A B^+-tree Based Indexing Technique for Necessity Measured Flexible Conditions on Fuzzy Numerical Data

Carlos D. Barranco
Division of Computer Science
School of Engineering
Pablo de Olavide University
Utrera Rd. Km. 1
41013 Sevilla (Spain)
cbarranco@upo.es

Jesús R. Campaña
Dept. of Computer Science
and Artificial Intelligence
University of Granada
Daniel Saucedo Aranda s/n
18071 Granada (Spain)
jesuscg@decsai.ugr.es

Juan M. Medina
Dept. of Computer Science
and Artificial Intelligence
University of Granada
Daniel Saucedo Aranda s/n
18071 Granada (Spain)
medina@decsai.ugr.es

Abstract

The paper proposes an indexing technique for fuzzy numerical data which increases the performance of query processing when the query involves an atomic necessity measured flexible condition. The proposal is based on a classical indexing technique for numerical crisp data, B^+-tree. Its efficiency is contrasted with another indexing method for similar data and queries. Results show that the proposal performance is similar to and more stable than the reference technique.

Keywords: Fuzzy numerical data indexing, Fuzzy databases.

1 Introduction

Database world has taken advantage of fuzzy set theory by using it as a way to manage imprecise, uncertain and inapplicable data (generally called *fuzzy data*) and to model and process flexible queries. As a result of this trend, there is a significant number of proposals on fuzzy database models, flexible querying and implementations of fuzzy database management systems (FDBMS).

Unfortunately, up to now FDBMSs are not generally integrated into real-world environments as the existing implementations do not provide the required performance. In fact, it is not the fault of implementations. The flexibility of fuzzy databases result in a increment of query results and makes the classical indexing techniques, which are the key for high performance in databases, inapplicable. Although a great deal of research has been carried out into fuzzy database models, not so much work has been done into indexing mechanisms for efficiently accessing fuzzy data.

This paper proposes an indexing technique for fuzzy numerical data to improve query processing when a particular kind of monoattribute flexible condition based on as the necessity measure is involved. The data structure and search algorithm of the proposed indexing mechanism is based on classical indexing structures for numerical crisp data, i.e. B^+-trees. This underlying indexing technique is not specifically designed to index fuzzy numerical data but it is a simple and well-optimized technique which is available in virtually every current DBMS. This near-to-general availability of B^+-tree indexing methods would result in a reduction of implementation, integration and optimization efforts to incorporate the proposed indexing technique to an FDBMS built on a crisp DBMS.

The paper is organized as follows. The concept of fuzzy numerical data and necessity measured flexible conditions in the context of the paper are described in Section 2. Section 3 briefly introduces related work on fuzzy data indexing and outlines the indexing principle on which the proposed and studied indexing techniques rely. Section 4 describes the proposed and considered indexing techniques. Section 5 presents the measures and procedures for evaluating the performance of

L. Magdalena, M. Ojeda-Aciego, J.L. Verdegay (eds): Proceedings of IPMU'08, pp. 1717–1724
Torremolinos (Málaga), June 22–27, 2008

the studied indexing techniques. Section 6 analyzes the performance results. Finally, Section 7 contains the concluding remarks and future works.

2 Basic Concepts

This section defines these concepts in the context of the paper.

A *fuzzy numerical value* is a convex possibility distribution on an underlying domain in which a linear order relation is defined.

Also for the purposes of this paper, a *fuzzy condition* is a restriction imposed on the values of an attribute which contains a fuzzy numerical value for each row. This restriction is specified as a fuzzy numerical value to which the restricted attribute value must possibly or necessarily be compatible. This paper only focuses on necessity measured conditions since possibility measured conditions has been previously studied [1].

