quad (6)$$

Neglecting the additional color value of $Building_2$ leads to an obviously incorrect result, which presents the two buildings as identical with respect to their colors. What is not represented in this similarity measure is the existence of one additional color for one of the buildings. To rectify this problem we may extend the smaller set in the assessment with *dne* nulls up to the cardinality of the larger set (Figure 3b). The addition of the pair $(blue, dne)$ in the formula that yields similarity for the sets (Equation 7) reflects the existence of one additional color in $Building_2$, and allows us to obtain a similarity result that corresponds better to the real-world situation.

		Building 1				Building 1		
		yellow	red			yellow	red	dne
Building 2 {	yellow	1	0.5	Building 2 {	yellow	1	0.5	0
	red	0.5	1		red	0.5	1	0
	blue	0.2	0.3		blue	0.2	0.3	0

Fig. 3. (a) Similarity between sets of different cardinalities, (b) Extending sets of different cardinalities with *dne* nulls

$$Sim_{color}(B_1, B_2) = \frac{Sim(yellow, yellow) + Sim(red, red) + Sim(blue, dne)}{3} \approx 0.7 . \quad (7)$$

6.3 Queries with *or* on the Same Attribute

The use of *or* requires that at least one of the values that it connects be present in the results. Terms joined by the *or*-operator are called disjuncts. In a typical database *or* query the disjuncts are values of the same attribute. If A is an attribute of a class of objects, the expression $A(x)$ *or* $A(y)$ means that the user wishes to retrieve objects for which the value for attribute A is either x or y . As in the case with relational operators there is not one query value but a set of query values. The difference, however, to queries expressed through relational operators is that the set of query terms is not represented by a range, but by a finite number of distinct values. Similarity is derived from Equation 8, where $Q=[x_1, x_2, \dots, x_n]$ is the set containing the n values connected by the *or*-operator in the query expression, and x_i is any stored value for attribute A in the database.

$$Sim_A(Q, x_i) = \begin{cases} \max(Sim_A(x_1, x_i), Sim_A(x_2, x_i), \dots, Sim_A(x_n, x_i)) & \text{if } x_i \notin Q \\ 1 & \text{if } x_i \in Q \end{cases} \quad (8)$$

Any value x_i contained in Q is an exact match. For values of x_i that do not belong to the set Q , the process consists of examining the similarity between x_i and all the values that belong to Q . The similarity is derived by the algorithm that has been assigned to the attribute A . Since all values in Q are exact matches, we choose the one that gives the largest similarity measure for x_i when compared with it, that is, the similarity of x_i is determined by its distance from the closest exact match. For example, assume a query on a quantitative attribute A asking to retrieve the records where the value of A is 400 or 600. The expression for this query is $A(400)$ *or* $A(600)$. Any object with a value of A that is 400 or 800 is an exact match. For objects that have a different value x_i for A , we choose the maximum similarity measure obtained for the pairs $(400, x_i)$ and $(600, x_i)$ as this is computed from the algorithm that has been assigned to attribute A (Figure 4).

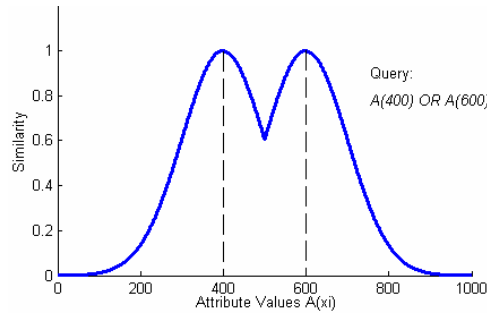


Fig. 4. Similar results to a logical *or*-query involving one attribute

6.4 Queries with *or* on Different Attributes

Specifying queries with *or* where the disjuncts are values of different attributes constitutes uncommon practice, but is still a viable option for the database users. If A_1 and A_2 are two attributes of a class, the expression $A_1(x)$ *or* $A_2(y)$ means that the user wants to retrieve those objects for which the attribute A_1 has the value of x , or those objects for which the attribute A_2 has the value of y . For each attribute, similarity between the query value and the stored database values is calculated with the algorithm that has been assigned to that attribute. If $Q=[A_1(x_q), A_2(x_q), \dots, A_n(x_q)]$ is the set containing n query values (x_q) of n different attributes connected by the *or*-operator, and $A_k(x_q)$ and $A_k(x_i)$ are the pair of query and database values for an attribute A_k with $k=1, \dots, n$, then the similarity between the reference object O_q , characterized by the query values of the user, and any other object (i.e., record) O_i in the database is derived from Equation 9.

