A Fast Refined Search Method based on Filtering by Query Difference

Joohyoun Park, Giseok Choe, Jongho Nang
Dept. of Computer Science and Engineering, Sogang University, Seoul, Korea

***Abstract*— This paper proposes a novel method for fast refined search based on the relevance feedback in various content based multimedia retrieval systems. The basic idea of the proposed method is filtering of the irrelevant multimedia objects using the intermediate results at the first search and the difference between the original and the refined query. The meaning of the intermediate results is the distances to the original query for all objects in a database. For each object, the approximation of distance to the refined query can be simply calculated by subtracting the query difference from the distance to the original query. This mechanism helps to reduce both CPU and I/O time of the refined search because many irrelevant objects can be filtered out by the approximations of distance which are calculated by only one operation without reading the high dimensional objects. Also, it can be used with the filtering based indexing methods which are developed to resolve the problem called "curse of dimensionality". Upon experimental results, the refined search using the proposed method is about 5 times faster than the simple sequential search.**

I. Introduction

Recently, relevance feedback is used as one of the solutions of "Semantic Gap[1]" problem in various CBMR systems. With this mechanism, users may find multimedia objects from a database with best wishes but the elapsed time for similarity search may be increased in direct proportion to the number of searches as well. Actually, only a search process would make the burden for a CBMR system because the similarity in CBMR is usually measured by the distance between the high dimensional features extracted from multimedia objects.

To improve the speed of retrieval process, the various indexing methods have been widely researched. Some conventional data partitioning approaches such as R-tree[2] or R*-tree[3] can be used for solving this problem. However, their performances are known to drastically degrade as the number of dimensions increases because of the problem called "curse of dimensionality[4]". Some researches such as VA-file[5] and HBI[6] have been proposed to resolve this problem. These approaches are called filtering approach because initially it filters-out many irrelevant objects by scanning the compact approximations of objects after that it computes the distances among the query and remaining relevant objects in order to find the similar objects. According to [5][6], these approaches are about 1.5-3 times faster than the sequential search. However, these indexing approaches can not solve the problem that the search time increases in proportion to the number of retrievals by relevance feedback.

This paper proposes a method for fast refined search caused by relevance feedback when a filtering based indexing method is used. The relevance feedback for CBMR usually proceeds as the following steps: first, a CBMR system retrieves a set of multimedia objects similar to a query. Second, the user selects those that are relevant and irrelevant. Third, the system produces a modified query made by the user's feedback. Fourth, a new set of multimedia objects is retrieved using the new query. These steps are repeated with the modified queries until the retrieved results satisfy the user's needs. Therefore, using the distances calculated at the previous search could help the CBMR system to calculate the distance to the current modified query. In detail, for an object in a database, an approximation of the distance to the modified query can be calculated by subtracting the difference between the two queries from the distance to the previous query. This idea is supported by second triangle inequality. Based on this simple idea, an efficient similarity search algorithm is proposed in this paper. Upon experimental results, the refined search using the proposed method is about 5 times faster than the simple sequential search.

II. A Fast Refined Search Algorithms based on the Filtering by Query Difference

The major reason to improve the speed of retrieval in filtering based indexing methods such as the VA-file[5] or the HBI[6] is that they have the filtering process where many irrelevant objects are quickly filtered out by calculating an approximation of the distance to a query. This filtering technique is directly applied to the proposed search algorithms with relevance feedback. Now, let us present how to calculate the approximation of distance and how to search using this approximation.

A. Calculating Approximation of Distance

For simplicity, let the space of three objects q_0, q_1, and p be the 2-dimensional vector space under the L_2-norm. Then,$\overline{q_1p}$ is bigger than $|\overline{q_0p} - \overline{q_0q_1}|$ and less than $\overline{q_0p} + \overline{q_0q_1}$ by triangle inequality, because these points make a triangle. Note that the notation $\overline{q_1p}$ implies that L_2-distance between q_1 and p. This basic idea would be expanded to a fast refined search algorithm for CBMR.

Consider a multimedia database $\Lambda = \{o_i | 1 \leq i \leq n\}$, where o_i is the i^{th} object in n multimedia objects. Let the space of objects be a d-dimensional vector space Ω^d under the L_p-norm. That is, the i^{th} object, o_i, in Ω^d is defined as $< o_i^1, o_i^2, ..., o_i^j, ..., o_i^d >$, where o_i^j is the attribute value of the object o_i at j^{th} dimension.