The necessity degree is called the fulfillment degree of the condition in the rest of the paper. This degree is computed as shown in eq. 1, where $D(A)$ is the underlying domain associated to the fuzzy attribute A, $\Pi_{A(r)}$ is the possibility distribution which describes the fuzzy value of the attribute A for the row r, μ_C is the membership function defining the fuzzy condition C and $\vee$ means for a t-conorm.

$$N(C/r) = \inf_{d \in D(A)} [(1 - \Pi_{A(r)}(d)) \vee \mu_C(d)] \quad (1)$$

A fuzzy condition is combined with a crisp relational comparator to set a threshold for its fulfillment degree. This threshold specifies the degree of flexibility in which the fuzzy condition is applied, from 1 (no flexibility) to 0 (maximum flexibility). The typical expression for applying a threshold T is $N(C/r) \geq T$, except when the threshold is 0. In the latter case, $N(C/r) > 0$ is applied. The combination of a fuzzy condition with a threshold is called an *atomic flexible condition* for the rest of this paper and it is notated $\langle C, T \rangle$.

3 Related Work

The works on fuzzy data indexing started, to the best of our knowledge, with the seminal paper [3]. It exposes the need for specific indexing techniques for fuzzy databases, and proposes two indexing principles for flexible querying using possibility and necessity measures on fuzzy attributes described as possibility distributions. For the sake of brevity this section only describes the indexing principle for necessity measured fuzzy conditions, on which this paper is focused. The reader is referred to [3, 1] for details on possibility measured ones.

The indexing principle allows the rows which do not satisfy an atomic flexible condition to be filtered out of a table. For necessity measured flexible conditions the article [3] proposes the one shown in Eq. 2, where, given a fuzzy set F, $S(F)$ means for the support of F and $S_\lambda(F)$ means for the λ-cut of F. It can be applied by indexing the cores of the indexed data.

$$N(C/r) > \lambda \rightarrow S_1(A(r)) \subset S_\lambda(C) \quad (2)$$

Since the publication of Bosc's seminal paper, some fuzzy data indexing techniques are proposed. The techniques [4, 9] and [10, 11, 12] are not based in the previous indexing principles and are designed for applications where the number of potential flexible conditions which can be used to build queries is finite and low. Whereas, [5] takes advantage of Bosc's indexing principles and supports any query but it is designed for low cardinality fuzzy data types. The limitations of the previous techniques make them unsuitable for indexing fuzzy numerical values.

The paper [8] proposes an indexing technique for convex possibility distributions defined on an ordered domain that relies on a crisp multidimensional indexing method. This technique is suitable for numerical fuzzy data indexing, on which this paper focuses, and it is described in the next section.

4 Considered and Proposed Indexing Techniques

This paper contrasts the performance of two fuzzy data indexing techniques based on the previously described indexing principle. Both tackle the problem of indexing the core of fuzzy data, which in fact is a closed interval, by mapping it to a point in a bidimensional space. This way, for each interval $[l, u]$ a point (l, u) is inserted in the indexing structure.

Given a necessity measured flexible condition $\langle C, T\rangle$, its preselection row set is obtained by retrieving all entries (x, y) in the index satisfying the range query $l_{S_T(C)} \leq x \leq u_{S_T(C)}, l_{S_T(C)} \leq y \leq u_{S_T(C)}$, where $l_{S_T(C)}$ and $u_{S_T(C)}$ are the lower and upper bounds of the interval corresponding with $S_T(C)$. For the rest of the paper $S_T(C)$ is called the *base* of the condition flexible condition $\langle C, T\rangle$. This range condition is a translation of the preselection criteria of eq. 2 to the described two-dimensional mapping scheme.

4.1 Fuzzy Data Indexing with a G-tree

Based in the previous proposal, [8] makes use of a G-tree [6] for indexing the bidimensional points representing the core of the possibility distributions modeling the indexed fuzzy data. A G-tree is a combination of a B^+-tree and a grid file for indexing multidimensional data points. This index structure supports single point queries as well as range queries. For the rest of the paper, this technique is called GT for the sake of conciseness.