$$Sim_{A_k}(O_q, O_i) = \begin{cases} \max(Sim_{A_k}(x_q, x_i)) & \text{if } A_k(x_i) \notin Q \\ 1 & \text{if } A_k(x_i) \in Q \end{cases} \quad (9)$$

A database object that matches any or a number of the values contained in Q for some attributes A_k is an exact match. When none of the object's values $A_k(x_i)$ is identical to the query value $A_k(x_q)$ for an attribute A_k , then we separately examine the similarity of the pairs $(A_k(x_i), A_k(x_q))$ for all attributes A_k connected by the *or*-operator. The similarity measure chosen is the maximum found during this process. For example, consider the query *Building_Type(hospital) or Capacity(150)* (Table 4). Regardless of its value for the attribute *Capacity*, object 1 is an exact match because it matches the query value for the attribute *Building_Type*. Similarly, object 2 is also an exact match because it matches the query value for the attribute *Capacity*. For objects 3 and 4 the similarity between their values and the query value for each of the attributes is calculated separately, and the larger of those measures is the overall similarity assigned to the object with respect to the user's query.

Table 4. Similar results to a logical *or*-query involving two attributes

ID	BuildingType	CapacityInBeds	Overall Similarity
1	hospital	100%	100%
2	clinic	83%	100%
3	health center	75%	75%
4	medical center	50%	50%

6.5 Queries with *not*

Values that the *not*-operator takes as arguments are not present in the results. If A is an attribute for a class of objects, the expression *not* $A(x)$ means that the user wants to retrieve any object, except those that have a value of x for attribute A . The similarity of an arbitrary database object O_i with respect to a query expressed by the *not*-

operator is derived by Equation 10, where x_i represents the stored database value for attribute A of object O_i , and x_q is the value specified by the *not* operator.

$$Sim_A(O_i) = \begin{cases} 0 & \text{if } x_q = x_i \\ 1 & \text{if } x_q \neq x_i \end{cases} \quad (10)$$

Negations are another area where fuzzy based implementations of similarity to complex queries suffer. Equation 1c is used to assess similarity of database objects to the query. Its effect is that it returns as most similar results the objects that are most dissimilar to the value negated in the query. This approach does not scale up well to human reasoning, and might even return paradoxical results. For instance, if a user queries for a hotel but not in the center of a city this does not necessarily mean that she would like to find a hotel in the middle of the desert or on the top of a mountain, while one in the suburbs would be acceptable. Therefore, the role of negations in information retrieval is as a means to avoid undesirable associations, or in general, to eliminate unwanted tuples from the set of retrieved results. Hence, it should be interpreted by a similarity query processor, as it has always been interpreted traditionally in the classic logic paradigm.

The combination of a conjunction and a negation over the same constraint— $A(x)$ and *not* $A(x)$ —can be interpreted as “find the objects that simultaneously have and do not have the value x for attribute A .” Although in classic logic this is a contradiction, in a similarity setting it can be interpreted as a request to retrieve all similar results for a query, excluding those that are exact matches.

7 Conclusions

To address the emerging needs for flexible yet powerful spatial information retrieval, we developed a model for spatial similarity assessment with complex constraints. First, we showed that an enhanced treatment of null values is possible if their different semantics are explicitly represented in the system with different identifiers that imply different degrees of uncertainty. Then we provided a comprehensive framework for dealing with similarity assessments under Boolean and relational operators. We followed widely accepted findings about similarity from the field of psychology.

