Visual Feature Mining for Adapting Query-by-Example over Large Image Databases

Anicet Kouomou Choupo, Laure Berti-Équille
IRISA, Campus universitaire de Beaulieu
F-35042 Rennes Cedex, France
Phone: +0033-299847390 Fax: +0033-299847171
{akouomou,berti}@irisa.fr

ABSTRACT. *At the pixel level still images can be retrieved by similarity searching on global visual features such as color, texture, layout or shape. The similarity between any two images is computed by first computing their similarities based on individual features and then, combining them to obtain the overall similarity. The content-based retrieval system then uses and combines all the available low-level features whose computing cost can be prohibitive and it ranks the images according to how well they match the submitted query-by-example and it returns the best few matches to the user in a ranked result list, the most similar images first followed by the less similar ones. But, a subset of features could be sufficient enough to answer very quickly while offering an acceptable quality of results. In this article, we propose a method for automatically selecting the relevant visual features using clustering and association rules discovery techniques. The method is designed to allow query execution plans for speeding up the content-based retrieval over very large still image databases. We also present a strategy for adapting the query-by-example processing in order to propose instantaneous intermediary results that are progressively merged together until the end of the query execution over the whole database. This offers the advantage to the user, on one hand, not to wait during the similarity distance computing over the whole database before getting the results and, on the other hand, to allow him to stop at any time the query execution and still get a result. We evaluate our method by comparing query execution time and result quality with the sequential search on all the low-level descriptors of the image database. Our results show that we can get the final result in less than half time of a sequential search, which is a promising result.*

KEYWORDS*: content-based retrieval, feature mining, indexing, association rules discovery, progressive query*

1. Introduction

Content-based retrieval in very large multimedia databases exploits the indexation of various intermediate descriptions of the contents whose computing costs can be considerable. For each media type (video, text, audio, image...), the set of potentially relevant descriptors is often very large. A still image database can be represented at the level pixel in various ways, by local visual features or global descriptors of color, texture, layout or shape. Many descriptors are proposed in the literature and standardized (e.g., MPEG-7 [14]). Each feature is defined according to specific information, specialists of signal or image processing want to extract. However, it's sometimes useless to search the entire database by computing similarity distances over every available low-level feature, because the discrimination power of the features (and their filtering capacities) often depends on the size of the database (i.e., some features may be very discriminative for small databases and they can be advantageously used for filtering the results but they loose their property when they are flushed in millions of images).

Another major drawbacks of content-based retrieval based on sequential similarity search is that the query must be carried out in an exhaustive way over the whole database, because of the poor efficiency of indexation scheme of very large image databases described by high-size multidimensional vectors. The consequence is an unacceptable waiting time for the user, until all the images of the database are compared to the image-query with the similarity distance.

Moreover, any stop during the query execution has, as a consequence, the loss of all the retrieval information due to the lack of recovery strategies, forcing the user to restart the query again.

Contrary to traditional databases [9,15], few works have been carried out on optimization and adaptive query processing for multimedia content-based retrieval. As far as we know, several works on progressive query have been recently proposed in [13], but these are limited to the periodic execution of the same sub-query on portions of multimedia databases (images or videos) without targeting, prioritizing nor scheduling the data subsets to process, nor merging results from heterogeneous descriptors. Concerning the scalability of the approach, neither the size of the databases nor the size of the database portions targeted by the sub-queries were mentioned [13].

In our paper, we propose a strategy for the automatic selection and scheduling of the search criteria (i.e., visual features) that are relevant to a given query-by-example on a large still image database. We use techniques of clustering and association rules discovery on descriptor values. We also present how the query-by-example processing can be adapted to propose instantaneous and intermediary results which are progressively merged with the advantage, for the user, on one hand, not to wait until the whole database has to be processed for finding the images that are the most similar to the submitted image-query and, on the other hand, to allow him to stop the current execution of the query. We evaluate our method on a still image database by comparison with the sequential search over every available feature followed by the fusion of the results.

The rest of the paper is organized as follows: in section 2, we present the related works on association rules discovery in the context of indexing and searching images. We present then, in section 3, our method of automatic selection of the search criteria, and then we formalize, in section 4, the progressive adaptation of query-by-example execution plans. In section 5, we describe the experiments that validate our approach. Section 6 offers concluding remarks.

2. Related works

Content-based retrieval consists in searching the nearest neighbors of a query-by-example among all the available images indexed by multidimensional descriptors (e.g., MPEG-7 image descriptor vectors with 7 to 84 dimensions). Indexation techniques that are either based on data partitioning (such as R-Tree [10]) or on space partitioning (such as k-d-Tree [3] or GridFile [17]) remain adapted to low-size vectors. The performances are very quickly deteriorating when the dimension of feature vectors becomes large. New indexation scheme have to be defined and adapted to the vectors of large dimension as well as approximate searching methods to improve the query performances.

In this context, several methods work out probabilistic models to control the precision of results [4]. And other works exploit the technique of association rules discovery in the context of multimedia content-based retrieval [6,20]. Djeraba uses an indexing technique that combines the construction of the clusters and the determination of association rules [6]. Several sub-groups of images (clusters) are thematically classified with labels (such as "animal", "plants"...) and the aim is to characterize and labellize these image groups by association rules. At the query time, an analysis of the query starting from the association rules drives the choice of the images sub-groups for the sequential search. The work of Djeraba aims at the improvement of the quality of results. The focus is not the fusion of results from several visual descriptors. The query time is not evaluated. Moreover, the approach imposes the creation of a dictionary (thesaurus) for the semantic labellisation of the clusters.

In our approach, the number as well as the type of descriptors is not limited and no labelling is required contrary to [6], because our objective is to technically improve the query performances at the pixel level without no ambition on the semantic level [21]. Our strategy combines the computing of clusters and the extraction of association rules. Thus, it's entirely non-supervised and is appropriate for very large databases.

