

# A Heuristic for Combining Fuzzy Results in Multimedia Databases

M.V. Ramakrishna

S. Nepal\*

P.K.Srivastava

School of Computer Science & Software Engineering  
Monash University, Caulfield 3145, Australia  
rama@csse.monash.edu.au

\*CSIRO Mathematical and Information Sciences  
Locked Bag 17, North Ryde NSW 1670, Australia  
Surya.Nepal@csiro.au

## Abstract

This paper deals with the issue of combining fuzzy results obtained from two individual systems in multimedia query processing. Consider a query such as *retrieve images similar to  $I_1$  by color AND with associated text flower*. Suppose we have a color subsystem which can return a sorted list of images based on similarity with  $I_1$  and a text subsystem which can return a sorted list of images based on similarity with *flower*. Our task is to combine these two results. In other words, we need to evaluate the fuzzy combining function AND, giving a sorted list of the images.

Fagin has proved that the probabilistic complexity of the problem is almost linear (Fagin 1996), and our multi-step algorithm (Nepal & M.V.Ramakrishna 1999) is an optimal uniform algorithm (Fagin, Lotem & Naor 2001). In view of this inherent limitation, we investigated a non-uniform heuristic approach. In this paper, we discuss the problems of processing such queries, also referred to as aggregation queries in multimedia databases. The experimental results presented show that our “minimum depth first search” heuristic approach outperforms other uniform algorithms in general and by over 90% when the distribution of similarity values is not uniform.

**Keywords:** Multimedia databases, query processing in image databases, query optimisation in multimedia databases, CBIR systems.

## 1 Introduction

Advances in computing technologies have made multimedia databases comprising large collections of images, video and sound possible. In these applications, the queries are specified by giving the target values of the attributes. A small set of objects whose attributes best match the query attributes, are to be returned to the user in response to his query. An exact matching result is seldom expected. The result set presented is expected to be ordered on the degree of match to the query attributes. There is increasing interest in such problems by the database community. As Gravano states, such problems have come into the main stream database research (Gravano 2001).

Although the rest of our paper is written in the context of image databases, the results are applicable for any other multimedia databases. Content-Based Image Retrieval (CBIR) systems are examples of multimedia databases (Flickner, Sawhney, Niblack, Ashley, Huang, Dom, Gorkani, Hafner, Lee, Petkovic & Steele 1995). CBIR systems extract image features (usually high dimensional vectors) and index them using an appropriate structure such as R\*-trees (Beckmann, Kriegel, Schneider & Seeger 1990).

<sup>1</sup>S. Nepal was associated with RMIT University when much of this work was carried out  
Copyright ©2001, Australian Computer Society, Inc. This paper appeared at the 13th Australian Database Conference (ADC2002), Melbourne, Australia. Conferences in Research and Practice in Information Technology, Vol. 5, Xiaofang Zhou, Ed. Reproduction for academic, not-for profit purposes permitted provided this text is included.

Users can pose queries by giving an example image, color pallet, texture patterns, drawing sketches, etc. An example query is  $Q_1$ : “*retrieve images similar to  $I_1$  based on color*”. This is a single feature CBIR query, and  $I_1$  is an image identifier. Processing this query involves extraction of color feature from the example image  $I_1$  and making a similarity search on the corresponding index.

Suppose we had some text document also associated with the image (such as in a web page). Consider the query  $Q_2$ : “*retrieve images similar to  $I_1$  based on color AND associated with text “flower”*”. Processing user queries based on text alone is well known. How to efficiently combine the results of the two, color and text, to arrive at the required result for the query? In other words (for this example) how to efficiently evaluate the aggregating function AND, using fuzzy result set available from the individual attributes (color and text in our example). Fagin has analysed the complexity of uniform algorithms and has proven optimality of one such algorithm (Fagin et al. 2001). These algorithms have an almost linear upper bound in the probabilistic sense. Algorithms which are not uniform were not considered by Fagin, as they can only be based on a “wild guess” in his words (Fagin et al. 2001).

