An Evolutionary Algorithm for the Processing of Fuzzy Quantified Queries using Clustering

Garima Singh
Department of Computer Science & Engineering
Madan Mohan Malaviya Engineering College
Gorakhpur-273010, U. P., INDIA
garimasingh2307@gmail.com

Awadhesh Kumar Sharma
Associate Professor, Department of Computer Science & Engineering
Madan Mohan Malaviya Engineering College
Gorakhpur-273010, U. P., INDIA
akscse@rediffmail.com

Abstract— **this paper is about to make a contribution in the field of fuzzy query processing. We are mainly interested in query mechanism for evaluation of fuzzy quantified queries on the relational database. We restrict our work to fuzzy quantified sentences of form S1, involving linguistic quantification of fuzzy conditions over crisp sets. We proposed a new algorithm for the processing of fuzzy quantified queries using clustering. This algorithm is an optimization of an existing query evaluation mechanism.**

Keywords- Fuzzy Quantifier; Fuzzy Query; Membership degree; Fuzzy Predicate; Fuzzy Modifier

I. INTRODUCTION

Often, a user faces the problem of how to express her or his information requirements in a formal query language supported by a given information system interface. These formal languages usually require a crisp (precise) specification of a query, while, for human beings, a query is best expressed in terms of a natural language. Thus, adding some flexibility to traditional querying systems seems to be a critical issue for enhancing their effectiveness and efficiency. As we know query is the statement requesting the retrieval of information from the database and query processing refers to the range of activities involved in extracting the data, satisfying a given condition from a database and delivered to the user according to the required format. Any query into which imprecise or vague terms are embedded are called fuzzy queries e.g. Give me a list of *young* patients suffering from fever. Fuzzy query allows users to express requirements involving preferences [2].

Quantifiers can be used to represent the amount of items satisfying a given predicate. In classic logic we have universal (for all, ∀) and existential (exists, ∃) quantifiers. In fuzzy logic basically we have absolute and proportional quantifiers. Absolute quantifiers represent amounts that are absolute in nature. For example, “much more than 10,” “close to 100,” and so forth. Proportional quantifiers (or relative quantifiers) are those as "at least half" or "most of" that are proportional in nature.

Functionally, linguistic quantifiers [3] are usually of one of three types:

1. **Increasing quantifiers** (as "at least n", "all", "most") are characterized by:
   μ_Q (a) ≤ μ_Q (b) for all a<b
2. **Decreasing quantifiers** (as "a few", "at most n") are characterized by:
   μ_Q (a) ≥ μ_Q (b) for all a<b
3. **Unimodal quantifiers** (as "about q") have the property that:
   For some c, μ_Q (c) =1 and μ_Q (a) ≤ μ_Q (b) for all a<b≤c and μ_Q (a) ≥ μ_Q (b) if c≤a<b.

There are essentially two kinds of quantified statements [4] [8]. First **“Q X’s are A”,** we call them “**sentences of form S1**”. Second “**Q A X’s are B**”, we call them “**sentences of form S2**”. In both where Q is a fuzzy quantifier, X is a regular set, A is a fuzzy predicate on X. In sentences of form S2, B is also a fuzzy predicate on X. Here we are using a query language, called SQLf [9] for the processing fuzzy quantified queries over relational database. SQLf is an extension of SQL, which is a standard for database querying.

II. RELATED WORK

The interpretation of fuzzy quantifiers has received attention from several researchers [1] [5] [6] [7] [10]. In spite of the existence of several interpretations for fuzzy quantifiers, there is not one totally appropriate for database querying and that can be evaluated in an efficient way [8] [9].

Leonid Jose's provide a well defined interpretation for each category of fuzzy quantifiers. This interpretation is based on a linguistic transformation principle: a translation of fuzzy quantified statements into statements using existential quantifier, conjunction and negation. The given interpretation is completely suitable for database flexible querying. This interpretation satisfies all the imposed constrains [8] in this application field. This feature is an advantage over most of the other known interpretations that do not satisfy all of these constraints so we have decided to extend this work.