Since Apriori [1], many algorithms for association rules discovery have been proposed in the literature, they are very effective for classical transactional databases whose structure is well defined. But, their application to the images requires some adaptations. An idea consists in transforming the image descriptions and in applying the traditional techniques of rule discovery. An image can be identified as a transaction and the various regions composing it as objects [18]. Under certain conditions, the repetition of objects may carry information. The traditional definition of association rules does not take into account the possibility of object repetition and it was necessary to develop new formalisms. Thus, Zaïane et al. [18] proposed a definition of association rules with recurrent objects as the following implication: $\alpha P_1 \wedge \beta P_2 \wedge \ \ldots \ \wedge \gamma P_n \rightarrow \delta Q_1 \wedge \lambda Q_2 \wedge \ldots \wedge \mu Q_m$

where P_i and Q_j are image descriptors ($i \in [1..n]$ and $j \in [1..m]$) and $\alpha, \beta, \gamma, \delta, \lambda, \mu$ are integers indicating the number of occurrences of the descriptors. The *MaxOccur* algorithm described in [20] is an adaptation of *Apriori* for the extraction of association rules with recurrent objects.

In our approach, we propose an application of the association rules for the indexing and query-by-example over a still image database. Our approach lies within the scope of the transformation of multimedia data and the application of the traditional techniques of association rules discovery. We use several MPEG-7 image descriptors and the images are gathered into clusters for each descriptor. We consider an image as a transaction and a cluster identifier of each descriptor as an attribute. The problem of recurrent objects does not arise since the clusters are identified in a single way for each descriptor.

3. Features Mining and Scheduling for Indexation and Retrieval

Let us now present the principle of our method of automatic selection of the searching criteria for query-by-example on large still image databases. It includes: 1) the indexing step, based on the clustering and the discovery of association rules between the visual descriptors, 2) the retrieval step, based on the selection and use of rules in order to choose the first rank descriptors as searching criteria (i.e., the most relevant for the query (for the result quality) and efficient for the performance (the query time)).

3.1. Indexing Process

Off-line indexing process is based on the organization of the still image database by computing the set of descriptor values for the characterization of the images. The image database is then gathered in clusters for each type of computed descriptors. Consider the image database B with N images noted $I_1, I_2, \ldots, I_N$. Let $D = \{d_1, d_2, \ldots, d_m\}$ the whole set of m available descriptors. We note $((n_j(I_i), d_j)$ the cluster of the descriptor d_j to which the image I_i belongs and B, the image database such as:

$$B = \{I/I = \{(n_1(I),d_1),(n_2(I),d_2),\ldots,(n_m(I),d_m)\}\}$$

We apply the Apriori algorithm [1] for discovering association rules on B and obtaining the implications between clusters of descriptors. We worked with a database of 30411 still images described by the MPEG-7 descriptors: *ColorLayout (CLD), ScalableColor (SCD), HomogeneousTexture (HTD), EdgeHistogram (EHD)* and *RegionShape (RSD)*. At the end of the indexing process, we obtain a set of clusters for each descriptor d_i and a set of association rules obtained with the *Apriori* algorithm which identify the relations between different the clusters of the database B such as the following examples:

```
r1: (19,HTD)∧(0,SCD) -> (29,CLD) <0.1,66.1>
r2: (23,RSD) ∧(1,HTD) ∧(23,EHD) -> (8,CLD) <0.1,55.2>
```

where `<0.1,66.1>` are respectively the support and the confidence values of the rule *r1*, expressed as percentages (idem with the support and confidence `<0.1,55.2>` of *r2*).

Intuitively, the semantics of a rule r selected for a query-by-example noted *Iq* is: the nearest neighbors of *Iq* that belong to all the clusters that are in the left hand side of r, probably belong also to the clusters that are in the right hand side of r.

We then use the most relevant association rules to schedule the cluster processing, starting first with clusters that have the best selectivity and are in the left hand side of selected rules.

3.2. Content-based Retrieval Process

When the user submits a query-by-example to the system, her goal is to find all the images similar to the image-query according to several visual criteria. The user may know how to choose the visual descriptors as searching criteria. But, generally she does not have any idea of what are the most discriminative descriptors for her query. In many cases, it's useless to search images on all the criteria (descriptors) already computed in the database. Our objective is to propose and to prioritize the descriptors that are the most relevant, especially in the context of progressive queries that can provide intermediate and instantaneous results during the query execution and before the end of the exhaustive processing of the database.

The end of what we call a searching phase corresponds to the presentation of an intermediate result. Given an image-query *Iq*, the system computes, for each descriptor d_j, a measure of selectivity of each cluster (defined in section 4.1.), then determines, for the given searching phase, the Top M clusters to start with the similarity searching and retrieves the nearest neighbors of I_q.

Consider *TopC(M, i)* the set of Top M clusters of better selectivity selected at the i^{th} searching phase. The purpose is to exploit the association rules between the clusters of *TopC(M, i)* and those selected during the previous searching phases.

