A Query-sensitive Cost Model for Similarity Queries with M-tree *

Paolo Ciaccia
DEIS - CSITE-CNR
Bologna, Italy
pciaccia@deis.unibo.it

Alessandro Nanni
Computer Science Lab.
Bologna, Italy
nannia@cs.unibo.it

Marco Patella
DEIS - CSITE-CNR
Bologna, Italy
mpatella@deis.unibo.it

Abstract. We introduce a cost model for the M-tree access method [Ciaccia et al., 1997] which provides estimates of CPU (distance computations) and I/O costs for the execution of similarity queries as a function of each single query. This model is said to be *query-sensitive*, since it takes into account, by relying on the novel notion of "witness", the "position" of the query point inside the metric space indexed by the M-tree. We describe the basic concepts underlying the model along with different methods which can be used for its implementation; finally, we experimentally validate the model over both real and synthetic datasets.

1 Introduction

Modern advanced database applications, such as sequence comparison in molecular biology [Chen and Aberer, 1997], shape matching [Huttenlocker et al., 1993], fingerprint recognition [Maio and Maltoni, 1996], and many others which typically occur in multimedia environments, often require the efficient evaluation of similarity (range and nearest neighbors) queries over a set of objects drawn from an arbitrary *metric space*. A metric space $\mathcal{M} = (\mathcal{U}, d)$ is defined by a value domain $\mathcal{U}$ and a metric d, satisfying the axioms of non-negativity, symmetry and triangular inequality ($d(O_i, O_j) \leq d(O_i, O_k) + d(O_k, O_j),\ \forall\ O_i, O_j, O_k \in \mathcal{U}$), which measures the distance (dis-similarity) of points (objects) of $\mathcal{U}$. As a particular case, metric spaces include multi-dimensional vector spaces, where objects are usually compared using the L_p distance, such as Euclidean (L_2) or Manhattan (L_1), but they are far more general. As an example, the domain $\mathcal{S}$ of text strings endowed with the *edit* (Levenshtein) distance, d_{edit}, which counts the minimal number of changes (insertions, deletions, substitutions) needed to transform a string into another one, is a metric space $(\mathcal{S}, d_{edit})$.

* This work has been funded by the EC ESPRIT LTR project no. 9141 HERMES, and Italian C.N.R. MIDA.

Metric trees are index structures developed to support similarity queries over metric spaces; most of them, such as *vp-tree* [Chiueh, 1994], *GNAT* [Brin, 1995], and *mvp-tree* [Bozkaya and Özsoyoglu, 1997], suffer from being intrinsically static. In contrast, the *M-tree* [Ciaccia et al., 1997] is a paged and balanced metric tree which acts as a dynamic database access method.

Theoretical analysis of metric trees is relevant for database design, query processing, and optimization, since it provides the means to understand and tune metric trees. Nowadays, the only *cost model for metric trees* is the one for M-tree, introduced in [Ciaccia et al., 1998]. In this paper we improve on this model by developing an effective *query-sensitive* cost model, which can change its estimates according to the "position" of the query object. As in [Ciaccia et al., 1998], the only information derivable from the analysis of the dataset and used by the model is (an estimate of) the *distance* distribution of objects (no information on *data* distribution is therefore used).

It has to be remarked that, among the many cost models [Kamel and Faloutsos, 1993; Faloutsos and Kamel, 1994; Theodoridis and Sellis, 1996; Papadopoulos and Manolopoulos, 1997; Berchtold et al., 1997] developed for multi-dimensional (spatial) access methods (such as R-tree [Guttman, 1984; Beckmann et al., 1990]), only the one proposed in [Theodoridis and Sellis, 1996] is query-sensitive (all the others only provide estimates for average costs). However, this model cannot be applied at all to generic metric spaces.

The rest of the paper is organized as follows. In Section 2 the basic principles of M-tree and of its existing average-case cost model are reviewed. In Section 3 we show that such a model becomes inadequate when cost estimates for single queries are needed. Section 4 introduces the concepts that lead to the new query-sensitive model and discusses several methods which can be used for its implementation. In Section 5 we provide some experimental results and, in Section 6, we draw our conclusions.