Leonid Jose's Interpretation [3] for Sentences of form S1

$$\mu\big(Q(X,fc)\big)$$

Where Q is a fuzzy quantifier, X is a regular set, fc is a fuzzy predicate on X

If Q is an increasing absolute quantifier:

$$\mu\big(Q(X,fc)\big) = \sup_{i\in\{0\ldots|X|\}} \left(\min_{x\in X}\left(\mu_Q(i), i_{\sup(\mu_{fc}(x))}\right)\right)$$

If Q is an increasing proportional quantifier:

$$\mu\big(Q(X,fc)\big) = \sup_{i\in\{0\ldots|X|\}} \left(\min_{x\in X}\left(\mu_Q\left(\frac{i}{|X|}\right), i_{\sup(\mu_{fc}(x))}\right)\right)$$

If Q is a decreasing absolute quantifier:

$$\mu\big(Q(X,fc)\big) = \sup_{i\in\{0\ldots|X|\}} \left(\min_{x\in X}\left(\mu_Q(i), i + 1_{\inf(1-\mu_{fc}(x))}\right)\right)$$

If Q is a decreasing proportional quantifier:

$$\mu\big(Q(X,fc)\big) = \sup_{i\in\{0\ldots|X|\}} \left(\min_{x\in X}\left(\mu_Q\left(\frac{i}{|X|}\right), i + 1_{\inf(1-\mu_{fc}(x))}\right)\right)$$

If Q is a unimodal absolute quantifier:

$$\mu\big(Q(X,fc)\big) = \min_{l,\, r\in\{0\ldots|X|\}} \begin{pmatrix} \sup_{x\in X}\big(\min\big(\mu_Q(l), l_{\sup(\mu_A(x))}\big)\big), \\ \sup_{x\in X}\left(\min\left(\mu_Q(r), r + 1_{\inf(1-\mu_{fc}(x))}\right)\right) \end{pmatrix}$$

If Q is a unimodal proportional quantifier:

$$\mu\big(Q(X,fc)\big) = \min_{l,\, r\in\{0\ldots|X|\}} \begin{pmatrix} \sup_{x\in X}\left(\min\left(\mu_Q\left(\frac{l}{|X|}\right), l_{\sup(\mu_A(x))}\right)\right), \\ \sup_{x\in X}\left(\min\left(\mu_Q\left(\frac{r}{|X|}\right), r + 1_{\inf(1-\mu_{fc}(x))}\right)\right) \end{pmatrix}$$

III. PROPOSED ALGORITHM

Although Leonid Jose's provide a well defined interpretation for each category of fuzzy quantifiers but to compute the satisfaction degree of each tuple in a large relation using this interpretation is very time consuming. Here we are proposing a clustering algorithm and then we implement it on the query base relation. After that on the resultant tuples we applied Leonid Jose's interpretation to find out the satisfaction degree of fuzzy quantified queries. By doing so the time required to process the query has been reduced.

A. Basics of Clustering

- **Cluster**
  Let X be a set of data, that is X = $\{x_1, x_2, \ldots x_n\}$. Clustering of X is its partition into m ($m \leq n$) clusters $C_i, \ldots, C_m$ so that
  1. None of the clusters is empty; $C_i \neq \emptyset$
  2. Every sample belongs to a cluster
  3. Every sample belongs to a single cluster (crisp clustering); $C_i \cap C_j = \emptyset$, i≠j
  Naturally, it is assumed that elements in cluster C_i are in some way "more similar" to each other than to the element in other clusters.

- **Average Distance (AD)** of the distance between every pair of neighboring numerical data value in the sorted data sequence can be calculated as follows:
  Assume that there is an ascending numerical data sequence shown as below:
  $d_{1,0} = d_{1,1} = \ldots < d_{2,0} = d_{2,1} = \ldots < \ldots < d_{s-1,0} = d_{s-1,1} = \ldots < d_{s,0} = d_{s,1} = \ldots$,
  where $d_{i,0}, d_{i,1}, \ldots$, and $d_{i,j}$ are the numerical data with the same value, $1 \leqslant i \leqslant s$, and $j \geqslant 0$ and s is the number of different numerical data in the sorted data sequence (here same data in the sorted data sequence are only counted once).
  $$AD = \frac{\sum_{i=0}^{s-1}(d_{i+1,0} - d_{i,0})}{(s-1)}$$

- **Cluster average distance (C_{AD})** is the average distance of distances between every pair of neighboring elements in a cluster.

