CODD’S RELATIONAL DATA MODEL AND FUZZY LOGIC: A PRACTICAL APPROACH TO FIND THE COMPUTER SOLUTION

Sabyasachi Chakraborty
Sr. Lecturer and Head of The Department
Department of IT and Computer Science
Icfai University-Sikkim

Abstract

The present paper deals with the relational database model of Codd and the application of fuzzy logic on it to derive the programming algorithm that aims to work on the fuzzy enabled database system. Firstly in this paper an analysis of codd’s relational model has taken place along with its fuzzy extension. In the next phase, a database solution along with a concrete computer algorithm is developed to yield the output of various fuzzy (imprecise) queries. In this paper a necessary modification or customization of classical and codd’s model has taken place in order to make a concrete practical solution.

Keywords: Data Base, Fuzzy Logic, Relation, Degree of Similarities, Ranking, Query, Functional Dependency, Algorithm.

I. Introduction

The relational database system and its development over the years has played a pivotal role in the area of software development and the areas related to data analysis, processing and data administration. The development of relational database management system and its driving components like relational algebra & relational calculus has facilitated the integration of diverse database entities and processing of static and dynamic queries for vast industrial data repositories. Introduction of Fuzzy Logic in the area of relational database system has brought a revolution. So far the relational database system used to yield integrated and complete information from the raw data but after the application of fuzzy logic to database systems, the generation of knowledge, answering to imprecise and indefinite queries and dealing with unstructured database entities are possible.

Fuzzy Logic breaks the barrier of [0,1] based Boolean system and assume all intermediate values between them to represent the quantification of indefinite parameter statements, thus the fuzzy base database management systems can answer queries with imprecise and indefinite parameter values.

The Codd’s relational model is extended with fuzzy basically to deal with imprecise queries and to generate knowledge by assigning the ranks or degrees to the different tuples of a database entity based on any given query and by assigning the degrees of similarity between a pair of values. Codd’s relational data model starts with data table and domain similarities then a row ranking is derived against a query parameter. In the stated model, the functional dependency in the relational database with fuzzy attributes are also analyzed. All these analyses can be applied to form practical solutions by forming the fuzzy relation and applying the fuzzy logic rules by the programming algorithm.

II. Literature review on Codd’s relational data model and fuzzy extension.

The similarity of tuples in a relation and the ranking of rows on the basis of that and the functional dependencies between different relations of the Codd’d model are surveyed in the paper “Codd’s relational model of data and fuzzy logic: comparisons, observations and some new results” by Radim Belohlavek and Vilem Vychodil of Department of Computer Science, Palacky University, Olomouc Tomkova of Czech Republic. The stated paper starts with some preliminary studies on fuzzy relational algebra dealing with fuzzy union, intersection, product etc. In the reviewed paper, firstly a data table is introduced that contains only three attributes: Candidate name, Age and Educational Background/Domain. The Domain similarity matrix is introduced to represent the degrees of similarity between all pairs of subjects. It has been shown in the reviewed paper that How a fuzzy or indefinite query can be solved by effective row ranking with respect to the value of the query parameter. The similarity matrix is designed by the random assignment of initial grades (for the first row only) on the basis of general perceptions but the remaining entries are made by obeying the reflexive, symmetric and transitive properties of similarity grades.

The discussion of the reviewed paper starts with the Data table as follows:
The Domain similarity matrix is as follows:

<table>
<thead>
<tr>
<th>Name</th>
<th>Age</th>
<th>Education</th>
</tr>
</thead>
<tbody>
<tr>
<td>Adams</td>
<td>30</td>
<td>Computer Science</td>
</tr>
<tr>
<td>Black</td>
<td>30</td>
<td>Computer Engineering</td>
</tr>
<tr>
<td>Chang</td>
<td>28</td>
<td>Accounting</td>
</tr>
<tr>
<td>Devis</td>
<td>27</td>
<td>Computer Engineering</td>
</tr>
<tr>
<td>Enke</td>
<td>36</td>
<td>Electrical Engineering</td>
</tr>
<tr>
<td>Francis</td>
<td>39</td>
<td>Business</td>
</tr>
</tbody>
</table>

Now the domain similarity is represented by a matrix with fuzzy values that represent the similarities between the subjects of education of the persons. The rules are:

\[ n_1 \approx n_2 \begin{cases} 1 & \text{if } n_1 = n_2 \\ 0 & \text{if } n_1 \neq n_2 \end{cases} \]

\[ a_1 \approx a_2 = s_a (| a_1 - a_2 |) \]