Current implementations of complex similarity assessments that use fuzzy logic have limitations, especially for conjunctions and negations. Although disjunctions perform realistically with a fuzzy logic interpretation of the *or*-operator, negations require a traditional classic logic interpretation. On the other hand, conjunctions need a compensatory use of the *and*-operator, with all of the individual similarity estimates contributing to the final score. An interesting case of conjunction occurs when the aggregated terms refer to values of the same attribute. Such queries are possible in systems that allow storage of multi-valued attributes. We developed a new set of methods to support them.

Future research will address the determination of weights for conjunctive queries based on context, and the design of appropriate user interfaces for similarity queries. We are also interested in finding optimal threshold specifications for the similar results and efficient ways of presenting them to the users.

8 Acknowledgments

This work was partially supported by the National Science Foundation under grants IIS-9970123 and EPS-9983432 and the National Imagery and Mapping Agency under grant numbers NMA202-97-1-1023 and NMA401-02-1-2009. Max Egenhofer's work is further supported by the National Institute of Environmental Health Sciences, NIH, under grant number 1 R 01 ES09816-01, and the National Imagery and Mapping Agency under grant numbers NMA201-00-1-2009 and NMA201-01-1-2003.

References

1. Bishr, Y. (1998) Overcoming the Semantic and Other Barriers to GIS Interoperability. *International Journal of Geographical Information Science* 12(4): 299-314.
2. Rodriguez, A., Egenhofer, M., and Rugg, R. (1999) Assessing Semantic Similarities Among Geospatial Feature Class Definitions. in: *The 2nd International Conference on Interoperating Geographic Information Systems*, vol. 1580, pp. 189-202, Vckovski, A., Brassel, K., and Schek, H., (Eds.), Zurich, Switzerland.
3. Mill, J. (1829) *Analysis of the Phenomenon of the Human Mind*. vol. 2, Baldwin and Cradock, London.
4. Richardson, M. (1938) Multidimensional Psychophysics. *Psychological Bulletin* 35: 659-660.
5. Torgerson, W. (1952) Multidimensional Scaling: I. Theory and Method. *Psychometrika* 17(4): 401-419.
6. Shepard, R. (1962) The Analysis of Proximities: Multidimensional Scaling with an Unknown Distance Function. I. *Psychometrika* 27(2): 125-140.
7. Shepard, R. (1962) The Analysis of Proximities: Multidimensional Scaling with an Unknown Distance Function. II. *Psychometrika* 27(3): 219-246.
8. Nosofsky, R. (1991) Stimulus Bias, Asymmetric Similarity, and Classification. *Cognitive Psychology* 23: 94-140.
9. Nosofsky, R. (1986) Attention, Similarity, and the Identification-Categorization Relationship. *Journal of Experimental Psychology: General* 115: 39-57.
10. Shepard, R. (1987) Toward a Universal Law of Generalization for Psychological Science. *Journal of Science* 237: 1317-1323.
11. Imai, S. (1977) Pattern Similarity and Cognitive Transformations. *Acta Psychologica* 41: 433-447.
12. Hahn, U. and Chater, N. (1997) Concepts and Similarity. in: *Knowledge, Concepts, and Categories*, Lamberts, L. and Shanks, D., (Eds.), Psychology Press/MIT Press, Hove, U.K.
13. Tversky, A. (1977) Features of Similarity. *Psychological Review* 84(4): 327-352.
14. Quillian, M. (1968) Semantic Memory. in: *Semantic Information Processing*, Minsky, M., (Ed.) MIT Press, Cambridge, MA.
15. Rada, R., Mili, H., Bicknell, E., and Blettner, M. (1989) Development and Application of a Metric on Semantic Nets. *IEEE Transactions on Systems, Man, and Cybernetics* 19(1): 17-30.
16. Budanitsky, A. (1999) *Lexical Semantic Relatedness and its Application in Natural Language Processing*. Computer Systems Research Group, University of Toronto, Toronto, Technical Report CSRG-390.
17. Elmasri, R. and Navathe, S. B. (2000) *Fundamentals of Database Systems*. Addison Wesley Longman Inc.