- **Cluster center**
  Assume that the numerical data have been clustered into m clusters, i.e. $C_i, \ldots, C_m$. Let C_c (i) denote the cluster center (C_c) of cluster C_i . Assume that $C_i = \{d_1, d_2, \ldots, d_p\}$, where p denotes the number of elements in C_i.
  $$C_c(i) = \frac{\sum_{i=0}^{p} d_i}{p}$$

- **Normalized cluster center (NC_c)**
  $$NC_c(i) = (C_c(i) - \text{low})/\text{range}$$
  where range = (high-low) , low is the smallest and high is the largest data value respectively in given numerical data sequence.

B. Proposed Clustering Algorithm

Step 1: Sort the numerical data in an ascending sequence.
Step 2: Calculate the average distance (AD) of the distance between every pair of neighboring numerical data value in the sorted data sequence, where the same data value are only counted once.
prev be the first numerical data and current be the second numerical data in the sorted data sequence. Put the first data value in the sorted data sequence into the first cluster. Let pre prev = prev.
If (current-prev ⩽ AD) then
  put the current into the first cluster and calculate cluster average distance(C_{AD}) of the cluster;
else
  put the current a new cluster;
prev = current and current = next data value in the sorted data sequence;
end
Step3: If prev the first element in a cluster then
  if (current-prev ⩽ AD) and (current-prev < prev-pre prev) then
    put current into the cluster and calculate the value of C_{AD} of the cluster;
  else
    put current into a new cluster;
  pre prev = prev, prev = current and current = next data value in the sorted data sequence;
else if (current-prev ⩽ AD) and (current-prev ⩽ C_{AD})then
  put current into the cluster and calculate the value of C_{AD} of the cluster;
else
  put current into a new cluster;

pre prev = prev, prev = current and current = next data value in the sorted data sequence;
end.

Step 4: If no data in the sorted data sequence need to be clustered then
Stop
else
go to Step 3.

C. A New Algorithm for the Processing of Fuzzy Quantified Queries using Clustering

Input Fuzzy Query
SELECT <A_s> FROM <R> GROUP BY <A_g> WHERE <Q> <X> <fm + fc>
THRESHOLD <α>
Where A_s is select attribute(s) list, $A_g$is grouping attribute(s) list, <Q> is any fuzzy quantifier, <X> is fuzzy attribute because it is an attribute of relation R on which fuzzy condition is mapped, <fm + fc> is a fuzzy pair in which <fm> is a fuzzy modifier (is optional) and <fc> is a fuzzy condition (fuzzy predicate), <t> is a threshold or we can say α-cut associated with the query (is optional).

Steps of Algorithm

1. First find out the fuzzy attribute from the fuzzy query i.e. X.
2. Apply the clustering algorithm on the numerical data values of fuzzy attribute X.
3. After applying the above clustering algorithm we will obtain clusters of the numerical data that is C_i,…, C_m
4. Calculate the cluster center of each cluster obtained above.
5. Calculate the normalized cluster center of each cluster obtained above.
6. Find the interval of fuzzy predicate (fuzzy condition) from the table of interval of fuzzy pair.
7. Find the normalized cluster center which falls in the interval of fuzzy predicate.
8. Put all the tuples whose attribute values of the fuzzy attribute fall in cluster of above normalized cluster center in a new table named as T_{final}.Now we got final set of tuples only on which we will perform further processing.
9. Find out the type of fuzzy quantifier i.e. absolute or proportional and its functional type that is: increasing, decreasing or unimodal.
10. According to its type select the appropriate formula given by Leonid Jose [3] and calculate $\mu(Q(X, fc))$ only on table T_{final} and represent the calculation in a new table $T_{membership\ degree}$. In $T_{membership\ degree}$ we will discard the column of attribute(s) that is not related with query
11. Apply threshold α if given in query and produce the result of fuzzy query in the table T_{output}.