Assuming that q_1 is the new query from the relevance feedback for the results of retrieval with q_0. Then, for an object

1-4244-1272-2/07/$25.00

o_i in Λ , the lower bound and the upper bound on L_p-distance between q_1 and o_i can be calculated as follows;

$$|L_p(q_0, o_i) - L_p(q_0, q_1)| \leq L_p(q_1, o_i) \leq L_p(q_0, o_i) + L_p(q_0, q_1) \quad (1)$$

In equation (1), $L_p(q_0, o_i)$ could be obtained without any extra calculation because the distances have been calculated at the previous search. Moreover, once $L_p(q_0, q_1)$ has been calculated, it could be used for all objects. That is, the lower bound and the upper bound on $L_p(q_1, o_i)$ can be calculated by only two operations with this mechanism. However, the equation (1) does not satisfy when the indexing methods based on filtering approach are used because the real distance is not calculated for most objects in the filtering approach. Therefore, the equation (1) has to be rewrote.

Let $A_p(a, b)$ be an approximation of distance (it means the approximate distance in HBI[6] and the lower bound on the real distance in VA-file[5]) between a and b in Ω^d, then $A_p(a, b)$ satisfies $A_p(a, b) \leq A_p(a, b)$. It implies;

$$A_p(q_0, o_i) - L_p(q_0, q_1) \leq L_p(q_0, o_i) - L_p(q_0, q_1) \text{ when, } A_p(q_0, o_i) \geq L_p(q_0, q_1) \quad (2)$$

From equation (3), the lower bound on $L_p(q_1, o_i)$ is still legal even if $L_p(q_0, o_i)$ is replaced with $A_p(q_0, o_i)$ in equation (1). Therefore, equation (1) must be revised as follows;

$$A_p(q_0, o_i) - L_p(q_0, q_1) \leq L_p(q_1, o_i) \text{ when, } A_p(q_0, o_i) \geq L_p(q_0, q_1) \quad (3)$$

B. A Fast Search Algorithms for r-Range and k-NN Search

The similarity search problems in the multimedia retrieval could be classified into two categories; one is r-range search which finds the objects whose distance from the query object is less than r, and the other one is k-NN (Nearest Neighbor) search that finds k objects with the smallest distance to the query object. Algorithms are shown in Fig. 2 and Fig. 3, respectively. Fig. 1 shows the common procedures for both algorithms.

Both have two filtering processes such as the first filtering process by triangle inequality and the second filtering process by an indexing method. In the first filtering process, every distance of new query is calculated by subtracting the distance between two queries from the distance saved at the previous searches in the first filtering process (**First_Filtering** procedure in Fig. 1). Since the approximation of distances are always less than the real distances as shown in the equation (1), the objects whose the approximation of distance to query is bigger than a threshold can be excluded from the set of final results. Note that the threshold is r for r-Range search and *kNNDist*, the maximal distance in the candidate set, for k-NN search. Of course, this process has to be skipped at the first search with the original query.

During the second filtering process, the distances are calculated by a filtering based indexing method for objects which are not filtered out yet in the first filtering process (**Second_Filtering** procedure in Fig. 1). This process could be skipped if no indexing method is being used. Finally, for the objects which survived from the above two filtering processes, L_p-distance to the query has to be calculated to get the final search results. The real distances and the approximations of distance which are calculated in the second filtering process should be saved in the additionally allocated memory for use of next search.

```
//q:the query object
//m:the m^th search in a session
//prevDist[i]:the array of distances between the previous query object and the i^th object
//prevQuery[i]:the array of search identifiers that indicate which query is used to calculate prevDist[i] for the i^th object
//queryDiff[m]:the array of distances between the current query object and the m^th query object
procedure FIRST_FILTERING(i, th)
    apxDist=prevDist[i]-queryDiff[prevQuery[i]];
    if apxDist ≥ th then
        return 1;
    else
        return 0;
    end if
end procedure
procedure SECOND_FILTERING(i, th, m)
    apxDist=Calculate_Ap_UsingIndex(i);
    SaveDistance(i, apxDist, m);
    if apxDist ≥ th then
        return 1;
    else
        return 0;
    end if
end procedure
procedure SAVEDISTANCE(i, distance, m)
    if prevDist[i]<distance then
        prevDist[i]=distance; prevQuery[i]=m;
    end if
end procedure
```