We present a multi-step algorithm, with minimum depth first search heuristic approach, for the efficient evaluation of aggregating fuzzy functions. Extensive experimental results presented with both real life data as well as randomly generated data show that the heuristic algorithm performs well in practice. An improvement of as much as 90% over the uniform algorithms, is achieved for many types of data. This leads to the conclusion that the multimedia query processor will benefit greatly by heuristics and statistics, similar that in relational query optimisation.

The rest of the paper is organized as follows. In the next section we provide a comprehensive background to the problem and elaborate on uniform algorithms. Our investigation into heuristic algorithms is presented in Section 3 along with experimental results. The last section draws some concluding remarks.

## 2 Background

We start with a brief description of the research reported in the related multimedia query optimisation. Chaudhuri and Gravano use the notion of optimal execution space for query optimisation (Chaudhuri & Gravano 1996). The main focus of their work is on the choice of indexes for optimal execution of queries with threshold filter conditions (such as “*retrieve images that are similar to an example image where color similarity is greater than 0.9 and texture similarity is greater than 0.8*”). Their results are not applicable for many multimedia database systems. A user query

seldom contains a threshold filter conditions. It is difficult for a user to provide a proper threshold condition without the knowledge of underlying similarity measures. A filter condition that does not have a threshold value or has a bad threshold value results in serial access (in the above query, the selectivity will be 1, if all the images in the database have texture similarity above 0.8).

Ortega et al. have dealt with processing of complex queries involving any number of query terms connected by AND, OR and NOT operations (Ortega, Rui, Chakrabarti, Mehrota & Huang 1998). Their techniques are implemented in the Multimedia Analysis and Retrieval System (MARS) they are developing. They considered only binary combining functions.

We now briefly describe some basic fuzzy concepts. We consider combining functions to evaluate  $m$ -ary queries. Combining functions evaluating conjunctive queries should satisfy the triangular norm (of conservation, monotonicity, commutativity and associativity) properties (Fagin 1996). Those evaluating disjunctive queries should satisfy triangular co-norm properties. We introduce the necessary notations and state the properties.

$x$ : an image

$q$ : a query term ( such as *color =red*)

$\mu_q(x)$ : the grade of  $x$  against the query  $q$

$t$ : combining function which combines the result of  $m$  grades  $V^m \rightarrow V$ , where  $V = [0, 1]$

$Q(q_1, \dots, q_m)$ : an  $m$ -ary query such as, ( $q_1$  AND ... AND  $q_m$ )

$\mu_{Q(q_1, \dots, q_m)}(x)$ : the grade of  $x$  against the query  $Q(q_1, \dots, q_m)$

Conservation property:

$$t(0, 0) = 0; \quad \forall g, t(g, 1) = t(1, g) = g \text{ for}$$

norm

$$\text{and } t(1, 1) = 1; \quad \forall g, t(g, 0) = t(0, g) = g \text{ for}$$

co-norm.

Monotonicity property:

$$t(g_1, g_2) \leq t(g_1', g_2') \text{ if } g_1 \leq g_1' \text{ and } g_2 \leq g_2'.$$

Commutative property:

$$t(g_1, g_2) = t(g_2, g_1)$$

Associative property:

$$t(t(g_1, g_2), g_3) = t(g_1, t(g_2, g_3))$$

There are several combining functions defined in fuzzy logic literature (Fagin 1996, Zadeh 1965). The commonly used *min* function for disjunction, and *max* function for conjunction are defined by Zadeh and obey the following rules. We will be using these two functions for most of our examples and experimental results.