2 The M-tree

The M-tree [Ciaccia et al., 1997] is a dynamic access structure able to index objects from a generic metric space $\mathcal{M} = (\mathcal{U}, d)$.[2] Given a set of objects $\mathcal{O} = \{O_1, \ldots, O_n\}$, $\mathcal{O} \subseteq \mathcal{U}$, the M-tree stores them into fixed-size leaf nodes, which correspond to regions (*balls*) of the metric space. Each entry in a leaf node has the format $[O_i, \text{oid}(O_i)]$, where $O_i \in \mathcal{O}$ and $\text{oid}(O_i)$ is a pointer to the corresponding description of O_i. Internal (non-leaf) nodes store entries with format $[O_r, r(N_r), \text{ptr}(N_r)]$, where O_r is a so-called *routing object*, $r(N_r) > 0$ is a *covering radius*, and $\text{ptr}(N_r)$ is a pointer to the child node N_r. The basic property of the covering radius is that *each object O_i in the sub-tree rooted at N_r satisfies the constraint $d(O_r, O_i) \leq r(N_r)$*, that is, all the objects O_i reachable from node N_r are in the ball of radius $r(N_r)$ centered in O_r. Clearly, the actual

[2] The code of M-tree, along with other information, is freely available at URL `http://www-db.deis.unibo.it/~patella/MMindex.html`.

"shape" of such balls depends on the specific metric space $(\mathcal{U}, d)$. For instance, balls are "diamonds" in $(\Re^2, L_1)$, circles in $(\Re^2, L_2)$, and squares in $(\Re^2, L_\infty)$. From an overall point of view, the M-tree organizes the database objects into a set of, possibly overlapping, balls, to which the same principle is recursively applied up to the root of the tree.

Similarity queries supported by the M-tree are of two basic types: (a) *range* queries, denoted $\texttt{range}(Q, r_Q)$, where, given a query (reference) object $Q \in \mathcal{U}$, all the objects whose distance from Q does not exceed r_Q are selected, and (b) *k-nearest neighbors* queries, $\texttt{NN}(Q, k)$, where the k ($k \geq 1$) closest objects to Q have to be retrieved, with ties arbitrarily broken.

2.1 The Average-case Cost Model

The existing average-case (A-C) cost model for M-tree [Ciaccia et al., 1998] gives only estimates of average costs and makes use of statistics on the structure of the M-tree. In the following we will consider only the so-called *level-based* variant, which exploits statistics collected on a per-level basis. Relevant symbols are listed in Table 1.

Symbol	Description	Symbol	Description
$F(x)$	distance distribution	N	number of indexed objects
$f(x)$	distance density function	M_l	no. of nodes at level l of the M-tree
d^+	upper bound on distances	$\overline{r_l}$	average covering radius of nodes at level l
Q	query object	h	height of the M-tree
r_Q	query radius	$nodes(\ldots)$	estimate of avg. no. of nodes accessed
k	no. of nearest neighbors	$dists(\ldots)$	estimate of avg. no. of computed distances

Table 1. Summary of symbols.

A central role in the model is held by the *overall distance distribution* (d.d.), whose cumulative and density functions are respectively defined as:

$$F(x) = \Pr\{d(\mathbf{O_1}, \mathbf{O_2}) \leq x\} \qquad f(x) = \frac{\mathrm{d}F(x)}{\mathrm{d}x} \tag{1}$$

where $\mathbf{O_1}$ and $\mathbf{O_2}$ are two (independent) random points (objects) of the domain $\mathcal{U}$. The *relative d.d.* of an object $O_i \in \mathcal{U}$ is obtained by setting $\mathbf{O_1} = O_i$, i.e.:

$$F_{O_i}(x) = \Pr\{d(O_i, \mathbf{O_2}) \leq x\} \tag{2}$$

Given a range query $\texttt{range}(Q, r_Q)$, a node at level l of the M-tree, having routing object O_r, has to be accessed iff the ball of radius r_Q centered in the query object Q and the region associated with the node intersect. Using average statistics on the tree, this is the case iff $d(Q, O_r) \leq \overline{r_l} + r_Q$; therefore, the probability that a node at level l has to be accessed can be expressed as:

$$\begin{aligned} \Pr\{\text{a node at level } l \text{ is accessed}\} &= \Pr\{d(Q, O_r) \leq \overline{r_l} + r_Q\} = \\ &= F_Q(\overline{r_l} + r_Q) \approx F(\overline{r_l} + r_Q) \end{aligned} \tag{3}$$

where the approximation is due to the use of the overall d.d. F in place of the d.d. relative to the query point Q.