Fig. 1. Common procedures

III. EXPERIMENTS

To verify the efficiency of the proposed method, both r-range search and k-NN search were performed with HBI[6] and BFS which is a simple sequential search without any indexing methods. Three real data sets (R1, R2, R3) and a synthetic data set (S1) were used in our experiments are where the parameters are summarized in TABLE I. For the search with HBI, 6, 10, 15, 7 bitmaps are used for R1, R2 and R3, respectively. All experiments are tested under Microsoft Windows XP on an Intel Pentium 4(3.0GHz) with 1GB RAM.

```
procedure FILTERING_r-SEARCH(q, r, m)
    for all o_i such that 1 ≤ i ≤ n do
        if m > 0 && First_Filtering(i, r) then
            continue; //filter out i^th object
        else
            prevDist[i]=0;
        end if
        if Second_Filtering(i, r, m) then
            continue;
        end if
        tmpSet = tmpSet ∪ o_i;
    end for
    return (tmpSet);
end procedure
procedure EXACT_r-SEARCH(q, C_{r-search}, r, m)
    for all o_i such that o_i ∈ C_{r-search} do
        realDist = L_p(q, o_i);
        SaveDistance(i, realDist, m)
        if realDist < r then
            tmpSet = tmpSet ∪ {o_i};
        end if
    end for
    return (tmpSet);
end procedure
procedure r-RANGESEARCH(q,r,m)
    C_{r-search}= Filtering_r-Search(q, r, m);
    A_{r-search}= Exact_r-Search(q, C_{r-search}, r, m);
end procedure
```

Fig. 2. Fast refined search algorithm for r-range search

TABLE I

DATA SETS USED IN THE EXPERIMENTS

ID	data	distribution	# of Dim	# of objects
R1	MPEG7 Color Structure [7]	Extracted from real images	256	25,160
R2	MPEG7 Edge Histogram [7]		80	25,160
R3	HSV Color Histogram		32	68,040
S1	Synthetic data[8]	Uniformly	256	100,000

To make a new query by relevance feedback, the modification of Rocchio's method[9] has been used with α=0.5, β=0.25, and γ=0.25. Also, the half of the relevant objects is randomly picked from search results, and the rest of them are regarded as irrelevant objects. The final experimental results were the average of the results from the 100 query generated randomly because the experimental results could be highly dependent on the query. Also, they were the average of the results of r-range and k-NN search since their results were very similar in our experiments.

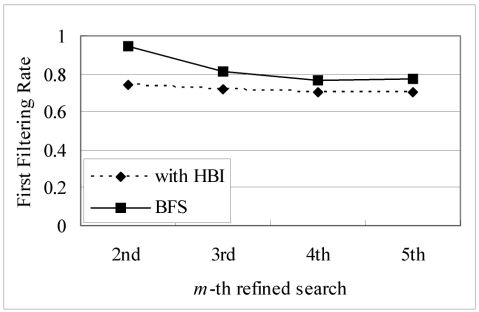
Fig. 4 shows the filtering rates in the first filtering processes and the total elapsed times of 5 retrievals in a search session for

```
//kNNDist:the maximum distance between the query object and the objects currently in C_{k-search}
//SelectMaxObject(C_{k-search}):a function that selects the object from C_{k-search} that has the maximum distance to query object
//FindMaxDist(C_{k-search}):a function that finds the maximum distance between the query object and the objects in C_{k-search}
procedure k-NNSEARCH(q, k, m)
    C_{k-search}=; //initially empty
    kNNDist = MaxDist; //initially maximum distance
    for all o_i such that 1 ≤ i ≤ n do
        if |C_{k-search}| < k then
            C_{k-search} = C_{k-search} ∪ {o_i}
        else
            apxDist = 0;
            if m > 0 && First_Filtering(i, kNNDist) then
                continue; //filter out i^th object
            else
                prevDist[i]=0;
            end if
        end if
        if Second_Filtering(i, kNNDist, m) then
            continue;
        end if
        realDist = L_p(q, o_i);
        SaveDistance(i, realDist, m);
        if realDist < kNNDist then
            o_max=SelectMaxObject(C_{k-search});
            C_{k-search} = C_{k-search} − {o_max} ∪ {o_i};
            kNNDist=FindMaxDist(C_{k-search});
        end if
    end for
end procedure
```

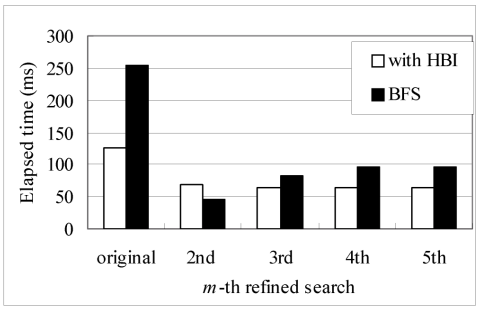
Fig. 3. Fast refined search algorithm for k-NN search

R1. A search session implies the set of retrieval processes to get the final results by relevance feedback including an original search. As shown in Fig. 4-(a), 70% of objects or over could be excluded from the candidate sets for all cases. It brings that the speed of an iterative search by relevance feedback is increased. Fig. 4-(b) shows that the total elapsed time of the second search is about 5 or 2 times faster than at the first search with BFS and HBI, respectively. Comparing the filtering rates of retrievals with HBI and BFS, those with BFS are rather higher than those of HBI on the same data sets. Also, as the number of the iterations of retrievals is being increased, the filtering rates of retrievals with BFS are slightly decreased while those with little changes of HBI. These results came from the correctness of the distances which were saved at the previous searches depends on the number of retrievals, and the existence of indexing method.