4.2 Fuzzy Data Indexing with a B^+-tree

This paper proposes a technique that takes advantage of classical B^+-tree indexing structures [2] for indexing bidimensional points representing the core of the indexed fuzzy numerical data.

B^+-trees are indexing structures designed for one-dimensional data. In order to make them suitable to index multidimensional data some additional work is required. One possible solution is to reduce multidimensional data to a one-dimensional counterpart, and then use a B^+-tree to index it. This reduction can be made with the help of a space-filling curve [7], a kind of curve that induces a linear order for multidimensional spaces. Among the variety of space-filling curves, in this paper Hilbert curve is chosen. The decision is motivated by its good performance contrasted with other popular possibilities.

Processing a multidimensional range query using the combination of a B^+-tree and a space filling-curve is an iterative process that comprises several one-dimensional range queries on the B^+-tree. Each iteration means a one-dimensional range query for each segment of the space-filling curve that cross the multidimensional query region. Figure 1 illustrates this process, where the center of each square forming the gray grid represents one point in a two-dimensional space, the order induced by the Hilbert curve is represented by a continuos black line and the two-dimensional range query is represented by a dotted rectangle. In the figure it can be seen that the Hilbert curve enters and leaves the query area several times. In order to find the indexed points included in the query area it is necessary to perform a one-dimensional range query on the B^+-tree for each curve segment inside the query region. For instance, for the case of the query region shown in the figure it is necessary to perform 4 one-dimensional range queries on the B^+-tree.

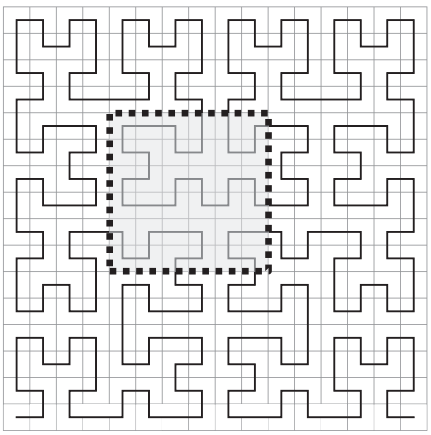

Figure 1: Number of query results and database size correlation

If the described method to solve multidimensional queries is applied directly it is unpredictably inefficient, as the number of one-dimensional queries necessary to perform varies for each multi-dimensional range query and so the index efficiency. This decreases as the number of necessary one-dimensional queries increases. This efficiency decrement is caused because each time a one-dimensional query is performed most of the non-leaf nodes of the B^+-tree must be read again. In order to solve this problem we propose to make use of a non-leaf node cache that will maintain only one node for each non-leaf level of the index. Taking into account that one-dimensional queries are applied following the order induced by the Hilbert curve, there is no need for a bigger cache. For instance, a 10,000,000 elements database of the same characteristics of the ones used in the experiments described later only requieres to maintain 2 non-leaf nodes in the worst case. Throughout the remainder of this paper this proposed indexing technique is called HBPT.

5 Performance evaluation

As it is remarked previously, HBPT is based on an one-dimensional indexing scheme, so this should result in lower performance in contrast with GT, which uses an specific multidimensional scheme. However, HBPT does not suffer from the performance degradation of GT due to low bucket usage caused by its partitioning method, which is extremely sensitive to data distribution. In fact, the stability of bucket usage of HBPT may neutralize the disadvantages due to its non multidimensional specific indexing scheme. This might enable the proposed fuzzy indexing technique to perform as well as GT.

In order to assess HBPT and GT performance, a quantitative performance evaluation has been carried out. This section describes the index performance measure used in this evaluation, the influential factors on index performance which has been taken into account, and the experiment designed to evaluate performance.