Example: Let us consider the relation STUDENT (name, roll. No., marks, branch code, age)

TABLE 1: STUDENT

S. No.	Name	Roll. No.	Marks	Branch Code	Age
1	Akansha	12001	95	1	18
2	Amrita	12007	97	1	27
3	Anjali	12004	97	1	20
4	Deepti	12009	23	1	19
5	Nidhi	12003	90	1	27
6	Nikita	12002	99	1	17
7	Shruti	12045	23	1	24
8	Swati	12047	24	1	15
9	Kavita	12046	90	1	26
10	Priyanka	12034	47	1	20
11	Shalini	12042	89	2	18
12	Varsha	12005	53	2	22
13	Kriti	12033	88	2	18
14	Shivani	12043	25	2	15
15	Surabhi	12006	48	2	21
16	Sakshi	12044	83	2	25
17	Neetu	12035	25	2	21
18	Prachi	12008	32	2	25
19	Rati	12048	53	2	18
20	Ishita	12049	83	2	25

21	Neha	12010	30	3	22
22	Ayushi	12036	53	3	18
23	Sneha	12011	84	3	24
24	Disha	12032	31	3	21
25	Aditi	12028	83	3	22
26	Anu	12040	33	3	19
27	Bhavna	12029	82	3	24
28	Megha	12039	55	3	23
29	Palak	12012	79	3	25
30	Sushmita	12013	33	3	18
31	Jyoti	12038	55	4	24
32	Manya	12017	54	4	17
33	Nitya	12030	78	4	19
34	Pooja	12041	82	4	23
35	Shubhi	12014	54	4	16
36	Vidhi	12031	82	4	22
37	Riya	12037	77	4	20
38	Sonali	12015	54	4	17
39	Arpita	12016	77	4	18
40	Shilpi	12052	88	4	16
41	Ankita	12050	44	5	19
42	Deepika	12022	46	5	19
43	Prerna	12025	64	5	21
44	Isha	12051	34	5	15
45	Nikunj	12026	72	5	17
46	Priya	12021	44	5	20
47	Nandita	12053	73	5	22
48	Vidhya	12020	35	5	19
49	Varnika	12054	64	5	16
50	Deepa	12027	45	5	20
51	Rashmi	12018	67	6	23
52	Khushi	12024	72	6	27
53	Monika	12019	44	6	15
54	Aradhna	12055	74	6	23
55	Shikha	12023	75	6	26
56	Preeti	12057	84	6	16
57	Shradha	12056	94	6	26
58	Prabha	12060	22	6	21
59	Anita	12058	96	6	22
60	Smriti	12059	98	6	25

The input fuzzy query “Find the branches where most of student having very good marks”, with a threshold 0.6, can be written as:
“SELECT <Name, Roll. No.> FROM <Student> GROUP BY <Branch Code> WHERE MOST_ OF Marks = very good THRESHOLD <0.8>”

In this query “Marks” is a fuzzy attribute, “MOST_OF” is a fuzzy quantifier and “very good” is a fuzzy pair in which “very” is a fuzzy modifier and “good” is a fuzzy condition (fuzzy predicate). Equations of membership function of MOST_OF:

$$f(q) = \begin{cases} 0 & if\ q \le \frac{2}{10} \\ (q - (\frac{2}{10}))/(\ \frac{6}{10} - \frac{2}{10}) & if\ \ \frac{2}{10} < q \le \frac{6}{10} \\ 1 & if\ q > \frac{6}{10} \end{cases}$$