The average number of pages accessed and the average number of distances computed (I/O and CPU costs, respectively) by a range query are estimated as:

$$nodes(\texttt{range}(Q, r_Q)) = \sum_{l=0}^{h-1} M_l \, F(\overline{r_l} + r_Q) \tag{4}$$

$$dists(\texttt{range}(Q, r_Q)) = \sum_{l=0}^{h-1} M_{l+1} \, F(\overline{r_l} + r_Q) \tag{5}$$

where M_l is the number of M-tree nodes at level l ($l \in [0, h-1]$, with root at level 0 and leaves at level $h-1$) and $M_h = N$ is the number of indexed objects.

Let us now consider a query of type $\texttt{NN}(Q, k)$; as a first step, the distribution of the distance $\texttt{nn}_{Q,k}$ between Q and its k-th nearest neighbor is determined. The probability that $\texttt{nn}_{Q,k}$ is at most r equals the probability that at least k objects are inside the ball of radius r centered in Q, that is:

$$P_{\texttt{nn}_{Q,k}}(r) = \Pr\{\texttt{nn}_{Q,k} \leq r\} = \ldots = 1 - \sum_{i=0}^{k-1} \binom{N}{i} F(r)^i \, (1 - F(r))^{N-i} \tag{6}$$

and the average costs for a k-nearest neighbors query are estimated as:

$$costs(\texttt{NN}(Q, k)) = \int_0^{d^+} costs(\texttt{range}(Q, r)) \, \frac{\mathrm{d}P_{\texttt{nn}_{Q,k}}(r)}{\mathrm{d}r} \, \mathrm{d}r \tag{7}$$

where $costs(\texttt{range}(Q, r))$ is given by Equation 4 (I/O costs) or by Equation 5 (CPU costs), and d^+ is a finite upper bound on distances.

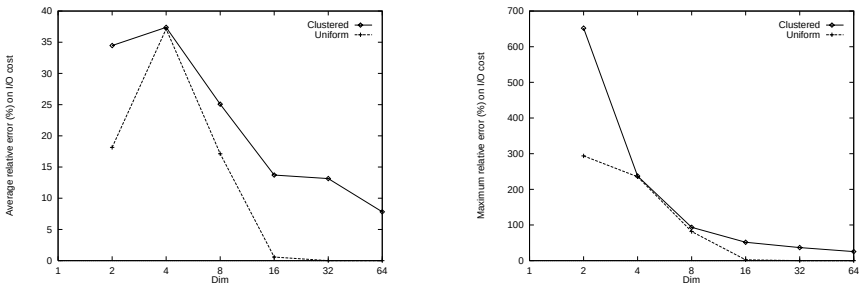
3 Limitations of the Average-Case Cost Model

In order to evaluate the accuracy of a cost model, it is important to precisely define how *estimate errors* are assessed. To this end, given an experimental testbed consisting of a set $\mathcal{Q} = \{q_1, \ldots, q_n\}$ of n queries, we consider three different kinds of error, as explained in the following.

- *Average absolute relative error*, defined as $\texttt{AvgErr} = \frac{1}{n} \sum_{q \in \mathcal{Q}} \left| \frac{\widehat{c}_q - c_q}{c_q} \right|$, where c_q is the cost of query q and $\widehat{c}_q$ is its estimate.
- *Maximum absolute relative error*, defined as $\texttt{MaxErr} = \max_{q \in \mathcal{Q}} \left\{ \left| \frac{\widehat{c}_q - c_q}{c_q} \right| \right\}$
- *Absolute relative error on average costs*, defined as $\texttt{AvgCaseErr} = \left| \frac{\widehat{c}_{avg} - c_{avg}}{c_{avg}} \right|$, where $c_{avg} = \frac{1}{n} \sum_{q \in \mathcal{Q}} c_q$ is the average cost due to the execution of the n queries and $\widehat{c}_{avg}$ is the model estimate for the average execution cost.

Note that the performance of average-case cost models is typically evaluated by the `AvgCaseErr` measure (see [Faloutsos and Kamel, 1994; Berchtold et al., 1997; Ciaccia et al., 1998]). In this light, as shown in [Ciaccia et al., 1998], the A-C cost model performs well both on real and synthetic datasets, with `AvgCaseErr` assuming values typically around 10%-15%.