5.1 Performance measurement

In order to measure the efficiency of the studied indexes independently of hardware and OS dependent factors [1] the index efficiency measure in equation 3 is used, where d is the minimum number of data blocks in which the result set can be fitted, and i is the number of blocks of index data accessed by the indexing technique (i.e. the index overhead).

$$eff = \frac{d}{d+i} \tag{3}$$

5.2 Influential factors for index performance

Performance of indexing techniques for fuzzy data are affected by a large number of factors. On one hand, a set of physical and logical factors related to the classical indexing techniques on which these indexes for fuzzy data are based must be taken into account. Even though the mentioned factors could increase index efficiency when tuned, they are basically hardware or particular case dependent, so their study does not provide a good insight into the general performance of the considered indexing techniques under general conditions. On the other hand, there is a set of factors related to the indexed data and the processed queries which would affect the index performance. These factors can not be tuned and would provide a good measure of index performance under different usage conditions.

The indexing principle on which both studied techniques are based calculates the preselection set as the set of fuzzy data elements whose core is contained inside the flexible condition base. The extent of these intervals and the amount of indexed data would affect to the number of results for a query. Therefore, this would relativize the index overhead and affect the index efficiency.

Both, the extent of fuzzy data core and the flexible condition base, are dependant of the shape of the fuzzy set modeling the fuzzy data element and flexible condition. This shape can be described as the extent of the support of the fuzzy set and the sharpness of its tran-

sition from its support to its core. The extent of the support means an upper bound for the flexible condition base and also for the fuzzy data core. The sharpness means the speed of reduction of the extent of the base of a flexible condition with respect to its threshold. This highlights that flexible condition threshold is also an influential factor. Additionally, the sharpness combined with the extent of the fuzzy set support determine the extent of the fuzzy set core. In order to generally describe these shape descriptors [1] *imprecision* and *fuzziness* degrees are proposed.

To sum up, the following six influential factors on the fuzzy index performance have been identified: the amount of indexed data, the imprecision and fuzziness of the fuzzy data and flexible queries, and the threshold of flexible queries.

5.3 Experiments

The same experiments using the same set of databases and queries have been conducted on both fuzzy data indexing techniques in order to asses their global performance and their particular performance under different data and query scenarios.

In order to isolate the experiment from physical and logical factors related to the underlying indexing techniques, both share the same fixed values chosen to generate worst case performance measures.

A test have been conducted on different randomly generated databases to evaluate the global performance of the evaluated indexes. The test data set is composed by 30 databases, each one randomly generated using a uniformly distributed random generator within the interval $[-1,000,000\ ,\ 1,000,000]$. The size of these databases has been fixed to 10,000, 20,000, 40,000, 80,000 and 160,000 elements.

The test query set is composed by 10,000 queries. Each flexible condition of the query test set has also been randomly generated using a uniformly distributed random generator within the interval $[-1,000,000\ ,\ 1,000,000]$.

In order to evaluate the index efficiency under different data and query scenarios, the same previous test have been conducted several times (once for a fixed value of each influential factor) on a modified test data set.

The original test data set has been modified by fixing the imprecision (in a first test) and fuzziness (in a second test) degree of the database elements. This way data elements are randomly generated except for the fixed factor.

For the first test, the imprecision degree has been fixed to values ranging from 0 to 0.9 by applying a 0.1 increment (i.e. 0, 0.1, 0.2, ..., 0.9). The imprecision degree 1 is ignored because it means that all the randomly generated data supports are the same as the extent of their support is equivalent to the extent of the underlying domain. A total of 10 test data sets composed by 30 databases (a total 18,600,000 randomly generated data elements) are considered in this test.

The same way, the fuzziness degree has been fixed to values ranging from 0 to 1 by applying a 0.1 increment for the second test. A total of 11 test data sets composed (a total 20,460,000 randomly generated data elements) are considered in this test.

In order to evaluate the influence of different query scenarios on index efficiency, the same global efficiency test has been carried out several times by applying each time a modified test query set.

In order to ensure that the influential parameter spectrum has been equably considered, the modified test query set has been generated by fixing the value of an influential factor, one factor for each modified test query set. This way, all the fuzzy sets modeling the restrictions of the applied flexible conditions of a test query set are randomly generated ensuring a fixed imprecision or fuzziness degree ranging from 0 to 1 by applying an increment of 0.1. Similarly, the spectrum of threshold values is explored by fixing threshold values ranging from 0 to 1 and an increase of 0.1.