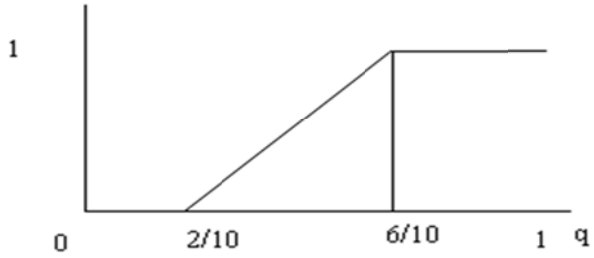

Figure 1: Membership function of MOST_OF

Equations of membership function of good:

f(marks) = marks/100

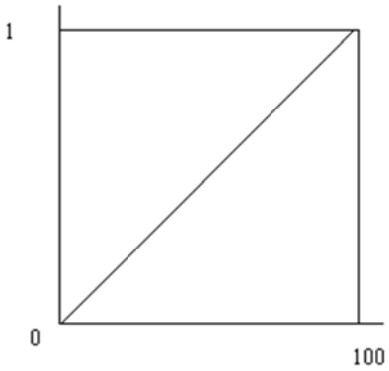

Figure 2: Membership function of good

TABLE 2: Interval of fuzzy pairs

Fuzzy pair	min	max
Good	0.65	0.85
very good	0.85	1.0
medium	0.45	0.65
Poor	0.25	0.45
very poor	0.0	0.25

On Applying Proposed Fuzzy Query Processing Algorithm
Step 1 Fuzzy attribute in the query is "Marks".
Step 2 Now apply Clustering Algorithm:
"Marks" is a fuzzy attribute so we will sort the data values under the column "Marks" in increasing order.
22, 23, 23, 24, 25, 25, 30, 31, 32, 33, 33, 34, 35, 44, 44, 44, 45, 46, 47, 48, 53, 53,53, 54, 54, 54, 55, 55, 64, 64, 67, 72, 72, 73, 74, 75, 77, 77, 78, 79, 82, 82, 82, 83,83, 83, 84, 84, 88, 88, 89, 90, 90, 94, 95, 96, 97, 97, 98, 99

Average distance (AD) = [(23-22) + (24-23) + (25-24) + (30-25) + (31-30) + (32-31)+ (33-32) + (34-33) + (35-34) + (44-35) + (45-44) + (46-45) + (47-46) + (48-47) + (53-48) + (54-53) + (55-54) + (64-55) + (67-64) + (72-67) + (73-72) + (74-73) + (75-74) + (77-75) + (78-77) + (79-78) + (82-79) + (83-82) + (84-83) + (88-84)+ (89-88) + (90-89) + (94-90) + (95-94) + (96-95) + (97-96) + (98-97) + (99-8)]/38
= (1 + 1 + 1 + 5 + 1 + 1 + 1 + 1 + 1 + 9 + 1 + 1 + 1 + 1 + 5 + 1 + 1 + 9 + 3 + 5 + 1 + 1 + 1 + 2 + 1 + 1 + 3 + 1 + 1 + 4 + 1 + 1 + 4 + 1 + 1 + 1 + 1 + 1)/38
= 2.026

Step 3 After applying clustering algorithm we obtained following clusters:
C_1 = {22, 23, 23, 24, 25, 25}
C_2 = {30, 31, 32, 33, 33, 34, 35}
C_3 = {44, 44, 44, 45, 46, 47, 48}
C_4 = {53, 53, 53, 54, 54, 54, 55, 55}
C_5 = {64, 64}
C_6 = {67}
C_7 = {72, 72, 73, 74, 75}

C_8 = {77, 77, 78, 79}
C_9 = {82, 82, 82, 83, 83, 83, 84, 84}
C_{10} = {88, 88, 89, 90, 90}
C_{11} = {94, 95, 96, 97, 97, 98, 99}