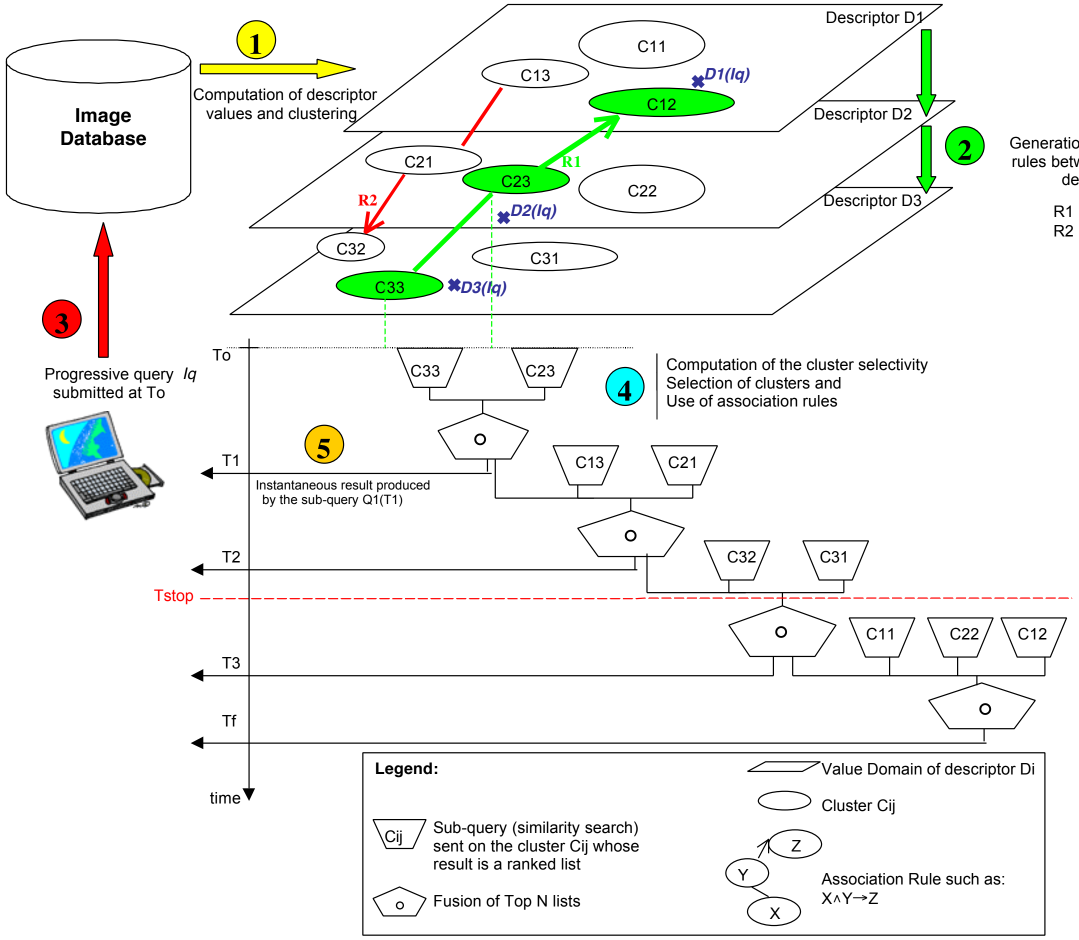
Consider *R* the set of association rules computed by the indexing process. For any $r \in R$, we note C_r, the set of clusters belonging to the expression of the rule (i.e., present in the right and left hand sides of *r*). Our assumption is as follows: for the rules whose support and confidence are higher than a given threshold, for each searching phase, a rule *r* is selected if $C_r \subseteq \bigcup_{i=1..k} TopC(M,i)$.

For the query processing using the association rules, the strategy is to first ignore the clusters present in the right hand side of the selected rules and their corresponding descriptors. In the previous example, if *r1* and *r2* are selected, the query execution will first ignore clusters #29 and #8 of the descriptor *CLD*.

4. Adaptating Query-By-Example Processing

Content-based retrieval generally requires the exhaustive processing of the whole image database before providing a result to the user. Our objective is here to improve the query processing while proposing: 1) a strategy of query-by-example execution plans which defines an optimal scheduling of the cluster to process with the similarity search, 2) the possibility to provide to the user instantaneous results which are progressively refreshed and merged during the query execution, 3) the possibility for the user to stop the current query execution if she's satisfied with the previously retrieved intermediary result.

The general principle is presented in the Figure 1. The indexing process is represented by the two first step (1) consisting in the computation of the descriptor values and the clustering and (2) for the generation of association rules. At the step (3), a query-by-example I_q is submitted at initial time T_o, and it is decomposed into several sub-queries targetting specific portions of the database (i.e. the Top M clusters) selected by the selectivity measure, and the priorities given by the association rules extracted beforehand at the indexing step (step (4)). The selectivity measure of a cluster depends on the size and the density of the cluster, and on its proximity to the description of the image-query relative to a given descriptor. In Figure 1, the cluster C_{12} is the nearest to $D_1(I_q)$, the description of the image-query relative to the D_1 descriptor. At step (5), the sub-queries provide instantaneous results that are progressively merged, updated and refined during the total query execution time over the whole image database. The advantage is here to provide instantaneous results to the user during the query execution. Moreover, the user will be able to stop the current execution (at T_{stop}) keeping the previous intermediate results. The intermediate results on each cluster of descriptors are progressively merged, and sent to the user (at T_1, T_2, T_3). When the whole database has been processed, the last instantaneous result is merged with the previous instantaneous results and is sent as the final result to the user at T_f (step (5)).



1
Computation of descriptor values and clustering
Image Database
Descriptor D1
Descriptor D2
Descriptor D3
C11
C12
C13
C21
C22
C23
C31
C32
C33
D1(Iq)
D2(Iq)
D3(Iq)
R1
R2
2
Generation of association rules between clusters of descriptors
R1 : C33,C23 →C12
R2 : C13,C21 →C32
3
Progressive query Iq submitted at To
To
4
Computation of the cluster selectivity
Selection of clusters and
Use of association rules
5
Instantaneous result produced by the sub-query Q1(T1)
T1
T2
Tstop
T3
Tf
time
Legend:
Value Domain of descriptor Di
Cluster Cij
Cij
Sub-query (similarity search) sent on the cluster Cij whose result is a ranked list
Fusion of Top N lists
Association Rule such as: X∧Y→Z
X
Y
Z

Figure 1. *Example of a Progressive Query Execution Plan*

4.1. Formal Definitions

Image Database

Consider k the fixed number of clusters per descriptor and D the set of descriptors. The image database B is defined as the union of the k clusters C_{ij} of each descriptor d_i available in the database such as:

$$B = \bigcup_{j} C_{ij} \quad \forall d_i \in D, i=1..card(D), j=1..k \qquad (1)$$

An image belongs to the database if its description for each descriptor present in the database belongs to a cluster.

$$I \in B \text{ if } \forall d_i \in D, \exists C_{ij}, d_i(I) \in C_{ij}$$

Query-by-example

A query-by-example $I_q = (n_q, \mathsf{O}_q, \mathsf{W}_q, \mathsf{D}_q)$ for the multidimensional space S consists of the following information:

- The number n_q of points in I_q
- A set of n_q points $\mathsf{O}_q = \{O_q^{(1)}, \ldots, O_q^{(n_q)}\}$ in the d_S-dimensional feature space S