As a result, 33 modified test query sets are

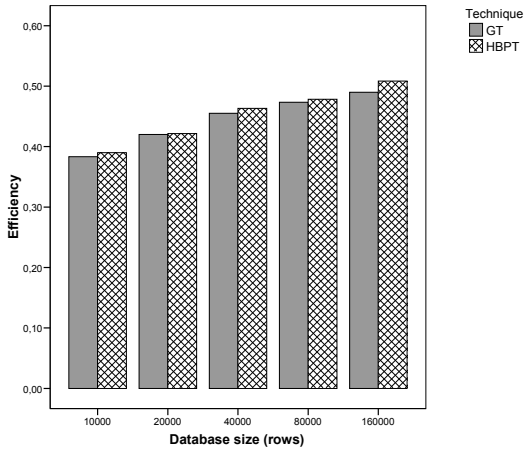

Figure 2: Comparison of efficiency under different database sizes

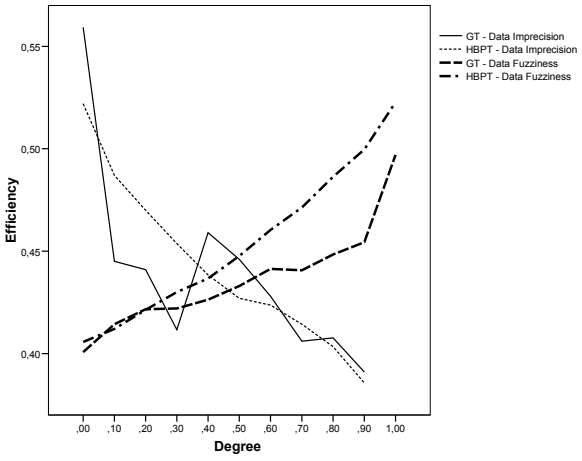

Figure 3: Comparison of efficiency under different data imprecision and fuzziness degrees

randomly generated. Each query test is then applied on the randomly generated test data set to measure the average efficiency of each index.

The described tests consider a total of 330,000 randomly generated queries and 9,900,000 query evaluations when they are applied to the 30 databases included in the test data set.

6 Experiment results

Results from the previously described experiment yield an average efficiency of the indexing techniques of 0.45 with a 0.04 standard deviation for HBPT, and 0.44 with a 0.04 standard deviation for GT. The minimum difference, which is within the standard deviation range, practically means a similar performance of both indexing techniques.

6.1 Influence of data related factors

The first considered influential factor is the database size which results in bigger index structures and may result in an increase in the cardinality of the results. Figure 2 shows the average efficiency of the compared techniques for different database sizes. The figures shows a similar performance for both indexing techniques, where HBPT slightly overperforms GT.

At first glance, it can be concluded that the larger the database size the greater the performance. This conclusion is counterintuitive as the larger the database size the greater the index overhead due to directory reads. Actually, a deep study reveals that the performance increment shown in fig. 2 is caused by an increase of the numbers of query results. The number of query results is directly related with the database size as the proposed test generates the test data set by randomly selecting values inside $[-1,000,000\ ,\ 1,000,000]$ whatever the database size is. This makes that the larger the database the greater the number of data elements inside a given interval (i.e. the greater the data density).

In conclusion, the results show that data density, not the database size, is an influential factor on index efficiency of both evaluated indexing methods. This is explained by the fact that a high data density means a greater number of query results that significatively reduces the impact of index overhead.

The results obtained by the previously described test for evaluating the influence of data imprecision on index efficiency are shown in fig. 3. In it, it can be seen that both indexing methods have a similar tendency, but also it can be noticed that HBPT is more stable than BT under data imprecision changes.