Step 4 Cluster center of above formed cluster:
$C_c(1)$ = (22 + 23 + 23 + 24 + 25 + 25) / 6 = 23.67
$C_c(2)$ = (30 + 31 + 32 + 33 + 33 + 34 + 35) / 7 = 32.57
$C_c(3)$ = (44 + 44 + 44 + 45 + 46 + 47 + 48) / 7 = 45.43
$C_c(4)$ = (53 + 53 + 53 + 54 + 54 + 54 + 55 + 55) / 8 = 53.88
$C_c(5)$ = (64 + 64) / 2 = 64.00
$C_c(6)$ = 67.00
$C_c(7)$ = (72 + 72 + 73 + 74 + 75) / 5 = 73.20
$C_c(8)$ = (77 + 77 + 78 + 79) / 4 = 77.75
$C_c(9)$ = (82 + 82 + 82 + 83 + 83 + 83 + 84 + 84) / 8 = 82.88
$C_c(10)$ = (88 + 88 + 89 + 90 + 90) / 5 = 89.00
$C_c(11)$ = (94 + 95 + 96 + 97 + 97 + 98 + 99) / 7 = 96.57

Step5 Normalized cluster center of above formed cluster:
$NC_c(1)$ = (23.67-22) / (99-22) = 0.02
$NC_c(2)$ = (32.57-22) / (99-22) = 0.14
$NC_c(3)$ = (45.43-22) / (99-22) = 0.30
$NC_c(4)$ = (53.88-22) / (99-22) = 0.41
$NC_c(5)$ = (64.00-22) / (99-22) = 0.55
$NC_c(6)$ = (67.00-22) / (99-22) = 0.58
$NC_c(7)$ = (73.20-22) / (99-22) = 0.66
$NC_c(8)$ = (77.75-22) / (99-22) = 0.72
$NC_c(9)$ = (82.88-22) / (99-22) = 0.79
$NC_c(10)$ = (89.00-22) / (99-22) = 0.87
$NC_c(11)$ = (96.57-22) / (99-22) = 0.97

Step 6 Interval of "very good" is [0.85, 1.0]
Step 7 Normalized cluster center $NC_c(10), NC_c(11)$ fall in this interval.
Step 8 Put all the tuples in the table T_{final} whose attribute values of fuzzy attribute (Marks) fall in the cluster C_{10}, C_{11}

TABLE 3: T_{final}

S. No.	Name	Roll. No.	Marks	Branch Code	Age
1	Kriti	12033	88	2	18
2	Shilpi	12052	88	4	16
3	Shalini	12042	89	2	18
4	Nidhi	12003	90	1	27
5	Kavita	12046	90	1	26
6	Shradha	12056	94	6	26
7	Akansha	12001	95	1	18
8	Anita	12058	96	6	22
9	Amrita	12007	97	1	27
10	Anjali	12004	97	1	20
11	Smriti	12059	98	1	25
12	Nikita	12002	99	4	17

Step 9 MOST_OF is an increasing proportional quantifier.
Step 10 Formula for $\mu(Q(X, fc))$ is:

$$\mu(Q(X,fc)) = \sup_{i \in \{0...|X|\}} \left(\min_{x \in X} \left(\mu_Q\left(\frac{i}{|X|}\right), i_{\sup(\mu_{fc}(x))} \right) \right)$$

here $|X| = 10$ for each branch since each branch have total 10 no. of student. We will discard the column "Age" as it is not related with query.

TABLE 4: $T_{membership\ degree}$

Branch Code	i	Name	Roll. No.	Marks	$\mu_{i=\ i_{\sup(\mu_{fc}\ (x))}}$ x∈X	$\mu_{Qi} = \mu_Q\left(\frac{i}{\lvert X\rvert}\right)$	$\min(\mu_{Qi}, \mu_{Qi})$	$\mu(Q(X, fc))$
1	1	Nidhi	12003	90	0.90	0.0	0.0	0.98
	2	Kavita	12046	90	0.90	0.0	0.0	
	3	Akansha	12001	95	0.95	0.25	0.25	
	4	Amrita	12007	97	0.97	0.5	0.5	
	5	Anjali	12004	97	0.97	0.75	0.75	
	6	Smriti	12059	98	0.98	1.0	0.98	
2	1	Kriti	12033	88	0.88	0.0	0.0	0.0
	2	Shalini	12042	89	0.89	0.0	0.0	
4	1	Shilpi	12052	88	0.88	0.0	0.0	0.0
	2	Nikita	12002	99	0.99	0.0	0.0	
6	1	Shradha	12056	94	0.94	0.0	0.0	0.0
	2	Anita	12058	96	0.96	0.0	0.0	