- A set of n_q weights $W_q=\{w_q^{(1)},\ldots,w_q^{(n_q)}\}$, the i^th^ weight $w_q^{(i)}$ being associated with the i^th^ object $O_q^{(i)}$ $\left(1\geq w_q^{(j)}\geq 0, \sum_{i=1}^{n_q} w_q^{(i)}=1\right)$
- A distance function D_q, which, given a point O in the space S, returns the distance between the query and the point. We assume D_q to be a weighted L_p metric, i.e., for a given value of p, the distance between two points T_1 and T_2 in S is given by:[1]

$$D_q(T_1,T_2)=\left[\sum_{j=1}^{d_s}\mu_q^{(j)}\left(\left|T_1[j]-T_2[j]\right|\right)^p\right]^{1/p} \tag{2}$$

where $\mu_q^{(j)}$ denotes the weight associated with the i^th^ dimension of S $\left(1\geq\mu_q^{(j)}\geq 0, \sum_{j=1}^{d_s}\mu_q^{(j)}=1\right)$. D_q specifies which L_p metric to use (i.e., the value of p) and the values of the dimension weights.

We use the point distance function D_q to construct the aggregate disance function $\mathsf{D}_q(I_q,O)$ between the multiple query objects O_q and the object O (in S). $\mathsf{D}_q(I_q,O)$ is the aggregate of the distances between O and the individual objects $O_q^{(i)}\in\mathsf{O}_q$:

$$\mathsf{D}_q(I_q,O)=\sum_{i=1}^{n_q} w_q^{(i)} D_q\left(O_q^{(i)},O\right) \tag{3}$$

We use a weighted sum as the aggregation function but any other function can be used as long as it is weighted and monotonic.

Searching phase

A i^*th*^ searching phase corresponds to the similarity search processing over the set of i^*th*^ rank clusters for each descriptor.

Selectivity of clusters

The selectivity of the cluster C for a given query-by-example I_q is a function combining the proximity of the cluster to the image-query (previously defined as the distance function D_q), the size and the local density of the cluster.

Selectivity measure of a cluster C depending on query-by-example I_q, noted *Selectivity(C, I~q~)* is defined as follows:

$$Selectivity(C,I_q)=\alpha\left(1-\frac{dist(I_q,C)}{\max_j(dist(I_q,C_j))}\right)+\beta\left(1-\frac{size(C)}{\max_j(size(C_j))}\right)+\gamma\ .\ density(C)$$

$$with\ \ \alpha\geq 0,\ \beta\geq 0,\ \gamma\geq 0\ \ and\ \ \alpha+\beta+\gamma=1$$

size(C) is the size of cluster *C, dist(I~q~,C)* is the distance of I_q to the centroid of cluster C and density(C) is the concentration of cluster C at it centroid.

We evaluate the concentration using the Gini index and the analytical formula is given by:

$$density(C)=\frac{\sum_{i=1}^{size(C)}\sum_{j=1}^{size(C)}\left|dist(I_i,C)-dist(I_j,C)\right|}{2\times(size(C)-1)\times\sum_{i=1}^{size(C)}dist(I_i,C)}\quad \text{with } I_i,I_j\in C \tag{5}$$

dist(I~i~,C) is the distance of I_i to the centroid of cluster C for $I_i\in C$.

Parameters α, β, γ are positive and chosen such as $0\leq Selectivity(C, I_q)\leq 1$.

The priority for processing a cluster C is high for a given query-by-example I_q if *Selectivity(C, I~q~)* is close to 1.

[1] Note that this assumption is general since most commonly used distance functions (e.g., Manhattan distance, Euclidean distance, Bounding Box distance) are special cases of the L_p metric.

Association rule between clusters

An association rule between several clusters of descriptors, noted C_{ij}, $C_{i'j'}$, $C_{i''j''}$, is defined as follows:

$$r : C_{ij} \wedge \ldots \wedge C_{i'j'} \rightarrow C_{i''j''} < supp, conf > \text{ with } i,i',i''=1..card(D), \quad i \neq i' \neq i'' \text{ and } j,j',j''=1..k \tag{6}$$

Semantics of the rule r is: an image whose description relative to the descriptors d_i and $d_{i'}$ belongs respectively to the clusters C_{ij} and $C_{i'j'}$ in the left hand side of the rule r, probably belongs to the cluster $C_{i''j''}$ (in the right hand side of the rule r), with *supp* and *conf* as support and confidence values for r.

Selection of association rules between clusters

For a given query-by-example *Iq*, the clusters are selected depending on their selectivity measure at the i^{th} searching phase. For the i^{th} searching phase of *Iq*, the set of the Top M clusters is noted: $TopC_M^i(I_q)$.

The corresponding set of selected descriptors is noted: $TopD_M^i(I_q)$ after rule selection.

An association rule is selected for query-by-example I_q if its support and confidence are higher than given thresholds and if the set of clusters composing the rule, noted *Cr*, is such as:

$$C_r \subseteq \bigcup_{i=1..k} TopC_M^i(I_q) \tag{7}$$

When a rule is selected, the query processing will first ignore clusters that are in the right hand side of the rule.

Query-by-example execution plan

A logical query execution plan P for the query-by-example I_q is composed of the list of clusters C_{ij} to process for retrieving the most similar image according to all the descriptors D of the database. A query execution plan P for I_q is composed of several subplans as follows:

$$P(I_q,D) \leftarrow P_1(I_q,d_1) \circ P_2(I_q,d_2) \circ \ldots \circ P_i(I_q,d_i) \circ \ldots \circ P_m(I_q,d_m) \tag{8}$$

$P_i(I_q,d_i)$ corresponds to the processing of the clusters for the descriptor d_i.

There are $\prod_i (n_c(d_i))!$ possible plans with $n_c(d_i)$, the number of clusters for the descriptor d_i.

$P(I_q, D)$ can be rewritten such as:

$$P(I_q,D) \leftarrow P_1(I_q,\{C_{11},\ldots,C_{1k}\}) \circ P_2(I_q,\{C_{21},\ldots,C_{2k}\}) \circ \ldots \circ P_m(I_q,\{C_{m1},\ldots,C_{mk}\})$$

or as minimal query subplans such as:

$$P(I_q,D) \leftarrow P_1(I_q,C_{11}) \circ \ldots \circ P_k(I_q,C_{1k}) \circ P_{k+1}(I_q,C_{21}) \circ \ldots \circ P_{2k}(I_q,C_{2k}) \circ \ldots \circ P_{k+m}(I_q,C_{m1}) \circ \ldots \circ P_{mk}(I_q,C_{mk})$$

with k clusters per descriptor and m descriptors.