From the observed tendency, it can be concluded that the greater the data imprecision the smaller the efficiency. This is an expected behavior as the greater the support of data

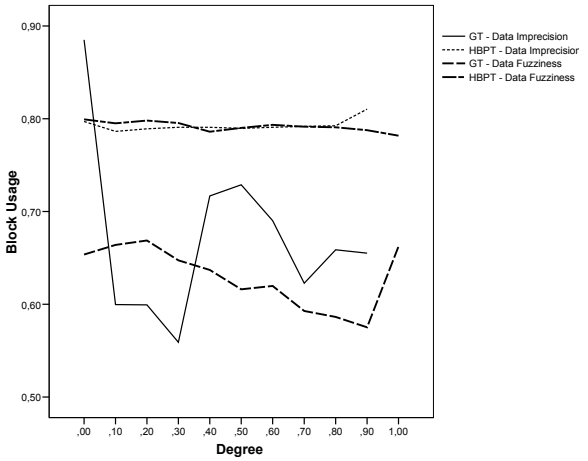

Figure 4: Block usage under different data imprecision and fuzziness degrees

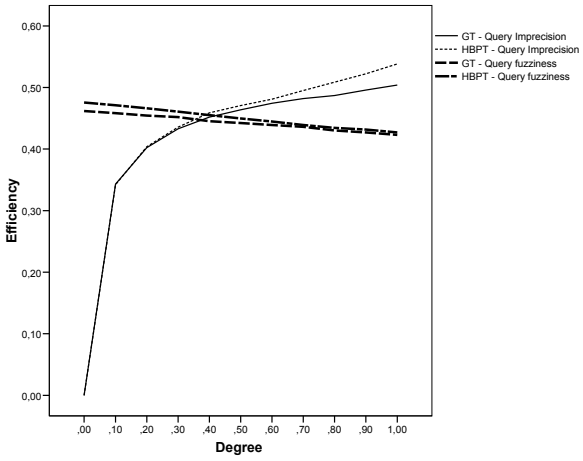

Figure 5: Comparison of efficiency under different query imprecision and fuzziness degrees

elements, the greater can be its core. Large cores of data elements reduce the number of query results, as it is less probable given a query to found data elements whose core is contained in the condition base. Finally, less query results means a greater impact of index overhead that results in a smaller index efficiency.

The aforementioned efficiency fluctuation of GT is directly caused by the fluctuation of its block usage. The block usage of GT is strongly data distribution dependent, as this indexing technique does not ensure a minimum block usage value. In contrast, HBPT ensures a minimum block usage of 0.5. Figure 4 shows the fluctuation of block usage in GT in contrast with the stability of HBPT block usage.

As previous results show a significative influence of the imprecision degree of the data, and taking into account the imprecision degree of data only influences indirectly on the extent of the core of data elements, it is expected that fuzziness degree would be also significantly influential.

The results shown in fig. 3 confirm this conjecture. Results show for both indexing techniques that the greater the fuzziness degree, the greater the efficiency. This is explained by the increment of query results related to an increment of the fuzziness degree, which causes smaller cores for data elements and thus increment the number of data elements whose core is contained is the condition base of a given query.

One more time, a fluctuation of GT efficiency can be observed. This fluctuation is the result of a fluctuation of the block usage of GT, shown in fig. 4, that makes evident its sensibility to data distribution in contrast with HBPT stability.

6.2 Influence of query related factors

With regards to the set of influential query factors, fig. 5 shows the relation between query imprecision and index efficiency. It can be seen that both indexing techniques are affected by this factor, as it is the lower bound of condition base extent, and so drastically determine the average number of query results. The efficiency is particularly low for the extreme case of a query imprecision of 0, which means a crisp condition. It can also be observed that the efficiency of GT does not grow at the rate of HBPT efficiency when query imprecision is high.