Step 11 α = 0.8 so we will take the values only having membership degree > 0.8
After applying α- cut output of the fuzzy query is shown in Table 5

TABLE 5: T_{output}

Branch Code	i	Name	Roll. No.	Marks	Membership degree
1	1	Nidhi	12003	90	0.98
	2	Kavita	12046	90	
	3	Akansha	12001	95	
	4	Amrita	12007	97	
	5	Anjali	12004	97	
	6	Smriti	12059	98	

As we know linguistic terms are user dependent so the meaning of these terms may vary from user to user. If the fuzzy set of any linguistic term will change then we do not required to perform clustering again.

Data in database is subject to change so if any new tuple gets inserted in the query base relation then there is no need to perform clustering again. In this case all that we have to do is given below.
Find

$$\min\ (\ |new\ data\ value - C_c\ (i)|\ )$$
$$i \in \{1,2,\ldots,m\}$$

where m is the total number cluster already exist.
This gives us the cluster to which new data value having minimum distance.
After that perform the following step:
If $\min\ (\ |new\ data\ value - C_c\ (i)|\ ) \leq$ (range of that cluster) then
put it in to the same cluster ;
else
form a new cluster and m= m+1 ;

Range of cluster can be calculated by taking the difference between highest and lowest data value in that cluster. In this way the Leonid Jose's interpretation has become more efficient to process the fuzzy quantified queries over relational database. This is all about the optimization of Leonid Jose's interpretation for fuzzy quantified queries.

IV. CONCLUSIONS

Database queries involving imprecise or fuzzy predicates are currently an evolving area of academic and industrial research. Next generation database applications for example, design; environmental and scientific applications have to cope with incomplete and uncertain data and queries. A new algorithm for the processing of fuzzy quantified queries using clustering is proposed to process the queries efficiently. This algorithm attempt to keep low the number of rows access in fuzzy querying thus reduced the evaluation time for the processing of fuzzy quantified queries. This algorithm also works well in case of change in query base relation or membership function of fuzzy sets.

REFERENCES

[1] L.A. Zadeh, “A computational approach to fuzzy quantifiers in natural languages,” Computer Mathematics with Applications, Vol. 8, pp 149-183, 1983.

[2] Galindo J., Urrutia A., Piattini M., “Fuzzy databases: modeling, design and implementation,” publish by Idea Group Publishing Hershey, USA, 2005.

[3] Leonid Jose and Tineo Rodríguez, “Extending RDBMS for allowing fuzzy quantified queries,” Lecture Notes in Computer Science, Vol. 1873, Mohamed Ibrahim-Josef Kung-Norman Revell (Eds.) Springer Verlag, pp. 407-41, 2000.

[4] Leonid Jose and Tineo Rodríguez, “A fuzzy quantifiers interpretation for database querying,” Lecture Notes in Computer Science, unpublished.

[5] R.R. Yager, “Quantified propositions in a linguistic logic,” J. Man Mach. Stud, Vol. 19, pp. 195–227, 1983.

[6] Janusz Kacprzyk and Andrzej Ziolkowski, “Database queries with fuzzy linguistic quantifiers,” IEEE Transactions on Systems, Man and Cybernetics, Vol. 16, No.3, pp. 474-478, 1986.

[7] R.R. Yager,”On ordered weighted averaging aggregation operators in multicriteria decision making,” IEEE Transaction on System, Man, and Cybernetics, Vol. 18, pp.183–190, Feb. 1988.

[8] P. Bosc, L. Lietard and O. Pivert, “Quantified statements and database fuzzy querying,” Fuzziness in Database Management Systems, Physica Verlag, pp. 275-308, 1995.

[9] P. Bosc, O. Pivert, “SQLf: A relational database language for fuzzy querying,” IEEE Transactions on Fuzzy Systems, Vol 3, No. 1, Feb 1995.

[10] R.R. Yager, "Connectives and quantifiers in fuzzay sets" Fuzzy Sets System. Vol. 40, pp. 39-76, 1991.