Progressive query-by-example

We suppose a partial order on the initial, intermediary and final times during the execution of the query such as: $T_0 < T_1 < T_2 < \ldots < T_f$.

A progressive query, noted $\vec{Q}(T_0, freq)$, submitted at initial time *To*, and finished at final time *Tf*, is a query-by-example whose intermediary results produced by each sub-queries (called instantaneous results) are regularly merged and sent to the user at times $Ti \in]T_0, T_f[$ as and when the query is executed depending on a fixed frequency, noted *freq*.

for freq = 1,

$$\vec{Q}(T_0,1) = Q_1(T_1) \circ Q_2(T_2) \circ \ldots \circ Q_{Nc}(T_f) \text{ with } Nc\text{, the total number of clusters.}$$

A minimal sub-query is a similarity search over one cluster of descriptor.

Frequency of a progressive query

The frequency of a progressive query depends on the required number of intermediate results and of the total number of clusters to process. The frequency is the ratio of the number of minimal sub-queries (i.e. the number of clusters) and the required number of instantaneous results such as:

$$freq=\frac{Nc}{Ni+1} \tag{9}$$

with Nc : the number of minimal sub-queries (i.e., the total number of clusters)
and Ni : the number of required instantaneous results ($1 \leq Ni \leq Nc-1$).
For a frequency $freq = 1$, the instantaneous results are sent to the user after the fusion of the results of each minimal sub-query. The corresponding progressive query is:

$$\vec{Q}(T_0,1)=Q_1(T_1)\circ Q_2(T_2)\circ\ldots\circ Q_n(T_f)$$

In Figure 1, the frequency is $freq = 9/4$, with 9 minimal subqueries and 3 intermediate results. The instantaneous results are sent to the user after the fusion of the results of two minimal sub-queries. Among the four possible execution plans that correspond to the progressive query, the one represented in the Figure 1 is:

$$\vec{Q}(T_0,9/4)=\Big(\Big(\Big((Q_1\circ Q_2)(T_1)\circ(Q_3\circ Q_4)\Big)(T_2)\circ(Q_5\circ Q_6)\Big)(T_3)\circ(Q_7\circ Q_8\circ Q_9)\Big)(T_f)$$

Instantaneous Result

An instantaneous result of a progressive query $\vec{Q}$, noted *InstantResult(Q)*, is the result of a sub-query on a portion of the database ; it's obtained by the fusion of the previous instantaneous results.

$$(InstantResult(Q_i))^{new}= (InstantResult(Q_{i-1}))^{old} \circ InstantResult(Q_i)$$

Final Result

The final result of a query is obtained:

- when the whole image database has been processed to answer to the initial query-by-example ; that is when all the clusters have been processed by similarity search:
  at Tf, $FinalResult(Q) = (InstantResult(Q_{Nc}))^{new}$ *with Nc, the total number of clusters*
- when the query execution has been volontarly stopped by the user
  at T_{stop}, $FinalResult(Q)= (InstantResult(Q_i))^{new}$ *with* $T_{stop} \in \left]T_0,T_f\right[$

4.2. Generation of Query Plans

For a given query-by-example, the system will generate all the possible query execution plans.

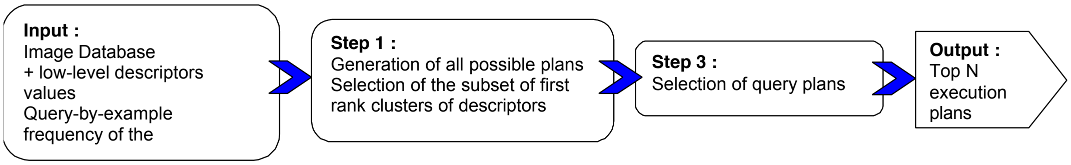


Figure 2. *Planification of A Progressive Query Execution*

In the example presented in Figure 1, the query concerns the set of three available descriptors (d_1, d_2, d_3) with three clusters per descriptor. As examples, here are three possible execution plans of the progressive query:

1. The first plan takes sequentially the clusters of each descriptor.
   P1(d1,d2,d3) <- (C11)(T_1) o (C12)(T_2) o (C13)(T_3) o (C21)(T_4) o (C22)(T_5) o (C23)(T_6) o (C31)(T_7) o (C32)(T_8) o (C33)(T_9)
2. The second plan is driven by the discovered association rules.
   P2(d1,d2,d3) <- (C33 o C23)(T_1) o (C13 o C21)(T_2) o (C32 o C31)(T_3) o (C11 o C22 o C12)(T_4=T_f)
   (represented in Figure 1)
3. The third plan uses in priority the proximity of the clusters to the query-by-example I_q.
   P3(d1,d2,d3) <- (C12 o C23 o C33)(T_1) o (C11 o C22)(T_2) o (C13 o C21)(T_3) o (C32 o C31)(T_4=T_f)

4.3. Selection of Query Plans

Consider the query-by-example I_q submitted to the image database and P_Q the set of all possible plans for the query, the search space for retrieving the N top plans is the set of size N of all the sub-sets of P_Q.

$$\boldsymbol{S}(Q) = \{ P' \subseteq P_Q \mid |P'| = N \}$$

To retrieve the N top plans including the cluster selection step, we adapted the Branch & Bound algorithm [16]. And we introduce the notion of upper bound limit for the plan P defined by the function *upperbound(P)* such as :

$$upperbound(P) \geq \max_{P' \supseteq P}[selectivity(P')] \tag{10}$$

For more details on the algorithm and the related experiments, we refer readers to an extended version of the paper in [11].