With scaling \( s_a : Z^+ \rightarrow [0,1] \)

The Domain similarity matrix is as follows:

<table>
<thead>
<tr>
<th></th>
<th>Acc</th>
<th>Buss</th>
<th>CE</th>
<th>CS</th>
<th>EE</th>
</tr>
</thead>
<tbody>
<tr>
<td>Acc</td>
<td>1</td>
<td>.7</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Buss</td>
<td>.7</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>CE</td>
<td>0</td>
<td>0</td>
<td>.9</td>
<td>1</td>
<td>.6</td>
</tr>
<tr>
<td>CS</td>
<td>0</td>
<td>0</td>
<td>.9</td>
<td>1</td>
<td>.7</td>
</tr>
<tr>
<td>EE</td>
<td>0</td>
<td>0</td>
<td>.6</td>
<td>.7</td>
<td>1</td>
</tr>
</tbody>
</table>

This matrix represents the degrees of similarities between all pairs of subjects and is expressed by fuzzy values. These degrees of similarities are determined on the basis of the knowledge of the subjects and their contents. In this paper Codd’s way of expressing row ranking against a certain query statement “List the suitability of the candidates where the desired age is about 30” is analyzed. Against the query, the row ranking may be done as follow:

<table>
<thead>
<tr>
<th>(t(A))</th>
<th>(t(B))</th>
<th>(t(C))</th>
</tr>
</thead>
<tbody>
<tr>
<td>Acc</td>
<td>Buss</td>
<td>CE</td>
</tr>
<tr>
<td>1.0</td>
<td>1.0</td>
<td>.9</td>
</tr>
<tr>
<td>.9</td>
<td>.8</td>
<td>.4</td>
</tr>
<tr>
<td>.5</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Here the query parameter is age and the pivot value is 30. The ranks of the rows with respect to the parameter and it’s desired value are the degrees of similarities of the rows for that parameter attribute with the desired value 30.

Now in discussing the similarity matrix, the plotted grades representing the degrees of similarities between the pairs of subjects are not just imaginary but based on three basic characteristics: reflexive \((U \approx U =1)\), symmetric \((U \approx V = V \approx U)\) and transitive \((U \approx V) \cdot (V \approx W) \rightarrow (U \approx W)\).

In case of \((U \approx V) \neq (V \approx W)\), \((U \approx V) = \min\{(U \approx V),(V \approx W)\}\) and in case of \((U \approx V) = (V \approx W)\), \((U \approx W) = \min\{(U \approx V),(V \approx W)\}\) or we can also say, \((U \approx V) \geq \min\{(U \approx V),(V \approx W)\}\). These rules are depicted from the assignment of fuzzy grades in the similarity matrix.

In the reviewed paper, the functional dependency of fuzzy relational database is discussed in a formularize way. The functional dependency of B on A is symbolized as: \(A \Rightarrow B\) where A and B are the fuzzy sets of attributes where a degree of similarity between any two tuples \(t_1\) and \(t_2\) is formulated as:

\[ t_1(C) \approx D t_2(C) \]

\[ = (D(t_1), D(t_2)) \rightarrow \cap_{y \in Y} (C(Y) \rightarrow (|t_1[y]| \approx t_2[y])) \]

For a ranked data table \(D\), the tuples \(t_1\) and \(t_2\) and \(C\) – a set of fuzzy attributes, the aforesaid degree represents the extents to which the tuples \(t_1\) and \(t_2\) have similar values on the attributes from \(C\). This similarity notation can be stated as: “if \(t_1\) and \(t_2\) are from \(D\) then for each attribute \(y\) from \(C\), \(t_1\) and \(t_2\) have similar values for \(y\)” or it can also be stated as, “The degree of similarity between \(t_1\) and \(t_2\) for each fuzzy attribute \(y\) \((y \in C)\) (if \(t_1\) and \(t_2\) are in \(D\)) is equal to the degree of similarity between \(t_1\) and \(t_2\) for entire set of fuzzy attributes \(C\)”

Now in the reviewed paper, the degree of fuzzy functional dependency of a set of attributes \(B\) on a set of attributes \(A\) is defined as:

A degree \(\|A \Rightarrow B\|_{D}\) to which functional dependency \(A \Rightarrow B\) is true in \(D\) is defined by:

\[\|A \Rightarrow B\|_{D} = \cap_{t_1, t_2} ((t_1(A) \approx_D t_2(A)) \rightarrow (t_1(B) \approx_D t_2(B)))\].