The inadequacy of the A-C model to estimate *the cost of single queries* is demonstrated by Figures 1 (a) and 1 (b), that show, respectively, `AvgErr` and `MaxErr` values resulting from the execution of 500 different range queries on synthetic datasets (see Table 2). It can be seen that `AvgErr` is generally around 20%-30% and sometimes close to 40%, whereas `MaxErr` assumes high values, tipically around 100%-200%, with peaks near 700%. It can also be observed that the accuracy of the A-C model increases with the space dimensionality. This is because, for high dimensionalities, the "Homogeneity of Viewpoints" of the metric space, as defined in [Ciaccia et al., 1998], is very high, i.e. relative d.d.'s of different objects are very similar; thus, the main approximation of Equation 3 has a minor impact on the performance of the model. The performance of the model for uniform datasets has a different explanation: in high dimensional spaces, the so-called *dimensionality curse* is such that, when $D \geq 16$, for (almost) every query the whole index has to be accessed [Weber et al., 1998], which makes quite easy to predict costs. For this reason, in subsequent analyses we will mainly concentrate on clustered datasets.



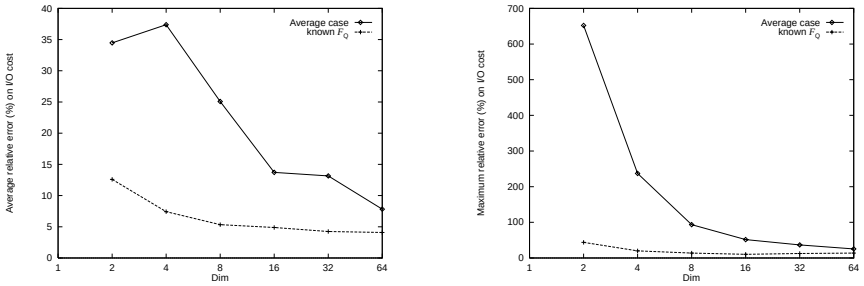
(a) (b)

Fig. 1. `AvgErr` (a) and `MaxErr` (b) for the A-C model — I/O costs for range queries on synthetic datasets.

4 The Query-sensitive Cost Model

Our approach to obviate the poor performance of the A-C model is to try to provide an estimate better than F of the d.d. F_Q relative to the query object Q (see Equation 3). Indeed, as Figures 2 (a) and 2 (b) show, both `AvgErr` and `MaxErr` are considerably reduced when F_Q is known. This means that the probabilistic arguments used by the A-C model are "good", but F_Q is poorly approximated by F.

The key idea around which we develop our query-sensitive (Q-S) cost model is to approximate F_Q by means of several relative d.d.'s, each one corresponding to a point of $\mathcal{U}$, conveniently called a *witness*. For every witness W_j $(j = 1, \ldots, nw)$



(a) (b)

Fig. 2. `AvgErr` (a) and `MaxErr` (b) for the model with known F_Q and for the A-C model — I/O costs for range queries on synthetic clustered datasets.

we store the relative d.d. $F_{W_j}(x) = \Pr\{d(W_j, \mathbf{O}) \leq x\}$, and exploit the "position" of the query object Q with respect to the witnesses (see Figure 3) to provide an approximation of F_Q. For this, two problems have to be addressed: (a) how witnesses have to be chosen among all the objects of the dataset, and (b) how their relative d.d.'s have to be combined.

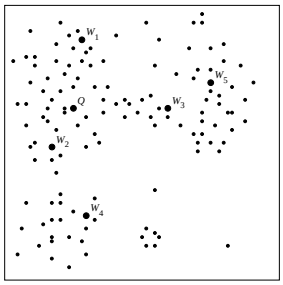


Fig. 3. The d.d. of query object Q is approximated by using d.d.'s of witnesses W_j.

4.1 How to Choose Witnesses

The choice we make about which objects of the dataset have to be designated as witnesses has its own importance, since it affects the way the metric space is "covered" by them: a bad choice, for instance, could lead to have regions of the metric space within which the relative d.d.'s are (highly) different and not enough witnesses are present to capture this variability. A related topic is also the choice for the number nw of witnesses to use: we expect the estimates of the model to improve with nw, since more information can be exploited.

Ideally, we should select witnesses in such a way that, for every "possible" query object Q, a witness with a d.d. arbitrarily close to F_Q exists. The basic heuristics we consider is to minimize the distance between each possible query object, Q, and the set of witnesses, by assuming that close objects have similar d.d.'s. To this end, witnesses should provide an appropriate "coverage" of the data space. With this in mind, we propose two basic criteria to choose the witnesses.