4.4. Fusion of Instantaneous Results

For a given searching phase, the similarity search is first done over a subset of descriptors selected among the available ones such as $\{d_1,d_2,\ldots,d_{m1}\} \subseteq D$ with $m_1 \leq card(D)$. Consider l_j the list of query results for the descriptor d_j ranked in a decreasing order of similarity distance ($1 \leq j \leq m_1$). For the final result merged from the ranked result lists of different descriptor, we define a fusion score for the image I such as:

$$S_f(I) = \frac{1}{2m_1}\left(\sum_{j=1}^{m_1}\left(f_{d_j}(I) + S_{d_j}(I)\right)\right) \text{ with } S_{d_j}(I) \in [0,1] \text{ and } f_{d_j}(I) = \begin{cases} 1 & if \quad I \in l_j \\ 0 & if \quad I \notin l_j \end{cases}. \tag{11}$$

$S_{d_j}(I)$ is the score of I for the descriptor d_j. It's proportional to the similarity distance between the query-by-example I_q and I. Similarity between two images is the distance between the respective multidimensional vectors of the descriptors. We use the Euclidian distances proposed by MPEG-7 in the definition of the descriptors. Merging Top N lists from heterogeneous descriptors (with different vectorial dimensions) is a difficult problem and we study other fusion functions [7,5] to implement in our experiments.

5. Experimental Results

The method of automatic selection and ranking of search criteria (i.e., visual descriptors) for the query-by-example processing presented above is implemented in C++ under Linux. We currently work with a database of 30411 still images but we will soon consider the processing of several hundreds of thousands of images.

Our experimental process follows two steps. At the indexation step, we exploit MPEG-7 descriptors for color (*ColorLayout*, *ScalableColor*), for texture (*HomogeneousTexture, EdgeHistogram*) and for form (*RegionShape*). For each MPEG-7 descriptor, we gather the images in clusters with a K-means-like algorithm. In this work, we observed that the very large clusters (containing lots of images) mostly appear in the right part of the association rules with high percentage of confidence. Compared to other works [8], we adopt an experimental approach in which the homogeneity of the size of the clusters determines the choice of the total number of clusters to consider. That is, we don't fix arbitrarly the number of clusters but we determine it depending on the number of robust association rules we consider (Experiment 1).

At the retrieval step, we study the contribution of association rules (Experiment 2) and the selection of clusters based on the selectivity measure (Experiment 3) we previousliy defined in section 4.1. In these experiments, we use 100 images as queries-by-examples, each belonging to the database and we search for the Top 15 nearest neighbors. We measure the average time and the average result quality. The result quality at a searching phase is defined as the number of common images in both cases: the partial and intermediary result and the final result obtained at the end of the sequential and exhaustive search over the whole image database with all the available descriptors. In this measurement of quality, we are not focused on the order of the retrieved images belonging to the result list but only on their presence/absence that could be satisfactory enough for an end-user.

5.1. Experiment 1: Choosing indexing parameters

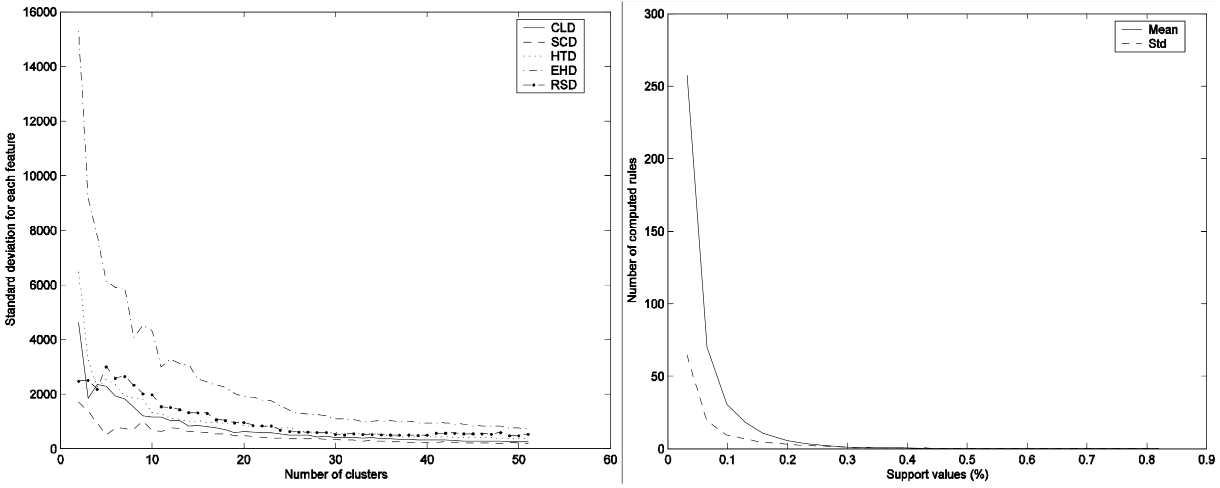
The goal here is to study the parameters of indexing such as the number of clusters, the thresholds of the support and confidence. We calculate, for a fixed value of the number of clusters, the standard deviation of the size of all the clusters. This experiment shows the size of clusters compared to the average number of images per cluster. Figure 3a shows the variations of the standard deviation compared to the number of clusters for each of the five descriptors we used. We observe that, when the number of clusters is lower than 10, the clusters are very disparate.

The number of cluster must be sufficiently large, so that the size does not deviate much from the average. This is true for all the descriptors that have the same distribution of the images in the clusters. For this reason, we can choose the same number of clusters for all the descriptors.

In our experiment over 30411 images, we fix the number of clusters for each descriptor such as 30 clusters per descriptor. The association rules are computed over the clusters of descriptors with the algorithm *Apriori* [1]. The threshold of confidence is fixed at 50 % to guarantee the robustness of the rules. The threshold of the support is chosen experimentally. Figure 3b gives the variations of the number of rules computed according to the threshold of the support. We obtain rules with very weak support in general. A high number of clusters involves the reduction of the support of the rules.

Suppose that all the 30 clusters contain exactly the same number of images, i.e. $\frac{30411}{30}$, then the support of a rule will be, at the very best, about $\frac{\left(\frac{30411}{30}\right)\times 100}{30411} \approx 3{,}33\%$ under the assumption that the right and left hand sides of the rules contain the same images.