The above equation can be stated as: “A set of attributes \(B\) is functionally dependent on a set of attributes \(A\) if for any tuples \(t_1\) and \(t_2\): if \(t_1\) and \(t_2\) have similar values on the attributes from \(A\) then they have similar values on the attributes from \(B\)” or we can say, the degree of functional dependency of \(B\) on \(A\) is equal to the extent by which the degree of similarity between these two tuples on \(A\) imply the degree of similarity between these two tuples on \(B\).

In this research paper these aspects are studied and a practical solution is searched after some minute modification and extension.

### III. Formulation of the Author

To formulate a practical model of fuzzy database solution, firstly the data table and the similarity matrix is combined to form a fuzzy relation consisting of attributes:

\{ slno, subject-1, subject-2, Degree of Similarity \}
This relation is given a name “Similarity Relation” consisting of nxn tuples or rows for n number of subjects. The entries can be made into this relation externally which should be reflexive, symmetric and transitive. Once we create this said relation and insert the fuzzy entries into it then any fuzzy query can be processed using it. The programming algorithm to derive the fuzzy similarity grades and insert them into the relation is as follows:

1. Access the Database Entity set “Data Table”
2. Start from the 1st record
3. Initialize variable C=0
4. If the empty row not reached then
   5. £c Value (Deg. Of similarity from the relation)
   6. C=C+1
   7. Go to next Record
8. Repeat 4. To 7 until 4. is false
9. initialize i=0
10. if i=< C-1 then
   11. DSi 1
   12. i=i+1
13. Repeat 10. To 12. Until 10. is false
14. initialize i=0
15. initialize j=i+1
16. if j=<n-1 then
   17. DSi j
   18. DSj i
   19. j=j+1
21. initialize i=0
22. if i<n then
   23. initialize j=i+1
24. if j<n then
25. if (i ≠1) AND (i≠ j) then
26. initialize Lower Limit=0
27. initialize k=0
28. if k<n then
29. if ( Lower Limit < DSik x DSkj ) then
30. Lower Limit = DSik x DSkj
31. k=k+1
33. Display (Lower Limit)
34. DSi j
   35. j=j+1
36. Repeat 24. To 35. Until 24. is false
37. i=i+1
38. Repeat 22. To 37. Until 22. is false
39. Access the blank relation “Similarity Relation”
40. Initialize variable S=0
41. initialize variable i = 0
42. if i=< n-1 then
43. initialize variable j=0
44. if j=<n-1 then
45. S=S+1
46. Add New Record into “Similarity Relation”
47. Data Field(1) S
48. Data Field(2) Ei
49. Data Field(3) Ei
50. Data Field(4) DSij
51. Update Relation
52. j=j+1
53. Repeat 44. To 52. Until 44. Is false
54. i=i+1
55. Repeat 42. To 54. Until 42. Is false.

Now the above algorithm represents the computations of degrees of similarities by ensuring the reflexive, symmetric and transitive properties of the fuzzy entries. After this computation the relational values in the arrays can be inserted to the four columns of the proposed relation. Now to design this algorithmic solution, the transitive property is slightly modified:

(U ≈ y W) ≥{(U ≈ y V) . (V≈ y W)}  for all V for any given U and W.

Here the logic behind this formulation is: when we say V is similar to U by a certain degree then that degree measure the containment of U in V and similarly the degree of similarity of W with V stand for the containment of V in W. Thus the containment of U in W will surely be at least the product of the previous two. In other way we can use the notation:

S_D(X,Z) ≥ max (min(S_D(X,Z), S_D(Y,Z))).

Y€D

Now for the sake of designing a logically effective algorithm the above relation is restated as:

S_D(X,Z) ≥ max ((S_D(X,Y) . S_D(Y,Z))).

Y€D

Here the “product(“ . ”)” is used in place of “fuzzy min” to clarify the logic of containment.

Now for a query, “ List the suitability of candidates for electrical engineering” on the derived relation, the query parameter “Electrical Engineering” is assigned to a variable p. Then a Ø join operation has to be used to make a relationship between the initial data table and the similarity relation.