'Random' – the nw witnesses are chosen in a random way. In this way, since witnesses are distributed like objects in $\mathcal{O}$, it is very likely that each query object will have a close witness, under the assumption of a *biased* query model, i.e. indexed and query objects follow the same distribution [Ciaccia et al., 1998].

'GNAT' – nw witnesses are chosen in a way similar to the one used in the GNAT access method [Brin, 1995] to designate "split points". The 'GNAT' method first extracts from the dataset a random sample $\mathcal{C}$ of $3 \cdot nw$ *candidate* objects; then, nw witnesses are chosen from $\mathcal{C}$ as follows: the first witness is picked at random and removed from $\mathcal{C}$; the i-th witness ($i = 2, \ldots, nw$) is the object in $\mathcal{C}$ which maximizes the minimum distance from the already chosen $i-1$ witnesses. This method aims to choose witnesses far apart each other and to "cover" the metric space in a somewhat homogeneous way, thus avoiding the "crowding" of witnesses in dense regions of the space. In other terms, 'GNAT' tries to minimize the maximum possible distance between a query object and its nearest witness.

4.2 How to Combine Relative Distance Distributions

Starting from the relative d.d.'s of the witnesses, the next problem is how to combine them to estimate F_Q. We consider two main criteria to cope with this problem, and we propose a specific variant for the second one. In all cases, the basic rationale is, again, that close objects have similar d.d.'s.

'Nearest Witness' Method This method estimates F_Q as follows:

$$F_Q(x) \simeq F_{W_{NW}}(x) \qquad \text{with} \;\; d(W_{NW}, Q) \leq d(W_j, Q), \;\; j = 1, \ldots, nw \tag{8}$$

that is, the unknown d.d. relative to Q is approximated with the one of its nearest witness. This method is quite easy to use and does not require a new d.d. to be computed.

'Distance Weighted' Method This method estimates F_Q as a weighted average of *all* the relative d.d.'s of the witnesses, that is:

$$F_Q(x) \simeq \sum_{j=1}^{nw} F_{W_j}(x) \cdot \alpha_j \Big/ \sum_{j=1}^{nw} \alpha_j \tag{9}$$

In order to assign appropriate α_j weights, we rely on the assumption that witnesses closer to Q are supposed to be more "reliable", since their d.d.'s are more similar to that of Q. Therefore, weights α_j should be inversely related to the distances between Q and the witnesses W_j. Our choice is to use the Exp-th power of the inverse of the distance between a witness and Q, that is, $\alpha_j = d(W_j, Q)^{-Exp}$.

We point out that the 'Nearest Witness' method is the limit of the 'Distance Weighted' method when Exp goes to infinity. On the other end, for $Exp = 0$ we obtain the classic arithmetic average, which equally weighs the contribution of each witness.

'Distance Weighted', Adaptive Method Since the 'Distance Weighted' method depends on the Exp parameter, it would be advisable to relate the value of Exp to the specific query at hand. The *adaptive* variant we propose determines Exp as a function of the distances between the W_j's and Q. When these distances are all somewhat "high", it does not make sense to rely on the nearest witness, thus a low value for Exp is chosen. On the other hand, when witnesses are (very) close to Q, Exp is assigned a higher value, since information provided by the nearest witness(es) is more reliable.

We define the average distance of witnesses from Q, normalized to maximum distance d^+, as follows:

$$m_{dw} = \sum_{j=1}^{nw} \frac{d(W_j, Q)}{d^+ \cdot nw} \tag{10}$$

Now, for a given Q, Exp can be computed as $Exp = (1 - m_{dw}) \cdot MaxExp$, where $MaxExp$ is the value of Exp for which the 'Distance Weighted' and the 'Nearest Witness' methods give very similar estimates. Experimentally we found out that an acceptable value is $MaxExp = 10$; this is also the value adopted in all the tests we carried out.

5 Experimental Evaluation

In order to evaluate the accuracy of our cost models, we ran several experiments on both synthetic and real datasets (see Table 2). Estimates are compared with actual results obtained by the M-tree, which was built using the `BulkLoading` algorithm described in [Ciaccia and Patella, 1998] with a node size of 4 Kbytes and a minimum node utilization of 30%.