Conjunction rule:

$$\mu_{q_1 \wedge q_2} = \min(\mu_{q_1}(x), \mu_{q_2}(x)).$$

Disjunction rule:

$$\mu_{q_1 \vee q_2} = \max(\mu_{q_1}(x), \mu_{q_2}(x)).$$

Negation rule:

$$\mu_{\neg q_1}(x) = 1 - \mu_{q_1}(x).$$

Fagin developed an algorithm given in Figure 1 for evaluation of combining functions, in the context of IBM's Garlic project, to combine image similarity with text (Cody, Haas, Niblack, Arya, Carey, Fagin, Flickner, Lee, Petkovic, Schwarz, Thomas, Roth & and 1995). We describe in terms of an example query, "retrieve 10 images similar to the given image  $I_1$  based on color AND texture". The problem is, how to efficiently evaluate the combining functions (AND in this case) to produce the required result. The combining functions could be simple, such as boolean AND and OR or more complex. We assume there exist individual index subsystems for each of the features (color and texture in this case). Each system

can return the next sorted element; That is the next most similar image to the given feature. Also, for any given image object, we can make a direct access to the subsystem to find its similarity with the given feature. Fagin's algorithm retrieves elements from individual attribute systems in sorted order until the intersection of the two has required number of elements. The final result is then evaluated. Fagin has proved that the database access cost of his (single-step algorithm) is  $O(N^{(m-1)/m}k^{1/m})$  with arbitrarily high probability, where  $N$  is the number of images in the database, and  $m$  is the number of attribute systems ( $m = 2$  for our example). The complexity is almost linear specially when  $m$  is larger. Our multi-step algorithm given in Figure 2 has been shown to be essentially same as the threshold algorithm by Fagin et. al. and proved to be optimal among uniform algorithms (Fagin 1996, Fagin et al. 2001, Nepal & M.V.Ramakrishna 1999).

Consider the example shown in Figure 3, where  $N = 5$  (the number of images in the database) and  $m = 2$  (the number of multimedia attributes in the aggregation query)). Suppose the user wants to retrieve best two matching images ( $k = 2$ ), the Fagin's algorithm requires 4 accesses from each subsystem to get the 2 common images. Only two accesses from texture attribute list and one access from color attribute list are needed, if we do not necessarily make equal number of accesses to each of the subsystems. We decided to investigate such non-uniform algorithms. Fagin refers to such an approach as *wild guess* (Fagin et al. 2001) and such algorithms are hard to find. We exploit the properties of combining functions and the nature of specific data to arrive at a non-uniform heuristic algorithm. We explain one of the heuristic approaches in the next section for an aggregate function *min*.

### 3 Heuristic Algorithms

As discussed above, non-uniform algorithms should be able to do better, if we some how ("wild guess") determine which system should be targeted for access. Often the similarity values are not uniformly distributed, or the ranges of the similarity values are not the same for different features. This observation is the basis of the heuristic approach to be used. The best heuristic is also likely to be dependent on the combining function used. After several trials, we decided to experiment with the following heuristic when the combining function is *min*.

The heuristic is to target the system which gave the minimum similarity value on the sorted accesses in the current iteration. In the next iteration, a sorted access is made to the targeted system and the the corresponding random accesses to the other systems. The threshold value is updated immediately (before making any more sorted accesses). The complete algorithm is given in Figure 4 and we refer to it as the "*minimum depth first search*" algorithm. We refer the readers to the cited literature for details of the basic single-step and multi-step algorithms. In the first step of the algorithm, we obtain the image( $x$ ) from each subsystem (e.g., color index), in the sorted order of the corresponding similarity( $\mu$ ). In the second step, for each image obtained in the first step, we find the similarity of the image from every other subsystem. In the third and fourth steps we compute the threshold grade and update the sets as indicated. The fifth step is the crucial one, where we determine the candidate subsystem to be targeted. Instead of continuing with uniform accesses, we start with one targeted subsystem. If we targeted correctly we save on the unnecessary accesses to other subsystems, as it happens in uniform algorithms.

Figure 1: Fagin’s single-step algorithm to evaluate combining functions

**Algorithm 1: Fagin’s Algorithm**

1. Sorted Access: From each subsystem  $i$  obtain pairs  $(x, \mu_{q_i}(x))$  in sorted order of  $\mu_{q_i}(x)$ . Let  $X_T^i$  denote the set of first  $T$  images output by  $i$ . Then obtain enough  $T$  images such that the set  $X_T^1 \cap X_T^2 \cap \dots \cap X_T^m$  contains at least  $k$  members.
2. Random Access: For each image  $x \in X_T^i$ , and  $x \notin X_T^j$ , find  $\mu_{q_j}(x)$  by a “random access” in each of the subsystem  $j$ .
3. Computation: Compute the grade  $\mu_Q(x) = t(\mu_{q_1}(x), \dots, \mu_{q_m}(x))$  for each image  $x$  that has been “visited”. Sort the images on their grades, and output the top  $k$  images.