(Data Table) (Similarity Relation)

In order to notational simplicity let us rename this relation as "REL"

1. Start
2. Access REL
3. Initialize by the 1st record of REL
4. Initialize variable i = 0
5. If the Empty row not reached then
6. If p= Field(4) of REL( sub-1) then
7. n_i

ISSN No: 2250-3536 Volume 2, Issue 4, July 2012 23
Now let us consider a fuzzy query: “List the suitability of “Adam” for all subjects”. Here the query parameter “Adam” has to be assigned to a variable p. The following algorithm is derived to process the query:

1. Start
2. Access “Data Table”
3. Initialize by the 1st record of “Data Table”
4. If the empty row not reached then
5. If p= Data Field (0) then
6. String sub ← Data Field(2)
7. Go to Next Record
9. Access “Similarity Relation”
10. Start from the first Record
11. Initialize variable i = 0
12. If the empty row not reached
13. If sub=Data Field(1) then
14. Subject ← Data Field(2)
15. DS ← Data Field(3)
16. i = i + 1
17. Go to Next Record
19. Initialize j=0
20. If j < i then
21. Display the combination: { Subject , DS} to represent the j th tuple of the combination of attributes { Subject, Degree of Suitability}
22. j = j+1
24. End

Now in the reviewed paper, the row ranking was derived against a query where “Age” was the parameter. Here the assignment of ranks based on the query “List the candidates aged about 30” – is intuitive and abruptly done without any pre-fixed rule. Now the ranks can be derived in a logical way rather that just assigning values to them. Considering the value of the fuzzy query parameter, the concept of range around that value can be introduced. We can introduce the range to be represented by the highest among the differences between the pivot parameter value and all other entered values. For example in the given data table, the query parameter is “Age” and the pivot value=30 as per the query statement. Now in the data table, the highest of the differences between the pivot parameter value and all other values is 9 as 39 differs most from 30 among all the entries. We thus have 9+1=10 points including 30 and 39.

Now the Row rankings can be inserted in the “row rank” attribute in the data table as:

1. Start
2. Access “Data table”
3. Start from the 1st Record
4. Initialize variable i = P – (Max-1)
5. If i =< P+(Max -1) then
7. End
6. If the empty row of “Data Table” is not reached then
7. If i = data Field(1) then
8. Data Field(3) ➔ R, (Inserting Rank)
9. Go to Next Record
10. Repeat 6. To 9. Until 6. is false
11. i = i + 1
12. Repeat 5. To 11. Until 5. is false.
13. End

Now against the given query, the row rankings are derived and inserted as shown in the algorithms shown above. In the reviewed paper, the first equation:

\[ t_1( C ) = D, t_2( C ) = ( D(t_1). D(t_2) ) \cap_{Y \in Y} (C(Y) \rightarrow ((t_1[y] \approx t_2[y])) \]

is stated as: The degree of similarity between \( t_1 \) and \( t_2 \) in \( C \) (a set of fuzzy attributes) is equal to the degree of similarity between \( t_1 \) and \( t_2 \) for each attribute \( y \) (\( y \in C \)) if \( t_1 \) and \( t_2 \) are from \( D \).

Now let us start with a query on the given “Data Table” in the reviewed paper for a practical implementation of this notation: “List the candidates in the ascending order of suitability for the post of computer Engineer where the preferred age is 30”

To perform this first of all we have to add two ranking columns representing the row ranking for “Age” and the row ranking for “subject”. The grade entries for the ranking for the subjects are as given and for the ages as derived as above:

<table>
<thead>
<tr>
<th>Name</th>
<th>Age</th>
<th>Subject</th>
<th>Rank (For suitability for Age=30)</th>
<th>Rank (For suitability for Subject = “Computer Engineering”)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Adams</td>
<td>30</td>
<td>Comp. Sc</td>
<td>1</td>
<td>.9</td>
</tr>
<tr>
<td>Black</td>
<td>30</td>
<td>Comp. Engg</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>Chang</td>
<td>28</td>
<td>Accounting</td>
<td>.77</td>
<td>0</td>
</tr>
<tr>
<td>Devis</td>
<td>27</td>
<td>Comp. Engg</td>
<td>.66</td>
<td>1</td>
</tr>
<tr>
<td>Enke</td>
<td>36</td>
<td>Elect. Engg</td>
<td>.33</td>
<td>.6</td>
</tr>
<tr>
<td>Francis</td>
<td>39</td>
<td>Business</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