Name	Description	Size	Dim. (D)	Metric
clustered	clustered distrib. points on $[0,1]^D$	$20,000$	$2-64$	L_∞
uniform	uniform distrib. points on $[0,1]^D$	$20,000$	$2-64$	L_∞
D	Decamerone	$17,396$		edit
DC	Divina Commedia	$12,701$		edit
GL	Gerusalemme Liberata	$11,973$		edit
OF	Orlando Furioso	$18,719$		edit
PS	Promessi Sposi	$19,486$		edit

Table 2. Datasets.

5.1 Datasets, Queries, and Distance Distributions

Synthetic datasets were obtained using the procedure described in [Jain and Dubes, 1988] which generates normally-distributed clusters in a D-dimensional vector space. In particular, this is the case of `clustered` datasets, for which the number of clusters is 10, the variance is $\sigma^2 = 0.1$, and clusters' centers are uniformly distributed. We also considered `uniform` datasets, that is, vectors

uniformly distributed in the D-dimensional space. Distance on these datasets was evaluated using the L_∞ metric, i.e. $L_\infty(O_x, O_y) = \max_{j=1}^{D} \{|O_x[j] - O_y[j]|\}$, which leads to hyper-cubic search (and covering) regions. *Real datasets* are sets of keywords extracted from 5 masterpieces of Italian literature; the metric used on them is the edit distance.

For each test, 500 similarity queries were executed; unless otherwise stated, for range queries we used a radius $r_Q = \sqrt[D]{0.01}/2$ for synthetic datasets and a radius $r_Q = 3$ for real datasets. For k-nearest neighbor queries we considered only $k = 1$, which is the most common case.

For each d.d. needed in the experiments we ran, an estimate of its density function f was obtained from an analysis of the dataset $\mathcal{O}$. In particular, for the overall d.d. we sampled all the $N(N-1)/2$ pairwise distances between all the N objects of $\mathcal{O}$; for the relative d.d. of a witness, we considered all the $N-1$ distances between the witness and the other objects of $\mathcal{O}$. In all cases the density function f was approximated by means of a 100 bins equi-width histogram of type ASH (Averaged Shifted Histogram [Scott, 1992]).

5.2 Experimental Results

In this Section we present some results showing the performance of the Q-S model, and also contrast it with the A-C model. The reported graphs only refer to I/O costs, since errors on estimates of CPU costs show a very similar trend.

Figure 4 shows how the number of witnesses and the way they are chosen can influence error estimates. For simplicity, we report only the values of `AvgErr` for range queries on synthetic clustered datasets when the 'Nearest Witness' method is used.

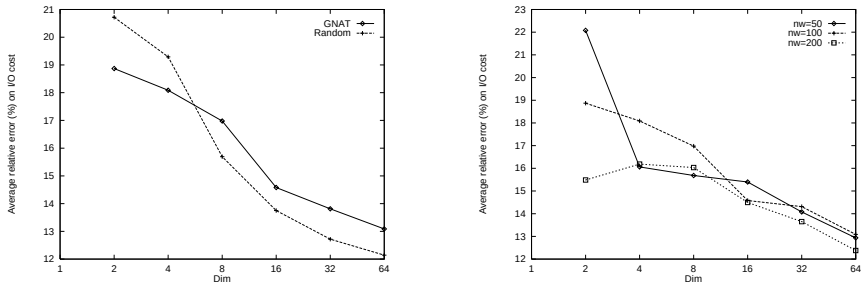
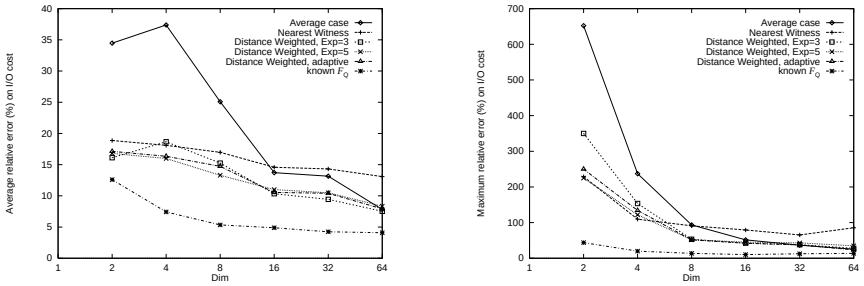


Fig. 4. `AvgErr` for range queries. 'Nearest Witness' method: (a) 'GNAT' vs. 'Random'; (b) effect of the number of witnesses ('GNAT' chosen).