18. Richter, M. (1992) Classification and Learning of Similarity Measures. in: *Studies in Classification, Data Analysis and Knowledge Organization*, Springer Verlag.
19. Bachman, C., Cohn, L., Florance, W., Kirshenbaum, F., Kuneke, H., Mairet, C., Scott, E., Sibley, E., Smith, D., Steel, T., Turner, J., and Yormark B., (1975) Interim Report: ANSI/X3/SPARC Study Group on Data Base Management Systems. *ACM SIGMOD Record* 7(2): 1-140.
20. Codd, E. (1979) Extending the Database Relational Model to Capture More Meaning. *ACM Transactions on Database Systems* 4(4): 397-434.
21. Codd, E. (1986) Missing Information (Applicable and Inapplicable) in Relational Databases. *ACM SIGMOD Record* 15(4): 53-78.
22. Zaniolo, C. (1982) Database Relations with Null Values. in: *Proceedings of the ACM Symposium on Principles of Database Systems*, pp. 27-33, Los Angeles, CA.
23. Vassiliou, Y. (1979) Null Values in Data Base Management: A Denotational Semantics Approach. in: *Proceedings of the 1979 ACM SIGMOD International Conference on Management of Data*, pp. 162-169, Bernstein, P., (Ed.), Boston, MA.
24. Morrissey, J. (1990) Imprecise Information and Uncertainty in Information Systems. *ACM Transactions on Information Systems* 8(2): 159-180.
25. Santini, S. and Ramesh, J. (1996) Similarity Queries in Image Databases. in: *Proceedings of CVPR '96, International IEEE Computer Vision and Pattern Recognition Conference*.
26. Ortega, M., Rui, Y., Chakrabarti, K., Porkaew, K., Mehrotra, S., and Huang, T. (1998) Supporting Ranked Boolean Similarity Queries in MARS. *IEEE Transactions on Knowledge and Data Engineering* 10(6): 905-925.
27. Torgerson, W. (1965) Multidimensional Scaling of Similarity. *Psychometrika* 30(4): 379-393.
28. Goldstone, R. (1994) The Role of Similarity in Categorization: Providing a Groundwork. *Cognition* 52: 125-157.
29. Stevens, S. (1946) On the Theory of Scales of Measurement. *Journal of Science* 103(2684): 677-680.
30. Chrisman, N. (1995) Beyond Stevens: A Revised Approach to Measurement for Geographic Information. in: *Twelfth International Symposium on Computer-Assisted Cartography, Auto-Carto 12*, Charlotte, NC.
31. Lipski, W. (1979) On Semantic Issues Connected with Incomplete Information Databases. *ACM Transactions on Database Systems*: 262-296.
32. Attneave, F. (1950) Dimensions of Similarity. *American Journal of Psychology* 63: 516-556.
33. Nosofsky, R. (1992) Similarity Scaling and Cognitive Process Models. *Annual Review of Psychology* 43.
34. Santini, S. and Ramesh, J. (1997) The Graphical Specification of Similarity Queries. *Journal of Visual Languages and Computing* 7(4): 403-421.
35. Fagin, R. (1998) Fuzzy Queries in Multimedia Database Systems. in: *Proceedings of the Seventeenth ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems*, Seattle, WA.
36. Ramakrishna, M., Nepal, S., and Srivastava, D. (2002) A Heuristic for Combining Fuzzy Results in Multimedia Databases. in: *Proceedings of the thirteenth Australasian conference on Database technologies*, Melbourne, Victoria, Australia.
37. Santini, S. and Ramesh, J. (2000) Integrated Browsing and Querying for Image Databases. *IEEE Multimedia* 7(3): 26-39.
38. Fagin, R. (1996) Combining Fuzzy Information From Multiple Systems. in: *15th ACM Symposium on Principles of Database Systems*.