Figure 2: Multi-step algorithm

**Algorithm 2: Multi-Step Algorithm**

1. From each subsystem  $i$  obtain pairs  $(x, \mu_{q_i}(x))$  in sorted order of  $\mu_{q_i}(x)$ .
2. For each image  $x$  returned by the subsystem  $i$  in the current iteration, find  $\mu_{q_j}(x)$  from every other system  $j$  by random access.
3. Compute the threshold grade,  $t_h^T$  for this iteration ( $T$ ),  $t_h^T = t(\mu_{q_1}(x_i^T), \dots, \mu_{q_m}(x_m^T))$ . Here  $x_i^T$  is the element retrieved by subsystem  $i$  in the current iteration  $T$ .
4. Compute the grade  $\mu_Q(x) = t(\mu_{q_1}(x), \dots, \mu_{q_m}(x))$  for all images  $x$  retrieved in this iteration ( $T$ ). Update  $Y$ , the set containing all images that have grade  $\mu_Q(x) \geq t_h^T$ .
5. If the set  $Y$  has  $k$  images and output the result  $Y$ , otherwise go back to step 1.

Figure 3: Sample data for an example query, “color = red AND texture = fine”

Color = “red”		texture = “fine”		Result	
objid	Grade	objid	Grade	objid	Grade
01	0.9	04	0.5	04	0.5
02	0.8	03	0.45	03	0.45
03	0.7	05	0.4	02	0.3
04	0.5	02	0.3	01	0.2
05	0.1	01	0.2	05	0.1

The working of all three algorithms for the example given in Figure 3 is shown in Table 1. The table shows the images retrieved for each of the sorted accesses by the subsystems, in the same order. The last column lists the number of sorted accesses at each stage. The Fagin’s algorithm makes four accesses to each subsystem, where as the multi-step algorithm needs only two accesses to each subsystem using the threshold values appropriately. The heuristic only makes one access to color system and two to texture system for a total of three accesses. After the first pair of accesses, the threshold is determined to be 0.5 and hence the next accesses is first made to texture system and not the color system (which returned 0.9 similarity earlier).

To evaluate the relative performance of the heuristic, we experimented by evaluating conjunctive queries using the CHITRA test bed system, with dif-

ferent data (Nepal, M.V.Ramakrishna & J.A.Thom 1998). We used the color features obtained by using the 13-color feature extraction algorithm as defined by Carson and Ogle (Carson & Ogle 1996). The Table 2 shows the summary of the comparative results obtained, showing the required number of sorted accesses with each of the three algorithms for retrieving  $k = 10$  best matches. The results shown are averages obtained from 500 different queries on the same image database as described before. We observe that for this data set, the heuristic based approach has remarkable improvement in the access cost. The sorted access cost of the heuristic algorithm is under 9% of the single-step cost, and under 15% of multi-step cost.

After checking our experimental details (for any errors) we were convinced of the remarkable improvement obtained with the heuristic. On further experimentation we found that this improvement is representative in general, but not always obtainable. We decided to study the heuristic further more systematically. Instead of using various real life data sets, we decided to use artificially generated data. Since this enables us to have better control on the experiments, we will be able to reach practically useful guidelines for the query optimiser. We used the random number generators to generate artificial similarity values with required distribution. We used the random number generators provided by the University of California, Los Angeles on their statistics home page (Dep n.d.). We generated random similarity values, in the range 0.0 to 1.0 with various distributions such as normal, uniform and exponential. This data was sorted and

used as sorted output of the attribute systems. The data set referred to as norm.2 corresponds to random similarity values which are normally distributed with mean of 0.2 and variance of 0.05. Similarly exp.2 corresponds to randomly generated similarity values with negative exponential distribution, having a variance of 0.2. The output of the random number generator was filtered to eliminate similarity values outside the valid range of 0.0 to 1.0.