Figure 4 (a) shows that usually we obtain more accurate estimates when witnesses are chosen in a 'GNAT' way, rather than using the 'Random' criterion. The explanation for this is that the 'GNAT' method typically leads to a lower average distance between the query object and its nearest witness, as compared to the distance obtained by the 'Random' method. This trend, however, is inverted for higher dimensionalities of the space, with estimate differences lower

than 1%. In Figure 4 (b) errors obtained by varying the number of witnesses are shown. As expected, by increasing the number of witnesses the estimates of the model improve; anyway, it can be seen that this improvement is not directly proportional to the increase introduced and sometimes it is not really appreciable.

In the following tests we analyze the behavior of the different variants of the Q-S model, comparing them with the A-C model. All these tests were carried out using 100 witnesses chosen in a 'GNAT' way.



(a) (b)

Fig. 5. `AvgErr` (a) and `MaxErr` (b) for range queries on synthetic datasets.

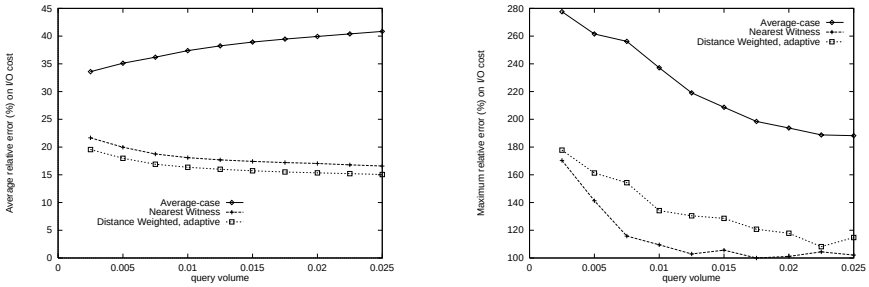
From the results shown in Figure 5 we can point out what follows:

- typically, Q-S methods exhibit a great improvement over the A-C model;
- when average-case estimates are affected by high errors, Q-S methods allow for an effective reduction of such errors;
- 'Distance Weighted' usually performs better than 'Nearest Witness';
- the adaptive variant of 'Distance Weighted' exhibits good results, often close to the best estimates over all Q-S methods.

To analyze the effect of query selectivity, in Figure 6 we compare the estimates of two Q-S methods — 'Nearest Witness' and the adaptive variant of 'Distance Weighted' — with the ones of the A-C model. We consider I/O costs for range queries on a clustered dataset with $D = 4$, and vary the query volume between 0.0025 and 0.025 (note that all previous results refer to a query volume of 0.01).

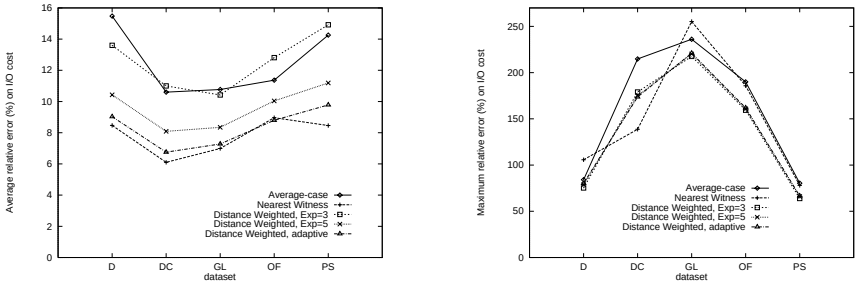
From Figure 6 the improvement of the Q-S model over the A-C model is evident. In the specific case, it can also be seen that the adaptive method performs better than 'Nearest Witness', as regards `AvgErr`. Considering `MaxErr`, instead, we observe exactly the opposite behavior.

Considering the results of experiments on real datasets, Figure 7 shows errors for range queries. Concerning `AvgErr`, it is interesting to observe that the A-C model exhibits quite good estimates on single queries, with relative errors between 10% and 16%. Anyway, the Q-S model performs even better: the adaptive 'Distance Weighted' method and the 'Nearest Witness' method are very accurate, limiting `AvgErr` to 6% – 10%.



(a) (b)

Fig. 6. `AvgErr` (a) and `MaxErr` (b) for range queries on a 4-D clustered dataset as a function of the query volume.