Fig. 5 shows the filtering rates of the first filtering process and

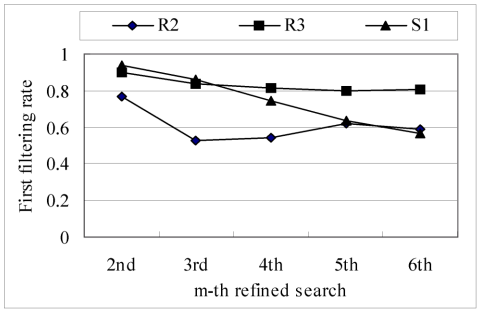


(a) The first filtering rate

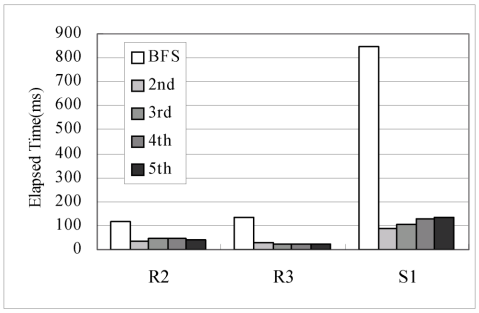


(b) Total elapsed time

Fig. 4. The first filtering rate and the total elpased time of the original search and its iterations of refined search with or without HBI for R1



(a) The first filtering rate



(b) Total elapsed time

Fig. 5. The first filtering rate and the total elpased time of the original search and its iterations of refined search with HBI for R2, R3, S1

the total elapsed time of the refined searches with HBI for the data sets in TABLE I. As shown in Fig. 5-(a), the filtering rates are lowered as the refined searches proceed. It is because the lower bound of the distance to the revised query is calculated by subtracting the relative difference between successive two queries from the distance calculated in the 1st search so that the new lower bound is always decreased as the search proceeds. Note that the filtering rates in 2nd and later searches are not high enough to get a satisfiable performance although its filtering time is negligible. It forces us to use the filtering algorithm using HBI to the objects that are not filtered-out by the first filtering process. As shown Fig. 5-(b), the times for 2nd and later search could be 2-2.5 times faster than the one for the original search with HBI, because a lot of irrelevant objects could be pre-filtered-out by the first filtering process with a negligible overhead and the remaining objects are tested and filtered-out again with HBI that helps to keep the filtering rate as high as possible. Compared to BFS(the original search without HBI), refined searches are at least 5 times faster.

IV. Conclusion

In this paper, we proposed a fast refined search method caused by relevance feedback, which reduces the number of operations required for calculating an approximation of distance. This method could filter out many irrelevant objects with an operation. Moreover, it could be applied to a CBMR system with a filtering based indexing method and also decrease the time of retrieval. Upon experimental results, 70-90% objects were filtered out by the proposed method on three sets composed of the visual features extracted from real images and one synthetic set with uniform distribution and their search time was averagely about 5 times faster than that of a simple sequential search. The proposed method could be used to build an efficient CBMR system with relevance feedback that guarantees a quick response time.

References

[1] R. Yates and B. Neto, *Modern Information Retrieval*. Addison Wesley, 1999.

[2] A. Guttman, "R-trees : A dynamic index structure for spatial searching," in *Proceedings of ACM SIGMOD International Conference on Management of Data*, pp. 47–57, 1984.

[3] N. Beckmann and H. Kriegel, "The R*-tree : An efficient and robust access method for points and rectangles," in *Proceedings of ACM SIGMOD International Conference on Management of Data*, pp. 322–331, 1990.

[4] S. Berchtold, C. Bohm, and H. Kriegel, "The Pyramid Technique : Towards breaking the course of dimensionality," in *Proceedings of ACM SIGMOD International Conference on Management of Data*, pp. 142–153, 1998.

[5] R. Weber, H. Schek, and S. Blott, "A quantitative analysis and performance study for similarity-search methods in high dimensional spaces," in *Proceedings of the International Conference on Very Large Database*, pp. 194–205, 1998.

[6] J. Park and J. Nang, "A hierarchical bitmap indexing method for content based multimedia retrieval," in *Proc.of EuroIMSA Int. Conf*, pp. 223–228, 2006.

[7] I. JTC1/SC29/WG11, *Information Technology Multimedia Content Description Interface-Part3: Visual*. 2001.

[8] T. Bozkaya and M. Ozsoyoglu, "Distance based indexing for high dimensional metric spaces," in *Proceedings of ACM SIGMOD Conference on Management of Data*, pp. 357–368, 1997.

[9] I. Ruthven and M. Lalmas, "A survey on the use of relevance feedback for information access systems," *The Knowledge Engineering Review*, vol. 18, no. 2, pp. 95–145, 2003.