The Table 3 gives a summary of the significant results comparing the sorted access costs of different algorithms. For example, the first row corresponds to results with two systems: one having uniformly distributed similarity values and the other having normally distributed (with mean of 0.2 and variance 0.05) similarity values. The sorted access cost for this case is 242 accesses under single-step algorithm, 26 accesses under multi-step algorithm which reduces to 14 accesses under the heuristic. We observe that the heuristic has enabled remarkable improvement in the access cost in this case. This improvement in the cost is dependent on the nature of the data set. The experiments corresponding to rows 5, 6 and 7 indicate that the performance gain of the heuristic is not as dramatic as the others. When the distribution of similarity values is the same among the different attribute systems, then the advantage of heuristic is only about 40 to 50%. When the distribution is different among the systems, then the gain of the heuristic algorithm is as much as 90% over the single-step algorithm. This is indicated by the rows 1 to 4, as well as the results with real life data presented earlier.

#### 4 Conclusions

In this paper, we addressed the problem of combining results from different attribute systems in multimedia databases. We have investigated a “minimum depth first search” heuristic approach with *min* combining function. Based on our experimental results, we summarise our results below.

- The proposed minimum depth first search heuristic approach performs significantly better than other two algorithms. Specially when the distribution of similarity values is not uniform, the improvement is over 90%.
- In our CHITRA database we observed that the distribution of similarity values is not always uniform. For such databases, the heuristic algorithm outperforms other two algorithms (refer Table 2).

Our experimental results show that the heuristic approach outperforms other uniform algorithms always. When the similarity values are not uniformly distributed the improvement is as high as 90%. We need to devise different heuristic algorithms for different data set. The application of a particular heuristic approach on a data set could be decided using statistical analysis of the data set based on the query optimization criteria. The proposed heuristic algorithm is applicable on the data set where the similarities values have non-uniform distribution in the systems. Thus we conclude that, the multimedia query processor will benefit greatly by heuristics and maintaining statistics of the data stored, similar to that in relational query optimisation.

#### References

Beckmann, N., Kriegel, H.-P., Schneider, R. & Seeger, B. (1990), The R\*-tree: An efficient and robust

access method for points and rectangles, in ‘Proceedings of the 1990 ACM SIGMOD International Conference on Management of Data’, Atlantic City, NJ, pp. 322–331.

Carson, C. & Ogle, V. E. (1996), ‘Storage and retrieval of feature data for a very large online image collection’, *Bulletin of the IEEE Computer Society Technical Committee on Data Engineering* **19**(4), 19–27.

Chaudhuri, S. & Gravano, L. (1996), ‘Optimizing queries over multimedia repositories’, *Bulletin of the IEEE Computer Society Technical Committee on Data Engineering* pp. 45–52.

Cody, W. F., Haas, L. M., Niblack, W., Arya, M., Carey, M. J., Fagin, R., Flickner, M., Lee, D., Petkovic, D., Schwarz, P. M., Thomas, J., Roth, M. T. & and, J. H. W. (1995), Querying multimedia data from multiple repositories by content: the Garlic project, Third Working Conference on Visual Database Systems (VDB-3), Lausanne, Switzerland, pp. 17 – 35.

Dep (n.d.), *Statisticals tables, plots and random numbers*. URL: [http:// ebook.stat. ucla.edu/ calculators/cdf/](http://ebook.stat.ucla.edu/calculators/cdf/).

Fagin, R. (1996), Combining fuzzy information from multiple systems, Proc. Fifteenth ACM Symp. on Principles of Database Systems, Montreal, pp. 216–226.

Fagin, R., Lotem, A. & Naor, M. (2001), Optimal aggregation algorithms for middleware, in ‘Proc 2001 ACM Symposium on PODS’. [http:// www.almaden. ibm.com/ cs/people/ fagin/pods01rj.ps](http://www.almaden.ibm.com/cs/people/fagin/pods01rj.ps).