(a) (b)

Fig. 7. `AvgErr` (a) and `MaxErr` (b) for range queries on real datasets.

Considering nearest neighbor queries, not shown here for brevity, the difference between query-sensitive estimates and those of A-C model reduces, since the errors on average-case estimates are better than the respective ones for range queries. Anyway, Q-S methods still obtain the best estimates.

6 Conclusions

In this work, starting from the existing cost model for M-tree [Ciaccia et al., 1998], we introduced some new concepts — mainly the ones of *witness* and *relative distance distributions* — which led to a new *query-sensitive* (Q-S) cost model, able to predict the cost of similarity queries by taking into account also the "position" of the query object. The concepts we have introduced are independent of the underlying cost model, and could therefore be applied to any model for metric trees which now makes use of the overall distance distribution.

Experimental results demonstrated that the Q-S model yields reliable estimates for both synthetic and real datasets, with a substantial improvement over the previous average-case cost model. Different variants were proposed and tested in order to determine how witnesses have to be chosen and how their relative distance distributions are to be combined; we also briefly analyzed the influence of the number of witnesses used.

In this work we have insisted on the usefulness of the proposed model in estimating execution costs for specific queries with M-tree. Issues concerning optimization of the algorithm and of the used structures, e.g. histograms compression, are left as subjects for future work.

References

Beckmann, N., Kriegel, H.-P., Schneider, R., and Seeger, B. (1990). The R*-tree: An efficient and robust access method for points and rectangles. SIGMOD'90, pp. 322–331, Atlantic City, NJ.

Berchtold, S., Böhm, C., Keim, D. A., and Kriegel, H.-P. (1997). A cost model for nearest neighbor search in high-dimensional data space. PODS'97, pp. 78–86, Tucson, AZ.

Bozkaya, T. and Özsoyoglu, M. (1997). Distance-based indexing for high-dimensional metric spaces. SIGMOD'97, pp. 357–368, Tucson, AZ.

Brin, S. (1995). Near neighbor search in large metric spaces. VLDB'95, pp. 574–584, Zurich, Switzerland.

Chen, W. and Aberer, K. (1997). Efficient querying on genomic databases by using metric space indexing techniques. PMIDS'97, Toulouse, France.

Chiueh, T. (1994). Content-based image indexing. VLDB'94, pp. 582–593, Santiago, Chile.

Ciaccia, P. and Patella, M. (1998). Bulk loading the M-tree. ADC'98, pp. 15–26, Perth, Australia.

Ciaccia, P., Patella, M., and Zezula, P. (1997). M-tree: An efficient access method for similarity search in metric spaces. VLDB'97, pp. 426–435, Athens, Greece.

Ciaccia, P., Patella, M., and Zezula, P. (1998). A cost model for similarity queries in metric spaces. PODS'97, pp. 59–68, Seattle, WA.

Faloutsos, C. and Kamel, I. (1994). Beyond uniformity and independence: Analysis of R-trees using the concept of fractal dimension. PODS'94, pp. 4–13, Minneapolis, MN.

Guttman, A. (1984). R-trees: A dynamic index structure for spatial searching. SIGMOD'94, pp. 47–57, Boston, MA.

Huttenlocker, D.P., Klanderman, G.A., and Rucklidge, W.J. (1993). Comparing images using the Hausdorff distance. *IEEE Trans. PAMI*, 15(9):850–863.

Jain, A. K. and Dubes, R. C. (1988). *Algorithms for Clustering Data.* Prentice-Hall.

Kamel, I. and Faloutsos, C. (1993). On packing R-trees. CIKM'93, pp. 490–499, Washington, DC.

Maio, D. and Maltoni, D. (1996). A structural approach to fingerprint classification. ICPR'96, volume C, pp. 578–585, Wien, Austria.

Papadopoulos, A. and Manolopoulos, Y. (1997). Performance of nearest-neighbor queries in R-trees. ICDT'97, pp. 394–408, Delphi, Greece.

Scott, D. W. (1992). *Multivariate Density Estimation. Theory, Practice and Visualization.* Wiley-Interscience, New York.

Theodoridis, Y. and Sellis, T. (1996). A model for the prediction of R-tree performance. PODS'96, pp. 161–171, Montreal, Canada.

Weber, R., Schek, H.-J., and Blott, S. (1998). A quantitative analysis and performance study for similarity-search methods in high-dimensional spaces. VLDB'98, pp. 196–205, New York, NY.