The rules give nevertheless information about the number of images that are found in the same cluster for several descriptors. Our objective is then to manage a great number of rules with a fixed threshold of the support of 0,033 %. A rule will be selected if its confidence is at least 50 % and if the left and right hand sides of the rule have at least 10 common images. Under these conditions, 30 clusters numbered from 0 to 29 are computed for all the descriptors and 279 rules are produced at the end of the indexing process.



(3a) ***(3b)***

Figure 3. *Indexing Parameters: the number of clusters (3a) and the support threshold (3b)*

The MPEG-7 descriptors *ColorLayout*, *ScalableColor*, *HomogeneousTexture*, *EdgeHistogram* and *RegionShape* are respectively noted *CLD, SCD, HTD, EHD,* and *RSD*.

5.2. Experiment 2: Using association rules for progressive query

The similarity search is made on clusters of descriptor ranked according to the proximity of their centroid to a given query I_q. At the first searching phase, the query processing of I_q is done on the first rank clusters (i.e. $TopC_M^1(I_q)$). At the next searching phase, the query processing is done on the second rank clusters and the query results are merged with the ones of the previous searching phase and updated. The database is organized into 30 clusters for each descriptor, and the query processing will thus have 30 searching phases. A progressive query means that intermediairy results can be sent at the end of each phase. We propose in this experiment to explore the strategy for the dynamic use of association rules: The rules are selected as and when the query is executed. This approach requires the browsing of the whole set of association rules each time a query is submitted to the system. At the 12th searching phase, the quality of the intermediate result is almost the same than for the final result for both sequential search and the dynamic use of association rules. But Figure 4 shows that, when we use dynamically association rules, the relative loss of result quality is about $\frac{1\times 100}{15}\approx 6{,}67\%$ and the relative gain of performance (as query time) is about $\frac{(4{,}6-3{,}55)\times 100}{4{,}6}\approx 22{,}9\%$.

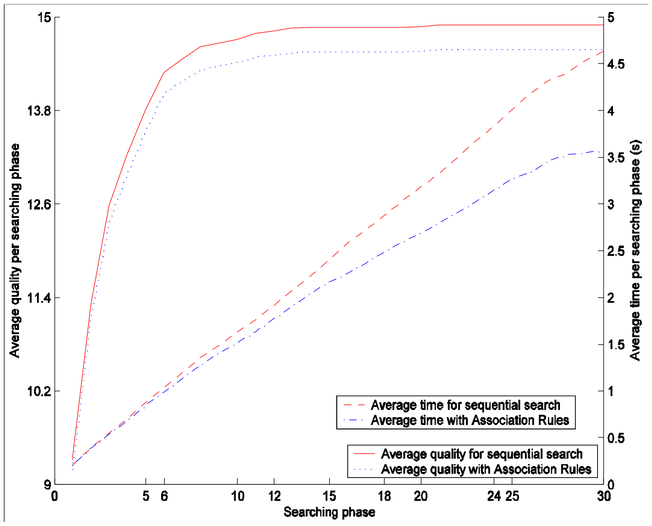


Figure 4. *Dynamical Use of Association Rules for progressive query processing*

The main advantage of progressive query execution planning is to retrieve the final result (with approximatively the same quality) after the half of the global query time necessary to a sequential search. And the dynamic use of association rules improve the query performance.

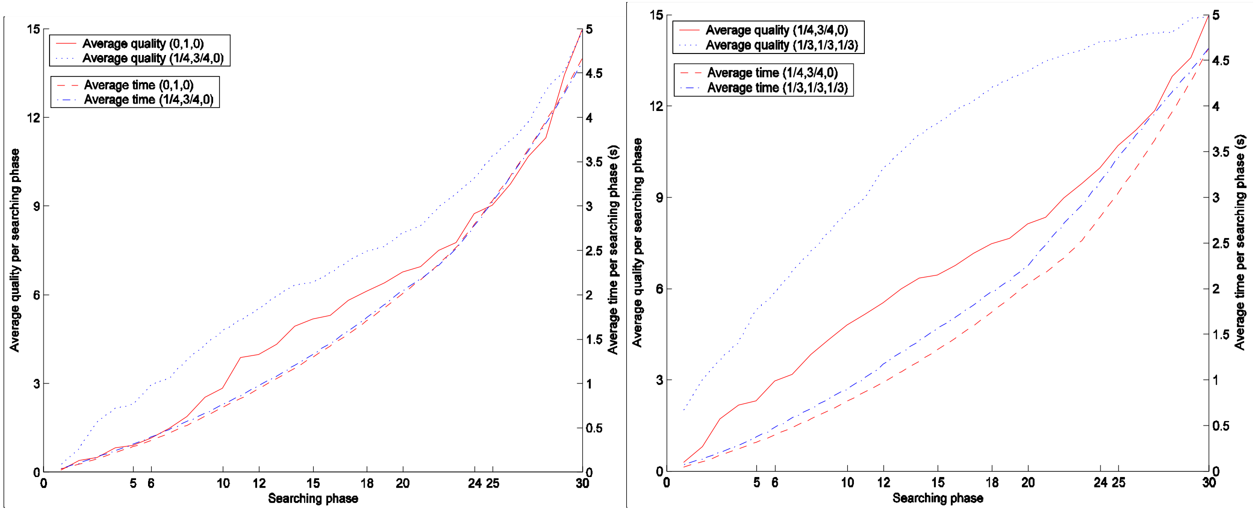
5.3. Experiment 3: Selecting clusters for progressive query

We study here the measure of selectivity to show the importance of the order of clusters in the context of a progressive query. This measure combines the proximity distance of the cluster centroid to the query image, the size of the cluster and the density (i.e., concentration) of a cluster at it centroid. It privileges the smallest, the nearest, and the most centroid concentrated clusters.

Based on the definition of selectivity measure given in section 4.1 (see formula (4)), for $\alpha = 1$, $\beta = 0$ and $\gamma=0$, clusters are ranked according to the proximity of their centroid to the query image and we have the same results shown at the previous section 5.2. Curves of quality and time for different values of α, β, γ are shown at the Figure 5.