Flickner, M., Sawhney, H., Niblack, W., Ashley, J., Huang, Q., Dom, B., Gorkani, M., Hafner, J., Lee, D., Petkovic, D. & Steele, D. (1995), ‘Query by image and video content: The QBIC system’, *Computer* **28**(9), 23–32.

Gravano, L. (2001), ‘Reminiscences on influential papers’, *SIGMOD Record* **30**(1), 55–56.

Nepal, S. & M.V.Ramakrishna (1999), Query processing issues in image(multimedia) databases, in ‘Fifteenth International Conference on Data Engineering (ICDE)’, March 23-26, Sydney, Australia, pp. 22–29.

Nepal, S., M.V.Ramakrishna & J.A.Thom (1998), A fuzzy system for content based image retrieval, in ‘Proceedings of the Second IEEE International Conference on Intelligent Processing Systems’, 4-7 August, Gold Coast, Australia, pp. 335–339.

Ortega, M., Rui, Y., Chakrabarti, K., Mehrota, S. & Huang, T. (1998), ‘Supporting ranked boolean similarity queries in MARS’, *IEEE Transaction on Knowledge and Data Engineering* **10**(6), 905–925.

Zadeh, L. (1965), ‘Fuzzy sets’, *Information and Control* **8**, 338–353.

Table 1: An illustration of the working of various algorithms to obtain the best two images for the query *color = "red" AND "texture=fine"*

Alg	color sys	texture	No. comm	threshold	k	No. sorted
Fagin	(I1, 0.9)	(I4, 0.5)	0	0.5	0	2
	(I2, 0.8)	(I3, 0.45)	0	0.45	0	4
	(I3, 0.7)	(I5, 0.4)	1	0.4	1	6
	(I5, 0.5)	(I2, 0.3)	3	0.3	2	8
Multi	(I1, 0.9)	(I4, 0.5)	0	0.5	1	2
	(I2, 0.8)	(I3, 0.45)	0	0.45	2	4
Heuristic	(I1, 0.9)	(I4, 0.5)	0	0.5	1	2
		(I3, 0.45)	0	0.45	2	3

Figure 4: Heuristic algorithm for combining function evaluation

**Algorithm 3: Minimum Depth First Search Approach**

1. For each subsystem  $i$  obtain pairs  $(x, \mu_{q_i}(x))$  in sorted order of  $\mu_{q_i}(x)$ .
2. For each image  $x$  returned by the subsystem  $i$  under sorted access, find  $\mu_{q_j}(x)$  from every other subsystem  $j$  by random access.
3. Compute the threshold grade,  $t_h$  for this iteration,  $t_h = \min(\mu_{q_i}(x_i), \dots, \mu_{q_m}(x_m))$ . Here  $x_i$  is the most recent element retrieved by the subsystem  $i$  under sorted access.
4. Compute the grade  $\mu_Q(x) = \min(\mu_{q_i}(x), \dots, \mu_{q_m}(x))$  for image  $x$  retrieved in this iteration. Update  $Y$ , the set containing all images that have grade  $\mu_Q(x) \geq t_h$ . If the set  $Y$  has  $k$  images, output the graded set  $\{(x, \mu_Q(x)) | x \in Y\}$ .
5. Find the candidate subsystem,  $i$ , which is the subsystem that has the minimum grade among the grades of the most recently retrieved elements from all the subsystems under sorted access, and retrieve the pair  $(x, \mu_{q_i}(x))$  from the subsystem  $i$  under sorted access and go to the step 2.

Table 2: Number of sorted access required by each algorithm with real image data

m	Single step	Multi step	Heuristic
2	208	104	48
3	567	303	75
4	996	608	88

Table 3: Number of sorted accesses required with randomly generated similarity values

No.	sys 1	sys 2	Single-step	Multi-step	Heuristic
1	unif	norm.2	242	26	14
2	norm.2	norm.6	192	20	11
3	unif	exp.05	208	40	21
4	exp.05	exp.1	206	86	44
5	exp.2	exp.2	212	178	90
6	unif	unif	220	198	127
7	norm.5	norm.5	202	186	94