Query fuzziness is another considered influential factor on index efficiency. Figure 5 shows the comparison of the efficiency of GT and HBPT under different query fuzziness conditions. As the fuzziness degree indirectly determines the extent of the core of conditions, and given that the extent of the core of con-

ditions means an upper bound for its condition base, the higher the imprecision the smaller the number of query results, and so the smaller the efficiency. Additionally, in the figure it can be seen that this factor is more influential for GT than for HBPT, specially for low fuzziness degrees where the difference of efficiency is larger.

Finally, the observed results and influence of query threshold is the same as the influence of fuzziness degree (Not included in fig. 5 for the sake of clarity). In fact, both factors indirectly determine the average number of query results by reducing the average extent of the base of conditions.

7 Concluding remarks and future works

In this paper, a new indexing technique, HBPT, has been proposed and evaluated. Experimental results reveal a similar average efficiency for both GT and HBPT. HBPT has proved to be a more stable indexing method than GT, which present instability issues relating to data distribution. It makes GT more affected by data related factors. Moreover, HBPT is less affected by the studied query related factors.

Future work will focus on providing and studying indexing mechanisms (with low implementation cost if possible) for other imprecise, uncertain and inapplicable data types, such as scalar fuzzy data, fuzzy objects and fuzzy collections.

Acknowledgements

This work has been partially supported by the Ministry of Education and Culture of Spain under grant TIN-68084-C02-00 and by the Council for Innovation, Science and Corporations of Andalusia (Spain) under grant P06-TIC-01570.

References

[1] C. Barranco, J. Campaña, and J. Medina. A B+-tree based indexing technique for fuzzy numerical data. *Fuzzy Sets and Systems*, In Press, 2008.

[2] R. Bayer and E. M. McCreight. Organization and maintenance of large ordered indexes. *Acta Informatica*, 1(3):173–189, 1972.

[3] P. Bosc and M. Galibourg. Indexing principles for a fuzzy data base. *Information Systems*, 14(6):493–499, 1989.

[4] P. Bosc and O. Pivert. Fuzzy querying in conventional databases. In *Fuzzy logic for the management of uncertainty*, chapter Fuzzy querying in conventional databases, pages 645–671. John Wiley & Sons, Inc., 1992.

[5] S. Helmer. Evaluating different approaches for indexing fuzzy sets. *Fuzzy Sets and Systems*, 140(1):167–182, 2003.

[6] A. Kumar. G-tree: a new data structure for organizing multidimensional data. *IEEE Transactions on Knowledge and Data Engineering*, 6(2):341–347, 1994.

[7] J. Lawder and P. King. Using space-filling curves for multi-dimensional indexing. In *Advances in Databases 17th British National Conference on Databases, BNCOD 17 Exeter, UK, July 3–5, 2000 Proceedings*, volume 1832 of *Lecture Notes in Computer Science*, pages 20–35, 2000.

[8] C. Liu, A. Ouksel, P. Sistla, J. Wu, C. Yu, and N. Rishe. Performance evaluation of g-tree and its application in fuzzy databases. In *CIKM '96: Proceedings of the fifth international conference on Information and knowledge management*, pages 235–242, New York, NY, USA, 1996. ACM Press.

[9] F. E. Petry and P. Bosc. *Fuzzy databases: principles and applications*. International Series in Intelligent Technologies. Kluwer Academic Publishers, 1996.

[10] A. Yazici and D. Cibiceli. An index structure for fuzzy databases. In *Proceedings of the Fifth IEEE International Conference on Fuzzy Systems*, volume 2, pages 1375–1381 vol.2, 1996.

[11] A. Yazici and D. Cibiceli. An access structure for similarity-based fuzzy databases. *Information Sciences*, 115(1-4):137–163, Apr. 1999.

[12] A. Yazici, C. Ince, and M. Koyuncu. An indexing technique for similarity-based fuzzy object-oriented data model. In H. Christiansen, M.-S. Hacid, T. Andreasen, and H. Larsen, editors, *Flexible Query Answering Systems*, volume 3055 of *Lecture Notes in Computer Science*, pages 334–347. Springer, 2004.