Now according to the query statement, for the age parameter, row 1 and 2 are the pivot rows and for “subject” parameter, row 2 is the pivot row. Now we apply the degree of similarity between the fuzzy attribute value in the pivot row and all other rows for both the attributes. We can derive an array of similarity for all rows of these two fuzzy rank attributes. Here it is proposed to apply the rule:

\[ \text{Min}(t^*_y, t_y) / \text{Max}(t^*_y, t_y) \]

to find the degree of similarity (on the basis of containment) of tuple \( t \) with pivot \( t^* \) for a fuzzy attribute \( y \). This is applicable however for any two tuples in the relation. Now we can have an average estimation of the degrees of similarity of all tuples with respect to the pivot tuple for both the fuzzy attributes. By doing so we get:

<table>
<thead>
<tr>
<th>Name</th>
<th>Net Grade</th>
</tr>
</thead>
<tbody>
<tr>
<td>Adams</td>
<td>.95</td>
</tr>
<tr>
<td>Black</td>
<td>1</td>
</tr>
<tr>
<td>Chang</td>
<td>.39</td>
</tr>
<tr>
<td>Devis</td>
<td>.83</td>
</tr>
<tr>
<td>Enke</td>
<td>.47</td>
</tr>
<tr>
<td>Francis</td>
<td>0</td>
</tr>
</tbody>
</table>

Now according to this “Net Grade” against the query, the list in descending order of suitability will be:

<table>
<thead>
<tr>
<th>Merit No</th>
<th>Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Black</td>
</tr>
<tr>
<td>2</td>
<td>Adams</td>
</tr>
<tr>
<td>3</td>
<td>Devis</td>
</tr>
<tr>
<td>4</td>
<td>Enke</td>
</tr>
<tr>
<td>5</td>
<td>Chang</td>
</tr>
<tr>
<td>6</td>
<td>Francis</td>
</tr>
</tbody>
</table>

Now this process is applicable between any two tuples \( t_1 \) and \( t_2 \). For example, if we choose 4th and 5th tuple then, for 4th tuple, the degree of similarity for the 4th (fuzzy) attribute is: \( \text{Min}(.33, .66) / \text{Max}(.33, .66) = .33 / .66 = .5 \). Now the same for the 5th (fuzzy) attribute is: \( \text{Min}(1, .6) / \text{Max}(1, .6) = .6 / 1 = .6 \). The net degree of similarity between 4th and 5th tuple for a set of attributes \( C \) (4th and 5th attribute) should be represented by the average of these two degrees for best explanation of the overall similarity. This value = \( \text{Avg}( .5, .6 ) = .55 \). So we can interpret that the candidates are .55 similar with respect to the query.

IV. Conclusion

The present paper basically works with the practical dimension of Codd’s relational data model and deals with the practical applications of the same enriched with fuzzy logic. Some critical notations of Codd’s relational Data model with fuzzy logic, had to be modified and customized a bit to fit into the practical algorithmic model. The concept of degree of similarity or likelihood is one of the utmost important aspect of knowledge generation from a database and lot of scopes of research remain in this area for the development of intelligent data processing and data mining. A lot of research on applying fuzzy logic is required in the area of database management systems to make the system capable of answering indefinite or unstructured queries.

The functional dependency and transitive dependency are two of the most crucial issues in relational database management.
systems and introducing the fuzzy degrees for those dependencies in an effective way can really bring revolution in the advancement of integrated and distributed data processing and data warehousing.

V. Acknowledgments

The author is thankful to IJATER Journal for the support to develop this document and grateful to the research guides Dr. Shankar Chandra Ghosh and Dr. Sreeyankar Acharya of West Bengal University of Technology and Budge Budge Institute of Technology, India, for providing with encouragement, support and guidance in carrying out this research work. The author is also thankful to his employer Icfai University Sikkim and its authorities for their constant support and encouragement in carrying out any research work in the subject area.

VI. References

Biographies

AUTHOR PROF. SABYASACHI CHAKRABORTY
received the M.Sc. degree in Computer Science, the M.Tech degree in Computer Science and Engineering and pursuing his Ph.D. in Computer Engineering from West Bengal University of Technology, India. The author also holds MBA degree in IT/Systems. Currently, He is a Senior Lecturer of I.T and Computer Science in Icfai University, Sikkim, India. His teaching and research areas include Computer Architecture, Relational Database System Development, Data Structure and algorithms, Programming logic, Fuzzy Logic and applications etc.. The author has a couple of published researched papers in international (IJSER) and national journal (NCPCC) and possesses a credit of performing and guiding over 20 software engineering and research projects. Author (Professor Sabyasachi Chakraborty) may be reached at scit75@gmail.com