The values α = 0, β = 1, γ=0 exclusively implie the clusters selection on the size criterion (Figure 5a). In this case, time grows slowly, but results convergence is slowed compared to the case (α = 1, β = 0, γ=0) of selection on proximity. We observe on Figure 5a that at equal time, partial results convergence speed is better for (α = 1/4, β = 3/4, γ=0). We can then say that cluster selection on the size criterion is good for reducing time but the proximity criterion contributes to keep partial results convergence high (i.e. relatively good quality compared to sequential search).

Partial results convergence is faster with (α = 1/3, β = 1/3, γ=1/3) than with α = 1/4, β = 3/4, γ=0), corresponding times varying similarly (Figure 5b). This observation confirms that additional density based criterion can speed up results convergence keeping retrieval time relatively low.
For more details on the experiments on density, we refer readers to an extended version on the paper [11].



(5a)

(5b)

Figure 5. *Selectivity measure for progressive query processing*

Using the selectivity measure for cluster selection during the progressive query execution planning improve the intermediate result quality.

6. Conclusion

We present an automatic strategy for selection of the search criteria for content-based retrieval over large image databases. This strategy is based on the use of clustering and association rules discovery techniques in order to prioritize the relevant descriptors for query-by-example and to schedule the first rank cluster to process. We introduce the measure of selectivity of clusters that we use to reduce the query time while keeping a suitable speed of convergence of the intermediate results of a progressive query close to the quality of results obtained by sequential search. All our experimental results are compared on the quality and the search time obtained with a sequential and exhaustive search on a database of 30411 fixed images described by five MPEG-7 descriptors. Our experimental results show that, on the one hand, a progressive query execution plan strategy makes it possible to have almost the totality of the final result (with the same quality) after half of the total query time of a sequential search and, on the other hand, clustering and association rules discovery over clusters of images accelerates the retrieval process. The use of association rules is thus a promising technique for the cluster selection and the improvement of the query performance. The perspectives of our work are now to prove the scalabitlity of our approach for indexing and querying very large images databases (with several hundreds of thousands of images). In this context, we may have to refine our measurement of selectivity and possibly to change the current method of clusters determination.

7. References

[1] Agrawal R., Imielinski T., Swami A., Mining association rules between sets of items in large databases, *ACM SIGMOD Int. Conf. on Management of Data*, 1993, p. 207-216.

[2] Amsaleg L., Gros P., Content-based retrieval using local descriptors: problems and issues from a database perspective, *Pattern Analysis and Applications, Special Issue on Image Indexation, vol. 4*, 2001, p. 108-124.

[3] Bentley J. L., Multidimensional binary search in database applications, *IEEE Transactions on Software Engineering, vol. 4, n° 5*, 1979, p. 333-340.

[4] Berrani S.A., Amsaleg L., Gros P., Approximate searches: k-Neighbors + Precision, *Proc. of the 12th Int. Conf. on Information and Knowledge Management*, 2003, p. 24-31.

[5] Berretti S., Del Bimbo A., Pala P., Ischia, Merging results for distributed content-based image retrieval, Proc. of the Int. Workshop On Multimedia Information Systems (MIS'03), 2003.

[6] Djeraba C., Association and content-based retrieval, *IEEE Transactions on Knowledge and Data Engineering, vol. 15, n° 1*, 2003, p. 118-135.

[7] Fagin R., Kumar R., Sivakumar D., Efficient similarity search and classification via rank aggregation, Proc. of the Int. ACM SIGMOD Conf. on Management of Data San Diego, California, USA, pp 301-312, 2003.

[8] Fernandez G., Meckaouche A., Peter P., Djeraba C., Intelligent Image Clustering, *Lecture Notes in Computer Science, vol. 2490*, 2002, p. 406-419.

[9] Gounaris A., Paton N., Fernandes A., Sakellariou R., Adaptive query processing, BNCOD 2002, *Lecture Notes in Computer Science*, vol. 2405, p. 11-25, 2002.

[10] Guttman A., R-trees: A dynamic Index Structure for Spatial Searching, *Proc. of the ACM SIGMOD Int. Conf. on Management of Data*, 1984, p. 47-57.

[11] Kouomou-Choupo A., Berti-Équille L., Feature Mining for Adapting Progressive Query-by-Example over Large Image Databases, Tech. Rep. INRIA, July 2004, *to appear*.

[12] Kouomou-Choupo A., Berti-Équille L., Morin A., Multimedia Indexing and Retrieval with Features Association Rules Mining. *Proc. of the IEEE Int. Conference on Multimedia and Expo* (ICME'2004), Taipei, Taiwan, 2004.

[13] Kiranyaz S., Gabbouj M., A novel multimedia retrieval techique: progressive query (why wait?), Proc. of the 5th Workshop of Image Analysis, Portugal, April 2004.

[14] Manjunath B. S., Salembier P., Sikora T., *Introduction to MPEG-7*, John Wiley & Sons, 2002.

[15] Manolescu I., Adaptive and self-tuning query processing, EDBT Summer School, 2002.

[16] Narendra P. M., Fukunaga K., A branch and bound algorithm for feature subset selection. IEEE Transactions on Computers, p. 917-922, September 1977.

[17] Nievergelt J., Hinterberger H., Sevcik K. C., The GridFile: An adaptable, symmetric multikey file structure, *ACM Transactions on Database Systems, vol. 9 n° 1*, 1984, p. 38-71.

[18] Ordonez C., Omiecinski E., Discovering association rules based on image content, *Proc. of the IEEE Advances in Digital Libraries Conf. (ADL'99)*, 1999, p. 38-49.

[19] Smeulders A.W.M., Worring M., Santini S., Gupta A., Jain R., Content-based image retrieval at the end of the early years, *IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. 22, n° 12*, 2000, p. 1349–1380.

[20] Zaïane O., Han J., Zhu H., Mining recurrent items in multimedia with progressive resolution re?nement, *Proc. of the* 16th *IEEE Int. Conf. on Data Engineering (ICDE'00)*, 2000, p. 461-476.

[21] Zhang J., Hsu W., Lee M. L., Image mining: issues, frameworks and techniques, *Proc. of the* 2nd *Int. Workshop on Multimedia Data Mining*, 2001, p. 13-20.